49 A POINT CLOUD SIMPLIFICATION ALGORITHM FOR MECHANICAL PART INSPECTION

Hao Song, Hsi-Yung Feng
Department of Mechanical and Materials Engineering
The University of Western Ontario
London, Ontario, Canada N6A 5B9
hsong2@uwo.ca, sfeng@eng.uwo.ca

A point cloud data set, a set of massive and dense coordinate data points sampled from the surface of a physical object, is emerging as a new representation format of 3D shapes. This is mostly attributed to recent advances in the range finding technology of high-speed 3D laser scanning. A typical laser scanned data set often contains millions of data points and this leads to significant computational challenges in processing the point cloud data for practical applications such as the high-speed laser inspection of mechanical parts. To reduce the number of the massive data points to facilitate geometric computation, this paper presents a simplification algorithm for the laser scanned point cloud data from manufactured mechanical parts, whose boundary surfaces include sharp edges. Due to the distinct feature represented by the points located on or near the sharp edges, these points are first identified and retained. The algorithm then repeatedly removes the least important point from the remaining data points until the specified data reduction ratio is reached. Quantification of a point's importance is based on points in its neighborhood and it indicates the point's contribution to the representation of the local surface geometry. The effectiveness of the proposed algorithm is shown through the simplification results of two practical point cloud data sets.

1. INTRODUCTION

In many disciplines, there is a need to faithfully represent the shape of a physical object in computer. In mechanical engineering, for example, it is often necessary to capture the shape of a manufactured part to evaluate the manufacturing error with respect to the original design. The need of capturing complex 3D shapes is obvious in many applications of computer graphics such as simulation and computer animation. In addition, some medical applications, such as radiation therapy and surgical planning (Lorensen and Cline, 1987), need the 3D surface representation of the anatomy.

Transferring a physical object into computer has been made possible by the development of 3D data acquisition technologies. Modern 3D data acquisition

Please use the following format when citing this chapter:

Song, H., Feng, H.-Y., 2006, in IFIP International Federation for Information Processing, Volume 220, Information Technology for Balanced Manufacturing Systems, ed. Shen, W., (Boston: Springer), pp. 461–468.

devices, such as laser scanner, can quickly capture points from the surface of the object to produce a dense point cloud data set. A typical laser scanned data set often contains millions of data points and this leads to significant computational challenges in processing the point cloud data for practical applications such as the high-speed laser inspection of mechanical parts. Two reasons may cause the data acquisition process to produce too many points. One is the high measurement precision and speed of modern 3D data acquisition devices. The other reason is that, when making measurement, the operator never knows how dense the points should be in order to achieve the description of the shape at a certain accuracy level. Frequently, far more points are captured. In practice, a simplification or decimation of the acquired point set is often necessary for the subsequent applications to operate at a reasonable computational cost.

As the early stage of point cloud processing, simplification may serve to facilitate different subsequent applications, and it is desirable to have some particular characteristics present in the simplified point set for the particular application at hand. In mechanical engineering, it is very common to have sharp edges and sharp corners in manufactured parts. In the point cloud representation of theses parts, the points representing the sharp edges and sharp corners are so unique that the local geometry cannot be correctly reflected without them. Therefore, for the situations that the sharp edge information is critical for the subsequent application, it is desirable to retain those points close to sharp edges or sharp corners in the simplified point set.

In the inspection of mechanical parts, for example, a segmentation process, which determines feature lines of the surface and separates the point set by this lines, is often needed in order to border the correct portion on which the manufacturing errors will be checked and to cut off points in unwanted regions. The feature lines are extracted from the points close to sharp edges or sharp corners and are interpreted as trimming curves in the segmentation process.

