Filling Holes in Complex Surfaces using Volumetric Diffusion

James Davis Stephen R. Marschner Matt Garr Marc Levoy

Computer Science Department
Stanford University

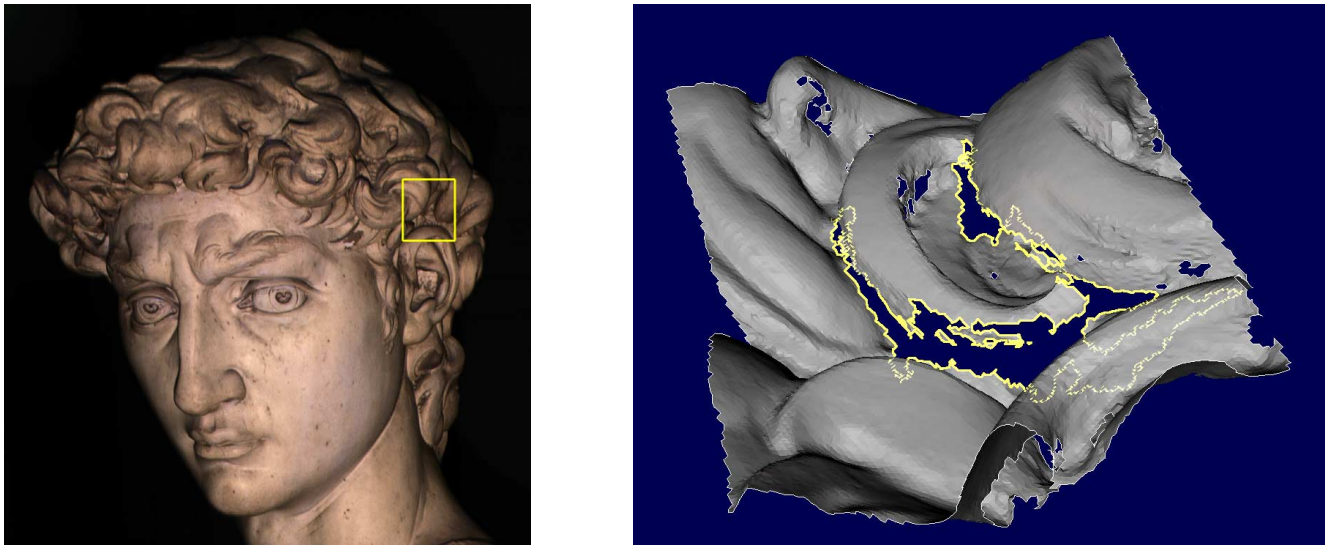


Figure 1: At left is a photograph of the head of Michelangelo's David. At right is a rendered 3D model of a section of his hair (corresponding to the square in the photograph). The section is 75 mm wide on the statue, and the spacing between triangle vertices is 1.0 mm. Although this surface was scanned dozens of times (from many different angles), occlusions prevented access to the deepest crevices. The hole highlighted in this rendering has a complex shape and multiple boundary components, i.e. islands floating in the hole. The largest boundary component contains 905 edges. (Hidden portions of this boundary are indicated with broken lines.)

Abstract

We address the problem of building watertight 3D models from surfaces that contain holes – for example, sets of range scans that observe most but not all of a surface. We specifically address situations in which the holes are too geometrically and topologically complex to fill using triangulation algorithms. Our solution begins by constructing a signed distance function, the zero set of which defines the surface. Initially, this function is defined only in the vicinity of observed surfaces. We then apply a diffusion process to extend this function through the volume until its zero set bridges whatever holes may be present. If additional information is available, such as known-empty regions of space inferred from the lines of sight to a 3D scanner, it can be incorporated into the diffusion process. Our algorithm is simple to implement, is guaranteed to produce manifold non-interpenetrating surfaces, and is efficient to run on large datasets because computation is limited to areas near holes.

1. Introduction

Modern rangefinding systems can measure the shape of an object's surface with high accuracy and resolution. However, these systems often cannot observe the entire surface, so the resulting 3D models may be incomplete. The most fundamental cause of holes is occlusion – recesses too deep to be observed using a particular triangulation angle. However, holes can also be caused by low reflectance, constraints on scanner placement, or simply missing views.

In some applications, an incomplete surface model is appropriate – it represents the surface exactly as measured, without adding fabricated geometry. However, other applications require a watertight surface that bounds a volume of space. Examples include computations of physical properties, fabrication of physical replicas, or presentation in contexts like schools and museums where holes would be confusing and unattractive. Algorithms

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

that are used in these applications often require that surfaces are valid 2-manifolds and/or that the geometry does not intersect itself.

To allow such uses while maintaining the data's accuracy and integrity, we need a surface reconstruction method that preserves the geometry where it exists and smoothly transitions to plausible geometry in unobserved areas. For scientific applications, it is also important to know which parts of the surface were observed and which parts are hypothetical.

One difficulty of hole filling is choosing appropriate topology. Many holes are simple and can be filled with disc topology; in these cases, triangulation algorithms can be employed. However, some holes have convoluted geometry, like a tangled loop of string (see figure 1). In our experience, such seemingly extreme cases occur frequently, especially when scanning objects that contain joints or crevices.

Even if the hole appears simple at a coarse scale, scanner noise may cause the polygons forming its boundary to point in essentially random directions (see figure 2). Once again, such extreme cases occur frequently, since scanner noise increases at the grazing observation angles that often accompany holes. When scanning marble statues using laser light, subsurface scattering of the laser beam contributes additional noise [Godin01]. Naive triangulation of such holes often yields self-intersecting geometry, as shown in the figure.

Other holes have multiple boundary components that should be filled, not with discs, but with patches that connect two or more loops of the boundary. This case occurs most frequently when the data is intermittent, due to low object reflectance, extreme specularity, grazing observation angles, or occlusions (see figure 7). A topologically inflexible approach may fail to find a valid manifold surface that passes through all the data.

To summarize, the ideal hole filling algorithm should:

- produce manifold, non-self-intersecting surfaces, even for noisy or convoluted data,
- choose appropriate (possibly non-disc) topology for holes that have multiple boundary components,
- construct plausible, visually pleasing geometry,
- distinguish in the output model between observed and fabricated surfaces,
- use all available information, including the scanner's lines of sight, and
- be efficient and scalable, since scanned models can contain upwards of a billion samples [Levoy00].

