Reconstructing Surface Discontinuities by Intersecting Tangent Planes of Advancing Mesh Frontiers

Luping Zhou, Indriyati Atmosukarto, Wee Kheng Leow, Zhiyong Huang
School of Computing, National University of Singapore

Contact person: Indriyati Atmosukarto
School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543, Singapore
Phone: (65) 874-4925
Fax: (65) 779-4580
Email: indri@comp.nus.edu.sg

Keywords: polygonization, intersecting tangent plane, surface normal re-estimation

Reconstructing Surface Discontinuities by Intersecting Tangent Planes of Advancing Mesh Frontiers

Luping Zhou, Indriyati Atmosukarto,
Wee Kheng Leow, Zhiyong Huang *
School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543, Singapore
zhoulupi, indri, leowwk, huangzy@comp.nus.edu.sg

Abstract

Reconstruction of surface discontinuities from unorganized 3D points is a difficult problem. Both the polygonization and surface fitting approaches to surface reconstruction face the same chicken-and-egg problem: to correctly reconstruct surface discontinuities, the points that lie on the same side of the discontinuities should be used to reconstruct the surfaces to form the discontinuities. However, to know whether the points lie on the same or different sides, the algorithm needs to know the locations and orientations of edge discontinuities, which are not directly available in the unorganized point set.

This paper presents an elegant method of overcoming the above problem. The method reconstructs an object's surface by constructing meshes at flat surfaces and advancing the mesh frontiers towards predicted surface discontinuities. As the frontiers of two or more meshes approach a discontinuity from its different sides, tangent planes at the mesh frontiers are estimated using the points within the frontiers, thus eliminating the problem of accidentally using points on different sides of the discontinuity to estimate a single tangent plane. Then, the tangent planes are intersected to form the discontinuity. Quantitative evaluation shows that this method can estimate the tangent planes very accurately, and can reconstruct surface discontinuities even when 3D points are not sampled at the discontinuities.

Keywords: polygonization, intersecting tangent plane, surface normal re-estimation.

*This research is supported by NUS ARF R-252-000-051-112

1. Introduction

In 3D model acquisition, an object is scanned and a large set of 3D points on the surface of the object are sampled. The object's surface can be reconstructed from the 3D points using one of two approaches: surface fitting and polygonization. The first approach accomplishes the task by finding a set of parametric surfaces to fit the 3D points as closely as possible [1, 8, 9, 11, 12, 14]. Typically, a single smooth, continuous surface is used to fit a subset of points, and neighboring surfaces are blended to form smooth joints. On the other hand, the second approach connects neighboring 3D points into triangles or polygons to approximate the object's surface [2, 3, 6, 7, 10].

These two approaches are different and each has its own strengths and weaknesses. Interestingly, when it comes to reconstructing surface discontinuities (i.e., edges, corners, etc.) from a set of unorganized points, both approaches face the same chicken-and-egg problem, which has not been solved in an elegant and efficient manner. For the first approach, 3D points lying on different sides of a discontinuity such as an edge should be fitted with different surfaces so that the discontinuity can be formed from the intersection of the surfaces. This approach thus requires knowledge about the location and orientation of the edge, which are not available in a set of unorganized points until the edge is reconstructed.

For the second approach, the frontier advancing algorithm that we proposed in [3] can be used to reconstruct a discontinuity as a natural consequence of meeting frontiers if 3D points on the discontinuity are sampled. The algorithm can estimate the location of the discontinuity but not the orientation of an edge. On the other hand, if the discontinuity is not sampled, the natural solution would be to estimate the tangent planes at the points near the discontinuity and to form the discontinuity by intersecting the tangent planes. The tangent plane can be estimated by computing its normal vector using the sample points. Now, we face the same problem: only the points lying on the same side of a discontinuity should be used to estimate a tangent plane, and this again requires knowledge about the location and orientation of edge discontinuity.

It turns out that this problem can be solved by performing tangent plane estimation and frontier advancing polygonization *at the same time*. Our frontier advancing polygonization algorithm has already been described in [3], with test results for general, irregularly shaped objects. A brief summary of the algorithm will be provided in Section 3. Complementing our earlier work, this article presents the method of reconstructing surface discontinuities by intersecting the tangent planes of advancing mesh frontiers. In particular, it focuses on the accurate estimation of tangent planes (Section 4), the formation of discontinuities by intersecting tangent plane (Section 5), and quantitative evaluation using synthetic 3D models (Section 6).

2. Related Work

Direct estimation of surface normals from unorganized 3D points is not a simple task. Although surface normals are inferred in some surface reconstruction algorithms, they are obtained mostly as by-products of the algorithms which depend on special reconstruction processes. For example, in Amenta's crust algorithm [2], surface normals are inferred from the Voronoi diagram built in previous stages of the algorithm. Nevertheless, there are several algorithms which directly estimate surface normals. These methods are discussed in this section.

Hoppe et al. [7] applied Principal Component Analysis on a small cluster of points to estimate the normal at the center of the cluster. It proves to be a robust method for points on smooth surfaces, but is not accurate at surface discontinuities because a cluster may contain points lying on different sides of a discontinuity (see Section 1 for details).

