DOI: 10.1007/s12541-010-0031-2

Rapid Surface Reconstruction from a Point Cloud Using the Least-Squares Projection

Dong-Jin Yoo[1,#]

1 Department of Computer Aided Mechanical Design Engineering, Daejin University, Pocheon-Si, Kyeonggi-Do, South Korea, 487-711
Corresponding Author / E-mail: djyoo@daejin.ac.kr, TEL: +82-31-539-2031, FAX: +82-31-539-1970

KEYWORDS: Point clouds, Surface reconstruction, Distance field, Least-squares projection, Mesh smoothing, Mesh refinement

A new approach for the rapid and robust surface reconstruction from a point cloud is presented based on the distance field and the least-squares projection (LSP) algorithm. This novel approach works directly on the point cloud without any explicit or implicit surface reconstruction procedure. First, a coarse base polygonal model was created directly from the distance field for the given point cloud through the iso-surface extraction. After acquiring a rough base polygonal model, we obtain a quality polygonal model through the iterative refinement and least-squares projection which projects current working polygonal model onto the point cloud in a least-squares sense. The main contribution of this work is the robust and fast surface reconstruction from randomly scattered 3D points only without any further information. We demonstrate the validity and efficiency of this new approach through a number of application examples.

Manuscript received: July 16, 2008 / Accepted: June 2, 2009

1. Introduction

With the remarkable development of high precision 3D scanners, we can easily obtain large and complex object models composed of point clouds sampled from the real-world objects. In recent years point clouds have gained increasing attention as an alternative surface representation. Various point-based techniques including point rendering, parameterization, simplification, smoothing, thinning or denoising, registration, and shape reconstruction become an active research area in Computer Aided Design (CAD), Computer Graphics (CG), and Reverse Engineering (RE).[1-4] In particular considerable works have been done to reconstruct surfaces from the point clouds. The main goal of this work is also to develop a novel method for surface reconstruction from unstructured point clouds.

There is a great deal of literature on the reconstruction of surfaces from a point cloud data. Many methods regarding the surface reconstruction from the scattered point set have been proposed in various ways, such as Delaunay tetrahedralization, the level set method, and the implicit surface interpolation method based on RBF (Radial Basis Function) or CSRBF (Compactly Supported Radial Basis Function). Methods based on Delaunay tetrahedralization[5] form surfaces by directly connecting the point set. However, interpolation may not be appropriate for noisy data, and may fail if sample noise approaches sample density. The level set method[6-7] evolves a surface over time until it approximates the given point cloud. This method involves defining a speed function which attracts the level set surface to the data points. The level set method is highly flexible and it is possible to include confidence measures and smoothing terms in the speed function. However, its implementation becomes expensive in time and memory if high accuracy reconstruction is required. The implicit surface reconstruction methods are attractive and become popular because they allow a complex shape to be interpolated by one formula. The main advantages of using implicits for surface reconstruction from scattered point sets are data repairing capabilities and opportunities to edit the resulting objects using standard implicit modeling operations. Finding a set of basis functions which forms an implicit surface reconstruction of a point set, is a difficult problem which has attracted a great deal of attention. Muraki[8] uses a linear combination of Gaussian blobs to fit an implicit surface to a point set. Unfortunately, the method was fairly slow since both blobs position and other parameters had to be inferred. More recently radial basis function (RBF) have been introduced to reconstruct an implicit surface from a point cloud. A linear combination of radial basis functions is found such that the zero level set interpolates or approximates the input points. The combination is found by solving a large linear system. One problem with RBF based methods is the

 Springer

need to place points inside and/or outside the object in order to have non-zero points for the function to interpolate. Another issue is the fact that the linear system is both large and dense. Carr et al.[9] attempted to apply an implicit surface to various types of large point clouds using the fast multipole method in order to improve performance of traditional RBF method. This approach, however, is not simple to implement due to a somewhat complex mathematical algorithm and the enormous amount of computation time required in treating large matrices of the linear system. To overcome this problem, Kojekine et al.[10] proposed compactly supported RBFs to reconstruct smooth surface from a point cloud data. In the method the large coefficient matrix of the linear system was transformed into a band-diagonal sparse matrix that could be solved more efficiently. However, the method was not robust for non-uniform distributions of points. Ohtake et al.[11] suggested a novel MPU (Multi-level Partition of Unity) implicit approach that reconstructs an implicit surface from unorganized data sets containing a huge number of points using weighted sums of different types of piecewise quadratic functions. Yoo[12] proposed a novel method to reconstruct a complete polygonal model from a raw incomplete polygonal model with many holes and overlapping triangles using an implicit surface scheme based on RBFs and the domain decomposition method. In the method the 3D spatial domain occupied by the given polygonal model was divided into several sub-domains. Local solutions were then obtained by interpolating points allocated in each sub-domain separately. After calculating the local solutions, they were blended together using a smooth blending function, forming a partition of unity to obtain a global solution as in Ohtake's method. Although great improvements have been made, radial basis functions remain relatively costly in time and memory. To overcome the difficulties, we present a novel method of reconstructing surfaces from a point cloud. The method is not using any explicit or implicit surface reconstruction procedure to the given point cloud. In some sense, this new method can be considered as an extension of the least-squares projection (LSP) algorithm introduced by earlier researchers. Azariadis[4] proposed a method for finding a parameterization of an unorganized point cloud using the point directed projection (DP) onto the point cloud along an associated projection vector. His method operates directly on the point cloud without any explicit or implicit surface reconstruction procedure. It has been applied to curve-drawing onto point clouds for point-based modeling, where the projection vectors are specified through a graphics interface tool. Liu et al.[13] applied the DP algorithm for projecting points onto a point cloud. They called the method a least-squares projection (LSP) algorithm. In the method they determined automatically the projection direction vectors using a newly proposed linear optimization method.

The major contribution of our work is to extend the LSP algorithm for reconstructing surfaces from a point cloud without any extra information except for the geometric position of the point set. First, a coarse base polygonal model is created by extracting the iso-surface from the distance field for the point cloud. Then a quality polygonal model is obtained through the iterative refinement and least-squares projection which projects the coarse base polygonal model onto the given point cloud in a least-squares sense. The proposed method has two advantages due to the combination of the distance field and LSP algorithm. The first advantage is that no expensive surface reconstruction procedure such as implicit or explicit surface reconstruction is needed for generating the coarse base polygonal model, therefore it is effective in computing time even when the number of points is gigantic. The second advantage is that a quality polygonal model is obtained through a simple and robust mesh refinement and LSP algorithm, so it does not require the expenditure of large amounts of time and memory.

