The Journal of the Society for Art and Science Vol. 14, No. 4, pp. 117-128

Feature-Preserving Simplification of Point Cloud by Using Clustering Approach Based on Mean Curvature

Xi Yang[1)] **Katsutsugu Matsuyama**[1)] **Kouichi Konno**[1)]
Yoshimasa Tokuyama[2)]

1) Graduate School of Engineering, Iwate University
2) Faculty of Engineering, Tokyo Polytechnic University

yangxi @ lk.cis.iwate-u.ac.jp

Abstract
For point cloud data obtained from 3D scanning devices, excessively large storage and long post-processing time are required. Due to this, it is very important to simplify the point cloud to reduce calculation cost. In this paper, we propose a new point cloud simplification method that can maintain the characteristics of surface shape for unstructured point clouds. In our method, a segmentation range based on mean curvature of point cloud can be controlled. The simplification process is completed by maintaining the position of the representative point and removing the represented points using the range. Our method can simplify results with highly simplified rate with preserving the form feature. Applying the proposed method to 3D stone tool models, the method is evaluated precisely and effectively.

Keywords: Point Cloud, Clustering, Mean Curvature, Simplification.

1 Introduction

The better the performance of 3D scanning devices[1] becomes year by year, the greater the number of generated point clouds will be. For some post-processing in the reverse engineering[2, 3], however, a large number of scan points will greatly consume the storage capacity for the process and the longer processing time is required. Simplification procedure for such point clouds is an efficient solution to these issues[4].

Point cloud simplification is a process that removes a large number of redundant points and it keeps the feature points representing 3D model sharp feature and boundaries. Currently, a lot of point cloud simplification algorithms have been developed. The algorithms are roughly divided into two categories: mesh-based methods and point-based ones. Simplification based on polygon meshes requires reconstruction of triangular meshes from a point cloud and then redundant points are removed[5, 6, 7]. In contrast, simplification based on points does not require reconstruction of a mesh model and it only relies on the information of points to simplify a point cloud.

On account of large amount of data to be processed and high computational complexity, the execution time of mesh-based simplification algorithm is extremely long[5]. In addition, triangular mesh reconstruction from a point cloud is very complicated. Therefore, the point-based simplification algorithm is applied more widely at present.

In this paper, we propose a new point cloud simplification algorithm based on curvature of points. Our simplification can be performed by using the pre-processing of Chida's method(see section 2.2). Simplification evaluation is optimized to find adjacent flakes with their method[21].

2 Related Work

2.1 Previous Simplification

The simplification algorithms based on point clouds have several typical methods. These methods are not efficient or available for stone tool models. Lee et al.[8] present a 3D point cloud simplification method by using 3D grids. Pauly et al.[9] introduced and analyzed different strategies for surface simplification of geometric models from unstructured point clouds. Moenning and Dodgson[10, 11] devised a coarse-to-fine uniform simplification algorithm with user-controlled density guarantee, based on the idea of progressive intrinsic farthest point sampling of a surface in point clouds. Lee et al.[13] presented a novel simplification method by adopting the Discrete Shape Operator to find the weight of the features of 3D models. Peng et al.[15] proposed a new simplification algorithm based on feature extraction for unstructured point clouds with unit normal vectors.

In addition, the previous simplification methods also have their own defects. For example, Song et al.[12] studied a global clustering point cloud simplification approach by searching for a subset of the original input data set according to a specified data reduction ratio. In their method, a global optimal result is obtained by minimizing the geometric deviation between the input point sets and the simplified ones. But when the number of points is reduced to become too small, the approximated point-to-surface distances may no longer get accurate values, and it is hugely time-consuming. Thus, it is inefficient for a large number of our stone tool models. Miao et al.[14] proposed a curvature aware re-sampling approach based on an adaptive mean-shift clustering scheme to simplify point clouds. An adaptive mean-shift clustering scheme is designed to generate a non-uniformly distributed simplification result. While it is difficult to incorporate the simplified geometric error in their algorithm, its simplified rate is low. Thus, it is hard to obtain a consistent error result for each stone tool models. Shi et al.[16] presented a new adaptive cluster subdivision simplification method using k-means clustering according to the two factors: user-defined space interval and normal vector tolerance. For stone tools, however, flake surfaces are smooth and the others may be very rough. Thus, the ridge lines represented by flake surfaces are deleted when user-defined parameters become large in order to raise the simplified rate. Furthermore, these methods are not show the evaluation of normalized distance, therefore the simplified results may not satisfy the requirements of our follow-up studies.

2.2 Search of Joining Material

Chida et al.[21] are studying on a method to search adjacent flake surfaces on stone implements for generating a joining material. In their method, a point cloud has to be simplified with maintaining form features to extract edges. Since the method[21] is based on polygon-based approach, it is necessary to generate a polygon mesh from a point cloud. In addition, the method[21] is to search an adjacent stone tool through the geometric matching score of flake surfaces. In general, however, since stone tools with matching flake surfaces are simplified respectively, the matching score is low.

