3D Photography

Course Notes for SIGGRAPH 2000
Los Angeles, California
July 24, 2000

Organizers

Brian Curless	University of Washington
Steven Seitz	Carnegie Mellon University

Speakers

Jean-Yves Bouguet	Intel Corporation
Brian Curless	University of Washington
Paul Debevec	University of California at Berkeley
Marc Levoy	Stanford University
Shree Nayar	Columbia University
Steven Seitz	Carnegie Mellon University

Course Abstract

3D photography is the process of using cameras and light to capture the shape and appearance of real objects. This process provides a simple way of acquiring graphical models of unparalleled detail and realism by scanning them in from the real world. This course provides an introduction to the emerging area of 3D photography, focusing on the current state of the art and the principles underlying several leading approaches.

After introducing fundamental concepts, the course surveys a variety of techniques and provides an in-depth analysis of a few successful approaches at the forefront of 3D photography, presented by leading researchers in the field. The focus is on passive and active optical methods, including stereo vision, photogrammetry, structured light, imaging radar, interferometry, and optical triangulation. The course concludes with a field study: capturing 3D photographs of Michelangelo's statues.

Scope

The course will cover a variety of methods for recovering shape and appearance from images. The course begins with novel 2D sensing technologies such as catadioptric cameras and high dynamic range sensors. A number of standard and emerging passive vision methods will be presented, including stereo, structure from motion, shape from focus/defocus, shape from shading, interactive photogrammetry, and voxel coloring. Active vision methods will include imaging radar, optical triangulation, moire, active stereo, active depth from defocus, and desktop shadow striping. An overview of reconstructing shape and appearance from range images will be followed by the first presentation of the Digital Michelangelo Project to the SIGGRAPH community.

Prerequisites

Participants will benefit from an understanding of basic techniques for representing and rendering surfaces and volumes. In particular, the course will assume familiarity with triangular meshes, voxels, and implicit functions (isosurfaces of volumes). Rendering concepts will include light interaction with surfaces (e.g., diffuse and specular reflection) and the mathematics of perspective projection. Understanding of basic image-processing will also be important. Experience with still photography will be helpful.

Course Notes Description

Course notes consist of copies of the speakers' slides, images and VRML files of some of the demonstrations, references to related work, and copies of related papers. Links to online 3D Photography resources, additional slides, and other materials may be found on the course web page at: **http://www.cs.cmu.edu/~seitz/3DPhoto.html**

Speaker Biographies

Brian Curless (co-organizer)
Assistant Professor
Dept. of Computer Science & Engineering
University of Washington
Sieg Hall, Box 352350
Seattle, WA 98195-2350
Tel: (206) 685-3796
Fax: (206) 543-2969
Email: curless@cs.washington.edu
Web: http://www.cs.washington.edu/homes/curless

Brian Curless is an assistant professor of Computer Science and Engineering at the University of Washington. He received a B.S. in Electrical Engineering from the University of Texas at Austin in 1988 and M.S. and Ph.D. degrees in Electrical Engineering from Stanford University in 1991 and 1997, respectively. Curless's recent research has focused on acquiring and building complex geometric models using structured light scanning systems. In the vision literature, he has published results on fundamentally better methods for optical triangulation, and at SIGGRAPH, he published a new method for combining range images that led to the first "3D fax" of a geometrically complex object. Curless currently sits on the Technical Advisory Board for Paraform, Inc., a company that is commercializing Stanford-developed technology for building CAD-ready models from range data and polygonal meshes. In the winter of 1999, Curless worked with Marc Levoy on the Digital Michelangelo Project in Florence where they captured the geometry and appearance of a number of Michelangelo's statues. His teaching experience includes both graduate and undergraduate graphics courses, including courses related to 3D photography taught at Stanford, the University of Washington, CVPR '99, and SIGGRAPH '99. Curless received a university-wide Outstanding Teaching Award from Stanford in 1992 and an NSF CAREER award (1999) and Sloan Fellowship (2000) at the University of Washington.

Steven Seitz (co-organizer)
Assistant Professor
The Robotics Institute
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
Tel: (412) 268-6795
Fax: (412) 268-5669
Email: seitz@cs.cmu.edu
Web: http://www.cs.cmu.edu/~seitz

Steven Seitz is an Assistant Professor of Robotics and Computer Science at Carnegie Mellon University, where he conducts research in image-based rendering, graphics, and computer vision. Before joining the Robotics Institute in August 1998, he spent a year visiting the Vision Technology Group at Microsoft Research, and a previous summer in the Advanced Technology Group at Apple Computer. He received his B.A. in computer science and mathematics at the University of California, Berkeley in 1991 and his Ph.D. in computer sciences at the University of Wisconsin, Madison in 1997. His current research focuses on the problem of acquiring and manipulating visual representations of real environments using semi- and fully-automated techniques. This effort has led to the development of "View Morphing" techniques for interpolating different images of a scene and voxel-based algorithms for computing photorealistic scene reconstructions. His work in these areas has appeared at SIGGRAPH and in international computer vision conferences and journals, and he co-organized courses on 3D Photography taught at CVPR '99 and SIGGRAPH 99. Seitz was awarded the 1999 David Marr Prize in Computational Vision for his co-authored paper on "Space Carving" at ICCV 99.

Jean-Yves Bouguet
California Intitute of Technology - MS 136-93
1200 East California Blvd.
Pasadena, CA 91125
Tel: (626) 395 3272
Fax: (626) 795 8649
Email: bouguetj@vision.caltech.edu
Web: http://www.vision.caltech.edu/bouguetj

Jean-Yves Bouguet is a researcher at Intel Corporation in the Microprocessor Research Labs. He received his diplome d'ingenieur from the Ecole Superieure d'ingenieurs en Electrotechnique et Electronique (ESIEE) in 1994 and the M.S. and Ph.D. degrees in Electrical Engineering from the California Institute of Technology (Caltech) in 1994 and 1999, respectively. His research interests cover passive and active techniques for three-dimensional scene modeling. He has developed a simple and inexpensive method for scanning objects using shadows. This work was first presented at ICCV'98 and a patent is pending on that invention. He also collaborated with Jim Arvo, Peter Schroder and Pietro Perona in teaching a class on 3D photography from 1996 to 1998 at Caltech. During his Ph.D. studies, Jean-Yves has also been working in collaboration with Larry Matthies at JPL on the development of passive visual techniques for three dimensional autonomous navigation targeted towards comet modeling and landing.

Paul Debevec
Research Scientist
University of California at Berkeley
387 Soda Hall #1776
Computer Science Division, UC Berkeley
Berkeley, CA 94720-1776
Tel: (510) 642-9940
Fax: (510) 642-5775
Email: debevec@cs.berkeley.edu
Web: http://www.cs.berkeley.edu/~debevec

Paul Debevec earned degrees in Math and Computer Engineering at the University of Michigan in 1992 and completed his Ph.D. at the University of California at Berkeley in 1996, where he is now a research scientist. Debevec has worked on a variety of image-based modeling and rendering projects, beginning in 1991 in deriving a 3D model of a Chevette from photographs for an animation project. Debevec has collaborated on projects at Interval Research Corporation in Palo Alto that used a variety of image-based techniques for interactive applications; the "Immersion '94" project done with Michael Naimark and John Woodfill developed an image-based walkthrough of the Banff national forest and his art installation "Rouen Revisited" done with Golan Levin showed at the SIGGRAPH 96 art show. His Ph.D. thesis under Jitendra Malik in collaboration with C.J. Taylor presented an interactive method of modeling architectural scenes from sparse sets of photographs and for rendering these scenes realistically. Debevec has directed several computer animations using image-based modeling, rendering, and lighting techniques including "The Campanile Movie", "Rendering with Natural Light" and "Fiat Lux" shown in the SIGGRAPH Electronic Theater. With Steven Gortler, Debevec organized the course "Image-Based Modeling and Rendering" at SIGGRAPH 98.

Marc Levoy
Associate Professor
Stanford University
Gates Computer Science Building
Room 366, Wing 3B
Stanford University
Stanford, CA 94305
Tel: (650) 725-4089
Fax: (650) 723-0033
Email: levoy@cs.stanford.edu
Web: http://graphics.stanford.edu/~levoy

Marc Levoy is an associate professor of Computer Science and Electrical Engineering at Stanford University. He received a B. Architecture in 1976 from Cornell University, an M.S. in 1978 from Cornell University, and a Ph.D. in Computer Science in 1989 from the University of North Carolina at Chapel Hill. Levoy's early research centered on computer-assisted cartoon animation, leading to development of a computer animation system for Hanna-Barbera Productions. His recent publications are in the areas of volume visualization, rendering algorithms, computer vision, geometric modeling, and user interfaces for imaging and visualization. His current research interests include digitizing the shape and appearance of physical objects using multiple sensing technologies, the creation, representation, and rendering of complex geometric models, image-based modeling and rendering, and applications of computer graphics in art history, preservation, restoration, and archeology. Levoy received the NSF Presidential Young Investigator Award in 1991 and the SIGGRAPH Computer Graphics Achievement Award in 1996 for his work in volume rendering.

Shree K. Nayar
Professor
Department of Computer Science
Columbia University
500 West, 120 Street
New York, NY 10027
Tel: (212) 939-7092
Fax: (212) 939-7172

Email: nayar@cs.columabia.edu
Web: http://www.cs.columbia.edu/~nayar/

Shree K. Nayar is a Professor at the Department of Computer Science, Columbia University. He received his PhD degree in Electrical and Computer Engineering from the Robotics Institute at Carnegie Mellon University in 1990. His primary research interests are in computational vision and robotics with emphasis on physical models for early visual processing, sensors and algorithms for shape recovery, learning and recognition of visual patterns, and vision for graphics. Dr. Nayar has authored and coauthored papers that have received the David Marr Prize at the 1995 International Conference on Computer Vision (ICCV'95) held in Boston, Siemens Outstanding Paper Award at the 1994 IEEE Computer Vision and Pattern Recognition Conference (CVPR'94) held in Seattle, 1994 Annual Pattern Recognition Award from the Pattern Recognition Society, Best Industry Related Paper Award at the 1994 International Conference on Pattern Recognition (ICPR'94) held in Jerusalem, and the David Marr Prize at the 1990 International Conference on Computer Vision (ICCV'90) held in Osaka. He holds several U.S. and international patents for inventions related to computer vision and robotics. Dr. Nayar was the recipient of the David and Lucile Packard Fellowship for Science and Engineering in 1992 and the National Young Investigator Award from the National Science Foundation in 1993.

Course Syllabus

(Note: these times are tentative and subject to change on the day of the course.)

A. 8:30 - 8:50, 20 min
Introduction (Seitz)
1. Overview of area and the course
2. Acquiring 3D models from images
3. Applications to computer graphics

B. 8:50 - 9:35, 45 min
Sensing for vision and graphics (Nayar)
1. The dimensions of visual sensing
2. Catadioptric vision
3. Panoramic and omnidirectional cameras
4. Spherical mosiacs
6. Radiometric self calibration
7. High dynamic range imaging

C. 9:35 - 10:15, 40 min
Overview of passive vision techniques (Seitz)
1. Cues for 3D inference (parallax, shading, focus, texture)
2. Camera Calibration
3. Single view techniques
4. Multiple view techniques
   - Stereo
   - Structure from motion
   - Photometric stereo
5. Strengths and Limitations

<> 10:15 - 10:30 **Break**

D. 10:30 - 11:20, 50 min
Façade: modeling architectural scenes (Debevec)

1. Constrained structure recovery
   - Architectural primitives
2. Photogrammetry
   - Recovering camera parameters
   - Importance of user-interaction
3. Model-based stereo
4. Connections to image-based rendering
   - Impact of geometric accuracy on rendering quality
   - Local vs. global 3D models

E. 11:20 - 12:00, 40 min

Voxels from images (Seitz)

1. Voxel-based scene representation
2. Volume intersection
   - Shape from silhouettes
3. Voxel coloring
   - Modeling radiance
   - Plane-sweep visibility
4. Space carving
   - General visibility modeling
   - Ambiguities in scene reconstruction
5. Related Techniques

<> 12:00 - 1:30 **Lunch**

F. 1:30 - 2:10, 40 min

Overview of active vision techniques (Curless)

1. Imaging radar
   - Time of flight
   - Amplititude modulation
2. Optical triangulation
   - Scanning with points and stripes
   - Spacetime analysis
3. Interferometry
   - Moire
4. Structured light applied to passive vision
   - Stereo
   - Depth from defocus
5. Reflectance capture
   - From shape-directed lighting
   - Using additional lighting

G. 2:10 - 2:50, 40 min

Desktop 3D Photography (Bouguet)

1. Traditional scanning is expensive, but...
   desklamp + pencil = structured light
2. Geometry of shadow scanning
   - Indoor: on the desktop
   - Outdoor: the sun as structured light
3. Image processing: Spacetime analysis for better accuracies
4. Calibration issues
   - Camera calibration
   - Light source calibration
5. Experimental results (indoor and outdoor)
6. Error analysis and Real-time implementation

H. 2:50 - 3:35, 45 min

Shape and appearance from images and range data (Curless)

1. Registration
2. Reconstruction from point clouds
3. Reconstruction from range images
   - Zippering
   - Volumetric merging
4. Modeling appearance

<> 3:35 - 3:50 **Break**

I. 3:50 – 5:00, 70 min

Application: The Digital Michelangelo Project (Levoy)

1. Scholarly and commercial motivations
2. Hardware and software
3. Scanning the David
4. Acquiring a big light field
5. Implications of 3D scanning
6. Lessons learned from the project
7. The problem of the Forma Urbis Romae

<> **Adjourn**

Contents

8. Overview of active vision techniques (Brian Curless)

Slides

Paper

Better Optical Triangulation through Spacetime Analysis
B. Curless and M. Levoy

9. Desktop 3D photography (Jean-Yves Bouguet)

Slides

Papers

3D Photography on Your Desk
J. Y. Bouguet and P. Perona

3D Photography Using Shadows in Dual-space Geometry
J. Y. Bouguet and P. Perona

VRML models (on CDROM only)

10. Shape and appearance from images and range data (Brian Curless)

Slides

Papers

Surface Reconstruction from Unorganized Points
H. Hoppe, T. DeRose, and T. Duchamp

Mesh Optimization
H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle

Zippered Polygon Meshes from Range Images
G. Turk and M. Levoy

A Volumetric Method for Building Complex Models from Range Images
B. Curless and M. Levoy

11. Application: The Digital Michelangelo Project (Marc Levoy)

Slides

Bibliography of papers included in this volume

- C. Tomasi and T. Kanade, *Shape and Motion from Image Streams under Orthography: A Factorization Method*, International Journal of Computer Vision, Vol. 9, No. 2, 1992, pp. 137-154.
- M. Okutomi and T. Kanade, *A Multiple-Baseline Stereo*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 15, No. 4, 1993, pp. 353-363.
- A. Criminisi, I. Reid, and A. Zisserman, *Single View Metrology*, Proc. 7th International Conference on Computer Vision, pp. 434-442, 1999.
- S. M. Seitz and C.R. Dyer, *Photorealistic Scene Reconstruction by Voxel Coloring*, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 1997, pp. 1067-1073.
- K. N. Kutulakos and S. M. Seitz, *A Theory of Shape by Space Carving*, International Journal of Computer Vision, 2000, to appear.
- P. E. Debevec, C. J. Taylor, and J. Malik, *Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach*, Proc. ACM SIGGRAPH 96, 1996, pp. 11-20.
- G.D. Borshukov and P. Debevec, *Recovering Arches in Façade Using Ray-Plane Intersections in 3-D.*
- G.D. Borshukov and P. Debevec, *Recovering the Radius and Offset of a Cross-Section in SORs Using Minimum Distance between Two Rays in 3-D.*
- B. Curless and M. Levoy, *Better Optical Triangulation through Spacetime Analysis*, IEEE International Conference on Computer Vision, 1995, pp. 987-994.
- J. Y. Bouguet and P. Perona, *3D photography on your desk*, Proc. IEEE International Conference on Computer Vision, 1998, pp. 43-50.
- J. Y. Bouguet and P. Perona, *3D Photography Using Shadows in Dual-space Geometry*, International Journal of Computer Vision, Vol. 35, No. 2, Nov./Dec. 1999, pp. 129-149.
- G. Turk and M. Levoy, *Zippered Polygon Meshes from Range Images*, Proc. ACM SIGGRAPH 94, 1994, pp. 311-318.
- H. Hoppe, T. DeRose, and T. Duchamp, *Surface Reconstruction from Unorganized Points*, Proc. ACM SIGGRAPH 92, 1992, pp. 71-78.
- H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, *Mesh Optimization*, Proc. ACM SIGGRAPH 93, 1993, pp. 19-26.
- B. Curless and M. Levoy, *A Volumetric Method for Building Complex Models from Range Images*, Proc. ACM SIGGRAPH 96, 1996, pp. 303-312.

Introduction

3D photography is an emerging technology for capturing richly detailed models of objects in the real world. Whereas traditional optical cameras capture scene appearance in the form of radiant light energy, 3D photographs measure surface characteristics like 3D geometry and reflectance—exactly what is needed to construct graphical models. Consequently this technology provides a means for acquiring graphical objects and scenes of unprecedented detail and realism by *scanning* them in from the real world.

Methods to digitize and reconstruct the shapes of complex three dimensional objects have evolved rapidly in recent years. The speed and accuracy of digitizing technologies owe much to advances in the areas of physics and electrical engineering, including the development of lasers, CCD's, and high speed sampling and timing circuitry. Such technologies allow us to take detailed shape measurements with precision better than 1 part per 1000 at rates exceeding 10,000 samples per second. To capture the complete shape of an object, many thousands, sometimes millions of samples must be acquired. The resulting mass of data requires algorithms that can efficiently and reliably generate computer models from these samples. The future of 3D photography will see systems that capture precise geometry and reflectance information at even larger spatial scales, enabling the acquisition of landscapes and complex urban scenes, and fast scanners that enable 3D video at real-time rates.

The applications of 3D photography are wide-ranging and include manufacturing, virtual simulation, human-computer interaction, scientific exploration, medicine, and consumer marketing.

Dissemination of museum artifacts

Museum artifacts represent one-of-a-kind objects that attract the interest of scientists and lay people world-wide. Traditionally, to visualize these objects, it has been necessary to visit potentially distant museums or obtain non-interactive images or video sequences. By digitizing these parts, museum curators can make them available for interactive visualization. For scientists, computer models afford the opportunity to study and measure artifacts remotely using powerful computer tools. A case in point is the *Digital Michelangelo Project* headed by Marc Levoy at Stanford University. The goal of this multi-year project is to create a high-quality 3D computer archive of the sculptures and architecture of Michelangelo. This course features the first presentation of the Digital Michelangelo Project to the SIGGRAPH community.

Special effects, games, and virtual worlds

Synthetic imagery is playing an increasingly prominent role in creating special effects for cinema. In addition, video games and gaming hardware are moving steadily toward interactive 3D graphics. Virtual reality as a means of simulating worlds of experience is also growing in popularity. All of these applications require 3D models that may be taken from real life or from sculptures created by artists. Digitizing the shapes of physical models will be essential to populating these synthetic environments.

Reverse engineering

Many manufacturable parts are currently designed with Computer Aided Design (CAD) software. However, in some instances, a mechanical part exists and belongs to a working system but has no computer model needed to regenerate the part. This is frequently the case for machines currently in service that were designed before the advent of computers and CAD systems, as well as for parts that were hand-tuned to fit into existing machinery. If such a part breaks, and neither spare parts nor casting molds exist, then it may be possible to remove a part from a working system and digitize it precisely for re-manufacture.

Collaborative design

While CAD tools can be helpful in designing parts, in some cases the most intuitive design method is physical interaction with the model. This is especially true when the model must have esthetic appeal, such as the exteriors of consumer products ranging from perfume bottles to automobiles. Frequently, companies employ sculptors to design these models in a medium such as clay. Once the sculpture is ready, it may be digitized and reconstructed on a computer. The computer model is

then suitable for dissemination to local engineers or remote clients for careful review, or it may serve as a starting point for constructing a CAD model suitable for manufacture.

Medicine

Applications of 3D Photography in medicine are wide ranging as well. Prosthetics can be custom designed when the dimensions of the patient are known to high precision. Plastic surgeons can use the shape of an individual's face to model tissue scarring processes and visualize the outcomes of surgery. When performing radiation treatment, a model of the patient's shape can help guide the doctor in directing the radiation accurately.

Web commerce

As the World Wide Web provides a backbone for interaction over the Internet, commercial vendors are taking advantage of the ability to market products through this medium. By making 3D models of their products available over the Web, vendors can allow the customer to explore their products interactively. Standards for disseminating 3D models over the web are already underway (e.g., the Virtual Reality Modeling Language (VRML)).

Course Objectives

In this course we will focus on the technology underlying the field of 3D photography, focusing on the current state-of-the-art and the principles underlying several leading approaches. Our intent is to cover the fundamentals but also to give an understanding of current research directions and exciting applications. With these objectives in mind we have designed a course that brings together several leading researchers and practitioners to present state-of-the-art 3D photography approaches from the ground up.

The course will cover a variety of methods for recovering shape and reflectance from images. The course begins with novel 2D sensing technologies such as catadioptric cameras and high dynamic range sensors. Several passive vision methods will be presented, including stereo, structure from motion, shape from shading, volume intersection, and voxel coloring. Active vision methods will include imaging radar, optical triangulation, moire, active stereo, active depth from defocus, and desktop shadow striping. The course concludes with a field study: capturing 3D photographs of Michelangelo's statues.

We hope the material in this volume will prove useful to you to help gain a deeper understanding of the concepts behind 3D photography, but moreover to help you build your own 3D photography system on your desktop. To this end, we have provided material that we believe is sufficient to design and build practical 3D photography systems from scratch.

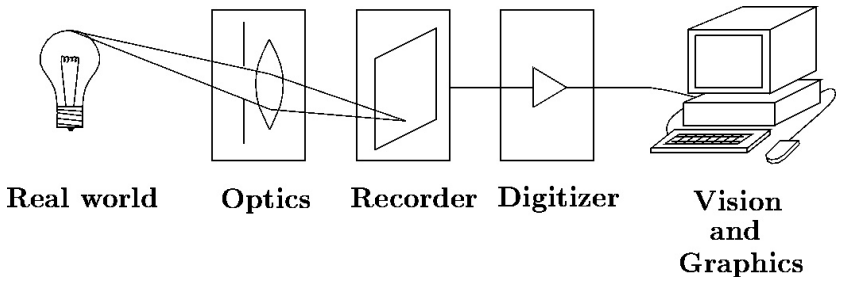
Steve Seitz
Brian Curless
April 2000

SIGGRAPH 2000 Course on
3D Photography

Acquiring Images

Brian Curless
University of Washington

The Imaging Pipeline



Overview

Pinhole camera

Lenses

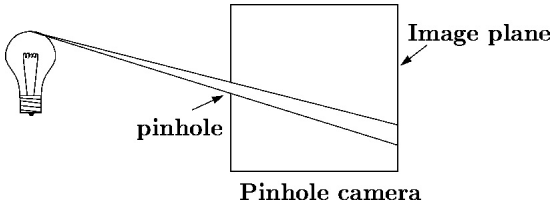
- **Principles of operation**
- **Limitations**

Charge-coupled devices

- **Principles of operation**
- **Limitations**

The pinhole camera

The first camera - "camera obscura" - known to Aristotle.



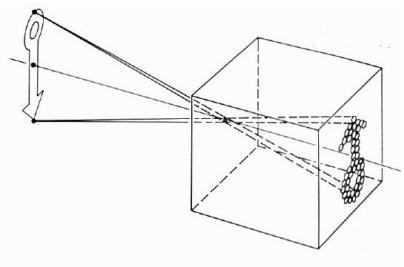
Pinhole camera

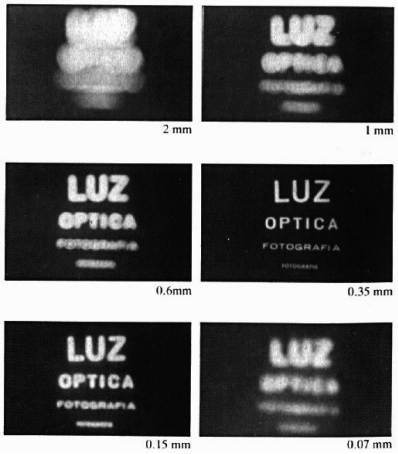
Small aperture = high fidelity

***but* requires long exposure or bright illumination**

Pinhole camera

If aperture is too small, then diffraction causes blur.

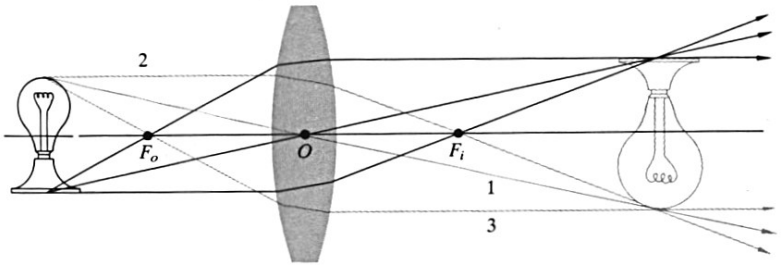




[Figure from Hecht87]

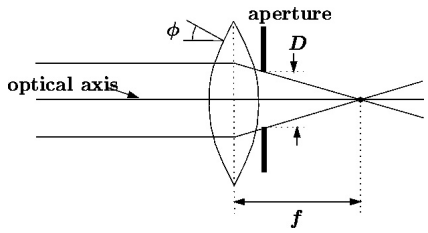
Lenses

Lenses focus a bundle of rays to one point.
=> can have larger aperture.



[Figure from Hecht87]

Lenses



A lens images a bundle of parallel rays to a focal point at a distance, *f*, beyond the plane of the lens.

Note: *f* is a function of the index of refraction of the lens.

An aperture of diameter, *D*, restricts the extent of the bundle of refracted rays.

Lenses

For economical manufacture, lens surfaces are usually spherical.

A spherical lens is behaves ideally if Φ is small:

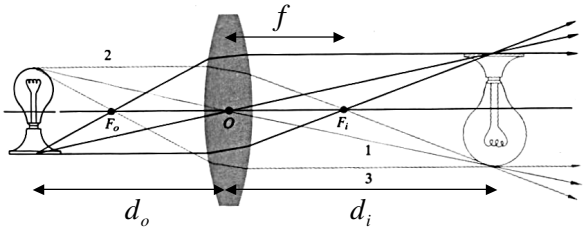
$$\sin\phi = \phi - \frac{\phi^3}{3!} + \frac{\phi^5}{5!} - \ldots \approx \phi$$

The angle restriction means we consider rays near the optical axis -- "paraxial rays."

Lenses

For a "thin" lens, we ignore lens thickness, and the paraxial approximation leads to the familiar Gaussian lens formula:

$$\frac{1}{d_o}+\frac{1}{d_i}=\frac{1}{f}$$



[Figure from Hecht87]

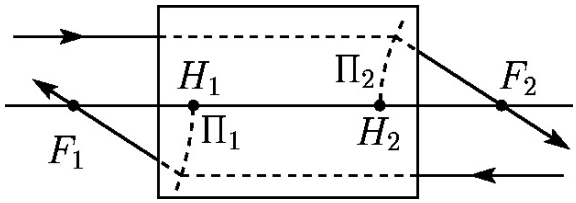
Cardinal points of a lens system

Most cameras do not consist of a single thin lens. Rather, they contain multiple lenses, some thick.

A system of lenses can be treated as a "black box" characterized by its *cardinal points*.

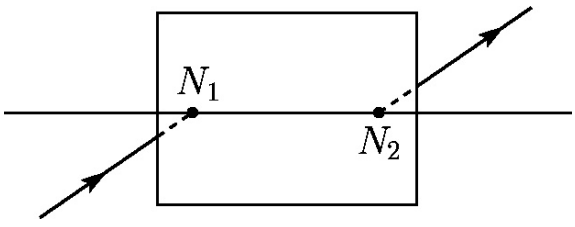
Focal and principal points

The focal and principal points and the principal "planes" describe the paths of rays parallel to the optical axis.



Nodal points

The nodal points describe the paths of rays that are not refracted, but are translated down the optical axis.



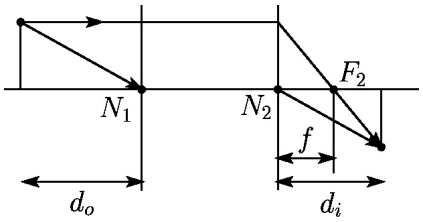
Cardinal points of a lens system

If:

- ***the optical system is surrounded by air***
- ***and the principal planes are assumed planar***

then

- ***the nodal and principal points are the same***



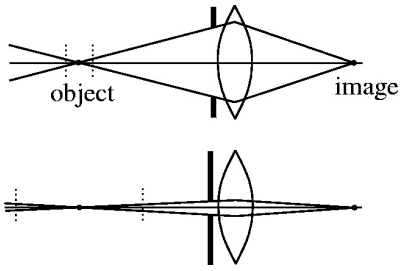
The system still obeys Gauss's law, but all distances are now relative to the principal points.

Depth of field

Lens systems do have some limitations.

First, points that are not in the object plane will appear out of focus.

The *depth of field* is a measure of how far from the object plane points can be before appearing "too blurry."



Monochromatic aberrations

Allowing for the next higher terms in the sinΦ approximation:

$$\sin\phi = \phi - \frac{\phi^3}{3!} + \frac{\phi^5}{5!} - \ldots \approx \phi - \frac{\phi^3}{3!}$$

…we arrive at the third order theory. Deviations from ideal optics are called the *primary* or *Seidel aberrations*:

- **Spherical aberration**
- **Coma**
- **Astigmatism**
- **Petzval curvature**
- **Distortion**

Distortion

Cause:

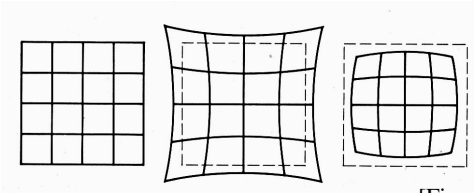
Oblique rays bent by the edges of the lens

Effect:

Non-radial lines curve out (barrel) or curve in (pin cushion)

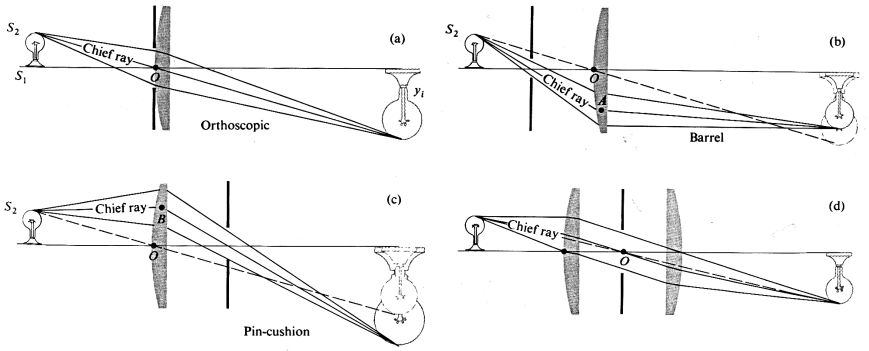
Ways of improving:

Symmetrical design.



[Figure from Hecht87]

Distortion



[Figures from Hecht87]

Chromatic aberration

Cause:

Index of refraction varies with wavelength.

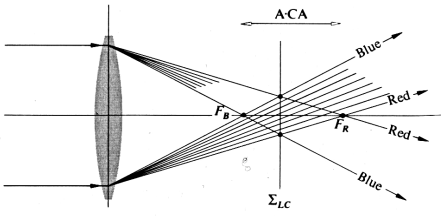
Effect:

Focus shifts with color, colored fringes on highlights

Ways of improving:

Achromatic designs

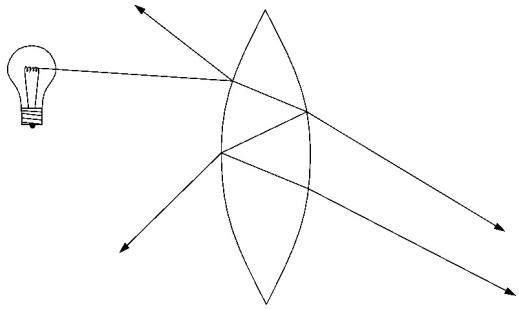
[Figure from Hecht87]



Flare

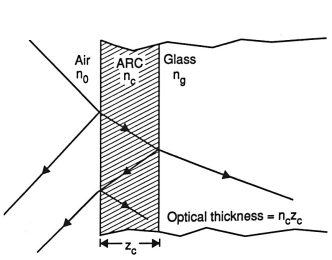
Light rays refract *and reflect* at the interfaces between air and the lens.

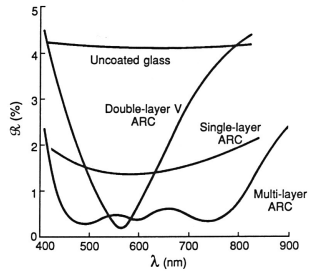
The "stray" light is not focused at the desired point in the image, resulting in ghosts or haziness.



Optical coatings

Optical coatings are tuned to cancel out reflections at certain angles and wavelengths.





[Figure from Burke96]

Vignetting

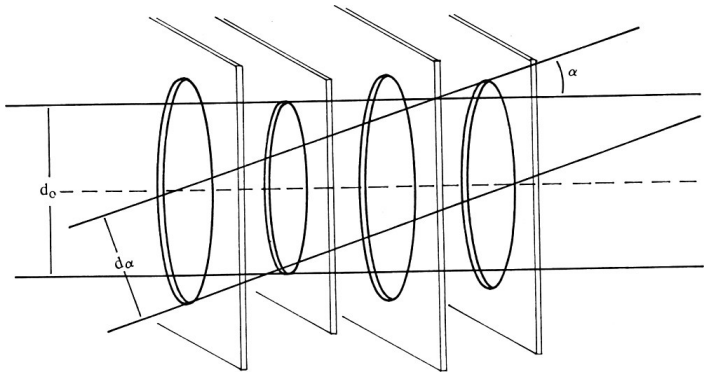
Light rays oblique to the lens will deliver less power per unit area (irradiance) due to:

- **mechanical vignetting**
- **optical vignetting**

Result: darkening at the edges of the image.

Mechanical vignetting

Occlusion by apertures and lens extents results in mechanical vignetting.



[Figure from Horn87]

Optical vignetting

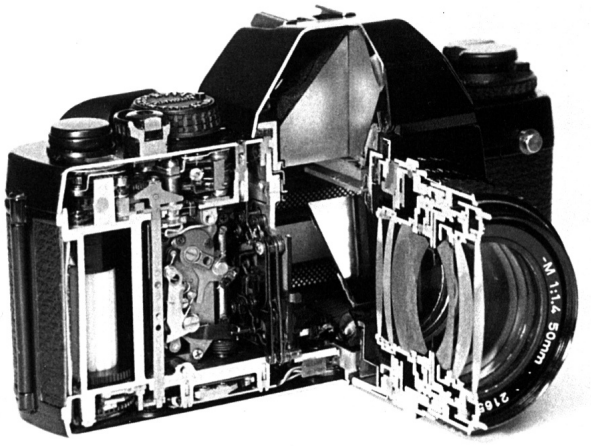
At grazing angles, less power per unit area is delivered to the image plane -- optical vignetting.

The irradiance at the sensor varies with the angle to the image plane, θ, as:

$$E \sim L\left(\frac{D}{f}\right)^2 \cos^4\theta$$

Note also: the irradiance is proportional to the radiance along the path.

The art of optical design...



[Figure from Goldberg92]

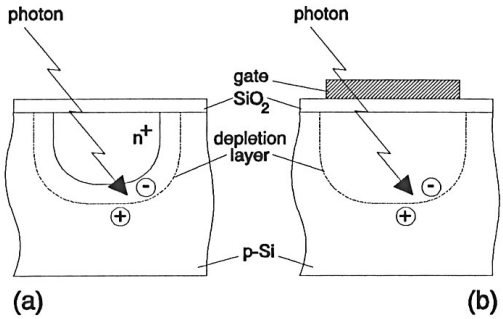
Charge-coupled devices

The most popular image recording technology for 3D photography is the charge-coupled device (CCD).

- ***Image is readily digitized***
- ***CCD cells respond linearly to irradiance***
  - > *But, camera makers often re-map the values to correct for TV monitor gamma or to behave like film*
- ***Available at low cost***

Photo-conversion

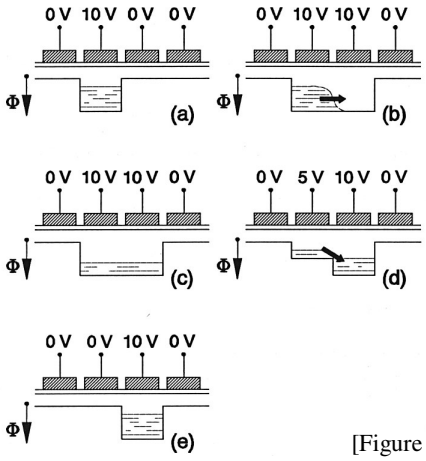
When a MOS capacitor is biased into "deep depletion," it can collect charges generated by photons.



[Figure from Theuwissen87]

Charge transfer

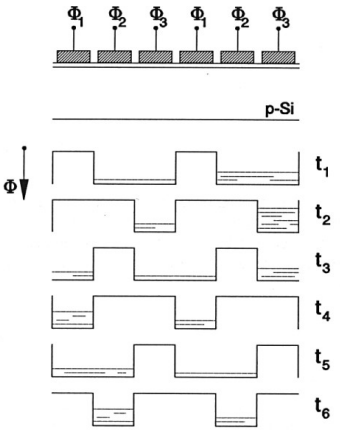
By manipulating voltages of neighboring cells, we can move a bucket of charge one gate to the right.



[Figure from Theuwissen87]

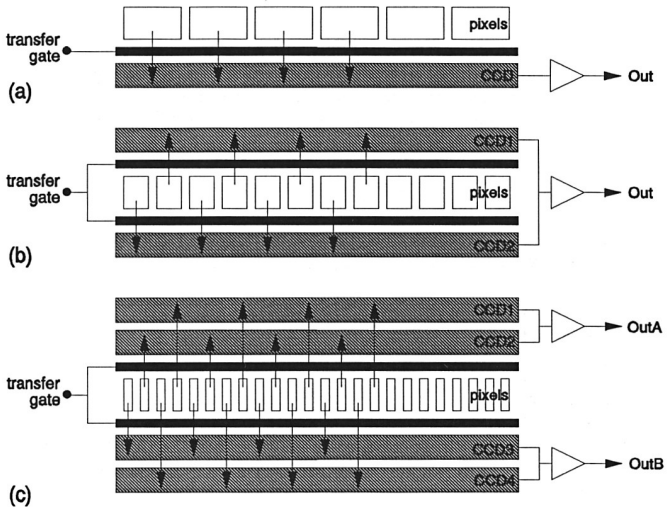
Three-phase clocking system

With three gates, we can move disjoint charge packets along a linear array of CCD's.



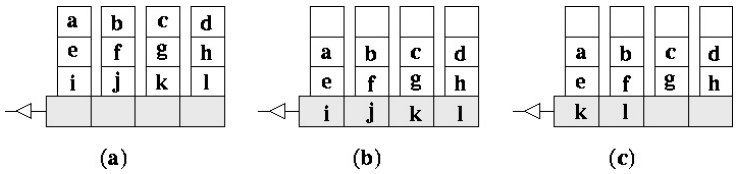
[Figure from Theuwissen87]

Linear array sensors

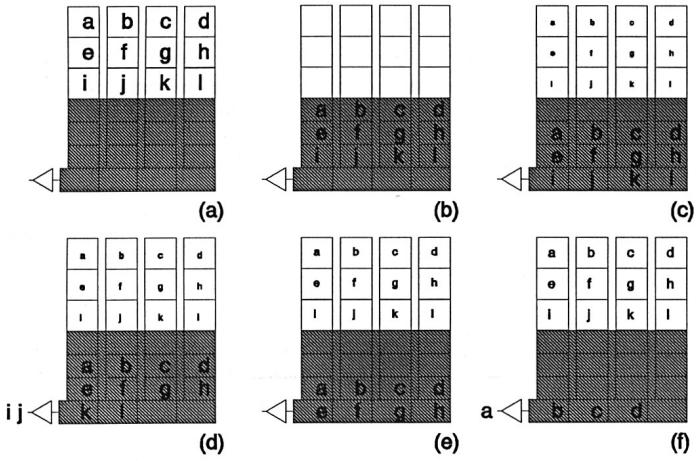


[Figure from Theuwissen87]

Full frame CCD

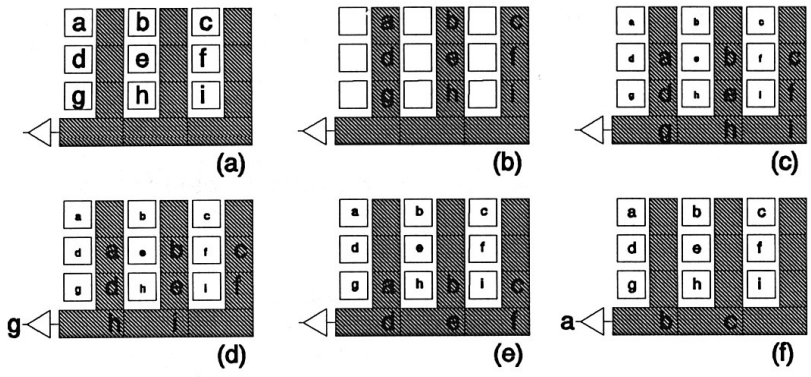


Frame transfer (FT) CCD



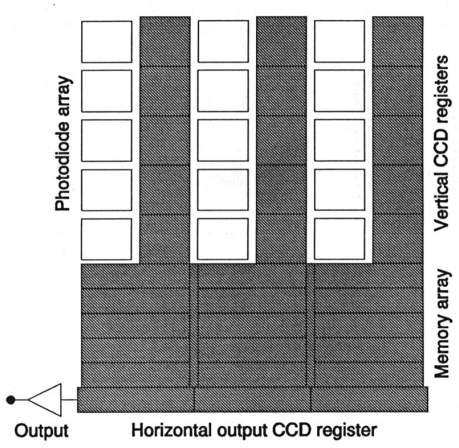
[Figure from Theuwissen87]

Interline transfer (IT) CCD



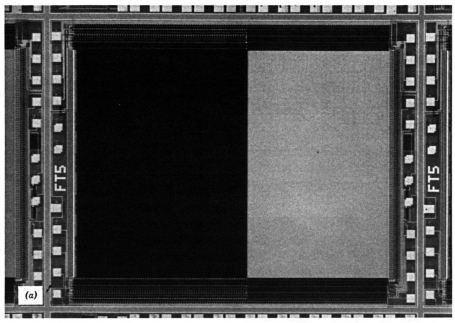
[Figure from Theuwissen87]

Frame interline transfer (FIT) CCD's

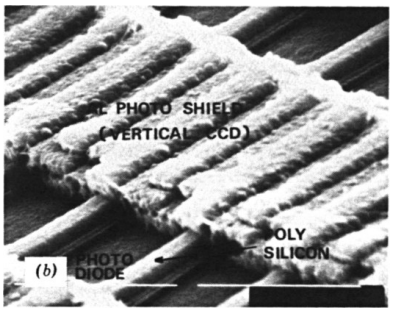


[Figure from Theuwissen87]

A closer look...



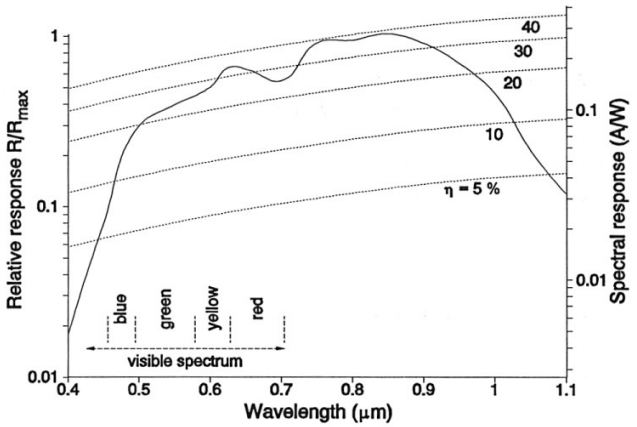
Frame transfer



Interline transfer

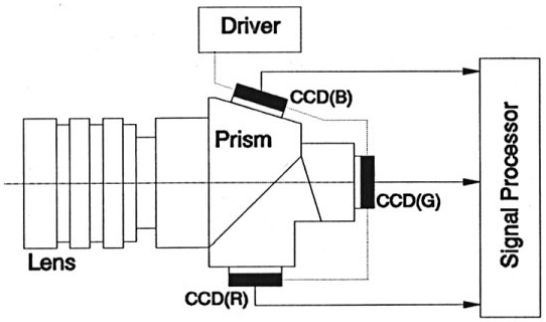
[Figure from Muller86]

Spectral response



[Figure from Theuwissen87]

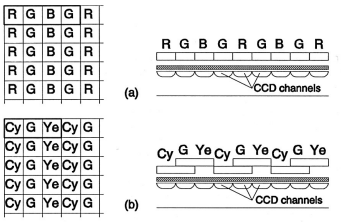
3-chip color cameras

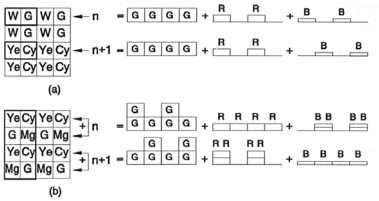


[Figure from Theuwissen87]

Single chip color filters

Stripe filters





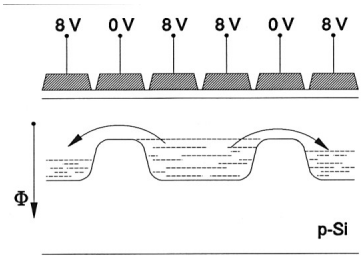
Mosaic filters

[Figures from Theuwissen87]

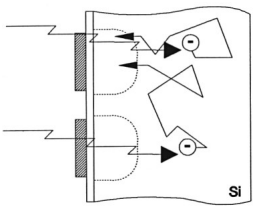
Limitations of CCD's

- **Smear vs. aliasing**
- **Blooming**
- **Diffusion**
- **Transfer efficiency**
- **Noise**
  - ***Processing defects***
  - ***Dark-current noise***
  - ***Output amplifier noise***
- **Dynamic range**

Blooming and diffusion



Blooming



Diffusion

[Figures from Theuwissen87]

Bibliography

Burke, M.W. Image Acquisition: Handbook of Machine Vision Engineering. Volume 1. New York. Chapman and Hall, 1996.

Goldberg, N. Camera Technology: The Dark Side of the Lens. Boston, Mass, Academic Press, Inc., 1992.

Hecht, E. Optics. Reading, Mass., Addison-Wesley Pub. Co., 2nd ed., 1987.

Horn, B.K.P. Robot Vision. Cambridgge, Mass., MIT Press, 1986.

Muller, R. and Kamins, T. Device Electronics for Integrated Circuits, 2nd Edition. John Wiley and Sons, New York, 1986.

Theuwissen, A. Solid-State Imaging with Charge-Coupled Devices. Kluwer Academic Publishers, Boston, 1995.

UNCONVENTIONAL VISION SENSORS

Shree K. Nayar

ACM SIGGRAPH 2000, Course Notes

Computer Science Department
Columbia University

Sponsors:
NSF, ONR MURI, DARPA,
Packard Foundation

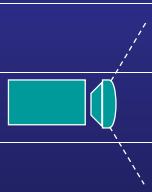
Depth Sensors

Depth from Focus and Defocus : (Nayar, Watanabe, Noguchi 95)

Performance : 512 x 480 Depth map at 30 frames per sec.

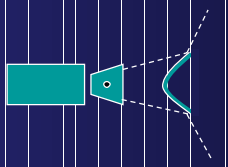
Different Approaches

Fish Eye Lens :



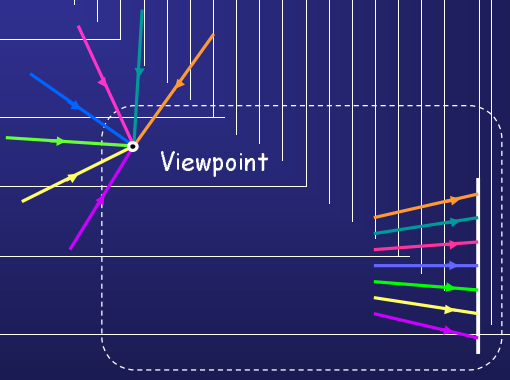
Examples: Wood 1906, Miyamoto 64, Hall 87, Zimmerman 93 , Poelstra 96 , Kuban et al. 94

Catadioptric Imaging :

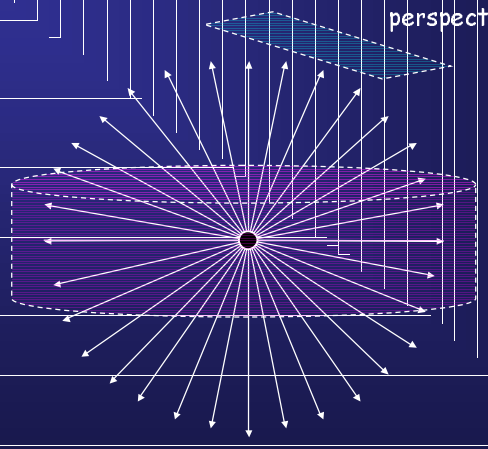


Examples: Rees 70, Rosendahl 83, Charles 87, Nayar 88, Yagi 90, Hong 91, Powell 95, Yamazawa 95, Bogner 95, Nalwa 96, Nayar 97, Chahl & Srinivasan 97

Single Viewpoint Imaging System

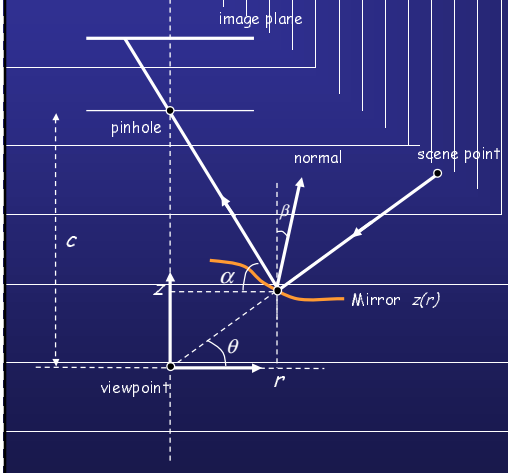


Single Viewpoint Omnidirectional Imaging



Single Mirror Systems : Problem Formulation

(Baker & Nayar 98)



Reflecting Mirror Point :

$$\tan\theta = \frac{z}{r}$$

Perspective Projection :

$$\tan\alpha = \frac{(c - z)}{r}$$

Specular Reflection :

$$2\beta = \alpha - \theta$$

Local Mirror Slope :

$$\frac{dz}{dr} = -\tan\beta$$

Single Mirror Systems : Solution

Quadratic First - Order Differential Equation :

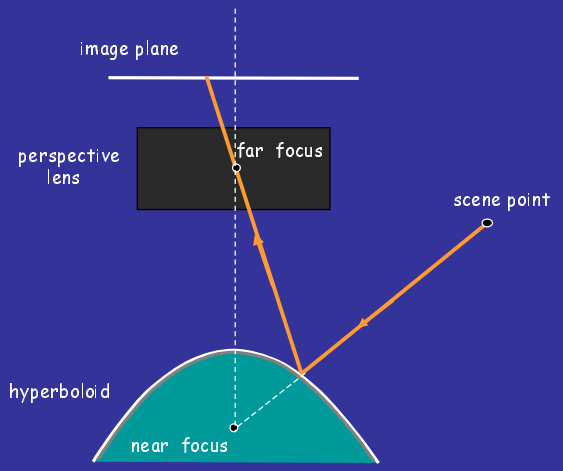
$$\frac{-2\frac{dz}{dr}}{1-\left(\frac{dz}{dr}\right)^2} = \frac{(c-2z)\,r}{r^2+cz-z^2}$$

Complete Class of Single Viewpoint Mirrors :

$$\left(z-\frac{c}{2}\right)^2 - r^2\left(\frac{k}{2}-1\right) = \frac{c^2}{4}\left(\frac{k-2}{k}\right) \qquad (k \geq 2)$$

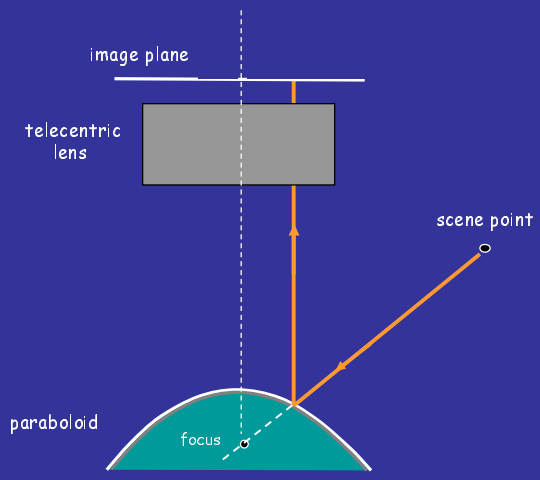
$$\left(z-\frac{c}{2}\right)^2 - r^2\left(1+\frac{c^2}{2k}\right) = \left(\frac{2k-c^2}{4}\right) \qquad (k > 0)$$

Single Mirror Systems



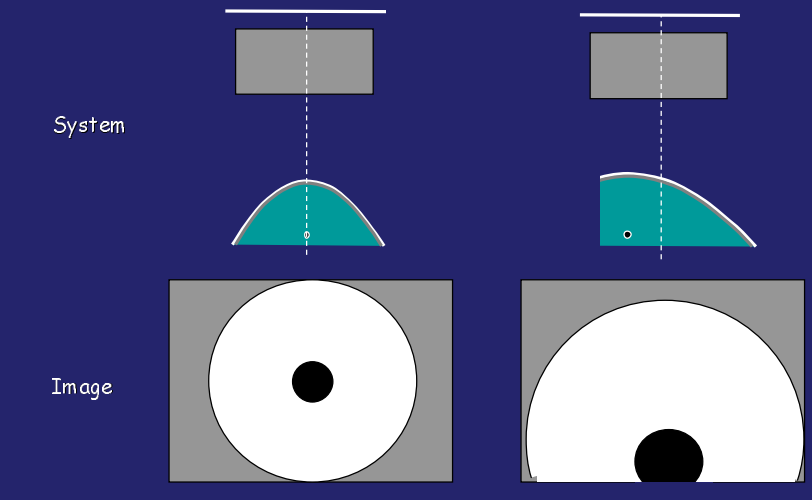
(Rees 70, Yamazawa 93)

Single Mirror Systems



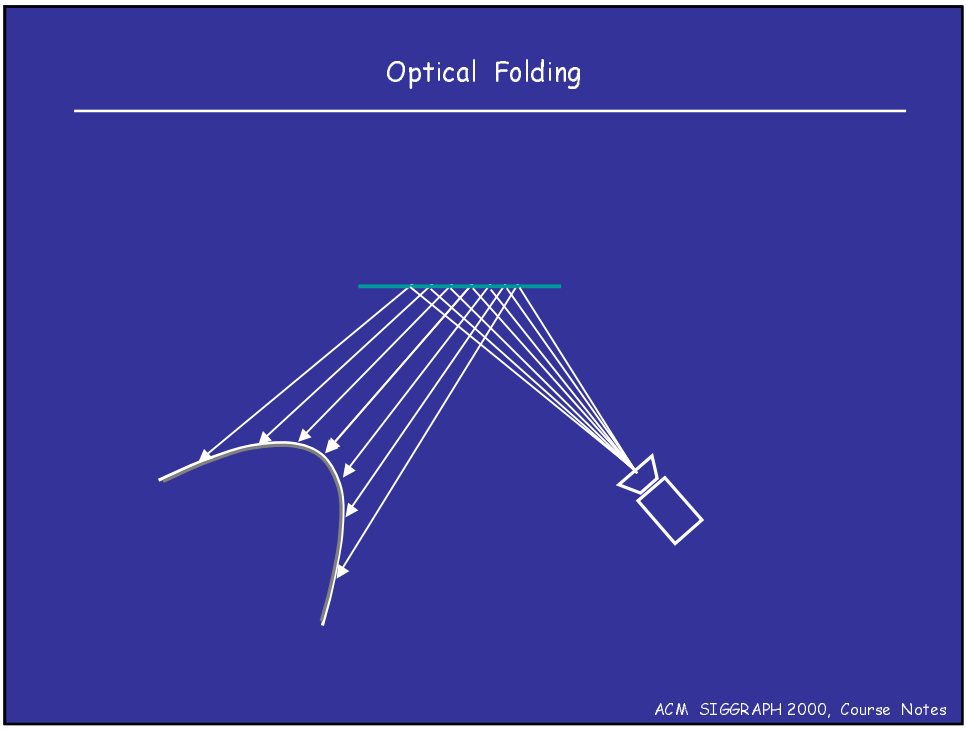
(Nayar 97)

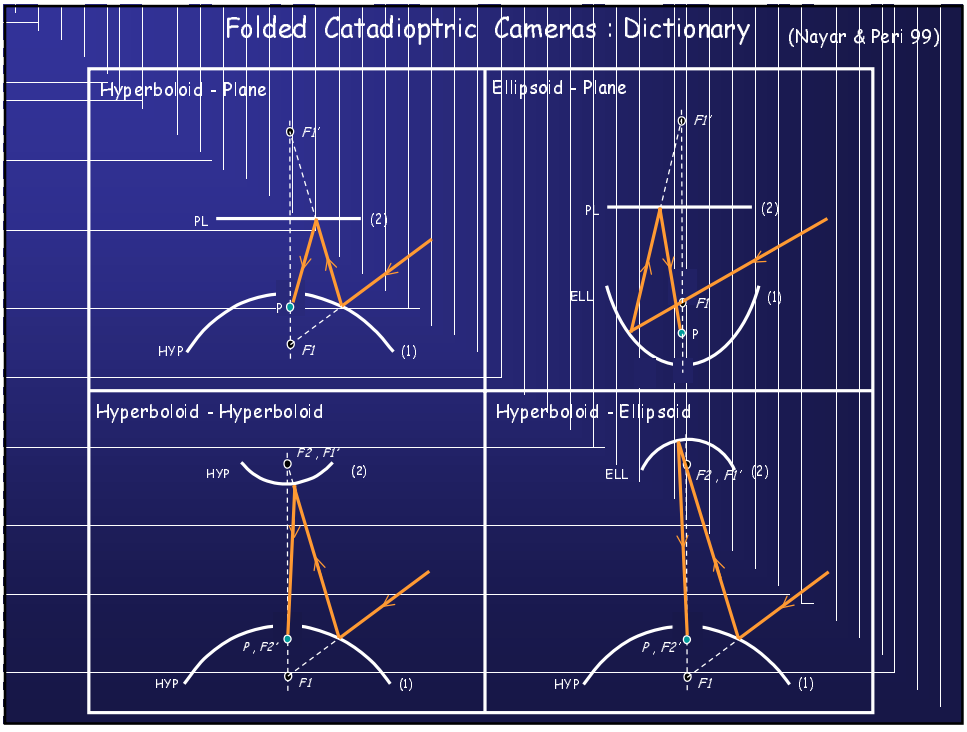
Flexibility: Resolution vs. FOV

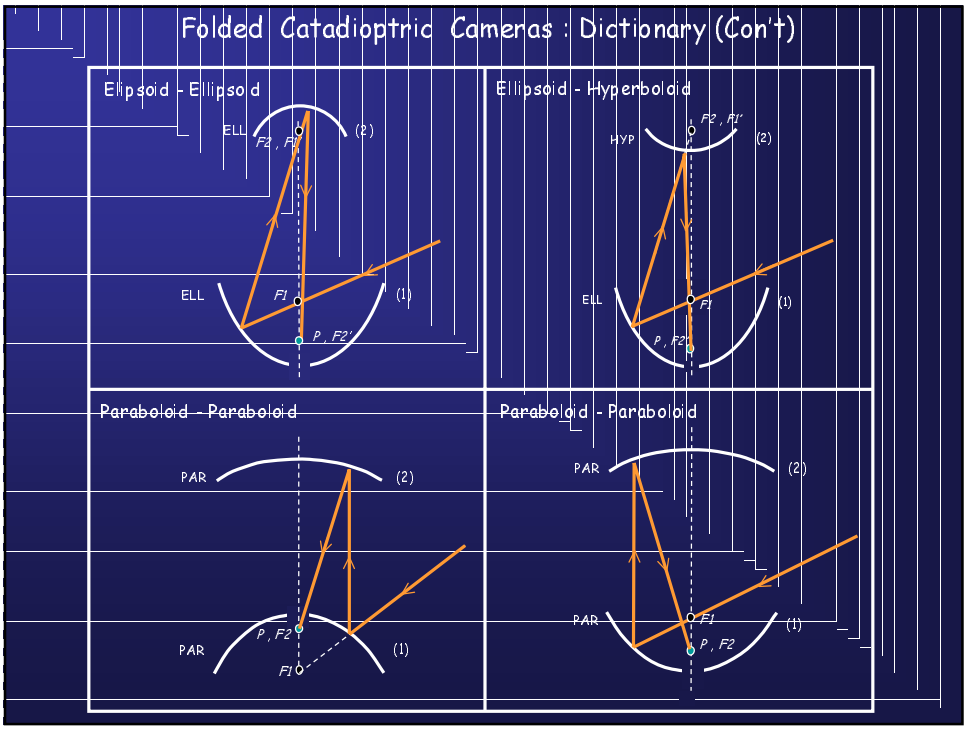


OneShot by RemoteReality

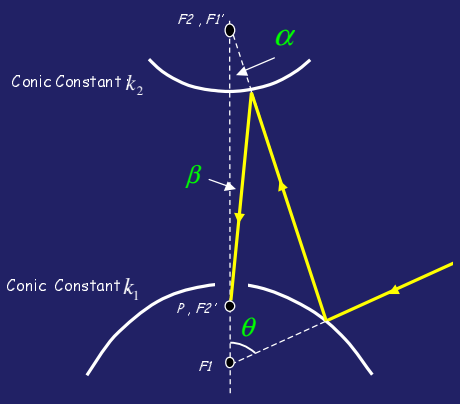
Megapixel (1680 x 1260)
360 degree still camera


Optical Folding
ACM SIGGRAPH 2000, Course Notes


Folded Catadioptric Cameras : Dictionary
(Nayar & Peri 99)
Hyperboloid - Plane
F1'
PL
(2)
P
HYP
F1
(1)
Ellipsoid - Plane
F1'
PL
(2)
ELL
F1
P
(1)
Hyperboloid - Hyperboloid
HYP
F2 , F1'
(2)
P , F2'
F1
(1)
Hyperboloid - Ellipsoid
ELL
F2 , F1'
(2)
P , F2'
HYP
F1
(1)



Equivalent Single Mirror System



Compression due to Primary Mirror :

$$\tan \alpha = \frac{(1+k_1)\sin\theta}{2\sqrt{-k_1}+(1-k_1)\cos\theta}$$

Compression due to Primary Mirror :

$$\tan \beta = \frac{(1+k_2)\sin\alpha}{2\sqrt{-k_2}+(1-k_2)\cos\alpha}$$

ACM SIGGRAPH 2000, Course Notes

Equivalent Single Mirror Systems

Total Compression for Folded System :

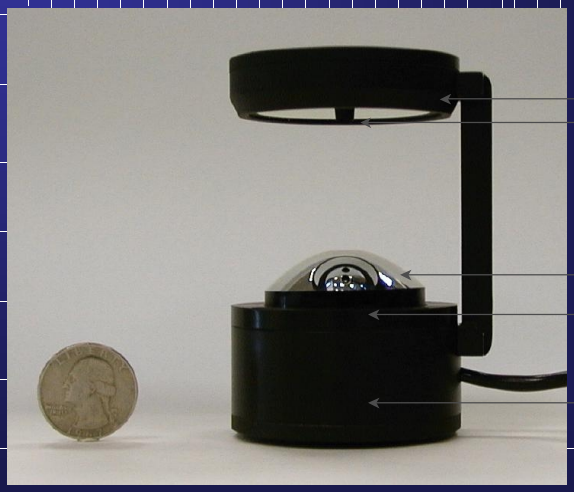
$$\tan^{-}\beta = -\frac{(1+k_1)(1+k_2)\sin\theta}{[2\sqrt{-k_1}+\sqrt{-k_2}-k_1\sqrt{-k_2}+\sqrt{-k_1}k_2 + (1-k_1(k_2-1)+4\sqrt{-k_1}\sqrt{-k_2}-k_2)\cos\theta]}$$

Conic Constant of Equivalent Single Mirror :

$$k_e = \left(\frac{\sqrt{-k_1}+\sqrt{-k_2}}{1+\sqrt{-k_1}\sqrt{-k_2}}\right)^2$$

Compact Panoramic Camera

(Nayar & Peri 99)



SIZE : 9 CM X 5 CM

Urbie : Tactical Mobile Robot

Collaborators : JPL , IS Robotics , CMU , Columbia , CycloVision

Sponsor : DARPA TMR Project

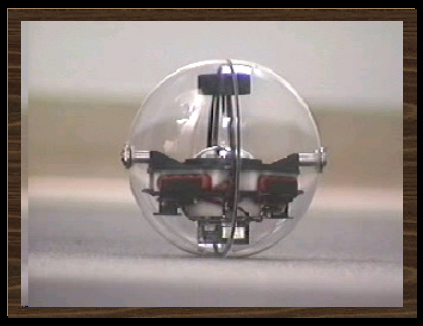
ACM SIGGRAPH 2000, Course Notes

Deployable Omnicameras

COLLABORATION: **COLUMBIA** (OPTICS AND IMAGING) AND **CMU** (MECHANICS - SCHEMPF GROUP)

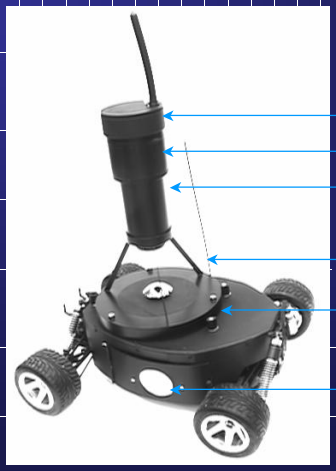
ACM SIGGRAPH 2000, Course Notes

Columbia-CMU Cyclops

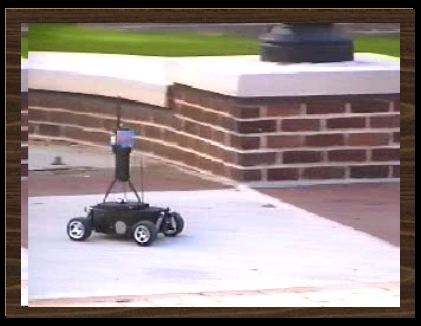


Remote Controlled ParaRover

(Lok & Nayar 98)
(Boult et al. 98)



ParaRover

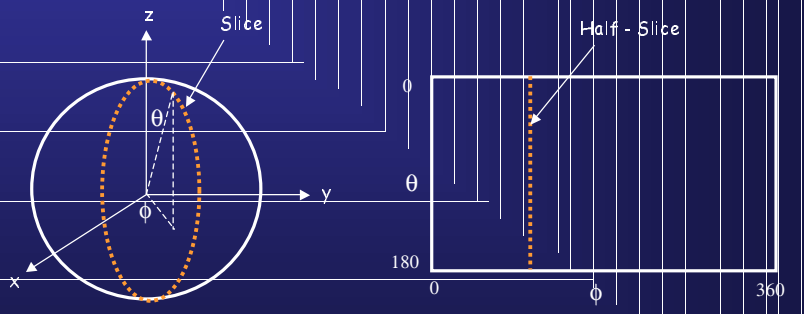


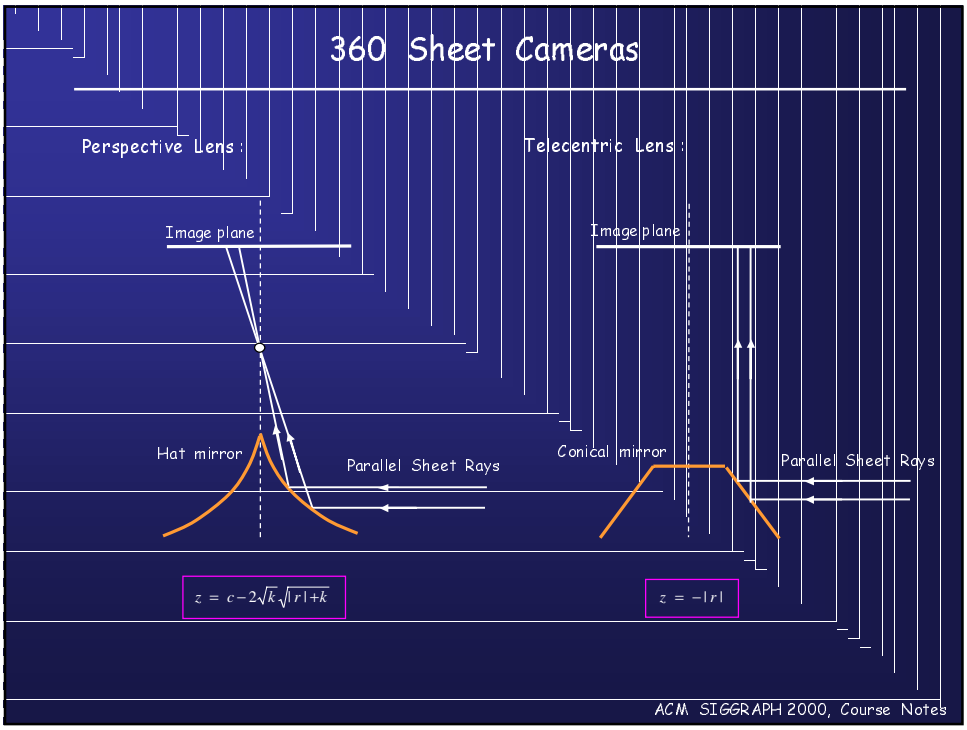
Spherical Mosaics from Slices

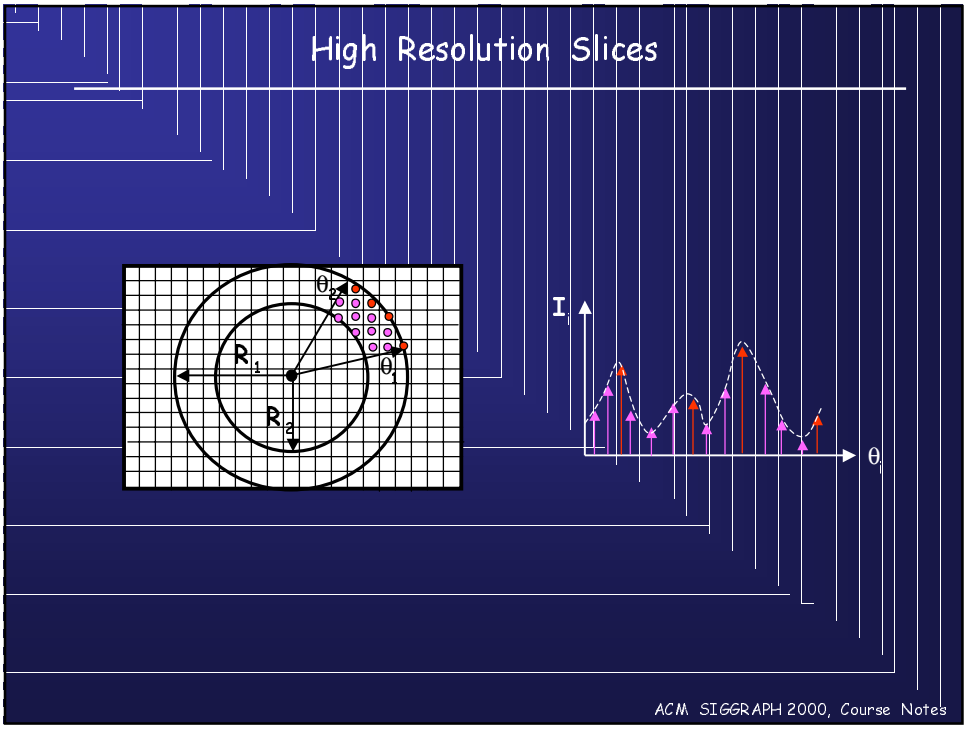
(Nayar & Karmarkar 99)

Unit Sphere :

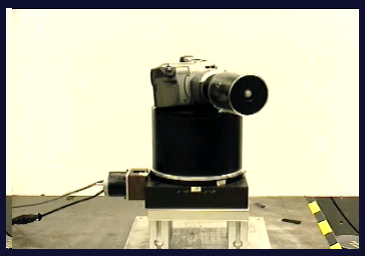
Spherical Panorama :

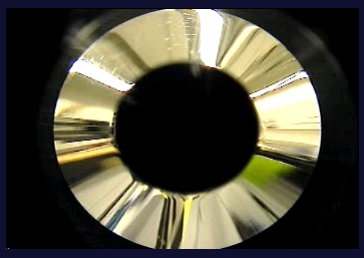



360 Sheet Cameras
Perspective Lens :
Telecentric Lens :
Image plane
Image plane
Hat mirror
Parallel Sheet Rays
Conical mirror
Parallel Sheet Rays
$z = c - 2\sqrt{k}\sqrt{|r|+k}$
$z = -|r|$
ACM SIGGRAPH 2000, Course Notes

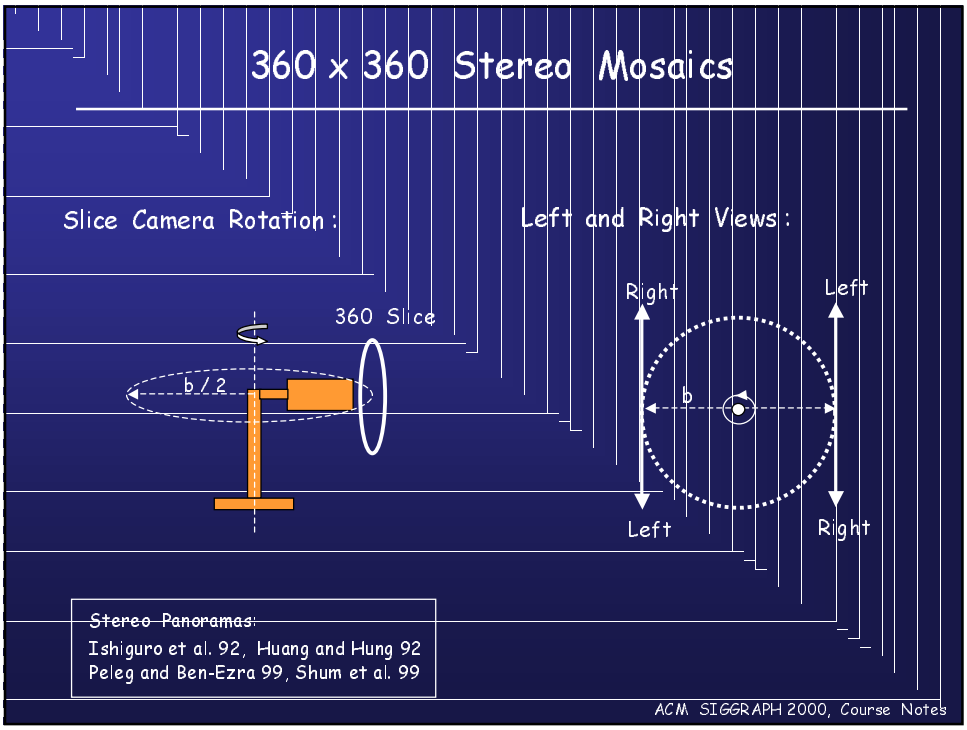

High Resolution Slices
R_1
R_2
θ_1
θ_2
I
θ
ACM SIGGRAPH 2000, Course Notes

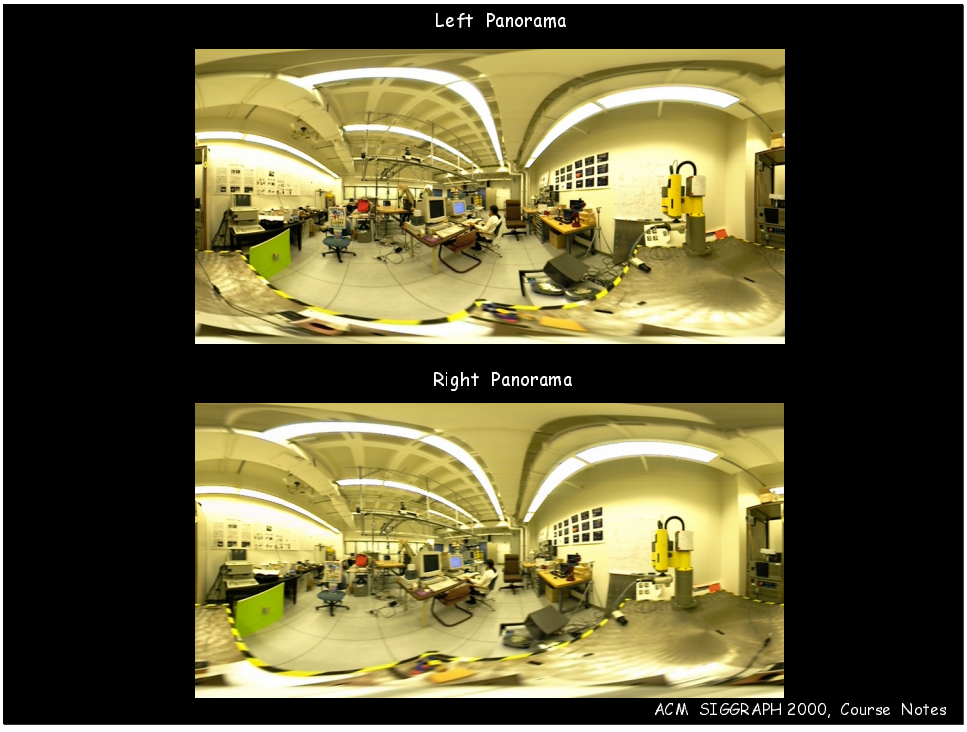
360 Sheet Camera Rotation



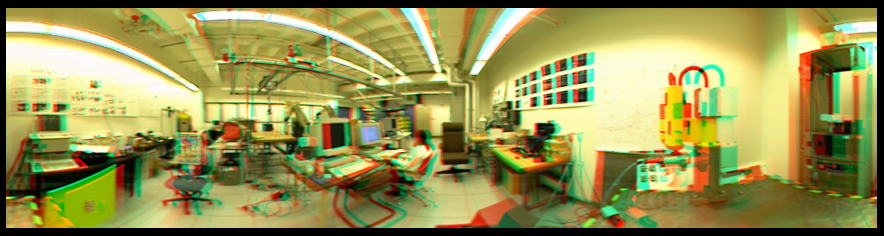


360 x 360 Spherical Mosaic


360 x 360 Stereo Mosaics
Slice Camera Rotation :
Left and Right Views :
360 Slice
b / 2
Right
Left
b
Left
Right
Stereo Panoramas:
Ishiguro et al. 92, Huang and Hung 92
Peleg and Ben-Ezra 99, Shum et al. 99
ACM SIGGRAPH 2000, Course Notes


Left Panorama
Right Panorama
ACM SIGGRAPH 2000, Course Notes

Stereoscopic Panorama



Problem of Dynamic Range

(Mitsunaga & Nayar 99)

- 8-bit Images and the Real World

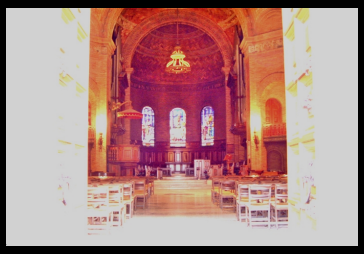


Image M_1
(High exposure)

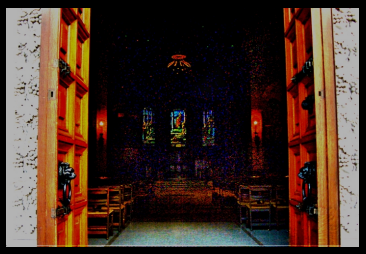
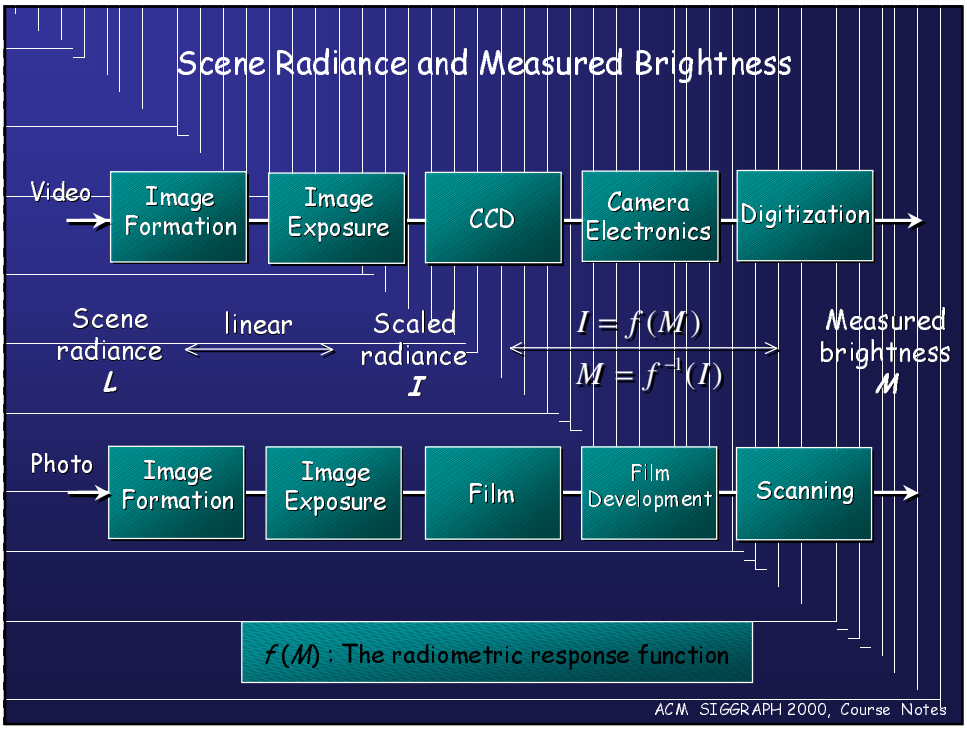


Image M_2
(Low exposure)



Response Function from Images : Previous Work

Mann and Picard (95) :

- Restrictive Model for f : $M = \alpha + \beta I^{\gamma}$
- Precisely Known Exposures

Debevec and Malik (97) :

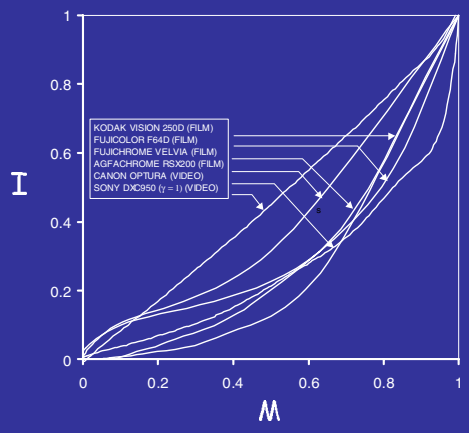
- General Model for f : Only Smoothness Constraint
- Precisely Known Exposures

Goal : Rough Exposure Ratios and Noisy Images

ACM SIGGRAPH 2000, Course Notes

Response Function : General Characteristics

Some Popular Imaging Systems :



High Order Polynomial :

$$I = f(M) = \sum_{n=0}^{N} c_n M^n$$

Scene Radiance — Image Brightness

- f is semi-monotonic
- f is smooth

Self Calibration Algorithm

(Mitsunaga and Nayar 98)

Radiance Ratio: $\frac{I_{p,q}}{I_{p,q+1}} = \frac{f(M_{p,q})}{f(M_{p,q+1})} = R_{q,q+1}$ (exposure ratio)

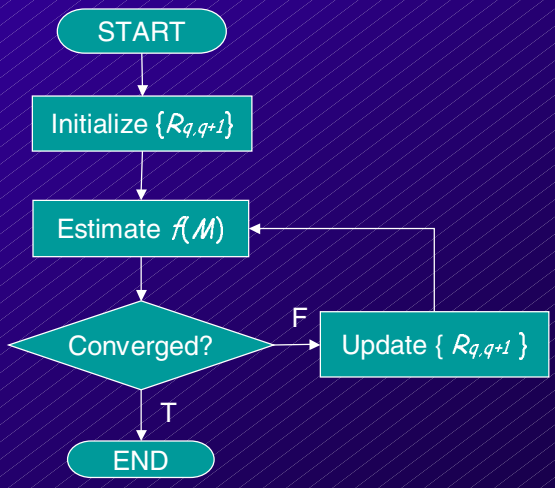
Using Polynomial Model : $\frac{\sum_{n=0}^{N} c_n M_{p,q}^{\ n}}{\sum_{n=0}^{N} c_n M_{p,q+1}^{\ n}} = R_{q,q+1}$

Objective Function : $$\varepsilon = \sum_{q=1}^{Q-1} \sum_{p=1}^{P} \left[\sum_{n=0}^{N} c_n M_{p,q}^{\ n} - R_{q,q+1} \sum_{n=0}^{N} c_n M_{p,q+1}^{\ n} \right]^2$$

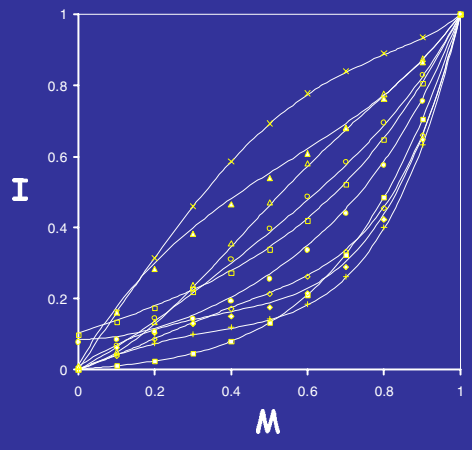
Iterative Algorithm : ratios $R_{q,q+1}$ ⇄ coeff c_n

Images: $q = 1,2,\ldots Q$
Pixels: $p = 1, 2, \ldots P$

Self Calibration Algorithm : Flow Chart



100 Tests with Noisy Synthetic Images

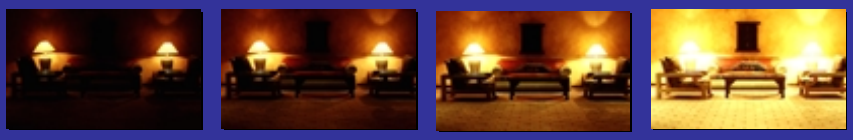


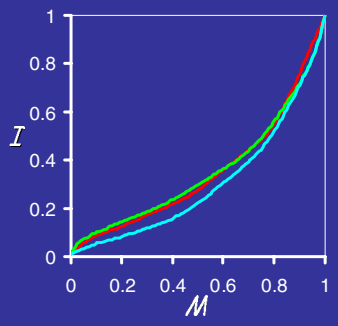
Actual Exposure Ratios : $0.45 < R < 0.55$
Initial Ratio Guess : $R = 0.5$

Solid : Computed Response Function
Dots : Actual Response Function

Results : Adobe Room

Captured images



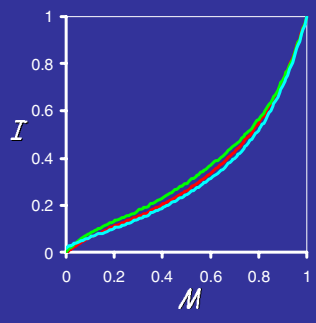


Computed response function

Computed radiance image

Results : Taos Clay Oven

Captured images



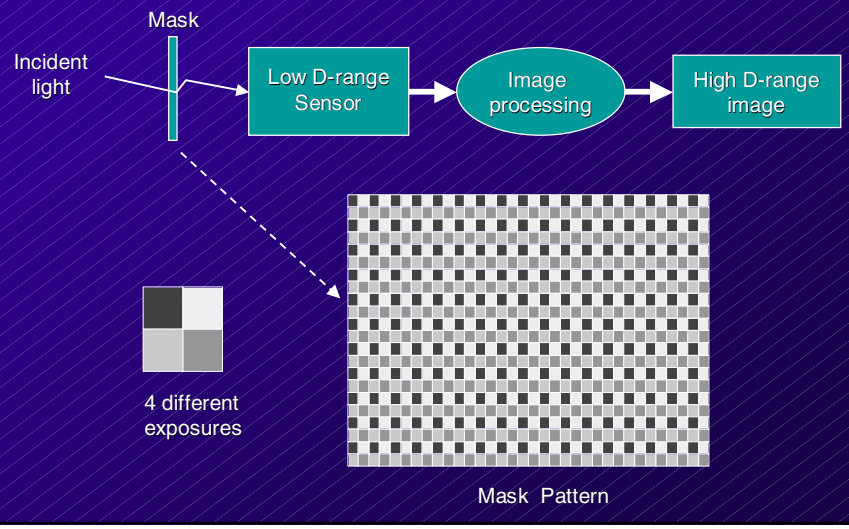
Computed response function

Computed radiance image

High Dynamic Range from Single Image

(Nayar & Mitsunaga 00)

- Spatially Varied Exposure (SVE) Camera



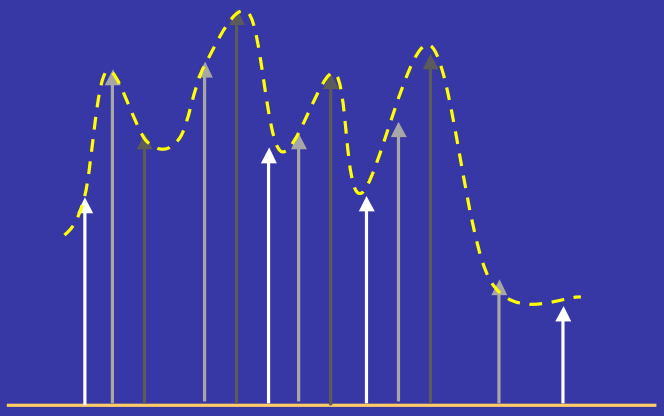
Characteristics

	Conventional Camera	SVE Camera
Dynamic Range	$DR = 20 \log \frac{I_{\max}}{I_{\min}}$ 48.13 db (8 bits)	$DR_{SVE} = 20 \log \frac{I_{\max}}{I_{\min}} \left(\frac{e_{\max}}{e_{\min}} \right)$ 64:1 84.25 db (14 bits)
Gray Levels	$Q = q$ 256	$Q_{SVE} = q + \sum_{k=0}^{K-1} Round((q-1) - (q-1)\frac{e_k}{e_{k-1}})$ 869

ACM SIGGRAPH 2000, Course Notes

High Dynamic Range Image Reconstruction

Exposure Normalized Image



ACM SIGGRAPH 2000, Course Notes

8-Bit Images with Different Exposures

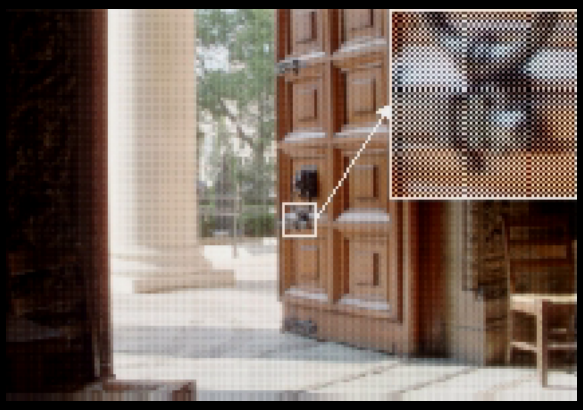
Exposure: T

Exposure: 4 T

Exposure: 16 T

Exposure: 64 T

Spatially Varying Exposure (SVE) Image



ACM SIGGRAPH 2000, Course Notes

Computed High Dynamic Range Image

ACM SIGGRAPH 2000, Course Notes

Summary

- Bigger Images and Clearer Images
- General Approach :

References

1 E. H. Adelson and J. R. Bergen.
The Plenoptic Function and the Elements of Early Vision.
MIT Press, Cambridge, MA, 1991.

2 S. Baker and S. K. Nayar.
Catadioptric Image Formation.
Proc. of International Conference on Computer Vision, January 1998.

3 S. Bogner.
Introduction to Panoramic Imaging.
Proc. of IEEE SMC Conference, pages 3100-3106, October 1995.

4 J. S. Chahl and M. V. Srinivasan.
Reflective surfaces for panoramic imaging.
36(31):8275-8285, 1997.

5 P. Debevec and J. Malik.
Recovering High Dynamic Range Radiance Maps from Photographs.
Proc. of ACM SIGGRAPH 1997, pages 369-378, 1997.

6 J. Gluckman and S. K. Nayar.
Egomotion and Omnidirectional Cameras.
Proc. of International Conference on Computer Vision, January 1998.

7 H. C. Huang and Y. P. Hung.
Panoramic stereo imaging system with automatic disparity warping and s eaming.
Graphical Models and Image Processing, 60(3):196-208, May 1998.

8 H. Ishiguro, H. Yamamoto, and S. Tsuji.
Omnidirectional Stereo.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 14:257-262, 1992.

9 B. Madden.
Extended Intensity Range Imaging.
Technical Report MS-CIS-93-96, Grasp Laboratory, University of Pennsylvania, 1993.

10 P. L. Manley.
Unusual Telescopes.

11 S. Mann and R. Picard.
Being `Undigital' with Digital Cameras: Extending Dynamic Range by Combining Differently Exposed Pictures.
Proc. of IST's 48th Annual Conference, pages 442-448, May 1995.

12 T. Mitsunaga and S. K. Nayar.
Radiometric Self Calibration.
In Proc. of Computer Vision and Pattern Recognition '99, volume 1, pages 374-380, June 1999.

13 K. Miyamoto.
Fish Eye Lens.
Journal of Optical Society of America, 54(8):1060-1061, August 1964.

14 J. R. Murphy.
Application of Panoramic Imaging to a Teleoperated Lunar Rover.
Proc. of IEEE SMC Conference, pages 3117-3121, October 1995.

15 V. Nalwa.
A True Omnidirectional Viewer.
Technical report, Bell Laboratories, Holmdel, NJ 07733, U.S.A., February 1996.

16 S. K. Nayar.
Sphereo: Recovering depth using a single camera and two specular spheres.
Proc. of SPIE: Optics, Illumination, and Image Sensing for Machine Vision II, November 1988.

17 S. K. Nayar.
Catadioptric Omnidirectional Camera.
Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, June 1997.

18 S. K. Nayar and T. Mitsunaga.
High Dynamic Range Imaging: Spatially varying pixel exposures.
In Proc. of Computer Vision and Pattern Recognition '00, June 2000.

19 S. K. Nayar and V. Peri.
Folded Catadioptric Cameras.
Proc. of IEEE Conference on Computer Vision and Pattern Recognition, June 1999.

20 Shree K. Nayar and Amruta Karmarkar.
360 x 360 mosaics.
In Proc. of Computer Vision and Pattern Recognition '00, 2000.

21 S. A. Nene and S. K. Nayar.
Stereo Using Mirrors.
Proc. of International Conference on Computer Vision, January 1998.

22 S. Peleg and M. Ben-Ezra.
Stereo Panorama with a Single Camera.
Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pages 395-401, June 1999.

23 S. Peleg and J. Herman.
Panoramic Mosaics by Manifold Projection.
Proc. of IEEE Conference on Computer Vision and Pattern Recognition, pages 338-343, June 1997.

24 V. Peri and S. K. Nayar.
Generation of Perspective and Panoramic Video from Omnidirectional Video.
Proc. of DARPA Image Understanding Workshop, May 1997.

25 D. W. Rees.
Panoramic Television Viewing System.
United States Patent, (3,505,465), April 1970.

26 H-Y. Shum and L-Wei He.
Rendering with Concentric Mosaics.
Proc. of ACM SIGGRAPH'99, August 1999.

27 H-Y. Shum, A. Kalai, and S. M. Seitz.
Omnivergent stereo.
Proc. of Seventh International Conference on Computer Vision, pages 22-29, September 1999.

28 R. Szeliski.
Image Mosaicing for Tele-reality Applications.
Proc. of IEEE Workshop on Applications of Computer Vision, pages 44-53, December 1994.

29 A. J. P. Theuwissen.
Solid State Imaging with Charge-Coupled Devices.
Kluwer Academic Press, Boston, 1995.

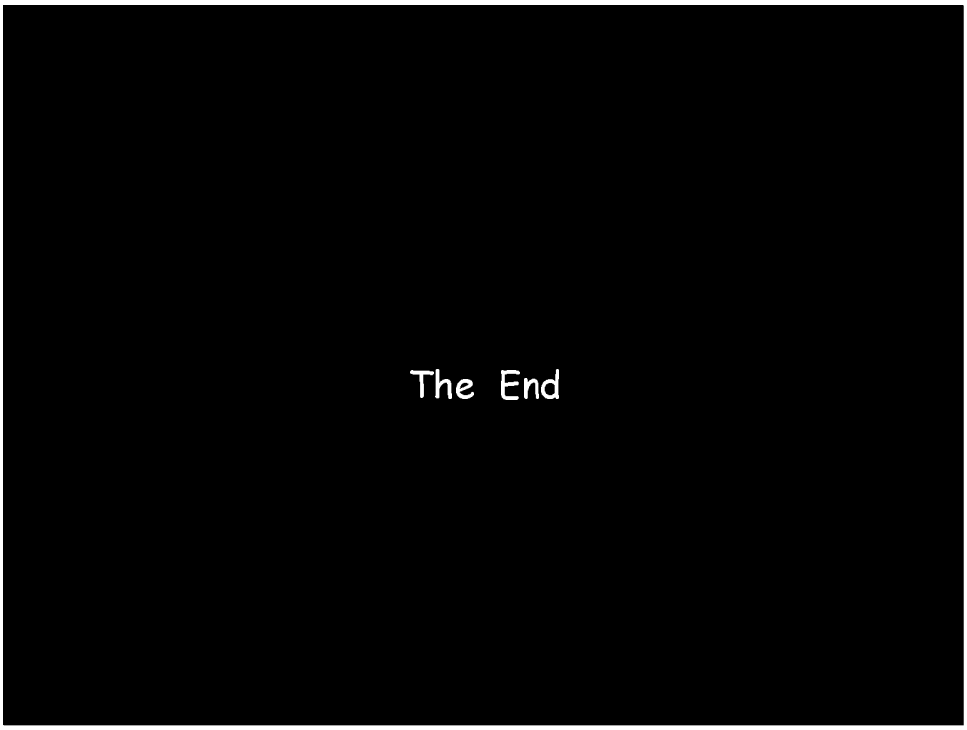
30 Y. Xiong and K. Turkowski.
Creating Image-Based VR Using a Self Calibrating Fishe ye Lens.
Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, June 1997.

31 Y. Yagi.
Omnidirectional Sensing and Its Applications.
IEICE Transactions, E82-D(3), March 1999.

32 Y. Yagi and S. Kawato.
Panoramic Scene Analysis with Conic Projection.
Proc. of International Conference on Robots and Systems (IROS), 1990.

33 K. Yamazawa, Y. Yagi, and M. Yachida.
Omnidirectional Imaging with Hyperboloidal Projection.
Proc. of International Conference on Robots and Systems (IROS), 1993.

34 J. Y. Zheng and S. Tsuji.
Panoramic Representation of Scenes for Route Understanding.
Proc. of the Tenth International Conference on Pattern Recognition, 1:161-167, June 1990.


The End

SIGGRAPH 2000 Course on
3D Photography

Passive 3D Photography

Steve Seitz
Carnegie Mellon University

http://www.cs.cmu.edu/~seitz

Talk Outline

1. *Visual Cues*
2. *Classical Vision Algorithms*
3. *State of the Art (video)*

Visual Cues

Motion

Visual Cues

Motion

Shading

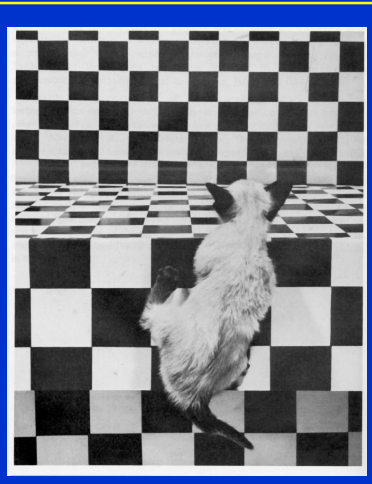
Merle Norman Cosmetics, Los Angeles

Visual Cues

Motion

Shading

Texture



The Visual Cliff, by William Vandivert, 1960

Visual Cues

Motion

Shading

Texture

Focus

From *The Art of Photography*, Canon

Visual Cues

Motion

Shading

Texture

Focus

Others:

- Highlights
- Shadows
- Silhouettes
- Inter-reflections
- Symmetry
- Light Polarization
- ...

Reconstruction Algorithms

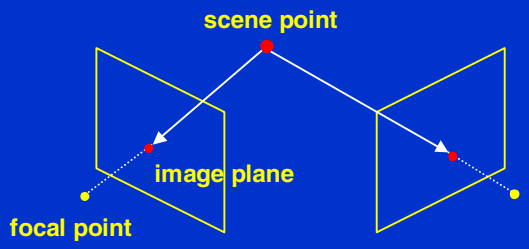
Shape From X

- ✔ Stereo (shape from parallax)
- ✔ Structure from motion
- Shape from shading
- Photometric stereo
- Shape from texture
- Shape from focus/defocus
- Shape from silhouettes, ...

Stereo

The Stereo Problem

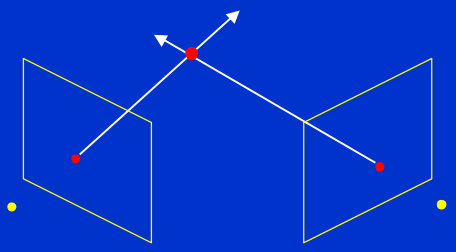
- Reconstruct scene geometry from two or more *calibrated* images



Stereo

The Stereo Problem

- Reconstruct scene geometry from two or more *calibrated* images



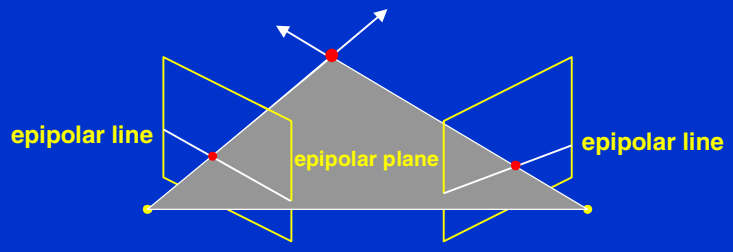
Basic Principle: Triangulation

- Gives reconstruction as intersection of two rays
- Requires ***point correspondence***

Stereo Correspondence

Determine Pixel Correspondence

- **Pairs of points that correspond to same scene point**



Epipolar Constraint

- **Reduces correspondence problem to 1D search along *conjugate epipolar lines***

Stereo Matching Algorithms

Match Pixels in Conjugate Epipolar Lines

- **Assume color of point does not change**
- **Pitfalls**
  - > specularities (non-Lambertian surfaces)
  - > ambiguity (low-contrast regions)
  - > missing data (occlusions)
  - > intensity error (quantization, sensor error)
  - > position error (camera calibration)
- **Numerous approaches**
  - > winner-take all
  - > dynamic programming [Ohta 85]
  - > smoothness functionals
  - > more images (trinocular, N-ocular) [Okutomi 93]

Structure from Motion

The SFM Problem

- Reconstruct scene geometry and camera motion from two or more images

Assume

- Pixel correspondence
  - > via tracking
- Projection model
  - > classic methods are orthographic

Orthographic Projection

$$\underset{2\times1}{\mathbf{u}} = \underset{2\times3}{\Pi}\ \underset{3\times1}{\mathbf{X}} + \underset{2\times1}{\mathbf{t}}$$

image point — projection matrix — scene point — image offset

Trick

- Choose scene origin to be centroid of 3D points
- Choose image origins to be centroid of 2D points
- Allows us to drop the camera translation:

$$\underset{2\times1}{\mathbf{u}} = \underset{2\times3}{\Pi}\ \underset{3\times1}{\mathbf{X}}$$

Shape by Factorization [Tomasi & Kanade, 92]

projection of *n* features in one image:

$$\underset{2\times n}{\begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix}} = \underset{2\times n}{\prod} \underset{3\times n}{\begin{bmatrix} X_1 & X_2 & \cdots & X_n \end{bmatrix}}$$

projection of *n* features in *f* images

$$\underset{2f\times n}{\begin{bmatrix} u_1^1 & u_2^1 & \cdots & u_n^1 \\ u_1^2 & u_2^2 & \cdots & u_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ u_1^f & u_2^f & \cdots & u_n^f \end{bmatrix}} = \underset{2f\times 3}{\begin{bmatrix} \Pi^1 \\ \Pi^2 \\ \vdots \\ \Pi^f \end{bmatrix}} \underset{3\times n}{\begin{bmatrix} X_1 & X_2 & \cdots & X_n \end{bmatrix}}$$

W measurement **M** motion **S** shape

Shape by Factorization [Tomasi & Kanade, 92]

known — $\underset{2f\times n}{W} = \underset{2f\times 3}{M}\ \underset{3\times n}{S}$ — solve for

Factorization Technique

- *W* is at most rank 3 (assuming no noise)
- We can use *singular value decomposition* to factor W:

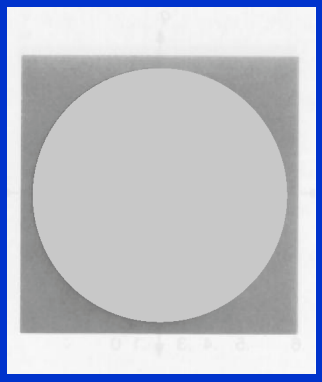
$$\underset{2f\times n}{W} = \underset{2f\times 3}{M'}\ \underset{3\times n}{S'}$$

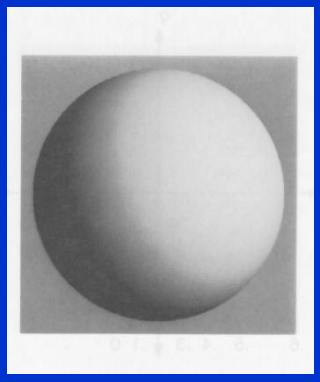
- *S'* differs from *S* by a linear transformation *A*:

$$W = M'S' = (MA^{-1})(AS)$$

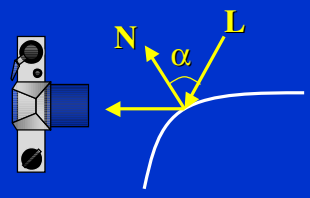
- Solve for *A* by enforcing constraints on *M*

Shape from Shading





Shape from Shading [Horn, 1970]

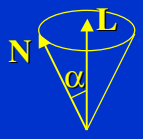


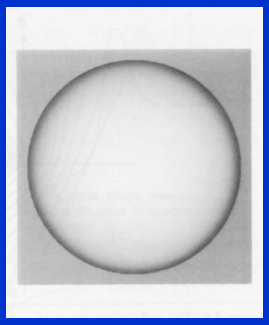
Classical Approach

- **Suppose reflected light depends only on α**

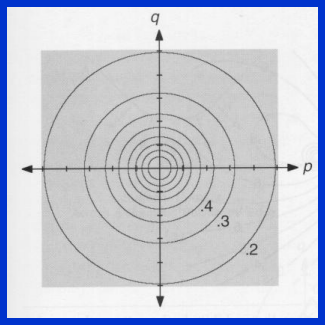
$$radiance = k \cos\alpha$$

The Reflectance Map





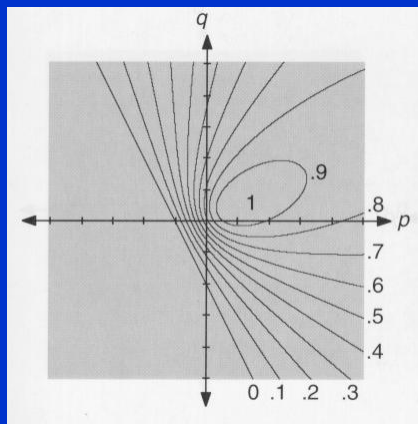
Image



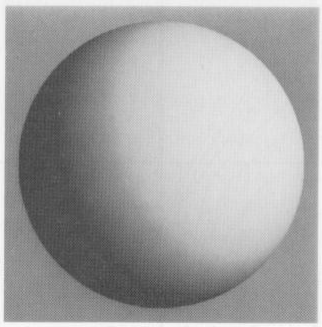
Reflectance Map: R

$$\mathbf{N} = [p \quad q \quad -1]$$

The Reflectance Map



Reflectance Map



Image

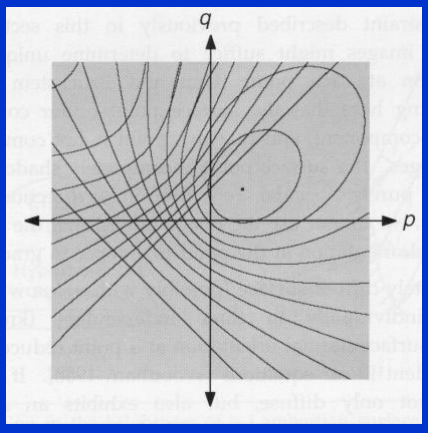
Finding a Unique Solution

Three Approaches

- **Characteristic Strip Method [Horn, 77]**
  - > select a few points where normal is known
  - > grow solution by moving direction of $\nabla \mathbf{R}$
- **Variational Method [Ikeuchi & Horn, 81]**
  - > start with an initial guess of surface shape
  - > define energy function
  - > refine to minimize energy function
- **Photometric Stereo [Woodham 80]**
  - > use more images

Photometric Stereo

Two Images Under Different Lighting



Need Three Images for Unique Solution

Photometric Stereo: Matrix Formulation

Write Equations in Matrix Form

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{L}}_1^T \\ \hat{\mathbf{L}}_2^T \\ \hat{\mathbf{L}}_3^T \end{bmatrix} \bullet k\hat{\mathbf{N}}$$

$$\mathbf{I}_{3\times 1} \quad \mathbf{L}_{3\times 3} \quad \tilde{\mathbf{N}}_{3\times 1}$$

$$\tilde{\mathbf{N}} = \mathbf{L}_1^{-1}\mathbf{I}$$

$$k = \left\| \tilde{\mathbf{N}} \right\|$$

Advantage:

- Can solve for variable reflectance *k*

Resources

Computer Vision Home Page

- http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/vision.html

Computer Vision Textbooks

- D. H. Ballard and C. M. Brown, *Computer Vision*, Prentice-Hall, 1982.
- O. Faugeras, *Three-Dimensional Computer Vision*, MIT Press, 1993.
- B. K. P. Horn, *Robot Vision*, McGraw-Hill, 1986.
- R. Jain, R. Kasturi and B. G. Schunck, *Machine Vision*, McGraw-Hill, 1995.
- R. Klette, K. Schluns and A. Koschan, *Computer Vision: Three-Dimensional Data from Images*, Springer-Verlag, 1998.
- V. S. Nalwa, *A Guided Tour of Computer Vision*, Addison-Wesley, 1993.
- M. Sonka, V. Hlavac and R. Boyle, *Image Processing, Analysis, and Machine Vision*, Brooks/Cole Publishing, 1999.
- E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice-Hall, 1998.
- D. Marr, *Vision*, Freeman, 1982.
- J. Koenderink, *Solid Shape*, MIT Press, 1990.

Bibliography

Stereo

- Yuichi Ohta & Takeo Kanade, "Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming", IEEE Trans. on Pattern Analysis and Machine Intelligence, 7(2), 1985, pp. 129-154.
- Masatoshi Okutomi & Takeo Kanade, "A Multiple-Baseline Stereo", IEEE Trans. on Pattern Analysis and Machine Intelligence", 15(4), 353-363, 1985.

Structure-from-Motion

- Carlo Tomasi & Takeo Kanade, "Shape and Motion from Image Streams Under Orthography: A Factorization Method", Int. Journal of Computer Vision, 9(2), 1992, pp. 137-154.

Shape from Shading

- B. Horn and M. Brooks, "Shape from Shading", 1989, MIT Press.
- L. Wolff, S. Shafer, and G. E. Healey, "Physics-Based Vision: Shape Recovery", 1992, Jones and Bartlett.
- R. J. Woodham, "Photometric Method for Determining Surface Orientation from Multiple Images", Optical Engineering, 1980, pp. 139-144.

Video

Shape and Motion from Image Streams: a Factorization Method

Full Report on the Orthographic Case

Carlo Tomasi Takeo Kanade

March 1992

Cornell TR 92-1270 and Carnegie Mellon CMU-CS-92-104

This research was sponsored by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-90-C-1465, ARPA Order No. 7597.

Keywords: computer vision, motion, shape, time-varying imagery

Abstract

Inferring scene geometry and camera motion from a stream of images is possible in principle, but is an ill-conditioned problem when the objects are distant with respect to their size. We have developed a *factorization method* that can overcome this difficulty by recovering shape and motion without computing depth as an intermediate step.

An image stream can be represented by the $2F \times P$ measurement matrix of the image coordinates of P points tracked through F frames. We show that under orthographic projection this matrix is of rank 3.

Using this observation, the factorization method uses the singular value decomposition technique to factor the measurement matrix into two matrices which represent object shape and camera motion respectively. The method can also handle and obtain a full solution from a partially filled-in measurement matrix, which occurs when features appear and disappear in the image sequence due to occlusions or tracking failures.

The method gives accurate results, and does not introduce smoothing in either shape or motion. We demonstrate this with a series of experiments on laboratory and outdoor image streams, with and without occlusions.

Chapter 1

Introduction

The structure from motion problem – recovering scene geometry and camera motion from a sequence of images – has attracted much of the attention of the vision community over the last decade. Yet it is common knowledge that existing solutions work well for perfect images, but are very sensitive to noise. We present a new method called the factorization method which can robustly recover shape and motion from a sequence of images without assuming a model of motion, such as constant translation or rotation.

More specifically, an image sequence can be represented as a $2F \times P$ measurement matrix W, which is made up of the horizontal and vertical coordinates of P points tracked through F frames. If image coordinates are measured with respect to their centroid, we prove the *rank theorem*: under orthography, the measurement matrix is of rank 3. As a consequence of this theorem, we show that the measurement matrix can be factored into the product of two matrices R and S. Here, R is a $2F \times 3$ matrix that represents camera rotation, and S is a $3 \times P$ matrix which represents shape in a coordinate system attached to the object centroid. The two components of the camera translation along the image plane are computed as averages of the rows of W. When features appear and disappear in the image sequence due to occlusions or tracking failures, the resultant measurement matrix W is only partially filled-in. The factorization method can handle this situation by growing a partial solution obtained from an initial full submatrix into a full solution with an iterative procedure.

The rank theorem precisely captures the nature of the redundancy that exists in an image sequence, and permits a large number of points and frames

to be processed in a conceptually simple and computationally efficient way to reduce the effects of noise. The resulting algorithm is based on the singular value decomposition, which is numerically well-behaved and stable. The robustness of the recovery algorithm in turn enables us to use an image sequence with a very short interval between frames (an *image stream*), which makes feature tracking relatively easy.

We have demonstrated the accuracy and robustness of the factorization method in a series of experiments on laboratory and outdoor sequences, with and without occlusions.

Chapter 2

Relation to Previous Work

In Ullman's original proof of existence of a solution [Ullman, 1979] for the structure from motion problem under orthography, as well as in the perspective formulation in [Roach and Aggarwal, 1979], the coordinates of feature points in the world are expressed in a world-centered system of reference. Since then, however, this choice has been replaced by most computer vision researchers with that of a camera-centered representation of shape [Prazdny, 1980], [Bruss and Horn, 1983], [Tsai and Huang, 1984], [Adiv, 1985], [Waxman and Wohn, 1985], [Bolles *et al.*, 1987], [Horn *et al.*, 1988], [Heeger and Jepson, 1989], [Heel, 1989], [Matthies *et al.*, 1989], [Spetsakis and Aloimonos, 1989], [Broida *et al.*, 1990]. With this representation, the position of feature points is specified by their image coordinates and by their depths, defined as the distances between the camera center and the feature points, measured along the optical axis. Unfortunately, although a camera-centered representation simplifies the equations for perspective projection, it makes shape estimation difficult, unstable, and noise sensitive.

There are two fundamental reasons for this. First, when camera motion is small, effects of camera rotation and translation can be confused with each other: for example, small rotation about the vertical axis and small translation along the horizontal axis both generate a very similar change in an image. Any attempt to recover or differentiate between these two motions, though doable mathematically, is naturally noise sensitive. Second, the computation of shape as relative depth, for example, the height of a building as the difference of depths between the top and the bottom, is very sensitive to noise, since it is a small difference between large values.

These difficulties are especially magnified when the objects are distant from the camera relative to their sizes, which is usually the case for interesting applications such as site modeling.

The factorization method we present in this paper takes advantage of the fact that both difficulties disappear when the problem is reformulated in world-centered coordinates, unlike the conventional camera-centered formulation. This new (old – in a sense) formulation links object-centered shape to image motion directly, without using retinotopic depth as an intermediate quantity, and leads to a simple and well-behaved solution. Furthermore, the mutual independence of shape and motion in world-centered coordinates makes it possible to cast the structure-from-motion problem as a factorization problem, in which a matrix representing image measurements is decomposed directly into camera motion and object shape.

We first introduced this factorization method in [Tomasi and Kanade, 1990a, Tomasi and Kanade, 1990b], where we treated the case of single-scanline images in a flat, two-dimensional world. In [Tomasi and Kanade, 1991] we presented the theory for the case of arbitrary camera motion in three dimensions and full two-dimensional images. This paper extends the factorization method for dealing with feature occlusions as well as presenting more experimental results with real-world images. Debrunner and Ahuja have pursued an approach related to ours, but using a different formalism [Debrunner and Ahuja, 1990, Debrunner and Ahuja, 1991]. Assuming that motion is constant over a period, they provide both closed-form expressions for shape and motion and an incremental solution (one image at a time) for multiple motions by taking advantage of the redundancy of measurements. Boult and Brown have investigated the factorization method for multiple motions [Boult and Brown, 1991], in which they count and segment separate motions in the field of view of the camera.

Chapter 3

The Factorization Method

Given an image stream, suppose that we have tracked P feature points over F frames. We then obtain trajectories of image coordinates $\{(u_{fp}, v_{fp}) \,|\, f = 1, \ldots, F, p = 1, \ldots, P\}$. We write the horizontal feature coordinates u_{fp} into an $F \times P$ matrix U: we use one row per frame, and one column per feature point. Similarly, an $F \times P$ matrix V is built from the vertical coordinates v_{fp}. The combined matrix of size $2F \times P$

$$W = \left[\frac{U}{\mathrm{V}}\right]$$

is called the *measurement matrix*. The rows of the matrices U and V are then registered by subtracting from each entry the mean of the entries in the same row:

$$\begin{aligned} \tilde{u}_{fp} &= u_{fp} - a_f \\ \tilde{v}_{fp} &= v_{fp} - b_f \ , \end{aligned} \tag{3.1}$$

where

$$\begin{aligned} a_f &= \frac{1}{P}\sum_{p=1}^{P} u_{fp} \\ b_f &= \frac{1}{P}\sum_{p=1}^{P} v_{fp} \ . \end{aligned}$$

This produces two new $F \times P$ matrices $\tilde{U} = [\tilde{u}_{fp}]$ and $\tilde{V} = [\tilde{v}_{fp}]$. The matrix

$$\widetilde{W} = \left[\frac{\tilde{U}}{\tilde{V}}\right]$$

is called the *registered measurement matrix*. This is the input to our factorization method.

3.1 The Rank Theorem

We now analyze the relation between camera motion, shape, and the entries of the registered measurement matrix $\widetilde{W}$. This analysis leads to the key result that $\widetilde{W}$ is highly rank-deficient.

Referring to Figure 3.1, suppose we place the origin of the world reference system $x - y - z$ at the centroid of the P points $\mathbf{s}_p = (x_p, y_p, z_p)^T, p = 1, \ldots, P\}$, in space which correspond to the P feature points tracked in the image stream. The orientation of the camera reference system corresponding to frame number f is determined by a pair of unit vectors, $\mathbf{i}_f$and $\mathbf{j}_f$, pointing along the scanlines and the columns of the image respectively, and defined with respect to the world reference system. Under orthography, all projection rays are then parallel to the cross product of $\mathbf{i}_f$and $\mathbf{j}_f$:

$$\mathbf{k}_f = \mathbf{i}_f \times \mathbf{j}_f \ .$$

From Figure 3.1 we see that the projection (u_{fp}, v_{fp}), *i.e.*, the image feature position, of point $\mathbf{s}_p = (x_p, y_p, z_p)^T$ onto frame f is given by the equations

$$\begin{aligned} u_{fp} &= \mathbf{i}_f{}^T(\mathbf{s}_p - \mathbf{t}_f) \\ v_{fp} &= \mathbf{j}_f{}^T(\mathbf{s}_p - \mathbf{t}_f) \ , \end{aligned}$$

where $\mathbf{t}_f = (a_f, b_f, c_f)^T$ is the vector from the world origin to the origin of image frame f. Here note that since the origin of the world coordinates is placed at the centroid of object points,

$$\frac{1}{P}\sum_{p=1}^{P} \mathbf{s}_p = 0 \ .$$

We can now write expressions for the entries $\tilde{u}_{fp}$ and $\tilde{v}_{fp}$ defined in (3.1) of the registered measurement matrix. For the the registered horizontal image projection we have

$$\tilde{u}_{fp} = u_{fp} - a_f$$

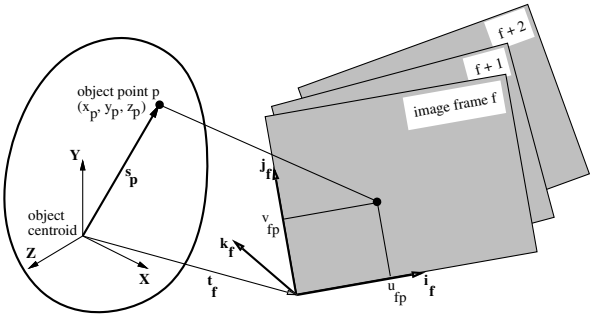


Figure 3.1: The systems of reference used in our problem formulation.

$$\begin{aligned} &= \mathbf{i}_f{}^T(\mathbf{s}_p - \mathbf{t}_f) - \frac{1}{P}\sum_{q=1}^{P}\mathbf{i}_f{}^T(\mathbf{s}_q - \mathbf{t}_f) \\ &= \mathbf{i}_f{}^T\left(\mathbf{s}_p - \frac{1}{P}\sum_{q=1}^{P}\mathbf{s}_q\right) \\ &= \mathbf{i}_f{}^T\mathbf{s}_p \ . \end{aligned} \tag{3.2}$$

We can write a similar equation for $\widetilde{v}_{fp}$. To summarize,

$$\begin{aligned} \widetilde{u}_{fp} &= \mathbf{i}_f{}^T\mathbf{s}_p \\ \widetilde{v}_{fp} &= \mathbf{j}_f{}^T\mathbf{s}_p \ . \end{aligned} \tag{3.3}$$

Because of the two sets of $F \times P$ equations (3.3), the registered measurement matrix $\widetilde{W}$ can be expressed in a matrix form:

$$\widetilde{W} = RS \tag{3.4}$$

where

$$R = \begin{bmatrix} \mathbf{i}_1^T \\ \vdots \\ \mathbf{i}_F^T \\ \mathbf{j}_1^T \\ \vdots \\ \mathbf{j}_F^T \end{bmatrix} \tag{3.5}$$

represents the camera rotation, and

$$S = \begin{bmatrix} \mathbf{s}_1 & \cdots & \mathbf{s}_P \end{bmatrix} \tag{3.6}$$

is the shape matrix. In fact, the rows of R represent the orientations of the horizontal and vertical camera reference axes throughout the stream, while the columns of S are the coordinates of the P feature points with respect to their centroid.

Since R is $2F \times 3$ and S is $3 \times P$, the equation (3.4) implies the following.

> **Rank Theorem**: *Without noise, the registered measurement matrix $\widetilde{W}$ is at most of rank three.*

The rank theorem expresses the fact that the $2F \times P$ image measurements are highly redundant. Indeed, they could all be described concisely by giving F frame reference systems and P point coordinate vectors, if only these were known.

From the first and the last line of equation (3.2), the original unregistered matrix W can be written as

$$W = RS + \mathbf{t}\mathbf{e}_P^T \,, \tag{3.7}$$

where $\mathbf{t} = (a_1, \ldots, a_F, b_1, \ldots, b_F)^T$ is a $2F$-dimensional vector that collects the projections of camera translation along the image plane (see equation (3.2)), and $\mathbf{e}_P^T = (1, \ldots, 1)$ is a vector of P ones. In scalar form,

$$\begin{aligned} u_{fp} &= \mathbf{i}_f^T \mathbf{s}_p + a_f \\ v_{fp} &= \mathbf{j}_f^T \mathbf{s}_p + b_f \,. \end{aligned} \tag{3.8}$$

Comparing with equations (3.1), we see that the two components of camera translation along the image plane are simply the averages of the rows of W.

In the equations above, $\mathbf{i}_f$ and $\mathbf{j}_f$ are mutually orthogonal unit vectors, so they must satisfy the constraints

$$|\mathbf{i}_f| = |\mathbf{j}_f| = 1 \quad \text{and} \quad \mathbf{i}_f^T \mathbf{j}_f = 0 \,. \tag{3.9}$$

Also, the rotation matrix R is unique if the system of reference for the solution is aligned, say, with that of the first camera position, so that:

$$\mathbf{i}_1 = (1,0,0)^T \quad \text{and} \quad \mathbf{j}_1 = (0,1,0)^T \,. \tag{3.10}$$

The registered measurement matrix $\widetilde{W}$ must be at most of rank three without noise. When noise corrupts the images, however, $\widetilde{W}$ will not be exactly of rank 3. However, the rank theorem can be extended to the case of noisy measurements in a well-defined manner. The next sectionintroduces the notion of approximate rank, using the concept of singular value decomposition [Golub and Reinsch, 1971].

3.2 Approximate Rank

Assuming [1] that $2F \geq P$, the matrix $\widetilde{W}$ can be decomposed [Golub and Reinsch, 1971] into a $2F \times P$ matrix O_1, a diagonal $P \times P$ matrix Σ, and a $P \times P$ matrix O_2,

$$\widetilde{W} = O_1 \Sigma O_2 \ , \tag{3.11}$$

such that $O_1^T O_1 = O_2^T O_2 = O_2 O_2^T = \mathcal{I}$, where $\mathcal{I}$ is the $P \times P$ identity matrix. Σ is a diagonal matrix whose diagonal entries are the *singular values* $\sigma_1 \geq \ldots \geq \sigma_P$ sorted in non-decreasing order. This is the *Singular Value Decomposition* (SVD) of the matrix $\widetilde{W}$.

Suppose that we pay attention only to the first three columns of O_1, the first 3×3 submatrix of Σ and the first three rows of O_2. If we partition the matrices O_1, Σ, and O_2 as follows:

$$\begin{aligned} O_1 &= \underbrace{\left[\ O_1' \ \right.}_{3} \underbrace{\left| \ O_1'' \ \right]}_{P-3} \}_{2F} \\ \Sigma &= \left[\begin{array}{c|c} \Sigma' & 0 \\ \hline 0 & \Sigma'' \end{array}\right] \begin{array}{l} \}_3 \\ \}_{P-3} \end{array} \\ & \quad \underbrace{}_{3} \ \underbrace{}_{P-3} \\ O_2 &= \left[\begin{array}{c} O_2' \\ \hline O_2'' \end{array}\right] \begin{array}{l} \}_3 \\ \}_{P-3} \end{array} \ , \\ & \quad \underbrace{}_{P} \end{aligned} \tag{3.12}$$

we have

$$O_1 \Sigma O_2 = O_1' \Sigma' O_2' + O_1'' \Sigma'' O_2'' \ .$$

Let $\widetilde{W}^*$ be the ideal registered measurement matrix, that is, the matrix we would obtain in the absence of noise. Because of the rank theorem, $\widetilde{W}^*$ has at most three non-zero singular values. Since the singular values in Σ are sorted in non-increasing order, Σ' must contain all the singular values of

[1] This assumption is not crucial: if $2F < P$, everything can be repeated for the transpose of $\widetilde{W}$.

$\widetilde{W}^*$ that exceed the noise level. As a consequence, the term $O_1''\Sigma''O_2''$ must be due entirely to noise, and the best possible rank-3 approximation to the ideal registered measurement matrix $\widetilde{W}^*$ is the product:

$$\hat{W} \;=\; O_1'\Sigma'O_2'$$

We can now restate our rank theorem for the case of noisy measurements.

> **Rank Theorem for Noisy Measurements**: *All the shape and rotation information in $\widetilde{W}$ is contained in its three greatest singular values, together with the corresponding left and right eigenvectors.*

Now if we define

$$\begin{aligned} \hat{R} &= O_1'[\Sigma']^{1/2} \\ \hat{S} &= [\Sigma']^{1/2}O_2' \;, \end{aligned}$$

we can write

$$\hat{W} = \hat{R}\hat{S} \;. \tag{3.13}$$

The two matrices $\hat{R}$ and $\hat{S}$ are of the same size as the desired rotation and shape matrices R and S: $\hat{R}$ is $2F \times 3$, and $\hat{S}$ is $3 \times P$. However, the decomposition (3.13) is not unique. In fact, if Q is *any* invertible 3×3 matrix, the matrices $\hat{R}Q$ and $Q^{-1}\hat{S}$ are also a valid decomposition of $\hat{W}$, since

$$(\hat{R}Q)(Q^{-1}\hat{S}) = \hat{R}(QQ^{-1})\hat{S} = \hat{R}\hat{S} = \hat{W} \;.$$

Thus, $\hat{R}$ and $\hat{S}$ are in general different from R and S. A striking fact, however, is that except for noise the matrix $\hat{R}$ is a linear transformation of the true rotation matrix R, and the matrix $\hat{S}$ is a linear transformation of the true shape matrix S. Indeed, in the absence of noise, R and $\hat{R}$ both span the column space of the registered measurement matrix $\widetilde{W} = \widetilde{W}^* = \hat{W}$. Since that column space is three-dimensional because of the rank theorem, R and $\hat{R}$ are different bases for the same space, and there must be a linear transformation between them.

Whether the noise level is low enough that it can be ignored at this juncture depends also on the camera motion and on shape. Notice, however, that the singular value decomposition yields sufficient information to make this decision: the requirement is that the ratio between the third and the fourth largest singular values of $\widetilde{W}$ be sufficiently large.

3.3 The Metric Constraints

We have found that the matrix $\hat{R}$ is a linear transformation of the true rotation matrix R. Likewise, $\hat{S}$ is a linear transformation of the true shape matrix S. More specifically, there exists a 3×3 matrix Q such that

$$\begin{aligned} R &= \hat{R}Q \\ S &= Q^{-1}\hat{S} \, . \end{aligned} \tag{3.14}$$

In order to find Q we observe that the rows of the true rotation matrix R are unit vectors and the first F are orthogonal to corresponding F in the second half of R. These *metric constraints* yield the over-constrained, quadratic system

$$\begin{aligned} \hat{\mathbf{i}_f}^T Q Q^T \hat{\mathbf{i}_f} &= 1 \\ \hat{\mathbf{j}_f}^T Q Q^T \hat{\mathbf{j}_f} &= 1 \\ \hat{\mathbf{i}_f}^T Q Q^T \hat{\mathbf{j}_f} &= 0 \end{aligned} \tag{3.15}$$

in the entries of Q. This is a simple data fitting problem which, though nonlinear, can be solved efficiently and reliably. Its solution is determined up to a rotation of the whole reference system, since the orientation of the world reference system was arbitrary. This arbitrariness can be removed by enforcing the constraints (3.10), that is, selecting the $x-y$ axes of the world reference system to be parallel with those of the first frame.

3.4 Outline of the Complete Algorithm

Based on the development in the previous chapters, we now have a complete algorithm for the factorization of the registered measurement matrix $\widetilde{W}$ derived from a stream of images into shape S and rotation R as defined in equations (3.4) - (3.6).

1. Compute the singular-value decomposition $\widetilde{W} = O_1 \Sigma O_2$.
2. Define $\hat{R} = O_1'(\Sigma')^{1/2}$ and $\hat{S} = (\Sigma')^{1/2} O_2'$, where the primes refer to the block partitioning defined in (3.12).
3. Compute the matrix Q in equations (3.14) by imposing the metric constraints (equations (3.15)).

4. Compute the rotation matrix R and the shape matrix S as $R = \hat{R}Q$ and $S = Q^{-1}\hat{S}$.

5. If desired, align the first camera reference system with the world reference system by forming the products RR_0 and $R_0^T S$, where the orthonormal matrix $R_0 = [\mathbf{i}_1 \ \mathbf{j}_1 \ \mathbf{k}_1]$ rotates the first camera reference system into the identity matrix.

Chapter 4

Experiment

We test the factorization method with two real streams of images: one taken in a controlled laboratory environment with ground-truth motion data, and the other in an outdoor environment with a hand-held camcorder.

4.1 "Hotel" Image Stream in a Laboratory

Some frames in this stream are shown in figure 4.1. The images depict a small plastic model of a building. The camera is a Sony CCD camera with a 200 mm lens, and is moved by means of a high-precision positioning platform. Camera pitch, yaw, and roll around the model are all varied as shown by the dashed curves in figure 4.2. The translation of the camera is such as to keep the building within the field of view of the camera.

For feature tracking, we extended the Lucas-Kanade method described in [Lucas and Kanade, 1981] to allow also for the automatic selection of image features. The Lucas-Kanade method of tracking obtains the displacement vector of the window around a feature as the solution of a linear 2×2 equation system. As good image features we select those points for which the above equation systems are stable. The details are presented in [Tomasi, 1991, Tomasi and Kanade, 1992].

The entire set of 430 features thus selected is displayed in figure 4.3, overlaid on the first frame of the stream. Of these features, 42 were abandoned during tracking because their appearance changed too much. The trajectories of the remaining 388 features are used as the measurement matrix for

the computation of shape and motion.

The motion recovery is precise. The plots in figure 4.2 compare the rotation components computed by the factorization method (solid curves) with the values measured mechanically from the mobile platform (dashed curves). The differences are magnified in figure 4.4. The errors are everywhere less than 0.4 degrees and on average 0.2 degrees. The computed motion follows closely also rotations with curved profiles, such as the roll profile between frames 1 and 20 (second plot in figure 4.2), and faithfully preserves all discontinuities in the rotational velocities: the factorization method does not smooth the results.

Between frames 60 and 80, yaw and pitch are nearly constant, and the camera merely rotates about its optical axis. That is, the motion is actually degenerate during this period, but still it has been correctly recovered. This demonstrates that the factorization method can deal without difficulty with streams that contain degenerate substreams, because the information in the stream is used *as a whole* in the method.

The shape results are evaluated qualitatively in figure 4.5, which shows the computed shape viewed from above. The view in figure 4.5 is similar to that in figure 4.6, included for visual comparison. Notice that the walls, the windows on the roof, and the chimneys are recovered in their correct positions.

To evaluate the shape performance quantitatively, we measured some distances on the actual house model with a ruler and compared them with the distances computed from the point coordinates in the shape results. Figure 4.7 shows the selected features. The diagram in figure 4.8 shows the distances between pairs of features measured on the actual model and those computed by the factorization method. The measured distances between the steps along the right side of the roof (7.2 mm) were obtained by measuring five steps and dividing the total distance (36 mm) by five. The differences between computed and measured results are of the order of the resolution of our ruler measurements (one millimeter).

Part of the errors in the results is due to the use of orthography as the projection model. However, it tends to be fairly small for many realistic situations. In fact, it has been shown that errors due to the orthogrphic distortion are approximately about the same percentage as the ratio of the object size in depth to the distance of the object from the camera [Tomasi, 1991].

4.2 Outdoor "House" Image Stream

The factorization method has been tested with an image stream of a real building, taken with a hand-held camera. Figure 4.9 shows some of the 180 frames of the building stream. The overall motion covers a relatively small rotation angle, approximately 15 degrees. Outdoor images are harder to process than those produced in a controlled environment of the laboratory, because lighting changes less predictably and the motion of the camera is more difficult to control. As a consequence, features are harder to track: the images are unpredictably blurred by motion, and corrupted by vibrations of the video recorder's head, both during recording and digitization. Furthermore, the camera's jumps and jerks produce a wide range of image disparities.

The features found by the selection algorithm in the first frame are shown in figure 4.10. There are many false features. The reflections in the window partially visible in the top left of the image move non-rigidly. More false features can be found in the lower left corner of the picture, where the vertical bars of the handrail intersect the horizontal edges of the bricks of the wall behind. We masked away these two parts of the image from the analysis.

In total, 376 features were found by the selection algorithm and tracked. Figure 4.11 plots the tracks of some (60) of the features for illustration. Notice the very jagged trajectories due to the vibrating motion of the hand-held camera.

Figures 4.12 and 4.13 show a front and a top view of the building as reconstructed by the factorization method. To render these figures for display, we triangulated the computed 3D points into a set of small surface patches and mapped the pixel values in the first frame onto the resulting surface. The structure of the visible part of the building's three walls has clearly been reconstructed. In these figures, the left wall appears to bend somewhat on the right where it intersects the middle wall. This occurred because the feature selector found features along the shadow of the roof just on the right of the intersection of the two walls, rather than at the intersection itself. Thus, the appearance of a bending wall is an artifact of the triangulation done for rendering.

This experiment with an image stream taken outdoors with the jerky motion produced by a hand-held camera demonstrates that the factorization method does not require a smooth motion assumption. The identification of

false features, that is, of features that do not move rigidly with respect of the environment, remains an open problem that must be solved for a fully autonomous system. An initial effort has been seen in [Boult and Brown, 1991].

Chapter 5

Occlusions

In reality, as the camera moves, features can appear and disappear from the image, because of occlusions. Also, a feature tracking method will not always succeed in tracking features throughout the image stream. These phenomena are frequent enough to make a shape and motion computation method unrealistic if it cannot deal with them.

Sequences with appearing and disappearing features result in a measurement matrix W which is only partially filled in. The factorization method introduced in chapter3 cannot be applied directly. However, there is usually sufficient information in the stream to determine all the camera positions and all the three-dimensional feature point coordinates. If that is the case, we can not only solve the shape and motion recovery problem from the incomplete measurement matrix W, but we can even hallucinate the unknown entries of W by projecting the computed three-dimensional feature coordinates onto the computed camera positions.

5.1 Solution for Noise-Free Images

Suppose that a feature point is not visible in a certain frame. If the same feature is seen often enough in other frames, its position in space should be recoverable. Moreover, if the frame in question includes enough other features, the corresponding camera position be recoverable as well. Then from point and camera positions thus recovered, we should also be able to reconstruct the missing image measurement. Formally, we have the following

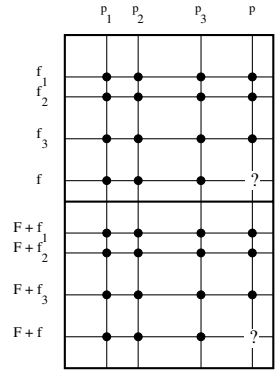


Figure 5.1: The Reconstruction Condition. If the dotted entries of the measurement matrix are known, the two unknown ones (question marks) can be reconstructed.

sufficient condition.