Hoppe et al. also made use of tangent planes in surface recovery. However, the tangent planes were used to define a signed distance for the sample points. A marching cube algorithm was then applied to compute the isosurface that formed the final polygonization result. On the other hand, we use the intersection of tangent planes to form surface discontinuities.

The method of Mediori [13] uses *Tensor Voting* to estimate surface normals. It is grounded on tensor calculus for data representation and tensor voting for data communication. The inputs of this method can be 3D points, 3D curve segments, or 3D surface patches. The possible surface normals at these input sites are represented as tensors. The actual normals at an input site is estimated according to a weighted sum of the tensors and the distance to its neighbor determines the weight of the contribution of the neighbor. Because the information about surface discontinuities is not available in the tensors, this method cannot accurately estimate surface normals at discontinuities.

Another general approach to estimating surface normals is to fit planar or quadratic surfaces to sample points [1, 8, 9, 11, 12, 14]. However, these methods suffer from the chicken-and-egg problem discussed in Section 1.

The method of Benko et al. [4, 5] is particularly interesting. First, it triangulates a point cloud to determine the neighborhood and connectivity of the sample points. Then, it segments the sample points into disjoint subsets such that the points in each subset can be fitted by one or more second order analytic surfaces. The surface normal at each sample point is then given by the normal of the fitted surface at the point. In addition, a least-square plane is fitted to each point and its neighbors. Points with large fitting errors are considered as lying on highly curved surfaces or at surface discontinuities. The triangles at these regions are removed. Next, an adjacency graph is constructed from the remaining fitted regions, which is used to determine the edges of the smooth faces (which consists of adjacent smooth regions). The edges are then constructed by intersections between the surfaces that represent the smooth faces.

This method consists of many stages, some of which are very complex. First, the region segmentation process involves complicated hypothesis testing. Second, at least three surface fitting processes are involved: one each for estimating surface normals, identifying surface discontinuities, and reconstructing complex smooth surfaces that cannot be fitted by a single second order surface. Moreover, the method requires an initial triangulation of point cloud which, as discussed in Section 1, may produce incorrect neighborhood and connectivity relationships between the points near surface discontinuities. The error incurred is expected to propagate to subsequent stages of the method, degrading the overall performance of the method.

In comparison, our method is simpler and more elegant. Moreover, as will be seen in Section 3, it can also accurately estimate surface normals at and near surface discontinuities.

3. Overview of Algorithms

The tangent plane intersection algorithm is built upon our frontier advancing polygonization algorithm described in [3]. Hence, it is useful to first provide a summary of the frontier advancing algorithm.

3.1 Summary of Frontier Advancing Polygonization Algorithm

The frontier advancing algorithm consists of the following stages:

1. Identifying reliable points

2a. Constructing a closed mesh around a point

2b. Merging meshes and advancing mesh frontiers

In stage 1, Principal Component Analysis (PCA) is used to estimate surface normals in a manner similar to [7]. Given a set of 3D points, PCA computes three eigenvectors $\mathbf{e}_i$ and eigenvalues λ_i, $i = 1, 2, 3$ in decreasing eigenvalue. If the set of points lie on a surface without discontinuity, the third eigenvector $\mathbf{e}_i$ would be parallel to the normal of the surface that best fit the points. However, when a surface discontinuity exists, points lying on different sides of the discontinuity may be used in the estimation. Consequently, the estimated normal would be an average of the normals of the different surfaces (Fig. 1).

If the set of points lie on a flat surface, the third eigenvalue λ_3 would be very small. These points are called *reliable points* because we are certain of the types of the surfaces at the points. For a set of points distributed near a surface discontinuity or a smooth curve surface with large curvature, λ_3 would be large compared to λ_1 and λ_2. Therefore, λ_3 can be used to distinguish between points lying on flat surfaces from other points. However, it cannot differentiate between points lying near discontinuities and those on smooth surfaces with large curvature. So, these points are called *ambiguous points*.

Stage 2a and 2b proceed simultaneously. After identifying reliable points, an initial mesh is first constructed around a randomly chosen reliable point based on the properties of Delaunay triangulation. It advances the mesh frontiers by adding neighboring reliable points until ambiguous points are encountered. This step is repeated for all reliable points.

After all the reliable points have been added to the mesh, there will be one or more disconnected meshes which are separated by ambiguous points. At this stage, the algorithm further extends the frontiers by completing the meshes around ambiguous points in increasing order of ambiguity. The mesh frontiers would finally meet at the most ambiguous points to form surface discontinuities. If 3D points along the edges and at the corners are sampled, then the algorithm will form an edge where two advancing frontiers meet and a corner where three or more frontiers meet. Otherwise, the algorithm will construct approximations of the edges and corners.

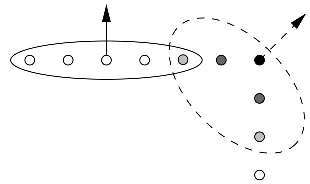


Figure 1. Third eigenvector (dashed arrow) of the points at the corner is not a correct estimate of the surface normals at the points. Darker points represent more ambiguous 3D points.

3.2 Overview of Tangent Plane Intersection Algorithm

As described in Section 3.1, the surface normals estimated by PCA may not be accurate for ambiguous points. To improve the estimation accuracy, surface normals at ambiguous points should be re-estimated and the re-estimation process should use only the points that lie on the same side of a discontinuity.