For a simplification algorithm of a point cloud, it is important to evaluate the simplification errors. In general, the geometric errors between the original point cloud and the simplified one should be measured, such as the maximum error and the average error[9, 14, 16].

Our method, however, will be used in searching matching adjacent flake surfaces. Stone flakes are assembled on the same stone core and the position and posture of the adjacent flake surfaces are restored to form a joining material of stone tools[21]. When flake surfaces are matched and the candidates for adjacent surfaces are detected, a threshold value of the normalized distance[22] is a very important parameter for selecting the optimal flake surface.

Then, the normalized distance is employed to evaluate the simplified result in this paper. The normalized distance D between the original point cloud PC and the simplified one PC' is measured by the following equations:

$$d_i = (\mathbf{V}_i - \mathbf{g}_i) \cdot \mathbf{n}_i \quad (1)$$

$$D = \frac{1}{S}\sum_{i=1}^{n}(d_i)^2 \quad (2)$$

For each point $\mathbf{V}_i \, \epsilon \, PC$, the geometric error is distance d_i between original point $\mathbf{V}_i$ and its corresponding triangle T_i in simplified point cloud PC'. Assuming that $\mathbf{n}_i$ is the normal vector of T_i and $\mathbf{g}_i$ is the geometric center of T_i, d_i can be calculated by equation(1). In order to get a valve that does not depend on a polygonal mesh area obtained from a point cloud, the sum of $(d_i)^2$ is divided by the sum of triangular areas that belong to PC' like equation(2). Since a polygonal mesh is required in equations (1) and (2), the simplified point cloud is temporally reconstructed by the method described in [17].

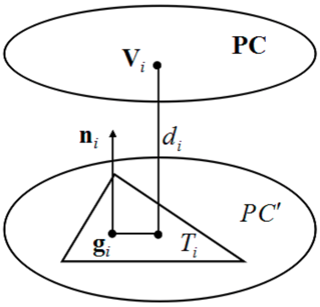


Figure 1: Normalized distance

3 Our Simplification Method

In this paper, a point cloud is simplified by the mean curvature of points. The flowchart of the method is shown in Figure 2. Firstly, the mean curvatures of points in a point cloud are computed. After that, all points in the entire point cloud are sorted in descending order of the mean curvatures. Then the segmentation range of the point with the maximum mean curvature is calculated. Next, the points in the point cloud in this range are removed. Finally, the point with the second-largest mean curvature in the rest of the point cloud is taken and the steps described earlier are repeated until all points in the point cloud are processed. The simplification process is completed.

3.1 Curvature Calculation of Point Cloud

For curvature estimation of point cloud, several algorithms are proposed[18, 19]. In [20], a method to calculate the curvature of the point

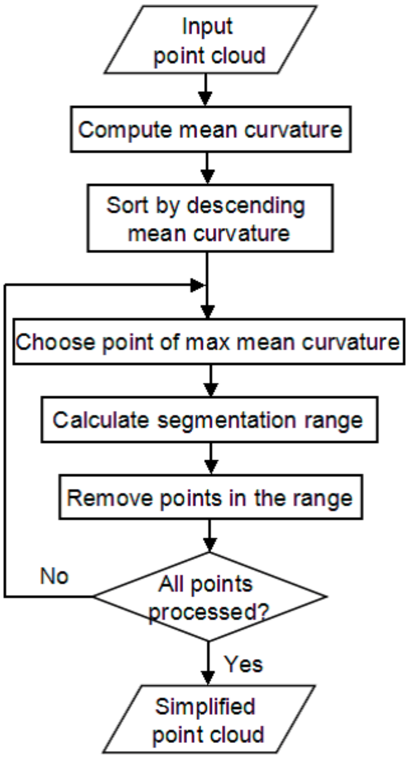


Figure 2: Flowchart of our algorithm

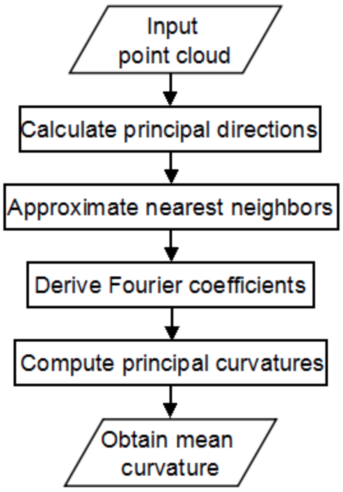

Input
point cloud
Calculate principal directions
Approximate nearest neighbors
Derive Fourier coefficients
Compute principal curvatures
Obtain mean
curvature

Figure 3: Mean curvature calculation in our method

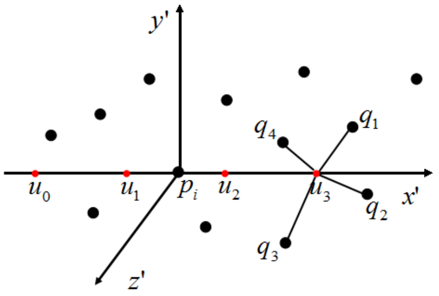


Figure 4: Equidistant sampling values generated from nearest neighbors of p_i

cloud is proposed. Specifically, the method represents shape features of the valley and the ridge lines by a frequency-domain. In addition, the method can be applied to noisy non-aligned point clouds. In our method, we calculate the principal curvatures of a point cloud according to the method[20].