> **Condition for Reconstruction**: In the absence of noise, an unknown image measurement pair (u_{fp}, v_{fp}) in frame f can be reconstructed if point p is visible in at least three more frames f_1, f_2, f_3, and if there are at least three more points p_1, p_2, p_3 that are visible in all the four frames: the original f and the additional f_1, f_2, f_3.

Referring to Figure 5.1, this means that the dotted entries must be known to reconstruct the question marks. This is equivalent to Ullman's result [Ullman, 1979] that three views of four points determine structure and motion. In this section, we prove the reconstruction condition in our formalism and develop the reconstruction procedure. To this end, we notice that the rows and columns of the noise-free measurement matrix W can always be permuted so that $f_1 = p_1 = 1$, $f_2 = p_2 = 2$, $f_3 = p_3 = 3$, $f = p = 4$. We can therefore suppose that u_{44} and v_{44} are the only two unknown entries in the

8×4 matrix

$$W = \left[\frac{U}{V}\right] = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ u_{41} & u_{42} & u_{43} & ? \\ v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \\ v_{41} & v_{42} & v_{43} & ? \end{bmatrix} .$$

Then, the factorization method can be applied to the first three rows of U and V, that is, to the 6×4 submatrix

$$W_{6\times 4} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \end{bmatrix} \tag{5.1}$$

to produce the partial translation and rotation submatrices

$$\mathbf{t}_{6\times 1} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \text{and} \quad R_{6\times 3} = \begin{bmatrix} \mathbf{i}_1^T \\ \mathbf{i}_2^T \\ \mathbf{i}_3^T \\ \mathbf{j}_1^T \\ \mathbf{j}_2^T \\ \mathbf{j}_3^T \end{bmatrix} \tag{5.2}$$

and the full shape matrix

$$S = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \mathbf{s}_3 & \mathbf{s}_4 \end{bmatrix} \tag{5.3}$$

such that

$$W_{6\times 4} = R_{6\times 3} S + \mathbf{t}_{6\times 1} \mathbf{e}_4^T$$

where $\mathbf{e}_4^T = (1, 1, 1, 1)$.

To complete the rotation solution, we need to compute the vectors $\mathbf{i}_4$ and $\mathbf{j}_4$. However, a registration problem must be solved first. In fact, only three points are visible in the fourth frame, while equation (5.3) yields all

four points in space. Since the factorization method computes the space coordinates with respect to the centroid of the points, we have $\mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4 = 0$, while the image coordinates in the fourth frame are measured with respect to the centroid of just three observed points (1, 2, 3). Thus, before we can compute $\mathbf{i}_4$ and $\mathbf{j}_4$ we must make the two origins coincide by referring all coordinates to the centroid

$$\mathbf{c} = \frac{1}{3}(\mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3)$$

of the three points that are visible in all four frames. In the fourth frame, the projection of $\mathbf{c}$ has coordinates

$$\begin{aligned} a'_4 &= \frac{1}{3}(u_{41} + u_{42} + u_{43}) \\ b'_4 &= \frac{1}{3}(v_{41} + v_{42} + v_{43}) \ , \end{aligned}$$

so we can define the new coordinates

$$\mathbf{s}'_p = \mathbf{s}_p - \mathbf{c} \quad \text{for} \quad p = 1,2,3$$

in space and

$$\begin{aligned} u'_{4p} &= u_{4p} - a'_4 \\ v'_{4p} &= v_{4p} - b'_4 \end{aligned} \quad \text{for} \quad p = 1,2,3$$

in the fourth frame. Then, $\mathbf{i}_4$ and $\mathbf{j}_4$ are the solutions of the two 3×3 systems

$$\begin{aligned} \begin{bmatrix} u'_{41} & u'_{42} & u'_{43} \end{bmatrix} &= \mathbf{i}_4^T \begin{bmatrix} \mathbf{s}'_1 & \mathbf{s}'_2 & \mathbf{s}'_3 \end{bmatrix} \\ \begin{bmatrix} v'_{41} & v'_{42} & v'_{43} \end{bmatrix} &= \mathbf{j}_4^T \begin{bmatrix} \mathbf{s}'_1 & \mathbf{s}'_2 & \mathbf{s}'_3 \end{bmatrix} \end{aligned} \tag{5.4}$$

derived from equation (3.4). The second equation in (5.2) and the solution to (5.4) yield the entire rotation matrix R, while shape is given by equation (5.3).

The components a_4 and b_4 of translation in the fourth frame with respect to the centroid of all four points can be computed by postmultiplying equation (3.7) by the vector $\eta_4 = (1,1,1,0)^T$:

$$W\eta_4 = RS\eta_4 + \mathbf{t}\mathbf{e}_4^T\eta_4 \ .$$

Since $\mathbf{e}_4^T \eta_4 = 3$, we obtain

$$\mathbf{t} = \frac{1}{3}(W - RS)\eta_4 \ . \tag{5.5}$$

In particular, rows 4 and 8 of this equation yield a_4 and b_4. Notice that the unknown entries u_{44} and v_{44} are multiplied by zeros in equation (5.5).

Now that both motion and shape are known, the missing entries u_{44}, v_{44} of the measurement matrix W can be found by orthographic projection (equation (3.8)):

$$\begin{aligned} u_{44} &= \mathbf{i}_4^T \mathbf{s}_4 + a_4 \\ v_{44} &= \mathbf{j}_4^T \mathbf{s}_4 + b_4 \ . \end{aligned}$$

The procedure thus completed factors the full 6×4 submatrix of W and then reasons on the three points that are visible in all the frames to compute motion for the fourth frame. Alternatively, one can first apply factorization to the 8×3 submatrix

$$W_{8\times 3} = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \\ u_{41} & u_{42} & u_{43} \\ v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \\ v_{41} & v_{42} & v_{43} \end{bmatrix} \tag{5.6}$$

to produce the full translation and rotation submatrices

$$\mathbf{t}' = \begin{bmatrix} a'_1 \\ a'_2 \\ a'_3 \\ a'_4 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \end{bmatrix} \quad \text{and} \quad R = \begin{bmatrix} \mathbf{i}_1^T \\ \mathbf{i}_2^T \\ \mathbf{i}_3^T \\ \mathbf{i}_4^T \\ \mathbf{j}_1^T \\ \mathbf{j}_2^T \\ \mathbf{j}_3^T \\ \mathbf{j}_4^T \end{bmatrix} \tag{5.7}$$

and the partial shape matrix

$$S_{3\times 3} = \begin{bmatrix} \mathbf{s}'_1 & \mathbf{s}'_2 & \mathbf{s}'_3 \end{bmatrix} \tag{5.8}$$

such that

$$W_{8\times 3} = RS'_{3\times 3} + \mathbf{t}'\mathbf{e}_3^T \ .$$

The primes here signal again that coordinates refer to the centroid of only the first three points. Then, this partial solution can be extended to $\mathbf{s}'_4$ by solving the following overconstrained system of six equations in the three unknown entries of $\mathbf{s}'_4$:

$$\begin{bmatrix} \mathbf{i}_1^T \\ \mathbf{i}_2^T \\ \mathbf{i}_3^T \\ \mathbf{j}_1^T \\ \mathbf{j}_2^T \\ \mathbf{j}_3^T \end{bmatrix} \mathbf{s}'_4 + \begin{bmatrix} a'_1 \\ a'_2 \\ a'_3 \\ b'_1 \\ b'_2 \\ b'_3 \end{bmatrix} = \begin{bmatrix} u'_{14} \\ u'_{24} \\ u'_{34} \\ v'_{14} \\ v'_{24} \\ v'_{34} \end{bmatrix} \tag{5.9}$$

where

$$\begin{matrix} u'_{f4} = u_{f4} - a'_f \\ v'_{f4} = v_{f4} - b'_f \end{matrix} \quad \text{for} \quad f = 1,2,3 \ .$$

The "primed" shape coordinates can now be registered with respect to their centroid to yield the "unprimed" coordinates:

$$\mathbf{s}_p = \mathbf{s}'_p - \frac{1}{4} S'\mathbf{e}_4 \quad \text{for} \quad p = 1,2,3,4$$

and the "unprimed" translation can again be found from equation (5.5).

In summary, the full motion and shape solution can be found in either of the following ways:

1. factor $W_{6\times 4}$ to find a partial motion and full shape solution, and propagate it to include motion for the remaining frame (equations (5.4)). This will be used for reconstructing the complete W by row-wise extension.

2. factor $W_{8\times 3}$ to find a full motion and partial shape solution, and propagate it to include the remaining feature point (equation (5.9)). This will be used for reconstructing the complete W by column-wise extension.

5.2 Solution in the Presence of Noise

The solution propagation method introduced in the previous sectioncan be extended to $2F \times P$ measurement matrices with $F \geq 4$ and $P \geq 4$. In fact, the only difference is that the propagation equations (5.4) and (5.9) now become overconstrained. If the measurement matrix W is noisy, this redundancy is beneficial, since equations (5.4) and (5.9) can be solved in the Least Square Error sense, and the effect of noise is reduced.

In the general case of a noisy $2F \times P$ matrix W the solution propagation method can be summarized as follows. A possibly large, full subblock of W is first decomposed by factorization. Then, this initial solution is grown one row or one column at a time by solving systems analogous to those in (5.4) or (5.9) in the Least Square Error sense.

However, because of noise, the order in which the rows and columns of W are incorporated into the solution can affect the exact values of the final motion and shape solution. Consequently, once the solution has been propagated to the entire measurement matrix W, it may be necessary to refine the results with a steepest-descent minimization of the residue

$$\|W - RS - \frac{1}{P}\mathbf{t}\mathbf{e}_P^T\|$$

(see equation (3.7)).

There remain the two problems of how to choose the initial full subblock to which factorization is applied and in what order to grow the solution. In fact, however, because of the final refinement step, neither choice is critical as long as the initial matrix is large enough to yield a good starting point. We illustrate this point in the next chapterof experiments.

Chapter 6

More Experiments

We will first test the propagation method with image streams which include substantial occlusions. We first use an image stream taken in a laboratory. Then, we demonstrate the robustness of the factorization method with another stream taken with a hand-held amateur camera.

6.1 "Ping-Pong Ball" Image Stream

A ping-pong ball with black dots marked on its surface is rotated 450 degrees in front of the camera, so features appear and disappear. The rotation between adjacent frames is 2 degrees, so the stream is 226 frames long. Figure 6.14 shows the first frame of the stream, with the automatically selected features overlaid.

Every 30 frames (60 degrees) of rotation, the feature tracker looks for new features. In this way, features that disappear on one side around the ball are replaced by new ones that appear on the other side. Figure 6.15 shows the tracks of 60 features, randomly chosen among the total 829 found by the selector.

If all measurements are collected into the noisy measurement matrix W, the U and V parts of W have the same fill pattern: if the x coordinate of a measurement is known, so is the y coordinate. Figure 6.16 shows this *fill matrix* for our experiment. This matrix has the same size as either U or V, that is, $F \times P$. A column corresponds to a feature point, and a row to a frame. Shaded regions denote known entries. The fill matrix shown has

$226 \times 829 = 187354$ entries, of which 30185 (about 16 percent) are known.

To start the motion and shape computation, the algorithm finds a large full submatrix by applying simple heuristics based on typical patterns of the fill matrix. The choice of the starting matrix is not critical, as long as it leads to a reliable initialization of the motion and shape matrices. The initial solution is then grown by repeatedly solving overconstrained versions of the linear system corresponding to (5.4) to add new rows, and of the system corresponding to (5.9) to add new columns. The rows and columns to add are selected so as to maximize the redundancy of the linear systems. Eventually, all of the motion and shape values are determined. As a result, the unknown 84 percent of the measurement matrix can be hallucinated from the known 16 percent.

Figure 6.17 shows two views of the final shape results, taken from the top and from the side. The missing features at the bottom of the ball in the side view correspond to the part of the ball that remained always invisible, because it rested on the rotating platform.

To display the motion results, we look at the $\mathbf{i}_f$ and $\mathbf{j}_f$ vectors directly. We recall that these unit vectors point along the rows and columns of the image frames f in $1, \ldots, F$. Because the ping-pong ball rotates around a fixed axis, both $\mathbf{i}_f$ and $\mathbf{j}_f$ should sweep a cone in space, as shown in Figure 6.18. The tips of $\mathbf{i}_f$ and $\mathbf{j}_f$ should describe two circles in space, centered along the axis of rotation. Figure 6.19 shows two views of these vector tips, from the top and from the side. Those trajectories indicate that the motion recovery was done correctly. Notice the double arc in the top part of figure 6.19 corresponding to more than 360 degrees rotation. If the motion reconstruction were perfect, the two arcs would be indistinguishable.

6.2 "Cup and Hand" Image Stream

In this sectionwe describe an experiment with a natural scene including occlusion as a dominant phenomenon. A hand holds a cup and rotates it by about ninety degrees in front of the camera mounted on a fixed stand. Figure 6.20 shows four out of the 240 frames of the stream.

An additional need in this experiment is figure/ground segmentation. Since the camera was fixed, however, this problem is easily solved: features that do not move belong to the background. Also, the stream includes some

nonrigid motion: as the hand turns, the configuration and relative position of the fingers changes slightly. This effect, however, is small and did not affect the results appreciably.

A total of 207 features was selected. Occlusions were marked by hand in this experiment. The fill matrix of figure 6.22 illustrates the occlusion pattern. Figure 6.21 shows the image trajectory of 60 randomly selected features.

Figures 6.23 and 6.24 show a front and a top view of the cup and the visible fingers as reconstructed by the propagation method. The shape of the cup was recovered, as well as the rough shape of the fingers. These renderings were obtained, as for the "House" image stream in section4.1, by triangulating the tracked feature points and mapping pixel values onto the resulting surface.

Chapter 7

Conclusion

The rank theorem, which is the basis of the factorization method, is both surprising and powerful. Surprising because it states that the correlation among measurements made in an image stream has a simple expression *no matter what the camera motion is and no matter what the shape of an object is*, thus making motion or surface assumptions (such as smooth, constant, linear, planar and quadratic) fundamentally superfluous. Powerful because the rank theorem leads to factorization of the measurement matrix into shape and motion in a well-behaved and stable manner.

The factorization method exploits the redundancy of the measurement matrix to counter the noise sensitivity of structure-from-motion and allows using very short inter-frame camera motion to simplify feature tracking. The structural insight into shape-from-motion afforded by the rank theorem led to a systematic procedure to solve the occlusion problem within the factorization method. The experiments in the lab demonstrate the high accuracy of the method, and the outdoor experiments show its robustness.

The rank theorem is strongly related to Ullman's twelve year old result that three pictures of four points determine structure and motion under orthography. Thus, in a sense, the theoretical foundation of our result has been around for a long time. The factorization method evolves the applicability of that foundation from mathematical images to actual noisy image streams.

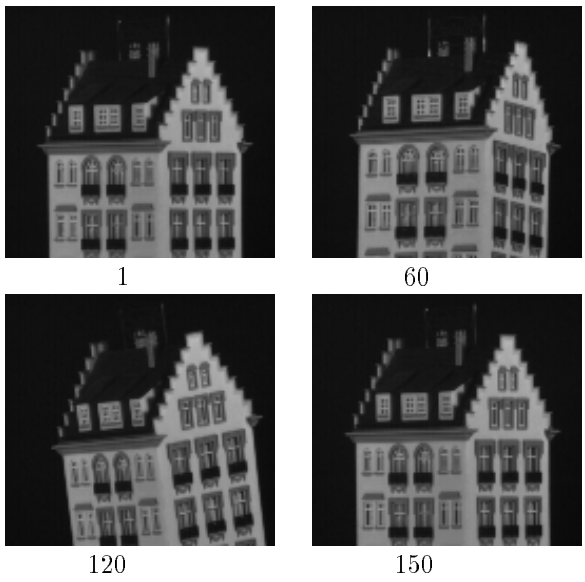


Figure 4.1: Some frames in the sequence. The whole sequence is 150 frames.

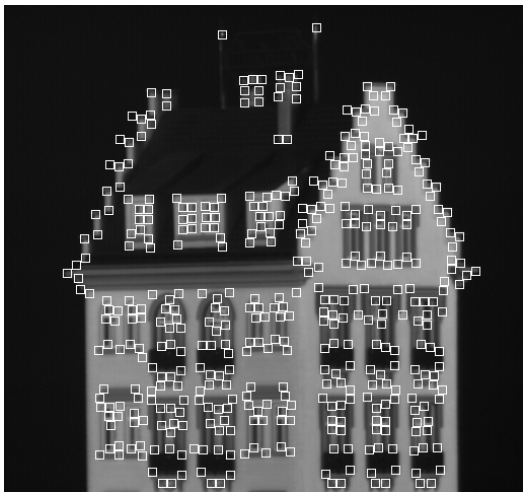


Figure 4.3: The 430 features selected by the automatic detection method.

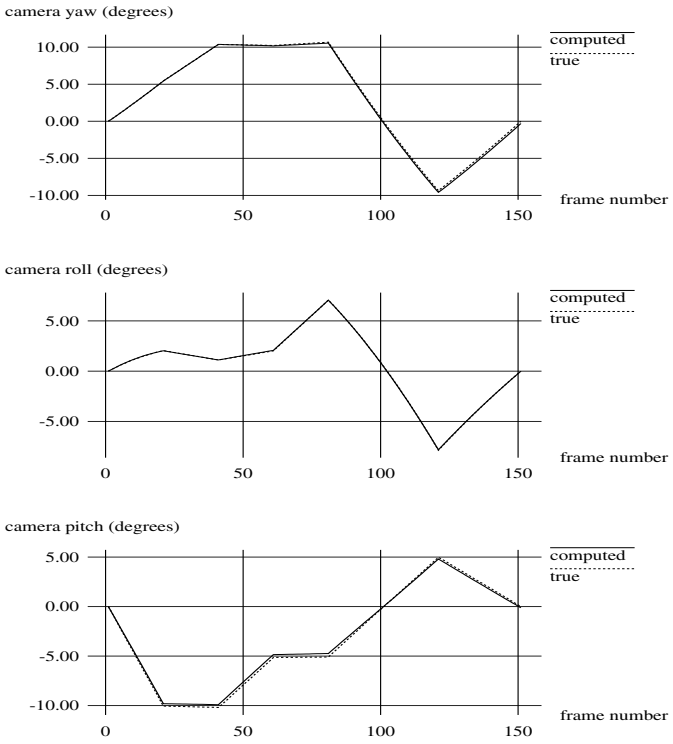


Figure 4.2: True and computed camera yaw, roll, pitch.

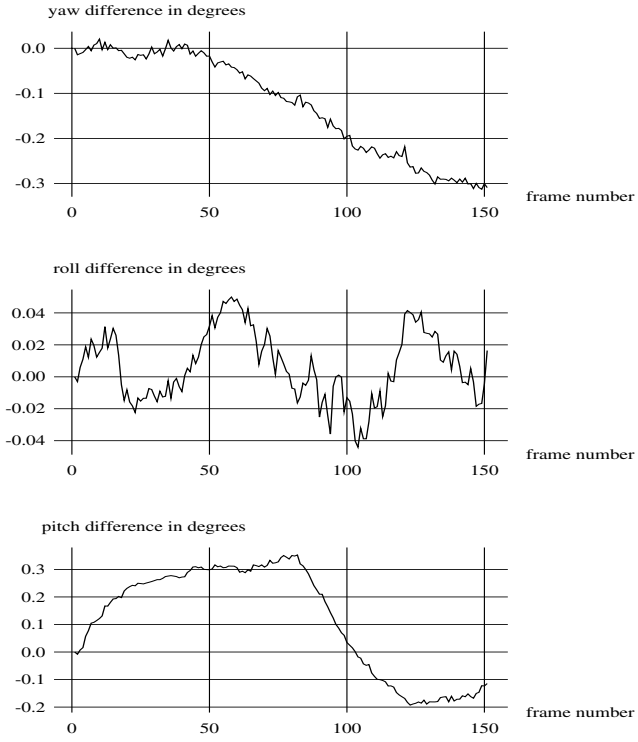


Figure 4.4: Blow-up of the errors in figure 4.2.

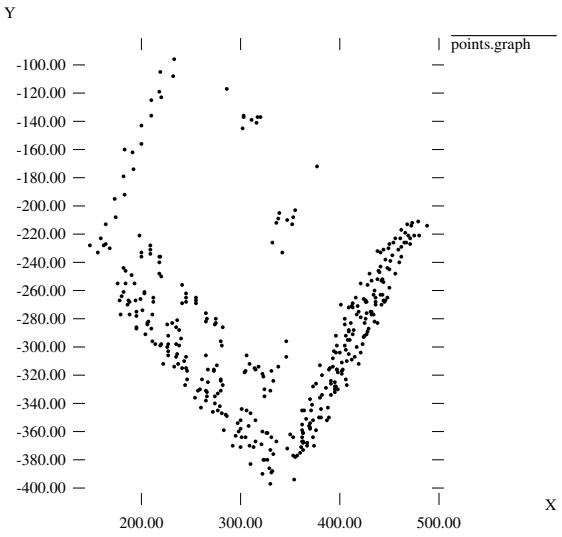


Figure 4.5: A view of the computed shape from approximately above the building (compare with figure 4.6).

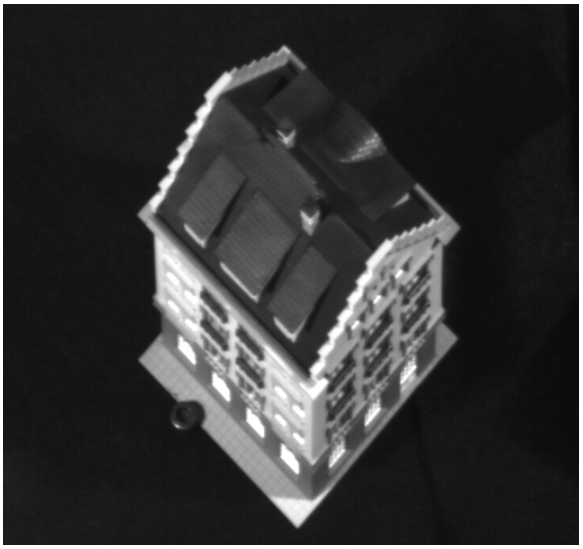


Figure 4.6: A real picture from above the building, similar to figure 4.5.

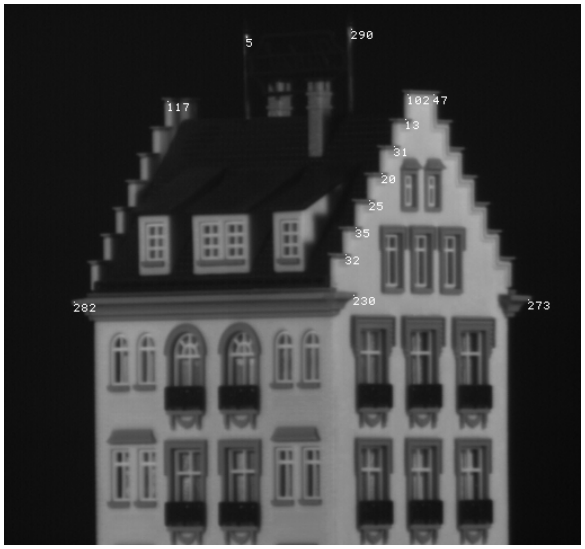


Figure 4.7: For a quantitative evaluation, distances between the features shown in the picture were measured on the actual model, and compared with the computed results. The comparison is shown in figure 4.8.

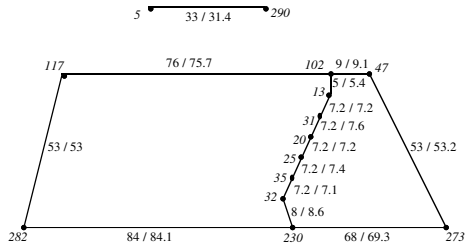


Figure 4.8: Comparison between measured and computed distances for the features in figure 4.7. The number before the slash is the measured distance, the one after is the computed distance. Lengths are in millimeters. Computed distances were scaled so that the computed distance between features 117 and 282 is the same as the measured distance.

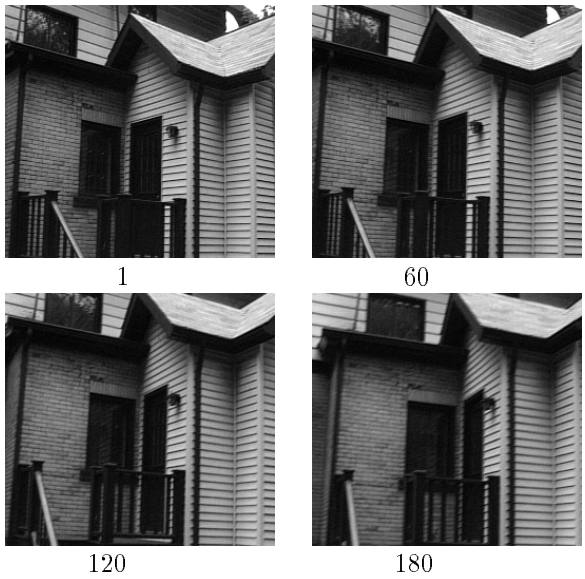


Figure 4.9: Four out of the 180 frames of the real house image stream.

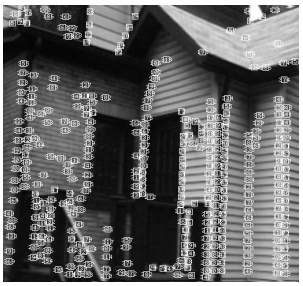


Figure 4.10: The features selected in the first frame of the real house stream (figure 4.9)
.

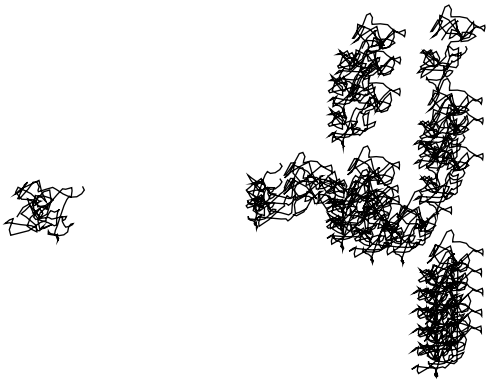


Figure 4.11: Tracks of 60 randomly selected features from the real house stream (figure 4.9.)

Figure 4.12: A front view of the three reconstructed walls, with the original image intensities mapped onto the resulting surface.

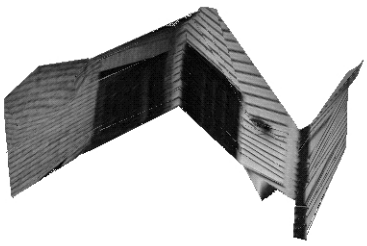


Figure 4.13: A view from above of the three reconstructed walls, with image intensities mapped onto the surface.

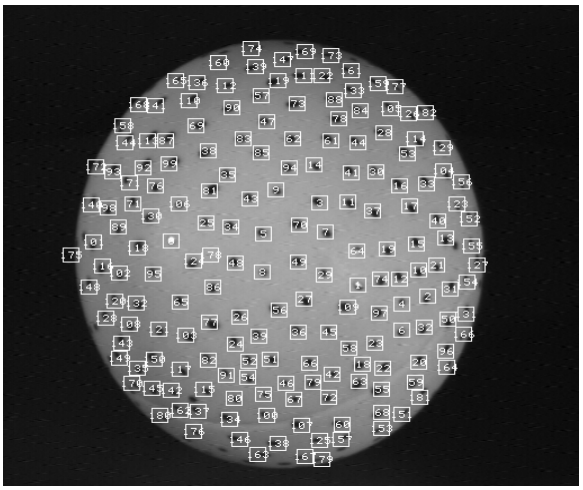


Figure 6.14: The first frame of the ping-pong stream, with overlaid features.

Figure 6.15: Tracks of 60 randomly selected features from the stream of figure 6.14.

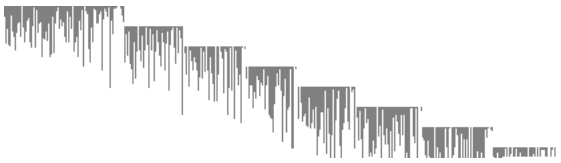


Figure 6.16: The fill matrix for the ping-pong ball experiment. Shaded entries are known.

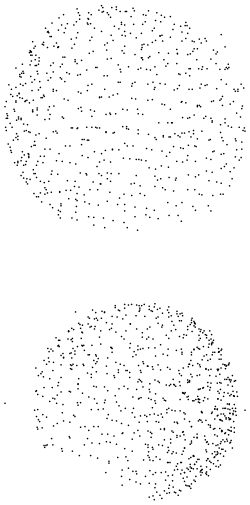


Figure 6.17: Top and side views of the reconstructed ping-pong ball.

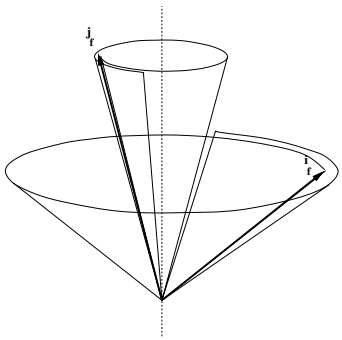


Figure 6.18: Rotational component of the camera motion for the ping-pong stream. Because rotation occurs around a fixed axis, the two mutually orthogonal unit vectors $\mathbf{i}_f$ and $\mathbf{j}_f$, pointing along rows and columns of the image sensor, sweep two 450-degree cones in space.

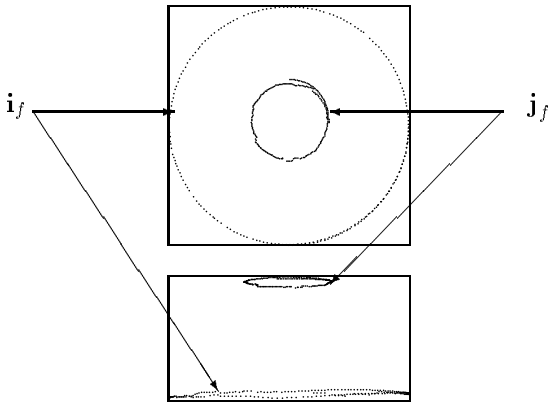


Figure 6.19: Top and side views of the $\mathbf{i}_f$ and $\mathbf{j}_f$ vectors identifying the camera rotation. See Figure 6.18.

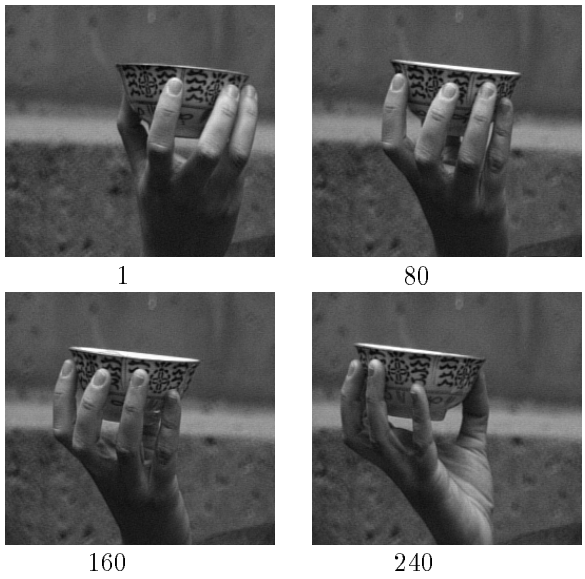


Figure 6.20: Four out of the 240 frames of the cup image stream.

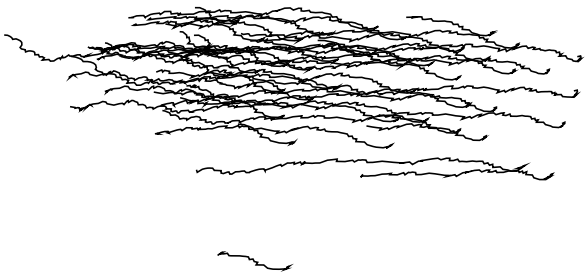


Figure 6.21: Tracks of 60 randomly selected features from the cup stream.

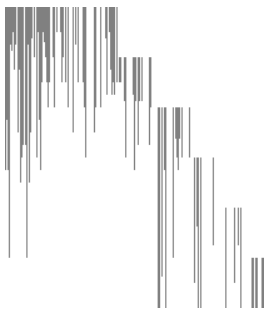


Figure 6.22: The 240×207 fill matrix for the cup stream (figure 6.20). Shaded entries are known.

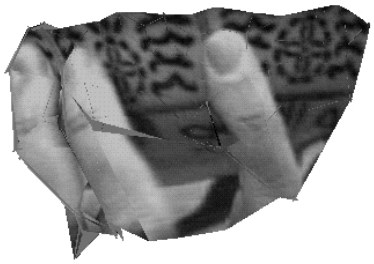


Figure 6.23: A front view of the cup and fingers, with the original image intensities mapped onto the resulting surface.

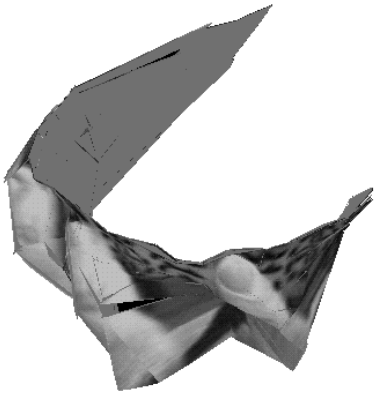


Figure 6.24: A view from above of the cup and fingers with image intensities mapped onto the surface.

Bibliography

[Adiv, 1985]
G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Pattern Analysis and Machine Intelligence*, 7:384–401, 1985.

[Bolles *et al.*, 1987]
R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1(1):7–55, 1987.

[Boult and Brown, 1991]
T. E. Boult and L. G. Brown. Factorization-based segmentation of motions. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 179–186, October 1991.

[Broida *et al.*, 1990]
T. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-d motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, July 1990.

[Bruss and Horn, 1983]
A. R. Bruss and B. K. P. Horn. Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21:3–20, 1983.

[Debrunner and Ahuja, 1990]
C. H. Debrunner and N. Ahuja. A direct data approximation based motion estimation algorithm. In *Proceedings of the 10th International Conference on Pattern Recognition*, pages 384–389, Atlantic City, NJ, June 1990.

[Debrunner and Ahuja, 1991]
C. H. DEBRUNNER AND N. AHUJA. Motion and structure factorization and segmentation of long multiple motion image sequences. Technical Report UI-BI-CV-5-91, CSL, Univ. of Illinois, Urbana-Champaign, IL, 1991.

[Golub and Reinsch, 1971]
G. H. GOLUB AND C. REINSCH. Singular value decomposition and least squares solutions, In *Handbook for Automatic Computation*, volume 2, chapter I/10, pages 134–151. Springer Verlag, New York, NY, 1971.

[Heeger and Jepson, 1989]
D. J. HEEGER AND A. JEPSON. Visual perception of three-dimensional motion. Technical Report 124, MIT Media Laboratory, Cambridge, Ma, December 1989.

[Heel, 1989]
J. HEEL. Dynamic motion vision. In *Proceedings of the DARPA Image Understanding Workshop*, pages 702–713, Palo Alto, Ca, May 23-26 1989.

[Horn *et al.*, 1988]
B. K. P. HORN, H. M. HILDEN, AND S. NEGAHDARIPOUR. Closed-form solution of absolute orientation using orthonormal matrices. *Journal of the Optical Society of America A*, 5(7):1127–1135, July 1988.

[Lucas and Kanade, 1981]
B. D. LUCAS AND T. KANADE. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, 1981.

[Matthies *et al.*, 1989]
L. MATTHIES, T. KANADE, AND R. SZELISKI. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3(3):209–236, September 1989.

[Prazdny, 1980]
K. PRAZDNY. Egomotion and relative depth from optical flow. *Biological Cybernetics*, 102:87–102, 1980.

[Roach and Aggarwal, 1979]
J. W. ROACH AND J. K. AGGARWAL. Computer tracking of objects

moving in space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):127–135, April 1979.

[Spetsakis and Aloimonos, 1989]
M. E. SPETSAKIS AND J. Y. ALOIMONOS. Optimal motion estimation. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 229–237, Irvine, California, March 1989.

[Tomasi and Kanade, 1990a]
C. TOMASI AND T. KANADE. Shape and motion without depth. In *Proceedings of the Third International Conference in Computer Vision (ICCV)*, Osaka, Japan, December 1990.

[Tomasi and Kanade, 1990b]
C. TOMASI AND T. KANADE. Shape and motion without depth. In *Proceedings of the DARPA Image Understanding Workshop*, pages 258–270, Pittsburgh, Pa, September 1990.

[Tomasi and Kanade, 1991]
C. TOMASI AND T. KANADE. Shape and motion from image streams: a factorization method - 2. point features in 3d motion. Technical Report CMU-CS-91-105, Carnegie Mellon University, Pittsburgh, PA, January 1991.

[Tomasi and Kanade, 1992]
C. TOMASI AND T. KANADE. Selecting and tracking features for image sequence analysis. *submitted to Robotics and Automation*, 1992.

[Tomasi, 1991]
C. TOMASI. *Shape and Motion from Image Streams: a Factorization Method.* PhD thesis, CMU, September 1991.

[Tsai and Huang, 1984]
R. Y. TSAI AND T. S. HUANG. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1):13–27, January 1984.

[Ullman, 1979]
S. ULLMAN. *The Interpretation of Visual Motion.* The MIT Press, Cambridge, Ma, 1979.

[Waxman and Wohn, 1985]
A. M. WAXMAN AND K. WOHN. Contour evolution, neighborhood deformation, and global image flow: planar surfaces in motion. *International Journal of Robotics Research*, 4:95–108, 1985.

A Multiple-Baseline Stereo

Masatoshi Okutomi, *Member, IEEE*, and Takeo Kanade, *Fellow, IEEE*

Abstract—**This paper presents a stereo matching method that uses multiple stereo pairs with various baselines to obtain precise distance estimates without suffering from ambiguity.**
In stereo processing, a short baseline means that the estimated distance will be less precise due to narrow triangulation. For more precise distance estimation, a longer baseline is desired. With a longer baseline, however, a larger disparity range must be searched to find a match. As a result, matching is more difficult, and there is a greater possibility of a false match. Therefore, there is a tradeoff between precision and accuracy in matching.
The stereo matching method presented in this paper uses multiple stereo pairs with different baselines generated by a lateral displacement of a camera. Matching is performed simply by computing the sum of squared-difference (SSD) values. The SSD functions for individual stereo pairs are represented with respect to the inverse distance (rather than the disparity, as is usually done) and are then simply added to produce the sum of SSD's. This resulting function is called the SSSD-in-inverse-distance. We show that the SSSD-in-inverse-distance function exhibits a unique and clear minimum at the correct matching position, even when the underlying intensity patterns of the scene include ambiguities or repetitive patterns. An advantage of this method is that we can eliminate false matches and increase precision without any search or sequential filtering.
This paper first defines a stereo algorithm based on the SSSD-in-inverse-distance and presents a mathematical analysis to show how the algorithm can remove ambiguity and increase precision. Then, a few experimental results with real stereo images are presented to demonstrate the effectiveness of the algorithm.

Index Terms— **Image matching, mulitple baselines, stereo vision, sum of squared differences, 3-D vision.**

I. Introduction

STEREO IS A useful technique for obtaining 3-D information from 2-D images in computer vision. In stereo matching, we measure the disparity d, which is the difference between the corresponding points of left and right images. The disparity d is related to the distance z by

$$d = BF\frac{1}{z} \tag{1}$$

where B and F are baseline and focal length, respectively.

Manuscript received January 16, 1991; revised February 7, 1992. This research was supported by the Defense Advanced Research Projects Agency (DOD) and monitored by the Avionics Laboratory, Air Force Wright Aeronautical Laboratories, Aeronautical Systems Division (AFSC) under Contract F33615-87-C-1499, ARPA Order No. 4976, Amendment 20. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or of the U.S. Government. Recommended for acceptance by Associate Editor T. Boult.

M. Okutomi is with the Information Systems Research Center, Canon Inc., Kawasaki, Japan

T. Kanade is with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.

IEEE Log Number 9206558.

This equation indicates that for the same distance, the disparity is proportional to the baseline or that the baseline length B acts as a magnification factor in measuring d in order to obtain z, that is, the estimated distance is more precise if we set the two cameras farther apart from each other, which means a longer baseline. A longer baseline, however, poses its own problem. Because a longer disparity range must be searched, matching is more difficult, and thus, there is a greater possibility of a false match. Therefore, there is a tradeoff between precision and accuracy (correctness) in matching.

One of the most common methods in dealing with the problem is a coarse-to-fine control strategy [1]–[5]. Matching is done at a low resolution to reduce false matches, and then, the result is used to limit the search range of matching at a higher resolution, where more precise disparity measurements are calculated. Using a coarse resolution, however, does not always remove false matches. This is especially true when there is inherent ambiguity in matching, such as a repeated pattern over a large part of the scene (e.g., a scene of a picket fence). Another approach to remove false matches and to increase precision is to use multiple images, especially a sequence of densely sampled images along a camera path [6]–[9]. A short baseline between a pair of consecutive images makes the matching or tracking of features easy, whereas the structure imposed by the camera motion allows integration of the possibly noisy individual measurements into a precise estimate. The integration has been performed either by exploiting constraints on the EPI [6], [7] or by a sequential Kalman filtering technique [8], [9].

The stereo matching method presented in this paper belongs to the second approach: use of multiple images with different baselines obtained by a lateral displacement of a camera. The matching technique, however, is based on the idea that global mismatches can be reduced by adding the sum of squared-difference (SSD) values from multiple stereo pairs, that is, the SSD values are computed first for each pair of stereo images. We represent the SSD values with respect to the inverse distance $1/z$ (rather than the disparity d, as is usually done). The resulting SSD functions from all stereo pairs are added together to produce the sum of SSD's, which we call SSSD-in-inverse-distance. We show that the SSSD-in-inverse-distance function exhibits a unique and clear minimum at the correct matching position, even when the underlying intensity patterns of the scene include ambiguities or repetitive patterns.

There have been stereo techniques that use multiple image pairs taken by cameras that are arranged along a line [10]–[12], in the form of a triangle [13]–[15] (called trinocular stereo), or in the other formation [16]. However, all of these techniques, except [10] and [16], decide candidate points

for correspondence in each image pair and then search for the correct combinations of correspondences among them using the geometrical consistencies they must satisfy. Since the intermediate decisions on correspondences are inherently noisy, ambiguous, and multiple, finding the correct combinations requires sophisticated consistency checks and search or filtering. In contrast, our method does not make any decisions about the correspondences in each stereo image pair; instead, it simply accumulates the measures of matching (SSD's) from all the stereo pairs into a single evaluation function, i.e., SSSD-in-inverse-distance, and then obtains one corresponding point from it. In other words, our method integrates *evidence* for a final decision rather than filtering intermediate *decisions*. In this sense, Tsai [16] employed strategy very similar to ours; he used multiple images to sharpen the peaks of his overall similarity measures, which he called JMM and WVM. However, the relationship between the improvement of the similarity measures and the camera baseline arrangement was not analyzed, nor was the method tested with real imagery. In this paper, we show both mathematical analysis and experimental results with real indoor and outdoor images, which demonstrate how the SSSD-in-inverse-distance function based on multiple image pairs from different baselines can greatly reduce false matches while improving precision.

In the next section, we present the method mathematically and show how ambiguity can be removed and precision increased by the method. Section III provides a few experimental results with real stereo images to demonstrate the effectiveness of the algorithm. Section IV presents conclusions.

II. Mathematical Analysis

The essence of stereo matching is, given a point in one image, to find in another image the corresponding point such that the two points are the projections of the same physical point in space. This task usually requires some criterion to measure similarity between images. The SSD of the intensity values (or values of preprocessed images, such as bandpass filtered images) over a window is the simplest and most effective criterion. In this section, we define the sum of SSD with respect to the inverse distance (SSSD-in-inverse-distance) for multiple-baseline stereo and mathematically show its advantage in removing ambiguity and increasing precision. For this analysis, we use 1-D stereo intensity signals, but the extension to 2-D images is straightforward.

A. SSD Function

Suppose that we have cameras at positions $P_0, P_1, \ldots, P_n$ along a line with their optical axes perpendicular to the line and a resulting set of stereo pairs with baselines $B_1, B_2, \ldots, B_n$, as shown in Fig. 1. Let $f_0(x)$ and $f_i(x)$ be the image pair at the camera positions P_0 and P_i, respectively. Imagine a scene point Z whose distance is z. Its disparity $d_{r(i)}$ for the image pair taken from P_0 and P_i is

$$d_{r(i)} = \frac{B_i F}{z}. \tag{2}$$

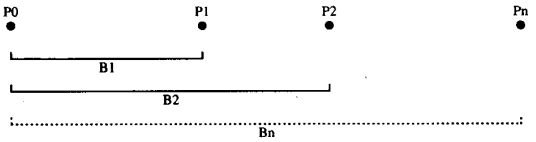


Fig. 1. Camera positions for stereo.

We model the image intensity functions $f_0(x)$ and $f_i(x)$ near the matching positions for Z as

$$\begin{aligned} f_0(x) &= f(x) + n_0(x) \\ f_i(x) &= f(x - d_{r(i)}) + n_i(x) \end{aligned} \tag{3}$$

assuming constant distance near Z and independent Gaussian white noise such that

$$n_0(x), n_i(x) \sim N(0, \sigma_n^2). \tag{4}$$

The SSD value $e_{d(i)}$ over a window W at a pixel position x of image $f_0(x)$ for the candidate disparity $d_{(i)}$ is defined as

$$e_{d(i)}(x, d_{(i)}) \equiv \sum_{j \in W} (f_0(x + j) - f_i(x + d_{(i)} + j))^2 \tag{5}$$

where the $\sum_{j \in W}$ means summation over the window. The $d_{(i)}$ that gives a minimum of $e_{d(i)}(x, d_{(i)})$ is determined as the estimate of the disparity at x. Since the SSD measurement $e_{d(i)}(x, d_{(i)})$ is a random variable, we will compute its expected value in order to analyze its behavior:

$$\begin{aligned} &E[e_{d(i)}(x, d_{(i)})] \\ &= E\left[\sum_{j \in W} (f(x + j) - f(x + d_{(i)} - d_{r(i)} + j) \right. \\ &\qquad \left. + n_0(x + j) - n_i(x + d_{(i)} + j))^2\right] \\ &= E\left[\sum_{j \in W} (f(x + j) - f(x + d_{(i)} - d_{r(i)} + j))^2\right] \\ &\quad + E\left[\sum_{j \in W} 2(f(x + j) - f(x + d_{(i)} - d_{r(i)} + j) \right. \\ &\qquad \left. \cdot (n_0(x + j) - n_i(x + d_{(i)} + j))\right] \\ &\quad + E\left[\sum_{j \in W} (n_0(x + j) - n_i(x + d_{(i)} + j))^2\right] \\ &= \sum_{j \in W} (f(x + j) - f(x + d_{(i)} - d_{r(i)} + j))^2 + 2N_w\sigma_n^2 \end{aligned} \tag{6}$$

where N_w is the number of the points within the window. For the rest of the paper, $E[]$ denotes the expected value of a random variable. In deriving the above equation, we have assumed that $d_{r(i)}$ is constant over the window. Equation (6) says that naturally, the SSD function $e_{d(i)}(x, d_{(i)})$ is *expected* to take a minimum when $d_{(i)} = d_{r(i)}$, i.e., at the right disparity.

Let us examine how the SSD function $e_{d(i)}(x, d_{(i)})$ behaves when there is ambiguity in the underlying intensity function. Suppose that the intensity signal $f(x)$ has the same pattern around pixel positions x and $x + a$

$$f(x + j) = f(x + a + j), \qquad j \in W \tag{7}$$

where $a \neq 0$ is a constant. Then, from (6)

$$E[e_{d(i)}(x, d_{r(i)})] = E[e_{d(i)}(x, d_{r(i)} + a)] = 2N_w\sigma_n^2. \tag{8}$$

This means that ambiguity is expected in matching in terms of positions of minimum SSD values. Moreover, the false match at $d_{r(i)} + a$ appears in exactly the same way for all i; it is separated from the correct match by a for all the stereo pairs. Using multiple baselines does not help to disambiguate.

B. SSD with Respect to Inverse Distance

Now, let us introduce the *inverse distance* ζ such that

$$\zeta = \frac{1}{z}. \tag{9}$$

From (2)

$$d_{r(i)} = B_i F \zeta_r \tag{10}$$

$$d_{(i)} = B_i F \zeta \tag{11}$$

where ζ_r and ζ are the real and the candidate inverse distance, respectively. Substituting (11) into (5), we have the SSD with respect to the inverse distance

$$e_{\zeta(i)}(x, \zeta) \equiv \sum_{j \in W} (f_0(x + j) - f_i(x + B_i F \zeta + j))^2 \tag{12}$$

at position x for a candidate inverse distance ζ. Its expected value is

$$E[e_{\zeta(i)}(x, \zeta)] = \sum_{j \in W} (f(x + j) - f(x + B_i F(\zeta - \zeta_r) + j))^2 + 2N_w\sigma_n^2. \tag{13}$$

Finally, we define a new evaluation function $e_{\zeta(12\cdots n)}(x, \zeta)$, which is the sum of SSD functions with respect to the inverse distance (SSSD-in-inverse-distance) for multiple stereo pairs. It is obtained by adding the SSD functions $e_{\zeta(i)}(x, \zeta)$ for individual stereo pairs:

$$e_{\zeta(12\cdots n)}(x, \zeta) = \sum_{i=1}^{n} e_{\zeta(i)}(x, \zeta). \tag{14}$$

Its expected value is

$$\begin{aligned} E[e_{\zeta(12\cdots n)}(x, \zeta)] &= \sum_{i=1}^{n} E[e_{\zeta(i)}(x, \zeta)] \\ &= \sum_{i=1}^{n} \sum_{j \in W} (f(x + j) \\ &\quad - f(x + B_i F(\zeta - \zeta_r) + j))^2 + 2nN_w\sigma_n^2. \end{aligned} \tag{15}$$

In the next three subsections, we will analyze the characteristics of these evaluation functions to see how ambiguity is removed and precision is improved.

C. Elimination of Ambiguity 1

As before, suppose the underlying intensity pattern $f(x)$ has the same pattern around x and $x + a$ (see (7)). Then, according to (13), we have

$$E[e_{\zeta(i)}(x, \zeta_r)] = E[e_{\zeta(i)}(x, \zeta_r + \frac{a}{B_i F})] = 2N_w\sigma_n^2. \tag{16}$$

We still have an ambiguity; a minimum is expected at a false inverse distance $\zeta_f = \zeta_r + \frac{a}{B_i F}$. However, an important point to be observed here is that this minimum for the false inverse distance ζ_f changes its position as the baseline B_i changes, whereas the minimum for the correct inverse distance ζ_r does not. This is the property that the new evaluation function, the SSSD-in-inverse-distance (14), exploits to eliminate the ambiguity. For example, suppose we use two baselines B_1 and B_2 ($B_1 \neq B_2$). From (15)

$$\begin{aligned} E[e_{\zeta(12)}(x, \zeta)] &= \\ &\sum_{j \in W} (f(x + j) - f(x + B_1 F(\zeta - \zeta_r) + j))^2 \\ &+ \sum_{j \in W} (f(x + j) - f(x + B_2 F(\zeta - \zeta_r) + j))^2 + 4N_w\sigma_n^2. \end{aligned} \tag{17}$$

We can prove that

$$E[e_{\zeta(12)}(x, \zeta)] > 4N_w\sigma_n^2 = E[e_{\zeta(12)}(x, \zeta_r)] \quad \text{for } \zeta \neq \zeta_r. \tag{18}$$

(refer to Appendix A) In words, $e_{\zeta(12)}(x, \zeta)$ is *expected* to have the smallest value at the correct ζ_r, that is, the ambiguity is likely to be eliminated by use of the new evaluation function with two different baselines.

We can illustrate this using synthesized data. Suppose the point whose distance we want to determine is at $x = 0$, and the underlying function $f(x)$ is given by

$$f(x) = \begin{cases} cos(\frac{\pi}{4}x) + 2 & \text{if } -4 < x < 12 \\ 1 & \text{if } x \leq -4 \text{ or } 12 \leq x. \end{cases} \tag{19}$$

Fig. 2(a) shows a plot of $f(x)$. Assuming that $d_{r(1)} = 5$, $\sigma_n^2 = 0.2$, and the window size is 5, the expected values of the SSD function $e_{d(1)}(x, d_{(1)})$ are as shown in Fig. 2(b). We see that there is an ambiguity; the minima occur at the correct match $d_{(1)} = 5$ and at the false match $d_{(1)} = 13$. The match that will be selected will depend on the noise, search range, and search strategy. Now, suppose we have a longer baseline B_2 such that $\frac{B_2}{B_1} = 1.5$. From equations (6) and (10), we obtain $E[e_{d(2)}]$ as shown in Fig. 2(c). Again, we encounter an ambiguity, and the separation of the two minima is the same.

Now, let us evaluate the SSD values with respect to the inverse distance ζ rather than the disparity d by using (12) through (15). The expected values of the SSD measurements $E[e_{\zeta(1)}]$ and $E[e_{\zeta(2)}]$ with baselines B_1 and B_2 are shown in Figs. 2(d) and (e), respectively (the plot is normalized such that $B_1 F = 1$). Note that the minima at the correct inverse distance ($\zeta = 5$) does not move, whereas the minima for the false match changes its position as the baseline changes. When the two functions are added to produce the SSSD-in-inverse-distance, its expected values $E[e_{\zeta(12)}]$ are as shown in Fig. 2(f). We can

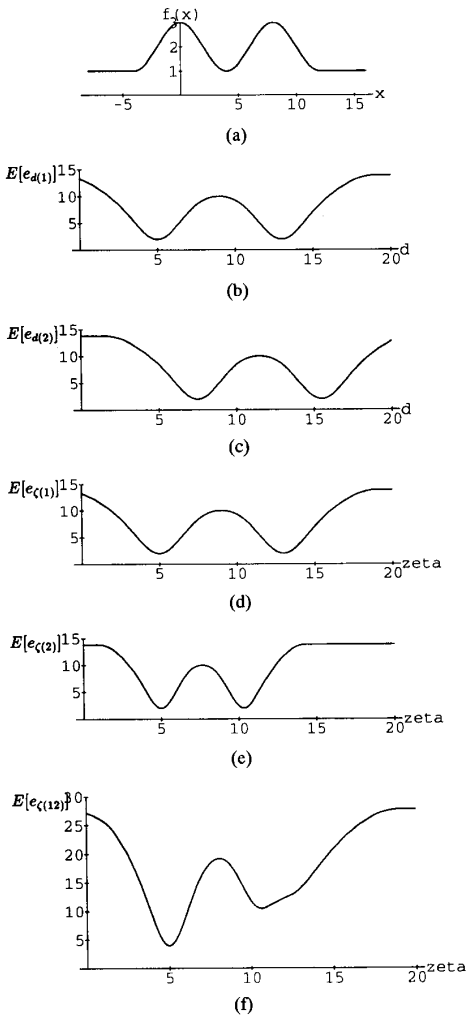


Fig. 2. Expected values of evaluation functions: (a) Underlying function; (b) $E[e_{d(1)}]$; (c) $E[e_{d(2)}]$; (d) $E[e_{\zeta(1)}]$; (e) $E[e_{\zeta(2)}]$; (f) $E[e_{\zeta(12)}]$.

see that the ambiguity has been reduced because the SSSD-in-inverse-distance has a smaller value at the correct match position than at the false match.

D. Elimination of Ambiguity 2

An extreme case of ambiguity occurs when the underlying function $f(x)$ is a periodic function, like a scene of a picket fence. We can show that this ambiguity can also be eliminated.

Let $f(x)$ be a periodic function with period T. Then, each $e_{\zeta(i)}(x,\zeta)$ is expected to be a periodic function of ζ with the period $\frac{T}{B_iF}$. This means that there will be multiple minima of $e_{\zeta(i)}(x,\zeta)$ (i.e., ambiguity in matching) at intervals of $\frac{T}{B_iF}$ in ζ. When we use two baselines and add their SSD values, the resulting $e_{\zeta(12)}(x,\zeta)$ will still be a periodic function of ζ, but its period T_{12} is increased to

$$T_{12} = LCM\left(\frac{T}{B_1F}, \frac{T}{B_2F}\right) \tag{20}$$

where $LCM()$ denotes least common multiple, that is, the period of the expected value of the new evaluation function can be made longer than that of the individual stereo pairs. Furthermore, it can be controlled by choosing the baselines B_1 and B_2 appropriately so that the expected value of the evaluation function has only one minimum within the search range. This means that using multiple-baseline stereo pairs simultaneously can eliminate ambiguity, although each individual baseline stereo may suffer from ambiguity.

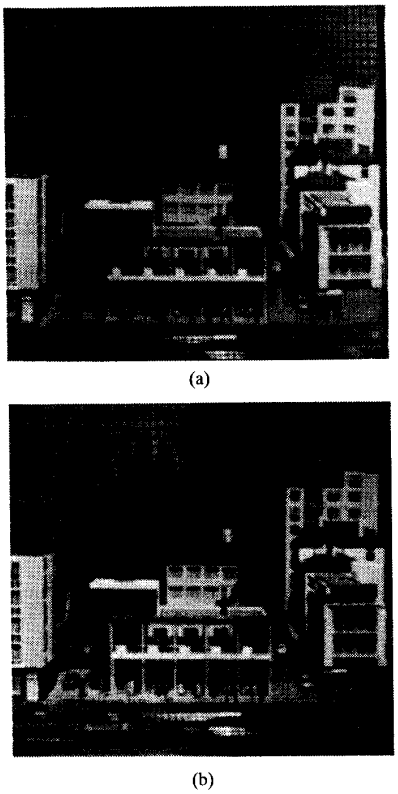


Fig. 3. "Town" data set: (a) Image0; (b) image9.

We illustrate this by using real stereo images. Fig. 3(a) shows an image of a sample scene. At the top of the scene, there is a grid board whose intensity function is nearly periodic. We took ten images of this scene by shifting the camera vertically as in Fig. 4. The actual distance between consecutive camera positions is 0.05 in. Let this distance be b. Fig. 3 shows the first and the last images of the sequence. We selected a point x within the repetitive grid board area in image9. The SSD values $e_{\zeta(i)}(x,\zeta)$ over 5-by-5-pixel windows are plotted for various baseline stereo pairs in Fig. 5. The horizontal axis of all the plots is the inverse distance, normalized such that $8bF = 1$. Fig. 5 illustrates the tradeoff between precision and ambiguity in terms of baselines, that is, for a shorter baseline, there are fewer minima (i.e., less ambiguity), but the SSD curve is flatter (i.e., less precise localization). On the other hand, for a longer baseline, there are more minima (i.e., more ambiguity), but the curve near the minimum is sharper, that is, the estimated distance is more precise if we can find the correct one.

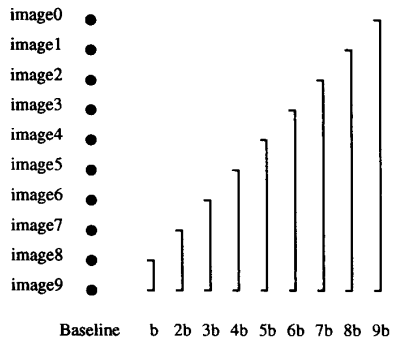


Fig. 4. "Town" data set image sequence.

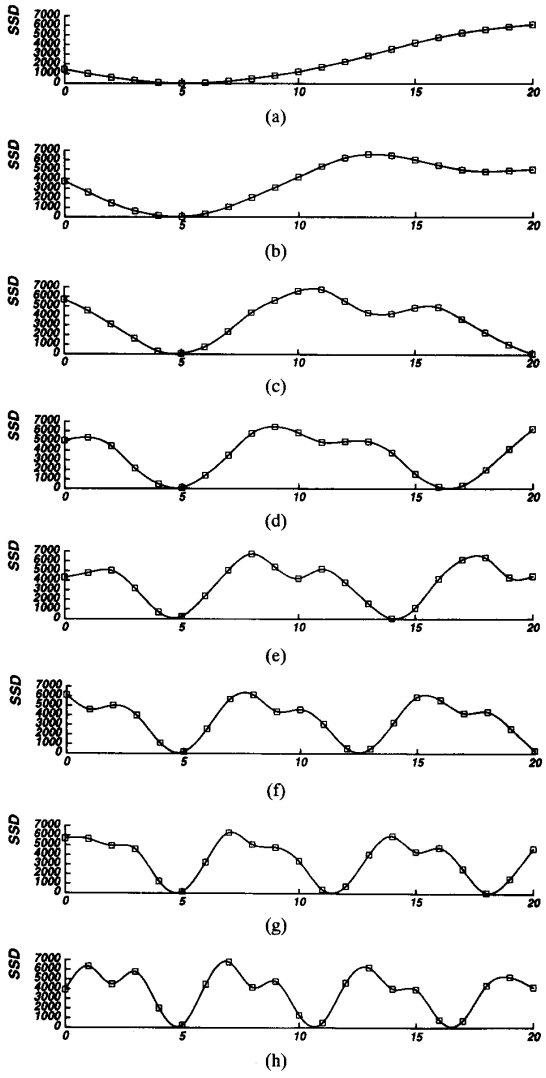


Fig. 5. SSD values versus inverse distance: (a) $B = b$; (b) $B = 2b$; (c) $B = 3b$; (d) $B = 4b$; (e) $B = 5b$; (f) $B = 6b$; (g) $B = 7b$; (h) $B = 8b$. The horizontal axis is normalized such that $8bF = 1$.

Now, let us take two stereo image pairs: one with $B = 5b$ and the other with $B = 8b$. In Fig. 6, the dashed curve and the dotted curve show the SSD for $B = 5b$ and $B = 8b$, respectively. Let us suppose the search range goes from 0 to 20 in the horizontal axis, which in this case corresponds to 12 to ∞ inches in distance. Although the SSD values take a minimum at the correct answer near $\zeta = 5$, there are also other minima for both cases. The solid curve shows the evaluation function for the multiple-baseline stereo, which is the sum of the dashed curve and the dotted curve. The solid curve shows only one clear minimum, that is, the ambiguity is resolved.

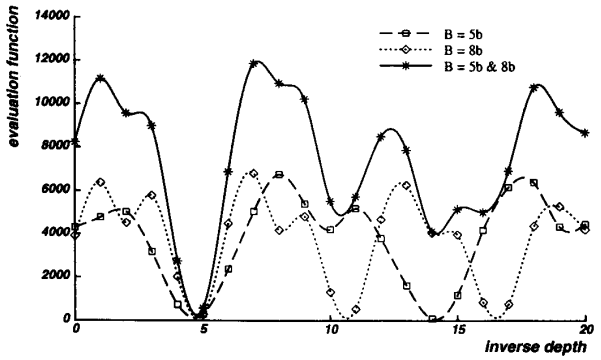


Fig. 6. Combining two stereo pairs with different baselines.

Thus far, we have considered using only two stereo pairs. We can easily extend the idea to multiple-baseline stereo, which uses more than two stereo pairs. Corresponding to (20), the period of $E[e_{\zeta(12\cdots n)}(x,\zeta)]$ becomes

$$T_{12,\ldots,n} = LCM\left(\frac{T}{B_1F}, \frac{T}{B_2F}, \ldots, \frac{T}{B_nF}\right) \qquad (21)$$

where $B_1, B_2, \ldots, B_n$ are baselines for each stereo pair.

We will demonstrate how the ambiguity can be further reduced by increasing the number of stereo pairs. From the data of Fig. 4, we first choose image1 and image9 as a long baseline stereo pair, i.e., 1) $B = 8b$. Then, we increase the number of stereo pairs by dividing the baseline between image1 and image9, i.e., 2) $B = 4b$ and $8b$, 3) $B = 2b$, $4b$, $6b$, and $8b$, 4) $B = b$, $2b$, $3b$, $4b$, $5b$, $6b$, $7b$, and $8b$. Fig. 7 demonstrates that the SSSD-in-inverse-distance shows the minimum at the correct position more clearly as more stereo pairs are used.

E. Precision

We have shown that ambiguities can be resolved by using the SSSD-in-inverse-distance computed from multiple baseline stereo pairs. The technique also increases precision in estimating the true inverse distance. We can show this by analyzing the statistical characteristics of the evaluation functions near the correct match.

From (3), (10), and (12), we have

$$e_{\zeta(i)}(x,\zeta) = \sum_{j\in W}(f(x+j) - f(x+B_iF(\zeta-\zeta_r)+j) + n_0(x+j) - n_i(x+B_iF\zeta+j))^2. \qquad (22)$$

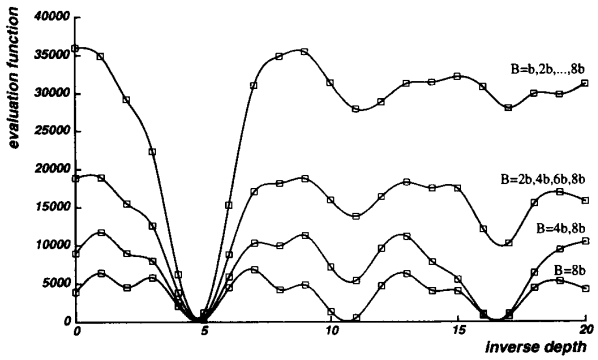


Fig. 7. Combining multiple baseline stereo pairs.

By taking the Taylor expansion about $\zeta = \zeta_r$ up to the linear terms, we obtain

$$f(x + B_i F(\zeta - \zeta_r) + j) \approx f(x + j) + B_i F(\zeta - \zeta_r) f'(x + j). \tag{23}$$

Substituting this into (22), we can approximate $e_{\zeta(i)}(x, \zeta)$ near ζ_r by a quadratic form of ζ:

$$\begin{aligned} e_{\zeta(i)}(x, \zeta) &\approx \sum_{j \in W} (-B_i F(\zeta - \zeta_r) f'(x + j) \\ &\quad + n_0(x + j) - n_i(x + B_i F \zeta + j))^2 \\ &= B_i^2 F^2 a(x)(\zeta - \zeta_r)^2 \\ &\quad + 2 B_i F b_i(x)(\zeta - \zeta_r) + c_i(x) \end{aligned} \tag{24}$$

where

$$a(x) = \sum_{j \in W} (f'(x + j))^2 \tag{25}$$

$$\begin{aligned} b_i(x) = \sum_{j \in W} & f'(x + j) \cdot \\ & (n_i(x + B_i F \zeta + j) - n_0(x + j)) \end{aligned} \tag{26}$$

$$c_i(x) = \sum_{j \in W} (n_i(x + B_i F \zeta + j) - n_0(x + j))^2. \tag{27}$$

The estimated inverse distance $\hat{\zeta}_{r(i)}$ is the value ζ that makes (24) minimum:

$$\hat{\zeta}_{r(i)} = \zeta_r - \frac{b_i(x)}{B_i F a(x)}. \tag{28}$$

Since $E[b_i(x)] = 0$, the expected value of the estimate $\hat{\zeta}_{r(i)}$ is the correct value ζ_r, but it varies due to the noise. The variance of this estimate is

$$\begin{aligned} Var(\hat{\zeta}_{r(i)}) &= \frac{Var(b_i(x))}{B_i^2 F^2 (a(x))^2} \\ &= \frac{2\sigma_n^2}{B_i^2 F^2 a(x)}. \end{aligned} \tag{29}$$

Basically, this equation states that for the same amount of image noise σ_n^2, the variance is smaller (the estimate is more precise) as the baseline B_i is longer or as the variation of intensity signal $a(x)$ is larger.

We can follow the same analysis for $e_{\zeta(12\cdots n)}(x, \zeta)$ of (14), which is the new evaluation function with multiple baselines. Near ζ_r, it is

$$\begin{aligned} e_{\zeta(12\cdots n)}(x, \zeta) \approx & \left(\sum_{i=1}^{n} B_i^2 \right) F^2 a(x)(\zeta - \zeta_r)^2 \\ & + 2F \left(\sum_{i=1}^{n} B_i b_i(x) \right) (\zeta - \zeta_r) + \sum_{i=1}^{n} c_i(x). \end{aligned} \tag{30}$$

The variance of the estimated inverse distance $\hat{\zeta}_{r(12\cdots n)}$ that minimizes this function is

$$Var(\hat{\zeta}_{r(12\cdots n)}) = \frac{2\sigma_n^2}{(\sum_{i=1}^{n} B_i^2) F^2 a(x)}. \tag{31}$$

From (29) and (31), we see that

$$\frac{1}{Var(\hat{\zeta}_{r(12\cdots n)})} = \sum_{i=1}^{n} \frac{1}{Var(\hat{\zeta}_{r(i)})}. \tag{32}$$

The inverse of the variance represents the precision of the estimate. Therefore, (32) means that by using the SSSD-in-inverse-distance with multiple baseline stereo pairs, the estimate becomes more precise. We can confirm this characteristic in Figs. 6 and 7 by observing that the curve around the correct inverse distance becomes sharper as more baselines are used.

III. Experimental Results

This section presents experimental results of the multiple-baseline stereo based on SSSD-in-inverse-distance with real 2-D images. A complete description of the algorithm is included in Appendix B.

The first result is for the "Town" data set that we showed in Fig. 3. Fig. 8 is the distance map and its isometric plot with a short baseline $B = 3b$. The result with a single long baseline $B = 9b$ is shown in Fig. 9. Comparing these two results, we observe that the distance map computed by using the long baseline is smoother on flat surfaces, i.e., more precise, but has gross errors in matching at the top of the scene because of the repeated pattern. These results illustrate the tradeoff between ambiguity and precision. Fig. 10, on the other hand, shows the distance map and its isometric plot obtained by the new algorithm using three different baselines $3b$, $6b$, and $9b$. For comparison, the corresponding oblique view of the scene is shown in Fig. 11. We can note that the computed distance map is less ambiguous *and* more precise than those of the single-baseline stereo.

Fig. 12 shows another data set used for our experiment. Figs. 13 and 14 compare the distance maps computed from the short baseline stereo and the long baseline stereo; the longer baseline is five times longer than the short one. For comparison, the actual oblique view roughly corresponding to the isometric plot is shown in Fig. 15. Although no repetitive patterns are apparent in the images, we can still observe gross errors in the distance map obtained with the long baseline due to false

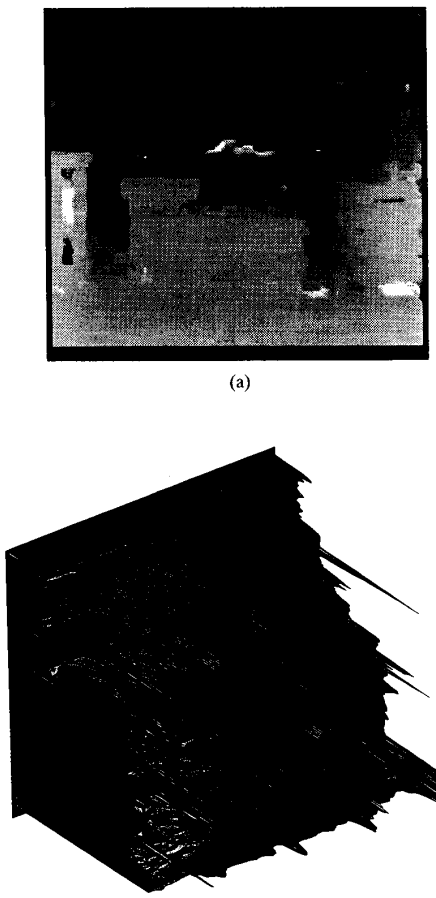


Fig. 8. Result with a short baseline $B = 3b$: (a) Distance map; (b) isometric plot of the distance map from the upper left corner. The matching is mostly correct but very noisy.

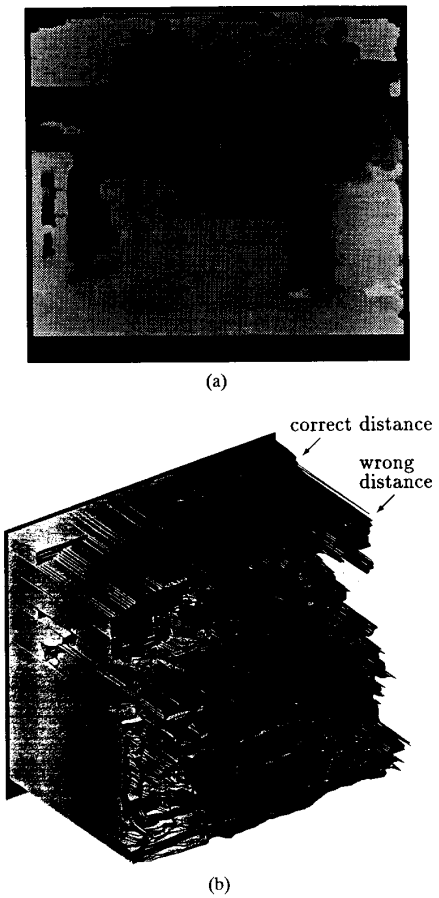
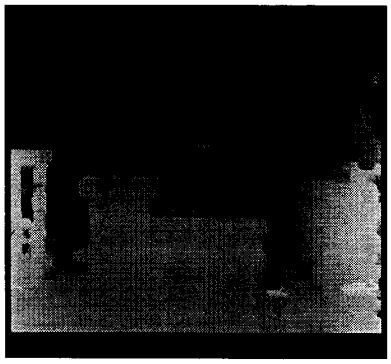


Fig. 9. Result with a long baseline $B = 9b$: (a) Distance map; (b) isometric plot. The matching is less noisy when it is correct. However, there are many gross mistakes, especially in the top of the image where, due to a repetitive pattern, the matching is completely wrong.

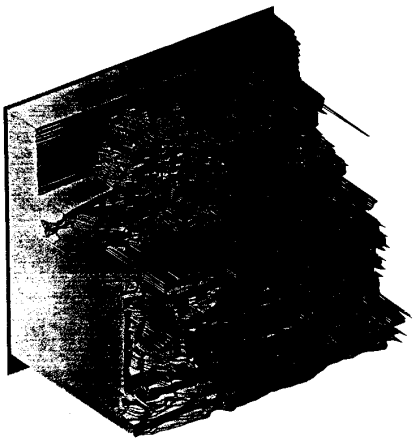
matching. In contrast, the result from the multiple-baseline stereo shown in Fig. 16 demonstrates both the advantage of unambiguous matching with a short baseline and that of precise matching with a long baseline.

Fig. 17(a) and (b) shows one of the real outdoor scenes to which the multiple-baseline stereo technique has been applied. The distance to the front object (curb) is roughly 20 m, and it is another 8 m to the building wall. We used a Sony CCD camera with a 50-mm lens and captured six images (five stereo image pairs) by moving the camera horizontally. The baseline between the neighboring camera positions is 1.9 cm so that the disparity is of the order of a few pixels (less than 15 pixels for the image pair with the longest baseline). Fig. 17(c) is the distance map obtained; we used a 9×9 window for SSD computation and used DOG-filtered images as input rather than the original intensity images in order to compensate for the change in sunlight during the data collection session. Pebbles on the road in front of the curb are detectable in the map, and the occlusion edges of the sign board are very sharp. Naturally, range measurements are noisy along the top edge of the curb, which is mostly horizontal. Note that the map is the direct output of the stereo algorithm with no smoothing or postprocessing applied.

During the experiments with this and other scenes, we found that we invariably obtained better results by using relatively short baselines. As seen in Fig. 17(a) and (b), the disparity is typically only 10 to 15 pixels, even for the closest objects in the image pair with the largest baseline. This is somewhat surprising since for precision, we anticipated that we would need much longer baselines, at least for one or two pairs. What is happening here seems to be the following. When the baselines become longer, the effect of photographic and geometric distortions, as well as occlusions, become severe. Use of the shorter baselines generally decreases precision but alleviates these problems, making the SSD functions show more consistent behavior. Yet, since we accumulate multiple observations, sufficient precision is still achievable. This is, in fact, an advantage of the method since it means fewer occluded parts in the final range map, and less computation as well, since the range of SSD computation is shorter. Moreover, after finding the unique minimum position of the SSSD function, we can compute the minimum positions of each individual pair's



(a)



(b)

Fig. 10. Result with multiple baselines $B = 3b$, $6b$, and $9b$: (a) Distance map; (b) isometric plot. Compared with Figs. 8(b) and 9(b), we see that the distance map is less noisy and that gross errors have been removed.

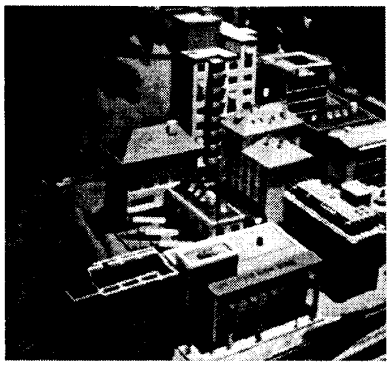
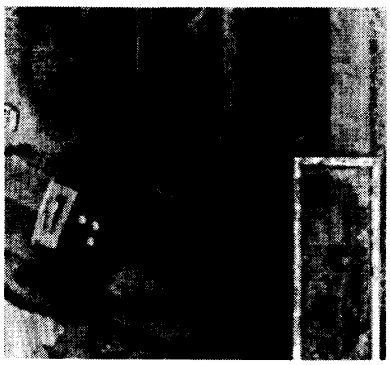


Fig. 11. Oblique view.

SSD functions near the overall minimum, their curvature at their minimums, and finally, their minimum values. We have found some indication that these can be used to evaluate the uncertainty of the correctness of the matching and, further, to classify the situation into occlusion, terminal edges, and specular reflections. We are investigating these issues further [17].



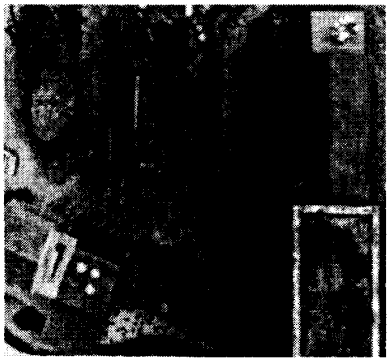


Fig. 12. "Coal mine" data set, long-baseline pair.

IV. Conclusions

In this paper, we have presented a new stereo matching method that uses multiple baseline stereo pairs. This method can overcome the tradeoff between precision and accuracy (avoidance of false matches) in stereo. The method is rather straightforward; we represent the SSD values for individual stereo pairs as a function of the inverse distance and add those functions. The resulting function (the SSSD-in-inverse-distance) exhibits an unambiguous and sharper minimum at the correct matching position. As a result, there is no need for search or sequential estimation procedures.

The key idea of the method is to relate SSD values to the inverse distance rather than the disparity. As an afterthought, this idea is natural. Whereas disparity is a function of the baseline, there is only one true (inverse) distance for each pixel position for all of the stereo pairs. Therefore, there must be a single minimum for the SSD values when they are summed and plotted with respect to the inverse distance. We have shown the advantage of the proposed method in removing ambiguity and improving precision by analytical and experimental results.

Appendix A
SSSD-in-Inverse Distance for Ambiguous Pattern

Proposition: Suppose that there are two and only two repetitions of the same pattern around positions x and $x + a$

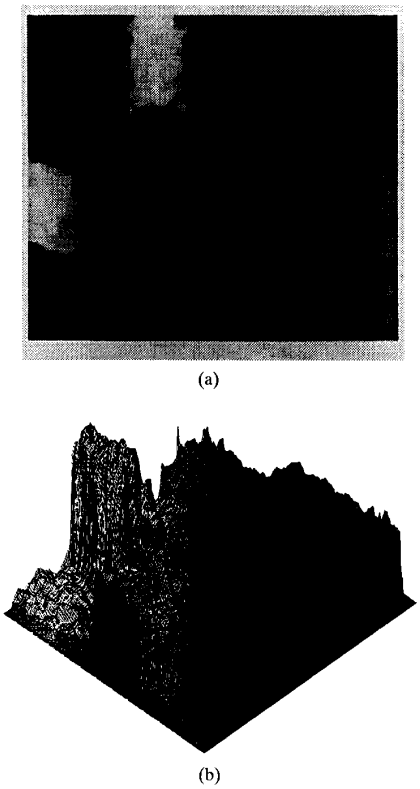


Fig. 13. Result with a short baseline: (a) Distance map; (b) isometric plot of the distance map viewed from the lower left corner.

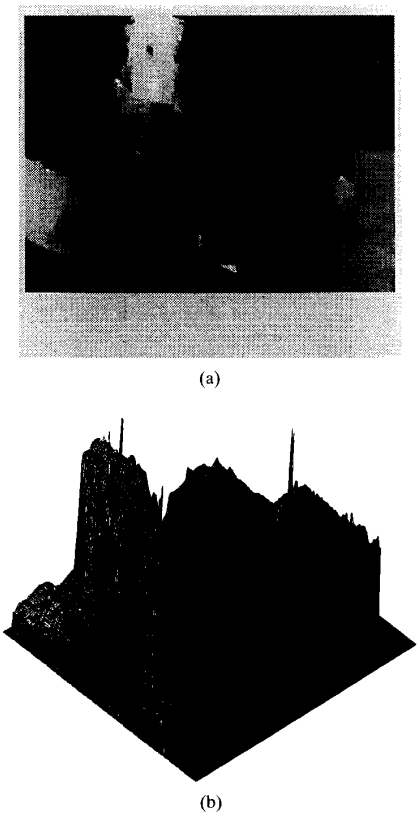


Fig. 14. Result with a long baseline: (a) Distance map; (b) isometric plot.

where $a \neq 0$ is a constant, that is, for $j \in W$

$$f(x+j) = f(\xi+j), \text{ if and only if } \xi = x \text{ or } \xi = x+a. \tag{33}$$

Then, if $B_1 \neq B_2$, for $\forall\zeta$, $\zeta \neq \zeta_r$

$$\begin{aligned} E[e_{\zeta(12)}(x,\zeta)] &= \sum_{j\in W}(f(x+j) \\ &\quad - f(x+B_1F(\zeta-\zeta_r)+j))^2 \\ &\quad + \sum_{j\in W}(f(x+j) \\ &\quad - f(x+B_2F(\zeta-\zeta_r)+j))^2 + 4N_w\sigma_n^2 \\ &> 4N_w\sigma_n^2 = E[e_{\zeta(12)}(x,\zeta_r)]. \end{aligned} \tag{34}$$

Proof: Tentatively suppose that for $\exists\zeta_f$, $\zeta_f \neq \zeta_r$

$$\begin{aligned} &\sum_{j\in W}(f(x+j) - f(x+B_1F(\zeta_f-\zeta_r)+j))^2 \\ &+ \sum_{j\in W}(f(x+j) - f(x+B_2F(\zeta_f-\zeta_r)+j))^2 = 0. \end{aligned} \tag{35}$$

Then, it must be the case that

$$\begin{aligned} f(x+j) &= f(x+a_1+j) \\ \text{and} \quad f(x+j) &= f(x+a_2+j) \end{aligned} \tag{36}$$

for $j \in W$, where

$$a_1 = B_1F(\zeta_f-\zeta_r)$$

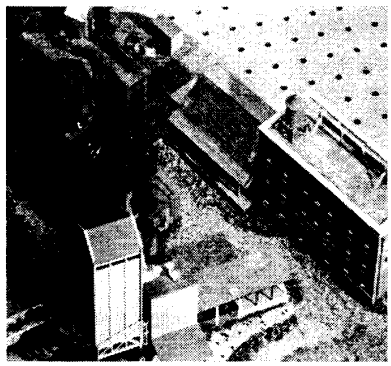


Fig. 15. Oblique view.

$$a_2 = B_2F(\zeta_f-\zeta_r).$$

Since $B_1 \neq B_2$ and $\zeta_r \neq \zeta_f$

$$a_1 \neq a_2. \tag{37}$$

Therefore, we have

$$f(x+j) = f(\xi+j), \quad \text{for } \xi = x,\ x+a_1, \text{ or } x+a_2. \tag{38}$$

Since this contradicts assumption (33), (35) does not hold. Its left-hand side must be positive. Hence, (34) holds.

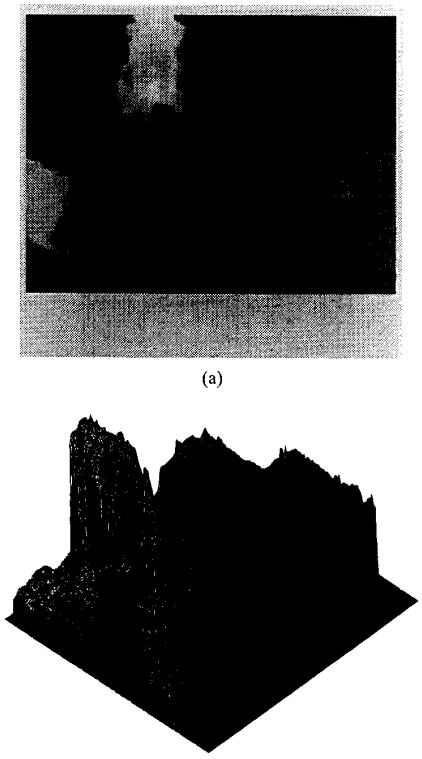


Fig. 16. Multiple baselines: (a) Distance map; (b) isometric plot.

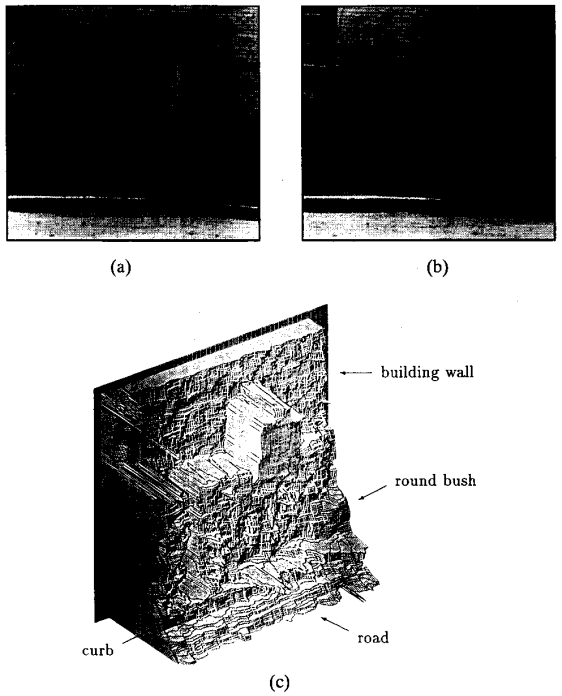


Fig. 17. Result with a real outdoor scene: (a),(b) Long baseline pair of images; (c) isometric plot of the distance map.

APPENDIX B
MULTIPLE-BASELINE STEREO ALGORITHM

We present a complete description of the stereo algorithm using multiple-baseline stereo pairs. The task is, given n stereo pairs, find the ζ that minimizes the SSSD-in-inverse-distance function

$$SSSD(x,\zeta) = \sum_{i=1}^{n} \sum_{j \in W} (f_0(x+j) - f_i(x+B_i F \zeta + j))^2. \quad (39)$$

We will perform this task in two steps: one at pixel resolution by minimum detection and the other at subpixel resolution by iterative estimation.

Minimum of SSSD at Pixel Resolution

For convenience, instead of using the inverse distance, we normalize the disparity values of individual stereo pairs with different baselines to the corresponding values for the largest baseline. Suppose $B_1 < B_2 < \cdots < B_n$. We define the baseline ratio R_i such that

$$R_i = \frac{B_i}{B_n}. \quad (40)$$

Then

$$B_i F \zeta = R_i B_n F \zeta = R_i d_{(n)} \quad (41)$$

where $d_{(n)}$ is the disparity for the stereo pair with baseline B_n. Substituting this into (39)

$$SSSD(x, d_{(n)}) = \sum_{i=1}^{n} \sum_{j \in W} (f_0(x+j) - f_i(x + R_i d_{(n)} + j))^2. \quad (42)$$

We compute the SSSD function for a range of disparity values at the pixel resolution and identify the disparity that gives the minimum. Note that pixel resolution for the image pair with the longest baseline (B_n) requires calculation of SSD values at subpixel resolution for other shorter baseline stereo pairs.

Iterative Estimation at Subpixel Resolution

Once we obtain disparity at pixel resolution for the longest baseline stereo, we improve the disparity estimate to subpixel resolution by an iterative algorithm presented in [12] and [18]. For this iterative estimation, we use only the image pair $f_0(x)$ and $f_n(x)$ with the longest baseline. This is due to a few reasons. First, since the pixel-level estimate was obtained by using the SSSD-in-inverse-distance, the ambiguity has been eliminated, and only improvement of precision is intended at this stage. Second, using only the longest-baseline image pair reduces the computational requirement for SSD calculation by a factor of n and yet does not degrade precision too significantly.

In the experiments shown in Section III, we used the following algorithm for subpixel estimation: Let $d_{0(n)}$ be the initial disparity estimate obtained at pixel resolution. Then, a more precise estimate is computed by calculating the following

two quantities:

$$\Delta\hat{d}_{(n)} = \frac{\sum_{j\in W}(f_0(x+j) - f_n(x+d_{0(n)}+j))f'_n(x+d_{0(n)}+j)}{\sum_{j\in W}(f'_n(x+d_{0(n)}+j))^2} \tag{43}$$

$$\sigma^2_{\Delta d_{(n)}} = \frac{2\sigma_n^2}{\sum_{j\in W}(f'_n(x+d_{0(n)}+j))^2}. \tag{44}$$

The value $\Delta\hat{d}_{(n)}$ is the estimate of the correction of the disparity to further minimize the SSD, and $\sigma^2_{\Delta d_{(n)}}$ is its variance. We iterate this procedure by replacing $d_{0(n)}$ by

$$d_{0(n)} \leftarrow d_{0(n)} + \Delta\hat{d}_{(n)} \tag{45}$$

until the estimate converges or up to a certain maximum number of iterations.

ACKNOWLEDGMENT

The authors would like to thank J. Krumm for his useful comments on this paper. K. Gremban, J. Rehg, and C. Novak have read the manuscript and improved its readability substantially. T. Nakahara performed experiments of the algorithm with outdoor scenes, which produced the result of Fig. 17.

REFERENCES

[1] D. Marr and T. Poggio, "A theory of human stereo vision," in *Proc. Roy. Soc.* (London), 1979, pp. 301–328, vol. B.
[2] W. E. L. Grimson, "Computational experiments with a feature based stereo algorithm," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-7, no. 1, pp. 17–34, Jan. 1985.
[3] S. T. Barnard, "Stochastic stereo matching over scale," *Int. J. Comput. Vision*, pp. 17–32, 1989.
[4] M. J. Hannah, "A system for digital stereo image matching," *Photogram. Eng. Remote Sensing*, vol. 55, no. 12, pp. 1765–1770, Dec. 1989.
[5] J. -S. Chen and G. Medioni, "Parallel multiscale stereo matching using adaptive smoothing," in *ECCV90*, 1990, pp. 99–103.
[6] R. C. Bolles, H. H. Baker, and D. H. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *Int. J. Comput. Vision*, vol. 1, no. 1, 1987.
[7] M. Yamamoto, "The image sequence analysis of three-dimensional dynamic scenes," Tech. Rep. 893, Electrotech. Lab., Agency of Indus. Sci. Technol., Tsukuba, Ibaraki, Japan, May 1988.
[8] L. Matthies, R. Szeliski, and T. Kanade, "Kalman filter-based algorithms for estimating depth from image sequences," *Int. J. Comput. Vision*, vol. 3, pp. 209–236, 1989.
[9] J. Heel, "Dynamic motion vision," in *Proc. DARPA Image Understanding Workshop* (Palo Alto, CA), May 23–26 1989, pp. 702–713.
[10] B. Wilcox, "Telerobotics and Mars rover research at JPL," Lecture at Carnegie Mellon Univ., Oct. 1987.
[11] H. P. Moravec, "Visual mapping by a robot rover," in *Proc. IJCAI*, 1979, pp. 598–600.
[12] L. Matthies and M. Okutomi, "A Bayesian foundation for active stereo vision," in *Proc. SPIE Sensor Fusion II: Human Machine Strategies*, Nov. 1989, pp. 62–74.
[13] M. Yachida, Y. Kitamura, and M. Kimachi, "Trinocular vision: New approach for correspondence problem," in *Proc. ICPR*, 1986, pp. 1041–1044.
[14] V. J. Milenkovic and T. Kanade, "Trinocular vision using photometric and edge orientation constraints," in *Proc. Image Understanding Workshop* (Miami Beach, FL), Dec. 1985, pp. 163–175.
[15] N. Ayache and F. Lustman, "Fast and reliable passive trinocular stereo vision," in *Proc. ICCV*, 1987, pp. 422–426.
[16] R. Y. Tsai, "Multiframe image point matching and 3-d surface reconstraction," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-5, no. 2, Mar. 1983.
[17] T. Kanade and T. Nakahara, "Experimental results of multibaseline stereo," in *IEEE Special Workshop Passive Ranging* (Princeton, NJ), Oct. 1991.
[18] M. Okutomi and T. Kanade, "A locally adaptive window for signal matching," in *Proc. Int. Conf. Comput. Vision*, Dec. 1990; also in *Int. J. Comput. Vision*, vol. 7, no. 2, pp. 143-162, 1992.

Masatoshi Okutomi (M'91) received the B.Eng. degree in mathematical engineering and information physics from the University of Tokyo, Tokyo, Japan, in 1981 and the M.Eng. degree in control engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1983.

He joined the Canon Research Center, Tokyo, Japan, in 1983. From 1987 to 1990, he was a visiting research scientist with the School of Computer Science at Carnegie Mellon University, Pittsburgh. Currently, he is a senior researcher with the Information Systems Research Center, Canon, Inc., Kawasaki, Japan. He has published research papers in the field of computer vision and robotics and has worked on industrial and medical applications of image processing and pattern recognition and the development of image processing systems.

Takeo Kanade (F'92) received the Ph.D. degree in electrical engineering from Kyoto University, Kyoto, Japan, in 1974.

After holding a faculty position at the Department of Information Science at Kyoto University, he joined Carnegie Mellon University, Pittsburgh, PA, in 1980, where he is currently a Professor of Computer Science and Director of the Robotics Institute. For education in robotics, he established the Robotics Ph.D. Program at Carnegie Mellon and is currently Chairman of the program.

Dr. Kanade is a Founding Fellow of the American Association of Artificial Intelligence, the founding editor of the *International Journal of Computer Vision*, and an Administrative Commitee member of the IEEE Robotics and Automation Society. He has received several awards including the Marr Prize Paper Award in 1990, and his paper was selected as one of the most influential papers that appeared in the *Artificial Intelligence* journal. He has served many government, industry, and university advisory panels, including the NASA Advanced Technology Advisory Committee and the Canadian Institute for Advanced Research.

Single View Metrology

A. Criminisi, I. Reid and A. Zisserman *
Department of Engineering Science
University of Oxford
Oxford, UK, OX1 3PJ
{criminis,ian,az}@robots.ox.ac.uk

Abstract

We describe how 3D affine measurements may be computed from a single perspective view of a scene given only minimal geometric information determined from the image. This minimal information is typically the vanishing line of a reference plane, and a vanishing point for a direction not parallel to the plane. It is shown that affine scene structure may then be determined from the image, without knowledge of the camera's internal calibration (e.g. focal length), nor of the explicit relation between camera and world (pose).

In particular, we show how to (i) compute the distance between planes parallel to the reference plane (up to a common scale factor); (ii) compute area and length ratios on any plane parallel to the reference plane; (iii) determine the camera's (viewer's) location. Simple geometric derivations are given for these results. We also develop an algebraic representation which unifies the three types of measurement and, amongst other advantages, permits a first order error propagation analysis to be performed, associating an uncertainty with each measurement.

We demonstrate the technique for a variety of applications, including height measurements in forensic images and 3D graphical modelling from single images.

1. Introduction

In this paper we describe how aspects of the affine 3D geometry of a scene may be measured from a single perspective image. We will concentrate on scenes containing planes and parallel lines, although the methods are not so restricted. The methods we develop extend and generalize previous results on single view metrology [8, 9, 13, 14].

It is assumed that images are obtained by perspective projection. In addition, we assume that the vanishing line of a *reference plane* in the scene may be determined from the image, together with a vanishing point for another *reference direction* (not parallel to the plane). We are then concerned with three canonical types of measurement: (i) measurements of the distance *between* any of the planes which are parallel to the reference plane; (ii) measurements *on* these planes (and comparison of these measurements to those obtained on any plane); and (iii) determining the camera's position in terms of the reference plane and direction. The measurement methods developed here are independent of the camera's internal parameters: focal length, aspect ratio, principal point, skew.

The ideas in this paper can be seen as reversing the rules for drawing perspective images given by Leon Battista Alberti [1] in his treatise on perspective (1435). These are the rules followed by the Italian Renaissance painters of the 15th century, and indeed we demonstrate the correctness of their mastery of perspective by analysing a painting by Piero della Francesca.

We begin in section 2 by giving geometric interpretations for the key scene features, and then give simple geometric derivations of how, in principle, three dimensional affine information may be extracted from the image. In section 3 we introduce an algebraic representation of the problem and show that this representation unifies the three canonical measurement types, leading to simple formulae in each case. In section 4 we describe how errors in image measurements propagate to errors in the 3D measurements, and hence we are able to compute confidence intervals on the 3D measurements, i.e. a quantitative assessment of accuracy. The work has a variety of applications, and we demonstrate two important ones: forensic measurement and virtual modelling in section 5.

2. Geometry

The camera model employed here is central projection. We assume that the vanishing line of a reference plane in the scene may be computed from image measurements, together with a vanishing point for another direction (not parallel to the plane). This information is generally easily obtainable from images of structured scenes [3, 11, 12]. Ef-

*The authors would like to thank Andrew Fitzgibbon for assistance with the TargetJr libraries, and David Liebowitz and Luc van Gool for discussions. This work was supported by the EU Esprit Project IMPROOFS.

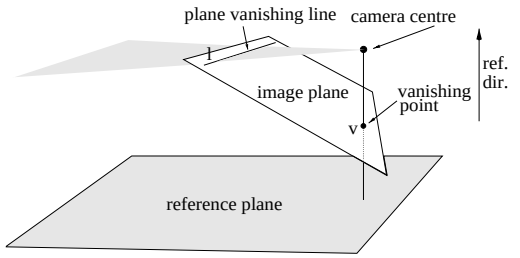


Figure 1: Basic geometry: The plane's vanishing line $\mathbf{l}$ is the intersection of the image plane with a plane parallel to the reference plane and passing through the camera centre. The vanishing point $\mathbf{v}$ is the intersection of the image plane with a line parallel to the reference direction through the camera centre.

fects such as radial distortion (often arising in slightly wide-angle lenses typically used in security cameras) which corrupt the central projection model can generally be removed [6], and are therefore not detrimental to our methods (see, for example, figure 9).

Although the schematic figures show the camera centre at a finite location, the results we derive apply also to the case of a camera centre at infinity, i.e. where the images are obtained by parallel projection. The basic geometry of the plane's vanishing line and the vanishing point are illustrated in figure 1. The vanishing line $\mathbf{l}$ of the reference plane is the projection of the line at infinity of the reference plane into the image. The vanishing point $\mathbf{v}$ is the image of the point at infinity in the reference direction. Note that the reference direction need not be vertical, although for clarity we will often refer to the vanishing point as the "vertical" vanishing point. The vanishing point is then the image of the vertical "footprint" of the camera centre on the reference plane.

It can be seen (for example, by inspection of figure 1) that the vanishing line partitions all points in scene space. Any scene point which projects onto the vanishing line is at the same distance from the plane as the camera centre; if it lies "above" the line it is further from the plane, and if "below" the vanishing line, then it is closer to the plane than the camera centre.

Two points on separate planes (parallel to the reference plane) *correspond* if the line joining them is parallel to the reference direction; hence the image of each point and the vanishing point are collinear. For example, if the direction is vertical, then the top of an upright person's head and the sole of his/her foot correspond.

2.1. Measurements between parallel planes

We wish to measure the distance between two parallel planes, specified by the image points $\mathbf{t}$ and $\mathbf{b}$, in the reference direction. Figure 2 shows the geometry, with points $\mathbf{t}$ and $\mathbf{b}$ in correspondence. The four points marked on the figure define a cross-ratio. The vanishing point is the image of a point at infinity in the scene [15]. In the image the value

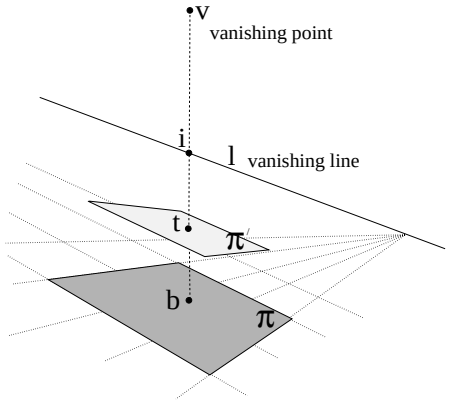


Figure 2: Cross ratio: The point $\mathbf{b}$ on the plane π corresponds to the point $\mathbf{t}$ on the plane π'. They are aligned with the vanishing point $\mathbf{v}$. The four points $\mathbf{v}$, $\mathbf{t}$, $\mathbf{b}$ and the intersection $\mathbf{i}$ of the line joining them with the vanishing line define a cross-ratio. The value of the cross-ratio determines a ratio of distances between planes in the world, see text.

of the cross-ratio provides an affine length ratio. In fact we obtain the ratio of the distance between the planes containing $\mathbf{t}$ and $\mathbf{b}$, to the camera's distance from the plane π (or π' depending on the ordering of the cross-ratio). The absolute distance can be obtained from this distance ratio once the camera's distance from π is specified. However it is usually more practical to determine the distance via a second measurement in the image, that of a known reference length.

Furthermore, since the vanishing line is the imaged axis of the pencil of planes parallel to the reference plane, the knowledge of the distance between *any* pair of the planes is sufficient to determine the absolute distance between another two of the planes.

Example. Figure 3 shows that a person's height may be computed from an image given a vertical reference height elsewhere in the scene. The formula used to compute this result is given in section 3.1.

2.2. Measurements on parallel planes

If the reference plane π is affine calibrated (we know its vanishing line) then from image measurements we can compute: (i) ratios of lengths of parallel line segments on the plane; (ii) ratios of areas on the plane. Moreover the vanishing line is shared by the pencil of planes parallel to the reference plane, hence affine measurements may be obtained for any other plane in the pencil. However, although affine measurements, such as an area ratio, may be made *on* a particular plane, the areas of regions lying on two parallel planes cannot be compared directly. If the region is parallel projected in the scene from one plane onto the other, affine measurements can then be made from the image since both regions are now on the same plane, and parallel projection between parallel planes does not alter affine properties.

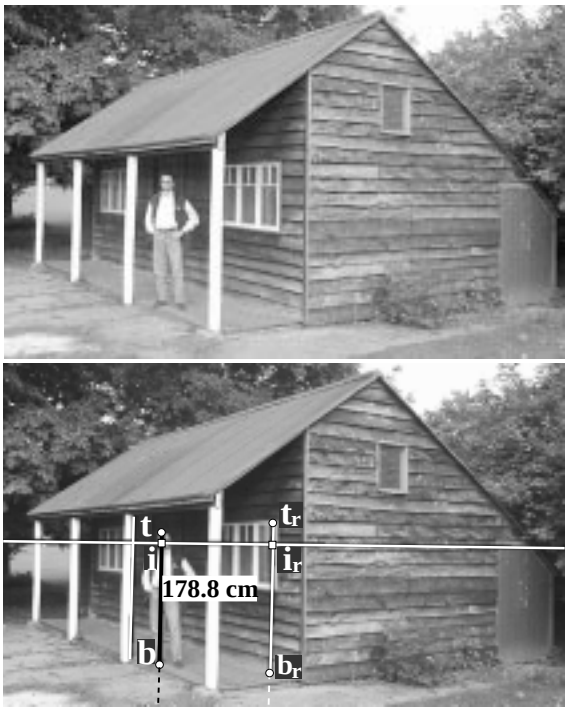


Figure 3: Measuring the height of a person: (top) original image; (bottom) the height of the person is computed from the image as 178.8cm (the true height is 180cm, but note that the person is leaning down a bit on his right foot). The vanishing line is shown in white and the reference height is the segment $(\mathbf{t}_r, \mathbf{b}_r)$. The vertical vanishing point is not shown since it lies well below the image. $\mathbf{t}$ is the top of the head and $\mathbf{b}$ is the base of the feet of the person while $\mathbf{i}$ is the intersection with the vanishing line.

A map in the world between parallel planes induces a map in the image between images of points on the two planes. This image map is a *planar homology* [15], which is a plane projective transformation with five degrees of freedom, having a line of fixed points, called the *axis* and a distinct fixed point not on the axis known as the *vertex*. Planar homologies arise naturally in an image when two planes related by a perspectivity in 3-space are imaged [16]. The geometry is illustrated in figure 4.

In our case the vanishing line of the plane, and the vertical vanishing point, are, respectively, the axis and vertex of the homology which relates a pair of planes in the pencil. This line and point specify four of the five degrees of freedom of the homology. The remaining degree of freedom of the homology is uniquely determined from any pair of image points which correspond between the planes (points $\mathbf{b}$ and $\mathbf{t}$ in figure 4).

This means that we can compare measurements made on two separate planes by mapping between the planes in the reference direction via the homology. In particular we may compute (i) the ratio between two parallel lengths, one length on each plane; (ii) the ratio between two areas, one area on each plane. In fact we can simply transfer all points from one plane to the reference plane using the homology and then, since the reference plane's vanishing line is known, make affine measurements in the plane, e.g. parallel length or area ratios.

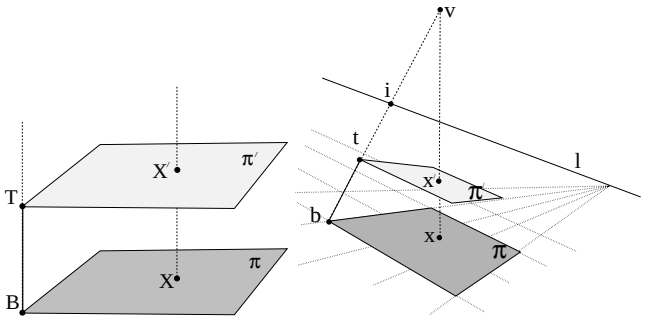


Figure 4: Homology mapping between parallel planes: (left) A point $\mathbf{X}$ on plane $\boldsymbol{\pi}$ is mapped into the point $\mathbf{X}'$ on $\boldsymbol{\pi}'$ by a parallel projection. (right) In the image the mapping between the images of the two planes is a homology, with $\mathbf{v}$ the vertex and $\mathbf{l}$ the axis. The correspondence $\mathbf{b} \rightarrow \mathbf{t}$ fixes the remaining degree of freedom of the homology from the cross-ratio of the four points: $\mathbf{v}$, $\mathbf{i}$, $\mathbf{t}$ and $\mathbf{b}$.

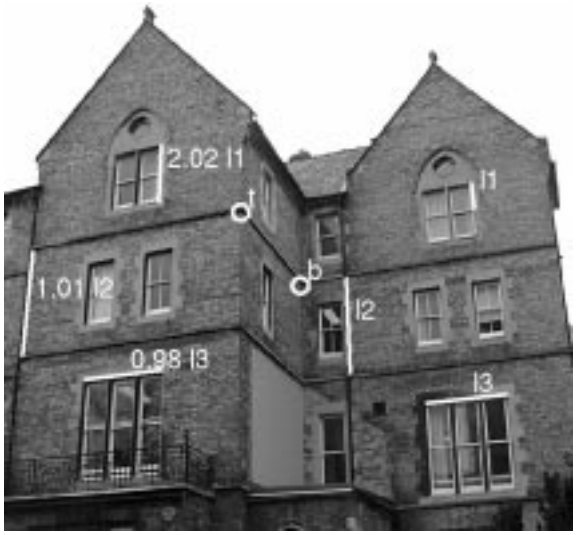


Figure 5: Measuring the ratio of lengths of parallel line segments lying on two parallel scene planes: The points $\mathbf{t}$ and $\mathbf{b}$ (together with the plane vanishing line and the vanishing point) define the homology between the two planes on the facade of the building.

Example. Figure 5 shows that given the reference vanishing line and vanishing point, and a point correspondence (in the reference direction) on each of two parallel planes, then the ratio of lengths of parallel line segments may be computed from the image. The formula used to compute this result is given in section 3.2.

2.3. Determining the camera position

In section 2.1, we computed distances between planes as a ratio relative to the camera's distance from the reference plane. Conversely, we may compute the camera's distance from a particular plane knowing a single reference distance.

Furthermore, by considering figure 1 it is seen that the location of the camera relative to the reference plane is the back-projection of the vanishing point onto the reference plane. This back-projection is accomplished by a homography which maps the image to the reference plane (and vice-versa). Although the choice of coordinate frame in the world is somewhat arbitrary, fixing this frame immediately defines the homography uniquely and hence the camera position.

We show an example in figure 12, where the location of the camera centre has been determined, and superimposed into a virtual view of the scene.

3. Algebraic Representation

The measurements described in the previous section are computed in terms of cross-ratios. In this section we develop a uniform algebraic approach to the problem which has a number of advantages over direct geometric construction: first, it avoids potential problems with ordering for the cross-ratio; second, it enables us to deal with both minimal or over-constrained configurations uniformly; third, we unify the different types of measurement within one representation; and fourth, in section 4 we use this algebraic representation to develop an uncertainty analysis for measurements.

To begin we define an affine coordinate system XYZ in space. Let the origin of the coordinate frame lie on the reference plane, with the X and Y-axes spanning the plane. The Z-axis is the reference direction, which is thus any direction not parallel to the plane. The image coordinate system is the usual xy affine image frame, and a point $\mathbf{X}$ in space is projected to the image point $\mathbf{x}$ via a 3×4 projection matrix $\mathtt{P}$ as:

$$\mathbf{x} = \mathtt{P}\mathbf{X} = \begin{bmatrix}\mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4\end{bmatrix}\mathbf{X}$$

where $\mathbf{x}$ and $\mathbf{X}$ are homogeneous vectors in the form: $\mathbf{x} = (x, y, w)^\top$, $\mathbf{X} = (X, Y, Z, W)^\top$, and '=' means equality up to scale.

If we denote the vanishing points for the X, Y and Z directions as (respectively) $\mathbf{v}_X$, $\mathbf{v}_Y$ and $\mathbf{v}$, then it is clear by inspectionthat the first three columns of $\mathtt{P}$ are the vanishing points; $\mathbf{v}_X = \mathbf{p}_1$, $\mathbf{v}_Y = \mathbf{p}_2$ and $\mathbf{v} = \mathbf{p}_3$, and that the final column of $\mathtt{P}$ is the projection of the origin of the world coordinate system, $\mathbf{o} = \mathbf{p}_4$. Since our choice of coordinate frame has the X and Y axes in the reference plane $\mathbf{p}_1 = \mathbf{v}_X$ and $\mathbf{p}_2 = \mathbf{v}_Y$ are two distinct points on the vanishing line. Choosing these points fixes the X and Y affine coordinate axes. We denote the vanishing line by $\mathbf{l}$, and to emphasise that the vanishing points $\mathbf{v}_X$ and $\mathbf{v}_Y$ lie on it, we denote them by $\mathbf{l}_1^\perp$, $\mathbf{l}_2^\perp$, with $\mathbf{l}_i^\perp \cdot \mathbf{l} = 0$.

Columns 1, 2 and 4 of the projection matrix are the three columns of the reference plane to image homography. This homography must have rank three, otherwise the reference plane to image map is degenerate. Consequently, the final column (the origin of the coordinate system) must not lie on the vanishing line, since if it does then all three columns are points on the vanishing line, and thus are not linearly independent. Hence we set it to be $\mathbf{o} = \mathbf{p}_4 = \mathbf{l}/||\mathbf{l}|| = \hat{\mathbf{l}}$.

Therefore the final parametrization of the projection matrix $\mathtt{P}$ is:

$$\mathtt{P} = \begin{bmatrix}\mathbf{l}_1^\perp & \mathbf{l}_2^\perp & \alpha\mathbf{v} & \hat{\mathbf{l}}\end{bmatrix} \tag{1}$$

where α is a scale factor, which has an important rôle to play in the remainder of the paper.

In the following sections we show how to compute various measurements from this projection matrix. Measurements between planes are independent of the first two (under-determined) columns of $\mathtt{P}$. For these measurements the only unknown quantity is α. Coordinate measurements within the planes depend on the first two and the fourth columns of $\mathtt{P}$. They define an affine coordinate frame within the plane. Affine measurements (e.g. area ratios), though, are independent of the actual coordinate frame and depend only on the fourth column of $\mathtt{P}$. If any metric information on the plane is known, we may impose constraints on the choice of the frame.

3.1. Measurements between parallel planes

We wish to measure the distance between scene planes specified by a base point $\mathbf{B}$ on the reference plane and top point $\mathbf{T}$ in the scene. These points may be chosen as respectively $(X, Y, 0)$ and (X, Y, Z), and their images are $\mathbf{b}$ and $\mathbf{t}$. If $\mathtt{P}$ is the projection matrix then the image coordinates are

$$\mathbf{b} = \mathtt{P}\begin{bmatrix}X\\Y\\0\\1\end{bmatrix}, \qquad \mathbf{t} = \mathtt{P}\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix}$$

The equations above can be rewritten as

$$\mathbf{b} = \rho(X\mathbf{p}_1 + Y\mathbf{p}_2 + \mathbf{p}_4) \tag{2}$$

$$\mathbf{t} = \mu(X\mathbf{p}_1 + Y\mathbf{p}_2 + Z\mathbf{p}_3 + \mathbf{p}_4) \tag{3}$$

where ρ and μ are unknown scale factors, and $\mathbf{p}_i$ is the ith column of the $\mathtt{P}$ matrix.

Taking the scalar product of (2) with $\hat{\mathbf{l}}$ yields $\rho = \hat{\mathbf{l}} \cdot \mathbf{b}$, and combining this with the third column of (1) and (3) we obtain

$$\alpha Z = \frac{-||\mathbf{b} \times \mathbf{t}||}{(\hat{\mathbf{l}} \cdot \mathbf{b})||\mathbf{v} \times \mathbf{t}||} \tag{4}$$

Since αZ scales linearly with α we have obtained affine structure. If α is known, then we immediately obtain a metric value for Z. Conversely, if Z is known (i.e. it is a reference distance) then we have a means of computing α, and hence removing the affine ambiguity.

Figure 6: Measuring heights using parallel lines: Given the vertical vanishing point, the vanishing line for the ground plane and a reference height, the distance of the top of the window on the right wall from the ground plane is measured from the distance between the two horizontal lines shown, one defined by the top edge of the window, and the other on the ground plane.

Example. In figure 6 heights from the ground plane are measured between two parallel lines, one off the plane (top) and one on the plane (base). In fact, thanks to the plane vanishing line, given one line parallel to the reference plane it is easy to compute the family of parallel lines. Computing the distance between them is a straightforward application of (4).

3.2. Measurements on parallel planes

The projection matrix $\mathtt{P}$ from the world to the image is defined above with respect to a coordinate frame on the reference plane. In this section we determine the projection matrix $\mathtt{P}'$ referred to the parallel plane π' and we show how the homology between the two planes can be derived directly from the two projection matrices.

Suppose the world coordinate system is translated from the plane π onto the plane π' along the reference direction, then it is easy to show that we can parametrize the new projection matrix $\mathtt{P}'$ as:

$$\mathtt{P}' = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \alpha\mathbf{v} & \alpha Z\mathbf{v} + \hat{\mathbf{l}} \end{bmatrix}$$

where Z is the distance between the planes. Note that if $Z = 0$ then $\mathtt{P}' = \mathtt{P}$ correctly.

The plane to image homographies can be extracted from the projection matrices ignoring the third column, to give:

$$\mathtt{H} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \hat{\mathbf{l}} \end{bmatrix}, \ \mathtt{H}' = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \alpha Z\mathbf{v} + \hat{\mathbf{l}} \end{bmatrix}$$

Then $\tilde{\mathtt{H}} = \mathtt{H}'\mathtt{H}^{-1}$ maps image points on the plane π onto points on the plane π' and so defines the homology.

A short computation gives the homology matrix $\tilde{\mathtt{H}}$ as:

$$\tilde{\mathtt{H}} = \mathtt{I} + \alpha Z\mathbf{v}\hat{\mathbf{l}}^\top \tag{5}$$

Given the homology between two planes in the pencil we can transfer all points from one plane to the other and make affine measurements in the plane (see fig 5 and fig 7).

Figure 7: Measuring ratios of areas on separate planes: The image points $\mathbf{t}$ and $\mathbf{b}$ together with the vanishing line of the two parallel planes and the vanishing point for the orthogonal direction define the homology between the planes. The ratio between the area of the window on the left plane and that of the window on the right plane is computed.

3.3. Determining camera position

Suppose the camera centre is $\mathbf{C} = (X_c, Y_c, Z_c, W_c)^\top$ in affine coordinates (see figure 1). Then since $\mathtt{P}\mathbf{C} = \mathbf{0}$ we have

$$\mathtt{P}\mathbf{C} = \mathbf{l}_1^\perp X_c + \mathbf{l}_2^\perp Y_c + \alpha\mathbf{v}Z_c + \hat{\mathbf{l}}W_c = \mathbf{0} \tag{6}$$

The solution to this set of equations is given (using Cramer's rule) by

$$\begin{array}{llll} X_c &= -\det\begin{bmatrix}\mathbf{l}_2^\perp & \mathbf{v} & \hat{\mathbf{l}}\end{bmatrix}, & Y_c &= \det\begin{bmatrix}\mathbf{l}_1^\perp & \mathbf{v} & \hat{\mathbf{l}}\end{bmatrix} \\ \alpha Z_c &= -\det\begin{bmatrix}\mathbf{l}_1^\perp & \mathbf{l}_2^\perp & \hat{\mathbf{l}}\end{bmatrix}, & W_c &= \det\begin{bmatrix}\mathbf{l}_1^\perp & \mathbf{l}_2^\perp & \mathbf{v}\end{bmatrix} \end{array} \tag{7}$$

Note that once again we obtain structure off the plane up to the affine scale factor α. As before, we may upgrade the distance to metric with knowledge of α, or use knowledge of camera height to compute α and upgrade the affine structure.

Note that affine viewing conditions (where the camera centre is at infinity) present no problem to the expressions in (7), since in this case we have $\hat{\mathbf{l}} = \begin{bmatrix}0 & 0 & *\end{bmatrix}^\top$ and $\mathbf{v} = \begin{bmatrix}* & * & 0\end{bmatrix}^\top$. Hence $W_c = 0$ so we obtain a camera centre on the plane at infinity, as we would expect. This point on π_∞ represents the viewing direction for the parallel projection.

If the viewpoint is finite (i.e. not affine viewing conditions) then the formula for αZ_c may be developed further by taking the scalar product of both sides of (6) with the vanishing line $\hat{\mathbf{l}}$. The result is: $\alpha Z_c = -(\hat{\mathbf{l}} \cdot \mathbf{v})^{-1}$.

4. Uncertainty Analysis

Feature detection and extraction – whether manual or automatic (e.g. using an edge detector) – can only be achieved

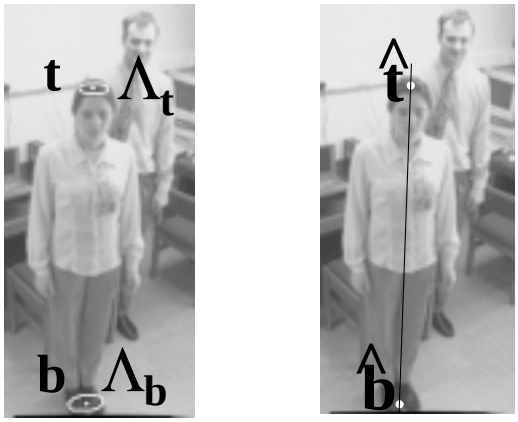


Figure 8: Maximum likelihood estimation of the top and base points (closeup of fig. 9): (left) The top and base uncertainty ellipses, respectively $\Lambda_{\mathbf{t}}$ and $\Lambda_{\mathbf{b}}$, are shown. These ellipses are specified by the user, and indicate a confidence region for localizing the points. (right) MLE top and base points $\hat{\mathbf{t}}$ and $\hat{\mathbf{b}}$ are aligned with the vertical vanishing point (outside the image).

to a finite accuracy. Any features extracted from an image, therefore, are subject to errors. In this section we consider how these errors propagate through the measurement formulae in order to quantify the uncertainty on the final measurements.

When making measurements between planes, uncertainty arises from the uncertainty in P, and from the uncertain image locations of the top and base points $\mathbf{t}$ and $\mathbf{b}$. The uncertainty in P depends on the location of the vanishing line, the location of the vanishing point, and on α, the affine scale factor. Since only the final two columns contribute, we model the uncertainty in P as a 6×6 homogeneous covariance matrix, $\Lambda_{\mathtt{P}}$. Since the two columns have only five degrees of freedom (two for $\mathbf{v}$, two for $\mathbf{l}$ and one for α), the covariance matrix is singular, with rank five. Details of its computation are given in [4] and are omitted here for brevity.

Likewise, the uncertainty in the top and base points (resulting largely from the finite accuracy with which these features may be located in the image) is modelled by covariance matrices $\Lambda_{\mathbf{b}}$ and $\Lambda_{\mathbf{t}}$. Since in the error-free case, these points must be aligned with the vertical vanishing point we can determine maximum likelihood estimates of their true locations ($\hat{\mathbf{t}}$ and $\hat{\mathbf{b}}$) by minimising the objective

$$(\mathbf{b}_2 - \hat{\mathbf{b}}_2)^\top \mathbf{\Lambda}_{\mathbf{b}_2}^{-1} (\mathbf{b}_2 - \hat{\mathbf{b}}_2) + (\mathbf{t}_2 - \hat{\mathbf{t}}_2)^\top \mathbf{\Lambda}_{\mathbf{t}_2}^{-1} (\mathbf{t}_2 - \hat{\mathbf{t}}_2)$$

(which is the sum of the Mahalanobis distances between the input points and the ML estimates, the subscript 2 indicates inhomogeneous 2-vectors) *subject to the alignment constraint* $\mathbf{v} \cdot (\hat{\mathbf{t}} \times \hat{\mathbf{b}}) = 0$. Using standard techniques [7] we obtain a first order approximation to the 4×4, rank three covariance of the parameters $\hat{\mathbf{z}} = (\ \hat{\mathbf{t}}_2^\top \quad \hat{\mathbf{b}}_2^\top\ )^\top$. Figure 8 illustrates the idea.

Figure 9: Uncertainty analysis on height measurements: The image shown was captured from a cheap security type camera which exhibited radial distortion. This has been corrected and the height of the man estimated (measurements are in cm). (left) The height of the man and the associated uncertainty are computed as 190.6cm (*c.f.* ground truth value 190cm). The vanishing line for the ground plane is shown in white at the top of the image. When one reference height is used the uncertainty (3-sigma) is ± 4.1cm, while (right) it reduces to ± 2.9cm as two more reference heights are introduced (the filing cabinet and the table on the left).

Now, assuming the statistical independence of $\hat{\mathbf{z}}$ and P we obtain a first order approximation for the variance of the distance measurement:

$$\sigma_h^2 = \boldsymbol{\nabla}_h \begin{pmatrix} \mathbf{\Lambda}_{\hat{\mathbf{z}}} & 0 \\ 0 & \mathbf{\Lambda}_{\mathtt{P}} \end{pmatrix} \boldsymbol{\nabla}_h{}^\top \tag{8}$$

where $\boldsymbol{\nabla}_h$ is the 1×10 Jacobian matrix of the function which maps the projection matrix and top and base points to a distance between them (4). The validity of all approximation has been tested by Monte Carlo simulations and by a number of measurements on real images where the ground truth was known.

Example. An image obtained from a poor quality security camera is shown in figure 9. It has been corrected for radial distortion using the method described in [6], and the floor taken as the reference plane. Vertical and horizontal lines are used to compute the P matrix of the scene. One reference height is used to obtain the affine scale factor α from (4), so other measurements in the same direction are metric.

The computed height of the man and an associated 3-standard deviation uncertainty are displayed in the figure. The height obtained differs by only 6mm from the known true value. As the number of reference distances is increased, so the uncertainty on P (in fact just on α) decreases, resulting in a decrease in uncertainty of the measured height, as theoretically expected.

5. Applications

5.1. Forensic science

A common requirement in surveillance images is to obtain measurements from the scene, such as the height of a felon. Although, the felon has usually departed the scene, reference lengths can be measured from fixtures such as tables and windows.

Figure 10: Measuring the height of a person in an outdoor scene: The ground plane is the reference plane, and its vanishing line is computed from the slabs on the floor. The vertical vanishing point is computed from the edges of the phone box, whose height is known and used as reference. The veridical height is 187cm, but note that the person is leaning slightly on his right foot.

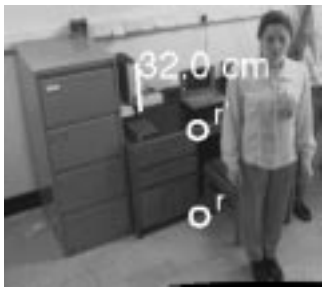


Figure 11: Measuring heights of objects on separate planes: Using the homology between the ground plane (initial reference) and the plane of the table, we can determine the height of the file on the table.

In figure 10 the edges of the paving stones on the floor are used to compute the vanishing line of the ground plane; the edges of the phonebox to compute the vertical vanishing point; and the height of the phone box provides the metric calibration in the vertical direction. The height of the person is then computed using (4).

Figure 11 shows an example where the homology is used to project points between planes so that a vertical distance may be measured given the distance between a plane and the reference plane.

5.2. Virtual modelling

In figure 12 we show an example of complete 3D reconstruction of a scene. Two sets of horizontal edges are used to compute the vanishing line for the ground plane, and vertical edges used to compute the vertical vanishing point. Four points with known Euclidean coordinates determine the metric calibration of the ground plane and thus for the pencil of horizontal planes which share the vanishing line. The distance of the top of the window to the ground, and the height of one of the pillars are used as reference

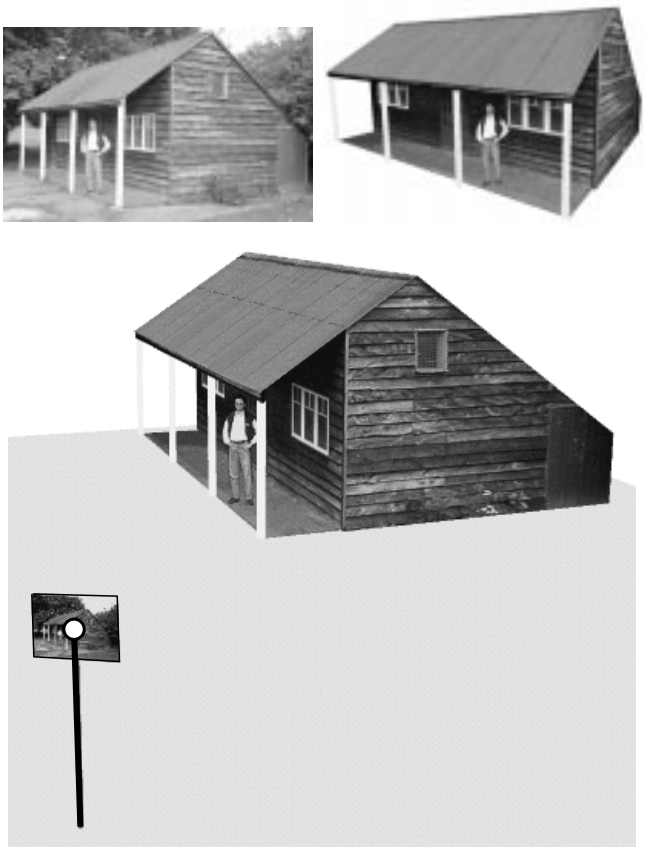


Figure 12: Complete 3D reconstruction of a real scene: (left) original image; (right) a view of the reconstructed 3D model; (bottom) A view of the reconstructed 3D model which shows the position of the camera centre (plane location X,Y and height) with respect to the scene.

lengths. The position of the camera centre is also estimated and is shown in the figure.

5.3. Modelling paintings

Figure 13 shows a masterpiece of Italian Renaissance painting, “La Flagellazione di Cristo” by Piero della Francesca (1416 - 1492). The painting faithfully follows the geometric rules of perspective, and therefore we can apply the methods developed here to obtain a correct 3D reconstruction of the scene.

Unlike other techniques [8] whose main aim is to create convincing new views of the painting regardless of the correctness of the 3D geometry, here we reconstruct a geometrically correct 3D model of the viewed scene.

In the painting analyzed here, the ground plane is chosen as reference and its vanishing line can be computed from the several parallel lines on it. The vertical vanishing point follows from the vertical lines and consequently the relative heights of people and columns can be computed. Furthermore the ground plane can be rectified from the square floor patterns and therefore the position on the ground of each

vertical object estimated [5, 10]. The measurements, up to an overall scale factor, are used to compute a three dimensional VRML model of the scene. Two different views of the model are shown in figure 13.

6. Summary and Conclusions

We have explored how the affine structure of 3-space may be partially recovered from perspective images in terms of a set of planes parallel to a reference plane and a reference direction not parallel to the reference plane. More generally, affine 3 space may be represented entirely by sets of parallel planes and directions [2]. We are currently investigating how this full geometry is best represented and computed from a single perspective image.

References

[1] L. B. Alberti. *De Pictura*. Laterza, 1980.

[2] M. Berger. *Geometry II*. Springer-Verlag, 1987.

[3] R. T. Collins and R. S. Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *Proc. ICCV*, pages 400–403, Dec 1990.

[4] A. Criminisi, I. Reid, and A. Zisserman. Computing 3D euclidean distance from a single view. Technical Report OUEL 2158/98, Dept. Eng. Science, University of Oxford, 1998.

[5] A. Criminisi, I. Reid, and A. Zisserman. A plane measuring device. *Image and Vision Computing*, 17(8):625–634, 1999.

[6] F. Devernay and O. Faugeras. Automatic calibration and removal of distortion from scenes of structured environments. In *SPIE*, volume 2567, San Diego, CA, Jul 1995.

[7] O. Faugeras. *Three-Dimensional Computer Vision: a Geometric Viewpoint*. MIT Press, 1993.

[8] Y. Horry, K. Anjyo, and K. Arai. Tour into the picture: Using a spidery mesh interface to make animation from a single image. In *Proc. ACM SIGGRAPH*, pages 225–232, 1997.

[9] T. Kim, Y. Seo, and K. Hong. Physics-based 3D position analysis of a soccer ball from monocular image sequences. *Proc. ICCV*, pages 721 – 726, 1998.

[10] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. In *Proc. EuroGraphics*, Sep 1999.

[11] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Proc. CVPR*, pages 482–488, Jun 1998.

[12] G. F. McLean and D. Kotturi. Vanishing point detection by line clustering. *IEEE T-PAMI*, 17(11):1090–1095, 1995.

[13] M. Proesmans, T. Tuytelaars, and L. Van Gool. Monocular image measurements. Technical Report Improofs-M12T21/1/P, K.U.Leuven, 1998.

[14] I. Reid and A. Zisserman. Goal-directed video metrology. In R. Cipolla and B. Buxton, editors, *Proc. ECCV*, volume II, pages 647–658. Springer, Apr 1996.

[15] C. E. Springer. *Geometry and Analysis of Projective Spaces*. Freeman, 1964.

[16] L. Van Gool, M. Proesmans, and A. Zisserman. Planar homologies as a basis for grouping and recognition. *Image and Vision Computing*, 16:21–26, Jan 1998.

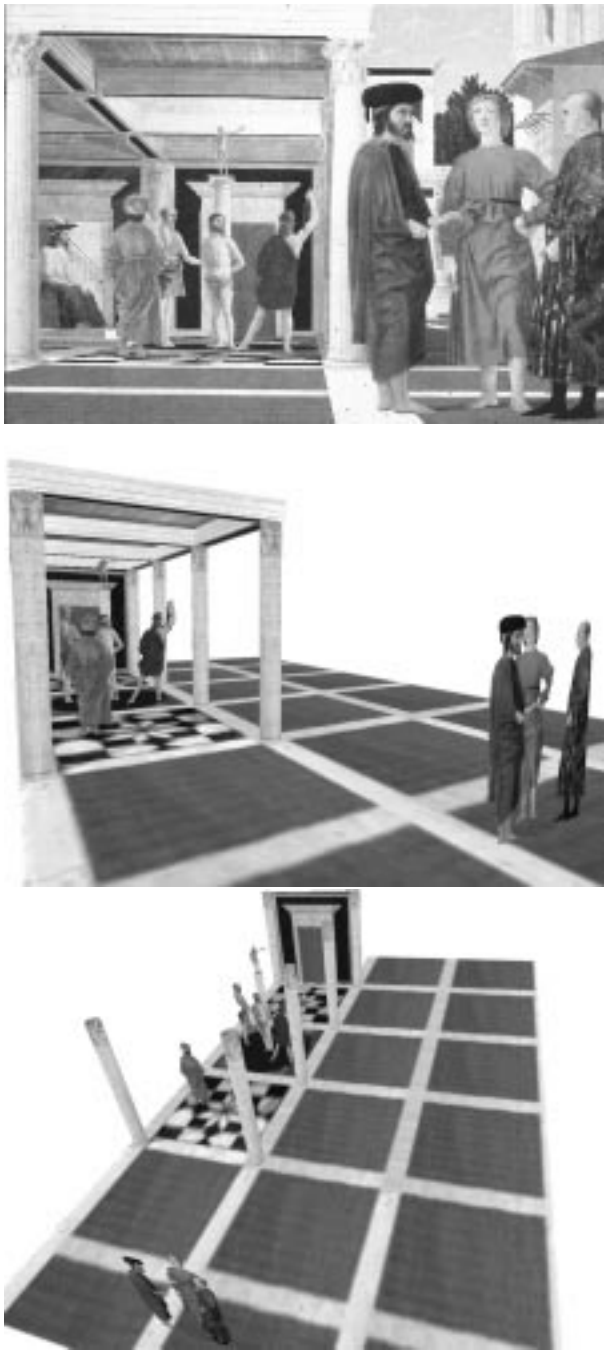


Figure 13: Complete 3D reconstruction of a Renaissance painting: (top) *La Flagellazione di Cristo*, (1460, Urbino, Galleria Nazionale delle Marche). (middle) A view of the reconstructed 3D model. The patterned floor has been reconstructed in areas where it is occluded by taking advantage of the symmetry of its pattern. (bottom) another view of the model with the roof removed to show the relative positions of people and architectural elements in the scene. Note the repeated geometric pattern on the floor in the area delimited by the columns (barely visible in the painting). Note that the people are represented simply as flat silhouettes since it is not possible to recover their volume from one image, they have been cut out manually from the original image. The columns have been approximated with cylinders.

SIGGRAPH 2000 Course on
3D Photography

Camera Calibration

Steve Seitz
Carnegie Mellon University

http://www.cs.cmu.edu/~seitz

Camera Calibration

Geometry

- Where is the camera?
- Where is it pointing?
- What are the internal parameters?
- What's the point spread function?

Radiometry

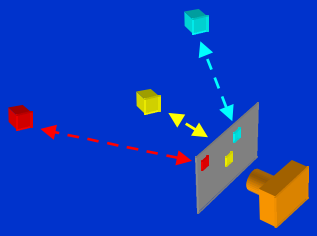
- What is the mapping from light to pixel values?
- [Debevec 97]

If Only Cameras Were "Smart" . . .

Geometric Camera Calibration

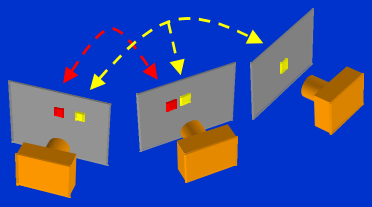
Augmented pin-hole camera model

- Focal point, orientation
- Focal length, aspect ratio, center, lens distortion



2D ⇔ 3D correspondence

"Classical" calibration



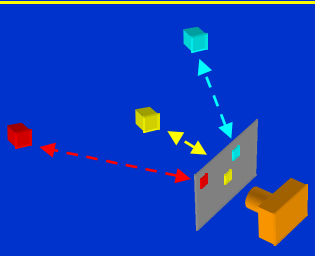
2D ⇔ 2D correspondence

SFM, "Self-calibration"

Linear Geometric Calibration

Know 3D coords, 2D coords

- Find projection matrix Π



$$d\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\mathbf{u} = \Pi \mathbf{X}$$

11 unknowns (up to scale)
2 equations per point
(eliminate d)

6 points is sufficient

Nonlinear Methods

Problems with Linear Method

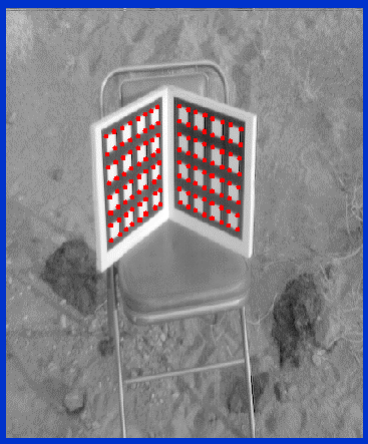
- **Too many free parameters**
- **Doesn't model lens distortion**

Nonlinear Methods **[Tsai, 1985]**

- **Parameterize Π in terms of**
  - > rotation: θ, ϕ, ψ
  - > translation: X, Y, Z
  - > intrinsics: f, aspect ratio, image center
  - > radial lens distortion: k_1, k_2

Code Available Via Course Web Page

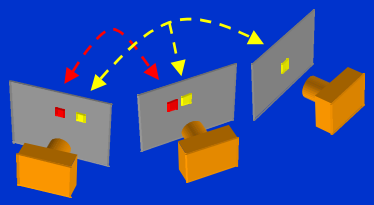
Calibration Patterns



Calibration grid
Z. Zhang, Microsoft Research

Chromaglyphs
Bruce Culbertson, HP-labs

Calibration From 2D Motion



Structure From Motion (SFM)

- Track points over a sequence of images
- Solve for 3D positions and camera positions
- Calibrate internal parameters beforehand

Self-Calibration

- Solve for internal *and* external parameters
- E.g., [Polleyfeys, 98]

Resources

Computer Vision Home Page

- http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/vision.html

Matlab and C Implementations

- Via course web page
- http://www.cs.cmu.edu/~seitz/course/3DPhoto.html

Bibliography

Geometric Calibration

- R. J. Tsai, A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses, IEEE Journal of Robotics and Automation, Vol. 3, No. 4, 1987, pp. 323-344.

Radiometric Calibration

- Paul E. Debevec and Jitendra Malik, “Recovering High Dynamic Range Radiance Maps from Photographs”, Proc. SIGGRAPH 97, pp. 369-378.

Structure-from-Motion

- Carlo Tomasi & Takeo Kanade, ”Shape and Motion from Image Streams Under Orthography: A Factorization Method", Int. Journal of Computer Vision, 9(2), 1992, pp. 137-154.

Self-Calibration

- Marc Pollefeys, Reinhard Koch, and Luc Van Gool, "Self-Calibration and Metric Reconstruction in spite of Varying Unknown Internal Camera Parameters", Proc. Sixth Int. Conf. on Computer Vision,1998, pp. 90-91.