This paper presents a simplification algorithm for the point cloud data scanned from mechanical parts, whose boundary surfaces include sharp edges. The algorithm selects a subset of the input point cloud based on points' importance in terms of defining their local geometries. The quantification of a point's importance is derived from its neighborhood – the point itself and a set of neighboring points chosen from the point cloud. The idea is to see whether or not the local geometry can be reliably implied from its neighboring points. If the local geometry cannot be reliably reflected by the neighboring points and their associated properties, the point is considered important in defining the geometry. The simplification is achieved by progressively removing points from the input cloud that are less important than others. Due to the discontinuity occurred at points close to sharp edges, the local geometry at these points cannot be reliably implied from the neighboring points. These points, considered as defining the important feature lines, are explicitly identified and retained in the simplified point set.

2. PREVIOUS WORK

Extensive research work has been reported on simplification of polygonal meshes, a closely related area to point cloud simplification. Actually, many of the reported

methods can be adapted to simplification of point clouds (Pauly et al., 2002). Luebke (2001) and Cignoni et al. (1998) presented good reviews on the algorithms of mesh simplification. Luebke (2001) described a useful taxonomy and discussed typical algorithms against some basic items of his classification such as simplification mechanism, the way of dealing with topology, and whether they are static, dynamic, or view-dependent simplifications. Cignoni et al. (1998) took an empirical approach. They tested and compared six representative algorithms by taking into account the computational cost and the quality of the resulting output meshes.

Compared to mesh simplification, direct simplification of point clouds is a new research area. Techniques are usually borrowed from related areas such as image processing and mesh simplification. Pauly and Gross (2001) proposed a spectral framework for processing point-sampled objects by introducing a concept of local frequencies on geometry. They split the model surface to a set of patches that can be represented as scalar height fields. The frequency spectrum of each patch was then obtained by the discrete Fourier transform (DFT). By directly applying signal processing theory, they re-sampled the patches individually and blended the patch boundaries. The simplification obtained by this method, however, unnecessarily depends on the specific layout of the patches.

Moenning and Dodgson (2004) applied the farthest point sampling principle, initially introduced by Eldar et al. (1997) in the context of image sampling, and offered control of density and feature sensitivity in their simplification procedure. The method takes a progressive approach which starts from a randomly picked sample point and repeatedly places the next sample point at the middle of the least-known sampling domain with the help of a pre-computed approximation of the geodesic Voronoi diagram of the input point cloud. Its density control is achieved by a user provided number indicating the upper bound of the geodesic distance from the next farthest point to the obtained (simplified) point set. The control of feature sensitivity, meaning that higher point density in highly curved region and lower density in flat region, is achieved by assigning weights to points according to the estimated local changes such as the mean curvature. Unfortunately, the relations among point set size, density condition, and feature sensitivity were not discussed, leaving their determination a difficult task to the user.

Pauly et al. (2002) adapted and generalized some popular methods in mesh simplification to the point cloud context. The methods of clustering, both incremental region growing and hierarchical clustering, iterative simplification, and particle simulation were tested against the criteria of accuracy, sampling distribution, and computational efficiency. By introducing an approximation of point-to-surface distance, where the surface is represented by a point cloud, they were able to quantitatively evaluate the error of the surface represented by the simplified point set to the surface represented by the original point cloud. Based on their comparison of the adapted methods, they gave useful hints to help choose the most appropriate method for different situations. However, none of their methods were originally designed to devote to any of the error criteria they used.

Lee et al. (2001) presented a 3D grid method for direct point data reduction. The method constructs connected 3D cells in the space to enclose the data points. The cells are generated by an octree-based spatial decomposition procedure and the criterion of the decomposition is defined as the standard deviation of the estimated

normal vectors of points within each cell. The simplification is then obtained by choosing a representative point from each cell. Since the number of cells, equivalently data points in the simplification, is related to the number of initial cells and the user-defined tolerance for the decomposition, this method does not produce a simplification with its size controlled in an intuitive way.

