Appl. Math. J. Chinese Univ.
2014, 29(4): 468-480

A fast and efficient mesh segmentation method based on improved region growing

YANG Fei[1] ZHOU Fan[1,*] WANG Ruo-mei[1,*] LIU Li[2]

LUO Xiao-nan[2]

Abstract. Mesh segmentation is one of the important issues in digital geometry processing. Region growing method has been proven to be a efficient method for 3D mesh segmentation. However, in mesh segmentation, feature line extraction algorithm is computationally costly, and the over-segmentation problem still exists during region merging processing. In order to tackle these problems, a fast and efficient mesh segmentation method based on improved region growing is proposed in this paper. Firstly, the dihedral angle of each non-boundary edge is defined and computed simply, then the sharp edges are detected and feature lines are extracted. After region growing process is finished, an improved region merging method will be performed in two steps by considering some geometric criteria. The experiment results show the feature line extraction algorithm can obtain the same geometric information fast with less computational costs and the improved region merging method can solve over-segmentation well.

§1 Introduction and notation

With the fast development of advanced modeling and visualization techniques, 3D models have been a more and more popular type of digital media. Highly detailed models can be easily acquired by the 3D laser scanning system and the techniques for efficient and robust processing of these models have become an active research area in computer graphics and digital geometry processing. Mesh segmentation is one of the important issues in digital geometry processing. Mesh segmentation can extract shape properties and structure characteristics from 3D Models. It is also an important step towards model understanding. Many applications can benefit from the mesh segmentation. In computer graphics, segmentation provides great prospect for mesh simplification[5], texture mapping[19], 3D shape retrieval[31], morphing[28], multiresolution modelling[6], geometry compression[12], collision detection[20], animation[13] and more. In these applications, an reliable mesh segmentation method is very important and can have a great influence on the results.

Received: 2013-10-21.
MR Subject Classification: 35B35, 65L15, 60G40.
Keywords: mesh segmentation, feature line, region growing, dihedral angle.
Digital Object Identifier(DOI): 10.1007/s11766-014-3240-0.
Supported by the National Natural Science Foundation of China (61272192,61379112) and the NSFC-Guangdong Joint Fund (U1135003).

According to the different purposes of mesh segmentation, mesh segmentation can be further classified into two categories: facet segmentation and component segmentation. The aim of the facet segmentation is to decompose a 3D mesh into regions and each region is homeomorphic to an open disk on the plane. The task of component segmentation is to partition a 3D mesh into a set of disjoint and meaningful parts which also keep consistent to the original model.

It is documented in various literatures about mesh segmentation, however there are still some problems, for example, the computation cost and the over-segmentation problem. In this paper, we present an improved mesh segmentation method based on region growing. The main contribution of our method is that we propose an improved region merging method to solve the over-segmentation problem well, at the same time, we present a feature line extraction algorithm based on dihedral angle and obtain the same geometric information with less computational cost. Compared with other algorithms, our algorithm has very good advantages: high efficiency, feasibility application and more accurate and robust.

§2 Related work

In the past few years, a lot of 3D mesh segmentation algorithms have been proposed. Garland, et al. presented a face clustering method which extended from Quadric error Metric[7] to partition a mesh into planar elements[8]. Shlafman, et al. proposed a method that uses K-means to cluster faces into meaningful parts[28]. Attene, et al. described a hierarchical face clustering algorithm based on fitting primitives[1]. Mangan, et al. presented a watershed algorithm by making use of the mean and Gaussian curvatures[23]. Zhang, et al. proposed a simple region growing algorithm to segment different regions based on curvatures[30]. Yamauchi, et al. proposed a simple scheme for clustering mesh normals[29]. Lee, et al. presented an intelligent scissoring operator for meshes which supports both automatic segmentation and manual cutting[18].