The remainder of this paper is organized as following. In Section 2, the procedure for the distance field calculation and iso-surface extraction is described. In Section 3, an iterative procedure for the mesh refinement and LSP algorithm is presented. In Section 4, some numerical examples for the surface reconstruction are presented. We conclude the paper with some discussions and ideas for future work in Section 5.

2. Distance field calculation and iso-surface extraction from a point cloud data

2.1 Point normal estimation

In this study, since a model is composed of point clouds without any extra information, we must estimate the normal vector in each point to calculate the signed distance field for generating a coarse base polygonal model. The normal vectors can be estimated by analyzing the local neighborhood of each sample point. Because there is no connectivity information available, these local neighborhoods are usually constructed using k-nearest neighborhoods.[14,15] If the points satisfy certain sampling criteria, like adaptation to the local feature size, then the neighborhood estimate is guaranteed to be reliable.

Let $\mathbf{V}_0$ be a sample point and $\{\mathbf{V}_1,\ldots,\mathbf{V}_k\}$ its k-nearest neighbors. The covariance matrix C(**P**) is defined as follows:[14]

$$C(\mathbf{P}) = \sum_{i=0}^{k} (\mathbf{V}_i - \mathbf{C})(\mathbf{V}_i - \mathbf{C})^T \in \mathbf{R}^{3x3}, \quad (1)$$

where **C** is the center of gravity of k-nearest neighborhoods. The eigenvector corresponding to the smallest eigenvalue gives an estimate for the normal direction. In addition, one of the most difficult problems in normal vector estimation is the establishment of a consistent inner and outer orientation, as shown in Fig. 1.

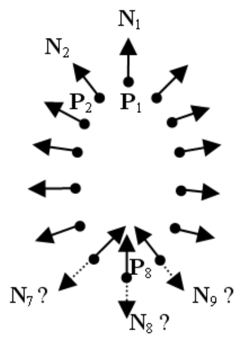


Fig. 1 Ambiguous normal orientation

Ou et al.[16] compare several algorithms for establishing the inner and outer directions. In this study, we only use a simple strategy similar to Hoppe et al.'s method,[14] in which the normal direction is propagated from an initial normal, as shown in Fig. 2. To assign an initial normal, the unit normal of the point which has the largest x coordinates is forced to point toward the +x axis.

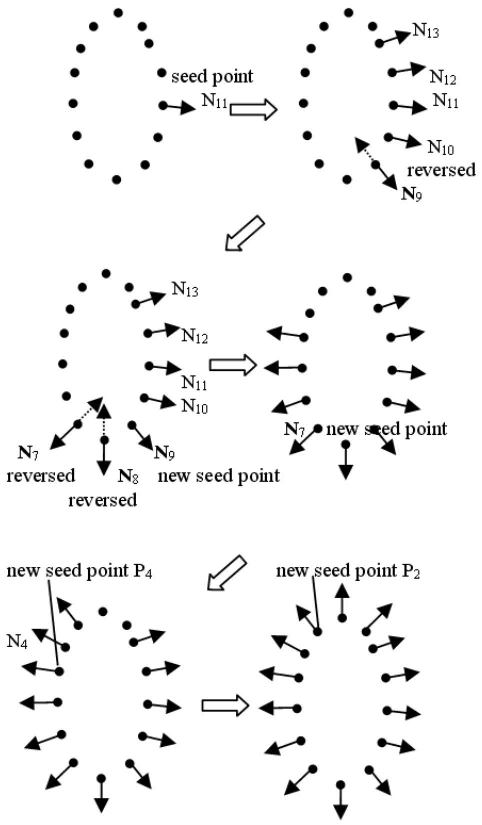


Fig. 2 An illustrating example of normal direction alignment

2.2 Distance field calculation for the given point cloud

After estimating the point normal, the distance field for the given point sets can be calculated easily. The distance field is an effective representation of shape. Traditionally, distance fields are defined as a scalar field of distances to a shape.[17-21, 28] Each element in a distance field specifies its minimum distance to the shape. Positive and negative distances are used to distinguish outside and inside of the shape, e. g., using negative values on the outside and positive on the inside, as shown in Fig. 3.

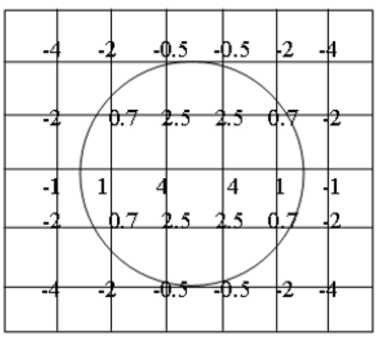


Fig. 3 Schematic diagram illustrating the concept of distance fields

Since the main purpose for the distance field calculation is to generate a rough base polygons, the spatial domain occupied by the point cloud is divided into a small number of voxels, typically 16 x 16 x 16 – 64 x 64 x 64 resolutions, according to the geometrical complexity of the given point cloud. The computation of the distance field is done by assigning a signed minimum distance value to each voxel grid point, as shown in Fig. 4.

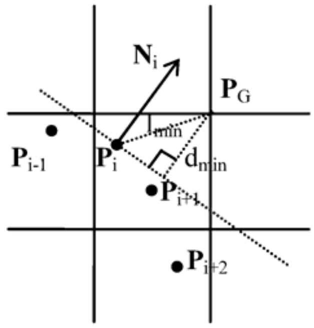


Fig. 4 Calculation of the signed minimum distance value for a voxel grid point $\mathbf{P}_G$

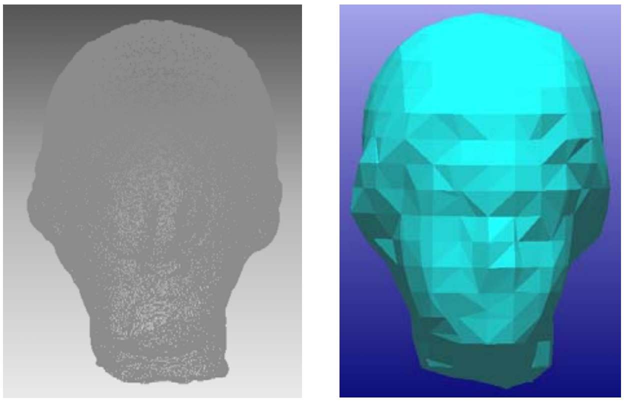
The minimum perpendicular distance d_{min} between a voxel grid point $\mathbf{P}_G$ and a point in the point cloud $\mathbf{P}_i$ is calculated as follows:

$$d_{min} = (\mathbf{P}_G - \mathbf{P}_i) \cdot \mathbf{N}_i, \tag{2}$$

where $\mathbf{N}_i$ is the normal vector at a point $\mathbf{P}_i$. The distance value for the particular voxel grid point can be determined by picking the closest point based on l_{min} denoted in Fig. 4, whereas the actual distance value used is d_{min} calculated by using equation (2). Of course the distance field calculated in this way can be considered as an approximated one. However, since the more detailed surface will be obtained through the iterative mesh refinement and LSP algorithm, it can be used successfully to generate the coarse base polygonal model.