Each point p_i of an input point cloud is processed as shown in Figure 3. First, the two principal directions of p_i are selected from three eigenvectors and calculated by the principal component analysis(PCA), according to K nearest neighbors of p_i. In the experiment, the size of neighbors K is set as 30. Let e_1, e_2, e_3 be the eigenvectors of PCA, corresponding to the eigenvalues λ_1, λ_2, λ_3 ($\lambda_1 \leqq \lambda_2 \leqq \lambda_3$). Eigenvector e_1 estimates the normal vector of p_i, eigenvectors e_2 and e_3 estimate the principal directions of p_i. Thus, a local coordinate system $(O'x'y'z')$ can be constructed at point p_i, the axes lie on $O'z'$, $O'y'$ and $O'x'$ along the eigenvectors e_1, e_2, e_3 respectively.

The point p_i and its K nearest neighbors are transformed to the local coordinate system and approximated by a truncated Fourier series in each principal direction. For example, a equidistant sampling points set u_l $(l = 0, ..., N-1)$ is created along the $O'x'$ axis to calculate principal curvature k_{ix} (N is set as 8 or 16 generally), as shown if Figure 4. The set center at p_i and its step size is chosen to be smaller than the average value $\overline{d}$ of the distances between each point of the input point cloud and its closest point. For each point u_l, its closest points q_j $(j = 0, 1, 2, 3)$ are selected from K nearest neighbors of p_i in the four spaces separated by plane $x'O'z'$ and plane $x'O'y'$. The sampling value $g(u_l)$ is computed by equation (3), where, $d(u_l, q_i)$ is a distance between u_l and p_i and z_{q_i} is the z value of q_i.

$$g(u_l) = \sum_{i=1}^{4} \frac{\frac{1}{d(u_l,q_i)}}{\sum_{j=1}^{4} \frac{1}{d(u_l,q_i)}} z_{q_i} \tag{3}$$

Next, the Fourier coefficients are derived by Fast Fourier Transform(FFT) according to the sampling values, as equation (4), where $Re(FFT(g(u_i)))$ and $Im(FFT(g(u_i)))$ are real and imaginary parts of the coefficients of $FFT(g(u_i))$.

$$\begin{aligned} a_0 &= \frac{1}{N} Re(FFT(g(u_0))), \\ Z \quad a_i &= \frac{2}{N} Re(FFT(g(u_i))), (i = 1, ..., \frac{N}{2} - 1), \\ a_i &= \frac{2}{N} Im(FFT(g(u_i))), (i = 1, ..., \frac{N}{2} - 1). \end{aligned} \tag{4}$$

After that, with the expressions of curvature and truncated Fourier series as equation (5), the principal curvature k_{ix} of p_i is computed from the approximated curves as equation (6), and k_{iy} can be computed in the same way along $O'y'$ axis.

$$f(x) \simeq \frac{1}{2} a_0 + \sum_{n=1}^{N} a_n cos \frac{n\pi x}{L} + \sum_{n=1}^{N} b_n sin \frac{n\pi x}{L} \tag{5}$$

$$k_{ix} = -\frac{\frac{\pi^2}{L^2} \sum_{n=1}^{N} n^2 a_n}{(1 + (\frac{\pi}{L} \sum_{n=1}^{N} n b_n)^2)^{\frac{3}{2}}} \tag{6}$$

Finally, mean curvature H_i of p_i is calculated by the average of the two principal curvatures k_{ix} and k_{iy} as the equation (7):

$$H_i = \frac{1}{2}(k_{ix} + k_{iy}) \tag{7}$$

3.2 Simplification Based on Curvature

For a 3D model, the feature points characterize sharp edges and surface boundaries that have large curvature. On the other hand, the points with small curvature, which are not feature points, are redundant to represent the surfaces of a model. A segmentation range of a point based on the mean curvature is defined by the circles as shown in Figure 5, which can represent the correspondence with the range. Thus, if representative points are selected according to the mean curvature values, the point cloud can be simplified by removing the points with small curvatures and maintaining ones with large curvatures.