Other methods include core extraction[14], spectral clustering[21], random walks[16], random cuts[9], leaning-based[11] and chipper[22]. The method of core extraction method transforms the vertices of the mesh into a pose-invariant representation using Multi-dimensional scaling(MDS). The method of spectral clustering applies spectral graph theory to solve the clustering of the eigenvectors of a symmetric affinity matrix. The method of random walks presents both an interactive and an automatic method of mesh segmentation based on random walks. The method of random cuts randomizes mesh segmentation algorithms to produce a function that captures the probability that an edge lies on a segmentation boundary (a cut) and to produce a ranked set of the most consistent cuts based on how much cuts overlap with others in a randomized set. The method of leaning-based presents a data-driven approach to simultaneous segmentation and labeling of parts in 3D meshes. The method of chipper partitions a 3D mesh into smaller parts so that each part can be printed by 3D printer and then be assembled to form the original mesh. A survey of segmentation techniques can be found in references [2,16].

Most previous methods decompose models based on some geometric criteria such as curvature, geodesic distances, concavity and dihedral angles. The choices of constraints and the criteria for deciding which elements belong to the same segment are the most important factors affecting the result. For most of these algorithms, the number of segments is determined by users. Only a few methods [14,27] can determine the segment number automatically. The method proposed by Kalogerakis, et al. is different from others that it requires some labeled training meshes[11]. Some researches have already considered to extend region growing method

from image segmentation to 3D mesh segmentation[30,17]. However, a serious problem of over-segmentation still exist. The quality and speed of segmentation method need to be researched in detail.

Our method is an extension of the region growing method. Dihedral angle of each non-boundary edge is used to detect the sharp edges to extract the feature lines. This feature line extraction algorithm can obtain the same geometric information fast with less computational cost. In view of some geometric criteria, an improved region merging method is reported. This region merging method has two steps to merge the regions. The experimental results show that our method has good robustness and adaptability which can settle the over-segmentation problem neatly.

§3 Mesh segmentation based on improved region growing

During past years, region growing method has been applied to mesh segmentation. Region growing is an iterative process and usually begins from a set of seeds that elements have similar attributes to a seed are classified into the same region. In general, region growing methods have something in common. First of all, some geometrical features of mesh need to be detected. For example, Lavoué, et al. used sharp edges and principal curvatures as the geometrical features[17]. The Gaussian curvature is estimated for each vertex on the surface[30]. Gaussian curvature and concaveness are regarded as the geometrical features[3]. Next, a region growing process is performed to cluster the similar elements into the same region. Finally, in order to reduce the over-segmentation problem which is generated in this step, a region merging operation must be performed in the next step. From the view of the former research, it should be worthy to pay more attention that region merging process plays a decisive part in mesh segmentation and affects the final result of the algorithm. The previous researchers have considered to overcome this difficulty. For instance, Chen and Georganas proposed a simple yet efficient region merging algorithm with two criteria[3]. One criterion is the region size, another one is the boundary length of the adjacent region.

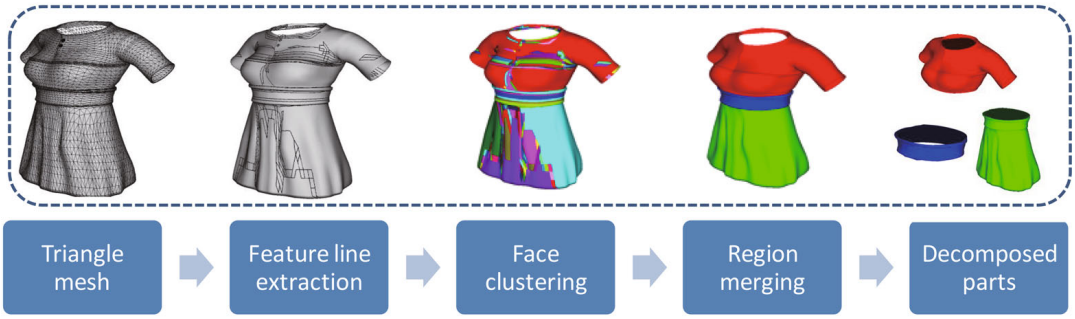

Figure 1: The flow diagram of our region growing method.

Our method improves the original method of region growing and mainly includes three steps. Figure 1 shows the flow diagram of our improved region growing method. firstly, the dihedral angle of each non-boundary edge is computed and sharp edges are detected for feature line extraction. That can obtain the same geometric information fast with less computational cost. Secondly, the faces are classified into different clusters by region growing method. Thirdly, a

novel region merging algorithm is processed that similar regions are merged. Our region merging method can solve the over-segmentation problem well by eliminating redundant feature lines and preserving the cutting contours which corresponding to real natural boundaries of the model.

