A New Shape Dissimilarity Measuring Technique Between 3-D Models Using Z-buffer and Surface Roving Method

In Kyu Park
Multimedia Lab.
Samsung Advanced Institute of Technology
YONGIN 449-712, KOREA
saitpik@sait.samsung.co.kr

Kyoung Mu Lee
Dept. of Electronics and Electrical Eng.
Hong-Ik University
SEOUL 121-791, KOREA
kmlee@wow.hongik.ac.kr

Sang Uk Lee
School of Electrical Eng.
Seoul National University
SEOUL 151-742, KOREA
sanguk@diehard.snu.ac.kr

Abstract

Estimation of the shape dissimilarity between 3-D models is very important problem in both computer vision and graphics for 3-D surface reconstruction, modeling, matching and compression. In this paper, we propose a novel method called "surface roving technique' to estimate the shape dissimilarity between 3-D models. Unlike conventional methods, our 'surface roving' approach exploits a virtual camera and Z-buffer, which is commonly used in 3-D graphics, so that the corresponding points on different 3-D model can be easily identified, and also the distance between them is determined efficiently, regardless of the representation types of the 3-D models. Moreover, by employing the viewpoint sampling technique, the overall computation can be greatly reduced so that the dissimilarity is obtained rapidly without loss of accuracy. Experimental results show that the proposed algorithm achieves fast and accurate measurement of shape dissimilarity for different types of 3-D object models.

1 Introduction

In 3-D computer vision and graphics, shape recovery and modeling have been one of the major research fields during last few years. For surface modeling, multiresolution surface representation, object recognition and 3-D data compression, it is essential to estimate the geometric distortion or shape dissimilarity in object space, since the performance of an algorithm can not be evaluated quantitatively without it. In surface modeling [11], the shape dissimilarity between the original data and the generated surface model should be compared, and in polygonal mesh reduction [2][3][4][5][6][7][8][9][10], the dissimilarity between the original and the simplified mesh should also be determined to guide the model simplification process. While, in 3-D data compression [12], the dissimilarity is needed for analyzing the rate-distortion property. Similarly, deformable model management and range view registration which employs the iterative closest point (ICP) algorithm also invoke a shape dissimilarity measurement.

Consider a typical problem of finding the shape dissimilarity between a simplified mesh to an original mesh. Most existing methods use vertex-to-vertex [1][8], vertex-to-plane [3][9], point-to-surface [2][5], and surface-to-surface distance [4][6][7][10]. However, most of them are difficult to implement and require massive computation as well. Note that although the vertex-to-vertex distance is easy to implement, it dose not provide any correct measurement. An alternative method is to find the vertex-to-plane. However, a direct implementation of it is not trivial, since it usually requires a brute force search which is impractical when the surface model is complex. Point-to-surface and surface-to-surface distance provide more exact measurements, while finding the corresponding point set is still a problem. Thus, in most literature, by imposing additional assumptions and constraints, the brute force search is replaced by a local search for practical use, and few methods can be found which deal with the problem directly in general setting.

In this paper, we propose an efficient method to evaluate the shape dissimilarity between 3-D models, even though they are represented in different types, including point cloud, polygonal mesh, parametric surface, and voxel model. The reference and test model can be any of them. Only exceptional case is that point cloud cannot be used for the test model. Unlike the conventional geometric methods, our approach utilizes a Z-buffer and virtual camera commonly used in 3-D graphics [13]. By using them, the distance between corresponding points on different models can be obtained efficiently, which is then used to compute the shape difference between the models. Since the operation is performed on the geometry engine in graphic hardware, processing time is greatly reduced using 3-D graphics accelerator.

This paper is organized as follow. In Section 2, we de-

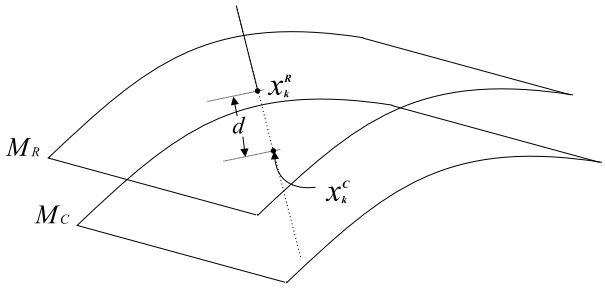


Figure 1. Distance between corresponding points.

fine the shape dissimilarity measure between 3-D shapes. Section 3 describes the 3-D graphics background which is used in the proposed approach. In Section 4, the proposed algorithm is described in detail. Experimental results are presented in Section 5, and the conclusion is drawn in Section 6.

