Fast mesh segmentation by approximated spherical patches

Zhenyu Shu[1,2], Guozhao Wang[1]

1. Institute of Computer Graphics and Image Processing, Department of Mathematics, Zhejiang University, Hangzhou 310027, China

2. Laboratory of Information and Optimization Technologies, Ningbo Institute of Technology, Zhejiang University, Ningbo, 315100, China

E-mail: littlerain_szy@sohu.com

Abstract: In this paper, we present a fast method for segmentation of triangular mesh surfaces into simple patches, where each patch is an approximation of a part of spherical surface. Given a mesh surface, all triangles of it cluster to a user-specified number of patches by energy minimization. All patches of result meshes are as spherical as possible with our algorithm. Experimental results show that our algorithm is efficient and robust.

Key Words: Mesh segmentation; Spherical surface; Clustering

1 INTRODUCTION

Mesh segmentation plays an important role in geometric modeling and digital processing. It is used in numerous areas including parameterization, texture mapping, shape matching, morphing, multiresolution modeling, mesh editing, compression, animation and more. The objective of mesh segmentation is to partition a mesh into several disjoint patches whose union corresponds to original model. Usually, segmentation decomposes a mesh into some simple pieces or ones which satisfied some criteria. In [1], input meshes are decomposed into some planar patches. In [2], meshes are segmented into several kinds of simple geometric surfaces, including plane, sphere, cylinder, and etc. Spherical surfaces are commonly used in geometry modeling for its simplicity and good property, such as convex. Therefore, decomposing a given mesh surface into several spherical patches is useful in some applications.

In this paper, we present a novel method for segmentation of surface meshes into user-specified number of approximated spherical patches. The algorithm maximizes patches' sphericity by global optimizing the sphericity metrics on the mesh directly. Through it, each patch produced by our algorithm could be well approximated by spherical surfaces. Experimental results show that our method is fast and robust.

2 RELATED WORK

In recent years, many mesh segmentation methods have been proposed to match different types of application needs. According to the difference of aim, they can be categorized into two classes. The first class focuses on partitioning the object defined by the mesh into meaningful volumetric components[3-5]. The second class tries to partition the mesh into surface patches by using surface geometric properties. In [6-8], an object is decomposed into geometric primitives such as planes, cylindrical patches, spherical parts, etc. Our algorithm falls in the second class and focuses on keeping segments planar. In many applications including texture mapping [9-11], building charts [12, 13] and geometry-image creation [14], the segment patch must be topologically equivalent to a disk and must not impose large distortion after parameterization onto 2D. Additionally, in remeshing or simplification[1, 15-20], patches are usually replaced either by one or a set of planar polygons. As a sequence, planarity is the desired property of the patches in these cases.

More recently, other types of proxies have been used to replace mesh patches defining different types of patch properties for spherical, cylindrical and rolling ball blends [21, 22]. Segmentation into general quadric surfaces is often sought in CAD for reverse engineering and modeling [23, 24]. For actual reconstruction and creation of physical models and toys, strips and quasi-developable patches are built in [25-27].

A different approach present in [2] is slippage analysis. Local points and regions having similar slippable motions are clustered to form segments. In [28], the authors use symmetry analysis to segment a mesh into components. Shlafman et al.[29] use k-means clustering to segment the models into meaningful pieces. Katz and Tal [30] use fuzzy clustering on geodesic and angular distances between surface elements to obtain a hierarchical mesh decomposition with non-jagged borders. An excellent survey on these methods is made in [31].

Our method is based on iterative clustering. Unlike previous techniques (In [1], the authors use a method based on Lloyd's relaxation to drive the distortion error down), we defined a global energy which reflects the sphericity of segments and iteratively update the boundaries of segments to minimize the energy directly. Benefiting from simplification of the energy term, the computational cost of our method is very low.

3 SPHERICITY METRIC

Given a 3D triangle mesh *M*, which is defined as a tuple {*V*, *E*, *F*} of vertices $V = \{ p_i \mid p_i \in R^3, 1 \le i \le m \}$, edges

This work is supported by the National Natural Science Foundation of China Grant (No. 60773179) and the Foundation of State Key Basic Research 973 Item (No. 2004CB318000).

978-1-4244-2723-9/09/$25.00 © 2009 IEEE

$E=\{e_{ij}=(p_i,p_j)\mid p_i,p_j\in V, i\neq j\}$, and faces $F=\{f_{ijk}=(p_i,p_j,p_k)\mid p_i,p_j,p_k\in V, i\neq j, i\neq k, j\neq k\}$. We want to decompose M into n sub-meshes M_i, i = 0,1,…,n-1, $M_i\subset M$ $M_i=\{V_i,E_i,F_i\}$ $V_i\in V, E_i\in E, F_i\in F$ $F_i\cap F_j=\varnothing$, $i\neq j$, here n is a number specified by the user.