3.1 Feature line extraction based on Dihedral angle

Generally, the curvature of vertex is a commonly-used feature in mesh segmentation algorithms. But, only taking curvature information into consideration does not contain all of the situations which may lead to a failure when there are concave vertexes on the mesh(see Figure 2). But the calculation of concave vertices enlarges the work remarkably especially when the model has a high resolution that you have to enlarge the normal 1-ring neighborhood to an extended Multi-Ring (XMR) neighborhood[3].

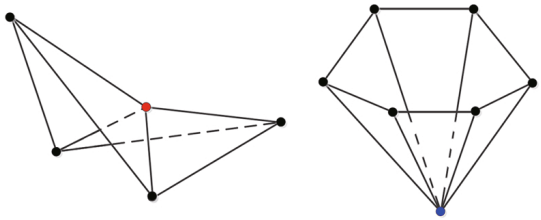

Figure 2: The red vertex is hyperbolic and the blue one is concave.
Two kinds of vertices are need to be considered[3].

Differently, to describe the geometric behavior of a surface,we just pay attention to dihedral angle. For a given edge which is a non-boundary edge, we can get the dihedral angle between two adjacent faces f_i and f_j.

$$\theta(f_i, f_j) = \pi - \arccos(N_i * N_j), \tag{1}$$

Where N_k is the normal to face f_k.

Now, we can define a sharp edge as follows: an edge is a boundary edge or shared by two adjacent triangles and its dihedral angle is smaller than a given threshold α. In our experiment, we have found $\alpha = 3.12$ for the general case, a larger value for high-resolution models and a small value for coarse models. It's important to note, however, unlike CAD models, only concave edges need to be considered for graphical models. Because we find that CAD models have many sharp features and are more sensitive than graphical models in terms of convex edges. It is common to see the dividing lines are composed of concave edges for graphical models.After all sharp edges are found, isolated and discontinuous sharp edges need to be eliminated, and then feature lines are extracted.

Since only the sign of the dihedral angle is used in preprocessing stage, we believe that this simple sharp edge detection approach is sufficient for our application. Actually, it is easy to prove that the edges which connect hyperbolic or concave vertices must be sharp edges. So instead of calculating discrete curvature and the concavity and convexity of each vertex, our method obtains the same geometry information at a low cost by only taking dihedral angle into account. From figure 1, we can see that the skirt's feature lines(the black lines) are extracted and the feature lines capture the curvature information of mesh well.

3.2 Face clustering

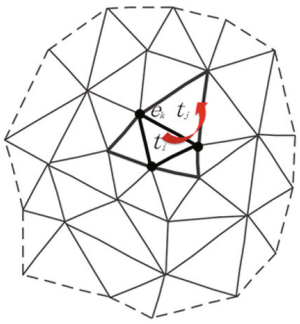

Figure 3: The face clustering process: (1)starting from the triangle t_i, the associated neighboring triangle t_j is classified into the same cluster, so does the other two adjacent triangles; (2)The same process is repeated for other triangles belonging to the cluster until sharp edges(dotted line) are encountered.

After all sharp edges are found and feature lines are extracted, a region growing operation is performed as follows. We start from a triangle t_i that has not classified into a cluster, then a new region is created, containing this triangle, associated with a new label L. Then a recursive process extends to this cluster: for each triangle t_j belonging to the cluster, for each non-sharp edge e_k of this triangle, we consider the opposite neighboring triangle to integrate into the region and assign the same label L. This recursive process will be terminated until the grown region is surrounded by sharp edges(see Figure 3). This growing algorithm is repeated for every other triangle that is still unlabelled.

With this process, all triangles are assigned a region label. Then, extra sharp edges which connect two triangles and have the same label need to be removed. This is an essential step at this point, because the region border will be used in the next stage.

3.3 Improved region merging

There is no doubt that the over-segmentation problem is inevitable in previous step due to small details and noise on the polygonal surface. Furthermore, the over-segmentation problem has a negative affect on the final segmentation results. So, how to solve the over-segmentation problem and preserving the cutting contours that match human perception is the most important in region merging stage. We have known from the previous research achievements based on the region growing algorithm that the over-segmentation problem haven't been solved very well, even to be discounted or ignored.