2.3 Generating a coarse base polygonal model

Various methods can be used to generate a polygonal model from the distance fields calculated in previous section. In this paper, the well-known marching cube algorithm[12] was used to extract a polygonal model from the distance fields. The 3D space occupied by the point set is divided into regular cells such as cubes. If the distance value takes on a mixture of positive and negative values at a corner of a given cube, then the surface must pass through the cube. At such cubes, a small set of polygons can be created that approximate the shape of the surface within the cube. A base mesh generated in this way is illustrated in Fig. 5. A coarse base polygonal model can be also obtained in voxel form, as shown in Fig. 6. This type of mesh can be obtained more easily than the one obtained by the marching cube algorithm which requires well defined look-up table for various cases, so the implementation for generating this type of voxel mesh is very simple and easy compared to the marching cube algorithm. Through our numerical experiments, it will be shown that these two extraction methods can be applied effectively to generate a coarse base mesh for the refinement and least-squares projection.



(a) Point cloud data (b) A base triangular mesh

Fig. 5 Generating a base mesh using the marching cube algorithm

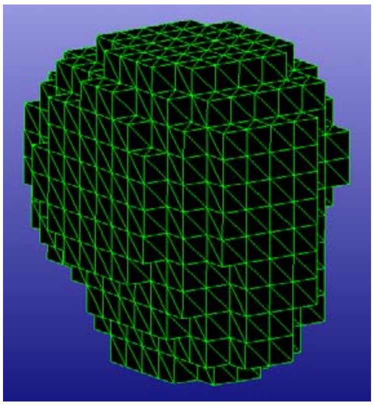
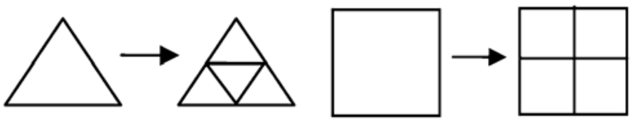


Fig. 6 A base mesh in voxel form

3. Fine surface generation using mesh refinement and LSP

As described in previous section, a coarse base model was generated in various forms including triangle meshes, rectangle meshes, and voxel meshes. Then each element was refined into four elements, as shown in Fig. 7. Finally, a quality polygonal model was obtained by smoothing the current working polygonal model and projecting it onto the given point cloud by LSP algorithm.



(a) Triangle meshes (b) Rectangle meshes

Fig. 7 Refinement of meshes

For the smoothing, a node was relocated to the averaged center of gravity of the neighboring polygons to improve the quality of the polygonal model as follows:[12]

$$\mathbf{P} = \frac{\sum_{i=1}^{n} A_i \mathbf{C}_i}{\sum_{i=1}^{n} A_i}, \tag{3}$$

where $\mathbf{P}$ is the position vector of the new location of a node, A_i is the area of the i^{th} element, $\mathbf{C}_i$ is the center of gravity of the i^{th} element, and n is the number of elements connected with the current node. After relocating all the nodes of current working polygonal model, relocated nodes were projected onto the point cloud. Azariadis[4] proposed an appropriate point cloud error function and used it to solve the problem of point projection onto a point cloud along a projection direction. Liu et al.[13] extended Azariadis's method to the LSP algorithm by determining automatically the projection direction. We review the detail of their works. Consider a point cloud C_N and a test point $\mathbf{P}$=(x, y, z) with an associated projection vector $\mathbf{n}_P$=(n_x, n_y, n_z). Each $\mathbf{P}_i$ is associated to a positive weight α_i. Let $\mathbf{P}^*$ be the projection point of $\mathbf{P}$, as shown in Fig. 8.

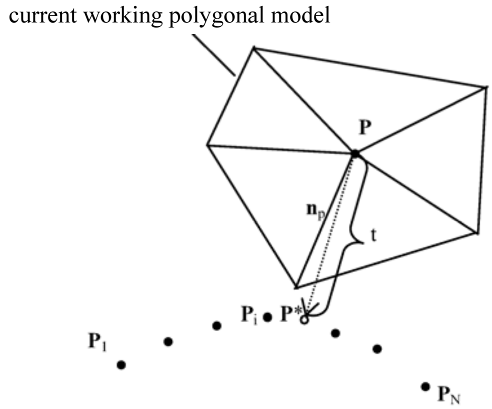


Fig. 8 Schematic diagram illustrating the projection of a point onto point sets

Then the projection problem can be considered as an optimization problem to find $\mathbf{P}^*$ by minimizing the following weighted sum of the squared distances:

$$E(\mathbf{P}^*) = \sum_{i=1}^{N} \alpha_i \left\| \mathbf{P}^* - \mathbf{P}_i \right\|^2, \tag{4}$$

where N is the number of points of a point cloud C_N. For given weights α_i, the projection point $\mathbf{P}^*$ can be described as follows:

$$\mathbf{P}^* = \mathbf{P} + t\mathbf{n}_p, \tag{5}$$

where t is the projection distance.

By substituting equation (5) into equation (4), the solution of minimizing equation (4) can be determined as follows:[13]

$$t = \frac{\beta - \mathbf{P} \cdot \mathbf{n}_p}{\left\| \mathbf{n}_p \right\|^2} \tag{6a}$$

$$\beta = \frac{\mathbf{C} \cdot \mathbf{n}_p}{C_0} \tag{6b}$$

$$C_0 = \sum_{i=1}^{N} \alpha_i, \quad C_1 = \sum_{i=1}^{N} \alpha_i x_i, \quad C_2 = \sum_{i=1}^{N} \alpha_i y_i, \quad C_3 = \sum_{i=1}^{N} \alpha_i z_i, \tag{6c}$$

where denotes the dot product and $\mathbf{C}$=(C_1, C_2, C_3). The weights α_i play a important role in the computation of $\mathbf{P}^*$, so they should be chosen carefully. In general, the weights α_i of $\mathbf{P}_i \in C_N$ should take a larger value when $\mathbf{P}_i$ is closer to the test point $\mathbf{P}$, and a descending value as the distance from $\mathbf{P}_i$ to $\mathbf{P}$ increases. In this work, we used one weight function which only takes into account the distance between $\mathbf{P}_i$ and $\mathbf{P}$ as follows:[13]