We describe a new technique for filling holes in scanned models by processing a volumetric representation – a signed distance function whose zero set is the observed surface. By applying diffusion to this volumetric representation, we extend the incomplete surface description until it forms a watertight (hole-free) model. In some cases the result does not match the topology of the object being scanned, but it is always topologically consistent (i.e. manifold), cannot self-intersect, and maintains fidelity to the original data wherever it exists.

2. Related work

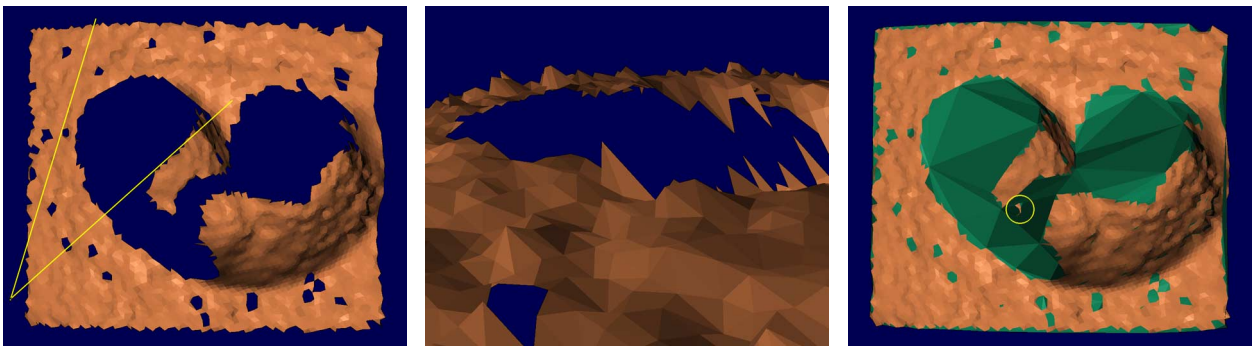
Hole filling can be performed as a post-processing operation, applied after surface reconstruction, or it can be integrated into a surface reconstruction algorithm. If integrated, we can further distinguish between reconstruction algorithms that operate on connected meshes of range samples and algorithms that operate on clouds of unorganized 3D points.

Hole filling as a post-process. For filling holes in an already reconstructed surface, one widely used approach is to triangulate each connected component of the surface's boundary, thereby filling each hole with a patch that has the topology of a disc. This technique works well for simple holes in nearly flat surfaces, but on convoluted holes it is likely to result in self-intersecting geometry, as mentioned earlier. For holes having multiple boundary components, many topologies are possible, hence many triangulations, and the problem becomes even more difficult.

Mesh-based reconstruction with hole filling. Mesh-based methods of surface reconstruction [Turk94, Curless96, Wheeler98] treat each scan (e.g. one sweep of a laser plane across the surface) as an ordered 2D array of range (i.e. depth) samples, sometimes called a range image. These samples can be triangulated to form a polygon mesh. These methods can perform hole filling as a post-process, or they can integrate it with surface reconstruction. The only mesh-based method we know of that integrates hole filling into surface reconstruction is Curless and Levoy's VRIP [Curless96]. This method converts each mesh to a signed distance function whose zero set is the observed surface, blends these distance functions together, and extracts the zero set as the final surface. To fill holes, they mark as empty the region of 3D space that lies along lines of sight between the scanner and the meshes, then they extract the boundary of this region as an additional surface.

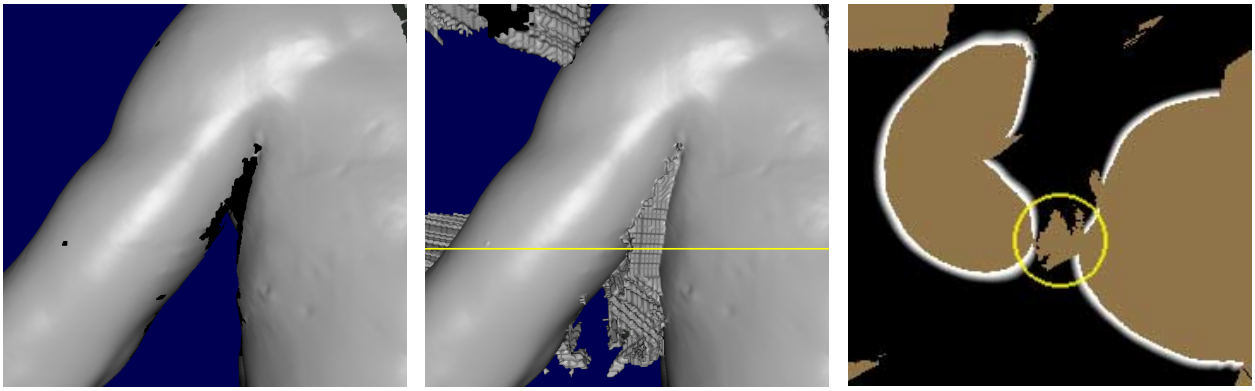
This so-called space carving method creates a surface that bounds the maximum region of space consistent with the scans, so it is guaranteed to produce a watertight surface. However, the method may lead to surfaces that are less plausible than smoothly extending the observed surfaces. Moreover, the method requires knowledge of

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE



(a) (b) (c)

Figure 2: Even holes having a single boundary component may be difficult to fill using triangulation. (a) A rendering of the range image resulting from one sweep of a laser triangulation scanner across the pupil of David's right eye. The section shown here is 25 mm wide on the statue, and the spacing between range samples is 0.29 mm. The heart-shaped pupil forms a "canyon" into which two strips of geometry extend. However, they do not completely cover its walls and floor, so a hole remains. The boundary of this hole contains 311 edges. (b) A view across the canyon from the left rim. This view corresponds to the frustum shown in (a). Scanner noise makes the hole ragged and gives its bounding polygons widely varying orientations. (c) Triangulation of such a hole almost invariably produces self-intersecting geometry; one such intersection is circled.