The basic idea of the tangent plane intersection algorithm is as follows. The first two stages (1 and 2a) of the algorithm are similar to those of the frontier advancing algorithm. The difference lies in how the two algorithms identify mesh edges. The frontier advancing algorithm uses a geometric approach to identify Delaunay edges whereas the tangent plane intersection algorithm uses the intersection of tangent planes. For sample points that lie on nearly flat surfaces, the tangent plane intersection algorithm reduces to the frontier advancing algorithm. Hence, tangent plane intersection is performed only for ambiguous points that are predicted to lie near surface discontinuities.

Before Stage 2b of the tangent plane intersection algorithm is executed, surface normals at ambiguous points must be re-estimated to obtain more accurate estimations. The re-estimation process first identifies sample points that are predicted to fall on the same side of a discontinuity as an ambiguous point near the mesh frontier (Section 4). The normals at these sample points are then used to estimate the normal at the ambiguous point. Once the normals of the ambiguous points near the mesh frontiers are re-estimated, tangent planes at the points are intersected to derive new mesh edges that are added to the mesh frontier (Section 5). Thus, the normals at the points in the mesh are reliable. In this way, normal re-estimation and frontier advancement alternate until all sample points are included in the mesh. With this concept in mind, we can now look at the normal re-estimation process in detail.

4. Surface Normal Re-estimation

Re-estimation of surface normals at ambiguous points proceed together with frontier advancement in increasing order of ambiguity. An ambiguous point next in queue at which the surface normal is to be re-estimated always lie just beyond a mesh frontier with no other intervening free points, which are the points not in the mesh. The surface normal re-estimation process consists of two steps:

1. Finding consistent neighbors
   Consistent neighbors refer to the points that are predicted to lie on the same side of a discontinuity.

2. Recomputing surface normals
   Normal re-estimation is performed only on consistent neighbors.

4.1 Determining Consistent Neighbors

Two methods of determining consistent neighbors have been investigated: point-based method and mesh-based method. The point-based method uses normals at the sample points whereas the mesh-based method uses the normals at the points that are inside the advancing mesh.

4.1.1 Point-based method

Conceptually, a point at a discontinuity has a locally maximum ambiguity. Therefore, a direct method of finding the consistent neighbors is to look for local maxima of ambiguity and collect consistent neighbors accordingly. However, this approach is computationally expensive.

An alternative method which is computationally more efficient is to first collect all the neighbors of a sample point p and then remove those points that are predicted to lie on different sides of a discontinuity as p. Because a discontinuity is a meeting of two or more surfaces, the difference in surface normal would be relatively large between points on different sides of the discontinuity. This could be used to predict whether two neighboring points are lying on the same surface.

Figure 2 illustrates this concept. Suppose that the neighbors p_1, p_2, and p_3 of an ambiguous point q are also ambiguous. Point p_1 has a similar normal as q, that is $||\mathbf{n} \cdot \mathbf{n}_1|| < \Gamma_n$, a fixed threshold, while points p_2 and p_3 have very different normals: $||\mathbf{n_2} \cdot \mathbf{n}_3|| \geq \Gamma_n$. We can draw a plane at p_2 orthogonal to the direction from q to p_2 and exclude all the points on the opposite side of the plane (Fig. 2, shaded area). Even if there is no discontinuity between p and p_2, it is still reasonable to exclude the points on the opposite side of the plane because their normals are very different from that of q. This method is repeated for each neighbor p_i whose normal $\mathbf{n}_i$ differs significantly from the normal $\mathbf{n}$ of q. The remaining neighbors are the consistent neighbors.

This method turns out to be not very accurate. As discussed in section 3.1, the initial PCA estimate at a discontinuity may include points on different sides of the discontinuity (Fig. 1). As a result, the initial estimates of the normals at ambiguous points may not be accurate. Without differentiating between accurate and inaccurate estimates, this method is therefore not expected to work well.

4.1.2 Mesh-based method

The mesh-based method is performed as follows. Given an ambiguous point q at which the normal is to be re-estimated, the mesh-based method first determines the

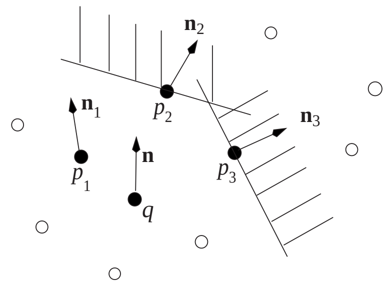


Figure 2. Using constraint planes to find consistent neighbors. Sample points on the shaded side are excluded because their normals are very different from the normal $\mathbf{n}$ of the sample point q. Dark points are ambiguous points while light points are reliable points.

neighboring points of q that are already in some meshes. Since normal re-estimation is performed together with frontier advancement, sample points that are already in the meshes always have accurately estimated normals. These neighboring points are grouped into clusters based on the similarity of their normals.

Next, for each cluster i, the point p_i nearest to the ambiguous point q are determined. Point p_i should be located near the frontier (Fig. 3). A pairwise intersection of the tangent planes at points p_i and p_j is performed. The resulting intersection line should be located near a discontinuity if one exists. Based on the resulting pairwise intersection lines, we determine on which sides of the intersection lines does q lie and collect the corresponding cluster of points (Fig. 3). The resulting cluster of points would be the consistent neighbors of q. If there is only one cluster after the grouping prorcess, then the points in this cluster are regarded as the consistent neighbors of q.