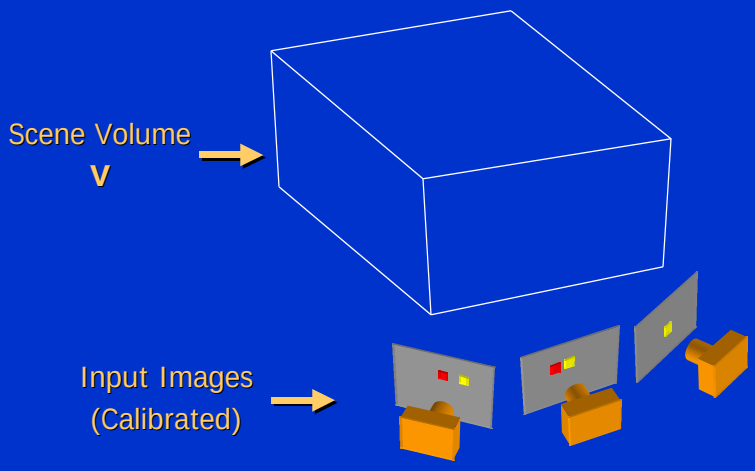
SIGGRAPH 2000 Course on
3D Photography

From Images to Voxels

Steve Seitz
Carnegie Mellon University

http://www.cs.cmu.edu/~seitz

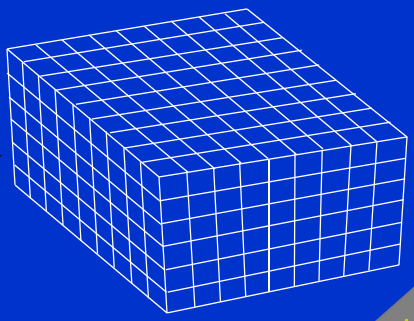
3D Reconstruction from Calibrated Images



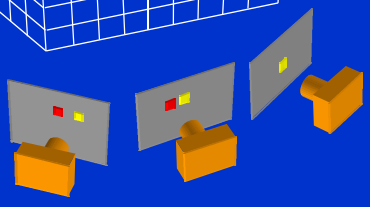
Goal: Determine transparency, radiance of points in V

Discrete Formulation: Voxel Coloring

Discretized Scene Volume →



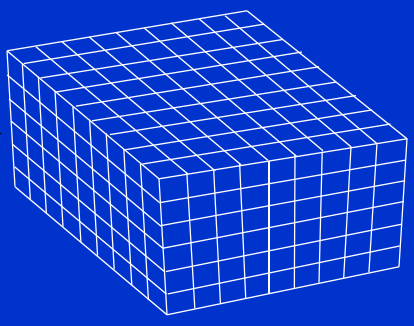
Input Images (Calibrated) →



Goal: Assign RGBA values to voxels in V *photo-consistent* with images

Complexity and Computability

Discretized Scene Volume →



N^3 voxels
C colors

G = space of all colorings (C^{N^3})
ℵ = space of all photo-consistent colorings (computable?)
S = true scene (not computable)

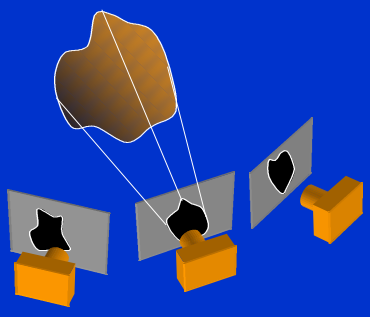
$S \in \aleph \subset G$

Voxel Coloring Solutions

1. *C=2 (silhouettes)*
   - Volume intersection [Martin 81, Szeliski 93]
2. *C unconstrained, viewpoint constraints*
   - Voxel coloring algorithm [Seitz & Dyer 97]
3. *General Case*
   - Space carving [Kutulakos & Seitz 98]

Reconstruction from Silhouettes (C = 2)

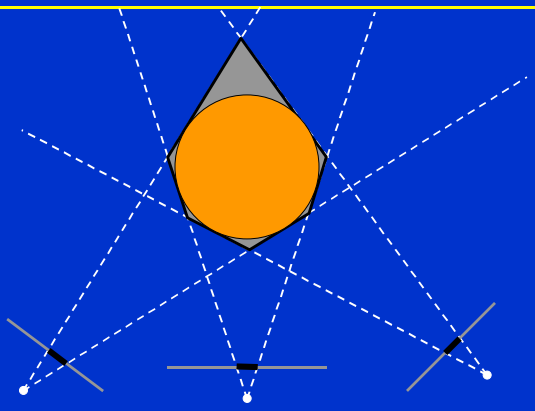
Binary Images →



Approach:

- *Backproject* each silhouette
- Intersect backprojected volumes

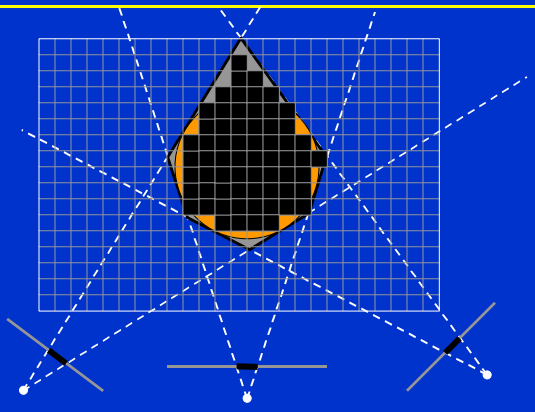
Volume Intersection



Reconstruction Contains the True Scene

- **But is generally not the same (no concavities)**
- **In the limit (all views) get *visual hull* or *line hull***
  > Complement of all lines that don't intersect S

Voxel Algorithm for Volume Intersection



Color voxel black if on silhouette in every image

- **O(MN^3), for M images, N^3 voxels**
- **Don't have to search 2$^{N^3}$ possible scenes!**

Properties of Volume Intersection

Pros

- Easy to implement, fast
- Accelerated via octrees [Szeliski 1993]

Cons

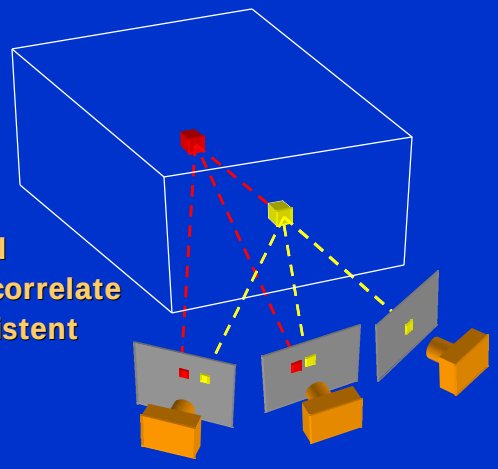
- No concavities
- Reconstruction is not photo-consistent
- Requires identification of silhouettes

Voxel Coloring Solutions

1. *C=2 (silhouettes)*
   - Volume intersection [Martin 81, Szeliski 93]
2. *C unconstrained, viewpoint constraints*
   - Voxel coloring algorithm [Seitz & Dyer 97]
3. *General Case*
   - Space carving [Kutulakos & Seitz 98]

Voxel Coloring Approach

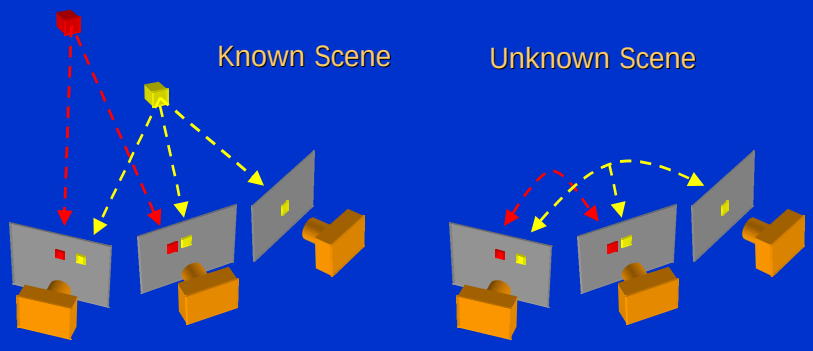
1. Choose voxel
2. Project and correlate
3. Color if consistent



Visibility Problem: in which images is each voxel visible?

The Global Visibility Problem

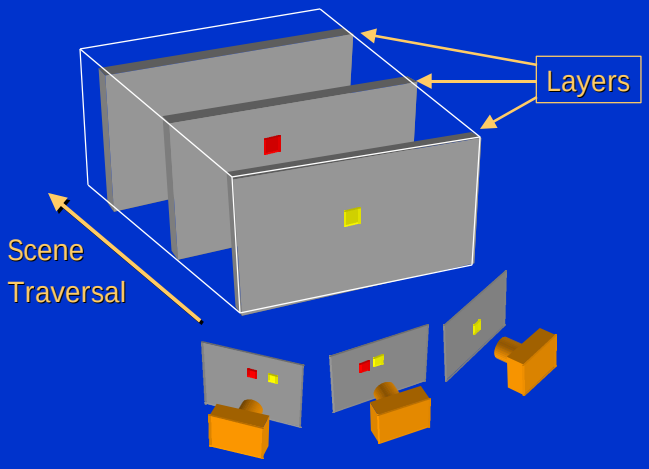
Which points are visible in which images?



Forward Visibility
known scene

Inverse Visibility
known images

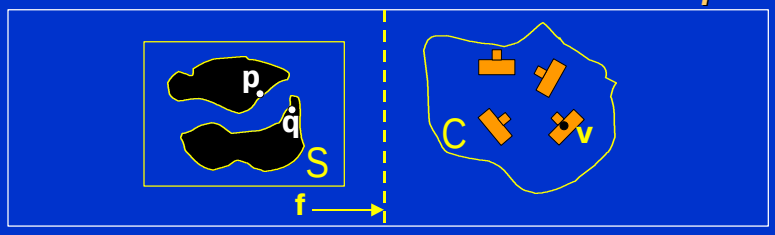
Depth Ordering: visit occluders first!



Condition: depth order is ***view-independent***

What is A *View-Independent* Depth Order?

A function f over a scene S and a camera space C



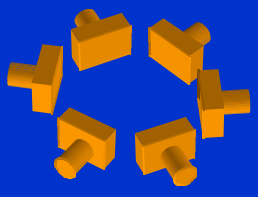
Such that for all p and q in S, v in C

p occludes q from v *only if* `f(p) < f(q)`

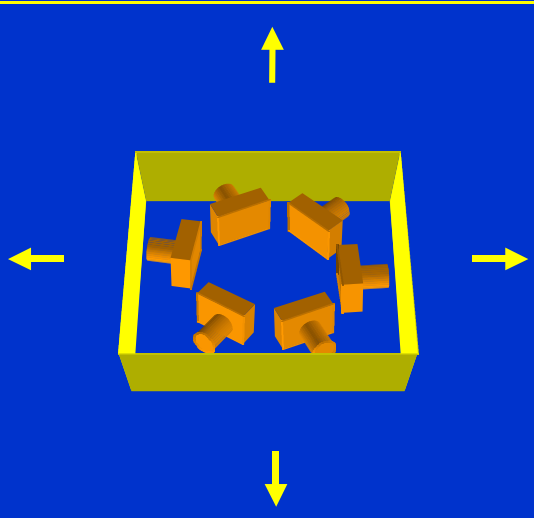
For example: f = distance from separating plane

Panoramic Depth Ordering

- Cameras oriented in many different directions
- Planar depth ordering does not apply

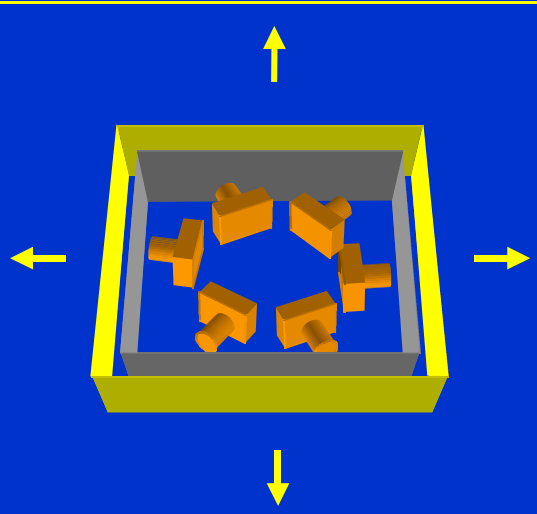


Panoramic Depth Ordering



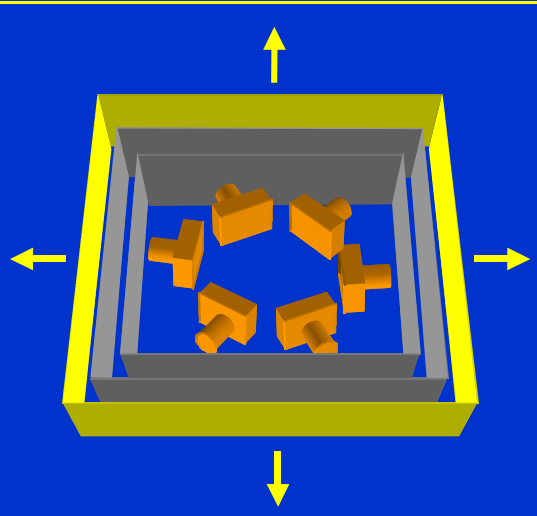
Layers radiate outwards from cameras

Panoramic Layering



Layers radiate outwards from cameras

Panoramic Layering

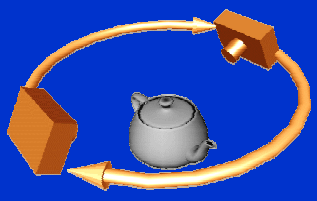


Layers radiate outwards from cameras

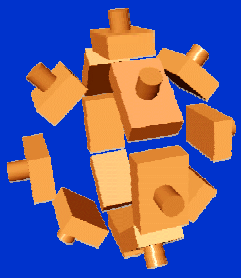
Compatible Camera Configurations

Depth-Order Constraint

- Scene outside convex hull of camera centers



Inward-Looking
cameras above scene



Outward-Looking
cameras inside scene

Calibrated Image Acquisition

Calibrated Turntable
360° rotation (21 images)

Selected Dinosaur Images

Selected Flower Images

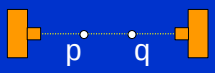
Voxel Coloring Results (Video)

Dinosaur Reconstruction
72 K voxels colored
7.6 M voxels tested
7 min. to compute
on a 250MHz SGI

Flower Reconstruction
70 K voxels colored
7.6 M voxels tested
7 min. to compute
on a 250MHz SGI

Limitations of Depth Ordering

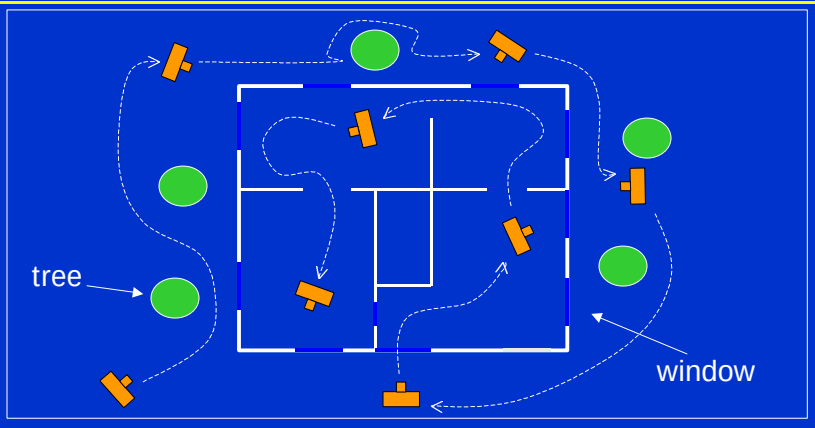
A view-independent depth order may not exist



Need more powerful general-case algorithms

- Unconstrained camera positions
- Unconstrained scene geometry/topology

A More Difficult Problem: Walkthrough



Input: calibrated images from arbitrary positions
Output: 3D model photo-consistent with all images

Voxel Coloring Solutions

1. *C=2 (silhouettes)*
   - Volume intersection [Martin 81, Szeliski 93]
2. *C unconstrained, viewpoint constraints*
   - Voxel coloring algorithm [Seitz & Dyer 97]
3. *General Case*
   - Space carving [Kutulakos & Seitz 98]

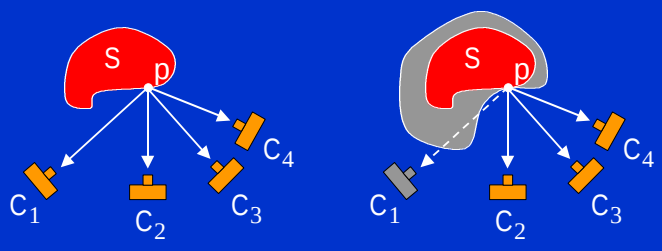
Space Carving Algorithm

- **Step 1:** **Initialize V to volume containing true scene**
- **Step 2:** **For every voxel on surface of V**
  - > test *photo-consistency* of voxel
  - > if voxel is inconsistent, carve it
- **Step 3:** **Repeat Step 2 until all voxels consistent**

Convergence:

- ***Always*** **converges to a photo-consistent model (when all assumptions are met)**
- **Good results on difficult real-world scenes**

Visibility Property



p Î S consistent Þ p Î consistent

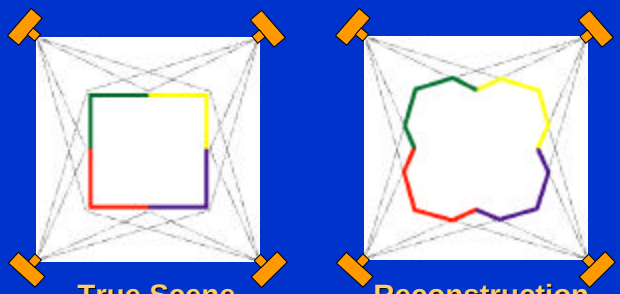
p Î inconsistent Þ p Î S inconsistent

This property ensures that carving converges

Space Carving Convergence Properties

Properties

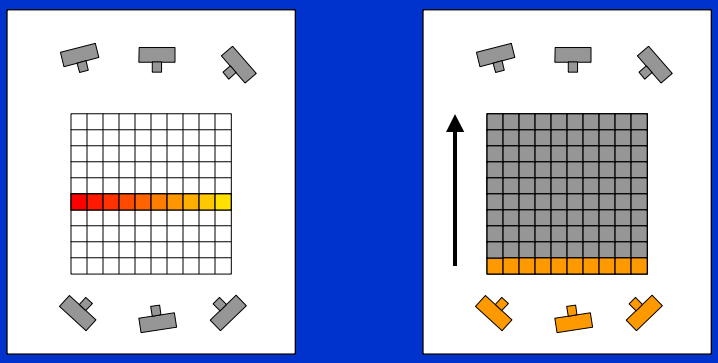
- **Guaranteed convergence to photo-consistent reconstruction (M) called the *photo hull***
  - > M = ∪ℵ --- union of all photo-consistent scenes
- **Tightest possible bound on true scene**
- ***Worst case* # consistency checks:** (# cameras)²(# voxels)



True Scene Reconstruction

Multi-Pass Plane Sweep

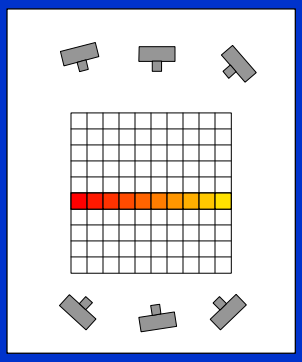
- **Sweep plane in each of 6 principle directions**
- **Consider cameras on only one side of plane**
- **Repeat until convergence**

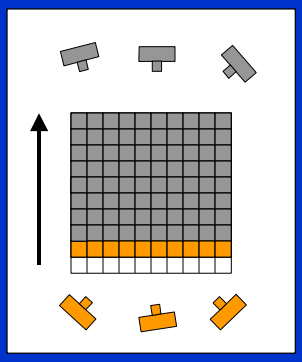


True Scene Reconstruction

Multi-Pass Plane Sweep

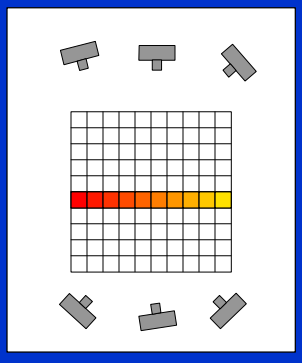
- Sweep plane in each of 6 principle directions
- Consider cameras on only one side of plane
- Repeat until convergence

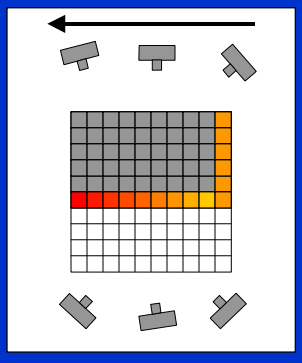




Multi-Pass Plane Sweep

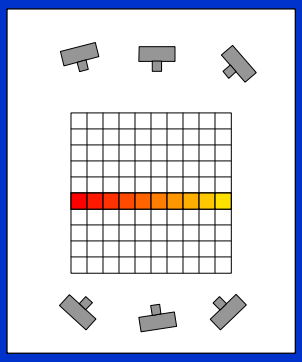
- Sweep plane in each of 6 principle directions
- Consider cameras on only one side of plane
- Repeat until convergence

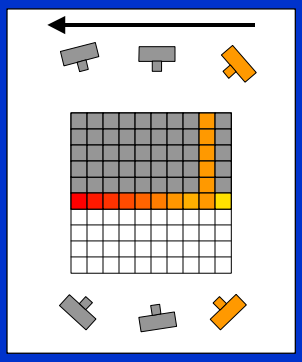




Multi-Pass Plane Sweep

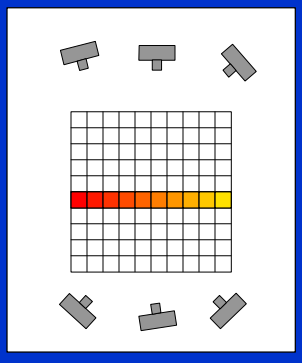
- Sweep plane in each of 6 principle directions
- Consider cameras on only one side of plane
- Repeat until convergence

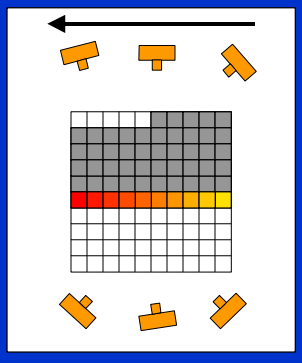




Multi-Pass Plane Sweep

- Sweep plane in each of 6 principle directions
- Consider cameras on only one side of plane
- Repeat until convergence





Space Carving Algorithm

Optimal algorithm is unwieldy

- Complex visibility update procedure

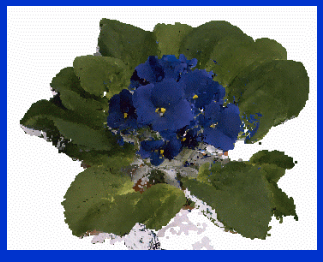
Alternative: multi-pass plane sweep

- Efficient, can use texture-mapping hardware
- Converges quickly in practice
- Easy to implement

Space Carving Results: African Violet

Input Image (1 of 45)

Reconstruction



Reconstruction

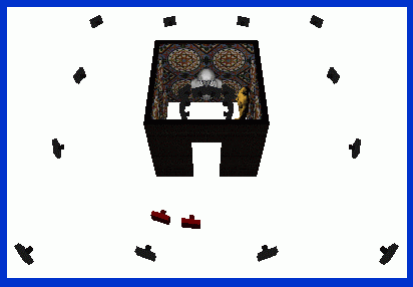
Reconstruction

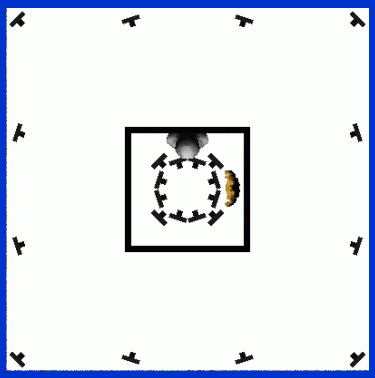
Space Carving Results: Hand

Input Image
(1 of 100)

Views of Reconstruction

House Walkthrough





24 rendered input views from inside and *outside*

Space Carving Results: House

Input Image
(true scene)

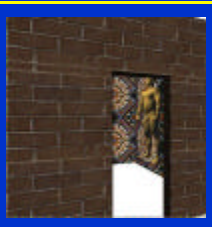
Reconstruction
370,000 voxels

Space Carving Results: House

Input Image
(true scene)

Reconstruction
370,000 voxels

Space Carving Results: House



New View (true scene)

Reconstruction

New View
(true scene)

Reconstruction

Reconstruction
(with new input view)

Other Features

Coarse-to-fine Reconstruction

- Represent scene as octree
- Reconstruct low-res model first, then refine

Hardware-Acceleration

- Use texture-mapping to compute voxel projections
- Process voxels an entire plane at a time

Limitations

- Need to acquire calibrated images
- Restriction to simple radiance models
- Bias toward maximal (fat) reconstructions
- Transparency not supported

Other Approaches

Level-Set Methods [Faugeras & Keriven 1998]
- Evolve implicit function by solving PDE's

Transparency and Matting [Szeliski & Golland 1998]
- Compute voxels with alpha-channel

Max Flow/Min Cut [Roy & Cox 1998]
- Graph theoretic formulation

Mesh-Based Stereo [Fua & Leclerc 95]
- Mesh-based but similar consistency formulation

Virtualized Reality [Narayan, Rander, Kanade 1998]
- Perform stereo 3 images at a time, merge results

Conclusions

Advantages of Voxels
- Non-parametric
  > can model arbitrary geometry
  > can model arbitrary topology
- Good reconstruction algorithms
- Good rendering algorithms (splatting, LDI)

Disadvantages
- Expensive to process hi-res voxel grids
- Large number of parameters
  > Simple scenes (e.g., planes) require lots of voxels
- Meshes simplify better

Bibliography

Volume Intersection

- Martin & Aggarwal, “Volumetric description of objects from multiple views”, Trans. Pattern Analysis and Machine Intelligence, 5(2), 1991, pp. 150-158.
- Szeliski, “Rapid Octree Construction from Image Sequences”, Computer Vision, Graphics, and Image Processing: Image Understanding, 58(1), 1993, pp. 23-32.

Voxel Coloring and Space Carving

- Seitz & Dyer, “Photorealistic Scene Reconstruction by Voxel Coloring”, Proc. Computer Vision and Pattern Recognition (CVPR), 1997, pp. 1067-1073.
- Seitz & Kutulakos, “Plenoptic Image Editing”, Proc. Int. Conf. on Computer Vision (ICCV), 1998, pp. 17-24.
- Kutulakos & Seitz, “A Theory of Shape by Space Carving”, U. Rochester C.S. Dept. TR #692, May 1998, to appear in Proc. ICCV 99.

Bibliography

Related References

- Faugeras & Keriven, “Variational principles, surface evolution, PDE's, level set methods and the stereo problem", IEEE Trans. on Image Processing, 7(3), 1998, pp. 336-344.
- Szeliski & Golland, “Stereo Matching with Transparency and Matting”, Proc. Int. Conf. on Computer Vision (ICCV), 1998, 517-524.
- Roy & Cox, “A Maximum-Flow Formulation of the N-camera Stereo Correspondence Problem”, Proc. ICCV, 1998, pp. 492-499.
- Fua & Leclerc, “Object-centered surface reconstruction: Combining multi-image stereo and shading", Int. Journal of Computer Vision, 16, 1995, pp. 35-56.
- Narayanan, Rander, & Kanade, “Constructing Virtual Worlds Using Dense Stereo”, Proc. ICCV, 1998, pp. 3-10.

Photorealistic Scene Reconstruction by Voxel Coloring

Steven M. Seitz Charles R. Dyer

Department of Computer Sciences
University of Wisconsin, Madison, WI 53706
E-mail: {seitz,dyer}@cs.wisc.edu
WWW: http://www.cs.wisc.edu/computer-vision

Abstract

A novel scene reconstruction technique is presented, different from previous approaches in its ability to cope with large changes in visibility and its modeling of intrinsic scene color and texture information. The method avoids image correspondence problems by working in a discretized scene space whose voxels are traversed in a fixed visibility ordering. This strategy takes full account of occlusions and allows the input cameras to be far apart and widely distributed about the environment. The algorithm identifies a special set of invariant voxels which together form a spatial and photometric reconstruction of the scene, fully consistent with the input images. The approach is evaluated with images from both inward- and outward-facing cameras.

1 Introduction

We consider the problem of acquiring photorealistic 3D models of real environments from widely distributed viewpoints. This problem has sparked recent interest in the computer vision community [1, 2, 3, 4, 5] as a result of new applications in telepresence, virtual walkthroughs, and other graphics-oriented problems that require realistic textured object models.

We use the term *photorealism* to describe 3D reconstructions of real scenes whose reprojections contain sufficient color and texture information to accurately reproduce images of the scene from a broad range of target viewpoints. To ensure accurate reprojections, the input images should be representative, i.e., sparsely distributed throughout the target range of viewpoints. Accordingly, we propose two criteria that a photorealistic reconstruction technique should satisfy:

- Photo Integrity: The reprojected model should accurately reproduce the input images, preserving color, texture and pixel resolution
- Broad Viewpoint Coverage: Reprojections should be accurate over a large range of target viewpoints. This requires that the *input images* are widely distributed about the environment

The support of DARPA and the National Science Foundation under Grant No. IRI-9530985 is gratefully acknowledged.

The photorealistic scene reconstruction problem, as presently formulated, raises a number of unique challenges that push the limits of existing reconstruction techniques. Photo integrity requires that the reconstruction be dense and sufficiently accurate to reproduce the original images. This criterion poses a problem for existing feature- and contour-based techniques that do not provide dense shape estimates. While these techniques can produce texture-mapped models [1, 3, 4], accuracy is ensured only in places where features have been detected. The second criterion means that the input views may be far apart and contain significant occlusions. While some stereo methods [6, 7] can cope with limited occlusions, handling visibility changes of greater magnitude appears to be beyond the state of the art.

Instead of approaching this problem as one of shape reconstruction, we instead formulate a *color reconstruction* problem, in which the goal is an assignment of colors (radiances) to points in an (unknown) approximately Lambertian scene. As a solution, we present a *voxel coloring* technique that traverses a discretized 3D space in "depth-order" to identify voxels that have a unique coloring, constant across all possible interpretations of the scene. This approach has several advantages over existing stereo and structure-from-motion approaches to scene reconstruction. First, occlusions are explicitly modeled and accounted for. Second, the cameras can be positioned far apart without degrading accuracy or run-time. Third, the technique integrates numerous images to yield dense reconstructions that are accurate over a wide range of target viewpoints.

The voxel coloring algorithm presented here works by discretizing scene space into a set of voxels that is traversed and colored in a special order. In this respect, the method is similar to Collins' Space-Sweep approach [8]

which performs an analogous scene traversal. However, the Space-Sweep algorithm does not provide a solution to the occlusion problem, a primary contribution of this paper. Katayama et al. [9] described a related method in which images are matched by detecting lines through slices of an epipolar volume, noting that occlusions could be explained by labeling lines in order of increasing slope. Our voxel traversal strategy yields a similar scene-space ordering but is not restricted to linear camera paths. However, their algorithm used a reference image, thereby ignoring points that are occluded in the reference image but visible elsewhere. Also related are recently developed panoramic stereo [10, 11] algorithms which avoid field of view problems by matching 360° panoramic images directly. Panoramic reconstructions can also be achieved using our approach, but without the need to build panoramic images (see Figs. 1 (b) and 4).

The remainder of the paper is organized as follows: Section 2 formulates and solves the voxel coloring problem, and describes its relations to shape reconstruction. Section 3 presents an efficient algorithm for computing the voxel coloring from a set of images, discussing complexity and related issues. Section 4 describes experiments on real and synthetic image sequences.

2 Voxel Coloring

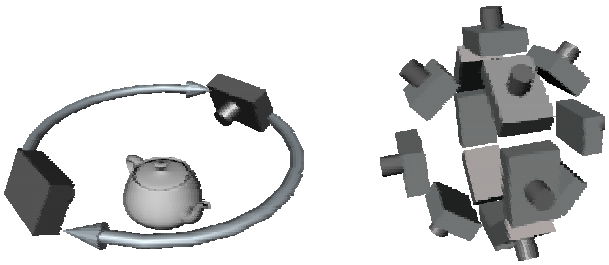
The voxel coloring problem is to assign colors (radiances) to voxels (points) in a 3D volume so as to maximize *photo integrity* with a set of input images. That is, rendering the colored voxels from each input viewpoint should reproduce the original image as closely as possible. In order to solve this coloring problem we must consider the following two issues:

- Uniqueness: Multiple voxel colorings can be consistent with a given set of images. How can the problem be well-defined?
- Computation: How can a voxel coloring be computed from a set of input images?

This section formalizes the voxel coloring problem and explores geometrical constraints on camera placement that enable an efficient solution. In order to address the uniqueness and computation issues, we introduce a novel visibility constraint that greatly simplifies the analysis. This *ordinal visibility constraint* enables the identification of certain *invariant* voxels whose colorings are uniquely defined. In addition, the constraint defines a depth-ordering of voxels by which the coloring can be computed in a single pass through the scene volume.

2.1 Notation

We assume that both the scene and lighting are stationary and that surfaces are approximately Lambertian. Under these conditions, the radiance at each point is isotropic and can therefore be described by a scalar value which we call *color*. We also use the term color to refer to the irradiance of an image pixel. The term's meaning should be clear by context.



(a) **(b)**

Figure 1: Compatible Camera Configurations. Both of the following camera configurations satisfy the ordinal visibility constraint: (a) a downward-facing camera moved 360 degrees around an object, and (b) a rig of outward-facing cameras distributed around a sphere.

A 3D scene $\mathcal{S}$ is represented as a finite[1] set of opaque voxels (volume elements), each of which occupies a finite homogeneous scene volume and has a fixed color. We denote the set of all voxels with the symbol $\mathcal{V}$. An image is specified by the set $\mathcal{I}$ of all its pixels. For now, assume that pixels are infinitesimally small.

Given an image pixel p and scene $\mathcal{S}$, we refer to the voxel $V \in \mathcal{S}$ that is visible and projects to p by $V = \mathcal{S}(p)$. The color of an image pixel $p \in \mathcal{I}$ is given by $color(p, \mathcal{I})$ and of a voxel V by $color(V, \mathcal{S})$. A scene $\mathcal{S}$ is said to be *complete* with respect to a set of images if, for every image $\mathcal{I}$ and every pixel $p \in \mathcal{I}$, there exists a voxel $V \in \mathcal{S}$ such that $V = \mathcal{S}(p)$. A complete scene is said to be *consistent* with a set of images if, for every image $\mathcal{I}$ and every pixel $p \in \mathcal{I}$,

$$color(p, \mathcal{I}) = color(\mathcal{S}(p), \mathcal{S}) \quad (1)$$

2.2 The Ordinal Visibility Constraint

For concreteness, a pinhole perspective projection model is assumed. To simplify the analysis, we introduce a constraint on the positions of the cameras relative to the scene. This constraint simplifies the task of resolving visibility relationships by establishing a fixed depth-order enumeration of points in the scene.

Let P and Q be scene points and $\mathcal{I}$ be an image from a camera centered at C. We say P *occludes* Q if P lies on the line segment $\overline{CQ}$. We require that the input cameras are positioned so as to satisfy the following constraint:

[1] It is assumed that the visible scene is spatially bounded.

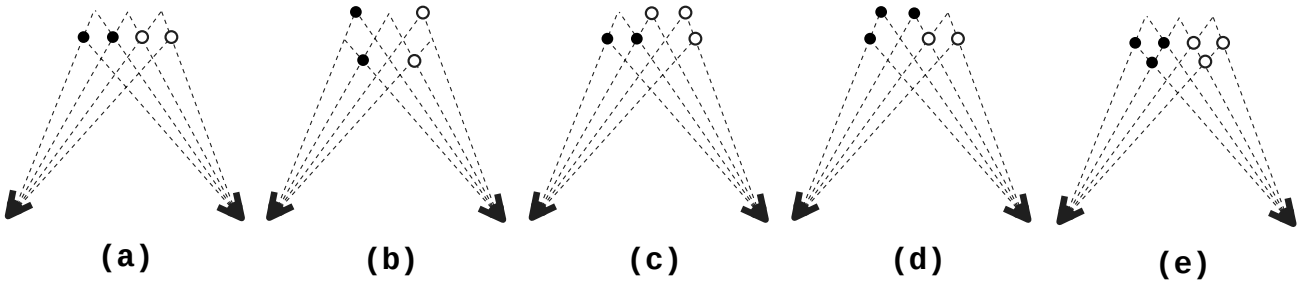


Figure 2: Ambiguity in Scene Reconstruction. All five scenes are indistinguishable from these two viewpoints. Shape ambiguity: scenes (a) and (b) have no points in common—no hard points exist. Color ambiguity: (c) and (d) share a point that has a different color assignment in the two scenes. Color invariants: each point in (e) has the same color in every consistent scene in which it is contained. These six points constitute the *voxel coloring* for these two views.

> **Ordinal visibility constraint:** There exists a norm $\|\cdot\|$ such that for all scene points P and Q, and input images $\mathcal{I}$, P occludes Q in $\mathcal{I}$ only if $\|P\| < \|Q\|$.

We call such a norm *occlusion-compatible*. For some camera configurations, it is not possible to define an occlusion-compatible norm. However, a norm *does* exist for a broad range of practical configurations. For instance, suppose the cameras are distributed on a plane and the scene is entirely below that plane, as shown in Fig. 1(a). For every such viewpoint, the relative visibility of any two scene points depends entirely on which point is closer to the plane, so we may define $\|\cdot\|$ to be distance to the plane. More generally, the ordinal visibility constraint is satisfied whenever **no scene point is contained within the convex hull $\mathcal{C}$ of the camera centers**. Here we use the occlusion-compatible norm $\|P\|_{\mathcal{C}}$, defined to be the Euclidean distance from P to $\mathcal{C}$. For convenience, $\mathcal{C}$ is referred to as the *camera volume*. Fig. 1 shows two useful camera configurations that satisfy this constraint. Fig. 1(a) depicts an inward-facing overhead camera rotating 360° around an object. Ordinal visibility is satisfied provided the camera is positioned slightly above the object. The constraint also enables panoramic reconstructions from outward-facing cameras, as in Fig. 1(b).

2.3 Color Invariance

The ordinal visibility constraint provides a depth-ordering of points in the scene. We now describe how this ordering can be used in scene reconstruction. Scene reconstruction is complicated by the fact that a set of images can be consistent with more than one rigid scene. Determining a scene's spatial occupancy is therefore an ill-posed task because a voxel contained in one consistent scene may not be contained in another. (see Fig. 2(a),(b)). Alternatively, a voxel may be part of two consistent scenes, but have different colors in each (Fig. 2(c),(d)).

Given a multiplicity of solutions to the color reconstruction problem, the only way to recover intrinsic scene information is through *invariants*— properties that are satisfied by *every* consistent scene. For instance, consider the set of voxels that are contained in every consistent scene. Laurentini [12] described how these invariants, called *hard points*, could be recovered by volume intersection from binary images. Hard points are useful in that they provide absolute information about the true scene. However, such points are relatively rare; some images may yield none (see, for example, Fig. 2). In this section we describe a more frequently occurring type of invariant relating to color rather than shape.

> A voxel V is a **color invariant** with respect to a set of images if, for every pair of scenes $\mathcal{S}$ and $\mathcal{S}'$ consistent with the images, $V \in \mathcal{S} \cap \mathcal{S}'$ implies $color(V, \mathcal{S}) = color(V, \mathcal{S}')$

Unlike shape invariance, color invariance does not require that a point be contained in every consistent scene. As a result, color invariants are more prevalent than hard points. In particular, any set of images satisfying the ordinal visibility constraint yields enough color invariants to form a complete scene reconstruction, as will be shown.

Let $\mathcal{I}_1, \ldots, \mathcal{I}_m$ be a set of images. For a given image point $p \in \mathcal{I}_j$ define V_p to be the voxel in $\{\mathcal{S}(p) \mid \mathcal{S}\ consistent\}$ that is closest to the camera volume. We claim that V_p is a color invariant. To establish this, observe that $V_p \in \mathcal{S}$ implies $V_p = \mathcal{S}(p)$, for if $V_p \neq \mathcal{S}(p)$, $\mathcal{S}(p)$ must be closer to the camera volume, which is impossible by the definition of V_p. It follows from Eq. (1) that V_p has the same color in every consistent scene; V_p is a color invariant.

> The **voxel coloring** of an image set $\mathcal{I}_1, \ldots, \mathcal{I}_m$ is defined to be:
> $\overline{\mathcal{S}} = \{V_p \mid p \in \mathcal{I}_i,\ 1 \leq i \leq m\}$

Fig. 2(e) shows the voxel coloring for the pair of images in Fig. 2. These six points have a unique color interpretation, constant in every consistent scene. They also comprise the closest consistent scene to the cameras in the following sense—every point in each consistent scene is either contained in the voxel coloring or is occluded by points in the voxel coloring. An interesting consequence of this closeness bias is that neighboring image pixels of the same color produce cusps in the voxel coloring, i.e., protrusions toward the camera volume. This phenomenon is clearly shown in Fig. 2(e), where the white and black points form two separate cusps. Also, observe that the voxel coloring is not a minimal reconstruction; removing the two closest points in Fig. 2(e) still leaves a consistent scene.

2.4 Computing the Voxel Coloring

In this section we describe how to compute the voxel coloring from a set of images that satisfy the ordinal visibility constraint. In addition it will be shown that the set of voxels contained in the voxel coloring form a complete scene reconstruction that is consistent with the input images.

The voxel coloring is computed one voxel at a time in an order that ensures agreement with the images at each step, guaranteeing that all reconstructed voxels satisfy Eq. (1). To demonstrate that voxel colorings form consistent scenes, we also have to show that they are complete, i.e., they account for every image pixel as defined in Section 2.1.

In order to make sure that the construction is incrementally consistent, i.e., agrees with the images at each step, we need to introduce a weaker form of consistency that applies to incomplete voxel sets. Accordingly, we say that a set of voxels with color assignments is *voxel-consistent* if its projection agrees fully with the subset of every input image that it overlaps. More formally, a set $\mathcal{S}$ is said to be voxel-consistent with images $\mathcal{I}_1, \ldots, \mathcal{I}_m$ if for every voxel $V \in \mathcal{S}$ and image pixels $p \in \mathcal{I}_i$ and $q \in \mathcal{I}_j$, $V = \mathcal{S}(p) = \mathcal{S}(q)$ implies $color(p, \mathcal{I}_i) = color(q, \mathcal{I}_j) = color(V, \mathcal{S})$. For notational convenience, define $\mathcal{S}_V$ to be the set of all voxels in $\mathcal{S}$ that are closer than V to the camera volume. Scene consistency and voxel consistency are related by the following properties:

1. If $\mathcal{S}$ is a consistent scene then $\{V\} \cup \mathcal{S}_V$ is a voxel-consistent set for every $V \in \mathcal{S}$.

2. Suppose $\mathcal{S}$ is complete and, for each point $V \in \mathcal{S}$, $V \cup \mathcal{S}_V$ is voxel-consistent. Then $\mathcal{S}$ is a consistent scene.

A consistent scene may be created using the second property by incrementally moving farther from the camera volume and adding voxels to the current set that maintain voxel-consistency. To formalize this idea, we define the following partition of 3D space into voxel layers of uniform distance from the camera volume:

$$\mathcal{V}_{\mathcal{C}}^{d} = \{V \mid \|V\|_{\mathcal{C}} = d\} \quad (2)$$

$$\mathcal{V} = \bigcup_{i=1}^{r} \mathcal{V}_{\mathcal{C}}^{d_i} \quad (3)$$

where $d_1, \ldots, d_r$ is an increasing sequence of numbers.

The voxel coloring is computed inductively as follows:

$$\begin{aligned} \mathcal{SP}_1 &= \{V \mid V \in \mathcal{V}_{d_1}, \{V\} \text{ voxel-consistent}\} \\ \mathcal{SP}_k &= \{V \mid V \in \mathcal{V}_{d_k}, \{V\} \cup \mathcal{SP}_{k-1} \text{ voxel-consistent}\} \\ \mathcal{SP} &= \{V \mid V = \mathcal{SP}_r(p) \text{ for some pixel } p\} \end{aligned}$$

We claim $\mathcal{SP} = \overline{\mathcal{S}}$. To prove this, first define $\overline{\mathcal{S}}_i = \{V \mid V \in \overline{\mathcal{S}}, \|V\|_{\mathcal{C}} \leq d_i\}$. $\overline{\mathcal{S}}_1 \subseteq \mathcal{SP}_1$ by the first consistency property. Inductively, assume that $\overline{\mathcal{S}}_{k-1} \subseteq \mathcal{SP}_{k-1}$ and let $V \in \overline{\mathcal{S}}_k$. By the first consistency property, $\{V\} \cup \overline{\mathcal{S}}_{k-1}$ is voxel-consistent, implying that $\{V\} \cup \mathcal{SP}_{k-1}$ is also voxel-consistent, because the second set includes the first and $\mathcal{SP}_{k-1}$ is itself voxel-consistent. It follows that $\overline{\mathcal{S}} \subseteq \mathcal{SP}_r$. Note also that $\mathcal{SP}_r$ is complete, since one of its subsets is complete, and hence consistent by the second consistency property. $\mathcal{SP}$ contains all the voxels in $\mathcal{SP}_r$ that are visible in any image, and is therefore consistent as well. Therefore $\mathcal{SP}$ is a consistent scene such that for each pixel p, $\mathcal{SP}(p)$ is at least as close to $\mathcal{C}$ as $\overline{\mathcal{S}}(p)$. Hence $\mathcal{SP} = \overline{\mathcal{S}}$. □

In summary, the following properties of voxel colorings have been shown:

- $\overline{\mathcal{S}}$ is a consistent scene
- Every voxel in $\overline{\mathcal{S}}$ is a color invariant
- $\overline{\mathcal{S}}$ is computable from any set of images satisfying the ordinal visibility constraint

3 Reconstruction by Voxel Coloring

In this section we present a voxel coloring algorithm for reconstructing a scene from a set of calibrated images. The algorithm closely follows the voxel coloring construction outlined in Section 2.4, adapted to account for image discretization and noise. As before, it is assumed that 3D space has been partitioned into a series of voxel layers $\mathcal{V}_{\mathcal{C}}^{d_1}, \ldots, \mathcal{V}_{\mathcal{C}}^{d_r}$ increasing in distance from the camera volume. The images $\mathcal{I}_1, \ldots, \mathcal{I}_m$ are assumed to be discretized into finite non-overlapping pixels. The cameras are assumed to satisfy the ordinal visibility constraint, i.e., no scene point lies within the camera volume.

If a voxel V is not fully occluded in image $\mathcal{I}_j$, its projection will overlap a nonempty set of image pixels, π_j. Without noise or quantization effects, a consistent voxel should project to a set of pixels with equal color values. In the presence of these effects, we evaluate the correlation of

the pixel colors to measure the likelihood of voxel consistency. Let s be the standard deviation and n the cardinality of $\bigcup_{j=1}^{m} \pi_j$. Suppose the sensor error (accuracy of irradiance measurement) is approximately normally distributed with standard deviation σ_0. If σ_0 is unknown, it can be estimated by imaging a homogeneous surface and computing the standard deviation of image pixels. The consistency of a voxel can be estimated using the likelihood ratio test: $\lambda_V = \frac{(n-1)s^2}{\sigma_0^2}$, distributed as χ^2 [13].

3.1 Voxel Coloring Algorithm

The algorithm is as follows:

```
S = ∅
for i = 1, ..., r do
  for every V ∈ V_C^{d_i} do
    project to I_1, ..., I_m, compute λ_V
    if λ_V < thresh then S = S ∪ {V}
```

The threshold, $thresh$, corresponds to the maximum allowable correlation error. An overly conservative (small) value of $thresh$ results in an accurate but incomplete reconstruction. On the other hand, a large threshold yields a more complete reconstruction, but one that includes some erroneous voxels. In practice, $thresh$ should be chosen according to the desired characteristics of the reconstructed model, in terms of accuracy vs. completeness.

Much of the work of the algorithm lies in the computation of λ_V. The set of overlapping pixels depends both on the shape of V's projection and the set $\mathcal{S}$ of possibly occluding voxels. To simplify the computation, our implementation used a square mask to approximate the projected voxel shape. The problem of detecting occlusions is solved by the scene traversal ordering used in the algorithm; the order is such that if V occludes V' then V is visited before V'. Therefore, occlusions can be detected by using a one-bit mask for each image pixel. The mask is initialized to 0. When a voxel V is processed, π_i is the set of pixels that overlap V's projection in $\mathcal{I}_i$ and have mask values of 0. These pixels are marked with masks of 1 if $\lambda_V < thresh$.

Voxel traversal can be made more efficient by employing alternative occlusion-compatible norms. For instance, using the axis-aligned bounding box of the camera volume instead of $\mathcal{C}$, and L_∞ instead of L_2, gives rise to a sequence of axis-aligned cube-shaped layers.

3.2 Discussion

The algorithm visits each voxel exactly once and projects it into every image. Therefore, the time complexity of voxel coloring is: $O(voxels * images)$. To determine the space complexity, observe that evaluating one voxel does not require access to or comparison with other voxels. Consequently, voxels need not be stored in main memory during the algorithm; the voxels making up the voxel coloring will simply be output one at a time. Only the images and one-bit masks need to be allocated. The fact that the space and time complexities of voxel coloring are linear in the number of images is essential in that large numbers of images can be processed at once.

The algorithm differs from stereo and optical-flow techniques in that it does not perform window-based image correlation in the reconstruction process. Correspondences are found during the course of scene traversal by voxel projection. A disadvantage of this searchless strategy is that it requires very precise camera calibration to achieve the triangulation accuracy of stereo methods. Accuracy and run-time also depend on the voxel resolution, a parameter that can be set by the user or determined automatically to match the pixel resolution, calibration accuracy, and computational resources.

Importantly, the approach reconstructs only one of the potentially numerous scenes consistent with the input images. Consequently, it is susceptible to aperture problems caused by image regions of near-uniform color. These regions cause cusps in the reconstruction (see Fig. 2(e)), since voxel coloring yields the reconstruction closest to the camera volume. This is a bias, just like smoothness is a bias in stereo methods, but one that guarantees a consistent reconstruction even with severe occlusions.

4 Experimental Results

The first experiment involved 3D reconstruction from twenty-one views spanning a 360° object rotation. Our strategy for calibrating the views was similar to that in [14]. Instead of a turntable, we placed the objects on a software-controlled pan-tilt head, viewed from above by a fixed camera (see Fig. 1(a)). Tsai's method [15] was used to calibrate the camera with respect to the head, by rotating a known object and manually selecting image features for three pan positions. The calibration error was approximately 3%.

Fig. 3 shows the voxel colorings computed from a complete revolution of a dinosaur toy and a rose. To facilitate reconstruction, we used a black background and eliminated most of the background points by thresholding the images. While background subtraction is not strictly necessary, leaving this step out results in background-colored voxels scattered around the edges of the scene volume. The threshold may be chosen conservatively since removing most of the background pixels is sufficient to eliminate this background scattering effect. The middle column in Fig. 3 shows the reconstructions from a viewpoint corresponding to one of the input images (shown at left), to demonstrate photo integrity. Note that even fine details such as the wind-up rod on the dinosaur and the leaves of the rose were re-

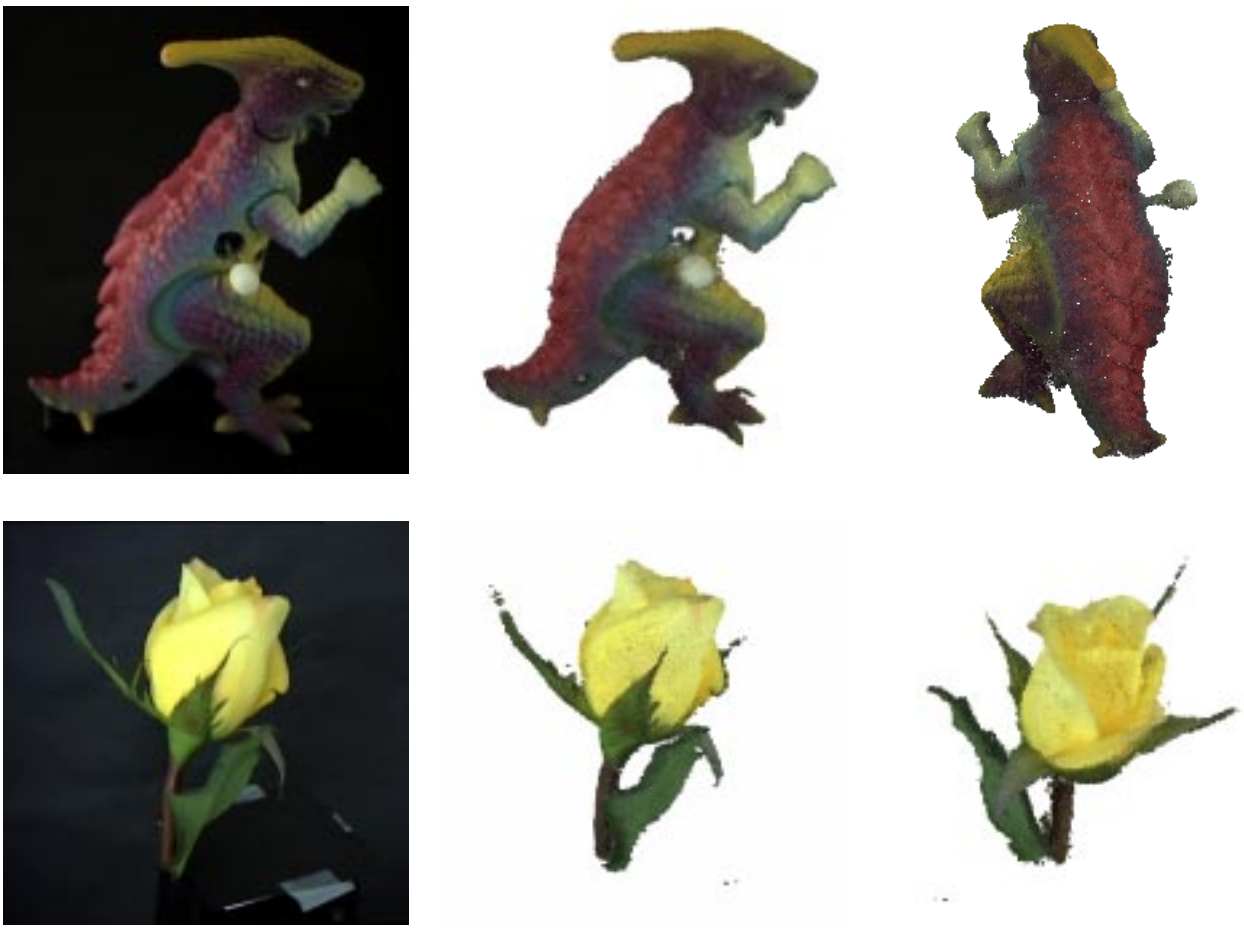


Figure 3: Voxel Coloring of Dinosaur Toy and Rose. The objects were rotated 360° below a camera. At left is one of 21 input images of each object. The other images show different views rendered from the reconstructions.

(a) **(b)** **(c)** **(d)**

Figure 4: Reconstruction of Synthetic Room Scene. The input images were all taken from cameras located inside the room. (a) shows the voxel coloring and (b) the original model from a new interior viewpoint. (c) and (d) show the reconstruction and original model, respectively, from a new viewpoint outside of the room.

constructed.

We experimented with different voxel resolutions to determine the effects of voxel sampling on reconstruction quality. Increasing the sampling rate improved the reconstruction quality, up to the limits of image quantization and calibration accuracy, at the cost of increased run-time. A low-resolution model can be built very quickly; a reconstruction (not shown) containing 980 voxels took less than a second to compute on a 250 MHz SGI Indigo2. In contrast, the 72,497-voxel dinosaur reconstruction shown in Fig. 3 required evaluating a volume of 7 million voxels and took roughly three minutes to compute.

The next experiment involved reconstructing a synthetic room from cameras *inside* the room. The room interior was highly concave, making reconstruction by volume intersection or other contour-based methods impractical. The room consisted of three texture-mapped walls and two shaded models. The models, a bust of Beethoven and a human figure, were illuminated diffusely from above. 24 cameras were placed at different positions and orientations throughout the room. The optical axes were parallel to the horizontal (XZ) plane.

Fig. 4 compares the original and reconstructed models from new viewpoints. The voxel coloring reproduced images from the room interior quite accurately (as shown in (a)), although some fine details were lost due to quantization effects. The overhead view (c) more clearly shows some discrepancies between the original and reconstructed shapes. For instance, the reconstructed walls are not perfectly planar, as some points lie just off the surface. This point drift effect is most noticeable in regions where the texture is locally homogeneous, indicating that texture information is important for accurate reconstruction. Not surprisingly, the quality of image (c) is worse than that of (a), since the former view was much farther from the input cameras. On the whole, Fig. 4 shows that the overall shape of the scene was captured quite well in the reconstruction. The recovered model contained 52,670 voxels and took 95 seconds to compute.

5 Conclusions

This paper presented a new scene reconstruction technique that incorporates intrinsic color and texture information for the acquisition of photorealistic scene models. Unlike existing stereo and structure-from-motion techniques, the method *guarantees* that a consistent reconstruction is found, even under large visibility differences across the input images. The method relies on a constraint on the input camera configuration that enables a simple solution for determining voxel visibility. A second contribution was the constructive proof of the existence of a set of color invariants. These points are useful in two ways: first, they provide information that is intrinsic, i.e., constant across all possible consistent scenes. Second, together they constitute a spatial and photometric reconstruction of the scene whose projections reproduce the input images.

References

[1] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," *Int. J. of Computer Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[2] T. Kanade, P. J. Narayanan, and P. W. Rander, "Virtualized reality: Concepts and early results," in *Proc. IEEE Workshop on Representation of Visual Scenes*, pp. 69–76, 1995.

[3] S. Moezzi, A. Katkere, D. Y. Kuramura, and R. Jain, "Reality modeling and visualization from multiple video sequences," *IEEE Computer Graphics and Applications*, vol. 16, no. 6, pp. 58–63, 1996.

[4] P. Beardsley, P. Torr, and A. Zisserman, "3D model acquisition from extended image sequences," in *Proc. European Conf. on Computer Vision*, pp. 683–695, 1996.

[5] L. Robert, "Realistic scene models from image sequences," in *Proc. Imagina 97 Conf.*, (Monaco), pp. 8–13, 1997.

[6] Y. Nakamura, T. Matsuura, K. Satoh, and Y. Ohta, "Occlusion detectable stereo–occlusion patterns in camera matrix," in *Proc. Computer Vision and Pattern Recognition Conf.*, pp. 371–378, 1996.

[7] P. N. Belhumeur and D. Mumford, "A Bayesian treatment of the stereo correspondence problem using half-occluded regions," in *Proc. Computer Vision and Pattern Recognition Conf.*, pp. 506–512, 1992.

[8] R. T. Collins, "A space-sweep approach to true multi-image matching," in *Proc. Computer Vision and Pattern Recognition Conf.*, pp. 358–363, 1996.

[9] A. Katayama, K. Tanaka, T. Oshino, and H. Tamura, "A viewpoint dependent stereoscopic display using interpolation of multi-viewpoint images," in *Proc. SPIE Vol. 2409A*, pp. 21–30, 1995.

[10] L. McMillan and G. Bishop, "Plenoptic modeling," in *Proc. SIGGRAPH 95*, pp. 39–46, 1995.

[11] S. B. Kang and R. Szeliski, "3-D scene data recovery using omnidirectional multibaseline stereo," in *Proc. Computer Vision and Pattern Recognition Conf.*, pp. 364–370, 1996.

[12] A. Laurentini, "How far 3D shapes can be understood from 2D silhouettes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 188–195, 1995.

[13] J. E. Freund, *Mathematical Statistics*. Englewood Cliffs, NJ: Prentice Hall, 1992.

[14] R. Szeliski, "Rapid octree construction from image sequences," *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 1, no. 58, pp. 23–32, 1993.

[15] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf cameras and lenses," *IEEE Trans. Robotics and Automation*, vol. 3, no. 4, pp. 323–344, 1987.

A Theory of Shape by Space Carving

Kiriakos N. Kutulakos (kyros@cs.rochester.edu)
Department of Computer Science and Department of Dermatology, Computer Studies Building, University of Rochester, Rochester, NY 14627, USA

Steven M. Seitz (seitz@cs.cmu.edu)
The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract. In this paper we consider the problem of computing the 3D shape of an unknown, arbitrarily-shaped scene from multiple photographs taken at known but arbitrarily-distributed viewpoints. By studying the equivalence class of all 3D shapes that reproduce the input photographs, we prove the existence of a special member of this class, the *photo hull*, that (1) can be computed directly from photographs of the scene, and (2) subsumes all other members of this class. We then give a provably-correct algorithm, called *Space Carving*, for computing this shape and present experimental results on complex real-world scenes. The approach is designed to (1) capture photorealistic shapes that accurately model scene appearance from a wide range of viewpoints, and (2) account for the complex interactions between occlusion, parallax, shading, and their view-dependent effects on scene-appearance.

Keywords: scene modeling, photorealistic reconstruction, multi-view stereo, space carving, voxel coloring, shape-from-silhouettes, visual hull, volumetric shape representations, metameric shapes, 3D photography

1. Introduction

A fundamental problem in computer vision is reconstructing the shape of a complex 3D scene from multiple photographs. While current techniques work well under controlled conditions (e.g., small stereo baselines (Okutomi and Kanade, 1993), active viewpoint control (Kutulakos and Dyer, 1994), spatial and temporal smoothness (Poggio et al., 1985; Bolles et al., 1987; Katayama et al., 1995), or scenes containing curved lines (Bascle and Deriche, 1993), planes (Pritchett and Zisserman, 1998), or texture-less surfaces (Cipolla and Blake, 1992; Vaillant and Faugeras, 1992; Laurentini, 1994; Szeliski and Weiss, 1994; Kutulakos and Dyer, 1995)), very little is known about scene reconstruction under general conditions. In particular, in the absence of *a priori* geometric information, what can we infer about the structure of an unknown scene from N arbitrarily positioned cameras at known viewpoints? Answering this question has many implications for reconstructing real objects and environments, which tend to be non-smooth, exhibit significant occlusions, and may contain both textured and texture-less surface regions (Figure 1).

In this paper, we develop a theory for reconstructing 3D scenes from photographs by formulating shape recovery as a constraint satisfaction problem. We show that any set of photographs of a rigid scene defines a collection of *picture constraints* that are satisfied by every scene projecting to those photographs. Furthermore, we characterize the set of all 3D shapes that satisfy these constraints and use the underlying theory to design a practical reconstruction algorithm, called *Space Carving*, that applies to fully-general shapes and camera configurations. In particular, we address three questions:

- Given N input photographs, can we characterize the set of all *photo-consistent shapes*, i.e., shapes that reproduce the input photographs?
- Is it possible to compute a shape from this set and if so, what is the algorithm?
- What is the relationship of the computed shape to all other photo-consistent shapes?

Our goal is to study the N-view shape recovery problem in the general case where no constraints are placed upon the scene's shape or the viewpoints of the input photographs. In particular, we address the above questions for the case when (1) no constraints are imposed on scene geometry or topology, (2) no constraints are imposed on the positions of the input cameras, (3) no information is available about the existence of specific image features in the input photographs (e.g., edges, points, lines, contours, texture, or color), and (4) no *a priori* correspondence information is available. Unfortunately, even though several algorithms have been proposed for recovering shape from multiple views that work under some of these conditions (e.g., work on stereo (Belhumeur, 1996; Cox et al., 1996; Stewart, 1995)), very little is currently known about how to answer the above questions, and even less so about how to answer them in this general case.

At the heart of our work is the observation that these questions become tractable when scene radiance belongs to a general class of radiance functions we call *locally computable*. This class characterizes scenes for which global illumination effects such as shadows, transparency and inter-reflections can be ignored, and is sufficiently general to include scenes with parameterized radiance models (e.g., Lambertian, Phong (Foley et al., 1990), Torrance-Sparrow (Torrance and Sparrow, 1967)). Using this observation as a starting point, we show how to compute, from N photographs of an unknown scene, a maximal shape called the *photo hull* that encloses the set of all photo-consistent reconstructions. The only requirements are that (1) the viewpoint of each photograph is

known in a common 3D world reference frame (Euclidean, affine (Koenderink and van Doorn, 1991), or projective (Mundy and Zisserman, 1992)), and (2) scene radiance follows a known, locally-computable radiance function. Experimental results demonstrating our method's performance are given for both real and simulated geometrically-complex scenes.

Central to our analysis is the realization that parallax, occlusion, and scene radiance all contribute to a photograph's dependence on viewpoint. Since our notion of photo-consistency implicitly ensures that all of these 3D shape cues are taken into account in the recovery process, our approach is related to work on stereo (Okutomi and Kanade, 1993; Cox et al., 1996; Hoff and Ahuja, 1989), shape-from-contour (Cipolla and Blake, 1992; Vaillant and Faugeras, 1992; Szeliski, 1993), as well as shape-from-shading (Epstein et al., 1996; Belhumeur and Kriegman, 1996; Woodham et al., 1991). These approaches rely on studying a single 3D shape cue under the assumptions that other sources of variability can be safely ignored, and that the input photographs contain features relevant to that cue (Bolles and Cain, 1982).[1] Unfortunately, these approaches cannot be easily generalized to attack the N-view reconstruction problem for arbitrary 3D scenes because neither assumption holds true in general. Implicit in this previous work is the view that untangling parallax, self-occlusion and shading effects in N arbitrary photographs of a scene leads to a problem that is either under-constrained or intractable. Here we challenge this view by showing that shape recovery from N arbitrary photographs of an unknown scene is not only a tractable problem but has a simple solution as well.

To our knowledge, no previous theoretical work has studied the equivalence class of solutions to the general N-view reconstruction problem or provably-correct algorithms for computing them. The Space Carving Algorithm that results from our analysis, however, is related to other 3D scene-space stereo algorithms that have been recently proposed (Fua and Leclerc, 1995; Collins, 1996; Seitz and Dyer, 1999; Seitz and Kutulakos, 1998; Zitnick and Webb, 1996; Narayanan et al., 1998; Szeliski and Golland, 1998; Roy and Cox, 1998). Of these, most closely related are mesh-based (Fua and Leclerc, 1995) and level-set (Faugeras and Keriven, 1998) algorithms, as well as methods that sweep a plane or other manifold through a discretized scene space (Collins, 1996; Seitz and Dyer, 1999; Seitz and Kutulakos, 1998; Szeliski and Golland, 1998; Langer and Zucker, 1994). While the algorithms in (Faugeras and Keriven, 1998; Fua and Leclerc, 1995) generate high-quality reconstructions and perform well in the presence of occlusions, their use of regularization techniques penalizes complex surfaces and shapes. Even more importantly, no formal study has been undertaken

to establish their validity for recovering arbitrarily-shaped scenes from unconstrained camera configurations (e.g., the one shown in Figure 1a). In contrast, our Space Carving Algorithm is provably correct and has no regularization biases. Even though space-sweep approaches have many attractive properties, existing algorithms (Collins, 1996; Seitz and Dyer, 1999; Seitz and Kutulakos, 1998; Szeliski and Golland, 1998) are not fully general i.e., they rely on the presence of specific image features such as edges and hence generate only sparse reconstructions (Collins, 1996), or they place strong constraints on the input viewpoints relative to the scene (Seitz and Dyer, 1999; Seitz and Kutulakos, 1998). Unlike all previous methods, Space Carving guarantees complete reconstruction in the general case.

Our approach offers six main contributions over the existing state of the art:

1. It introduces an algorithm-independent analysis of the shape recovery problem from N arbitrary photographs, making explicit the assumptions required for solving it as well as the ambiguities intrinsic to the problem. This analysis not only extends previous work on reconstruction but also puts forth a concise geometrical framework for analyzing the general properties of recently-proposed scene-space stereo techniques (Fua and Leclerc, 1995; Collins, 1996; Seitz and Dyer, 1999; Seitz and Kutulakos, 1998; Zitnick and Webb, 1996; Narayanan et al., 1998; Szeliski and Golland, 1998; Roy and Cox, 1998). In this respect, our analysis has goals similar to those of theoretical approaches to structure-from-motion (Faugeras and Maybank, 1990), although the different assumptions employed (i.e., unknown vs. known correspondences, known vs. unknown camera motion), make the geometry, solution space, and underlying techniques completely different.

2. Our analysis provides a volume which is the tightest possible bound on the shape of the true scene that can be inferred from N photographs. This bound is important because it tells us precisely what shape information we can hope to extract from N photographs, in the absence of *a priori* geometric and point correspondence information, *regardless of the specific algorithm being employed.*

3. The Space Carving Algorithm presented in this paper is the only provably-correct method, to our knowledge, that enables scene reconstruction from input cameras at arbitrary positions. As such, the algorithm enables reconstruction of complex scenes from viewpoints distributed throughout an unknown 3D environment—an extreme example is shown in Fig. 11a where the interior and exterior of a

house are reconstructed simultaneously from cameras distributed throughout the inside and outside of the house.

4. Because no constraints on the camera viewpoints are imposed, our approach leads naturally to global reconstruction algorithms (Kutulakos and Dyer, 1995; Seitz and Dyer, 1995) that recover 3D shape information from all photographs in a single step. This eliminates the need for complex partial reconstruction and merging operations (Curless and Levoy, 1996; Turk and Levoy, 1994) in which partial 3D shape information is extracted from subsets of the photographs (Narayanan et al., 1998; Kanade et al., 1995; Zhao and Mohr, 1996; Seales and Faugeras, 1995), and where global consistency with the entire set of photographs is not guaranteed for the final shape.

5. We describe an efficient multi-sweep implementation of the Space Carving Algorithm that enables recovery of photo-realistic 3D models from multiple photographs of real scenes, and exploits graphics hardware acceleration commonly available on desktop PC's.

6. Because the shape recovered via Space Carving is guaranteed to be photo-consistent, its reprojections will closely resemble photographs of the true scene. This property is especially significant in computer graphics, virtual reality, and tele-presence applications (Tomasi and Kanade, 1992; Kanade et al., 1995; Moezzi et al., 1996; Zhang, 1998; Kang and Szeliski, 1996; Sato et al., 1997) where the photo-realism of constructed 3D models is of primary importance.

1.1. Least-Commitment Shape Recovery

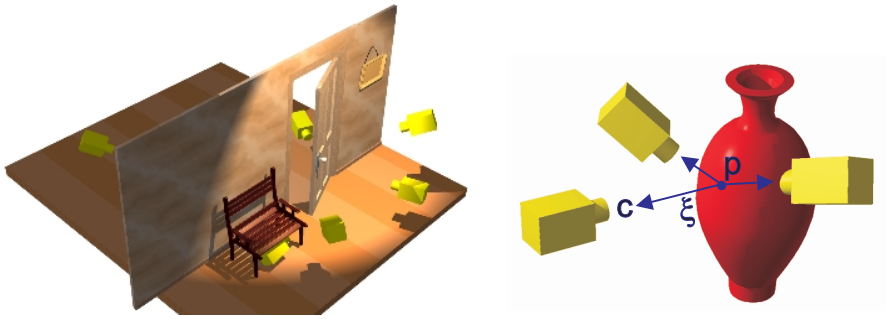
A key consequence of our photo-consistency analysis is that there are 3D scenes for which no finite set of input photographs can uniquely determine their shape: in general, there exists an uncountably-infinite equivalence class of shapes each of which reproduces all of the input photographs exactly. This result is yet another manifestation of the well-known fact that 3D shape recovery from a set of images is generally ill-posed (Poggio et al., 1985), i.e., there may be multiple shapes that are consistent with the same set of images.[2] Reconstruction methods must therefore choose a particular scene to reconstruct from the space of all consistent shapes. Traditionally, the most common way of dealing with this ambiguity has been to apply smoothness heuristics and regularization techniques (Poggio et al., 1985; Aloimonos, 1988) to obtain reconstructions that are as smooth as possible. A drawback of this

type of approach is that it typically penalizes discontinuities and sharp edges, features that are very common in real scenes.

The notion of the photo hull introduced in this paper and the Space Carving Algorithm that computes it lead to an alternative, *least commitment principle* (Marr, 1982) for choosing among all of the photo-consistent shapes: rather than making an arbitrary choice, we choose the only photo-consistent reconstruction that is guaranteed to subsume (i.e., contain within its volume) all other photo-consistent reconstructions of the scene. By doing so we not only avoid the need to impose *ad hoc* smoothness constraints, which lead to reconstructions whose relationship to the true shape are difficult to quantify, we also ensure that the recovered 3D shape can serve as a description for the entire equivalence class of photo-consistent shapes.

While our work shows how to obtain a consistent scene reconstruction without imposing smoothness constraints or other geometric heuristics, there are many cases where it may be advantageous to impose *a priori* constraints, especially when the scene is known to have a certain structure (Debevec et al., 1996; Kakadiaris and Metaxas, 1995). Least-commitment reconstruction suggests a new way of incorporating such constraints: rather than imposing them as early as possible in the reconstruction process, we can impose them after first recovering the photo hull. This allows us to delay the application of *a priori* constraints until a later stage in the reconstruction process, when tight bounds on scene structure are available and where these constraints are used only to choose among shapes within the class of photo-consistent reconstructions. This approach is similar in spirit to "stratification" approaches of shape recovery (Faugeras, 1995; Koenderink and van Doorn, 1991), where 3D shape is first recovered *modulo* an equivalence class of reconstructions and is then refined within that class at subsequent stages of processing.

The remainder of this paper is structured as follows. Section 2 analyzes the constraints that a set of photographs place on scene structure given a known, locally-computable model of scene radiance. Using these constraints, a theory of photo-consistency is developed that provides a basis for characterizing the space of all reconstructions of a scene. Sections 3 and 4 then use this theory to present the two central results of the paper, namely the existence of the photo hull and the development of a provably-correct algorithm called Space Carving that computes it. Section 5 then presents a discrete implementation of the Space Carving Algorithm that iteratively "carves" out the scene from an initial set of voxels. This algorithm can be seen as a generalization of silhouette-based techniques like volume intersection (Martin and Aggarwal, 1983; Szeliski, 1993; Kutulakos, 1997; Moezzi et al., 1996)



(a) (b)

Figure 1. Viewing geometry. The scene volume and camera distribution covered by our analysis are both completely unconstrained. Examples include (a) a 3D environment viewed from a collection of cameras that are arbitrarily dispersed in free space, and (b) a 3D object viewed by a single camera moving around it.

to the case of gray-scale and full-color images, and generalizes voxel coloring (Seitz and Dyer, 1999) and plenoptic decomposition (Seitz and Kutulakos, 1998) to the case of arbitrary camera geometries.[3] Section 6 concludes with experimental results on real and synthetic images.

2. Picture Constraints

Let $\mathcal{V}$ be a shape defined by a closed and opaque set of points that occupy a volume in space.[4] We assume that $\mathcal{V}$ is viewed under perspective projection from N known positions $c_1, \ldots, c_N$ in $\mathbb{R}^3 - \mathcal{V}$ (Figure 1b). The *radiance* of a point p on the shape's surface, $Surf(\mathcal{V})$ is a function $rad_p(\xi)$ that maps every oriented ray ξ through the point to the color of light reflected from p along ξ. We use the term *shape-radiance scene description* to denote the shape $\mathcal{V}$ together with an assignment of a radiance function to every point on its surface. This description contains all the information needed to reproduce a photograph of the scene for any camera position.[5]

Every photograph of a 3D scene taken from a known location partitions the set of all possible shape-radiance scene descriptions into two families, those that reproduce the photograph and those that do not. We characterize this constraint for a given shape and a given radiance assignment by the notion of *photo-consistency*:[6]

Definition 1 (Point Photo-Consistency) *Let $\mathcal{S}$ be an arbitrary subset of $\mathbb{R}^3$. A point $p \in \mathcal{S}$ that is visible from c is photo-consistent with*

the photograph at c if (1) p does not project to a background pixel, and (2) the color at p's projection is equal to $\mathrm{rad}_p(\vec{pc})$. *If p is not visible from c, it is trivially photo-consistent with the photograph at c.*

Definition 2 (Shape-Radiance Photo-Consistency) *A shape-radiance scene description is photo-consistent with the photograph at c if all points visible from c are photo-consistent and every non-background pixel is the projection of a point in* $\mathcal{V}$.

Definition 3 (Shape Photo-Consistency) *A shape* $\mathcal{V}$ *is photo-consistent with a set of photographs if there is an assignment of radiance functions to the visible points of* $\mathcal{V}$ *that makes the resulting shape-radiance description photo-consistent with all photographs.*

Our goal is to provide a concrete characterization of the family of all scenes that are photo-consistent with N input photographs. We achieve this by making explicit the two ways in which photo-consistency with N photographs can constrain a scene's shape.

2.1. Background Constraints

Photo-consistency requires that no point of $\mathcal{V}$ projects to a background pixel. If a photograph taken at position c contains identifiable background pixels, this constraint restricts $\mathcal{V}$ to a cone defined by c and the photograph's non-background pixels. Given N such photographs, the scene is restricted to the *visual hull*, which is the volume of intersection of their corresponding cones (Laurentini, 1994).

When no *a priori* information is available about the scene's radiance, the visual hull defines all the shape constraints in the input photographs. This is because there is always an assignment of radiance functions to the points on the surface of the visual hull that makes the resulting shape-radiance description photo-consistent with the N input photographs.[7] The visual hull can therefore be thought of as a "least commitment reconstruction" of the scene—any further refinement of this volume must rely on assumptions about the scene's shape or radiance.

While visual hull reconstruction has often been used as a method for recovering 3D shape from photographs (Szeliski, 1993; Kutulakos, 1997), the picture constraints captured by the visual hull only exploit information from the background pixels in these photographs. Unfortunately, these constraints become useless when photographs contain no background pixels (i.e., the visual hull degenerates to $\mathbb{R}^3$) or when background identification (Smith and Blinn, 1996) cannot be performed accurately. Below we study picture constraints from non-background

pixels when the scene's radiance is restricted to a special class of radiance models. The resulting constraints lead to photo-consistent scene reconstructions that are subsets of the visual hull, and unlike the visual hull, can contain concavities.

2.2. Radiance Constraints

Surfaces that are not transparent or mirror-like reflect light in a coherent manner, i.e., the color of light reflected from a single point along different directions is not arbitrary. This coherence provides additional picture constraints beyond what can be obtained from background information. In order to take advantage of these constraints, we focus on scenes whose radiance satisfies the following criteria:

Consistency Check Criteria:

1. An algorithm $\texttt{consist}_K()$ is available that takes as input at least $K \leq N$ colors $col_1, \ldots, col_K$, K vectors $\xi_1, \ldots, \xi_K$, and the light source positions (non-Lambertian case), and decides whether it is possible for a single surface point to reflect light of color col_i in direction ξ_i simultaneously for all $i = 1, \ldots, K$.
2. $\texttt{consist}_K()$ is assumed to be monotonic, i.e., $\texttt{consist}_K(col_1, \ldots, col_j, \xi_1, \ldots, \xi_j)$ implies that $\texttt{consist}_K(col_1, \ldots, col_{j-1}, \xi_1, \ldots, \xi_{j-1})$ for every permutation of $1, \ldots, j$.

Given a shape $\mathcal{V}$, the Consistency Check Criteria give us a way to establish the photo-consistency of every point on $\mathcal{V}$'s surface. These criteria define a general class of radiance models, that we call *locally computable*, that are characterized by a locality property: the radiance at any point is independent of the radiance of all other points in the scene. The class of locally-computable radiance models therefore restricts our analysis to scenes where global illumination effects such as transparency (Szeliski and Golland, 1998), inter-reflection (Forsyth and Zisserman, 1991), and shadows can be ignored. For example, inter-reflection and shadows in Lambertian scenes viewed under fixed illumination are correctly accounted for because scene radiance is isotropic even when such effects are present. As a result, the class of locally-computable radiance models subsumes the Lambertian ($K = 2$) and other parameterized models of scene radiance.[8]

Given an *a priori* locally computable radiance model for the scene, we can determine whether or not a given shape $\mathcal{V}$ is photo-consistent with a collection of photographs. Even more importantly, when the scene's radiance is described by such a model, the *non*-photo-consistency

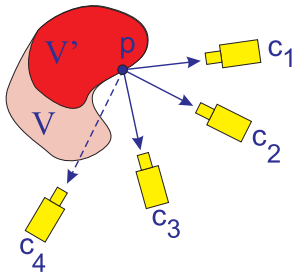


Figure 2. Illustration of the Visibility and Non-Photo-Consistency Lemmas. If p is non-photo-consistent with the photographs at c_1, c_2, c_3, it is non-photo-consistent with the entire set $Vis_{\mathcal{V}'}(p)$, which also includes c_4.

of a shape $\mathcal{V}$ tells us a great deal about the shape of the underlying scene. We use the following two lemmas to make explicit the structure of the family of photo-consistent shapes. These lemmas provide the analytical tools needed to describe how the non-photo-consistency of a shape $\mathcal{V}$ affects the photo-consistency of its subsets (Figure 2):

Lemma 1 (Visibility Lemma) *Let p be a point on $\mathcal{V}$'s surface,* $\mathrm{Surf}(\mathcal{V})$*, and let* $\mathrm{Vis}_{\mathcal{V}}(p)$ *be the collection of input photographs in which $\mathcal{V}$ does not occlude p. If $\mathcal{V}' \subset \mathcal{V}$ is a shape that also has p on its surface,* $\mathrm{Vis}_{\mathcal{V}}(p) \subseteq \mathrm{Vis}_{\mathcal{V}'}(p)$.

Proof. Since $\mathcal{V}'$ is a subset of $\mathcal{V}$, no point of $\mathcal{V}'$ can lie between p and the cameras corresponding to $Vis_{\mathcal{V}}(p)$. QED

Lemma 2 (Non-Photo-Consistency Lemma) *If $p \in \mathrm{Surf}(\mathcal{V})$ is not photo-consistent with a subset of* $\mathrm{Vis}_{\mathcal{V}}(p)$*, it is not photo-consistent with* $\mathrm{Vis}_{\mathcal{V}}(p)$.

Intuitively, Lemmas 1 and 2 suggest that both visibility and non-photo-consistency exhibit a form of "monotonicity:" the Visibility Lemma tells us that the collection of photographs from which a surface point $p \in Surf(\mathcal{V})$ is visible strictly expands as $\mathcal{V}$ gets smaller (Figure 2). Analogously, the Non-Photo-Consistency Lemma, which follows as a direct consequence of the definition of photo-consistency, tells us that each new photograph can be thought of as an additional constraint on the photo-consistency of surface points—the more photographs are available, the more difficult it is for those points to achieve photo-consistency. Furthermore, once a surface point fails to be photo-consistent no new photograph of that point can re-establish photo-consistency.

The key consequence of Lemmas 1 and 2 is given by the following theorem which shows that *non*-photo-consistency at a point rules out the photo-consistency of an entire family of shapes:

Theorem 1 (Subset Theorem) *If $p \in \text{Surf}(\mathcal{V})$ is not photo-consistent, no photo-consistent subset of $\mathcal{V}$ contains p.*

Proof. Let $\mathcal{V}' \subset \mathcal{V}$ be a shape that contains p. Since p lies on the surface of $\mathcal{V}$, it must also lie on the surface of $\mathcal{V}'$. From the Visibility Lemma it follows that $Vis_{\mathcal{V}}(p) \subseteq Vis_{\mathcal{V}'}(p)$. The theorem now follows by applying the Non-Photo-Consistency Lemma to $\mathcal{V}'$ and using the locality property of locally computable radiance models. QED

We explore the ramifications of the Subset Theorem in the next section.

3. The Photo Hull

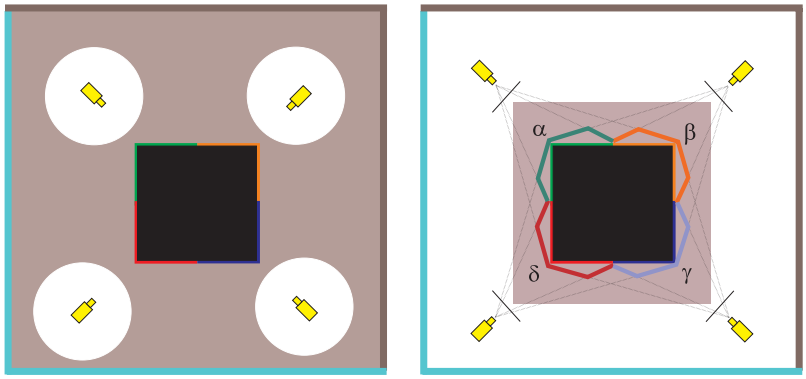
The family of all shapes that are photo-consistent with N photographs defines the ambiguity inherent in the problem of recovering 3D shape from those photographs. When there is more than one photo-consistent shape it is impossible to decide, based on those photographs alone, which photo-consistent shape corresponds to the true scene. This ambiguity raises two important questions regarding the feasibility of scene reconstruction from photographs:

- Is it possible to compute a shape that is photo-consistent with N photographs and, if so, what is the algorithm?
- If a photo-consistent shape can be computed, how can we relate that shape to all other photo-consistent 3D interpretations of the scene?

Before providing a general answer to these questions we observe that when the number of input photographs is finite, the first question can be answered with a trivial shape (Figure 3a). In general, trivial shape solutions such as this one can be eliminated with the incorporation of *free space* constraints, i.e., regions of space that are known not to contain scene points. Our analysis enables the (optional) inclusion of such constraints by specifying an arbitrary set $\mathcal{V}$ within which a photo-consistent shape is known to lie.[9]

In particular, our answers to both questions rest on the following theorem. Theorem 2 shows that for any shape $\mathcal{V}$ there is a unique photo-consistent shape that subsumes, i.e., contains within its volume, all other photo-consistent shapes in $\mathcal{V}$ (Figure 3b):

Theorem 2 (Photo Hull Theorem) *Let $\mathcal{V}$ be an arbitrary subset of $\mathbb{R}^3$. If $\mathcal{V}^*$ is the union of all photo-consistent shapes in $\mathcal{V}$, every point on the surface of $\mathcal{V}^*$ is photo-consistent. We call $\mathcal{V}^*$ the* photo hull.[10]



(a) (b)

Figure 3. Photo-consistent shapes for a two-dimensional scene viewed by four cameras. The scene consists of a black square whose sides are painted diffuse red, blue, orange, and green. (a) Trivial shape solutions in the absence of free-space constraints. Carving out a small circle around each camera and projecting the image onto the interior of that circle yields a trivial photo-consistent shape, shown in gray. (b) Illustration of the Photo Hull Theorem. The gray-shaded region corresponds to an arbitrary shape $\mathcal{V}$ containing the square in (a). $\mathcal{V}^*$ is a polygonal region that extends beyond the true scene and whose boundary is defined by the polygonal segments α, β, γ, and δ. When these segments are colored as shown, $\mathcal{V}^*$'s projections are indistinguishable from that of the true object and *no* photo-consistent shape in the gray-shaded region can contain points outside $\mathcal{V}^*$.

Proof. (By contradiction) Suppose that p is a surface point on $\mathcal{V}^*$ that is not photo-consistent. Since $p \in \mathcal{V}^*$, there exists a photo-consistent shape, $\mathcal{V}' \subset \mathcal{V}^*$, that also has p on its surface. It follows from the Subset Theorem that $\mathcal{V}'$ is not photo-consistent. QED

Corollary 1 *If $\mathcal{V}^*$ is closed, it is a photo-consistent shape.*

Theorem 2 provides an explicit relation between the photo hull and all other possible 3D interpretations of the scene: the theorem guarantees that every such interpretation is a subset of the photo hull. The photo hull therefore represents a least-commitment reconstruction of the scene.

While every point on the photo hull is photo-consistent, the hull itself is not guaranteed to be closed, i.e., it may not satisfy our definition of a *shape*. Specific cases of interest where $\mathcal{V}^*$ is closed include (1) discretized scene volumes, i.e., scenes that are composed of a finite number of volume elements, and (2) instances where the number of photo-consistent shapes in a volume is finite. We describe a volumetric algorithm for computing discretized photo hulls in the next section.

The general case, where the photo hull is an infinite union of shapes, is considered in the Appendix.

4. Reconstruction by Space Carving

An important feature of the photo hull is that it can be computed using a simple, discrete algorithm that "carves" space in a well-defined manner. Given an initial volume $\mathcal{V}$ that contains the scene, the algorithm proceeds by iteratively removing (i.e. "carving") portions of that volume until it converges to the photo hull, $\mathcal{V}^*$. The algorithm can therefore be fully specified by answering four questions: (1) how do we select the initial volume $\mathcal{V}$, (2) how should we represent that volume to facilitate carving, (3) how do we carve at each iteration to guarantee convergence to the photo hull, and (4) when do we terminate carving?

The choice of the initial volume has a considerable impact on the outcome of the reconstruction process (Figure 3). Nevertheless, selection of this volume is beyond the scope of this paper; it will depend on the specific 3D shape recovery application and on information about the manner in which the input photographs were acquired.[11] Below we consider a general algorithm that, given N photographs and *any* initial volume that contains the scene, is guaranteed to find the (unique) photo hull contained in that volume.

In particular, let $\mathcal{V}$ be an arbitrary finite volume that contains the scene as an unknown sub-volume. Also, assume that the surface of the true scene conforms to a radiance model defined by a consistency check algorithm $\texttt{consist}_K()$. We represent $\mathcal{V}$ as a finite collection of voxels $v_1, \ldots, v_M$. Using this representation, each carving iteration removes a single voxel from $\mathcal{V}$.

The Subset Theorem leads directly to a method for selecting a voxel to carve away from $\mathcal{V}$ at each iteration. Specifically, the theorem tells us that if a voxel v on the surface of $\mathcal{V}$ is not photo-consistent, the volume $\mathcal{V} = \mathcal{V} - \{v\}$ must still contain the photo hull. Hence, if only non-photo-consistent voxels are removed at each iteration, the carved volume is guaranteed to converge to the photo hull. The order in which non-photo-consistent voxels are examined and removed is not important for guaranteeing correctness. Convergence to this shape occurs when no non-photo-consistent voxel can be found on the surface of the carved volume. These considerations lead to the following algorithm for computing the photo hull:[12]

Space Carving Algorithm

Step 1: Initialize $\mathcal{V}$ to a volume containing the true scene.

Step 2: Repeat the following steps for voxels $v \in \mathit{Surf}(\mathcal{V})$ until a non-photo-consistent voxel is found:

a. Project v to all photographs in $\mathit{Vis}_{\mathcal{V}}(v)$. Let $\mathit{col}_1, \ldots, \mathit{col}_j$ be the pixel colors to which v projects and let $\xi_1, \ldots, \xi_j$ be the optical rays connecting v to the corresponding optical centers.

b. Determine the photo-consistency of v using $\texttt{consist}_K(\mathit{col}_1, \ldots, \mathit{col}_j, \xi_1, \ldots, \xi_j)$.

Step 3: If no non-photo-consistent voxel is found, set $\mathcal{V}^* = \mathcal{V}$ and terminate. Otherwise, set $\mathcal{V} = \mathcal{V} - \{v\}$ and repeat Step 2.

The key step in the algorithm is the search and voxel consistency checking of Step 2. The following proposition gives an upper bound on the number of voxel photo-consistency checks:

Proposition 1 *The total number of required photo-consistency checks is bounded by $N * M$ where N is the number of input photographs and M is the number of voxels in the initial (i.e., uncarved) volume.*

Proof. Since (1) the photo-consistency of a voxel v that remains on $\mathcal{V}$'s surface for several carving iterations can change only when $\mathit{Vis}_{\mathcal{V}}(v)$ changes due to $\mathcal{V}$'s carving, and (2) $\mathit{Vis}_{\mathcal{V}}(v)$ expands monotonically as $\mathcal{V}$ is carved (Visibility Lemma), the photo-consistency of v must be checked at most N times. QED

5. A Multi-Sweep Implementation of Space Carving

Despite being relatively simple to describe, the Space Carving Algorithm as described in Section 4 requires a difficult update procedure because of the need to keep track of scene visibility from all of the input cameras. In particular, every time a voxel is carved a new set of voxels becomes newly visible and must be re-evaluated for photo-consistency. Keeping track of such changes necessitates computationally-expensive ray-tracing techniques or memory-intensive spatial data structures (Culbertson et al., 1999). To overcome these problems, we instead describe a multi-sweep implementation of the Space Carving Algorithm that enables efficient visibility computations with minimal memory requirements.

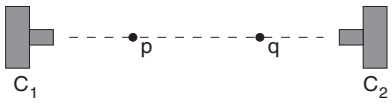


Figure 4. A Visibility Cycle. Voxel p occludes q from c_1, whereas q occludes p from c_2. Hence, no visibility order exists that is the same for both cameras.

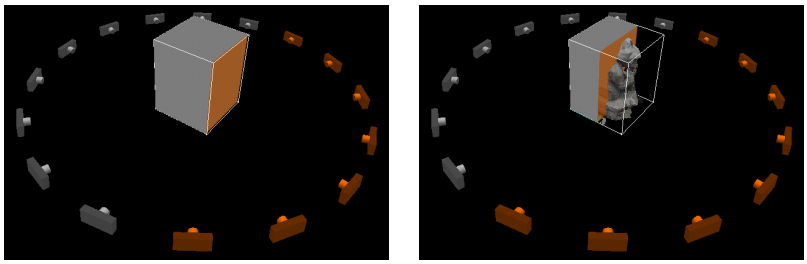
5.1. Multi-view Visibility Ordering

A convenient method of keeping track of voxel visibility is to evaluate voxels *in order of visibility*, i.e., visit occluders before the voxels that they occlude. The key advantage of this approach is that backtracking is avoided—carving a voxel affects only voxels encountered later in the sequence. For a single camera, visibility ordering amounts to visiting voxels in a front-to-back order and may be accomplished by depth-sorting (Newell et al., 1972; Fuchs et al., 1980). The problem of defining visibility orders that apply simultaneously to *multiple* cameras is more difficult, however, because it requires that voxels occlude each other in the same order from different viewpoints. More precisely, voxel p is evaluated before q only if q does not occlude p from *any one* of the input viewpoints.

It is known that multi-view visibility orders exist for cameras that lie on one side of a plane (Langer and Zucker, 1994). Recently, Seitz and Dyer (Seitz and Dyer, 1999) generalized this case to a range of interesting camera configurations by showing that multi-view visibility orders always exist when the scene lies outside the convex hull of the camera centers. When this constraint is satisfied, evaluating voxels in order of increasing distance to this camera hull yields a multi-view visibility order that may be used to reconstruct the scene. The convex hull constraint is a significant limitation, however, because it strongly restricts the types of scenes and range of views that are reconstructible. In fact, it can be readily shown that no multi-view visibility constraint exists in general (Fig. 4). Therefore, different techniques are needed in order to reconstruct scenes like Fig. 4 that violate the convex hull constraint.

5.2. Plane-Sweep Visibility

While multi-view visibility orders do not exist in the general case, it *is* possible to define visibility orders that apply to a subset of the input cameras. In particular, consider visiting voxels in order of increasing X coordinate and, for each voxel $p = (X_p, Y_p, Z_p)$, consider only cameras whose X coordinates are less than X_p. If p occludes q from a camera



(a) (b)

Figure 5. Plane-Sweep Visibility. The plane-sweep algorithm ensures that voxels are visited in order of visibility with respect to all active cameras. The current plane and active set of cameras is shown in orange. (b) The shape evolves and new cameras become active as the plane moves through the scene volume.

at c, it follows that p is on the line segment cq and therefore $X_p < X_q$. Consequently, p is evaluated before q, i.e., occluders are visited before the voxels that they occlude.

Given this ordering strategy, the Space Carving Algorithm can be implemented as a multi-sweep volumetric algorithm in which a solid block of voxels is iteratively carved away by sweeping a single plane through the scene along a set of pre-defined sweep directions (Fig. 5). For each position of the plane, voxels on the plane are evaluated by considering their projections into input images from viewpoints on one side of the plane. In the above example, a plane parallel to the Y-Z axis is swept in the increasing X direction.

Plane Sweep Algorithm

Step 1: Given an initial volume $\mathcal{V}$, initialize the sweep plane Π such that $\mathcal{V}$ lies below Π (i.e., Π is swept towards $\mathcal{V}$).

Step 2: Intersect Π with the current shape $\mathcal{V}$.

Step 3: For each surface voxel v on Π:

- **a.** let $c_1, \ldots, c_j$ be the cameras above Π for which v projects to an *unmarked* pixel;
- **b.** determine the photo-consistency of v using $\texttt{consist}_K(col_1, \ldots, col_j, \xi_1, \ldots, \xi_j)$;
- **c.** if v is inconsistent then set $\mathcal{V} = \mathcal{V} - \{v\}$, otherwise mark the pixels to which v projects.

Step 4: Move Π downward one voxel width and repeat Step 2 until $\mathcal{V}$ lies above Π.

The dominant costs of this algorithm are (1) projecting a plane of voxels into N images, and (2) correlating pixels using $\texttt{consist}_K(col_1, \ldots, col_j, \xi_1, \ldots, \xi_j)$. Our implementation exploits texture-mapping graphics hardware (the kind found on standard PC graphics cards) to project an entire plane of voxels at a time onto each image. We have found that when this optimization is used, the pixel correlation step dominates the computation.

5.3. Multi-Sweep Space Carving

The Plane Sweep Algorithm considers only a subset of the input cameras for each voxel, i.e., the cameras on one side of the sweep plane. Consequently, it may fail to carve voxels that are inconsistent with the entire set of input images but are consistent with a proper subset of these images. To ensure that all cameras are considered, we repeatedly perform six sweeps through the volume, corresponding to the six principle directions (increasing and decreasing X, Y, and Z directions). Furthermore, to guarantee that all cameras visible to a voxel are taken into account, we perform an additional round of voxel consistency checks that incorporate the voxel visibility information collected from individual sweeps. The complete algorithm is as follows:

Multi-Sweep Space Carving Algorithm

Step 1: Initialize $\mathcal{V}$ to be a superset of the true scene.

Step 2: Apply the Plane Sweep Algorithm in each of the six principle directions and update $\mathcal{V}$ accordingly.

Step 3: For every voxel in $\mathcal{V}$ whose consistency was evaluated in more than one plane sweep:

- **a.** let $c_1, \ldots, c_j$ be the cameras that participated in the consistency check of v in *some* plane sweep during Step 2;
- **b.** determine the photo-consistency of v using $\texttt{consist}_K(col_1, \ldots, col_j, \xi_1, \ldots, \xi_j)$.

Step 4: If no voxels were removed from $\mathcal{V}$ in Steps 2 and 3, set $\mathcal{V}^* = \mathcal{V}$ and terminate; otherwise, repeat Step 2.

5.4. Lambertian Scenes

We give special attention to case of Lambertian scenes, in which the Consistency Check Criteria can be defined using the standard deviation

of colors, $col_1, \ldots, col_K$, at a voxel's projection. To account for errors in the image formation process due to quantization, calibration, or other effects, we call a voxel *photo-consistent* if σ is below a given threshold. This threshold is chosen by considering σ to be a statistical measure of voxel photo-consistency. In particular, suppose the sensor error (accuracy of irradiance measurement) is normally distributed[13] with standard deviation σ_0. The photo-consistency of a voxel v can be estimated using the likelihood ratio test, distributed as χ^2 with $K-1$ degrees of freedom (Freund, 1992):

$$\lambda_v = \frac{(K-1)\sigma^2}{\sigma_0^2}. \tag{1}$$

This formulation of the Consistency Check Criterion allows us to incorporate two additional optimizations to the Multi-Sweep Carving Algorithm. First, we maintain sufficient per-voxel color statistics between sweeps to integrate information from all input images, therefore eliminating the need for Step 3 of the multi-sweep algorithm. This is because the standard deviation of K monochrome pixel values of intensity col_i, can be computed using the following recursive formula:

$$\sigma^2 = \frac{1}{K}\left(\sum_{i=1}^{K} col_i^2 - \sum_{i=1}^{K} col_i\right). \tag{2}$$

It is therefore sufficient to maintain three numbers per voxel, namely $\sum_{i=1}^{K} col_i$, $\sum_{i=1}^{K} col_i^2$, and K (i.e., seven numbers for three-component color pixels). Second, to ensure that no camera is considered more than once per voxel in the six sweeps, we further restrict the cameras considered in each sweep to a pyramidal beam defined by the voxel center and one of its faces, as shown in Fig. 6. This strategy partitions the cameras into six non-overlapping sets to be processed in the six respective sweeps, thereby ensuring that each camera is considered exactly once per voxel during the six sweeps.

6. 3D Photography by Space Carving

6.1. Image Acquisition

In the Space Carving Algorithm, every input photograph can be thought of as a *shape constraint* that forces the reconstructed scene volume to contain only voxels consistent with the photograph. To ensure that the algorithm's output closely resembles the shape and appearance of

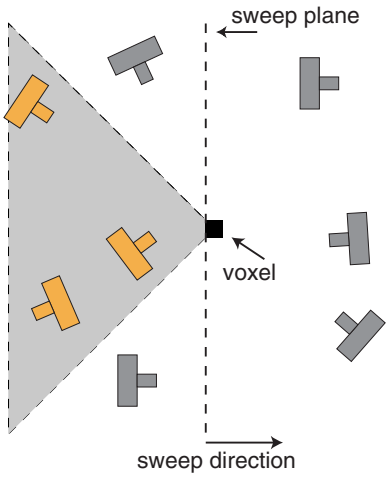


Figure 6. To ensure that a camera is processed at most once per voxel during the six plane sweeps, the set of cameras considered in each sweep is clipped to a pyramidal beam defined by the center of the voxel and one of its faces.

a complicated 3D scene it is therefore important to acquire enough photographs of the scene itself. In a typical image acquisition session, we take between 10 and 100 calibrated images around the scene of interest using a Pulnix TMC-9700 color CCD camera (Fig. 7).

A unique property of the Space Carving Algorithm is that it can be forced to automatically segment a 3D object of interest from a larger scene using two complementary methods. The first method, illustrated in the sequence of Fig. 7, involves slightly modifying the image acquisition process—before we take a photograph of the object of interest from a new viewpoint, we manually alter the object's background. This process enabled segmentation and complete reconstruction of the gargoyle sculpture; the Space Carving Algorithm effectively removed all background pixels in all input photographs because the varying backgrounds ensured that photo-consistency could not be enforced for points projecting to non-object pixels. Note that image subtraction or traditional matting techniques (Smith and Blinn, 1996) cannot be applied to this image sequence to segment the sculpture since every photograph was taken from a *different* position in space and therefore the background is different in each image. The second method, illustrated in Fig. 9, involves defining an initial volume $\mathcal{V}$ (e.g., a bounding box) that is "tight enough" to ensure reconstruction of only the object of interest. This process enabled segmentation of the hand because the initial volume did not intersect distant objects such as the TV monitor.

6.2. Reconstruction Results

In this section we present results from applying our Multi-Sweep implementation of the Space Carving Algorithm to a variety of image sequences. In all examples, a Lambertian model was used for the Consistency Check Criterion, i.e., it was assumed that a voxel projects to pixels of the same color in every image. The standard deviation of these pixels was therefore used to determine whether or not a voxel should be carved, as described in Section 5.

We first ran the Space Carving Algorithm on 16 images of a gargoyle sculpture (Fig. 7). The sub-pixel calibration error in this sequence enabled using a small threshold of 6% for the RGB component error. This threshold, along with the voxel size and the 3D coordinates of a bounding box containing the object were the only parameters given as input to our implementation. Fig. 8 shows selected input images and new views of the reconstruction. This reconstruction consisted of 215 thousand surface voxels that were carved out of an initial volume of approximately 51 million voxels. It took 250 minutes to compute on an SGI O2 R10000/175MHz workstation. Some errors are still present in the reconstruction, notably holes that occur as a result of shadows and other illumination changes due to the object's rotation inside a static, mostly diffuse illumination environment. These effects were not modeled by the Lambertian model and therefore caused voxels on shadowed surfaces to be carved. The finite voxel size, calibration error, and image discretization effects resulted in a loss of some fine surface detail. Voxel size could be further reduced with better calibration, but only up to the point where image discretization effects (i.e., finite pixel size) become a significant source of error.

Results from a sequence of one hundred images of a hand are shown in Figs. 9 and 10. Note that the near-perfect segmentation of the hand from the rest of the scene was performed not in image-space, but in 3D object space—the background lay outside the initial block of voxels and was therefore not reconstructed. This method of 3D background segmentation has significant advantages over image subtraction and chroma-keying methods because it (1) does not require the background to be known and (2) will never falsely eliminate foreground pixels, as these former techniques are prone to do (Smith and Blinn, 1996).

Two kinds of artifacts exist in the resulting reconstructions. First, voxels that are not visible from any input viewpoint do not have a well-defined color assignment and are given a default color. These artifacts can be eliminated by acquiring additional photographs to provide adequate coverage of the scene's surfaces. Second, stray voxels may be reconstructed in unoccupied regions of space due to accidental agree-

Figure 7. Nine of sixteen 486x720 RGB images of a gargoyle stone sculpture. The sequence corresponds to a complete circumnavigation of the object, performed in approximately 22.5 degree increments.

ments between the input images. Such artifacts can be easily avoided by re-applying the Space Carving Algorithm on an initial volume that does not contain those regions or by post-filtering the reconstructed voxel model.

In a final experiment, we applied our algorithm to images of a synthetic building scene rendered from both its interior and exterior (Figure 11). This placement of cameras yields an extremely difficult stereo problem, due to the drastic changes in visibility between interior and exterior cameras.[14] Figure 11 compares the original model and the reconstruction from different viewpoints. The model's appearance is very good near the input viewpoints, as demonstrated in Figs. 11b-c. Note that the reconstruction tends to "bulge" out and that the walls are not perfectly planar (Figure 11e). This behavior is exactly as predicted by Theorem 2—the algorithm converges to the *largest possible* shape that is consistent with the input images. In low-contrast regions where

Figure 8. Reconstruction of a gargoyle sculpture. One of 16 input images is shown (a), along with views of the reconstruction from the same (b) and new (c-d) viewpoints.

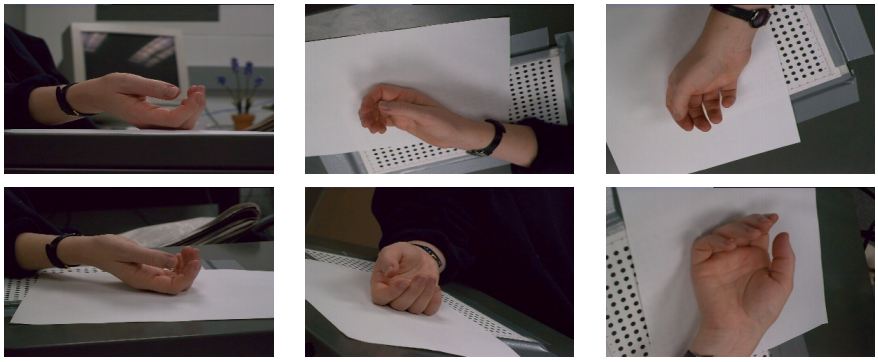


Figure 9. Six out of one hundred photographs of a hand sequence.

shape is visually ambiguous, this causes significant deviations between the computed photo hull and the true scene. While these deviations do not adversely affect scene appearance near the input viewpoints, they can result in noticeable artifacts for far-away views. These deviations and the visual artifacts they cause are easily remedied by including images from a wider range of camera viewpoints to further constrain the scene's shape, as shown in Figure 11f.

Our experiments highlight a number of advantages of our approach over previous techniques. Existing multi-baseline stereo techniques (Okutomi and Kanade, 1993) work best for densely textured scenes and suffer in the presence of large occlusions. In contrast, the hand sequence contains many low-textured regions and dramatic changes in visibility. The low-texture and occlusion properties of such scenes cause problems for feature-based structure-from-motion methods (Tomasi and Kanade, 1992; Seitz and Dyer, 1995; Beardsley et al., 1996; Pollefeys et al., 1998), due to the difficulty of locating and tracking a sufficient number of features throughout the sequence. While contour-based techniques like volume intersection (Martin and Aggarwal, 1983; Szeliski, 1993) often work well for similar scenes, they require detecting silhouettes or occluding contours. For the gargoyle sequence, the background was unknown and heterogeneous, making the contour detection problem extremely difficult. Note also that Seitz and Dyer's voxel coloring technique (Seitz and Dyer, 1999) would not work for any of the above sequences because of the constraints it imposes on camera placement. Our approach succeeds because it integrates both texture and contour information as appropriate, without the need to explicitly detect features or contours, or constrain viewpoints. Our results indicate the approach is highly effective for both densely textured and untextured objects and scenes.

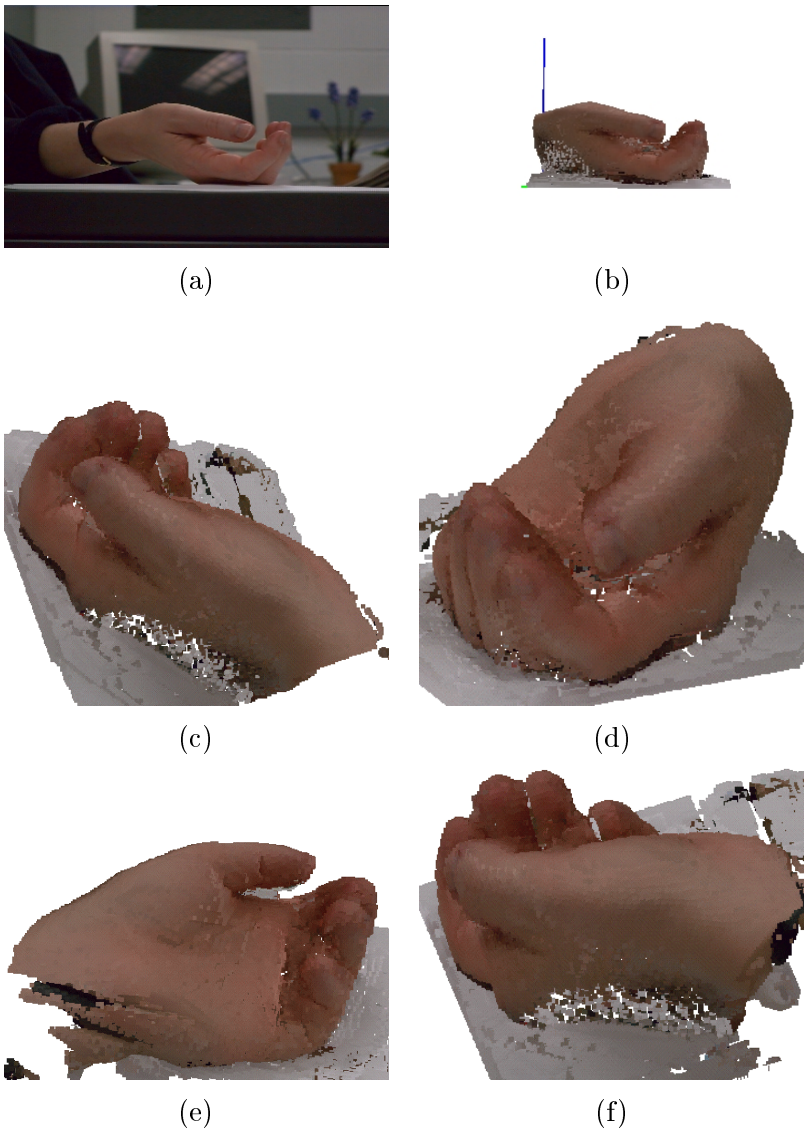


Figure 10. Reconstruction of a hand. An input image is shown in (a) along with views of the reconstruction from the same (b) and other (d-f) viewpoints. The reconstructed model was computed using an RGB component error threshold of 6%. The model has 112 thousand voxels and took 53 seconds to compute. The blue line in (b) indicates the z-axis of the world coordinate system.

7. Concluding Remarks

This paper introduced *photo-consistency theory* as a new, general mathematical framework for analyzing the 3D shape recovery problem from multiple images. We have shown that this theory leads to a "least commitment" approach for shape recovery and a practical algorithm called Space Carving that together overcome several limitations in the current state of the art. First, the approach allows us to analyze and characterize the set of all possible reconstructions of a scene, without placing constraints on geometry, topology, or camera configuration. Second, this is the only provably-correct method, to our knowledge, capable of reconstructing non-smooth, free-form shapes from cameras positioned and oriented in a completely arbitrary way. Third, the performance of the Space Carving Algorithm was demonstrated on real and synthetic image sequences of geometrically-complex objects, including a large building scene photographed from both interior and exterior viewpoints. Fourth, the use of photo-consistency as a criterion for 3D shape recovery enables the development of reconstruction algorithms that allow faithful image reprojections and resolve the complex interactions between occlusion, parallax, and shading effects in shape analysis.

While the Space Carving Algorithm's effectiveness was demonstrated in the presence of low image noise, the photo-consistency theory itself is based on an idealized model of image formation. Extending the theory to explicitly model image noise, quantization and calibration errors, and their effects on the photo hull is an open research problem (Kutulakos, 2000). Extending the formulation to handle non-locally computable radiance models (e.g., shadows and inter-reflections) is another important topic of future work. Other research directions include (1) developing space carving algorithms for images with significant pixel noise, (2) investigating the use of surface-based rather than voxel-based techniques for finding the photo hull, (3) incorporating *a priori* shape constraints (e.g., smoothness), and (4) analyzing the topological structure of the set of photo-consistent shapes. Finally, an on-line implementation of the Space Carving Algorithm, that performs image capture and scene reconstruction simultaneously, would be extremely useful both to facilitate the image acquisition process and to eliminate the need to store long video sequences.

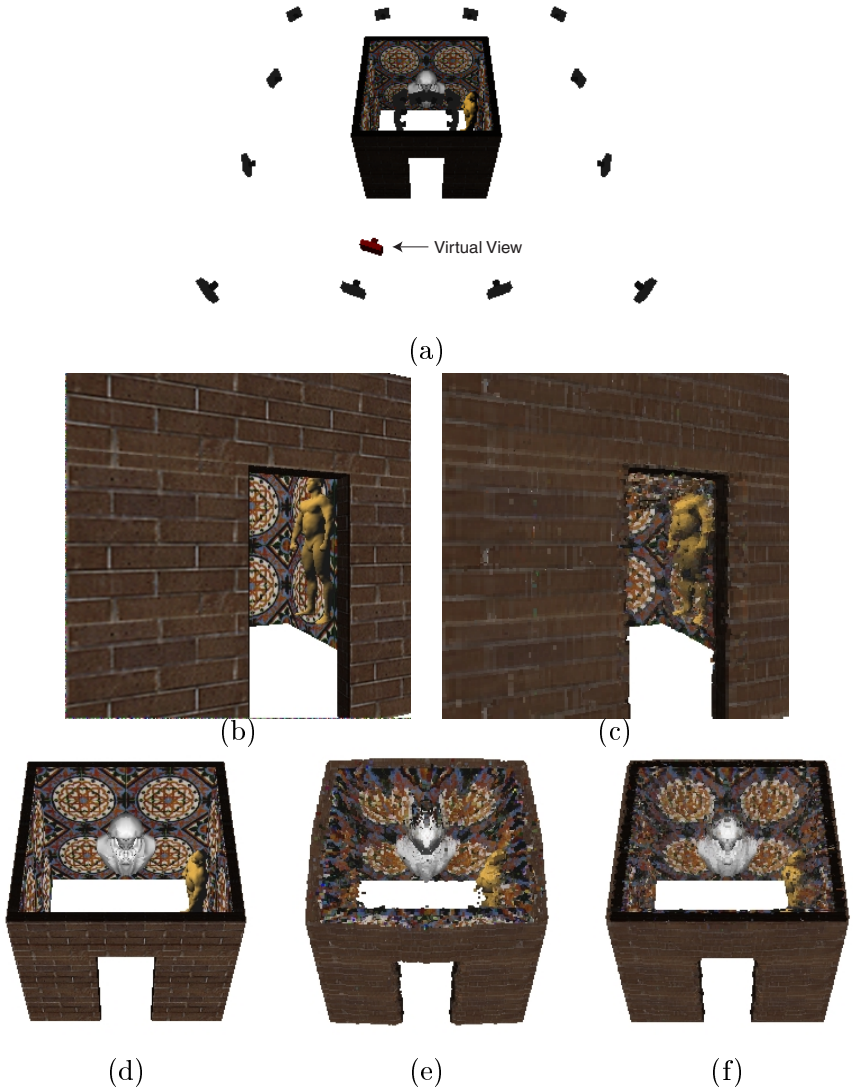


Figure 11. Reconstruction of a synthetic building scene. (a) 24 Cameras were placed in both the interior and exterior of a building to enable simultaneous, complete reconstruction of its exterior and interior surfaces. The reconstruction contains 370,000 voxels, carved out of a $200 \times 170 \times 200$ voxel block. (b) A rendered image of the building for a viewpoint near the input cameras (shown as "virtual view" in (a)) is compared to the view of the reconstruction (c). (d-f) Views of the reconstruction from far away camera viewpoints. (d) shows a rendered top view of the original building, (e) the same view of the reconstruction, and (f) a new reconstruction resulting from adding image (d) to the set of input views. Note that adding just a single top view dramatically improves the quality of the reconstruction.

Acknowledgements

The authors would like to thank Prof. Terry Boult for many discussions and suggestions on the technical aspects of the paper's proofs. Kiriakos Kutulakos gratefully acknowledges the support of the National Science Foundation under Grant No. IIS-9875628, of Roche Laboratories, Inc., and of the Dermatology Foundation. Part of this work was conducted while Steven Seitz was employed by the Vision Technology Group at Microsoft Research. The support of the Microsoft Corporation is gratefully acknowledged.

Appendix

In general, the photo hull, $\mathcal{V}^*$, of a set $\mathcal{V}$ is the union of a potentially infinite collection of shapes in $\mathcal{V}$. Such unions do not always correspond to a closed subset of $\mathbb{R}^3$ (Armstrong, 1983). As a result, even though all points of the photo hull are photo-consistent, the photo hull itself may not satisfy the definition of a 3D shape given in Section 2. In this Appendix we investigate the properties of the closure, $\overline{\mathcal{V}^*}$, of $\mathcal{V}^*$ which is always a valid shape.[15] In particular, we show that $\overline{\mathcal{V}^*}$ satisfies a slightly weaker form of photo-consistency called *directional ϵ-photo-consistency*, defined below. This property leads to a generalization of Theorem 2:

Theorem 3 (Closed Photo Hull Theorem) *Let $\mathcal{V}$ be an arbitrary shape in $\mathbb{R}^3$ and let $\overline{\mathcal{V}^*}$ be the closure of the union of all photo-consistent shapes in $\mathcal{V}$. The shape $\overline{\mathcal{V}^*}$ is directionally ϵ-photo-consistent and is called the* closed photo hull.

A.1. The Strong Visibility Condition

Because we impose no constraints on the structure of the photo-consistent shapes in $\mathcal{V}$ that are considered in our analysis (e.g., smoothness), it is possible to define degenerate shapes that defy one's "intuitive" notions of visibility and occlusion. More specifically, the standard definition of visibility of a surface point p from a camera c requires that the open line segment pc does not intersect the shape itself; otherwise, p is defined to be occluded. When $\mathcal{V}$ is arbitrary, however, it is possible to define shapes whose surface gets infinitesimally close to this line segment at one or more points other than p. Intuitively, surface points that have this property are not occluded under the above definition but are not

"fully visible" either. We therefore refine the notion of visibility in a way that excludes such degeneracies. In particular, let $B(p,\epsilon) \subset \mathbb{R}^3$ be the open 3-ball of radius ϵ that is centered at p:

Definition 4 (Strong Visibility Condition) *A point p on the surface of a shape $\mathcal{V}$ is strongly visible to a set of cameras if it is visible from those cameras and if, for every $\epsilon > 0$, there exists a closed set N and an $\epsilon' < e$ such that the following two properties are satisfied:*

1. *N contains all its occluders, i.e., for every camera c and point $p \in N$, if q occludes p from c then $q \in N$, and*
2. *$B(p,\epsilon') \subset N \subset B(p,\epsilon)$.*

Intuitively, the strong visibility condition is equivalent to the standard definition of point visibility for shapes that are "well-behaved"—it differs from this definition only in cases where the ray from point p to a camera comes arbitrarily close to the shape outside p's neighborhood. An illustration of a strong visibility neighborhood N is given in Fig. 12b.

A.2. Directional ϵ-Photo-Consistency

When $\overline{\mathcal{V}^*}$ and $\mathcal{V}^*$ are not equal, the closed photo hull will contain limit points that do not belong to any photo-consistent subset of $\mathcal{V}$. These limit points are not always photo-consistent (Fig. 12a). Fortunately, even though the photo-consistency of these points cannot be guaranteed, these points (as well as the rest of $\overline{\mathcal{V}^*}$) do satisfy the directional ϵ-photo-consistency property:

Definition 5 (Strongly Visible Camera Set) *If $p \in \mathcal{V}$, Π_p is a plane through p, and C is the set of cameras in $\mathrm{Vis}_{\mathcal{V}}(p)$ that are strictly above Π_p, define*

$$\mathrm{SVis}_{\mathcal{V}}(\Pi_p) = \begin{cases} C & \text{if } p \text{ is strongly visible to } C, \\ \emptyset & \text{otherwise.} \end{cases} \tag{3}$$

Definition 6 (Directional Point Photo-Consistency) *A point p in $\mathcal{V}$ is directionally photo-consistent if for every oriented plane Π_p through p, the point p is photo-consistent with all cameras in $\mathrm{SVis}_{\mathcal{V}}(\Pi_p)$.*

Definition 7 (Directional ϵ-photo-consistency) *A point p in $\mathcal{V}$ is directionally ϵ-photo-consistent if for every $\epsilon > 0$ and every oriented plane Π_p through p, there exists a point $q \in B(p,e)$ that is photo-consistent with all cameras in $\mathrm{SVis}_{\mathcal{V}}(\Pi_p)$.*

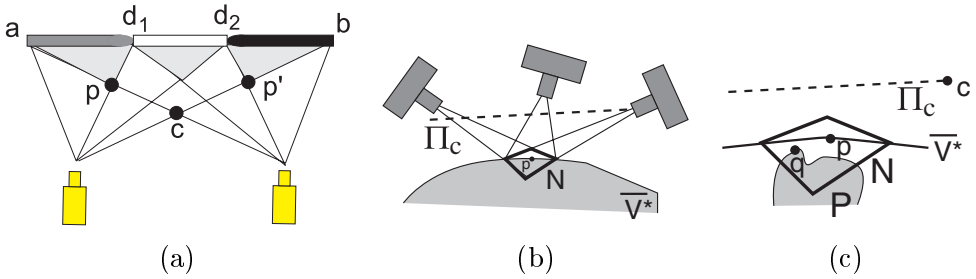


Figure 12. (a) Non-photo-consistent points on the closed photo hull. The 2D scene is composed of a closed thick line segment ab that is painted gray, white, and black. The points d_1, d_2, corresponding to color transitions, are painted white. When $\mathcal{V}$ is defined by the triangle abc, the closed photo hull, $\overline{\mathcal{V}^*}$, is defined by the region shown in light gray. Note that even though $p \in \overline{\mathcal{V}^*}$ is directionally ϵ-photo-consistent, it is not photo-consistent: p projects to a white pixel in the left camera and a gray pixel in the right one. (b)-(c) Proof of Theorem 3. (b) A point p is strongly visible to three cameras by means of neighborhood N. (c) The closest point $q \in N \cap \mathcal{P}$ to Π_c is visible to all cameras on or above Π_c.

Compared to the definition of point photo-consistency (Definition 1), directional photo-consistency relaxes the requirement that p's radiance assignment must agree with all visible cameras. Instead, it requires the ability to find radiance assignment(s) that force agreement only with visible cameras within the same half-space. Directional ϵ-photo-consistency goes a step further, lifting the requirement that every surface point p must have a directionally consistent radiance assignment. The only requirement is that p is infinitesimally close to a point for which directional consistency can be established with respect to the cameras from which p is strongly visible.

Despite their differences, photo-consistency and directional ϵ-photo-consistency share a common characteristic: we can determine whether or not these properties hold for a given shape $\mathcal{V}$ without having *any* information about the photo-consistent shapes contained in $\mathcal{V}$. This is especially important when attempting to characterize $\overline{\mathcal{V}^*}$ because it establishes a direct link between $\overline{\mathcal{V}^*}$ and the image observations that does not depend on explicit knowledge of the family of photo-consistent shapes.

A.3. Proof of Theorem 3

Since points that are not strongly visible are always directionally ϵ-photo-consistent, it is sufficient to consider only strongly visible points $p \in \overline{\mathcal{V}^*}$. More specifically, it suffices to show that every open ball, $B(p, \epsilon)$, contains a point q on some photo-consistent shape $\mathcal{P}$ such that

the set $Vis_{\mathcal{P}}(q)$ contains all cameras in $SVis_{\overline{\mathcal{V}^*}}(\Pi_p)$. For if q is photo-consistent with $Vis_{\mathcal{P}}(q)$, it follows that q is photo-consistent with any of its subsets.

We proceed by first choosing a photo-consistent shape $\mathcal{P}$ and then constructing the point q (Figs. 12b,c). In particular, let c be a camera in $SVis_{\overline{\mathcal{V}^*}}(\Pi_p)$ that is closest to Π_p, and let Π_c be the plane through c that is parallel to Π_p. Fix ϵ such that $0 < \epsilon < k$, where k is the distance from c to Π_p.

Let $N \subset B(p, \epsilon)$ be a set that establishes p's strong visibility according to Definition 4. According to the definition, N contains an open ball $B(p, \epsilon')$ for some $\epsilon' < \epsilon$. By the definition of the photo hull, there exists a photo-consistent shape $\mathcal{P}$ that intersects $B(p, \epsilon')$.

We now construct point q and consider the set of cameras from which q is visible. Let q be a point in the set $\mathcal{P} \cap N$ that minimizes perpendicular distance to Π_c.[16] By construction, no point in $N \cap \mathcal{P}$ occludes q from the cameras in $SVis_{\overline{\mathcal{V}^*}}(\Pi_p)$. Moreover, since $q \in N$, Definition 4 tells us that no point in $\mathcal{P} - N$ can occlude q from the cameras in $SVis_{\overline{\mathcal{V}^*}}(\Pi_p)$. It follows that $Vis_{\mathcal{P}}(q) \supseteq SVis_{\overline{\mathcal{V}^*}}(\Pi_p)$. QED

Notes

[1] Examples include the use of the small baseline assumption in stereo to simplify correspondence-finding and maximize joint visibility of scene points (Kanade et al., 1996), the availability of easily-detectable image contours in shape-from-contour reconstruction (Vaillant and Faugeras, 1992), and the assumption that all views are taken from the same viewpoint in photometric stereo (Woodham et al., 1991).

[2] Faugeras (Faugeras, 1998) has recently proposed the term *metameric* to describe such shapes, in analogy with the term's use in the color perception (Alfvin and Fairchild, 1997) and structure-from-motion literature (van Veen and Werkhoven, 1996).

[3] Note that both of these generalizations represent significant improvements in the state of the art. For instance, silhouette-based algorithms require identification of silhouettes, fail at surface concavities, and treat only the case of binary images. While (Seitz and Dyer, 1999; Seitz and Kutulakos, 1998) also used a volumetric algorithm, their method worked only when the scene was outside the convex hull of the cameras. This restriction strongly limits the kinds of environments that can be reconstructed, as discussed in Section 6.

[4] More formally, we use the term *shape* to refer to any closed set $\mathcal{V} \subseteq I\!R^3$ for which every point $p \in \mathcal{V}$ is infinitesimally close to an open 3-ball inside $\mathcal{V}$. That is, for every $\epsilon > 0$ there is an open 3-ball, $B(p, \epsilon)$, that contains an open 3-ball lying inside $\mathcal{V}$. Similarly, we define the *surface* of $\mathcal{V}$ to be the set of points in $\mathcal{V}$ that are infinitesimally close to a point outside $\mathcal{V}$.

[5] Note that even points on a radiance discontinuity must have a unique radiance function assigned to them. For example, in the scene of Fig. 3, the point of transition between red and blue surface points must be assigned either a red or a blue color.

[6] In the following, we make the simplifying assumption that pixel values in the image measure scene radiance directly.

[7] For example, set $rad_p(\vec{pc})$ equal to the color at p's projection.

[8] Strictly speaking, locally-computable radiance models cannot completely account for surface normals and other neighborhood-dependent quantities. However, it is possible to estimate surface normals based purely on radiance information and thereby approximately model cases where the light source changes (Seitz and Kutulakos, 1998) or when reflectance is normal-dependent (Sato et al., 1997). Specific examples include (1) using a mobile camera mounted with a light source to capture photographs of a scene whose reflectance can be expressed in closed form (e.g., using the Torrance-Sparrow model (Torrance and Sparrow, 1967; Sato et al., 1997)), and (2) using multiple cameras to capture photographs of an approximately Lambertian scene under arbitrary unknown illumination (Figure 1).

[9] Note that if $\mathcal{V} = I\!R^3$, the problem reduces to the case when no constraints on free space are available.

[10] Our use of the term *photo hull* to denote the "maximal" photo-consistent shape defined by a collection of photographs is due to a suggestion by Leonard McMillan.

[11] Examples include defining $\mathcal{V}$ to be equal to the visual hull or, in the case of a camera moving through an environment , $I\!R^3$ minus a tube along the camera's path.

[12] Convergence to this shape is provably guaranteed only for scenes representable by a discrete set of voxels.

[13] Here we make the simplifying assumption that σ_0 does not vary as a function of wavelength.

[14] For example, the algorithms in (Seitz and Dyer, 1999; Seitz and Kutulakos, 1998) fail catastrophically for this scene because the distribution of the input views and the resulting occlusion relationships violate the assumptions used by those algorithms.

[15] To see this, note that $\overline{\mathcal{V}^*}$ is, by definition, a closed subset of $I\!R^3$. Now observe that every point $p \in \overline{\mathcal{V}^*}$ is infinitesimally close to a point on *some* photo-consistent shape $\mathcal{V}'$. It follows that p is infinitesimally close to an open 3-ball inside $\mathcal{V}' \subseteq \overline{\mathcal{V}^*}$. The closed photo hull therefore satisfies our definition of a shape.

[16] Note that such a point does exist since $\mathcal{P} \cap N$ is a closed and bounded subset of $I\!R^3$ and hence it is compact (Armstrong, 1983).

References

Alfvin, R. L. and M. D. Fairchild: 1997, 'Observer variability in metameric color matches using color reproduction media'. *Color Research & Application* **22**(3), 174–178.

Aloimonos, Y.: 1988, 'Visual Shape Computation'. *Proc. IEEE* **76**, 899–916.

Armstrong, M. A.: 1983, *Basic Topology*. Springer-Verlag.

Bascle, B. and R. Deriche: 1993, 'Stereo Matching, Reconstruction and Refinement of 3D Curves Using Deformable Contours'. In: *Proc. 4th Int. Conf. Computer Vision*. pp. 421–430.

Beardsley, P., P. Torr, and A. Zisserman: 1996, '3D Model Acquisition from Extended Image Sequences'. In: *Proc. 4th European Conf. on Computer Vision*. pp. 683–695.

Belhumeur, P. N.: 1996, 'A Bayesian Approach to Binocular Stereopsis'. *Int. J. on Computer Vision* **19**(3), 237–260.

Belhumeur, P. N. and D. J. Kriegman: 1996, 'What is the Set of Images of an Object Under All Possible Lighting Conditions'. In: *Proc. Computer Vision and Pattern Recognition*. pp. 270–277.

Bolles, R. C., H. H. Baker, and D. H. Marimont: 1987, 'Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion'. *Int. J. Computer Vision* **1**, 7–55.

Bolles, R. C. and R. A. Cain: 1982, 'Recognizing and Locating Partially-Visible Objects: The Local-Feature-Focus Method'. *Int. J. Robotics Research* **1**(3), 57–82.

Cipolla, R. and A. Blake: 1992, 'Surface Shape from the Deformation of Apparent Contours'. *Int. J. Computer Vision* **9**(2), 83–112.

Collins, R. T.: 1996, 'A space-sweep approach to true multi-image matching'. In: *Proc. Computer Vision and Pattern Recognition Conf.* pp. 358–363.

Cox, I., S. Hingorani, S. Rao, and B. Maggs: 1996, 'A Maximum Likelihood Stereo Algorithm'. *CVIU: Image Understanding* **63**(3), 542–567.

Culbertson, W. B., T. Malzbender, and G. Slabaugh: 1999, 'Generalized Voxel Coloring'. In: *Workshop on Vision Algorithms: Theory and Practice*. Corfu, Greece.

Curless, B. and M. Levoy: 1996, 'A Volumetric Method for Building Complex Models from Range Images'. In: *Proc. SIGGRAPH'96*. pp. 303–312.

Debevec, P. E., C. J. Taylor, and J. Malik: 1996, 'Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach'. In: *Proc. SIGGRAPH'96*. pp. 11–20.

Epstein, R., A. L. Yuille, and P. N. Belhumeur: 1996, 'Learning Object Representations from Lighting Variations'. In: J. Ponce, A. Zisserman, and M. Hebert (eds.): *Object Representation in Computer Vision II*. pp. 179–199.

Faugeras, O.: 1995, 'Stratification of three-dimensional vision: projective, affine, and metric representations'. *J. Opt. Soc. Am. A* **12**(3), 465–484.

Faugeras, O. and R. Keriven: 1998, 'Complete Dense Stereovision Using Level Set Methods'. In: *Proc. 5th European Conf. on Computer Vision*. pp. 379–393.

Faugeras, O. D.: 1998, 'Personal communication'.

Faugeras, O. D. and S. Maybank: 1990, 'Motion from point matches: multiplicity of solutions'. *Int. J. Computer Vision* **4**, 225–246.

Foley, J. D., A. van Dam, S. K. Feiner, and J. F. Hughes: 1990, *Computer Graphics Principles and Practice*. Addison-Wesley Publishing Co.

Forsyth, D. and A. Zisserman: 1991, 'Reflections on Shading'. *IEEE Trans. Pattern Anal. Machine Intell.* **13**(7), 671–679.

Freund, J. E.: 1992, *Mathematical Statistics*. Englewood Cliffs, NJ: Prentice Hall.

Fua, P. and Y. G. Leclerc: 1995, 'Object-Centered Surface Reconstruction: Combining Multi-Image Stereo and Shading'. *Int. J. Computer Vision* **16**, 35–56.

Fuchs, H., Z. Kedem, and B. F. Naylor: 1980, 'On Visible Surface Generation by A Priori Tree Structures'. In: *Proc. SIGGRAPH '80*. pp. 39–48.

Hoff, W. and N. Ahuja: 1989, 'Surfaces from Stereo: Integrating Feature Matching, Disparity Estimation, and Contour Detection'. *IEEE Trans. Pattern Anal. Machine Intell.* **11**, 121–136.

Kakadiaris, I. A. and D. Metaxas: 1995, '3D Human Body Model Acquisition from Multiple Views'. In: *Proc. Int. Conf. on Computer Vision*. pp. 618–623.

Kanade, T., P. J. Narayanan, and P. W. Rander: 1995, 'Virtualized Reality: Concepts and Early Results'. In: *Proc. Workshop on Representations of Visual Scenes*. pp. 69–76.

Kanade, T., A. Yoshida, K. Oda, H. Kano, and M. Tanaka: 1996, 'A Stereo Machine for Video-rate Dense Depth Mapping and its New Applications'. In: *Proc. Computer Vision and Pattern Recognition Conf.*

Kang, S. B. and R. Szeliski: 1996, '3-D Scene Data Recovery using Omnidirectional Multibaseline Stereo'. In: *Proc. Computer Vision and Pattern Recognition Conf.* pp. 364–370.

Katayama, A., K. Tanaka, T. Oshino, and H. Tamura: 1995, 'A viewpoint dependent stereoscopic display using interpolation of multi-viewpoint images'. In: *Proc. SPIE*, Vol. 2409A. pp. 21–30.

Koenderink, J. J. and A. J. van Doorn: 1991, 'Affine structure from motion'. *J. Opt. Soc. Am.* **A**(2), 377–385.

Kutulakos, K. N.: 1997, 'Shape from the Light Field Boundary'. In: *Proc. Computer Vision and Pattern Recognition*. pp. 53–59.

Kutulakos, K. N.: 2000, 'Approximate N-View Stereo'. In: *Proc. European Conf. on Computer Vision*. To appear.

Kutulakos, K. N. and C. R. Dyer: 1994, 'Recovering Shape by Purposive Viewpoint Adjustment'. *Int. J. Computer Vision* **12**(2), 113–136.

Kutulakos, K. N. and C. R. Dyer: 1995, 'Global Surface Reconstruction by Purposive Control of Observer Motion'. *Artificial Intelligence Journal* **78**(1-2), 147–177.

Langer, M. S. and S. W. Zucker: 1994, 'Shape-from-shading on a cloudy day'. *J. Opt. Soc. Am. A* **11**(2), 467–478.

Laurentini, A.: 1994, 'The Visual Hull Concept for Silhouette-Based Image Understanding'. *IEEE Trans. Pattern Anal. Machine Intell.* **16**(2), 150–162.

Marr, D.: 1982, *Vision*. Freeman.

Martin, W. N. and J. K. Aggarwal: 1983, 'Volumetric Descriptions of Objects from Multiple Views'. *IEEE Proc. Pattern Anal. Machine Intell.* **5**(2), 150–158.

Moezzi, S., A. Katkere, D. Y. Kuramura, and R. Jain: 1996, 'Reality Modeling and Visualization from Multiple Video Sequences'. *IEEE Computer Graphics and Applications* **16**(6), 58–63.

Mundy, J. L. and A. Zisserman (eds.): 1992, *Geometric Invariance in Computer Vision*. MIT Press.

Narayanan, P. J., P. W. Rander, and T. Kanade: 1998, 'Constructing Virtual Worlds Using Dense Stereo'. In: *Proc. Int. Conf. on Computer Vision*. pp. 3–10.

Newell, M. E., R. G. Newell, and T. L. Sancha: 1972, 'A Solution to the Hidden Surface Problem'. In: *Proc. ACM National Conference*. pp. 443–450.

Okutomi, M. and T. Kanade: 1993, 'A multiple-baseline stereo'. *IEEE Trans. Pattern Anal. Machine Intell.* **15**(4), 353–363.

Poggio, T., V. Torre, and C. Koch: 1985, 'Computational Vision and Regularization Theory'. *Nature* **317**(26), 314–319.

Pollefeys, M., R. Koch, and L. V. Gool: 1998, 'Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters'. In: *Proc. 6th Int. Conf. on Computer Vision*. pp. 90–95.

Pritchett, P. and A. Zisserman: 1998, 'Wide Baseline Stereo Matching'. In: *Proc. 6th Int. Conf. on Computer Vision*. pp. 754–760.

Roy, S. and I. J. Cox: 1998, 'A Maximum-Flow Formulation of the N-camera Stereo Correspondence Problem'. In: *Proc. 6th Int. Conf. on Computer Vision*. pp. 492–499.

Sato, Y., M. D. Wheeler, and K. Ikeuchi: 1997, 'Object Shape and Reflectance Modeling from Observation'. In: *Proc. SIGGRAPH'97*. pp. 379–387.

Seales, W. B. and O. Faugeras: 1995, 'Building Three-Dimensional Object Models from Image Sequences'. *Computer Vision and Image Understanding* **61**(3), 308–324.

Seitz, S. M. and C. R. Dyer: 1995, 'Complete Scene Structure from Four Point Correspondences'. In: *Proc. 5th Int. Conf. on Computer Vision*. pp. 330–337.

Seitz, S. M. and C. R. Dyer: 1999, 'Photorealistic Scene Reconstruction by Voxel Coloring'. *Int. J. Computer Vision* **35**(2), 151–173.

Seitz, S. M. and K. N. Kutulakos: 1998, 'Plenoptic Image Editing'. In: *Proc. 6th Int. Conf. Computer Vision*. pp. 17–24.