(a) (b) (c)

Figure 3: Space carving using line-of-sight constraints sometimes produces unwanted geometry. (a) A rendering of a coarse model (2.0 mm) of David's back and left arm. The model was created using VRIP [Curless96], but with space carving disabled. Despite multiple scans, a large hole remains in his armpit. (b) If space carving is enabled, some holes are filled, but his armpit is now bridged. The cause can be seen by examining a slice (c) through the VRIP volume at the plane denoted with a horizontal line in (b). In this visualization, the signed distance function is shown as a band of white shading off to black, with mid-gray representing the observed surface. Solid black areas are known to be empty due to line-of-sight constraints. Brown areas have not been seen and are therefore unknown. A large unknown area bridges David's torso and arm (circled in the image). The boundary of this area is the bridge surface seen in (b). The aliasing in this surface is due to a flaw in the current implementation of space carving; it is not fundamental to the algorithm.

scanner lines of sight, and it performs poorly if these lines of sight do not adequately cover the volume outside the object (see figure 3). Additional lines of sight can be obtained by scanning backdrops placed behind the object but still within the scanner's working volume [Curless97], or by using a separate sensor to detect object silhouettes [Wojciech00]. However, these solutions may be difficult to deploy outside the laboratory. Our algorithm can take advantage of line-of-sight information if it is available, but it can also operate without it.

Point-cloud reconstruction with hole filling. The third branch of this taxonomy is point cloud methods [Amenta98, Bajaj95, Bernardini99, Edelsbrunner92, Hoppe92, Whitaker98], which treat the union of all the scans as an unorganized set of 3D points to be fit with a

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

continuous surface. With no connectivity between range samples, the large gap across a hole is conceptually equivalent to the space between adjacent samples, so these methods effectively fill holes during reconstruction.

One class of point-cloud methods interpolates the original samples using alpha shapes [Edelsbrunner92, Bajaj95], crusts [Amenta98, Dey01], or balls [Bernardini99]. However, interpolation may not be appropriate for noisy data, and these algorithms may fail if sample noise approaches sample density – which it often does. Also, in algorithms based on alpha shapes or balls, it may be difficult to find a single alpha (or ball radius) that bridges holes without also bridging fine surface details. The authors in [Bernardini99] address the noise problem by smoothing the data beforehand, and they address the bridging problem by running the algorithm several times with increasing ball radius.

A second class of point-cloud methods evolves a surface over time until it approximates the data. Examples include inflating a polygonal mesh [Chen95] or evolving a 3D signed distance function, a level set of which is the intended surface [Whitaker98, Zhao01]. A third class of point-cloud methods fits a set of 3D radial basis functions to the data; a weighted sum of these functions forms a new function, a level set of which is the intended surface [Dinh01, Carr01].

Although the latter two classes resemble our method in their use of a level set to represent the surface, there are important differences. First, they fill holes during reconstruction, so they must be applied to the entire surface, even though holes typically cover only a small fraction of the total area – perhaps only a few percent. Our diffusion process, on the other hand, operates only near holes. Second, since our algorithm operates after surface reconstruction is complete, it is compatible with any reconstruction method. VRIP is one such reconstruction method, which is known to be fast; the algorithm in [Bernardini99] is also fast[1]. Finally, our algorithm provides a mechanism for including additional constraints, such as regions of space known to be empty. Although it might be possible to extend the former methods to include these additional constraints, we know of no such extension.

[1] Working from results reported in their papers, we estimate the following surface reconstruction (and hole filling) rates, measured in input range points reconstructed per hour per gigahertz: [Carr01] = 250K, [Zhao01] = 350K, [Curless96] with space carving = 13M, [Curless96] without space carving + this paper for hole filling = 5M. Direct comparisons between surface reconstruction algorithms are difficult, and reconstruction quality has not been controlled, so these numbers should only be taken as approximations.

Other related work. Although diffusion methods have not previously been applied to surface reconstruction, they have a long history in the image processing community. An application of diffusion to filling gaps in sampled data is image inpainting [Bertalmio00]. Like our method, Bertalimio et al. iteratively apply a sequence of operators, one of which is anisotropic diffusion [Perona90], in order to propagate information from known regions of the image into unknown (e.g. scratched) regions. Although these methods have been extended to vector-valued images and parametric domains on 2-manifolds [Sapiro01], it is not obvious how to extend them to filling holes in surfaces.

3. Volumetric diffusion

Our hole filling algorithm can be used on any 3D surface model. The surface is first converted to our volumetric representation, which is a regularly spaced 3D grid of values of a clamped signed distance function $d_s(\mathbf{x})$. This function is defined only in a narrow band near the observed surface, and it is positive inside the surface and negative outside. Since the units of the function are not important, we define it to be in the range from -1 to 1 over the width of the band. The thickness of the band does not have a significant influence on the result; we use about 5 voxels on either side of the observed surface. The observed surface is the zero set of d_s[2].

One can construct d_s in many ways; our implementation uses the VRIP algorithm [Curless96] to build this function directly from a collection of range scans. One could alternatively build d_s using volumetric scan conversion from a reconstructed surface [Frisken00], or one could identify points known to be outside (or inside) the object, then use these to build the sidedness function [Carr01]. At the same time we define an associated weight function w_s, which ranges from 0 to 1 and measures our confidence in the value of d_s. In most areas $w_s = 1$, but it typically decreases near boundaries of the observed surface, where noise increases.

The goal of our algorithm is to extend d_s to a function d that is defined over the entire volume, though in practice we only compute d near the surface – in fact only near holes in the surface. We achieve this by diffusing the values of d_s outward from the observed surface into adjoining undefined areas. As the function spreads, so does its zero set. In particular, the diffused function propagates inward across the holes, eventually spanning them. Once

[2] An alternative definition, equivalent in practice, is a filtered sidedness function: –1 outside the object's surface and +1 inside.

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

diffusion is complete, the zero set of this function is the desired hole-free surface.