Instead of performing all $m(m-1)/2$ pairwise intersections of the tangent planes of m clusters, the consistent neighbors can also be determined by a more efficient $O(m)$ algorithm:

> Choose an initial cluster M_1 and set p_1 as point w.
> Repeat for each nearest neighbor p_i, $i = 2, \ldots, m$,
>   Intersect the tangent planes at w and p_i
>   to produce an intersection line L.
>   If q is on the same side of L as p_i then,
> $w = p_i$.

At the end of the algorithm, the mesh points in the cluster that contains point w are the consistent neighbors.

Consider Fig. 3 for example. Suppose p_1 is considered first, and the algorithm intersects the tangent planes at points p_1 and p_2 to produce an intersection line L_{12}. Since

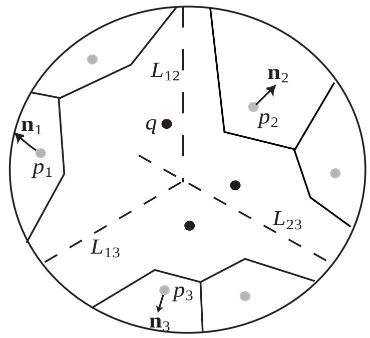


Figure 3. The mesh-based method for collecting consistent neighbors. There are three groups of mesh points in the neighborhood of the ambiguous point q. The consistent neighbors are those on the same side of the predicted discontinuity L_1 and L_2 as point q.

q lies on the same side as p_1, w is set as p_1. The algorithm continues by intersecting the tangent planes at points p_1 and p_3 and produces the intersection line L_{13}. Since q lies on the same side as p_1, w is set as p_1 and the mesh points in p_1's cluster are the consistent neighbors of q.

Now consider another possibility. Suppose p_2 is considered first, and the tangent planes at p_2 and p_3 are intersected to produce the intersection line L_{23}. Since q lies on the same side as p_2, w is set as p_2. Next, the tangent planes at p_1 and p_2 are intersected to produce the intersection line L_{12}. Because p_1 lies on the same side as q, w is set as p_1 and the mesh points in the cluster that contains p_1 are still considered the consistent neighbors. Therefore, the algorithm correctly identifies consistent neighbors regardless of the order in which the points p_i are considered.

4.2 Estimating Normal from Consistent Neighbors

A straightforward method to estimate the normal is to apply PCA on the consistent neighbors. This method is not accurate because PCA's estimate is correct for the normal at the centroid of the neighborhood but the ambiguous point tends to fall at the fringe of the neighborhood (Fig. 3). Therefore, a more accurate method is to perform linear extrapolation.

The linear extrapolation method uses both the positions and the normals of the consistent neighbors to infer the normals at the ambiguous point. To perform linear extrapolation correctly, the normals of the consistent neighbors must be pointing at similar directions, i.e, $\mathbf{n}_i \cdot \mathbf{n}_j \geq 0$ for consistent neighbors i, j. Since the normals are initially estimated using PCA (Stage 1) and PCA does not guarantee a consistent orientation of the estimated normals, a local alignment of normals similar to the approach of Hoppe et al. [7] is applied.

After aligning the normals, linear extrapolation is applied. Since the points p_i and ambiguous point q lie on the same side of a predicted discontinuity, the surface containing these points would be a plane or a small patch of a curved surface. Hence, $\mathbf{n}_i$ would be approximately linearly related to p_i:

$$\mathbf{n}_i = \mathbf{A}\mathbf{p}_i + \mathbf{B} \quad \text{for each } i \tag{1}$$

where $\mathbf{A}$ is a 3×3 matrix and $\mathbf{B}$ is a 3×1 vector. The matrices $\mathbf{A}$ and $\mathbf{B}$ are computed by finding the matrices that minimize the error E

$$E = \sum_{i=1}^{n} ||\mathbf{n}_i - (\mathbf{A}\mathbf{p}_i + \mathbf{B})||^2. \tag{2}$$

After obtaining $\mathbf{A}$ and $\mathbf{B}$, the normal $\mathbf{n}$ at q can be computed as $\mathbf{n} = \mathbf{A}\mathbf{q} + \mathbf{B}$.

5. Computing Mesh Polygon

The polygonization process constructs a polygon around a sample point c, and merges the polygon into the existing mesh. The polygon consists of vertices v_i, $i = 1, \ldots, n$, with $v_{n+1} = v_1$. The polygon edges are contained in the intersection lines $L(c, p_i)$ produced by the intersection of the tangent planes at c and at its neigbor p_i. The vertex v_{i+1} is computed as the intersection of line $L(c, p_i)$ and $L(c, p_{i+1})$. In practice, it is easier to compute the vertex v_{i+1} by intersecting 3 planes, the tangent planes at c, p_i, and p_{i+1} instead of two 3-D lines. That is, v_{i+1} is the solution of the following equation:

$$\begin{aligned} (\mathbf{x} - \mathbf{c}) \cdot \mathbf{n} &= 0 \\ (\mathbf{x} - \mathbf{p}_i) \cdot \mathbf{n}_i &= 0 \\ (\mathbf{x} - \mathbf{p}_{i+1}) \cdot \mathbf{n}_{i+1} &= 0 \end{aligned} \tag{3}$$

In this polygon construction process, the first point p_1 is the sample point nearest to c. The algorithm determines the next point p_2 by finding a point whose intersection point v_2 is nearer to the line from c to p_1 than are any other possible intersections. All subsequent intersection points v_{i+1} are computed in a similar manner except they are the inersections that are closest to their predecessors v_i along the intersection lines $L(c, p_i)$ (Fig. 4).

