

Introducción a la Fotografía 3D

UBA/FCEN Marzo 27 – Abril 12 2013

Clase 5 : Lunes Abril 8

Gabriel Taubin

Brown University

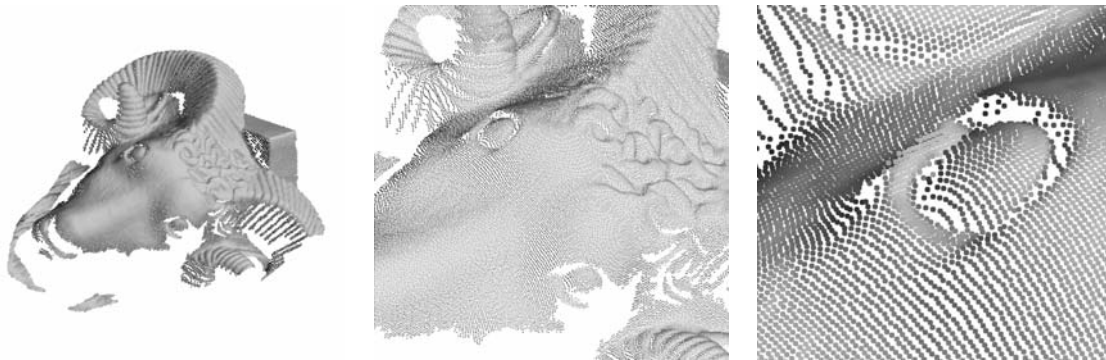


BROWN

Course Schedule

- Introduction
- The Mathematics of 3D Triangulation
- 3D Scanning with Swept-Planes
- Camera and Swept-Plane Light Source Calibration
- ***Reconstruction and Visualization using Point Clouds***
- Combining Point Clouds Recovered from Multiple Views

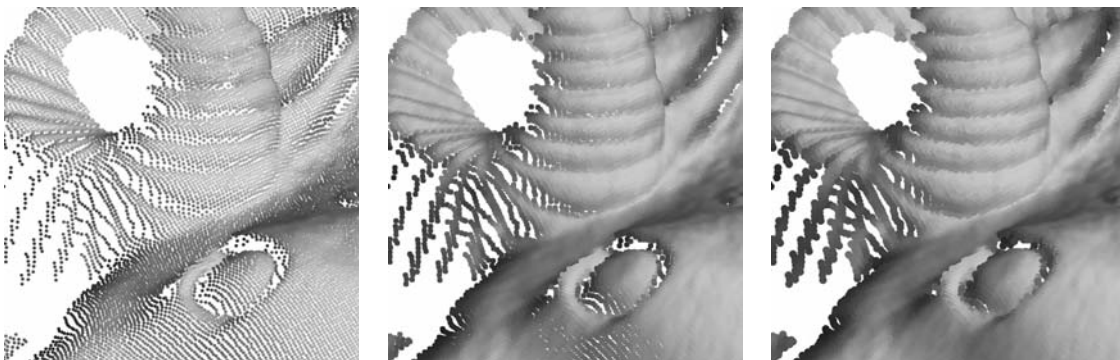
Visualizing Point Clouds: Point-based Rendering via Splatting



- Swept-plane scanner produces a *colored point cloud*: a set of 3D points
- Problem: how to render a point cloud to make it look like as a continuous surface?
- Splatting: render points as overlapping colored disks
- If normal vectors are measured as well, render points as shaded ellipses

*See the SIGGRAPH 2009 course: [Point Based Graphics – State of the Art and Recent Advances](#) by Markus Gross.

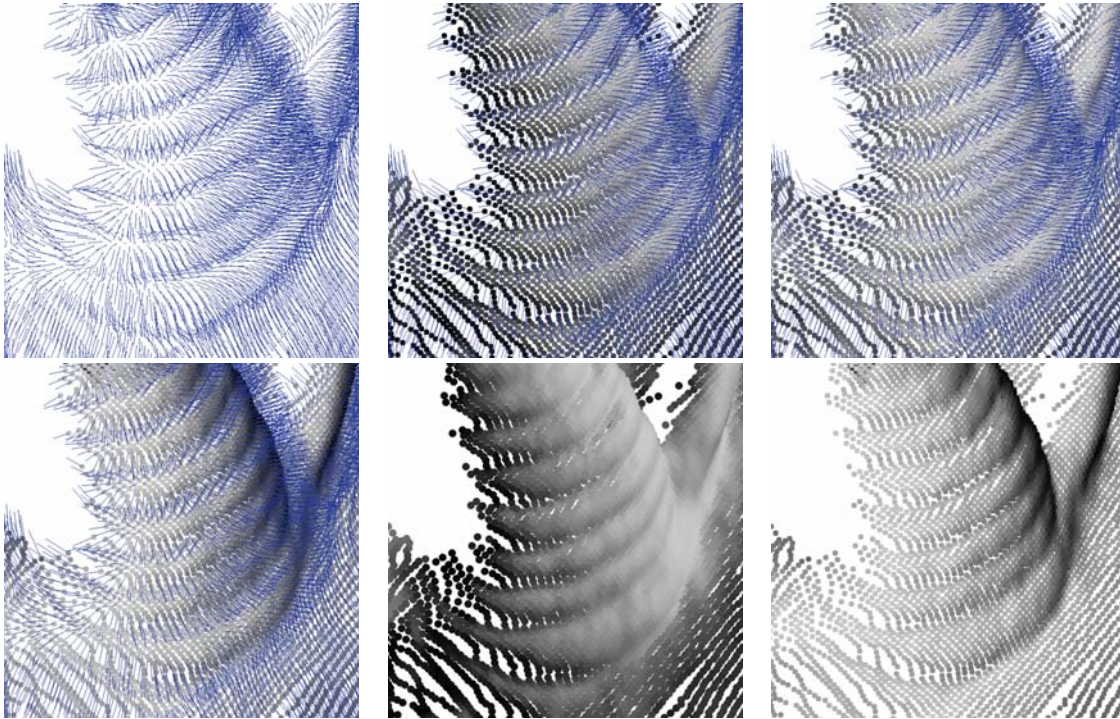
Visualizing Point Clouds: Point-based Rendering via Splatting



- Swept-plane scanner produces a *colored point cloud*: a set of 3D points
- Problem: how to render a point cloud to make it look like as a continuous surface?
- Splatting: render points as overlapping colored disks
- If normal vectors are measured as well, render points as shaded ellipses

*See the SIGGRAPH 2009 course: [Point Based Graphics – State of the Art and Recent Advances](#) by Markus Gross.

