An efficient local clustering approach for simplification of 3D point-based computer graphics models

Zhiwen Yu
Department of Computer Science
City University of Hong Kong
yuzhiwen@cs.cityu.edu.hk

Hau-san Wong
Department of Computer Science
City University of Hong Kong
cshswong@cityu.edu.hk

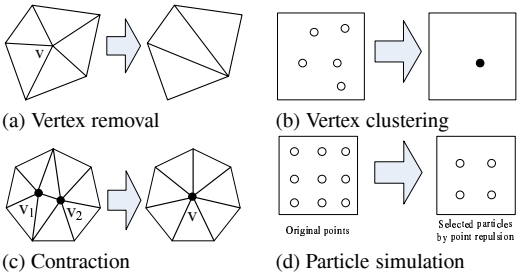
Abstract

Given a point-based 3D computer graphics model which is defined by a point set P ($P = \{p_i \in R^3\}$) and a desired reduced number of output samples N_s, the simplification approach finds a point set P_s which (i) satisfies $|P_s| = N_s$ ($|P_s|$ is the cardinality of P_s) and (ii) minimizes the difference of the corresponding surface S_s(defined by P_s) and the original surface S(defined by P). Although a number of previous approaches have been proposed for simplification, most of them (i) do not focus on point-based 3D models, (ii)do not consider efficiency, quality and generality together. In this paper, we introduce an adaptive simplification method (ASM) which is an efficient technique for simplifying point-based complex 3D model . ASM achieves low running time by clustering the points locally based on the preservation of geometric characteristics. Finally, we analyze the performance of ASM and show that it outperforms most of the current state-of-the-art methods in terms of efficiency, quality and generality.

1. Introduction

Early work in the simplification of 3D computer graphics model focuses on polygonal models because other model representations, such as spline, volumetric and implicit-surface can be converted to a polygonal one. There are four main types of algorithms for simplifying polygonal meshes: (i) Vertex removal (Figure 1(a)) (e.g. [1]); (ii) Vertex clustering (Figure 1(b)) (e.g. [2], [3]); (iii) Edge contraction(Figure 1(c)) (e.g. [4], [5]); (iv) Particle simulation(Figure 1(d))(e.g. [6], [7]). However, most of the above algorithms cannot be applied to point-based surface directly. The main motivation of this paper is to investigate whether there is a new approach which (i)inherits the advantage of simplification algorithms for polygonal meshes and (ii) works well on point-based surface.

With the popularity of 3D scanners, more and more point-based 3D models are available. Point-based 3D models have three characteristics: (i) most of them consists of millions of 3D points; (ii) the boundary surfaces of the models are represented by a discrete point set; (iii) there does not exist any topological relationship between these points. Traditional methods first convert a discrete point cloud to a continuous surface representation, such as spline, volumetric, implicit-surface, or polygon. Then, the models are simplified by the above simplification algorithms. However, the process of conversion is time consuming and memory consuming, especially with the increasing cardinality of the point set. As a result, another motivation of this paper is to investigate whether there is a new approach which can simplify the point set directly and avoid the process of conversion.



(a) Vertex removal (b) Vertex clustering

(c) Contraction (d) Particle simulation

Figure 1. Different kinds of simplification methods

The contribution of this paper are threefold. First, we propose a local clustering approach which is very efficient for clustering the points in a grid cell structure based on the preservation of geometric characteristics. Second, we introduce a novel error metric , known as key point error (KPE), to evaluate the error of the simplified model. Third, we introduce an adaptive simplification method (ASM) which combines local clustering with a global error-bound.

The rest of the paper is organized as follows. Section 2 surveys the related work on simplification methods for point-based models. Section 3 describes local clustering methods, key point error, and the ASM algorithm. Section 4 evaluates ASM experimentally. Section 5 is conclusion and future work.

2. Background

Recently, point clouds receive much attention as a representation of models in computer graphics. Point clouds which are produced by 3D scanning devices can be rendered and visualized directly by point-based rendering and visualization methods.

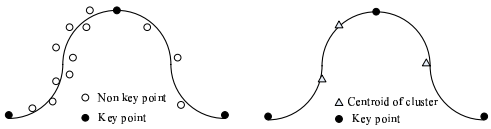
Alexa et al.[8] and Linsen et al[9] first proposed simplification algorithm for point-based models. Their algorithms belong to the class of vertex removal. Then, Pauly and Gross[10] provided a resampling strategy based on one of signal processing theory. The most related work to us is proposed by Pauly, et al [11]. They

provided four surface simplification methods which are adapted from the simplification methods for polygonal surface: incremental clustering, hierarchical clustering, iterative simplification and particle simulation.