To make each patch Mi as spherical as possible, we need a metric to measure sphericity. We use the metric below to measure the sphericity of segment:

$$E_i=\int_{M_i}\rho(x)\left\|K-\overline{K}_i\right\|^2 dx, \qquad (1)$$

where K is the Gaussian curvature at vertex x on M_i, $\rho(x)$ is a density function of M_i, $\overline{K}_i$ is the average of K on M_i and can be formulated as $\overline{K}_i=\dfrac{\int_{M_i}\rho(x)Kdx}{\int_{M_i}\rho(x)dx}$.

Intuitively, if a surface S is a part of a sphere, the corresponding E_i on S equals to 0 for K remain constant on the whole surface of sphere. As a sequence, to make all patch M_i as spherical as possible, we only need to minimize the energy given as

$$E=\sum_{i=0}^{n-1}\int_{M_i}\rho(x)\left\|K-\overline{K}_i\right\|^2 dx. \qquad (2)$$

Because $\overline{K}_i$ can be rewritten as $\overline{K}_i=\dfrac{\sum_{t_j\in M_i}\int_{t_j}\rho(x)Kdx}{\sum_{t_j\in M_i}\int_{t_j}\rho(x)dx}$, if we denote the area of the triangle T_j as s_j, which can be computed as $s_j=\int_{t_j}\rho(x)dx$, and denote the unit normal of the triangle T_j as n_j, we can rewrite the energy E as follows:

$$E=\sum_{i=0}^{n-1}\left(\sum_{t_j\in M_i}s_j\left\|K_j\right\|^2-\frac{\left\|\sum_{t_j\in M_i}s_jK_j\right\|^2}{\sum_{t_j\in M_i}s_j}\right). \qquad (4)$$

Therefore, to construct segmentation which maximize the sphericity of each segment, we only need to find a clustering of all faces of M, such that it minimizes (4).

4 ALGORITHM

4.1 Clustering construction

Our algorithm is based on the construction of a clustering which minimizes (4). We propose to minimize E with an iterative algorithm that updates the clusters according to boundary tests. The algorithm is similar with [32]. As the boundaries of M_i is a subset of mesh edges, to find a proper configuration of clusters, a local test for each edge e between two different clusters is processed. Let us assume an edge e is on the boundary between two sub-meshes M_i and M_j and e has two adjacent triangles T_k and T_l belonging respectively to M_i and M_j (Fig.1). We compute E for three cases:

E: T_k belongs to M_i and T_l belongs to M_j.

E_1: both T_k and T_l belong to M_i.

E_2: both T_k and T_l belong to M_j.

Then, the case resulting in the smallest energy E is chosen, and the clusters configuration is updated. By looping all of the boundary edges between adjacent patches, the energy E are iteratively decreased. Because $E\geq 0$, the convergence of our algorithm can be guaranteed.

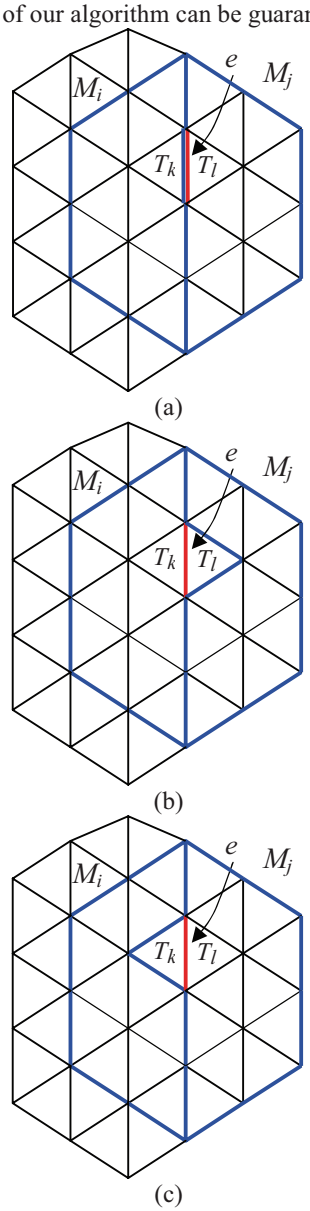

Fig.1. Compute energy E' for three cases and the case resulting maximum E' is chosen. (a) T_k belongs to M_i and T_l belongs to M_j; (b) both T_k and T_l belong to M_i; (c) both T_k and T_l belong to M_j.

4.2 Implementation details

To build an initial cluster configuration, we randomly pick n triangles T_i (i = 0, 1, …, n-1) as n initial clusters. When iteration proceeds, we loop over all the edges on the boundary of patches. For any edge e on the boundary of M_i, if one of the adjacent triangles T_k belongs to M_i and the other triangle T_l does not belong to any clusters, we add T_l to M_i directly. Else, if T_k belongs to M_i, while T_l belongs to another cluster M_j ($j \neq i$), we perform the boundary test described in section 4.1 and update the cluster configuration to decrease the energy E'.