In this section, we propose a novel and efficient region merging algorithm. The region growing stage proceeds in two steps(see Figure 4). Before we start to introduce the specific algorithm. A few notations are given.

Definition 3.1. An *EdgeSection* is composed of edges which are shared by two different clusters.

Definition 3.2. For a given cluster with area s and perimeter c, we define the planarity ρ of the cluster which measures the ratio of its squared perimeter c^2 to its area s.

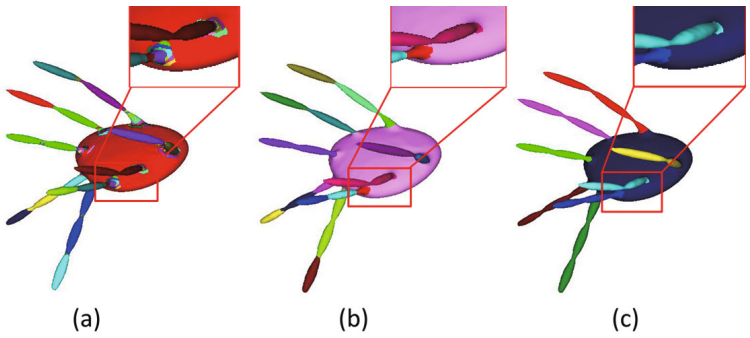

Figure 4: The procedure of region merging algorithm contains two steps: (a)The mesh after face clustering; (b)A large number of small clusters are merged with other clusters; (c)Both planarity and small regions are estimated.

$$\rho = \frac{c^2}{4\pi s} \tag{2}$$

As we consider that the size of initial clusters are considered small, so, at first, a large number of small clusters will be merged into large clusters by some simple criterions. Given the number of segmentations k by users, in the first step, the number of clusters n will be reduced to a value grater than k(for example, we use $2k$ in our experiment)by choosing the cluster iteratively with the smallest area to merge with a neighboring cluster. In every iteration step, once the cluster is selected, we first find the cluster's all *EdgeSections*. Our goal is to assign a *EdgeSection* ES_i's probability be P_{ij} (relative to *EdgeSection* ES_j) of being removed to merge the two adjacent clusters. Let $Length(ES_i)$ be the length of *EdgeSection*, and $Avg(\theta_i)$ be the average dihedral angle of *EdgeSection*. We define P_{ij} as follows:

$$P_{ij} = \beta\left(\frac{Length(ES_i)}{Length(ES_i) + Length(ES_j)}\right) + (1-\beta)\left(\frac{Avg(\theta_i)}{Avg(\theta_i) + Avg(\theta_j)}\right) \tag{3}$$

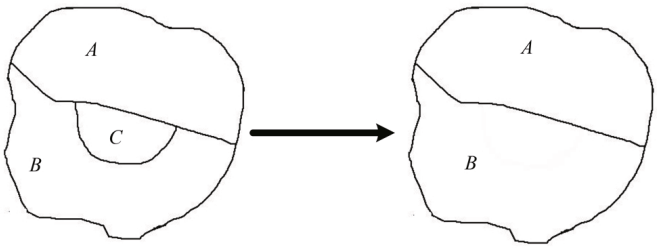

Figure 5: A selected region will be merged with its adjacent region that has both long and flat EdgeSection. In this figure, region C merges with region B.

We can see this merging process in Figure 5. The principle of this metric is straightforward.

If *EdgeSection* ES_i is longer and more flat than *EdgeSection* ES_j, the probability of removing ES_i is larger than the probability of removing ES_j. We also find that the average angles of the *EdgeSection* is more important than the length of the *EdgeSection*. So, we set $\beta = 0.1$ in our experiment and acquire a good result. Finally, we can get $P_{ji} = 1 - P_{ij}$ and $0 < P_{ij} < 1$. After each region merging, we update the neighbor information of associated clusters.