2 Distance Measure

The shape difference is defined as the average distance from sample points on a reference model M_R to their corresponding points on a test model M_C. Ideally, the sample points are the whole surface S, yielding the difference in a surface integral on S using proper metric $d(\cdot,\cdot)$ as follows.

$$D(M_R, M_C) = \frac{1}{A}\iint_S d(x^R, x^C)dS\ , \qquad (1)$$

where A is the surface area of M_R, x^R is a test point on M_R, and x^C is its corresponding point on M_C, as shown in Figure 1.

A numerical approximation of (1) can be obtained by sampling a finite set of point correspondences, $\{(x_i^R, x_i^C)\}$. For example, each vertex of M_R can be a sample point in a mesh model. Let N denote the number of the point correspondences, then (1) can be approximated by a discrete form as

$$D(M_R, M_C) \approx \frac{1}{N}\sum_{k=0}^{N-1} d(x_k^R, x_k^C). \qquad (2)$$

In order to evaluate the distance $d(x_k^R, x_k^C)$ between x_k^R and x_k^C, we can simply select appropriate $d(\cdot\,,\cdot)$ among many distance measures including the signed distance, the absolute distance, and the squared distance.

Then, the only remaining problem is how to find the corresponding point x_k^C, *i.e.,* for each $x_k^R, k = 0, \ldots, N-1$. Usually, the correspondence problem can be solved with reasonable accuracy by simply determining the nearest

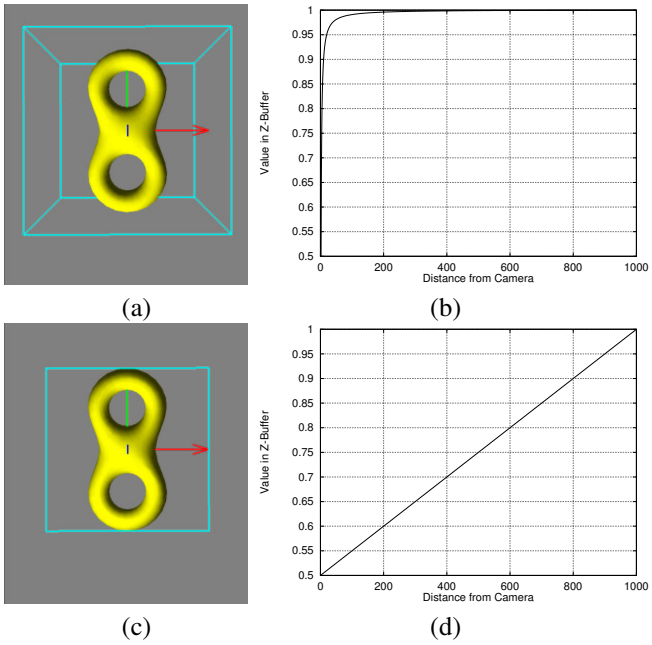


Figure 2. Camera models and their property of depth evaluation in OpenGL. (a) A perspective camera. (b) Depth change in a perspective camera. (c) An orthographic camera. (d) Depth change in an orthographic camera.

point from each other if the models consist of dense point cloud or mesh. An improved method is to find x_k^C by the intersection of a perpendicular line through x_k^R and M_C as shown in Figure 1. Note that this provides more accurate result but is computationally more difficult. In this paper, the distance measure is defined as the latter case. Note that this distance is not a metric, since it is not symmetric.

3 Virtual Camera and Z-Buffer

In this section, related issues on 3-D graphics are addressed, especially on virtual camera and Z-buffer. They are used efficiently in the proposed method for measuring shape dissimilarity, which is described in detail in Section 4.

The most popular camera model used in computer vision and graphics is a perspective projection model. A special case of the perspective camera model is an orthographic camera, in which the focal length is infinite. An example of both models is shown in Figure 2 (a) and (c), which is implemented using OpenGL [14]. As will be described later, since we are considering just one point in the scene which is positioned on the image center, both cameras can be used in our approach.