The vertices v_i, $i = 1, \ldots, n$, make up the polygon which includes the sample point c within the polygon. Once a polygon is constructed around a point c, it is merged into the existing mesh frontier in the same way as the frontier advancing algorithm in [3].

6. Test Results and Discussions

In order to perform quantitative analysis of the normal estimation methods, synthetic data with known normals were used. These data points were sampled randomly from

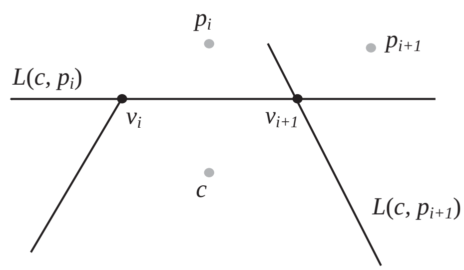


Figure 4. The next polygon vertex v_{i+1} is computed by the intersection between $L(c, p_i)$ and $L(c, p_{i+1})$, and v_{i+1} is nearer than other intersections are to v_i.

Table 1. Numbers of reliable and ambiguous points in each test case.

test case	reliable points	ambiguous points
cube	394	207
cylinder	894	258
hemisphere	690	171

the surfaces of a cube, a cylinder, and a hemisphere (Figs. 5, 6, 7). The 3 test data had different numbers of reliable and ambiguous points (Table 1). To test the reconstruction process, the edges and the corners were not sampled so as to demonstrate that surface discontinuities can be reconstructed using the tangent plane intersection algorithm. On the other hand, to assess the accuracy of normal estimations, points were sampled along discontinuities.

Five test cases were performed for each test data: (1) PCA: apply PCA on 3D sample points; (2) PCA/P: apply PCA on consistent neighbors obtained using the point-based method; (3) LE/P: apply linear extrapolation on consistent neighbors obtained using the point-based method; (4) PCA/M: apply PCA on consistent neighbors obtained using the mesh-based method, and (5) LE/M: apply linear extrapolation on consistent neighbors obtained using the mesh-based method. The mean error measured in terms of the angle between the actual and the estimated normals was computed. For points lying exactly on a surface discontinuity, the normals of the surfaces that form the discontinuity were taken as the possible normal vectors. So, an edge point would have 2 possible normals, and a corner would have more than 2 possible normals. The measured errors at these points were taken as the smallest error between the estimated value and the possible normals.

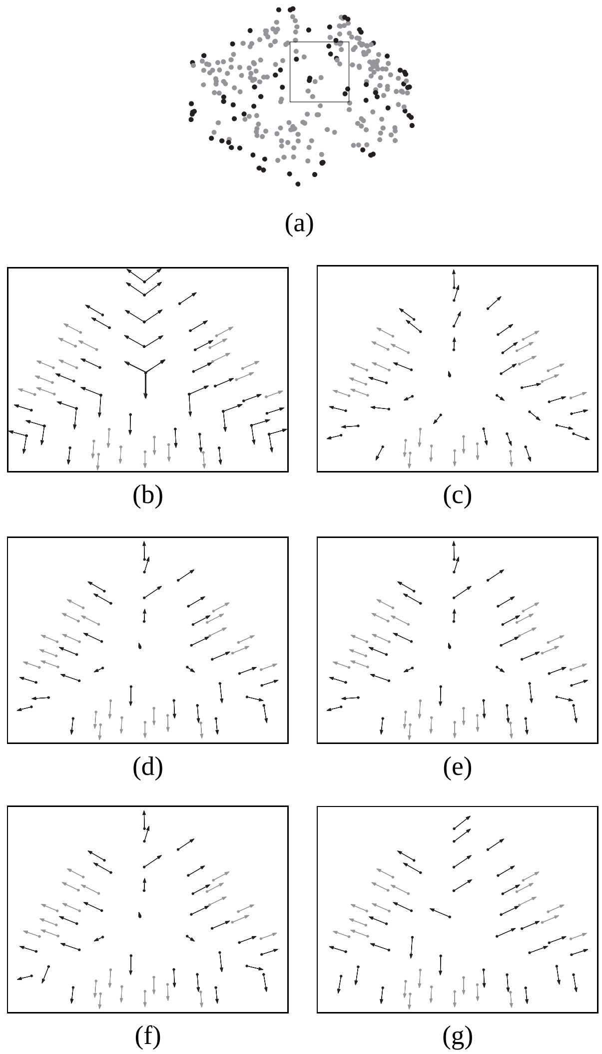


Figure 5. Visual comparison of normal estimation for the cube. Dark color denotes ambiguous points and light color denotes reliable points. (a) Random points sampled on the surface. (b) Actual normals. (c-g) Normal vectors estimated by (c) PCA, (d) PCA/P, (e) LE/P, (f) PCA/M, and (g) LE/M.