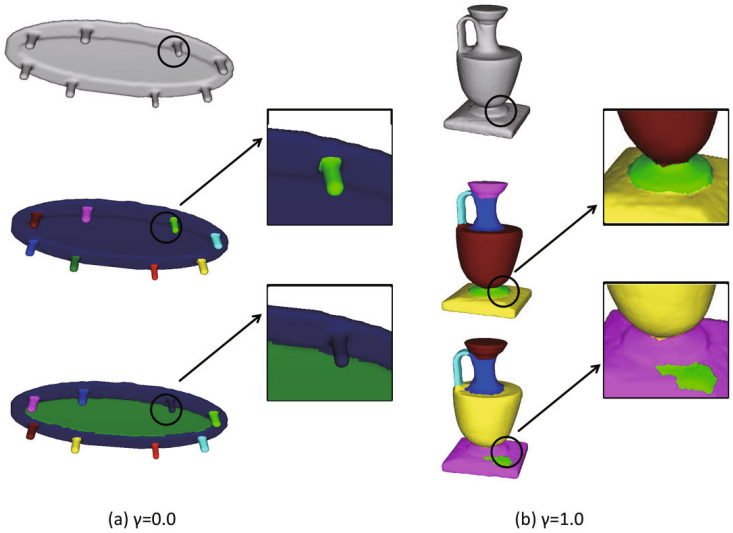

Figure 6: Both of area and planarity are needed to be considered in this merging process, and the role of them can be seen by comparing two segmentation results for two models by different coefficients: (a)Segmentation of table with $\gamma = 0.0$(bottom); (b)Segmentation of vase with $\gamma = 1.0$(bottom)

When small regions are merged into large regions, both planarity and small regions will be estimated. At this stage, for any two clusters C_i and C_j, we define a probability $P_j(C_i)$ that cluster C_i will be selected to combine with other clusters relative to C_j as follows:

$$P_j(C_i) = 1 - \gamma \left(\frac{e^{-\rho_i}}{e^{-\rho_i} + e^{-\rho_j}} \right) - (1 - \gamma) \left(\frac{Area(C_i)}{Area(C_i) + Area(C_j)} \right), \tag{4}$$

where $Area(C_i)$ is the area of cluster C_i and γ represents the weight of planarity. We find when $\gamma = 0.6$, the results are better. The planarity represent the ratio of the squared perimeter to area. The perimeter consists of the length of cluster's non-boundary edges and the area is composed of the area of cluster's each triangle. We can easily know that a circle will have planarity $\rho = 1$ and larger values of ρ correspond to more planar and irregular regions in a general way. In our experiment, we find that almost all of redundant clusters are planar and irregular. So these clusters need to be selected to merge with other clusters and the planarity of finally segments will be small relatively. The role of these factors can also be seen in Figure 6. Although some models use one of these factors can obtain a good segmentation results as well, both of them need to be considered for general cases. From Figure 6, we can see that the effect of the area and planarity, and the error segmentation results by lacking any of them. One of table's leg is selected to merged with other region by only considering area factor, because

it has a smaller area. Instead, the bottom of table is flatter than the leg. So the planarity factor cannot be ignored for this table. The vase model is partitioned wrongly by only taking planarity factor into account. The area of the region is also important in merging process for it. So these two factors can be combined to get a satisfactory result. In this stage, the number of clusters will be reduce to k which is given by user. In each iteration process, once the cluster is chosen, the following process is the same as the last step that we choose a proper adjacent cluster to merge.

After the region merging process listed above is finished, a large number of small and irregular clusters are removed and segments are generated.

§4 Experimental results

In this section, we evaluate our segmentation method on the Princeton segmentation benchmark[4]. The benchmark provides both human-generated and algorithm-generated segmentations. The Princeton segmentation benchmark evaluates segmentation algorithms against human-generated segmentations on 19 categories of models and each category has 20 models. The benchmark also provides several measures to compare these algorithms like Cut Discrepancy[10], Hamming Distance, Rand Index[25] and Consistency Error[24]. According to different people, the number of segmentations of model is different. So, in order to remove abnormal segmentations, we select the number, a choice for most people, as the labeling number, and choose one of these segmentations to be used as the ground-truth segmentation. One snapshot of segmentation result from applying our approach to one model from every category is shown in Figure 7. From Figure 7, we can see that the results of our algorithm match human perception well with cutting contours along the geometric features.