$$\alpha_i = \frac{1}{1+\|\mathbf{P}-\mathbf{P}_i\|^4}. \tag{7}$$

Liu et al.[13] determined the projection direction $\mathbf{n}_p$ by minimizing the projection distance t under the assumption that t is a function with respect to $\mathbf{n}_p$. They had to determine the projection direction iteratively in order to apply the point projection procedure to a number of application examples including thinning a point cloud, point normal estimation, projecting curves onto a point cloud and others. However, in this work, the projection direction can be calculated through a simple vector operation using the connectivity information of current working polygonal model as follows:

$$\mathbf{M}_p = \frac{\sum_{i=1}^{n} A_i \mathbf{M}_i}{\sum_{i=1}^{n} A_i}, \quad \mathbf{n}_p = \frac{\mathbf{M}_p}{\|\mathbf{M}_p\|} \tag{8}$$

where A_i is the area of the i[th] element, $\mathbf{M}_i$ is the normal vector of the i[th] element, and n is the number of elements connected with the current node. After determining the normal vector at each node using equation (8), all relocated nodes of current working polygonal model were projected onto the point cloud using equation(5). The refinement and projection combined with smoothing operation were repeated until the desired level of accuracy was attained. Figure 9 shows the whole process of reconstructing surfaces from a point cloud.

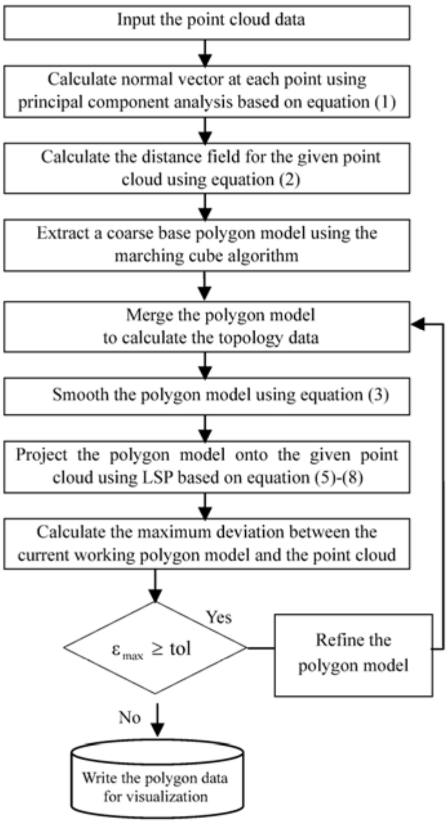
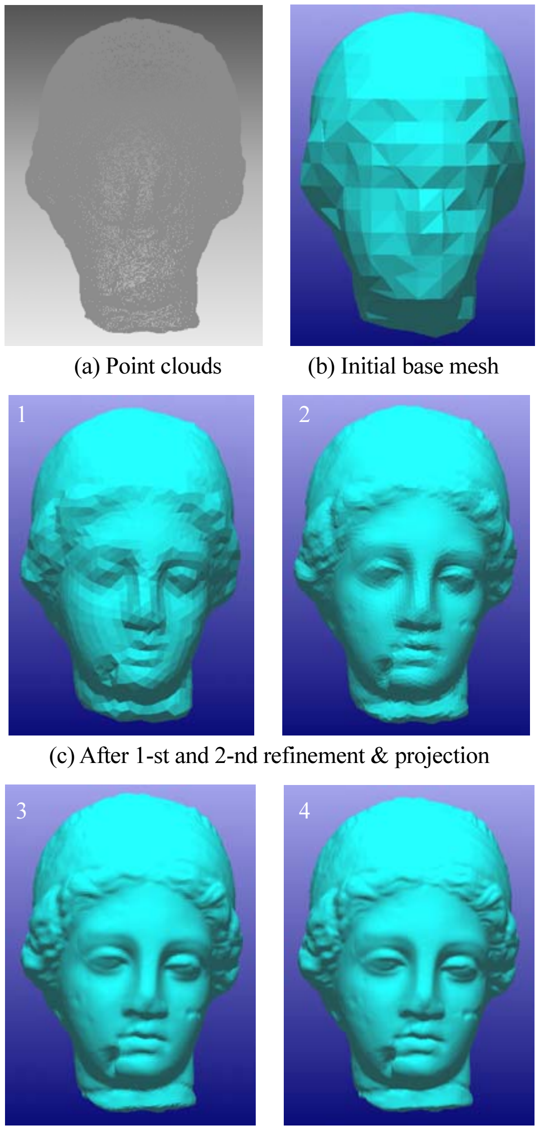


Fig. 9 Flowchart showing the procedure of surface reconstruction from a point cloud

4. Experimental results and Discussion

Various surface reconstructions were performed for large and complex point clouds with arbitrary shapes and topologies to verify the effectiveness and validity of the proposed surface reconstruction algorithm. The proposed algorithm has been implemented in C language on a 3 GHz Pentium IV computer with 512 MB memory.



(a) Point clouds (b) Initial base mesh

(c) After 1-st and 2-nd refinement & projection

(d) After 3-rd and 4-th refinement & projection

Fig. 10 Shape reconstruction of Igea model (I)

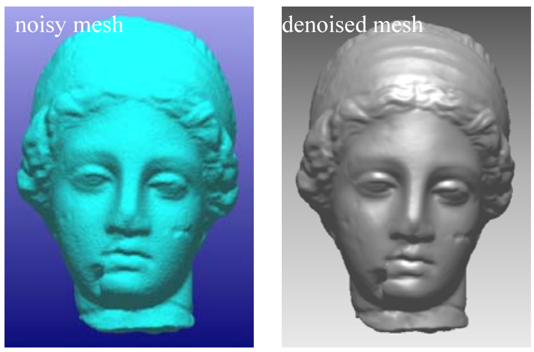
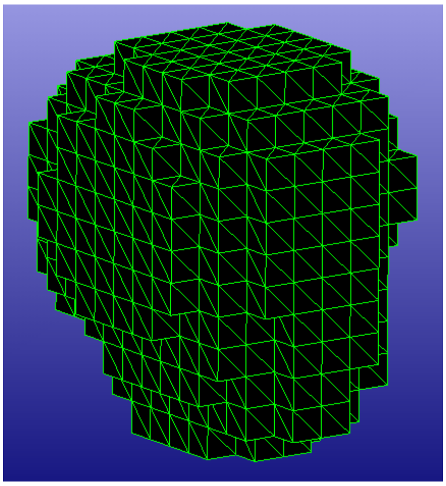
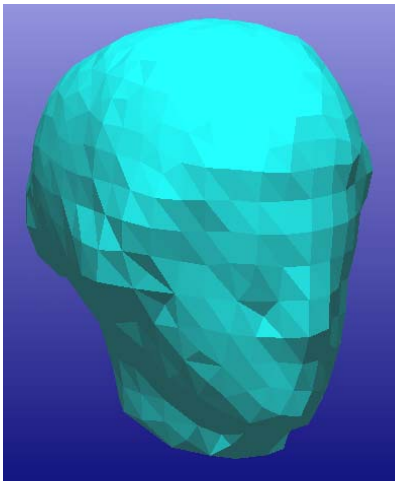