Smith, A. R. and J. F. Blinn: 1996, 'Blue Screen Matting'. In: *Proc. SIGGRAPH'96*. pp. 259–268.

Stewart, C. V.: 1995, 'MINPRAN: A New Robust Estimator for Computer Vision'. *IEEE Trans. Pattern Anal. Machine Intell.* **17**(10), 925–938.

Szeliski, R.: 1993, 'Rapid Octree Construction from Image Sequences'. *CVGIP: Image Understanding* **58**(1), 23–32.

Szeliski, R. and P. Golland: 1998, 'Stereo Matching with Transparency and Matting'. In: *Proc. 6th Int. Conf. on Computer Vision*. pp. 517–524.

Szeliski, R. and R. Weiss: 1994, 'Robust Shape Recovery from Occluding Contours Using a Linear Smoother'. In: C. M. Brown and D. Terzopoulos (eds.): *Real-time Computer Vision*. Cambridge University Press, pp. 141–165.

Tomasi, C. and T. Kanade: 1992, 'Shape and Motion from Image Streams under Orthography: A Factorization Method'. *Int. J. Computer Vision* **9**(2), 137–154.

Torrance, K. E. and E. M. Sparrow: 1967, 'Theory of off-specular reflection from roughened surface'. *Journal of the Optical Society of America* **57**, 1105–1114.

Turk, G. and M. Levoy: 1994, 'Zippered Polygon Meshes from Range Images'. In: *Proc. SIGGRAPH'94*. pp. 311–318.

Vaillant, R. and O. D. Faugeras: 1992, 'Using Extremal Boundaries for 3-D Object Modeling'. *IEEE Trans. Pattern Anal. Machine Intell.* **14**(2), 157–173.

van Veen, J. A. J. C. and P. Werkhoven: 1996, 'Metamerisms in structure-from-motion perception'. *Vision Research* **36**(14), 2197–2210.

Woodham, R. J., Y. Iwahori, and R. A. Barman: 1991, 'Photometric Stereo: Lambertian Reflectance and Light Sources with Unknown Direction and Strength'. Technical Report 91-18, University of British Columbia, Laboratory for Computational Intelligence.

Zhang, Z.: 1998, 'Image-Based Geometrically-Correct Photorealistic Scene/Object Modeling (IBPhM): A Review'. In: *Proc. 3rd Asian Conf. on Computer Vision*. pp. 340–349.

Zhao, C. and R. Mohr: 1996, 'Global Three-Dimensional Surface Reconstruction from Occluding Contours'. *Computer Vision and Image Understanding* **64**(1), 62–96.

Zitnick, C. L. and J. A. Webb: 1996, 'Multi-baseline stereo using surface extraction'. Technical Report CMU-CS-96-196, Carnegie Mellon University, Pittsburgh, PA.

Modeling and Rendering Architecture from Photographs

Paul Debevec

USC Institute for Creative Technologies
www.debevec.org
paul@debevec.org

Notes for SIGGRAPH 2000 Course #19
3D Photography
Organized by Brian Curless and Steve Seitz
July 24, 2000

Contents

This section of the course notes is organized as follows:

1. Introductory material for this section. This includes a brief overview of related and complimentary material to photogrammetric modeling, such as structure from motion, stereo correspondence, shape from silhouettes, camera calibration, laser scanning, and image-based rendering.
2. A bibliography of related papers.
3. A reprint of:

   Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. *Modeling and Rendering Architecture from Photographs.* In SIGGRAPH 96, August 1996, pp. 11-20.
4. Notes on photogrammetric recovery of arches and surfaces of revolution written by George Borshukov.
5. Copies of the slides used for the presentation.

More information can be found in [10], [5], and [13], available at:
`http://www.cs.berkeley.edu/~debevec/Thesis`

1 Introduction

The creation of three-dimensional models of existing architectural scenes with the aid of the computer has been commonplace for some time, and the resulting models have been both entertaining virtual environments as well as valuable visualization tools. Large-scale efforts have pushed the campuses of Iowa State University, California State University – Chico, and swaths of downtown Los Angeles [23] through the graphics pipeline. Unfortunately, the modeling methods employed in such projects are very labor-intensive. They typically involve surveying the site, locating and digitizing architectural plans (if available), and converting existing CAD data (if available). Moreover, the renderings of such models are noticeably computer-generated; even those that employ large number of texture-maps generally fail to resemble real photographs.

Already, efforts to build computer models of architectural scenes have produced many interesting applications in computer graphics; a few such projects are shown in Fig. 1. Unfortunately, the traditional methods of constructing models (Fig. 2a) of existing architecture, in which a modeling program is used to manually position the elements of the scene, have several drawbacks. First, the process is extremely labor-intensive, typically involving surveying the site, locating and digitizing architectural plans (if available), or converting existing CAD data (again, if available). Second, it is difficult to verify whether the resulting model is accurate.

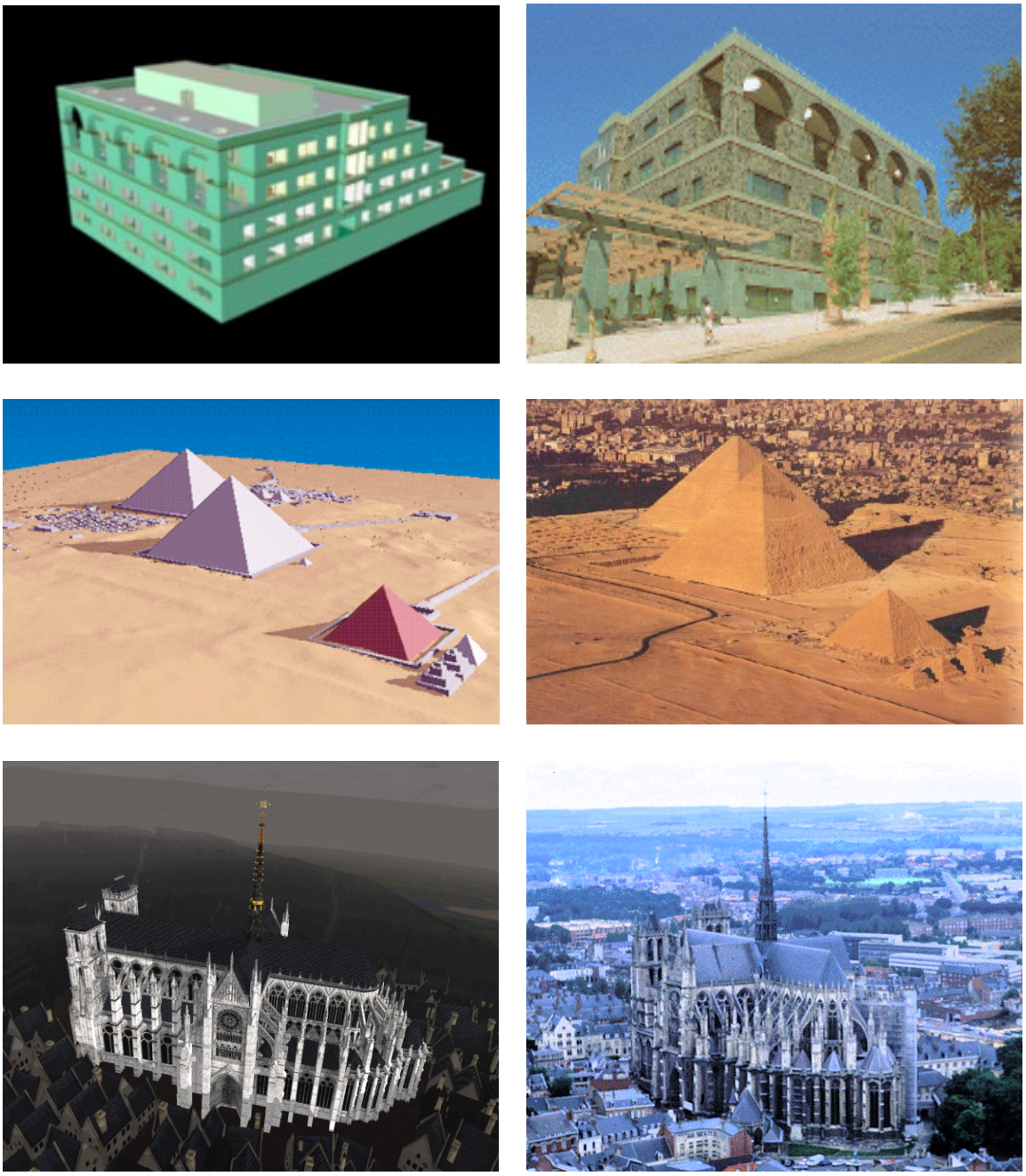


Figure 1: Three ambitious projects to model architecture with computers, each presented with a rendering of the computer model and a photograph of the actual architecture. Top: Soda Hall Walkthru Project [47, 19], University of California at Berkeley. Middle: Giza Plateau Modeling Project, University of Chicago. Bottom: Virtual Amiens Cathedral, Columbia University. Using traditional modeling techniques (Fig. 2a), each of these models required many person-months of effort to build, and although each project yielded enjoyable and useful renderings, the results are qualitatively different from actual photographs of the architecture.

Most disappointing, though, is that the renderings of the resulting models are noticeably computer-generated; even those that employ liberal texture-mapping generally fail to resemble real photographs. As a result, it is easy to distinguish the computer renderings from the real photographs in Fig. 1.

Recently, creating models directly from digital images has received increased interest in both computer vision and in computer graphics under the title of image-based modeling and rendering. Since real images are used as input, such an image-based system (Fig. 2c) has an advantage in producing photorealistic renderings as output. Some of these promising systems (e.g. [26, 32, 31, 44, 39], see also Figs. 3 and 4) employ the computer vision technique of computational stereopsis to automatically determine the structure of the scene from the multiple photographs available. As a consequence, however, these systems are only as strong as the underlying stereo algorithms. This has caused problems because state-of-the-art stereo algorithms have a number of significant weaknesses; in particular, the photographs need to have similar viewpoints for reliable results to be obtained. Because of this, current image-based techniques must use many closely spaced images, and in some cases employ significant amounts of user input for each image pair to supervise the stereo algorithm. In this framework, capturing the data for a realistically renderable model would require an impractical number of closely spaced photographs, and deriving the depth from the photographs could require an impractical amount of user input. These concessions to the weakness of stereo algorithms would seem to bode poorly for creating large-scale, freely navigable virtual environments from photographs.

The techniques presented in these notes aim to make the process of obtaining basic models of architectural scenes more convenient, more accurate, and more photorealistic than the methods currently available. The approach developed draws on the strengths of both geometry-based and image-based methods, as illustrated in Fig. 2b. The result is that our approach to modeling and rendering architecture requires only a sparse set of photographs and can produce realistic renderings from arbitrary viewpoints. In our approach, a basic geometric model of the architecture is recovered semi-automatically with an easy-to-use photogrammetric modeling system (explained in the following reprinted paper [12]), novel views are created using view-dependent texture mapping [12, 13], and additional geometric detail can be recovered through model-based stereo correspondence [12, 10]. The final images can be rendered with current image-based rendering techniques or with traditional texture-mapping hardware. Because only photographs are required, our approach to modeling architecture is neither invasive nor does it require architectural plans, CAD models, or specialized instrumentation such as surveying equipment, GPS sensors or laser range scanners.

2 Work Related to Photogrammetric Modeling

The process of recovering 3D structure from 2D images has been a central endeavor within computer vision, and the process of rendering such recovered structures is an emerging topic in computer graphics. Although no general technique exists to derive models from images, several areas of research have provided results that are applicable to the problem of modeling and rendering architectural scenes. The particularly relevant areas reviewed here are: Camera Calibration, Structure from Motion, Shape from Silhouette Contours, Stereo Correspondence, and Image-Based Rendering.

2.1 Camera calibration

Recovering 3D structure from images becomes a simpler problem when the images are taken with *calibrated* cameras. For our purposes, a camera is said to be *calibrated* if the mapping between image coordinates and directions relative to the camera center are known. However, the position of the camera in space (i.e. its translation and rotation with respect to world coordinates) is not necessarily known. An excellent presentation of the algebraic and matrix representations of perspective cameras may be found in [17].

Considerable work has been done in both photogrammetry and computer vision to calibrate cameras and lenses for both their perspective intrinsic parameters and their distortion patterns. Some successful methods include [49], [16], and [15]. While there has been recent progress in the use of uncalibrated views for 3D reconstruction [18], this method does not consider non-perspective camera distortion which prevents high-precision results for images taken with real cameras. We have found camera calibration to be a straightforward

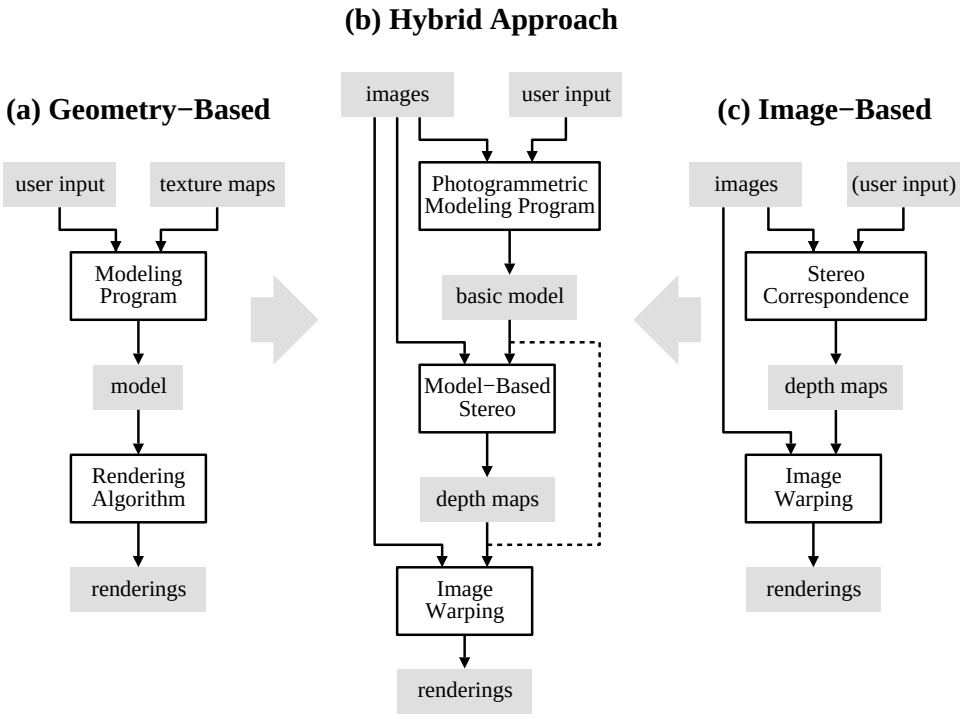


Figure 2: Schematic of how our hybrid approach combines geometry-based and image-based approaches to modeling and rendering architecture from photographs. The geometry-based approach illustrated places the majority of the modeling task on the user, whereas the image-based approach places the majority of the task on the computer. Our method divides the modeling task into two stages, one that is interactive, and one that is automated. The dividing point we have chosen capitalizes on the strengths of both the user and the computer to produce the best possible models and renderings using the fewest number of photographs. The dashed line in the geometry-based schematic indicates that images may optionally be used in a modeling program as texture-maps. The dashed line in the image-based schematic indicates that in some systems user input is used to initialize the stereo correspondence algorithm. The dashed line in the hybrid schematic indicates that view-dependent texture-mapping (discussed later in these notes and in [10, 13, 36]) can be used without performing stereo correspondence.

Figure 3: The Immersion ’94 [32] stereo image sequence capture rig, being operated by Michael Naimark of Interval Research Corporation. Immersion ’94 was one project that attempted to create navigable, photorealistic virtual environments from photographic data. The stroller supports two identical 16mm movie cameras, and has an encoder on one wheel to measure the forward motion of the rig. The cameras are motor-driven and can be programmed to take pictures in synchrony at any distance interval as the camera rolls forward. For much of the work done for the Immersion project, the forward motion distance between acquired stereo pairs was one meter.

process that considerably simplifies the problem of 3D reconstruction, although the methods presented here can also solve for focal lengths and other intrinsic parameters if necessary. [10], Chapter 4 provides a more detailed overview of the issues involved in camera calibration and discusses the camera calibration process used in this work.

2.2 Structure from motion

Given the 2D projection of a point in the world, its position in 3D space could be anywhere on a ray extending out in a particular direction from the camera’s optical center. However, when the projections of a sufficient number of points in the world are observed in multiple images from different positions, it is mathematically possible to deduce the 3D locations of the points as well as the positions of the original cameras, up to an unknown factor of scale.

This problem has been studied in the area of photogrammetry for the principal purpose of producing topographic maps. In 1913, Kruppa [25] proved the fundamental result that given two views of five distinct points, one could recover the rotation and translation between the two camera positions as well as the 3D locations of the points (up to a scale factor). Since then, the problem’s mathematical and algorithmic aspects have been explored starting from the fundamental work of Ullman [51] and Longuet-Higgins [29], in the early 1980s. Faugeras’s book [17] overviews the state of the art as of 1992. So far, a key realization has been that the recovery of structure is very sensitive to noise in image measurements when the translation between the available camera positions is small.

Attention has turned to using more than two views with image stream methods such as [48] or recursive approaches [1]. Tomasi and Kanade [48] (see Fig. 5) showed excellent results for the case of orthographic cameras, but direct solutions for the perspective case remain elusive. In general, linear algorithms for the problem fail to make use of all available information while nonlinear optimization methods are prone to difficulties arising from local minima in the parameter space. An alternative formulation of the problem by Taylor and Kriegman [46] (see Fig. 6) uses lines rather than points as image measurements, but the previously stated concerns were shown to remain largely valid. For purposes of computer graphics, there is yet another problem: the models recovered by these algorithms consist of sparse point fields or individual line segments, which are

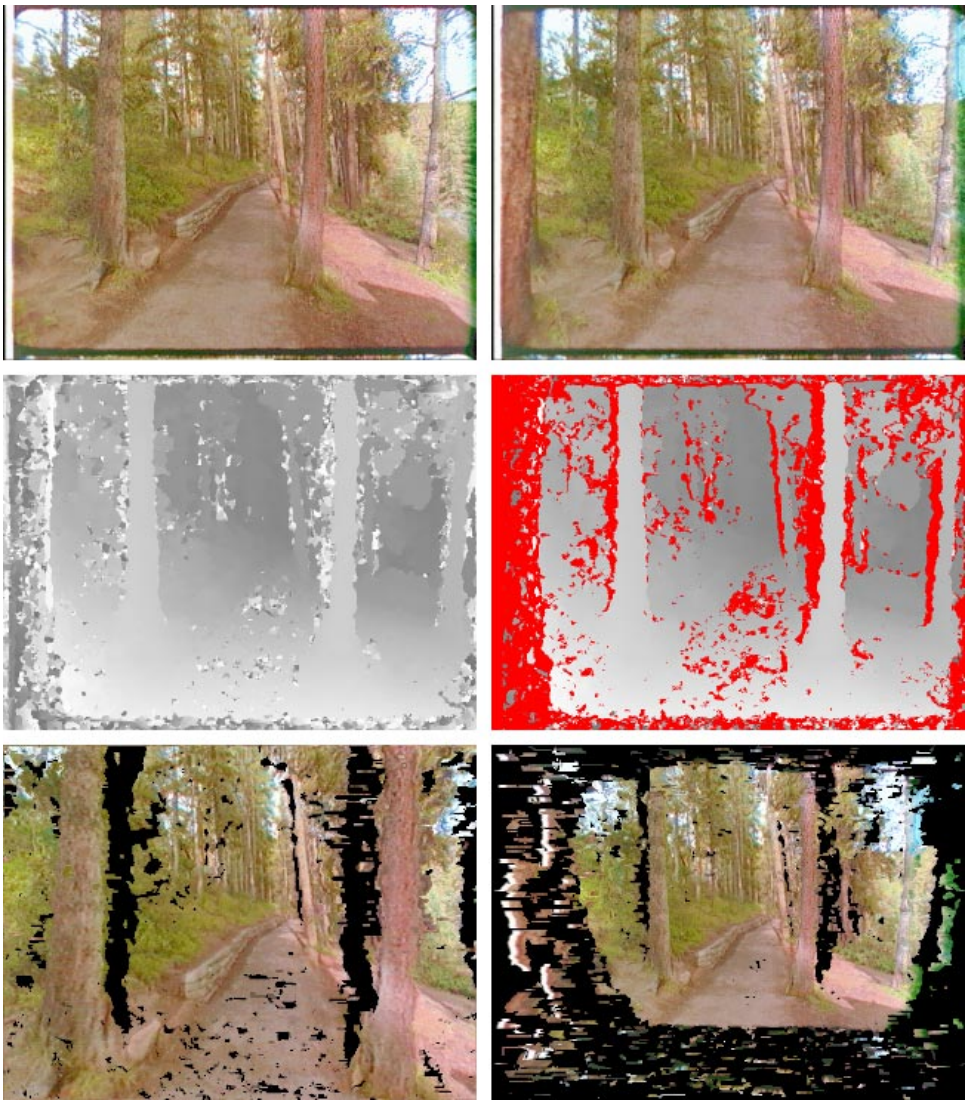


Figure 4: The Immersion ’94 [32] image-based modeling and rendering (see Fig. 2c) project. The top two photos are a stereo pair (reversed for cross-eyed stereo viewing) taken with the apparatus in Fig. 3 in Canada’s Banff National Forest. The film frame was overscanned to assist in image registration. The middle left photo is a stereo disparity map produced by a parallel implementation of the Zabih-Woodfill stereo algorithm [55]. To its right the map has been processed using a left-right consistency check to invalidate regions where running stereo based on the left image and stereo based on the right image did not produce consistent results. Below are two virtual views generated by casting each pixel out into space based on its computed depth estimate, and reimaging the pixels into novel camera positions. On the left is the result of virtually moving one meter forward, on the right is the result of virtually moving one meter backward. Note the dark de-occluded areas produced by these virtual camera moves; these areas were not seen in the original stereo pair. In the Immersion ’94 animations, these regions were automatically filled in from neighboring stereo pairs.

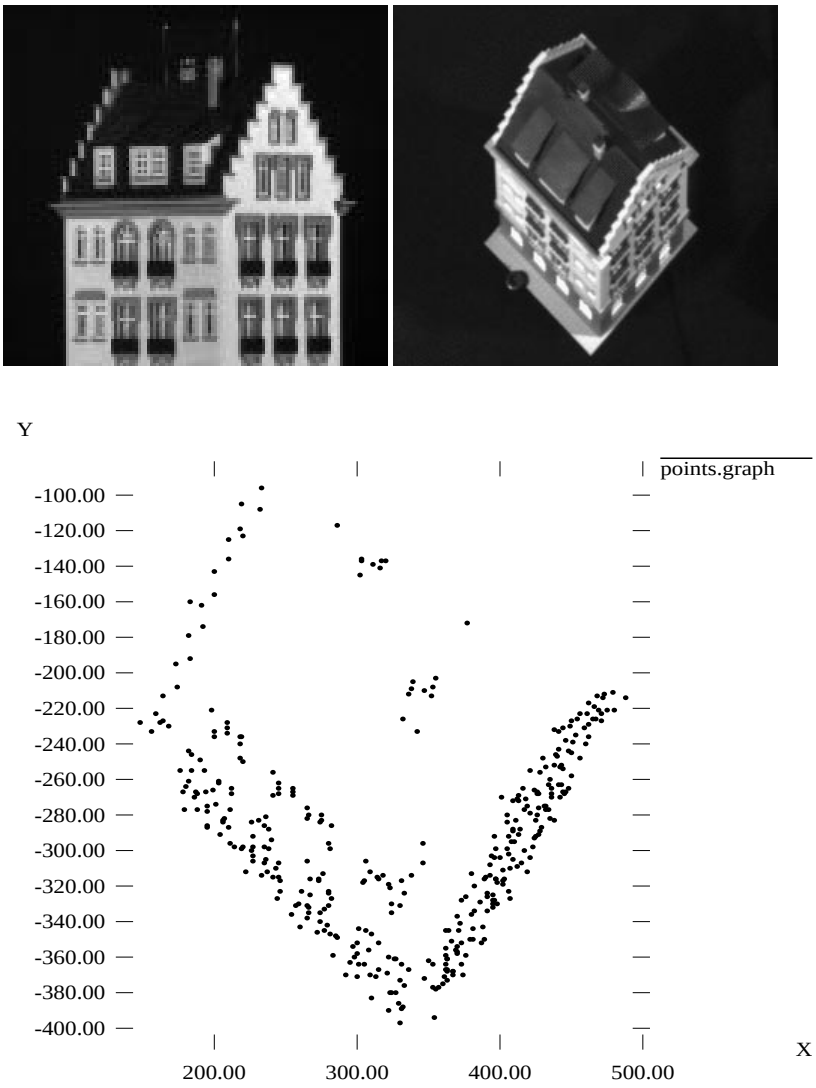


Figure 5: Images from the 1992 Tomasi-Kanade structure from motion paper [48]. In this paper, feature points were automatically tracked in an image sequence of a model house rotating. By assuming the camera was orthographic (which was approximated by using a telephoto lens), they were able to solve for the 3D structure of the points using a linear factorization method. The above left picture shows a picture from the original sequence, the above right picture shows a second image of the model from above (not in the original sequence), and the plot below shows the 3D recovered points from the same camera angle as the above right picture. Although an elegant and fundamental result, this approach is not directly applicable to real-world scenes because real camera lenses (especially those typically used for architecture) are too wide-angle to be approximated as orthographic.

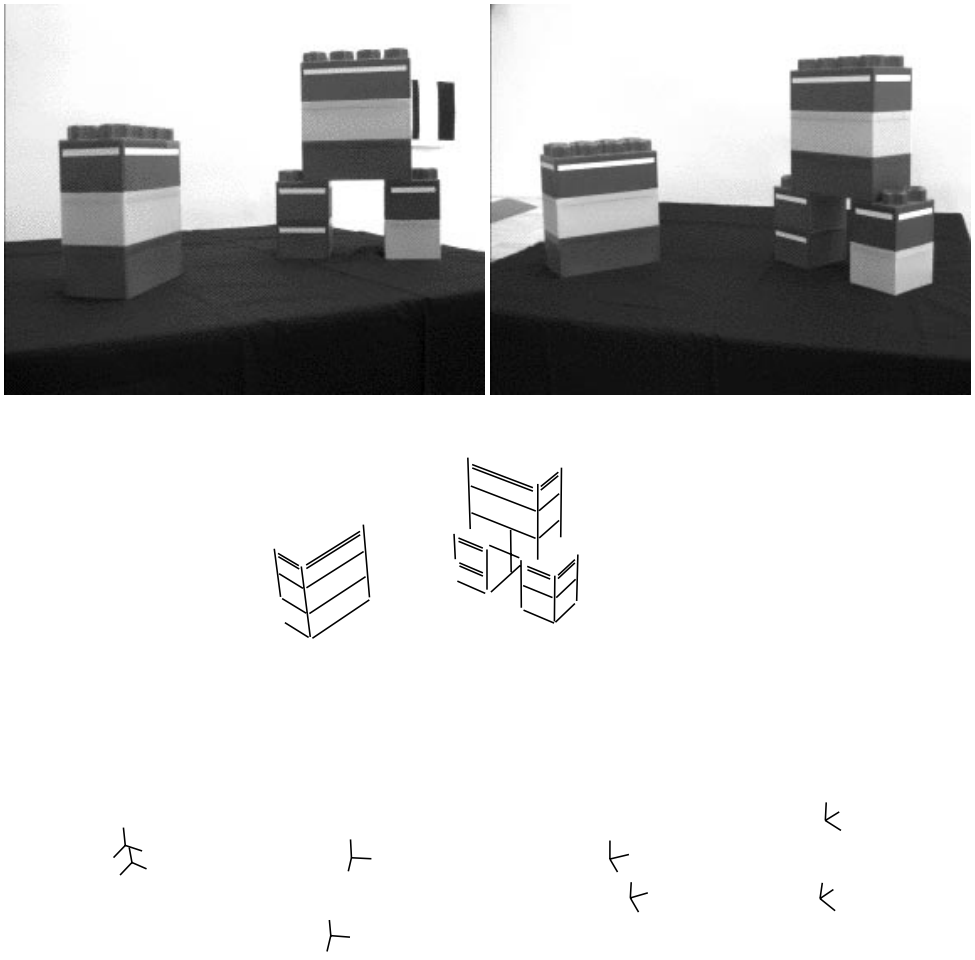


Figure 6: Images from the 1995 Taylor-Kriegman structure from motion paper [46]. In this work, structure from motion is recast in terms of line segments rather than points. A principal benefit of this is that line features are often more easily located in architectural scenes than point features. Above are two of eight images of a block scene; edge correspondences among the images were provided to the algorithm by the user. The algorithm then employed a nonlinear optimization technique to solve for the 3D positions of the line segments as well as the original camera positions, show below. This work used calibrated cameras, but allowed a full perspective model to be used in contrast to Tomasi and Kanade [48]. However, the optimization technique was prone to getting caught in local minima unless good initial estimates of the camera orientations were provided. This work was extended to become the basis of the photogrammetric modeling method presented in this section of these notes.

not directly renderable as solid 3D models.

In our approach, we exploit the fact that we are trying to recover geometric models of architectural scenes, not arbitrary three-dimensional point sets. This enables us to include additional constraints not typically available to structure from motion algorithms and to overcome the problems of numerical instability that plague such approaches. Our approach is demonstrated in an interactive system for building architectural models from photographs, described in the following paper.

2.3 Shape from silhouette contours

Some work has been done in both computer vision and computer graphics to recover the shape of objects from their silhouette contours in multiple images. If the camera geometry is known for each image, then each contour defines an infinite, cone-shaped region of space within which the object must lie. An estimate for the geometry of the object can thus be obtained by intersecting multiple such regions from different images. As a greater variety of views of the object are used, this technique can eventually recover the ray hull[1] of the object. A simple version of the basic technique was demonstrated in [8], shown in Fig. 7. In this project, three nearly orthographic photographs of a car were used to carve out its shape, and the images were mapped onto this geometry to produce renderings. Although just three views were used, the recovered shape is close to the actual shape because the views were chosen to align with the mostly boxy geometry of the object. A project in which a continuous stream of views was used to reconstruct object geometry is presented in [45, 44]; see also Fig. 8. A similar silhouette-based technique was used to provide an approximate estimate of object geometry to improve renderings in the Lumigraph image-based modeling and rendering system [20].

In modeling from silhouettes, qualitatively better results can be obtained for curved objects by assuming that the object surface normal is perpendicular to the viewing direction at every point of the contour. Using this constraint, [43] developed a surface fitting technique to recover curved models from images.

In general, silhouette contours can be used effectively to recover approximate geometry of individual objects, and the process can be automated if there is known camera geometry and the objects can be automatically segmented out of the images. Silhouette contours can also be used very effectively to recover the precise geometry of surfaces of revolution in images. However, for the general shape of an arbitrary building that has many sharp corners and concavities, silhouette contours alone can not provide adequately accurate model geometry.

Although not adequate for general building shapes, silhouette contours could be useful in recovering the approximate shapes of trees, bushes, and topiary in architectural scenes. Techniques such as those presented in [35] could then be used to synthesize detailed plant geometry to conform to the shape and type of the original flora. This technique would seem to hold considerably more promise for practically recovering plant structure than trying to reconstruct the position and coloration of each individual leaf and branch of every tree in the scene.

2.4 Stereo correspondence

The geometrical theory of structure from motion assumes that one is able to solve the *correspondence* problem, which is to identify the points in two or more images that are projections of the same point in the world. In humans, corresponding points in the two slightly differing images on the retinas are determined by the visual cortex in the process called binocular stereopsis. Two terms used in reference to stereo are *baseline* and *disparity*. The baseline of a stereo pair is the distance between the camera locations of the two images. Disparity refers to the difference in image location between corresponding features in the two images, which is projectively related to the depth of the feature in the scene.

Years of research (e.g. [2, 14, 21, 24, 30, 33, 34]) have shown that determining stereo correspondences by computer is difficult problem. In general, current methods are successful only when the images are similar in appearance, as in the case of human vision, which is usually obtained by using cameras that are closely spaced

[1] The ray hull of an object is the complement of the union of all rays in space which do not intersect the object. The ray hull can capture some forms of object concavities, but not, in general, complicated concave structure.

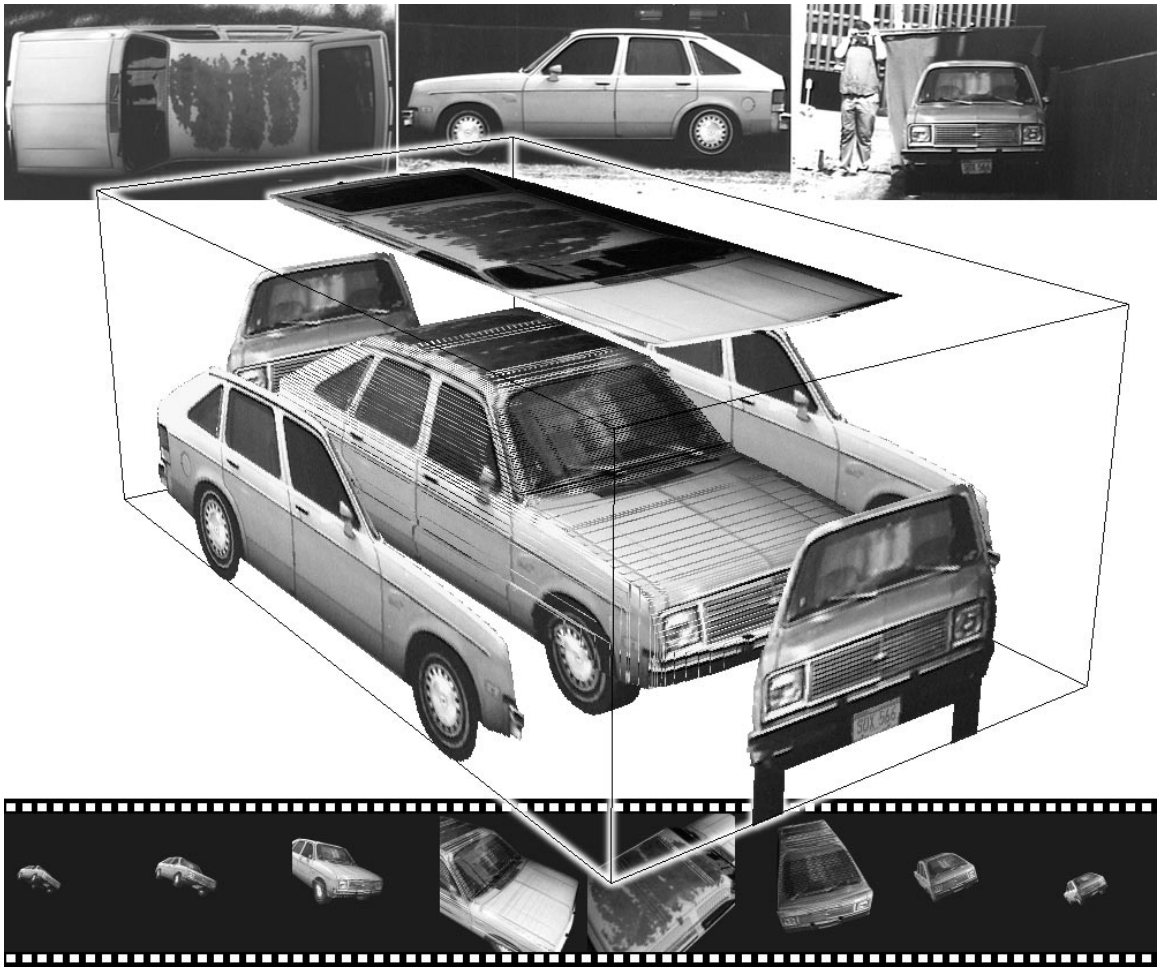


Figure 7: Images from the 1991 Chevette Modeling project [8]. The top three images show pictures of the 1980 Chevette photographed with a 210mm lens from the top, side, and front. The Chevette was semi-automatically segmented from each image, and these images were then registered with each other approximating the projection as orthographic. The registered photographs are shown placed in proper relation to each other on the faces of a rectangular box in the center of the figure. The shape of the car is then carved out from the box volume by perpendicularly sweeping each of the three silhouettes like a cookie-cutter through the box volume. The recovered volume (shown inside the box) is then textured-mapped by projecting the original photographs onto it. The bottom of the figure shows a sampling of frames from a synthetic animation of the car flying across the screen. Although (and perhaps because) the final model has flaws resulting from specularities, missing concavities, and imperfect image registration, it unequivocally evokes an uncanny sense of the actual vehicle.

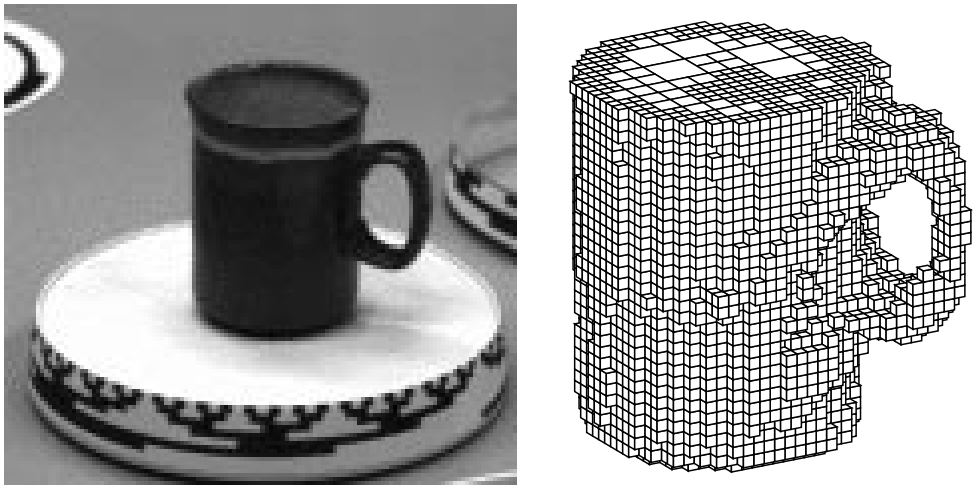


Figure 8: Images from a silhouette modeling project by Rick Szeliski [45, 44]. The cup was videotaped on a rotating platform (left), and the extracted contours from this image sequence were used to automatically recover the shape of the cup (right).

relative to the objects in the scene. As the distance between the cameras (often called the *baseline*) increases, surfaces in the images exhibit different degrees of foreshortening, different patterns of occlusion, and large disparities in their locations in the two images, all of which makes it much more difficult for the computer to determine correct stereo correspondences. To be more specific, the major sources of difficulty include:

1. **Foreshortening**. Surfaces in the scene viewed from different positions will be foreshortened differently in the images, causing the image neighborhoods of corresponding pixels to appear dissimilar. Such dissimilarity can confound stereo algorithms that use local similarity metrics to determine correspondences.

2. **Occlusions**. Depth discontinuities in the world can create half-occluded regions in an image pair, which also poses problems for local similarity metrics.

3. **Lack of Texture**. Where there is an absence of image intensity features it is difficult for a stereo algorithm to correctly find the correct match for a particular point, since many point neighborhoods will be similar in appearance.

Unfortunately, the alternative of improving stereo correspondence by using images taken from nearby locations has the disadvantage that computing depth becomes very sensitive to noise in image measurements. Since depth is computed by taking the inverse of disparity, image pairs with small disparities tend to give rise to noisy depth estimates. Geometrically, depth is computed by triangulating the position of a matched point from its imaged position in the two cameras. When the cameras are placed close together, this triangle becomes very narrow, and the distance to its apex becomes very sensitive to the angles at its base. Noisy depth estimates mean that novel views will become visually unconvincing very quickly as the virtual camera moves away from the original viewpoint[2].

Thus, computing scene structure from stereo leaves us with a conundrum: image pairs with narrow baselines (relative to the distance of objects in the scene) are similar in appearance and make it possible to auto-

[2] The error present in a synthetic view as a function of stereo correspondence accuracy can be described as the **re-rendering equation**. If the novel view is at the same position as the original view, then no amount of depth estimation error can effect the appearance of the re-rendered view; it will always be the same as the original view up to rotation. However, if the novel view is displaced up to one baseline away from the original view, then a stereo correspondence error of n pixels will cause up to n pixels of error in the reprojected position of the mis-corresponded pixel. For a displacement up to k baselines away from the original viewpoint, the reprojection error will be up to kn pixels in the reprojected view, with this bound realized if the camera motion is parallel to the baseline between the cameras. Thus, it is advisable to limit novel viewpoints to be within a few baselines of the original views, lest correspondence errors distort the images very noticeably.

matically compute stereo correspondences, but give noisy depth estimates. Image pairs with wide baselines can give very accurate depth localization for matched points, but the images usually exhibit large disparities, significant regions of occlusion, and different forms of foreshortening which makes it very difficult to automatically determine correspondences.

In these notes, we help address this problem by showing that having an approximate model of the photographed scene can be used to robustly determine stereo correspondences from images taken from widely varying viewpoints. Specifically, the model enables us to warp the images to eliminate unequal foreshortening and to predict major instances of occlusion *before* trying to find correspondences. This technique is a generalization of the plane-plus-parallax parameterization [38] which we call *model-based stereo*; it is presented in the following paper and in [10], Chapter 7.

2.5 Range scanning

Instead of the approach of using multiple images to reconstruct scene structure, an alternative technique is to use range imaging sensors (e.g. [4]) to directly measure depth to various points in the scene. Range imaging sensors determine depth either by triangulating the position of a projected laser stripe, or by measuring the time of flight of a directional laser pulse. While existing versions of these sensors are generally slow, cumbersome and expensive, active development of this technology is making it of practical use for more and more applications. Indeed, the improved practicality of these devices combined with their amazing resolution and range precision will advocate their use in more and more modeling projects. In particular, the Digital Michaelangelo project [27] being directed by Professor Marc Levoy of Stanford University will undoubtedly serve as a watershed event in the practical use of laser range devices and digital photography for creating realistic models of both objects and environments for computer graphics applications.

Algorithms for combining multiple range images from different viewpoints have been developed both in computer vision [53, 42, 41] and in computer graphics [22, 50, 6]; see also Fig. 9. In many ways, range image based techniques and photographic techniques are complementary and each have advantages and disadvantages. Some advantages of modeling from photographic images are that (a) still cameras are inexpensive and widely available, (b) for some architecture that no longer exists (historic buildings, disassembled film sets) all that is available are photographs, and (c) photogrammetry works at arbitrary distances, and is always eye-safe. Of course, geometry alone is insufficient for producing realistic renderings of a scene; photometric information from photographs is also necessary. In general, any image-based rendering technique that can work with geometry acquired from photogrammetry or stereo can work equally well or better with geometry acquired from range scanning.

2.6 Image-based modeling and rendering

In traditional image-based rendering systems, the model consists of a set of images of a scene and their corresponding depth maps. When the depth of every point in an image is known, the image can be re-rendered from any nearby point of view by projecting the pixels of the image to their proper 3D locations and reprojecting them onto a new image plane. Thus, a new image of the scene is created by warping the images according to their depth maps. A principal attraction of image-based rendering is that it offers a method of rendering arbitrarily complex scenes with a constant amount of computation required per pixel. Using this property, [52] demonstrated how regularly spaced synthetic images (with their computed depth maps) could be warped and composited in real time to produce a virtual environment.

In the Immersion '94 project [32], (Fig. 4) stereo photographs with a baseline of eight inches were taken every meter along a trail in a forest. Depth was extracted from each stereo pair using a census stereo algorithm [55]. Novel views were produced by supersampled z-buffered forward pixel splatting based on the stereo depth estimate of each pixel. ([26] describes a different rendering approach that implicitly triangulated the depth maps.) By manually determining relative camera pose between successive stereo pairs, it was possible to optically combine re-renderings from neighboring stereo pairs to fill in missing texture information. The project was able to produce very realistic synthetic views looking forward along the trail from any position within a meter of the original camera path, which was adequate for producing a realistic virtual experience of

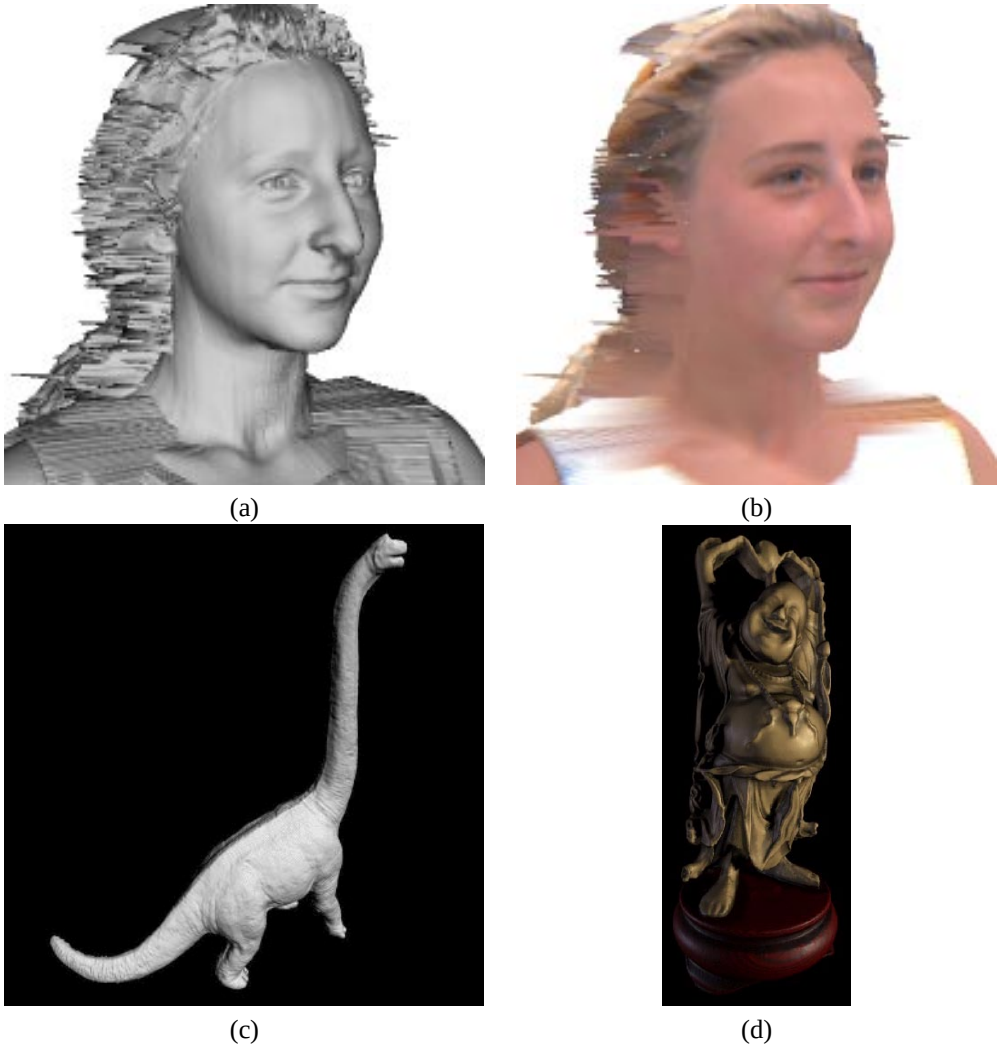


Figure 9: Several models constructed from triangulation-based laser range scanning techniques. **(a)** A model of a person's head scanned using a commercially available Cyberware laser range scanner, using a cylindrical scan. **(b)** A texture-mapped version of this model, using imagery acquired by the same video camera used to detect the laser stripe. **(c)** A more complex geometry assembled by zippering together several triangle meshes obtained from separate linear range scans of a small object from [50]. **(d)** An even more complex geometry acquired from approximately sixty range scans using the volumetric recovery method in [6].

walking down the trail. Thus, for mostly linear environments such as a forest trail, this method of capture and rendering seems promising.

[31] presented a real-time image-based rendering system that used panoramic photographs with depth computed, in part, from stereo correspondence. One observation of the paper was that extracting reliable depth estimates from stereo is very difficult. The method was nonetheless able to obtain acceptable results for nearby views using user input to aid the stereo depth recovery: the correspondence map for each image pair was seeded with 100 to 500 user-supplied point correspondences and also post-processed. Even with user assistance, the images used still had to be closely spaced; the largest baseline described in the paper was five feet.

The requirement that samples be close together is a serious limitation to generating a freely navigable virtual environment. Covering the size of just one city block could require thousands of panoramic images spaced five feet apart. Clearly, acquiring so many photographs is impractical. Moreover, even a dense lattice of ground-based photographs would only allow renderings to be generated from within a few feet of the original camera level, precluding any virtual fly-bys of the scene. Extending the dense lattice of photographs into three dimensions would clearly make the acquisition process even more difficult.

The modeling and rendering approach described in these notes takes advantage of the structure in architectural scenes so that only a sparse set of photographs can be used to recover both the geometry and the appearance of an architectural scene. As an example, the approach was able to create a virtual fly-around of the UC Berkeley bell tower and the surrounding campus from just twenty photographs (see the following slides and the web site http://www.cs.berkeley.edu/~debevec/Campanile).

Some research done concurrently with the work presented here [3] also shows that taking advantage of architectural constraints can simplify image-based scene modeling. This work specifically explored the constraints associated with the cases of parallel and coplanar edge segments.

An interesting aspect of image-based modeling and rendering is that the accuracy of the geometry can traded off with the number of images acquired and the freedom of movement attainable. [40] for example, uses no explicit geometry but rather a set of correspondences between two views of a scene to generate arbitrary views intermediate to the two original ones. And [20, 28] blend between a very large array images of an object or scene in a view-dependent manner to create the appearance of a 3D object, when the actual geometry being used can be as simple as a single plane passing through the object.

3 Conclusion

The philosophy of the work presented here is that geometry is a good thing to have, and that it should be acquired as accurately as possible. The particular techniques presented here make it possible to acquire the basic geometry for many sorts of architectural scenes, using just a set of still photographs and some effort by a trained user of the system. With the geometry recovered, the full realism of the scene can be rendered by projecting the original photographs onto the geometry, preferably combining them with a form of view-dependent texture mapping. Note that this technique can only render the scene in the original lighting conditions, and that it will not be able to convincingly render particularly shiny surfaces, which change in appearance too much with angle to be captured adequately in a sparse set of views. Addressing these problems requires obtaining estimates of the lighting conditions and material properties of the scene, which is the subject of work in image-based lighting [11, 9], BRDF recovery [7, 37], and Inverse Global Illumination [54].

More extensive information on Image-Based Modeling, Rendering, and Lighting and how it relates to 3D Photography may be found in the SIGGRAPH 99 Course notes for Course #39, “Image-Based Modeling, Rendering, and Lighting”.

Acknowledgments

Many thanks to C.J. Taylor, Jitendra Malik, George Borshukov, Yizhou Yu, and Golan Levin for their contributions to this work, and to Interval Research Corporation, the National Science Foundation, Silicon Graphics,

the California MICRO program, Rockwell International, the ONR MURI Program, and the JSEP program for their sponsorship.

References

[1] Ali Azarbayejani and Alex Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(6):562–575, June 1995.

[2] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In *Proceedings of the Seventh IJCAI, Vancouver, BC*, pages 631–636, 1981.

[3] Shawn Becker and V. Michael Bove Jr. Semiautomatic 3-d model extraction from uncalibrated 2-d camera views. In *SPIE Symposium on Electronic Imaging: Science and Technology*, Feb 1995.

[4] P. Besl. Active optical imaging sensors. In J. Sanz, editor, *Advances in Machine Vision: Architectures and Applications*. Springer Verlag, 1989.

[5] George Borshukov. New algorithms for modeling and rendering architecture from photographs. Master's thesis, University of California at Berkeley, Computer Science Division, Berkeley CA, May 1997.

[6] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH 96*, pages 303–312, 1996.

[7] K. J. Dana, B. Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and texture of real-world surfaces. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 151–157, 1997.

[8] Paul Debevec. The Chevette Project. Summer 1991.

[9] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *SIGGRAPH 98*, July 1998.

[10] Paul E. Debevec. *Modeling and Rendering Architecture from Photographs*. PhD thesis, University of California at Berkeley, Computer Science Division, Berkeley CA, 1996. http://www.cs.berkeley.edu/~debevec/Thesis.

[11] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH 97*, pages 369–378, August 1997.

[12] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH 96*, pages 11–20, August 1996.

[13] Paul E. Debevec, Yizhou Yu, and George D. Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. In George Drettakis and Nelson Max, editors, *9th Eurographics workshop on Rendering, Vienna, Austria*, pages 105–116, June 1998.

[14] D.J.Fleet, A.D.Jepson, and M.R.M. Jenkin. Phase-based disparity measurement. *CVGIP: Image Understanding*, 53(2):198–210, 1991.

[15] O.D. Faugeras, Q.-T. Luong, and S.J. Maybank. Camera self-calibration: theory and experiments. In *European Conference on Computer Vision*, pages 321–34, 1992.

[16] Oliver Faugeras and Giorgio Toscani. The calibration problem for stereo. In *Proceedings IEEE CVPR 86*, pages 15–20, 1986.

[17] Olivier Faugeras. *Three-Dimensional Computer Vision*. MIT Press, 1993.

[18] Olivier Faugeras, Stephane Laveau, Luc Robert, Gabriella Csurka, and Cyril Zeller. 3-d reconstruction of urban scenes from sequences of images. Technical Report 2572, INRIA, June 1995.

[19] Thomas Funkhauser and C. H. Sequin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *SIGGRAPH 93*, pages 247–254, 1993.

[20] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The Lumigraph. In *SIGGRAPH 96*, pages 43–54, 1996.

[21] W. E. L. Grimson. *From Images to Surface*. MIT Press, 1981.

[22] H. Hoppe, T. DeRose, T. DUchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *ACM SIGGRAPH 94 Proc.*, pages 295–302, 1994.

[23] William Jepson, Robin Liggett, and Scott Friedman. An environment for real-time urban simulation. In *Proceedings of the Symposium on Interactive 3D Graphics*, pages 165–166, 1995.

[24] D. Jones and J. Malik. Computational framework for determining stereo correspondence from a set of linear spatial filters. *Image and Vision Computing*, 10(10):699–708, December 1992.

[25] E. Kruppa. Zur ermittlung eines objectes aus zwei perspektiven mit innerer orientierung. *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. Ila.*, 122:1939–1948, 1913.

[26] Stephane Laveau and Olivier Faugeras. 3-D scene representation as a collection of images. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 689–691, 1994.

[27] Marc Levoy. The Digital Michaelangelo Project. http://www-graphics.stanford.edu/projects/mich/, 1999.

[28] Marc Levoy and Pat Hanrahan. Light field rendering. In *SIGGRAPH 96*, pages 31–42, 1996.

[29] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.

[30] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London*, 204:301–328, 1979.

[31] Leonard McMillan and Gary Bishop. Plenoptic Modeling: An image-based rendering system. In *SIGGRAPH 95*, 1995.

[32] Michael Naimark, John Woodfill, Paul Debevec, and Leo Villareal. Immersion '94. Interval Research Corporation image-based modeling and rendering project from Summer 1994, presented at SIGGRAPH 95 Panel "Museums Without Walls: New Media for New Museums".

[33] H. K. Nishihara. Practical real-time imaging stereo matcher. *Optical Engineering*, 23(5):536–545, 1984.

[34] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. A stereo correspondence algorithm using a disparity gradient limit. *Perception*, 14:449–470, 1985.

[35] Przemyslaw Prusinkiewicz, Mark James, and Radomir Mech. Synthetic topiary. In *SIGGRAPH 94*, pages 351–358, July 1994.

[36] Kari Pulli, Michael Cohen, Tom Duchamp, Hugues Hoppe, Linda Shapiro, , and Werner Stuetzle. View-based rendering: Visualizing real objects from scanned range and color data. In *Proceedings of 8th Eurographics Workshop on Rendering, St. Etienne, France*, pages 23–34, June 1997.

[37] Yoichi Sato, Mark D. Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. In *SIGGRAPH 97*, pages 379–387, 1997.

[38] Harpreet S. Sawhney. Simplifying motion and structure analysis using planar parallax and image warping. In *International Conference on Pattern Recognition*, 1994.

[39] D. Scharstein. Stereo vision for view synthesis. In *Computer Vision and Pattern Recognition*, June 1996.

[40] Steven M. Seitz and Charles R. Dyer. View morphing. In *SIGGRAPH 96*, pages 21–30, August 1996.

[41] H. Shum, M. Hebert, K. Ikeuchi, and R. Reddy. An integral approach to free-formed object modeling. *ICCV*, pages 870–875, 1995.

[42] M. Soucy and D. Lauendeau. Multi-resolution surface modeling from multiple range views. In *Proc. IEEE Computer Vision and Pattern Recognition*, pages 348–353, 1992.

[43] Steve Sullivan and Jean Ponce. Constructing 3d object models from photographs. Technical Sketch, Siggraph 1996, Unpublished.

[44] R. Szeliski. Image mosaicing for tele-reality applications. In *IEEE Computer Graphics and Applications*, 1996.

[45] Richard Szeliski and Rich Weiss. Robust shape recovery from occluding contours using a linear smoother. Technical Report 93/7, Digital Equipment Corporation, December 1993.

[46] Camillo J. Taylor and David J. Kriegman. Structure and motion from line segments in multiple images. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(11), November 1995.

[47] S. J. Teller and C. H. Sequin. Visibility preprocessing for interactive walkthroughs. In *SIGGRAPH 91*, pages 61–69, 1991.

[48] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[49] Roger Tsai. A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, August 1987.

[50] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *SIGGRAPH 94*, pages 311–318, 1994.

[51] S. Ullman. *The Interpretation of Visual Motion*. The MIT Press, Cambridge, MA, 1979.

[52] Lance Williams and Eric Chen. View interpolation for image synthesis. In *SIGGRAPH 93*, 1993.

[53] Y.Chen and G. Medioni. Object modeling from multiple range images. *Image and Vision Computing*, 10(3):145–155, April 1992.

[54] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *SIGGRAPH 99*, August 1999.

[55] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. In *European Conference on Computer Vision*, pages 151–158, May 1994.

Modeling and Rendering Architecture from Photographs

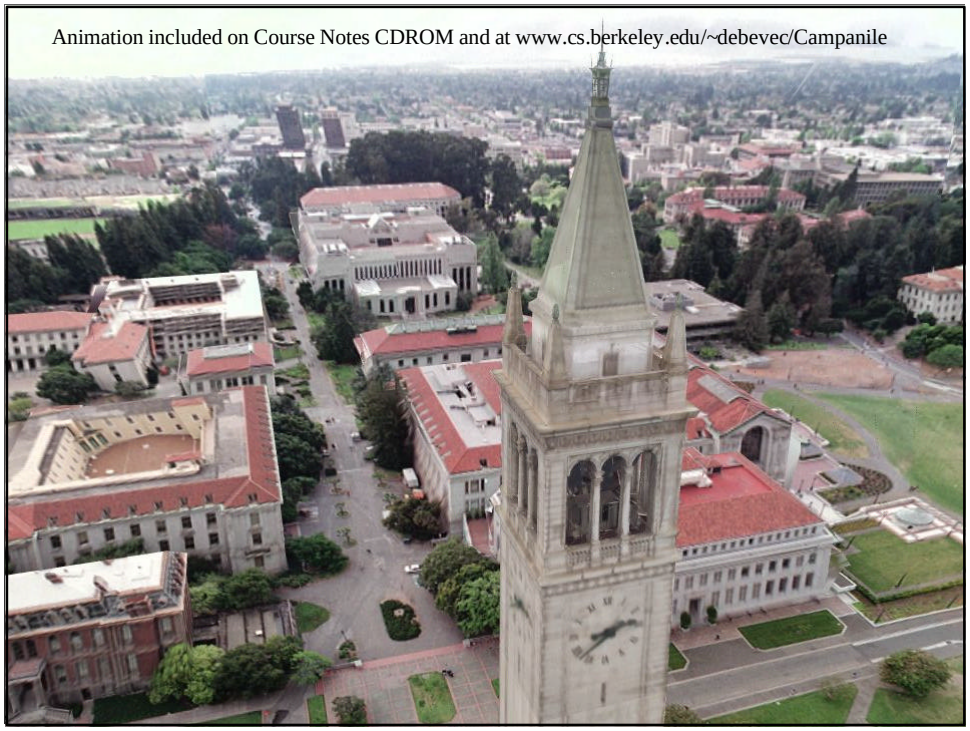
Paul Debevec

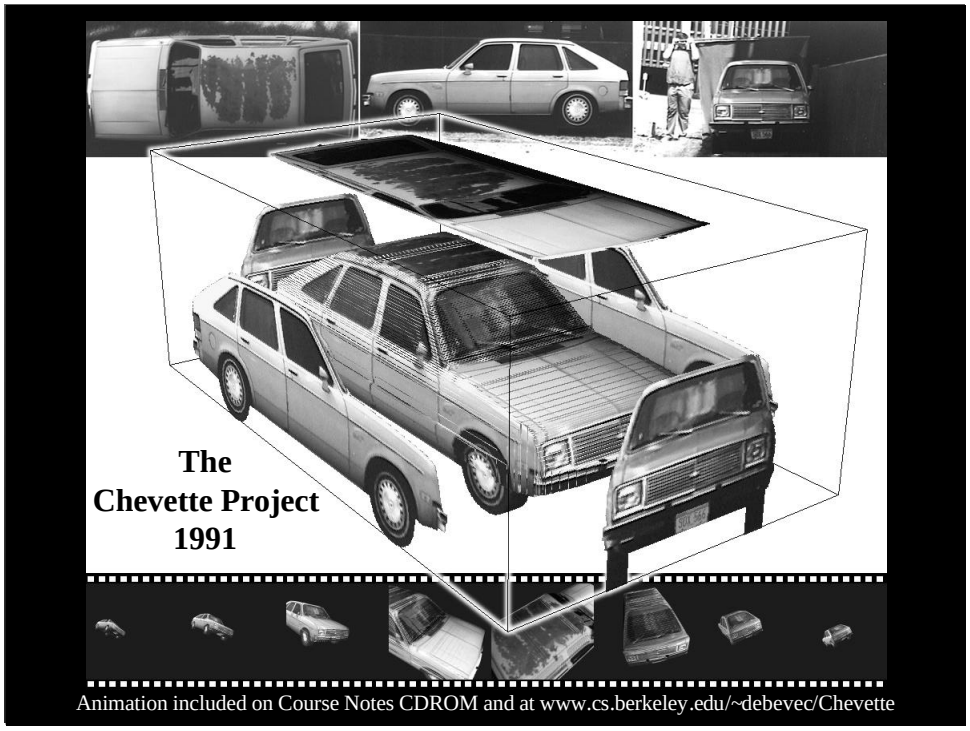
University of Southern California
Institute for Creative Technologies

SIGGRAPH 2000 Course #19, 3D Photography
Brian Curless and Steve Seitz, organizers

July 24, 2000

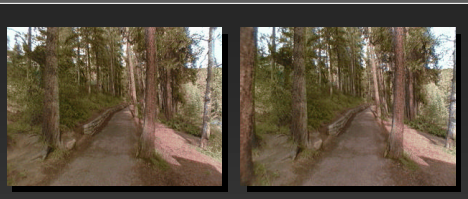
http://www.debevec.org/



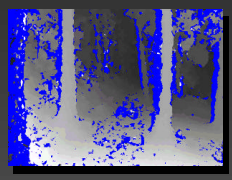


Animation included on Course Notes CDROM and at www.cs.berkeley.edu/~debevec/Chevette

Stereo Image Capture Rig



Stereo Image Pair



Depth Map

Synthetic Views

Immersion '94
Michael Naimark
John Woodfill
Paul Debevec
Leo Villareal
Ramin Zabih
Interval Research Corporation

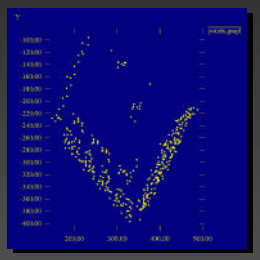
Ramin Zabih and John Woodfill. Non-parametric local transforms for determining visual correspondence. ECCV, May 1994.

Animation included on Course Notes CDROM and at www.cs.berkeley.edu/~debevec/Immersion

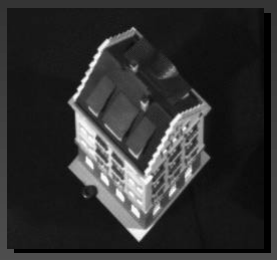
Structure from Motion

Tomasi and Kanade 1992

Image from sequence



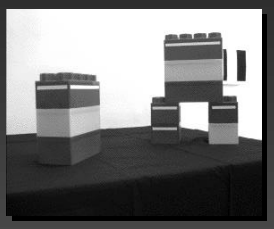
Recovered model

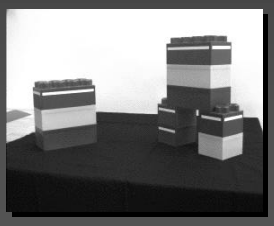


Scene viewed from same position

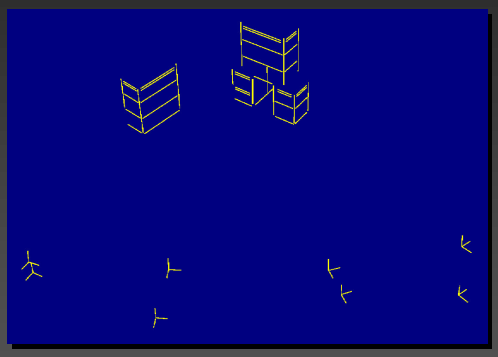
Structure from Motion

Taylor and Kriegman 1995





Two of eight original images



Recovered model

Façade's Modeling Method:

The user represents the scene as a collection of geometric primitives

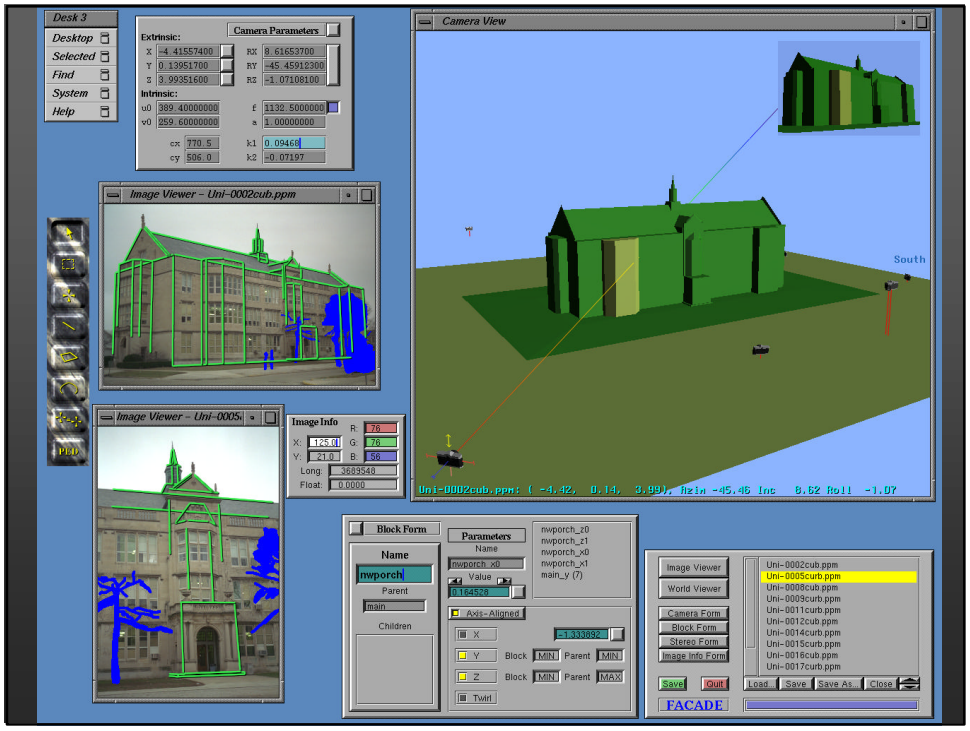
The computer solves for the sizes and positions of the blocks according to user-supplied edge correspondences

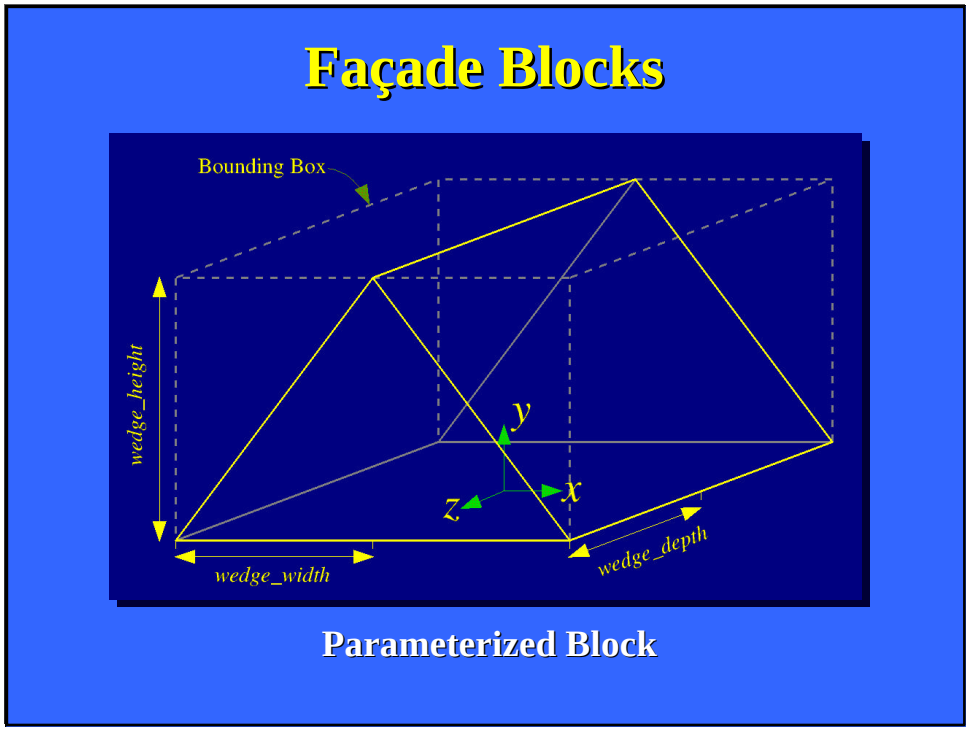
Modeling and Rendering Architecure from Photographs (Debevec, Taylor, and Malik 1996)

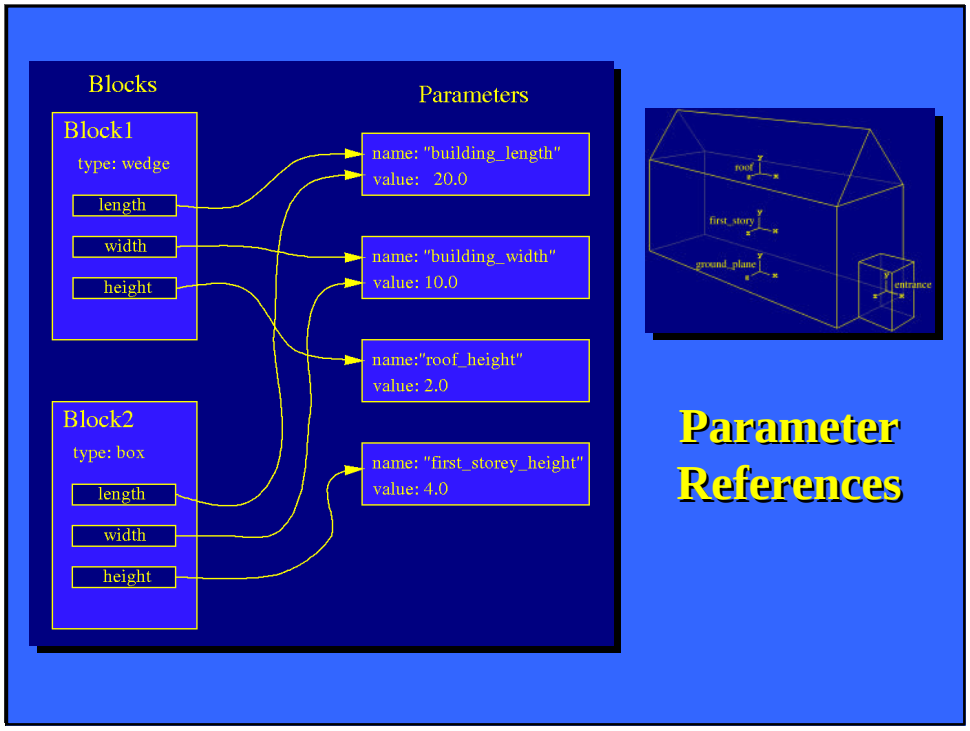
Block Model

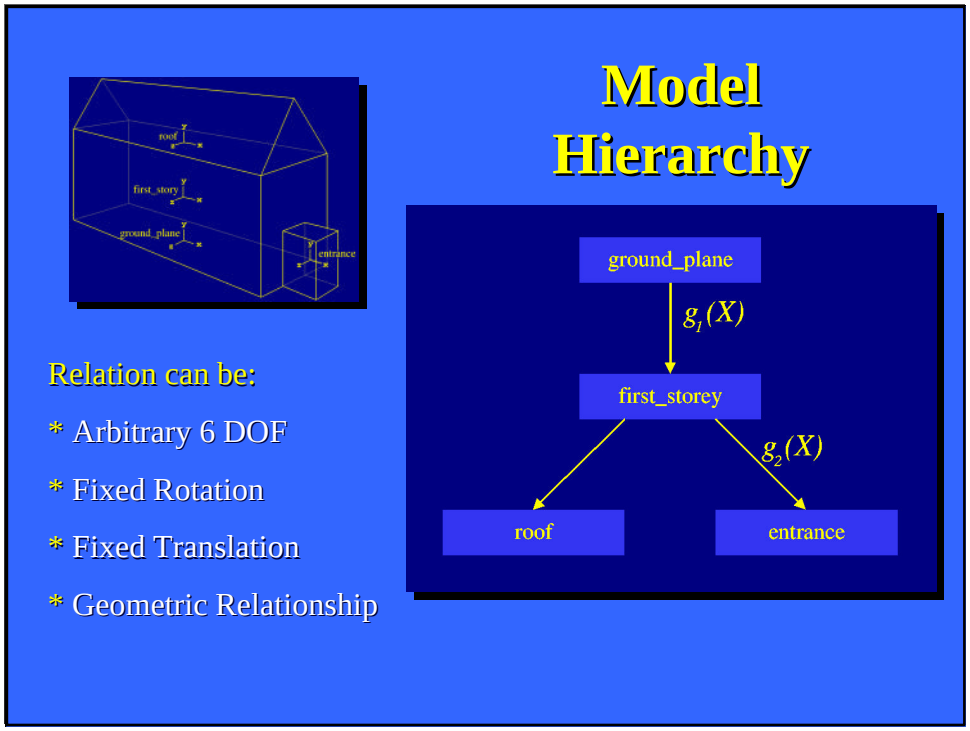
User-Marked Edges

Recovered Model


Desk 3
Desktop
Selected
Find
System
Help
Camera Parameters
Extrinsic:
Intrinsic:
Image Viewer – Uni-0002cub.ppm
Camera View
South
Image Viewer – Uni-0005
Image Info
Block Form
Name
nwporch
Parameters
Axis-Aligned
Image Viewer
World Viewer
Camera Form
Block Form
Stereo Form
Image Info Form
Save
Quit
FACADE


Façade Blocks
Bounding Box
wedge_height
wedge_width
wedge_depth
y
x
z
Parameterized Block


Blocks
Parameters
Block1
type: wedge
length
width
height
Block2
type: box
length
width
height
name: "building_length"
value: 20.0
name: "building_width"
value: 10.0
name:"roof_height"
value: 2.0
name: "first_storey_height"
value: 4.0
roof
first_story
ground_plane
entrance
Parameter References

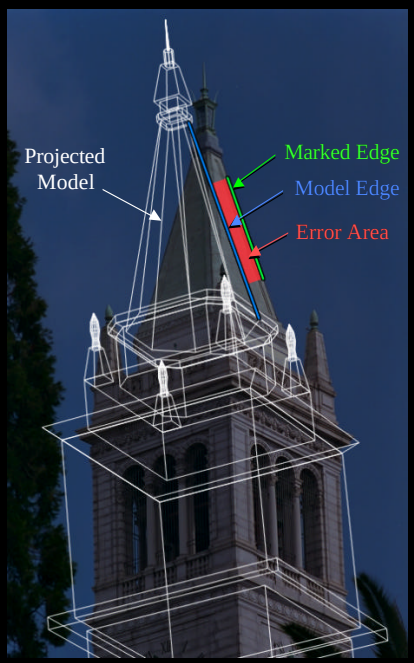

roof
first_story
ground_plane
entrance
Model Hierarchy
Relation can be:
* Arbitrary 6 DOF
* Fixed Rotation
* Fixed Translation
* Geometric Relationship
ground_plane
$g_1(X)$
first_storey
$g_2(X)$
roof
entrance

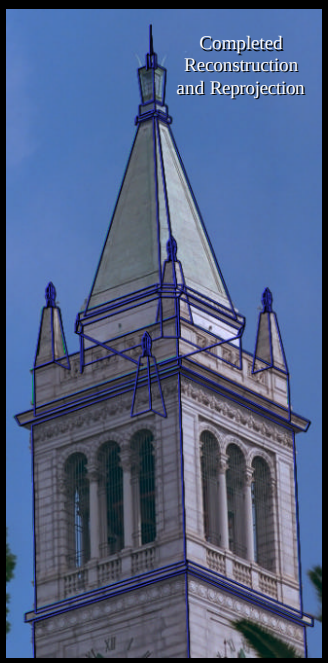
Reconstruction Algorithm

An objective function *O* measures the misalignment between the marked edges and the corresponding projected edges of the model

O is minimized with respect to the model parameters and camera positions

An initial estimate is obtained by a separate procedure





Algorithm with Initial Estimate Procedure

1. Solve for camera rotations, independently, based on edge orientations
2. Hold camea rotations fixed; solve for other parameters (often linear)
3. Perform full non-linear optimization, starting from near the solution

Photogrammetric Modeling Summary

Modeling with blocks works because:

Convenient for architecture

Recovers Complete Models

Reduces number of model parameters, e.g.

Campanile model has:

2,896 parameters as independent edges

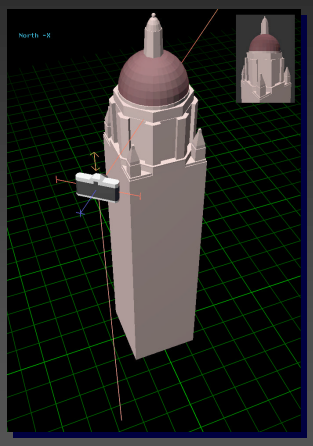
240 parameters as independent blocks

33 parameters as constrained blocks

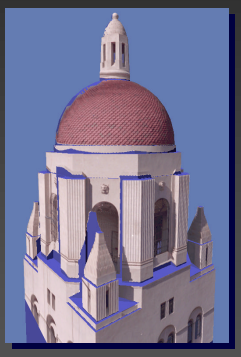
- → Few marked features required
- → Easier to solve

Surfaces of Revolution

Photograph

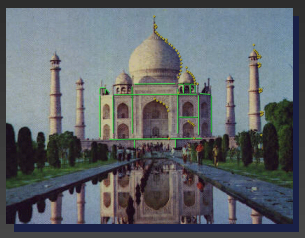


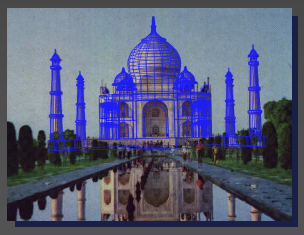
Recovered Model



Synthetic View

Arches and Surfaces of Revolution





Taj Mahal modeled from one photograph

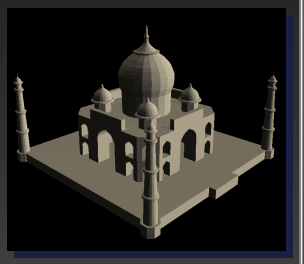


Image-Based Modeling, Rendering, and Lighting

SIGGRAPH 2000 Course #35

Tuesday, July 25, 2000

8:30am - 5:00pm

Paul Debevec
UC Berkeley

Michael Cohen
Microsoft Research

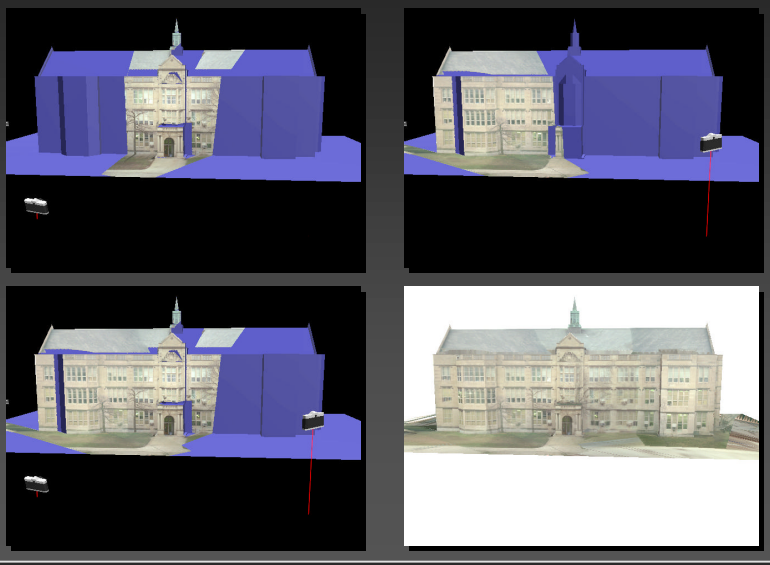
Leonard McMillan
MIT

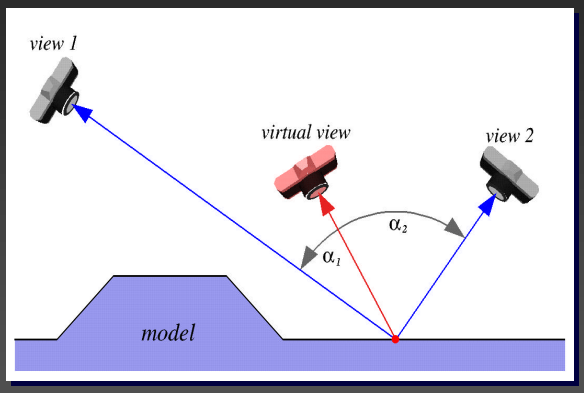
Chris Bregler
Stanford University

Richard Szeliski
Microsoft Research

François Sillion
iMAGIS - GRAVIR/IMAG

Rendering with Projective Texture Mapping

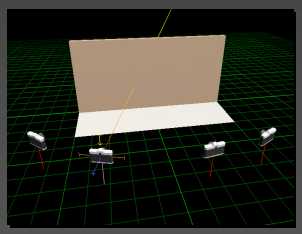




View-Dependent Weighting Function

Scene with Geometric Detail

Model-Based Stereo



Approximate Block Model

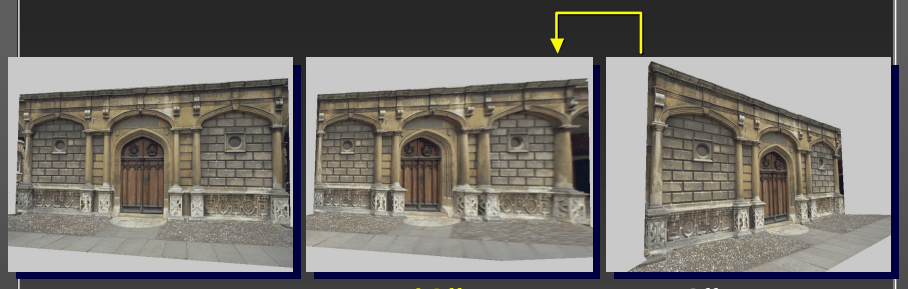
Model-Based Stereo

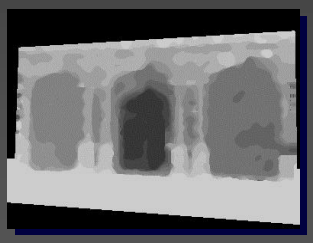
Given a key and an offset image,

- Project the offset image onto the model
- View the model through the key camera
  → Warped offset image

Stereo becomes feasible between key and warped offset images because:

- Disparities are small
- Foreshortening is greatly reduced

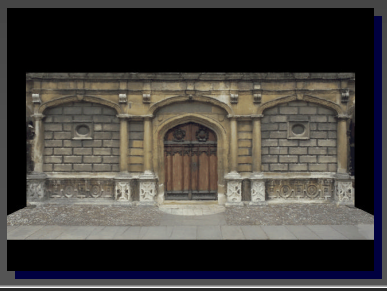




Disparity Map

Synthetic Views of Refined Model

Four images composited with Model-Based Stereo and VDTM

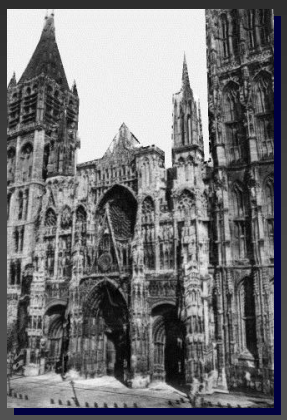


Application: Rouen Revisited

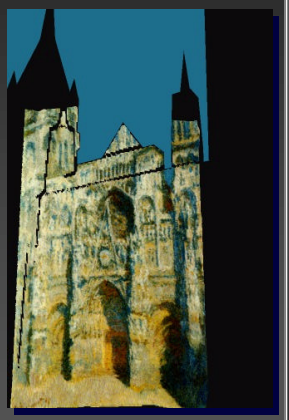
(Golan Levin and Paul Debevec)

SIGGRAPH 96 Art Show

Synthetic View: 1996



Synthetic View: 1896



Synthetic View: Monet Painting

(Uncalibrated Views)

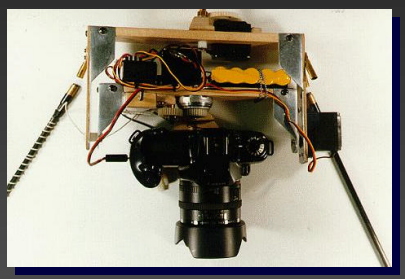
Application: The Campanile Movie

Created by: George Borshukov, Yizhou Yu, Jason Luros, Vivian Jiang, Chris Wright, Sami Khoury, Charles Benton, Tim Hawkins, Charles Ying, and Paul Debevec

Thanks to Jitendra Malik, Jeff Davis, Susan Marquez, Al Vera, Peter Bosselman, Camillo Taylor, Eric Paulos, Michael Naimark, Dorrice Pyle, Russell Bayba, Lindsay Krisel, Oliver Crow, and Peter Pletcher, as well as Charlie and Thomas Benton, Linda Branagan, John Canny, Magdalene Crowley, Brett Evans, Eva Marie Finney, Lisa Sardegna, Ellen Perry, and Camillo J. Taylor.

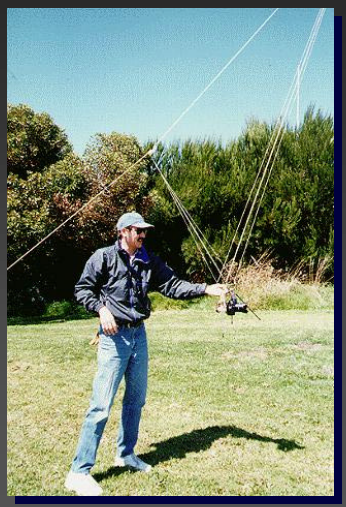
Additional thanks: the Berkeley Computer Vision Group, the Berkeley Multimedia Research Center, the Berkeley Computer Graphics Group, the ONR MURI Program, Interval Research Corporation, and Silicon Graphics, Inc.

Cris Benton: Kite Aerial Photography



http://www-archfp.ced.berkeley.edu/kap/

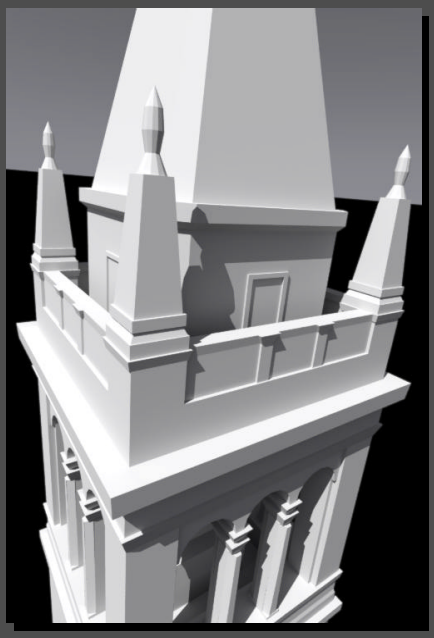
Cris Benton: Kite Aerial Photography



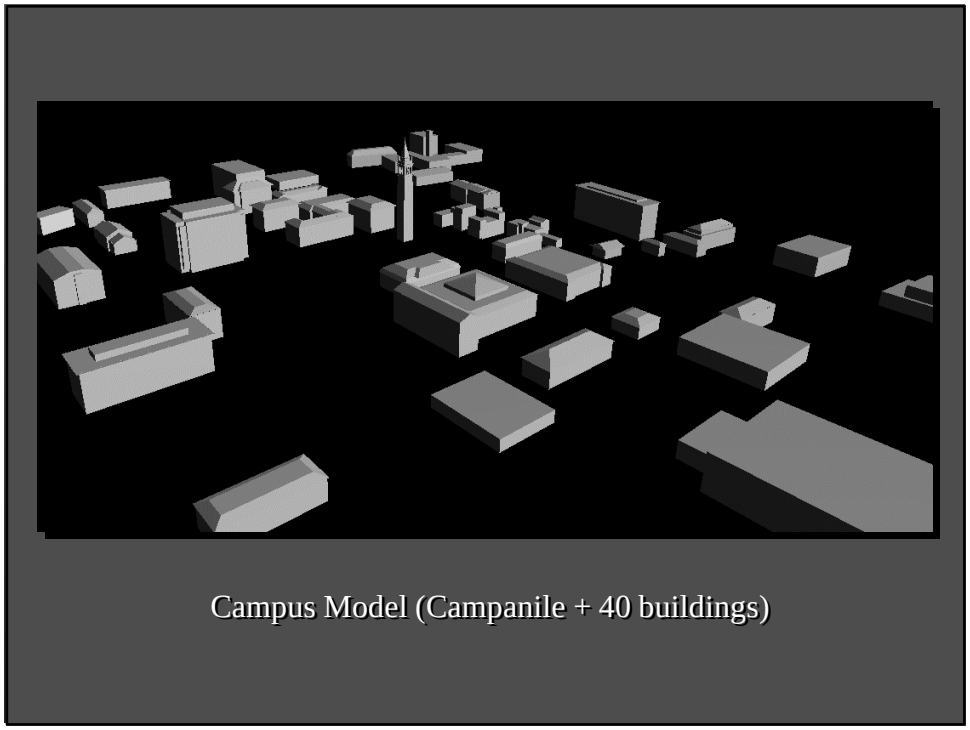
http://www-archfp.ced.berkeley.edu/kap/

Tower Photographs

Campanile Model



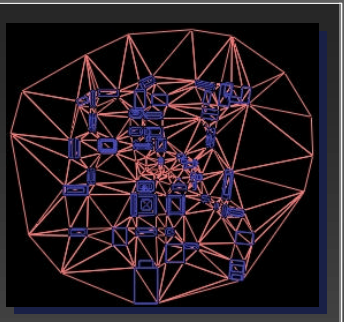
Environment Photographs

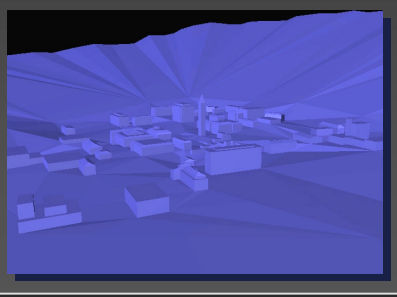


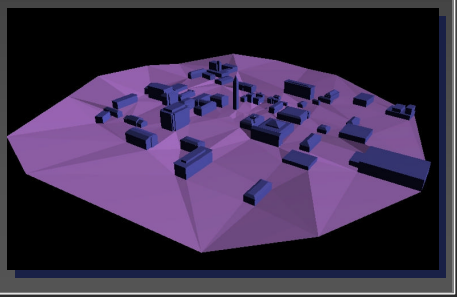
Campus Model (Campanile + 40 buildings)

Terrain Modeling

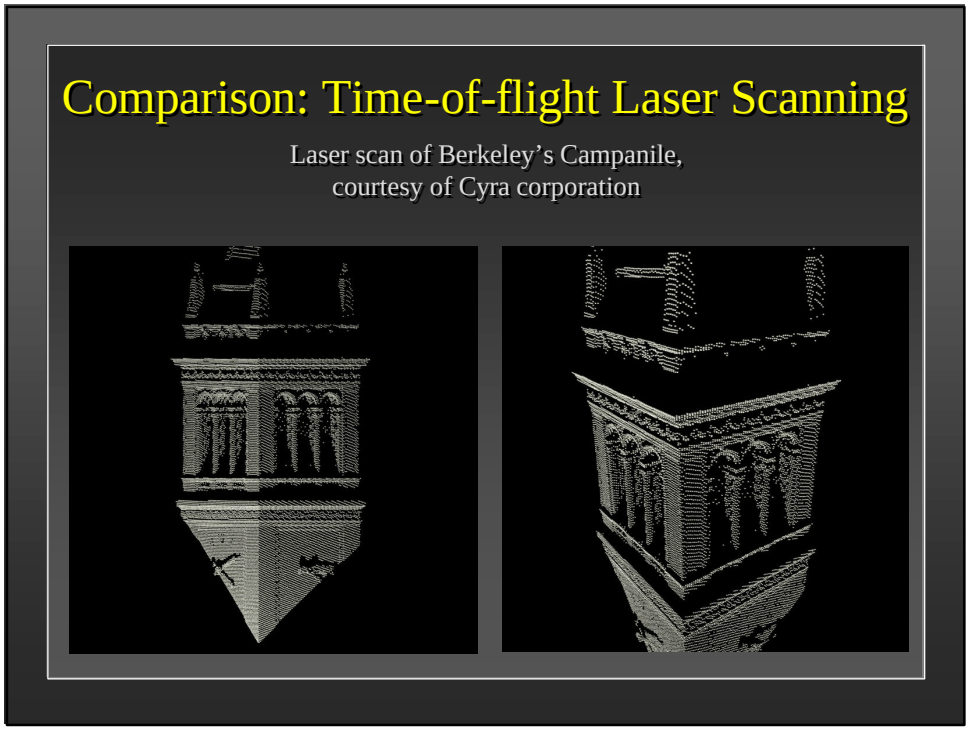
- *Delaunay triangulation of building bases + other recovered ground points*
- *Extension out to horizon*








A view from too far away

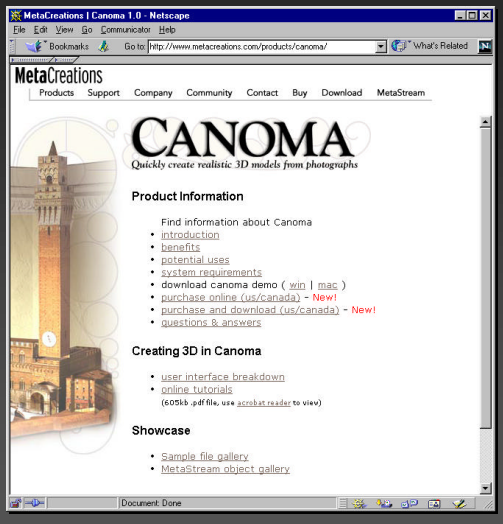

Comparison: Time-of-flight Laser Scanning
Laser scan of Berkeley's Campanile,
courtesy of Cyra corporation

Application: The Matrix

www.mvfx.com

George Borshukov, Dan Piponi, Kim Libreri, and John Gaeta, MANEX Entertainment

Commercial Product: Metacreations Canoma

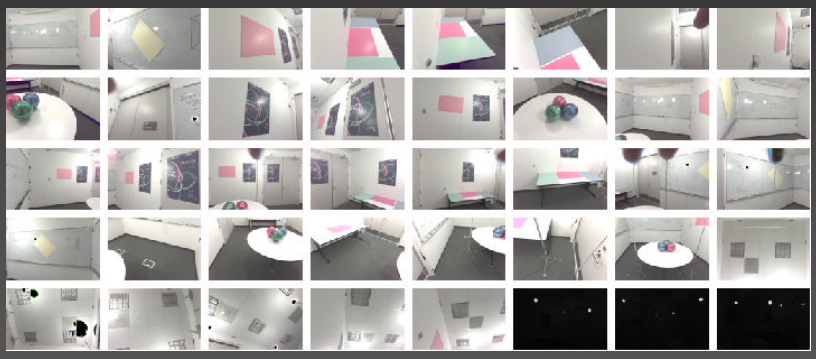


www.metacreations.com/canoma
www.canoma.com

Application: Inverse Global Illumination

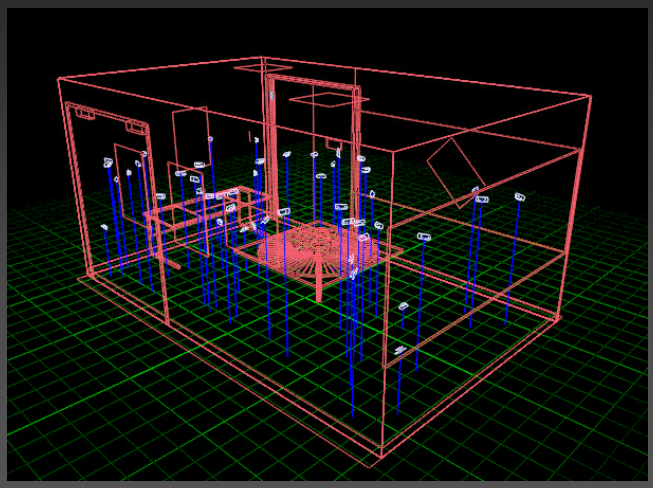
Yizhou Yu, Paul Debevec, Jitendra Malik, Tim Hawkins

SIGGRAPH 99, Thursday, 11:50-12:15pm, West Hall A



40 radiance maps of a room

Recovered Geometry and Viewpoints



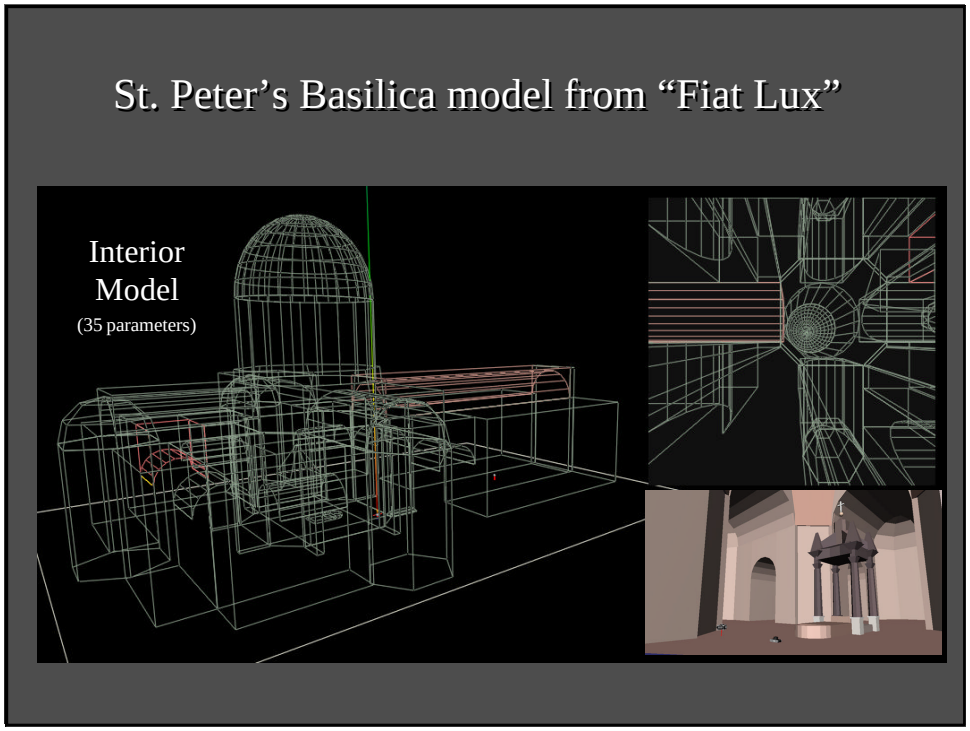
Real/Synthetic Comparison

Same viewpoints, Same lighting, Same objects

Real/Synthetic Comparison

New viewpoint, New lighting, New object


Interior Illumination Model
St. Peter's Basilica


St. Peter's Basilica model from "Fiat Lux"
Interior
Model
(35 parameters)

Thanks

George Borshukov, Christine Cheng, H-P Duiker, Tal Garfinkel, Tim Hawkins, Jenny Huang, Sami Khoury, Jason Luros, Jitendra Malik, Westley Sarokin, Camillo Taylor, Chris Wright, Yizhou Yu

Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach

Paul E. Debevec Camillo J. Taylor Jitendra Malik

University of California at Berkeley[1]

ABSTRACT

We present a new approach for modeling and rendering existing architectural scenes from a sparse set of still photographs. Our modeling approach, which combines both geometry-based and image-based techniques, has two components. The first component is a *photogrammetric modeling* method which facilitates the recovery of the basic geometry of the photographed scene. Our photogrammetric modeling approach is effective, convenient, and robust because it exploits the constraints that are characteristic of architectural scenes. The second component is a *model-based* stereo algorithm, which recovers how the real scene deviates from the basic model. By making use of the model, our stereo technique robustly recovers accurate depth from widely-spaced image pairs. Consequently, our approach can model large architectural environments with far fewer photographs than current image-based modeling approaches. For producing renderings, we present *view-dependent texture mapping*, a method of compositing multiple views of a scene that better simulates geometric detail on basic models. Our approach can be used to recover models for use in either geometry-based or image-based rendering systems. We present results that demonstrate our approach's ability to create realistic renderings of architectural scenes from viewpoints far from the original photographs.

CR Descriptors: I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding - *Modeling and recovery of physical attributes*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism - *Color, shading, shadowing, and texture* I.4.8 [**Image Processing**]: Scene Analysis - *Stereo*; J.6 [**Computer-Aided Engineering**]: Computer-aided design (CAD).

1 INTRODUCTION

Efforts to model the appearance and dynamics of the real world have produced some of the most compelling imagery in computer graphics. In particular, efforts to model architectural scenes, from the Amiens Cathedral to the Giza Pyramids to Berkeley's Soda Hall, have produced impressive walk-throughs and inspiring flybys. Clearly, it is an attractive application to be able to explore the world's architecture unencumbered by fences, gravity, customs, or jetlag.

[1]Computer Science Division, University of California at Berkeley, Berkeley, CA 94720-1776. {debevec,camillo,malik}@cs.berkeley.edu. See also http://www.cs.berkeley.edu/~debevec/Research

Unfortunately, current geometry-based methods (Fig. 1a) of modeling existing architecture, in which a modeling program is used to manually position the elements of the scene, have several drawbacks. First, the process is extremely labor-intensive, typically involving surveying the site, locating and digitizing architectural plans (if available), or converting existing CAD data (again, if available). Second, it is difficult to verify whether the resulting model is accurate. Most disappointing, though, is that the renderings of the resulting models are noticeably computer-generated; even those that employ liberal texture-mapping generally fail to resemble real photographs.

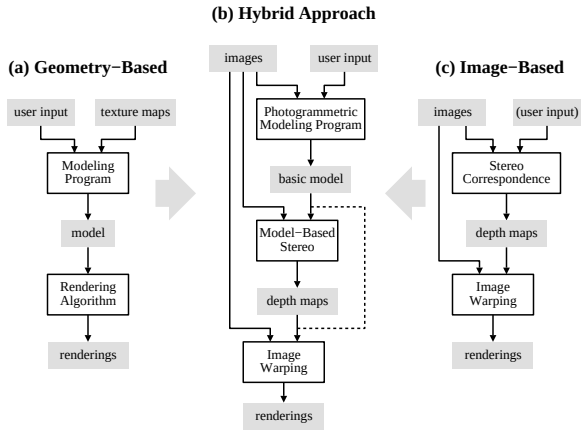


Figure 1: *Schematic of how our hybrid approach combines geometry-based and image-based approaches to modeling and rendering architecture from photographs.*

Recently, creating models directly from photographs has received increased interest in computer graphics. Since real images are used as input, such an image-based system (Fig. 1c) has an advantage in producing photorealistic renderings as output. Some of the most promising of these systems [16, 13] rely on the computer vision technique of computational stereopsis to automatically determine the structure of the scene from the multiple photographs available. As a consequence, however, these systems are only as strong as the underlying stereo algorithms. This has caused problems because state-of-the-art stereo algorithms have a number of significant weaknesses; in particular, the photographs need to appear very similar for reliable results to be obtained. Because of this, current image-based techniques must use many closely spaced images, and in some cases employ significant amounts of user input for each image pair to supervise the stereo algorithm. In this framework, capturing the data for a realistically renderable model would require an impractical number of closely spaced photographs, and deriving the depth from the photographs could require an impractical amount of user input. These concessions to the weakness of stereo algorithms bode poorly for creating large-scale, freely navigable virtual environments from photographs.

Our research aims to make the process of modeling architectural

scenes more convenient, more accurate, and more photorealistic than the methods currently available. To do this, we have developed a new approach that draws on the strengths of both geometry-based and image-based methods, as illustrated in Fig. 1b. The result is that our approach to modeling and rendering architecture requires only a sparse set of photographs and can produce realistic renderings from arbitrary viewpoints. In our approach, a basic geometric model of the architecture is recovered interactively with an easy-to-use photogrammetric modeling system, novel views are created using view-dependent texture mapping, and additional geometric detail can be recovered automatically through stereo correspondence. The final images can be rendered with current image-based rendering techniques. Because only photographs are required, our approach to modeling architecture is neither invasive nor does it require architectural plans, CAD models, or specialized instrumentation such as surveying equipment, GPS sensors or range scanners.

1.1 Background and Related Work

The process of recovering 3D structure from 2D images has been a central endeavor within computer vision, and the process of rendering such recovered structures is a subject receiving increased interest in computer graphics. Although no general technique exists to derive models from images, four particular areas of research have provided results that are applicable to the problem of modeling and rendering architectural scenes. They are: Camera Calibration, Structure from Motion, Stereo Correspondence, and Image-Based Rendering.

1.1.1 Camera Calibration

Recovering 3D structure from images becomes a simpler problem when the cameras used are *calibrated*, that is, the mapping between image coordinates and directions relative to each camera is known. This mapping is determined by, among other parameters, the camera's focal length and its pattern of radial distortion. Camera calibration is a well-studied problem both in photogrammetry and computer vision; some successful methods include [20] and [5]. While there has been recent progress in the use of uncalibrated views for 3D reconstruction [7], we have found camera calibration to be a straightforward process that considerably simplifies the problem.

1.1.2 Structure from Motion

Given the 2D projection of a point in the world, its position in 3D space could be anywhere on a ray extending out in a particular direction from the camera's optical center. However, when the projections of a sufficient number of points in the world are observed in multiple images from different positions, it is theoretically possible to deduce the 3D locations of the points as well as the positions of the original cameras, up to an unknown factor of scale.

This problem has been studied in the area of photogrammetry for the principal purpose of producing topographic maps. In 1913, Kruppa [10] proved the fundamental result that given two views of five distinct points, one could recover the rotation and translation between the two camera positions as well as the 3D locations of the points (up to a scale factor). Since then, the problem's mathematical and algorithmic aspects have been explored starting from the fundamental work of Ullman [21] and Longuet-Higgins [11], in the early 1980s. Faugeras's book [6] overviews the state of the art as of 1992. So far, a key realization has been that the recovery of structure is very sensitive to noise in image measurements when the translation between the available camera positions is small.

Attention has turned to using more than two views with image stream methods such as [19] or recursive approaches (e.g. [1]). [19] shows excellent results for the case of orthographic cameras, but direct solutions for the perspective case remain elusive. In general, linear algorithms for the problem fail to make use of all available information while nonlinear minimization methods are prone to difficulties arising from local minima in the parameter space. An alternative formulation of the problem [17] uses lines rather than points as image measurements, but the previously stated concerns were shown to remain largely valid. For purposes of computer graphics, there is yet another problem: the models recovered by these algorithms consist of sparse point fields or individual line segments, which are not directly renderable as solid 3D models.

In our approach, we exploit the fact that we are trying to recover geometric models of architectural scenes, not arbitrary three-dimensional point sets. This enables us to include additional constraints not typically available to structure from motion algorithms and to overcome the problems of numerical instability that plague such approaches. Our approach is demonstrated in a useful interactive system for building architectural models from photographs.

1.1.3 Stereo Correspondence

The geometrical theory of structure from motion assumes that one is able to solve the *correspondence* problem, which is to identify the points in two or more images that are projections of the same point in the world. In humans, corresponding points in the two slightly differing images on the retinas are determined by the visual cortex in the process called binocular stereopsis.

Years of research (e.g. [2, 4, 8, 9, 12, 15]) have shown that determining stereo correspondences by computer is difficult problem. In general, current methods are successful only when the images are similar in appearance, as in the case of human vision, which is usually obtained by using cameras that are closely spaced relative to the objects in the scene. When the distance between the cameras (often called the *baseline*) becomes large, surfaces in the images exhibit different degrees of foreshortening, different patterns of occlusion, and large disparities in their locations in the two images, all of which makes it much more difficult for the computer to determine correct stereo correspondences. Unfortunately, the alternative of improving stereo correspondence by using images taken from nearby locations has the disadvantage that computing depth becomes very sensitive to noise in image measurements.

In this paper, we show that having an approximate model of the photographed scene makes it possible to robustly determine stereo correspondences from images taken from widely varying viewpoints. Specifically, the model enables us to warp the images to eliminate unequal foreshortening and to predict major instances of occlusion *before* trying to find correspondences.

1.1.4 Image-Based Rendering

In an image-based rendering system, the model consists of a set of images of a scene and their corresponding depth maps. When the depth of every point in an image is known, the image can be re-rendered from any nearby point of view by projecting the pixels of the image to their proper 3D locations and reprojecting them onto a new image plane. Thus, a new image of the scene is created by warping the images according to their depth maps. A principal attraction of image-based rendering is that it offers a method of rendering arbitrarily complex scenes with a constant amount of computation required per pixel. Using this property, [23] demonstrated how regularly spaced synthetic images (with their computed depth maps) could be warped and composited in real time to produce a virtual environment.

More recently, [13] presented a real-time image-based rendering system that used panoramic photographs with depth computed, in part, from stereo correspondence. One finding of the paper was that extracting reliable depth estimates from stereo is "very difficult". The method was nonetheless able to obtain acceptable results for nearby views using user input to aid the stereo depth recovery: the correspondence map for each image pair was seeded with 100 to 500 user-supplied point correspondences and also post-processed. Even

with user assistance, the images used still had to be closely spaced; the largest baseline described in the paper was five feet.

The requirement that samples be close together is a serious limitation to generating a freely navigable virtual environment. Covering the size of just one city block would require thousands of panoramic images spaced five feet apart. Clearly, acquiring so many photographs is impractical. Moreover, even a dense lattice of ground-based photographs would only allow renderings to be generated from within a few feet of the original camera level, precluding any virtual fly-bys of the scene. Extending the dense lattice of photographs into three dimensions would clearly make the acquisition process even more difficult. The approach described in this paper takes advantage of the structure in architectural scenes so that it requires only a sparse set of photographs. For example, our approach has yielded a virtual fly-around of a building from just twelve standard photographs.

1.2 Overview

In this paper we present three new modeling and rendering techniques: photogrammetric modeling, view-dependent texture mapping, and model-based stereo. We show how these techniques can be used in conjunction to yield a convenient, accurate, and photorealistic method of modeling and rendering architecture from photographs. In our approach, the photogrammetric modeling program is used to create a basic volumetric model of the scene, which is then used to constrain stereo matching. Our rendering method composites information from multiple images with view-dependent texture-mapping. Our approach is successful because it splits the task of modeling from images into tasks which are easily accomplished by a person (but not a computer algorithm), and tasks which are easily performed by a computer algorithm (but not a person).

In Section 2, we present our **photogrammetric modeling** method. In essence, we have recast the structure from motion problem not as the recovery of individual point coordinates, but as the recovery of the parameters of a constrained hierarchy of parametric primitives. The result is that accurate architectural models can be recovered robustly from just a few photographs and with a minimal number of user-supplied correspondences.

In Section 3, we present **view-dependent texture mapping**, and show how it can be used to realistically render the recovered model. Unlike traditional texture-mapping, in which a single static image is used to color in each face of the model, view-dependent texture mapping interpolates between the available photographs of the scene depending on the user's point of view. This results in more lifelike animations that better capture surface specularities and unmodeled geometric detail.

Lastly, in Section 4, we present **model-based stereo**, which is used to automatically refine a basic model of a photographed scene. This technique can be used to recover the structure of architectural ornamentation that would be difficult to recover with photogrammetric modeling. In particular, we show that projecting pairs of images onto an initial approximate model allows conventional stereo techniques to robustly recover very accurate depth measurements from images with widely varying viewpoints.

2 Photogrammetric Modeling

In this section we present our method for photogrammetric modeling, in which the computer determines the parameters of a hierarchical model of parametric polyhedral primitives to reconstruct the architectural scene. We have implemented this method in *Façade*, an easy-to-use interactive modeling program that allows the user to construct a geometric model of a scene from digitized photographs. We first overview Façade from the point of view of the user, then we describe our model representation, and then we explain our reconstruction algorithm. Lastly, we present results from using Façade to reconstruct several architectural scenes.

2.1 The User's View

Constructing a geometric model of an architectural scene using Façade is an incremental and straightforward process. Typically, the user selects a small number of photographs to begin with, and models the scene one piece at a time. The user may refine the model and include more images in the project until the model meets the desired level of detail.

Fig. 2(a) and (b) shows the two types of windows used in Façade: image viewers and model viewers. The user instantiates the components of the model, marks edges in the images, and corresponds the edges in the images to the edges in the model. When instructed, Façade computes the sizes and relative positions of the model components that best fit the edges marked in the photographs.

Components of the model, called *blocks*, are parameterized geometric primitives such as boxes, prisms, and surfaces of revolution. A box, for example, is parameterized by its length, width, and height. The user models the scene as a collection of such blocks, creating new block classes as desired. Of course, the user does not need to specify numerical values for the blocks' parameters, since these are recovered by the program.

The user may choose to constrain the sizes and positions of any of the blocks. In Fig. 2(b), most of the blocks have been constrained to have equal length and width. Additionally, the four pinnacles have been constrained to have the same shape. Blocks may also be placed in constrained relations to one other. For example, many of the blocks in Fig. 2(b) have been constrained to sit centered and on top of the block below. Such constraints are specified using a graphical 3D interface. When such constraints are provided, they are used to simplify the reconstruction problem.

The user marks edge features in the images using a point-and-click interface; a gradient-based technique as in [14] can be used to align the edges with sub-pixel accuracy. We use edge rather than point features since they are easier to localize and less likely to be completely obscured. Only a section of each edge needs to be marked, making it possible to use partially visible edges. For each marked edge, the user also indicates the corresponding edge in the model. Generally, accurate reconstructions are obtained if there are as many correspondences in the images as there are free camera and model parameters. Thus, Façade reconstructs scenes accurately even when just a portion of the visible edges and marked in the images, and when just a portion of the model edges are given correspondences.

At any time, the user may instruct the computer to reconstruct the scene. The computer then solves for the parameters of the model that cause it to align with the marked features in the images. During the reconstruction, the computer computes and displays the locations from which the photographs were taken. For simple models consisting of just a few blocks, a full reconstruction takes only a few seconds; for more complex models, it can take a few minutes. For this reason, the user can instruct the computer to employ faster but less precise reconstruction algorithms (see Sec. 2.4) during the intermediate stages of modeling.

To verify the the accuracy of the recovered model and camera positions, Façade can project the model into the original photographs. Typically, the projected model deviates from the photographs by less than a pixel. Fig. 2(c) shows the results of projecting the edges of the model in Fig. 2(b) into the original photograph.

Lastly, the user may generate novel views of the model by positioning a virtual camera at any desired location. Façade will then use the view-dependent texture-mapping method of Section 3 to render a novel view of the scene from the desired location. Fig. 2(d) shows an aerial rendering of the tower model.

2.2 Model Representation

The purpose of our choice of model representation is to represent the scene as a surface model with as few parameters as possible: when

(a) (b) (c) (d)

Figure 2: **(a)** *A photograph of the Campanile, Berkeley's clock tower, with marked edges shown in green.* **(b)** *The model recovered by our photogrammetric modeling method. Although only the left pinnacle was marked, the remaining three (including one not visible) were recovered from symmetrical constraints in the model. Our method allows any number of images to be used, but in this case constraints of symmetry made it possible to recover an accurate 3D model from a single photograph.* **(c)** *The accuracy of the model is verified by reprojecting it into the original photograph through the recovered camera position.* **(d)** *A synthetic view of the Campanile generated using the view-dependent texture-mapping method described in Section 3. A real photograph from this position would be difficult to take since the camera position is 250 feet above the ground.*

the model has fewer parameters, the user needs to specify fewer correspondences, and the computer can reconstruct the model more efficiently. In Façade, the scene is represented as a constrained hierarchical model of parametric polyhedral primitives, called *blocks*. Each block has a small set of parameters which serve to define its size and shape. Each coordinate of each vertex of the block is then expressed as linear combination of the block's parameters, relative to an internal coordinate frame. For example, for the wedge block in Fig. 3, the coordinates of the vertex P_o are written in terms of the block parameters $width$, $height$, and $length$ as $P_o = (-width, -height, length)^T$. Each block is also given an associated bounding box.

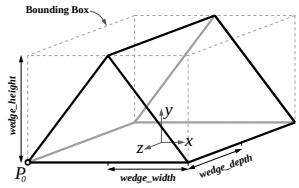
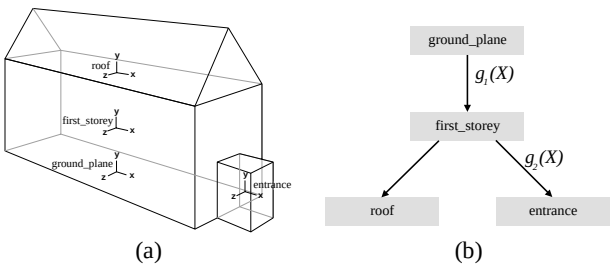


Figure 3: *A wedge block with its parameters and bounding box.*



(a) (b)

Figure 4: **(a)** *A geometric model of a simple building.* **(b)** *The model's hierarchical representation. The nodes in the tree represent parametric primitives (called* blocks*) while the links contain the spatial relationships between the blocks.*

The blocks in Façade are organized in a hierarchical tree structure as shown in Fig. 4(b). Each node of the tree represents an individual block, while the links in the tree contain the spatial relationships between blocks, called *relations*. Such hierarchical structures are also used in traditional modeling systems.

The relation between a block and its parent is most generally represented as a rotation matrix R and a translation vector t. This representation requires six parameters: three each for R and t. In architectural scenes, however, the relationship between two blocks usually has a simple form that can be represented with fewer parameters, and Façade allows the user to build such constraints on R and t into the model. The rotation R between a block and its parent can be specified in one of three ways: first, as an unconstrained rotation, requiring three parameters; second, as a rotation about a particular coordinate axis, requiring just one parameter; or third, as a fixed or null rotation, requiring no parameters.

Likewise, Façade allows for constraints to be placed on each component of the translation vector t. Specifically, the user can constrain the bounding boxes of two blocks to align themselves in some manner along each dimension. For example, in order to ensure that the roof block in Fig. 4 lies on top of the first story block, the user can require that the maximum y extent of the first story block be equal to the minimum y extent of the roof block. With this constraint, the translation along the y axis is computed ($t_y = (first\_story_y^{MAX} - roof_y^{MIN})$) rather than represented as a parameter of the model.

Each parameter of each instantiated block is actually a reference to a named symbolic variable, as illustrated in Fig. 5. As a result, two parameters of different blocks (or of the same block) can be equated by having each parameter reference the same symbol. This facility allows the user to equate two or more of the dimensions in a model, which makes modeling symmetrical blocks and repeated structure more convenient. Importantly, these constraints reduce the number of degrees of freedom in the model, which, as we will show, simplifies the structure recovery problem.

Once the blocks and their relations have been parameterized, it is straightforward to derive expressions for the world coordinates of the block vertices. Consider the set of edges which link a specific block in the model to the ground plane as shown in Fig. 4.

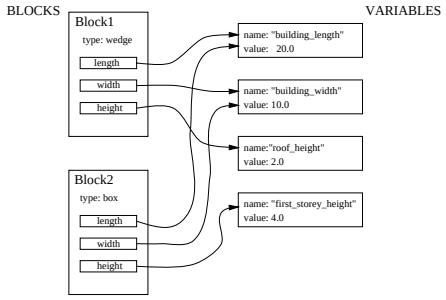


Figure 5: *Representation of block parameters as symbol references. A single variable can be referenced by the model in multiple places, allowing constraints of symmetry to be embedded in the model.*

Let $g_1(X), ..., g_n(X)$ represent the rigid transformations associated with each of these links, where X represents the vector of all the model parameters. The world coordinates $P_w(X)$ of a particular block vertex $P(X)$ is then:

$$P_w(X) = g_1(X)...g_n(X)P(X) \tag{1}$$

Similarly, the world orientation $v_w(X)$ of a particular line segment $v(X)$ is:

$$v_w(X) = g_1(X)...g_n(X)v(X) \tag{2}$$

In these equations, the point vectors P and P_w and the orientation vectors v and v_w are represented in homogeneous coordinates.

Modeling the scene with polyhedral blocks, as opposed to points, line segments, surface patches, or polygons, is advantageous for a number of reasons:

- Most architectural scenes are well modeled by an arrangement of geometric primitives.
- Blocks implicitly contain common architectural elements such as parallel lines and right angles.
- Manipulating block primitives is convenient since they are at a suitably high level of abstraction; individual features such as points and lines are less manageable.
- A surface model of the scene is readily obtained from the blocks, so there is no need to infer surfaces from discrete features.
- Modeling in terms of blocks and relationships greatly reduces the number of parameters that the reconstruction algorithm needs to recover.

The last point is crucial to the robustness of our reconstruction algorithm and the viability of our modeling system, and is illustrated best with an example. The model in Fig. 2 is parameterized by just 33 variables (the unknown camera position adds six more). If each block in the scene were unconstrained (in its dimensions and position), the model would have 240 parameters; if each line segment in the scene were treated independently, the model would have 2,896 parameters. This reduction in the number of parameters greatly enhances the robustness and efficiency of the method as compared to traditional structure from motion algorithms. Lastly, since the number of correspondences needed to suitably overconstrain the minimization is roughly proportional to the number of parameters in the model, this reduction means that the number of correspondences required of the user is manageable.

2.3 Reconstruction Algorithm

Our reconstruction algorithm works by minimizing an objective function $\mathcal{O}$ that sums the disparity between the projected edges of the model and the edges marked in the images, i.e. $\mathcal{O} = \sum Err_i$ where Err_i represents the disparity computed for edge feature i. Thus, the unknown model parameters and camera positions are computed by minimizing $\mathcal{O}$ with respect to these variables. Our system uses the the error function Err_i from [17], described below.

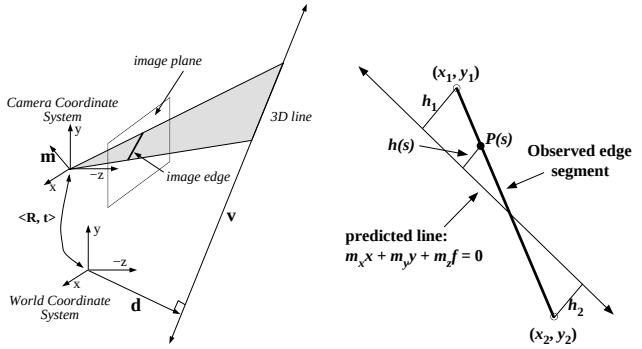


Figure 6: **(a)** *Projection of a straight line onto a camera's image plane.* **(b)** *The error function used in the reconstruction algorithm. The heavy line represents the observed edge segment (marked by the user) and the lighter line represents the model edge predicted by the current camera and model parameters.*

Fig. 6(a) shows how a straight line in the model projects onto the image plane of a camera. The straight line can be defined by a pair of vectors $\langle v, d \rangle$ where v represents the direction of the line and d represents a point on the line. These vectors can be computed from equations 2 and 1 respectively. The position of the camera with respect to world coordinates is given in terms of a rotation matrix R_j and a translation vector t_j. The normal vector denoted by $\mathbf{m}$ in the figure is computed from the following expression:

$$\mathbf{m} = R_j(\mathbf{v} \times (\mathbf{d} - t_j)) \tag{3}$$

The projection of the line onto the image plane is simply the intersection of the plane defined by m with the image plane, located at $z = -f$ where f is the focal length of the camera. Thus, the image edge is defined by the equation $m_x x + m_y y - m_z f = 0$.

Fig. 6(b) shows how the error between the observed image edge $\{(x_1, y_1), (x_2, y_2)\}$ and the predicted image line is calculated for each correspondence. Points on the observed edge segment can be parameterized by a single scalar variable $s \in [0, l]$ where l is the length of the edge. We let $h(s)$ be the function that returns the shortest distance from a point on the segment, $p(s)$, to the predicted edge.

With these definitions, the total error between the observed edge segment and the predicted edge is calculated as:

$$Err_i = \int_0^l h^2(s)ds = \frac{l}{3}(h_1^2 + h_1 h_2 + h_2^2) = \mathbf{m}^T(A^T B A)\mathbf{m} \tag{4}$$

where:

$$\begin{aligned} \mathbf{m} &= (m_x, m_y, m_z)^T \\ A &= \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{pmatrix} \\ B &= \frac{l}{3(m_x^2 + m_y^2)} \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} \end{aligned}$$

The final objective function $\mathcal{O}$ is the sum of the error terms resulting from each correspondence. We minimize $\mathcal{O}$ using a variant of the Newton-Raphson method, which involves calculating the gradient and Hessian of $\mathcal{O}$ with respect to the parameters of the camera

and the model. As we have shown, it is simple to construct symbolic expressions for m in terms of the unknown model parameters. The minimization algorithm differentiates these expressions symbolically to evaluate the gradient and Hessian after each iteration. The procedure is inexpensive since the expressions for d and v in Equations 2 and 1 have a particularly simple form.

2.4 Computing an Initial Estimate

The objective function described in Section 2.3 section is non-linear with respect to the model and camera parameters and consequently can have local minima. If the algorithm begins at a random location in the parameter space, it stands little chance of converging to the correct solution. To overcome this problem we have developed a method to directly compute a good initial estimate for the model parameters and camera positions that is near the correct solution. In practice, our initial estimate method consistently enables the non-linear minimization algorithm to converge to the correct solution.

Our initial estimate method consists of two procedures performed in sequence. The first procedure estimates the camera rotations while the second estimates the camera translations and the parameters of the model. Both initial estimate procedures are based upon an examination of Equation 3. From this equation the following constraints can be deduced:

$$m^T R_j \mathbf{v} = 0 \quad (5)$$

$$m^T R_j (\mathbf{d} - t_j) = 0 \quad (6)$$

Given an observed edge $\mathbf{u}_{ij}$ the measured normal $\mathbf{m}'$ to the plane passing through the camera center is:

$$\mathbf{m}' = \begin{pmatrix} x_1 \\ y_1 \\ -f \end{pmatrix} \times \begin{pmatrix} x_2 \\ y_2 \\ -f \end{pmatrix} \quad (7)$$

From these equations, we see that any model edges of known orientation constrain the possible values for R_j. Since most architectural models contain many such edges (e.g. horizontal and vertical lines), each camera rotation can be usually be estimated from the model independent of the model parameters and independent of the camera's location in space. Our method does this by minimizing the following objective function $\mathcal{O}_1$ that sums the extents to which the rotations R_j violate the constraints arising from Equation 5:

$$\mathcal{O}_1 = \sum_i (m^T R_j \mathbf{v}_i)^2, \quad \mathbf{v}_i \in \{\hat{x}, \hat{y}, \hat{z}\} \quad (8)$$

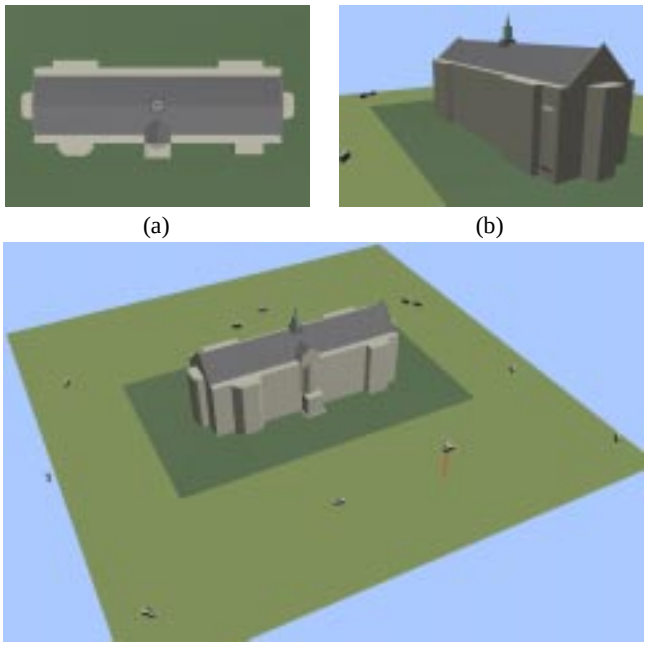
Once initial estimates for the camera rotations are computed, Equation 6 is used to obtain initial estimates of the model parameters and camera locations. Equation 6 reflects the constraint that all of the points on the line defined by the tuple $\langle \mathbf{v}, \mathbf{d} \rangle$ should lie on the plane with normal vector $\mathbf{m}$ passing through the camera center. This constraint is expressed in the following objective function $\mathcal{O}_2$ where $P_i(X)$ and $Q_i(X)$ are expressions for the vertices of an edge of the model.

$$\mathcal{O}_2 = \sum_i (m^T R_j (P_i(X) - t_j))^2 + (m^T R_j (Q_i(X) - t_j))^2 \quad (9)$$

In the special case where all of the block relations in the model have a known rotation, this objective function becomes a simple quadratic form which is easily minimized by solving a set of linear equations.

Once the initial estimate is obtained, the non-linear minimization over the entire parameter space is applied to produce the best possible reconstruction. Typically, the minimization requires fewer than ten iterations and adjusts the parameters of the model by at most a few percent from the initial estimates. The edges of the recovered models typically conform to the original photographs to within a pixel.

Figure 7: *Three of twelve photographs used to reconstruct the entire exterior of University High School in Urbana, Illinois. The superimposed lines indicate the edges the user has marked.*



(a) (b)

(c)

Figure 8: *The high school model, reconstructed from twelve photographs.* **(a)** *Overhead view.* **(b)** *Rear view.* **(c)** *Aerial view showing the recovered camera positions. Two nearly coincident cameras can be observed in front of the building; their photographs were taken from the second story of a building across the street.*

Figure 9: *A synthetic view of University High School. This is a frame from an animation of flying around the entire building.*

(a) (b) (c)

Figure 10: *Reconstruction of Hoover Tower, Stanford, CA* **(a)** *Original photograph, with marked edges indicated.* **(b)** *Model recovered from the single photograph shown in (a).* **(c)** *Texture-mapped aerial view from the virtual camera position indicated in (b). Regions not seen in (a) are indicated in blue.*

2.5 Results

Fig. 2 showed the results of using Façade to reconstruct a clock tower from a single image. Figs. 7 and 8 show the results of using Façade to reconstruct a high school building from twelve photographs. (The model was originally constructed from just five images; the remaining images were added to the project for purposes of generating renderings using the techniques of Section 3.) The photographs were taken with a calibrated 35mm still camera with a standard 50mm lens and digitized with the PhotoCD process. Images at the 1536×1024 pixel resolution were processed to correct for lens distortion, then filtered down to 768×512 pixels for use in the modeling system. Fig. 8 shows some views of the recovered model and camera positions, and Fig. 9 shows a synthetic view of the building generated by the technique in Sec. 3.

Fig. 10 shows the reconstruction of another tower from a single photograph. The dome was modeled specially since the reconstruction algorithm does not recover curved surfaces. The user constrained a two-parameter hemisphere block to sit centered on top of the tower, and manually adjusted its height and width to align with the photograph. Each of the models presented took approximately four hours to create.

3 View-Dependent Texture-Mapping

In this section we present view-dependent texture-mapping, an effective method of rendering the scene that involves projecting the original photographs onto the model. This form of texture-mapping is most effective when the model conforms closely to the actual structure of the scene, and when the original photographs show the scene in similar lighting conditions. In Section 4 we will show how view-dependent texture-mapping can be used in conjunction with model-based stereo to produce realistic renderings when the recovered model only approximately models the structure of the scene.

Since the camera positions of the original photographs are recovered during the modeling phase, projecting the images onto the model is straightforward. In this section we first describe how we project a single image onto the model, and then how we merge several image projections to render the entire model. Unlike traditional texture-mapping, our method projects different images onto the model depending on the user's viewpoint. As a result, our view-dependent texture mapping can give a better illusion of additional geometric detail in the model.

3.1 Projecting a Single Image

The process of texture-mapping a single image onto the model can be thought of as replacing each camera with a slide projector that projects the original image onto the model. When the model is not convex, it is possible that some parts of the model will shadow others with respect to the camera. While such shadowed regions could be determined using an object-space visible surface algorithm, or an image-space ray casting algorithm, we use an image-space shadow map algorithm based on [22] since it is efficiently implemented using z-buffer hardware.

Fig. 11, upper left, shows the results of mapping a single image onto the high school building model. The recovered camera position for the projected image is indicated in the lower left corner of the image. Because of self-shadowing, not every point on the model within the camera's viewing frustum is mapped.

3.2 Compositing Multiple Images

In general, each photograph will view only a piece of the model. Thus, it is usually necessary to use multiple images in order to render the entire model from a novel point of view. The top images of Fig. 11 show two different images mapped onto the model and rendered from a novel viewpoint. Some pixels are colored in just one of the renderings, while some are colored in both. These two renderings can be merged into a composite rendering by considering the corresponding pixels in the rendered views. If a pixel is mapped in only one rendering, its value from that rendering is used in the composite. If it is mapped in more than one rendering, the renderer has to decide which image (or combination of images) to use.

It would be convenient, of course, if the projected images would agree perfectly where they overlap. However, the images will not necessarily agree if there is unmodeled geometric detail in the building, or if the surfaces of the building exhibit non-Lambertian reflection. In this case, the best image to use is clearly the one with the viewing angle closest to that of the rendered view. However, using the image closest in angle at every pixel means that neighboring rendered pixels may be sampled from different original images. When this happens, specularity and unmodeled geometric detail can cause visible seams in the rendering. To avoid this problem, we smooth these transitions through weighted averaging as in Fig. 12.

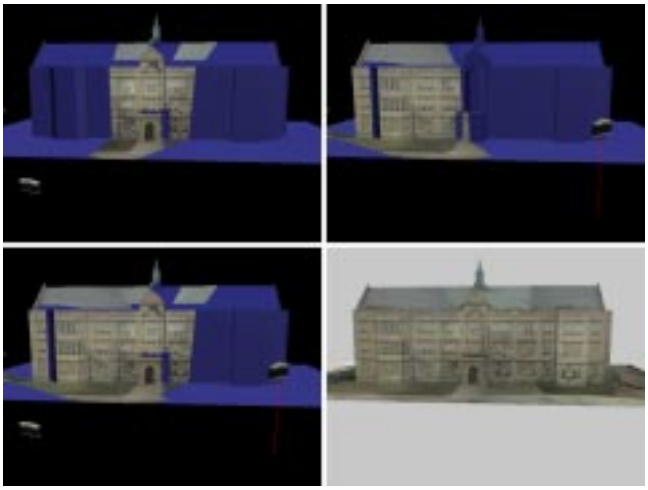


Figure 11: *The process of assembling projected images to form a composite rendering. The top two pictures show two images projected onto the model from their respective recovered camera positions. The lower left picture shows the results of compositing these two renderings using our view-dependent weighting function. The lower right picture shows the results of compositing renderings of all twelve original images. Some pixels near the front edge of the roof not seen in any image have been filled in with the hole-filling algorithm from [23].*

Even with this weighting, neighboring pixels can still be sampled from different views at the boundary of a projected image, since the contribution of an image must be zero outside its boundary. To

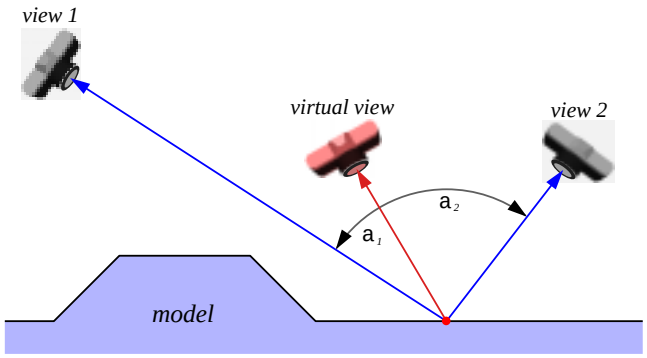


Figure 12: *The weighting function used in view-dependent texture mapping. The pixel in the virtual view corresponding to the point on the model is assigned a weighted average of the corresponding pixels in actual views 1 and 2. The weights w_1 and w_2 are inversely inversely proportional to the magnitude of angles α_1 and α_2. Alternately, more sophisticated weighting functions based on expected foreshortening and image resampling can be used.*

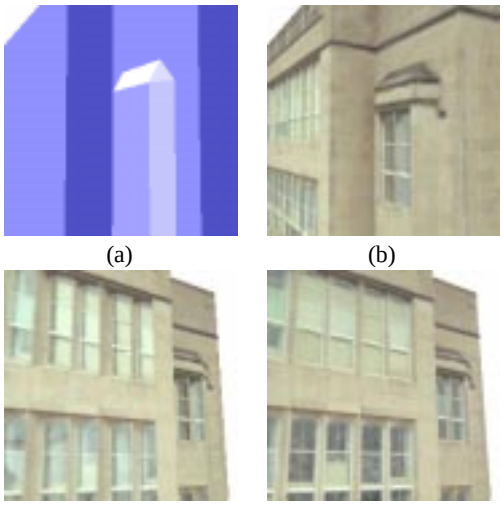


Figure 13: *View-dependent texture mapping.* **(a)** *A detail view of the high school model.* **(b)** *A rendering of the model from the same position using view-dependent texture mapping. Note that although the model does not capture the slightly recessed windows, the windows appear properly recessed because the texture map is sampled primarily from a photograph which viewed the windows from approximately the same direction.* **(c)** *The same piece of the model viewed from a different angle, using the same texture map as in (b). Since the texture is not selected from an image that viewed the model from approximately the same angle, the recessed windows appear unnatural.* **(d)** *A more natural result obtained by using view-dependent texture mapping. Since the angle of view in (d) is different than in (b), a different composition of original images is used to texture-map the model.*

address this, the pixel weights are ramped down near the boundary of the projected images. Although this method does not guarantee smooth transitions in all cases, we have found that it eliminates most artifacts in renderings and animations arising from such seams.