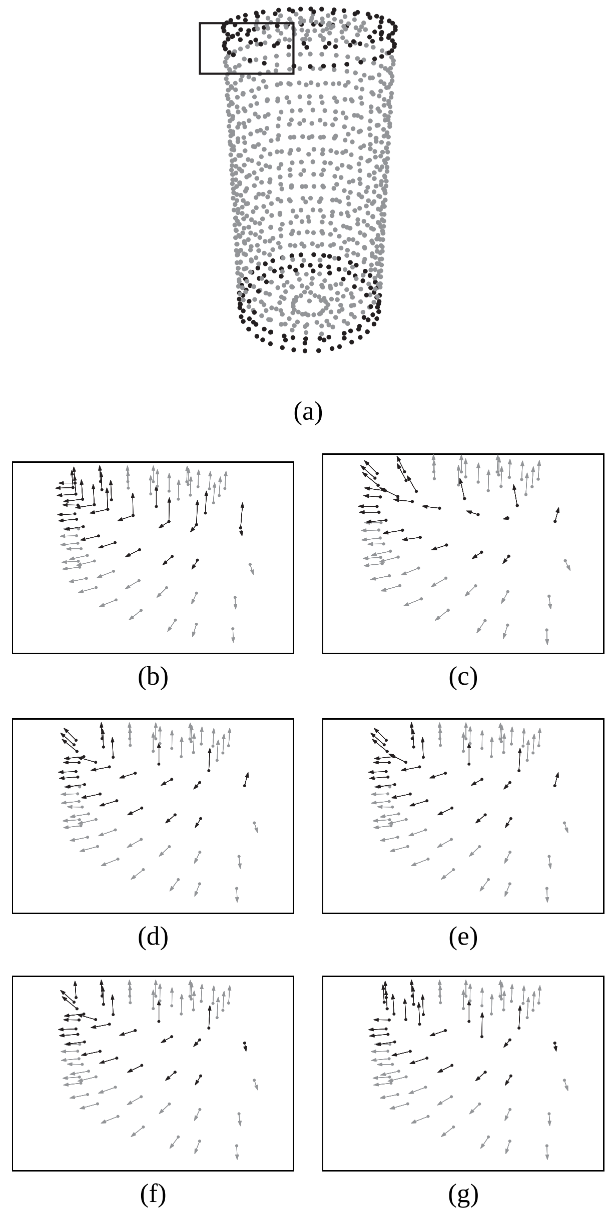


Figure 6. Visual comparison of normal estimation for the cylinder. (a) Random points on the surface. (b) Actual normals. (c-g) Normal vectors estimated by (c) PCA, (d) PCA/P, (e) LE/P, (f) PCA/M, and (g) LE/M.

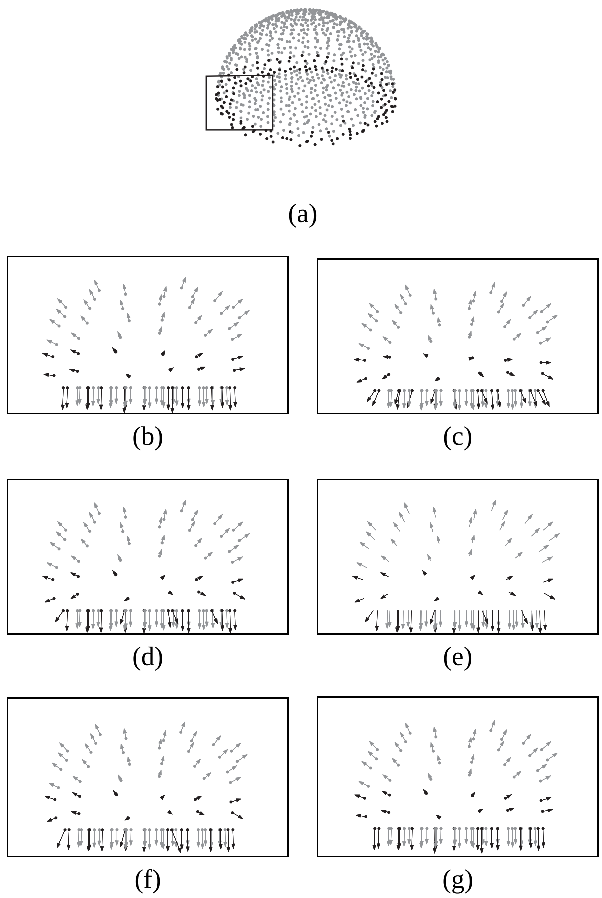


Figure 7. **Visual comparison of normal estimation for the hemisphere. (a) Random points sampled on the surface. (b) Actual normals. (c-g) Normal vectors estimated by (c) PCA, (d) PCA/P, (e) LE/P, (f) PCA/M, and (g) LE/M.**

Table 2 shows the results produced by five methods of normal estimation. The results show that both PCA/P and LE/P can produce more accurate normal estimation than PCA. LE/P's error estimation for reliable points is slightly larger than that of PCA/P. However, its estimation for ambiguous points is more accurate than that of PCA/P. This is because an ambiguous point tends to fall on one side of its consistent neighbors. As a result, the centroid of the consistent neighbors may not coincide with the position of the ambiguous point. However, PCA always computes the third eigenvector with respect to the centroid. Therefore, the normals at the ambiguous points estimated by PCA are less accurate than those estimated using linear extrapolation.