Firstly, all points in a point cloud are sorted by the absolute value of their mean curvatures, and the average mean curvature $\overline{H}$ of the point cloud is calculated as equation (8), $\overline{H}$ will be used to calculate the segmentation range later.

$$\overline{H} = \frac{1}{n} \sum_{i=1}^{n} |H_i| \tag{8}$$

Next, starting from the point p_i of current maximum curvature, the radius of its segmentation range r_i is calculated by equation (9).

$$r_i = \alpha \cdot \frac{\overline{H}}{|H_i|} \tag{9}$$

According to equation (9), segmentation range r_i can be determined by $\frac{\overline{H}}{|H_i|}$. Therefore, in the regions with larger curvatures the range is small, while in the regions with smaller curvatures, the range is large as shown in Figure 5. Additionally, users can control the scale of segmentation range by a positive value α. Points inside the circle of r_i are the target ones for local simplification. The number of points in a simplified point cloud can be estimated by α. Furthermore, in order to keep the balance of the simplified point clouds, the minimum value of the range can be arbitrarily set by users. In our implementation, this value is set to $\alpha/3$.

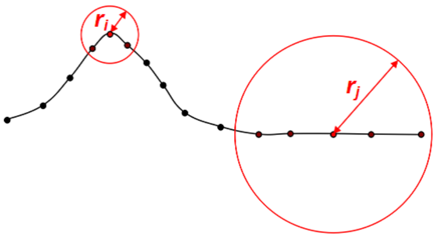


Figure 5: Segmentation range

After the representative point and segmentation range are determined, other points in the segmentation range are removed. Firstly the points in the range are searched by using the k-d tree[25]. The computational complexity of k-d tree construction, insertion and deletion is high while search is low. Due to this complexity, deletion of elements in the k-d tree is replaced by marking elements in the list. Therefore, the k-d tree of a point cloud needs to be built once in the beginning and deletion operation need not be repeated for other point clouds. Thus, the list and the k-d tree are retained in the memory during the operation and points can be searched in the k-d tree and marked in list, as shown in Figure 6.

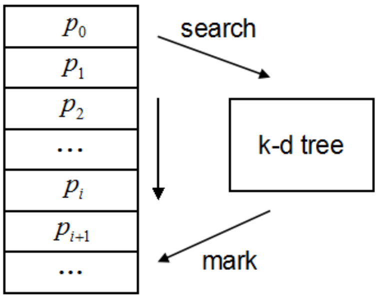


Figure 6: Removal of redundant points

After the points in the range are marked, the segmentation range of the next unmarked point is calculated, which has the currently maximum mean curvature current. Repeat these steps until all points in the point cloud are treated.

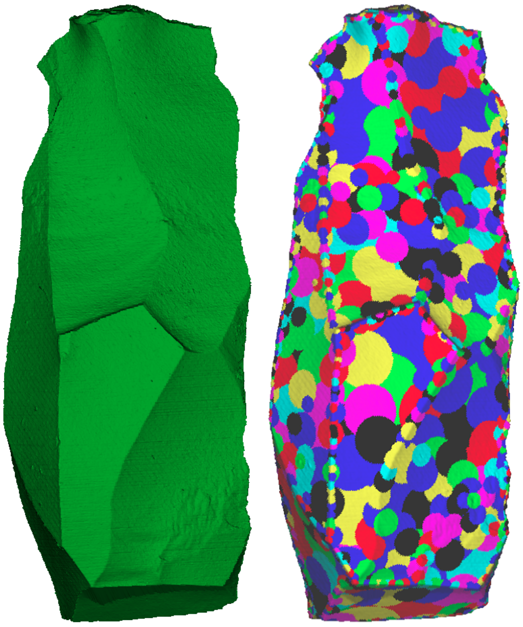
3.3 Evaluation

In the study of joining material searching[21], the normalized distance of each pair of flake surfaces is employed to evaluate the matching score for similarity. Since our new simplification method is used in the pre-processing stage of flake surface searching, the normalized distance for each flake surface of a stone tool between the original point cloud and the simplified one should be measured in the same manner as the evaluation of simplification result. Our method prevents from increasing normalized distance because removed points have small curvature. Therefore, the shape of simplified model is suitable for [21] by the proposed method.

4 Experimental Results and Limitation

We have implemented our algorithm using C++ and OpenGL, and tested on a PC with an Intel Core i5-3470 CPU and 8.00GB memory. The simplified point clouds are reconstructed by the method described in [17]. The original point clouds and the reconstructed simplified ones are displayed.

4.1 Clustering Result



(a) Original point cloud (b) Result of clustering

Figure 7: Result of clustering

Figure 7 shows the stone tool of No.L0197A0180 model and the result of clustering for this model, obtained by the proposed method. In the figure, the circles of different colors represent different clusters. Because of color matching, the different adjacent clusters may have the same color. The information of the model surface is shown clearly in Figure 7 (a). As the two figures are compared, we can clearly see that small clusters are placed on the boundaries and ridges of the model, while large clusters are placed for faces. So the method can maintain the contour information of the model and the redundant information can be removed efficiently.