The Z-buffer, or depth buffer, is commonly used in 3-D graphics to remove hidden surfaces. It is a memory array, in which each element contains the distance from camera

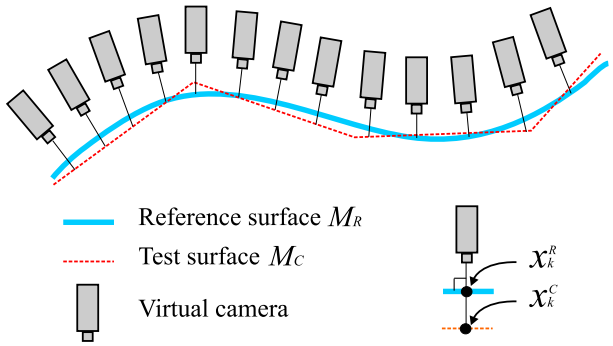


Figure 3. Surface roving method with a virtual camera.

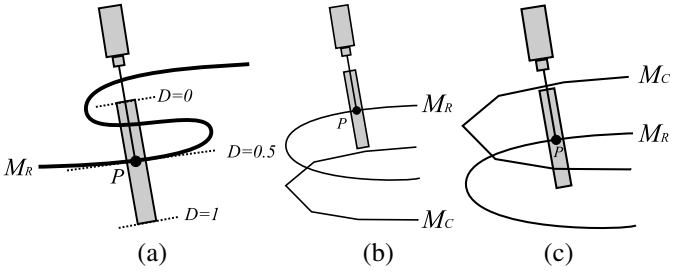


Figure 4. 2-D examples of degenerate case with geometric ambiguity. (a) Self-occlusion. (b) No detection (No hit in the view volume). (c) False alarm (Wrong hit in the view volume).

to the object surface drawn at a specific pixel position. The value in Z-buffer is increasing monotonically as the distance increases.

In OpenGL, there is an important difference in depth buffering between the two camera models. In a perspective camera, the value in Z-buffer is not linear. That is, the depth value in the Z-buffer is not linearly increasing as the distance from the camera to the object increases. This behavior is well illustrated in Figure 2 (b), in which depth values are retrieved and plotted while the distance from viewpoint and the target object is increasing. On the other hand, it is linear in an orthographic camera, as shown in Figure 2 (d). Since the Z-buffer value is going to be used for finding the actual depth in Z-direction by simple scaling, we prefer the linear property in depth buffer values according to the Z position. Therefore, in this paper, the orthographic projection is adopted. Note that the orthographic projection also simplifies the mathematical computation involved, yielding fast measuring process of shape dissimilarity.

4 Measuring Shape Dissimilarity

In this section, a novel method to evaluate the dissimilarity in (2) using a virtual camera and Z-buffer called surface roving technique is presented. And then a fast and asymptotic version of the surface roving method based on the view sampling technique is described as well.

4.1 Surface Roving Method

Let $\mathbf{X_s} = \{X_0, X_1, \cdots, X_{n-1}\}$ be the set of sample points on M_R, on which the approximation error is to be measured. In the proposed approach, a virtual camera is set to 'observe' X_k, moving around for $k = 0, 1, \cdots, n-1$ to observe the whole surface. The optical axis of the camera is aligned with the opposite normal direction of each sample point. Based on this configuration, it is obvious that x_k^R and the corresponding point x_k^C are projected onto the same position in the image coordinates. Note that our objective is to measure the distance from a sample point x_k^R to M_C along the perpendicular line through it as described in Section 2. Based on this observation, the distance from x_k^R to x_k^C can be evaluated without explicitly finding x_k^C, because the distance from the camera to x_k^R and to x_k^C is recorded in the Z-buffer when M_R and M_C is drawn, respectively. We can retrieve those corresponding depth values in the Z-buffer and simply compute the difference, yielding both signed and unsigned distance from x_k^R to x_k^C. This procedure is called 'surface roving', since the virtual camera visits all the test points over the reference model just like a satellite roving above the earth, as shown in Figure 3.

4.2 Removing Geometric Ambiguity

Generally, surface roving method works well when M_R is locally smooth enough and M_C approximates M_R to some extent. However, for the completion of the argument, a few degenerate cases with geometric ambiguity should be addressed.