Visualizing Point Clouds: Splatting with normal vectors and colors



Visualizing Point Clouds: File Formats

- No standard file format to store point clouds
- Point = (x,y,z) plus (R,G,B) and/or (Nx,Ny,Nz)
- It is easy to create an ad-hoc file format
- Scene graph based file format: VRML
- International standard: ISO/IEC 14772-1:97 VRML'97
- PointSet node includes coordinates (x,y,z) and optional colors (R,G,B), but no normals

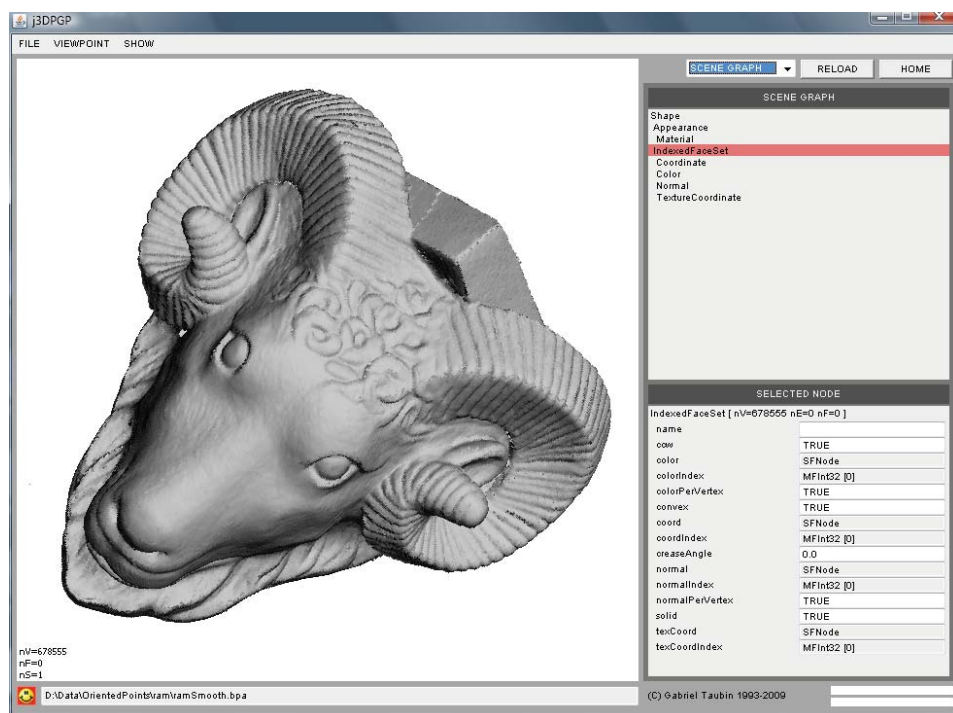
```
PointSet {  
  coord Coordinate {  
    point [  
      0 -1 2,  1 0 0,  
      -2 3 -1  
    ]  
  }  
  color Color {  
    color [  
      1 0 0, 0 1 0, 1 1 0  
    ]  
  }  
}
```

Visualizing Point Clouds: File Formats

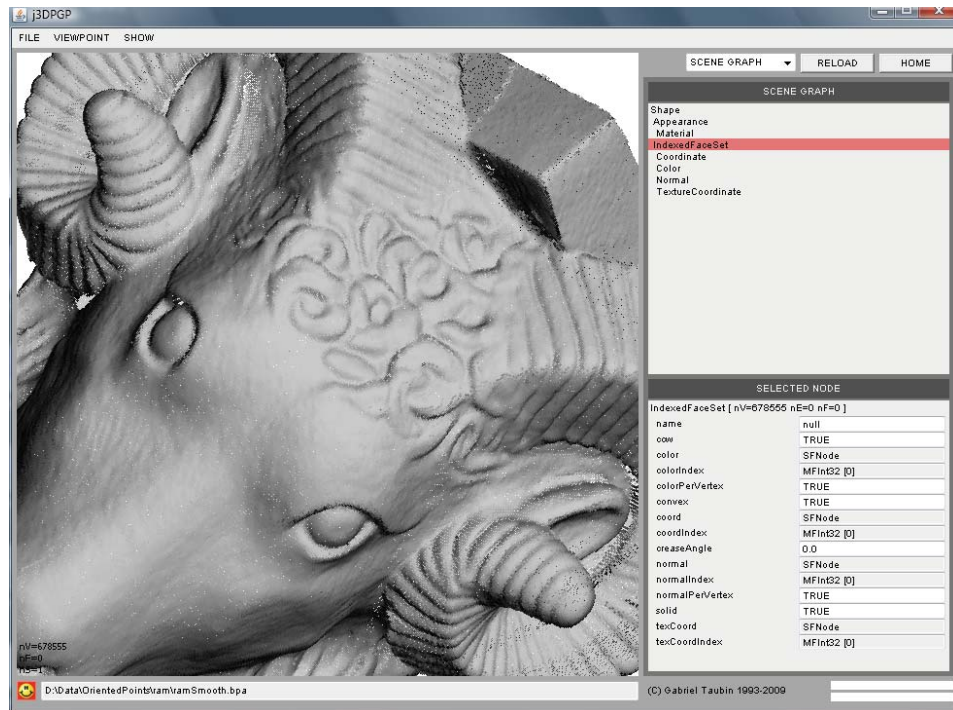
- IndexedFaceSet node designed to store a polygon mesh can be used to store point clouds with optional colors and/or normal vectors
- Store point coordinates as vertices
- Store point colors as colors per vertex
- Store point normal vectors as normals per vertex
- Degenerate polygon mesh with no faces is valid VRML syntax

```
IndexedFaceSet {
  coord Coordinate {
    point [
      0 -1 2,
      1 0 0,
      -2 3 -1
    ]
  }
  colorPerVertex TRUE
  color Color {
    color [
      1 0 0,
      0 1 0,
      1 1 0
    ]
  }
  normalPerVertex TRUE
  normal Normal {
    vector [
      1 0 0,
      0 1 0,
      0 0 1
    ]
  }
}
```