3. Adaptive simplifi cation algorithm

3.1. Key point

Without loss of generality, we assume that a point-based surface S_c in a grid cell c can be approximated by a local surface function $f(p)$. Usually, local surface function is a polynomial function which is determined by a set of points $p_i (p_i \in c, 1 \leq i \leq n_c$, where n_c is the number of the points in a grid cell c). The point p_j ($p_j \in c, 1 \leq j \leq n_k \leq n_c$, where n_k is the number of the key points in c) is called a "Key point" if its contribution is high in the process of determining the geometric characteristics of a local surface. Figure 2 illustrates the key points(the black points) on the curve in a 2D view. Key points in figure 2(a) determine the extent of the curve and the maximum value and the minimum value of the curve. If key points are preserved, most of the geometric characteristics of the curve are preserved as well in the process of simplification. Key points include three kinds of points: (i) start points and end points; (ii) stationary points; (iii) inflection points. These key points capture the skeleton of the curve or the surface. The preservation of key points decrease the difference between the original surface S_c before simplification and the corresponding surface $S_c^{'}$ after simplification.



(a) key points of the curve (b) simplifying the curve

Figure 2. Key point

3.2. Local clustering

The computation cost for finding key points is very high because local surface function f_p and its derivatives $f_p^{'}$, $f_p^{''}$ are difficult to compute. In order to improve the efficiency, we propose another approach based on several interesting observations:(1) It is not necessary to preserve inflection points, since most of the inflection points are close to the center of the clusters, as shown in Figure 2(b). (2) It is not necessary to preserve the start point and the end point for closed surfaces. (3) The coordinates of the key points achieve either a maximum or minimum value when compared with the corresponding coordinates of a number of neighboring points along an axis with small variance.(4) The data points between key points can easily be partitioned into one or several clusters along the axis with maximum variance. For example, the data points between the black points in figure 2 are easy to form clusters along the x-axis.

Based on these observations , we first calculate the eigenvectors and eigenvalues of the covariance matrix of the points in grid cell c. Then, the eigenvectors are sorted in descending order according to their corresponding eigenvalues.**Note that**, the objective here is to determine the variance of the local surface normal. Finally, all the points are projected to the new coordinate system which is formed by the eigenvectors. The following Lemma 1 and Lemma 2 are obtained by the property of key points.(We only focus on stationary points)

Lemma 1. *If the coordinate of a point p along the axis v_2 with the smallest eigenvalue has the maximum (minimum)value, p is a global maximum(minimum) point in grid cell c.*

Lemma 2. *If the coordinate of a point p along the axis v_2 with the smallest eigenvalue is greater(smaller) than that of its neighbors, p is a local maximum (minimum)point in grid cell c.*

The proofs are given in [13].

There are two ways for the algorithm to determine the key points according to the lemmas:(i) It obtains the points with the maximum or minimum value along the axis v_2 as key points directly; (ii) It considers the points one by one. If the point satisfies Lemma2, the point is a local maximum (minimum) point. Although searching for the nearest neighbor of a point is very fast, computational cost will be expensive for a large point set. In order to save computation cost, the algorithm adopts a partition strategy: (i) it calculates the difference between the maximum value MAX_{v_2} and the minimum value MIN_{v_2} along v_2. (ii) A partition threshold α ($0 \leq \alpha \leq \frac{1}{2}$) is specified. There exists two planes perpendicular to v_2 and whose intersections with v_2 are given by $MAX_{v_2} - \alpha \cdot (MAX_{v_2} - MIN_{v_2})$ and $MIN_{v_2} - \alpha \cdot (MAX_{v_2} - MIN_{v_2})$ respectively. They divide the space in the grid cell c into three parts. (iii) The algorithm clusters the points in the three parts based on another two axis v_0, v_1. **Note that**, we take the center of the clusters above the plane $v_2 = (MAX_{v_2} - \alpha \cdot (MAX_{v_2} - MIN_{v_2}))$ or below the plane $v_2 = MIN_{v_2} - \alpha \cdot (MAX_{v_2} - MIN_{v_2})$ as key points, since adopting the center of the clusters (i)alleviates the influence of noise and (ii) the centers are close to the key points if α is small. The value of α determines the number of missing key points. If α is large, the number of missing key points is small. However, in this case, the center of the clusters are not close to the key points any more and cannot achieve the objective of preserving key points. According to our experiments, $\alpha = \frac{1}{8}$ represents a suitable choice.