Figure 7: Segmentation results produced by applying our approach on the Princeton segmentation benchmark[4]. We can see that the results of our algorithm match human perception well with cutting contours along the geometric features.

We have tested our segmentation algorithm and made a comparison with previous methods on the entire 3D Segmentation Benchmark dataset. Table 1 shows a comparison of the performance of the three algorithms for each object category and lists the detailed Rand Index

Table 1: The result of Rand Index scores for Randomized Cuts, Random Walks and Region Growing(our method) and smaller scores suggest better segmentation results.

Model Categories	Number of models	Randomized Cuts	Random Walks	Region Growing
Human	16	1.32	4.46	1.46
Cup	19	3.41	6	0.89
Glasses	20	1.05	6.8	1.16
Airplane	20	2.29	5.27	1.27
Ant	20	0.27	1.19	0.43
Chair	20	2.94	2.75	1.02
Octopus	20	1.28	1.72	0.48
Table	20	8.2	2.06	0.25
Teddy	20	0.61	2.36	0.56
Hand	20	1.37	3.87	1.56
Plier	20	1.85	4.28	3.38
Fish	20	5.93	7.65	1.9
Bird	18	1.34	4.36	1.58
Armadillo	18	1.19	1.78	1.62
Bust	19	2.59	4.53	3.01
Mech	20	4.55	2.94	0.23
Bearing	17	0.44	3.49	0.07
Vase	17	1.38	4.35	1.07
Fourleg	20	3.79	5.51	3.57
Average	-	0.125	0.197	0.069

scores for each category. Note that we have ignored some models that the human-generated number of segments can not be distinguished apparently. We mainly uses Rand Index measure to compare segmentations. Rand Index measures the probability that two triangles are either in the same segment in two segmentations, or in different segments in both segmentations. A more detail information about Rand Index can be referred to reference [4].

We can also obtain more details about the comparative results for each category from Figure 8. The performance of our method tends to stable relative to other two algorithms. For the vast majority of categories, our region growing method is superior to the other two algorithms. We give more segmentation results for different models in Figure 9. Figure 9 provides a visual comparison with other methods for Chair, Fish, Bird, Hand, Human. Among the given methods, Region Growing, Randomized Cuts and Random Walks, we can see that our proposed Region Growing algorithm achieves the best performance, significantly outgoing the other two.

As described in Sections 3.1 and 3.3, dihedral angle has been devised to efficiently extract the feature lines and the region merging method solve the over-segmentation problem well. All our experiments are run on a commodity PC with Intel Pentium 2.90GHz processor and 2GB RAM. Table 2 lists the mesh information and the computing time for the models presented in Figures 4, 6 and 9. In the view of the timings, we can see that the performance of our algorithm in the step of preprocessing and feature line extraction is quite good. The time of this step is no more than 0.5s. However, the average time of computing the principal curvature takes about 1.72s (respectively, Chair(2.62s), Fish(0.93s), Bird(0.75s), Hand(1.40s), Table(1.59s), Vase(2.47s), Octopus(1.36s)), and that doesn't include the costs of computing the concavity and convexity of vertices in Multi-Ring(XMR) neighborhood environments. The process of face clustering is also effective and the average of time is about 0.5s. Because of taking more factors into account

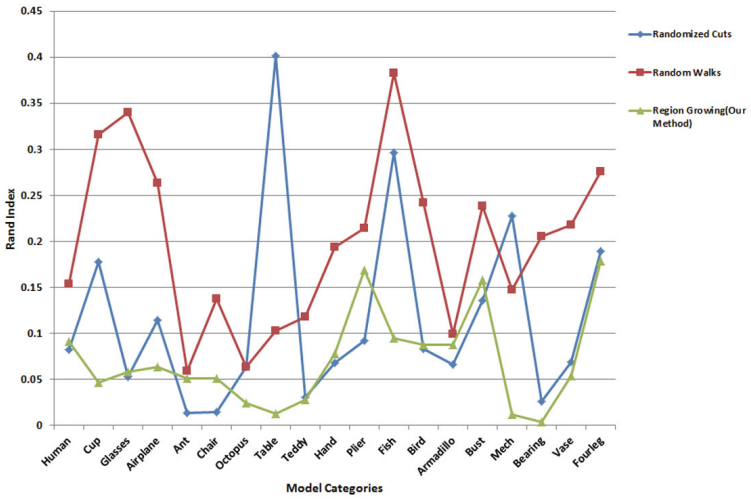

Figure 8: The analysis of average Rand Index score for each categories.