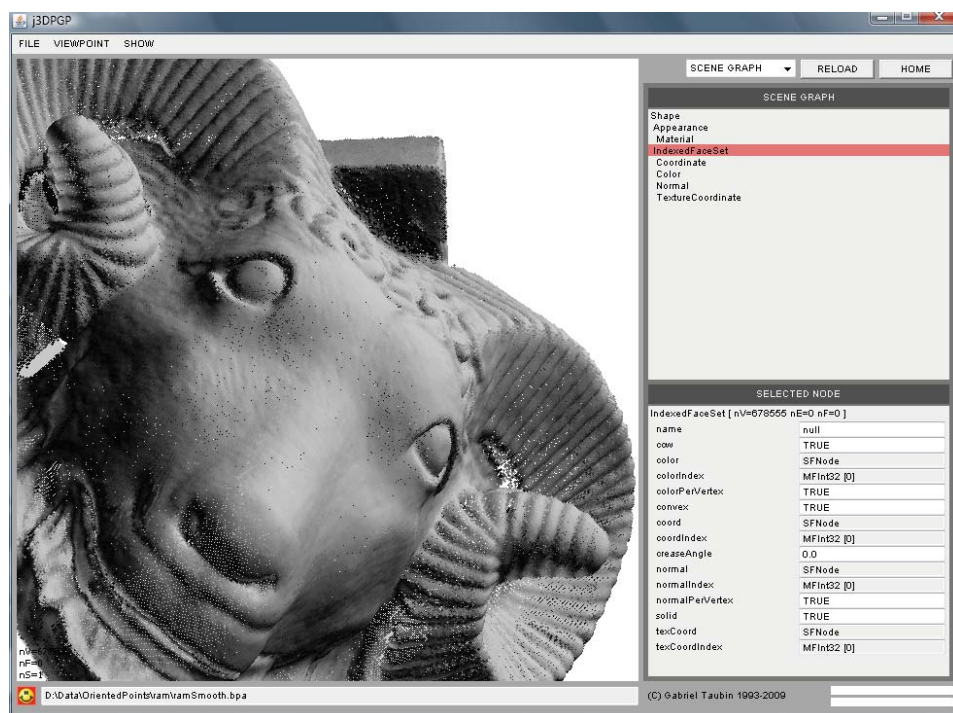
Visualizing Point Clouds: BYO3D Java Viewer



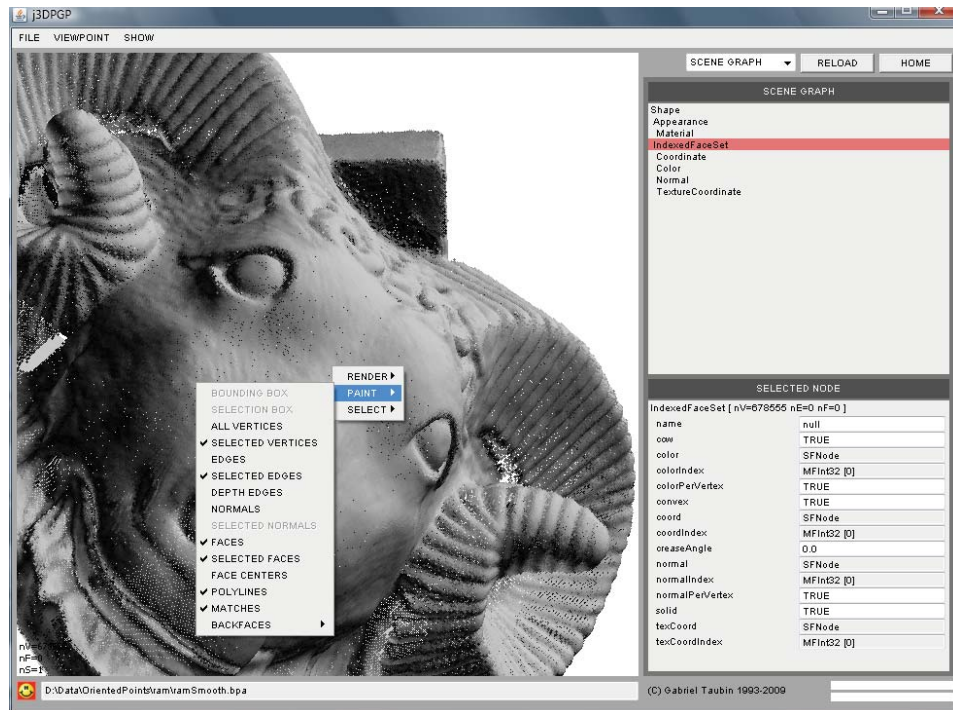
Visualizing Point Clouds: BYO3D Java Viewer



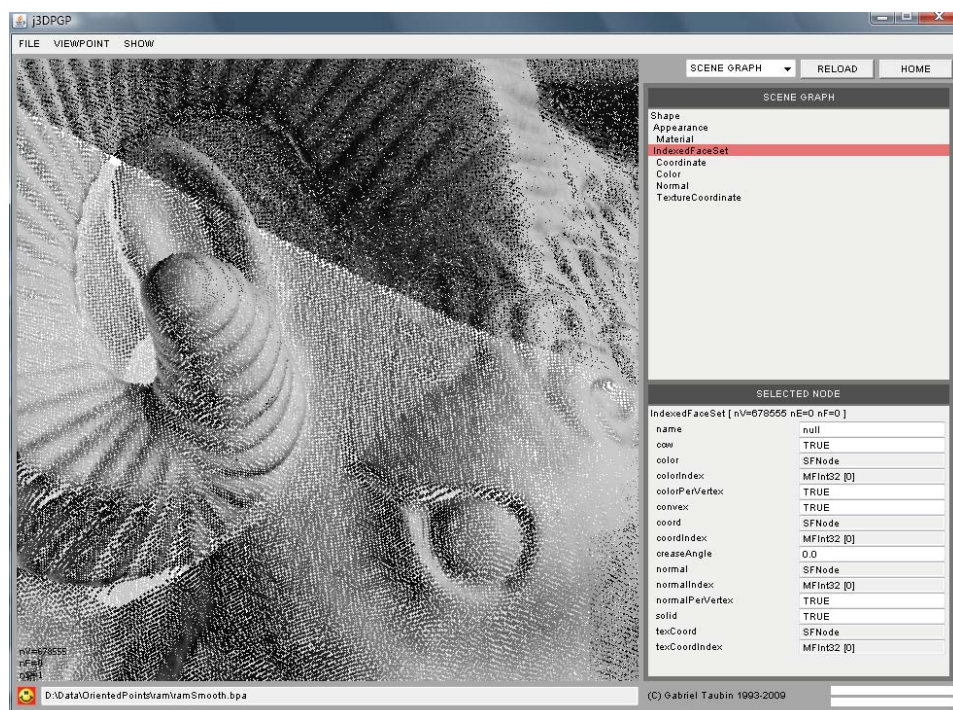
Visualizing Point Clouds: BYO3D Java Viewer