If an original photograph features an unwanted car, tourist, or other object in front of the architecture of interest, the unwanted object will be projected onto the surface of the model. To prevent this from happening, the user may mask out the object by painting over the obstruction with a reserved color. The rendering algorithm will then set the weights for any pixels corresponding to the masked regions to zero, and decrease the weights of the pixels near the boundary as before to minimize seams. Any regions in the composite image which are occluded in every projected image are filled in using the hole-filling method from [23].

In the discussion so far, projected image weights are computed at every pixel of every projected rendering. Since the weighting function is smooth (though not constant) across flat surfaces, it is not generally not necessary to compute it for every pixel of every face of the model. For example, using a single weight for each face of the model, computed at the face's center, produces acceptable results. By coarsely subdividing large faces, the results are visually indistinguishable from the case where a unique weight is computed for every pixel. Importantly, this technique suggests a real-time implementation of view-dependent texture mapping using a texture-mapping graphics pipeline to render the projected views, and α-channel blending to composite them.

For complex models where most images are entirely occluded for the typical view, it can be very inefficient to project every original photograph to the novel viewpoint. Some efficient techniques to determine such visibility *a priori* in architectural scenes through spatial partitioning are presented in [18].

4 Model-Based Stereopsis

The modeling system described in Section 2 allows the user to create a basic model of a scene, but in general the scene will have additional geometric detail (such as friezes and cornices) not captured in the model. In this section we present a new method of recovering such additional geometric detail automatically through stereo correspondence, which we call *model-based* stereo. Model-based stereo differs from traditional stereo in that it measures how the actual scene deviates from the approximate model, rather than trying to measure the structure of the scene without any prior information. The model serves to place the images into a common frame of reference that makes the stereo correspondence possible even for images taken from relatively far apart. The stereo correspondence information can then be used to render novel views of the scene using image-based rendering techniques.

As in traditional stereo, given two images (which we call the *key* and *offset*), model-based stereo computes the associated depth map for the key image by determining corresponding points in the key and offset images. Like many stereo algorithms, our method is *correlation-based*, in that it attempts to determine the corresponding point in the offset image by comparing small pixel neighborhoods around the points. As such, correlation-based stereo algorithms generally require the neighborhood of each point in the key image to resemble the neighborhood of its corresponding point in the offset image.

The problem we face is that when the key and offset images are taken from relatively far apart, as is the case for our modeling method, corresponding pixel neighborhoods can be foreshortened very differently. In Figs. 14(a) and (c), pixel neighborhoods toward the right of the key image are foreshortened horizontally by nearly a factor of four in the offset image.

The key observation in model-based stereo is that even though two images of the same scene may appear very different, they appear similar after being projected onto an approximate model of the scene. In particular, projecting the offset image onto the model and viewing it from the position of the key image produces what we call the *warped offset* image, which appears very similar to the key image. The geometrically detailed scene in Fig. 14 was modeled as two flat surfaces with our modeling program, which also determined the relative camera positions. As expected, the warped offset image (Fig. 14(b)) exhibits the same pattern of foreshortening as the key image.

In model-based stereo, pixel neighborhoods are compared between the key and warped offset images rather than the key and off-

(a) *Key Image* (b) *Warped Offset Image* (c) *Offset Image* (d) *Computed Disparity Map*

Figure 14: **(a)** *and* **(c)** *Two images of the entrance to Peterhouse chapel in Cambridge, UK. The Façade program was used to model the façade and ground as a flat surfaces and to recover the relative camera positions.* **(b)** *The warped offset image, produced by projecting the offset image onto the approximate model and viewing it from the position of the key camera. This projection eliminates most of the disparity and foreshortening with respect to the key image, greatly simplifying stereo correspondence.* **(d)** *An unedited disparity map produced by our model-based stereo algorithm.*

set images. When a correspondence is found, it is simple to convert its disparity to the corresponding disparity between the key and offset images, from which the point's depth is easily calculated. Fig. 14(d) shows a disparity map computed for the key image in (a).

The reduction of differences in foreshortening is just one of several ways that the warped offset image simplifies stereo correspondence. Some other desirable properties of the warped offset image are:

- Any point in the scene which lies on the approximate model will have zero disparity between the key image and the warped offset image.
- Disparities between the key and warped offset images are easily converted to a depth map for the key image.
- Depth estimates are far less sensitive to noise in image measurements since images taken from relatively far apart can be compared.
- Places where the model occludes itself relative to the key image can be detected and indicated in the warped offset image.
- A linear epipolar geometry (Sec. 4.1) exists between the key and warped offset images, despite the warping. In fact, the epipolar lines of the warped offset image coincide with the epipolar lines of the key image.

4.1 Model-Based Epipolar Geometry

In traditional stereo, the *epipolar constraint* (see [6]) is often used to constrain the search for corresponding points in the offset image to searching along an epipolar line. This constraint simplifies stereo not only by reducing the search for each correspondence to one dimension, but also by reducing the chance of selecting a false matches. In this section we show that taking advantage of the epipolar constraint is no more difficult in model-based stereo case, despite the fact that the offset image is non-uniformly warped.

Fig. 15 shows the epipolar geometry for model-based stereo. If we consider a point P in the scene, there is a unique *epipolar plane* which passes through P and the centers of the key and offset cameras. This epipolar plane intersects the key and offset image planes in *epipolar lines* e_k and e_o. If we consider the projection p_k of P onto the key image plane, the epipolar constraint states that the corresponding point in the offset image must lie somewhere along the offset image's epipolar line.

In model-based stereo, neighborhoods in the key image are compared to the warped offset image rather than the offset image. Thus, to make use of the epipolar constraint, it is necessary to determine where the pixels on the offset image's epipolar line project to in the warped offset image. The warped offset image is formed by projecting the offset image onto the model, and then reprojecting the model onto the image plane of the key camera. Thus, the projection p_o of P in the offset image projects onto the model at Q, and then reprojects to q_k in the warped offset image. Since each of these projections occurs within the epipolar plane, any possible correspondence

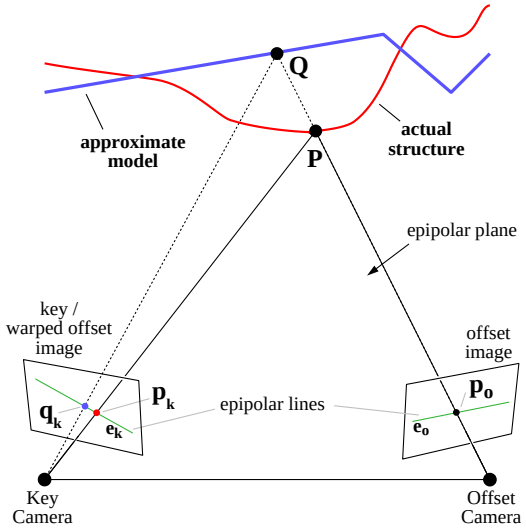


Figure 15: *Epipolar geometry for model-based stereo.*

for p_k in the key image must lie on the *key* image's epipolar line in the warped offset image. In the case where the actual structure and the model coincide at P, p_o is projected to P and then reprojected to p_k, yielding a correspondence with zero disparity.

The fact that the epipolar geometry remains linear after the warping step also facilitates the use of the ordering constraint [2, 6] through a dynamic programming technique.

4.2 Stereo Results and Rerendering

While the warping step makes it dramatically easier to determine stereo correspondences, a stereo algorithm is still necessary to actually determine them. The algorithm we developed to produce the images in this paper is described in [3].

Once a depth map has been computed for a particular image, we can rerender the scene from novel viewpoints using the methods described in [23, 16, 13]. Furthermore, when several images and their corresponding depth maps are available, we can use the view-dependent texture-mapping method of Section 3 to composite the multiple renderings. The novel views of the chapel façade in Fig. 16 were produced through such compositing of four images.

5 Conclusion and Future Work

To conclude, we have presented a new, photograph-based approach to modeling and rendering architectural scenes. Our modeling approach, which combines both geometry-based and image-based modeling techniques, is built from two components that we have developed. The first component is an easy-to-use photogrammet-

Figure 16: *Novel views of the scene generated from four original photographs. These are frames from an animated movie in which the façade rotates continuously. The depth is computed from model-based stereo and the frames are made by compositing image-based renderings with view-dependent texture-mapping.*

ric modeling system which facilitates the recovery of a basic geometric model of the photographed scene. The second component is a model-based stereo algorithm, which recovers precisely how the real scene differs from the basic model. For rendering, we have presented view-dependent texture-mapping, which produces images by warping and compositing multiple views of the scene. Through judicious use of images, models, and human assistance, our approach is more convenient, more accurate, and more photorealistic than current geometry-based or image-based approaches for modeling and rendering real-world architectural scenes.

There are several improvements and extensions that can be made to our approach. First, surfaces of revolution represent an important component of architecture (e.g. domes, columns, and minarets) that are not recovered in our photogrammetric modeling approach. (As noted, the dome in Fig. 10 was manually sized by the user.) Fortunately, there has been much work (e.g. [24]) that presents methods of recovering such structures from image contours. Curved model geometry is also entirely consistent with our approach to recovering additional detail with model-based stereo.

Second, our techniques should be extended to recognize and model the photometric properties of the materials in the scene. The system should be able to make better use of photographs taken in varying lighting conditions, and it should be able to render images of the scene as it would appear at any time of day, in any weather, and with any configuration of artificial light. Already, the recovered model can be used to predict shadowing in the scene with respect to an arbitrary light source. However, a full treatment of the problem will require estimating the photometric properties (i.e. the bidirectional reflectance distribution functions) of the surfaces in the scene.

Third, it is clear that further investigation should be made into the problem of selecting which original images to use when rendering a novel view of the scene. This problem is especially difficult when the available images are taken at arbitrary locations. Our current solution to this problem, the weighting function presented in Section 3, still allows seams to appear in renderings and does not consider issues arising from image resampling. Another form of view selection is required to choose which pairs of images should be matched to recover depth in the model-based stereo algorithm.

Lastly, it will clearly be an attractive application to integrate the models created with the techniques presented in this paper into forthcoming real-time image-based rendering systems.

Acknowledgments

This research was supported by a National Science Foundation Graduate Research Fellowship and grants from Interval Research Corporation, the California MICRO program, and JSEP contract F49620-93-C-0014. The authors also wish to thank Tim Hawkins, Carlo Séquin, David Forsyth, and Jianbo Shi for their valuable help in revising this paper.

References

[1] Ali Azarbayejani and Alex Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(6):562–575, June 1995.

[2] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In *Proceedings of the Seventh IJCAI, Vancouver, BC*, pages 631–636, 1981.

[3] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. Technical Report UCB//CSD-96-893, U.C. Berkeley, CS Division, January 1996.

[4] D.J.Fleet, A.D.Jepson, and M.R.M. Jenkin. Phase-based disparity measurement. *CVGIP: Image Understanding*, 53(2):198–210, 1991.

[5] Oliver Faugeras and Giorgio Toscani. The calibration problem for stereo. In *Proceedings IEEE CVPR 86*, pages 15–20, 1986.

[6] Olivier Faugeras. *Three-Dimensional Computer Vision*. MIT Press, 1993.

[7] Olivier Faugeras, Stephane Laveau, Luc Robert, Gabriella Csurka, and Cyril Zeller. 3-d reconstruction of urban scenes from sequences of images. Technical Report 2572, INRIA, June 1995.

[8] W. E. L. Grimson. *From Images to Surface*. MIT Press, 1981.

[9] D. Jones and J. Malik. Computational framework for determining stereo correspondence from a set of linear spatial filters. *Image and Vision Computing*, 10(10):699–708, December 1992.

[10] E. Kruppa. Zur ermittlung eines objectes aus zwei perspektiven mit innerer orientierung. *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa.*, 122:1939–1948, 1913.

[11] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.

[12] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London*, 204:301–328, 1979.

[13] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH '95*, 1995.

[14] Eric N. Mortensen and William A. Barrett. Intelligent scissors for image composition. In *SIGGRAPH '95*, 1995.

[15] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. A stereo correspondence algorithm using a disparity gradient limit. *Perception*, 14:449–470, 1985.

[16] R. Szeliski. Image mosaicing for tele-reality applications. In *IEEE Computer Graphics and Applications*, 1996.

[17] Camillo J. Taylor and David J. Kriegman. Structure and motion from line segments in multiple images. *IEEE Trans. Pattern Anal. Machine Intell.*, 17(11), November 1995.

[18] S. J. Teller, Celeste Fowler, Thomas Funkhouser, and Pat Hanrahan. Partitioning and ordering large radiosity computations. In *SIGGRAPH '94*, pages 443–450, 1994.

[19] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[20] Roger Tsai. A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, August 1987.

[21] S. Ullman. *The Interpretation of Visual Motion*. The MIT Press, Cambridge, MA, 1979.

[22] L Williams. Casting curved shadows on curved surfaces. In *SIGGRAPH '78*, pages 270–274, 1978.

[23] Lance Williams and Eric Chen. View interpolation for image synthesis. In *SIGGRAPH '93*, 1993.

[24] Mourad Zerroug and Ramakant Nevatia. Segmentation and recovery of shgcs from a real intensity image. In *European Conference on Computer Vision*, pages 319–330, 1994.

Recovering Arches in Facade using Ray - Plane intersections in 3-D

G. D. Borshukov and P. Debevec
Department of Electrical Engineering and Computer Science,
University of California, Berkeley, CA 94720

1 Assumptions

1. Focal length f and image plane center (u_0, v_0) in pixels are known.
2. Camera locations $\mathbf{R^C}, \mathbf{T^C}$ in the world coordinate system are previously reconstructed by the minimization algorithm.
3. The arch is initially created as a box which parent and relation are specified. Then, its widht, depth, and height are reconstructed by the minimization algorithm.
4. We know image points like (x_i, y_i) that lie on the arch contour in the image plane.

2 Derivation

First the image measurement (x_i, y_i) is converted into camera coordinates $\mathbf{p}_i = [\ x \quad y \quad -1\ ]^T$ where

$$\begin{aligned} (x &= x_i - u_0)\frac{1}{f} \\ (y &= y_i - v_0)\frac{1}{f} \end{aligned} \tag{1}$$

Now $\mu\mathbf{p}_i$ is a ray from the camera's COP passing through (x_i, y_i). This ray intersects the face of the arch box where the arch begins at point $\mu_0\mathbf{p}_i$.

To find this intersection, i.e. the value of μ_0, we use a point $\mathbf{P}_c$ on the face (the middle of the bottom edge) with world coordinates $\mathbf{p}_c^{\mathbf{W}}$ and camera coordinates $\mathbf{p}_c^{\mathbf{C}} = \mathbf{R^C}(\mathbf{p}_c^{\mathbf{W}} - \mathbf{T^C})$, and the face normal $\mathbf{n}$. The point $\mu_0\mathbf{p}_i$ lies on the face, therefore, its distance from the face:

$$[\mu_0\mathbf{p}_i - \mathbf{p}_c^{\mathbf{C}}]^T(\mathbf{R^C}\mathbf{n}) = 0 \tag{2}$$

which gives:

$$\mu_0 = \frac{[\mathbf{p}_c^{\mathbf{C}}]^T(\mathbf{R^C}\mathbf{n})}{\mathbf{p}_i^T(\mathbf{R^C}\mathbf{n})} \tag{3}$$

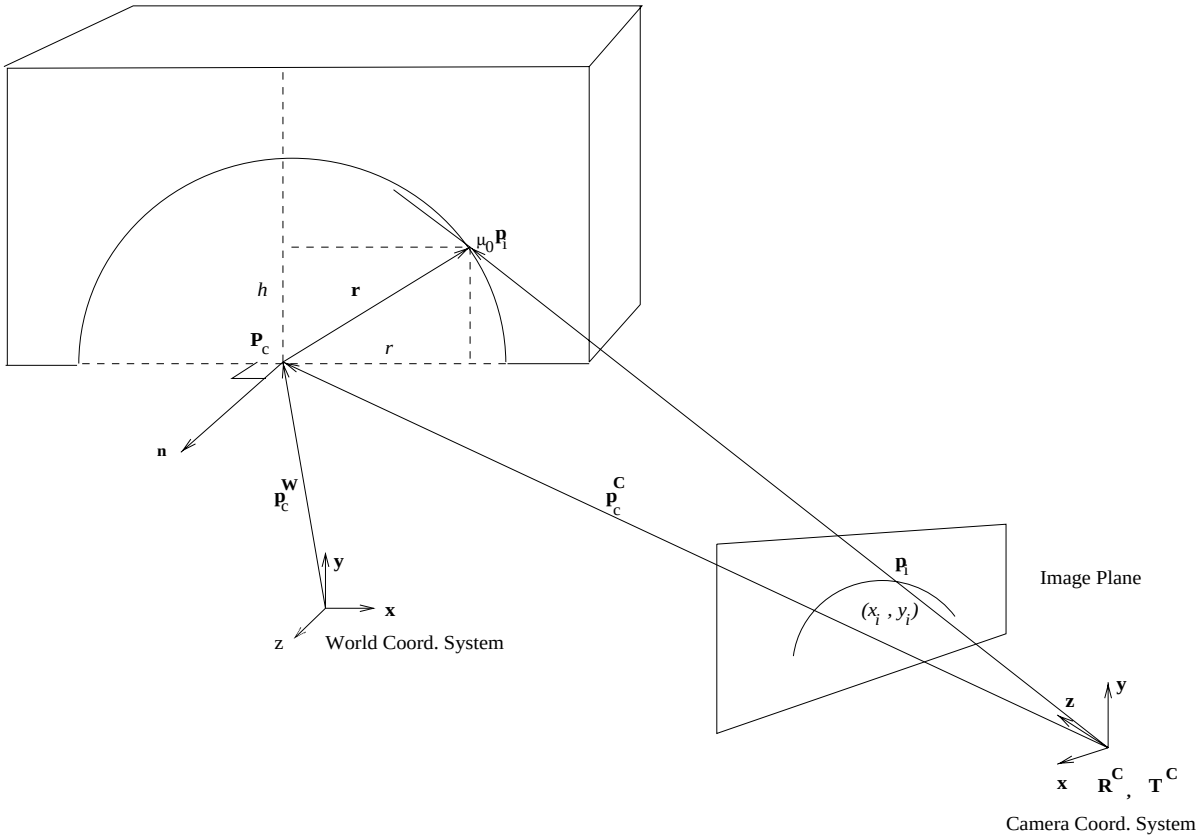


Figure 1: Geometry and notation

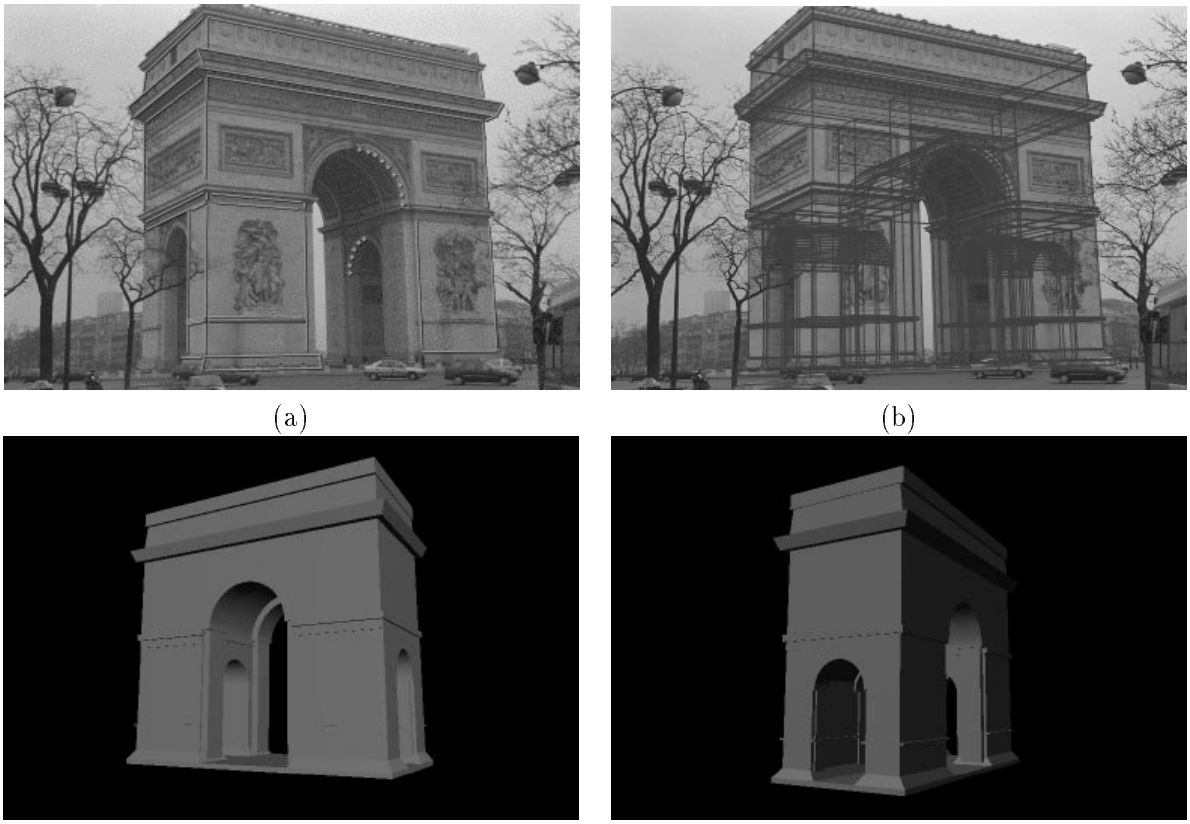
We need to rotate the vector $(\mu_0 \mathbf{p}_i - \mathbf{p}_c^{\mathbf{C}})$ back into world coordinates to obtain the desired vector $\mathbf{r}$:

$$\mathbf{r} = [\mathbf{R}^{\mathbf{C}}]^{-1}(\mu_0 \mathbf{p}_i - \mathbf{p}_c^{\mathbf{C}}) \tag{4}$$

The algorithm uses the projections *r* and *h* of this vector onto the bottom edge and the middle axis of the face to automatically generate the arch surface.

1.2.3 Results

Fig. 1.3 shows the results of reconstructing a 3-D model of the Arc de Triomphe using the new arch recovery tools.



(a) (b)

(c) (d)

Figure 1.3: Model of the Arc de Triomphe demonstrating the new arch recovery capabilities of Façade. **(a)** One of three photographs used to reconstruct the Arc de Triomphe, with marked features indicated. **(b)** Reconstructed model edges projected into the original photograph. **(c)** Recovered model of the Arc de Triomphe. **(d)** Another view of the recovered 3-D model.

Recovering the Radius and Offset of a Cross-Section in SORs using Minimum Distance between Two Rays in 3-D

G. D. Borshukov and P. Debevec
Department of Electrical Engineering and Computer Science,
University of California, Berkeley, CA 94720

1 Assumptions

1. Focal length f and image plane center (u_0, v_0) in pixels are known.
2. Camera locations $\mathbf{R^C}, \mathbf{T^C}$ in the world coordinate system are previously reconstructed by the minimization algorithm. The camera coordinate system is aligned with the world coordinate system. From now on all vectors will be in world coordinates.
3. The SOR central axis is known, i.e. we know a point on the axis $\mathbf{p_b}$ (usually the base point) and the axis direction $\mathbf{m}$ (usually $[\ 0 \quad 1 \quad 0\ ]^T$).
4. We know image points like (x_i, y_i) that lie on the occluding contour in the image plane.

2 Derivation

First the image measurement (x_i, y_i) is converted into camera coordinates $\mathbf{p}_i = [\ x \quad y \quad -1\ ]^T$ where

$$\begin{aligned} (x &= x_i - u_0)\frac{1}{f} \\ (y &= y_i - v_0)\frac{1}{f} \end{aligned} \tag{1}$$

We want find the minimum distance between the rays $\mathbf{p_b} + \lambda\mathbf{m}$ and $\mu\mathbf{p_i}$. Exploiting the fact that the minimum distance vector

$$\mathbf{d_0} = (\mu_0\mathbf{p_i} - \mathbf{p_b} - \lambda_0\mathbf{m}) \tag{2}$$

must be perpendicular to each ray, conveniently, our task boils down to solving the following simultaneous equationswith respect to λ_0 and μ_0.

$$\mu_0\mathbf{p_i}^T\mathbf{d_0} = 0 \tag{3}$$
$$\lambda_0\mathbf{m}^T\mathbf{d_0} = 0 \tag{4}$$

Excluding the trivial solutions $\lambda_0 = 0$ and $\mu_0 = 0$ and substituting (1) into (2) and (3) we get

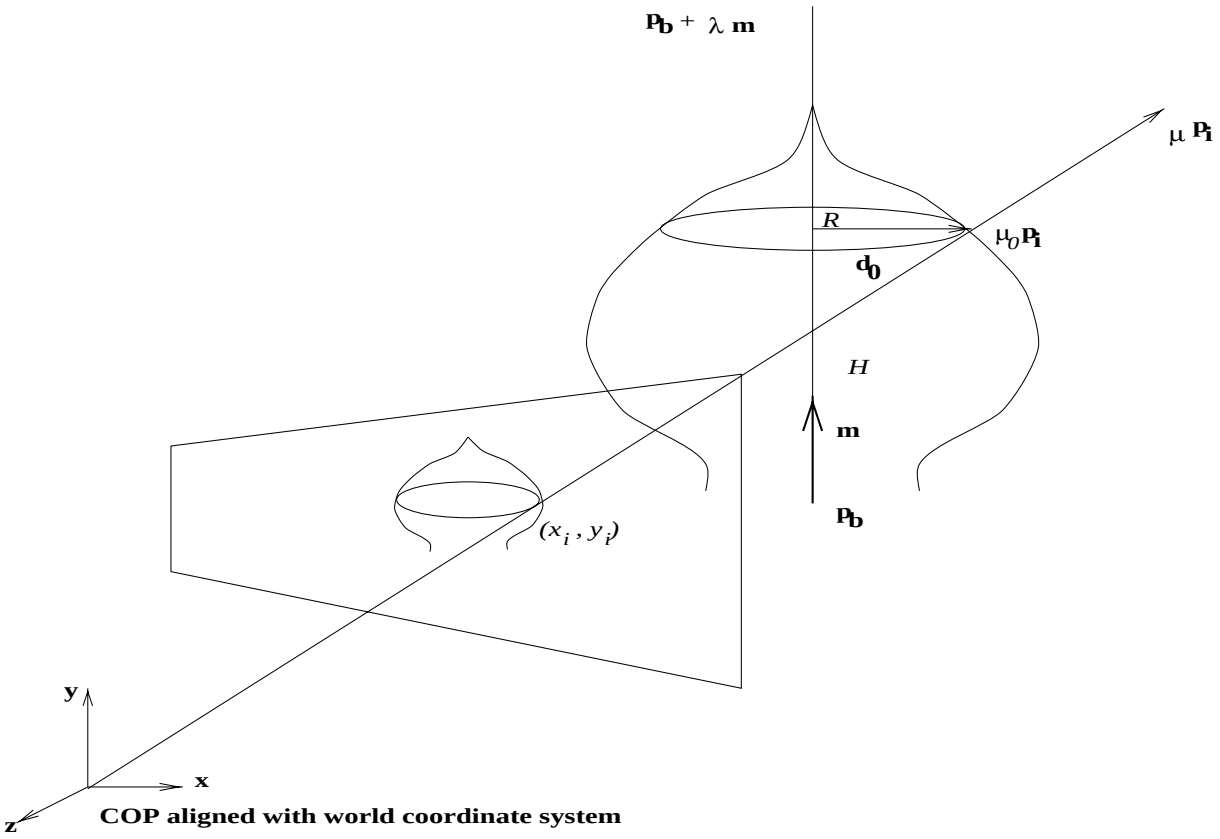


Figure 1: Geometry and notation

$$A\mu_0 - C\lambda_0 = B \quad (5)$$
$$C\mu_0 - E\lambda_0 = D \quad (6)$$

where

$$A = \mathbf{p_i}^T\mathbf{p_i} \quad (7)$$
$$B = \mathbf{p_i}^T\mathbf{p_b} \quad (8)$$
$$C = \mathbf{p_i}^T\mathbf{m} \quad (9)$$
$$D = \mathbf{m}^T\mathbf{p_b} \quad (10)$$
$$E = \mathbf{m}^T\mathbf{m} \quad (11)$$

Further using (4) we obtain

$$\mu_0 = \frac{B + C\lambda_0}{A} \quad (12)$$

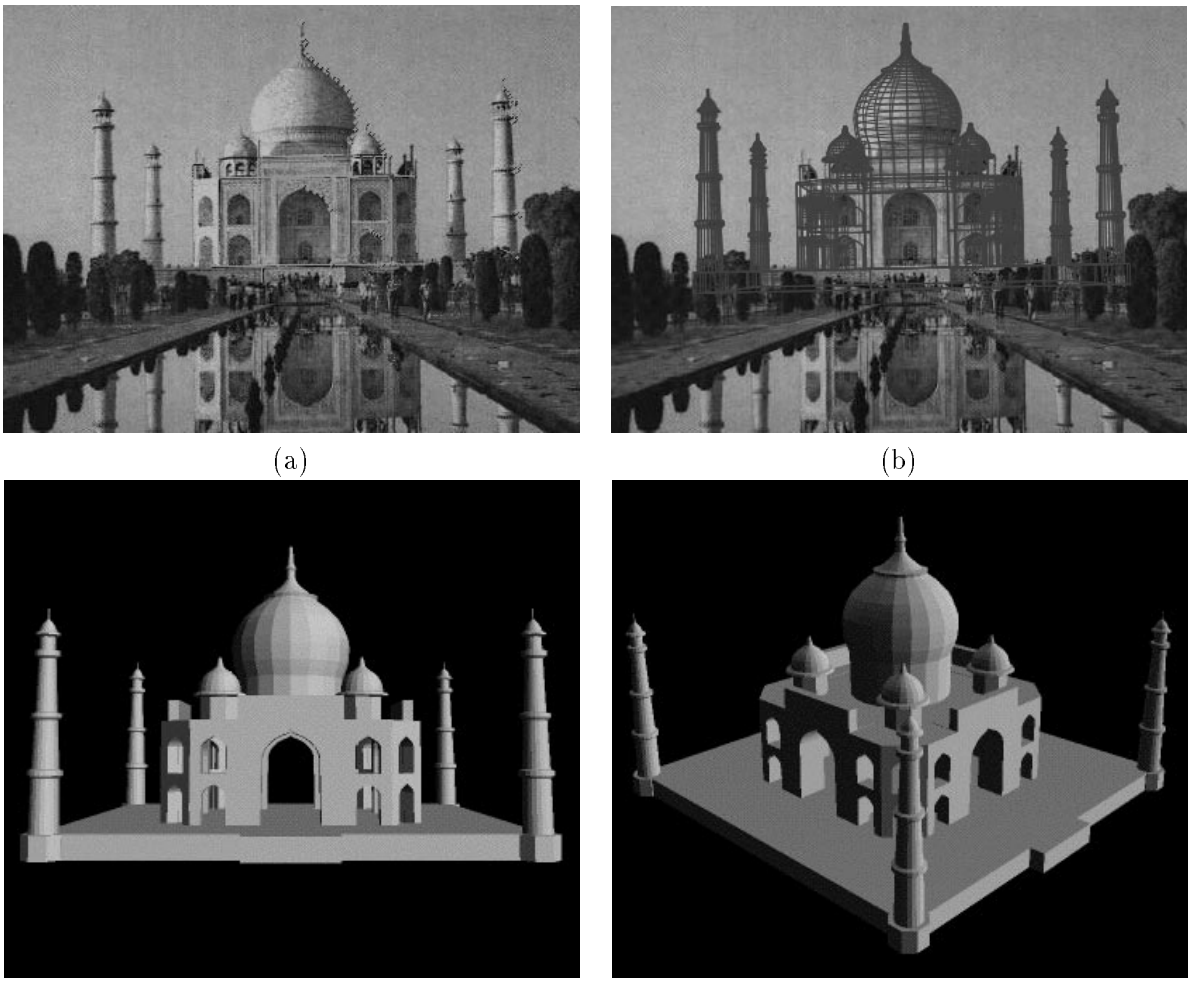
which could be substituted in (5) to yield

$$\lambda_0 = \frac{BC - AD}{AE - C^2} \quad (13)$$

Now knowing λ_0 and μ_0, the radius R of a circular cross section offset by $H = \lambda_0 \mathbf{m}$ from $\mathbf{p_b}$(usually on the base plane) can be expressed by

$$R = \sqrt{\mathbf{d_0}^T \mathbf{d_0}} \tag{14}$$

The algorithm uses the quantities H and R to automatically generate the surface of revolution.



(a) (b)

(c) (d)

Figure 1.5: Model of the majestic Taj Mahal created with the new surface of revolution and arch reconstruction tools. **(a)** A single low-resolution photograph of the Taj Mahal obtained from the Internet, with marked features shown. **(b)** Reconstructed model edges projected onto the original photograph. **(c)** 3-D model of the Taj Mahal, complete with domes and minarets, recovered from the single photograph in less than an hour of modeling time. **(d)** Another view of the recovered 3-D model.

SIGGRAPH 2000 Course on
3D Photography

Overview of Active Vision Techniques

Brian Curless
University of Washington

Overview

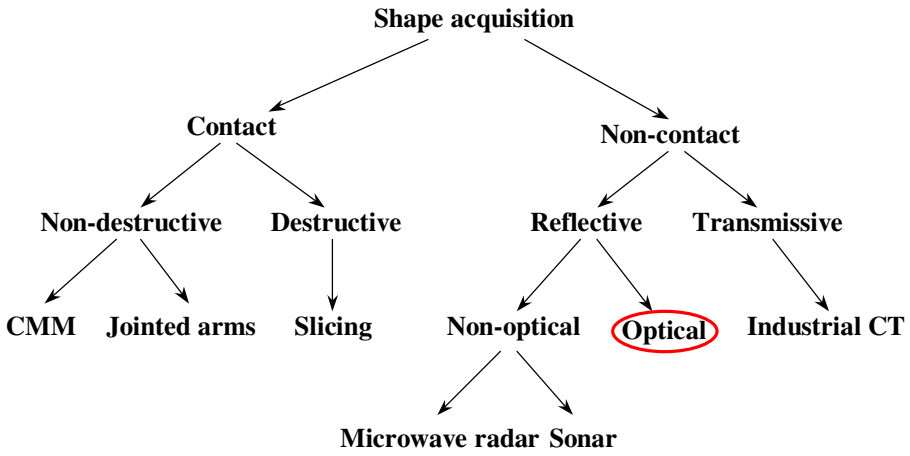
Introduction

Active vision techniques

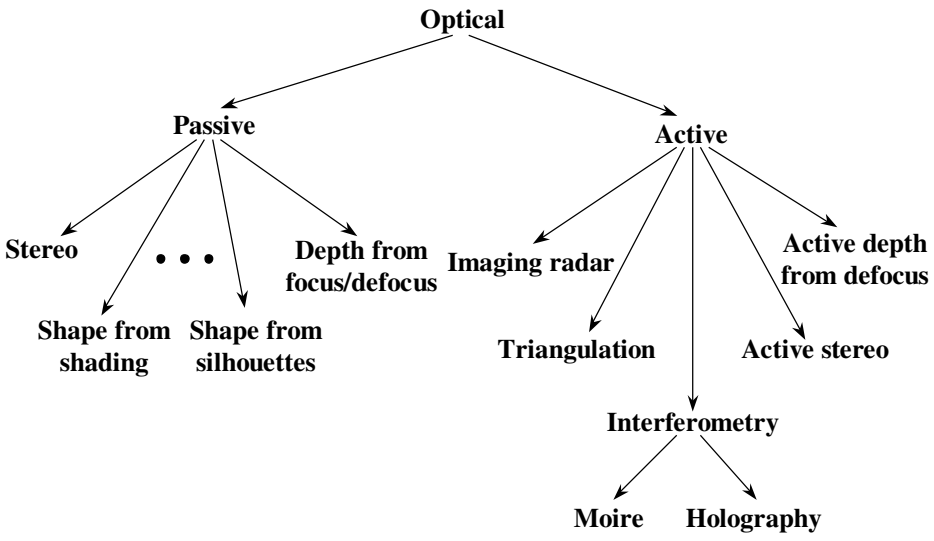
- **Imaging radar**
- **Triangulation**
- **Moire**
- **Active Stereo**
- **Active depth-from-defocus**

Capturing appearance

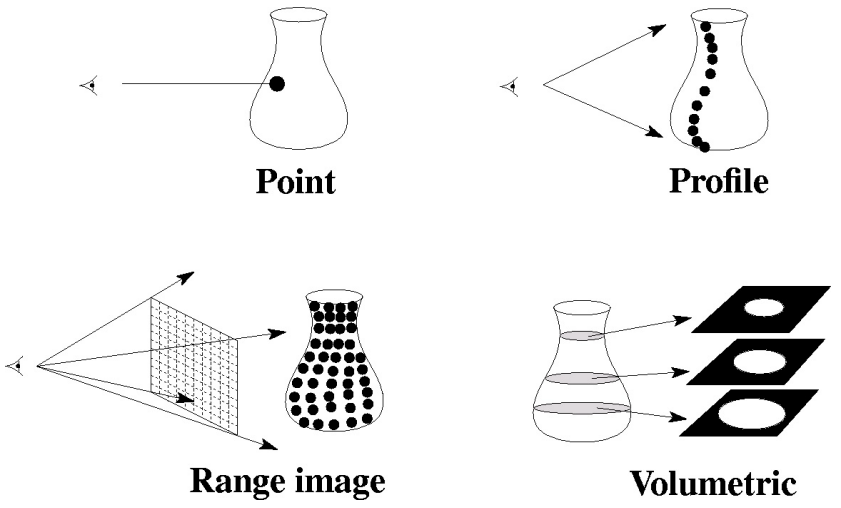
A taxonomy



A taxonomy



Structure of the data



Quality measures

Resolution

Smallest change in depth that sensor can report?
Quantization? Spacing of samples?

Accuracy

Statistical variations among repeated measurements of known value.

Repeatability

Do the measurements drift?

Environmental sensitivity

Does temperature or wind speed influence measurements?

Speed

Optical range acquisition

Strengths

- ***Non-contact***
- ***Safe***
- ***Inexpensive (?)***
- ***Fast***

Limitations

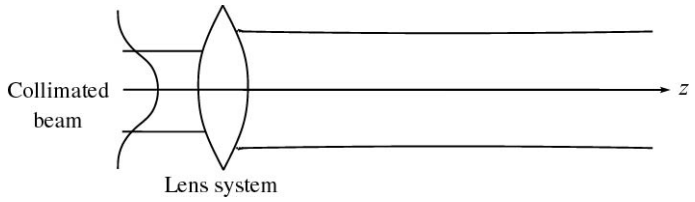
- ***Can only acquire visible portions of the surface***
- ***Sensitivity to surface properties***
  - > *transparency, shininess, rapid color variations, darkness (no reflected light), subsurface scatter*
- ***Confused by interreflections***

Illumination

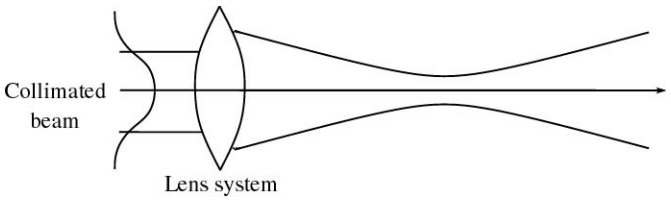
Why are lasers a good idea?

- ***Compact***
- ***Low power***
- ***Single wavelength is easy to isolate***
- ***No chromatic aberration***
- ***Tight focus over long distances***

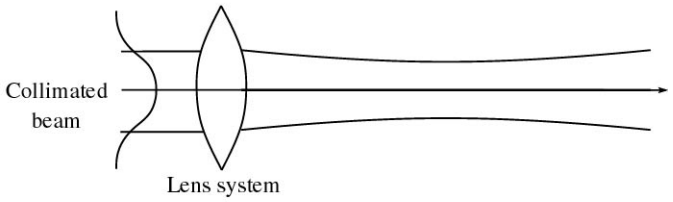
Illumination



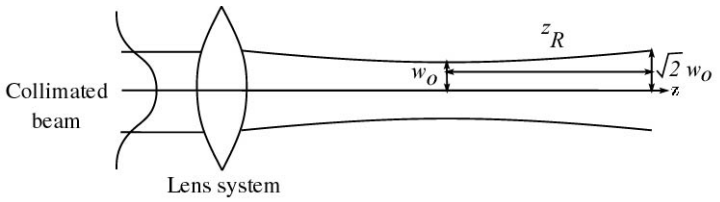
Illumination



Illumination



Illumination



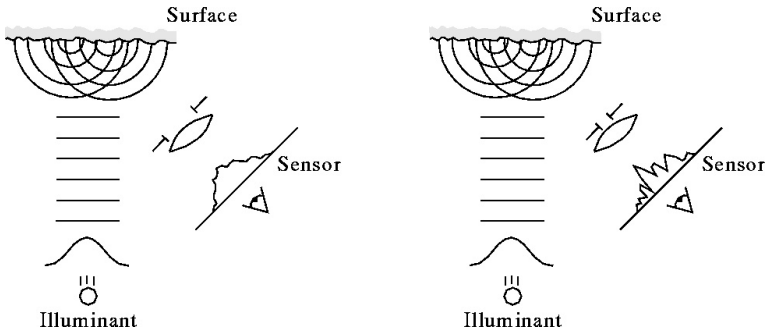
$$z_R = \frac{\pi w_o^2}{\lambda}$$

z_R = Rayleigh range
w_o = beam waist (narrowest laser width)
λ = wavelength of laser

Illumination

Limitations of lasers

- ***Eye safety concerns***
- ***Laser speckle adds noise***
  > *Narrowing the aperture increases the noise*

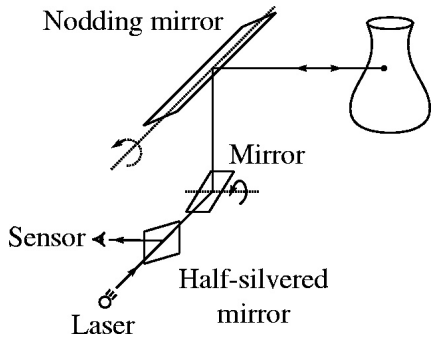


Imaging radar: time of flight

A pulse of light is emitted, and the time of the reflected pulse is recorded:

c t = 2 r = roundtrip distance

Typical scanning configuration:



Imaging radar: Amplitude Modulation

The current to a laser diode is driven at frequency:

$$f_{AM} = \frac{c}{\lambda_{AM}}$$

The phase difference between incoming and outgoing signals gives the range:

$$2r(\Delta\varphi) = n\lambda_{AM} + \frac{\Delta\varphi}{2\pi}\lambda_{AM} \Rightarrow r(\Delta\varphi) = \frac{1}{2}\lambda_{AM}\frac{(\Delta\varphi + 2\pi n)}{2\pi}$$

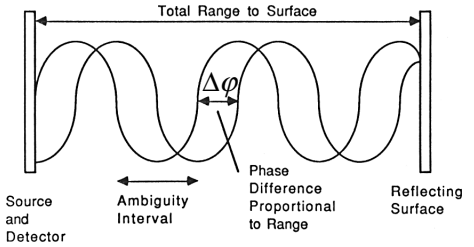


Figure from [Besl89]

Imaging radar: Amplitude Modulation

Note the ambiguity due to the + 2π*n*. This translates into range ambiguity:

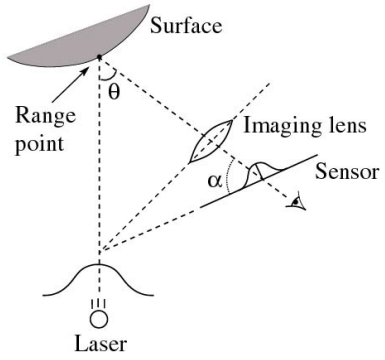
$$r_{ambig} = \frac{n\lambda_{AM}}{2}$$

The ambiguity can be overcome with sweeps of increasingly finer wavelengths.

Optical triangulation

A beam of light strikes the surface, and some of the light bounces toward an off-axis sensor.

The center of the imaged reflection is triangulated against the laser line of sight.



Optical triangulation

Lenses map planes to planes. If the object plane is tilted, then so should the image plane.

The image plane tilt is described by the Scheimpflug condition:

$$\tan\alpha = \frac{\tan\theta}{M}$$

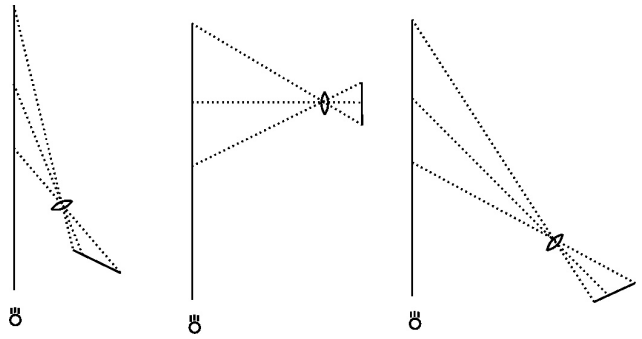
where M is the on-axis magnification.

Triangulation angle

When designing an optical triangulation, we want:

- **Small triangulation angle**
- **Uniform resolution**

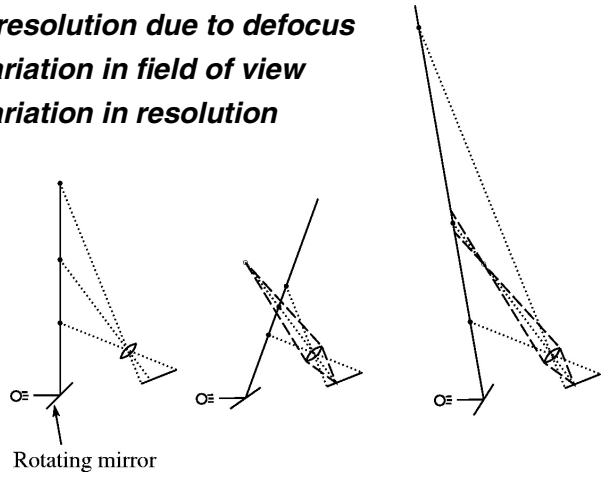
These requirements are at odds with each other.



Triangulation scanning configurations

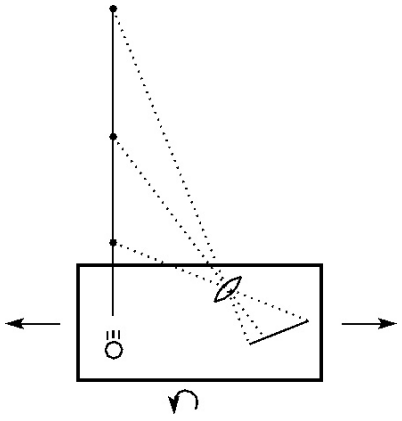
A scene can be scanned by sweeping the illuminant. Problems:

- ***Loss of resolution due to defocus***
- ***Large variation in field of view***
- ***Large variation in resolution***



Triangulation scanning configurations

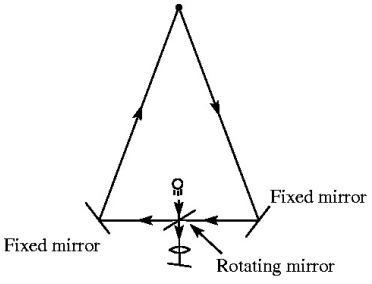
Can instead move the laser and camera together, e.g., by translating or rotating a scanning unit.

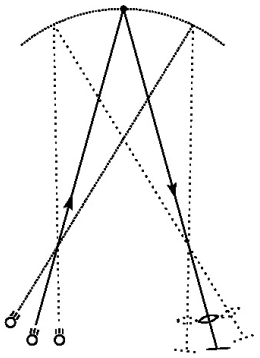


Triangulation scanning configurations

A novel design was created and patented at the NRC of Canada [Rioux'87].

Basic idea: sweep the laser and sensor *simultaneously*.



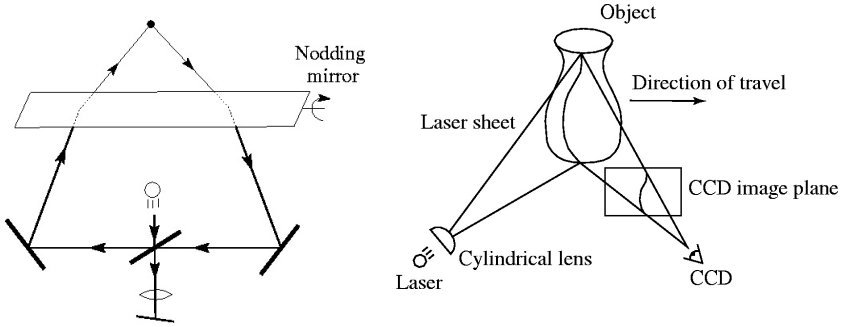


After unfolding the optics

Triangulation scanning configurations

Extension to 3D achievable as:

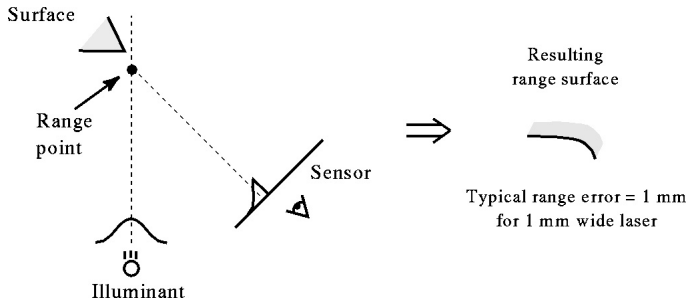
- ***flying spot***
- ***sweeping light stripe***
- ***hand-held light stripe on jointed arm***



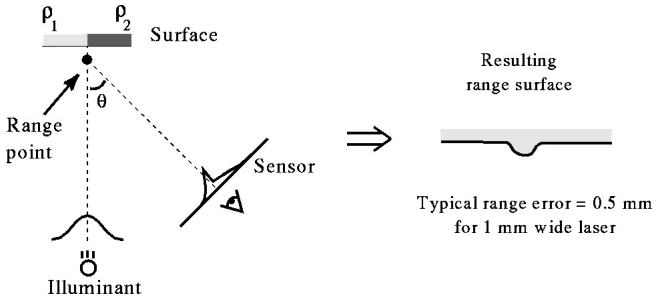
Errors in optical triangulation

Finding the center of the imaged pulse is tricky.

If the surface exhibits variations in reflectance or shape, then laser width limits accuracy.

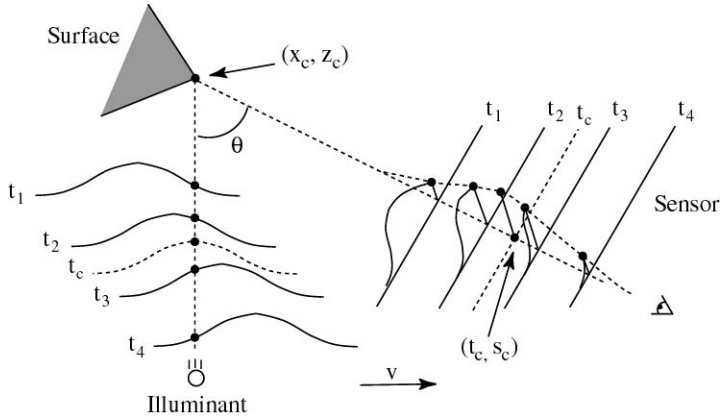


Errors in optical triangulation

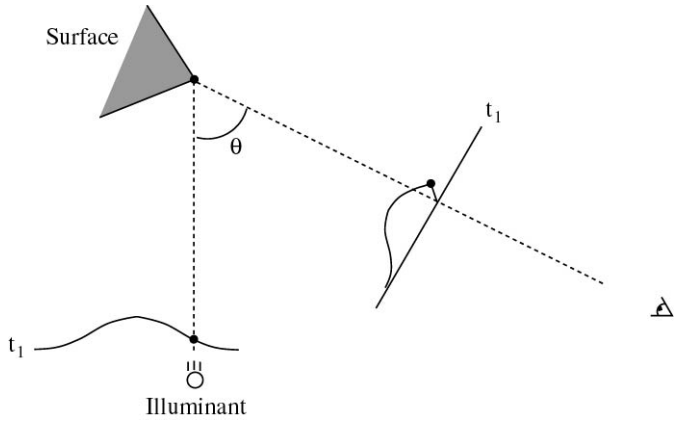


Spacetime analysis

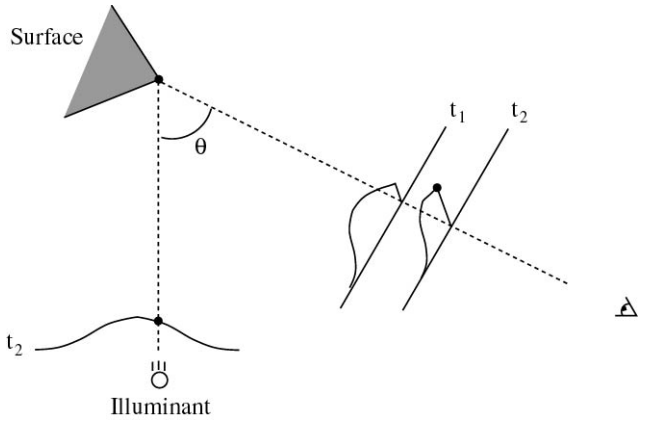
A solution to this problem is spacetime analysis [Curless 95]:



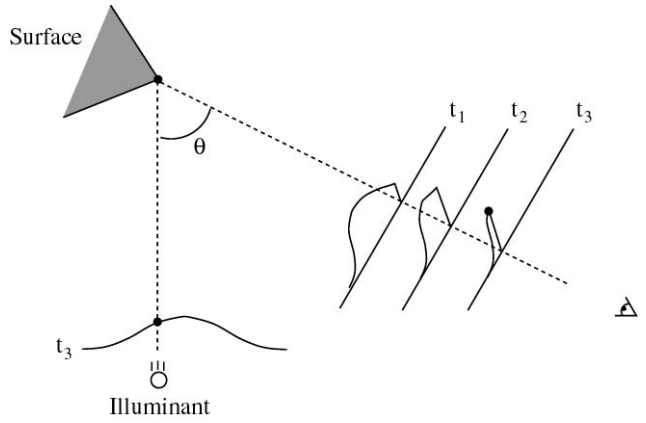
Spacetime analysis



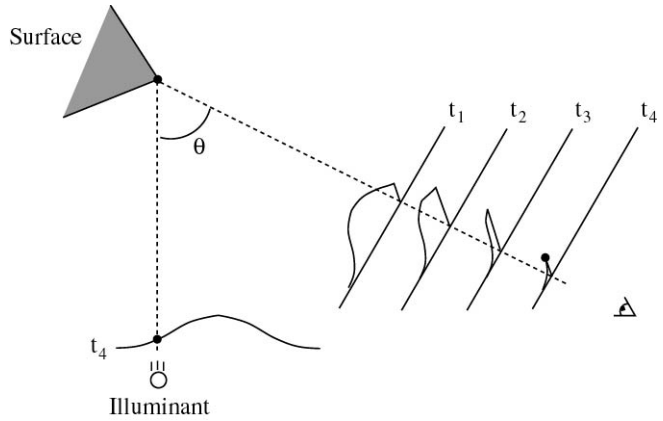
Spacetime analysis



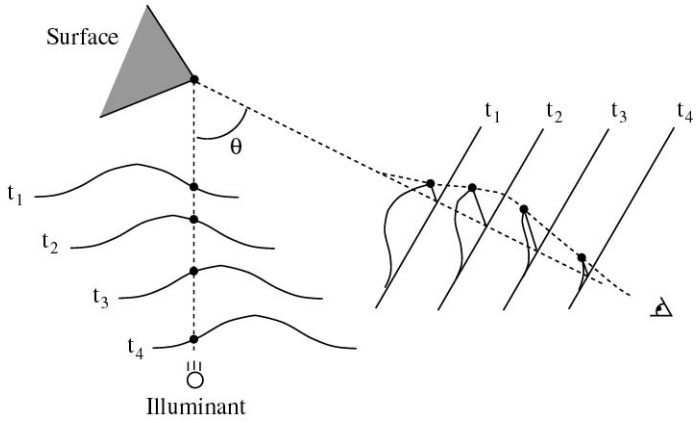
Spacetime analysis



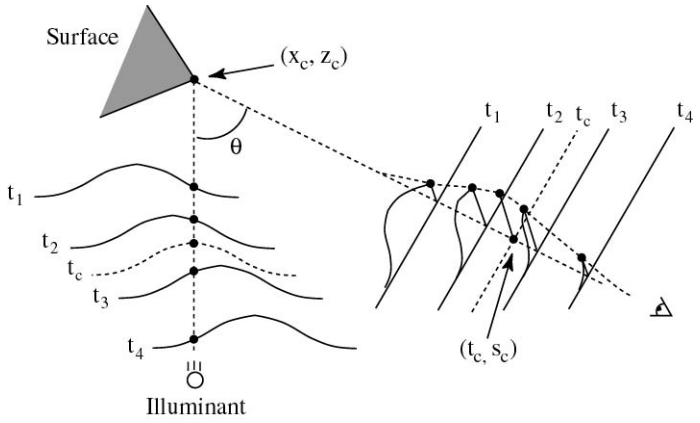
Spacetime analysis



Spacetime analysis



Spacetime analysis

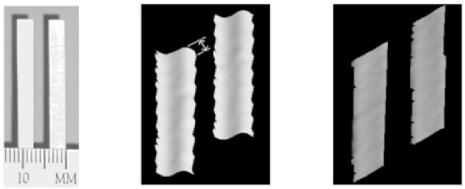


Spacetime analysis: results

Reflectance correction

Reflectance card Traditional analysis Spacetime analysis

Edge curl reduction



Two thin strips Traditional analysis Spacetime analysis

Improved shape extraction

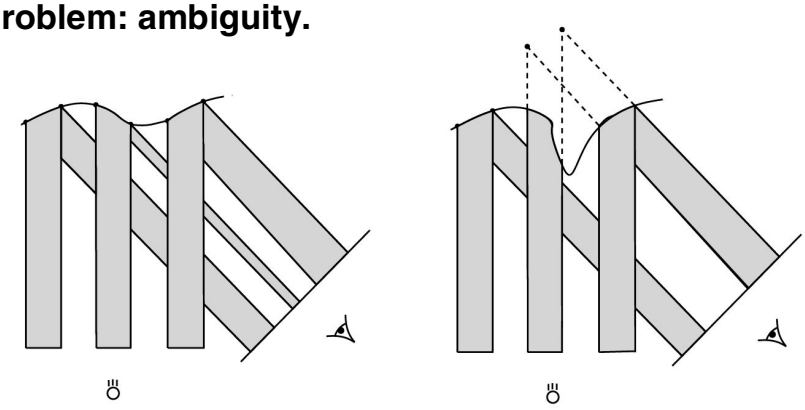
Shape ribbon Traditional analysis Spacetime analysis

Multi-spot and multi-stripe triangulation

For faster acquisition, some scanners use multiple spots or stripes.

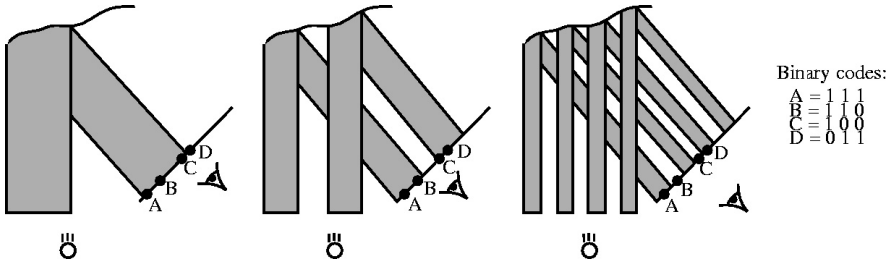
Trade off depth-of-field for speed.

Problem: ambiguity.



Binary coded illumination

Alternative: resolve visibility hierarchically (logN).



Moire

Moire methods extract shape from interference patterns:

- ***Illuminate a surface with a periodic grating.***
- ***Capture image as seen at an angle through another grating.***

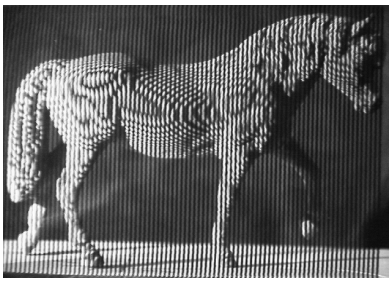
  => interference pattern, phase encodes shape
- ***Low pass filter the image to extract the phase signal.***

Requires that the shape varies slowly so that phase is low frequency, much lower than grating frequency.

Example: shadow moire

Shadow moire:

- ***Place a grating (e.g., stripes on a transparency) near the surface.***
- ***Illuminate with a lamp.***
- ***Instant moire!***



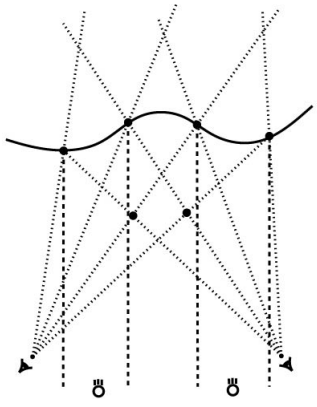
Shadow moire

Filtered image

Active stereo

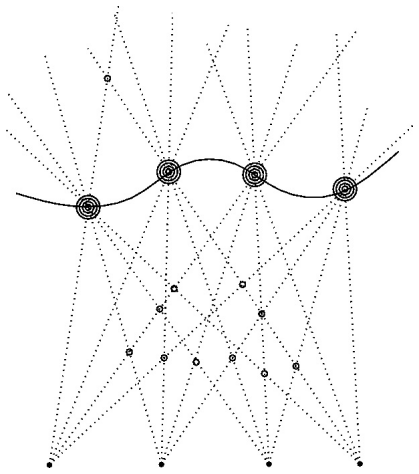
Passive stereo methods match features observed by two cameras and triangulate.

Active stereo simplifies feature finding with structured light. Problem: ambiguity.



Active multi-baseline stereo

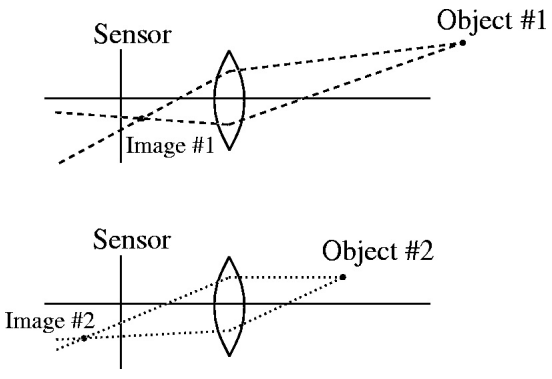
Using multiple cameras reduces likelihood of false matches.



Active depth from defocus

Depth of field for large apertures will cause the image of a point to blur.

The amount of blur indicates distance to the point.



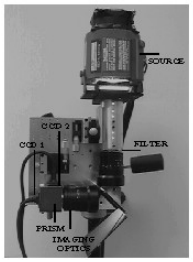
Active depth from defocus

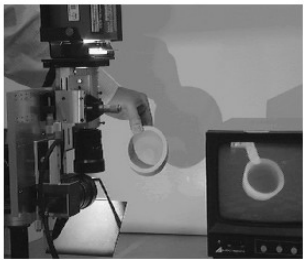
Depth ambiguity can be resolved with two sensor planes.

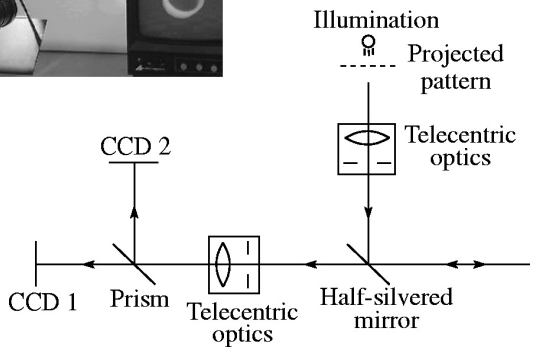
Amount of defocus depends on presence of texture. Solution: project structured lighting onto surface.

[Nayar'95] demonstrates a real-time system utilizing telecentric optics.

Active depth from defocus







Capturing appearance

"Appearance" refers to the way an object reflects light to a viewer.

We can think of appearance under:

- ***fixed lighting***
- ***variable lighting***

Appearance under fixed lighting

Under fixed lighting, a static radiance field forms.

Each point on the object reflects a 2D (directional) radiance function.

We can acquire samples of these radiance functions with photographs registered to the geometry.

Appearance under variable lighting

To re-render the surface under novel lighting, we must capture the BRDF -- the bi-directional reflectance distribution function.

In the general case, this problem is *hard*:

- ***The BRDF is a 4D function -- may need many samples.***
- ***Interreflections imply the need to perform difficult inverse rendering calculations.***

Here, we mention ways of capturing the data needed to estimate the BRDF.

BRDF capture

To capture the BRDF, we must acquire images of the surface under known lighting conditions.

[Sato'97] captures color images with point source illumination. The camera and light are calibrated, and pose is determined by a robot arm.

[Baribeau'92] uses a white laser that is also used for optical triangulation. Reflectance samples are registered to range samples.

Key advantage: minimizes interreflection.

Bibliography

Baribeau, R., Rioux, M., and Godin, G., "Color reflectance modeling using a polychromatic laser range scanner," IEEE Transactions on PAMI, vol. 14, no. 2, Feb., 1992, pp. 263-269.

Besl, P. *Advances in Machine Vision*. "Chapter 1: Active optical range imaging sensors," pp. 1-63, Springer-Verlag, 1989.

Curless, B. and Levoy, M., "Better optical triangulation through spacetime analysis." In Proceedings of IEEE International Conference on Computer Vision, Cambridge, MA, USA, 20-23 June 1995, pp. 987-994.

Nayar, S.K., Watanabe, M., and Noguchi, M. "Real-time focus range sensor", Fifth International Conference on Computer Vision (1995), pp. 995-1001.

Rioux, M., Bechthold, G., Taylor, D., and Duggan, M. "Design of a large depth of view three-dimensional camera for robot vision," Optical Engineering (1987), vol. 26, no. 12, pp. 1245-1250.

Sato, Y., Wheeler, M.D., Ikeuchi, K., "Object shape and reflectance modeling from observation." SIGGRAPH '97, p.379-387.

Reprinted, with permission, from IEEE Int't Conference on Computer Vision '95, pp. 987-994, June, 1995.

Better Optical Triangulation through Spacetime Analysis

Brian Curless

curless@cs.stanford.edu
Computer Systems Laboratory
Stanford University
Stanford, CA 94305

Marc Levoy

levoy@cs.stanford.edu
Computer Science Department
Stanford University
Stanford, CA 94305

Abstract

The standard methods for extracting range data from optical triangulation scanners are accurate only for planar objects of uniform reflectance illuminated by an incoherent source. Using these methods, curved surfaces, discontinuous surfaces, and surfaces of varying reflectance cause systematic distortions of the range data. Coherent light sources such as lasers introduce speckle artifacts that further degrade the data. We present a new ranging method based on analyzing the time evolution of the structured light reflections. Using our spacetime analysis, we can correct for each of these artifacts, thereby attaining significantly higher accuracy using existing technology. We present results that demonstrate the validity of our method using a commercial laser stripe triangulation scanner.

1 Introduction

Active optical triangulation is one of the most common methods for acquiring range data. Although this technology has been in use for over twenty years, its speed and accuracy has increased dramatically in recent years with the development of geometrically stable imaging sensors such as CCD's and lateral effect photodiodes. The range acquisition literature contains many descriptions of optical triangulation range scanners, of which we list a handful [2] [8] [10] [12] [14] [17] . The variety of methods differ primarily in the structure of the illuminant (typically point, stripe, multi-point, or multi-stripe), the dimensionality of the sensor (linear array or CCD grid), and the scanning method (move the object or move the scanner hardware).

Figure 1 shows a typical system configuration in two dimensions. The location of the center of the reflected light pulse imaged on the sensor corresponds to a line of sight that intersects the illuminant in exactly one point, yielding a depth value. The shape of the object is acquired by translating or rotating the object through the beam or by scanning the beam across the object.

The accuracy of optical triangulation methods hinges on the ability to locate the "center" of the imaged pulse at each time step. For optical triangulation systems that extract range from single imaged pulses at a time, variations in surface reflectance and shape result in systematic range errors. Several researchers have observed one or both of these accuracy limitations [4] [12] [16]. For the case of coherent illumination, the images of reflections from rough surfaces are also subject to laser speckle noise, introducing noise into the range data. Researchers have studied the effect of speckle on range determination and have indicated that it is a fundamental limit to the accuracy of laser range triangulation, though its effects can be reduced with well-known speckle reduction techniques [1] [5]. Mundy and Porter [12] attempt to correct for variations in surface reflectance by noting that two imaged pulses, differing in position or wavelength are sufficient to overcome the reflectance errors, though some restrictive assumptions are necessary for the case of differing wavelengths. Kanade, et al, [11] describe a rangefinder that finds peaks in time for a stationary sensor with pixels that view fixed points on an object. This method of peak detection is very similar to the one presented in this paper for solving some of the problems of optical triangulation; however, the authors in [11] do not indicate that their design solves or even addresses these problems. Further, we show that the principle generalizes to other scanning geometries.

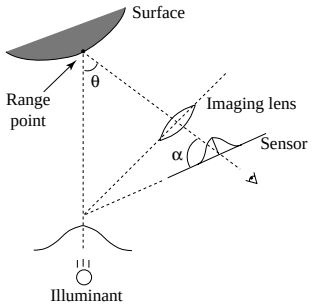


Figure 1: Optical triangulation geometry. The angle θ is the triangulation angle while α is the tilt of the sensor plane needed to keep the laser plane in focus.

In the following sections, we first show how range errors arise with traditional triangulation techniques. In section 3, we show that by analyzing the time evolution of structured light reflections, a process we call spacetime analysis, we can overcome the accuracy limitations caused by shape and reflectance variations. Experimental evidence also indicates that laser speckle behaves in a manner that allows us to reduce its distorting effect as well.

In sections 4 and 5, we describe our hardware and software implementation of the spacetime analysis using a commercial scanner and a video digitizer, and we demonstrate a significant improvement in range accuracy. Finally, in section 6, we conclude and describe future directions.

2 Error in triangulation systems

For optical triangulation systems, the accuracy of the range data depends on proper interpretation of imaged light reflections. The most common approach is to reduce the problem to one of finding the "center" of a one dimensional pulse, where the "center" refers to the position on the sensor which hopefully maps to the center of the illuminant. Typically, researchers have opted for a statistic such

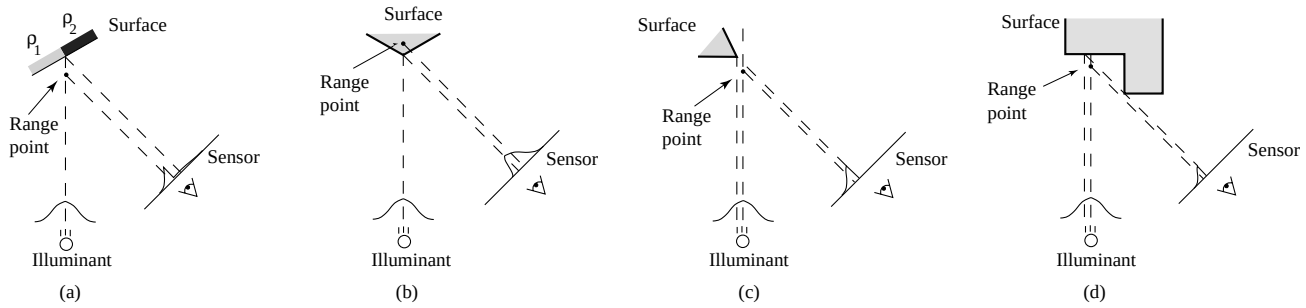


Figure 2: Range errors using traditional triangulation methods. (a) Reflectance discontinuity. (b) Corner. (c) Shape discontinuity with respect to the illumination. (d) Sensor occlusion.

as mean, median or peak of the imaged light as representative of the center. These statistics give the correct answer when the surface is perfectly planar, but they are generally inaccurate whenever the surface perturbs the shape of the illuminant.

2.1 Geometric intuition

Perturbations of the shape of the imaged illuminant occur whenever:

- The surface reflectance varies.
- The surface geometry deviates from planarity.
- The light paths to the sensor are partially occluded.
- The surface is sufficiently rough to cause laser speckle.

In Figure 2, we give examples of how the first three circumstances result in range errors even for an ideal triangulation system with infinite sensor resolution and perfect calibration. For purposes of illustration, we omit the imaging optics of Figure 1 and treat the sensor as a one dimensional orthographic sensor. Further, we assume an illuminant of Gaussian cross-section, and we use the mean for determining the center of an imaged pulse. Figure 2a shows how a step reflectance discontinuity results in range points that do not lie on the surface. Figure 2b and 2c provide two examples of shape variations resulting in range errors. Note that in Figure 2c, the center of the illuminant is not even striking a surface. In this case, a measure of the center of the pulse results in a range value, when in fact the correct answer is to return no range value whatever. Finally, Figure 2d shows the effect of occluding the line of sight between the illuminated surface and the sensor. This range error is very similar to the error encountered in Figure 2c.

The fourth source of range error is laser speckle, which arises when coherent laser illumination bounces off of a surface that is rough compared to a wavelength [7]. The surface roughness introduces random variations in optical path lengths, causing a random interference pattern throughout space and at the sensor. The result is an imaged pulse with a noise component that affects the mean pulse detection, causing range errors even from a planar target.

2.2 Quantifying the error

To quantify the errors inherent in using mean pulse analysis, we have computed the errors introduced by reflectance and shape variations for an ideal triangulation system with a single Gaussian illuminant. We take the beam width, w, to be the distance between the beam center and the e^{-2} point of the irradiance profile, a convention common to the optics literature. We present the range errors in a scale invariant form by dividing all distances by the beam width. Figure 3 illustrates the maximum deviation from planarity introduced by scanning reflectance discontinuities of varying step magnitudes for varying triangulation angles. As the size of the step increases, the error increases correspondingly. In addition, smaller triangulation angles, which are desirable for reducing the likelihood of missing data due to sensor occlusions, actually result in larger range errors. This result is not surprising, as sensor mean positions are converted to depths through a division by $sin\theta$, where θ is the triangulation angle, so that errors in mean detection translate to larger range errors for smaller triangulation angles.

Figure 4 shows the effects of a corner on range error, where the error is taken to be the shortest distance between the computed range data and the exact corner point. The corner is oriented so that the illumination direction bisects the corner's angle as shown in Figure 2b. As we might expect, a sharper corner results in greater compression of the left side of the imaged Gaussian relative to the right side, pushing the mean further to the right on the sensor and pushing the triangulated point further behind the corner. In this case, the triangulation angle has little effect as the division by $sin\theta$ is offset almost exactly by the smaller observed left/right pulse compression imbalance.

One possible strategy for reducing these errors would be to decrease the width of the beam and increase the resolution of the sensor. However, diffraction limits prevent us from focusing the beam to an arbitrary width. The limits on focusing a Gaussian beam with spherical lenses are well known [15]. In recent years, Bickel, et al, [3] have explored the use of axicons (e.g., glass cones and other surfaces of revolution) to attain tighter focus of a Gaussian beam. The refracted beam, however, has a zeroth order Bessel function cross-section; i.e., it has numerous side-lobes of non-negligible irradiance. The influence of these side-lobes is not well-documented and would seem to complicate triangulation.

3 A New Method: Spacetime Analysis

The previous section clearly demonstrates that analyzing each imaged pulse using a low order statistic leads to systematic range errors. We have found that these errors can be reduced or eliminated by analyzing the time evolution of the pulses.

3.1 Geometric intuition

Figure 5 illustrates the principle of spacetime analysis for a laser triangulation scanner with Gaussian illuminant and orthographic sen-

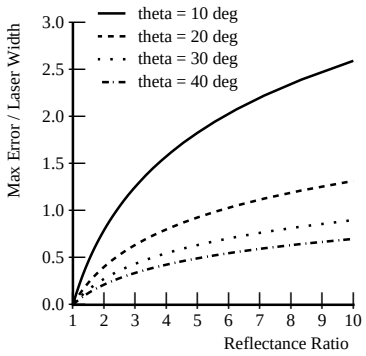


Figure 3: Plot of errors due to reflectance discontinuities for varying triangulation angles (theta).

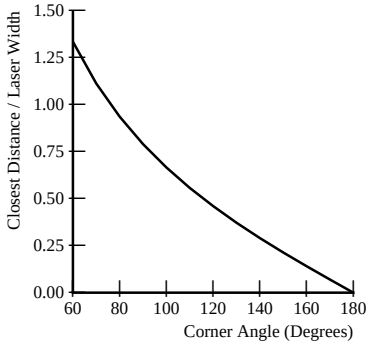


Figure 4: Plot of errors due to corners.

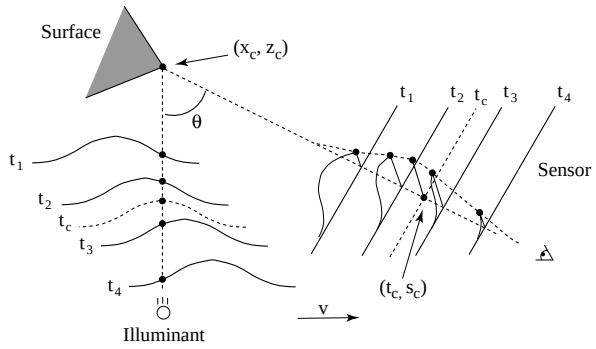


Figure 5: Spacetime mapping of a Gaussian illuminant. As the light sweeps across the corner point, the sensor images the shape of the illuminant over time.

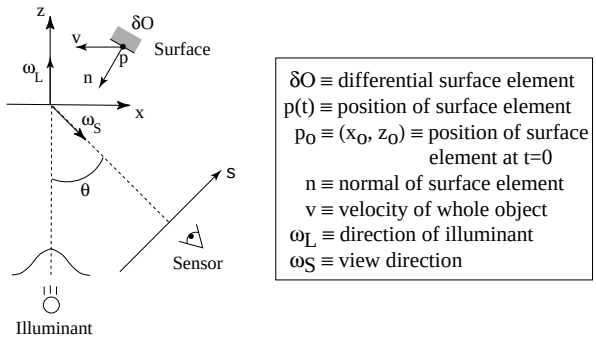


Figure 6: Triangulation scanner coordinate system. A depiction of the coordinate systems and the vectors relevant to a moving differential element.

sor as it translates across the edge of an object. As the scanner steps to the right, the sensor images a smaller and smaller portion of the laser cross-section. By time t_3, the sensor no longer images the center of the illuminant, and conventional methods of range estimation fail. However, if we look along the lines of sight from the corner to the laser and from the corner to the sensor, we see that the profile of the laser is being imaged *over time* onto the sensor (indicated by the dotted Gaussian envelope). Thus, we can find the coordinates of the corner point (x_c, z_c) by searching for the mean of a Gaussian along a constant line of sight through the sensor images. We can express the coordinates of this mean as a time and a position on the sensor, where the time is in general between sensor frames and the position is between sensor pixels. The position on the sensor indicates a depth, and the time indicates the lateral position of the center of the illuminant. In the example of Figure 5, we find that the spacetime Gaussian corresponding to the exact corner has its mean at position s_c on the sensor at a time t_c between t_2 and t_3 during the scan. We extract the corner's depth by triangulating the center of the illuminant with the line of sight corresponding to the sensor coordinate s_c, while the corner's horizontal position is proportional to the time t_c.

3.2 A complete derivation

For a more rigorous analysis, we consider the time evolution of the irradiance from a translating differential surface element, δO, as recorded at the sensor. We refer the reader to Figure 6 for a description of coordinate systems; note that in contrast to the previous section, the surface element is translating instead of the illuminant-sensor assembly. The element has a normal $\hat{n}$ and an initial position $\vec{p}_o$ and is translating with velocity $\vec{v}$, so that:

$$\vec{p}(t) = \vec{p}_o + t\vec{v} \tag{1}$$

Our objective is to compute the coordinates $\vec{p}_o = (x_o, z_o)$ given the temporal irradiance variations on the sensor. For simplicity, we assume that $\vec{v} = (-v, 0)$. The illuminant we consider is a laser with a unidirectional Gaussian radiance profile. We can describe the total radiance reflected from the element to the sensor as:

$$L(\vec{p}(t), \hat{\omega}_S) = f_r(\hat{\omega}_L, \hat{\omega}_S)|\hat{n} \cdot \hat{\omega}_L| I_L e^{\frac{-2(x_o - vt)^2}{w^2}} \tag{2}$$

where f_r is the bidirectional reflection distribution function (BRDF) of the point $\vec{p}_o$, $|\hat{n} \cdot \hat{\omega}_L|$ is the cosine of the angle between the surface and illumination. The remaining terms describe a point moving in the x-direction under the Gaussian illuminant of width w and power I_L.

Projecting the point $\vec{p}(t)$ onto the sensor, we find:

$$s = (x_o - vt)cos\theta - z_o sin\theta \tag{3}$$

where s is the position on the sensor and θ is the angle between the sensor and laser directions. We combine Equations 2-3 to give us an equation for the irradiance observed at the sensor as a function of time and position on the sensor:

$$E_S(t, s) \quad = \quad f_r(\hat{\omega}_L, \hat{\omega}_S)|\hat{n} \cdot \hat{\omega}_L| I_L e^{\frac{-2(x_o - vt)^2}{w^2}}$$

$$\delta(s - (x_o - vt)cos\theta - z_o sin\theta) \quad (4)$$

To simplify this expression, we condense the light reflection terms into one measure:

$$\beta \equiv f_r(\hat{\omega}_L, \hat{\omega}_S)|\hat{n} \cdot \hat{\omega_L}| \quad (5)$$

which we will refer to as the reflectance coefficient of point $\vec{p}$ for the given illumination and viewing directions. We also note that $x = vt$ is a measure of the relative x-displacement of the point during a scan, and $z = s/sin\theta$ is the relation between sensor coordinates and depth values along the center of the illuminant. Making these substitutions we have:

$$\begin{aligned} E_S(x,z) &= \beta I_L e^{\frac{-2(x-x_o)^2}{w^2}} \\ &\quad \delta((x - x_o)cos\theta + (z - z_o)sin\theta) \end{aligned} \quad (6)$$

This equation describes a Gaussian running along a *tilted line* through the spacetime sensor plane or "spacetime image". We define the "spacetime image" to be the image whose columns are filled with sensor scanlines that evolve over time. Through the substitutions above, position within a column of this image represents displacement in depth, and position within a row represents time or displacement in lateral position. Figure 7 shows the theoretical spacetime image of a single point based on the derivation above, while Figures 8a and 8b shows the spacetime image generated during a real scan. From Figure 7, we see that the tilt angle is $-\theta$ with respect to the z-axis, and the width of the Gaussian along the line is:

$$w' = w/cos\theta \quad (7)$$

The peak value of the Gaussian is βI_L, and its mean along the line is located at (x_o, z_o), the exact location of the range point. Note that the angle of the line and the width of the Gaussian are solely determined by the fixed parameters of the scanner, *not* the position, orientation, or BRDF of the surface element.

Thus, extraction of range points should proceed by computing low order statistics along tilted lines through the sensor spacetime image, rather than along columns (scanlines) as in the conventional method. As a result, we can determine the position of the surface element independently of the orientation and BRDF of the element and independently of any other nearby surface elements. In theory, the decoupling of range determination from local shape and reflectance is complete. In practice, optical systems and sensors have filtering and sampling properties that limit the ability to resolve neighboring points. In Figure 8d, for instance, the extracted edges extend slightly beyond their actual bounds. We attribute this artifact to filtering which blurs the exact cutoffs of the edges into neighboring pixels in the spacetime image, causing us to find additional range values.

As a side effect of the spacetime analysis, the peak of the Gaussian yields the irradiance at the sensor due to the point. Thus, we automatically obtain an intensity image precisely registered to the range image.

3.3 Generalizing the geometry

We can easily generalize the previous results to other scanner geometries under the following conditions:

- The illuminant direction is constant over the path of each range point.
- The sensor is orthographic.
- The motion is purely translational.

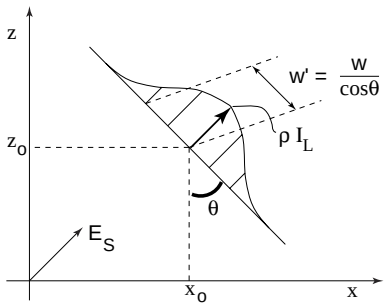


Figure 7: Spacetime image of a point passing through a Gaussian illuminant.

These conditions ensure that the reflectance coefficient, $\beta = f_r(\hat{\omega}_L, \hat{\omega}_S)|\hat{n} \cdot \hat{\omega}_L|$, is constant. Note that the illumination need only be directional; coherent or incoherent light of any pattern is acceptable. Further, the translational motion need not be of constant speed, only constant direction; we can correct for known variations in speed by applying a suitable warp to the spacetime image.

We can weaken each of these restrictions if β does not vary appreciably for each point as it passes through the illuminant. A perspective sensor is suitable if the changes in viewing directions are relatively small for neighboring points inside the illuminant. This assumption of "local orthography" has yielded excellent results in practice. In addition, we can tolerate a rotational component to the motion as long as the radius of curvature of the point path is large relative to the beam width, again minimizing the effects on β.

3.4 Correcting laser speckle

The discussion in sections 3.1-3.3 show how we can go about extracting accurate range data in the presence of shape and reflectance variations, as well as occlusions. But what about laser speckle? Empirical observation of the time evolution of the speckle pattern with our optical triangulation scanner strongly suggests that the image of laser speckle moves as the surface moves. The streaks in the spacetime image of Figure 8b correspond to speckle noise, for the object has uniform reflectance and should result in a spacetime image with uniform peak amplitudes. These streaks are tilted precisely along the direction of the spacetime analysis, indicating that the speckle noise adheres to the surface of the object and behaves as a noisy reflectance variation. Other researchers have observed a "stationary speckle" phenomenon as well [1]. Proper analysis of this problem is an open question, likely to be resolved with the study of the governing equations of scalar diffraction theory for imaging of a rough translating surface under coherent Gaussian beam illumination [6].

4 Implementation

We have implemented the spacetime analysis presented in the previous section using a commercial laser triangulation scanner and a real-time digital video recorder.

4.1 Hardware

The optical triangulation system we use is a Cyberware MS platform scanner. This scanner collects range data by casting a laser stripe on the object and by observing reflections with a CCD camera positioned at an angle of 30^o with respect to the plane of the

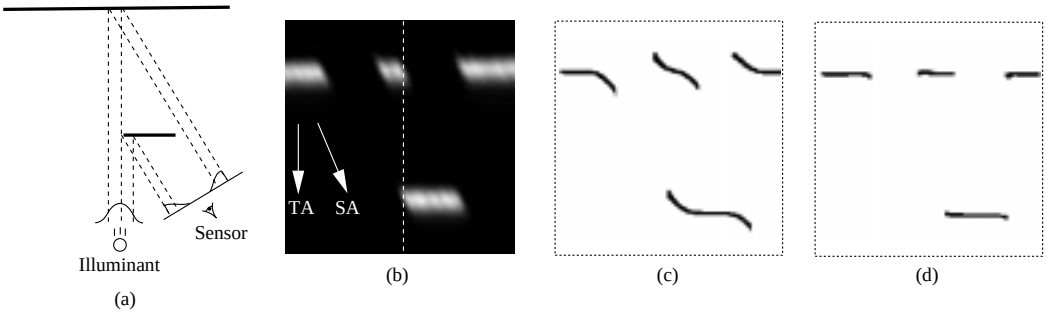


Figure 8: From geometry to spacetime image to range data. (a) The original geometry. (b) The resulting spacetime image. TA indicates the direction of traditional analysis, while SA is the direction of the spacetime analysis. The dotted line corresponds to the scanline generated at the instant shown in (a). (c) Range data after traditional mean analysis. (d) Range data after spacetime analysis.

laser. The platform can either translate or rotate an object through the field of view of the triangulation optics. The laser width varies from 0.8 mm to 1.0 mm over the field of view which is approximately 30 cm in depth and 30 cm in height. Each CCD pixel images a portion of the laser plane roughly 0.5 mm by 0.5 mm. Although the Cyberware scanner performs a form of peak detection in real time, we require the actual video frames of the camera for our analysis. We capture these frames with an Abekas A20 video digitizer and an Abekas A60 digital video disk, a system that can acquire 486 by 720 size frames at 30 Hz. These captured frames have approximately the same resolution as the Cyberware range camera, though they represent a resampling of the reconstructed CCD output.

4.2 Algorithms

Using the principles of section 3, we can devise a procedure for extracting range data from spacetime images:

1. Perform the range scan and capture the spacetime images.
2. Rotate the spacetime images by $-\theta$.
3. Find the statistics of the Gaussians in the rotated coordinates.
4. Rotate the means back to the original coordinates.

In order to implement step 1 of this algorithm, we require a sequence of CCD images. Most commercial optical triangulation systems discard each CCD image after using it (e.g. to compute a stripe of the range map). As described in section 4.1, we have assembled the necessary hardware to record the CCD frames. In section 3, we discussed a one dimensional sensor scenario and indicated that perspective imaging could be treated as locally orthographic. For a two dimensional sensor, we can imagine the horizontal scanlines as separate one dimensional sensors with varying vertical (y) offsets. Each scanline generates a spacetime image, and by stacking the spacetime images one atop another, we define a spacetime *volume*. In general, we must perform our analysis along the paths of points, paths which may cross scanlines within the spacetime volume. However, we have observed for our system that the illuminant is sufficiently narrow and the perspective of the range camera sufficiently weak, that these paths essentially remain within scanlines. This observation allows us to perform our analysis on each spacetime image separately.

In step 2, we rotate the spacetime images so that Gaussians are vertically aligned. In a practical system with different sampling rates in x and z, the correct rotation angle can be computed as:

$$tan\theta = \frac{\tau_z}{\tau_x} tan\theta_T \tag{8}$$

where θ is the new rotation angle, τ_x and τ_z are the sample spacing in x and z respectively, and θ_T is the triangulation angle. In order to determine the rotation angle, θ, for a given scanning rate and region of the field of view of our Cyberware scanner, we first determined the local triangulation angle and the sample spacings in depth (z) and lateral position (x). Equation 8 then yields the desired angle.

In step 3, we compute the statistics of the Gaussians along each rotated spacetime image raster. Our method of choice for computing these statistics is a least squares fit of a parabola to the log of the data. We have experimented with fitting the data directly to Gaussians using the Levenberg-Marquardt non-linear least squares algorithm [13], but the results have been substantially the same as the log-parabola fits. The Gaussian statistics consist of a mean, which corresponds to a range point, as well as a width and a peak amplitude, both of which indicate the reliability of the data. Widths that are far from the expected width and peak amplitudes near the noise floor of the sensor imply unreliable data which may be downweighted or discarded during later processing (e.g., when combining multiple range meshes [18]). For the purposes of this paper, we discard unreliable data.

Finally, in step 4, we rotate the range points back into the global coordinate system.

Traditionally, researchers have extracted range data at sampling rates corresponding to one range point per sensor scanline per unit time. Interpolation of shape between range points has consisted of fitting primitives (e.g., linear interpolants like triangles) to the range points. Instead, we can regard the spacetime volume as the primary source of information we have about an object. After performing a real scan, we have a sampled representation of the spacetime volume, which we can then reconstruct to generate a continuous function. This function then acts as our range oracle, which we can query for range data at a sampling rate of our choosing. In practice, we can magnify the sampled spacetime volume prior to applying the range imaging steps described above. The result is a range grid with a higher sampling density based directly on the imaged light reflections.

5 Results

5.1 Reflectance correction

To evaluate the tolerance of the spacetime method to changes in reflectance, we performed two experiments, one quantitative and the other qualitative. For the first experiment, we generated planar cards with step reflectance changes varying from about 1:1 to 10:1 and scanned them at an angle of 30^o (roughly facing the sen-

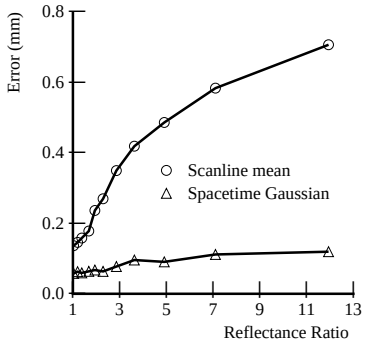


Figure 9: Measured error due to varying reflectance steps.

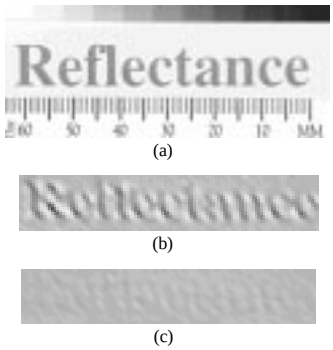


Figure 10: Reflectance card. (a) Photograph of a planar card with the word "Reflectance" printed on it, and shaded renderings of the range data generated by (b) mean pulse analysis and (c) spacetime analysis.

sor). Figure 9 shows a plot of maximum deviations from planarity when using traditional per scanline mean analysis and our spacetime analysis. The spacetime method has clearly improved over the old method, yielding up to 85% reductions in range errors.

For qualitative comparison, we produced a planar sheet with the word "Reflectance" printed on it. Figure 10 shows the results. The old method yields a surface with the characters well-embossed into the geometry, whereas the spacetime method yields a much more planar surface indicating successful decoupling of geometry and reflectance.

5.2 Shape correction

We conducted several experiments to evaluate the effects of shape variation on range acquisition. In the first experiment, we generated corners of varying angles by abutting sharp edges of machined aluminum wedges which are painted white. Figure 11 shows the range errors that result for traditional and spacetime methods. Again, we see an increase in accuracy, though not as great as in the reflectance case.

We also scanned two 4 mm strips of paper at an angle of 30^o (roughly facing the sensor) to examine the effects of depth continuity. Figure 12b shows the "edge curl" observed with the old method, while the spacetime method in Figure 12c shows a significant reduction of this artifact under spacetime analysis. We have found that the spacetime method reduces the length of the edge curl from an average of 1.1 mm to an average of approximately 0.35

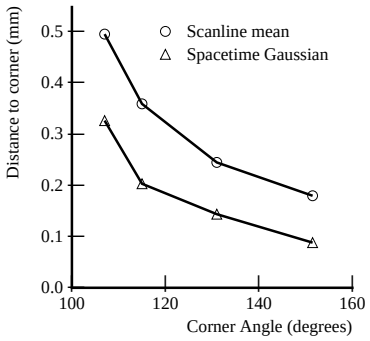


Figure 11: Measured error due to corners of varying angles.

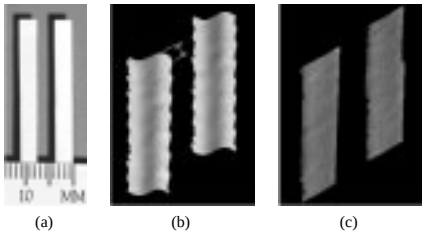


Figure 12: Depth discontinuities and edge curl. (a) Photograph of two strips of paper, and shaded renderings of the range data generated by (b) mean pulse analysis and (c) spacetime analysis. The "edge curl" indicated by the hash-marks in (b) is 1.1mm.

mm.

Finally, we impressed the word "shape" onto a plastic ribbon using a commonly available label maker. In Figure 10, we wanted the word "Reflectance" to disappear because it represented changes in reflectance rather than in geometry. In Figure 13, we want the word "Shape" to stay because it represents real geometry. Furthermore, we wish to resolve it as highly as possible. Figure 13 shows the result. Using the scanline mean method, the word is barely visible. Using the new spacetime analysis, the word becomes legible.

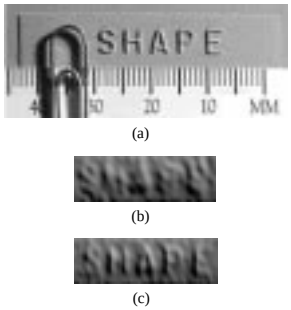


Figure 13: Shape ribbon. (a) Photograph of a surface with raised lettering (letters are approx. 0.3 mm high), and renderings of the range data generated by (b) mean pulse analysis and (c) spacetime analysis.

5.3 Speckle reduction

We performed range scans on the planar surfaces and generated range points using the traditional and spacetime methods. After fitting planes to range points, we found a 30-60% reduction in average deviation from planarity when using the spacetime analysis.

5.4 A complex object

Figure 14 shows the results of scanning a model tractor. Figure 14b is a rendering of the data generated by the Cyberware scanner hardware and is particularly noisy. This added noisiness results from the method of pulse analysis performed by the hardware, a method similar to peak detection. Peak detection is especially susceptible to speckle noise, because it extracts a range point based on a single value or small neighborhood of values on a noisy curve. Mean analysis tends to average out the speckle noise, resulting in smoother range data as shown in Figure 14c. Figure 14d shows our spacetime results and Figure 14e shows the spacetime results with 3X interpolation and resampling of the spacetime volume as described in section 4.2. Note the sharper definition of features on the body of the tractor and less jagged edges in regions of depth discontinuity.

5.5 Remaining sources of error

The results we presented in this section clearly show that the spacetime analysis yields more accurate range data, but the results are imperfect due to system limitations. These limitations include:

- CCD noise
- Finite sensor resolution
- Optical blurring and electronic filtering
- Quantization errors
- Calibration errors
- Surface-surface inter-reflections

In addition, we observed some electronic artifacts in our Cyberware scanner that influenced our results. We expect, however, that any measures taken to reduce the effects of the limiting factors described above will lead to higher accuracy. By contrast, if one uses traditional methods of range extraction, then increasing sensor resolution and reducing the effects of filtering alone will *not* significantly increase tolerance to reflectance and shape changes when applying the traditional methods of range extraction.

6 Conclusion

We have described several of the systematic limitations in traditional methods of range acquisition with optical triangulation range scanners, including intolerance to reflectance and shape changes and speckle noise. By analyzing the time evolution of the reflected light imaged onto the sensor, we have shown that distortions induced by shape and reflectance changes can be corrected, while the influence of laser speckle can be reduced. In practice, we have demonstrated that we can significantly reduce range distortions with existing hardware. Although the spacetime method does not completely eliminate range artifacts in practice, it has proven to reduce the artifacts in all experiments we have conducted.

In future work, we plan to incorporate the improved range data with algorithms that integrate partial triangulation scans into complete, unified meshes. We expect this improved data to ease the process of estimating topology, especially in areas of high curvature which are prone to edge curl artifacts. We will also investigate methods for increasing the resolution of the existing hardware by registering and deblurring multiple spacetime images [9]. Finally, we hope to apply the results of scalar diffraction theory to put the achievement of speckle reduction on sound theoretical footing.

Acknowledgments

We thank the people of Cyberware for the use of the range scanner and for their help in accessing the raw video output from the range camera.

References

[1] R. Baribeau and M. Rioux. Influence of speckle on laser range finders. *Applied Optics*, 30(20):2873–2878, July 1991.

[2] Paul Besl. *Advances in Machine Vision*, chapter 1 - Active optical range imaging sensors, pages 1–63. Springer-Verlag, 1989.

[3] G. Bickel, G. Haulser, and M. Maul. Triangulation with expanded range of depth. *Optical Engineering*, 24(6):975–977, December 1985.

[4] M. Buzinski, A. Levine, and W.H. Stevenson. Performance characteristics of range sensors utilizing optical triangulation. In *Proceedings of the IEEE 1992 National Aerospace and Electronics Conference, NAECON 1992*, pages 1230–1236, May 1992.

[5] R.G. Dorsch, G. Hausler, and J.M. Herrmann. Laser triangulation: fundamental uncertainty in distance measurement. *Applied Optics*, 33(7):1306–1314, March 1994.

[6] Joseph W. Goodman. *Introduction to Fourier optics*. McGraw-Hill, 1968.

[7] J.W. Goodman. *Laser Speckle and Related Phenomena*, chapter 1 - Statistical properties of laser speckle patterns, pages 9–76. Springer-Verlag, 1984.

[8] G. Hausler and W. Heckel. Light sectioning with large depth and high resolution. *Applied Optics*, 27(24):5165–5169, Dec 1988.

[9] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP: Graphical Models and Image Processing*, 53(3):231–239, May 1991.

[10] R.A. Jarvis. A perspective on range-finding techniques for computer vision. *IEEE Trans. Pattern Analysis Mach. Intell.*, 5:122–139, March 1983.

[11] T Kanade, A Gruss, and L Carley. A very fast vlsi rangefinder. In *1991 IEEE International Conference on Robotics and Automation*, volume 39, pages 1322–1329, April 1991.

[12] J.L. Mundy and G.B. Porter. *Three-dimensional machine vision*, chapter 1 - A three-dimensional sensor based on structured light, pages 3–61. Kluver Academic Publishers, 1987.

[13] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 1986.

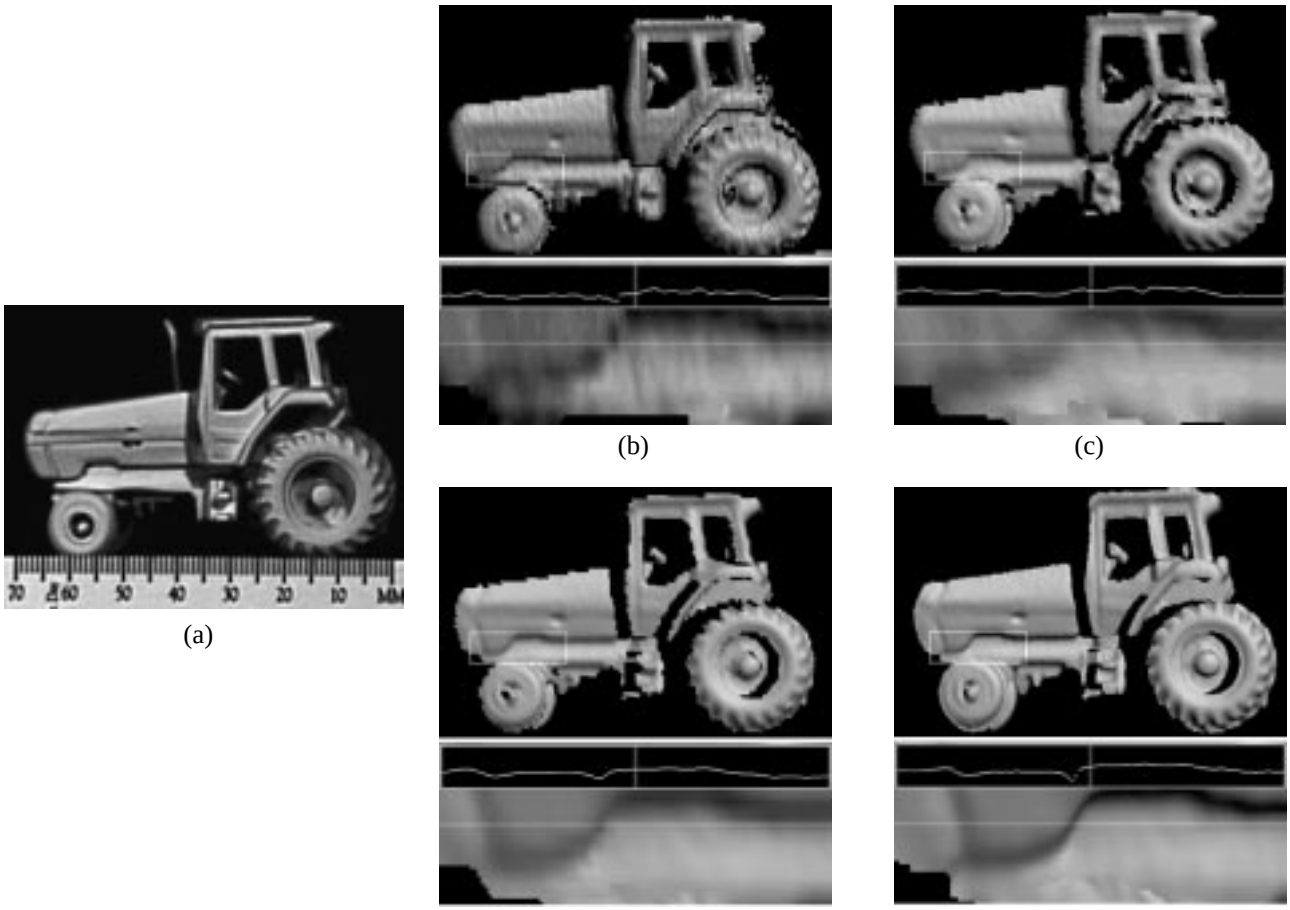


Figure 14: Model tractor. (a) Photograph of original model and shaded renderings of range data generated by (b) the Cyberware scanner hardware, (c) mean pulse analysis, (d) our spacetime analysis, and (e) the spacetime analysis with 3X interpolation of the spacetime volume before fitting the Gaussians. Below each of the renderings is a blow-up of one section of the tractor body (indicated by rectangle on rendering) with a plot of one row of pixel intensities.

[14] M. Rioux, G. Bechthold, D. Taylor, and M. Duggan. Design of a large depth of view three-dimensional camera for robot vision. *Optical Engineering*, 26(12):1245–1250, Dec 1987.

[15] A.E. Siegman. *Lasers*. University Science Books, 1986.

[16] M. Soucy, D. Laurendeau, D. Poussart, and F. Auclair. Behaviour of the center of gravity of a reflected gaussian laser spot near a surface reflectance discontinuity. *Industrial Metrology*, 1(3):261–274, Sept 1990.

[17] T. Strand. Optical three dimensional sensing. *Optical Engineering*, 24(1):33–40, Jan-Feb 1983.

[18] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *SIGGRAPH 94 Conference Proceedings*, pages 311–318, July 1994.

Desktop 3D Photography

Jean-Yves Bouguet

Intel Corporation
Microprocessor Research Lab

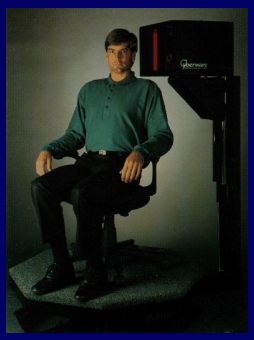
Pietro Perona

California Institute of Technology
Computer Vision Group

http://www.vision.caltech.edu/bouguetj

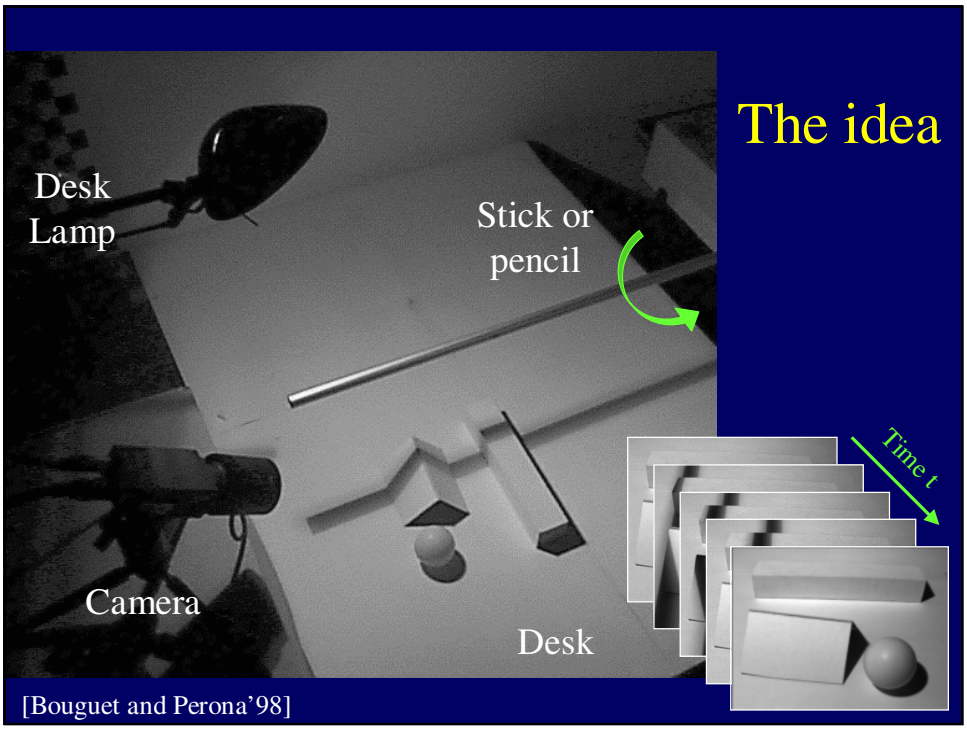
Goal: 3D reconstruction

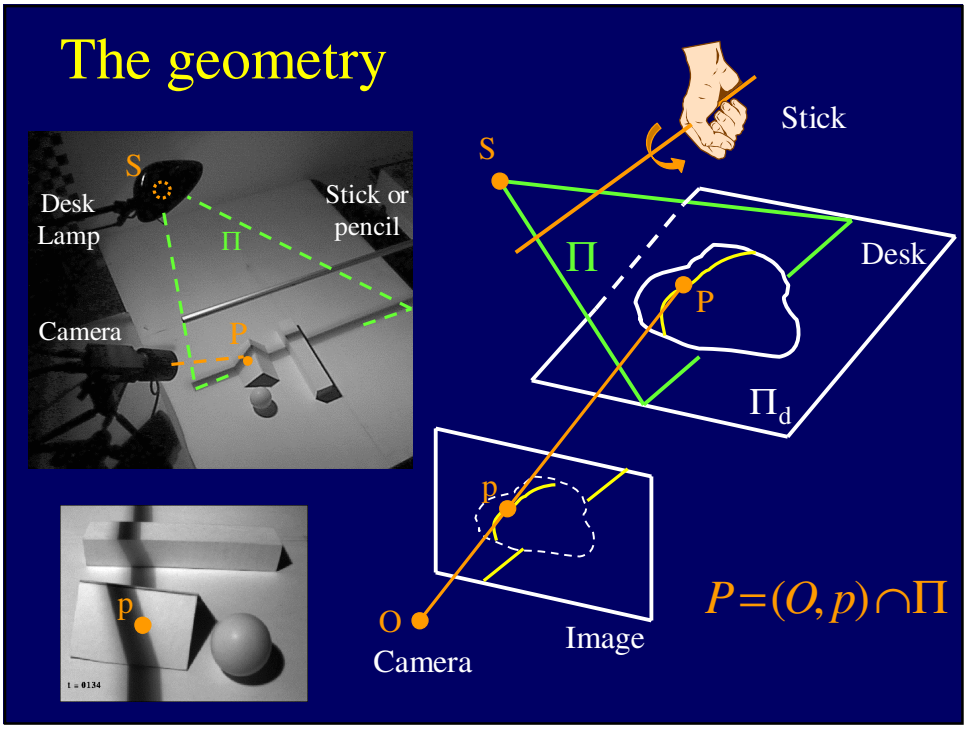
State of the art



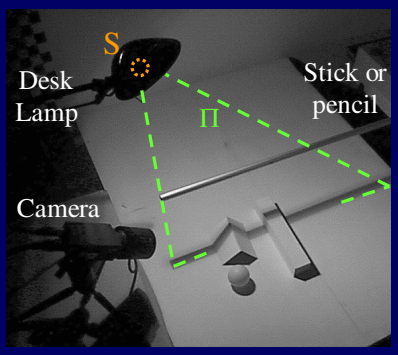
- Accurate
- Bulky
- Complicated
- Cost: >10k$

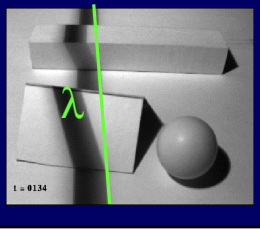
Weak structured lighting system

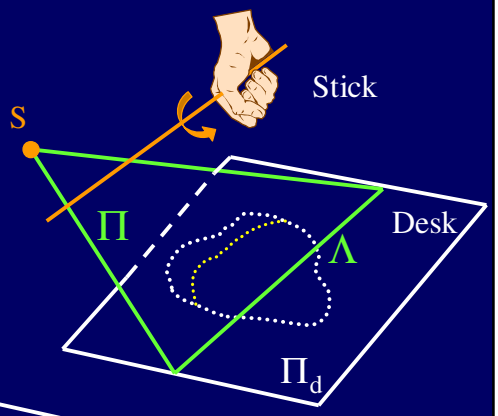

The idea
Desk Lamp
Stick or pencil
Camera
Desk
Time t
[Bouguet and Perona'98]

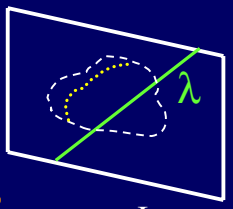

The geometry
S
Desk Lamp
Stick or pencil
Π
Camera
P
p
Stick
S
Π
Desk
P
Π_d
p
O
Camera
Image
$P=(O,p)\cap\Pi$

The geometry







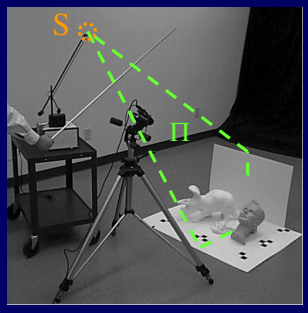


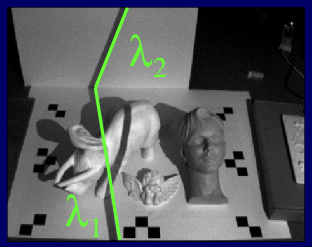
O
Camera
Image

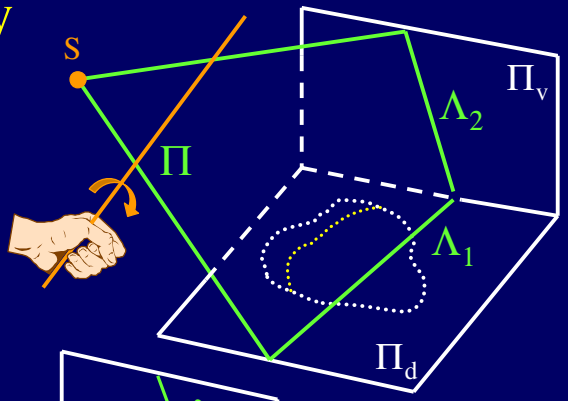
$\Lambda = (O, \lambda) \cap \Pi_d$

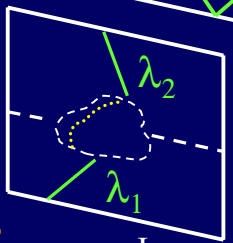
$\Pi = (S, \Lambda)$

The geometry









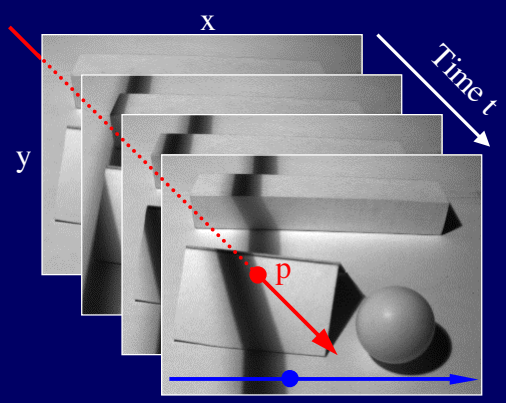
O
Camera
Image

$\Lambda_1 = (O, \lambda_1) \cap \Pi_d$

$\Lambda_2 = (O, \lambda_2) \cap \Pi_v$

$\Pi = (\Lambda_1, \Lambda_2)$

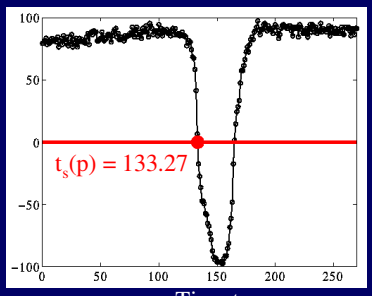
Spatio-temporal processing



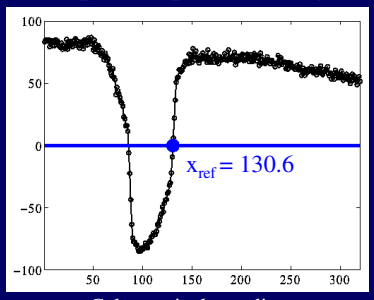
$t_s(p) = 133.27$

[Kanade'91,Curless'95]

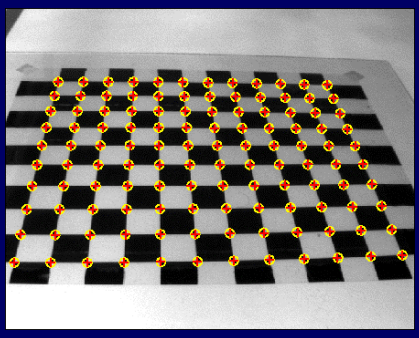
Temporal processing



Spatial processing



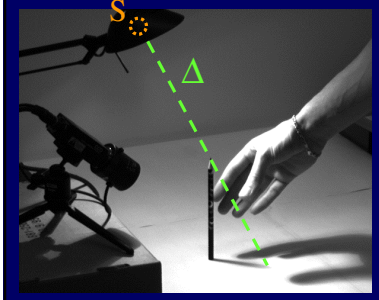
Camera calibration

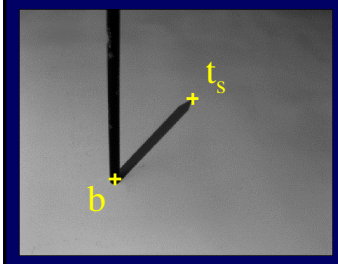


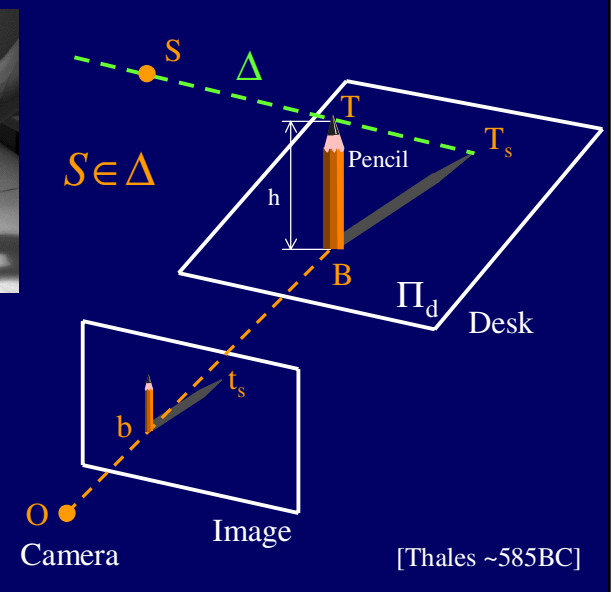
[Tsai'87, Abdel-Aziz and Karara'71]

- Position of the desk plane
- Internal parameters of the camera

Lamp calibration



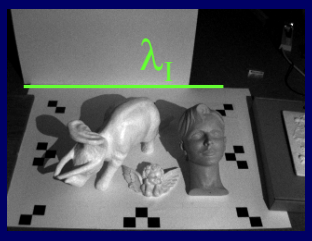


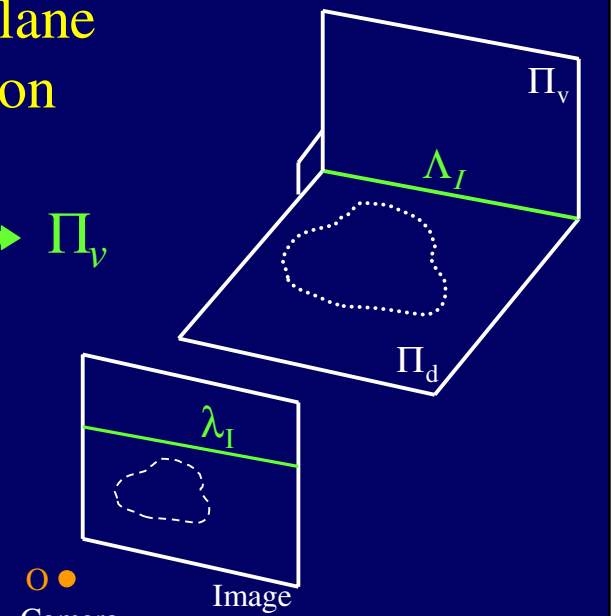


[Thales ~585BC]

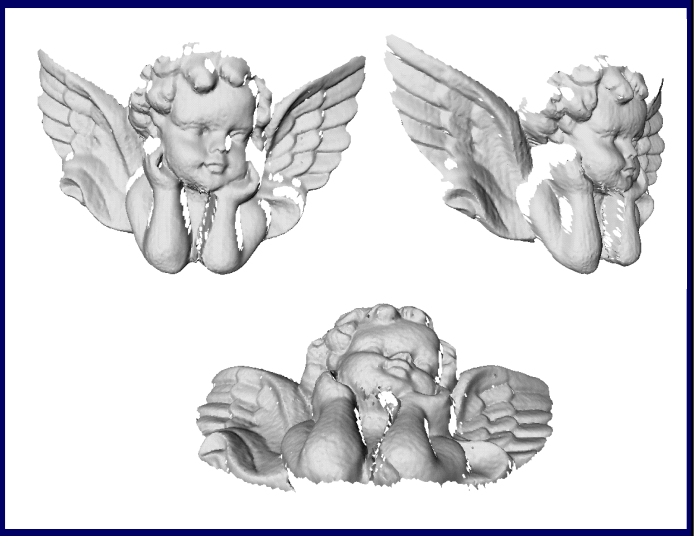
Vertical plane calibration

$\{\Pi_d, \lambda_I\} \rightarrow \Pi_v$



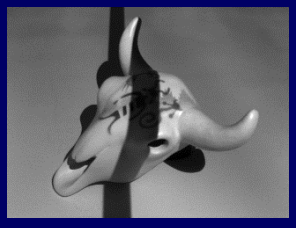


Angel experiment



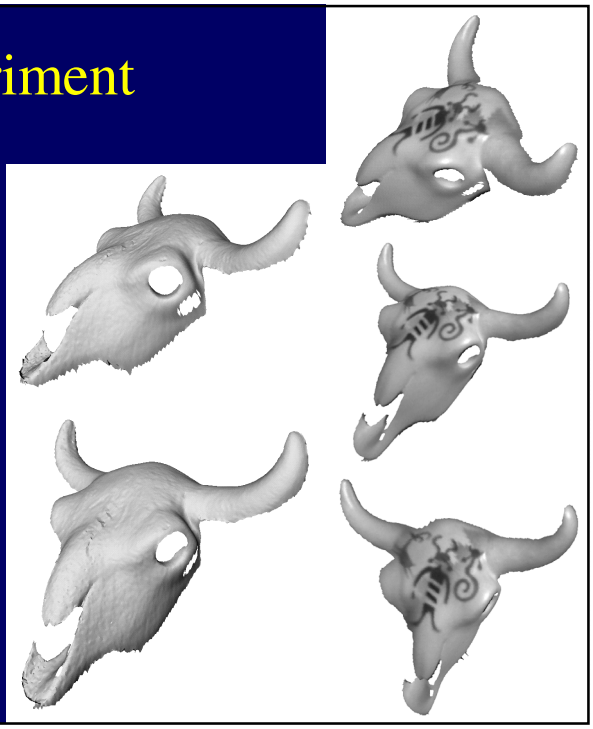
Accuracy: 0.1mm over 10cm → ~ 0.1% error

Skull experiment

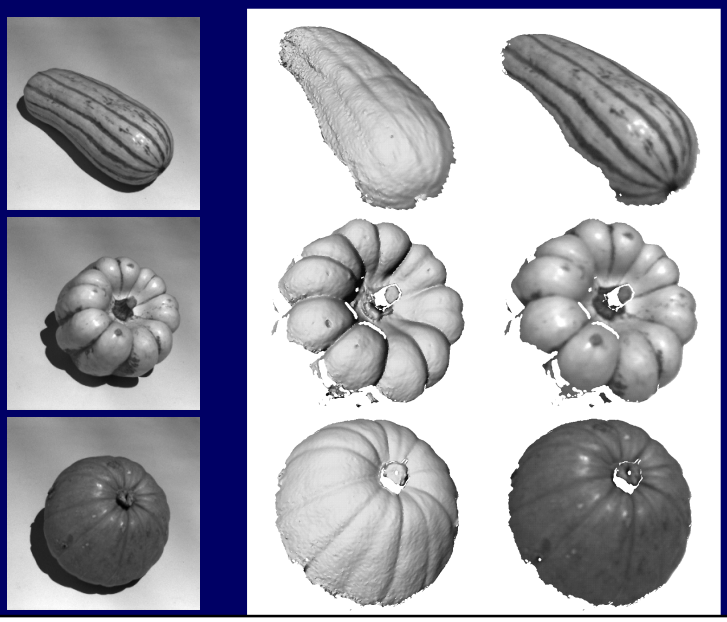


Accuracy: 0.1mm over 10cm

→ ~ 0.1% error

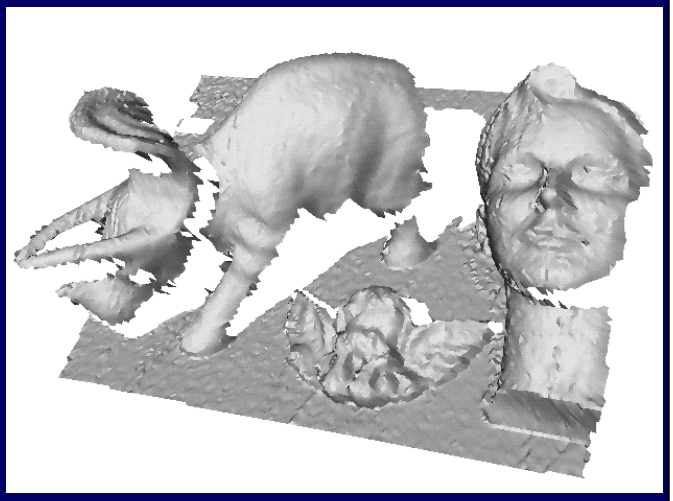


Textured objects



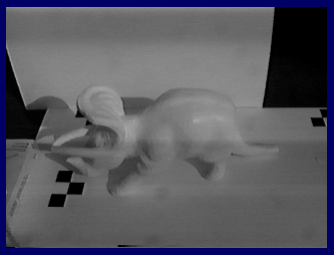
Other objects

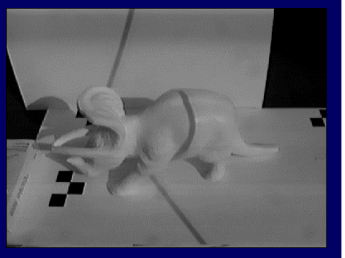
Pot-pourri scan



Accuracy: 0.5mm over 50cm → ~ 0.1% error

Scanning with the sun





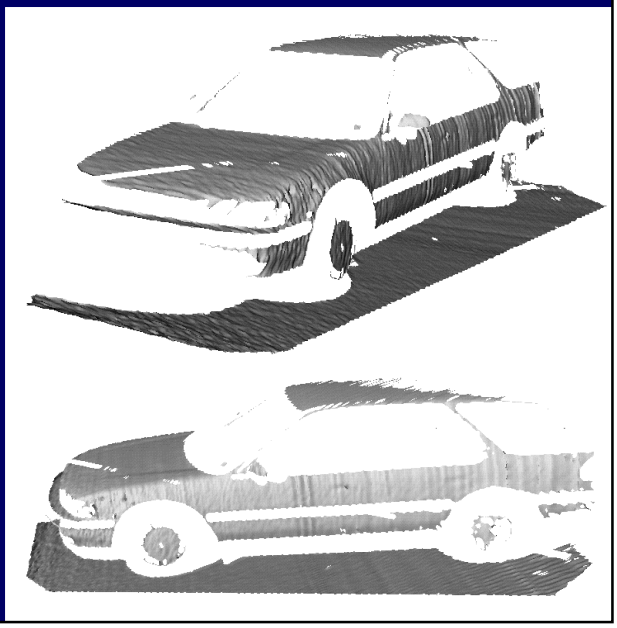
Accuracy: 1mm over 50cm

→ ~ 0.5% error

Scanning with the sun

Accuracy: 1cm over 2m

→ ~ 0.5% error



Error analysis

$$\sigma_Z^2 \propto \frac{1}{d^2} \cdot \frac{1}{\nabla I^2} \cdot \sigma_I^2$$

Variance of the error in depth estimate

d : distance of the shadow plane Π to the camera optical center

∇I : shadow edge sharpness (image gradient)

Image brightness noise

[Bouguet'99]

Real-time implementation

- Performance: 30Hz, 320x240, Pentium II 300MHz
- Single shadow pass: 20 - 30 seconds (600-900 frames)
- Refined scanning: 1 - 2 minutes

Conclusions

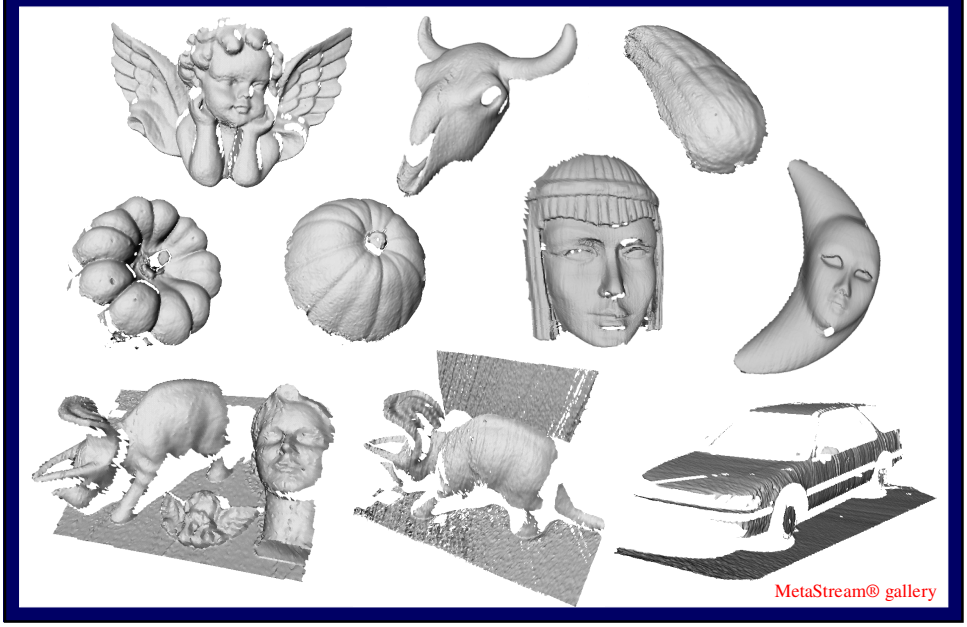
✓ Low cost and simple technique for dense 3D shape acquisition

✗ Does not work with specular or dark objects

What's next?

- Registration of multiple scans
  - ➡ complete models [Turk'94, Curless'96]

VRML gallery



References (1)

Space-time analysis:

- B. Curless and M. Levoy, "Better optical triangulation through spacetime analysis", ICCV95, pages 987-993, June 1995
- T. Kanade, A. Gruss and L. Carley, "A very fast VLSI rangefinder", IEEE International Conference on Robotics and Automation, volume 39, pages 1322-1329, April 1991

Camera calibration:

- R. Y. Tsai, "A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses", IEEE J. Robotics Automat., RA-3(4):323-344, 1987
- D.C. Brown, Calibration of close range cameras, Proc. 12th Congress Int. Soc. Photogrammetry, Ottawa, Canada
- Y. I. Abdel-Aziz and H. M. Karara, "Direct linear transformation into object space coordinates in close-range photogrammetry", Proc. ASP Symposium on Close-Range Photogrammetry, Urbana, Illinois, pages 1-18, 1971
- J.-Y. Bouguet and P. Perona, Camera calibration tutorial and Matlab code available at: http://www.vision.caltech.edu/bouguetj/calib_doc/

References (2)

Multiple view registration:

- G. Turk and M. Levoy, "Zippered polygon meshes from range images", SIGGRAPH'94, pages 311-318, July 1994
- B. Curless and M. Levoy, "A volumetric method for building complex models from range images", SIGGRAPH'96, 1996

Shadow scanning:

- J.-Y. Bouguet and P. Perona, "3D Photography Using Shadows in Dual-Space Geometry", Int. Journal of Computer Vision 35(2), 129-149, 1999 available at: http://www.vision.caltech.edu/bouguetj/ICCV98/
- J.-Y. Bouguet and P. Perona, "3D Photography on your desk", ICCV'98, pages 43-50, January 1998 available at: http://www.vision.caltech.edu/bouguetj/ICCV98/
- J.-Y. Bouguet, "Visual methods for three-dimensional modeling", Ph.D. thesis, California Institute of Technology, June 1999 available at: http://www.vision.caltech.edu/bouguetj/thesis/thesis.html

References (3)

Related work on shape from shadows:

- D. J. Kriegman and P. N. Belhumeur, "What Shadows Reveal About Object Structure", ECCV'98, pages 399-414, June 1998
- J. J. Clark, and L. Wang ," Trajectories for Optimal Temporal Integration in Active Vision Systems", Proceedings of the International Conference on Robotics and Automation, Albuquerque, April, 1997, pages 431-436
- M. Daum and G. Dudek, "On 3-D Surface Reconstruction Using Shape from Shadows", CVPR'98, pages 461-468, June 1998
- J.-Y. Bouguet, M. Weber and P. Perona, "What do planar shadows tell us about scene geometry?", CVPR'99, June 1999

1998 IEEE. Reprinted, with permission, from Proc. International Conference on Computer Vision
Bombay, India - January 1998 - pp. 43-50

3D photography on your desk

Jean-Yves Bouguet† and Pietro Perona†‡

† California Institute of Technology, 136-93, Pasadena, CA 91125, USA
‡ Università di Padova, Italy
{bouguetj,perona}@vision.caltech.edu

Abstract

A simple and inexpensive approach for extracting the three-dimensional shape of objects is presented. It is based on 'weak structured lighting'; it differs from other conventional structured lighting approaches in that it requires very little hardware besides the camera: a desk-lamp, a pencil and a checkerboard. The camera faces the object, which is illuminated by the desk-lamp. The user moves a pencil in front of the light source casting a moving shadow on the object. The 3D shape of the object is extracted from the spatial and temporal location of the observed shadow. Experimental results are presented on three different scenes demonstrating that the error in reconstructing the surface is less than 1%.

1 Introduction and Motivation

One of the most valuable functions of our visual system is informing us about the shape of the objects that surround us. Manipulation, recognition, and navigation are amongst the tasks that we can better accomplish by seeing shape. Ever-faster computers, progress in computer graphics, and the widespread expansion of the Internet have recently generated much interest in systems that may be used for imaging both the geometry and surface texture of object. The applications are numerous. Perhaps the most important ones are animation and entertainment, industrial design, archiving, virtual visits to museums and commercial on-line catalogues.

In designing a system for recovering shape, different engineering tradeoffs are proposed by each application. The main parameters to be considered are: cost, accuracy, ease of use and speed of acquisition. So far, the commercial 3D scanners (e.g. the Cyberware scanner) have emphasized accuracy over the other parameters. These systems use motorized transport of the object, and active (laser, LCD projector) lighting of the scene, which makes them very accurate, but expensive and bulky [1, 15, 16, 12, 2].

An interesting challenge for computer vision researchers is to take the opposite point of view: emphasize cost and simplicity, perhaps sacrificing some amount of accuracy, and design 3D scanners that demand little more hardware than a PC and a video camera, by now almost standard equipment both in offices and at home, by making better use of the data that is available in the images.

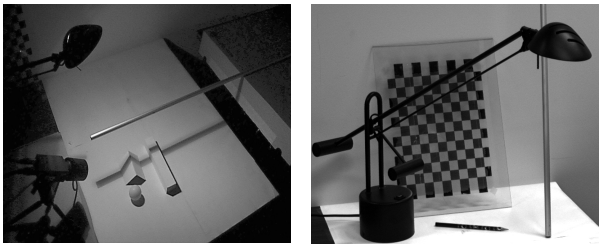


Figure 1: **The general setup of the proposed method:** The camera is facing the scene illuminated by a halogen desk lamp (left). The scene consists of objects on a plane (the desk). When an operator freely moves a stick in front of the lamp (over the desk), a shadow is cast on the scene. The camera acquires a sequence of images $I(x, y, t)$ as the operator moves the stick so that the shadow scans the entire scene. This constitutes the input data to the 3D reconstruction system. The variables x and y are the pixel coordinates (also referred to as spatial coordinates), and t the time (or frame number). The three dimensional shape of the scene is reconstructed using the spatial and temporal properties of the shadow boundary throughout the input sequence. The right-hand figure shows the necessary equipment besides the camera: a desk lamp, a calibration grid and a pencil for calibration, and a stick for the shadow. One could use the pencil instead of the stick.

A number of passive cues have long been known to contain information on 3D shape: stereoscopic disparity, texture, motion parallax, (de)focus, shadows, shading and specularities, occluding contours and other surface discontinuities amongst them. At the current state of vision research stereoscopic disparity is the single passive cue that gives reasonable accuracy. Unfortunately it has two major drawbacks: (a) it requires two cameras thus increasing complexity and cost, (b) it cannot be used on untextured surfaces (which are common for industrially manufactured objects).

We propose a method for capturing 3D surfaces that is based on 'weak structured lighting'. It yields good accuracy and requires minimal equipment besides a computer and a camera: a pencil (two uses), a checkerboard and a desk-lamp – all readily available in most homes; some intervention by a human operator, acting as a low precision motor, is also required.

We start with a description of the method in Sec. 2, followed in Sec. 3 by a noise sensitivity analysis, and in Sec. 4 by a number of experiments that assess the convenience and accuracy of the system. We end with a discussion and conclusions in Sec. 5.

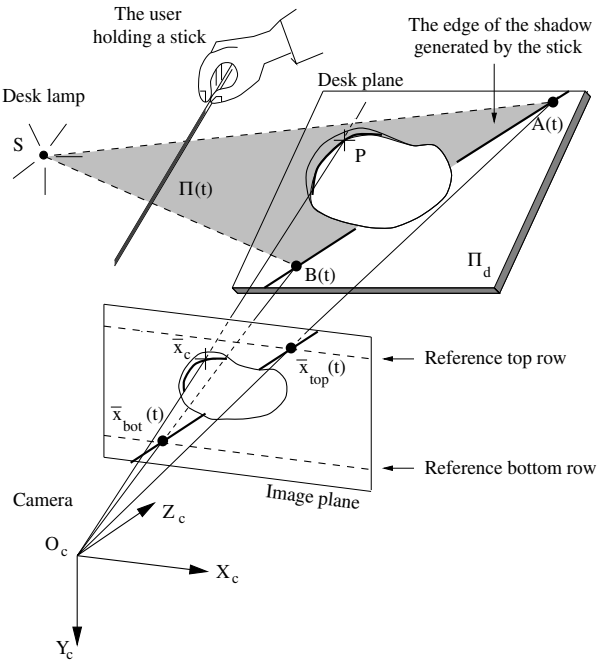


Figure 2: **Geometrical principle of the method:** Approximate the light source with a point S, and denote by Π_d the desk plane. Assume that the positions of the light source S and the plane Π_d in the camera reference frame are known from calibration. The goal is to estimate the 3D location of the point P in space corresponding to every pixel $\overline{x}_c$ in the image. Call t the time at which a given pixel $\overline{x}_c$ 'sees' the shadow boundary (later referred to as the shadow time). Denote by $\Pi(t)$ the corresponding shadow plane at that time t. Assume that two portions of the shadow projected on the desk plane are visible on two given rows of the image (top and bottom rows in the figure). After extracting the shadow boundary along those rows $\overline{x}_{\rm top}(t)$ and $\overline{x}_{\rm bot}(t)$, we find two points on the shadow plane $A(t)$ and $B(t)$ by intersecting Π_d with the optical rays $(O_c, \overline{x}_{\rm top}(t))$ and $(O_c, \overline{x}_{\rm bot}(t))$ respectively. The shadow plane $\Pi(t)$ is then inferred from the three points in space S, $A(t)$ and $B(t)$. Finally, the point P corresponding to $\overline{x}_c$ is retrieved by intersecting $\Pi(t)$ with the optical ray $(O_c, \overline{x}_c)$. This final stage is called triangulation. Notice that the key steps in the whole scheme are: (a) estimate the shadow time $t_s(\overline{x}_c)$ at every pixel $\overline{x}_c$ (*temporal processing*), and (b) locate the reference points $\overline{x}_{\rm top}(t)$ and $\overline{x}_{\rm bot}(t)$ at every time instant t (*spatial processing*). These two are discussed in detail in section 2.2.