Consider the degenerate cases shown in Figure 4 (a)∼(c), which are most likely in real application. 2-D curves imply the intersection of 3-D surface and a normal plane. Figure 4 (a) shows self-occlusion case of M_R, in which other part of M_R hides the test point P from the virtual camera. Since the viewing parameters of the virtual camera can be adjusted easily, the consequent view volume can also be constructed properly. Therefore, if we choose the view volume of the virtual camera as a rectangular parallelepiped centered at the test point and aligned with the optical axis, the depth value D of the test point would be always 0.5, regardless of the occlusion. We don't need to render the scene in order to compute the depth and therefore occlusion does not influence the depth value at all. In our implementation, the view volume of each virtual camera is built in the same way such that the Z-buffer evaluation is necessary only for M_C.

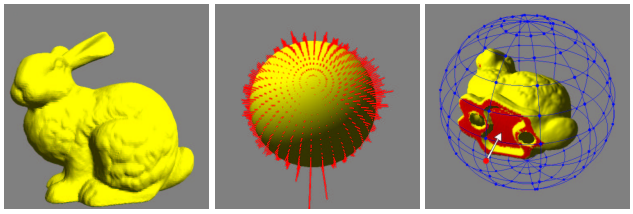
Other two cases occur when the approximation performance is very poor. Note that, they seldom occur in our application, since our concern is a guided multiresolution modeling and minor shape distortion by geometry compression. In case of no detection as shown in Figure 4 (b), we provide a predetermined MAX_ERROR value to the test point. A reasonable value of MAX_ERROR could be the half-height of the view volume.

Figure 4 (c) is the worst case. In this case, since the corresponding point is determined incorrectly, the measured depth difference becomes small even though the actual dissimilarity is large. Note that, any other distance measure cannot determine the exact difference in this case. Therefore, a high-level shape matching algorithm is required in order to find the exact correspondence, which is another research topic in computer vision.

4.3 Cooperative Method: Accelerating Surface Roving by Viewpoint Sampling

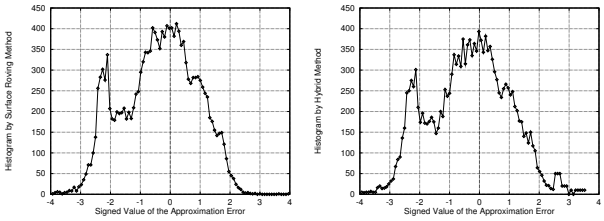
Although the proposed surface roving method is quite accurate, we have to 'observe' tremendous number of sample points where the difference is going to be measured. This might cause redundant operations especially for those sample points with similar normal direction. However, note that since an orthographic camera model assumes parallel rays, by employing it, we don't need to visit all those sample points individually. For example, for the Bunny model shown in Figure 5 (a), most of sample points at the bottom part have similar surface orientations, which is marked in red as shown in Figure 5 (c). Therefore, if we construct an Extended Gaussian Image (EGI) of the model, a sharp spike should be existing on the corresponding point on EGI. This is illustrated in Figure 5 (b). In this case, surface roving for those points can be replaced by only one orthographic projection from the viewpoint shown as a red point in Figure 5 (c), reducing the number of observations significantly compared with the original surface roving. Thus, with a proper selection of viewpoint samples, it is possible to evaluate an asymptotic measurement of the difference of models using the surface roving technique.

One problem in using viewpoint sample is that there could be self-occlusion when a object is seen from a view sample. There may be unobservable part of the object, especially when the object is not convex. For example, the long ears of the bunny model may occlude the main body. Thus, for those sample points on the occluded region, the viewpoint sampling method does not work. In order to solve this problem, cooperative method is employed in our approach, in which the surface roving is applied to the occluded region, while the viewpoint sampling is used for the unoccluded region, respectively. Note that whether a sample point is occluded or not can be determined easily by comparing the depth difference with prespecified threshold. In general, if the shape of the object is not too much complex, most of the surface region is observable from one of the viewpoint samples.



(a) (b) (c)

Figure 5. Viewpoint sampling method. (a) Bunny model. (b) Extended Gaussian Image (EGI) of the Bunny model. (c) View samples and a specific mapping.



(a) (b)

Figure 6. Histogram of the approximation error between the Bunny mesh models. (a) Result of applying exact measure by surface roving (Original surface roving). (b) Result of applying asymptotic measure by viewpoint sampling plus surface roving (Cooperative method).

Note that although the cooperative method is actually an asymptotic implementation of the original surface roving using (2), it provides almost the same performance of the exact measure. For instance, for the mesh models in Figure 5 (a) and Figure 8 (a), the shape distortions are measured using both methods. The histograms of the measured error are shown in Figure 6, in which both distributions are almost similar, while the number of observations required for the cooperative method is reduced to only 5% of that of the original surface roving.