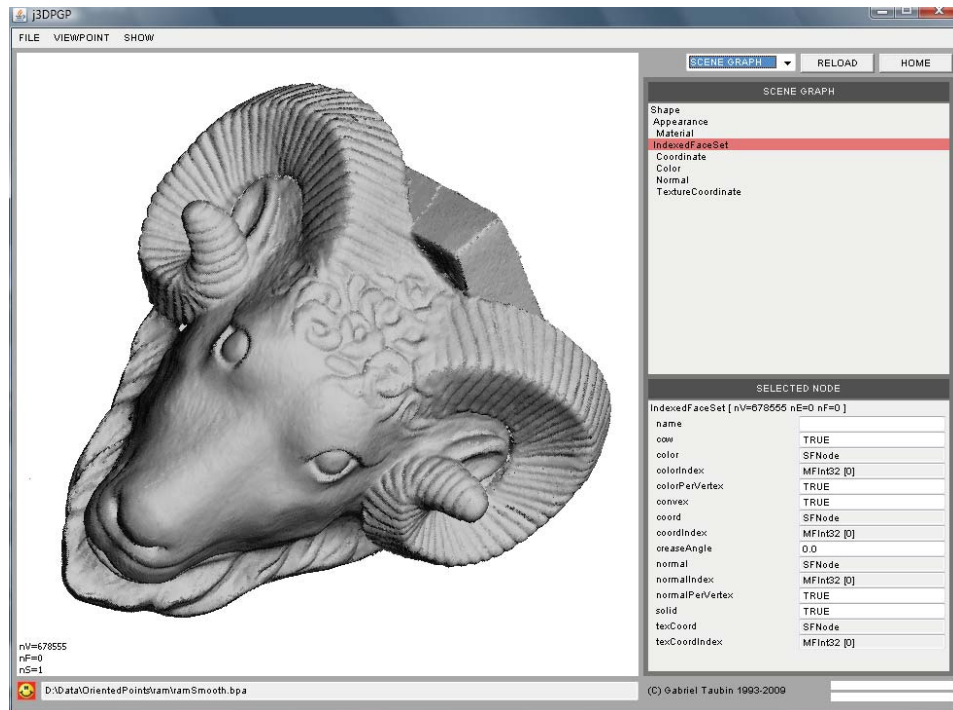
Visualizing Point Clouds: BYO3D Java Viewer



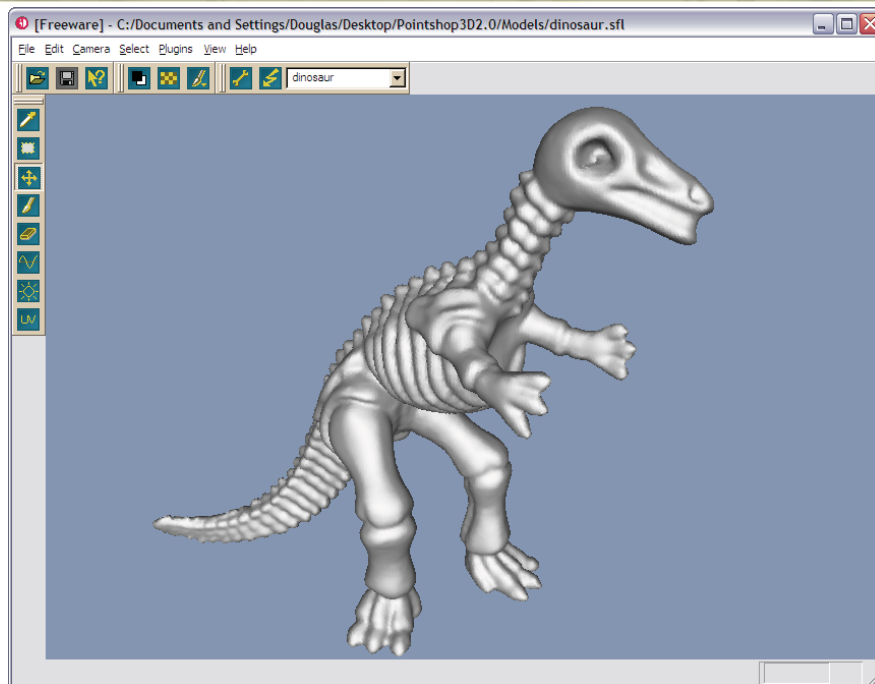
Visualizing Point Clouds: BYO3D Java Viewer



Visualizing Point Clouds: BYO3D Java Viewer

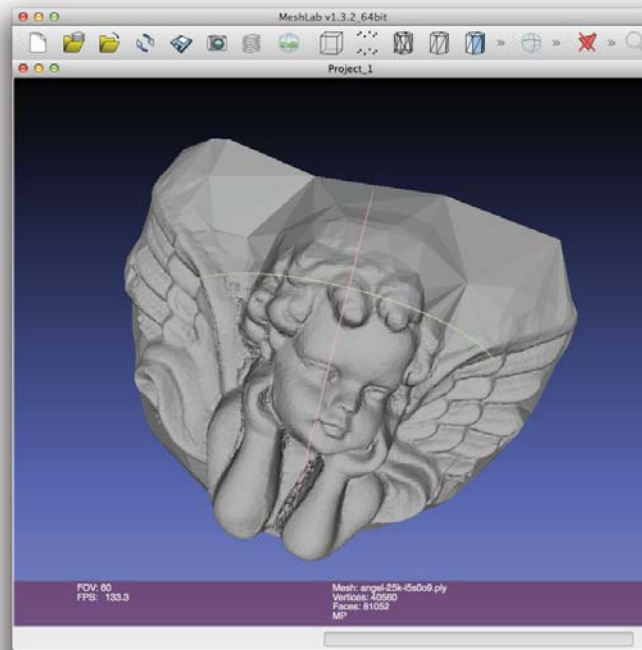


Visualizing Point Clouds: Pointshop 3D [Zwicker et al. 2002]