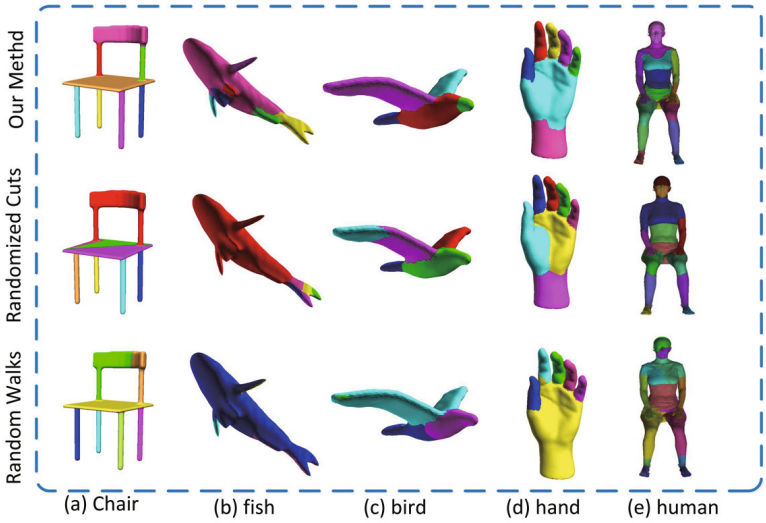

Figure 9: A visual comparison with other methods. Top row: results of our method. Middle row: results of Randomized Cuts. Bottom row: results of Random Walks.

Table 2: Statistics of the different steps of our segmentation for models presented in Figures 4, 6 and 9.

Model (Figure)	# of vertices	# of triangles	# of segments	preprocessing and feature line extraction times (s)	face clustering times (s)	region merging times (s)	total time(s)
Chair(9(a))	14372	28744	8	0.30	0.54	3.52	4.36
Fish(9(b))	10186	20368	7	0.16	0.37	12.3	12.83
Bird(9(c))	5054	10104	5	0.12	0.19	1.74	2.05
Hand(9(d))	8647	17290	7	0.21	0.33	4.98	5.52
Human(9(e))	15223	30450	15	0.37	0.62	9.66	10.65
Table(6(a))	9802	19600	9	0.21	0.37	5.73	6.31
Vase(6(b))	14476	28952	6	0.36	0.55	11.07	11.98
Octopus(4)	9124	18244	9	0.22	0.35	3.03	8.60

in our region merging algorithm, the time of this process is longer than other two steps, but it is worthwhile for our method. Even so, the processing speed of our method is also competitive.

§5 Conclusion

In this paper, we have presented a simple and efficient segmentation method by region growing. Instead of using Gaussian curvature and concaveness estimation as 3D feature extraction, we simply consider dihedral angle as the geometric criteria. After faces are clustered into different classes, we design a novel region growing algorithm to solve the over-segmentation problem. The experimental results are satisfactory, and the method is easy to implement and also sufficiently efficient to be useful in applications.

In the future, we intend to explore extension of our method to Spectral Clustering[21]. The bottleneck of Spectral Clustering is inefficiency both in time and memory usage. The computations in Spectral Clustering method are dominated by solving the eigenvectors and eigenvalues which cannot be processed when the number of faces is very large. According to the analysis, we can cluster the faces into classes at first in order to reduce the dimension of affinity matrix. Then this particular problem can be solved easily.

In recent years, dominant-quad remeshing has become a hot research field. But adaptive mesh resolution is difficult to be achieved in conforming quad meshes[15]. So, we can apply our segmentation method to dominant-quad remeshing that the size of the faces can adapt to the local curvature.

References

[1] M Attene, B Falcidieno, and M Spagnuolo. *Hierarchical mesh segmentation based on fitting primitives*, Visual Comput, 2006, 22: 181-193.