4.2 Simplified Results of Our Method

Figure 8 shows a group of No.L0197A0151 model's simplified results (triangulated) by the new method with different segmentation ranges. The number of original points was 319K (a) and the numbers of simplified ones were 160K (b), 27K (c) and 9K (d). These examples demonstrated that the features of model were well maintained by the proposed method, even though the number of points was reduced to 2.84% of those in the original point cloud.

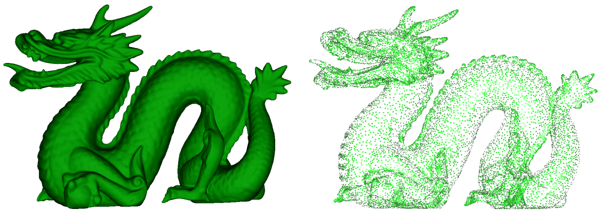
Figure 9 shows three stone tool models simplified by using the proposed method. The number of points of No.L0197A0178 model was reduced to 3.73% of those in its original model, that of No.S008A00010 model was reduced to 5.63%, and that of the No.S008A0003 model was reduced to 8.12%. In the right column of Figure 9, the preserved points of models were clearly shown by surfels(surface elements proposed in [23]). The experimental results indicate that the new method can maintain the boundaries and ridges of 3D models.

Table 1 shows the number of original points, the execution time of each step, the number of preserved points, and the maximum normalized distances of each flake surface. The execution times show the method is efficient. For the matching study[21], the normalized distance of each flake surface segmented from the stone tool models is calculated. The evaluation results of the maximum normalized distance indicate that the error between simplified result and the original point cloud is very small, and the simplified results are more than adequate requirements of matching study[21]; that is, the new simplification method has a good effect on the preprocessing of the matching study.

Table 2: Comparison of Dragon model.

Method	Ours	Miao et al.
Num. of original points	437,645	437,645
Eexecution time (sec.)	44.20	53.60
Num. of preserved points	34,861	34,049
Simplification ratio	92.03%	92.22%
Normalized average error	1.04×10^{-4}	5.29×10^{-4}

Stanford Dragon model was also tested and shown in Figures 10. In order to provide the standard of evaluation, the normalized average geometric error[24, 14] was computed, shown in Table 2. In paper [14], Dragon model was tested and the number of simplified point cloud was 34,049, the normalized average error was 5.29×10^{-4}. While the normalized average error of our method was 1.04×10^{-4}. According to paper[14], the total execution time of Miao et al. was 53.60 seconds, and that of our method was 44.20 seconds. This contrast could be used as a reference for the efficiency of our method, although they were measured in different experiment environments. As the result, normalized average error was smaller than one of the method[14], when the shape was simplified as same as the number of the model. Our method prevents from increasing normalized distance because removed points have small curvature. Therefore, the superb evaluation result of normalized average error is also obtained. This example demonstrated the efficiency of our method, and our method could obtain good simplified results for 3D models.



(a) Original point cloud (b) Simplified point cloud

Figure 10: Dragon

Concerning the execution of our method for very large data, an excavated relic model shown in Figure 11 by surfels[23] was tested. The number of points of original model was 2,277,128 and that of the preserved points was 10,144. The total execution time was approximately 253 seconds. This example indicated that our method has a good performance for big data.

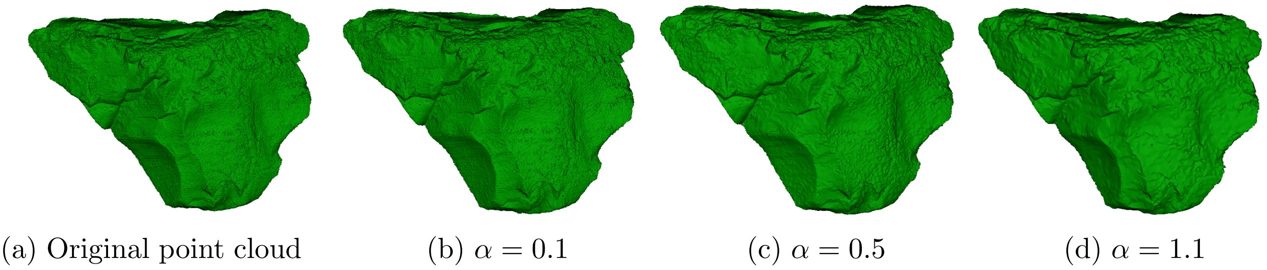
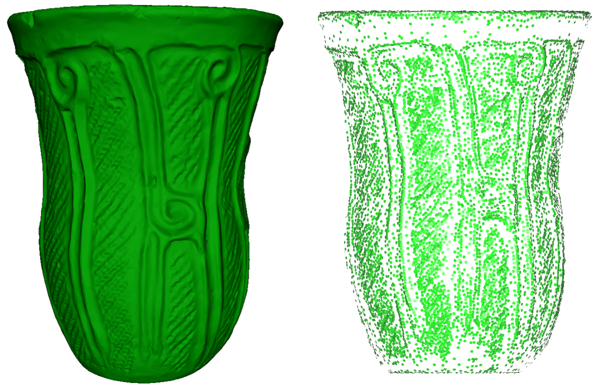


Figure 8: Results of simplification with different segmentation ranges (triangulated)

Table 1: Status of three stone tool models.