M. Zwicker, M. Pauly, O. Knoll, M. Gross. *Pointshop 3D: An Interactive System for Point-Based Surface Editing*. ACM SIGGRAPH, 2002

Visualizing Point Clouds: MeshLab

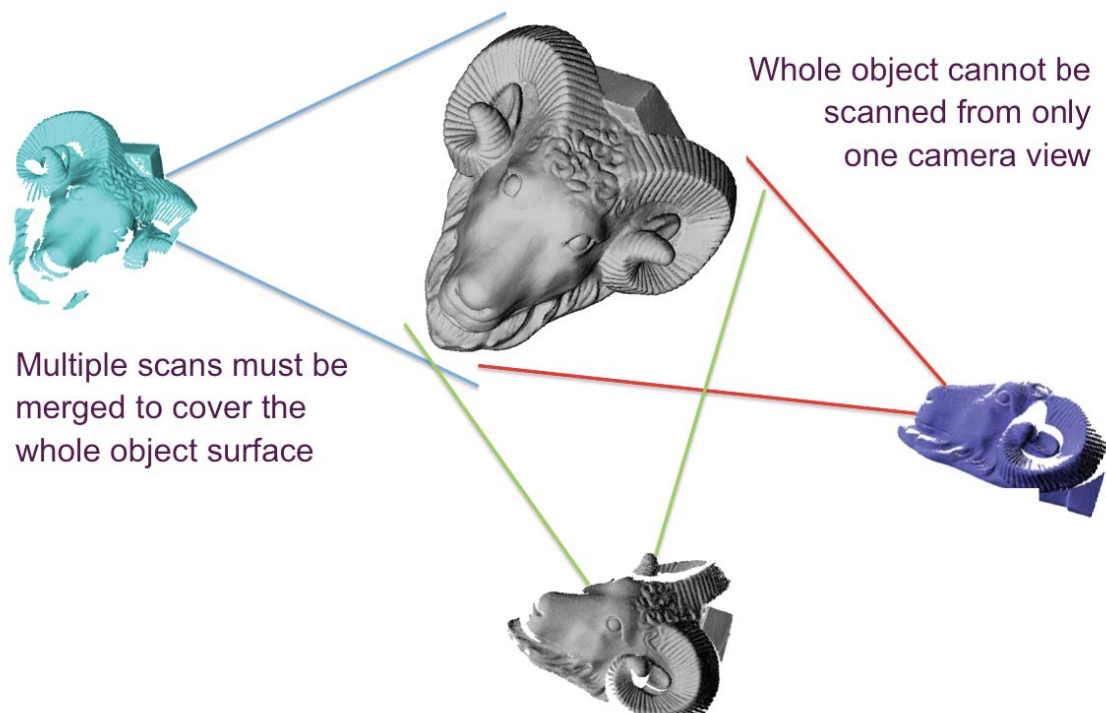


A screenshot of the MeshLab website homepage. The page features the "MeshLab" logo at the top left. Below the logo is a paragraph describing the software as an open source, portable, and extensible system for processing and editing of unstructured 3D triangular meshes. It mentions that the system is based on the VCG library developed at the Visual Computing Lab of ISTI - CNR. The page also includes a "Download Latest Version (03 August 2012) V1.3.2 (changes)" link. A section titled "Features" lists various capabilities such as interactive selection, input/output in many formats, mesh cleaning, remeshing, and various colorization/inspection filters. On the right side, there is a "SOURCEFORGE.NET" logo, a Facebook "Like" button, and a list of download links for different operating systems (Windows, Linux, MacOSX) and mobile devices (iOS, Android). The page also includes links to the MeshLab's Blog, Documentation, and Bug Reporting.

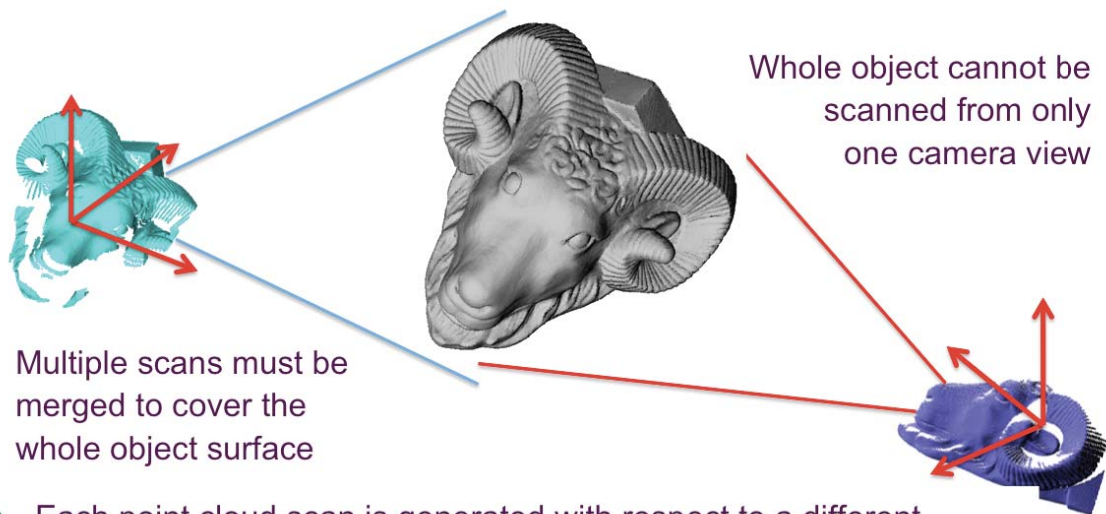
Course Schedule

- Introduction
- The Mathematics of 3D Triangulation
- 3D Scanning with Swept-Planes
- Camera and Swept-Plane Light Source Calibration
- Reconstruction and Visualization using Point Clouds
- **Combining Point Clouds Recovered from Multiple Views**