The algorithm is divided into six steps: (i) the mean point p_c of all the data points p_i in the grid cell c is computed by the equation $p_c = \frac{\sum_{i=1}^{n_c} p_i}{n_c}$ (n_c is the number of points in a grid cell c). (ii) A 3×3 covariance matrix C_M is calculated by the equation $C_M = Cov(axis\ k, axis\ l) = \frac{\sum_{i=1}^{n_c}(p_{ki}-p_{kc})(p_{li}-p_{lc})}{n_c}$, where $axis\ k, axis\ l \in \{x-axis, y-axis, z-axis\}$. ($iii$) the eigenvectors and eigenvalues of the covariance matrix C_M are determined by the equation $C_M \cdot v_k = \lambda_k \cdot v_k$ ($0 \leq k \leq 2$). (iv) All the points are transformed to the new coordinate system defined by the eigenvectors v_0, v_1, v_2. (v)The points are partitioned into three parts by the above partition strategy. (vi)The points in the top and bottom parts are clustered as key points, while the points between the key points are clustered by their values along v_0, v_1. The centroids of the clusters become the new sample points.

Figure 3 illustrates this local clustering approach based on PCA using a 2D view. Figure 3(a) shows the original data points,

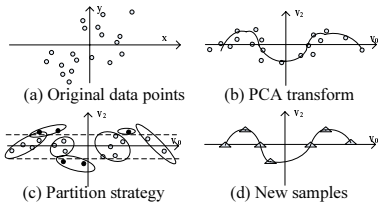


Figure 3. Local clustering

while Figure 3(b) illustrates the change after PCA. The points in Figure 3(c) are partitioned into three parts by two dashed lines ($v_2 = MAX_{v_2} - \alpha \cdot (MAX_{v_2} - MIN_{v_2})$ and $v_2 = MIN_{v_2} - \alpha \cdot (MAX_{v_2} - MIN_{v_2})$)according to the partition strategy. It is obvious that there are two clusters in the top partition, while only one cluster is in the bottom . The centers of the clusters which contain the black points are viewed as key points. Finally, all the points in the middle part are clustered between the key points along v_0. Figure 3(d) shows the new samples after clustering.

Figure 4 shows the results of the local clustering method based on PCA to simplify the model of a dragon (437646 points) to different output model size (N_s). The arrows in the top row show some key points which are preserved by the algorithm, while the arrows in the bottom row point out the change in mean curvature.

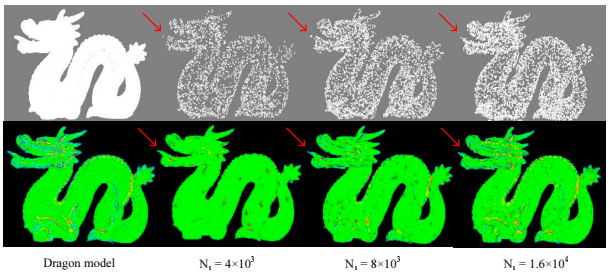


Figure 4. Dragon model

3.3. Combination with random sampling

The clustering algorithm has two major disadvantages: (i) the surface quality is low; (ii)we are unable to control the number of samples. The objective of key point preservation is to improve the quality of the surface. In order to control the number of the output samples, we combine local clustering with random sampling. First, the system gives the number of samples N_c for the grid cell c. Then, the algorithm clusters the black points in Figure 3(c).(iii) if ($N_c > C_n$)(where C_n is the number of the centers of the clusters), it selects $N_c - C_n$ samples by clustering the white points. (iv) if ($N_c < C_n$), it randomly selects N_c from C_n points.

3.4. Error analysis

We propose a novel error estimation method based on the observation that the output surface S' is similar to the original surface S which means the key points in S' are close to the corresponding key points in S. The error between S' and S is measured by the average of the distances between key points, which we refer to as the Average key point error(AKPE). We can also adopt a Maximum key point error(MKPE) as defined in Eq(2) below:

$$AKPE = \frac{1}{n_k} \sum_{k=1}^{n_k} d(p_k[i], p'_k[i]) \quad (1)$$

$$MKPE = Max_{1 \le k \le n_k} d(p_k[i], p'_k[i]) \quad (2)$$

$$d(p_k[i], p'_k[i]) = \sum_{i=0}^{2} (p_k[i] - p'_k[i])^2 \quad (3)$$

where n_k is the number of selected key points, $p_k[i]$ is the coordinate of the key point($p_k \in S$) along the ith dimension , $p'_k[i]$ is the coordinate of the corresponding key point ($p'_k \in S'$). $d(p_k[i], p'_k[i])$ is the distance measure.