PCA/M's error estimation is better than both PCA/P and LE/P for reliable and am-

Table 2. Comparison of mean error (angle) of normal estimation. PCA: Principal Components Analysis on 3D sample points, LE: linear extrapolation, P: the point-based method for finding consistent neighbors, M: the mesh-based method for finding consistent neighbors.

	cube		cylinder		hemisphere	
methods	reliable points	ambiguous points	reliable points	ambiguous points	reliable points	ambiguous points
PCA	1.478	24.764	2.983	30.163	2.654	35.428
PCA/P	0.369	9.072	1.430	14.328	1.647	18.466
LE/P	0.369	9.069	1.468	13.621	1.832	16.545
PCA/M	0.115	5.433	0.953	7.877	1.430	10.326
LE/M	0.000	0.018	0.334	2.336	0.598	2.869

biguous points because the mesh-based method is more reliable than the point-based method in determining consistent neighbors. LE/M combines the strengths of both linear extrapolation and mesh-based method of determining consistent neighbors. Therefore, its estimation of surface normals is most accurate. In particular, its estimation of the normals at ambiguous point is roughly one order of magnitude better than those of PCA.

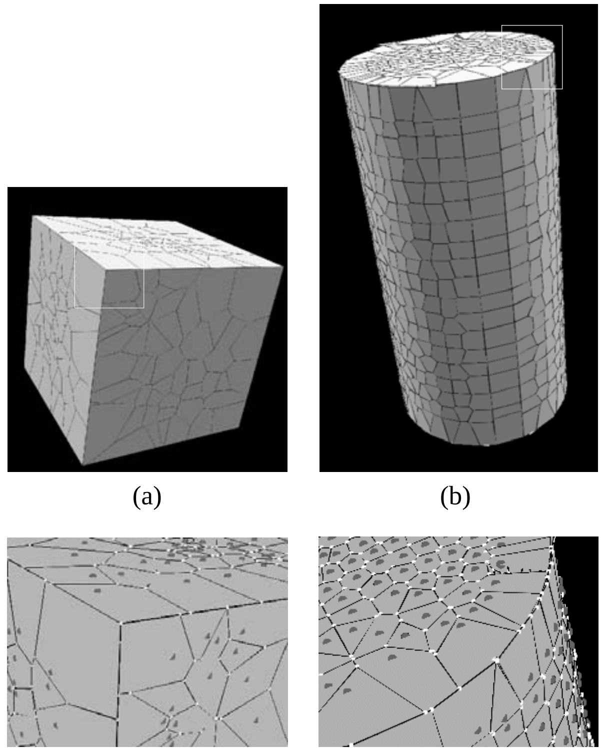
Visual comparisons of selected regions of the 3 test data are shown in Figs. 5, 6, and 7. These figures clearly show that the normals at the ambiguous points (dark arrows) estimated by PCA differ significantly from the actual normals. The normals estimated by PCA/P and linear extrapolation are more accurate than that of PCA and the estimation of mesh-based linear extrapolation is the most accurate. Zoom-in views of the reconstructed surfaces (Figs. 8, 9) show that the tangent plane intersection method can derive the points at the edges and the corners even though they are not sampled.

A comparison of the polygonization produced by the tangent plane intersection method with the frontier advancing Delaunay triangulation method [3] is shown in Figure 9. The curved edges produced by frontier advancing Delaunay triangulation (Fig. 9(c)) is jagged and is not as smooth as those constructed by the tangent plane intersection method (Fig. 9(a, b)). This result clearly demonstrates the strength of the tangent plane intersection algorithm.

Figure 10 shows the experiments conducted on the foot and the mannequin standard data points. The algorithm is able to derive the surface discontinuities on the foot such as the toes and on the face such as the eyes, nose, mouth, and ears.

Our tangent plane intersection method is similar in strategy to that of Benko et al. [4, 5], but is simpler and more elegant:

1. Our method uses linear estimation to determine ambiguous points and estimate surface normals. It is simpler than the second order surface fitting of Benko et al., and yet produces very accurate estimations as shown in the test results (Table 2). On the other hand, complicated region segmentation and hypothesis testing are