Merging Multiple Point Cloud Scans

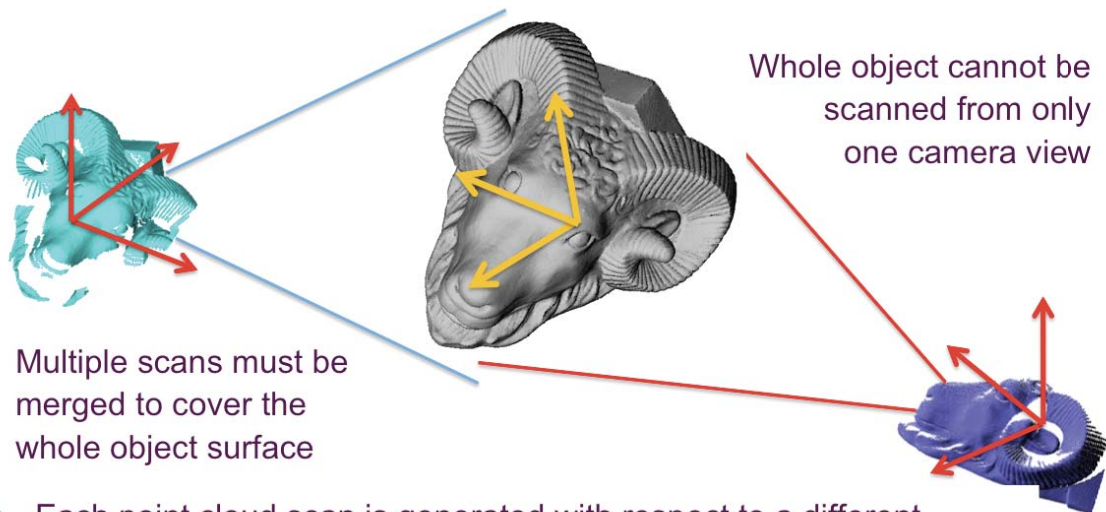


Merging Multiple Point Cloud Scans



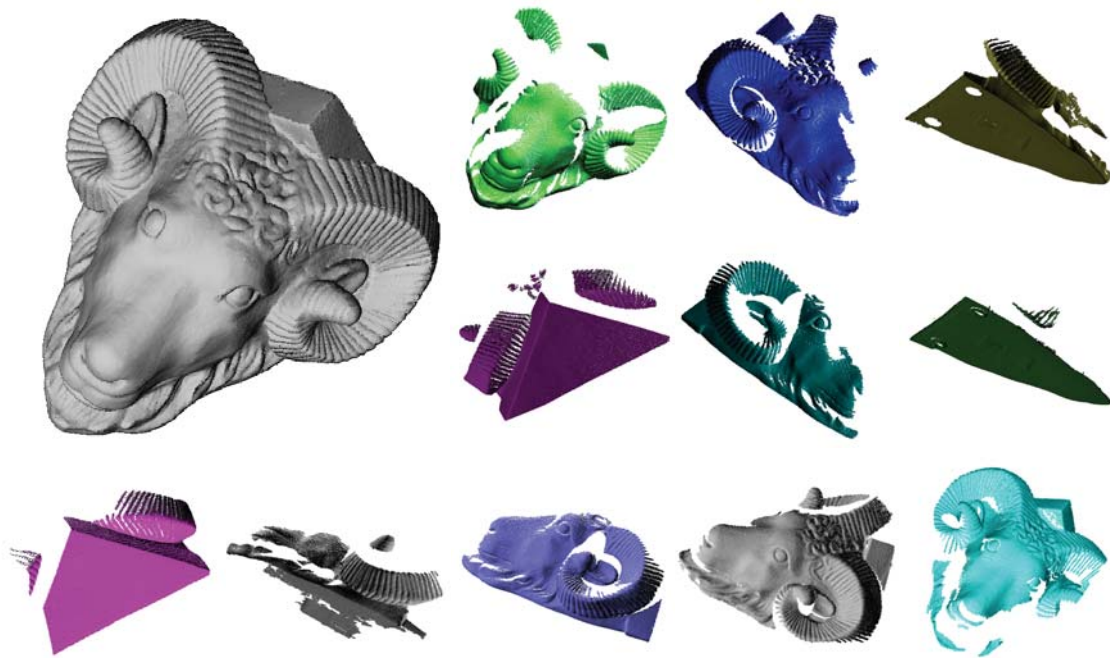
- Each point cloud scan is generated with respect to a different camera coordinate system

Merging Multiple Point Cloud Scans



- Each point cloud scan is generated with respect to a different camera coordinate system
- Relative position and orientation of each scan with respect to a global coordinate system must be determined to produce a single merged point cloud