3.1. Basic algorithm

The diffusion process consists of alternating steps of blurring and compositing. We begin with $d = d_s$, and each iteration first convolves d with a lowpass filter h, then composites d_s back into the volume using the *over* operator [Porter84]. The algorithm uses two volumes, the diffusion volume and the source volume. The diffusion volume, which is where the computation takes place, has two values at each point, $d_i(\mathbf{x}) \in [-1, 1]$, the value of d after i iterations, and $v_i(\mathbf{x}) \in \{0, 1\}$, which indicates where the value of d_i is valid. The source volume represents the observed surface, and contains two values, $d_s(\mathbf{x}) \in [-1, 1]$ and $w_s(\mathbf{x}) \in [0, 1]$. The initialization is:

$$(d_0, v_0) \;=\; (d_s, [w_s > 0]),$$

where $[p]$ evaluates the predicate p and returns 1 if p is true and 0 otherwise. A single iteration is:

$$(\hat{d}_i, v_i) \;=\; h * (d_{i-1}, v_{i-1})$$

$$d_i \;=\; w_s d_s + (1 - w_s)\hat{d}_i.$$

During the convolution only valid voxels ($v = 1$) are used; h is renormalized to include only these voxels. For the Boolean volume v, convolution with h can be interpreted as inflating the region where $v = 1$ by the support of h. Due to the repeated blurring, the choice of h is not critical; we use a $3 \times 3 \times 3$ box or 7-point ''plus'' filter, because they are faster to evaluate than larger filters. The output surface is constructed by running Marching Cubes [Lorensen87] (with corrections from [Montani94]) once, after diffusion is complete, to extract the $d = 0$ isosurface. To insure a closed mesh, voxels outside the volume boundaries are treated as if they had the value $d = +\infty$. Note that this surface does not in general interpolate the original range samples [Curless96]. Figure 4 shows the stages of diffusion in 2D.

Through a sequence of algebraic manipulations, it is possible to show that the algorithm we have just described is identical to the heat equation [Landau87]. In heat diffusion, a scalar field representing temperature is propagated from each node in a computational domain to its neighbors according to the material's thermal conductivity. Optionally, after each diffusion step, a source term is added into each node, representing the addition (or removal) of heat to (or from) the system. These propagation and addition steps correspond precisely to our convolution and compositing steps.

The equivalence between our algorithm and the heat diffusion equation has two important implications. First, it tells us that our process is guaranteed to converge. Second, it says that once the high-frequency components of the heat distribution have been damped out – which happens quickly – the equation converges rather slowly to equilibrium. In the context of our problem, once the holes are closed and changes in surface shape on each iteration fall below the noise inherent in our scanner (and hence in our input geometry), we stop.

3.2. Accelerations

This volumetric diffusion algorithm, implemented naively, would consume time and memory proportional to n^3, the number of voxels in the volume. Because most of the volume is empty, and only a small fraction of the surface contains holes, this is inefficient for large models. We take two measures to accelerate the computation: we use a sparse representation of the volume that avoids using memory for undefined areas, and we limit the computation to voxels that are not more than a certain predetermined distance from a hole boundary in the original surface.

To implement the first acceleration, we represent both the source volume and the working volume using a simple block structure. The volume is divided into fixed-sized cubical blocks (we use $8 \times 8 \times 8$ voxels), and storage is only allocated for blocks containing valid voxels. To implement the second acceleration, we simply flag the voxels that are within m voxels from a hole boundary [3] and process only those voxels during diffusion. The choice of m depends on the size of the largest hole to be filled; m must be greater than half the width of that hole. Typical values of m for the examples shown in this paper are 15 to 30 voxels. If the algorithm fails to close a hole because m was too small, it suffices to increase m and continue diffusing.

With these two accelerations, the algorithm requires space proportional to surface area times block width, and time that depends on m and the area and size of holes. If we let k be the fraction of the surface area that is within distance m of a hole boundary, the processing time for a diffusion iteration is proportional to kn^2m. The value of k reflects both the size and the shape of the holes, but is typically small (a few percent). For example, given an object embedded in a 1000^3 volume, where holes cover 5% of the surface area and the largest hole is 50 voxels across (in

[3] A voxel is on a hole boundary if it is valid, has at least one invalid neighbor (in v_0), and has at least one valid neighbor with the opposite sign (in d_0).

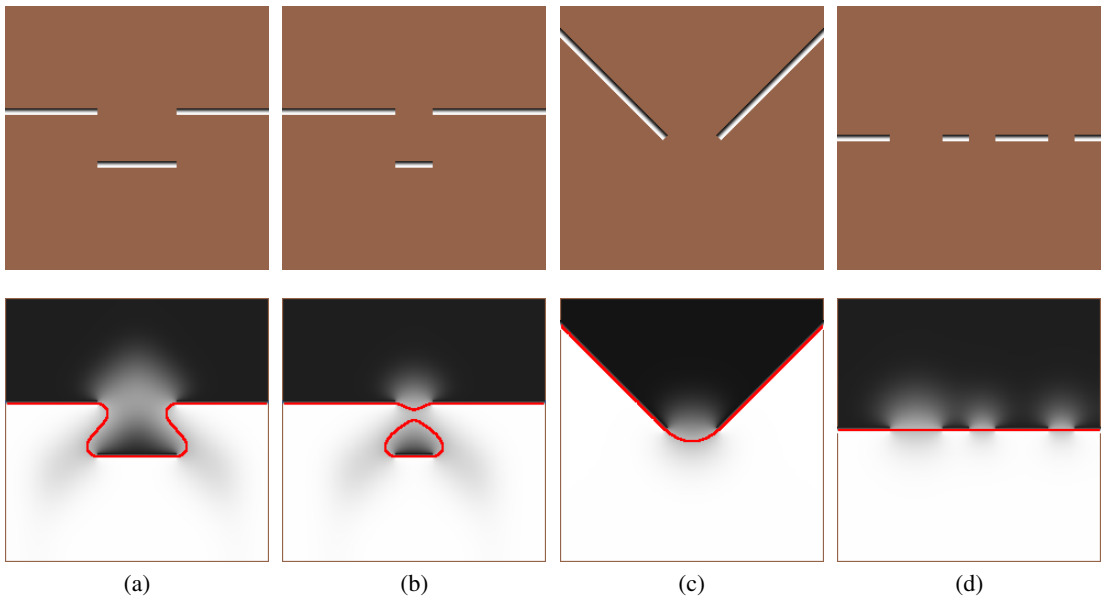
Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

(a) (b) (c) (d) (e)

Figure 4: Illustration of 2D diffusion in progress. (a) The source term. Grayscale values encode signed distance, with black and white corresponding to outside ($d = -1$) and inside ($d = 1$), respectively. Brown denotes invalid voxels ($v = 0$). (b) Our diffusion process extends the surfaces. The red curve marks the zero set. (c) The surfaces begin to interact. (d) The hole closes. (e) The shape is converged.