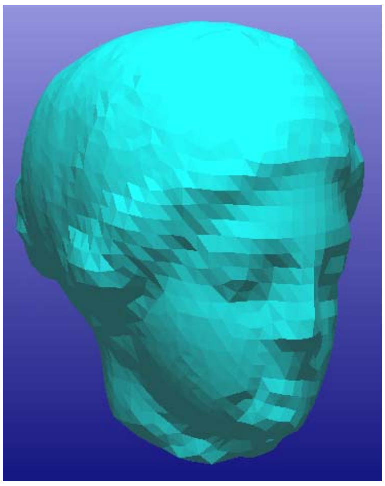
Fig. 11 Denoising noisy mesh



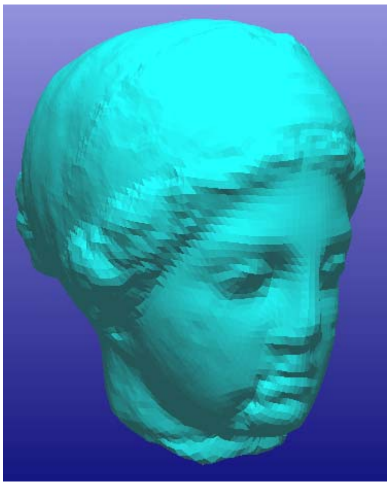
(a) Initial base mesh



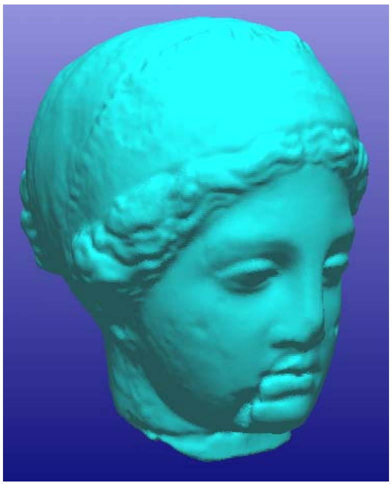
(b) After 1-st refinement & projection



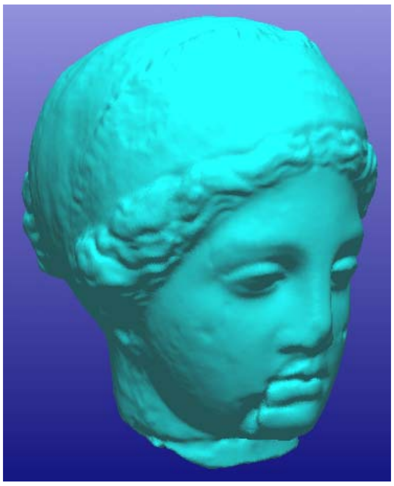
(c) After 2-nd refinement & projection



(d) After 3-rd refinement & projection



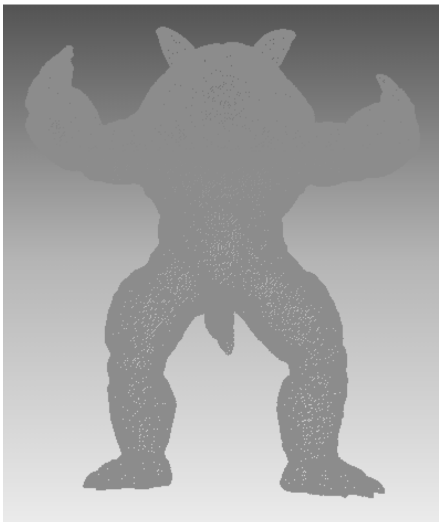
(e) After 4-th refinement & projection



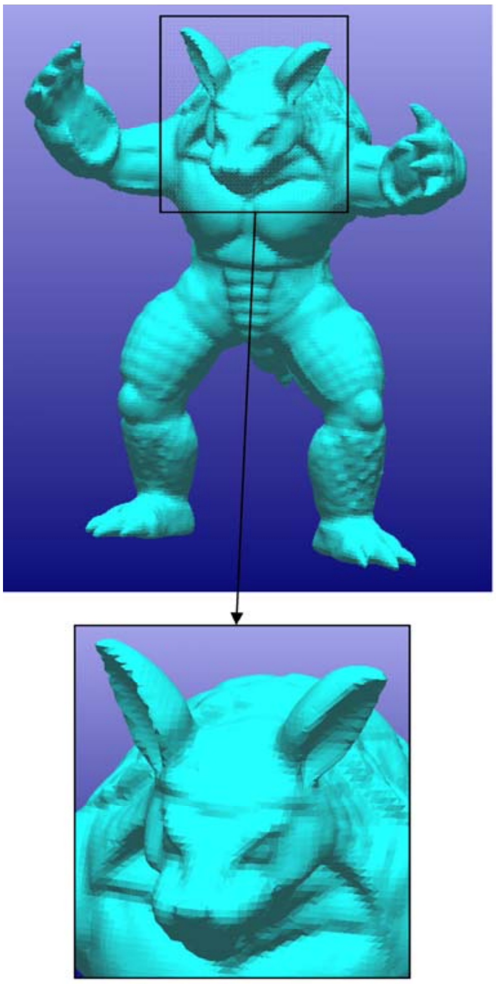
(f) After 5-th refinement & projection

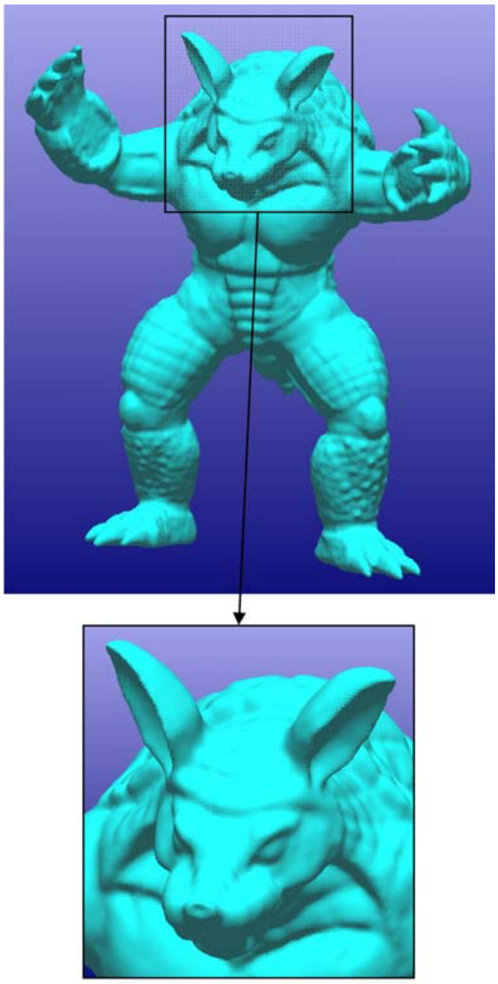
Fig. 12 Shape reconstruction of Igea model (II)