Model	Number of original points	Execution time of each step (sec.)			Number of preserved points	Simplification ratio	Max normalization distance
		File reading	Curvature calculation	Simplification			
L0197A0178	115030	0.522	12.676	0.353	4287	96.27%	0.0137
S008A00010	129145	0.566	12.001	0.417	7267	94.37%	0.0123
S008A0003	146214	0.688	13.834	0.508	11876	91.88%	0.0140



(a) Original point cloud (b) Simplified point cloud

Figure 11: An excavated relic

4.3 Limitation

For a 3D model has thin flake shape as shown in Figure 12, the feature point p_1 may be removed in the segmentation range of point p_0 on the opposite side, when the curvature of p_1 smaller than p_0.

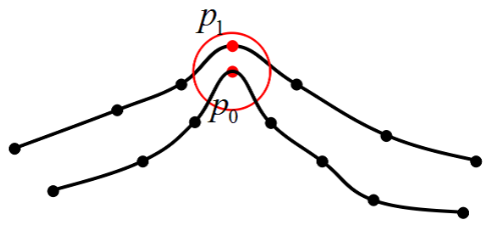
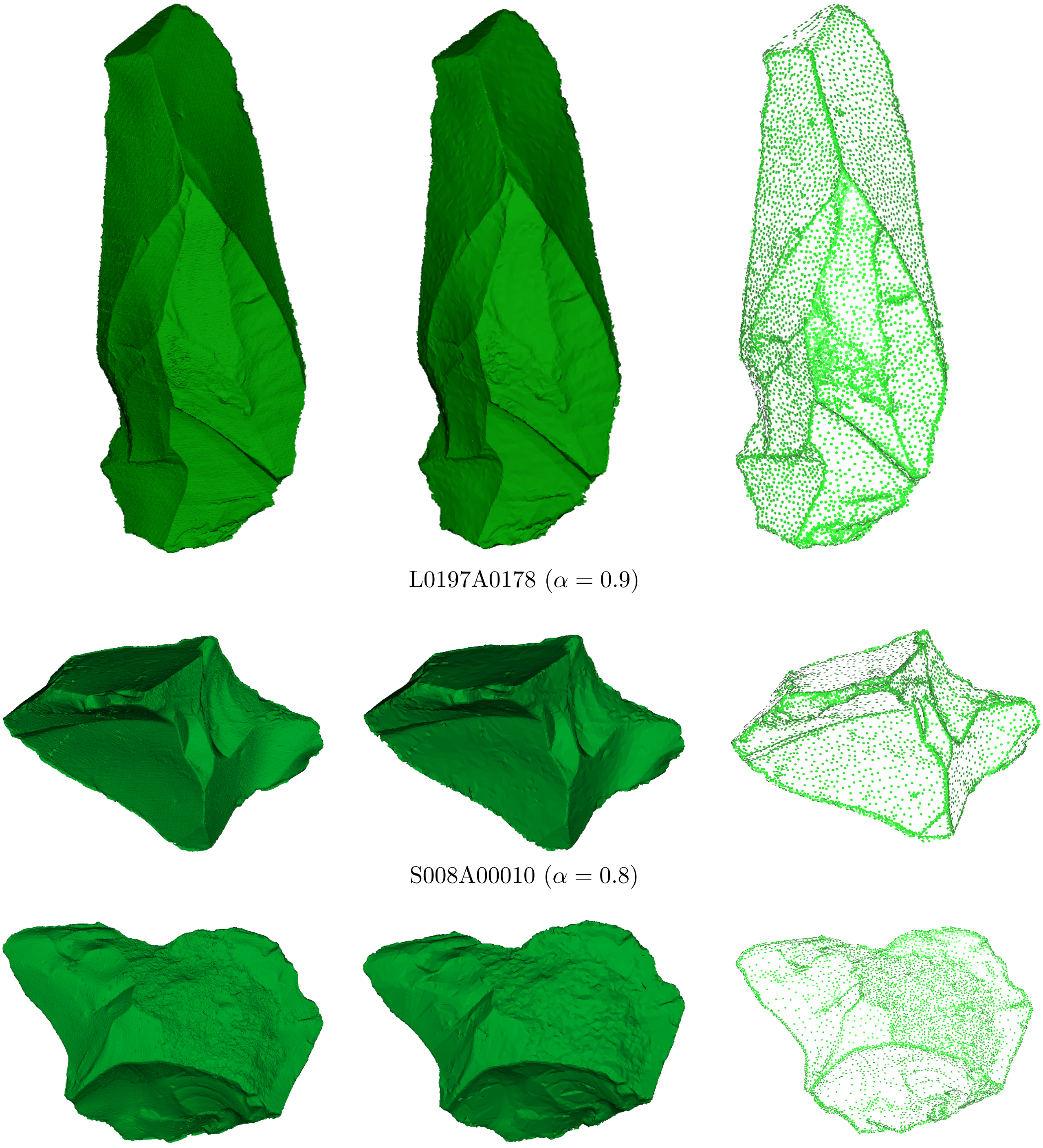