A straightforward approach is to judge the importance of each point in the point cloud. The importance is quantified by a measure which indicates either the amount of information contributed to the description of the shape or the redundancy of the point. The simplification is then achieved by removing points that contain the least amount of information or the largest redundancy. Linsen (2001) defined the information content of a point as a weighted sum of factors such as distances to its neighbors, non-planarity, and change of normal vectors evaluated in the selected neighborhood. Alexa et al. (2001) presented a measure which involved the Moving Least-Squares (MLS) surface (Levin, 2004). It is computed as the distance from a point to its projection onto the MLS surface derived from all the other points of the point cloud. They iteratively removed the point with the smallest distance and, after each removal, updated the distances of the affected points. Kalaiah and Varshney (2003) measured the redundancy of each individual point and iteratively removed the point with the largest redundancy. The evaluation of the redundancy is based on the local geometric properties derived from the point's neighborhood.

All the existing simplification methods are based on some smoothness condition of the underlying surface. Manufactured parts, however, frequently show sharp edges and sharp corners which naturally separate the surface into different patches. Although existing methods are able to work on surfaces with sharp features and small blends and produce simplification with relative dense points in the sharp feature regions, they do not explicitly identify the points on or close to the sharp edges and pass the information to subsequent applications.

3. THE SIMPLIFICATION ALGORITHM

The proposed simplification algorithm has two steps. The first step identifies points located on or close to sharp edges (referred as boundary points thereafter). These points are retained in the point cloud and will not be processed by the second step. The second step performs the simplification by progressively removing non-boundary points from the cloud. The removing is based on the quantification of each point's importance. Each time, the least important point is removed and the importance values of points affected by the removal are updated. The process continues until the specified number of points for the simplification is reached.

3.1 Identifying Boundary Points

Different methods have been proposed to identify points located on or close to sharp edges. These methods all involve the quantification of the smoothness of the underlying surface at a point's vicinity and the determination of a threshold. The method employed in this paper establishes a measure derived from the deviation of the normal vectors in a point's vicinity (Song and Feng). It attempts to compensate for the deviation caused by surface curvature and focuses on the component caused

by tangential discontinuity. The benefit of using such a measure is that the threshold can be automatically determined by identifying outlying values from the general trend shown by the values calculated at points in smooth regions. As a result, it minimizes user involvement in the process of boundary point detection.

The proposed method needs the information of the unit normal vector of the underlying surface at each data point. The estimation of the normal vectors from the input data cloud is done by the algorithm suggested by OuYang and Feng (2005). The algorithm features a novel definition of neighborhood (a set of neighboring points) for each data point, which is called the Local Voronoi Mesh Neighbors. Establishment of the neighborhood is based on the Voronoi diagram of the input point cloud P. Let $DN_{\mathbf{p}}$ denote the desired neighborhood for point $\mathbf{p}$. If $VN_{\mathbf{p}}$ is the set of points whose corresponding Voronoi cells share a facet with the Voronoi cell of $\mathbf{p}$ in the Voronoi diagram of P, then the algorithm chooses $DN_{\mathbf{p}}$ as a subset of $VN_{\mathbf{p}}$: $DN_{\mathbf{p}} \subseteq VN_{\mathbf{p}}$. The idea is to choose those in $VN_{\mathbf{p}}$ who are close to $\mathbf{p}$ and contribute to the construction of a local mesh centered at $\mathbf{p}$ to reflect the local geometry. In this paper, $VN_{\mathbf{p}}$ is also called the *Voronoi neighbors* of $\mathbf{p}$.