For an efficient implementation, we cache two values $c_1 = \sum_{t_j \in M_i} s_j K_j$ and $c_2 = \sum_{t_j \in M_i} s_j$ for each patch M_i when clustering proceeding. When a triangle T_k is added to M_i or removed from M_i, two delta values $\Delta c_1 = s_k K_k$ and $\Delta c_2 = s_k$ are added to or removed from c_1 and c_2 respectively.

In practice, when clustering carries on, some patches may be separated into several disconnected parts, which is not our expectation. To avoid occurrence of this situation, we apply a heuristic constraint to each patch M_i. When a triangle T_k is removed from M_i and added to another patch M_j (Fig.2 (a)), if it will lead to the shape of M_i's boundary looking like ' ∞ ' (Fig.2 (b)), i.e. there is a vertex P which has four connected edges on the M_i's boundary, let the value of corresponding energy E' be 0. It will prevent T_k from being removed from M_i and also prevent M_i from being divided into several disconnected parts. Although we use this method each time when energy testing, the computational expense of this method is low because the boundary information has already been cached when clustering. In our experience, this step is efficient and costs only about 10% of the total clustering time. Using this method, no patch will ever fall into several disconnected parts when clustering.

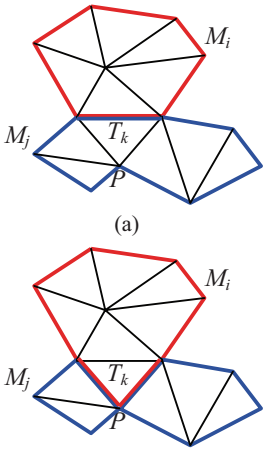

Fig.2. Preventing each segment from falling into several disconnected parts. M_i's boundary is blue and M_j's boundary is red.

5 EXAMPLES

We test our algorithm on several models: Dinosaur; Bunny; Laurent's hand. (Fig.3-5)

Table 1 shows our results using our method. The processing time was measured on a low-end AT/AT compatible PC with Intel Celeron-M CPU 1.4 GHz and 1GB RAM memory. From the experimental results, we can see our algorithm is fast and almost real time to small models. An observation from our results is that the processing time is reduced when the number of segments increases for a given model. We think the main reason is that our method is based on face clustering and the count of the iteration steps increases when segment number decreases. Another advantage of our algorithm is that users can specify the count of final segments. It is an important feature to some applications such as mesh simplification and etc. While using many other segmentation algorithms, the number of segments can not be controlled beforehand.

Our algorithm partitions input meshes into given number of spherical segments. On some meshes, minimizing our energy leads the sharp edges on the model to be the common boundaries of neighbor patches. But in some cases, the common boundaries of segments are not as smooth as we expected. It is a defect of our method, although we can use a post-process to improve the smoothness of the boundaries. We will address this issue in the future.

Table 1 Our results obtained for reference models

Fig	Model	Face number	Segment number	Time (sec)
Fig.3	Dinosaur	5,094	20	0.25
Fig.4	Bunny	69,674	100	7.029
Fig.5	Laurent's hand	99,999	20	53.17

6 CONCLUSION

In this paper we present an efficient algorithm for mesh segmentation, which is suitable for meshes with arbitrary topology. By introducing an appropriate measure of sphericity, we convert the mesh segmentation to an energy minimization problem and solve it by a fast face clustering method. Experiments show that our algorithm is fast, effective and robust. In future work, we plan to improve the performance of the algorithm further. And also, we will investigate smoothing of the boundaries if it is needed in some cases.

Fig.3. Dinosaur model with 5094 faces, which is partitioned to 20 segments.

Fig.4. Bunny model with 69674 faces, which is partitioned to 100 segments.

Fig.5. Laurent's hand model with 99999 faces, which is partitioned to 20 segments.

ACKNOWLEDGEMENTS

The authors thank Dr. Ligang Liu for his suggestion. The Laurent's hand model was provided courtesy of INRIA by the AIM@SHAPE Shape Repository. The Bunny model was provided courtesy of Stanford 3D Scanning Repository.