2 Description of the method

The general principle consists of casting a shadow onto the scene with a pencil or another stick, and using the image of the deformed shadow to estimate the three dimensional shape of the scene. Figure 1 shows the required hardware and the setup of the system. The objective is to extract scene depth at every pixel in the image. Figure 2 gives a geometrical description of the method that we propose to achieve that goal.

2.1 Calibration

The goal of calibration is to recover the geometry of the setup (that is, the location of the desk plane Π_d and that of the light source S) as well as the *intrinsic parameters* of the camera (focal length, optical center and radial distortion factor). We decompose the procedure into two successive stages: first camera calibration and then lamp calibration.

Camera calibration: Estimate the intrinsic camera parameters and the location of the desk plane Π_d (tabletop) with respect to the camera. The procedure consists of first placing a planar checkerboard pattern (see figure 1) on the desk in the location of the objects to scan. From the image captured by the camera, we infer the *intrinsic* and *extrinsic* (rigid motion between camera and desk reference frame) parameters of the camera, by matching the projections onto the image plane of the known grid corners with the expected projection directly measured on the image (extracted corners of the grid). This method is very much inspired by the algorithm proposed by Tsai [13]. Note that since our calibration rig is planar, the optical center cannot be recovered through that process, and therefore is assumed to be fixed at the center of the image. A description of the whole procedure can be found in [3]. The reader can also refer to Faugeras [6] for further insights on camera calibration. Notice that the extrinsic parameters directly lead to the position of the tabletop Π_d in the camera reference frame.

Lamp calibration: After camera calibration, estimate the 3D location of the point light source S. Figure 3 gives a description of our method.

2.2 Spatial and temporal shadow edge localization

A fundamental stage of the method is the detection of the line of intersection of the shadow plane $\Pi(t)$ with the desktop Π_d; a simple approach may be used if we make sure that the top and bottom edges of the image are free from objects. Then the two tasks to accomplish are: **(a)** Localize the edge of the shadow that is directly projected on the tabletop $(\overline{x}_{\rm top}(t), \overline{x}_{\rm bot}(t))$ at every time instant t (every frame), leading to the set of all shadow planes $\Pi(t)$, **(b)** Estimate the time $t_s(\overline{x}_c)$ (*shadow time*) where the edge of the shadow passes through any given pixel $\overline{x}_c = (x_c, y_c)$ in the image. Curless and Levoy demonstrated in [4] that such a spatio-temporal approach is appropriate to preserve sharp discontinuities in the scene. Details of our implementation are given in figure 4. Notice that the shadow was scanned from the left to the right side of the scene. This explains why the right edge of the shadow corresponds to the front edge of the temporal profile in figure 4.

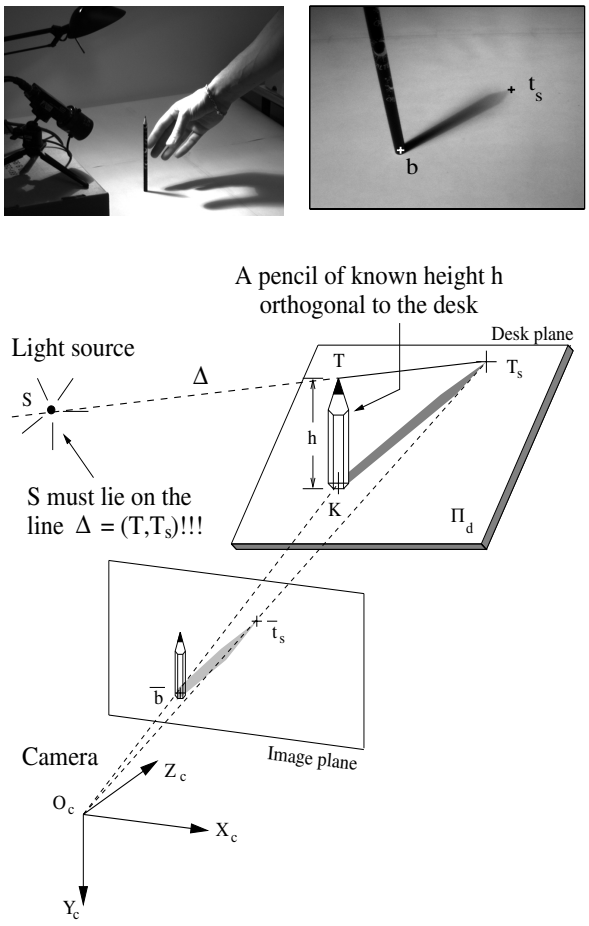


Figure 3: **Lamp calibration:** The operator places a pencil on the desk plane Π_d, orthogonal to it (top-left). The camera observes the shadow of the pencil projected on the tabletop. The acquired image is shown on the top-right. From the two points $\overline{b}$ and $\overline{t}_s$ on this image, one can infer the positions in space of K and T_s, respectively the base of the pencil, and the tip of the pencil shadow (see bottom figure). This is done by intersecting the optical rays $(O_c, \overline{b})$ and $(O_c, \overline{t}_s)$ with Π_d (known from camera calibration). In addition, given that the height of the pencil h is known, the coordinates of its tip T can be directly inferred from K. Then, the light source point S has to lie on the line $\Delta = (T, T_s)$ in space. This yields one linear constraint on the light source position. By taking a second view, with the pencil at a different location on the desk, one can retrieve a second independent constraint with another line Δ'. A closed form solution for the 3D coordinate of S is then derived by intersecting the two lines Δ and Δ' (in the least squares sense). Notice that since the problem is linear, one can easily integrate the information from more than 2 views and then make the estimation more accurate. If $N > 2$ images are used, one can obtain a closed form solution for the best intersection point $\tilde{S}$ of the N inferred lines (in the least squares sense). We also estimate the uncertainty on that estimate from the distance of $\tilde{S}$ from each one of the Δ lines. That indicates how consistently the lines intersect a single point in space. Refer to [3] for the complete derivations.

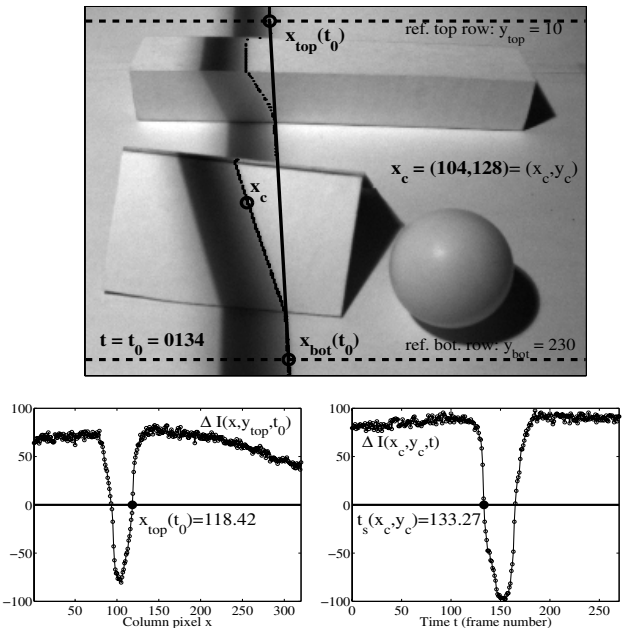


Figure 4: **Spatial and temporal shadow location:** The first step consists of localizing spatially the shadow edge $(\overline{x}_{\text{top}}(t_0), \overline{x}_{\text{bot}}(t_0))$ at every integer time t_0 (i.e. every frame). The top and bottom rows are $y_{\text{top}} = 10$ and $y_{\text{bot}} = 230$ on the top figure. This leads to an estimate of the shadow plane $\Pi(t_0)$ at every frame. The second processing step consists of extracting at every pixel $\overline{x}_c$, the time $t_s(\overline{x}_c)$ of passage of the shadow edge. For any given pixel $\overline{x}_c = (x, y)$, define $I_{\min}(x, y) \doteq \min_t (I(x, y, t))$ and $I_{\max}(x, y) \doteq \max_t (I(x, y, t))$ as its minimum and maximum brightness throughout the entire sequence. We then define the shadow edge to be the locations (in space-time) where the image $I(x, y, t)$ intersects with the threshold image $I_{\text{shadow}}(x, y) \doteq (I_{\min}(x, y) + I_{\max}(x, y))/2$. This may be also regarded as the zero crossings of the difference image $\Delta I(x, y, t) \doteq I(x, y, t) - I_{\text{shadow}}(x, y)$. The two bottom plots illustrate the shadow edge detection in the spatial domain (to find $\overline{x}_{\text{top}}$ and $\overline{x}_{\text{bot}}$) and in the temporal domain (to find $t_s(\overline{x}_c)$). The bottom-left figure shows the profile of $\Delta I(x, y, t)$ along the top reference row $y = y_{\text{top}} = 10$ at time $t = t_0 = 134$ versus the column pixel coordinate x. The second zero crossing of that profile corresponds to the top reference point $\overline{x}_{\text{top}}(t_0) = (118.42, 10)$ (computed at sub-pixel accuracy). Identical processing is applied on the bottom row to obtain $\overline{x}_{\text{bot}}(t_0) = (130.6, 230)$. Similarly, the bottom-right figure shows the temporal profile $\Delta I(x_c, y_c, t)$ at the pixel $\overline{x}_c = (x_c, y_c) = (104, 128)$ versus time t (or frame number). The shadow time at that pixel is defined as the first zero crossing location of that profile: $t_s(104, 128) = 133.27$ (computed at sub-frame accuracy).

Notice that the pixels corresponding to regions in the scene that are not illuminated by the lamp (shadows due to occlusions) do not provide any relevant depth information. For this reason we can restrict the processing to pixels that have sufficient swing between maximum and minimum brightness. Therefore, we only process pixels with contrast value $I_{\text{contrast}}(x, y) \doteq I_{\max}(x, y) - I_{\min}(x, y)$ larger than a pre-defined threshold I_{thresh}. This threshold was 70 in all experiments reported in this paper (recall that the intensity values

are encoded from 0 for black to 255 for white).

We do not apply any spatial filtering on the images; that would generate undesired blending in the final depth estimates, especially noticeable at depth discontinuities (at occlusions for example). However, it would be acceptable to low-pass filter the brightness profiles of the top and bottom rows (there is no depth discontinuity on the tabletop) and low-pass filter the temporal brightness profiles at every pixel. These operations would preserve sharp spatial discontinuities, and might decrease the effect of local processing noise by accounting for smoothness in the motion of the stick.

Experimentally, we found that this thresholding approach for shadow edge detection allow for some internal reflections in the scene [9, 8, 14]. However, if the light source is not close to an ideal point source, the mean value between maximum and minimum brightness may not always constitute the optimal value for the threshold image I_{shadow}. Indeed, the shadow edge profile becomes shallower as the distance between the stick and the surface increases. In addition, it deforms asymmetrically as the surface normal changes. These effects could make the task of detecting the shadow boundary points challenging. In the future, we intend to develop a geometrical model of extended light sources and incorporate it in the system.

Although $I_{\min}$ and $I_{\max}$ are needed to compute I_{shadow}, there exists an implementation of that algorithm that does not require storage of the complete image sequence in memory and therefore leads itself to real-time implementations. All that one needs to do is update at each frame five different arrays $I_{\max}(x,y)$, $I_{\min}(x,y)$, $I_{\text{contrast}}(x,y)$, $I_{\text{shadow}}(x,y)$ and the shadow time $t_s(x,y)$, as the images $I(x,y,t)$ are acquired. For a given pixel (x,y), the maximum brightness $I_{\max}(x,y)$ is collected at the very beginning of the sequence (the first frame), and then, as time goes, the incoming images are used to update the minimum brightness $I_{\min}(x,y)$ and the contrast $I_{\text{contrast}}(x,y)$. Once $I_{\text{contrast}}(x,y)$ crosses I_{thresh}, the adaptive threshold $I_{\text{shadow}}(x,y)$ starts being computed and updated at every frame (and activated). This process goes on until the pixel brightness $I(x,y,t)$ crosses $I_{\text{shadow}}(x,y)$ for the first time (in the upwards direction). That time instant is registered as the shadow time $t_s(x,y)$. In that form of implementation, the left edge of the shadow is tracked instead of the right one, however the principle remains the same.

2.3 Triangulation

Once the shadow time $t_s(\overline{x}_c)$ is estimated at a given pixel $\overline{x}_c$, one can identify the corresponding shadow plane $\Pi(t_s(\overline{x}_c))$. Then, the 3D point P associated to $\overline{x}_c$ is retrieved by intersecting $\Pi(t_s(\overline{x}_c))$ with the optical ray $(O_c, \overline{x}_c)$ (see figure 2). Notice that the shadow time $t_s(\overline{x}_c)$ acts as an index to the shadow plane list $\Pi(t)$. Since $t_s(\overline{x}_c)$ is estimated at sub-frame accuracy, the final plane $\Pi(t_s(\overline{x}_c))$ actually results from linear interpolation between the two planes $\Pi(t_0 - 1)$ and $\Pi(t_0)$ if $t_0 - 1 < t_s(\overline{x}_c) < t_0$ and t_0 integer. Once the range data are recovered, a mesh may be generated by connecting neighboring points in triangles. Rendered views of three reconstructed surface structures can be seen in figures 6, 7 and 8.

3 Noise Sensitivity

The overall scheme is based on first extracting from every frame (i.e. every time instants t) the x coordinates of the two reference points $x_{\text{top}}(t)$ and $x_{\text{bot}}(t)$, and second estimating the shadow time $t_s(\overline{x}_c)$ at every pixel $\overline{x}_c$. Those input data are used to estimate the depth Z_c at every pixel. The purpose of the noise sensitivity analysis is to quantify the effect of the noise in the measurement data $\{x_{\text{top}}(t), x_{\text{bot}}(t), t_s(\overline{x}_c))\}$ on the final reconstructed scene depth map. One key step in the analysis is to transfer the noise affecting the shadow time $t_s(\overline{x}_c)$ into a scalar noise affecting the x coordinate of $\overline{x}_c$ after scaling by the local shadow speed on the image at that pixel. Let V be the volume of the parallelepiped formed by the three vectors $\overline{O_cA}$, $\overline{O_cB}$ and $\overline{O_cS}$, originating at O_c (see figure 2):

$$V = \overline{X}_S^T . \left\{ (\overline{X}_B - \overline{X}_S) \times (\overline{X}_A - \overline{X}_S) \right\}$$

where $\overline{X}_S = [X_S \; Y_S \; Z_S]^T$, $\overline{X}_A = [X_A \; Y_A \; Z_A]^T$ and $\overline{X}_B = [X_B \; Y_B \; Z_B]^T$ are the coordinate vectors of S, A and B in the camera reference frame ($\times$ is the standard outer product operator). Notice that V is computed at the triangulation stage, and therefore is always available (see [3]). Define $\overline{X}_c = [X_c \; Y_c \; Z_c]^T$ as the coordinate vector in the camera reference frame of the point in space corresponding to $\overline{x}_c$. Assume that the x coordinates of the top and bottom reference points (after normalization) are affected by additive white Gaussian noise with zero mean and variances σ_{t}^2 and σ_{b}^2 respectively. Assume in addition that the variance on the x coordinate of $\overline{x}_c$ is $\sigma_{x_c}^2$ (different at every pixel). The following expression for the variance $\sigma_{Z_c}^2$ of the induced noise on the depth estimate Z_c was derived by taking first order derivatives of Z_c with respect to the 'new' noisy input variables x_{top}, x_{bot} and $\overline{x}_c$ (notice that the time variable does not appear any longer in the analysis):

$$\sigma_{Z_c}^2 = \frac{Z_c^2}{V^2} \Big\{ W^2 h_S^2 Z_c^2 \sigma_{x_c}^2 + \left(\alpha_1 + \beta_1 Y_c + \gamma_1 Z_c\right)^2 \sigma_{\text{t}}^2 + \left(\alpha_2 + \beta_2 Y_c + \gamma_2 Z_c\right)^2 \sigma_{\text{b}}^2 \Big\} \quad (1)$$

where W, h_S, α_1, β_1, γ_1, α_2, β_2 and γ_2 are constants depending only on the geometry (see figure 5):

$$\begin{aligned}
\alpha_1 &= Z_A\,(Z_B\,Y_S - Y_B\,Z_S)\\
\beta_1 &= -Z_A\,(Z_B - Z_S)\\
\gamma_1 &= Z_A\,(Y_B - Y_S)\\
\alpha_2 &= Z_B\,(Y_A\,Z_S - Z_A\,Y_S)\\
\beta_2 &= Z_B\,(Z_A - Z_S)\\
\gamma_2 &= -Z_B\,(Y_A - Y_S)
\end{aligned}$$

The first term in equation 1 comes from the temporal noise (on $t_s(\overline{x}_c)$ transferred to $\overline{x}_c$); the second and third terms from the spatial noise (on $\overline{x}_{\rm top}$ and $\overline{x}_{\rm bot}$). Let σ_I be the standard deviation of the image brightness noise. Given that we use linear interpolation of the temporal brightness profile to calculate the shadow time $t_s(\overline{x}_c)$, we can write σ_{x_c} as a function of the horizontal spatial image gradient $I_x(\overline{x}_c)$ at $\overline{x}_c$ at time $t = t_s(\overline{x}_c)$:

$$\sigma_{x_c} = \frac{\sigma_I}{|I_x(\overline{x}_c)|} \qquad (2)$$

Since σ_{x_c} in inversely proportional to the image gradient, the accuracy improves with shadow edge sharpness. This justifies the improvement in experiment 3 after removing the lamp reflector (thereby significantly increasing sharpness). In addition, observe that σ_{x_c} does not depend on the local shadow speed. Therefore, decreasing the scanning speed would not increase accuracy. However, for the analysis leading to equation 2 to remain valid, the temporal pixel profile must be sufficiently sampled within the transition area of the shadow edge (the penumbra). Therefore, if the shadow edge were sharper, the scanning should also be slower so that the temporal profile at every pixel would be properly sampled. Decreasing further the scanning speed would benefit the accuracy only if the temporal profile were appropriately low-pass filtered before extraction of $t_s(\overline{x}_c)$. This is an issue for future research.

Notice that σ_{Z_c}, aside from quantifying the uncertainties on the depth estimate Z_c at every pixel $\overline{x}_c$, it also constitutes a good indicator of the overall accuracies in reconstruction, since most of the errors are located along the Z direction of the camera frame. In addition, we found numerically that most of the variations in the variance $\sigma^2_{Z_c}$ are due to the variation of volume V within a single scan. This explains why the reconstruction noise is systematically larger in portions of the scene further away from the lamp (see figures 6, 7 and 8). Indeed, it can be shown that, as the shadow moves into the opposite direction of the lamp (e.g. to the right if the lamp is on the left of the camera), the absolute value of the volume $|V|$ strictly decreases, making $\sigma^2_{Z_c}$ larger (see [3] for details).

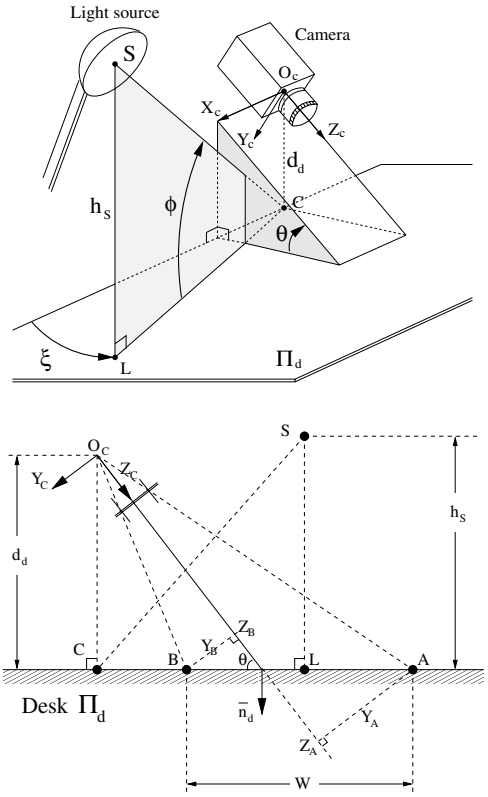


Figure 5: **Geometric setup:** The camera is positioned at a distance d_d away from the desk plane Π_d and tilted down towards it at an angle θ. The light source is located at a height h_S, with its direction defined by the azimuth and elevation angles ξ and ϕ. Notice the sign of $\cos\xi$ directly relates to which side of the camera the lamp is standing: positive on the right, and negative on the left. The bottom figure is a side view of the system (in the (O_c, Y_c, Z_c) plane). The points A and B are the reference points on the desk plane (see figure 2).

In order to obtain a uniformly accurate reconstruction of the entire scene, one may take two scans of the same scene with the lamp at two different locations (on the left (L) and on the right (R) of the camera), and merge them together using at each pixel the estimated reliability of the two measurements. Assume that the camera position, as well as the height h_S of the lamp, are kept identical for both scans. Suppose in addition that the scanning speeds were approximately the same. Then, at every pixel $\overline{x}_c$ in the image, the two scan data sets provide two estimates Z_c^L and Z_c^R of the same depth Z_c with respective reliabilities $\sigma^2_{Z_L}$ and $\sigma^2_{Z_R}$ given by equation 1. In addition, if we call V_L and V_R the two respective volumes, then the relative uncertainty between Z_c^L and Z_c^R reduces to a function of the volumes: $\sigma^2_{Z_R}/\sigma^2_{Z_L} = (V_L/V_R)^2$. Notice that calculating that relative uncertainty does not require any extra computation, since V_L and V_R are available from the two triangulations. The final depth is computed by weighted average of Z_c^L and Z_c^R: $Z_c \doteq \omega_L\,Z_c^L + \omega_R\,Z_c^R$. If Z_c^R and Z_c^L were Gaus-

sian distributed, and independent, they would be optimally averaged using the inverse of the variances as weights [10]: $\omega_L = \sigma^2_{Z_R}/(\sigma^2_{Z_L} + \sigma^2_{Z_R}) = \alpha^2/(1+\alpha^2)$ and $\omega_R = \sigma^2_{Z_L}/(\sigma^2_{Z_L} + \sigma^2_{Z_R}) = 1/(1+\alpha^2)$, where $\alpha = V_L/V_R$. Experimentally, we found that this choice does not yield very good merged surfaces. It makes the noisy areas of one view interact too significantly with the clean corresponding areas in the other view, degrading the overall final reconstruction. This happens possibly because the random variables Z_c^L and Z_c^R are not Gaussian. A heuristic solution to that problem is to use sigmoid functions to calculate the weights: $\omega_L = (1+\exp\{-\beta\Delta V\})^{-1}$, and $\omega_R = (1+\exp\{\beta\Delta V\})^{-1}$ with $\Delta V = (V_L^2 - V_R^2)/(V_L^2 + V_R^2) = (\alpha^2-1)/(\alpha^2+1)$. The positive coefficient β controls the amount of diffusion between the left and the right regions, and should be determined experimentally. In the limit, as β tends to infinity, merging reduces to a hard decision: $Z_c = Z_c^L$ if $V_L > V_R$, and $Z_c = Z_c^R$ otherwise. Our merging technique presents two advantages: (a) obtaining more coverage of the scene and (b) reducing the estimation noise. Moreover, since we do not move the camera between scans, we do not have to solve for the difficult problem of view alignment [11, 7, 5]. One merging example is presented in experiment 3.

Independently from local variations in accuracy within one scan, one would also wish to maximize the global (or average) accuracy of reconstruction throughout the entire scene. In this paper, scanning is vertical (shadow parallel to the y axis of the image). Therefore, the average relative depth error $|\sigma_{Z_c}/Z_c|$ is inversely proportional to $|\cos\xi|$ (see [3]). The two best values for the azimuth angle are then $\xi = 0$ and $\xi = \pi$ corresponding to the lamp standing either to the right ($\xi = 0$) or to the left ($\xi = \pi$) of the camera (see figure 5-top).

4 Experimental Results

4.1 Calibration accuracies

Camera calibration. For a given setup, we acquired 10 images of the checkerboard (see figure 1), and performed independent calibrations on them. The checkerboard consisted of approximately 90 visible corners on a 8×9 grid. Then, we computed both mean values and standard deviations of all the parameters independently: the focal length f_c, radial distortion factor k_c and desk plane position Π_d. Regarding the desk plane position, it is convenient to look at the height d_d and the surface normal vector $\overline{n}_d$ of Π_d expressed in the camera reference frame. An additional geometrical quantity related to $\overline{n}_d$ is the tilt angle θ (see figure 5). The following table summarizes the calibration results (notice that the relative error on the angle θ is computed referring to 360 degrees):

Parameters	Estimates	Relative errors
f_c (pixels)	857.3 ± 1.3	0.2%
k_c	-0.199 ± 0.002	1%
d_d (cm)	16.69 ± 0.02	0.1%
$\overline{n}_d$	$\begin{pmatrix} -0.0427 \pm 0.0003 \\ 0.7515 \pm 0.0003 \\ 0.6594 \pm 0.0004 \end{pmatrix}$	0.06%
θ (degrees)	41.27 ± 0.02	0.006%

Lamp calibration. Similarly, we collected 10 images of the pencil shadow (like figure 3-top-right) and performed calibration of the light source on them. See section 2.1. Notice that the points $\overline{b}$ and t_s were manually extracted from the images. Define $\overline{S}_c$ as the coordinate vector of the light source in the camera frame. The following table summarizes the calibration results (refer to figure 5 for notation):

Parameters	Estimates	Relative errors
$\overline{S}_c$ (cm)	$\begin{pmatrix} -13.7 \pm 0.1 \\ -17.2 \pm 0.3 \\ -2.9 \pm 0.1 \end{pmatrix}$	$\approx 2\%$
h_S (cm)	34.04 ± 0.15	0.5%
ξ (degrees)	146.0 ± 0.8	0.2%
ϕ (degrees)	64.6 ± 0.2	0.06%

The estimated lamp height agrees with the manual measure (with a ruler) of 34 ± 0.5 cm.

Our method yields an accuracy of approximately 3 mm (in standard deviation) in localizing the light source. This accuracy is sufficient for final shape recovery without significant deformation, as we discuss in the next section.

4.2 Scene reconstructions

On the first scene (figure 6), we evaluated the accuracy of reconstruction based on the sizes and shapes of the plane at the bottom left corner and the corner object on the top of the scene (see figure 4-top).

Planarity of the plane: We fit a plane across the points lying on the planar patch and estimated the standard deviation of the set of residual distances of the points to the plane to 0.23 mm. This corresponds to the granularity (or roughness) noise on the planar surface. The fit was done over a surface patch of approximate size 4 cm $\times$ 6 cm. This leads to a relative non planarity of approximately 0.23mm/5cm = 0.4%. To check for possible global deformations due to errors in calibration, we also fit a quadratic patch across those points. We noticed a decrease of approximately 6% in residual standard deviation after quadratic warping. This leads us to believe that global geometric deformations are negligible compared to local surface noise. In other words, one may assume that the errors of calibration do not induce significant global deformations on the final reconstruction.

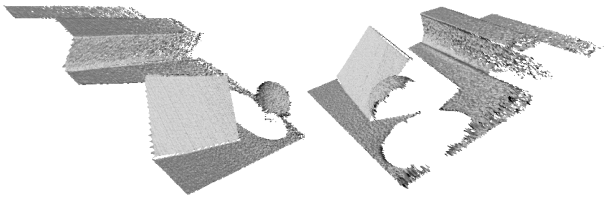


Figure 6: **Experiment 1 - The plane/ball/corner scene:** Two views of the mesh generated from the cloud of points obtained after triangulation. The original sequence was 270 frames long, the images being 320×240 pixels each. At 60 Hz acquisition frequency, the entire scanning take 5 seconds. The camera was positioned at distance $d_d = 16.7$ cm from the desk plane, tilted down by $\theta = 41.3$ degrees. The light source was at height $h_S = 37.7$ cm, on the left of the camera at angles $\xi = 157.1$ degrees and $\phi = 64.8$ degrees. From the right-hand figure we notice that the right-hand side of the reconstructed scene is more noisy than the left-hand side. This was expected since the lamp was standing on the left of the camera (refer to section 3 for details).

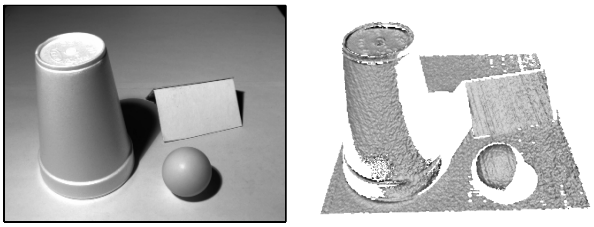


Figure 7: **Experiment 2 - The cup/plane/ball scene:** The scanned objects were a cup, the plane and the ball. The initial image of the scene is shown on the left, and the final reconstructed mesh on the right. We found agreement between the estimated height of the cup from the 3D reconstruction, 11.04 ± 0.09 cm, and the measured height (obtained using a ruler), 10.95 ± 0.05 cm. Once again the right portion on the reconstructed scene is noisier than the left portion. This was expected since the light source was, once again, standing to the left of the camera. Geometrical parameters: $d_d = 22.6$ cm, $\theta = 38.2$ degrees, $h_S = 43.2$ cm, $\xi = 155.9$ degrees, and $\phi = 69$ degrees.

Geometry of the corner: We fit 2 planes to the corner structure, one corresponding to the top surface (the horizontal plane) and the other one to the frontal surface (vertical plane). We estimated the surface noise of the top surface to 0.125 mm, and that of the frontal face to 0.8 mm (almost 7 times larger). This noise difference between the two planes can be observed on figure 6. Once again, after fitting quadratic patches to the two planar portions, we did not notice any significant global geometric distortion in the scene (from planar to quadratic warping, the residual noise decreased by only 5% in standard deviation). From the reconstruction, we estimated the height H and width D of the right angle structure, as well as the angle ψ between the two reconstructed planes, and compared them to their true values:

Parameters	Estimates	True values	Relative errors
H (cm)	2.57 ± 0.02	2.65 ± 0.02	3%
D (cm)	3.06 ± 0.02	3.02 ± 0.02	1.3%
ψ (degrees)	86.21	90	1%

The overall reconstructed structure does not have any major noticeable global deformation (it seems that the calibration process gives good enough estimates). The most noticeable source of errors is the surface noise due to local image processing. A figure of merit to keep in mind is a surface noise between 0.1 mm (for planes roughly parallel to the desk) and 0.8 mm (for frontal plane in the right corner). In most portions of the scene, the errors are of the order of 0.3 mm, i.e. less than 1%. Notice that these figures may very well vary from experiment to experiment, especially depending on how fast the scanning is performed. In all the presented experiments, we kept the speed of the shadow approximately uniform.

Figures 7 and 8 report the reconstruction results achieved on two other scenes.

5 Conclusion and future work

We have presented a simple, low cost system for extracting surface shape of objects. The method requires very little processing and image storage so that it can be implemented in real time. The accuracies we obtained on the final reconstructions are reasonable (at most 1% or 0.5 mm noise error) considering the little hardware requirement. In addition, the final outcome is a dense coverage of the surface (one point in space for each pixel in the image) allowing for direct texture mapping.

An error analysis was presented together with the description of a simple technique for merging multiple 3D scans together in order to (a) obtain a better coverage of the scene, and (b) reduce the estimation noise. The overall calibration procedure, even in the case of multiple scans, is very intuitive, simple, and sufficiently accurate.

Another advantage of our approach is that it easily scales to larger scenarios indoors – using more powerful lamps like photo-floods – and outdoors where the sun may be used as a calibrated light source (given latitude, longitude, and time of day). These are experiments that we wish to carry out in the future.

Other extensions of this work relate to multiple view integration. We wish to extend the alignment technique to a method allowing the user to move freely the object in front of the camera and the lamp between scans in order to achieve a full coverage. That is necessary to construct complete 3D models.

It is also part of future work to incorporate a geometrical model of extended light source to the shadow edge detection process, in addition to developing an uncalibrated (or projective) version of the method.

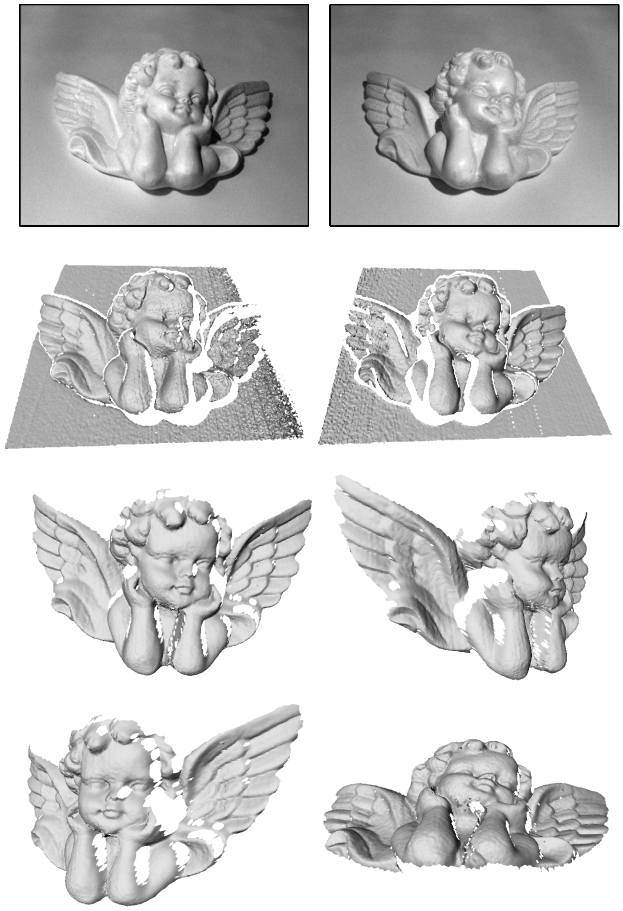


Figure 8: **Experiment 3 - The angel scene:** We took two scans of the angel with the lamp first on the left side (top-left) and then on the right side (top-right) of the camera. The two resulting meshes are shown on the second row, left and right. As expected, the portions further away from the light source are noisier. The two meshes were then merged together following the technique described in section 3, with diffusion coefficient $\beta = 15$. Four different views of the final mesh (47076 triangles) are presented. Notice the small surface noise: we estimated it to 0.09 mm throughout the entire reconstructed surface. Over a depth variation of approximately 10 cm, this means a relative error of 0.1%. The few white holes correspond to the occluded portions of the scene (not observed from the camera or not illuminated). Most of the geometrical constants in the setup were kept roughly identical in both scans: $d_d = 22$ cm, $\theta = 40$ degrees, $h_S = 62$ cm, $\phi \approx 70$ degrees; we only changed the azimuth angle ξ from π (lamp on the left) to 0 (lamp on the right). In this experiment we took the lamp reflector off, leaving the bulb naked. Consequently, we noticed a significant improvement in the sharpness of the projected shadow compared to the two first experiments. We believe that this operation was the main reason for the noticeable improvement in reconstruction quality. Once again, there was no significant global deformation in the final structured surface: we fit a quadratic model through the reconstructed set of points on the desk plane and noticed from planar to quadratic warping a decrease of only 2% on the standard deviation of surface noise.

Acknowledgments

This work is supported in part by the California Institute of Technology; an NSF National Young Investigator Award to P.P.; the Center for Neuromorphic Systems Engineering funded by the National Science Foundation at the California Institute of Technology; and by the California Trade and Commerce Agency, Office of Strategic Technology. We wish to thank all the colleagues that helped us throughout this work, especially Luis Goncalves, George Barbastathis, Mario Munich, and Arrigo Benedetti for very useful discussions. Comments from the anonymous reviewers were very helpful in improving a previous version of the paper.

References

[1] Paul Besl, *Advances in Machine Vision*, chapter 1 - Active optical range imaging sensors, pages 1–63, Springer-Verlag, 1989.

[2] P.J. Besl and N.D. McKay, "A method for registration of 3-d shapes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[3] Jean-Yves Bouguet and Pietro Perona, "3D Photography on your Desk", Technical report, California Institute of Technology, 1997, available at: http://www.vision.caltech.edu/bouguetj/ICCV98.

[4] Brian Curless and Marc Levoy, "Better optical triangulation through spacetime analysis", *Proc. 5^{th} Int. Conf. Computer Vision, Boston, USA*, pages 987–993, 1995.

[5] Brian Curless and Marc Levoy, "A volumetric method for building complex models from range images", *SIGGRAPH96, Computer Graphics Proceedings*, 1996.

[6] O.D. Faugeras, *Three dimensional vision, a geometric viewpoint*, MIT Press, 1993.

[7] Berthold K.P. Horn, "Closed-form solution of absolute orientation using unit quaternions", *J. Opt. Soc. Am. A*, 4(4):629–642, 1987.

[8] Jurgen R. Meyer-Arendt, "Radiometry and photometry: Units and conversion factors", *Applied Optics*, 7(10):2081–2084, October 1968.

[9] Shree K. Nayar, Katsushi Ikeuchi, and Takeo Kanade, "Shape from interreflections", *Int. J. of Computer Vision*, 6(3):173–195, 1991.

[10] Athanasios Papoulis, *Probability, Random Variables and Stochastic Processes*, Mac Graw Hill, 1991, Third Edition, page 187.

[11] A.J. Stoddart and A. Hilton, "Registration of multiple point sets", *Proceedings of the 13th Int. Conf. of Pattern Recognition*, 1996.

[12] Marjan Trobina, "Error model of a coded-light range sensor", Technical Report BIWI-TR-164, ETH-Zentrum, 1995.

[13] R.Y. Tsai, "A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses", *IEEE J. Robotics Automat.*, RA-3(4):323–344, 1987.

[14] John W. T. Walsh, *Photometry*, Dover, NY, 1965.

[15] Y.F. Wang, "Characterizing three-dimensional surface structures from visual images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):52–60, 1991.

[16] Z. Yang and Y.F. Wang, "Error analysis of 3D shape construction from structured lighting", *Pattern Recognition*, 29(2):189–206, 1996.

1999 Reprinted with permission of Kluwer Academic Publishers,
from the International Journal of Computer Vision - Vol. 35, No. 2, Nov./Dec. 1999 - pp. 129-149

3D photography using shadows in dual-space geometry

Jean-Yves Bouguet† and Pietro Perona†‡

† California Institute of Technology, 136-93, Pasadena, CA 91125, USA
‡ Università di Padova, Italy
{bouguetj,perona}@vision.caltech.edu

Abstract

A simple and inexpensive approach for extracting the three-dimensional shape of objects is presented. It is based on 'weak structured lighting'. It requires very little hardware besides the camera: a light source (a desk-lamp or the sun), a stick and a checkerboard. The object, illuminated by the light source, is placed on a stage composed of a ground plane and a back plane; the camera faces the object. The user moves the stick in front of the light source, casting a moving shadow on the scene. The 3D shape of the object is extracted from the spatial and temporal location of the observed shadow. Experimental results are presented on five different scenes (indoor with a desk lamp and outdoor with the sun) demonstrating that the error in reconstructing the surface is less than 0.5% of the size of the object. A mathematical formalism is proposed that simplifies the notation and keep the algebra compact. A real-time implementation of the system is also presented.

1 Introduction and motivation

One of the most valuable functions of our visual system is informing us about the shape of the objects that surround us. Manipulation, recognition, and navigation are amongst the tasks that we can better accomplish by seeing shape. Ever-faster computers, progress in computer graphics, and the widespread expansion of the Internet have recently generated interest in imaging both the geometry and surface texture of objects. The applications are numerous. Perhaps the most important ones are animation and entertainment, industrial design, archiving, virtual visits to museums, and commercial on-line catalogues.

In designing a system for recovering shape, different engineering tradeoffs are proposed by each application. The main parameters to be considered are cost, accuracy, ease of use and speed of acquisition. So far the commercial 3D scanners (e.g. the Cyberware scanner) have emphasized accuracy over the other parameters. Active illumination systems are popular in industrial applications where a fixed installation with controlled lighting is possible. These systems use motorized transport of the object and active (laser, LCD projector) lighting of the scene which makes them very accurate, but unfortunately expensive [2, 23, 26, 38, 43]. Furthermore most active systems fail under bright outdoor scenes except those based upon synchronized scanning. One such system has been presented by Riou in [33].

An interesting challenge for vision scientists is to take the opposite point of view: emphasize low cost and simplicity and design 3D scanners that demand little more hardware than a PC and a video camera by making better use of the data that is available in the images.

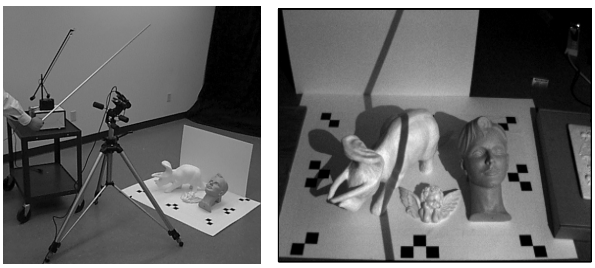


Figure 1: **The general setup of the proposed method:** The camera is facing the scene illuminated by the light source (top-left). The figure illustrates an indoor scenario when a desk lamp (without reflector) is used as light source. In outdoor the lamp is substituted by the sun. The objects to scan are positioned on the ground floor (horizontal plane), in front of a background plane. When an operator freely moves a stick in front of the light, a shadow is cast on the scene. The camera acquires a sequence of images $I(x, y, t)$ as the operator moves the stick so that the shadow scans the entire scene. A sample image is shown on the top right figure. This constitutes the input data to the 3D reconstruction system. The three dimensional shape of the scene is reconstructed using the spatial and temporal properties of the shadow boundary throughout the input sequence.

A number of passive cues have long been known to contain information on 3D shape: stereoscopic disparity, texture, motion parallax, (de)focus, shadows, shading and specularities, occluding contours and other surface discontinuities. At the current state of vision research stereoscopic disparity is the single passive cue that reliably gives reasonable accuracy. Unfortunately it has two major drawbacks: it requires two cameras thus increasing complexity and cost, and it cannot be used on untextured surfaces, which are common for industrially manufactured objects.

We propose a method for capturing 3D surfaces that is based on what we call 'weak structured lighting.' It yields good accuracy and requires minimal equipment besides a computer and a camera: a stick, a checkerboard, and a point light source. The light source may be a desk lamp for indoor scenes, and the sun for outdoor scenes. A human operator, acting as a low precision motor, is also required.

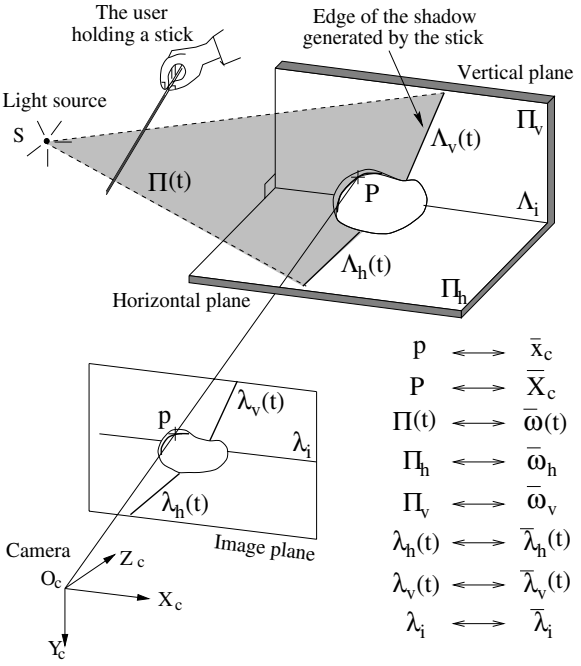


Figure 2: **Geometrical principle of the method**

We start with the description of the scanning method in Sec. 2, followed in Sec. 3 by a number of experiments that assess the convenience and accuracy of the system in indoor as well as outdoor scenarios. We end with a discussion and conclusions in Sec. 4. In addition, we show that expressing the problem in dual-space geometry [12] enables to explore and compute geometrical properties of three dimensional scenes with simple and compact notation. This formalism is discussed in the appendix together with a complete error analysis of the method.

2 Description of the method

The general principle consists of casting a moving shadow with a stick onto the scene, and estimating the three dimensional shape of the scene from the sequence of images of the deformed shadow. Figure 1 shows a typical setup. The objective is to extract scene depth at every pixel in the image. The point light source and the leading edge of the stick define, at every time instant, a plane; therefore, the boundary of the shadow that is cast by the stick on the scene is the intersection of this plane with the surface of the object. We exploit this geometrical insight for reconstructing the 3D shape of the object. Figure 2 illustrates the geometrical principle of the method. Approximate the light source with a point S, and denote by Π_h the horizontal plane (ground) and Π_v a vertical plane orthogonal to Π_h. Assume that the position of the plane Π_h in the camera reference frame is known from calibration (sec. 2.1). We infer the location of Π_v from the projection λ_i (visible in the image) of the intersection line Λ_i between Π_h and Π_v (sec. 2.2). The goal is to estimate the 3D location of the point P in space corresponding to every pixel p (of coordinates $\overline{x}_c$) in the image. Call t the time when the shadow boundary passes by a given pixel $\overline{x}_c$ (later referred to as the *shadow time*). Denote by $\Pi(t)$ the corresponding shadow plane at that time t. Assume that two portions of the shadow projected on the two planes Π_h and Π_v are visible on the image: $\lambda_h(t)$ and $\lambda_v(t)$. After extracting these two lines, we deduce the location in space of the two corresponding lines $\Lambda_h(t)$ and $\Lambda_v(t)$ by intersecting the planes $(O_c, \lambda_h(t))$ and $(O_c, \lambda_v(t))$ with Π_h and Π_v respectively. The shadow plane $\Pi(t)$ is then the plane defined by the two non-collinear lines $\Lambda_h(t)$ and $\Lambda_v(t)$ (sec. 2.5). Finally, the point P corresponding to $\overline{x}_c$ is retrieved by intersecting $\Pi(t)$ with the optical ray (O_c, p). This final stage is called triangulation (sec. 2.6). Notice that the key steps are: (a) estimate the shadow time $t_s(\overline{x}_c)$ at every pixel $\overline{x}_c$ (*temporal processing*), (b) locate the two reference lines $\lambda_h(t)$ and $\lambda_v(t)$ at every time instant t (*spatial processing*), (c) calculate the shadow plane, and (d) triangulate and calculate depth. These tasks are described in sections 2.4, 2.5 and 2.6.

Goshtasby *et al.* [22] also designed a range scanner using a shadow generated by a fine wire in order to reconstruct the shape of dental casts. In their system, the wire was motorized and its position calibrated.

Notice that if the light source is at a known location in space, then the shadow plane $\Pi(t)$ may be directly inferred from the point S and the line $\Lambda_h(t)$. Consequently, in such cases, the additional plane $\Pi_v(t)$ is not required. We describe here two versions of the setup: one containing two calibrated planes and an uncalibrated (possibly moving) light source; the second containing one calibrated plane and a calibrated light source.

2.1 Camera calibration

The goal of calibration is to recover the location of the ground plane Π_h and the *intrinsic* camera parameters (focal length, principal point and radial distortion factor). The procedure consists of first placing a planar checkerboard pattern on the ground in the location of the objects to scan (see figure 3-left). From the image captured by the camera (figure 3-right), we infer the intrinsic and extrinsic parameters of the camera, by matching the projections onto the image plane of the known grid corners with the expected projection directly measured on the image (extracted corners of the grid); the method is proposed by Tsai in [39]. We use a first order symmetric radial distortion model for the lens, as proposed in [11, 39, 25]. When using a single image of a planar calibration rig, the principal point (i.e. the intersection of the optical axis with the image plane) cannot be recovered [25, 37]. In that case it is assumed to be identical to the image center. In order to fit a full camera model (principal

point included), we propose to extend that approach by integrating multiple images of the planar grid positioned at different locations in space (with different orientations). This method has been suggested, studied and demonstrated by Sturm and Maybank in [37]. Theoretically, a minimum of two images is required to recover two focals (along x and y), the principal point coordinates, and the lens distortion factor. We recommend to use that method with three or four images for best accuracies on the intrinsic parameters [37]. In our experience, in order to achieve good 3D reconstruction accuracies, it is sufficient to use the simple approach with a single calibration image without estimating the camera principal point. In other words, the quality of reconstruction is quite insensitive to errors on the principal point position. A whole body of work supporting that observation may be found in the literature. We especially advise the reader most interested in issues on sensitivity of 3D Euclidean reconstruction results with respect to intrinsic calibration errors, to refer to recent publications on self-calibration, such as Bougnoux [5] or Pollefeys *et al.* [28, 31, 32].

For more general insights on calibration techniques, we refer the reader to the work of Faugeras [19] and others [10, 11, 14, 18, 36, 42]. A 3D rig should be used for achieving maximum accuracy.

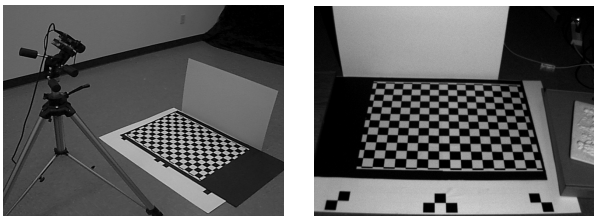


Figure 3: **Camera calibration**

2.2 Vertical plane localization Π_v

Call $\overline{\omega}_h$ and $\overline{\omega}_v$ respectively the coordinate vectors of Π_h and Π_v (refer to figure 2 and Appendix A for notation). After calibration, $\overline{\omega}_h$ is known. The two planes Π_h and Π_v intersect along the line Λ_i observed on the image plane at λ_i. Therefore, according to proposition 1 in Appendix A, $\overline{\omega}_h - \overline{\omega}_v$ is parallel to $\overline{\lambda}_i$, coordinate vector of λ_i, or equivalently, there exists a scalar α such that $\overline{\omega}_v = \overline{\omega}_h + \alpha\overline{\lambda}_i$. Since the two planes Π_h and Π_v are by construction orthogonal, we have $\langle \overline{\omega}_h, \overline{\omega}_v \rangle = 0$. That leads to the closed-form expression for calculating $\overline{\omega}_v$:

$$\overline{\omega}_v = \overline{\omega}_h - \frac{\langle \overline{\omega}_h, \overline{\omega}_h \rangle}{\langle \overline{\lambda}_i, \overline{\omega}_h \rangle} \overline{\lambda}_i.$$

Notice that in every realistic scenario, the two planes Π_h and Π_v do not contain the camera center O_c. Their coordinate vectors $\overline{\omega}_h$ and $\overline{\omega}_v$ in dual-space are therefore always well defined (see Appendix A and sections 2.6 and 2.7 for further discussions).

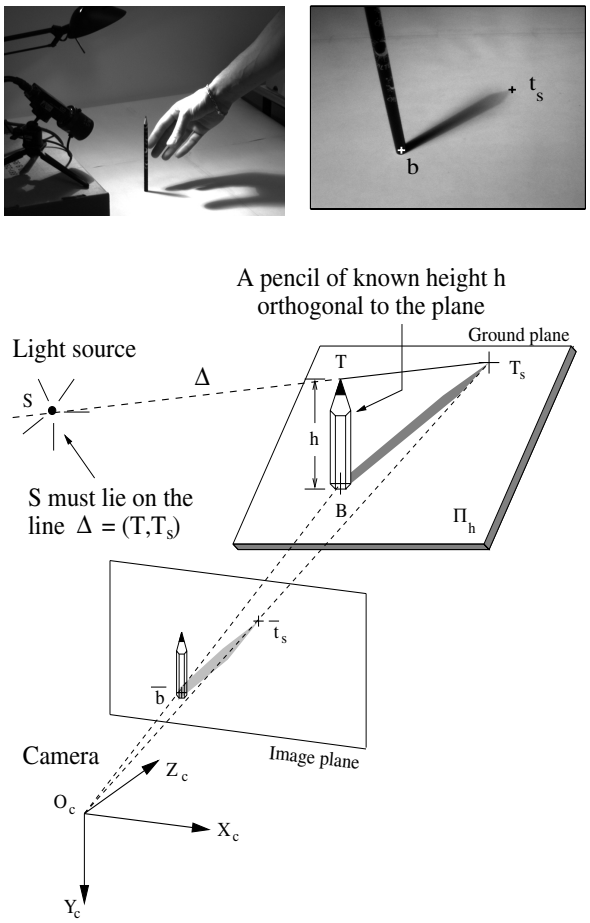


Figure 4: **Light source calibration**

2.3 Light source calibration

When using a single reference plane for scanning (for example Π_h without Π_v), it is required to know the location of the light source in order to infer the shadow plane location $\Pi(t)$ (see section 2.5 for details). Figure 4 illustrates a simple technique for calibrating the light source that requires minimal extra equipment: a pencil of known length. The operator stands a pencil on the reference plane Π_h (see fig. 4-top-left). The camera observes the shadow of the pencil projected on the ground plane. The acquired image is shown on figure 4-top-right. From the two points $\overline{b}$ and $\overline{t}_s$ on this image, one can infer the positions in space of B and T_s, respectively the base of the pencil, and the tip of the pencil shadow (see bottom figure). This is done by intersecting the optical rays $(O_c, \overline{b})$ and $(O_c, \overline{t}_s)$ with Π_h (known from camera calibration). In addition, given that the height of the pencil h is known, the coordinates of its tip T can be directly inferred from B. The point light source S has to lie on the line $\Delta = (T, T_s)$ in space. This yields one linear constraint on the light source position. By taking a second view, with the pencil at a

different location on the plane, one retrieves a second independent constraint with another line Δ'. A closed form solution for the 3D coordinate of S is then derived by intersecting the two lines Δ and Δ' (in the least squares sense). Notice that since the problem is linear, one can integrate the information from more than 2 views and make the estimation more accurate. If $N > 2$ images are used, one can obtain a closed form solution for the closest point $\tilde{S}$ to the N inferred lines (in the least squares sense). We also estimate the uncertainty on that estimate from the distance of $\tilde{S}$ to each one of the Δ lines. That indicates how consistently the lines intersect a single point in space. Refer to [7, 8, 6] for the complete derivations.

2.4 Spatial and temporal shadow edge localization

A fundamental stage of the method is the detection of the lines of intersection of the shadow plane $\Pi(t)$ with the two planes Π_h and Π_v; a simple approach to extract $\overline{\lambda}_h(t)$ and $\overline{\lambda}_v(t)$ may be used if we make sure that a number of pixel rows at the top and bottom of the image are free from objects. Then the two tasks to accomplish are: **(a)** Localize the edges of the shadow that are directly projected on the two orthogonal planes $\lambda_h(t)$ and $\lambda_v(t)$ at every discrete time t (every frame), leading to the set of all shadow planes $\Pi(t)$ (see sec. 2.5), **(b)** Estimate the time $t_s(\overline{x}_c)$ (*shadow time*) where the edge of the shadow passes through any given pixel $\overline{x}_c = (x_c, y_c)$ in the image (see figure 5). Curless and Levoy [16] demonstrated that such a spatio-temporal approach is appropriate for preserving sharp discontinuities in the scene as well as reducing range distortions. A similar temporal processing for range sensing was used by Gruss, Tada and Kanade in [23, 27].

Both processing tasks correspond to finding the edge of the shadow, but the search domains are different: one operates on the spatial coordinates (image coordinates) and the other one on the temporal coordinate. Although independent in appearance, the two search procedures need to be compatible: if at time t_0 the shadow edge passes through pixel $\overline{x}_c = (x_c, y_c)$, the two searches should find the exact same point (x_c, y_c, t_0) (in space/time). One could observe that this property does not hold for all techniques. One example is the image gradient approach for edge detection (e.g. Canny edge detector [13]). Indeed, the maximum spatial gradient point does not necessarily match with the maximum temporal gradient point (which is function of the scanning speed). In addition, the spatial gradient is a function both of changes in illumination due to the shadow and changes in albedo and changes in surface orientation. Furthermore, it is preferable to avoid any spatial filtering on the images (e.g. smoothing) which would produce blending in the final depth estimates, especially noticeable at

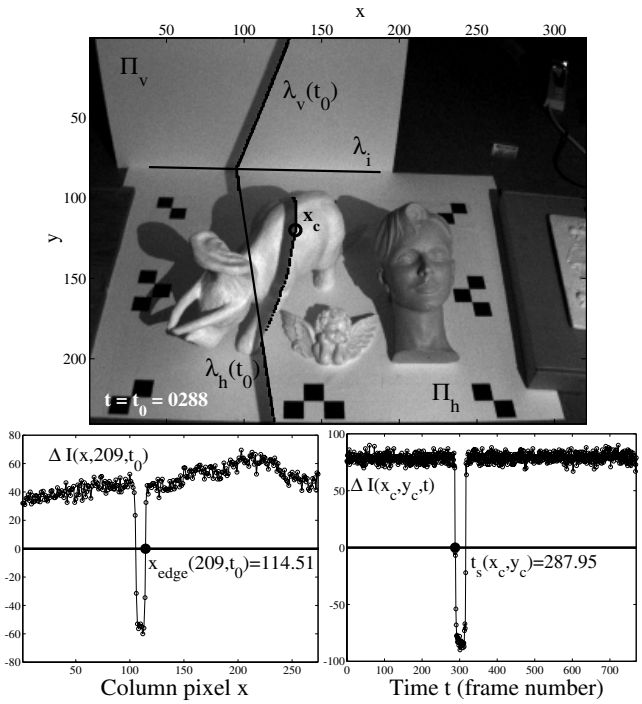


Figure 5: **Spatial and temporal shadow localization**

occlusions and surface discontinuities (corners for example). These observations were also addressed by Curless and Levoy in [16].

It is therefore necessary to use a unique criterion that would equally describe shadow edges in space (image coordinates) and time and is insensitive to changes in surface albedo and surface orientation. The simple technique we propose here that satisfies that property is spatio-temporal thresholding. This is based on the following observation: as the shadow is scanned across the scene, each pixel (x, y) sees its brightness intensity going from an initial maximum value $I_{\max}(x, y)$ (when there is no shadow yet) down to a minimum value $I_{\min}(x, y)$ (when the pixel is within the shadow) and then back up to its initial value as the shadow goes away. This profile is characteristic even when there is a fair amount of internal reflections in the scene [29, 41].

For any given pixel $\overline{x}_c = (x, y)$, define $I_{\min}(x, y)$ and $I_{\max}(x, y)$ as its minimum and maximum brightness throughout the entire sequence:

$$\begin{cases} I_{\min}(x,y) & \doteq & \min\limits_t \{I(x,y,t)\} \\ I_{\max}(x,y) & \doteq & \max\limits_t \{I(x,y,t)\} \end{cases} \quad (1)$$

We define the shadow edge to be the locations (in space-time) where the image $I(x, y, t)$ intersects with the threshold image $I_{\text{shadow}}(x, y)$ defined as the mean value between $I_{\max}(x, y)$ and $I_{\min}(x, y)$:

$$I_{\text{shadow}}(x,y) \doteq \frac{1}{2}\left(I_{\max}(x,y) + I_{\min}(x,y)\right) \quad (2)$$

This may be also regarded as the zero crossings of the difference image $\Delta I(x, y, t)$ defined as follows:

$$\Delta I(x, y, t) \doteq I(x, y, t) - I_{\text{shadow}}(x, y) \qquad (3)$$

The two bottom plots of fig. 5 illustrate shadow edge detection in the spatial domain (to calculate $\lambda_h(t)$ and $\lambda_v(t)$) and in the temporal domain (to calculate $t_s(\overline{x}_c)$). The bottom-left plot shows the profile of $\Delta I(x, y, t)$ along row $y = 209$ at time $t = t_0 = 288$ versus the column pixel coordinate x. The second zero crossing of that profile corresponds to one point $\overline{x}_{\text{edge}}(t_0) = (114.51, 209)$ belonging to $\lambda_h(t_0)$, the right edge of the shadow (computed at subpixel accuracy by linear interpolation). Identical processing is applied on 39 other rows for $\lambda_h(t_0)$ and 70 rows for $\lambda_v(t_0)$ in order to retrieve the two edges (by least square line fitting across the two sets of points on the image). Similarly, the bottom-right figure shows the temporal profile $\Delta I(x_c, y_c, t)$ at the pixel $\overline{x}_c = (x_c, y_c) = (133, 120)$ versus time t (or frame number). The shadow time at that pixel is defined as the first zero crossing location of that profile: $t_s(133, 120) = 287.95$ (computed at sub-frame accuracy by linear interpolation). Notice that the right edge of the shadow corresponds to the front edge of the temporal profile, because the shadow was scanned from left to right in all experiments. Intuitively, pixels corresponding to occluded regions in the scene do not provide any relevant depth information. Therefore, we only process pixels with contrast value $I_{\text{contrast}}(x, y) \doteq I_{\max}(x, y) - I_{\min}(x, y)$ larger than a pre-defined threshold I_{thresh}. This threshold was 30 in all experiments reported in this paper (recall that the intensity values are encoded from 0 for black to 255 for white). This threshold should be proportional to the level of noise in the image.

Due to the limited dynamic range of the camera, it is clear that one should avoid saturating the sensor, and that one would expect poor accuracy in areas of the scene that reflect little light towards the camera due to their orientation with respect to the light source and/or low albedo. Our experiments were designed to test the extent of this problem.

2.5 Shadow plane estimation $\Pi(t)$

Denote by $\overline{\omega}(t)$, $\overline{\lambda}_h(t)$ and $\overline{\lambda}_v(t)$ the coordinate vectors of the shadow plane $\Pi(t)$ and of the shadow edges $\lambda_h(t)$ and $\lambda_v(t)$ at time t. Since $\lambda_h(t)$ is the projection of the line of intersection $\Lambda_h(t)$ between $\Pi(t)$ and Π_h, then $\overline{\omega}(t)$ lies on the line passing through $\overline{\omega}_h$ with direction $\overline{\lambda}_h(t)$ in dual-space (from Appendix A). That line, denoted $\hat{\Lambda}_h(t)$, is the dual image of $\Lambda_h(t)$ in dual-space (see Appendix A). Similarly, $\overline{\omega}(t)$ lies on the line $\hat{\Lambda}_v(t)$ passing through $\overline{\omega}_v$ with direction $\overline{\lambda}_v(t)$ (dual image of $\Lambda_v(t)$). Therefore, in dual-space, the coordinate vector of the shadow plane $\overline{\omega}(t)$ is at the intersection between the two known lines $\hat{\Lambda}_h(t)$ and $\hat{\Lambda}_v(t)$. In the presence of noise these two lines will not exactly intersect (equivalently, the 3 lines λ_i, $\lambda_h(t)$ and $\lambda_v(t)$ do not necessarily intersect at one point on the image plane, or their coordinate vectors $\overline{\lambda}_i$, $\overline{\lambda}_h(t)$ and $\overline{\lambda}_v(t)$ are not coplanar). However, one may still identify $\overline{\omega}(t)$ with the point that is the closest to the lines in the least-squares sense. The solution to that problem reduces to:

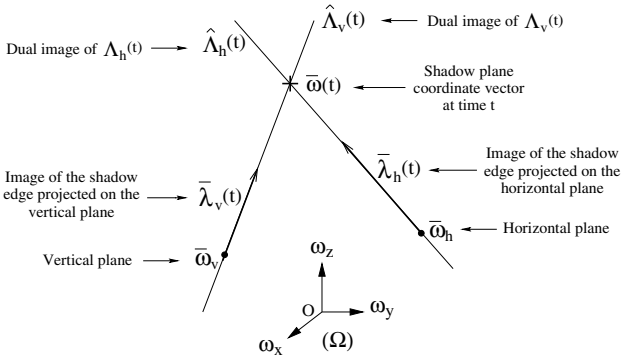


Figure 6: **Shadow plane estimation using two planes:** The coordinate vector of the shadow plane $\overline{\omega}(t)$ is the intersection point of the two dual lines $\hat{\Lambda}_h(t)$ and $\hat{\Lambda}_v(t)$ in dual-space (Ω). In presence of noise, the two lines do not intersect. The vector $\overline{\omega}(t)$ is then the best intersection point between the two lines (in the least squares sense).

$$\overline{\omega}(t) = \frac{1}{2}\left(\overline{\omega}_1(t) + \overline{\omega}_2(t)\right), \qquad (4)$$

with

$$\begin{array}{rcl} \overline{\omega}_1(t) & \doteq & \overline{\omega}_h + \alpha_h \overline{\lambda}_h(t) \\ \overline{\omega}_2(t) & \doteq & \overline{\omega}_v + \alpha_v \overline{\lambda}_v(t) \end{array} \qquad (5)$$

if $[\alpha_h \quad \alpha_v]^T = \mathbf{A}^{-1}\mathbf{b}$, where $\mathbf{A}$ and $\mathbf{b}$ are defined as follows (for clarity, the variable t is omitted):

$$\mathbf{A} \doteq \begin{bmatrix} \langle \overline{\lambda}_h, \overline{\lambda}_h \rangle & -\langle \overline{\lambda}_h, \overline{\lambda}_v \rangle \\ -\langle \overline{\lambda}_h, \overline{\lambda}_v \rangle & \langle \overline{\lambda}_v, \overline{\lambda}_v \rangle \end{bmatrix}, \quad \mathbf{b} \doteq \begin{bmatrix} \langle \overline{\lambda}_h, \overline{\omega}_v - \overline{\omega}_h \rangle \\ \langle \overline{\lambda}_v, \overline{\omega}_h - \overline{\omega}_v \rangle \end{bmatrix}$$

Note that the two vectors $\overline{\omega}_1(t)$ and $\overline{\omega}_2(t)$ are the orthogonal projections, in dual-space, of $\overline{\omega}(t)$ onto $\hat{\Lambda}_h(t)$ and $\hat{\Lambda}_v(t)$ respectively. The norm of the difference between these two vectors may be used as an estimate of the error in recovering $\Pi(t)$. If the two edges $\lambda_h(t)$ and $\lambda_v(t)$ are estimated with different reliabilities, a weighted least squares method may still be used.

Figure 6 illustrates the principle of shadow plane estimation in dual-space when using the two edges $\lambda_h(t)$ and $\lambda_v(t)$. This reconstruction method was used in experiments 1, 4 and 5.

Notice that the additional vertical plane Π_v enables us to extract the shadow plane location without requiring the knowledge of the light source position.

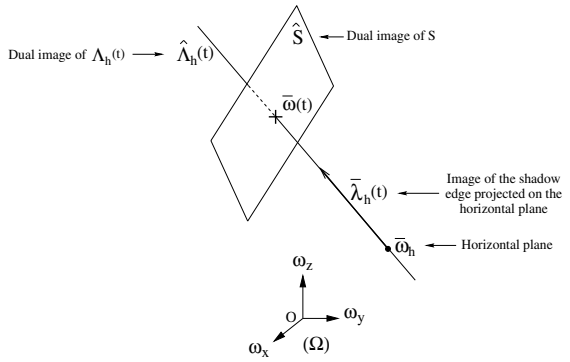


Figure 7: **Shadow plane estimation using one plane and the light source position:** In dual-space, the coordinate vector of the shadow plane $\overline{\omega}(t)$ is the intersection point of the line $\hat{\Lambda}_h(t)$ and the plane $\hat{S}$, dual image of the point light source S. This method requires the knowledge of the light source position. A light source calibration method is presented in section 2.3.

Consequently, the light source is allowed to move during the scan (this may be the case of the sun, for example).

When the light source is of fixed and known location in space, the plane Π_v is not required. Then, one may directly infer the shadow plane position from the line $\lambda_h(t)$ and from the light source position S:

$$\overline{\omega}(t) = \overline{\omega}_h + \alpha_h \overline{\lambda}_h(t) \tag{6}$$

where

$$S \in \Pi(t) \quad \Leftrightarrow \quad \langle \overline{\omega}(t), \overline{X}_S \rangle = 1 \quad \Leftrightarrow \quad \alpha_h = \frac{1 - \langle \overline{\omega}_h, \overline{X}_S \rangle}{\langle \overline{\lambda}_h(t), \overline{X}_S \rangle}$$

where $\overline{X}_S = [X_S \quad Y_S \quad Y_S]^T$ is the coordinate vector of the light source S in the camera reference frame. In dual-space geometry, this corresponds to intersecting the line $\hat{\Lambda}_h(t)$ with the plane $\hat{S}$, dual image of the source point S. This process is illustrated in figure 7. Notice that $\langle \overline{\lambda}_h(t), \overline{X}_S \rangle = 0$ corresponds to the case where the shadow plane contains the camera center of projection O_c. This is singular configuration that makes the triangulation fail ($\|\overline{\omega}(t)\| \to \infty$). This approach requires an additional step of estimating the position of S. Section 2.3 describes a simple method for light source calibration. This reconstruction method was used in experiments 2 and 3.

It is shown in Appendix B that $1 - \langle \overline{\omega}_h, \overline{X}_S \rangle = h_S / d_h$ where h_S and d_h are the orthogonal distances of the light source S and the camera center O_c to the plane Π_h (see figure 8). Then, the constant α_h may be written as:

$$\alpha_h = \frac{h_S/d_h}{\langle \overline{\lambda}_h(t), \overline{X}_S \rangle} = \frac{1/d_h}{\langle \overline{\lambda}_h(t), \overline{X}_S/h_S \rangle} \tag{7}$$

This expression highlights the fact that the algebra naturally generalizes to cases where the light source is located at infinity (and calibrated). Indeed, in those cases, the ratio $\overline{X}_S/h_S$ reduces to $\overline{d}_S/\sin\phi$ where $\overline{d}_S$ is the normalized light source direction vector (in the camera reference frame) and ϕ the elevation angle of the light source with respect to the plane Π_h (defined on figure 8). In dual-space, the construction of the shadow plane vector $\overline{\omega}(t)$ remains the same: it is still at the intersection of $\hat{\Lambda}_h(t)$ with $\hat{S}$. The only difference is that the dual image $\hat{S}$ is a plane crossing the origin in dual-space. The surface normal of that plane is simply the vector $\overline{d}_S$.

2.6 Triangulation

Once the shadow time $t_s(\overline{x}_c)$ is estimated at a given pixel $\overline{x}_c = [x_c \quad y_c \quad 1]^T$ (in homogeneous coordinates), one can identify the corresponding shadow plane $\Pi(t_s(\overline{x}_c))$ (with coordinate vector $\overline{\omega}_c \doteq \overline{\omega}(t_s(\overline{x}_c))$). Then, the point P in space associated to $\overline{x}_c$ is retrieved by intersecting $\Pi(t_s(\overline{x}_c))$ with the optical ray $(O_c, \overline{x}_c)$ (see figure 2):

$$Z_c = \frac{1}{\langle \overline{\omega}_c, \overline{x}_c \rangle} \quad \Longrightarrow \quad \overline{X}_c = Z_c \, \overline{x}_c = \frac{\overline{x}_c}{\langle \overline{\omega}_c, \overline{x}_c \rangle}, \tag{8}$$

if $\overline{X}_c = [X_c \quad Y_c \quad Z_c]^T$ is defined as the coordinate vector of P in the camera reference frame.

Notice that the shadow time $t_s(\overline{x}_c)$ acts as an index to the shadow plane list $\Pi(t)$. Since $t_s(\overline{x}_c)$ is estimated at sub-frame accuracy, the plane $\Pi(t_s(\overline{x}_c))$ (actually its coordinate vector $\overline{\omega}_c$) results from linear interpolation between the two planes $\Pi(t_0 - 1)$ and $\Pi(t_0)$ if $t_0 - 1 < t_s(\overline{x}_c) < t_0$ and t_0 integer:

$$\overline{\omega}_c = \Delta t \, \overline{\omega}(t_0 - 1) + (1 - \Delta t) \, \overline{\omega}(t_0),$$

where $\Delta t = t_0 - t_s(\overline{x}_c)$, $0 \leq \Delta t < 1$ (see figure 17).

Once the range data are recovered, a mesh is generated by connecting neighboring points in triangles. The connectivity is directly given by the image: two vertices are neighbors if their corresponding pixels are neighbors in the image. In addition, since every vertex corresponds to a unique pixel, texture mapping is also a straightforward task. Figures 9, 11, 12, 13 and 14 show experimental results.

Similarly to stereoscopic vision, when the baseline becomes shorter, as the shadow plane moves closer to the camera center triangulation becomes more and more sensitive to noise. In the limit, if the plane crosses the origin (or equivalently $\|\overline{\omega}_c\| \to \infty$) triangulation becomes impossible. This is why it is not a big loss that we cannot represent planes going through the origin with our parameterization. This observation will appear again in the next section on error analysis.

2.7 Design Issues - Error analysis

When designing the scanning system, it is important to choose a spatial configuration of the camera and the light source that maximizes the overall quality of reconstruction of the scene.

The analysis conducted in Appendix C leads to an expression for the variance $\sigma^2_{Z_c}$ of the error of the depth estimate Z_c of a point P belonging to the scene (equation 18):

$$\sigma^2_{Z_c} = Z_c^4 \left(\frac{\omega_x \cos\varphi + \omega_y \sin\varphi}{f_c \, \|\overline{\nabla} I(\overline{x}_c)\|} \right)^2 \sigma_I^2 \qquad (9)$$

where $\overline{x}_c$ is the coordinate vector of the projection p of P on the image plane, $\overline{\omega}_c = [\omega_x \quad \omega_y \quad \omega_z]^T$ is the shadow plane vector at time $t = t_s(\overline{x}_c)$, $\overline{\nabla} I(\overline{x}_c) = [I_x(\overline{x}_c) \quad I_y(\overline{x}_c)]^T = \|\overline{\nabla} I(\overline{x}_c)\| \, [\cos\varphi \quad \sin\varphi]^T$ is the spatial gradient vector of the image brightness at the shadow edge at $\overline{x}_c$ at time $t = t_s(\overline{x}_c)$ (in units of brightness per pixel), σ_I is the standard deviation of the image brightness noise (in units of brightness), and f_c is the camera focal length (in pixels).

Three observations may be drawn from equation 9. First, since $\sigma^2_{Z_c}$ is inversely proportional to $\|\overline{\nabla} I(\overline{x}_c)\|^2$, the reconstruction accuracy increases with the sharpness of the shadow boundary. This behavior has been observed in past experiments, and discussed in [8]. This might explain why scans obtained using the sun (experiments 4 and 5) are more noisy that those with a desk lamp (as the penumbra is larger with the sun by a factor of approximately 5). Second, notice that $\sigma^2_{Z_c}$ is proportional to $\|\overline{\omega}_c\|^2$ (through the terms ω_x^2 and ω_y^2), or, equivalently, inversely proportional to the square of the distance of the shadow plane to the camera center O_c. Therefore, as the shadow plane moves closer to the camera, triangulation becomes more and more sensitive to noise (see discussion in section 2.6). The third observation is less intuitive: one may notice that σ_{Z_c} does not explicitly depend on the local shadow speed at $\overline{x}_c$, at time $t = t_s(\overline{x}_c)$. Therefore, decreasing the scanning speed would not increase accuracy. However, for the analysis leading to equation 9 to remain valid (see Appendix C), the temporal pixel profile must be sufficiently sampled within the transition area of the shadow edge (the penumbra). Therefore, if the shadow edge were sharper, the scanning should also be slower so that the temporal profile at every pixel would be properly sampled. Decreasing further the scanning speed would benefit the accuracy only if the temporal profile were appropriately low-pass filtered or otherwise interpolated before extraction of $t_s(\overline{x}_c)$. This is an issue for future research.

An experimental validation of the variance expression (9) is reported in section 3 (figure 10).

In the case where the light source position is known, it is possible to write the "average" depth variance as a direct function of the variables defining the geometry of the system (Appendix C, equation 22):

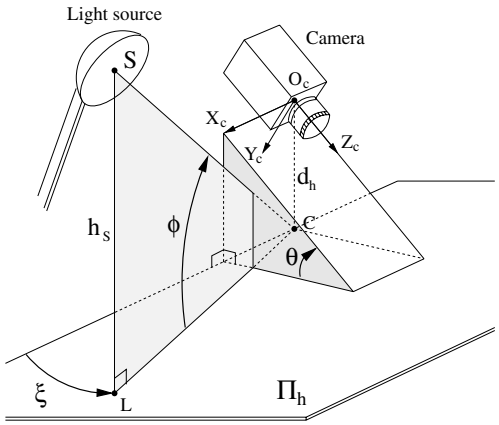


Figure 8: **Geometric setup:** The camera is positioned at a distance d_h away from the plane Π_h and tilted down towards it at an angle θ. The light source is located at a height h_S, with its direction defined by the azimuth and elevation angles ξ and ϕ in the reference frame attached to the plane Π_h. Notice that the sign of $\cos\xi$ directly relates to which side of the camera the lamp is standing: positive on the right, and negative on the left.

$$\sigma_{Z_c}|_{\text{average}} \approx d_h \, \frac{\tan\phi}{\sin^2\theta \, |\cos\xi|} \, \frac{\sigma_I}{f_c \, |I_x(\overline{x}_c)|} \qquad (10)$$

where the quantities d_h, θ, ϕ and ξ characterize the spatial configuration of the camera and the light source with respect to the reference plane Π_h (figure 8). Notice that this quantity may even be computed prior to scanning right after calibration.

In order to maximize the overall reconstruction quality, the position of the light source needs then to be chosen so that the norm of the ratio $\tan\phi / \cos\xi$ is minimized. Therefore, the two optimal values for the azimuth angle are $\xi = 0$ and $\xi = \pi$ corresponding to positioning the lamp either to the right ($\xi = 0$) or to the left ($\xi = \pi$) of the camera (see figure 8). Regarding the elevation angle ϕ, it would be beneficial to make $\tan\phi$ as small as possible. However, making ϕ arbitrarily small is not practically possible. Indeed, setting $\phi = 0$ would constrain the light source to lie on the plane Π_h which would then drastically reduce the effective coverage of the scene due to large amount of self-shadows cast on the scenery. A reasonable tradeoff for ϕ is roughly between 60 and 70 degrees. Regarding the camera position, it would also be beneficial to make $\sin\theta$ as large as possible (ideally equal to one). However, it is very often not practical to make θ arbitrarily close to $\pi/2$. Indeed, having $\theta = \pi/2$ brings the reference plane Π_h parallel to the image plane. Then, standard visual camera calibration algorithms are known to fail (due to lack of depth perspective in the image). In most experiments, we set θ to roughly $\pi/4$.

Once again, for validation purposes, we may use

equation 10 to estimate the reconstruction error of the scans performed in experiment 3 (figure 12). From a set of 10 images, we first estimate the average image brightness noise ($\sigma_I = 2$), and the shadow edge sharpness ($\|\overline{\nabla} I\| \approx 50$). After camera and light source calibration, the following set of parameters is recovered: $f_c = 428$ pixels, $d_h = 22$ cm, $\theta = 39.60$ degrees, $h_S = 53.53$ cm, $\xi = -4.91$ degrees and $\phi = 78.39$ degrees. Equation 10 returns then an estimate of the reconstruction error ($\sigma_{Z_c} \approx 0.2$ mm) very close to the actual error experimentally measured on the final reconstructed surface (between 0.1 mm and 0.2 mm). The first expression given in equation 9 may also be used for obtaining a more accurate estimate of σ_{Z_c} specific to every point in the scene.

2.8 Merging scans

The range data can only be retrieved at pixels corresponding to regions in the scene illuminated by the light source and seen by the camera. In order to obtain better coverage of the scene, one may take multiple scans of the same scene having the light source at different locations each time, while keeping the camera position fixed. Consider the case of two scans with the lamp first on the right, and then on the left of the camera (see figure 9). Assume that at a given pixel $\overline{x}_c$ on the image, two shadow planes are available from the two scans: Π_c^L and Π_c^R. Denote by $\overline{\omega}_c^L$ and $\overline{\omega}_c^R$ their respective coordinate vectors. Then, two estimates Z_c^L and Z_c^R of the corresponding depth at $\overline{x}_c$ are available (from equation 8):

$$\left\{ \begin{array}{rcl} Z_c^L & = & 1 / \left\langle \overline{\omega}_c^L, \overline{x}_c \right\rangle \\ Z_c^R & = & 1 / \left\langle \overline{\omega}_c^R, \overline{x}_c \right\rangle \end{array} \right. \tag{11}$$

One may then calculate the depth estimate at $\overline{x}_c$ by taking a weighted average of Z_c^L and Z_c^R:

$$Z_c \doteq \alpha_L \, Z_c^L + \alpha_R \, Z_c^R \tag{12}$$

where the weights α_L and α_R are computed based on the respective reliabilities of the two depth estimates. Assuming that Z_c^L and Z_c^R are random variables with independent noise terms, they are optimally averaged (in the minimum variance sense) using the inverse of the variances as weights [30]:

$$\frac{\alpha_L}{\alpha_R} = \frac{\sigma_R^2}{\sigma_L^2} \implies \left\{ \begin{array}{l} \alpha_L = \sigma_R^2 / (\sigma_R^2 + \sigma_L^2) \\ \alpha_R = \sigma_L^2 / (\sigma_R^2 + \sigma_L^2) \end{array} \right. \tag{13}$$

where σ_L^2 and σ_R^2 are the variances of the error attached to Z_c^L and Z_c^R respectively.

A sensitivity analysis of the method described in Appendix C provides expressions for those variances (given in equation 9). This technique was used in experiment 1 for merging two scans (see figure 9).

2.9 Real-time implementation

We implemented a real-time version of our 3D scanning algorithm in collaboration with Silvio Savarese of the university of Naples, Italy. In that implementation the process is divided into two main steps. In the first step, the minimum and maximum images $I_{\min}(x, y)$ and $I_{\max}(x, y)$ (eq. 1) are computed through a first fast shadow sweep over the scene (with no shadow edge detection). That step allows to pre-compute the threshold image $I_{\text{shadow}}(x, y)$ (eq. 2) which is useful to compute in real-time the difference image $\Delta I(x, y, t)$ (eq. 3) during the next stage: the scanning procedure itself. During scanning, temporal and spatial shadow edge detections are performed as described in section 2.4: As a new image $I(x, y, t_0)$ is acquired at time $t = t_0$, the corresponding difference image $\Delta I(x, y, t_0)$ is first computed. Then, a given pixel (x_c, y_c) is selected as a pixel lying on the edge of the shadow if $\Delta I(x_c, y_c, t)$ crosses zero between times $t = t_0 - 1$ and $t = t_0$. In order to make that decision, and then compute its corresponding sub-frame shadow time $t_s(x_c, y_c)$, only the previous image difference $\Delta I(x, y, t_0 - 1)$ needs to be stored in memory. Once a pixel (x_c, y_c) is activated and its sub-frame shadow time $t_s(x_c, y_c)$ computed, one may directly identify its corresponding shadow plane Π by linear interpolation between the current shadow plane $\Pi(t_0)$ and the previous one $\Pi(t_0 - 1)$ (see sec. 2.5). Therefore, the 3D coordinates of the point may be directly computed by triangulation (see sec. 2.6). As a result, in that implementation, neither the shadow times $t_s(x, y)$, nor the entire list of shadow planes $\Pi(t)$ need to be stored in memory, only the previous difference image $\Delta I(x, y, t_0 - 1)$ and the previous shadow plane $\Pi(t_0 - 1)$. In addition, scene depth map (or range data) is computed in real-time. The final implementation that we designed also takes advantage of possible multiple passes of the shadow edge over a given pixel in the image by integrating all the successive depth measurements together based on their relative reliabilities (equations 11, 12 and 13 in section 2.8). Details of the implementation may be found in [34].

The real-time program was developed under Visual C++ and works at 30 frames a second on images of size 320×240 on a Pentium 300MHz machine: it takes approximately 30 seconds to scan a scene with a single shadow pass (i.e. $30 \times 30 = 900$ frames), and between one and two minutes for a refined scan using multiple shadow passes. The system uses the PCI frame grabber PXC200 from Imagenation, a NTSC black and white SONY XC-73/L camera (1/3 inch CCD) with a 6mm COSMICAR lens (leading to a 45^o horizontal field of view). Source code (`matlab` for calibration and `C` for scanning) and complete hardware references and specifications are available online at `http://www.vision.caltech.edu/bouguetj/ICCV98`. At the same location, a short demonstration movie of

the working system is also available.

3 Experimental Results

3.1 Calibration accuracy

Camera calibration. For a given setup, we acquired 5 images of the checkerboard pattern (see figure 3-right), and performed independent calibrations on them. The checkerboard, placed at different positions in each image, consisted of 187 visible corners on a 16×10 grid. We computed both mean values and standard deviations of all the parameters independently: the focal length f_c, radial distortion factor k_c and ground plane position Π_h. Regarding the ground plane position, it is convenient to look at its distance d_h to the camera origin O_c and its normal vector $\overline{n}_h$ expressed in the camera reference frame (recall: $\overline{\omega}_h = \overline{n}_h / d_h$). The following table summarizes the calibration results:

Parameters	Estimates	Relative errors
f_c (pixels)	426.8 ± 0.8	0.2%
k_c	-0.233 ± 0.002	1%
d_h (cm)	112.1 ± 0.1	0.1%
$\overline{n}_h$	$\begin{pmatrix} -0.0529 \pm 0.0003 \\ 0.7322 \pm 0.0003 \\ 0.6790 \pm 0.0003 \end{pmatrix}$	0.05%
$\overline{\omega}_h$ (m^{-1})	$\begin{pmatrix} -0.0472 \pm 0.0003 \\ 0.653 \pm 0.006 \\ 0.606 \pm 0.006 \end{pmatrix}$	0.1%

Lamp calibration. Similarly, we collected 10 images of the pencil shadow (like figure 4-top-right) and performed calibration of the light source on them. See section 2.3. Notice that the points $\overline{b}$ and t_s were manually extracted from the images. Define $\overline{X}_S$ as the coordinate vector of the light source in the camera reference frame. The following table summarizes the calibration results obtained for the setup shown in figure 4 (refer to figure 8 for notation):

Parameters	Estimates	Relative errors
$\overline{X}_S$ (cm)	$\begin{pmatrix} -13.7 \pm 0.1 \\ -17.2 \pm 0.3 \\ -2.9 \pm 0.1 \end{pmatrix}$	$\approx 2\%$
h_S (cm)	34.04 ± 0.15	0.5%
ξ (degrees)	146.0 ± 0.8	0.2%
ϕ (degrees)	64.6 ± 0.2	0.06%

The estimated lamp height agrees with the manual measure (with a ruler) of 34 ± 0.5 cm.

This accuracy is sufficient for not inducing any significant global distortion onto the final recovered shape, as we discuss in the next section.

3.2 Scene reconstructions

Experiment 1 - Indoor scene: We took two scans of the same scene with the desk lamp first on the right side and then on the left side of the camera. The two resulting meshes are shown on the top row on figure 9. The meshes were then merged together following the technique described in section 2.8. The bottom figure shows the resulting mesh composed of 66,579 triangles. We estimated the surface error (σ_{Z_c}) to approximately .7 mm in standard deviation over 50 cm large objects, leading to a relative reconstruction error of 0.15%. The white holes in the mesh images correspond to either occluded regions (not observed from the camera, or not illuminated) or very low albedo areas (such as the black squares on the horizontal plane). There was no significant global deformation in the final structured surface: after fitting a quadratic model through sets of points on the two planes, we only noticed a decrease of approximately 5% in standard deviation of the surface error. One may therefore conclude that the calibration procedure returns sufficiently accurate estimates. The original input sequences were respectively 665 and 501 frames long, each image being 320×240 pixels large, captured with a grayscale camera.

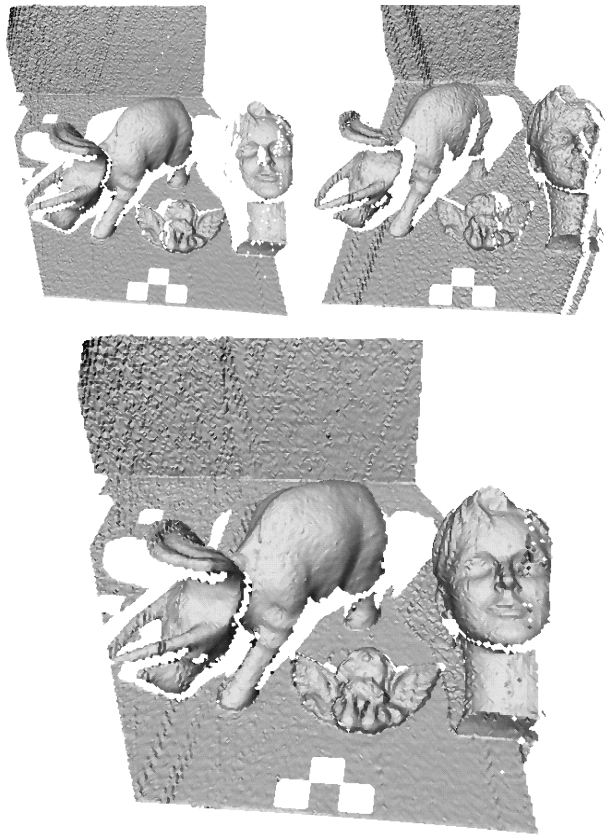
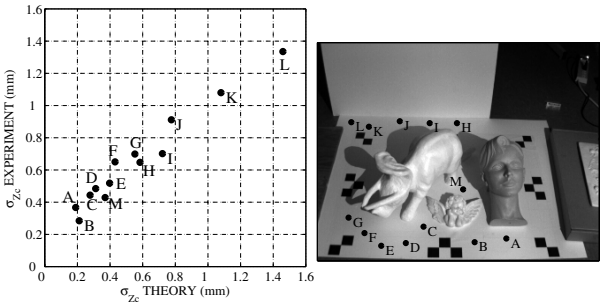


Figure 9: **Experiment 1 - Indoor scene**

Figure 10 reports a comparison test between the theoretical depth variances obtained from expression (9) and that computed from the reconstructed surface. This test was done on the first scan of the scene shown on figure 9-top-left. In that test, we experimentally compute the standard deviation σ_{Z_c} of the error on the depth estimate Z_c at 13 points $p = (A, B, \ldots, M)$



p	$\overline{\nabla} I$	$[\omega_x \ \omega_y]^T$	Z_c	σ_{Z_c} th.	σ_{Z_c} exp.
A	71.5 18.0	1.6591 0.2669	1332.4	0.19	0.37
B	69.0 12.0	1.7755 0.3762	1317.2	0.21	0.28
C	61.0 11.0	1.9639 0.3576	1355.6	0.28	0.44
D	52.0 12.0	2.0788 0.3071	1300.0	0.31	0.48
E	40.5 14.0	2.2454 0.2170	1286.2	0.40	0.52
F	42.0 12.0	2.3455 0.1606	1318.6	0.43	0.65
G	37.5 10.0	2.5048 0.1101	1363.4	0.55	0.70
H	46.5 9.0	1.7752 0.3776	1800.8	0.58	0.65
I	38.5 9.5	1.8700 0.3608	1789.6	0.72	0.70
J	38.0 9.5	2.0038 0.3491	1786.1	0.78	0.91
K	28.0 7.5	2.1815 0.2523	1749.7	1.08	1.08
L	21.5 7.0	2.2834 0.1953	1769.0	1.46	1.34
M	51.0 10.0	1.7905 0.3765	1495.2	0.37	0.43

Figure 10: **Comparison of measured and predicted reconstruction error σ_{Z_c}:** The standard deviation σ_{Z_c} of the depth estimate error are experimentally calculated at 13 points $p = (A, B, \ldots, M)$ picked randomly on the horizontal plane Π_h and computed theoretically using equation 9. The experimental estimates are reported in the last column of the table (in mm) and the second last column reports the corresponding theoretical estimates. The terms involved in equation 9 are also given: $\overline{\nabla} I$ (in units of brightness per pixel), $[\omega_x \ \omega_y]^T$ (in m^{-1}) and Z_c (in mm). The image noise was experimentally estimated to $\sigma_I = 2$ brightness values, and the focal value used was $f_c = 426$ pixels. The top-left figure shows a plot is the theoretical standard deviations versus the experimental ones. Observe that the theoretical error model captures quite faithfully the actual variations in accuracy of reconstruction within the entire scene: as the point of interest moves from the left to the right part of the scenery, accuracy increases due to sharper edges, and a smaller shadow plane vector $\overline{\omega}_c$; in addition, deeper areas in the scene are more noisy mainly because of larger absolute depths Z_c and shallower shadow edges (smaller $\|\overline{\nabla} I\|$). We conclude from that experiment that equation 9 returns an accurate estimate for σ_{Z_c}.

picked randomly on the horizontal plane Π_h of the scan data shown on figure 9-top-left. Figure 10-top-right shows the positions of those points in the scene. The standard deviation σ_{Z_c} at a given point p in the image is experimentally calculated by first taking the 9×9 pixel neighborhood around p resulting into a set of 81 points in space that should lie on Π_h. We then fit a plane across those 81 points (in the least squares sense) and set σ_{Z_c} as the standard deviation of the residual algebraic distances of the entire set of points to this best fit plane. The experimental estimates for σ_{Z_c} are reported in the last column of the table (in mm). The second last column reports the corresponding theoretical estimates of σ_{Z_c} (in mm) computed using equation 9. The terms involved in that equation are also given: $\overline{\nabla} I$ (in units of brightness per pixel), $[\omega_x \ \omega_y]^T$ (in m^{-1}) and Z_c (in mm). The image noise was experimentally estimated to $\sigma_I = 2$ brightness values (calculation based on 100 acquired images of the same scene), and the focal value used was $f_c = 426$ pixels. See sec. 2.7 for a complete description of those quantities. The top-left figure shows a plot of the theoretical standard deviations versus the experimental ones. Observe that the theoretical error model captures quite faithfully the actual variations in accuracy of reconstruction within the entire scene: as the point of interest moves from the left to the right part of the scenery, accuracy increases due to sharper edges, and a smaller shadow plane vector $\overline{\omega}_c$; in addition, deeper areas in the scene are more noisy mainly because of larger absolute depths Z_c and shallower shadow edges (smaller $\|\overline{\nabla} I\|$). We conclude from that experiment that equation 9 returns a valid estimate for σ_{Z_c}.

Experiment 2 - Scanning of a textured skull: We took one scan of a small painted skull, using a single reference plane Π_h, with known light source position (pre-calibrated). Two images of the sequence are shown on the top row of figure 11. The recovered shape is presented on the second row (33,533 triangles), and the last row shows three views of the mesh textured by the top left image. Notice that the textured regions of the object are nicely reconstructed (although these regions have smaller contrast I_{contrast}). Small artifacts observable at some places on the top of the skull are due to the saturation of the pixel values to zero during shadow passage. This effect induces a positive bias on the threshold I_{shadow} (since $I_{\min}$ is not as small as it should be). Consequently, those pixels take on slightly too small shadow times t_s and are triangulated with shadow planes that are shifted to the left. In effect, their final 3D location is slightly off the surface of the object. One possible solution to that problem consists of taking multiple scans of the object with different camera apertures, and retain each time the range results for the pixels that do not suffer from saturation. The overall

reconstruction error was estimated to approximately 0.1 mm over a 10 cm large object leading to a relative error of approximately 0.1%. In order to check for global distortion, we measured the distances between three characteristic points on the object: the tip of the two horns, and the top medium corner of the mouth. The values obtained from physical measurements on the object and the ones from the retrieved model agreed within the error of measurement (on the order of 0.5mm over distances of approximately 12 to 13cm). The sequence of images was 670 frames long, each image being 320×240 pixels large (acquired with a grayscale camera).

Experiment 3 - Textured and colored fruits: Figure 12 shows the reconstruction results on two textured and colored fruits. The second row shows the reconstructed shapes. The two meshes with the pixel images textured on them are shown on the third row. Similar reconstruction errors to the previous experiment (Experiment 2) were estimated on that data set. Notice that both textured and colored regions of the objects were well reconstructed: the local surface errors was estimated between 0.1 mm and 0.2 mm, leading to relative errors of approximately 0.1%.

Experiment 4 - Outdoor scene: In this experiment, the sun was the light source. See figure 13. The final mesh is shown on the bottom figure (106,982 triangles). The reconstruction error was estimated to 1mm in standard deviation, leading to a relative error of approximately 0.2%. The larger reconstruction error is possibly due to the fact that the sun is not well approximated by a point light source (as discussed in Appendix C). Once again, there was no noticeable global deformation induced by calibration. After fitting a quadratic model to sets of points on the planes, we only witnessed a decrease of approximately 5% on the standard deviation of the residual error. The original sequence was 790 images long acquired with a consumer electronics color camcorder (at 30 Hz). After digitization, and de-interlacing, each image was 640×240 pixel large. The different digitalization technique may also explain the larger reconstruction error.

Experiment 5 - Outdoor scanning of a car: Figure 14 shows the reconstruction results on scanning a car with the sun. The two planes (ground floor and back wall) approach was used to infer the shadow plane (without requiring the sun position). The initial sequence was 636 frames long acquired with a consumer electronics color video-camera (approximately 20 seconds long). Similarly to Experiment 4, the sequence was digitized resulting to 640×240 pixel large non-interlaced images. Two images of the sequence are presented on the top row, as well as two views of the reconstructed 3D mesh after scanning. The reconstruction errors were estimated to approximately 1 cm, or 0.5% of the size of the car (approximately 3 meters).

4 Conclusion and future work

We have presented a simple, low cost system for 3D scanning. The system requires very little equipment (a light source, and a straight edge to cast the shadow) and is very simple and intuitive to use and to calibrate. This technique scales well to large objects and may be used in brightly lit scenes where most active lighting methods are impractical (expect synchronized scanning systems [33]). In outdoor scenarios, the sun is used as light source and is allowed to move during a scan. The method requires very little processing and image storage and has been implemented in real time (30 Hz) on a Pentium 300MHz machine. The accuracies that we obtained on the final reconstructions are reasonable (error at most 0.5% of the size of the scene). In addition, the final outcome is a dense and conveniently organized coverage of the surface (one point in space for each pixel in the image), allowing direct triangular meshing and texture mapping. We also showed that using dual-space geometry enables us to keep the mathematical formalism simple and compact throughout the successive steps of the method. An error analysis was presented together with a description of a simple technique for merging multiple 3D scans in order to obtain a better coverage of the scene, and reduce the estimation error. The overall calibration procedure, even in the case of multiple scans, is intuitive, simple, and accurate.

Our method may be used to construct complete 3D object models. One may take multiple scans of the object at different locations in space, and then align the sets of range images. For that purpose, a number of algorithms have been explored and shown to yield excellent results [3, 21, 40]. The final step consists of constructing the final object surface from the aligned views [1, 17, 40].

It is part of future work to incorporate a geometrical model of extended light source to the shadow edge detection process, in addition to developing an uncalibrated (projective) version of the method. One step towards an uncalibrated system may be found in [9]. In this paper, we study the case of 3D reconstruction from a set of planar shadows when there is no calibrated background plane in the scene.

A Dual-space formalism

Let $(E) = \mathbb{R}^3$ be the 3D Euclidean space. A plane Π in (E) is uniquely represented by the 3-vector $\overline{\omega} = [\omega_x \quad \omega_y \quad \omega_z]^T$ such that any point P of coordinate vector $\overline{X}_c = [X_c \quad Y_c \quad Z_c]^T$ (expressed in the camera reference frame) lies on Π if and only if $\langle \overline{\omega}, \overline{X}_c \rangle = 1$ ($\langle ., . \rangle$ is the standard scalar product operator). Notice that $\overline{\omega} \doteq \overline{n}/d$ where $\overline{n}$ is the unitary normal vector of the plane and $d \neq 0$ the plane's distance to the origin. Let $(\Omega) = \mathbb{R}^3$. Since every point $\overline{\omega} \in (\Omega)$

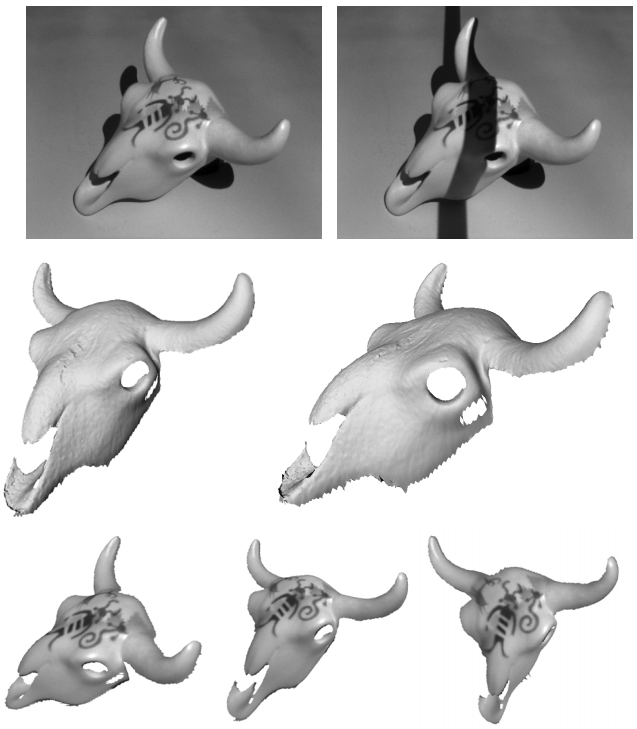


Figure 11: **Experiment 2 - Scanning of a textured skull**

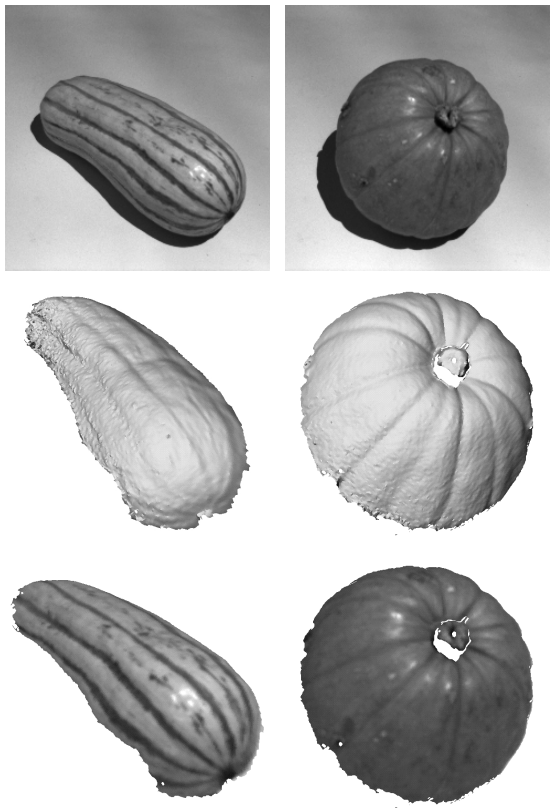


Figure 12: **Experiment 3 - Textured and colored fruits**

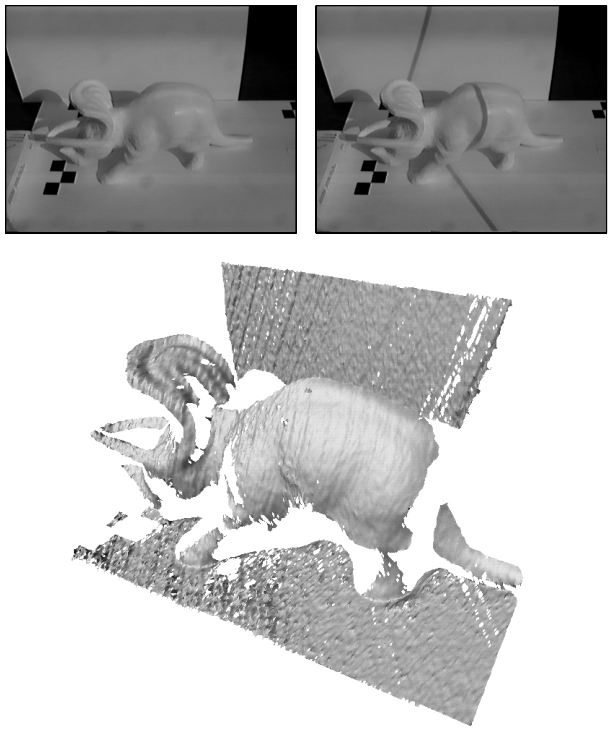


Figure 13: **Experiment 4 - Outdoor scanning of an object**

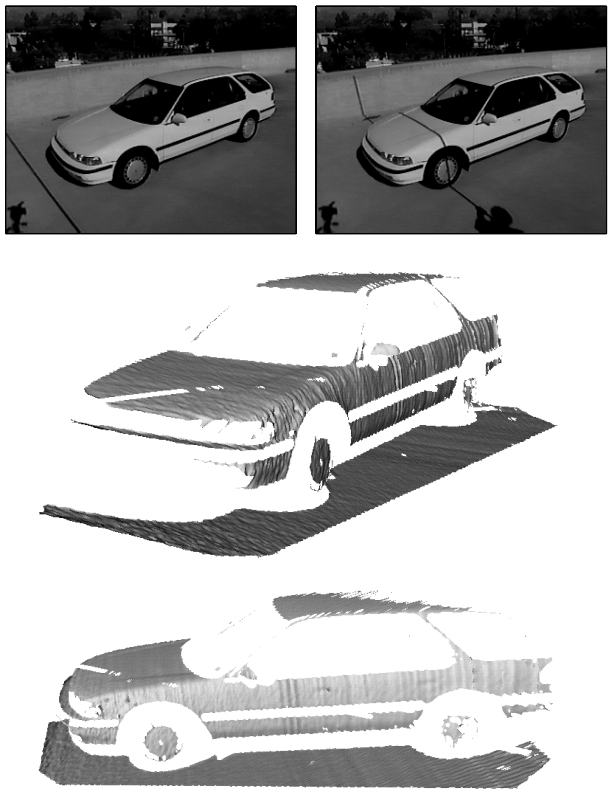


Figure 14: **Experiment 5 - Outdoor scanning of a car**

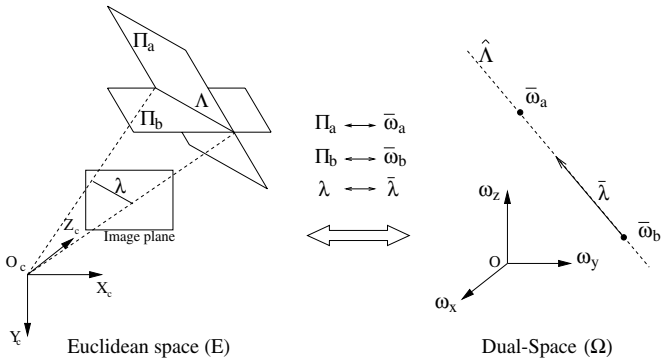


Figure 15: **Proposition 1:** The direction of the line connecting two planes vectors $\overline{\omega}_a$ and $\overline{\omega}_b$ in dual-space (Ω) is precisely $\overline{\lambda}$, the coordinate vector of the perspective projection λ of the line of intersection Λ between the two planes Π_a and Π_b in Euclidean space (E).

corresponds to a unique plane Π in (E), we refer to (Ω) as the 'dual-space'. Conversely, every plane Π that does not contain the origin has a valid coordinate vector $\overline{\omega}$ in (Ω). Notice that the set of plane crossing the origin cannot be parameterized in (Ω) space, since the $\overline{\omega}$ diverges to infinity as d gets closer to zero.

Similarly, a line λ on the image plane is represented by the 3-vector $\overline{\lambda}$ (up to scale) such that any point p of coordinates $\overline{x}_c = [x_c \;\; y_c \;\; 1]^T$ lies on this line if and only if $\langle \overline{\lambda}, \overline{x}_c \rangle = 0$. See [20, 24, 35].

Originally, the dual-space of a given vector space (E) is defined as the set of linear forms on (E) (linear functions of (E) into the reals $\mathbb{R}$). See [4]. In the case where (E) is the three dimensional Euclidean space, each linear form may be interpreted as a plane Π in space that is typically parameterized by a homogeneous 4-vector $\overline{\pi} = [\pi_1 \;\; \pi_2 \;\; \pi_3 \;\; \pi_4]^T$. A point P of homogeneous coordinates $\overline{X} = [X \;\; Y \;\; Z \;\; 1]^T$ lies on a generic plane Π of coordinates $\overline{\pi}$ if and only if $\langle \overline{\pi}, \overline{X} \rangle = 0$ (see [12]). Our $\overline{\omega}-$parameterization differs from the conventional parameterization in that it does not allow to represent planes crossing the origin (the correspondence between the two parameterizations is $\overline{\omega} = -[\pi_1 \;\; \pi_2 \;\; \pi_3]^T/\pi_4$, therefore $\pi_4 \neq 0$). However, that does not constitute a limitation in our application since none of the planes we need to parameterize are allowed to cross the origin (as discussed in sections 2.2 and 2.6). Furthermore, this new representation exhibits useful properties allowing to naturally relate objects in 3D (planes, lines and points) to their perspective projections on the image plane (lines and points) in addition to providing very compact analytical results in error sensitivity analysis.

The following proposition constitutes the major property associated to our choice of parameterization:

Proposition 1: Consider two planes Π_a and Π_b in space, with respective coordinate vectors $\overline{\omega}_a$ and $\overline{\omega}_b$ ($\overline{\omega}_a \neq \overline{\omega}_b$), and let $\Lambda = \Pi_a \cap \Pi_b$ be the line of intersection between them. Let λ be the perspective projection of Λ on the image plane, and $\overline{\lambda}$ its representative vector. Then $\overline{\lambda}$ is parallel to $\overline{\omega}_a - \overline{\omega}_b$ (see figure 15). In other words, $\overline{\omega}_a - \overline{\omega}_b$ is a valid coordinate vector of the line λ.

Proof: Let $P \in \Lambda$ and let p be the projection of P on the image plane. Call $\overline{X} = [X \;\; Y \;\; Z]^T$ and $\overline{x} = \frac{1}{Z}\overline{X}$ the respective coordinates of P and p. We successively have:

$$
\begin{aligned}
P \in \Lambda \quad &\Longleftrightarrow \quad \begin{cases} P \in \Pi_a \\ P \in \Pi_b \end{cases} \\
&\Longleftrightarrow \quad \begin{cases} \langle \overline{\omega}_a, \overline{X} \rangle = 1 \\ \langle \overline{\omega}_b, \overline{X} \rangle = 1 \end{cases} \\
&\Longrightarrow \quad \langle \overline{\omega}_a - \overline{\omega}_b, \overline{x} \rangle = 0.
\end{aligned}
$$

Therefore $(\overline{\omega}_a - \overline{\omega}_b)$ *is* a representative vector of λ and must be parallel to $\overline{\lambda}$. ■

Consequently, the coordinate vector $\overline{\omega}$ of any plane Π containing the line Λ will lie on the line connecting $\overline{\omega}_a$ and $\overline{\omega}_b$ in dual-space (Ω). We denote that line by $\hat{\Lambda}$ and call it the *dual image* of Λ. The following definition generalizes that concept of dual image:

Definition: Let $\mathcal{A}$ be a submanifold of (E) (e.g. a point, line, plane, surface or curve). The *dual image* $\hat{\mathcal{A}}$ of $\mathcal{A}$ is defined as the set coordinates vectors $\overline{\omega}$ in dual-space (Ω) representing the tangent planes to $\mathcal{A}$. Following that standard definition (see [12]), the dual images of points, lines and planes in (E) may be shown to be respectively planes, lines and points in dual-space (Ω), as illustrated in figure 16. Further properties regarding non-linear sub-manifolds may be observed, such as for quadric surfaces in [15].

B Proof of $h_S/d_h = 1 - \langle \overline{\omega}_h, \overline{X}_S \rangle$

Since $\overline{\omega}_h$ is the coordinate vector of the plane Π_h, the vector $\overline{n}_h = d_h\,\overline{\omega}_h$ is the normal vector of the plane Π_h in the camera reference frame (see figure 8). Let P be a point in Euclidean space (E) of coordinate vector $\overline{X}$. The quantity $d_h - \langle \overline{n}_h, \overline{X} \rangle$ is then the (algebraic) orthogonal distance of P to Π_h (positive quantity if the P is on the side of the camera, negative otherwise). In particular, if P lies on Π_h, then $\langle \overline{n}_h, \overline{X} \rangle = d_h$, which is equivalent to $\langle \overline{\omega}_h, \overline{X} \rangle = 1$. The orthogonal distance of the light source S to Π_h is denoted h_S on figure 8. Therefore $h_S = d_h - \langle \overline{n}_h, \overline{X} \rangle$, or equivalently $1 - \langle \overline{\omega}_h, \overline{X}_S \rangle = h_S/d_h$. ■

C Sensitivity Analysis

This appendix presents a complete error analysis for the whole reconstruction scheme. As first mentioned in section 2, the method proposes to associate to every pixel $\overline{x}_c$ the time instant $t_s(\overline{x}_c)$ at which the shadow crosses that particular pixel. That given time corresponds to the shadow plane $\Pi(t_s(\overline{x}_c))$ in space (of coordinate vector $\overline{\omega}_c$), used at the triangulation step

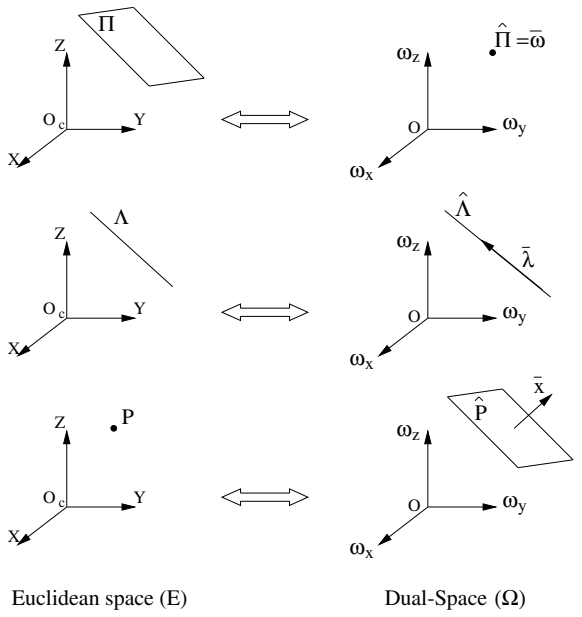


Figure 16: **Duality principle:** The dual images of a plane Π, a line Λ and a point P. Notice that the perspective projection $\overline{\lambda}$ of the line Λ is directly observable in dual-space as the direction vector of its dual image $\hat{\Lambda}$. Similarly, the coordinate vector $\overline{x}$ of the projection of P is precisely the normal vector the plane $\hat{P}$ (dual image of P).

to retrieve the coordinates of the point P in space (see figure 2). In addition, at every time instant t, a shadow plane $\Pi(t)$ is estimated based on two line segments $\lambda_h(t)$ and $\lambda_v(t)$ extracted from the image plane (see section 2.4).

Therefore, one clearly identifies two possible sources of error affecting the overall reconstruction: errors in localizing the two edges $\lambda_h(t)$ and $\lambda_v(t)$ leading to error in estimating the shadow plane $\Pi(t)$ (or error on the vector $\overline{\omega}(t)$), and errors in finding the shadow time $t_s(\overline{x}_c)$ (at every pixel $\overline{x}_c$) leading to an error in shadow plane assignment.

Experimentally, we found that the error coming from spatial processing (shadow plane localization) was much smaller than the one coming from temporal processing (shadow time computation). In other words, in all the experiments we carried out, the shadow planes were localized to such a degree of accuracy that the errors induced by the noise on $\overline{\omega}_c$ were negligible compared to the errors induced by the noise on $t_s(\overline{x}_c)$. This experimental observation is reasonable because the shadow edges $\lambda_h(t)$ and $\lambda_v(t)$ are recovered by fitting lines through many points on the image plane (an order of 50 points per line) while shadow time $t_s(\overline{x}_c)$ is estimated on a basis of a single pixel. Notice that this is experiment dependent, and may very well not be true if fewer points were used to extract the shadow edges, or if the image were more noisy, or more distorted. In those cases, both error terms should be retained. In the present analysis, we propose to derive an expression of the variance of the error in depth estimation $\sigma^2_{Z_c}$ assuming that the main source of noise comes from temporal processing. In the experimental section, we verify that the final variance expression agrees numerically with accuracies achieved on real scan data.

C.1 Derivation of the depth variance $\sigma^2_{Z_c}$

Every pixel $\overline{x}_c$ on the image sees the shadow passing at time a $t_s(\overline{x}_c)$, called the shadow time, that is estimated through temporal processing (see section 2.4). This estimation is naturally subject to errors, leading to inaccuracies in the final 3D reconstruction. The purpose of that analysis is to study how damaging those errors truly are on the final structure, and quantify them. Assume that for a given pixel $\overline{x}_c$, an additive temporal error $\delta t_s(\overline{x}_c)$ is made on its shadow time estimate: $\tilde{t}_s(\overline{x}_c) = t_s(\overline{x}_c) + \delta t_s(\overline{x}_c)$. This typically leads the algorithm to assign to the pixel $\overline{x}_c$ the "wrong" shadow plane $\Pi(t_s(\overline{x}_c) + \delta t_s(\overline{x}_c))$ for the geometrical triangulation step. Equivalently, one can think that the plane $\Pi(t_s(\overline{x}_c) + \delta t_s)$ has been associated with the "wrong" pixel $\overline{x}_c$ in the image. Although it does not change anything to the problem, that way of centering the reasoning onto the shadow plane instead of the pixel actually significantly simplifies the whole analysis. Indeed, as we will show in the following, if we assign the noise to the pixel location itself, the time variable can then be omitted.

To be more precise, let us first define $\overline{v}(\overline{x}_c) = [v_x(\overline{x}_c) \quad v_y(\overline{x}_c)]^T$ to be the velocity vector of the shadow at the pixel $\overline{x}_c$ that is orthogonal to the shadow edge. Then, the closest point to $\overline{x}_c$ that has truly been lit by the shadow plane $\Pi(t_s(\overline{x}_c) + \delta t_s(\overline{x}_c))$ is $\overline{x}_c + \delta t_s(\overline{x}_c)\,\overline{v}(\overline{x}_c)$. Therefore, by picking $\overline{x}_c$ instead, we introduce an additive pixel error $\delta\overline{x}_c \doteq -\delta t_s(\overline{x}_c)\,\overline{v}(\overline{x}_c)$. This is the equivalent noise that can be attributed to the pixel location $\overline{x}_c$ before triangulation.

One can then see that this equivalent image coordinate noise is naturally related to the speed of the shadow. Indeed, even if we assume that the time estimation error δt_s is identical for every pixel in the image, the corresponding pixel error $\delta\overline{x}_c$ is generally not uniform, neither in direction, nor in magnitude. Typically, fast moving shadow regions will be subject to larger errors than slow moving shadow regions. Variations in apparent shadow speed can be caused by a change in the actual speed at which the stick is moved, a change in local surface orientation of the scene, or both.

Before triangulation, the pixel coordinates have to be normalized by the intrinsic parameters of the camera. Let us assume, for simplicity in the notation, that $\overline{x}_c = [x_c \quad y_c \quad 1]^T$ is directly the normalized, homogeneous coordinate vector associated to the pixel. The two coordinates x_c and y_c are affected by the

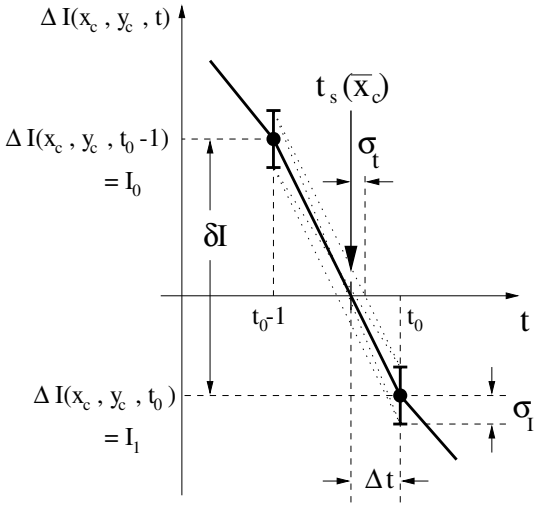


Figure 17: **Estimation error on the shadow time:** The shadow time $t_s(\overline{x}_c)$ is estimated by linearly interpolating the difference temporal brightness function $\Delta I(x_c, y_c, t)$ between times $t_0 - 1$ and t_0. The pixel noise (of standard deviation σ_I) on $I_0 \doteq \Delta I(x_c, y_c, t_0 - 1)$ and $I_1 \doteq \Delta I(x_c, y_c, t_0)$ induces errors on the estimation of Δt, or equivalently $t_s(\overline{x}_c)$. This error has variance σ_t^2.

error vector $\delta\overline{x}_c$ whose variance-covariance matrix is denoted $\Sigma_{\overline{x}_c}$ (a 2×2 matrix). Let us derive an expression for that matrix as a function of the image brightness noise.

Lemma: Let σ_I be the standard deviation of the image brightness noise (estimated experimentally). We can write $\Sigma_{\overline{x}_c}$ as a function of the image gradient $\overline{\nabla} I(\overline{x}_c)$ at pixel $\overline{x}_c$ at time $t = t_s(\overline{x}_c)$:

$$\Sigma_{\overline{x}_c} = \frac{\sigma_I^2}{f_c^2 \, \|\overline{\nabla} I(\overline{x}_c)\|^2} \begin{bmatrix} \cos^2\varphi & \cos\varphi \sin\varphi \\ \cos\varphi \sin\varphi & \sin^2\varphi \end{bmatrix} \tag{14}$$

where f_c is the focal length of the camera (in pixels), $\overline{\nabla} I(\overline{x}_c)$ is the gradient vector of the image brightness at the shadow, and φ the orientation angle of that vector (orientation of the shadow edge at pixel $\overline{x}_c$):

$$\overline{\nabla} I(\overline{x}_c) = \begin{bmatrix} I_x(\overline{x}_c) \\ I_y(\overline{x}_c) \end{bmatrix} = \|\overline{\nabla} I(\overline{x}_c)\| \begin{bmatrix} \cos\varphi \\ \sin\varphi \end{bmatrix}$$

where:

$$I_x(\overline{x}_c) \doteq \left. \frac{\partial I(\overline{x}, t)}{\partial x} \right|_{\overline{x}=\overline{x}_c, t=t_s(\overline{x}_c)}$$

$$I_y(\overline{x}_c) \doteq \left. \frac{\partial I(\overline{x}, t)}{\partial y} \right|_{\overline{x}=\overline{x}_c, t=t_s(\overline{x}_c)}$$

Proof of lemma (eq. 14): Figure 17 shows the principle of computing the shadow time $t_s(\overline{x}_c)$ from the difference image ΔI (refer to section 2.5). For clarity in the notation, define $I_0 \doteq \Delta I(x_c, y_c, t_0 - 1)$ and $I_1 \doteq \Delta I(x_c, y_c, t_0)$. Then, the shadow time $t_s(\overline{x}_c)$ is given by:

$$t_s(\overline{x}_c) = t_0 - \Delta t$$

where:

$$\Delta t \doteq \frac{I_1}{I_1 - I_0}$$

Let σ_t^2 be the variance of the error $\delta t_s(\overline{x}_c)$ attached to the shadow time $t_s(\overline{x}_c)$. In normal sampling conditions (if the temporal brightness is sufficiently sampled within the shadow transition area), the same error is on the variable Δt, and therefore σ_t may be directly expressed as a function of σ_I, the variance of pixel noise on I_0 and I_1:

$$\sigma_t^2 = \left(\left(\frac{\partial \Delta t}{\partial I_0} \right)^2 + \left(\frac{\partial \Delta t}{\partial I_1} \right)^2 \right) \sigma_I^2$$

$$\sigma_t^2 = \frac{I_0^2 + I_1^2}{\delta I^4} \, \sigma_I^2 \tag{15}$$

where $\delta I \doteq I_1 - I_0$ is the temporal brightness variation at the zero crossing (or equivalently at the shadow time). One may notice from equation 15 that, as the brightness difference δI increases, the error in shadow time decreases. That is a very intuitive behavior given that higher shadow contrasts should give rise to better accuracies. Notice however that the variance σ_t^2 is not only a function of δI but also of the absolute brightness values I_0 and I_1. One may then consider the maximum value of σ_t^2 for a fixed δI over all I_0 and I_1, subject to the constraint $I_1 = I_0 + \delta I$:

$$\sigma_t^2 = \max_{0 < I_0 < -\delta I} \left\{ \frac{2\, I_0^2 + 2\, I_0\, \delta I + \delta I^2}{\delta I^4} \right\} \sigma_I^2$$

leading to the following simplified expression for σ_t^2:

$$\sigma_t^2 = \frac{\sigma_I^2}{\delta I^2} \tag{16}$$

To motivate that simplification, one may notice that the minimum and maximum values of σ_t^2 over all values I_0 and I_1 are quite similar anyway: $\sigma_I^2/(2\,\delta I^2)$ (minimum) and $\sigma_I^2/\delta I^2$ (maximum). The maximum may be thought as an upper bound on the error. Notice that δI is nothing but the first temporal derivative of the image brightness at the pixel $\overline{x}_c$, at the shadow time:

$$\delta I = \left. \frac{\partial I(\overline{x}, t)}{\partial t} \right|_{\overline{x}=\overline{x}_c, t=t_s(\overline{x}_c)}$$

This temporal derivative may also be expressed as a function of the image gradient vector $\overline{\nabla} I(\overline{x}_c) =$

$[I_x(\overline{x}_c) \quad I_y(\overline{x}_c)]^T$ and the shadow edge velocity vector $\overline{v}(\overline{x}_c) = [v_x(\overline{x}_c) \quad v_y(\overline{x}_c)]^T$:

$$\delta I = -\overline{\nabla} I(\overline{x}_c)^T \, \overline{v}(\overline{x}_c) = -I_x(\overline{x}_c)\, v_x(\overline{x}_c) - I_y(\overline{x}_c)\, v_y(\overline{x}_c)$$

By definition, the edge velocity vector $\overline{v}(\overline{x}_c)$ is orthogonal to the shadow edge. Therefore it may be also written as a direct function of the gradient vector $\overline{\nabla} I(\overline{x}_c)$:

$$\overline{v}(\overline{x}_c) = s \, \|\overline{v}(\overline{x}_c)\| \, \frac{\overline{\nabla} I(\overline{x}_c)}{\|\overline{\nabla} I(\overline{x}_c)\|} = s \, \|\overline{v}(\overline{x}_c)\| \begin{bmatrix} \cos\varphi \\ \sin\varphi \end{bmatrix}$$

where s is either $+1$ or -1 depending on the direction of motion of the edge. Therefore,

$$\delta I = (-s) \, \frac{\overline{\nabla} I(\overline{x}_c)^T \, \overline{\nabla} I(\overline{x}_c)}{\|\overline{\nabla} I(\overline{x}_c)\|} \, \|\overline{v}(\overline{x}_c)\|$$

$$\delta I = (-s) \, \|\overline{\nabla} I(\overline{x}_c)\| \, \|\overline{v}(\overline{x}_c)\| \qquad (17)$$

Consequently, by substituting (17) into (16), we obtain a new expression for the temporal variance σ_t^2:

$$\sigma_t^2 = \frac{\sigma_I^2}{\|\overline{\nabla} I(\overline{x}_c)\|^2 \, \|\overline{v}(\overline{x}_c)\|^2}$$

Then, the error vector $\delta\overline{x}_c$ transfered on the image plane is also related to the shadow edge velocity $\overline{v}(\overline{x}_c)$ and the temporal error $\delta t_s(\overline{x}_c)$:

$$\delta\overline{x}_c = -\delta t_s(\overline{x}_c) \, \overline{v}(\overline{x}_c)$$

$$\delta\overline{x}_c = (-s) \, \|\overline{v}(\overline{x}_c)\| \, \delta t_s(\overline{x}_c) \begin{bmatrix} \cos\varphi \\ \sin\varphi \end{bmatrix}$$

Then, the variance-covariance matrix of the noise $\delta\overline{x}_c$ is (recall that $s^2 = 1$):

$$\Sigma_{\overline{x}_c} = \|\overline{v}(\overline{x}_c)\|^2 \, \sigma_t^2 \begin{bmatrix} \cos^2\varphi & \cos\varphi\sin\varphi \\ \cos\varphi\sin\varphi & \sin^2\varphi \end{bmatrix}$$

$$\Sigma_{\overline{x}_c} = \frac{\sigma_I^2}{\|\overline{\nabla} I(\overline{x}_c)\|^2} \begin{bmatrix} \cos^2\varphi & \cos\varphi\sin\varphi \\ \cos\varphi\sin\varphi & \sin^2\varphi \end{bmatrix}$$

Finally, note that this relation is valid if x_c is expressed in pixel coordinates. After normalization, this variance must be scaled by the square of the inverse of focal length f_c:

$$\Sigma_{\overline{x}_c} = \frac{\sigma_I^2}{f_c^2 \, \|\overline{\nabla} I(\overline{x}_c)\|^2} \begin{bmatrix} \cos^2\varphi & \cos\varphi\sin\varphi \\ \cos\varphi\sin\varphi & \sin^2\varphi \end{bmatrix}$$

which ends the proof of the lemma (eq. 14). ■

Notice that if the shadow edge is roughly vertical on the image, one may assume $\varphi = 0$, and therefore simplify quite significantly the variance expression:

$$\Sigma_{\overline{x}_c} = \frac{\sigma_I^2}{f_c^2 \, I_x^2(\overline{x}_c)} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

In that case, we reach the very intuitive result that only the first coordinate of $\overline{x}_c$ is affected by noise.

Since $\Sigma_{\overline{x}_c}$ in inversely proportional to the image gradient, accuracy improves with shadow edge sharpness. In addition, observe that $\Sigma_{\overline{x}_c}$ does not directly depend upon the local shadow speed. Therefore, decreasing the scanning speed would not increase accuracy. However, for the analysis leading to equation 14 to remain valid, the temporal pixel profile must be sufficiently sampled within the transition area of the shadow edge (the penumbra). Therefore, if the shadow edge were sharper, the scanning should also be slower so that the temporal profile at every pixel would be properly sampled. Further discussions may be found in section 2.7. Another consequence of equation 14 is that one may experimentally compute the variance $\Sigma_{\overline{x}_c}$ of the transfered error directly from the original input sequence: $\overline{\nabla} I(\overline{x}_c)$ is the image gradient at the shadow edge and σ_I is the pixel noise on the image. In addition, assuming that the sharpness of the shadow is approximately uniform over the entire image, then $\Sigma_{\overline{x}_c}$ may also be assumed to be uniform to a first approximation. That constitutes an additional simplification that does not have to be retained in practice.

The final expression of the variance $\sigma_{Z_c}^2$ of the error attached to the depth estimate Z_c may be written as follows:

$$\sigma_{Z_c}^2 = \left(\frac{\partial Z_c}{\partial \overline{x}_c}\right) \Sigma_{\overline{x}_c} \left(\frac{\partial Z_c}{\partial \overline{x}_c}\right)^T$$

One may derive the expression for the Jacobian matrix $\left(\frac{\partial Z_c}{\partial \overline{x}_c}\right)$ from the triangulation equation 8:

$$Z_c = \frac{1}{\langle \overline{\omega}_c, \overline{x}_c \rangle} \implies \frac{\partial Z_c}{\partial \overline{x}_c} = Z_c^2 \begin{bmatrix} \omega_x & \omega_y \end{bmatrix}$$

where ω_x and ω_y are the two first coordinates of the shadow plane vector $\overline{\omega}_c$. This allows to expand the expression of $\sigma_{Z_c}^2$:

$$\sigma_{Z_c}^2 = Z_c^4 \left(\frac{\omega_x \cos\varphi + \omega_y \sin\varphi}{f_c \, \|\overline{\nabla} I(\overline{x}_c)\|}\right)^2 \sigma_I^2 \qquad (18)$$

This expression is directly computable from the original input sequence, and used for scan merging (refer to section 2.8). Several observations regarding that expression may be found in section 2.7.

C.2 System Design Issues

Let us consider the scanning setup as it is presented on figure 8 where the scan is done roughly vertically. In that case, $\varphi \approx 0$, and $I_y^2(\overline{x}_c) \ll I_x^2(\overline{x}_c)$ (see figure 10). Then, the depth variance expression (18) may be further simplified to:

$$\sigma_{Z_c}^2 \approx \frac{Z_c^4\,\omega_x^2}{f_c^2\,I_x^2(\overline{x}_c)}\,\sigma_I^2 \tag{19}$$

It appears then that the first coordinate ω_x of the shadow plane vector $\overline{\omega}_c$ carries most of the variations in accuracy of reconstruction within a given scan. When designing the scanning system, an important issue is to choose the spatial configurations of the camera and the light source that maximize the overall quality of reconstruction, or equivalently minimize $|\omega_x|$. In order to address this issue, it is necessary to further expand the term ω_x, and study its dependence upon the geometrical variables characterizing the system. Since the light source position is of interest here, let us consider the case where a single plane Π_h is used for scanning. In that case, the shadow plane vector $\overline{\omega}_c$ appears as a function of the light source position vector $\overline{X}_S$, as stated by equation 6. Assume that $\overline{\lambda}_h = [\lambda_x \;\; \lambda_y \;\; \lambda_z]^T$ is normalized such that $\lambda_x = 1$. In addition, assume that the (O_c, X_c) axis of the camera is approximately parallel to the plane Π_h (as suggested in figure 8). This implies that the first coordinate of $\overline{\omega}_h$ is zero. Then, the first coordinate ω_x of $\overline{\omega}_c$ reduces to:

$$\omega_x = \frac{1 - \langle \overline{\omega}_h, \overline{X}_S \rangle}{\langle \overline{\lambda}_h, \overline{X}_S \rangle} = \frac{h_S/d_h}{\langle \overline{\lambda}_h, \overline{X}_S \rangle} \tag{20}$$

where d_h and h_S are the respective orthogonal distances of the camera center O_c and the light source S to the plane Π_h.

For simplification purposes, let us assume that the shadow edge λ_h appears vertically on the image plane, and let x be its horizontal position (on the image). As the shadow moves from left to right, x varies from negative values to positive values, crossing zero when the shadow is at the center of the image. In that specific scenario, the shadow edge vector reduces to: $\overline{\lambda}_h = [1 \;\; 0 \;\; -x]^T$ simplifying equation 20:

$$\frac{1}{\omega_x} = \frac{d_h}{h_S}(X_S - x\,Z_S) \tag{21}$$

The problem of maximizing the reconstruction quality corresponds then to maximizing $|1/\omega_x|$. Since that quantity is function of the shadow edge location x, we may observe that the accuracy of reconstruction is not uniform throughout the scene for a given scan (unless the depth of the light source in the camera reference frame is zero: $Z_S = 0$). A better understanding of that relation may be achieved by expressing the light source coordinate vector $\overline{X}_S$ as a function of the angular coordinates θ, ϕ, and ξ defining the mutual positions of the camera and the light source with respect to the plane Π_h (see figure 8):

$$\overline{X}_S = \begin{bmatrix} X_S \\ Y_S \\ Z_S \end{bmatrix} = \begin{bmatrix} h_S \frac{\cos\xi}{\tan\phi} \\ -h_S \frac{\sin\theta\sin\xi}{\tan\phi} + (d_d - h_S)\cos\theta \\ h_S \frac{\cos\theta\sin\xi}{\tan\phi} + (d_d - h_S)\sin\theta \end{bmatrix}$$

Following this notation, the inverse of ω_x may be written as follows:

$$\frac{1}{\omega_x} = d_h \left(\frac{\cos\xi}{\tan\phi} - x \left(\frac{\cos\theta\sin\xi}{\tan\phi} + \frac{d_h - h_S}{h_S}\sin\theta \right) \right)$$

Since during scanning, the shadow edge coordinate x spans a range of values going from negative to positive values, we may consider that taking $x = 0$ gives us an indication of the "average" reconstruction quality:

$$\left.\frac{1}{\omega_x}\right|_{\text{average}} \approx \left.\frac{1}{\omega_x}\right|_{x=0} = d_h\,\frac{\cos\xi}{\tan\phi}$$

Equation 19 may then be used to infer an expression for the "average" depth variance:

$$\left.\sigma_{Z_c}^2\right|_{\text{average}} \approx \frac{Z_c^4}{d_h^2}\,\frac{\tan^2\phi}{\cos^2\xi}\,\frac{\sigma_I^2}{f_c^2\,I_x^2(\overline{x}_c)}$$

A next simplification step may be applied, by observing that the average depth of the scene is approximately related to the height d_h and the tilt angle θ of the camera through the following expression:

$$\left.Z_c\right|_{\text{average}} \approx \frac{d_h}{\sin\theta}$$

That relation leads us to a new expression for the "average" σ_{Z_c}:

$$\left.\sigma_{Z_c}\right|_{\text{average}} \approx d_h\,\frac{\tan\phi}{\sin^2\theta\;|\cos\xi|}\,\frac{\sigma_I}{f_c\;|I_x(\overline{x}_c)|} \tag{22}$$

Notice that this quantity may be computed prior to scanning knowing the geometrical configuration of the system. From that expression, it is also possible to identify optimal configurations of the camera and the light source that maximize the overall quality of the reconstruction. See section 2.7.

Acknowledgments

This work is supported in part by the California Institute of Technology; an NSF National Young Investigator Award to

P.P.; a STC fund; the Center for Neuromorphic Systems Engineering funded by the National Science Foundation at the California Institute of Technology. We wish to thank all the colleagues that helped us throughout this work, especially Peter Schröder, Paul Debevec, Wolfgang Stürzlinger, Luis Goncalves, George Barbastathis and Mario Munich for very useful discussions. Very special thanks go to Silvio Savarese for his work on the real-time implementation of our algorithm.

References

[1] C.L. Bajaj, F. Bernardini, and G. Xu Xu, "Automatic reconstruction of surfaces and scalar fields from 3D scans", *In SIGGRAPH '95, Los Angeles, CA*, pages 109–118, August 1995.

[2] Paul Besl, *Advances in Machine Vision*, chapter 1 - Active optical range imaging sensors, pages 1–63, Springer-Verlag, 1989.

[3] P.J. Besl and N.D. McKay, "A method for registration of 3-d shapes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[4] R.L. Bishop and S.I. Goldberg, *Tensor analysis on manifold*, Dove Publications, 1980.

[5] Sylvain Bougnoux, "From projective to euclidean space under any practical situation, a criticism of self-calibration", *Proc. 6^{th} Int. Conf. Computer Vision, Bombay, India*, pages 790–796, January 1998.

[6] Jean-Yves Bouguet, *Visual methods for three-dimensional modeling*, PhD thesis, California Institute of Technology, 1999. Available at: http://www.vision.caltech.edu/bouguetj/thesis/thesis.html.

[7] Jean-Yves Bouguet and Pietro Perona, "3D Photography on your Desk", Technical report, California Institute of Technology, 1997, available at: http://www.vision.caltech.edu/bouguetj/ICCV98.

[8] Jean-Yves Bouguet and Pietro Perona, "3D Photography on your Desk", *Proc. 6^{th} Int. Conf. Computer Vision, Bombay, India*, pages 43–50, January 1998.

[9] Jean-Yves Bouguet, Markus Weber, and Pietro Perona, "What do planar shadows tell us about scene geometry?", *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, I:514–520, 1999.

[10] D. C. Brown, "Analytical calibration of close range cameras", *Proc. Symp. Close Range Photogrammetry, Melbourne, FL*, 1971.

[11] D. C. Brown, "Calibration of close range cameras", *Proc. 12^th Congress Int. Soc. Photogrammetry, Ottawa, Canada*, 1972.

[12] J.W. Bruce, "Lines, surfaces and duality", Technical report, Dept. of Pure Mathematics, University of Liverpool, 1992.