Merging Point Cloud Scans



Complex Models May Require 100s of Scans



Shape

Appearance

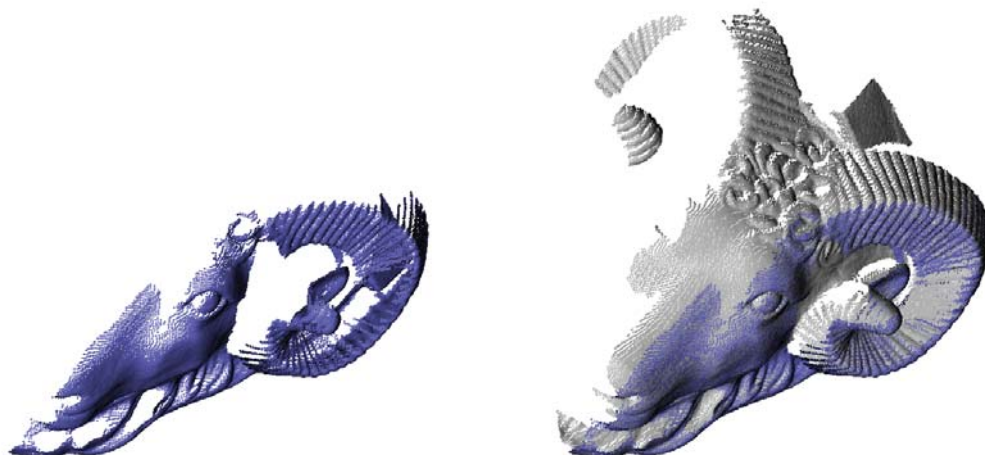


<http://www.research.ibm.com/pieta>



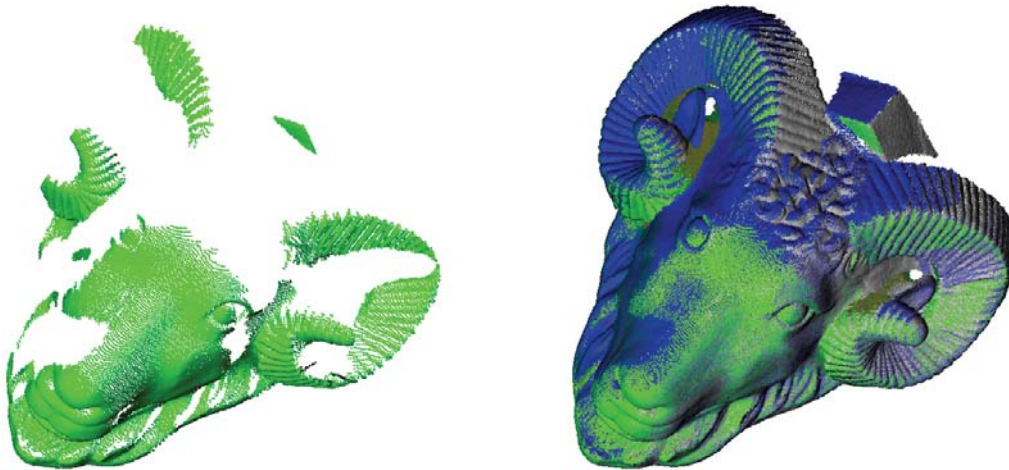
Merging Point Cloud Scans

- Incremental registration and merging
- Followed by global relaxation to remove accumulated errors



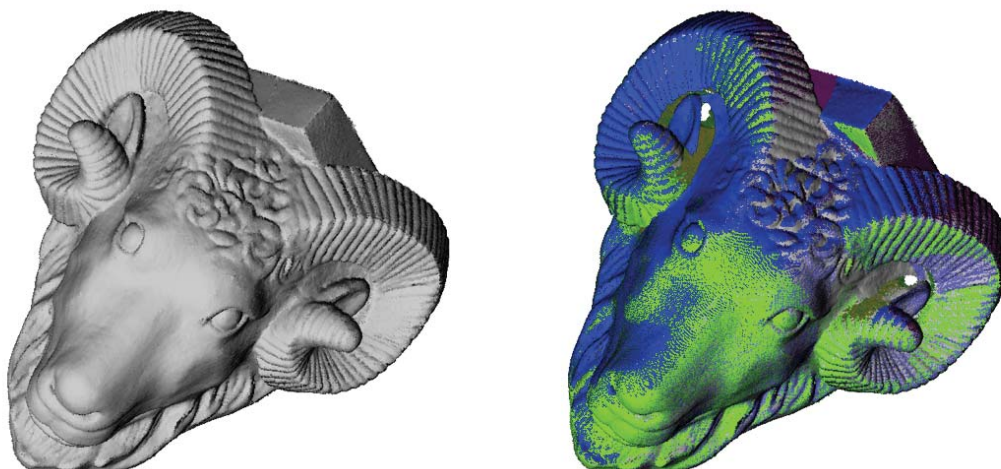
Merging Point Cloud Scans

- Incremental registration and merging
- Followed by global relaxation to remove accumulated errors



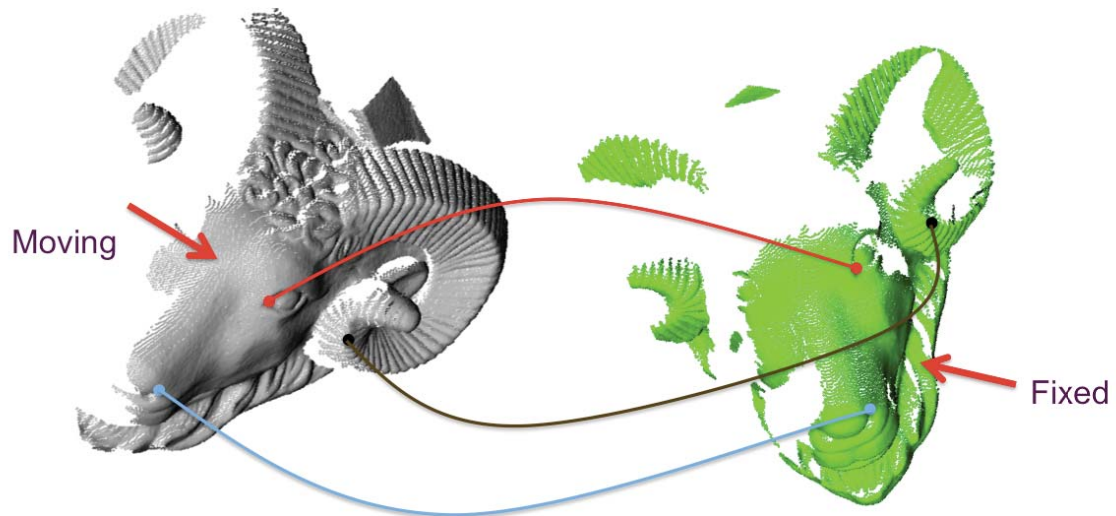
Merging Point Cloud Scans

- Incremental registration and merging
- Followed by global relaxation to remove accumulated errors