As seen in Fig. 10, the proposed approach facilitates the reconstruction of surfaces from a point cloud and gives a significant reduction in computing time and required memory, since the method does not require any explicit or implicit surface representation. A given point cloud might have noise as seen in the left figure of Fig. 11. The problem of noise can be handled by projecting the points onto the point cloud surface themselves. For this experiment noisy point clouds are produced by adding noise to each points of the Igea model of the right figure of Fig. 10(d) along the normals. The right figure of Fig. 11 shows the result of denoising the initial mesh with noise. The projections were repeated until the noise was completely removed. The result of this kind of projection procedure is a thin point cloud, and the procedure is often called the thinning operation in the literature. Figure 12 shows



(a) Point clouds

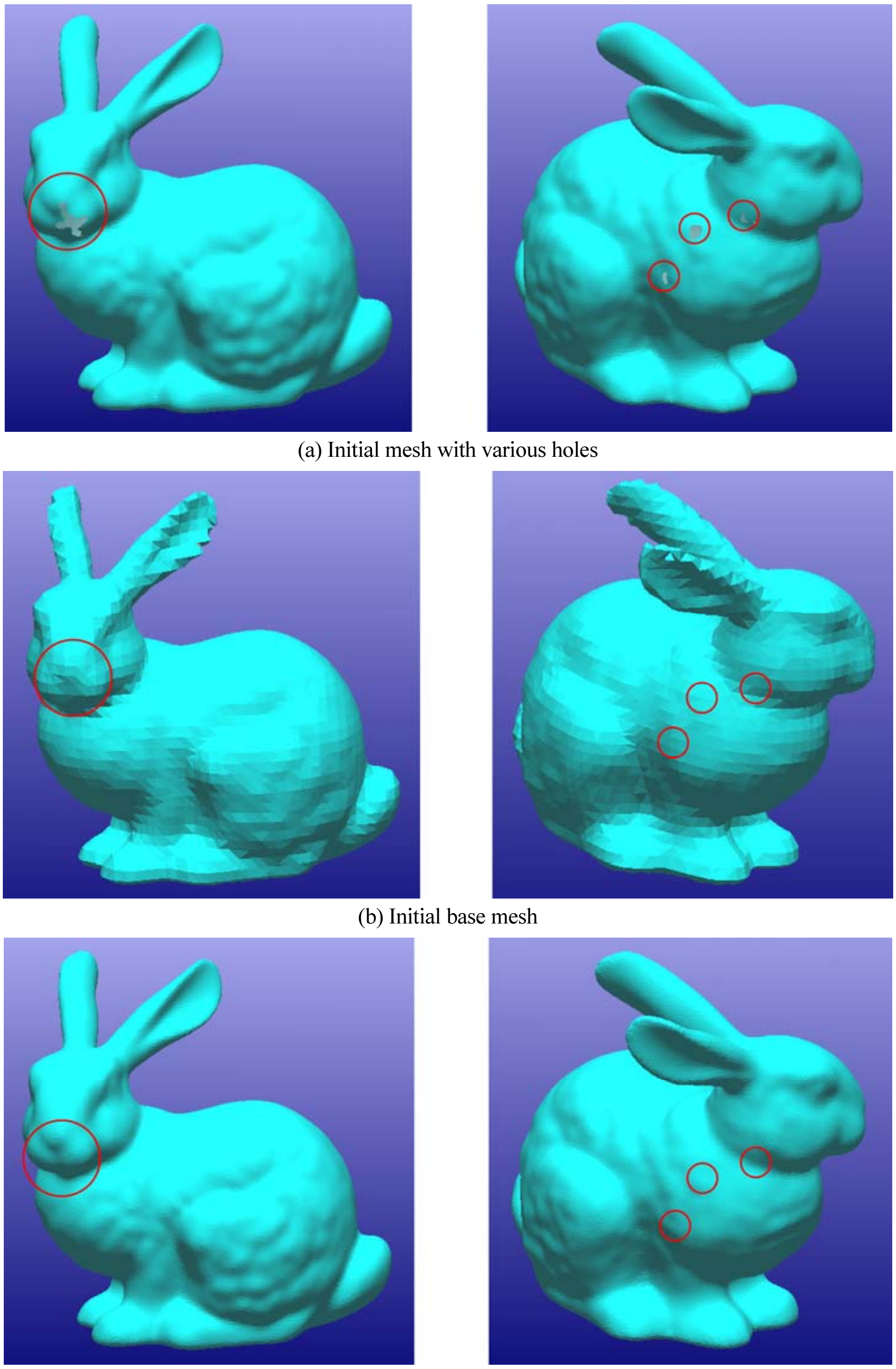




(b) Initial base mesh (c) After 1-st refinement & projection

Fig. 13 Shape reconstruction of Armadillo model

a surface reconstruction result from the Igea point cloud of Fig. 10(a). To test the effect of the type of initial base mesh, we construct the initial base mesh in voxel form composed of triangle meshes, as shown in Fig. 12(a). The accuracy and quality of the final reconstructed surface were almost same as those of the right figure of Fig. 10(d). The robustness of the proposed method was verified again using a more large point cloud. Figure 13 shows the surface reconstruction result of Armadillo model. The point cloud of Armadillo model was composed of 320,532 points. The initial base mesh consisting of 153,400 triangles was generated using the marching cube algorithm as described in Section 2.3. Since the initial base mesh was fairly faithful to the given point cloud, only one refinement and projection procedure was needed to get the fine surface of Fig. 13(c). Figure 14 shows the data repairing capability