Figure 12: Thin flake shape section

5 Conclusions and Further Work

In this paper, a new curvature-based point cloud simplification algorithm is proposed. The segmentation range is proposed for using one point that represents the others in the range. Then the method of removing redundant points in the range is applied according to the order of curvatures to simplify the point cloud.

The normalized distance is employed to estimate the error of simplified results, and the user can control the degree of simplification by a space interval parameter. The larger parameters can lead to higher degree of simplification and larger simplification errors. Experiment results show that the proposed new algorithm can obtain ef-



L0197A0178 ($\alpha = 0.9$)

S008A00010 ($\alpha = 0.8$)

S008A0003 ($\alpha = 0.9$)

Figure 9: Simplified results of the new method. Left: original point clouds(triangulated); middle: simplified point clouds(triangulated); right: simplified point clouds(surfel[23])

ficient results that have high simplification rates and low simplification errors.

This segmentation spherical range, however, is not a good way for some extreme cases. There are three types of the local shape of point cloud : flat, rugged and ridge areas. These three types can be expressed by principal curvatures in two directions. Therefore, in the future, three different ellipsoids will be designed to calculate the segmentation spaces according to the two principal curvatures against the different local shapes. This will obtain higher simplification accuracy.

The basic concept of our method has already been presented in NICOGRAPH 2014[26] and we extended the concept in this paper. We are extremely grateful for lots of efficient advice from the paper reviewers.

Acknowledgements

We would like to thank the Tokyo National Museum for providing data of their stone tool models, and the Morioka City Study Museum of Archeological Site for providing an excavated relic model in our experiment. This work was partially supported by KAKENHI(26420090).

References

[1] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, et al. The digital Michelangelo project: 3D scanning of large statues, *Proceedings of ACM SIGGRAPH 2000*, pp. 131-144, 2000.

[2] T. Varady, R. Martin, J. Cox, Reverse Engineering of Geometric Models - An Introduction, *Computer-Aided Design*, Vol. 29, No. 4, pp. 255-268, 1997.

[3] R. D. Toledo, B. Levy, J. Paul, REVERSE ENGINEERING FOR INDUSTRIAL-ENVIRONMENT CAD MODELS, *Proceedings of TMCE 2008*, 2008.

[4] K. H. Lee, H. Woo, T. Suk, Data reduction methods for reverse engineering, *The International Journal of Advanced Manufacturing Technology*, Vol. 17, No. 10, pp. 735-743, 2001.

[5] P. Cignoni , C. Montani , R. Scopigno, A comparison of mesh simplification algorithms, *Computers and Graphics*, 1997.

[6] D. P. Luebke, A Developer ' s Survey of Polygonal Simplification Algorithms, *IEEE Computer Graphics and Applications*, Vol. 21, No. 3, pp. 24-35, 2001.

[7] Y. Yoshida, K. Konno, Y. Tokuyama, A Distributed Simplification Method with PC Cluster, *The Journal of the Society for Art and Science*, Vol. 7, No. 3, pp. 113-123, 2008.

[8] K. H. Lee, H. Woo, T. Suk, Point Data Reduction Using 3D Grids, *The International Journal of Advanced Manufacturing Technology*, Vol. 18, No. 3, pp. 201-210, 2001.

[9] M. Pauly, M. Gross, L. P. Kobbelt, Efficient Simplification of Point-Sampled Surfaces, *Proceedings of the conference on Visualization '02*, pp. 163-170, 2002.

[10] C. Moenning, N. A. Dodgson, A new point cloud simplification algorithm, *Proceedings 3rd IASTED Conference on Visualization, Imaging and Image Processing*, 2003.

[11] C. Moenning, N. A. Dodgson, Intrinsic point cloud simplification, *Graphic Conference 2004*, 2006.

[12] H. Song, H. Y. Feng, A global clustering approach to point cloud simplification with a specified data reduction ratio, *Computer-Aided Design*, Vol. 40, No. 3, pp. 281-292, 2007.

[13] P. F. Lee, B. S. Jong, Point-based Simplification Algorithm, *Journal WSEAS Transactions on Computer Research*, Vol. 3, No. 1, pp. 61-66, 2008.

[14] Y. Miao, R. Pajarolac, J. Feng, Curvature-aware adaptive re-sampling for point-sampled geometry, *Computer-Aided Design*, Vol. 41, No. 6, pp. 395-403, 2009.