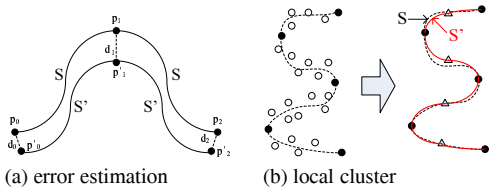


Figure 5. Error estimation

Figure 5(a) illustrates the process of calculating KPE. The algorithm first obtains the key points $\{p_0, p_1, p_2\}$ in the original surface S and the corresponding key points $\{p'_0, p'_1, p'_2\}$ in the output surface S' by the methods in section 3.2. Then, the distances (d_0, d_1, d_2)between key point pairs $(p_0, p'_0), (p_1, p'_1), (p_2, p'_2)$are computed respectively. Finally, KPE is calculated by Eqs (1) and (2).

3.5. The overview of the algorithm

The algorithm for ASM is shown in Figure 6. In order to control the total number of output samples, an evaluation function $f(c_i)$ is proposed to determine the contribution of each grid cell c_i as follows.

$$f(c_i) = \frac{\omega \cdot c_s \cdot |c_i|}{|P_s|} \quad (4)$$

where ω is the value of weight which reflects the importance of the grid cell c_i, c_s is the output model size, $|c_i|$ is the number of the points in the grid cell c_i, and $|P_s|$ is the cardinality of the point set P_s. Obviously, the value of $f(c_i)$ is proportional to $|c_i|$.

The error tolerance ϵ is given by the user which bounds the maximum key point error (MKPE)for local clustering. If MKPE is greater than ϵ in the process of local clustering, we reduce the value of α(α is the control parameter to determine the partition plane) and adjust the corresponding partition plane by the fitness function $\psi(\alpha)$

$$\psi(\alpha) = \frac{MAX_{v_2} - \frac{(MAX_{v_2} - \alpha_1 \cdot (MAX_{v_2} - MIN_{v_2})) \cdot \epsilon}{MKPE}}{MAX_{v_2} - MIN_{v_2}} \quad (5)$$

where MAX_{v_2} and MIN_{v_2} are the maximum and minimum value along the eigenvector v_2 with the smallest eigenvalue respectively, and α_1 is the value of α before adjusting. Maximum

key point error $MKPE$ is then computed by Eq (2). **Note that**, it is obvious that the key point with maximum key point error is the global maximum point or the global minimum point in the grid cell C_i. As a result, the algorithm first calculates $MKPE$. If $MKPE > \epsilon$, it adjusts the value of α by Eq (5). Otherwise, it clusters other points by the approach based on PCA.

Algorithm **ASM**(Point set P, Sample number N_s, Tolerance ϵ)
/*ϵ is the maximum error tolerance */
/*$|P'_s|$ is the cardinality of the point set P after clustering */
/*$MKPE_N$ is the key point error after clustering */
1. Hashes all the points to the grid G;
2. **For** each grid cell c_i
3. Compute $f(c_i)$ by Eq(4);
4. Calculate $MKPE$ by Eq(2);
5. **If** $(MKPE > \epsilon)$
6. Compute α by Eq(5);
7. Perform local clustering;

Figure 6. The algorithm for ASM

4. Experiment

All the experiments presented are executed with a Pentium 2.8 GHz CPU with 1 GByte memory. Our datasets are obtained from the Stanford 3D scanning repository([12]), and other geometric models archives. The cardinality of the models are shown in Table 1. The time for simplifying models (the size of output model is 5% of the size of input model) and the Average key point errors(AKPE) are illustrated in Table 2

Model name	Cardinality	Output	Model name	Cardinality	Output
Human Brain	19459	973	Knee	37889	1895
Ball Joint	137063	6854	Armadillo	172975	8649
Skeleton hand	327324	16366	Dragon	437646	21882
Hip	530169	26509	HappyBuddha	543653	27183
Malaysia	1815771	90789	Asian Dragon	3609601	180481
Thai Statue	4999997	250000	Lucy	14027873	701394