(a) (b) (c) (d) (e)

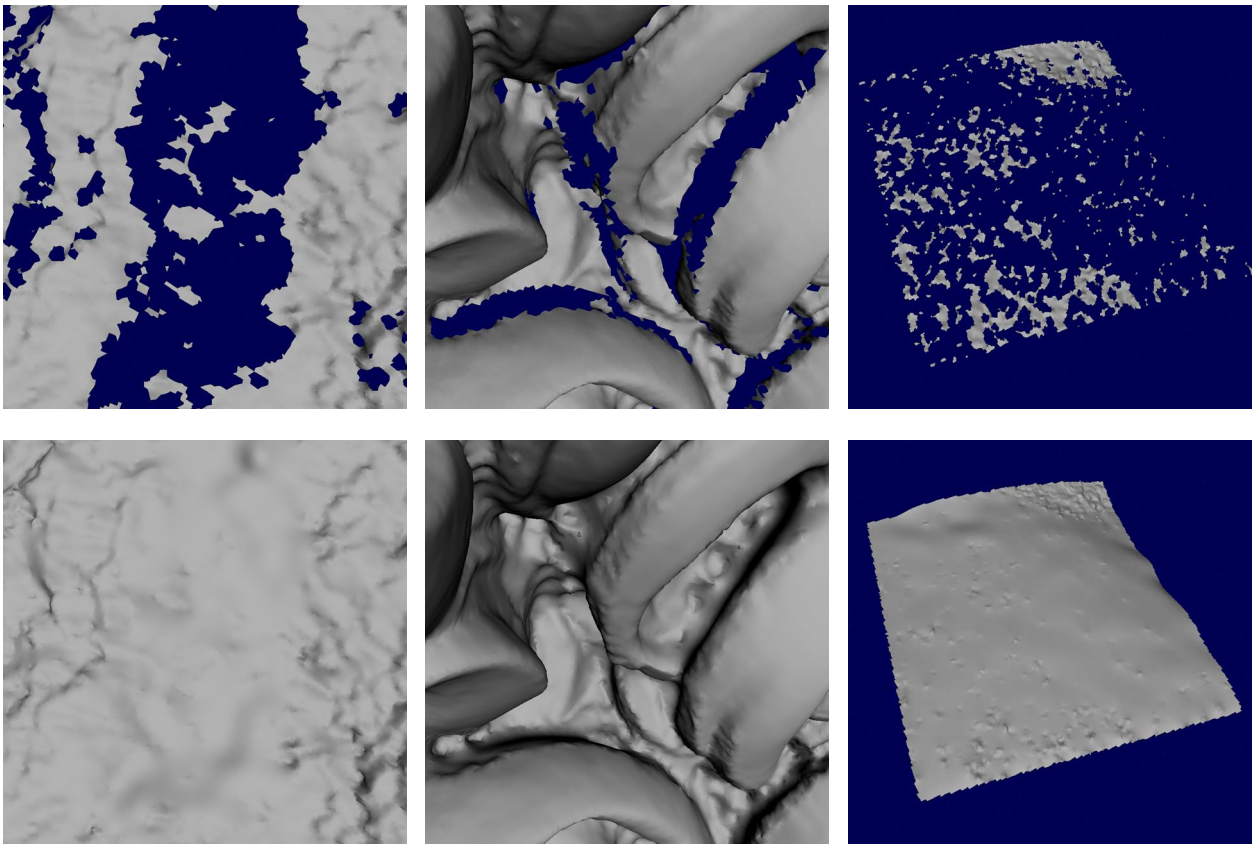
Figure 5: The same example as figure 4, but with line-of-sight constraints. Incorporating this additional information into the diffusion process causes the converged surface to stay mostly out of the region known to be empty, while remaining smooth. (a) Blue pixels mark where the additional source term indicates that space is empty ($w_c > 0$, visible here only when $w_s = 0$).



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

Figure 6: Examples of diffusion in 2D for several kinds of holes. Above: source term; below: diffusion result with zero set marked in red. (a) A frontal (i.e. perpendicular) scan of a surface with a recessed portion leads to two holes, which our diffusion algorithm fills with curved steps. (b) If the recessed portion is sufficiently narrow, the diffusion algorithm instead builds a bridge across it. This bridging can be prevented, when appropriate, using line-of-sight constraints. (c) If two angled surfaces are scanned frontally (relative to each surface), the joint between them may be missed. Our algorithm fills this hole with a smooth fillet. (d) A flat surface with several holes, which are filled with the expected geometry.

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE



(a) (b) (c)

Figure 7: Results of our algorithm on holes that occur in practice. (a) A section 40mm wide of a fragment of the Forma Urbis Romae, a marble map of ancient Rome. The surface is relatively flat, but the hole has many islands, which would be thrown away by algorithms that triangulate its boundary. (b) Another view of the section of David's hair shown in figure 1. Our diffusion process creates plausible surfaces to fill these holes. (The bumps in the crevices are drill holes made by Michelangelo; they are not artifacts of our hole filling process.) (c) A section 50 mm wide from Michelangelo's highly polished statue of Night. The statue's specularity and the grazing scanning angle in this example created an area that is mostly hole, rather than mostly surface. Our diffusion process successfully fits a surface through what little data does exist.

its narrowest direction), 2.5 million voxels must be touched per iteration. This represents only 0.25 percent of the volume. If the diffusion process is allowed 100 cycles to converge – twice the width of the largest hole – then the total number of voxel read/writes is 250 million, still only a fraction of the 1 billion voxels required if we stored the entire volume. For more examples, see section 4.

3.3. Line-of-sight constraints

Range scanners provide information about where surfaces are not as well as where they are, based on lines of sight between the range scanner and the observed surface. In triangulation scanners, each range sample defines two lines of sight; from the surface to the laser, and from the surface to the camera. In order to observe a range sample, both lines must be empty.