[15] X. Peng, W. Huang, P. Wen, X. Wu, Simplification of Scattered Point Cloud Based on Feature Extraction, *Genetic and Evolutionary Computing, 2009. WGEC '09. 3rd International Conference*, pp. 335-338, 2009.

[16] B. Q. Shi, J. Liang, Q. Liu, Adaptive simplification of point cloud using k-means clustering, *Computer-Aided Design*, Vol. 43, No. 8, pp. 910-922, 2011.

[17] E. Altantsetseg, Y. Muraki, F. Chiba, and K. Konno, 3D Surface Reconstruction of Stone Tools by Using Four-Directional Measurement Machine, *The International Journal of Virtual Reality (IJVR)*, Vol. 10, No. 1, pp. 37-43, 2011.

[18] Y. Qiu, X. Zhou, P. Yang, X. Qian, Curvature Estimation of Point Set Data Based on the Moving-Least Square Surface, *Journal of Shanghai Jiaotong University (Science)*, Vol. 16, No. 4, pp. 402-411, 2011.

[19] G. Mullineux, S. T. Robinson, Fairing point sets using curvature, *Computer-Aided Design*, Vol. 39, No. 1, pp. 27-34, 2007.

[20] E. Altantsetseg, Y. Muraki, K. Matsuyama, K. Konno, Feature Line Extraction from Unorganized Noisy Point Clouds Using Truncated Fourier Series, *The Visual Computer*, Vol. 29, No. 6-8, pp. 617-626, 2013.

[21] A. Chida, K. Matsuyama, F. Chiba, K. Konno, A Rapid Searching Method of Adjacent Flake Surfaces in Stone Implements by Using Sets of Measured Points for Generating a Joining Material, *The Journal of the Society for Art and Science*, Vol. 13, No. 2, pp. 107-115, 2014.

[22] K. Yamahara, K. Konno, F. Chiba, M. Sato, A Method of Detecting Adjacent Flakes in Stone Tool Restoration by Extracting Peeling Surfaces, *Japan Society for Archaeological Information*, Vol. 17, No. 1.2, pp. 23-32, 2011.

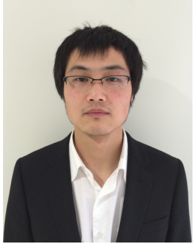
[23] H. Pfister, M. Zwicker, J. van Baar, M. Gross, Surfels: Surface Elements as Rendering Primitives, *SIGGRAPH '00 Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 335-342, 2000.

[24] P. Cignoni, C. Rocchini and R. Scopigno, Metro: measuring error on simplified surfaces, *Computer Graphics Forum*, Vol. 17, No. 2, pp. 167-174, 1998.

[25] J. L. Bentley, Multidimensional binary search trees used for associative searching, *Communications of the ACM*, Vol. 18, No. 9, pp. 509-517, 1975.

[26] X. Yang, K. Matsuyama, K. Konno, Y. Tokuyama, A Feature Preserving Simplification of Point Cloud by Using Clustering Approach Based on Mean Curvature, *NICOGRAPH 2014*, 2014.

Xi Yang



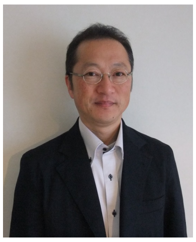
Xi Yang is currently a postgraduate student in Faculty of Engineering at Iwate University. He received the BE degree in College of Information Engineering from Northwest A&F University in 2012. His research interests include geometric modeling and computer graphics.

Katsutsugu Matsuyama

Katsutsugu Matsuyama is currently an assistant professor at Iwate University. His research interests include computer graphics, information visualization and interactive systems. He received BE, ME, DE degrees in computer science from

Iwate University in 1999, 2001 and 2005, respectively. He was a research associate at Future University-Hakodate from 2005 to 2011.

Kouichi Konno



Kouichi Konno is a professor of Faculty of Engineering at Iwate University. He received a BS in Information Science in 1985 from the University of Tsukuba. He earned his Dr.Eng. in precision machinery engineering from the University of Tokyo in 1996. He joined the solid modeling project at RICOH from 1985 to 1999, and the XVL project at Lattice Technology in 2000. He worked on an associate professor of Faculty of Engineering at Iwate University from 2001 to 2009. His research interests include virtual reality, geometric modeling, 3D measurement systems, and computer graphics. He is a member of IEEE.

Yoshimasa Tokuyama

Yoshimasa Tokuyama is a professor of Department of Media and Image Technology, Faculty of Engineering, Tokyo Polytechnic University. He received his MS in Mechanical Engineering in 1986 and doctor degree in Computer Graphics in 2000 from The University of Tokyo. He was a member of the 3D CAD project at RICOH's Software Division from 1986 to 2002. His areas of research interest include computer graphics, game, haptic interface, virtual reality, shape modeling, and their applications. He is a member of Information Processing Society of Japan, Institute of Image Information and Television Engineers of Japan, the Institute of Image Electronics Engineers of Japan, the Society for Art and Science.