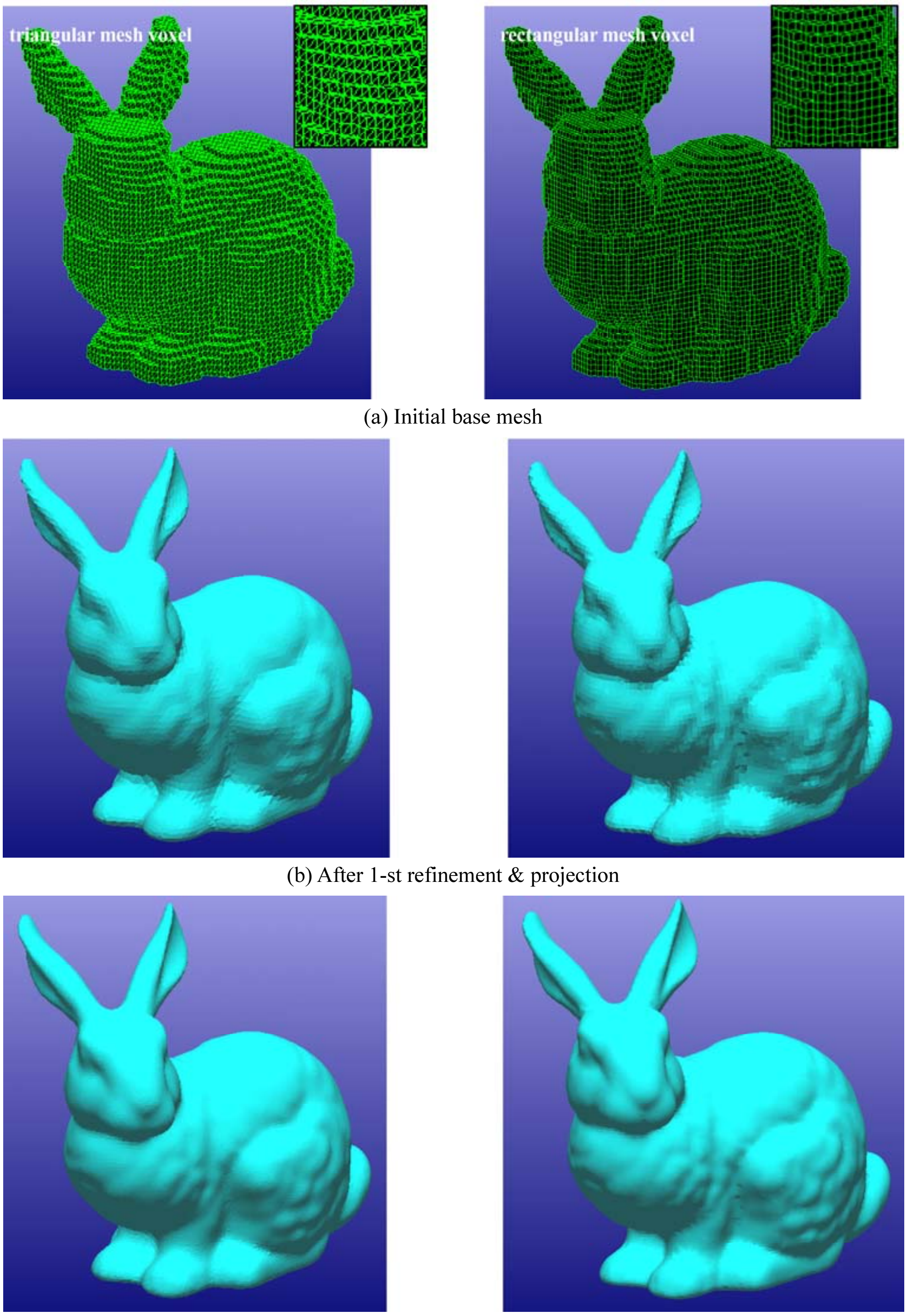
(a) Initial mesh with various holes

(b) Initial base mesh

(c) After refinement & projection

Fig. 14 Shape reconstruction of Stanford Bunny model (I)

of the proposed method. The many holes existed in the polygonal model were completely and automatically removed by the surface reconstruction procedure. In the case of a fairly dense point cloud, almost every voxel grid point near the point cloud can receive a well-defined distance value, so the initial base mesh generated from the distance field can have a complete model without any holes, as shown in Fig. 14(b). Of course, if the point cloud is very sparse and extremely non-uniform, some pre-processing such as points smoothing and filling hole must be carried out properly in advance. A review of the many available methods for points smoothing and filling hole is beyond the scope of this paper. The reader may consult related works[22-27] for detailed expositions. Figure 15 shows the surface reconstruction result of Bunny model in the case of using a base mesh of voxel form. Irrespective of the type of base mesh, we can get almost same results in both quality and accuracy. The computation time of the numerical examples illustrated in this paper is summarized in Table 1. Our method is very fast and robust compared to other methods of surface reconstruction for unorganized point sets. It is hard to make a truly fair comparison between one's own method and those used in other papers. Through



(a) Initial base mesh

(b) After 1-st refinement & projection

(c) After 2-nd refinement & projection

Fig. 15 Shape reconstruction of Stanford Bunny model (II)

the investigation of the experimental results, it was found that the performance of the proposed method is about two or three times faster than author's previous work[12] due to the introduction of a hybrid method combined with the distance field and the least-squares projection.

5. Conclusions and future works

In this section, we summarize the results from this research and suggest a few future directions. In this paper, a novel surface reconstruction method was proposed to get a complete model from an unorganized point cloud. Our method is a hybrid method using the distance field and least-squares projection (LSP) without any explicit or implicit surface reconstruction procedure. Therefore, the implementation of algorithm is very simple and robust. In addition, the proposed method is very fast compared to the earlier RBFs based methods. In general, the iso-surface extraction using marching cube algorithm is very time consuming in the case of using volume resolutions larger than 256 x 256 x 256. In our

Table 1 Computational results

Model	# of points	Process	# of triangles and nodes after processing	Proposed method (time : sec)	Previous method[12] (time : sec)
Igea(Fig. 10)	213,425	surface reconstruction	# of nodes : 173,058 # of triangles : 346,112	60.6	133.3
Igea(Fig. 11)	173,058	denoising	# of nodes : 173,058 # of triangles : 346,112	27.6	impossible
Igea(Fig. 12)	213,425	surface reconstruction	# of nodes : 145,922 # of triangles : 291,840	49.7	119.3
Armadillo	320,532	surface reconstruction	# of nodes : 305,622 # of triangles : 613,600	91.8	192.7
Bunny(Fig. 14)	75,191	surface reconstruction	# of nodes : 80,299 # of triangles : 160,600	21.3	57.5
Bunny(left figure of Fig. 15)	75,191	surface reconstruction	# of nodes : 154,846 # of triangles : 309,696	37.4	108.5
Bunny(right figure of Fig. 15)	75,191	surface reconstruction	# of nodes : 154,846 # of rectangles : 154,848	36.8	103.1

(H/W: Pentium Ⅳ, 3 GHz CPU, 512MB RAM)

method, the iso-surface extraction was used only to get an initial coarse base mesh from the distance field having volume resolutions little than 64 x 64 x 64. Then, the final detailed surface was obtained by the iterative mesh refinement and LSP algorithm. This combination of the traditional distance field method and LSP algorithm could reduce the computing time and memory consumption remarkably. Another advantage of the method is that the amount of surface detail we can reconstruct is unlimited by using a simple mesh refinement scheme. Our method has also several limitations. First, we assume that the given point cloud is dense and uniform enough to get a fairly detailed resulting surface. If the point cloud is very sparse and non-uniform, proper processing such as points smoothing and filling hole should be performed previously. Second, the resulting surface is composed of a large number of meshes. For more practical applications, the general standard surface type such as B-spline surface, and NURBS surface will be more helpful. This is an open area for future research for this study.