3.2 Evaluating Points' Importance

For a point $\mathbf{p} \in P$ in the smooth region of S, where P is the input point cloud and S is the surface represented by P, suppose it has K neighbors $\mathbf{q}_1$, $\mathbf{q}_2$, ... $\mathbf{q}_K$, and they are all in the smooth region of S (non-boundary points). Let the unit normal vectors at $\mathbf{q}_1$, $\mathbf{q}_2$, ... $\mathbf{q}_K$ be $\mathbf{n}_1$, $\mathbf{n}_2$, ... $\mathbf{n}_K$, respectively, which are estimated from P by the algorithm of OuYang and Feng (2005). The importance of $\mathbf{p}$ is calculated as

$$i(\mathbf{p}) = \sum_{l=1}^{K} \left| (\mathbf{p} - \mathbf{q}_l) \cdot \mathbf{n}_l \right| \Big/ K \tag{1}$$

Geometrically, Eq. (1) calculates the average of the Euclidean distances from $\mathbf{p}$ to the estimated tangent planes at $\mathbf{q}_1$, $\mathbf{q}_2$, ... $\mathbf{q}_K$, respectively. The value reflects the deviation from $\mathbf{p}$ to the underlying surface represented by its neighbors. A large value indicates that $\mathbf{p}$'s position cannot be reliably implied by its neighboring points. Therefore, $\mathbf{p}$ plays an important role in defining the local shape.

The evaluation takes advantage of the established neighborhood and the normal vector information obtained in the previous step for each data point. In the case that some of $\mathbf{p}$'s neighbors are boundary points, they are simply ignored in the evaluation as their associated normal vectors are not reliable.

3.3 Removing Points

The simplification is achieved by progressively removing points from P. In each iteration, it removes one point in the current configuration of P that has the smallest importance value. Suppose $\mathbf{p}$ is chosen to be removed. Those points that previously have $\mathbf{p}$ as one of their neighbors will change their neighborhood configurations. So the normal vectors at these points need to be re-estimated. Since the neighborhood for $\mathbf{p} \in P$ is a subset of $\mathbf{p}$'s Voronoi neighbors, the point that has $\mathbf{p}$ as one of its neighbors can be easily found by exploring $\mathbf{p}$'s Voronoi neighbors. Let $\mathbf{q}$ be such a point that $\mathbf{p}$ is one of $\mathbf{q}$'s neighbors, that is, $\mathbf{p} \in DN_{\mathbf{q}}$. Because $DN_{\mathbf{q}} \subseteq VN_{\mathbf{q}}$, then

$\mathbf{p} \in VN_{\mathbf{q}}$. Due to the characteristic of Voronoi diagram, it is easy to see that $\mathbf{q} \in VN_{\mathbf{p}}$. Therefore, searching in $VN_{\mathbf{p}}$ will cover all the points that previously have $\mathbf{p}$ as one of their neighbors. By updating the Voronoi diagram of P and re-establishing the neighborhoods, their associated normal vectors can be easily updated.

The impact on the importance values is even bigger. Figure 1 shows the example of the affected points due to the removal of $\mathbf{p}$. Suppose $\mathbf{p}$ is a neighbor of $\mathbf{q}_1$ (it is also a neighbor of $\mathbf{q}_2$, $\mathbf{q}_3$, $\mathbf{q}_4$, and $\mathbf{q}_5$). Due to the removal of $\mathbf{p}$, the estimated normal vector at $\mathbf{q}_1$ changes. The changed normal vector, equivalently the tangent plane at $\mathbf{q}_1$, will in turn have impact on the importance values of the points that have $\mathbf{q}_1$ as one of their neighbors (this type of points are marked as triangles in Fig.1). Therefore, if the affected normal vectors are restricted to the immediate ring of points $\mathbf{q}_1$, $\mathbf{q}_2$, ..., $\mathbf{q}_5$ around $\mathbf{p}$, the affected importance values will spread into the second ring of points: $\mathbf{t}_1$, $\mathbf{t}_2$, ..., $\mathbf{t}_9$. As a result, finding these points needs to explore not only in $VN_{\mathbf{p}}$ but also in $VN_{\mathbf{q}_1}$, $VN_{\mathbf{q}_2}$, ..., $VN_{\mathbf{q}_5}$.