The reference model can be any of point cloud, polygonal mesh, parametric surface, and voxel model, as far as the normal direction at a specific test point can be determined. The test model can also be polygonal mesh, parametric surface, and voxel model. However, since point cloud has holes and does not record the depth on it, the proposed method is not applicable in this case.

Especially, if the reference model is represented by voxel, the number of view samples reduces to only six, since

(a) (b) (c)

Figure 7. 3-D reference models. (a) Teeth model with 69,451 triangles. (b) Rocker Arm model with 116,602 triangles. (c) Venus model with 134,342 triangles.

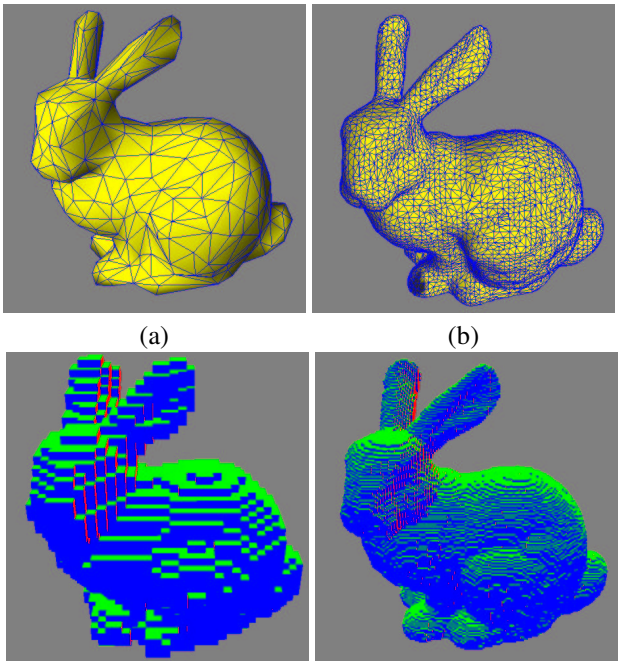
each voxel is actually a cube with six faces. Thus, in this case, the measurement process could be extremely fast.

5 Experimental Results

In this section, we present the experimental results of measuring the shape dissimilarity using the proposed cooperative surface roving method. First, it is investigated how much the viewpoint sampling reduces the number of observations. For the reference models shown in Figure 7, the numbers of observations required in both the original surface roving and the cooperative methods are counted for a few resolution of viewpoint sampling. We summarize the result in Table 1, for various horizontal and vertical resolution, H and V, respectively. As shown in Table 1, the number of observations is reduced significantly compared with that of the original surface roving method, ranging from 10% (Rocker Arm model) to 0.65% (Venus model) of the original ones. Viewpoint sampling replaces the surface roving if the test point is visible from a sample viewpoint in its normal orthographic direction. In case of Rocker Arm model, since the interior wall around the hole is not visible, surface roving should be employed for those area. On the contrary, Venus model is roughly sphere-like such that most of surface region is visible from the corresponding view sample. Therefore, the number of observations can be reduced significantly for the Venus model than for the Rocker Arm model.

In order to show the efficacy of the proposed technique, experiments are carried out on four different representations of the Bunny model. The reference model is dense mesh with 69,451 triangles shown in Figure 5 (a), and the test models are simplified meshes with different resolutions (800 and 10,459 triangles, respectively), rough voxel ($32\times32\times32$) and fine voxel ($128\times128\times128$), as shown in Figure 8.

Distances from the reference to the test models as well as the execution times on a 933MHz Pentium III CPU are evaluated by using the proposed method, and shown in Ta-



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

Figure 8. Different representations of the Bunny model. (a) Simplified mesh at low resolution. (b) Simplified mesh at middle resolution. (c) Rough voxel. (d) Fine voxel.

Table 1. The number of observations for surface roving and cooperative methods.

Model Name	Surface Roving	Cooperative (H×V)		
		36×18	24×12	12×6
Bunny	69,451	3,422	3,286	5,396
Teeth	116,602	3,705	3,412	4,619
Rocker Arm	60,264	7,962	7,193	6,307
Venus	134,342	978	868	2,265

Table 2. Measured shape dissimilarity. (SiAD: Signed average distance, USiAD: Unsigned average distance, STD: Standard deviation, MAX: Maximum distance).