[13] J.F. Canny, "A computational approach to edge detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[14] B. Caprile and V. Torre, "Using vanishing points for camera calibration", *IJCV*, 4(2):127–140, March 1990.

[15] Geoffrey Cross and Andrew Zisserman, "Quadric Reconstruction from Dual-Space Geometry", *Proc. 6^{th} Int. Conf. Computer Vision, Bombay, India*, pages 25–31, 1998.

[16] Brian Curless and Marc Levoy, "Better optical triangulation through spacetime analysis", *Proc. 5^{th} Int. Conf. Computer Vision, Boston, USA*, pages 987–993, 1995.

[17] Brian Curless and Marc Levoy, "A volumetric method for building complex models from range images", *SIGGRAPH96, Computer Graphics Proceedings*, 1996.

[18] K. Daniilidis and J. Ernst, "Active intrinsic calibration using vanishing points", *PRL*, 17(11):1179–1189, September 1996.

[19] O.D. Faugeras, *Three dimensional vision, a geometric viewpoint*, MIT Press, 1993.

[20] Olivier Faugeras and Bernard Mourrain, "On the geometry and algebra of the point and line correspondence between n images", *Proc. 5^{th} Int. Conf. Computer Vision, Boston, USA*, pages 851–856, 1994.

[21] H. Gagnon, M. Soucy, R. Bergevin, and D. Laurendeau, "Registration of multiple range views for automatic 3-D model building", *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 581–586, June 1994.

[22] A.A. Goshtasby, S. Nambala, W.G. deRijk, and S.D. Campbell, "A System for Digital Reconstruction of Gypsum Dental Casts", *IEEE Transactions on Medical Imaging*, 16(5):664–674, October 1987.

[23] A. Gruss, S. Tada, and T. Kanade, "A VLSI Smart Sensor for Fast Range Imaging", In *DARPA93*, pages 977–986, 1993.

[24] Richard I. Hartley, "A linear method for reconstruction from lines and points", *Proc. 5^{th} Int. Conf. Computer Vision, Boston, USA*, pages 882–887, 1994.

[25] Janne Heikkila and Olli Silven, "A four-step camera calibration procedure with implicit image correction", *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 1106–1112, 1997.

[26] R. A. Jarvis, "A perspective on range-finding techniques for computer vision", *IEEE Trans. Pattern Analysis Mach. Intell.*, 5:122–139, March 1983.

[27] T. Kanade, A. Gruss, and L. Carley, "A Very Fast VLSI Rangefinder", In *IEEE International Conference on Robotics and Automation*, volume 39, pages 1322–1329, April 1991.

[28] Reinhard Koch, Marc Pollefeys, and Luc Van Gool, "Multi viewpoint stereo from uncalibrated video sequence", *Proc. 5^{th} European Conf. Computer Vision, Freiburg, Germany*, pages 55–71, June 1998.

[29] Jurgen R. Meyer-Arendt, "Radiometry and photometry: Units and conversion factors", *Applied Optics*, 7(10):2081–2084, October 1968.

[30] Athanasios Papoulis, *Probability, Random Variables and Stochastic Processes*, Mac Graw Hill, 1991, Third Edition, page 187.

[31] Marc Pollefeys, Reinhard Koch, and Luc Van Gol, "Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters", *Proc. 6^{th} Int. Conf. Computer Vision, Bombay, India*, pages 90–95, January 1998.

[32] Marc Pollefeys and Luc Van Gool, "A stratified approach to metric self-calibration", *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 407–412, 1997.

[33] Riou, "High resolution digital 3-d imaging of large structures", *SPIE Proceedings, 3-D Image Capture, San Jose*, 3023:109–118, February 1997.

[34] Silvio Savarese, "Scansione tridimensionale con metodi a luce debolmente strutturata", *Tesi di Laurea, Universita degli Studi di Napoli Federico II*, 1998.

[35] A. Shashua and M. Werman, "Trilinearity of three perspective views and its associated tensor", *Proc. 5^{th} Int. Conf. Computer Vision, Boston, USA*, pages 920–925, 1995.

[36] G.P. Stein, "Accurate Internal Camera Calibration Using Rotation, with Analysis of Sources of Error", In *Proc. 5^{th} Int. Conf. Computer Vision, Boston, USA*, pages 230–236, 1995.

[37] Peter F. Sturm and Stephen J. Maybank, "On plane-based camera calibration: A general algorithm, singularities, applications", *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, I:432–437, 1999.

[38] Marjan Trobina, "Error model of a coded-light range sensor", Technical Report BIWI-TR-164, ETH-Zentrum, 1995.

[39] R.Y. Tsai, "A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses", *IEEE J. Robotics Automat.*, RA-3(4):323–344, 1987.

[40] G. Turk and M. Levoy, "Zippered polygon meshes from range images", *In SIGGRAPH '94*, pages 311–318, July 1994.

[41] John W. T. Walsh, *Photometry*, Dover, NY, 1965.

[42] L.L. Wang and W.H. Tsai, "Computing camera parameters using vanishing-line information from a rectangular parallelepiped", *MVA*, 3(3):129–141, 1990.

[43] Y.F. Wang, "Characterizing three-dimensional surface structures from visual images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):52–60, 1991.

SIGGRAPH 2000 Course on
3D Photography

Shape and Appearance from Images and Range Data

Brian Curless
University of Washington

Overview

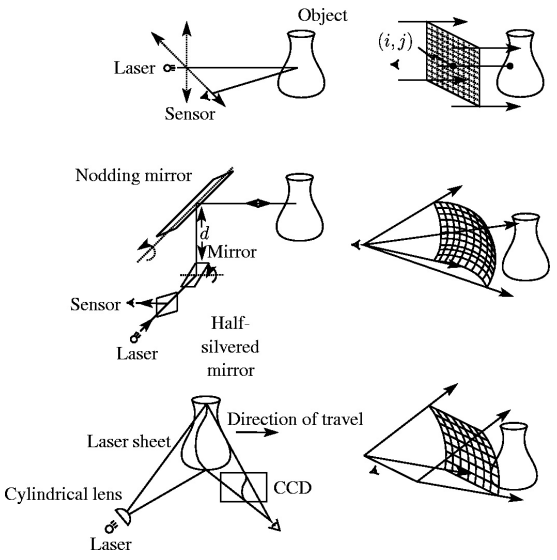
Range images vs. point clouds
Registration
Reconstruction from point clouds
Reconstruction from range images
Modeling appearance

Range images

For many structured light scanners, the range data forms a highly regular pattern known as a *range image*.

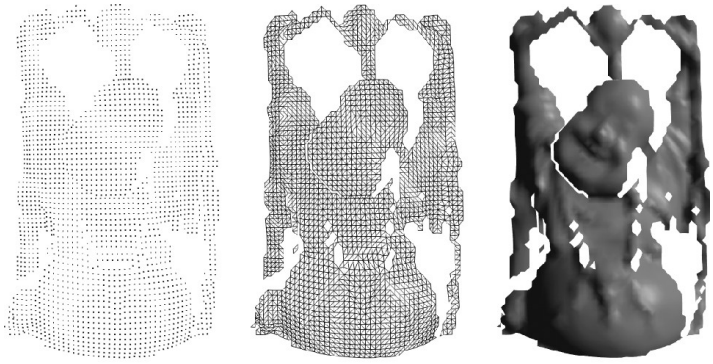
The sampling pattern is determined by the specific scanner.

Examples of sampling patterns



Range images and range surfaces

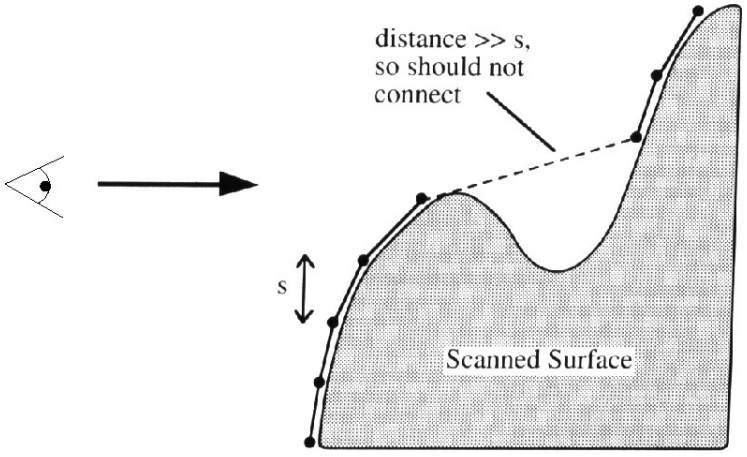
Given a range image, we can perform a preliminary reconstruction known as a *range surface*.



Range image Tesellation Range surface

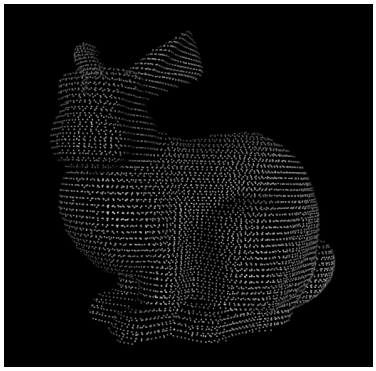
Tessellation threshold

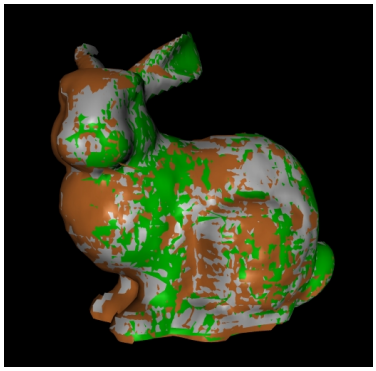
To avoid "prematurely aggressive" reconstruction, a tessellation threshold is employed:



Point clouds vs. range images

We can view the entire set of range data as a point cloud or as a group of overlapping range surfaces.





Registration

Any surface reconstruction algorithm strives to use all of the detail in the range data.

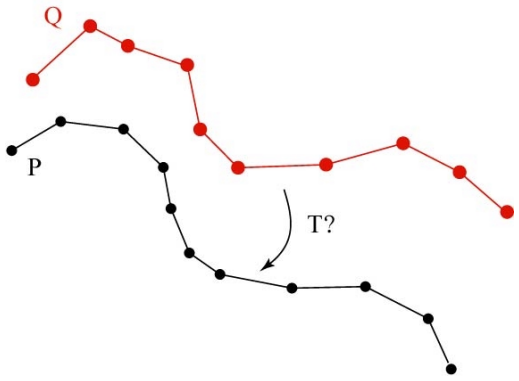
To preserve this detail, the range data must be precisely registered.

Accurate registration may require:

- ***Calibrated scanner positioning***
- ***Software optimization***
- ***Both***

Registration as optimization

Given two overlapping range scans, we wish to solve for the rigid transformation, *T*, that minimizes the distance between them.

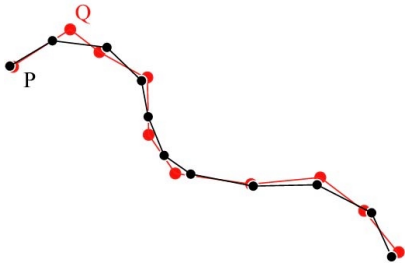


Registration as optimization

An approximation to the distance between range scans is:

$$E = \sum_{i}^{N_P} \left\| Tq_i - p_i \right\|^2$$

Where the q_i are samples from scan Q and the p_i are the *corresponding* points of scan P. These points may lay on the range surface derived from P.



Registration as optimization

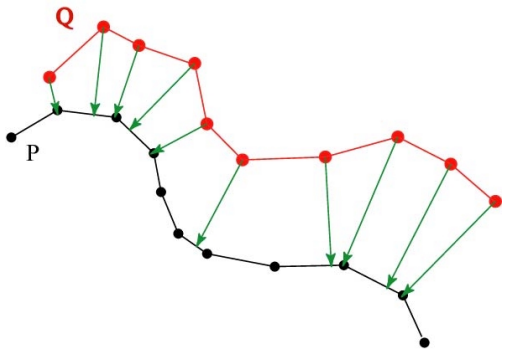
If the correspondences are known a priori, then there is a closed form solution for *T*.

However, the correspondences are not known in advance.

Registration as optimization

Iterative solutions such as [Besl92] proceed in steps:

- ***Identify nearest points***
- ***Compute the optimal T***
- ***Repeat until E is small***



Registration as optimization

This approach is troubled by slow convergence when surfaces need to slide along each other.

Chen and Medioni [Chen92] describe a method that does not penalize sliding motions.

The Chen and Medioni method was the method of choice for pairwise alignment on the Digital Michelangelo Project.

Global registration

Pairwise alignment leads to accumulation of errors when walking across the surface of an object.

The optimal solution minimizes distances between all range scans *simultaneously*. This is sometimes called the *global registration* problem.

Finding efficient solution methods to the global registration problem is an active area of research.

Surface reconstruction

Given a set of registered range points or images, we want to reconstruct a 2D manifold that closely approximates the surface of the original model.

Desirable properties

Desirable properties for surface reconstruction:

- ***No restriction on topological type***
- ***Representation of range uncertainty***
- ***Utilization of all range data***
- ***Incremental and order independent updating***
- ***Time and space efficiency***
- ***Robustness***
- ***Ability to fill holes in the reconstruction***

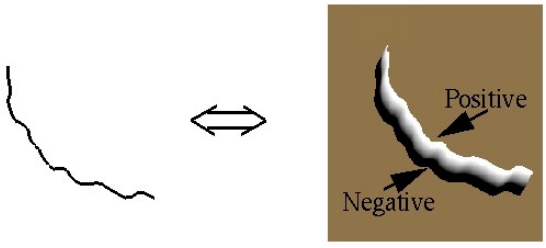
Reconstruction methods

Surface reconstruction from range data has been an active area of research for many years.

A number of methods reconstruct from unorganized points. Such methods:

- ***are general***
- ***typically do not use all available information***

Parametric vs. implicit



Parametric Surface

Implicit surface
F(x) = 0

Reconstruction from unorganized points

Methods that construct triangle meshes directly:

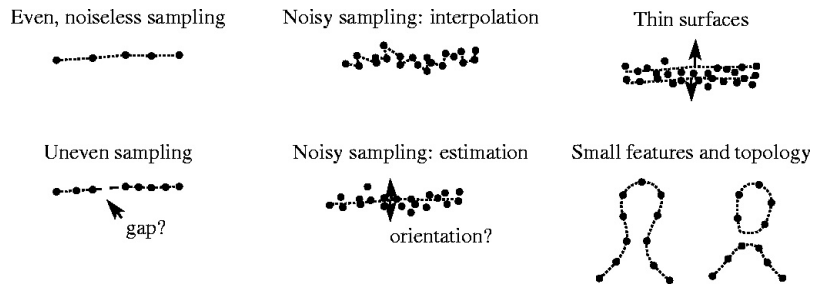
- ***Alpha shapes [Edelsbrunner92]***
- ***Local Delaunay triangulations [Boissonat94]***
- ***Crust algorithm [Amenta98]***

Methods that construct implicit functions:

- ***Voxel-based signed distance functions [Hoppe92]***
- ***Bezier-Bernstein polynomials [Baja95]***

Hoppe treats his reconstruction as a topologically correct approximation to be followed by mesh optimization [Hoppe93].

Reconstruction from unorganized points



Reconstruction from range images

Methods that construct triangle meshes directly:

- *Re-triangulation in projection plane [Soucy92]*
- *Zippering in 3D [Turk94]*

Methods that construct implicit functions:

- *Signed distances to nearest surface [Hilton96]*
- *Signed distances to sensor + space carving [Curless96]*

We will focus on the two reconstruction algorithms of [Turk94] and [Curless96].

Zippering

A number of methods combine range surfaces by stitching polygon meshes together.

Zippering [Turk94] is one such method.

Overview:

- *Tessellate range images and assign weights to vertices*
- *Remove redundant triangles*
- *Zipper meshes together*
- *Extract a consensus geometry*

Weight assignment

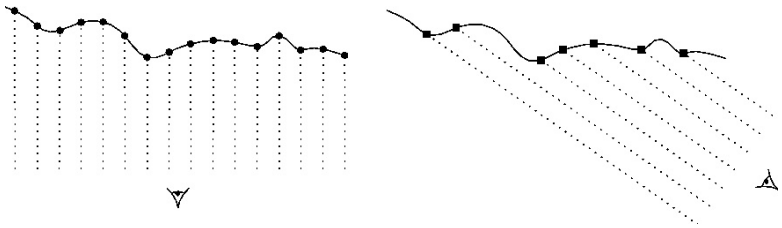
Final surface will be weighted combination of range images.

Weights are assigned at each vertex to:

- ***Favor views with higher sampling rates***
- ***Encourage smooth blends between range images***

Weights for sampling rates

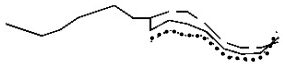
Sampling rate over the surface is highest when view direction is parallel to surface normal.



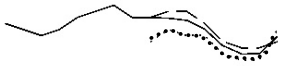
Weights for smooth blends

To assure smooth blends, weights are forced to taper in the vicinity of boundaries:

Two range surfaces



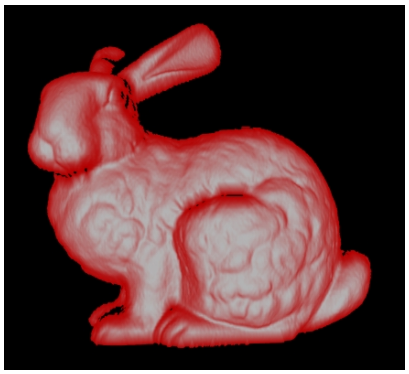
After unweighted blending



After weighted blending

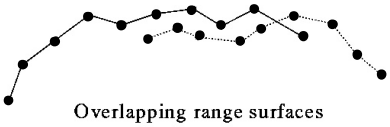
Example

Range surface

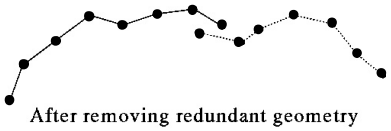


Confidence rendering

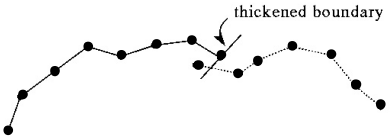
Redundancy removal and zippering



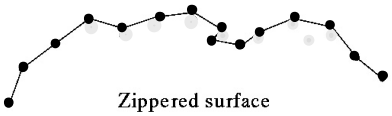
Overlapping range surfaces



After removing redundant geometry

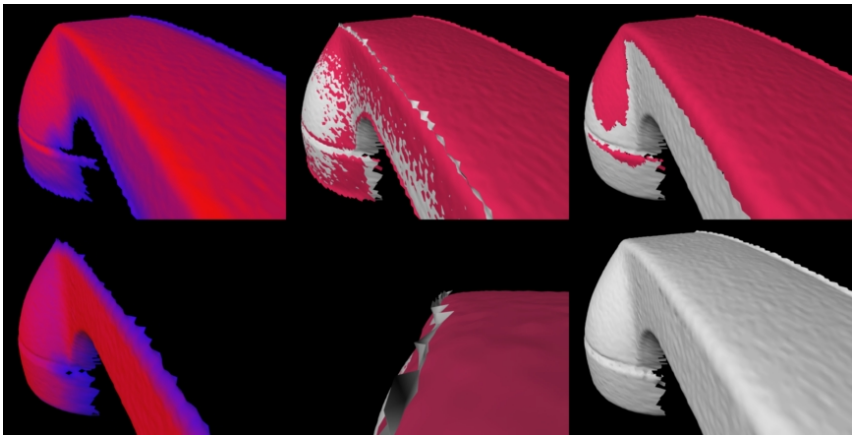


Thickening the mesh boundary for clipping

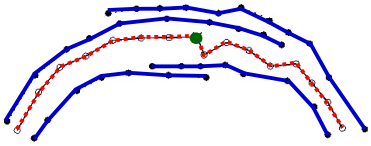


Zippered surface

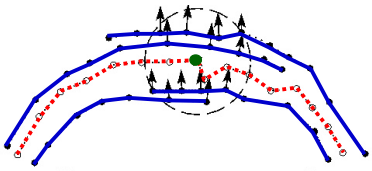
Example



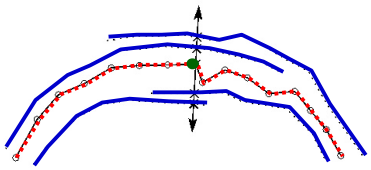
Consensus geometry



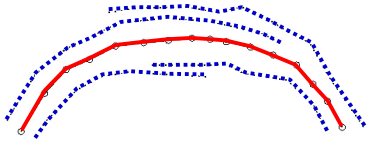
Zippered geometry + range surfaces



Compute consensus normal

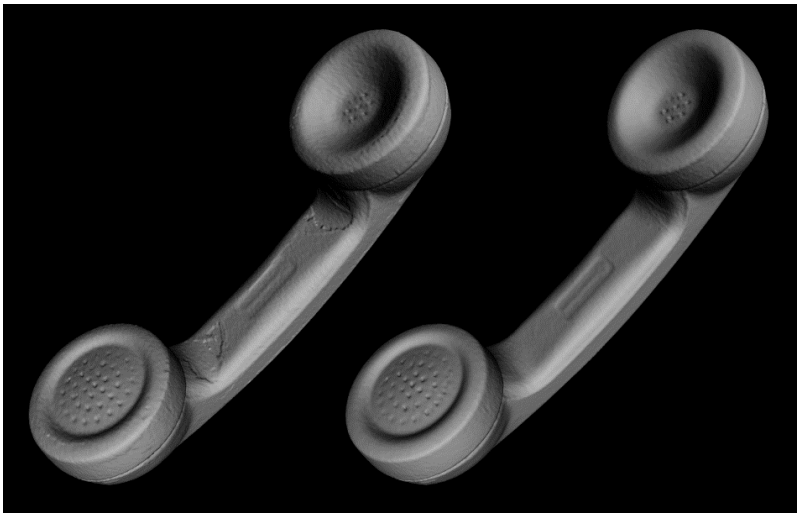


Find vertex positions on range surfaces by intersection with consensus normal



Compute weighted average of vertex positions

Example



Volumetrically combining range images

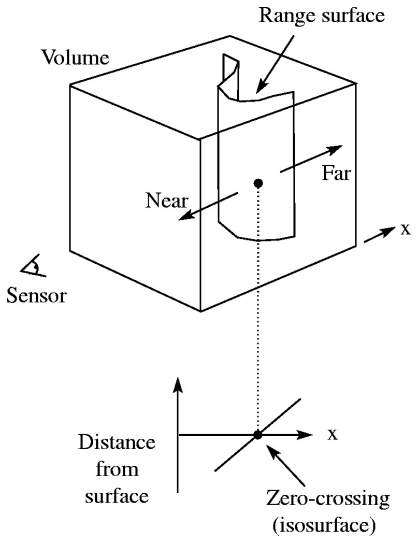
Combining the meshes volumetrically can overcome difficulties of stitching polygon meshes.

Here we describe the method of [Curless96].

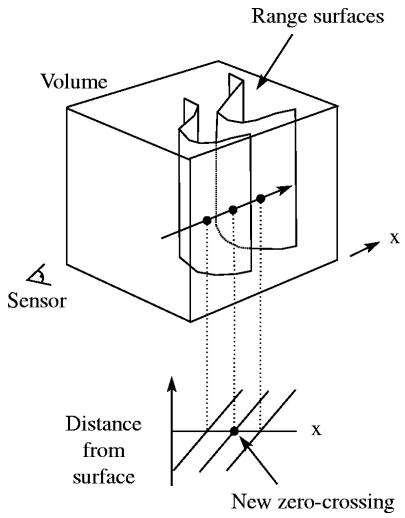
Overview:

- ***Convert range images to signed distance functions***
- ***Combine signed distance functions***
- ***Carve away empty space***
- ***Extract hole-free isosurface***

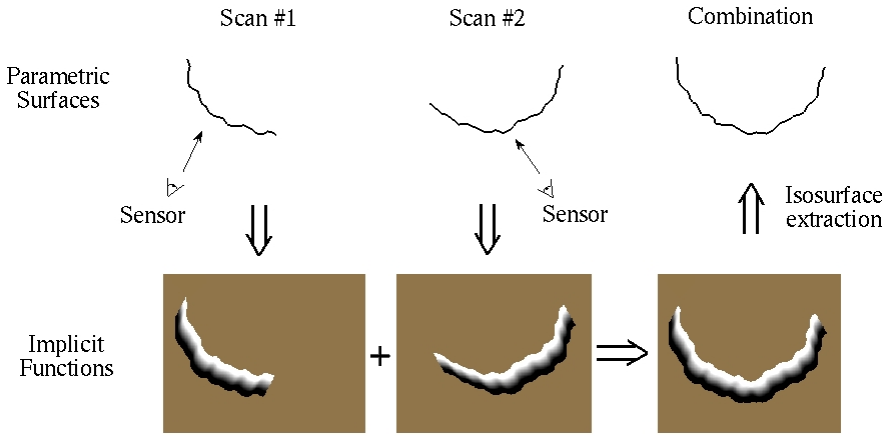
Signed distance function



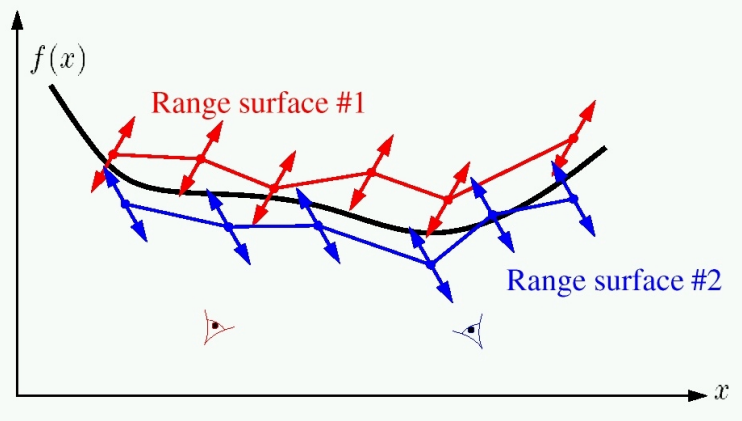
Combining signed distance functions



Merging surfaces in 2D



Least squares solution



Least squares solution

$$E(f) = \sum_{i=1}^{N} \int d_i^2(x, f)\,dx$$

Error per point

Error per range surface

Finding the f(x) that minimizes E yields the optimal surface.

This f(x) is exactly the zero-crossing of the combined signed distance functions.

Hole filling

We have presented an algorithm that reconstructs the observed surface. Unseen portions appear as holes in the reconstruction.

A hole-free mesh is useful for:

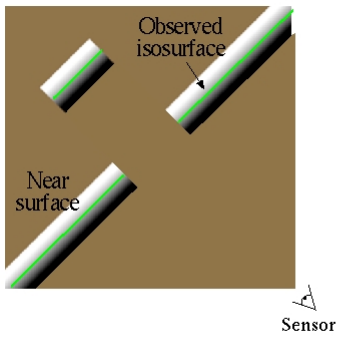
- ***Fitting surfaces to meshes***
- ***Manufacturing models (e.g., stereolithography)***
- ***Aesthetic renderings***

Hole filling

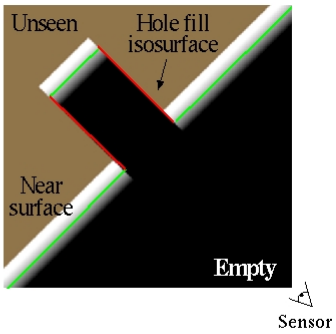
We can fill holes in the polygonal model directly, but such methods:

- ***are hard to make robust***
- ***do not use all available information***

Space carving

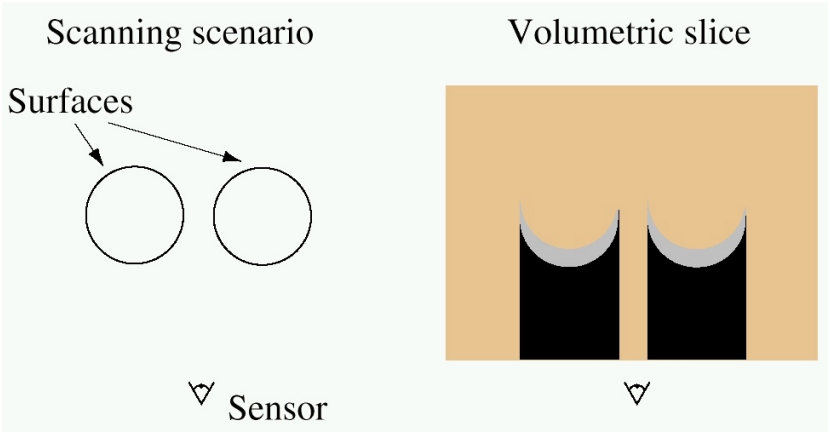


Without space carving

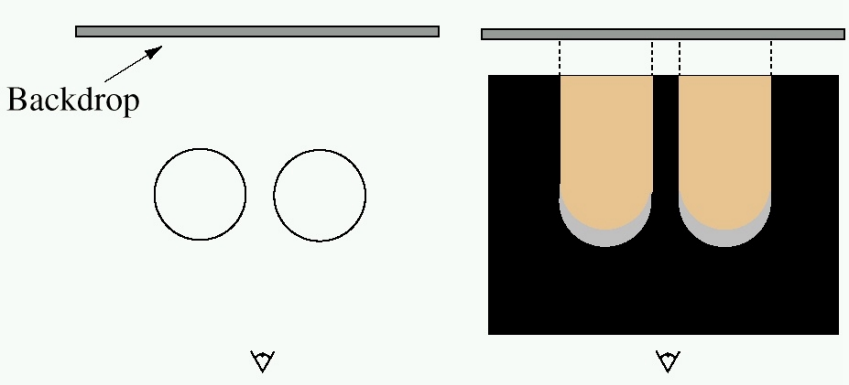


With space carving

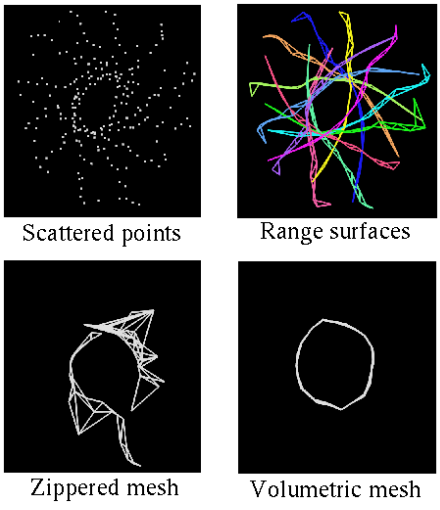
Carving *without* a backdrop



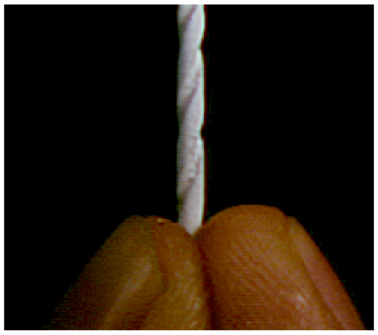
Carving *with* a backdrop



Merging 12 views of a drill bit



Merging 12 views of a drill bit

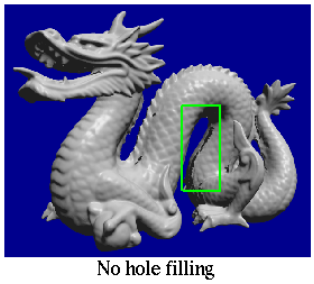


Photograph of painted drill bit

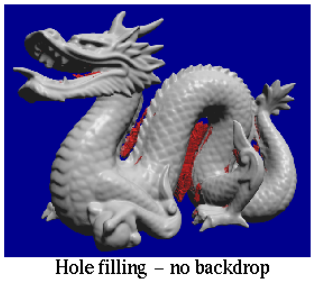
Zippered mesh

Volumetric mesh

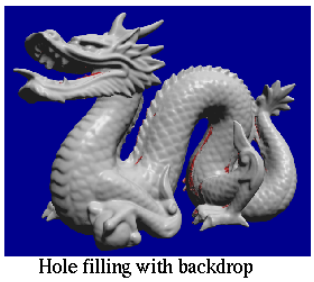
Dragon model



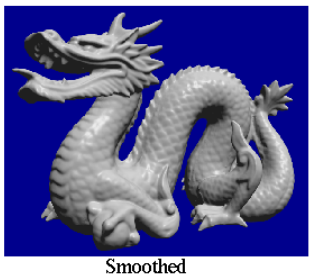
No hole filling



Hole filling – no backdrop

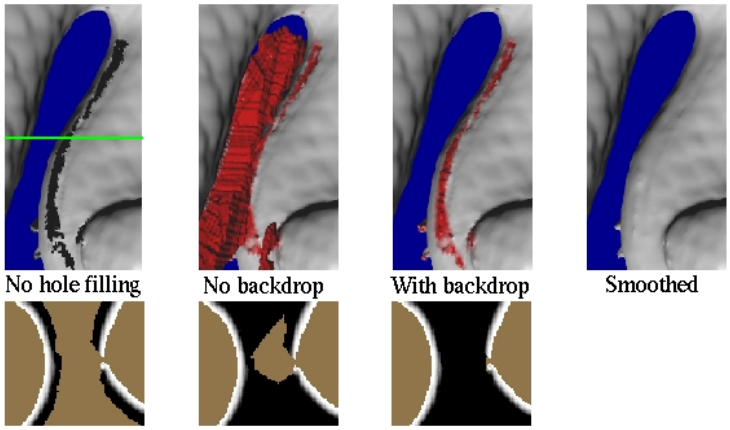


Hole filling with backdrop

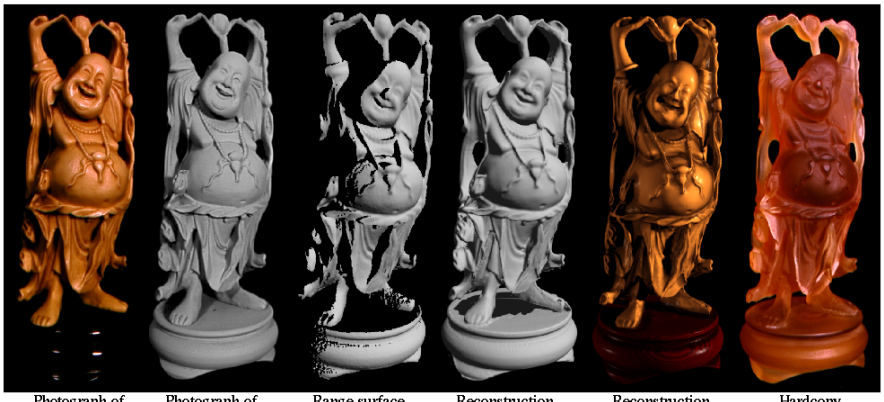


Smoothed

Dragon model



Happy Buddha



Modeling appearance

When describing appearance capture, we distinguish fixed from variable lighting.

Fixed lighting yields samples of the radiance function over the surface.

This radiance function can be re-rendered using methods such as lumigraph rendering or view-dependent texture mapping.

BRDF modeling

To re-render under new lighting conditions, we must model the BRDF.

Modeling the BRDF accurately is hard:

- ***BRDF is 4D in general.***
- ***Interreflections require solving an inverse rendering problem.***

Simplifications:

- ***Assume no interreflections***
- ***Assume a reflectance model with few parameters***

BRDF modeling

[Sato97] assume no interreflections and a Torrance-Sparrow BRDF model.

Procedure:

- ***Extract diffuse term where there are no specular highlights***
- ***Compute specular term at the specular highlights***
- ***Interpolate specular term over the surface***

BRDF modeling

Some researchers have modeled the impact of interreflections.

[Nayar91] assumes diffuse reflectance and extracts shape and reflectance from photometric stereo.

More recently, [Yu99] has demonstrated a method that computes diffuse and specular terms given geometry, even in the presence of interreflections.

Bibliography

Amenta, N., Bern, M., and Kamvysselis, M., "A new voronoi-based surface reconstructino algorithm," SIGGRAPH '98. Orlando, FL, USA, 19-24 July 1992. p. 415-421.

Bajaj, C.L., Bernardini, F. and Xu, G, "Automatic reconstruction of surfaces and scalar fields from 3D scans," SIGGRAPH '95 (Los Angeles, CA, Aug. 6-11, 1995, ACM Press, pp. 109--118.

Besl, P.J. and McKay, H.D., "A method for registration of 3-D shapes," IEEE Transactions on Pattern Analysis and Machine Intelligence, Feb. 1992, 2(4), pp. 239--256.

Boissonnat, J.-D, "Geometric Structures for Three-Dimensional Shape Representation," ACM Transactions on Graphics, October, 1994, 3(4), pp. 266-286.

Chen, Y. and Medioni, G., "Object modeling by registration of multiple range images," Image and Vision Computing, 10(3), April, 1992, pp. 145-155.

Curless, B. and Levoy, M., "A volumetric method for building complex models from range images." In Proceedings of SIGGRAPH '96, pp. 303-312. ACM Press, August 1996.

Edelsbrunner, H., Mucke, E.P. "Three-dimensional alpha shapes." ACM Transactions on Graphics (Jan. 1994) vol.13, no.1, pp. 43-72.

Hilton, A., Stoddart, A.J., Illingworth, J., and Windeatt, T, "Reliable Surface Reconstruction from Multiple Range Images," Fourth European Conference on Computer Vision, April 1996, vol. 1, pp. 117-126.

Bibliography

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., "Surface reconstruction from unorganized points." SIGGRAPH '92. Chicago, IL, USA, 26-31 July 1992. p. 71-8.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., "Mesh optimization." SIGGRAPH '93. Anaheim, CA, USA, 1-6 Aug. 1993, pp. 19-26.

Nayar, S.K., Ikeuchi, K., Kanade, T., "Recovering shape in the presence of interreflections." 1991 IEEE International Conference on Robotics and Automation, pp. 1814-19.

Sato, Y., Wheeler, M.D., Ikeuchi, K., "Object shape and reflectance modeling from observation." SIGGRAPH '97, p.379-387.

Soucy, M. and Laurendeau, D., "Multi-resolution surface modeling from multiple range views," Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, held in Champaign, IL, USA, 15-18 June 1992, pp. 348-353.

Turk, G. and Levoy, M., "Zippered polygon meshes from range images." In Proceedings of SIGGRAPH '94, pp. 311-318. ACM Press, July 1994.

Yu, Y., Debevec, P., and Malik, J., "Inverse global illumination: recovering reflectance models of real scenes from photographs," to appear in SIGGRAPH '99.

Surface Reconstruction from Unorganized Points

Hugues Hoppe* Tony DeRose* Tom Duchamp†
John McDonald‡ Werner Stuetzle‡

University of Washington
Seattle, WA 98195

Abstract

We describe and demonstrate an algorithm that takes as input an unorganized set of points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^3$ on or near an unknown manifold M, and produces as output a simplicial surface that approximates M. Neither the topology, the presence of boundaries, nor the geometry of M are assumed to be known in advance — all are inferred automatically from the data. This problem naturally arises in a variety of practical situations such as range scanning an object from multiple view points, recovery of biological shapes from two-dimensional slices, and interactive surface sketching.

CR Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling.

Additional Keywords: Geometric Modeling, Surface Fitting, Three-Dimensional Shape Recovery, Range Data Analysis.

1 Introduction

Broadly speaking, the class of problems we are interested in can be stated as follows: Given partial information of an unknown surface, construct, to the extent possible, a compact representation of the surface. Reconstruction problems of this sort occur in diverse scientific and engineering application domains, including:

- *Surfaces from range data:* The data produced by laser range scanning systems is typically a rectangular grid of distances from the sensor to the object being scanned. If the sensor and object are fixed, only objects that are "point viewable" can be fully digitized. More sophisticated systems, such as those produced by Cyberware Laboratory, Inc., are capable of digitizing cylindrical objects by rotating either the sensor or the object. However, the scanning of topologically more complex objects, including those as simple as a coffee cup with a handle (a surface of genus 1), or the object depicted in Figure 1a (a surface of genus 3), cannot be accomplished by either of these methods. To adequately scan these objects, multiple view points must be used. Merging the data generated from multiple view points to reconstruct a polyhedral surface representation is a non-trivial task [11].

- *Surfaces from contours:* In many medical studies it is common to slice biological specimens into thin layers with a microtome. The outlines of the structures of interest are then digitized to create a stack of contours. The problem is to reconstruct the three-dimensional structures from the stacks of two-dimensional contours. Although this problem has received a good deal of attention, there remain severe limitations with current methods. Perhaps foremost among these is the difficulty of automatically dealing with branching structures [3, 12].

- *Interactive surface sketching:* A number of researchers, including Schneider [21] and Eisenman [6], have investigated the creation of curves in $\mathbb{R}^2$ by tracing the path of a stylus or mouse as the user sketches the desired shape. Sachs et al. [19] describe a system, called 3-Draw, that permits the creation of free-form curves in $\mathbb{R}^3$ by recording the motion of a stylus fitted with a Polhemus sensor. This can be extended to the design of free-form surfaces by ignoring the order in which positions are recorded, allowing the user to move the stylus arbitrarily back and forth over the surface. The problem is then to construct a surface representation faithful to the unordered collection of points.

Reconstruction algorithms addressing these problems have typically been crafted on a case by case basis to exploit partial structure in the data. For instance, algorithms solving the surface from contours problem make heavy use of the fact that data are organized into contours (i.e., closed polygons), and that the contours lie in parallel planes. Similarly, specialized algorithms to reconstruct surfaces from multiple view point range data might exploit the adjacency relationship of the data points within each view.

In contrast, our approach is to pose a unifying general problem that does not assume any structure on the data points. This approach has both theoretical and practical merit. On the theoretical side, abstracting to a general problem often sheds light on the truly critical aspects of the problem. On the practical side, a single algorithm that solves the general problem can be used to solve any specific problem instance.

*Department of Computer Science and Engineering, FR-35
†Department of Mathematics, GN-50
‡Department of Statistics, GN-22

This work was supported in part by Bellcore, the Xerox Corporation, IBM, Hewlett-Packard, the Digital Equipment Corporation, the Department of Energy under grant DE-FG06-85-ER25006, the National Library of Medicine under grant NIH LM-04174, and the National Science Foundation under grants CCR-8957323 and DMS-9103002.

1.1 Terminology

By a *surface* we mean a "compact, connected, orientable two-dimensional manifold, possibly with boundary, embedded in $\mathbb{R}^3$" (cf. O'Neill [17]). A surface without boundary will be called a *closed surface*. If we want to emphasize that a surface possesses a non-empty boundary, we will call it a *bordered surface*. A piecewise linear surface with triangular faces will be referred to as a *simplicial surface*. We use $\|\mathbf{x}\|$ to denote the Euclidean length of a vector $\mathbf{x}$, and we use $d(X, Y)$ to denote the Hausdorff distance between the sets of points X and Y (the Hausdorff distance is simply the distance between the two closest points of X and Y).

Let $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be sampled data points on or near an unknown surface M (see Figure 1b). To capture the error in most sampling processes, we assume that each of the points $\mathbf{x}_i \in X$ is of the form $\mathbf{x}_i = \mathbf{y}_i + \mathbf{e}_i$, where $\mathbf{y}_i \in M$ is a point on the unknown surface and $\mathbf{e}_i \in \mathbb{R}^3$ is an error vector. We call such a sample X *δ-noisy* if $\|\mathbf{e}_i\| \leq \delta$ for all i. A value for δ can be estimated in most applications (e.g., the accuracy of the laser scanner). Features of M that are small compared to δ will obviously not be recoverable.

It is also impossible to recover features of M in regions where insufficient sampling has occurred. In particular, if M is a bordered surface, such as a sphere with a disc removed, it is impossible to distinguish holes in the sample from holes in the surface. To capture the intuitive notion of sampling density we need to make another definition: Let $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\} \subset M$ be a (noiseless) sample of a surface M. The sample Y is said to be *ρ-dense* if any sphere with radius ρ and center in M contains at least one sample point in Y. A δ-noisy sample $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^3$ of a surface M is said to be ρ-dense if there exists a noiseless ρ-dense sample $\{\mathbf{y}_1, \ldots, \mathbf{y}_n\} \subset M$ such that $\mathbf{x}_i = \mathbf{y}_i + \mathbf{e}_i$, $\|\mathbf{e}_i\| \leq \delta$, $i = 1, \ldots, n$.

1.2 Problem Statement

The goal of *surface reconstruction* is to determine a surface M' (see Figure 2f) that approximates an unknown surface M (Figure 1a), using a sample X (Figure 1b) and information about the sampling process, for example, bounds on the noise magnitude δ and the sampling density ρ.

We are currently working to develop conditions on the original surface M and the sample X that are sufficient to allow M to be reliably reconstructed. As that work is still preliminary, we are unable to give guarantees for the algorithm presented here. However, the algorithm has worked well in practice where the results can be compared to the original surface (see Section 4).

2 Related Work

2.1 Surface Reconstruction

Surface reconstruction methods can be classified according to the way in which they represent the reconstructed surface.

Implicit reconstruction methods attempt to find a smooth function $f : \mathbb{R}^3 \to \mathbb{R}$ such that $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is close to the zero set $Z(f)$. They differ with respect to the form of f and the measure of closeness. Pratt [18] and Taubin [25] minimize the sum of squared Hausdorff distances from the data points to the zero set of a polynomial in three variables. Muraki [15] takes f to be a linear combination of three-dimensional Gaussian kernels with different means and spreads. His goodness-of-fit function measures how close the values of f at the data points are to zero, and how well the unit normals to the zero set of f match the normals estimated from the data. Moore and Warren [13] fit a piecewise polynomial recursively and then enforce continuity using a technique they call *free form blending*.

In contrast to implicit reconstruction techniques, parametric reconstruction techniques represent the reconstructed surface as a topological embedding $f(\Lambda)$ of a 2-dimensional parameter domain Λ into $\mathbb{R}^3$. Previous work has concentrated on domain spaces with simple topology, i.e. the plane and the sphere. Hastie and Stuetzle [9] and Vemuri [26, 27] discuss reconstruction of surfaces by a topological embedding $f(\Lambda)$ of a planar region Λ into $\mathbb{R}^3$. Schudy and Ballard [22, 23] and Brinkley [4] consider the reconstruction of surfaces that are slightly deformed spheres, and thus choose Λ to be a sphere. Sclaroff and Pentland [24] describe a hybrid implicit/parametric method for fitting a deformed sphere to a set of points using deformations of a superquadric.

Compared to the techniques mentioned above, our method has several advantages:

- It requires only an unorganized collection of points on or near the surface. No additional information is needed (such as normal information used by Muraki's method).
- Unlike the parametric methods mentioned above, it can reconstruct surfaces of arbitrary topology.
- Unlike previously suggested implicit methods, it deals with boundaries in a natural way, and it does not generate spurious surface components not supported by the data.

2.2 Surface Reconstruction vs Function Reconstruction

Terms like "surface fitting" appear in reference to two distinct classes of problems: surface reconstruction and function reconstruction. The goal of surface reconstruction was stated earlier. The goal of function reconstruction may be stated as follows: Given a surface M, a set $\{\mathbf{x}_i \in M\}$, and a set $\{y_i \in \mathbb{R}\}$, determine a function $f : M \to \mathbb{R}$, such that $f(\mathbf{x}_i) \approx y_i$.

The domain surface M is most commonly a plane embedded in $\mathbb{R}^3$, in which case the problem is a standard one considered in approximation theory. The case where M is a sphere has also been extensively treated (cf. [7]). Some recent work under the title *surfaces on surfaces* addresses the case when M is a general curved surface such as the skin of an airplane [16].

Function reconstruction methods can be used for surface reconstruction in simple, special cases, where the surface to be reconstructed is, roughly speaking, the graph of a function over a *known* surface M. It is important to recognize just how limited these special cases are — for example, not every surface homeomorphic to a sphere is the graph of a function over the sphere. The point we want to make is that function reconstruction must not be misconstrued to solve the general surface reconstruction problem.

3 A Description of the Algorithm

3.1 Overview

Our surface reconstruction algorithm consists of two stages. In the first stage we define a function $f : D \to \mathbb{R}$, where $D \subset \mathbb{R}^3$ is a region near the data, such that f estimates the signed geometric distance to the unknown surface M. The zero set $Z(f)$ is our estimate for M. In the second stage we use a contouring algorithm to approximate $Z(f)$ by a simplicial surface.

Although the *unsigned* distance function $|f|$ would be easier to estimate, zero is not a regular value of $|f|$. Zero is, however, a regular value of f, and the implicit function theorem thus guarantees that our approximation $Z(f)$ is a manifold.

The key ingredient to defining the signed distance function is to associate an oriented plane with each of the data points. These

tangent planes serve as local linear approximations to the surface. Although the construction of the tangent planes is relatively simple, the selection of their orientations so as to define a globally consistent orientation for the surface is one of the major obstacles facing the algorithm. As indicated in Figure 2b, the tangent planes do not directly define the surface, since their union may have a complicated non-manifold structure. Rather, we use the tangent planes to define the signed distance function to the surface. An example of the simplicial surface obtained by contouring the zero set of the signed distance function is shown in Figure 2e. The next several sections develop in more detail the successive steps of the algorithm.

3.2 Tangent Plane Estimation

The first step toward defining a signed distance function is to compute an oriented tangent plane for each data point. The tangent plane $Tp(\mathbf{x}_i)$ associated with the data point $\mathbf{x}_i$ is represented as a point $\mathbf{o}_i$, called the center, together with a unit normal vector $\hat{\mathbf{n}}_i$. The signed distance of an arbitrary point $\mathbf{p} \in \mathbb{R}^3$ to $Tp(\mathbf{x}_i)$ is defined to be $\text{dist}_i(\mathbf{p}) = (\mathbf{p} - \mathbf{o}_i) \cdot \hat{\mathbf{n}}_i$. The center and normal for $Tp(\mathbf{x}_i)$ are determined by gathering together the k points of X nearest to $\mathbf{x}_i$; this set is denoted by $Nbhd(\mathbf{x}_i)$ and is called the k-neighborhood of $\mathbf{x}_i$. (We currently assume k to be a user-specified parameter, although in Section 5 we propose a method for determining k automatically.) The center and unit normal are computed so that the plane $\{\text{dist}_i(\mathbf{p}) = 0\}$ is the least squares best fitting plane to $Nbhd(\mathbf{x}_i)$. That is, the center $\mathbf{o}_i$ is taken to be the centroid of $Nbhd(\mathbf{x}_i)$, and the normal $\hat{\mathbf{n}}_i$ is determined using principal component analysis. To compute $\hat{\mathbf{n}}_i$, the covariance matrix of $Nbhd(\mathbf{x}_i)$ is formed. This is the symmetric 3×3 positive semi-definite matrix

$$\text{CV} = \sum_{\mathbf{y} \in Nbhd(\mathbf{x}_i)} (\mathbf{y} - \mathbf{o}_i) \otimes (\mathbf{y} - \mathbf{o}_i)$$

where $\otimes$ denotes the outer product vector operator[1]. If $\lambda_i^1 \geq \lambda_i^2 \geq \lambda_i^3$ denote the eigenvalues of CV associated with unit eigenvectors $\hat{\mathbf{v}}_i^1, \hat{\mathbf{v}}_i^2, \hat{\mathbf{v}}_i^3$, respectively, we choose $\hat{\mathbf{n}}_i$ to be either $\hat{\mathbf{v}}_i^3$ or $-\hat{\mathbf{v}}_i^3$. The selection determines the orientation of the tangent plane, and it must be done so that nearby planes are "consistently oriented".

3.3 Consistent Tangent Plane Orientation

Suppose two data points $\mathbf{x}_i, \mathbf{x}_j \in X$ are geometrically close. Ideally, when the data is dense and the surface is smooth, the corresponding tangent planes $Tp(\mathbf{x}_i) = (\mathbf{o}_i, \hat{\mathbf{n}}_i)$ and $Tp(\mathbf{x}_j) = (\mathbf{o}_j, \hat{\mathbf{n}}_j)$ are nearly parallel, i.e. $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j \approx \pm 1$. If the planes are consistently oriented, then $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j \approx +1$; otherwise, either $\hat{\mathbf{n}}_i$ or $\hat{\mathbf{n}}_j$ should be flipped. The difficulty in finding a consistent global orientation is that this condition should hold between all pairs of "sufficiently close" data points.

We can model the problem as graph optimization. The graph contains one node N_i per tangent plane $Tp(\mathbf{x}_i)$, with an edge (i, j) between N_i and N_j if the tangent plane centers $\mathbf{o}_i$ and $\mathbf{o}_j$ are sufficiently close (we will be more precise about what we mean by sufficiently close shortly). The cost on edge (i, j) encodes the degree to which N_i and N_j are consistently oriented and is taken to be $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j$. The problem is then to select orientations for the tangent planes so as to maximize the total cost of the graph. Unfortunately, this problem can be shown to be NP-hard via a reduction to MAX-CUT [8]. To efficiently solve the orientation problem we must therefore resort to an approximation algorithm.

Before describing the approximation algorithm we use, we must decide when a pair of nodes are to be connected in the graph. Since the surface is assumed to consist of a single connected component, the graph should be connected. A simple connected graph for a set of points that tends to connect neighbors is the Euclidean Minimum Spanning Tree (EMST). However, the EMST over the tangent plane centers $\{\mathbf{o}_1, \ldots, \mathbf{o}_n\}$ (Figure 1c) is not sufficiently dense in edges to serve our purposes. We therefore enrich it by adding a number of edges to it. Specifically, we add the edge (i, j) if either $\mathbf{o}_i$ is in the k-neighborhood of $\mathbf{o}_j$, or $\mathbf{o}_j$ is in the k-neighborhood of $\mathbf{o}_i$ (where k-neighborhood is defined over $\{\mathbf{o}_1, \ldots, \mathbf{o}_n\}$ as it was for X). The resulting graph (Figure 1d), called the *Riemannian Graph*, is thus constructed to be a connected graph that encodes geometric proximity of the tangent plane centers.

A relatively simple-minded algorithm to orient the planes would be to arbitrarily choose an orientation for some plane, then "propagate" the orientation to neighboring planes in the Riemannian Graph. In practice, we found that the order in which the orientation is propagated is important. Figure 3b shows what may result when propagating orientation solely on the basis of geometric proximity; a correct reconstruction is shown in Figure 3c. Intuitively, we would like to choose an order of propagation that favors propagation from $Tp(\mathbf{x}_i)$ to $Tp(\mathbf{x}_j)$ if the unoriented planes are nearly parallel. This can be accomplished by assigning to each edge (i, j) in the Riemannian Graph the cost $1 - |\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j|$. In addition to being non-negative, this assignment has the property that a cost is small if the unoriented tangent planes are nearly parallel. A favorable propagation order can therefore be achieved by traversing the *minimal spanning tree* (MST) of the resulting graph. This order is advantageous because it tends to propagate orientation along directions of low curvature in the data, thereby largely avoiding ambiguous situations encountered when trying to propagate orientation across sharp edges (as at the tip of the cat's ears in Figure 3b). In the MST shown in Figure 2a, the edges are colored according to their cost, with the brightly colored edges corresponding to regions of high variation (where $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j$ is somewhat less than 1).

To assign orientation to an initial plane, the unit normal of the plane whose center has the largest z coordinate is forced to point toward the $+z$ axis. Then, rooting the tree at this initial node, we traverse the tree in depth-first order, assigning each plane an orientation that is consistent with that of its parent. That is, if during traversal, the current plane $Tp(\mathbf{x}_i)$ has been assigned the orientation $\hat{\mathbf{n}}_i$ and $Tp(\mathbf{x}_j)$ is the next plane to be visited, then $\hat{\mathbf{n}}_j$ is replaced with $-\hat{\mathbf{n}}_j$ if $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j < 0$.

This orientation algorithm has been used in all our examples and has produced correct orientations in all the cases we have run. The resulting oriented tangent planes are represented as shaded rectangles in Figure 2b.

3.4 Signed Distance Function

The signed distance $f(\mathbf{p})$ from an arbitrary point $\mathbf{p} \in \mathbb{R}^3$ to a known surface M is the distance between $\mathbf{p}$ and the closest point $\mathbf{z} \in M$, multiplied by ± 1, depending on which side of the surface $\mathbf{p}$ lies. In reality M is not known, but we can mimic this procedure using the oriented tangent planes as follows. First, we find the tangent plane $Tp(\mathbf{x}_i)$ whose center $\mathbf{o}_i$ is closest to $\mathbf{p}$. This tangent plane is a local linear approximation to M, so we take the signed distance $f(\mathbf{p})$ to M to be the signed distance between $\mathbf{p}$ and its projection $\mathbf{z}$ onto $Tp(\mathbf{x}_i)$; that is,

$$f(\mathbf{p}) = \text{dist}_i(\mathbf{p}) = (\mathbf{p} - \mathbf{o}_i) \cdot \hat{\mathbf{n}}_i.$$

If M is known not to have boundaries, this simple rule works well. However, the rule must be extended to accommodate surfaces that might have boundaries. Recall that the set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is assumed to be a ρ-dense, δ-noisy sample of M. If there was no noise, we could deduce that a point $\mathbf{z}$ with $d(\mathbf{z}, X) > \rho$ cannot be

[1] If $\mathbf{a}$ and $\mathbf{b}$ have components a_i and b_j respectively, then the matrix $\mathbf{a} \otimes \mathbf{b}$ has $a_i b_j$ as its ij-th entry.

a point of M since that would violate X being ρ-dense. Intuitively, the sample points do not leave holes of radius larger than ρ. If the sample is δ-noisy, the radius of the holes may increase, but by no more than δ. We therefore conclude that a point $\mathbf{z}$ cannot be a point of M if $d(\mathbf{z}, X) > \rho + \delta$. If the projection $\mathbf{z}$ of $\mathbf{p}$ onto the closest tangent plane has $d(\mathbf{z}, X) > \rho + \delta$, we take $f(\mathbf{p})$ to be undefined. Undefined values are used by the contouring algorithm of Section 3.5 to identify boundaries.

Stated procedurally, our signed distance function is defined as:

$i \leftarrow$ index of tangent plane whose center is closest to $\mathbf{p}$

{ *Compute* $\mathbf{z}$ *as the projection of* $\mathbf{p}$ *onto* $Tp(\mathbf{x}_i)$ }
$\mathbf{z} \leftarrow \mathbf{o}_i - ((\mathbf{p} - \mathbf{o}_i) \cdot \hat{\mathbf{n}}_i)\, \hat{\mathbf{n}}_i$

if $d(\mathbf{z}, X) < \rho + \delta$ **then**
    $f(\mathbf{p}) \leftarrow (\mathbf{p} - \mathbf{o}_i) \cdot \hat{\mathbf{n}}_i$ $\quad \{= \pm\|\mathbf{p} - \mathbf{z}\|\}$
else
    $f(\mathbf{p}) \leftarrow$ **undefined**
endif

The simple approach outlined above creates a zero set $Z(f)$ that is piecewise linear but contains discontinuities. The discontinuities result from the implicit partitioning of space into regions within which a single tangent plane is used to define the signed distance function. (These regions are in fact the Voronoi regions associated with the centers $\mathbf{o}_i$.) Fortunately, the discontinuities do not adversely affect our algorithm. The contouring algorithm discussed in the next section will discretely sample the function f over a portion of a 3-dimensional grid near the data and reconstruct a *continuous* piecewise linear approximation to $Z(f)$.

3.5 Contour Tracing

Contour tracing, the extraction of an isosurface from a scalar function, is a well-studied problem [1, 5, 28]. We chose to implement a variation of the marching cubes algorithm (cf. [1]) that samples the function at the vertices of a cubical lattice and finds the contour intersections within tetrahedral decompositions of the cubical cells.

To accurately estimate boundaries, the cube size should be set so that edges are of length less than $\rho + \delta$. In practice we have often found it convenient to set the cube size somewhat larger than this value, simply to increase the speed of execution and to reduce the number of triangular facets generated.

The algorithm only visits cubes that intersect the zero set by pushing onto a queue only the appropriate neighboring cubes (Figure 2c). In this way, the signed distance function f is evaluated only at points close to the data. Figure 2d illustrates the signed distance function by showing line segments between the query points $\mathbf{p}$ (at the cube vertices) and their associated projected points $\mathbf{z}$. As suggested in Section 3.4, no intersection is reported within a cube if the signed distance function is undefined at any vertex of the cube, thereby giving rise to boundaries in the simplicial surface.

The resulting simplicial surface can contain triangles with arbitrarily poor aspect ratio (Figure 2e). We alleviate this problem using a post-processing procedure that collapses edges in the surface using an aspect ratio criterion.[2] The final result is shown in Figure 2f. Alternatively, other contouring methods exist that can guarantee bounds on the triangle aspect ratio [14].

[2]The edges are kept in a priority queue; the criterion to minimize is the product of the edge length times the minimum inscribed radius of its two adjacent faces. Tests are also performed to ensure that edge collapses preserve the topological type of the surface.

4 Results

We have experimented with the reconstruction method on data sets obtained from several different sources. In all cases, any structure (including ordering) that might have been present in the point sets was discarded.

Meshes : Points were randomly sampled from a number of existing simplicial surfaces[3]. For instance, the mesh of Figure 3a was randomly sampled to yield 1000 unorganized points, and these in turn were used to reconstruct the surface in Figure 3c. This particular case illustrates the behavior of the method on a bordered surface (the cat has no base and is thus homeomorphic to a disc). The reconstructed knot (original mesh from Rob Scharein) of Figure 3d is an example of a surface with simple topology yet complex geometrical embedding.

Ray Traced Points : To simulate laser range imaging from multiple view points, CSG models were ray traced from multiple eye points. The ray tracer recorded the point of first intersection along each ray. Eight eye points (the vertices of a large cube centered at the object) were used to generate the point set of Figure 1b from the CSG object shown in Figure 1a. This is the point set used in Section 3 to illustrate the steps of the algorithm (Figures 1a-2f).

Range Images : The bust of Spock (Figure 3e) was reconstructed from points taken from an actual cylindrical range image (generated by Cyberware Laboratory, Inc.). Only 25% of the original points were used.

Contours : Points from 39 planar (horizontal) slices of the CT scan of a femur were combined together to obtain the surface of Figure 3f.

The algorithm's parameters are shown in the next table for each of the examples. The execution times were obtained on a 20 MIPS workstation. The parameter $\rho + \delta$ and the marching cube cell size are both expressed as a fraction of the object's size. The parameter $\rho + \delta$ is set to infinity for those surfaces that are known to be closed.

Object	n	k	$\rho + \delta$	cell size	time (seconds)
cat	1000	15	.06	1/30	19
knot	10000	20	∞	1/50	137
mechpart	4102	12	∞	1/40	54
spock	21760	8	.08	1/80	514
femur	18224	40	.06	1/50	2135

5 Discussion

5.1 Tangent Plane Approximation

The neighborhood $Nbhd(\mathbf{x}_i)$ of a data point $\mathbf{x}_i$ is defined to consist of its k nearest neighbors, where k is currently assumed to be an input parameter. In the case where the data contains little or no noise, k is not a critical parameter since the output has been empirically observed to be stable over a wide range of settings. However, it would be best if k could be selected automatically. Furthermore, allowing k to adapt locally would make less stringent the requirement that the data be uniformly distributed over the surface. To select and adapt k, the algorithm could incrementally gather points while monitoring the changing eigenvalues of the covariance matrix (see Section 3.2). For small values of k, data noise tends to dominate, the eigenvalues are similar, and the eigenvectors do not reveal the surface's true tangent plane. At the other extreme, as k becomes

[3]Discrete inverse transform sampling [10, page 469] on triangle area was used to select face indices from the mesh, and uniform sampling was used within the faces.

large, the k-neighborhoods become less localized and the surface curvature tends to increase the "thickness" λ_i^3 of the neighborhood. Another possible criterion is to compare λ_i^3 to some local or global estimate of data noise. Although we have done some initial experimentation in this direction, we have not yet fully examined these options.

If the data is obtained from range images, there exists some knowledge of surface orientation at each data point. Indeed, each data point is known to be visible from a particular viewing direction, so that, unless the surface incident angle is large, the point's tangent plane orientation can be inferred from that viewing direction. Our method could exploit this additional information in the tangent plane orientation step (Section 3.3) by augmenting the Riemannian Graph with an additional pseudo-node and n additional edges.

5.2 Algorithm Complexity

A spatial partitioning Abstract Data Type greatly improves performance of many of the subproblems discussed previously. The critical subproblems are (with their standard time complexity):

- EMST graph ($O(n^2)$)
- k-nearest neighbors to a given point ($O(n + k \log n)$)
- nearest tangent plane origin to a given point ($O(n)$)

Hierarchical spatial partitioning schemes such as octrees [20] and k-D trees [2] can be used to solve these problems more efficiently. However, the uniform sampling density assumed in our data allows simple spatial cubic partitioning to work efficiently. The axis-aligned bounding box of the points is partitioned by a cubical grid. Points are entered into sets corresponding to the cube to which they belong, and these sets are accessed through a hash table indexed by the cube indices. It is difficult to analyze the resulting improvements analytically, but, empirically, the time complexity of the above problems is effectively reduced by a factor of n, except for the k-nearest neighbors problem which becomes $O(k)$.

As a result of the spatial partitioning, the Riemannian Graph can be constructed in $O(nk)$ time. Because the Riemannian Graph has $O(n)$ edges (at most $n+nk$), the MST computation used in finding the best path on which to propagate orientation requires only $O(n \log n)$ time. Traversal of the MST is of course $O(n)$.

The time complexity of the contouring algorithm depends only on the number of cubes visited, since the evaluation of the signed distance function f at a point $\mathbf{p}$ can be done in constant time (the closest tangent plane origin $\mathbf{o}_i$ to $\mathbf{p}$ and the closest data point $\mathbf{x}_j$ to the projected point $\mathbf{z}$ can both be found in constant time with spatial partitioning).

6 Conclusions and Future Work

We have developed an algorithm to reconstruct a surface in three-dimensional space with or without boundary from a set of unorganized points scattered on or near the surface. The algorithm, based on the idea of determining the zero set of an estimated signed distance function, was demonstrated on data gathered from a variety of sources. It is capable of automatically inferring the topological type of the surface, including the presence of boundary curves.

The algorithm can, in principle, be extended to reconstruct manifolds of co-dimension one in spaces of arbitrary dimension; that is, to reconstruct $d-1$ dimensional manifolds in d dimensional space. Thus, essentially the same algorithm can be used to reconstruct curves in the plane or volumes in four-dimensional space.

The output of our reconstruction method produced the correct topology in all the examples. We are trying to develop formal guarantees on the correctness of the reconstruction, given constraints on the sample and the original surface. To further improve the geometric accuracy of the fit, and to reduce the space required to store the reconstruction, we envision using the output of our algorithm as the starting point for a subsequent spline surface fitting procedure. We are currently investigating such a method based on a nonlinear least squares approach using triangular Bézier surfaces.

References

[1] E. L. Allgower and P. H. Schmidt. An algorithm for piecewise linear approximation of an implicitly defined manifold. *SIAM Journal of Numerical Analysis*, 22:322–346, April 1985.

[2] J. L. Bentley. Multidimensional divide and conquer. *Comm. ACM*, 23(4):214–229, 1980.

[3] Y. Breseler, J. A. Fessler, and A. Macovski. A Bayesian approach to reconstruction from incomplete projections of a multiple object 3D domain. *IEEE Trans. Pat. Anal. Mach. Intell.*, 11(8):840–858, August 1989.

[4] James F. Brinkley. Knowledge-driven ultrasonic three-dimensional organ modeling. *IEEE Trans. Pat. Anal. Mach. Intell.*, 7(4):431–441, July 1985.

[5] David P. Dobkin, Silvio V. F. Levy, William P. Thurston, and Allan R. Wilks. Contour tracing by piecewise linear approximations. *ACM TOG*, 9(4):389–423, October 1990.

[6] John A. Eisenman. Graphical editing of composite bezier curves. Master's thesis, Department of Electrical Engineering and Computer Science, M.I.T., 1988.

[7] T.A. Foley. Interpolation to scattered data on a spherical domain. In M. Cox and J. Mason, editors, *Algorithms for Approximation II*, pages 303–310. Chapman and Hall, London, 1990.

[8] Michael R. Garey and David S. Johnson. *Computers and Intractability*. W. H. Freeman and Company, 1979.

[9] T. Hastie and W. Stuetzle. Principal curves. *JASA*, 84:502–516, 1989.

[10] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, Inc., second edition, 1991.

[11] Marshal L. Merriam. Experience with the cyberware 3D digitizer. In *NCGA Proceedings*, pages 125–133, March 1992.

[12] David Meyers, Shelly Skinner, and Kenneth Sloan. Surfaces from contours: The correspondence and branching problems. In *Proceedings of Graphics Interface '91*, pages 246–254, June 1991.

[13] Doug Moore and Joe Warren. Approximation of dense scattered data using algebraic surfaces. TR 90-135, Rice University, October 1990.

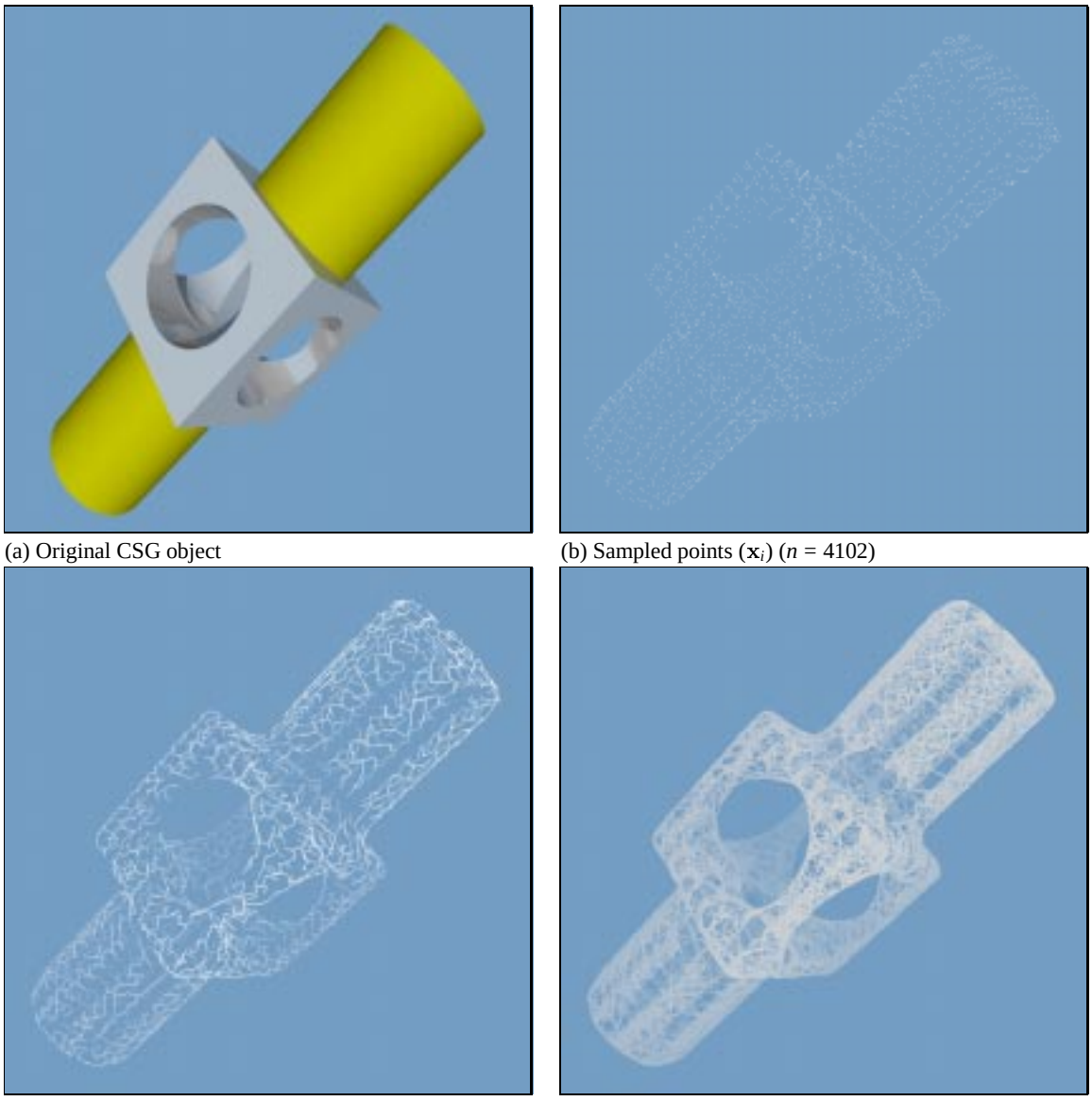
[14] Doug Moore and Joe Warren. Adaptive mesh generation ii: Packing solids. TR 90-139, Rice University, March 1991.

[15] Shigeru Muraki. Volumetric shape description of range data using "blobby model". *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):227–235, July 1991.

[16] Gregory M. Nielson, Thomas A. Foley, Bernd Hamann, and David Lane. Visualizing and modeling scattered multivariate data. *IEEE CG&A*, 11(3):47–55, May 1991.

[17] Barrett O'Neill. *Elementary Differential Geometry*. Academic Press, Orlando, Florida, 1966.

[18] Vaughan Pratt. Direct least-squares fitting of algebraic surfaces. *Computer Graphics (SIGGRAPH '87 Proceedings)*, 21(4):145–152, July 1987.



(a) Original CSG object (b) Sampled points ($\mathbf{x}_i$) (n = 4102)

(c) EMST of tangent plane centers $\mathbf{o}_i$ (d) Riemannian Graph over $\mathbf{o}_i$

Figure 1: Reconstruction of ray-traced CSG object (simulated multi-view range data).

[19] Emanuel Sachs, Andrew Roberts, and David Stoops. 3-Draw: A tool for designing 3D shapes. *IEEE Computer Graphics and Applications*, 11(6):18–26, November 1991.

[20] Hanan Samet. *Applications of Spatial Data Structures*. Addison-Wesley, 1990.

[21] Philip J. Schneider. Phoenix: An interactive curve design system based on the automatic fitting of hand-sketched curves. Master's thesis, Department of Computer Science, U. of Washington, 1988.

[22] R. B. Schudy and D. H. Ballard. Model detection of cardiac chambers in ultrasound images. Technical Report 12, Computer Science Department, University of Rochester, 1978.

[23] R. B. Schudy and D. H. Ballard. Towards an anatomical model of heart motion as seen in 4-d cardiac ultrasound data. In *Proceedings of the 6th Conference on Computer Applications in Radiology and Computer-Aided Analysis of Radiological Images*, 1979.

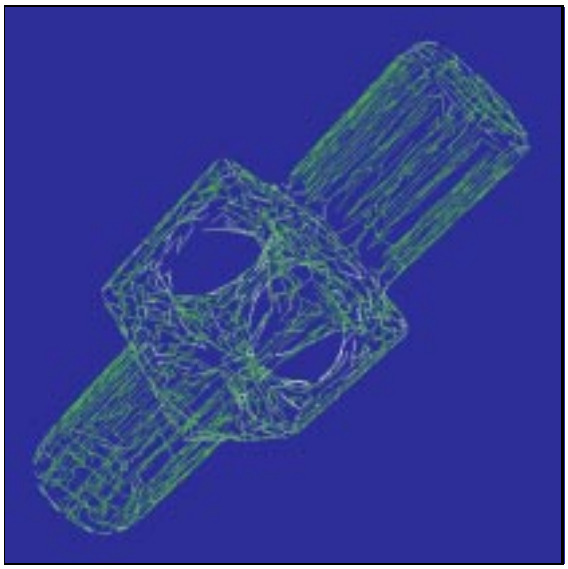
[24] Stan Sclaroff and Alex Pentland. Generalized implicit functions for computer graphics. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):247–250, July 1991.

[25] G. Taubin. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation. Technical Report LEMS-66, Division of Engineering, Brown University, 1990.

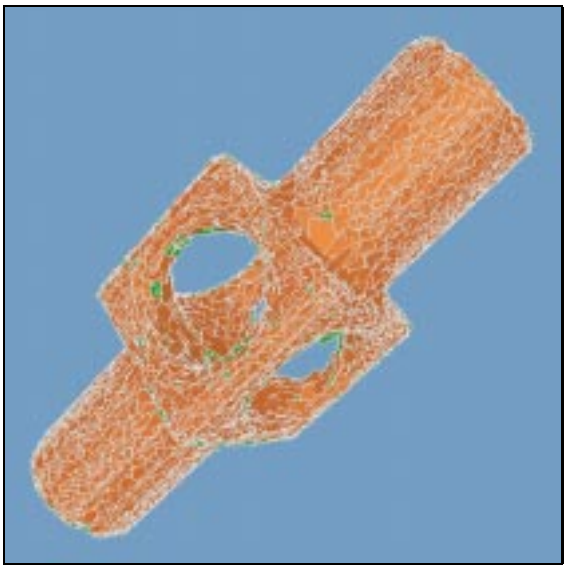
[26] B. C. Vemuri. *Representation and Recognition of Objects From Dense Range Maps*. PhD thesis, Department of Electrical and Computer Engineering, University of Texas at Austin, 1987.

[27] B. C. Vemuri, A. Mitiche, and J. K. Aggarwal. Curvature-based representation of objects from range data. *Image and Vision Computing*, 4(2):107–114, 1986.

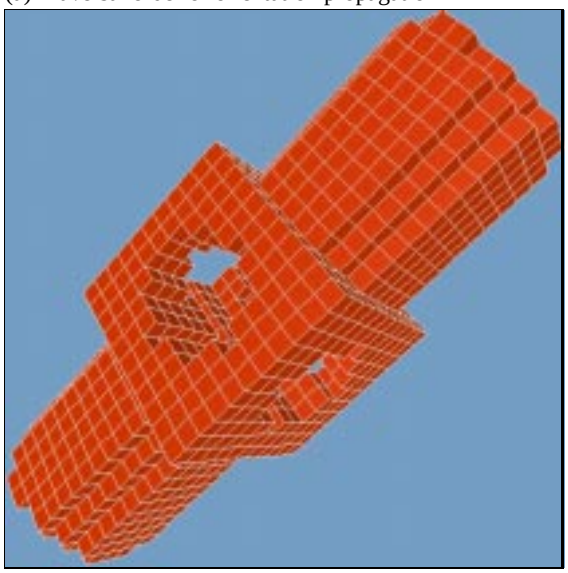
[28] G. Wyvill, C. McPheeters, and B. Wyvill. Data structures for soft objects. *The Visual Computer*, 2(4):227–234, August 1986.



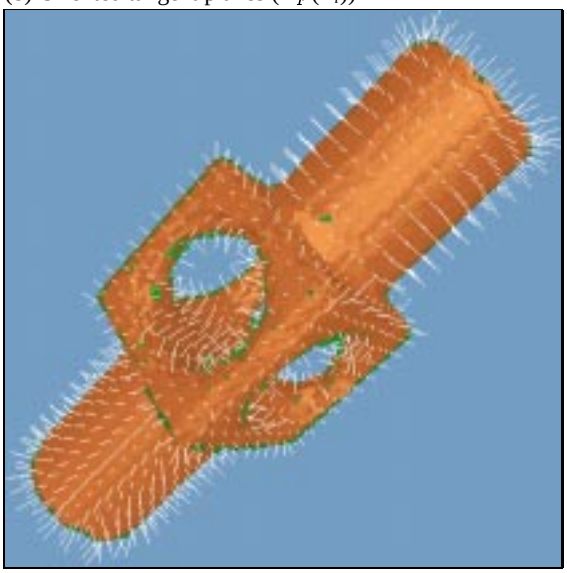
(a) Traversal order of orientation propagation



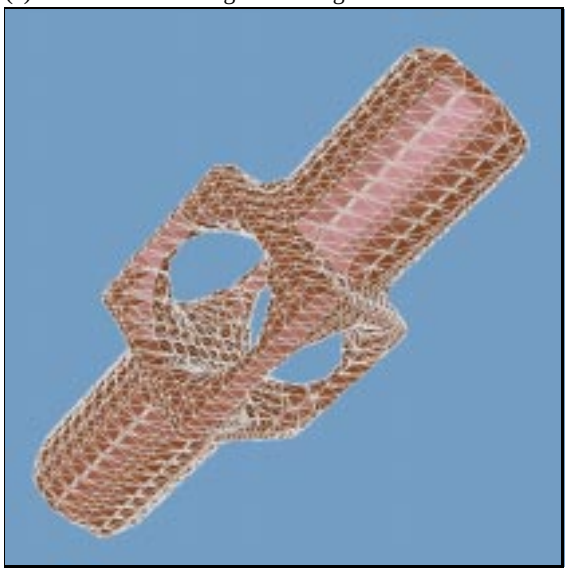
(b) Oriented tangent planes ($Tp(\mathbf{x}_i)$)



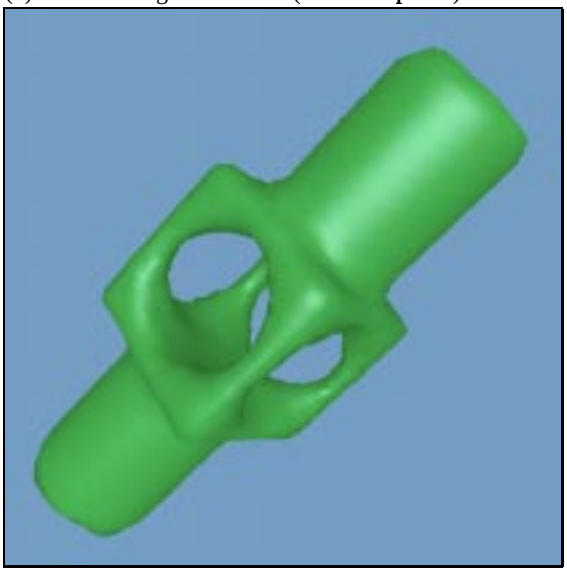
(c) Cubes visited during contouring



(d) Estimated signed distance (shown as $\mathbf{p} - \mathbf{z}$)

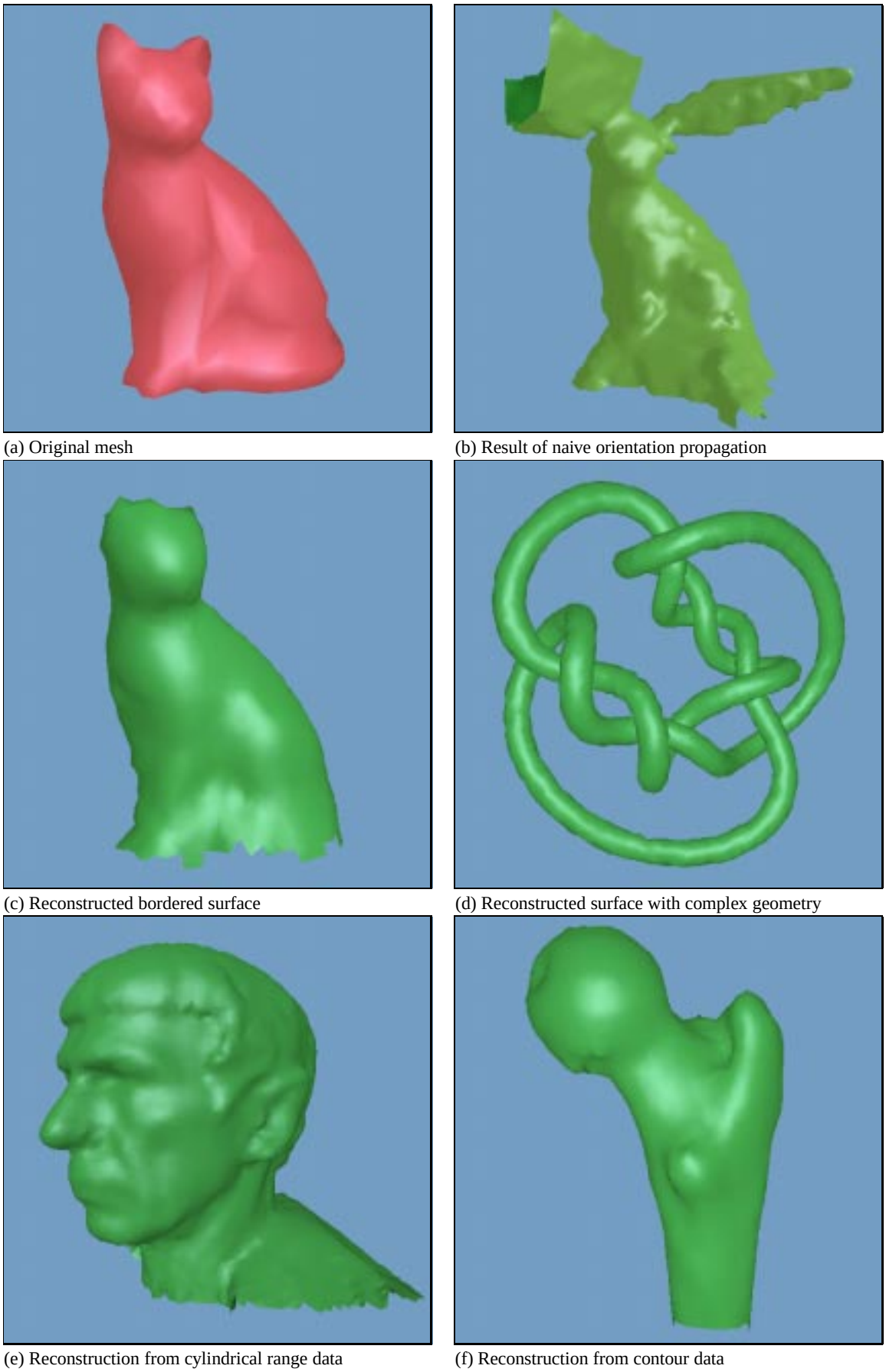


(e) Output of modified marching cubes



(f) Final surface after edge collapses

Figure 2: Reconstruction of ray-traced CSG object (continued).



(a) Original mesh

(b) Result of naive orientation propagation

(c) Reconstructed bordered surface

(d) Reconstructed surface with complex geometry

(e) Reconstruction from cylindrical range data

(f) Reconstruction from contour data

Figure 3: Reconstruction examples.

Mesh Optimization

Hugues Hoppe* Tony DeRose* Tom Duchamp†
John McDonald‡ Werner Stuetzle‡

University of Washington
Seattle, WA 98195

Abstract

We present a method for solving the following problem: Given a set of data points scattered in three dimensions and an initial triangular mesh M_0, produce a mesh M, of the same topological type as M_0, that fits the data well and has a small number of vertices. Our approach is to minimize an energy function that explicitly models the competing desires of conciseness of representation and fidelity to the data. We show that mesh optimization can be effectively used in at least two applications: surface reconstruction from unorganized points, and mesh simplification (the reduction of the number of vertices in an initially dense mesh of triangles).

CR Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling.

Additional Keywords: Geometric Modeling, Surface Fitting, Three-Dimensional Shape Recovery, Range Data Analysis, Model Simplification.

1 Introduction

The *mesh optimization* problem considered in this paper can be roughly stated as follows: Given a collection of data points X in $\mathbf{R}^3$ and an initial triangular mesh M_0 near the data, find a mesh M of the same topological type as M_0 that fits the data well and has a small number of vertices.

As an example, Figure 7b shows a set of 4102 data points sampled from the object shown in Figure 7a. The input to the mesh optimization algorithm consists of the points together with the initial mesh shown in Figure 7c. The optimized mesh is shown in Figure 7h. Notice that the sharp edges and corners indicated by the data have been faithfully recovered and that the number of vertices has been significantly reduced (from 1572 to 163).

*Department of Computer Science and Engineering, FR-35
†Department of Mathematics, GN-50
‡Department of Statistics, GN-22

This work was supported in part by Bellcore, the Xerox Corporation, IBM, Hewlett-Packard, AT&T Bell Labs, the Digital Equipment Corporation, the Department of Energy under grant DE-FG06-85-ER25006, the National Library of Medicine under grant NIH LM-04174, and the National Science Foundation under grants CCR-8957323 and DMS-9103002.

To solve the mesh optimization problem we minimize an *energy function* that captures the competing desires of tight geometric fit and compact representation. The tradeoff between geometric fit and compact representation is controlled via a user-selectable parameter c_{rep}. A large value of c_{rep} indicates that a sparse representation is to be strongly preferred over a dense one, usually at the expense of degrading the fit.

We use the input mesh M_0 as a starting point for a non-linear optimization process. During the optimization we vary the number of vertices, their positions, and their connectivity. Although we can give no guarantee of finding a global minimum, we have run the method on a wide variety of data sets; the method has produced good results in all cases (see Figure 1).

We see at least two applications of mesh optimization: surface reconstruction and mesh simplification.

The problem of surface reconstruction from sampled data occurs in many scientific and engineering applications. In [2], we outlined a two phase procedure for reconstructing a surface from a set of unorganized data points. The goal of phase one is to determine the topological type of the unknown surface and to obtain a crude estimate of its geometry. An algorithm for phase one was described in [5]. The goal of phase two is to improve the fit and reduce the number of faces. Mesh optimization can be used for this purpose.

Although we were originally led to consider the mesh optimization problem by our research on surface reconstruction, the algorithm we have developed can also be applied to the problem of mesh simplification. Mesh simplification, as considered by Turk [15] and Schroeder et al. [10], refers to the problem of reducing the number of faces in a dense mesh while minimally perturbing the shape. Mesh optimization can be used to solve this problem as follows: sample data points X from the initial mesh and use the initial mesh as the starting point M_0 of the optimization procedure. For instance, Figure 7q shows a triangular approximation of a minimal surface with 2032 vertices. Application of our mesh optimization algorithm to a sample of 6752 points (Figure 7r) from this mesh produces the meshes shown in Figures 7s (487 vertices) and 7t (239 vertices). The mesh of Figure 7s corresponds to a relatively small value of c_{rep}, and therefore has more vertices than the mesh of Figure 7t which corresponds to a somewhat larger value of c_{rep}.

The principal contributions of this paper are:

- It presents an algorithm for fitting a mesh of arbitrary topological type to a set of data points (as opposed to volume data, etc.). During the fitting process, the number and connectivity of the vertices, as well as their positions, are allowed to vary.
- It casts mesh simplification as an optimization problem with an energy function that directly measures deviation of the final mesh from the original. As a consequence, the final mesh

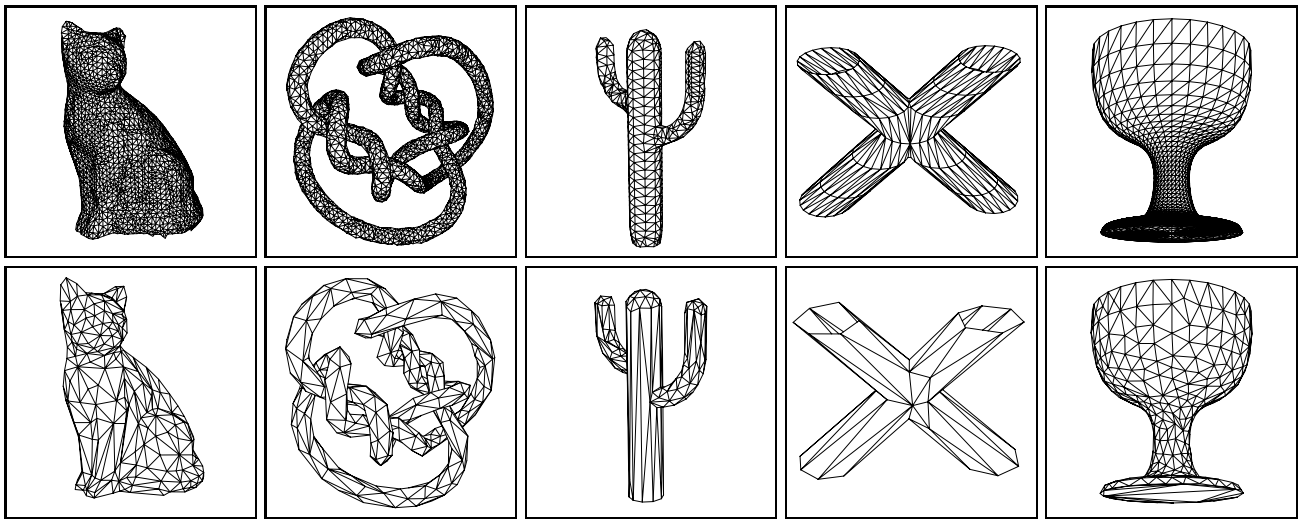


Figure 1: Examples of mesh optimization. The meshes in the top row are the initial meshes M_0; the meshes in the bottom row are the corresponding optimized meshes. The first 3 columns are reconstructions; the last 2 columns are simplifications.

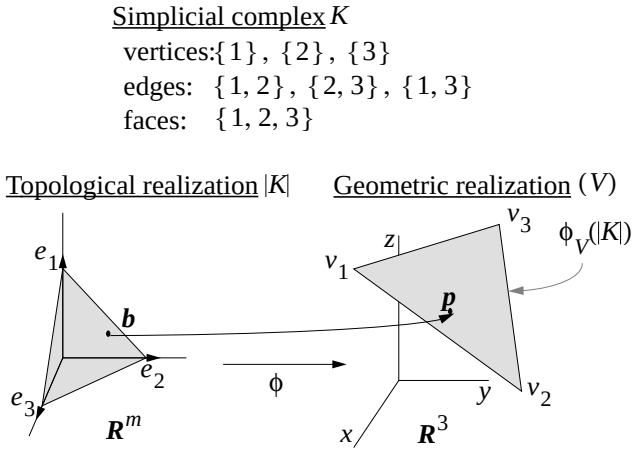


Figure 2: Example of mesh representation: a mesh consisting of a single face.

naturally adapts to curvature variations in the original mesh.

- It demonstrates how the algorithm's ability to recover sharp edges and corners can be exploited to automatically segment the final mesh into smooth connected components (see Figure 7i).

2 Mesh Representation

Intuitively, a *mesh* is a piecewise linear surface, consisting of triangular faces pasted together along their edges. For our purposes it is important to maintain the distinction between the connectivity of the vertices and their geometric positions. Formally, a mesh M is a pair (K, V), where: K is a *simplicial complex* representing the connectivity of the vertices, edges, and faces, thus determining the topological type of the mesh; $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$, $\mathbf{v}_i \in \mathbf{R}^3$ is a set of vertex positions defining the shape of the mesh in $\mathbf{R}^3$ (its geometric realization).

A simplicial complex K consists of a set of vertices $\{1, \ldots, m\}$, together with a set of non-empty subsets of the vertices, called the simplices of K, such that any set consisting of exactly one vertex is a simplex in K, and every non-empty subset of a simplex in K is again a simplex in K (cf. Spanier [14]). The 0-simplices $\{i\} \in K$ are called vertices, the 1-simplices $\{i, j\} \in K$ are called edges, and the 2-simplices $\{i, j, k\} \in K$ are called faces.

A geometric realization of a mesh as a surface in $\mathbf{R}^3$ can be obtained as follows. For a given simplicial complex K, form its *topological realization* $|K|$ in $\mathbf{R}^m$ by identifying the vertices $\{1, \ldots, m\}$ with the standard basis vectors $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ of $\mathbf{R}^m$. For each simplex $s \in K$ let $|s|$ denote the convex hull of its vertices in $\mathbf{R}^m$, and let $|K| = \cup_{s \in K} |s|$. Let $\phi : \mathbf{R}^m \to \mathbf{R}^3$ be the linear map that sends the i-th standard basis vector $\mathbf{e}_i \in \mathbf{R}^m$ to $\mathbf{v}_i \in \mathbf{R}^3$ (see Figure 2).

The *geometric realization* of M is the image $\phi_V(|K|)$, where we write the map as ϕ_V to emphasize that it is fully specified by the set of vertex positions $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$. The map ϕ_V is called an *embedding* if it is 1-1, that is if $\phi_V(|K|)$ is not self-intersecting. Only a restricted set of vertex positions V result in ϕ_V being an embedding.

If ϕ_V is an embedding, any point $\mathbf{p} \in \phi_V(|K|)$ can be parameterized by finding its unique pre-image on $|K|$. The vector $\mathbf{b} \in |K|$ with $\mathbf{p} = \phi_V(\mathbf{b})$ is called the *barycentric coordinate vector* of $\mathbf{p}$ (with respect to the simplicial complex K). Note that barycentric coordinate vectors are convex combinations of standard basis vectors $\mathbf{e}_i \in \mathbf{R}^m$ corresponding to the vertices of a face of K. Any barycentric coordinate vector has at most three non-zero entries; it has only two non-zero entries if it lies on an edge of $|K|$, and only one if it is a vertex.

3 Definition of the Energy Function

Recall that the goal of mesh optimization is to obtain a mesh that provides a good fit to the point set X and has a small number of vertices. We find a simplicial complex K and a set of vertex positions V defining a mesh $M = (K, V)$ that minimizes the energy function

$$E(K, V) = E_{dist}(K, V) + E_{rep}(K) + E_{spring}(K, V).$$

The first two terms correspond to the two stated goals; the third term is motivated below.

The distance energy E_{dist} is equal to the sum of squared distances

from the points $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ to the mesh,

$$E_{dist}(K, V) = \sum_{i=1}^{n} d^2(\mathbf{x}_i, \phi_V(|K|)).$$

The representation energy E_{rep} penalizes meshes with a large number of vertices. It is set to be proportional to the number of vertices m of K:

$$E_{rep}(K) = c_{rep} m.$$

The optimization allows vertices to be both added to and removed from the mesh. When a vertex is added, the distance energy E_{dist} is likely to be reduced; the term E_{rep} makes this operation incur a penalty so that vertices are not added indefinitely. Similarly, one wants to remove vertices from a dense mesh even if E_{dist} increases slightly; in this case E_{rep} acts to encourage the vertex removal. The user-specified parameter c_{rep} provides a controllable trade-off between fidelity of geometric fit and parsimony of representation.

We discovered, as others have before us [8], that minimizing $E_{dist} + E_{rep}$ does not produce the desired results. As an illustration of what can go wrong, Figure 7d shows the result of minimizing E_{dist} alone. The estimated surface has several spikes in regions where there is no data. These spikes are a manifestation of the fundamental problem that a minimum of $E_{dist} + E_{rep}$ may not exist.

To guarantee the existence of a minimum [6], we add the third term, the spring energy E_{spring}. It places on each edge of the mesh a spring of rest length zero and spring constant κ:

$$E_{spring}(K, V) = \sum_{\{j,k\} \in K} \kappa \|\mathbf{v}_j - \mathbf{v}_k\|^2$$

It is worthwhile emphasizing that the spring energy is not a smoothness penalty. Our intent is not to penalize sharp dihedral angles in the mesh, since such features may be present in the underlying surface and should be recovered. We view E_{spring} as a regularizing term that helps guide the optimization to a desirable local minimum. As the optimization converges to the solution, the magnitude of E_{spring} can be gradually reduced. We return to this issue in Section 4.4.

For some applications we want the procedure to be scale-invariant, which is equivalent to defining a unitless energy function E. To achieve invariance under Euclidean motion and uniform scaling, the points X and the initial mesh M_0 are pre-scaled uniformly to fit in a unit cube. After optimization, a post-processing step can undo this initial transformation.

4 Minimization of the Energy Function

Our goal is to minimize the energy function

$$E(K, V) = E_{dist}(K, V) + E_{rep}(K) + E_{spring}(K, V)$$

over the set $\mathcal{K}$ of simplicial complexes K homeomorphic to the initial simplicial complex K_0, and the vertex positions V defining the embedding. We now present an outline of our optimization algorithm, a pseudo-code version of which appears in Figure 3. The details are deferred to the next two subsections.

To minimize $E(K, V)$ over both K and V, we partition the problem into two nested subproblems: an inner minimization over V for fixed simplicial complex K, and an outer minimization over K.

In Section 4.1 we describe an algorithm that solves the inner minimization problem. It finds $E(K) = \min_V E(K, V)$, the energy of the best possible embedding of the fixed simplicial complex K, and the corresponding vertex positions V, given an initial guess for V. This corresponds to the procedure OptimizeVertexPositions in Figure 3.

```
OptimizeMesh(K0,V0) {
    K := K0
    V := OptimizeVertexPositions(K0,V0)
    – Solve the outer minimization problem.
    repeat {
        (K',V') := GenerateLegalMove(K,V)
        V' = OptimizeVertexPositions(K',V')
        if E(K', V') < E(K, V) then
            (K,V) := (K',V')
        endif
    } until convergence
    return (K,V)
}

– Solve the inner optimization problem
–   E(K) = min_V E(K, V)
– for fixed simplicial complex K.
OptimizeVertexPositions(K,V) {
    repeat {
        – Compute barycentric coordinates by projection.
        B := ProjectPoints(K,V)
        – Minimize E(K, V, B) over V using conjugate gradients.
        V := ImproveVertexPositions(K,B)
    } until convergence
    return V
}

GenerateLegalMove(K,V) {
    Select a legal move K ⇒ K'.
    Locally modify V to obtain V' appropriate for K'.
    return (K',V')
}
```

Figure 3: An idealized pseudo-code version of the minimization algorithm.

Whereas the inner minimization is a continuous optimization problem, the outer minimization of $E(K)$ over the simplicial complexes $K \in \mathcal{K}$ (procedure OptimizeMesh) is a discrete optimization problem. An algorithm for its solution is presented in Section 4.2.

The energy function $E(K, V)$ depends on two parameters c_{rep} and κ. The parameter c_{rep} controls the tradeoff between conciseness and fidelity to the data and should be set by the user. The parameter κ, on the other hand, is a regularizing parameter that, ideally, would be chosen automatically. Our method of setting κ is described in Section 4.4.

4.1 Optimization for Fixed Simplicial Complex
(Procedure OptimizeVertexPositions)

In this section, we consider the problem of finding a set of vertex positions V that minimizes the energy function $E(K, V)$ for a given simplicial complex K. As $E_{rep}(K)$ does not depend on V, this amounts to minimizing $E_{dist}(K, V) + E_{spring}(K, V)$.

To evaluate the distance energy $E_{dist}(K, V)$, it is necessary to compute the distance of each data point $\mathbf{x}_i$ to $M = \phi_V(|K|)$. Each of these distances is itself the solution to the minimization problem

$$d^2(\mathbf{x}_i, \phi_V(|K|)) = \min_{\mathbf{b}_i \in |K|} \|\mathbf{x}_i - \phi_V(\mathbf{b}_i)\|^2,$$

in which the unknown is the barycentric coordinate vector $\mathbf{b}_i \in |K| \subset \mathbf{R}^m$ of the projection of $\mathbf{x}_i$ onto M. Thus, minimizing

$E(K, V)$ for fixed K is equivalent to minimizing the new objective function

$$\begin{aligned} E(K, V, B) &= \sum_{i=1}^{n} \|\mathbf{x}_i - \phi_V(\mathbf{b}_i)\|^2 + E_{spring}(K, V) \\ &= \sum_{i=1}^{n} \|\mathbf{x}_i - \phi_V(\mathbf{b}_i)\|^2 + \sum_{\{j,k\} \in K} \kappa \|\mathbf{v}_j - \mathbf{v}_k\|^2 \end{aligned}$$

over the vertex positions $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$, $\mathbf{v}_i \in \mathbf{R}^3$ and the barycentric coordinates $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, $\mathbf{b}_i \in |K| \subset \mathbf{R}^m$.

To solve this optimization problem (procedure OptimizeVertexPositions), our method alternates between two subproblems:

1. For fixed vertex positions V, find optimal barycentric coordinate vectors B by *projection* (procedure ProjectPoints).
2. For fixed barycentric coordinate vectors B, find optimal vertex positions V by solving a *linear* least squares problem (procedure ImproveVertexPositions).

Because we find optimal solutions to both of these subproblems, $E(K, V, B)$ can never increase, and since it is bounded from below, it must converge. In principle, one could iterate until some formal convergence criterion is met. Instead, as is common, we perform a fixed number of iterations. As an example, Figure 7e shows the result of optimizing the mesh of Figure 7c over the vertex positions while holding the simplicial complex fixed.

It is conceivable that procedure OptimizeVertexPositions returns a set V of vertices for which the mesh is self-intersecting, i.e. ϕ_V is not an embedding. While it is possible to check *a posteriori* whether ϕ_V is an embedding, constraining the optimization to always produce an embedding appears to be difficult. This has not presented a problem in the examples we have run.

4.1.1 Projection Subproblem
(Procedure ProjectPoints)

The problem of optimizing $E(K, V, B)$ over the barycentric coordinate vectors $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, while holding the vertex positions $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ and the simplicial complex K constant, decomposes into n separate optimization problems:

$$\mathbf{b}_i = \operatorname*{argmin}_{\mathbf{b} \in |K|} \|\mathbf{x}_i - \phi_V(\mathbf{b})\|$$

In other words, $\mathbf{b}_i$ is the barycentric coordinate vector corresponding to the point $\mathbf{p} \in \phi_V(|K|)$ closest to $\mathbf{x}_i$.

A naive approach to computing $\mathbf{b}_i$ is to project $\mathbf{x}_i$ onto all of the faces of M, and then find the projection with minimal distance. To speed up the projection, we first enter the faces of the mesh into a spatial partitioning data structure (similar to the one used in [16]). Then for each point $\mathbf{x}_i$ only a nearby subset of the faces needs to be considered, and the projection step takes expected time $O(n)$. For additional speedup we exploit coherence between iterations. Instead of projecting each point globally onto the mesh, we assume that a point's projection lies in a neighborhood of its projection in the previous iteration. Specifically, we project the point onto all faces that share a vertex with the previous face. Although this is a heuristic that can fail, it has performed well in practice.

4.1.2 Linear Least Squares Subproblem
(Procedure ImproveVertexPositions)

Minimizing $E(K, V, B)$ over the vertex positions V while holding B and K fixed is a linear least squares problem. It decomposes into three independent subproblems, one for each of the three coordinates of the vertex positions. We will write down the problem for the first coordinate.

Let e be the number of edges (1-simplices) in K; note that e is $O(m)$. Let $\mathbf{v}^1$ be the m-vector whose i-th element is the first coordinate of $\mathbf{v}_i$. Let $\mathbf{d}^1$ be the $(n+e)$-vector whose first n elements are the first coordinates of the data points $\mathbf{x}_i$, and whose last e elements are zero. With these definitions we can express the least squares problem for the first coordinate as minimizing $\|A\mathbf{v}^1 - \mathbf{d}^1\|^2$ over $\mathbf{v}^1$. The design matrix A is an $(n+e) \times m$ matrix of scalars. The first n rows of A are the barycentric coordinate vectors $\mathbf{b}_i$. Each of the trailing e rows contains 2 non-zero entries with values $\sqrt{\kappa}$ and $-\sqrt{\kappa}$ in the columns corresponding to the indices of the edge's endpoints. The first n rows of the least squares problem correspond to $E_{dist}(K, V)$, while the last e rows correspond to $E_{spring}(K, V)$. An important feature of the matrix A is that it contains at most 3 non-zero entries in each row, for a total of $O(n + m)$ non-zero entries.

To solve the least squares problem, we use the conjugate gradient method (cf. [3]). This is an iterative method guaranteed to find the exact solution in as many iterations as there are distinct singular values of A, i.e. in at most m iterations. Usually far fewer iterations are required to get a result with acceptable precision. For example, we find that for m as large as 10^4, as few as 200 iterations are sufficient.

The two time-consuming operations in each iteration of the conjugate gradient algorithm are the multiplication of A by an $(n+e)$-vector and the multiplication of A^T by an m-vector. Because A is sparse, these two operations can be executed in $O(n + m)$ time. We store A in a sparse form that requires only $O(n + m)$ space. Thus, an acceptable solution to the least squares problem is obtained in $O(n+m)$ time. In contrast, a typical noniterative method for solving dense least squares problems, such as QR decomposition, would require $O((n + m)m^2)$ time to find an exact solution.

4.2 Optimization over Simplicial Complexes
(Procedure OptimizeMesh)

To solve the outer minimization problem, minimizing $E(K)$ over K, we define a set of three elementary transformations, *edge collapse*, *edge split*, and *edge swap*, taking a simplicial complex K to another simplicial complex K' (see Figure 4).

We define a *legal move* to be the application of one of these elementary transformations to an edge of K that leaves the topological type of K unchanged. The set of elementary transformations is complete in the sense that *any* simplicial complex in $\mathcal{K}$ can be obtained from K_0 through a sequence of legal moves[1].

Our goal then is to find such a sequence taking us from K_0 to a minimum of $E(K)$. We do this using a variant of random descent: we randomly select a legal move, $K \Rightarrow K'$. If $E(K') < E(K)$, we accept the move, otherwise we try again. If a large number of trials fails to produce an acceptable move, we terminate the search.

More elaborate selection strategies, such as steepest descent or simulated annealing, are possible. As we have obtained good results with the simple strategy of random descent, we have not yet implemented the other strategies.

Identifying Legal Moves An edge split transformation is always a legal move, as it can never change the topological type of K. The other two transformations, on the other hand, can cause a change of topological type, so tests must be performed to determine if they are legal moves.

[1] In fact, we prove in [6] that edge collapse and edge split are sufficient; we include edge swap to allow the optimization procedure to "tunnel" through small hills in the energy function.

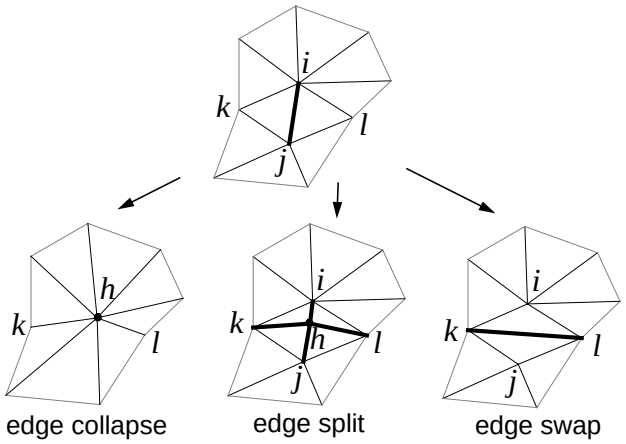


Figure 4: Local simplicial complex transformations

We define an edge $\{i,j\} \in K$ to be a *boundary edge* if it is a subset of only one face $\{i,j,k\} \in K$, and a vertex $\{i\}$ to be a *boundary vertex* if there exists a boundary edge $\{i,j\} \in K$.

An edge collapse transformation $K \Rightarrow K'$ that collapses the edge $\{i,j\} \in K$ is a legal move if and only if the following conditions are satisfied (proof in [6]):

- For all vertices $\{k\}$ adjacent to both $\{i\}$ and $\{j\}$ ($\{i,k\} \in K$ and $\{j,k\} \in K$), $\{i,j,k\}$ is a face of K.
- If $\{i\}$ and $\{j\}$ are both boundary vertices, $\{i,j\}$ is a boundary edge.
- K has more than 4 vertices if neither $\{i\}$ nor $\{j\}$ are boundary vertices, or K has more than 3 vertices if either $\{i\}$ or $\{j\}$ are boundary vertices.

An edge swap transformation $K \Rightarrow K'$ that replaces the edge $\{i,j\} \in K$ with $\{k,l\} \in K'$ is a legal move if and only if $\{k,l\} \notin K$.

4.3 Exploiting Locality

The idealized algorithm described so far is too inefficient to be of practical use. In this section, we describe some heuristics which dramatically reduce the running time. These heuristics capitalize on the fact that a local change in the structure of the mesh leaves the optimal positions of distant vertices essentially unchanged.

4.3.1 Heuristics for Evaluating the Effect of Legal Moves

Our strategy for selecting legal moves requires evaluation of $E(K') = \min_V E(K', V)$ for a simplicial complex K' obtained from K through a legal move. Ideally, we would use procedure OptimizeVertexPositions of Section 4.1 for this purpose, as indicated in Figure 3. In practice, however, this is too slow. Instead, we use fast local heuristics to estimate the effect of a legal move on the energy function.

Each of the heuristics is based on extracting a submesh in the neighborhood of the transformation, along with the subset of the data points projecting onto the submesh. The change in overall energy is estimated by only considering the contribution of the submesh and the corresponding point set. This estimate is always pessimistic, as full optimization would only further reduce the energy. Therefore, the heuristics never suggest changes that will increase the true energy of the mesh.

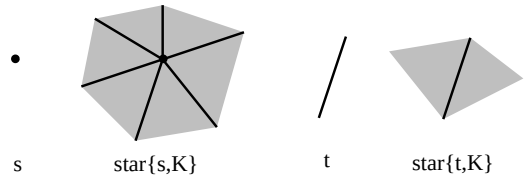


Figure 5: Neighborhood subsets of K.

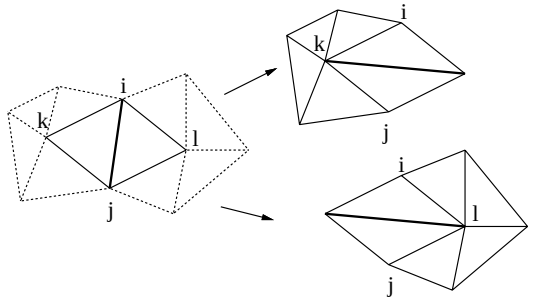


Figure 6: Two local optimizations to evaluate edge swap

Definition of neighborhoods in a simplicial complex To refer to neighborhoods in a simplicial complex, we need to introduce some further notation. We write $s' \leq s$ to denote that simplex s' is a non-empty subset of simplex s. For simplex $s \in K$, $\text{star}(s;K) = \{s' \in K : s \leq s'\}$ (Figure 5).

Evaluation of Edge Collapse To evaluate a transformation $K \Rightarrow K'$ collapsing an edge $\{i,j\}$ into a single vertex $\{h\}$ (Figure 4), we take the submesh to be $\text{star}(\{i\};K) \cup \text{star}(\{j\};K)$, and optimize over the single vertex position $\mathbf{v}_h$ while holding all other vertex positions constant.

Because we perform only a small number of iterations (for reasons of efficiency), the initial choice of $\mathbf{v}_h$ greatly influences the accuracy of the result. Therefore, we attempt three optimizations, with $\mathbf{v}_h$ starting at $\mathbf{v}_i$, $\mathbf{v}_j$, and $\frac{1}{2}(\mathbf{v}_i + \mathbf{v}_j)$, and accept the best one.

The edge collapse should be allowed only if the new mesh does not intersect itself. Checking for this would be costly; instead we settle for a less expensive heuristic check. If, after the local optimization, the maximum dihedral angle of the edges in $\text{star}(\{h\};K')$ is greater than some threshold, the edge collapse is rejected.

Evaluation of Edge Split The procedure is the same as for edge collapse, except that the submesh is defined to be $\text{star}(\{i,j\};K)$, and the initial value of the new vertex $\mathbf{v}_h$ is chosen to be $\frac{1}{2}(\mathbf{v}_i + \mathbf{v}_j)$.

Evaluation of Edge Swap To evaluate an edge swap transformation $K \Rightarrow K'$ that replaces an edge $\{i,j\} \in K$ with $\{k,l\} \in K'$, we consider two local optimizations, one with submesh $\text{star}(\{k\};K')$, varying vertex $\mathbf{v}_k$, and one with submesh $\text{star}(\{l\};K')$, varying vertex $\mathbf{v}_l$ (Figure 6). The change in energy is taken to best of these. As is the case in evaluating an edge collapse, we reject the transformation if the maximum dihedral angle after the local optimization exceeds a threshold.

4.3.2 Legal Move Selection Strategy
(Procedure GenerateLegalMove)

The simple strategy for selecting legal moves described in Section 4.2 can be improved by exploiting locality. Instead of selecting edges completely at random, edges are selected from a candidate set. This candidate set consists of all edges that may lead to beneficial moves, and initially contains all edges.

To generate a legal move, we randomly remove an edge from the candidate set. We first consider collapsing the edge, accepting the move if it is legal and reduces the total energy. If the edge collapse is not accepted, we then consider edge swap and edge split in that order. If one of the transformations is accepted, we update the candidate set by adding all neighboring edges. The candidate set becomes very useful toward the end of optimization, when the fraction of beneficial moves diminishes.

4.4 Setting of the Spring Constant

We view the spring energy E_{spring} as a regularizing term that helps guide the optimization process to a good minimum. The spring constant κ determines the contribution of this term to the total energy. We have obtained good results by making successive calls to procedure OptimizeMesh, each with a different value of κ, according to a schedule that gradually decreases κ.

As an example, to obtain the final mesh in Figure 7h starting from the mesh in Figure 7c, we successively set κ to $10^{-2}, 10^{-3}, 10^{-4}$, and 10^{-8} (see Figures 7f–7h). This same schedule was used in all the examples.

5 Results

5.1 Surface Reconstruction

From the set of points shown in Figure 7b, phase one of our reconstruction algorithm [5] produces the mesh shown in Figure 7c; this mesh has the correct topological type, but it is rather dense, is far away from the data, and lacks the sharp features of the original model (Figure 7a). Using this mesh as a starting point, mesh optimization produces the mesh in Figure 7h.

Figures 7i–7k,7m–7o show two examples of surface reconstruction from actual laser range data (courtesy of Technical Arts, Redmond, WA). Figures 7j and 7n show sets of points obtained by sampling two physical objects (a distributor cap and a golf club head) with a laser range finder. The outputs of phase one are shown in Figures 7k and 7o. The holes present in the surface of Figure 7k are artifacts of the data, as self-shadowing prevented some regions of the surface from being scanned. Adaptive selection of scanning paths preventing such shadowing is an interesting area of future research. In this case, we manually filled the holes, leaving a single boundary at the bottom. Figures 7l and 7p show the optimized meshes obtained with our algorithm.

5.2 Mesh Simplification

For mesh simplification, we first sample a set of points randomly from the original mesh using uniform random sampling over area. Next, we add the vertices of the mesh to this point set. Finally, to more faithfully preserve the boundaries of the mesh, we sample additional points from boundary edges.

As an example of mesh simplification, we start with the mesh containing 2032 vertices shown in Figure 7q. From it, we obtain a sample of 6752 points shown in Figure 7r (4000 random points, 2032 vertex points, and 720 boundary points). Mesh optimization, with $c_{rep} = 10^{-5}$, reduces the mesh down to 487 vertices (Figure 7s).

Fig.	#vert. m	#faces	#data n	Parameters c_{rep}	Parameters κ	Resulting energies E_{dist}	Resulting energies E	time (min.)
7c	1572	3152	4102	-	-	8.57×10^{-2}	-	-
7e	1572	3152	4102	10^{-5}	10^{-2}	8.04×10^{-4}	4.84×10^{-2}	1.5
7f	508	1024	4102	10^{-5}	10^{-2}	6.84×10^{-4}	3.62×10^{-2}	(+3.0)
7g	270	548	4102	10^{-5}	10^{-3}	6.08×10^{-4}	6.94×10^{-3}	(+2.2)
7h	163	334	4102	10^{-5}	*varied*	4.86×10^{-4}	2.12×10^{-3}	17.0
7k	9220	18272	12745	-	-	6.41×10^{-2}	-	-
7l	690	1348	12745	10^{-5}	*varied*	4.23×10^{-3}	1.18×10^{-2}	47.0
7o	4059	8073	16864	-	-	2.20×10^{-2}	-	-
7p	262	515	16864	10^{-5}	*varied*	2.19×10^{-3}	4.95×10^{-3}	44.5
7q	2032	3832	-	-	-	-	-	-
7s	487	916	6752	10^{-5}	*varied*	1.86×10^{-3}	8.05×10^{-3}	9.9
7t	239	432	6752	10^{-4}	*varied*	9.19×10^{-3}	4.39×10^{-2}	10.2

Table 1: Performance statistics for meshes shown in Figure 7.

By setting $c_{rep} = 10^{-4}$, we obtain a coarser mesh of 239 vertices (Figure 7t).

As these examples illustrate, basing mesh simplification on a measure of distance between the simplified mesh and the original has a number of benefits:

- Vertices are dense in regions of high Gaussian curvature, whereas a few large faces span the flat regions.
- Long edges are aligned in directions of low curvature, and the aspect ratios of the triangles adjust to local curvature.
- Edges and vertices of the simplified mesh are placed near sharp features of the original mesh.

5.3 Segmentation

Mesh optimization enables us to detect sharp features in the underlying surface. Using a simple thresholding method, the optimized mesh can be segmented into smooth components. To this end, we build a graph in which the nodes are the faces of mesh. Two nodes of this graph are connected if the two corresponding faces are adjacent and their dihedral angle is smaller than a given threshold. The connected components of this graph identify the desired smooth segments. As an example, Figure 7i shows the segmentation of the optimized mesh into 11 components. After segmentation, vertex normals can be estimated from neighboring faces within each component, and a smoothly shaded surface can be created (Figure 7m).

5.4 Parameter Settings and Performance Statistics

Table 1 lists the specific parameter values of c_{rep} and κ used to generate the meshes in the examples, along with other performance statistics. In all these examples, the table entry *"varied"* refers to a spring constant schedule of $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-8}\}$. In fact, all meshes in Figure 1 are also created using the same parameters (except that c_{rep} was changed in two cases). Execution times were obtained on a DEC uniprocessor Alpha workstation.

6 Related Work

Surface Fitting There is a large body of literature on fitting embeddings of a rectangular domain; see Bolle and Vemuri [1] for a review. Schudy and Ballard [11, 12] fit embeddings of a sphere to point data. Goshtasby [4] works with embeddings of cylinders and tori. Sclaroff and Pentland [13] consider embeddings of a deformed superquadric. Miller et al. [9] approximate an isosurface of volume data by fitting a mesh homeomorphic to a sphere. While it appears that their method could be extended to finding isosurfaces of arbitrary topological type, it it less obvious how it could be modified to

handle point instead of volume data. Mallet [7] discusses interpolation of functions over simplicial complexes of arbitrary topological type.

Our method allows fitting of a parametric surface of arbitrary topological type to a set of three-dimensional points. In [2], we sketched an algorithm for fitting a mesh of *fixed* vertex connectivity to the data. The algorithm presented here is an extension of this idea in which we also allow the number of vertices and their connectivity to vary. To the best of our knowledge, this has not been done before.

Mesh Simplification Two notable papers discussing the mesh simplification problem are Schroeder et al. [10] and Turk [15].

The motivation of Schroeder et al. is to simplify meshes generated by "marching cubes" that may consist of more than a million triangles. In their iterative approach, the basic operation is removal of a vertex and re-triangulation of the hole thus created. The criterion for vertex removal in the simplest case (interior vertex not on edge or corner) is the distance from the vertex to the plane approximating its surrounding vertices. It is worthwhile noting that this criterion only considers deviation of the new mesh from the mesh created in the previous iteration; deviation from the original mesh does not figure in the strategy.

Turk's goal is to reduce the amount of detail in a mesh while remaining faithful to the original topology and geometry. His basic idea is to distribute points on the existing mesh that are to become the new vertices. He then creates a triangulation containing both old and new vertices, and finally removes the old vertices. The density of the new vertices is chosen to be higher in areas of high curvature.

The principal advantage of our mesh simplification method compared to the techniques mentioned above is that we cast mesh simplification as an optimization problem: we find a new mesh of lower complexity that is as close as possible to the original mesh. This is recognized as a desirable property by Turk (Section 8, p. 63): "Another topic is finding measures of how closely matched a given re-tiling is to the original model. Can such a quality measure be used to guide the re-tiling process?". Optimization automatically retains more vertices in areas of high curvature, and leads to faces that are elongated along directions of low curvature, another property recognized as desirable by Turk.

7 Summary and Future Work

We have described an energy minimization approach to solving the mesh optimization problem. The energy function we use consists of three terms: a distance energy that measures the closeness of fit, a representation energy that penalizes meshes with a large number of vertices, and a regularizing term that conceptually places springs of rest length zero on the edges of the mesh. Our minimization algorithm partitions the problem into two nested subproblems: an inner continuous minimization and an outer discrete minimization. The search space consists of all meshes homeomorphic to the starting mesh.

Mesh optimization has proven effective as the second phase of our method for surface reconstruction from unorganized points, as discussed in [5]. (Phase two is responsible for improving the geometric fit and reducing the number of vertices of the mesh produced in phase one.)

Our method has also performed well for mesh simplification, that is, the reduction of the number of vertices in a dense triangular mesh. It produces meshes whose edges align themselves along directions of low curvature, and whose vertices concentrate in areas of high Gaussian curvature. Because the energy does not penalize surfaces with sharp dihedral angles, the method can recover sharp edges and corners.

A number of areas of future research still remain, including:

- Investigate the use of more sophisticated optimization methods, such as simulated annealing for discrete optimization and quadratic methods for non-linear least squares optimization, in order to avoid undesirable local minima in the energy and to accelerate convergence.
- Gain more insight into the use of the spring energy as a regularizing term, especially in the presence of appreciable noise.
- Improve the speed of the algorithm and investigate implementations on parallel architectures.
- Develop methods for fitting higher order splines to more accurately and concisely model curved surfaces.
- Experiment with sparse, non-uniform, and noisy data.
- Extend the current algorithm to other distance measures such as maximum error (L^∞ norm) or average error (L^1 norm), instead of the current L^2 norm.

References

[1] Ruud M. Bolle and Baba C. Vemuri. On three-dimensional surface reconstruction methods. *IEEE PAMI*, 13(1):1–13, January 1991.

[2] T. DeRose, H. Hoppe, T. Duchamp, J. McDonald, and W. Stuetzle. Fitting of surfaces to scattered data. *SPIE*, 1830:212–220, 1992.

[3] Gene Golub and Charles Van Loan. *Matrix Computations*. John Hopkins University Press, 2nd edition, 1989.

[4] Ardeshir Goshtasby. Surface reconstruction from scattered measurements. *SPIE*, 1830:247–256, 1992.

[5] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):71–78, July 1992.

[6] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. TR 93-01-01, Dept. of Computer Science and Engineering, University of Washington, January 1993.

[7] J.L. Mallet. Discrete smooth interpolation in geometric modeling. *CAD*, 24(4):178–191, April 1992.

[8] Samuel Marin and Philip Smith. Parametric approximation of data using ODR splines. GMR 7057, General Motors Research Laboratories, May 1990.

[9] J.V. Miller, D.E. Breen, W.E. Lorensen, R.M. O'Bara, and M.J. Wozny. Geometrically deformed models: A method for extracting closed geometric models from volume data. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):217–226, July 1991.

[10] William Schroeder, Jonathan Zarge, and William Lorensen. Decimation of triangle meshes. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):65–70, July 1992.

[11] R. B. Schudy and D. H. Ballard. Model detection of cardiac chambers in ultrasound images. Technical Report 12, Computer Science Department, University of Rochester, 1978.

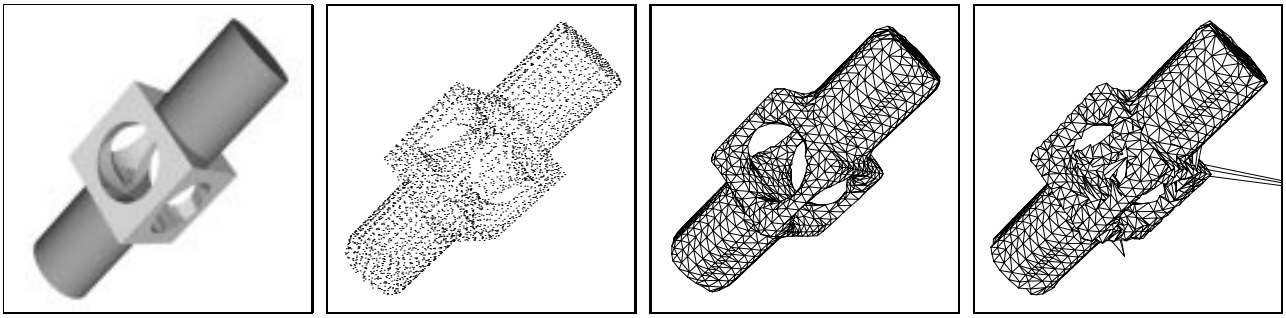
[12] R. B. Schudy and D. H. Ballard. Towards an anatomical model of heart motion as seen in 4-d cardiac ultrasound data. In *Proceedings of the 6th Conference on Computer Applications in Radiology and Computer-Aided Analysis of Radiological Images*, 1979.

[13] Stan Sclaroff and Alex Pentland. Generalized implicit functions for computer graphics. *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4):247–250, July 1991.

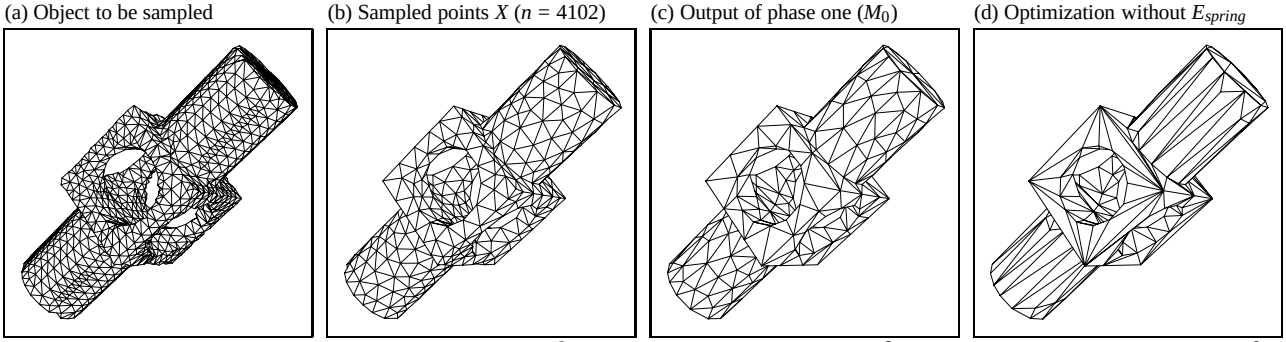
[14] E. H. Spanier. *Algebraic Topology*. McGraw-Hill, New York, 1966.

[15] Greg Turk. Re-tiling polygonal surfaces. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2):55–64, July 1992.

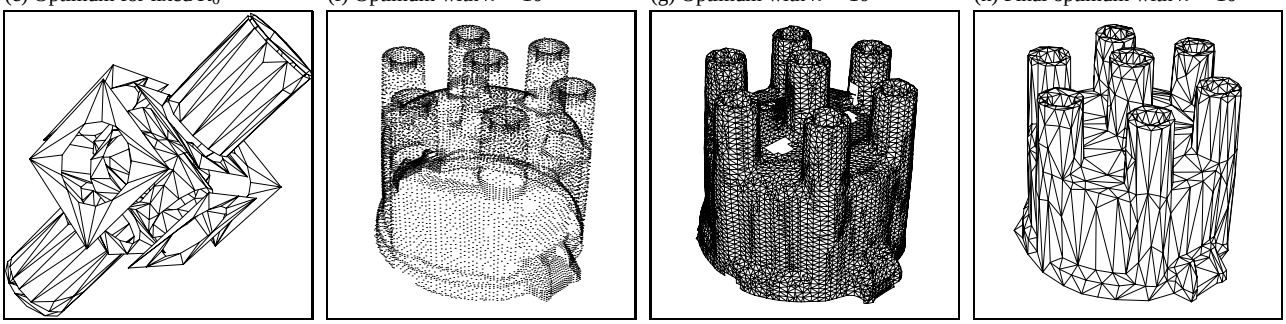
[16] G. Wyvill, C. McPheeters, and B. Wyvill. Data structures for soft objects. *The Visual Computer*, 2(4):227–234, August 1986.



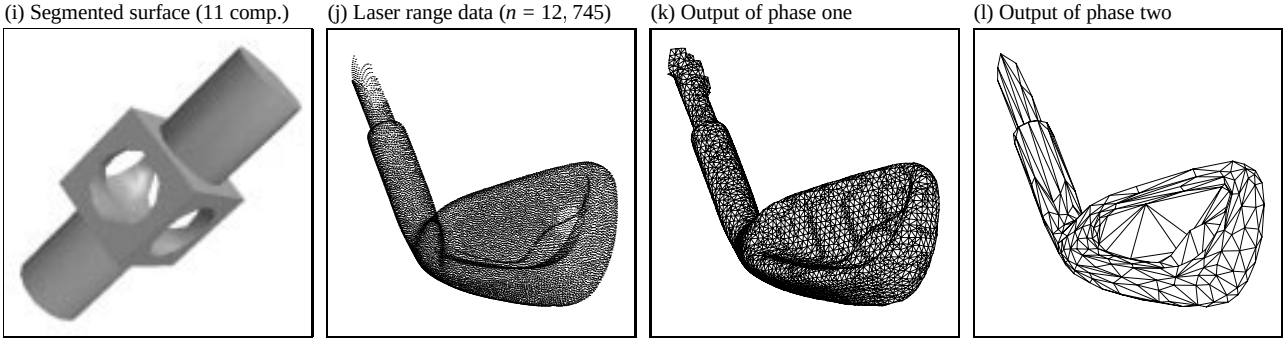
(a) Object to be sampled (b) Sampled points X (n = 4102) (c) Output of phase one (M_0) (d) Optimization without E_{spring}



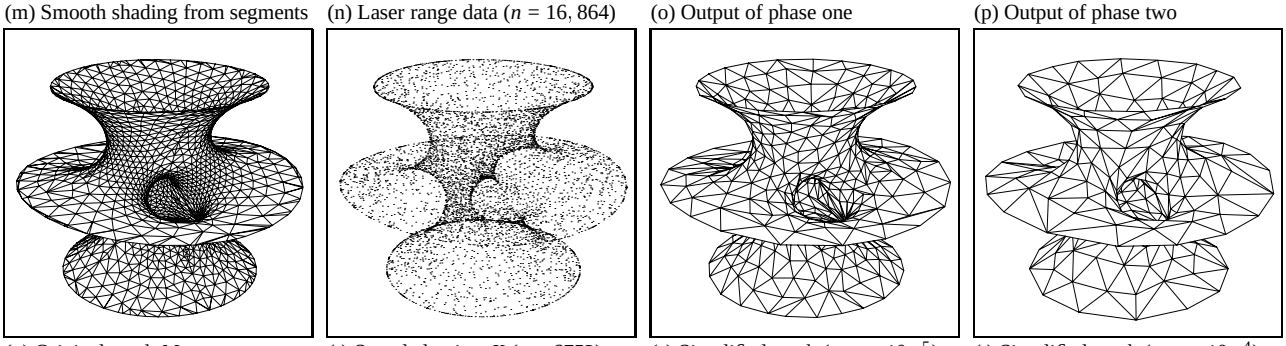
(e) Optimum for fixed K_0 (f) Optimum with $\kappa = 10^{-2}$ (g) Optimum with $\kappa = 10^{-3}$ (h) Final optimum with $\kappa = 10^{-8}$



(i) Segmented surface (11 comp.) (j) Laser range data (n = 12, 745) (k) Output of phase one (l) Output of phase two



(m) Smooth shading from segments (n) Laser range data (n = 16, 864) (o) Output of phase one (p) Output of phase two



(q) Original mesh M_0 (r) Sampled points X (n = 6752) (s) Simplified mesh ($c_{rep} = 10^{-5}$) (t) Simplified mesh ($c_{rep} = 10^{-4}$)

Figure 7: Examples of surface reconstruction and mesh simplification.

Zippered Polygon Meshes from Range Images

Greg Turk and Marc Levoy
Computer Science Department
Stanford University

Abstract

Range imaging offers an inexpensive and accurate means for digitizing the shape of three-dimensional objects. Because most objects self occlude, no single range image suffices to describe the entire object. We present a method for combining a collection of range images into a single polygonal mesh that completely describes an object to the extent that it is visible from the outside.

The steps in our method are: 1) align the meshes with each other using a modified iterated closest-point algorithm, 2) zipper together adjacent meshes to form a continuous surface that correctly captures the topology of the object, and 3) compute local weighted averages of surface positions on all meshes to form a consensus surface geometry.

Our system differs from previous approaches in that it is incremental; scans are acquired and combined one at a time. This approach allows us to acquire and combine large numbers of scans with minimal storage overhead. Our largest models contain up to 360,000 triangles. All the steps needed to digitize an object that requires up to 10 range scans can be performed using our system with five minutes of user interaction and a few hours of compute time. We show two models created using our method with range data from a commercial rangefinder that employs laser stripe technology.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling.
Additional Key Words: Surface reconstruction, surface fitting, polygon mesh, range images, structured light range scanner.

1 Introduction

This paper presents a method of combining multiple views of an object, captured by a range scanner, and assembling these views into one unbroken polygonal surface. Applications for such a method include:

- Digitizing complex objects for animation and visual simulation.
- Digitizing the shape of a found object such as an archaeological artifact for measurement and for dissemination to the scientific community.
- Digitizing human external anatomy for surgical planning, remote consultation or the compilation of anatomical atlases.
- Digitizing the shape of a damaged machine part to help create a replacement.

E-mail: turk@redclay.stanford.edu, levoy@cs.stanford.edu
Web site: www-graphics.stanford.edu

There is currently no procedure that will allow a user to easily capture a digital description of a physical object. The dream tool would allow one to set an industrial part or a clay figure onto a platform, press a button, and have a complete digital description of that object returned in a few minutes. The reality is that much digitization is done by a user painstakingly touching a 3D sensing probe to hundreds or thousands of positions on the object, then manually specifying the connectivity of these points. Fortunately range scanners offer promise in replacing this tedious operation.

A *range scanner* is any device that senses 3D positions on an object's surface and returns an array of distance values. A *range image* is an $m \times n$ grid of distances (*range points*) that describe a surface either in Cartesian coordinates (a height field) or cylindrical coordinates, with two of the coordinates being implicitly defined by the indices of the grid. Quite a number of measurement techniques can be used to create a range image, including structured light, time-of-flight lasers, radar, sonar, and several methods from the computer vision literature such as depth from stereo, shading, texture, motion and focus. The range images used to create the models in this paper were captured using structured light (described later), but our techniques can be used with any range images where the uncertainties of the distance values are smaller than the spacing between the samples.

Range scanners seem like a natural solution to the problem of capturing a digital description of physical objects. Unfortunately, few objects are simple enough that they can be fully described by a single range image. For instance, a coffee cup handle will obscure a portion of the cup's surface even using a cylindrical scan. To capture the full geometry of a moderately complicated object (e.g. a clay model of a cat) may require as many as a dozen range images.

There are two main issues in creating a single model from multiple range images: *registration* and *integration*. Registration refers to computing a rigid transformation that brings the points of one range image into alignment with the portions of a surface that is shares with another range image. Integration is the process of creating a single surface representation from the sample points from two or more range images.

Our approach to registration uses an iterative process to minimize the distance between two triangle meshes that were created from the range images. We accelerate registration by performing the matching on a hierarchy of increasingly more detailed meshes. This method allows an object to be scanned from any orientation without the need for a six-degree-of-freedom motion device.

We separate the task of integration into two steps: 1) creating a mesh that reflects the topology of the object, and 2) refining the vertex positions of the mesh by averaging the geometric detail that is present in all scans. We capture the topology of an object by merging pairs of triangle meshes that are each created from a single range image. Merging begins by converting two meshes that may have considerable overlap into a pair of meshes that just barely overlap along portions of their boundaries. This is done by simultaneously eating back the boundaries of each mesh that lie directly on top of the other mesh. Next, the meshes are *zippered* together: the triangles of one mesh are clipped to the boundary of the other mesh and the vertices on the boundary are shared. Once all the meshes have been combined, we allow all of the scans to contribute to the surface detail by finding the *consensus geometry*. The final position of a vertex is found by taking an average of nearby positions from each of the original range images. The order in which we perform zippering and consensus geometry is important. We deliberately postpone the refinement of surface geometry until after the overall shape of the object has been determined. This eliminates discontinuities that may be introduced during zippering.

The remainder of this paper is organized as follows. Section 2 describes previous work on combining range images. Section 3 covers the basic principles of a structured light range scanner. Section 4 presents the automatic registration process. Section 5 describes zippering meshes into one continuous surface. Section 6 describes how surface detail is captured through consensus geometry. Section 7 shows examples of digitized models and compares our approach to other methods of combining range data. Section 8 concludes this paper by discussing future work.