We do not require lines of sight in our basic algorithm. (We do require knowing inside from outside, which we derive from lines of sight, but this same information could be derived from surface normals.) When line-of-sight information is available, it may be incorporated into the diffusion process as a second source term, (d_c, w_c). For volumes of space that are known to be empty, we set $d_c = -1$, indicating that those areas are outside the surface, and we set $w_c = \alpha$. At the boundaries between empty and unseen volumes of space, w_c ramps to zero over a length dependent on the precision of the scanner. The parameter α can be thought of as our confidence that these voxels are indeed empty. In practice, it controls how far into the empty region surfaces will be built; for high α the surface will turn sharply to avoid empty regions, and for lower values it will remain smoother at the expense of

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

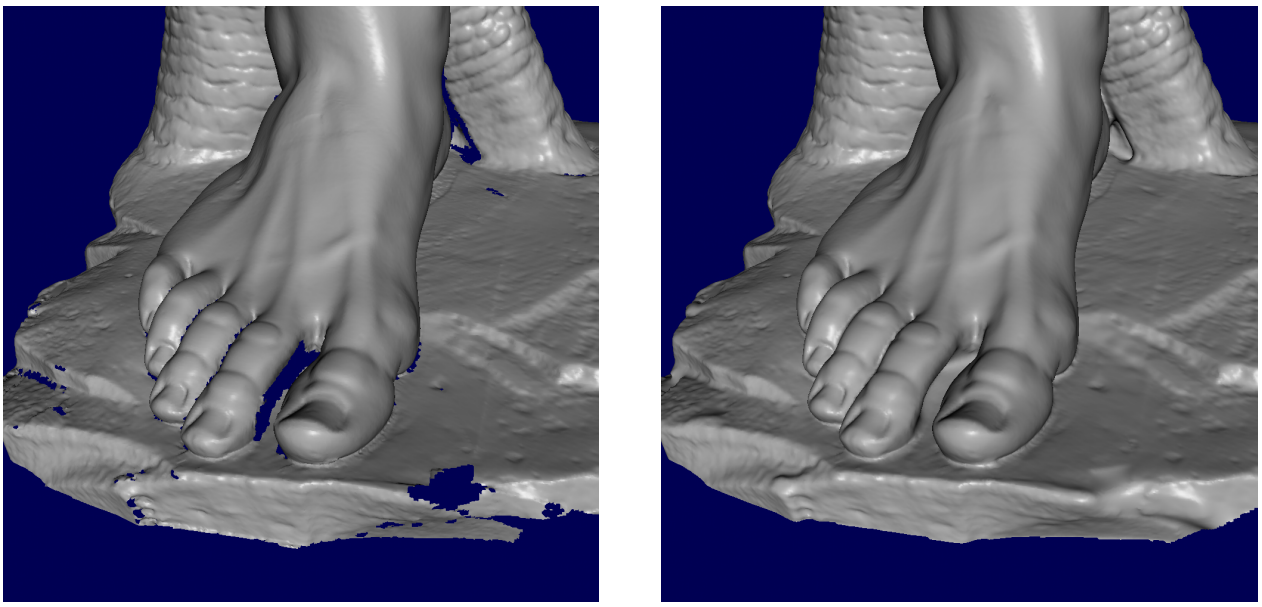


Figure 8: Volumetric diffusion applied to a larger dataset, the foot of Michelangelo's David with 1.0 mm spacing. The boundary of the largest hole in this dataset (between the big and second toes) contains 966 edges. See table 1 for performance statistics.

protruding slightly into the empty region. Typical values for α range from 0.001 to 0.01.

When using line-of-sight constraints in the diffusion, we still initialize d_0 to d_s, but during each iteration we composite with (d_c, w_c), then (d_s, w_s). This has the effect of applying a constant, gentle pressure toward -1 (outside) as the function diffuses into the known-empty region of space. An example in 2D is shown in figure 5.

4. Results

To illustrate the behavior of our algorithm, we show results in figure 6 for several synthetic 2D test cases. These images show that the algorithm generates plausible, smooth surfaces for a variety of configurations, and that it can generate different topologies. Determining whether the topology it produces matches that of the object being scanned requires either additional geometry or a high-level understanding of the object's shape, which our algorithm does not have.

To test our method on holes that arise in practice, we have adapted our algorithm to use the clamped distance function generated by VRIP for d_s, with w_s derived from VRIP's confidence values. These confidences account for several sources of uncertainty, most notably surface orientation relative to the scanner's line of sight and distance to the nearest edge of the observed surface.

size of input volume	440 M voxels
fraction of voxels touched	4.5%
size of output mesh	4.5 M triangles
total memory allocated	550 MB
processing time including I/O	20 min

Table 1: Some statistics for our hole fill of David's foot (figure 8). Processing time is on a 1 GHz Pentium III PC. The input volume comes from VRIP [Curless96], but the processing time does not include running VRIP.

Figure 7 shows results for three scanned models from the Digital Michelangelo Project [Levoy00]. These examples illustrate the algorithm's ability to fill a variety of holes using the same computation. To demonstrate the scalability of the system, Figure 8 shows results for the entire foot of the David with 1.0 mm voxel spacing. Performance statistics for this example are given in table 1. To run our algorithm on the entire David at full resolution (0.25 mm voxels) would take several days and require tens of gigabytes of memory. For such a large model, we could partition the volume into overlapping blocks and process each block separately. The required amount of overlap between the blocks would depend on the width of the largest hole to be filled. We have not yet implemented this.