REFERENCES

[1] Cohen-steiner D, Alliez P, Desbrun M. Variational shape approximation[C]. Los Angeles, California: ACM, 2004.
[2] Gelfand N, Guibas L J. Shape segmentation using local slippage analysis[C]. Nice, France: ACM, 2004.
[3] Reniers D, Telea A. Hierarchical part-type segmentation using voxel-based curve skeletons[J]. The Visual Computer. 2008, 24(6): 383-395.
[4] Lai Y K, Hu S M, Martin R R, et al. Fast mesh segmentation using random walks[C]. Stony Brook, New York: ACM, 2008.
[5] Mortara M, Patan G, Spagnuolo M. From geometric to semantic human body models[J]. Computers & Graphics. 2006, 30(2): 185-196.
[6] Varady T, Martin R R, Cox J. Reverse engineering of geometric models--an introduction[J]. Computer-Aided Design. 1997, 29(4): 255-268.
[7] Shah J J, Anderson D, Kim Y S, et al. A discourse on geometric feature recognition from CAD models[J]. J. Comput. Info. Sci. Eng. 2001, 1(1): 41-51.
[8] Ip C Y, Regli W C. Manufacturing Classification of CAD Models Using Curvature and SVMs[C]. IEEE Computer Society, 2005.
[9] Sander P V, Snyder J, Gortler S J, et al. Texture mapping progressive meshes[C]. ACM, 2001.
[10] Sorkine O, Cohen-or D, Goldenthal R, et al. Bounded-distortion piecewise mesh parameterization[C]. Boston, Massachusetts: IEEE Computer Society, 2002.
[11] Zhang E, Mischaikow K, Turk G. Feature-based surface parameterization and texture mapping[J]. ACM Trans. Graph. 2005, 24(1): 1-27.
[12] Levy B, Petitjean S, Ray N, et al. Least squares conformal maps for automatic texture atlas generation[J]. ACM Trans. Graph. 2002, 21(3): 362-371.
[13] Zhou K, Synder J, Guo B, et al. Iso-charts: stretch-driven mesh parameterization using spectral analysis[C]. Nice, France: ACM, 2004.
[14] Sander P V, Wood Z J, Gortler S J, et al. Multi-chart geometry images[C]. Aachen, Germany: Eurographics Association, 2003.
[15] Kalvin A D, Taylor R H. Superfaces: Polygonal Mesh Simplification with Bounded Error[J]. IEEE Comput. Graph. Appl. 1996, 16(3): 64-77.
[16] Garland M, Willmott A, Heckbert P S. Hierarchical face clustering on polygonal surfaces[C]. ACM, 2001.
[17] Sheffer A. Model simplification for meshing using face clustering[J]. Computer-Aided Design. 2001, 33(13): 925-934.
[18] Zuckerberger E, Tal A, Shlafman S. Polyhedral surface decomposition with applications[J]. Computers & Graphics. 2002, 26(5): 733-743.
[19] Boier-martin I M. Domain decomposition for multiresolution analysis[C]. Aachen, Germany: Eurographics Association, 2003.
[20] Inoue K, Itoh T, Yamada A, et al. Face clustering of a large-scale CAD model for surface mesh generation[J]. Computer-Aided Design. 2001, 33(3): 251-261.
[21] Jianhua W, Leif K. Structure Recovery via Hybrid Variational Surface Approximation[J]. Computer Graphics Forum. 2005, 24(3): 277-284.
[22] Attene M, Falcidieno B, Spagnuolo M. Hierarchical mesh segmentation based on fitting primitives[J]. The Visual Computer. 2006, 22(3): 181-193.
[23] Petitjean S. A survey of methods for recovering quadrics in triangle meshes[J]. ACM Comput. Surv. 2002, 34(2): 211-262.
[24] Lavou G, Dupont F, Baskurt A. A new CAD mesh segmentation method, based on curvature tensor analysis[J]. Computer-Aided Design. 2005, 37(10): 975-987.
[25] Mitani J, Suzuki H. Making papercraft toys from meshes using strip-based approximate unfolding[J]. ACM Trans. Graph. 2004, 23(3): 259-263.
[26] Dan J, Kraevoy V, Sheffer A. D-Charts: Quasi-Developable Mesh Segmentation[J]. Computer Graphics Forum. 2005, 24(3): 581-590.
[27] Shatz I, Tal A, Leifman G. Paper craft models from meshes[J]. The Visual Computer. 2006, 22(9): 825-834.
[28] Podolak J, Shilane P, Golovinskiy A, et al. A planar-reflective symmetry transform for 3D shapes[C]. Boston, Massachusetts: ACM, 2006.
[29] Shymon S A. Metamorphosis of Polyhedral Surfaces using Decomposition[J]. Computer Graphics Forum. 2002, 21(3): 219-228.
[30] Katz S, Tal A. Hierarchical mesh decomposition using fuzzy clustering and cuts[J]. ACM Trans. Graph. 2003, 22(3): 954-961.
[31] Shamir A. A survey on Mesh Segmentation Techniques[J]. Computer Graphics Forum. 2008, 27(6): 1539-1556.
[32] Valette S, Chassery J. Approximated Centroidal Voronoi Diagrams for Uniform Polygonal Mesh Coarsening[J]. Computer Graphics Forum. 2004, 23(3): 381-389