Model Type	Shape Dissimilarity			
	SiAD	USiAD	STD	MAX
Low-Resolution Mesh	-0.2085	0.6777	0.8562	6.2169
Middle-Resolution Mesh	0.0156	0.0549	0.2632	6.1912
Sparse Voxel	1.6348	1.6633	0.9922	5.1713
Dense Voxel	0.4436	0.4455	0.2249	4.9414

Table 3. Execution time on a 933MHz Pentium III CPU.

Model Type	Execution Time (sec)		
	Cooperative	Surface Roving	Vertex-to-Vertex [1][8]
Low-Resolution Mesh	27.6	87.5	≥ 3000
Middle-Resolution Mesh	64.9	1137.0	≥ 3000
Sparse Voxel	38.4	57.9	N/A
Dense Voxel	152.0	440.0	N/A

ble 2 and Table 3, respectively. It can be seen that the proposed approach is computationally efficient and measures the shape dissimilarity for several different types of 3-D models accurately. Compared with the original surface roving method and naive implementation of the conventional method used in [1][8], the cooperative algorithm reduces the execution time significantly. In this experiment, a total of 24×12 view samples are used, spaced 15 degrees in both latitudinal and longitudinal directions. Note that the execution time can be further decreased if the number of samples is reduced by selecting larger spacing. However, in this case, the measurement accuracy may decrease as well.

6 Conclusion

In this paper, we proposed an efficient method to evaluate the shape dissimilarity for different types of 3-D models. Unlike the conventional geometric methods, our approach called 'surface roving method' utilizes the Z-buffer and virtual camera commonly used in 3-D graphics to obtain the distance between corresponding points on different 3-D surface models. In order to make the surface roving faster, an asymptotic implementation of the surface roving method was developed, in which viewpoint sampling and simultaneous orthographic projection of virtual camera were used efficiently. Since the operation is performed on the geometry engine, it can be sped up by adopting a faster hardware accelerator. Experimental result shows the efficacy of the proposed approach, in which the shape dissimilarity is measured for two popular types of 3-D models, including mesh and voxel model, rapidly and accurately.

References

[1] J. Rossignac and P. Borrel, "Multi-Resolution 3D approximations for rendering," *Modeling in Computer Graphcis,* pp. 455-465, Springer-Verlag 1993.

[2] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," *Proc. of SIGGRAPH '93*, pp. 19-26, August 1993.

[3] R. Ronfard and J. Rossignac, "Full-range approximation of triangulated polyhedra," *Computer Graphics Forum,* vol. 15, no. 3, pp. 67-76, August 1996.

[4] R. Klein, G. Liebich and W. Strasser, "Mesh reduction with error control," *Proc. of IEEE Visualization Conference*, October 1996.

[5] H. Hoppe, "Progressive meshes," *Proc. of SIGGRAPH '96*, pp. 99-108, August 1996.

[6] J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, F. Brooks, and W. Wright, "Simplification envelopes," *Proc. of SIGGRAPH '96*, pp. 119-128, August 1996.

[7] J. Cohen, D. Manocha, M. Olano , "Simplifying polygonal models using successive mappings," *Proc. of IEEE Visualization Conference*, pp. 395-402, October 1997.

[8] D. Luebke and C. Erikson, "View-dependent simplification of arbitrary polygonal environments," *Proc. of SIGGRAPH '97*, pp. 199-208, August 1997.

[9] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," *Proc. of SIGGRAPH '97*, pp. 209-216, August 1997.

[10] A. Guéziec, "Locally toleranced surface simplification," *IEEE Trans. on Visualization and Computer Graphics,* vol. 5, no. 2, pp. 168-189, April-June 1999.

[11] I. K. Park, I. D. Yun, and S. U. Lee, "Automatic 3-D model synthesis from measured range data," *IEEE Trans. on Circuits and Systems for Video Technology,* vol 10, no. 2, pp. 293-301, March 2000.

[12] C. S. Kim and S. U. Lee, "Compact encoding of 3D voxel surfaces based on pattern code representation," accepted for publication in *IEEE Trans. on Image Processing*.

[13] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice,* Addison Wesley, 1992.

[14] M. Woo, J. Neider, T. Davis, and D. Shreiner, *OpenGL Programming Guide,* Addison Wesley, 1999.