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

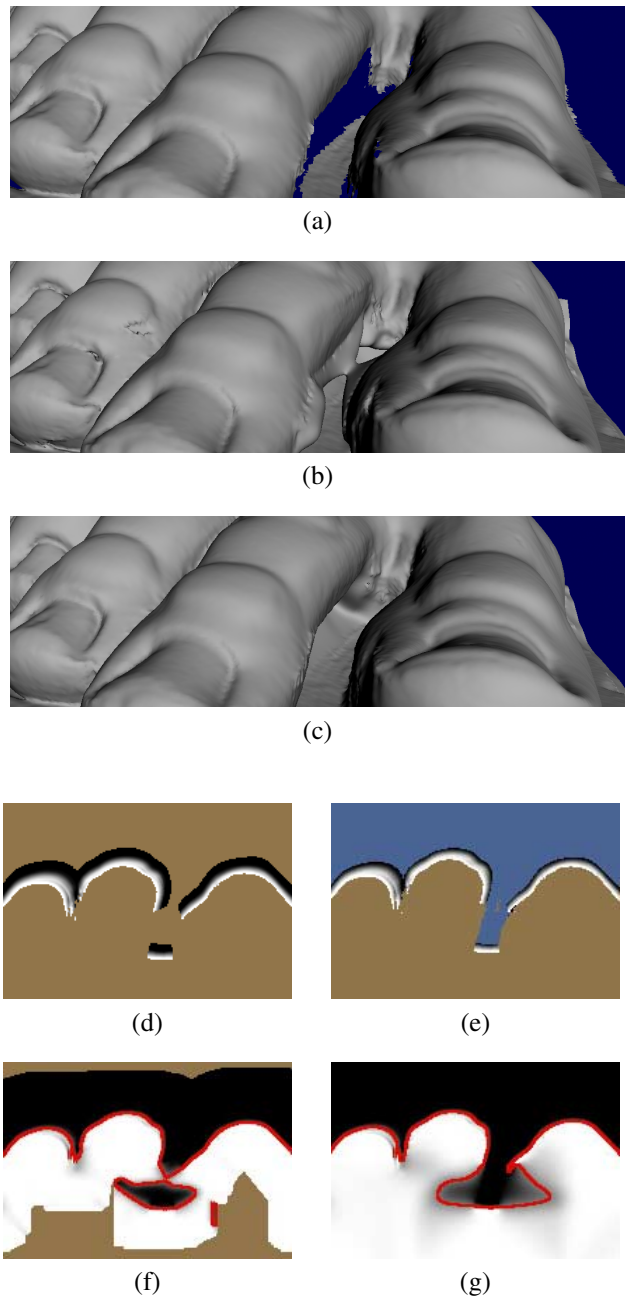


Figure 9: Using line-of-sight constraints. (a) Between two of David's toes, occlusion causes a hole with ambiguous topology. (b) Filling the hole results in an inappropriate bridge. (c) The addition of line-of-sight information forces the correct topology. (d,e) Slices of the source volume without and with this constraint, respectively. (f,g) The same slices of the diffused volume. Note the bridge that forms in (f) but is avoided in (g). Further diffusion will reduce the protrusion in (g) near where the bridge formed in (f).

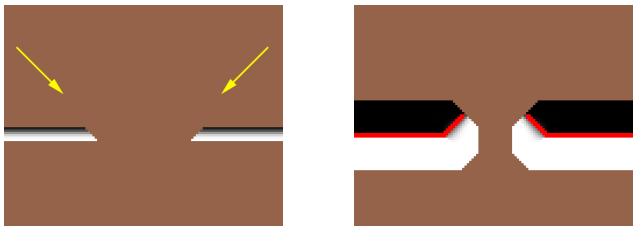


Figure 10: The problem of oblique scans, illustrated in 2D (in closeup). (a) The surfaces bordering this hole were scanned at an angle, indicated by the arrows. If VRIP is used to derive the source term, we obtain angled boundaries between valid and invalid voxels, as shown (compare to figure 4a). (b) Diffusion of this source term causes the zero set (red line) to incorrectly angle upwards.

The value of line-of-sight constraints is demonstrated in figure 9. This example shows the first two toes of the David's right foot. In this case, the diffusion process made a topological choice to build a bridge between the toes. While the bridge is consistent with the shape of this large hole, it is not the correct topology. By adding line-of-sight information, we can prevent this bridge from forming, as shown in the figure. (Note that Figure 8 did not use space carving; the bridge occurs only for specific parameter settings, and it did not happen to form in that particular run.)

One limitation of our present implementation is that, since our source term comes from VRIP, the scanner's lines of sight define the boundary between $w_s = 0$ and $w_s > 0$. Generally this boundary is approximately perpendicular to the scanned surface, but when all available scans were taken obliquely it may be angled. Since the zero set of our diffusion process tends to propagate perpendicularly to the boundary of w_s, this can cause undesired kinks in the reconstructed surface, as shown in figure 10. This behavior seems independent of the choice of convolution filter; every isotropic filter we have tried behaves similarly. The correct solution to this problem is to create a clamped signed distance function (or a filtered sidedness function) directly from the reconstructed surface [Frisken00], while retaining VRIP's line-of-sight information. We are currently addressing this problem.

5. Conclusions and future work

We have presented a new technique for filling holes in range scans by using diffusion to complete a volumetric representation of the surface. The method is simple and effective, and it always produces a closed, manifold triangle mesh without self-intersections. We have

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

demonstrated its feasibility on scanned datasets of significant size.

Although our algorithm is robust, it contains a number of free parameters. The distance at which we clamp our source terms, the size of our convolution filter h, the distance m from a hole boundary that we include in our computational domain, and the number of iterations for which we run our computation, are all parameters that affect our algorithm's running time, but have little or no effect on the shape of the resulting surface. On the other hand, the confidence α that we assign to known-empty voxels significantly affects the surface shape. At present, we have no principled method for deriving values for these parameters.

Diffusion processes and level set methods are very general computational frameworks [Sethian96, Osher01], so our algorithm can be extended in several ways. One extension would be to employ multigrid methods to further accelerate our diffusion process. Another would be to detect when a hole closes, and switch to a smaller computational domain that tracks the moving surface. Additionally, although we have incorporated into our algorithm one important form of ancillary information (line-of-sight), there are other constraints we might wish to add:

- Our algorithm leads to surfaces that generally blend well with the observed surface. However, greater control over properties of this surface – for example curvature or area – might be desirable. This can be achieved by incorporating terms related to these properties into the diffusion process [Sethian96, Whitaker98].
- Although line-of-sight constraints can often resolve ambiguous topology, there will always be cases that require high-level knowledge to disambiguate. We believe this knowledge is best provided through user intervention. The user could manipulate d_s directly – by sculpting [Perry01], or indirectly – by marking points in the volume as outside or inside.

Another area in which we believe progress can be made is in understanding what kinds of holes can arise during 3D scanning. Ignoring noise and absence of data due to low reflectance, it is probable that the holes caused by occlusion of the laser or camera line of sight have interesting geometric properties, and we might be able to make use of these properties in our hole filling algorithm. For example, given the geometry of a particular scanner, and making some assumptions about the coverage with which a scene was scanned, it might be possible to determine how far from a hole boundary the diffusion must grow searching for data to incorporate into the surface that closes that hole.