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

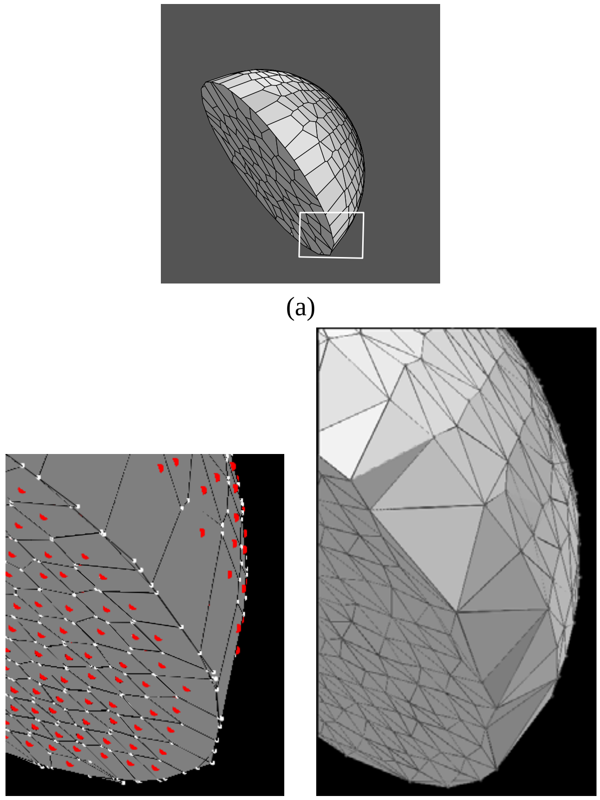
Figure 8. Polygonization of (a) a cube and (b) a cylinder. (c, d) Zoom-in views of surface discontinuities. Tangent plane intersection method could derive the points at surface discontinuities (light points) from the sample points (dark points) even though the discontinuity were not sampled.

required in their method.

2. In our method, surface normal estimation and polygonization, which includes reconstruction of surface discontinuities, are performed together in a single process, thereby completely resolving the chicken-and-egg problem. In contrast, the method of [4, 5] involves a lot of stages, and does not completely resolve the problem.

7. Conclusions

This paper presented a method of reconstructing surface discontinuities by intersecting the tangent planes of advancing mesh frontiers. The method incorporates intersec-

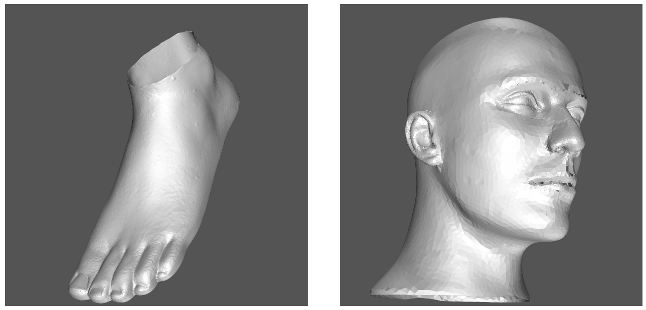


(a)

(b) (c)

Figure 9. Polygonization of a hemisphere. Since the curve edges were not sampled, Delaunay triangulation produced jagged edges (c). On the other hand, the tangent plane intersection algorithm reconstructed smooth edges (a, b).

tion tangent planes into the frontier advancing algorithm. By performing surface normal estimation together with frontier advancement, the method can accurately estimate surface normals for points at and near surface disconituities. Quantitative evaluation shows that the normal estimation obtained is accurate. In particular, the combination of linear extrapolation and mesh-based method is roughly one order of magnitude more accurate than PCA. Using the accurately estimated normals, the method can accurately reconstruct sharp discontinuities even when they are not sampled.



(a) (b)

Figure 10. Polygonization of (a) the foot and (b) the mannequin standard data points.

References

[1] M.-E. Algorri and F. Schmitt. Surface reconstruction from unstructured 3D data. In *In Proc. Computer Graphics Forum*, pages 47–60, 1996.

[2] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *Proc. SIGGRAPH '98*, pages 415–421, 1998.

[3] I. Atmosukarto, L. Zhou, W. K. Leow, and Z. Huang. Polygonizing non-uniformly distributed 3D points by advancing mesh frontiers. In *Proc. CGI '01*, 2001.

[4] P. Benko, G. Lukacs, R. R. Martin, and T. Varady. Algorithms to create B-rep models from measured point clouds. *Research Report of Geometric Modelling Laboratory, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest*, 6, 1998.

[5] P. Benko, R. R. Martin, and T. Varady. Algorithms for reverse engineering boundary representation models. *Computer Aided Design*, 33(11):839–851, 2001.

[6] H. Edelsbrunner and E. P. Mucke. Three-dimensional alpha shapes. *ACM TOG*, 13:43–72, 1994.

[7] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proc. SIGGRAPH '92*, pages 71–78, 1992.

[8] G. Kos, R. R. Martin, and T. Varady. Methods to recover constant radius rolling ball blends in reverse engineering. *Computer Aided Geometry Design*, 17(2):127 – 160, 2000.

[9] S. Lodha. Surface approximation with low degree patches with multiple representations. *Ph.D Thesis, Computer Science, Rice University, Houston, Texas*, 1992.

[10] C. Oblonsek and N. Guid. A fast surface-based procedure for object reconstruction from scattered points. *Computer Vision and Image Understanding*, 69(2):185–195, 1998.

[11] R. Poli, G. Coppini, and G. Valli. Recovery of 3D closed surfaces from sparse data. *Computer Vision, Graphics and Image Processing*, 60(1):1–25, 1994.

[12] D. Ruprecht and H. Muller. Free form deformation with scattered data interpolation methods. *In Proc. Geometric Modelling*, pages 171–200, 1995.

[13] C. K. Tang and G. Medioni. Inference of integrated surface, curve and junction descriptions from sparse 3D data. *IEEE Trans. PAMI*, 20(11), 1998.

[14] D. Terzopoulos, A. Witkin, and M. Kass. Constrains on deformable models: recovering 3D shape and non-rigid motion. *Artificial Intelligence*, 36:91–123, 1988.