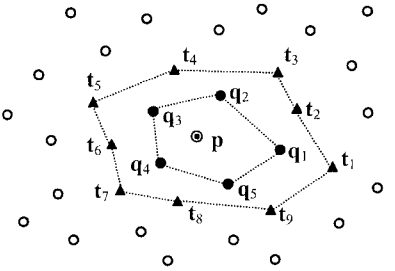


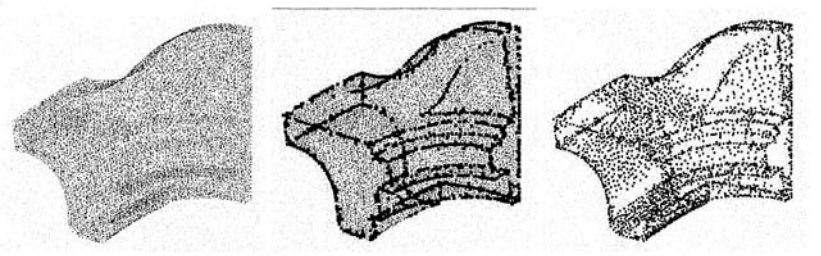
Figure 1 – The affected points (marked as solid dots and triangles) due to the removal of $\mathbf{p}$ from P.

4. IMPLEMENTATION AND RESULTS

The proposed algorithm relies on the maintenance of the Voronoi diagram of P when P is changing. Actually, it is the spatial structure established by the Voronoi diagram that helps explore the points whose normal vectors and importance values are affected by the removal. Both the construction of the Voronoi diagram and its dynamic maintenance are implemented by using the 3D Delaunay triangulation programs in CGAL (Computational Geometry Algorithms Library) (Boissonnat et al., 2002).

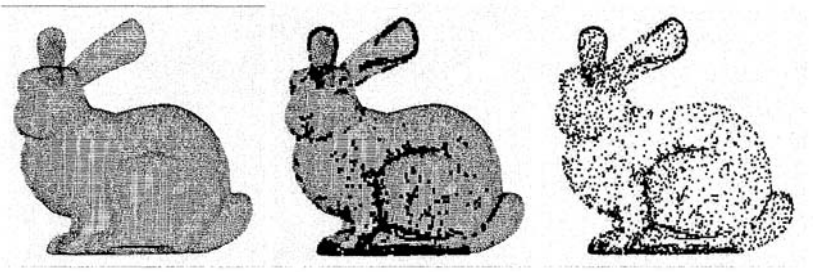
In addition, due to the spatial structure set by the Voronoi diagram, the order in which the points are evaluated does not affect the importance values. In the current implementation, the points are presented as an array and each element of the array records a point's position, associated normal vector, and indices of its neighbors.

The simplification algorithm has been tested using practical data sets. Figures 2 and 3 show the simplification of the point clouds of the Fandisk (16,475 points) and the Bunny (35,947 points). For each example, the identified boundary points are also highlighted to visualize the results obtained from the first step of the algorithm. The running times for identification of boundary points are 190 and 373 seconds for



(a) (b) (c)

Figure 2 – Fandisk: (a) the original point set; (b) the highlighted boundary points; and (c) the point set simplified to 20% of its original size.



(a) (b) (c)

Figure 3 – Bunny: (a) the original point set; (b) the highlighted boundary points; and (c) the point set simplified to 10% of its original size.

Fandisk and Bunny, respectively, on a Pentium 4, 3.0 GHz personal computer. The removing process took 1,253 seconds for Fandisk and 1,720 seconds for Bunny.

For the model of Fandisk which has clear boundaries (sharp edges) between different surface patches, the simplified point sets show clear sharp edge information as expected. In smooth regions, there is also a clear trend that points in regions of higher curvatures are denser than those in relatively flat regions. Result of the Bunny, featuring free form surfaces, also show varied point density in regions of different curvatures, which is a favorable characteristic for the simplified point set to achieve both detailed description of the shape and reduction of the data size.