6. Acknowledgements

The idea of using volumetric diffusion to fill holes was first suggested to us by Lucas Pereira. Early discussions of this idea with Olaf Hall-Holt were invaluable. We thank Leo Guibas for guiding us through the computational geometry literature, Ron Fedkiw for helpful discussions about the heat equation, Szymon Rusinkiewicz for the triangulation algorithm used to generate figure 2, and all the members of the Digital Michelangelo Project team for providing the data and motivation for this research. We also thank our anonymous reviewers for many useful comments. Our sponsors are Stanford University, the National Science Foundation. the Mellon Foundation, the Paul G. Allen Foundation for the Arts, Interval Research Corporation, Intel, Sony, and MERL.

7. References

[Amenta98] Amenta, N., Bern, M., Kamvysselis, M., "A New Voronoi-Based Surface Reconstruction Algorithm," *Proc. SIGGRAPH '98*, ACM, 1998.

[Bajaj95] Bajaj, C.L., Bernardini, F., Xu, G., "Automatic Reconstruction of Surfaces and Scalar Fields From 3D Scans," *Proc. SIGGRAPH '95*, ACM, 1995.

[Bernardini99] Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G., "The Ball-Pivoting Algorithm for Surface Reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, October-December, 1999.

[Bertalmio00] Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C., "Image Inpainting," *Proc. SIGGRAPH 2000*, ACM, 2000.

[Carr01] Carr, J.C., Beatson, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R., "Reconstruction and Representation of 3D Objects with Radial Basis Functions," *Proc. SIGGRAPH 2001*, ACM, 2001.

[Chen95] Chen, Y., Medioni, G., "Description of Complex Objects from Multiple Range Images Using an Inflating Balloon Model," *Computer Vision and Image Understanding*, Vol. 61, No. 3, May, 1995.

[Curless96] Curless, B., Levoy, M., "A Volumetric Method for Building Complex Models from Range Images," *Proc. SIGGRAPH '96*, ACM, 1996.

[Curless97] Curless, B., *New Methods for Surface Reconstruction from Range Images*, PhD dissertation, Computer Science Department, Stanford University, 1997.

[Dey01] Dey, T.K., Giesen, J., Hudson, J., "Delaunay Based Shape Reconstruction from Large Data," *Proc. Symposium on Parallel and Large Data Visualization and Graphics* Surfaces and Parallel Rendering" , ACM, October, 2001.

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE

[Dinh01] Dinh, H., Turk, G., Slabaugh, G., Reconstructing Surfaces Using Anisotropic Basis Functions Proc. ICCV 2001.

[Edelsbrunner92] Edelsbrunner, H., Mücke, E.P., "Three-Dimensional Alpha Shapes," *ACM Transactions on Graphics*, Vol. 13, No. 1, 1994.

[Frisken00] Frisken, S., Perry, R., Rockwood, A., Jones, T., "Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics," *Proc. SIGGRAPH 2000*, ACM, 2000.

[Godin01] Godin, G., Beraldin, J.-A., Rioux, M., Levoy, M., Cournoyer, L., Blais, F., "An Assessment of Laser Range Measurement of Marble Surfaces," *Proc. Fifth Conference on optical 3-D measurement techniques*, Vienna University of Technology, Vienna, Austria, 2001.

[Hoppe92] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., "Surface reconstruction from unorganized points," *Proc. SIGGRAPH '92*, ACM, 1992.

[Landau87] Landau, L.D., Lipshitz, E.M., *Fluid Mechanics*, 2nd edition, Butterworth Heinemann Oxford, 1987.

[Levoy00] Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., Fulk, D., "The Digital Michelangelo Project: 3D scanning of large statues," *Proc. SIGGRAPH 2000*, ACM, 2000.

[Lorensen87] Lorensen, W.E., Cline, H.E., "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *Proc. SIGGRAPH '87*, ACM, 1987.

[Montani94] Montani, C., Scateni, R. Scopigno, R., "A modified look-up table for implicit disambiguation of marching cubes," *Visual Computer*, Vol. 10, No. 6, 1994.

[Osher01] Osher, S., Fedkiw, R., "Level Set Methods: An Overview and Some Recent Results," *J. Comput. Phys*, Vol. 169, 2001, pp. 463-502.

[Perona90] Perona, P., Malik, J., "Scale-space and edge detection using anisotropic diffusion," *IEEE PAMI*, Vol. 12, 1990, pp. 629-639.

[Perry01] Perry, R., Frisken, S., "Kizamu: A System for Sculpting Digital Characters," *Proc. SIGGRAPH 2001*, ACM, 2001.

[Porter84] Porter, T., Duff, T., "Compositing digital images," *Proc. SIGGRAPH '84*, ACM, 1984.

[Sapiro01] Sapiro, G., *Geometric partial differential equations and image analysis*, Cambridge University Press, 2001.

[Sethian96] Sethian, J.A., *Level set methods: evolving interfaces in geometry,* fluid mechanics, computer vision, and materials science" , Cambridge University Press, 1996.

[Turk94] Turk, G., Levoy, M., "Zippered Polygon Meshes from Range Images," *Proc. SIGGRAPH '94*, ACM, 1994.

[Whitaker98] Whitaker, R., "A Level-set Approach to 3D Reconstruction from range data," *International Journal of Computer Vision*, Vol. 29, No. 3, October, 1998.

[Wheeler98] Wheeler, M.D., Sato, Y., Ikeuchi, K., "Consensus Surfaces for Modeling 3D Objects from Multiple Range Images," *Proc. ICCV '98*.

[Wojciech00] Wojciech, W., Buehler, C., Raskar, R., Gortler, S.J., McMillan, L., "Image-Based Visual Hulls," *Proc. SIGGRAPH 2000*, ACM, 2000.

[Zhao01] Zhao, H.-K., Osher, S. Fedkiw, R., "Fast Surface Reconstruction using the Level Set Method," *Proc. First IEEE Workshop on Variational and Level Set Methods*, in conjunction with Proc. ICCV '01, IEEE, 2001, pp. 194-202.

Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT'02)
0-7695-1521-5/02 $17.00 © 2002 IEEE