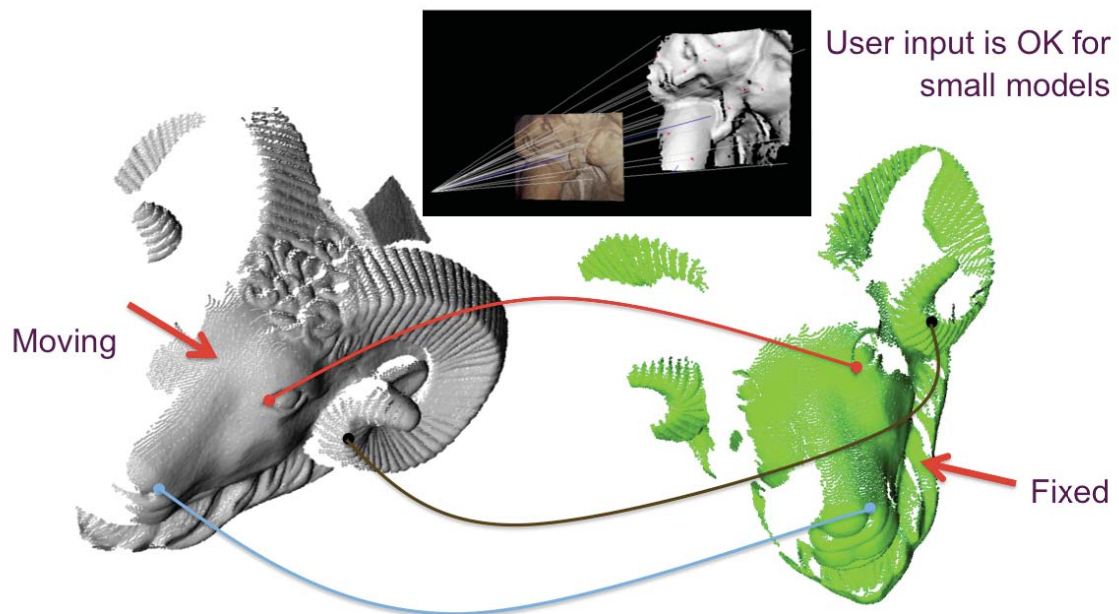
Merging Two Point Cloud Scans

- Select at least 3 pairs of corresponding points, but ideally $N \gg 3$



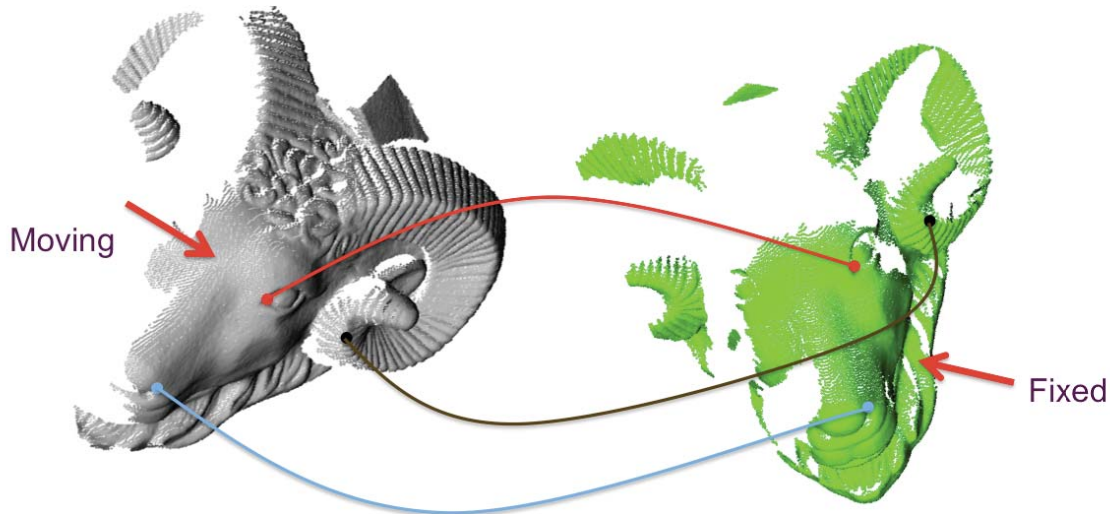
Merging Two Point Cloud Scans

- Select at least 3 pairs of corresponding points, but ideally $N \gg 3$



Merging Two Point Cloud Scans

- Select at least 3 pairs of corresponding points, but ideally $N \gg 3$
- Solve in close form for the matching rigid body transformation
- Refine solution using the Iterative Closest Point Algorithm (ICP)



P. Besl, N.D. McKey, A method for Registration of 3D Shapes.
IEEE Transactions on PAMI, 1992

Computing the Matching Transformation

- Given N pairs of corresponding 3D points $(p_1, q_1), \dots, (p_n, q_n)$ we are looking for a rotation matrix R and a translation vector T so that

$$Rp_j + T = q_j \quad j = 1, \dots, n$$

- In general, solution does not exist: solve in the Least-Squares sense
- Now we are looking for the minimizer of the quadratic energy function

$$E(R, T) = \frac{1}{n} \sum_{j=1}^n \|Rp_j + T - q_j\|^2$$

- This problem has a closed form solution

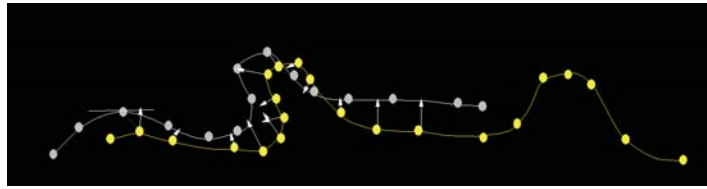
$$R = V^t U, \quad T = \bar{q} - R\bar{p}$$

- Where

$$\bar{p} = \frac{1}{n} \sum_{j=1}^n p_j \quad \bar{q} = \frac{1}{n} \sum_{j=1}^n q_j \quad M = \frac{1}{n} \sum_{j=1}^n (p_j - \bar{p})(q_j - \bar{q})^t$$

- And $M = U\Delta V^t$ is the Singular Value Decomposition (SVD) of M

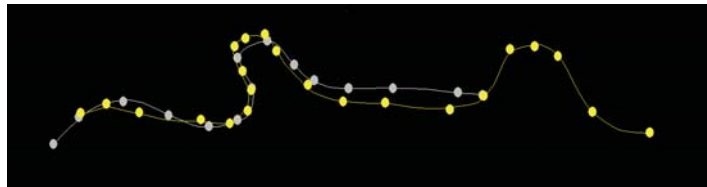
Iterative Closest Point Algorithm (ICP)



1. Automatically select N points p_1, \dots, p_n
2. Find closest corresponding points q_1, \dots, q_n
3. Solve in close form for the matching rigid body transformation which minimizes the energy function

$$E(R, T) = \frac{1}{n} \sum_{j=1}^n \|Rp_j + T - q_j\|^2$$

4. Repeat 1-3 while until convergence



Finding Closest Points

- Problem: find the point of the set $D = \{p_1, \dots, p_n\}$ closest to the point q
- Naïve algorithm: sequential search $O(N)$
- Too expensive if the same computation must be performed for many points q_1, \dots, q_n
- Efficient algorithm requires space partition data structure
Quadtree/Octree, BSP tree