5. CONCLUSIONS

A simplification algorithm for point cloud data has been presented. The obtained simplification completely preserves the sharp edge information and preserves the shape of smooth regions at a reasonable level of detail by progressively removing points in these regions. The removing process is based on the quantification of points' importance calculated from each point's neighborhood. The quantification indicates the point's contribution to the representation of the local geometry.

The algorithm needs the establishment of a neighborhood and the normal vector information for each point. Due to the particular way of establishing the

neighborhood, which is the basis for both normal vector estimation and the evaluation of the point importance, finding the points that need to update their associated normal vectors and importance values after a removal becomes efficient. This is achieved by the dynamic maintenance of the Voronoi diagram of the remaining data points.

6. ACKNOWLEDGMENTS

This work was funded in part by the Natural Sciences and Engineering Research Council of Canada.

The authors would like to thank the team of CGAL for making their programs publicly available. The point set of Fandisk was obtained from the web page of Dr. Hugues Hoppe at Microsoft Research. The point set of Bunny was obtained from the Large Geometric Models Archive at Georgia Institute of Technology, which was originally created at the Stanford Computer Graphics Laboratory.

7. REFERENCES

1. Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D, Silva CT. Point set surfaces. in: Proceedings of the IEEE Conference on Visualization (VIS'01), 2001, pp. 21-28.
2. Boissonnat J-D, Devillers O, Pion S, Teillaud M, Yvinec M. Triangulations in CGAL. Computational Geometry: Theory and Applications 2002; 22(1-3): 5-19.
3. Cignoni P, Montani C, Scopigno R. A comparison of mesh simplification algorithms. Computers & Graphics 1998; 22(1): 37-54.
4. Eldar Y, Lindenbaum M, Porat M, Zeevi YY. The farthest point strategy for progressive image sampling. IEEE Transactions on Image Processing 1997; 6(9): 1305-1315.
5. Kalaiah A, Varshney A. Modeling and rendering of points with local geometry. IEEE Transactions on Visualization and Computer Graphics 2003; 9(1): 30-42.
6. Lee KH, Woo H, Suk T. Point data reduction using 3D grids. International Journal of Advanced Manufacturing Technology 2001; 18(3): 201-210.
7. Levin D. "Mesh-independent surface interpolation". in: Geometric Modeling for Scientific Visualization, Brunnett G, Hamann B, Muller H, Linsen L. eds., Springer-Verlag, 2004, pp. 37-49.
8. Linsen L. Point cloud representation. Technical Report, Faculty of Informatics, University of Karlsruhe, Germany, 2001.
9. Lorensen WE, Cline HE. Marching cubes: a high resolution 3D surface construction algorithm. in: Proceedings of the 14th annual conference on Computer graphics and interactive techniques (SIGGRAPH'87), 1987, pp. 163-169.
10. Luebke DP. A developer's survey of polygonal simplification algorithms. IEEE Computer Graphics and Applications 2001; 21(3): 24-35.
11. Moenning C, Dodgson NA. Intrinsic point cloud simplification. in: Proceedings of the 14th International Conference on Computer Graphics and Vision (GraphiCon), Moscow, Russia, September 2004.
12. Pauly M, Gross M. Spectral processing of point-sampled geometry. in: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01), 2001, pp. 379-386.
13. Pauly M, Gross M, Kobbelt LP. Efficient simplification of point-sampled surfaces. in: Proceedings of the 13th IEEE Visualization Conference, Boston, MA, October 2002, pp. 163-170.
14. OuYang D, Feng H-Y. On the normal vector estimation for point cloud data from smooth surfaces. Computer-Aided Design 2005; 37(10): 1071-1079.
15. Song H, Feng H-Y. Automatic detection of tangential discontinuities in point cloud data. ASME Journal of Computing and Information Science in Engineering, in revision.