2 Previous Work

There is a great deal of published work on registration and integration of depth information, particularly in the vision literature. Our literature review only covers work on registration or integration of dense range data captured by an active range scanner, and where the product of the integration is a polygon mesh.

2.1 Registration

Two themes dominate work in range image registration: matching of "created" features in the images to be matched, and minimization of distances between all points on the surface represented by the two images. In the first category, Wada and co-authors performed six degree of freedom registration by matching distinctive facets from the convex hulls of range images [Wada 93]. They computed a rotation matrix from corresponding facets using a least squares fit of the normal vectors of the facets.

In the second category, Champleboux and co-workers used a data structure called an octree-spline that is a sampled representation of distances to an object's surface [Champleboux 92]. This gave them a rapid way to determine distances from a surface (and the distance gradient) with a low overhead in storage. Chen and Medioni establish a correspondence between points on one surface and nearby tangent planes on the other surface [Chen 92]. They find a rigid motion that minimizes the point-to-tangent collection directly and then iterate. Besl and McKay use an approach they call the *iterated closest-point* algorithm [Besl 92]. This method finds the nearest positions on one surface to a collection of points on the other surface and then transforms one surface so as to minimize the collective distance. They iterate this procedure until convergence.

Our registration method falls into the general category of direct distance minimization algorithms, and is an adaptation of [Besl 92]. It differs in that we do not require that one surface be a strict subset of the other. It is described in Section 4.

2.2 Integration

Integration of multiple range scans can be classified into *structured* and *unstructured* methods. Unstructured integration presumes that one has a procedure that creates a polygonal surface from an arbitrary collection of points in 3-space. Integration in this case is performed by collecting together all the range points from multiple scans and presenting them to the polygonal reconstruction procedure. The Delaunay triangulation of a set of points in 3-space has been proposed as the basis of one such reconstruction method [Boissonnat 84]. Another candidate for surface reconstruction is a generalization of the convex hull of a point set known as the alpha shape [Edelsbrunner 92]. Hoppe and co-authors use graph traversal techniques to help construct a signed distance function from a collection of unorganized points [Hoppe 92]. An isosurface extraction technique produces a polygon mesh from this distance function.

Structured integration methods make use of information about how each point was obtained, such as using error bounds on a point's position or adjacency information between points within one range image. Soucy and Laurendeau use a structured integration technique to combine multiple range images [Soucy 92] that is similar in several respects to our algorithm. Given n range images of an object, they first partition the points into a number of sets that are called *common surface sets*. The range points in one set are then used to create a grid of triangles whose positions are guided by a weighted average of the points in the set. Subsets of these grids are stitched together by a constrained Delaunay triangulation in one of n projections onto a plane. We compare our method to Soucy's in Section 7.

3 Structured Light Range Scanners

In this section we describe the operating principles of range scanners based on structured light. We do this because it highlights issues common to many range scanners and also because the range images used in this article were created by such a scanner.

3.1 Triangulation

Structured light scanners operate on the principle of triangulation (see Figure 1, left). One portion of the scanner projects a specific pattern of light onto the object being scanned. This pattern of light is observed by the sensor of the scanner along a viewing direction that is off-axis from the source of light. The position of the illuminated part of the object is determined by finding the intersection of the light's projected direction and the viewing direction of the sensor. Positions can be accumulated across the length of the object while the object is moved across the path of the projected light. Some of the patterns that have been used in such scanners include a spot, a circle, a line, and several lines at once. Typically the sensor is a CCD array or a lateral effect photodiode.

The scanner used for the examples in this paper is a Cyberware Model 3030 MS. It projects a vertical sheet of He-Ne laser light onto the surface of an object. The laser sheet is created by spreading a laser beam using a cylindrical lens into a sheet roughly 2 mm wide and 30 cm high. The sensor of the Cyberware scanner is a 768×486 pixel CCD array. A typical CCD image shows a ribbon of laser light running from the top to the bottom (see Figure 2). A range point is created by looking across a scanline for the peak intensity of this ribbon. A range point's distance from the scanner (the "depth") is given by the horizontal position of this peak and the vertical position of the range point is given by the number of the scanline. Finding the peaks for each scanline in one frame gives an entire column of range points, and combining the columns from multiple frames as the object is moved through the laser sheet gives the full range image.

3.2 Sources of Error

Any approach to combining range scans should attempt to take into account the possible sources of error inherent in a given scanner. Two sources of error are particularly relevant to integration. One is a result of light falling on the object at a grazing angle. When the projected light falls on a portion of the object that is nearly parallel to the light's path, the sensor sees a dim and stretched-out version of the pattern. Finding the center of the laser sheet when it grazes the

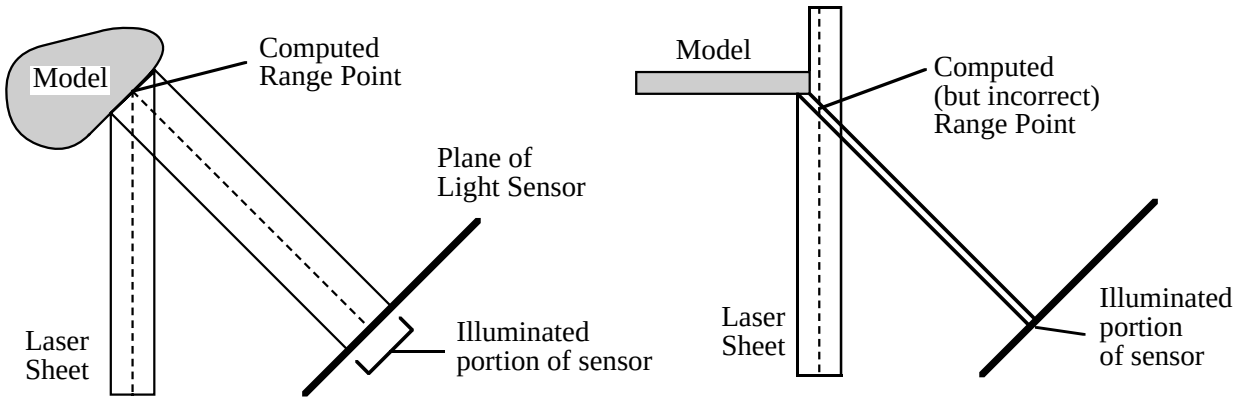


Figure 1: Structured light triangulation (left) and false edge extension in the presence of a partially illuminated edge (right).

object becomes difficult, and this adds uncertainty to the position of the range points. The degree of uncertainty at a given range point can be quantified, and we make use of such information at several stages in our approach to combining range images.

A second source of inaccuracy occurs when only a portion of the laser sheet hits an object, such as when the laser sheet falls off the edge of a book that is perpendicular to the laser sheet (see Figure 1, right). This results in a false position because the peak-detection and triangulation method assumes that the entire width of the sheet is visible. Such an assumption results in edges of objects that are both curled and extended beyond their correct position. This false extension of a surface at edges is an issue that needs to be specifically addressed when combining range images.

3.3 Creating Triangle Meshes from Range Images

We use a mesh of triangles to represent the range image data at all stages of our integration method. Each sample point in the $m \times n$ range image is a potential vertex in the triangle mesh. We take special care to avoid inadvertently joining portions of the surface together that are separated by depth discontinuities (see Figure 3).

To build a mesh, we create zero, one or two triangles from four points of a range image that are in adjacent rows and columns. We find the shortest of the two diagonals between the points and use this to identify the two triplets of points that may become triangles. Each of these point triples is made into a triangle if the edge lengths fall below a distance threshold. Let s be the maximum distance between adjacent range points when we flatten the range image, that is, when we don't include the depth information (see Figure 3). We take the distance threshold be a small multiple of this sampling distance, typically 4s. Although having such a distance threshold may prevent joining some range points that should in fact be connected, we can rely on other range images (those with better views of the location in question) to give the correct adjacency information.

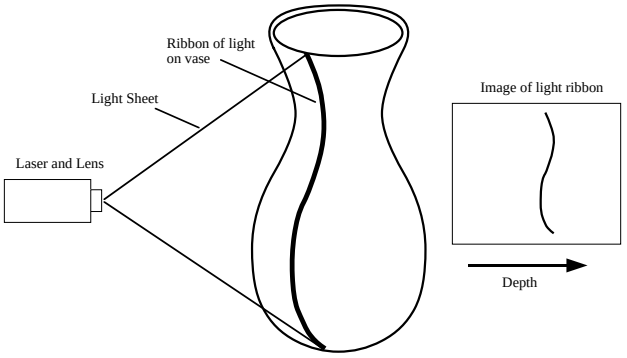


Figure 2: Light-stripe projected on vase (left) and corresponding CCD image (right).

This willingness to discard questionable data is representative of a deliberate overall strategy: to acquire and process large amounts of data rather than draw hypotheses (possibly erroneous) from sparse data. This strategy appears in several places in our algorithm.

4 Registration of Range Images

Once a triangle mesh is created for each range image, we turn to the task of bringing corresponding portions of different range images into alignment with one another. If all range images are captured using a six-degree of freedom precision motion device then the information needed to register them is available from the motion control software. This is the case when the object or scanner is mounted on a robot arm or the motion platform of a precision milling machine. Inexpensive motion platforms are often limited to one or two degrees of freedom, typically translation in a single direction or rotation about an axis. One of our goals is to create an inexpensive system. Consequently, we employ a registration method that does not depend on measured position and orientation. With our scanner, which offers translation and rotation around one axis, we typically take one cylindrical and four translational scans by moving the object with the motion device. To capture the top or the underside of the object, we pick it up by hand and place it on its side. Now the orientation of subsequent scans cannot be matched with those taken earlier, and using a registration method becomes mandatory.

4.1 Iterated Closest-Point Algorithm

This section describes a modified iterated closest-point (ICP) algorithm for quickly registering a pair of meshes created from range images. This method allows a user to crudely align one range image with another on-screen and then invoke an algorithm that snaps the position of one range image into accurate alignment with the other.

The iterated closest-point of [Besl 92] cannot be used to register range images because it requires that *every* point on one surface have

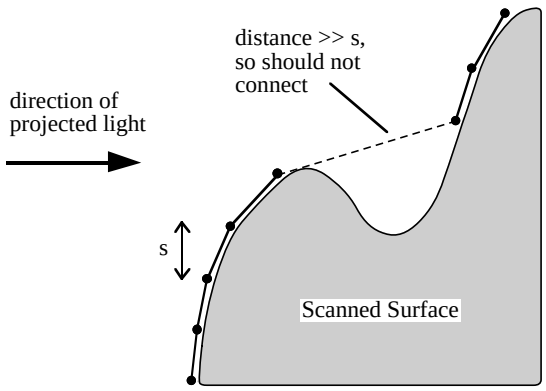


Figure 3: Building triangle mesh from range points.

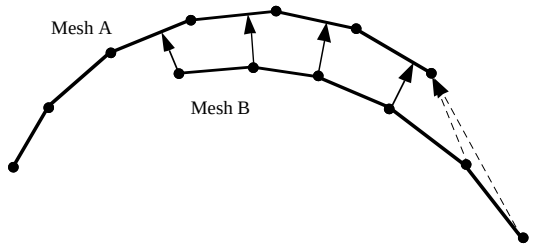


Figure 4: Finding corresponding points for mesh registration. Dotted arrows show matches that should be avoided because they will cause mesh *B* to be erroneously dragged up and left.

a corresponding point on the other surface. Since our scans are overlapping, we seldom produce data that satisfies this requirement. Thus we have developed our own variant of this algorithm. Its steps are:

1) Find the nearest position on mesh *A* to each vertex of mesh *B*.
2) Discard pairs of points that are too far apart.
3) Eliminate pairs in which either points is on a mesh boundary.
4) Find the rigid transformation that minimizes a weighted least-squared distance between the pairs of points.
5) Iterate until convergence.
6) Perform ICP on a more detailed mesh in the hierarchy.

In step 1, it is important to note that we are looking for the 3-space position A_i on the surface of mesh *A* that is closest to a given vertex B_i of mesh *B* (see Figure 4). The nearest point A_i may be a vertex of *A*, may be a point within a triangle, or may lie on a triangle's edge. Allowing these points A_i to be anywhere on a C^0 continuous surface means that the registration between surfaces can have greater accuracy than the spacing *s* between range points.

4.2 Constraints on ICP

Our ICP algorithm differs from Besl's in several ways. First, we have added a distance threshold to the basic iterated closest-point method to avoid matching any vertex B_i of one mesh to a remote part of another mesh that is likely to not correspond to B_i. Such a vertex B_i from mesh *B* might be from a portion of the scanned object that was not captured in the mesh *A*, and thus no pairing should be made to any point on *A*. We have found that excellent registration will result when this distance threshold is set to twice the spacing *s* between range points. Limiting the distance between pairs of corresponding points allows us to perform step 2 (eliminating remote pairs) during the nearest points search in step 1.

The nearest points search can be accelerated considerably by placing the mesh vertices in a uniform subdivision of space based on the distance threshold. Because the triangle size is limited in the mesh creation step, we can search over all triangles within a fixed distance and guarantee that we miss no nearby portion of any triangle. Because we will use this constrained nearest-point search again later, it is worth giving a name to this query. Let nearest_on_mesh(*P*,*d*,*M*) be a routine that returns the nearest position on a mesh *M* to a given point *P*, or that returns nothing if there is no such point within the distance *d*.

Second, we have added the restriction that we never allow boundary points to be part of a match between surfaces. Boundary points are those points that lie on the edge of a triangle and where that edge is not shared by another triangle. Figure 4 illustrates how such matches can drag a mesh in a contrary direction to the majority of the point correspondences.

4.3 Best Rigid Motion

The heart of the iterated closest-point approach is in finding a rigid transformation that minimizes the least-squared distance between the point pairs. Berthold Horn describes a closed-form solution to this problem [Horn 87] that is linear in time with respect to the number of point pairs. Horn's method finds the translation vector *T* and the rotation **R** such that:

$$E = \sum_{i=1}^{n} |A_i - \mathbf{R}(B_i - B_c) - T|^2$$

is minimized, where A_i and B_i are given pairs of positions in 3-space and B_c is the centroid of the B_i. Horn showed that *T* is just the difference between the centroid of the points A_i and the centroid of the points B_i. **R** is found by constructing a cross-covariance matrix between centroid-adjusted pairs of points. The final rotation is given by a unit quaternion that is the eigenvector corresponding to the largest eigenvalue of a matrix constructed from the elements of this cross-covariance matrix. Details can be found in both [Horn 87] and [Besl 92].

As we discussed earlier, not all range points have the same error bounds on their position. We can take advantage of an optional weighting term in Horn's minimization to incorporate the positional uncertainties into the registration process. Let a value in the range from 0 to 1 called *confidence* be a measure of how certain we are of a given range point's position. For the case of structured light scanners, we take the *confidence* of a point *P* on a mesh to be the dot product of the mesh normal *N* at *P* and the vector *L* that points from *P* to the light source of the scanner. (We take the normal at *P* to be the average of the normals of the triangles that meet at *P*.) Additionally, we lower the confidence of vertices near the mesh boundaries to take into account possible error due to false edge extension and curl. We take the confidence of a pair of corresponding points A_i and B_i from two meshes to be the product of their confidences, and we will use w_i to represent this value. The problem is now to find a *weighted* least-squares minimum:

$$E = \sum_{i=1}^{n} w_i |A_i - \mathbf{R}(B_i - B_c) - T|^2$$

The weighted minimization problem is solved in much the same way as before. The translation factor *T* is just the difference between the weighted centroids of the corresponding points. The solution for **R** is described by Horn.

4.4 Alignment in Practice

The above registration method can be made faster by matching increasingly more detailed meshes from a hierarchy. We typically use a mesh hierarchy in which each mesh uses one-forth the number of range points that are used in the next higher level. The less-detailed meshes in this hierarchy are constructed by sub-sampling the range images. Registration begins by running constrained ICP on the lowest-level mesh and then using the resulting transformation as the initial position for the next level up in the hierarchy. The matching distance threshold *d* is halved with each move up the hierarchy.

Besl and McKay describe how to use linear and quadratic extrapolation of the registration parameters to accelerate the alignment process. We use this technique for our alignment at each level in the hierarchy, and find it works well in practice. Details of this method can be found in their paper.

The constrained ICP algorithm registers only two meshes at a time, and there is no obvious extension that will register three or more meshes simultaneously. This is the case with all the registration algorithms we know. If we have meshes *A*, *B*, *C* and *D*, should we register *A* with *B*, then *B* with *C* and finally *C* with *D*, perhaps compounding registration errors? We can minimize this problem by registering all meshes to a single mesh that is created from a cylindrical range image. In this way the cylindrical range image acts as a common anchor for all of the other meshes. Note that if a cylindrical scan covers an object from top to bottom, it captures all the surfaces that lie on the convex hull of the object. This means that,

for almost all objects, there will be some common portions between the cylindrical scan and all linear scans, although the degree of this overlap depends on the extent of the concavities of the object. We used such a cylindrical scan for alignment when constructing the models shown in this paper.

5 Integration: Mesh Zippering

The central step in combining range images is the integration of multiple views into a single model. The goal of integration is to arrive at a description of the overall topology of the object being scanned. In this section we examine how two triangle meshes can be combined into a single surface. The full topology of a surface is realized by zippering new range scans one by one into the final triangle mesh.

Zippering two triangle meshes consists of three steps, each of which we will consider in detail below:

1) Remove overlapping portions of the meshes.
2) Clip one mesh against another.
3) Remove the small triangles introduced during clipping.

5.1 Removing Redundant Surfaces

Before attempting to join a pair of meshes, we eat away at the boundaries of both meshes until they just meet. We remove those triangles in each mesh that are in some sense "redundant," in that the other mesh includes an unbroken surface at that same position in space. Although this step removes triangles from the meshes, we are not discarding data since all range points eventually will be used to find the consensus geometry (Section 6). Given two triangle meshes *A* and *B*, here is the process that removes their redundant portions:

Repeat until both meshes remain unchanged:
  Remove redundant triangles on the boundary of mesh *A*
  Remove redundant triangles on the boundary of mesh *B*

Before we can remove a given triangle *T* from mesh *A*, we need to determine whether the triangle is redundant. We accomplish this by querying mesh *B* using the nearest_on_mesh() routine that was introduced earlier. In particular, we ask for the nearest positions on mesh *B* to the vertices V_1, V_2 and V_3 of *T*. We will declare *T* to be redundant if the three queries return positions on *B* that are within a tolerance distance *d* and if none of these positions are on the boundary of *B*. Figure 7 shows two overlapping surfaces before and after removing their redundant triangles. In some cases this particular decision procedure for removing triangles will leave tiny gaps where the meshes meet. The resulting holes are no larger than the maximum triangle size and we currently fill them in an automatic post-processing step to zippering. Using the fast triangle redundancy check was an implementation decision for the sake of efficiency, not a necessary characteristic of our zippering approach, and it could easily be replaced by a more cautious redundancy check that leaves no gaps. We have not found this necessary in practice.

If we have a measure of confidence of the vertex positions (as we do for structured light scanners), then the above method can be altered to preserve the more confident vertices. When checking to see if the vertices V_1, V_2 and V_3 of *T* lie within the distance tolerance of mesh *B*, we also determine whether at least two of these vertices have a lower confidence measure than the nearby points on *B*. If this is the case, we allow the triangle to be removed. When no more triangles can be removed from the boundaries of either mesh, we drop this confidence value restriction and continue the process until no more changes can be made. This procedure results in a pair of meshes that meet along boundaries of nearly equal confidences.

5.2 Mesh Clipping

We now describe how triangle clipping can be used to smoothly join two meshes that slightly overlap. The left portion of Figure 5 shows two overlapping meshes and the right portion shows the result of clipping. Let us examine the clipping process in greater detail, and for the time being make the assumption that we are operating on two meshes that lie in a common plane.

To clip mesh *A* against the boundary of mesh *B* we first need to add new vertices to the boundary of *B*. Specifically, we place a new vertex wherever an edge of a triangle from mesh *A* intersects the boundary of mesh *B*. Let *Q* be the set of all such new vertices. Together, the new vertices in *Q* and the old boundary vertices of mesh *B* will form a common boundary that the triangles from both meshes will share. Once this new boundary is formed we need to incorporate the vertices *Q* into the triangles that share this boundary. Triangles from mesh *B* need only to be split once for each new vertex to be incorporated (shown in Figure 5, right). Then we need to divide each border triangle from *A* into two parts, one part that lies inside the boundary of *B* that should be discarded and the other part that lies outside of this boundary and should be retained (See Figure 5, middle). The vertices of the retained portions of the triangle are passed to a constrained triangulation routine that returns a set of triangles that incorporates all the necessary vertices (Figure 5, right).

The only modification needed to extend this clipping step to 3-space is to determine precisely how to find the points of intersection *Q*. In 3-space the edges of mesh *A* might very well pass above or below the boundary of *B* instead of exactly intersecting the boundary. To correct for this we "thicken" the boundary of mesh *B*. In essence we create a wall that runs around the boundary of *B* and that is roughly perpendicular to *B* at any given location along the boundary. The portion of the wall at any given edge *E* is a collection of four triangles, as shown in Figure 6. To find the intersection points with the edges of *A*, we only need to note where these edges pass through the wall of triangles. We then move this intersection point down to the nearest position on the edge *E* to which the intersected portion of the wall belongs. The rest of the clipping can proceed as described above.

5.3 Removing Small Triangles

The clipping process can introduce arbitrarily small or thin triangles into a mesh. For many applications this does matter, but in situations where such triangles are undesirable they can easily be removed. We use vertex deletion to remove small triangles: if any of a triangle's altitudes fall below a user-specified threshold we delete one of the triangle's vertices and all the triangles that shared this vertex. We then use constrained triangulation to fill the hole that is left by deleting these triangles (see [Bern 92]). We preferentially delete vertices that were introduced as new vertices during the clipping process. If all of a triangle's vertices are original range points then the vertex opposite the longest side is deleted.

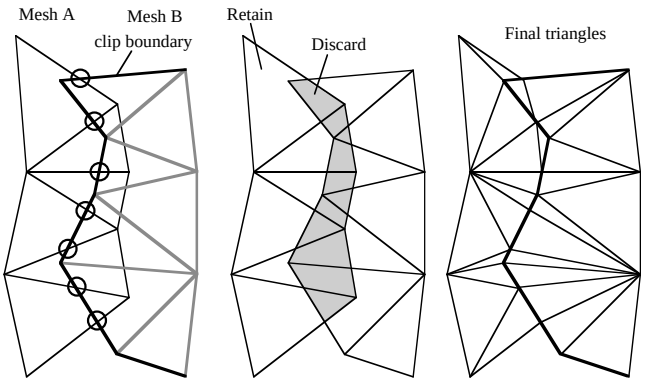


Figure 5: Mesh *A* is clipped against the boundary of mesh *B*. Circles (left) show intersection between edges of *A* and *B*'s boundary. Portions of triangles from *A* are discarded (middle) and then both meshes incorporate the points of intersection (right).

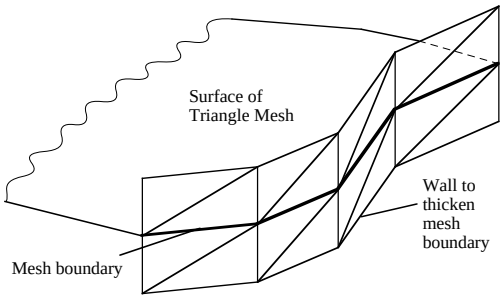


Figure 6: Thickened boundary for clipping in 3-space.

5.4 False Edge Extension

As described in Section 3.2, range points from a structured light scanner that are near an object's silhouette are extended and curled away from the true geometry. These extended edges typically occur at corners. If there is at least one scan that spans both sides of the corner, then our method will correctly reconstruct the surface at the corner. Since we lower the confidence of a surface near the mesh boundaries, triangles at the false edge extensions will be eliminated during redundant surface removal because there are nearby triangles with higher confidence in the scan that spans the corner. For correct integration at a corner, it is the user's responsibility to provide a scan that spans both sides of the corner. Figure 7 illustrates correct integration at a corner in the presence of false edge extension. Unfortunately, no disambiguating scan can be found when an object is highly curved such as a thin cylinder.

Although the problem of false edge extension is discussed in the structured light literature [Businski 92], we know of no paper on surface integration from such range images that addresses or even mentions this issue. We are also unaware of any other integration methods that will correctly determine the geometry of a surface at locations where there are false extensions. Our group has developed a method of reducing false edge extensions when creating the range images (to appear in a forthcoming paper) and we are exploring algorithms that will lessen the effect of such errors during integration. It is our hope that by emphasizing this issue we will encourage others to address this topic in future research on range image integration.

6 Consensus Geometry

When we have zippered the meshes of all the range images together, the resulting triangle mesh captures the *topology* of the scanned object. This mesh may be sufficient for some applications. If surface detail is important, however, we need to fine-tune the *geometry* of the mesh.

The final model of an object should incorporate all the information available about surface detail from each range image of the object. Some of this information may have been discarded when we removed redundant triangles during mesh zippering. We re-introduce the information about surface detail by moving each vertex of our zippered mesh to a consensus position given by a weighted average of positions from the original range images. Vertices are moved only in the direction of the surface normal so that features are not blurred by lateral motion. This is in contrast to unstructured techniques which tend to blur small features isotropically. Our preference for averaging only in the direction of the surface normal is based on the observation that most points in range scans are generally accurately placed with respect to other points in the same scan, but may differ between scans due to alignment errors such as uncorrected optical distortion in the camera. Let M_1, M_2,..., M_n refer to the original triangle meshes created from the range images. Then the three steps for finding the consensus surface are:

1) Find a local approximation to the surface normal.
2) Intersect a line oriented along this normal with each original range image.
3) Form a weighted average of the points of intersection.

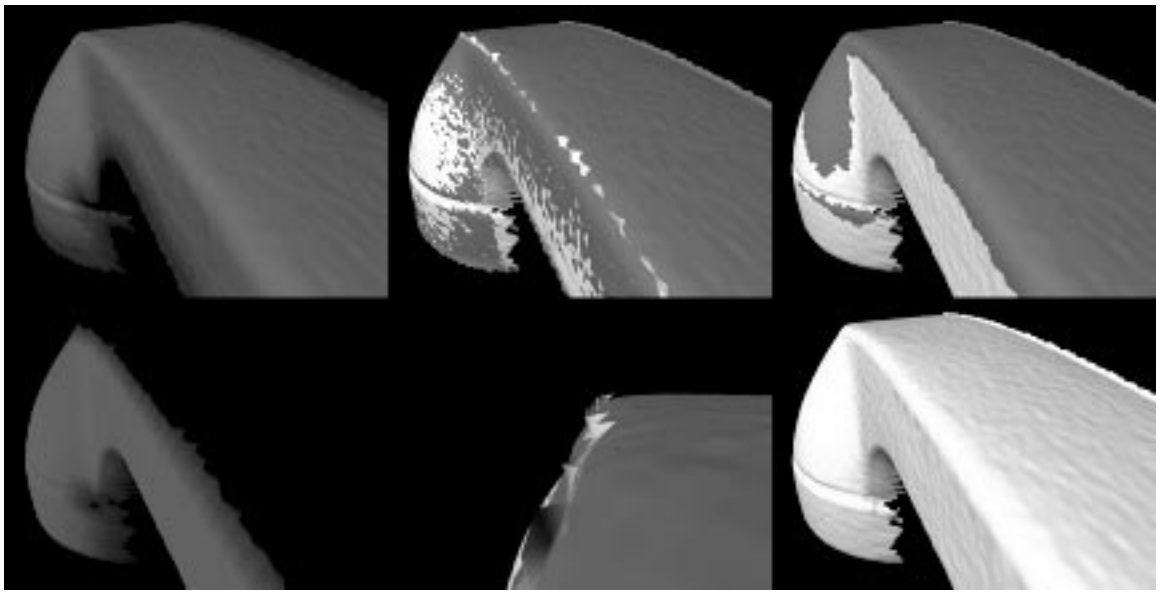


Figure 7: Left (top and bottom): Meshes created from two range images of a telephone. Red denotes locations of high confidence and blue shows low confidence. Note the low confidence at the edges to account for false edge extensions. Top middle: The two meshes (colored red and white) after alignment. Bottom middle: Close-up of aligned meshes that shows a jagged ridge of triangles that is the false edge extension of the white mesh at a corner. Top right: The meshes after redundant surface removal. Bottom right: The meshes after zippering.

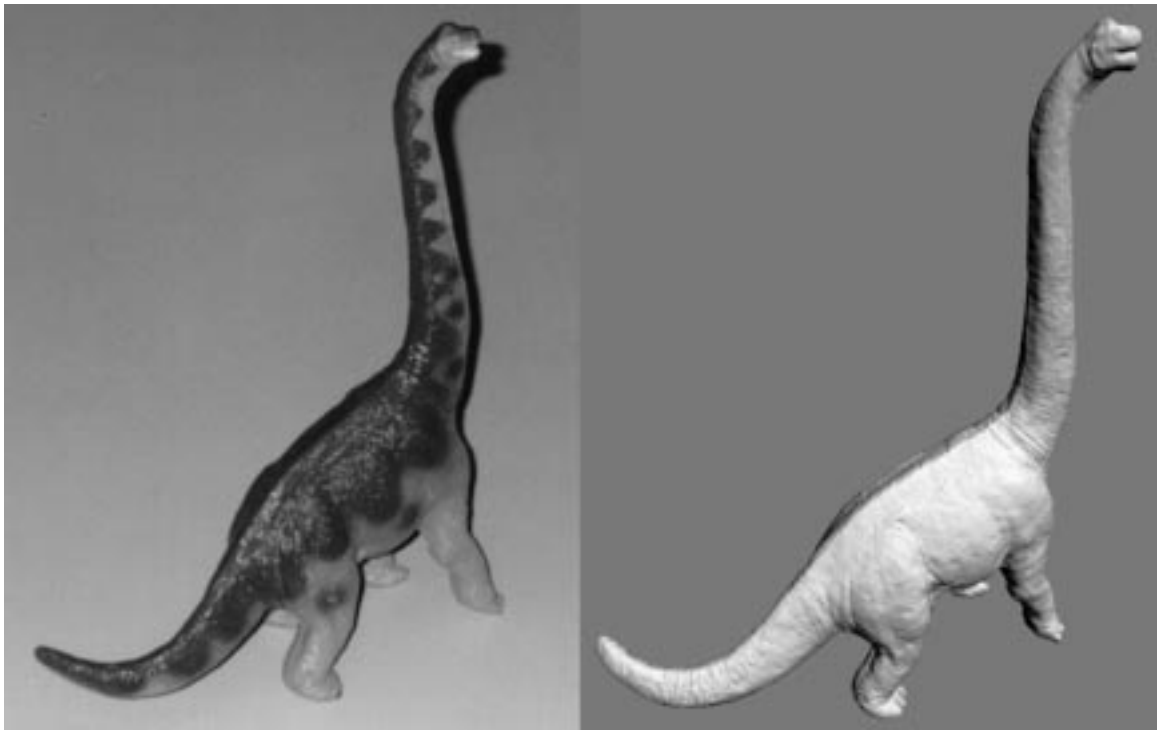


Figure 8: Photograph of a plastic dinosaur model (left) and a polygon mesh created by registering and zippering together 14 range images that were taken of the model (right). The mesh consists of more than 360,000 polygons.

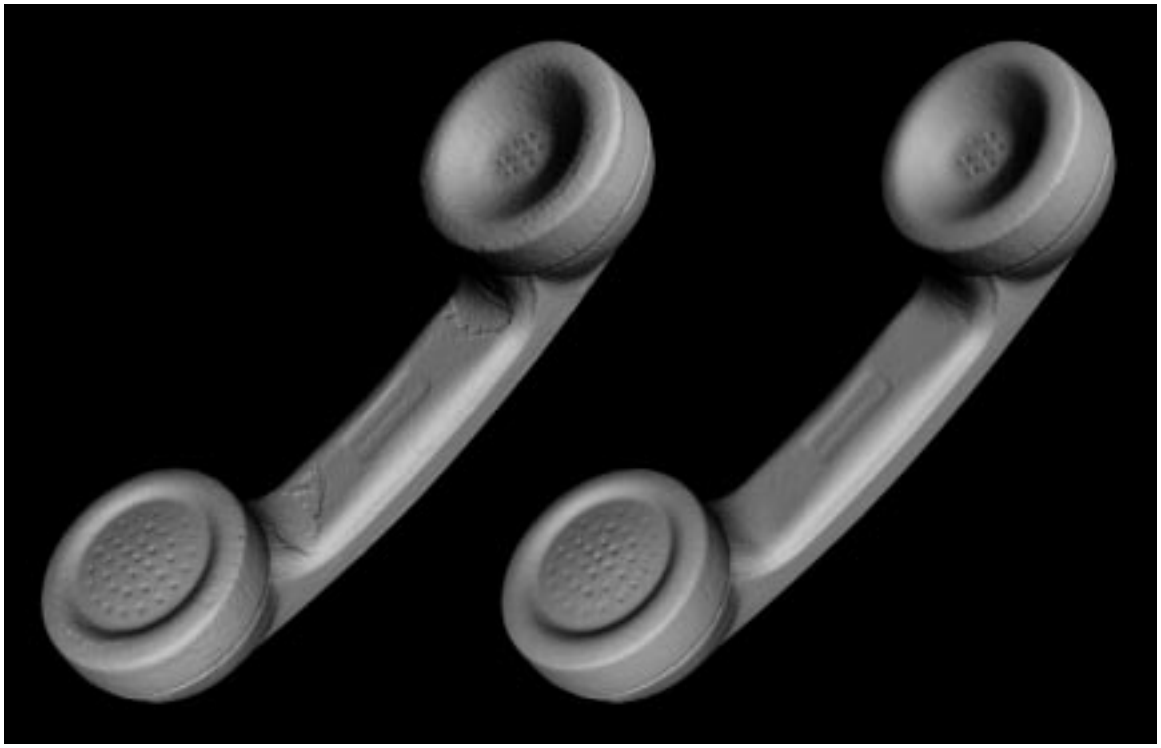


Figure 9: Left: This model of a telephone handset was created by zippering together meshes from ten range images. The mesh consists of more than 160,000 triangles. Right: The final positions of the vertices in the mesh have been moved to an average of nearby positions in the original range images. We call this the consensus geometry.

We approximate the surface normal N at a given vertex V by taking an average over all vertex normals from the vertices in all the meshes M_i that fall within a small sphere centered at V. We then intersect each of the meshes M_i with the line passing through V along the direction N. Let P be the set of all intersections that are near V. We take the consensus position of V to be the average of all the points in P. If we have a measure of confidence for positions on a mesh we use this to weight the average.

7 Results and Discussion

The dinosaur model shown in Figure 8 was created from 14 range images and contains more than 360,000 triangles. Our integration method correctly joined together the meshes at all locations except on the head where some holes due to false edge extensions were filled manually. Such holes should not occur once we eliminate the false extensions in the range images. The dinosaur model was assembled from a larger quantity of range data (measured either in number of scans or number of range points) than any published model known to us. Naturally, we plan to explore the use of automatic simplification methods with our models [Schroeder 92] [Turk 92] [Hoppe 93]. Figure 9 shows a model of a phone that was created from ten range images and contains over 160,000 triangles. The mesh on the right demonstrates that the consensus geometry both reduces noise from the range images without blurring the model's features and also that it eliminates discontinuities at zippered regions.

A key factor that distinguishes our approach from those using unstructured integration ([Hoppe 92] and others) is that our method attempts to retain as much of the triangle connectivity as is possible from the meshes created from the original range images. Our integration process concentrates on a one-dimensional portion of the mesh (the boundary) instead of across an entire two-dimensional surface, and this makes for rapid integration.

Our algorithm shares several characteristics with the approach of Soucy and Laurendeau, which is also a structured integration method [Soucy 92]. The most important difference is the order in which the two methods perform integration and geometry averaging. Soucy's method first creates the final vertex positions by averaging between range images and then stitches together the common surface sets. By determining geometry before connectivity, their approach may be sensitive to artifacts of the stitching process. This is particularly undesirable because their method can create seams between as many as 2^n common surface sets from n range images. Such artifacts are minimized in our approach by performing geometry averaging after zippering.

In summary, we use zippering of triangle meshes followed by refinement of surface geometry to build detailed models from range scans. We expect that in the near future range image technology will replace manual digitization of models in several application areas.

8 Future Work

There are several open problems related to integration of multiple range images. One issue is how an algorithm might automatically determine the next best view to capture more of an object's surface. Another important issue is merging reflectance information (including color) with the geometry of an object. Maybe the biggest outstanding issue is how to create higher-order surface descriptions such as Bezier patches or NURBS from range data, perhaps guided by a polygon model.

Acknowledgments

We thank David Addleman, George Dabrowski and all the other people at Cyberware for the use of a scanner and for educating us about the issues involved in the technology. We thank all the members of our scanner group for numerous helpful discussions. In particular, Brian Curless provided some key insights for interpreting the range data and also wrote code to help this work. Thanks to Phil Lacroute for help with the color figures. This work was supported by an IBM Faculty Development Award, The Powell Foundation, and the National Science Foundation under contract CCR-9157767.

References

[Bern 92] Bern, Marshall and David Eppstein, "Mesh Generation and Optimal Triangulation," Technical Report P92-00047, Xerox Palo Alto Research Center, March 1992.

[Besl 92] Besl, Paul J. and Neil D. McKay, "A Method of Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2 (February 1992), pp. 239–256.

[Boissonnat 84] Boissonnat, Jean-Daniel, "Geometric Structures for Three-Dimensional Shape Representation," *ACM Transactions on Graphics*, Vol. 3, No. 4 (October 1984), pp. 266–286.

[Businski 92] Businski, M., A. Levine and W. H. Stevenson, "Performance Characteristics of Range Sensors Utilizing Optical Triangulation," *IEEE National Aerospace and Electronics Conference*, Vol. 3 (1992), pp. 1230–1236.

[Champleboux 92] Champleboux, Guillaume, Stephane Lavallee, Richard Szeliski and Lionel Brunie, "From Accurate Range Imaging Sensor Calibration to Accurate Model-Based 3-D Object Localization," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Champaign, Illinois, June 15-20, 1992, pp. 83–89.

[Chen 92] Chen, Yang and Gerard Medioni, "Object Modelling by Registration of Multiple Range Images," *Image and Vision Computing*, Vol. 10, No. 3 (April 1992), pp. 145–155.

[Edelsbrunner 92] Edelsbrunner, Herbert and Ernst P. Mücke, "Three-dimensional Alpha Shapes," *Proceedings of the 1992 Workshop on Volume Visualization*, Boston, October 19-20, 1992, pp. 75–82.

[Hoppe 92] Hoppe, Hugues, Tony DeRose, Tom Duchamp, John McDonald and Werner Stuetzle, "Surface Reconstruction from Unorganized Points," *Computer Graphics*, Vol. 26, No. 2 (SIGGRAPH '92), pp. 71–78.

[Hoppe 93] Hoppe, Hugues, Tony DeRose, Tom Duchamp, John McDonald and Werner Stuetzle, "Mesh Optimization," *Computer Graphics Proceedings*, Annual Conference Series (SIGGRAPH '93), pp. 19–26.

[Horn 87] Horn, Berthold K. P., "Closed-Form Solution of Absolute Orientation Using Unit Quaternions," *Journal of the Optical Society of America. A*, Vol. 4, No. 4 (April 1987), pp. 629–642.

[Schroeder 92] Schroeder, William J., Jonathan A. Zarge and William E. Lorensen, "Decimation of Triangle Meshes," *Computer Graphics*, Vol. 26, No. 2 (SIGGRAPH '92), pp. 65–70.

[Soucy 92] Soucy, Marc and Denis Laurendeau, "Multi-Resolution Surface Modeling from Multiple Range Views," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Champaign, Illinois, June 15-20, 1992, pp. 348–353.

[Turk 92] Turk, Greg, "Re-Tiling Polygonal Surfaces," *Computer Graphics*, Vol. 26, No. 2 (SIGGRAPH '92), pp. 55–64.

[Wada 93] Wada, Nobuhiko, Hiroshi Toriyama, Hiromi T. Tanaka and Fumio Kishino, "Reconstruction of an Object Shape from Multiple Incomplete Range Data Sets Using Convex Hulls," *Computer Graphics International '93*, Lausanne, Switzerland, June 21-25, 1993, pp. 193–203.

A Volumetric Method for Building Complex Models from Range Images

Brian Curless and Marc Levoy
Stanford University

Abstract

A number of techniques have been developed for reconstructing surfaces by integrating groups of aligned range images. A desirable set of properties for such algorithms includes: incremental updating, representation of directional uncertainty, the ability to fill gaps in the reconstruction, and robustness in the presence of outliers. Prior algorithms possess subsets of these properties. In this paper, we present a volumetric method for integrating range images that possesses all of these properties.

Our volumetric representation consists of a cumulative weighted signed distance function. Working with one range image at a time, we first scan-convert it to a distance function, then combine this with the data already acquired using a simple additive scheme. To achieve space efficiency, we employ a run-length encoding of the volume. To achieve time efficiency, we resample the range image to align with the voxel grid and traverse the range and voxel scanlines synchronously. We generate the final manifold by extracting an isosurface from the volumetric grid. We show that under certain assumptions, this isosurface is optimal in the least squares sense. To fill gaps in the model, we tessellate over the boundaries between regions seen to be empty and regions never observed.

Using this method, we are able to integrate a large number of range images (as many as 70) yielding seamless, high-detail models of up to 2.6 million triangles.

CR Categories: I.3.5 [Computer Graphics] Computational Geometry and Object Modeling
Additional keywords: Surface fitting, three-dimensional shape recovery, range image integration, isosurface extraction

1 Introduction

Recent years have witnessed a rise in the availability of fast, accurate range scanners. These range scanners have provided data for applications such as medicine, reverse engineering, and digital film-making. Many of these devices generate *range images*; i.e., they produce depth values on a regular sampling lattice. Figure 1 illustrates how an optical triangulation scanner can be used to acquire a range image. By connecting nearest neighbors with triangular elements, one can construct a *range surface* as shown in Figure 1d. Range images are typically formed by sweeping a 1D or 2D sensor linearly across an object or circularly around it, and generally do not contain enough information to reconstruct the entire object being scanned. Accordingly, we require algorithms that can merge multiple range images into a single description of the surface. A set of desirable properties for such a surface reconstruction algorithm includes:

- *Representation of range uncertainty*. The data in range images typically have asymmetric error distributions with primary directions along sensor lines of sight, as illustrated for optical triangulation in Figure 1a. The method of range integration should reflect this fact.
- *Utilization of all range data*, including redundant observations of each object surface. If properly used, this redundancy can reduce sensor noise.
- *Incremental and order independent updating*. Incremental updates allow us to obtain a reconstruction after each scan or small set of scans and allow us to choose the next best orientation for scanning. Order independence is desirable to ensure that results are not biased by earlier scans. Together, they allow for straightforward parallelization.
- *Time and space efficiency*. Complex objects may require many range images in order to build a detailed model. The range images and the model must be represented efficiently and processed quickly to make the algorithm practical.
- *Robustness*. Outliers and systematic range distortions can create challenging situations for reconstruction algorithms. A robust algorithm needs to handle these situations without catastrophic failures such as holes in surfaces and self-intersecting surfaces.
- *No restrictions on topological type*. The algorithm should not assume that the object is of a particular genus. Simplifying assumptions such as "the object is homeomorphic to a sphere" yield useful results in only a restricted class of problems.
- *Ability to fill holes in the reconstruction*. Given a set of range images that do not completely cover the object, the surface reconstruction will necessarily be incomplete. For some objects, no amount of scanning would completely cover the object, because some surfaces may be inaccessible to the sensor. In these cases, we desire an algorithm that can automatically fill these holes with plausible surfaces, yielding a model that is both "watertight" and esthetically pleasing.

In this paper, we present a volumetric method for integrating range images that possesses all of these properties. In the next section, we review some previous work in the area of surface reconstruction. In section 3, we describe the core of our volumetric algorithm. In section 4, we show how this algorithm can be used to fill gaps in the reconstruction using knowledge about the emptiness of space. Next, in section 5, we describe how we implemented our volumetric approach so as to keep time and space costs reasonable. In section 6, we show the results of surface reconstruction from many range images of complex objects. Finally, in section 7 we conclude and discuss limitations and future directions.

2 Previous work

Surface reconstruction from dense range data has been an active area of research for several decades. The strategies have proceeded along two basic directions: reconstruction from unorganized points, and

Authors' Address: Computer Science Department, Stanford University, Stanford, CA 94305
E-mail: {curless,levoy}@cs.stanford.edu
World Wide Web: http://www-graphics.stanford.edu

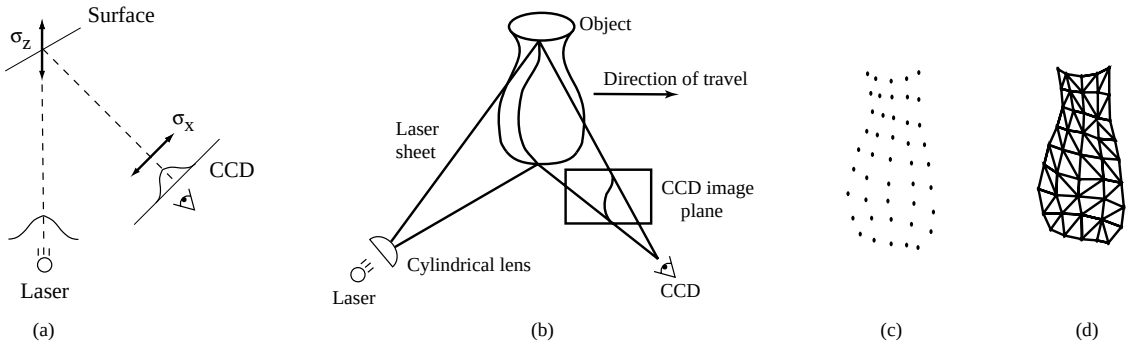


Figure 1. From optical triangulation to a range surface. (a) In 2D, a narrow laser beam illuminates a surface, and a linear sensor images the reflection from an object. The center of the image pulse maps to the center of the laser, yielding a range value. The uncertainty, σ_x, in determining the center of the pulse results in range uncertainty, σ_z along the laser's line of sight. When using the spacetime analysis for optical triangulation [6], the uncertainties run along the lines of sight of the CCD. (b) In 3D, a laser stripe triangulation scanner first spreads the laser beam into a sheet of light with a cylindrical lens. The CCD observes the reflected stripe from which a depth profile is computed. The object sweeps through the field of view, yielding a range image. Other scanner configurations rotate the object to obtain a cylindrical scan or sweep a laser beam or stripe over a stationary object. (c) A range image obtained from the scanner in (b) is a collection of points with regular spacing. (d) By connecting nearest neighbors with triangles, we create a piecewise linear range surface.

reconstruction that exploits the underlying structure of the acquired data. These two strategies can be further subdivided according to whether they operate by reconstructing parametric surfaces or by reconstructing an implicit function.

A major advantage of the unorganized points algorithms is the fact that they do not make any prior assumptions about connectivity of points. In the absence of range images or contours to provide connectivity cues, these algorithms are the only recourse. Among the parametric surface approaches, Boissanat [2] describes a method for Delaunay triangulation of a set of points in 3-space. Edelsbrunner and Mücke [9] generalize the notion of a convex hull to create surfaces called alpha-shapes. Examples of implicit surface reconstruction include the method of Hoppe, et al [16] for generating a signed distance function followed by an isosurface extraction. More recently, Bajaj, et al [1] used alpha-shapes to construct a signed distance function to which they fit implicit polynomials. Although unorganized points algorithms are widely applicable, they discard useful information such as surface normal and reliability estimates. As a result, these algorithms are well-behaved in smooth regions of surfaces, but they are not always robust in regions of high curvature and in the presence of systematic range distortions and outliers.

Among the structured data algorithms, several parametric approaches have been proposed, most of them operating on range images in a polygonal domain. Soucy and Laurendeau [25] describe a method using Venn diagrams to identify overlapping data regions, followed by re-parameterization and merging of regions. Turk and Levoy [30] devised an incremental algorithm that updates a reconstruction by eroding redundant geometry, followed by zippering along the remaining boundaries, and finally a consensus step that reintroduces the original geometry to establish final vertex positions. Rutishauser, et al [24] use errors along the sensor's lines of sight to establish consensus surface positions followed by a re-tessellation that incorporates redundant data. These algorithms typically perform better than unorganized point algorithms, but they can still fail catastrophically in areas of high curvature, as exemplified in Figure 9.

Several algorithms have been proposed for integrating structured data to generate implicit functions. These algorithms can be classified as to whether voxels are assigned one of two (or three) states or are samples of a continuous function. Among the discrete-state volumetric algorithms, Connolly [4] casts rays from a range image accessed as a quad-tree into a voxel grid stored as an octree, and generates results for synthetic data. Chien, et al [3] efficiently generate octree models under the severe assumption that all views are taken from the directions corresponding to the 6 faces of a cube. Li and Crebbin [19] and Tarbox and Gottschlich [28] also describe methods for generating binary voxel grids from range images. None of these methods has been used to generate surfaces. Further, without an underlying continuous function, there are no mechanism for representing range uncertainty or for combining overlapping, noisy range surfaces.

The last category of our taxonomy consists of implicit function methods that use samples of a continuous function to combine structured data. Our method falls into this category. Previous efforts in this area include the work of Grosso, et al [12], who generate depth maps from stereo and average them into a volume with occupancy ramps of varying slopes corresponding to uncertainty measures; they do not, however, perform a final surface extraction. Succi, et al [26] create depth maps from stereo and optical flow and integrate them volumetrically using a straight average. The details of his method are unclear, but they appear to extract an isosurface at an arbitrary threshold. In both the Grosso and Succi papers, the range maps are sparse, the directions of range uncertainty are not characterized, they use no time or space optimizations, and the final models are of low resolution. Recently, Hilton, et al [14] have developed a method similar to ours in that it uses weighted signed distance functions for merging range images, but it does not address directions of sensor uncertainty, incremental updating, space efficiency, and characterization of the whole space for potential hole filling, all of which we believe are crucial for the success of this approach.

Other relevant work includes the method of probabilistic occupancy grids developed by Elfes and Matthies [10]. Their volumetric space is a scalar probability field which they update using a Bayesian formulation. The results have been used for robot navigation, but not for surface extraction. A difficulty with this technique is the fact that the best description of the surface lies at the peak or ridge of the probability function, and the problem of ridge-finding is not one with robust solutions [8]. This is one of our primary motivations for taking an isosurface approach in the next section: it leverages off of well-behaved surface extraction algorithms.

The discrete-state implicit function algorithms described above also have much in common with the methods of extracting volumes from silhouettes [15] [21] [23] [27]. The idea of using backdrops to help carve out the emptiness of space is one we demonstrate in section 4.

3 Volumetric integration

Our algorithm employs a continuous implicit function, $D(\mathbf{x})$, represented by samples. The function we represent is the weighted signed distance of each point $\mathbf{x}$ to the nearest range surface along the line of

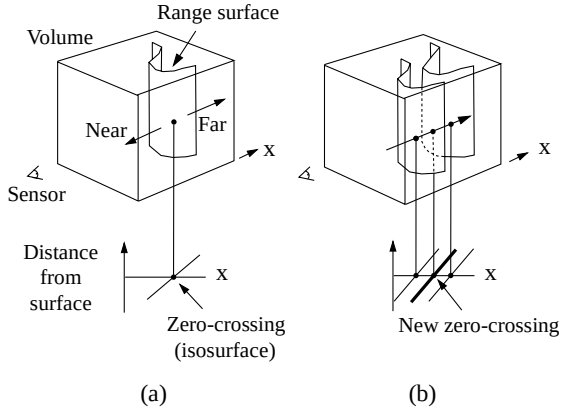


Figure 2. Unweighted signed distance functions in 3D. (a) A range sensor looking down the x-axis observes a range image, shown here as a reconstructed range surface. Following one line of sight down the x-axis, we can generate a signed distance function as shown. The zero crossing of this function is a point on the range surface. (b) The range sensor repeats the measurement, but noise in the range sensing process results in a slightly different range surface. In general, the second surface would interpenetrate the first, but we have shown it as an offset from the first surface for purposes of illustration. Following the same line of sight as before, we obtain another signed distance function. By summing these functions, we arrive at a cumulative function with a new zero crossing positioned midway between the original range measurements.

sight to the sensor. We construct this function by combining signed distance functions $d_1(\mathbf{x})$, $d_2(\mathbf{x})$, ... $d_n(\mathbf{x})$ and weight functions $w_1(\mathbf{x})$, $w_2(\mathbf{x})$, ... $w_n(\mathbf{x})$ obtained from range images 1 ... n. Our combining rules give us for each voxel a cumulative signed distance function, $D(\mathbf{x})$, and a cumulative weight $W(\mathbf{x})$. We represent these functions on a discrete voxel grid and extract an isosurface corresponding to $D(\mathbf{x}) = 0$. Under a certain set of assumptions, this isosurface is optimal in the least squares sense. A full proof of this optimality is beyond the scope of this paper, but a sketch appears in appendix A.

Figure 2 illustrates the principle of combining unweighted signed distances for the simple case of two range surfaces sampled from the same direction. Note that the resulting isosurface would be the surface created by averaging the two range surfaces along the sensor's lines of sight. In general, however, weights are necessary to represent variations in certainty across the range surfaces. The choice of weights should be specific to the range scanning technology. For optical triangulation scanners, for example, Soucy [25] and Turk [30] make the weight depend on the dot product between each vertex normal and the viewing direction, reflecting greater uncertainty when the illumination is at grazing angles to the surface. Turk also argues that the range data at the boundaries of the mesh typically have greater uncertainty, requiring more down-weighting. We adopt these same weighting schemes for our optical triangulation range data.

Figure 3 illustrates the construction and usage of the signed distance and weight functions in 1D. In Figure 3a, the sensor is positioned at the origin looking down the +x axis and has taken two measurements, r_1 and r_2. The signed distance profiles, $d_1(x)$ and $d_2(x)$ may extend indefinitely in either direction, but the weight functions, $w_1(x)$ and $w_2(x)$, taper off behind the range points for reasons discussed below.

Figure 3b is the weighted combination of the two profiles. The combination rules are straightforward:

$$D(\mathbf{x}) = \frac{\Sigma w_i(\mathbf{x}) d_i(\mathbf{x})}{\Sigma w_i(\mathbf{x})} \tag{1}$$

$$W(\mathbf{x}) = \Sigma w_i(\mathbf{x}) \tag{2}$$

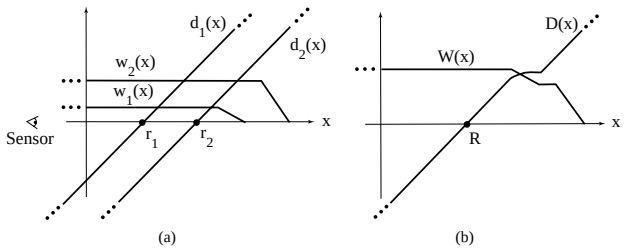


Figure 3. Signed distance and weight functions in one dimension. (a) The sensor looks down the x-axis and takes two measurements, r_1 and r_2. $d_1(x)$ and $d_2(x)$ are the signed distance profiles, and $w_1(x)$ and $w_2(x)$ are the weight functions. In 1D, we might expect two sensor measurements to have the same weight magnitudes, but we have shown them to be of different magnitude here to illustrate how the profiles combine in the general case. (b) $D(x)$ is a weighted combination of $d_1(x)$ and $d_2(x)$, and $W(x)$ is the sum of the weight functions. Given this formulation, the zero-crossing, R, becomes the weighted combination of r_1 and r_2 and represents our best guess of the location of the surface. In practice, we truncate the distance ramps and weights to the vicinity of the range points.

where, $d_i(\mathbf{x})$ and $w_i(\mathbf{x})$ are the signed distance and weight functions from the ith range image.

Expressed as an incremental calculation, the rules are:

$$D_{i+1}(\mathbf{x}) = \frac{W_i(\mathbf{x}) D_i(\mathbf{x}) + w_{i+1}(\mathbf{x}) d_{i+1}(\mathbf{x})}{W_i(\mathbf{x}) + w_{i+1}(\mathbf{x})} \tag{3}$$

$$W_{i+1}(\mathbf{x}) = W_i(\mathbf{x}) + w_{i+1}(\mathbf{x}) \tag{4}$$

where $D_i(\mathbf{x})$ and $W_i(\mathbf{x})$ are the cumulative signed distance and weight functions after integrating the ith range image.

In the special case of one dimension, the zero-crossing of the cumulative function is at a range, R given by:

$$R = \frac{\Sigma w_i r_i}{\Sigma w_i} \tag{5}$$

i.e., a weighted combination of the acquired range values, which is what one would expect for a least squares minimization.

In principle, the distance and weighting functions should extend indefinitely in either direction. However, to prevent surfaces on opposite sides of the object from interfering with each other, we force the weighting function to taper off behind the surface. There is a trade-off involved in choosing where the weight function tapers off. It should persist far enough behind the surface to ensure that all distance ramps will contribute in the vicinity of the final zero crossing, but, it should also be as narrow as possible to avoid influencing surfaces on the other side. To meet these requirements, we force the weights to fall off at a distance equal to half the maximum uncertainty interval of the range measurements. Similarly, the signed distance and weight functions need not extend far in front of the surface. Restricting the functions to the vicinity of the surface yields a more compact representation and reduces the computational expense of updating the volume.

In two and three dimensions, the range measurements correspond to curves or surfaces with weight functions, and the signed distance ramps have directions that are consistent with the primary directions of sensor uncertainty. The uncertainties that apply to range image integration include errors in alignment between meshes as well as errors inherent in the scanning technology. A number of algorithms for aligning sets of range images have been explored and shown to yield excellent results [11][30]. The remaining error lies in the scanner itself. For optical triangulation scanners, for example, this error has been shown to be ellipsoidal about the range points, with the major axis of the ellipse aligned with the lines of sight of the laser [13][24].

Figure 4 illustrates the two-dimensional case for a range curve derived from a single scan containing a row of range samples. In practice, we use a fixed point representation for the signed distance func-

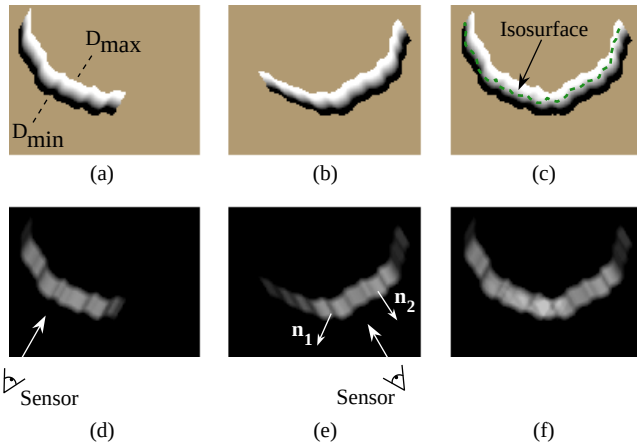


Figure 4. Combination of signed distance and weight functions in two dimensions. (a) and (d) are the signed distance and weight functions, respectively, generated for a range image viewed from the sensor line of sight shown in (d). The signed distance functions are chosen to vary between D_{min} and D_{max}, as shown in (a). The weighting falls off with increasing obliquity to the sensor and at the edges of the meshes as indicated by the darker regions in (e). The normals, $\mathbf{n_1}$ and $\mathbf{n_2}$ shown in (e), are oriented at a grazing angle and facing the sensor, respectively. Note how the weighting is lower (darker) for the grazing normal. (b) and (e) are the signed distance and weight functions for a range image of the same object taken at a 60 degree rotation. (c) is the signed distance function $D(\mathbf{x})$ corresponding to the per voxel weighted combination of (a) and (b) constructed using equations 3 and 4. (f) is the sum of the weights at each voxel, $W(\mathbf{x})$. The dotted green curve in (c) is the isosurface that represents our current estimate of the shape of the object.

tion, which bounds the values to lie between D_{min} and D_{max} as shown in the figure. The values of D_{min} and D_{max} must be negative and positive, respectively, as they are on opposite sides of a signed distance zero-crossing.

For three dimensions, we can summarize the whole algorithm as follows. First, we set all voxel weights to zero, so that new data will overwrite the initial grid values. Next, we tessellate each range image by constructing triangles from nearest neighbors on the sampled lattice. We avoid tessellating over step discontinuities (cliffs in the range map) by discarding triangles with edge lengths that exceed a threshold. We must also compute a weight at each vertex as described above.

Once a range image has been converted to a triangle mesh with a weight at each vertex, we can update the voxel grid. The signed distance contribution is computed by casting a ray from the sensor through each voxel near the range surface and then intersecting it with the triangle mesh, as shown in figure 5. The weight is computed by linearly interpolating the weights stored at the intersection triangle's vertices. Having determined the signed distance and weight we can apply the update formulae described in equations 3 and 4.

At any point during the merging of the range images, we can extract the zero-crossing isosurface from the volumetric grid. We restrict this extraction procedure to skip samples with zero weight, generating triangles only in the regions of observed data. We will relax this restriction in the next section.

4 Hole filling

The algorithm described in the previous section is designed to reconstruct the observed portions of the surface. Unseen portions of the surface will appear as holes in the reconstruction. While this result is an accurate representation of the known surface, the holes are esthetically unsatisfying and can present a stumbling block to follow-on algorithms that expect continuous meshes. In [17], for example, the authors describe a method for parameterizing patches that entails

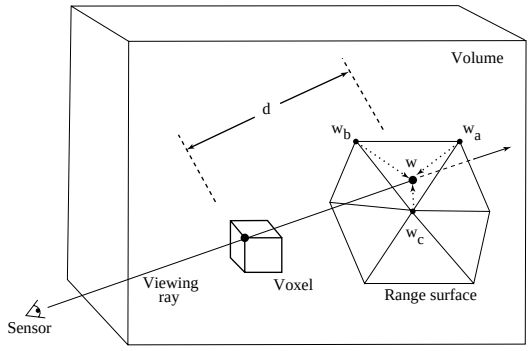


Figure 5. Sampling the range surface to update the volume. We compute the weight, w, and signed distance, d, needed to update the voxel by casting a ray from the sensor, through the voxel onto the range surface. We obtain the weight, w, by linearly interpolating the weights (w_a, w_b, and w_c) stored at neighboring range vertices. Note that for a translating sensor (like our Cyberware scanner), the sensor point is different for each column of range points.

generating evenly spaced grid lines by walking across the edges of a mesh. Gaps in the mesh prevent the algorithm from creating a fair parameterization. As another example, rapid prototyping technologies such as stereolithography typically require a "watertight" model in order to construct a solid replica [7].

One option for filling holes is to operate on the reconstructed mesh. If the regions of the mesh near each hole are very nearly planar, then this approach works well. However, holes in the meshes can be (and frequently are) highly non-planar and may even require connections between unconnected components. Instead, we offer a hole filling approach that operates on our volume, which contains more information than the reconstructed mesh.

The key to our algorithm lies in classifying all points in the volume as being in one of three states: unseen, empty, or near the surface. Holes in the surface are indicated by frontiers between unseen regions and empty regions (see Figure 6). Surfaces placed at these frontiers offer a plausible way to plug these holes (dotted in Figure 6). Obtaining this classification and generating these hole fillers leads to a straightforward extension of the algorithm described in the previous section:

1. Initialize the voxel space to the "unseen" state.
2. Update the voxels near the surface as described in the previous section. As before, these voxels take on continuous signed distance and weight values.
3. Follow the lines of sight back from the observed surface and mark the corresponding voxels as "empty". We refer to this step as *space carving*.
4. Perform an isosurface extraction at the zero-crossing of the signed distance function. Additionally, extract a surface between regions seen to be empty and regions that remain unseen.

In practice, we represent the unseen and empty states using the function and weight fields stored on the voxel lattice. We represent the unseen state with the function values $D(\mathbf{x}) = D_{max}, W(\mathbf{x}) = 0$ and the empty state with the function values $D(\mathbf{x}) = D_{min}, W(\mathbf{x}) = 0$, as shown in Figure 6b. The key advantage of this representation is that we can use the same isosurface extraction algorithm we used in the previous section without the restriction on interpolating voxels of zero weight. This extraction finds both the signed distance and hole fill isosurfaces and connects them naturally where they meet, i.e., at the corners in Figure 6a where the dotted red line meets the dashed green line. Note that the triangles that arise from interpolations across voxels of zero weight are distinct from the others: they are hole fillers.

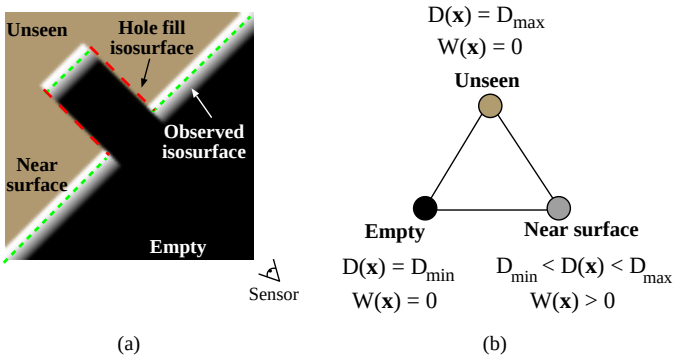


Figure 6. Volumetric grid with space carving and hole filling. (a) The regions in front of the surface are seen as empty, regions in the vicinity of the surface ramp through the zero-crossing, while regions behind remain unseen. The green (dashed) segments are the isosurfaces generated near the observed surface, while the red (dotted) segments are hole fillers, generated by tessellating over the transition from empty to unseen. In (b), we identify the three extremal voxel states with their corresponding function values.

We take advantage of this distinction when smoothing surfaces as described below.

Figure 6 illustrates the method for a single range image, and provides a diagram for the three-state classification scheme. The hole filler isosurfaces are "false" in that they are not representative of the observed surface, but they do derive from observed data. In particular, they correspond to a boundary that confines where the surface could plausibly exist. In practice, we find that many of these hole filler surfaces are generated in crevices that are hard for the sensor to reach.

Because the transition between unseen and empty is discontinuous and hole fill triangles are generated as an isosurface between these binary states, with no smooth transition, we generally observe aliasing artifacts in these areas. These artifacts can be eliminated by prefiltering the transition region before sampling on the voxel lattice using straightforward methods such as analytic filtering or super-sampling and averaging down. In practice, we have obtained satisfactory results by applying another technique: post-filtering the mesh after reconstruction using weighted averages of nearest vertex neighbors as described in [29]. The effect of this filtering step is to blur the hole fill surface. Since we know which triangles correspond to hole fillers, we need only concentrate the surface filtering on the these portions of the mesh. This localized filtering preserves the detail in the observed surface reconstruction. To achieve a smooth blend between filtered hole fill vertices and the neighboring "real" surface, we allow the filter weights to extend beyond and taper off into the vicinity of the hole fill boundaries.

We have just seen how "space carving" is a useful operation: it tells us much about the structure of free space, allowing us to fill holes in an intelligent way. However, our algorithm only carves back from observed surfaces. There are numerous situations where more carving would be useful. For example, the interior walls of a hollow cylinder may elude digitization, but by seeing through the hollow portion of the cylinder to a surface placed behind it, we can better approximate its geometry. We can extend the carving paradigm to cover these situations by placing such a backdrop behind the surfaces being scanned. By placing the backdrop outside of the voxel grid, we utilize it purely for carving space without introducing its geometry into the model.

5 Implementation

5.1 Hardware

The examples in this paper were acquired using a Cyberware 3030 MS laser stripe optical triangulation scanner. Figure 1b illustrates the scanning geometry: an object translates through a plane of laser light while the reflections are triangulated into depth profiles through a CCD camera positioned off axis. To improve the quality of the data, we apply the method of spacetime analysis as described in [6]. The benefits of this analysis include reduced range noise, greater immunity to reflectance changes, and less artifacts near range discontinuities.

When using traditional triangulation analysis implemented in hardware in our Cyberware scanner, the uncertainty in triangulation for our system follows the lines of sight of the expanding laser beam. When using the spacetime analysis, however, the uncertainty follows the lines of sight of the camera. The results described in section 6 of this paper were obtained with one or the other triangulation method. In each case, we adhere to the appropriate lines of sight when laying down signed distance and weight functions.

5.2 Software

The creation of detailed, complex models requires a large amount of input data to be merged into high resolution voxel grids. The examples in the next section include models generated from as many as 70 scans containing up to 12 million input vertices with volumetric grids ranging in size up to 160 million voxels. Clearly, time and space optimizations are critical for merging this data and managing these grids.

5.2.1 Run-length encoding

The core data structure is a run-length encoded (RLE) volume with three run types: empty, unseen, and varying. The varying fields are stored as a stream of varying data, rather than runs of constant value. Typical memory savings vary from 10:1 to 20:1. In fact, the space required to represent one of these voxel grids is usually *less* than the memory required to represent the final mesh as a list of vertices and triangle indices.

5.2.2 Fast volume traversal

Updating the volume from a range image may be likened to inverse volume rendering: instead of reading from a volume and writing to an image, we read from a range image and write to a volume. As a result, we leverage off of a successful idea from the volume rendering community: for best memory system performance, stream through the volume and the image simultaneously in scanline order [18]. In general, however, the scanlines of a range image are not aligned with the scanlines of the voxel grid, as shown in Figure 7a. By suitably resampling the range image, we obtain the desired alignment (Figure 7b). The resampling process consists of a depth rendering of the range surface using the viewing transformation specific to the lines of sight of the range sensor and using an image plane oriented to align with the voxel grid. We assign the weights as vertex "colors" to be linearly interpolated during the rendering step, an approach equivalent to Gouraud shading of triangle colors.

To merge the range data into the voxel grid, we stream through the voxel scanlines in order while stepping through the corresponding scanlines in the resampled range image. We map each voxel scanline to the correct portion of the range scanline as depicted in Figure 7d, and we resample the range data to yield a distance from the range surface. Using the combination rules given by equations 3 and 4, we update the run-length encoded structure. To preserve the linear memory structure of the RLE volume (and thus avoid using linked lists of runs scattered through the memory space), we read the voxel scanlines from the current volume and write the updated scanlines to a second RLE volume; i.e., we double-buffer the voxel grid. Note that depending on the scanner geometry, the mapping from voxels to range image pixels may not be linear, in which case care must be taken to resample appropriately [5].

For the case of merging range data only in the vicinity of the surface, we try to avoid processing voxels distant from the surface. To that end, we construct a binary tree of minimum and maximum depths for every adjacent pair of resampled range image scanlines. Before processing each voxel scanline, we query the binary tree to decide

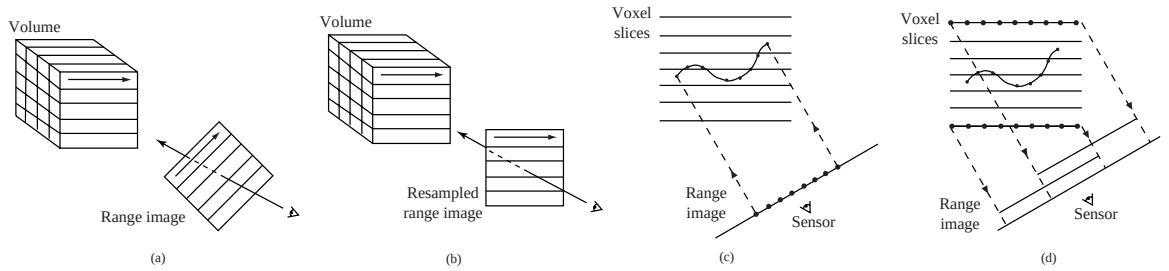


Figure 7. Range image resampling and scanline order voxel updates. (a) Range image scanlines are not in general oriented to allow for coherently streaming through voxel and range scanlines. (b) By resampling the range image, we can obtain the desired range scanline orientation. (c) Casting rays from the pixels on the range image means cutting across scanlines of the voxel grid, resulting in poor memory performance. (d) Instead, we run along scanlines of voxels, mapping them to the correct positions on the resampled range image.

which voxels, if any, are near the range surface. In this way, only relevant pieces of the scanline are processed. In a similar fashion, the space carving steps can be designed to avoid processing voxels that are not seen to be empty for a given range image. The resulting speedups from the binary tree are typically a factor of 15 without carving, and a factor of 5 with carving. We did not implement a brute-force volume update method, however we would expect the overall algorithm described here would be much faster by comparison.

5.2.3 Fast surface extraction

To generate our final surfaces, we employ a Marching Cubes algorithm [20] with a lookup table that resolves ambiguous cases [22]. To reduce computational costs, we only process voxels that have varying data or are at the boundary between empty and unseen.

6 Results

We show results for a number of objects designed to explore the robustness of our algorithm, its ability to fill gaps in the reconstruction, and its attainable level of detail. To explore robustness, we scanned a thin drill bit using the traditional method of optical triangulation. Due to the false edge extensions inherent in data from triangulation scanners [6], this particular object poses a formidable challenge, yet the volumetric method behaves robustly where the zippering method [30] fails catastrophically. The dragon sequence in Figure 11 demonstrates the effectiveness of carving space for hole filling. The use of a backdrop here is particularly effective in filling the gaps in the model. Note that we do not use the backdrop at all times, in part because the range images are much denser and more expensive to process, and also because the backdrop tends to obstruct the path of the object when automatically repositioning it with our motion control platform. Finally, the "Happy Buddha" sequence in Figure 12 shows that our method can be used to generate very detailed, hole-free models suitable for rendering and rapid manufacturing.

Statistics for the reconstruction of the dragon and Buddha models appear in Figure 8. With the optimizations described in the previous section, we were able to reconstruct the observed portions of the surfaces in under an hour on a 250 MHz MIPS R4400 processor. The space carving and hole filling algorithm is not completely optimized, but the execution times are still in the range of 3-5 hours, less than the time spent acquiring and registering the range images. For both models, the RMS distance between points in the original range images and points on the reconstructed surfaces is approximately 0.1 mm. This figure is roughly the same as the accuracy of the scanning technology, indicating a nearly optimal surface reconstruction.

7 Discussion and future work

We have described a new algorithm for volumetric integration of range images, leading to a surface reconstruction without holes. The algorithm has a number of desirable properties, including the representation of directional sensor uncertainty, incremental and order independent updating, robustness in the presence of sensor errors, and the ability to fill gaps in the reconstruction by carving space. Our use of a run-length encoded representation of the voxel grid and synchronized processing of voxel and resampled range image scanlines make the algorithm efficient. This in turn allows us to acquire and integrate a large number of range images. In particular, we demonstrate the ability to integrate up to 70 scans into a high resolution voxel grid to generate million polygon models in a few hours. These models are free of holes, making them suitable for surface fitting, rapid prototyping, and rendering.

There are a number of limitations that prevent us from generating models from an arbitrary object. Some of these limitations arise from the algorithm while others arise from the limitations of the scanning technology. Among the algorithmic limitations, our method has difficulty bridging sharp corners if no scan spans both surfaces meeting at the corner. This is less of a problem when applying our hole-filling algorithm, but we are also exploring methods that will work without hole filling. Thin surfaces are also problematic. As described in section 3, the influences of observed surfaces extend behind their estimated positions for each range image and can interfere with distance functions originating from scans of the opposite side of a thin surface. In this respect, the apexes of sharp corners also behave like thin surfaces. While we have limited this influence as much as possible, it still places a lower limit on the thickness of surface that we can reliably reconstruct without causing artifacts such as thickening of surfaces or rounding of sharp corners. We are currently working to lift this restriction by considering the estimated normals of surfaces.

Other limitations arise from the scanning technologies themselves. Optical methods such as the one we use in this paper can only provide data for external surfaces; internal cavities are not seen. Further, very complicated objects may require an enormous amount of scanning to cover the surface. Optical triangulation scanning has the additional problem that both the laser and the sensor must observe each point on the surface, further restricting the class of objects that can be scanned completely. The reflectance properties of objects are also a factor. Optical methods generally operate by casting light onto an object, but shiny surfaces can deflect this illumination, dark objects can absorb it, and bright surfaces can lead to interreflections. To minimize these effects, we often paint our objects with a flat, gray paint.

Straightforward extensions to our algorithm include improving the execution time of the space carving portion of the algorithm and demonstrating parallelization of the whole algorithm. In addition, more aggressive space carving may be possible by making inferences about sensor lines of sight that return no range data. In the future, we hope to apply our methods to other scanning technologies and to large scale objects such as terrain and architectural scenes.

Model	Scans	Input triangles	Voxel size (mm)	Volume dimensions	Exec. time (min)	Output triangles	Holes
Dragon	61	15 M	0.35	712x501x322	56	1.7 M	324
Dragon + fill	71	24 M	0.35	712x501x322	257	1.8 M	0
Buddha	48	5 M	0.25	407x957x407	47	2.4 M	670
Buddha + fill	58	9 M	0.25	407x957x407	197	2.6 M	0

Figure 8. Statistics for the reconstruction of the dragon and Buddha models, with and without space carving.

Acknowledgments

We would like to thank Phil Lacroute for his many helpful suggestions in designing the volumetric algorithms. Afra Zomorodian wrote the scripting interface for scanning automation. Homan Igehy wrote the fast scan conversion code, which we used for range image resampling. Thanks to Bill Lorensen for his marching cubes tables and mesh decimation software, and for getting the 3D hardcopy made. Matt Pharr did the accessibility shading used to render the color Buddha, and Pat Hanrahan and Julie Dorsey made helpful suggestions for RenderMan tricks and lighting models. Thanks also to David Addleman and George Dabrowski of Cyberware for their help and for the use of their scanner. This work was supported by the National Science Foundation under contract CCR-9157767 and Interval Research Corporation.

References

[1] C.L. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *Proceedings of SIGGRAPH '95 (Los Angeles, CA, Aug. 6-11, 1995)*, pages 109–118. ACM Press, August 1995.

[2] J.-D. Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics*, 3(4):266–286, October 1984.

[3] C.H. Chien, Y.B. Sim, and J.K. Aggarwal. Generation of volume/surface octree from range data. In *The Computer Society Conference on Computer Vision and Pattern Recognition*, pages 254–60, June 1988.

[4] C. I. Connolly. Cumulative generation of octree models from range data. In *Proceedings, Intl. Conf. Robotics*, pages 25–32, March 1984.

[5] B. Curless. *Better optical triangulation and volumetric reconstruction of complex models from range images.* PhD thesis, Stanford University, 1996.

[6] B. Curless and M. Levoy. Better optical triangulation through spacetime analysis. In *Proceedings of IEEE International Conference on Computer Vision*, pages 987–994, June 1995.

[7] A. Dolenc. Software tools for rapid prototyping technologies in manufacturing. *Acta Polytechnica Scandinavica: Mathematics and Computer Science Series*, Ma62:1–111, 1993.

[8] D. Eberly, R. Gardner, B. Morse, S. Pizer, and C. Scharlach. Ridges for image analysis. *Journal of Mathematical Imaging and Vision*, 4(4):353–373, Dec 1994.

[9] H. Edelsbrunner and E.P. Mücke. Three-dimensional alpha shapes. In *Workshop on Volume Visualization*, pages 75–105, October 1992.

[10] A. Elfes and L. Matthies. Sensor integration for robot navigation: combining sonar and range data in a grid-based representation. In *Proceedings of the 26th IEEE Conference on Decision and Control*, pages 1802–1807, December 1987.

[11] H. Gagnon, M. Soucy, R. Bergevin, and D. Laurendeau. Registration of multiple range views for automatic 3-D model building. In *Proceedings 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 581–586, June 1994.

[12] E. Grosso, G. Sandini, and C. Frigato. Extraction of 3D information and volumetric uncertainty from multiple stereo images. In *Proceedings of the 8th European Conference on Artificial Intelligence*, pages 683–688, August 1988.

[13] P. Hebert, D. Laurendeau, and D. Poussart. Scene reconstruction and description: geometric primitive extraction from multiple viewed scattered data. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 286–292, June 1993.

[14] A. Hilton, A.J. Toddart, J. Illingworth, and T. Windeatt. Reliable surface reconstruction from multiple range images. In *Fourth European Conference on Com-*

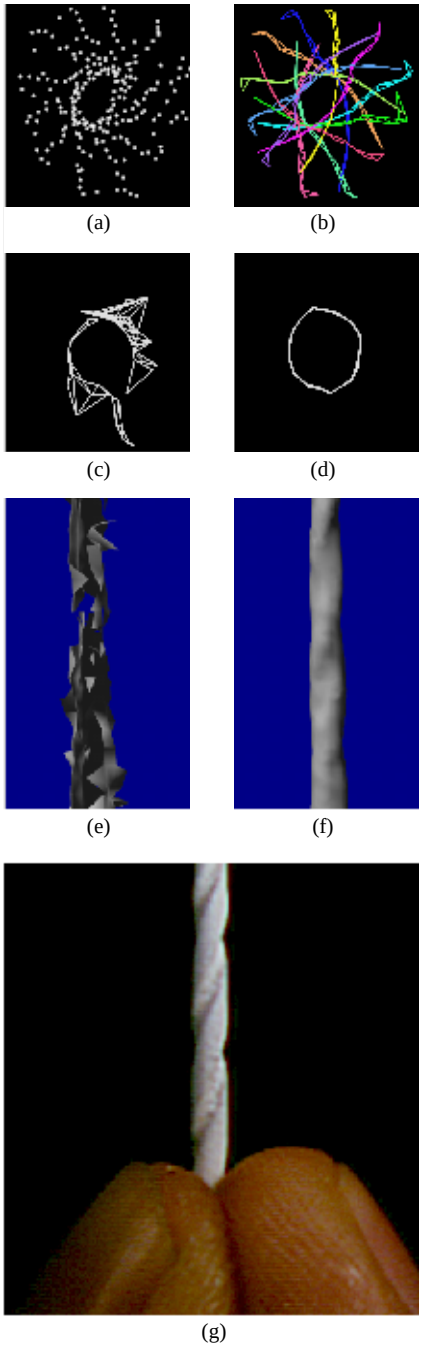


Figure 9. Merging range images of a drill bit. We scanned a 1.6 mm drill bit from 12 orientations at a 30 degree spacing using traditional optical triangulation methods. Illustrations (a) - (d) each show a plan (top) view of a slice taken through the range data and two reconstructions. (a) The range data shown as unorganized points: algorithms that operate on this form of data would likely have difficulty deriving the correct surface. (b) The range data shown as a set of wire frame tessellations of the range data: the false edge extensions pose a challenge to both polygon and volumetric methods. (c) A slice through the reconstructed surface generated by a polygon method: the zippering algorithm of Turk [31]. (d) A slice through the reconstructed surface generated by the volumetric method described in this paper. (e) A rendering of the zippered surface. (f) A rendering of the volumetrically generated surface. Note the catastrophic failure of the zippering algorithm. The volumetric method, however, produces a watertight model. (g) A photograph of the original drill bit. The drill bit was painted white for scanning.

puter Vision, volume I, pages 117–126, April 1996.

[15] Tsai-Hong Hong and M. O. Shneier. Describing a robot's workspace using a sequence of views from a moving camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(6):721–726, November 1985.

[16] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 71–78, July 1992.

[17] V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In these proceedings.

[18] P. Lacroute and M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. In *Proceedings of SIGGRAPH '94 (Orlando, FL, July 24-29, 1994)*, pages 451–458. ACM Press, July 1994.

[19] A. Li and G. Crebbin. Octree encoding of objects from range images. *Pattern Recognition*, 27(5):727–739, May 1994.

[20] W.E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 163–169, July 1987.

[21] W.N. Martin and J.K. Aggarwal. Volumetric descriptions of objects from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):150–158, March 1983.

[22] C. Montani, R. Scateni, and R. Scopigno. A modified look-up table for implicit disambiguation of marching cubes. *Visual Computer*, 10(6):353–355, 1994.

[23] M. Potmesil. Generating octree models of 3D objects from their silhouettes in a sequence of images. *Computer Vision, Graphics, and Image Processing*, 40(1):1–29, October 1987.

[24] M. Rutishauser, M. Stricker, and M. Trobina. Merging range images of arbitrarily shaped objects. In *Proceedings 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 573–580, June 1994.

[25] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):344–358, April 1995.

[26] G. Succi, G. Sandini, E Grosso, and M. Tistarelli. 3D feature extraction from sequences of range data. In *Robotics Research. Fifth International Symposium*, pages 117–127, August 1990.

[27] R. Szeliski. Rapid octree construction from image sequences. *CVGIP: Image Understanding*, 58(1):23–32, July 1993.

[28] G.H Tarbox and S.N. Gottschlich. IVIS: An integrated volumetric inspection system. In *Proceedings of the 1994 Second CAD-Based Vision Workshop*, pages 220–227, February 1994.

[29] G. Taubin. A signal processing approach to fair surface design. In *Proceedings of SIGGRAPH '95 (Los Angeles, CA, Aug. 6-11, 1995)*, pages 351–358. ACM Press, August 1995.

[30] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of SIGGRAPH '94 (Orlando, FL, July 24-29, 1994)*, pages 311–318. ACM Press, July 1994.

[31] Robert Weinstock. *The Calculus of Variations, with Applications to Physics and Engineering*. Dover Publications, 1974.

A Isosurface as least squares minimizer

It is possible to show that the isosurface of the weighted signed distance function is equivalent to a least squares minimization of squared distances between points on the range surfaces and points on the desired reconstruction. The key assumptions are that the range sensor is orthographic and that the range errors are independently distributed along sensor lines of sight. A full proof is beyond the scope of this paper, but we provide a sketch here. See [5] for details.

Consider a region, R, on the desired surface, f, which is observed by n range images. We define the error between an observed range surface and a possible reconstructed surface as the integral of the weighted squared distances between points on the range surface and the reconstructed surface. These distances are taken along the lines of sight of the sensor, commensurate with the predominant directions of uncertainty (see Figure 10). The total error is the sum of the integrals for the n range images:

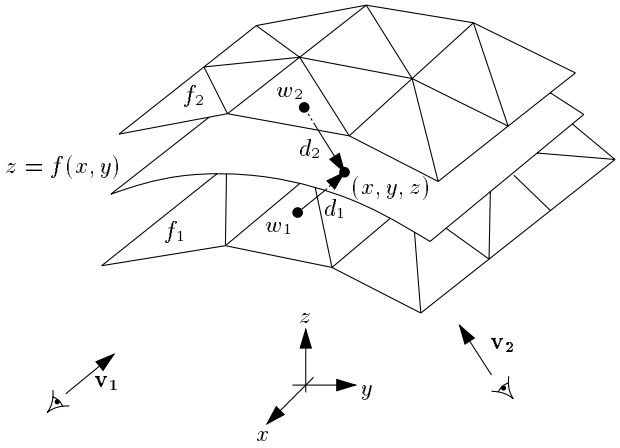


Figure 10. Two range surfaces, f_1 and f_2, are tessellated range images acquired from directions v_1 and v_2. The possible range surface, $z = f(x, y)$, is evaluated in terms of the weighted squared distances to points on the range surfaces taken along the lines of sight to the sensor. A point, (x, y, z), is shown here being evaluated to find its corresponding signed distances, d_1 and d_2, and weights, w_1 and w_2.

$$E(f) = \sum_{i=1}^{n} \iint_{A_i} w_i(s,t,f) d_i(s,t,f)^2 ds dt \tag{6}$$

where each (s, t) corresponds to a particular sensor line of sight for each range image, A_i is the domain of integration for the i'th range image, and $w_i(s, t, f)$ and $d_i(s, t, f)$ are the weights and signed distances taken along the i'th range image's lines of sight.

Now, consider a canonical domain, A, on a parameter plane, (x, y), over which R is a function $z = f(x, y)$. The total error can be rewritten as an integration over the canonical domain:

$$E(z) = \iint_A \sum_{i=1}^{n} \left[w_i(x,y,z) d_i(x,y,z)^2 \right] \left[\mathbf{v_i} \cdot (\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}, -1) \right] dx dy \tag{7}$$

where $\mathbf{v_i}$ is the sensing direction of the i'th range image, and the weights and distances are evaluated at each point, (x, y, z), by first mapping them to the lines of sight of the corresponding range image. The dot product represents a correction term that relates differential areas in A to differential areas in A_i. Applying the calculus of variations [31], we can construct a partial differential equation for the z that minimizes this integral. Solving this equation we arrive at the following relation:

$$\sum_{i=1}^{n} \partial_{\mathbf{v_i}} \left[w_i(x,y,z) d_i(x,y,z)^2 \right] = 0 \tag{8}$$

where $\partial_{\mathbf{v_i}}$ is the directional derivative along $\mathbf{v_i}$. Since the weight associated with a line of sight does not vary along that line of sight, and the signed distance has a derivative of unity along the line of sight, we can simplify this equation to:

$$\sum_{i=1}^{n} w_i(x,y,z) d_i(x,y,z) = 0 \tag{9}$$

This weighted sum of signed distances is the same as what we compute in equations 1 and 2, without the division by the sum of the weights. Since the this divisor is always positive, the isosurface we extract in section 3 is exactly the least squares minimizing surface described here.

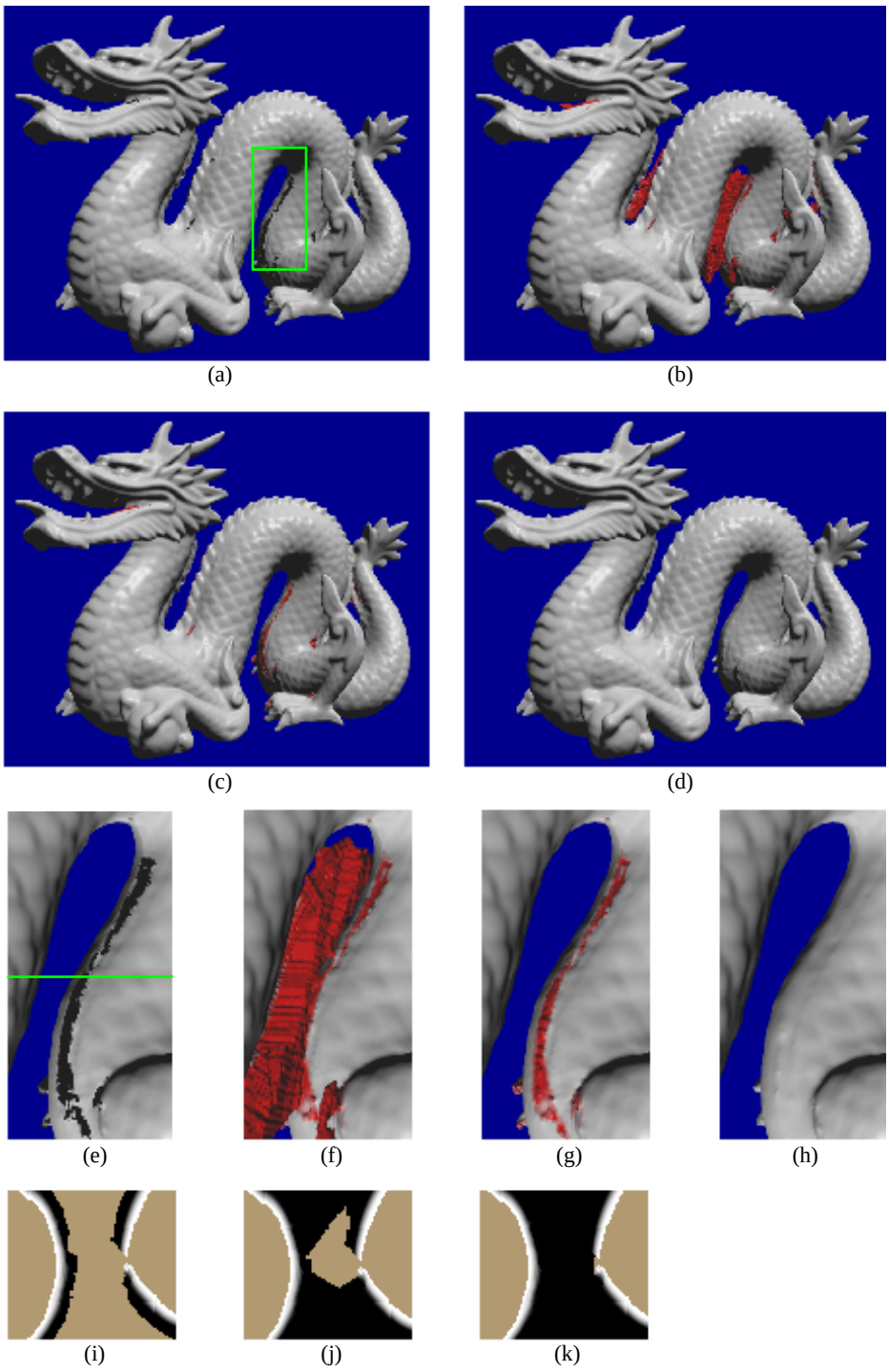
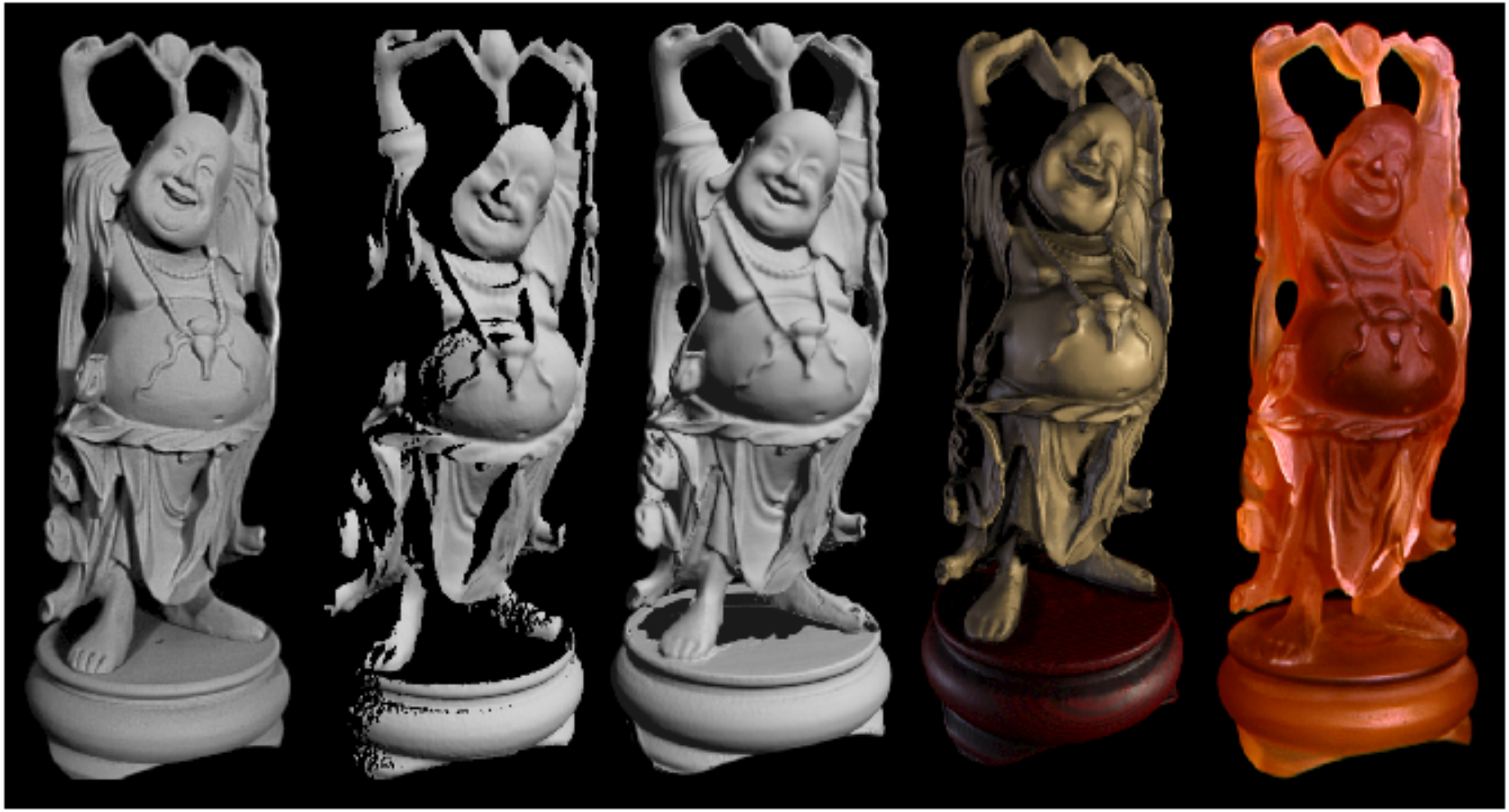


Figure 11. Reconstruction of a dragon. Illustrations (a) - (d) are full views of the dragon. Illustrations (e) - (h) are magnified views of the section highlighted by the green box in (a). Regions shown in red correspond to hole fill triangles. Illustrations (i) - (k) are slices through the corresponding volumetric grids at the level indicated by the green line in (e). (a)(e)(i) Reconstruction from 61 range images without space carving and hole filling. The magnified rendering highlights the holes in the belly. The slice through the volumetric grid shows how the signed distance ramps are maintained close to the surface. The gap in the ramps leads to a hole in the reconstruction. (b)(f)(j) Reconstruction with space carving and hole filling using the same data as in (a). While some holes are filled in a reasonable manner, some large regions of space are left untouched and create extraneous tessellations. The slice through the volumetric grid reveals that the isosurface between the unseen (brown) and empty (black) regions will be connected to the isosurface extracted from the distance ramps, making it part of the connected component of the dragon body and leaving us with a substantial number of false surfaces. (c)(g)(k) Reconstruction with 10 additional range images using "backdrop" surfaces to effect more carving. Notice how the extraneous hole fill triangles nearly vanish. The volumetric slice shows how we have managed to empty out the space near the belly. The bumpiness along the hole fill regions of the belly in (g) corresponds to aliasing artifacts from tessellating over the discontinuous transition between unseen and empty regions. (d)(h) Reconstruction as in (c)(g) with filtering of the hole fill portions of the mesh. The filtering operation blurs out the aliasing artifacts in the hole fill regions while preserving the detail in the rest of the model. Careful examination of (h) reveals a faint ridge in the vicinity of the smoothed hole fill. This ridge is actual geometry present in all of the renderings, (e)-(h). The final model contains 1.8 million polygons and is watertight.



(a) (b) (c) (d) (e)

Figure 12. Reconstruction and 3D hardcopy of the "Happy Buddha". The original is a plastic and rosewood statuette that stands 20 cm tall. Note that the camera parameters for each of these images is different, creating a slightly different perspective in each case. (a) Photograph of the original after spray painting it matte gray to simplify scanning. (b) Gouraud-shaded rendering of one range image of the statuette. Scans were acquired using a Cyberware scanner, modified to permit spacetime triangulation [6]. This figure illustrates the limited and fragmentary nature of the information available from a single range image. (c) Gouraud-shaded rendering of the 2.4 million polygon mesh after merging 48 scans, but before hole-filling. Notice that the reconstructed mesh has at least as much detail as the single range image, but is less noisy; this is most apparent around the belly. The hole in the base of the model corresponds to regions that were not observed directly by the range sensor. (d) RenderMan rendering of an 800,000 polygon decimated version of the hole-filled and filtered mesh built from 58 scans. By placing a backdrop behind the model and taking 10 additional scans, we were able to see through the space between the base and the Buddha's garments, allowing us to carve space and fill the holes in the base. (e) Photograph of a hardcopy of the 3D model, manufactured by 3D Systems, Inc., using stereolithography. The computer model was sliced into 500 layers, 150 microns apart, and the hardcopy was built up layer by layer by selectively hardening a liquid resin. The process took about 10 hours. Afterwards, the model was sanded and bead-blasted to remove the stair-step artifacts that arise during layered manufacturing.

The Digital Michelangelo Project

Marc Levoy

Computer Science Department
Stanford University

Executive overview

Create a 3D computer archive of the principal statues and architecture of Michelangelo

Scholarly motivations
- pushes technology
- scientific tool
- cultural experiment
- lasting archive

Commercial motivations
- virtual museums
- art reproduction
- 3D stock photography
- 2nd generation multimedia

Outline of talk

- hardware and software
- scanning the David
- ~~acquiring a big light field~~
- implications of 3D scanning
- lessons learned from the project
- the problem of the Forma Urbis Romae

Scanners used in the Digital Michelangelo Project

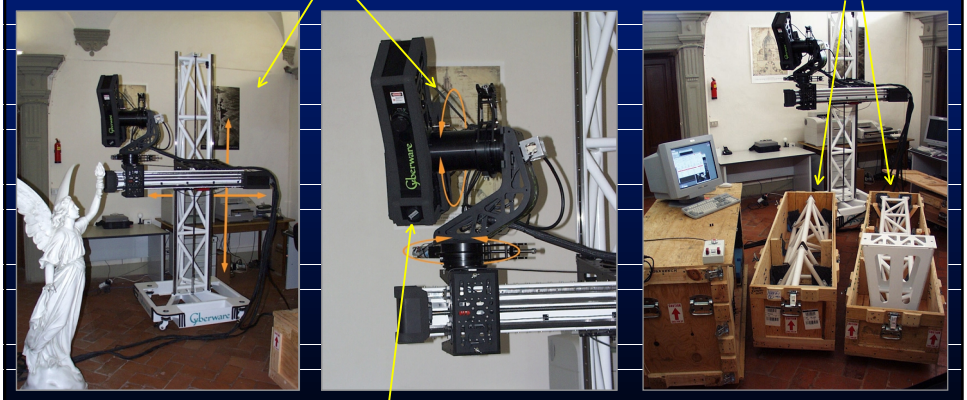
1. Cyberware
   - main 3D scanner for statues
   - planar light field scanner

2. Faro + 3D Scanners
   - for tight spots
   - handheld light field scanner?

3. Cyra
   - for architecture
   - low-res models for view planning?

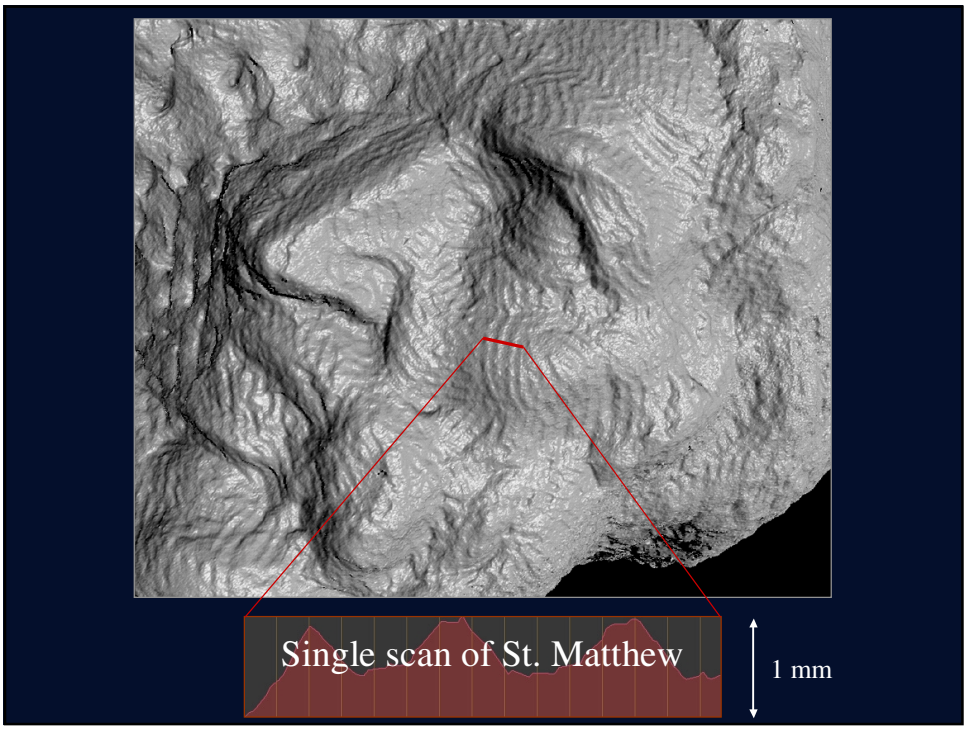
- *All scanners acquire range and color*

Laser triangulation scanner customized for large statues

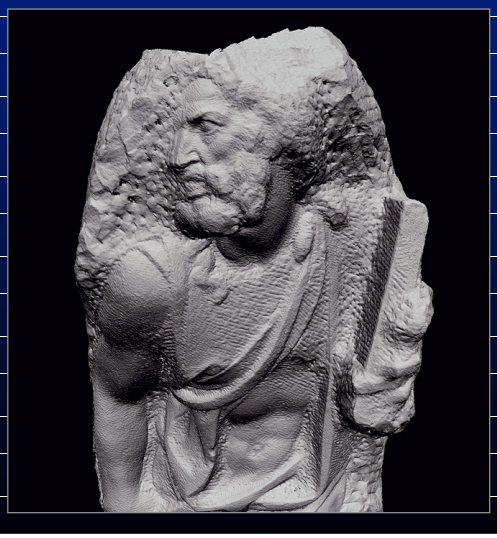


Scanning St. Matthew

working in the museum | scanning geometry | scanning color



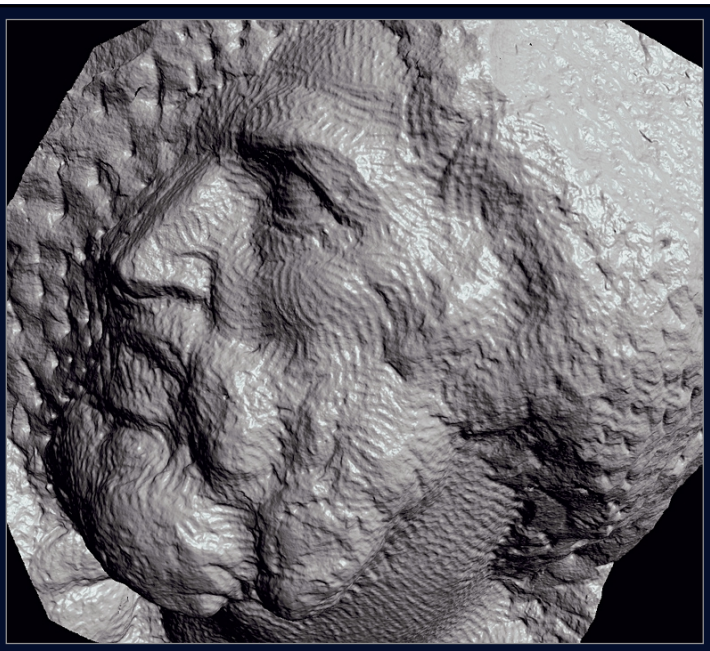
Our scan of St. Matthew



- 104 scans
- 800 million polygons
- 4,000 color images
- 15 gigabytes
- 1 week of scanning

Post-processing pipeline

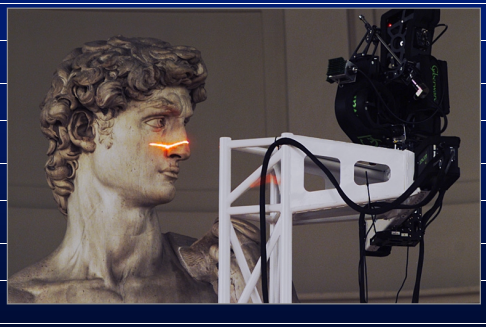
- range data
  - align scans from different gantry positions
  - combine using a volumetric algorithm
  - fill holes using space carving
- color data
  - compensate for ambient lighting
  - discard shadows or reflections
  - factor out surface orientation



Artificial surface reflectance

Estimated diffuse reflectance

Scanning the David



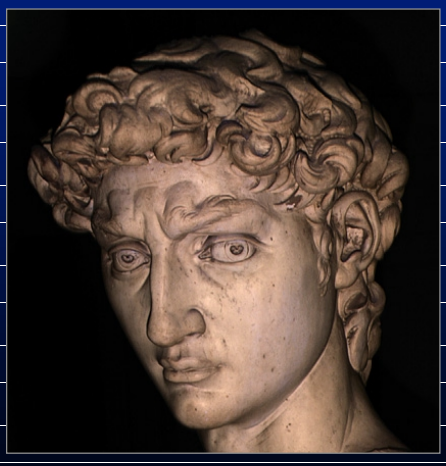
maximum height of gantry: 7.5 meters

weight including subbase: 800 kilograms

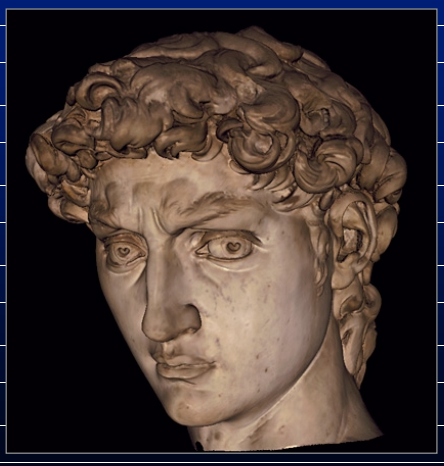
Statistics about the scan

- 480 individually aimed scans
- 2 billion polygons
- 7,000 color images
- 32 gigabytes
- 30 nights of scanning
- 1,080 man-hours
- 22 people

Head of Michelangelo's David

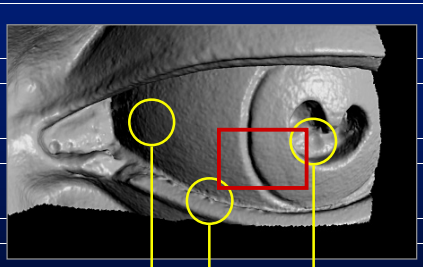


photograph



computer model

David's left eye

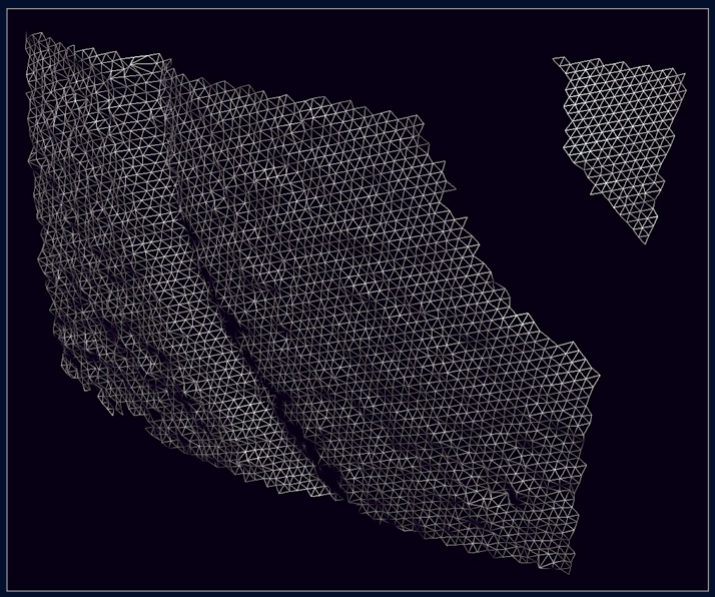


- 0.25mm model
  - rendering of full statue would be 20,000 pixels high
- space carving used to fill holes

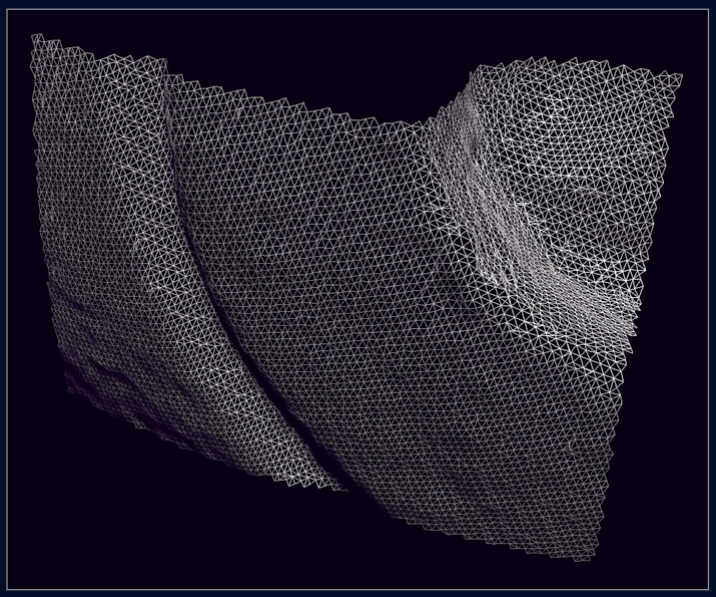
holes from Michelangelo's drill

artifacts from space carving

noise from laser scatter

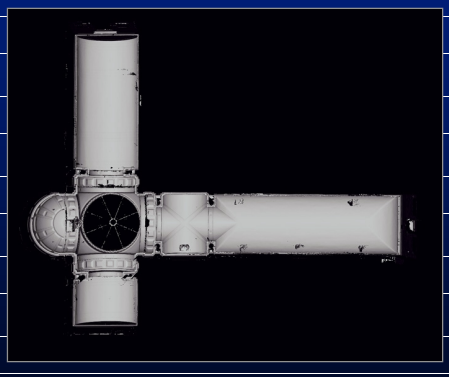


Single scan of David's cornea



Mesh constructed from several scans

Model of Galleria dell'Accademia



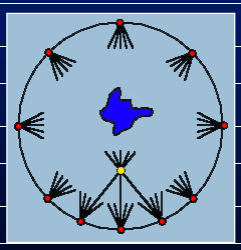
- 4mm model
- 15 million polygons
- Cyra time-of-flight scanner

Computer representations of architectural objects

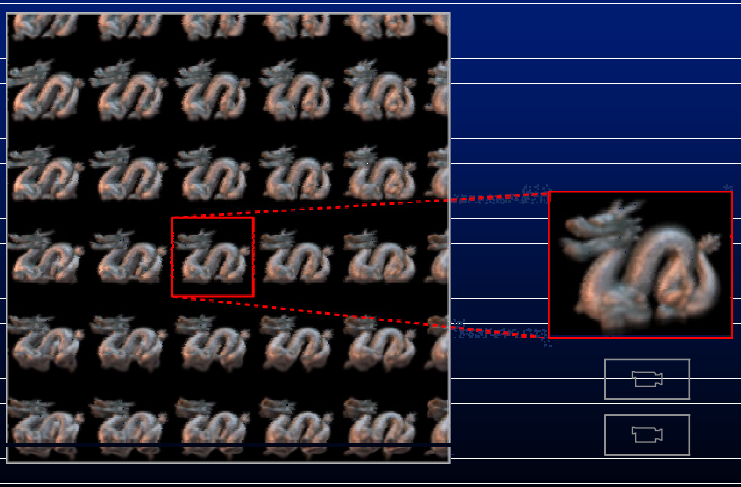
- unstructured mesh
- line drawings
- structured 3D model

Light field rendering

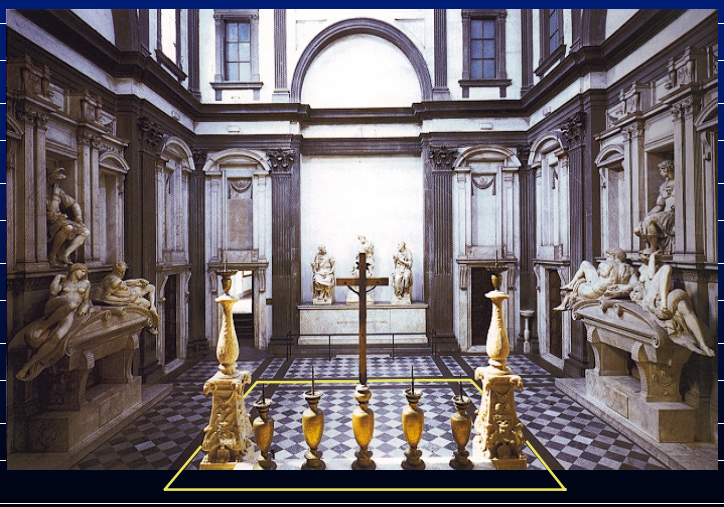
- a form of image-based rendering (IBR)
- make new views by rebinning old views
- Advantages
  - doesn't need a 3D model
  - less computation than rendering a model
  - rendering cost independent of scene complexity
- Disadvantages
  - fixed lighting
  - static scene geometry
  - must stay outside convex hull of object



A light field is an array of images

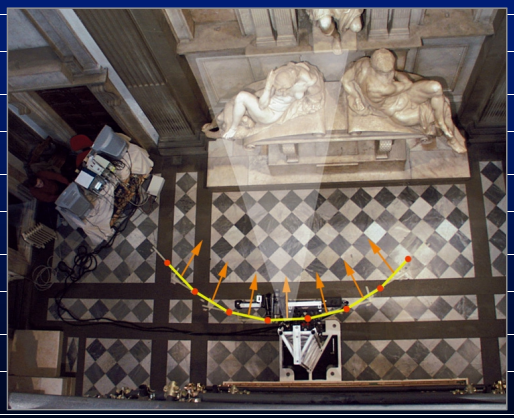


Our planned light field of the Medici Chapel

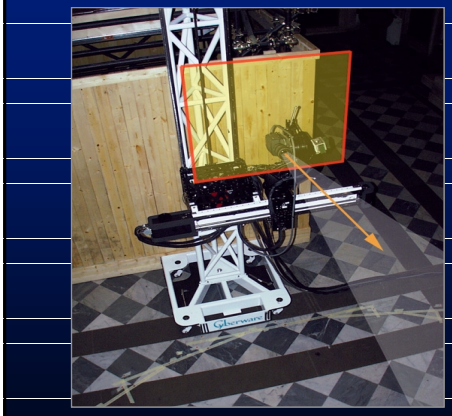


What got in the way of this plan

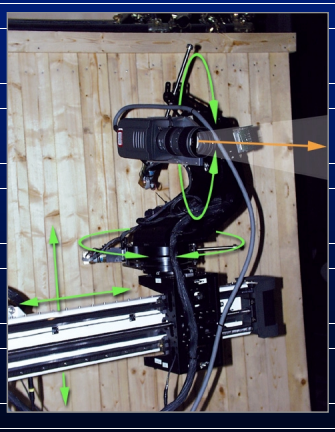
Acquiring a light field of Michelangelo's statue of Night



the light field consists of 7 slabs, each 70cm x 70cm



each slab contains 56 x 56
images spaced 12.5mm apart



the camera is always aimed
at the center of the statue

Sample image from center slab

Statistics about the light field

- 1300 x 1000 pixels per image
- 56 x 56 x 7 = 21,952 images
- 16 gigabytes (using 6:1 JPEG)
- 35 hours of shooting (over 4 nights)
- also acquired a 0.25mm 3D model of statue

Implications of 3D scanning on the viewing of art

- type of reproduction
  - scripted computer graphics
  - interactive computer graphics
  - physical copy
- pros and cons
  - \+ flexible viewing
  - \+ increased accessibility
  - – increased ubiquity
  - – separation from context

Flexible viewpoint

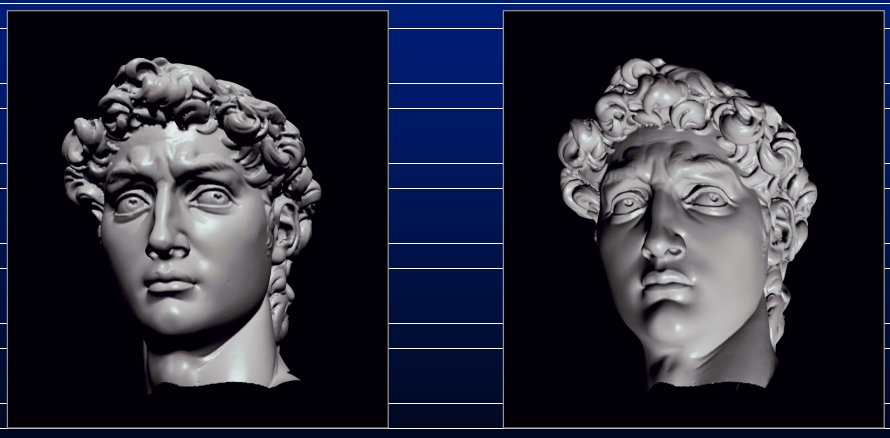
classic 3/4 view

left profile

Flexible viewpoint

face-on view

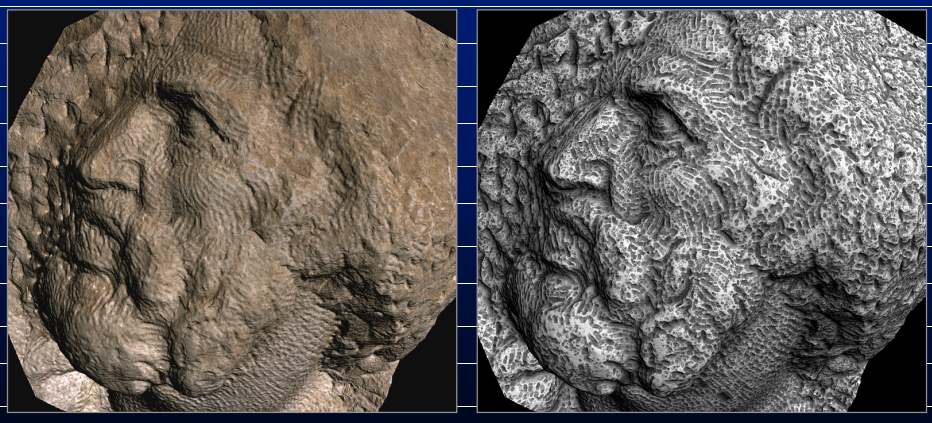
Flexible lighting



lit from above

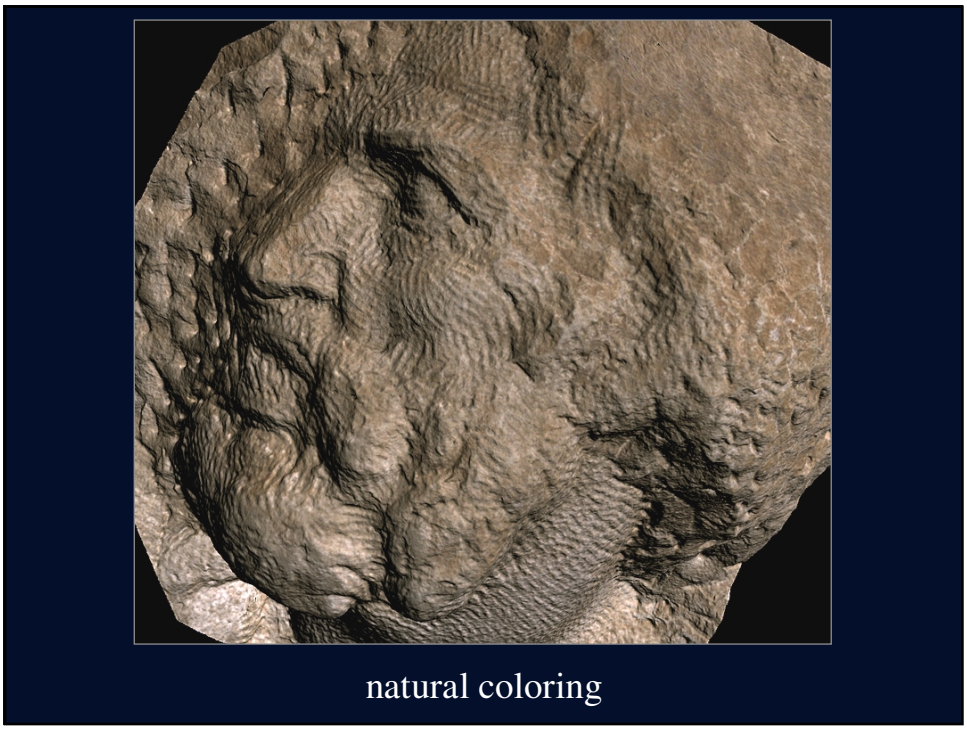
lit from below

Flexible shading

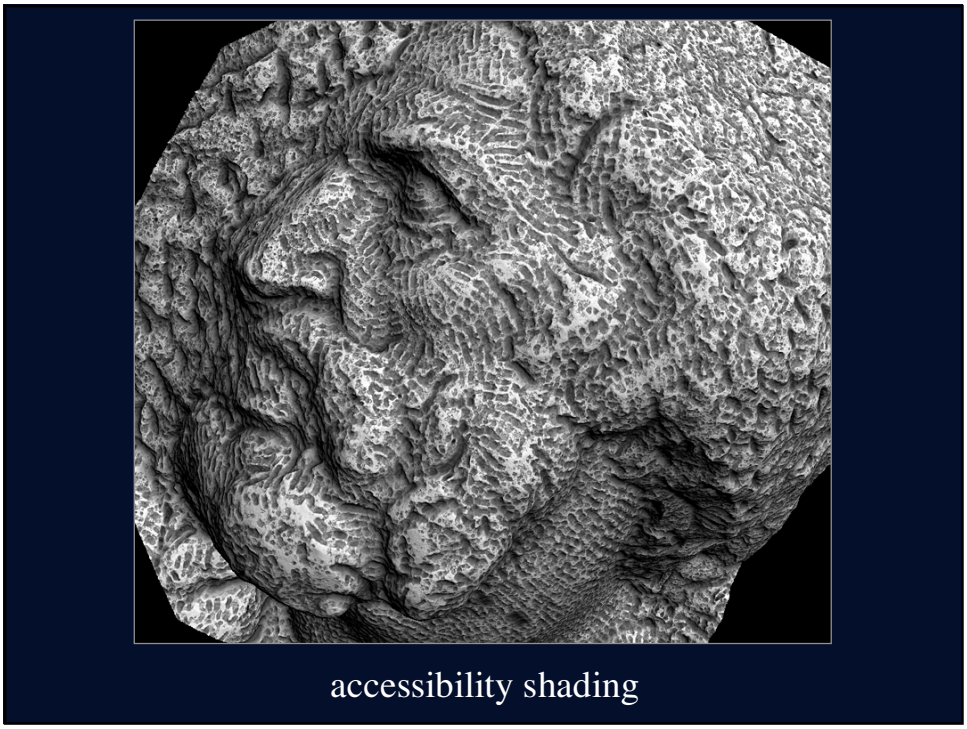


natural coloring

accessibility shading



natural coloring



accessibility shading

Implications of 3D scanning on the viewing of art

- type of reproduction
  - – scripted computer graphics
  - – interactive computer graphics
  - – physical copy
- pros and cons
  - + flexible viewing
  - ⇒ + increased accessibility
  - – increased ubiquity
  - – separation from context

Implications of 3D scanning for art historians

- restoration record
- permanent archive
- diagnostic maps
- geometric calculations
- projection of images onto statues

Diagnostic imaging of David

under white light

under ultraviolet light

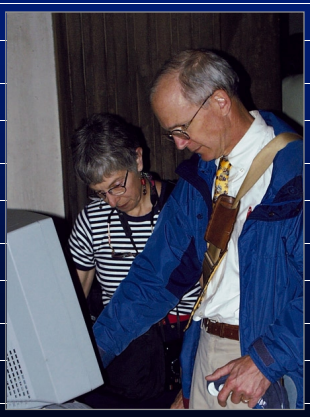
Implications of 3D scanning for art historians

- restoration record
- permanent archive
- diagnostic maps
- ⇒ geometric calculations
- projection of images onto statues

Implications of 3D scanning for educators and museums

- virtual exhibitions
- augmented exhibitions
- enhanced documentaries
- interactive multimedia
- physical replicas

Letting the tourists play with our model of Dawn



They came...

Letting the tourists play with our model of Dawn

They saw...

Letting the tourists play with our model of Dawn

They played...

What really happened?

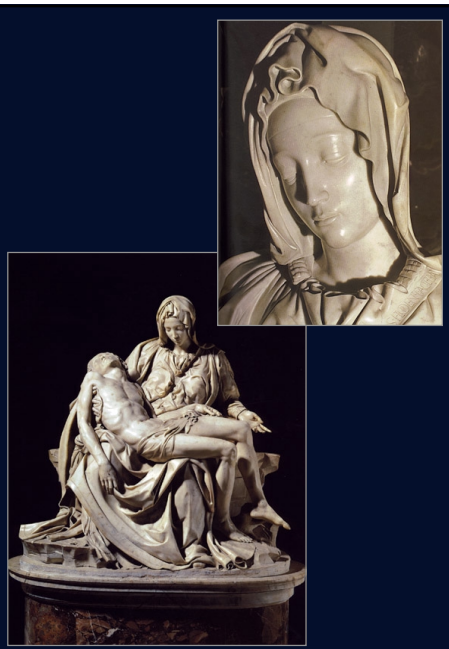
- Kids immediately crowd around.
  Some adults step right up; others need invitations.
- Kids but don't take turns very well.
  Some adults don't either.
- A woman will try it only if a man is not nearby.
  Same for girls and boys.
- Adults usually rotate the statue slowly.
  Kids fly around wildly, but are surprisingly good at it.

What really happened?

- It's amazing how much trouble people can get into.
  Zooming too close is the worst offender.
- People enjoy changing the lighting
  as much as they do rotating the statue.
- People are fascinated by the raw 3D points,
  which they see when the model is in motion.
- People spend a lot of time looking back and forth
  between the screen and the real statue.

Implications of 3D scanning for educators and museums

- virtual exhibitions
- augmented exhibitions
- ⇒ enhanced documentaries
- interactive multimedia
- physical replicas



Michelangelo's Pieta

handmade replica

Logistical challenges

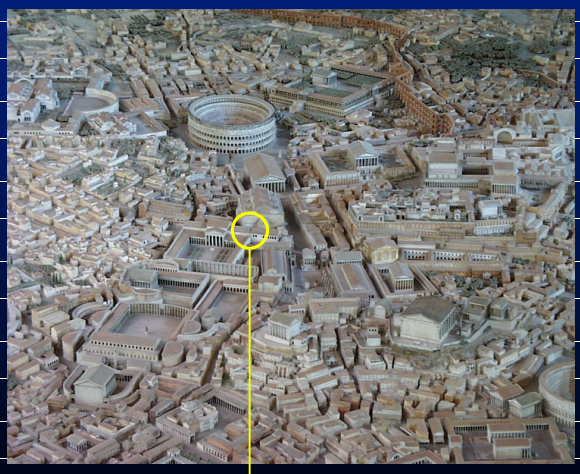
- size of the datasets
- safety for the statues
- intellectual property rights

Lessons learned

- hardware and software
  - variable standoff distance
  - tracking of gantry, not manual alignment of scans
  - autocalibration, not stiff gantry
  - automatic view planning
- logistics
  - scan color quickly - things change
  - need a large team - scanning is tedious work
  - post-processing takes time and people
  - 50% of time on first 90%, 50% on next 9%, ignore last 1%

Il Plastico:
a model of ancient Rome

- made in the 1930's
- measures 60 feet on a side
- at the Museum of Roman Civilization



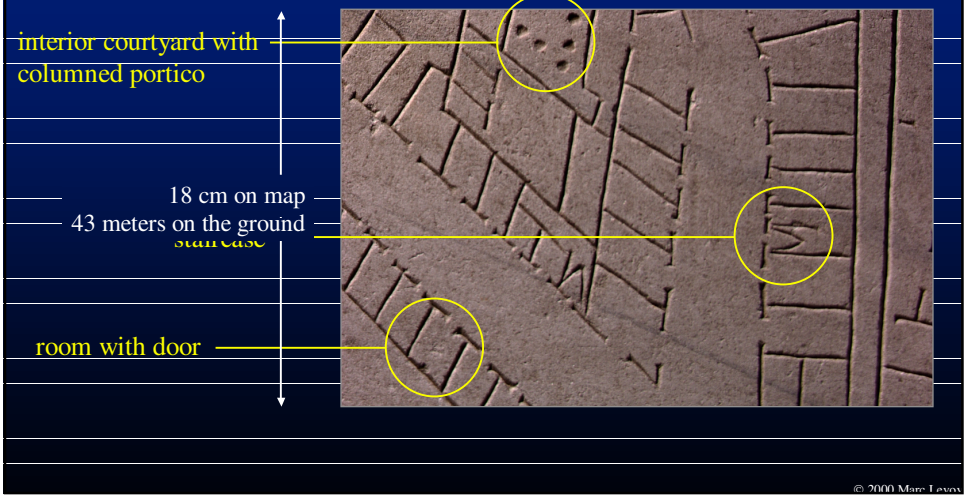
The Forma Urbis Romae: a map of ancient Rome

- carved circa 200 A.D.
- 60 wide x 45 feet high
- marble, 4 inches thick
- showed the entire city at 1:240
- single most important document about ancient Roman topography

its back wall still exists, and on it was hung...

Fragment #10g

Fragment #10g



Solving the jigsaw puzzle

- 1,163 fragments
  - 200 identified
  - 500 unidentified
  - 400 unincised
- 15% of map remains
  - but strongly clustered
- available clues
  - fragment shape (2D or 3D)
  - incised patterns
  - marble veining
  - matches to ruins

Scanning the fragments

uncrating...

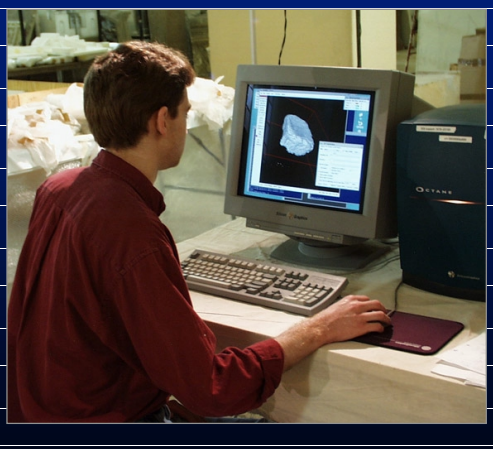
Scanning the fragments

positioning...

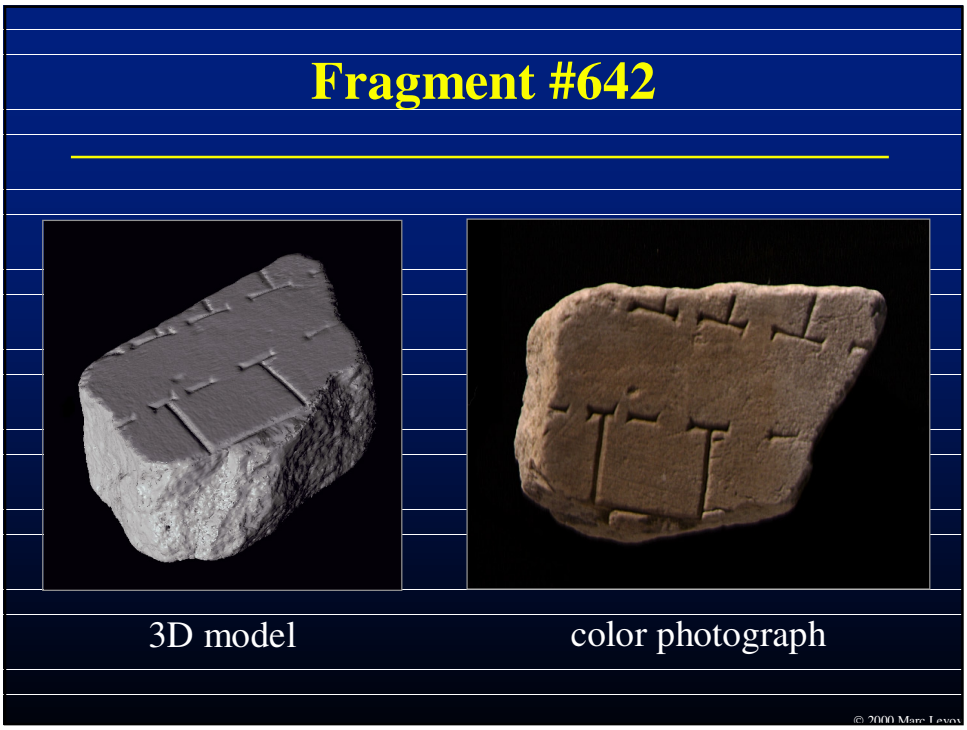
Scanning the fragments

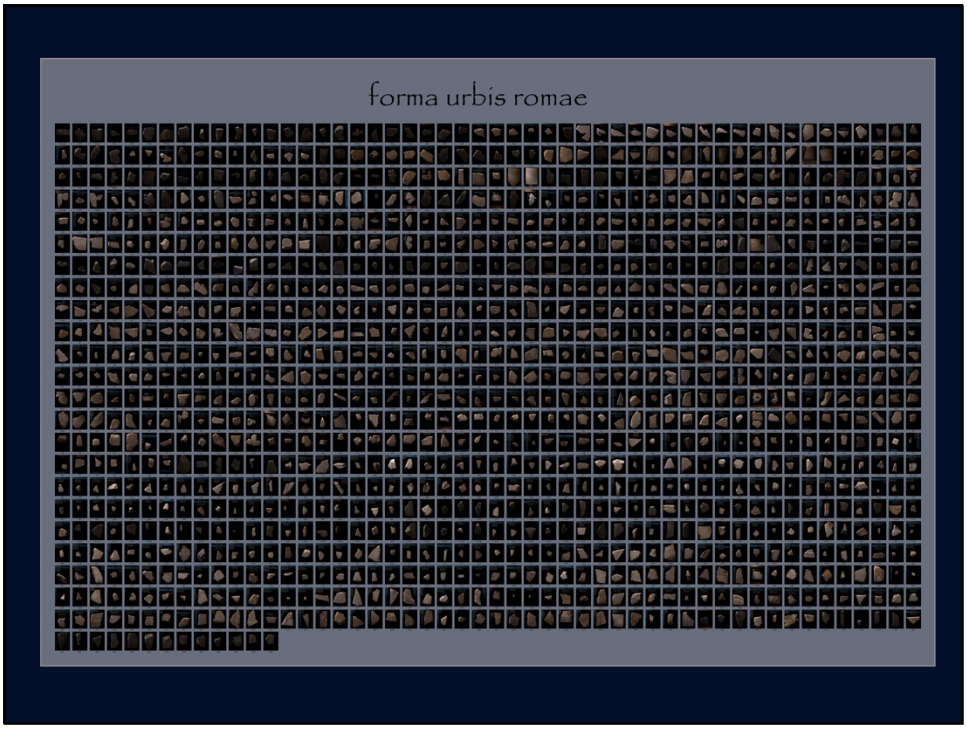
scanning...

Scanning the fragments



aligning...


Fragment #642
3D model
color photograph
© 2000 Marc Levoy


forma urbis romae

Acknowledgements

Faculty and staff

Prof. Brian Curless, John Gerth, Jelena Curless, Prof. Marc Levoy, Lisa Pacelle, Domi Pitturo, Dr. Kari Pulli

Graduate students

Sean Anderson, Barbara Caputo, James Davis, Dave Koller, Lucas Pereira, Szymon Rusinkiewicz, Jonathan Shade, Marco Tarini, Daniel Wood

Undergraduates

Alana Chan, Kathryn Chinn, Jeremy Ginsberg, Matt Ginzton, Unnur Gretarsdottir, Rahul Gupta, Wallace Huang, Dana Katter, Ephraim Luft, Dan Perkel, Semira Rahemtulla, Alex Roetter, Joshua David Schroeder, Maisie Tsui, David Weekly

In Florence

Dott.ssa Cristina Acidini, Dott.ssa Franca Falletti, Dott.ssa Licia Bertani, Alessandra Marino, Matti Auvinen

In Rome

Prof. Eugenio La Rocca, Dott.ssa Susanna Le Pera, Dott.ssa Anna Somella, Dott.ssa Laura Ferrea

In Pisa

Roberto Scopigno

Sponsors

Interval Research, Paul G. Allen Foundation for the Arts, Stanford University

Equipment donors

Cyberware, Cyra Technologies, Faro Technologies, Intel, Silicon Graphics, Sony, 3D Scanners

http://graphics.stanford.edu/projects/mich/ levoy@cs.stanford.edu