Table 1. The cardinality of the models

Model name	time(s)	AKPE	Model name	time(s)	AKPE
Human Brain	0.126	7.03×10^{-5}	Knee	0.211	6.83×10^{-5}
Ball Joint	0.843	6.52×10^{-5}	Armadillo	1.421	6.51×10^{-5}
Skeleton hand	2.112	6.48×10^{-5}	Dragon	2.314	6.47×10^{-5}
Hip	3.161	6.41×10^{-5}	Happy Buddha	3.161	6.41×10^{-5}
Malaysia	8.121	6.27×10^{-5}	Asian Dragon	15.061	6.13×10^{-5}
Thai Statue	25.014	6.01×10^{-5}	Lucy	35.263	5.81×10^{-5}

Table 2. The simplification time and AKPE

Figure 7 illustrates the results of ASM when applied to simplify a ball joint model (137063 points) to different output model size (N_s). The arrows in the top row show some key points which are preserved by the algorithm, while the arrows in the bottom row point out the change in mean curvature. Blue color denotes high curvature, while green color denotes low curvature. **Note that,** the key point preservation leads to the changes in mean curvature around the key point. In general, the change of the mean curvature is determined by the distribution of the points around the key point.

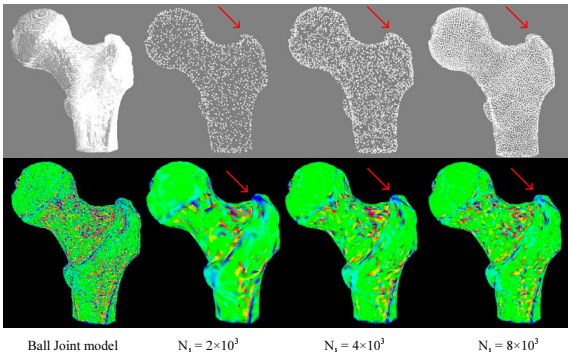


Figure 7. Ball joint model

5. Conclusion and future work

The paper investigates the problem of meshless simplification. Our contribution is an efficient simplification method which we refer to as the adaptive simplification method(ASM). ASM is based on PCA which preserves most of the key points with a small computational cost. A key point error metric is then introduced to measure the difference between the original surface and the surface after simplification. ASM guarantees a global error bound in the simplification process. In the future, it will be interesting to apply ASM to different kinds of point-based models with different surface properties.

Acknowledgments

The work described in this paper was partially supported by grants from the Research Grants Council of Hong Kong Special Administrative Region, China [Project No. CityU 1197/03E and CityU 121005] and grants from City University of Hong Kong [Project No. 7001596 and 9360091].

References

[1] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. Computer Graphics (SIGGRAPH 92 Proc.), 26(2):65-70, July 1992.

[2] Jarek Rossignac and Paul Borrel. Multi-resolution 3D approximations for rendering complex scenes. In B. Falcidieno and T. Kunii, editors, Modeling in Computer Graphics: Methods and Applications, pages 455-465, 1993.

[3] David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. In SIGGRAPH 97 Proc., August 1997.

[4] Garland, M., Heckbert, P. Surface simplification using quadric error metrics. SIGGRAPH 97, 1997.

[5] Michael Garland, Yuan Zhou. Quadric-based simplification in any dimension . ACM Transactions on Graphics (TOG), Volume 24 Issue 2.April 2005.

[6] Turk, G. Re-Tiling Polygonal Surfaces. SIGGRAPH 92,1992.

[7] Witkin, A., Heckbert, P. Using Particles To Sample and Control Implicit Surfaces. SIGGRAPH 94, 1994.

[8] Alexa,M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, T. Point Set Surfaces, IEEE Visualization 01, 2001.

[9] Linsen, L. Point Cloud Representation. Technical Report, Faculty of Computer Science, University of Karlsruhe, 2001.

[10] Pauly, M., Gross, M. Spectral Processing of Point-Sampled Geometry, SIGGRAPH 01, 2001.

[11] Mark Pauly, Markus Gross, Leif P. Kobbelt. Efficient Simplification of Point-Sampled Surfaces. Proceedings of the conference on Visualization '02.October 2002.

[12] $http://www-graphics.stanford.edu/data/3Dscanrep/$

[13] ZhiWen Yu, Hau-San Wong, "An efficient local clustering approach for simplification of 3D point-based computer graphics models", Technical report, 2006.