ACKNOWLEDGEMENT

This work was supported by Daejin University Research Grants in 2010.

REFERENCES

1. Lee, I. K., "Curve reconstruction from unorganized points," Computer Aided Geometric Design, Vol. 17, No. 2, pp. 161-177, 2000.
2. Pauly, M., Gross, M. and Kobbelt, L., "Efficient simplification of point-sampled surfaces," Proceedings of IEEE Visualization '02, pp. 163-170, 2002.
3. Kobbelt, L. and Botsch, M., "A survey of point-based techniques in computer graphics," Computers and Graphics, Vol. 28, No. 6, pp. 801-814, 2004.
4. Azariadis, P., "Parameterization of clouds of unorganized points using dynamic base surfaces," Computer-Aided Design, Vol. 36, No. 7, pp. 607-623, 2004.
5. Amenta, N., Choi, S. and Kolluri, R., "The power crust," Proceedings of the sixth ACM Symposium on Solid modeling and applications, pp. 249-266, 2001.
6. Whitaker, R., "A level-set approach to 3D reconstruction from range data," International Journal of Computer Vision, Vol. 29, No. 3, pp. 203-231, 1998.
7. Zhao, H., Osher, S. and Fedkiw, R., "Fast surface reconstruction using the level set method," 1st IEEE Workshop on Variational and Level Set Methods, pp. 194-202, 2001.
8. Muraki, S., "Volumetric shape description of range data using blobby model," ACM SIGGRAPH Computer Graphics, Vol. 25, No. 4, pp. 227-235, 1991.
9. Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C. and Evans, T. R., "Reconstruction and representation of 3D objects with radial basis functions," Proceedings of SIGGRAPH '01, pp. 67-76, 2001.
10. Kojekine, N., Hagiwara, I. and Savchenko, V., "Software tools using CSRBFs for processing scattered data," Computers and Graphics, Vol. 27, No. 2, pp. 311-319, 2003.
11. Ohtake, Y., Belyaev, A., Alexa, M., Turk, G. and Seidel, H. P., "Multi-level partition of unity implicits," ACM Transactions on Graphics, Vol. 22, No. 3, pp. 463-470, 2003.
12. Yoo, D. J., "Filling Holes in Large Polygonal models Using an Implicit Surface Scheme and the Domain Decomposition Method," Int. J. Precis. Eng. Manuf., Vol. 8, No. 1, pp. 3-10, 2007.
13. Liu, Y. S., Paul, J. C., Yang, J. H., Yu, P. Q., Zhang, H., Sun, J. G. and Ramani, K., "Automatic least-squares projection of points onto point clouds with applications in reverse engineering," Computer-Aided Design, Vol. 38, No. 12, pp. 1251-1263, 2006.

14. Hoppe, H., DeRose, T. and Duchamp, T., "Surface Reconstruction from Unorganized Points," Proceedings of SIGGRAPH '92, pp. 71-78, 1992.

15. Lange, C. and Polthier, K., "Anisotropic smoothing of point sets," Computer Aided Geometric Design, Vol. 17, No. 2, pp. 161-177, 2000.

16. Ou, Y. D. and Feng, H. Y., "On the normal vector estimation for point cloud data from smooth surfaces," Computer-Aided Design, Vol. 37, No. 10, pp. 1071-1079, 2005.

17. Barentzen, J. A. and Aanas, H., "Signed distance computation using the angle weighted pseudo-normal," Transactions on Visualization and Computer Graphics, Vol. 11, No. 3, pp. 243-253, 2005.

18. Gueziec, A., "Meshsweeper: Dynamic point-to-polygonal mesh distance and applications," IEEE Transactions on Visualization and Computer Graphics, Vol. 7, No. 1, pp. 47-60, 2001.

19. Sud, A., Otaduy, M. A. and Manocha, D., "DiFi: Fast 3D distance field computation using graphics hardware," Proc. of Euro-graphics, Vol. 23, No. 3, pp. 557-566, 2004.

20. Frisken, S. F., Perry, R. N., Rockwood, A. P. and Jones, T. R., "Adaptively sampled distance fields: a general representation of shape for computer graphics," Proceedings of SIGGRAPH '00, pp. 249-254, 2000.

21. Huang, J., Li, Y., Crawfis, R., Lu, S. C. and Liou, S. Y., "A complete distance field representation," Proceedings of Visualization '01, pp. 247-254, 2001.

22. Yoo, D. J., "A Study on Filling Holes of the Polygonal model using Implicit Surface Scheme," Journal of the Korean Society for Precision Engineering, Vol. 22, No. 3, pp. 107-114, 2005.

23. Yang, X. N. and Wang, G. Z., "Planar point set fairing and fitting by arc splines," Computer-Aided Design, Vol. 33, No. 1, pp. 35-43, 2001.

24. Hu, G., Peng, Q. and Forrest, A. R., "Mean shift denoising of point-sampled surfaces," Visual Computation, Vol. 22, No. 3, pp. 147-157, 2006.

25. Schall, O., Belyaev, A. and Seidel, H. P., "Robust filtering of noisy scattered point data," Proceedings of Eurographics Symposium on Point-Based Graphics, pp. 71-77, 2005.

26. Yoo, D. J. and Kwon, H. H., "Shape Reconstruction, Shape Manipulation, and Direct Generation of Input Data from Point Clouds for Rapid Prototyping," Int. J. Precis. Eng. Manuf., Vol. 10, No. 1, pp. 103-113, 2009.

27. Yoo, D. J., "Three-dimensional Morphing of Similar Shapes Using a Template Mesh," Int. J. Precis. Eng. Manuf., Vol. 10, No. 1, pp. 55-66, 2009.

28. Yoo, D. J., "General 3D Offsetting of a Triangular Net Using an Implicit Function and the Distance Fields," Int. J. Precis. Eng. Manuf., Vol. 10, No. 4, pp. 131-142, 2009.