[2] M Attene, S Katz, M Mortara, G Patane, M Spagnuolo, and A Tal. *Mesh Segmentation−A Comparative Study*, Proc IEEE Int Conf Shape Model Appl, 2006, 7-18.

[3] L J Chen, N D Georganas. *An efficient and robust algorithm for 3D mesh segmentation*, Multimedia Tools Appl 29, 2006, 109-125.

[4] X B Chen, A Golovinskiy, and T Funkhouser. *A benchmark for 3D mesh segmentation*, ACM Trans Graph 28, 2009, 73: 1-12.

[5] D Cohen-Steiner, P Alliez, and M Desbrun. *Variational shape approximation*, ACM SIGGRAPH 2004 Papers, 2004, 905-914.

[6] T Funkhouser, M Kazhdan, P Shilane, P Min, W Kiefer, A Tal, S Rusinkiewicz, and D Dobkin. *Modeling by example*, ACM SIGGRAPH 2004 Papers, 2004, 652-663.

[7] M Garland, P S Heckbert. *Surface simplification using quadric error metrics*, In: *Proc 24th Ann Conf Comput Graph Interactive Tech*, SIGGRAPH '97, 1997, 209-216.

[8] M Garland, A Willmott, and P S Heckbert. *Hierarchical face clustering on polygonal surfaces*, In: *Proc 2001 Symp Interactive 3D Graph*, I3D'01, 2001, 49-58.

[9] A Golovinskiy, T Funkhouser. *Randomized cuts for 3d mesh analysis*, ACM Trans Graph 27, 2008, 145: 1-12.

[10] H Qian, B Dom. *Quantitative methods of evaluating image segmentation*, In: *Image Process, 1995. Proc Int Conf*, 3: 53-56.

[11] E Kalogerakis, A Hertzmann, and K Singh. *Learning 3d mesh segmentation and labeling*, ACM SIGGRAPH 2010 Papers, SIGGRAPH '10, 2010, 102: 1-12.

[12] Z Karni, C Gotsman. *Spectral compression of mesh geometry*, In: *Proc 27th Ann Conf Comput Graph Interactive Tech*, SIGGRAPH '00, 2000, 279-286.

[13] S Katz, A Tal. *Hierarchical mesh decomposition using fuzzy clustering and cuts*, ACM Trans Graph 22, 2003, 954-961.

[14] S Katz, G Leifman, and A Tal. *Mesh segmentation using feature point and core extraction*, Visual Comput 21, 2005, 649-658.

[15] Y K Lai, L Kobbelt, and S M Hu.*An incremental approach to feature aligned quad dominant remeshing*, In: *Proc 2008 ACM Symp Solid Phys Model*, SPM '08, 2008, 137-145.

[16] Y K Lai, S M Hu, R R Martin, and P L Rosin. *Fast mesh segmentation using random walks*, In: *Proc 2008 ACM Symp Solid Phys Model*, SPM '08, 2008, 183-191.

[17] L Guillaume, D Florent, and B Atilla. *A new cad mesh seg-mentation method, based on curvature tensor analysis*, Comput Aided Design 37, 2005, 975-987.

[18] Y J Lee, S Y Lee, A Shamir, D Cohen-Or, and H P Seidel. *Intelligent mesh scissoring using 3d snakes*, In: *Proc Comput Graph Appl, 12th Pacific Conf*, PG '04, 2004, 279-287.

[19] B Lévy, S Petitjean, N Ray, and J Maillote. *Least squares conformal maps for automatic texture atlas generation*, ACM Trans Graph 21, 2002, 362-371.

[20] X T Li, T W Woon, T S Tan, and Z Y Huang. *Decomposing polygon meshes for interactive applications*, In: *Proc 2001 Symp Int 3D graphics*, I3D '01, 2001, 35-42.

[21] R Liu, H Zhang. *Segmentation of 3d meshes through spectral clustering*, In: *Comput Graph Appl, 2004. PG 2004. Proc 12th Pacific Conf*, 2004, 298-305.

[22] L J Luo, I Baran, S Rusinkiewicz, and W Matusik. *Chop-per: Partitioning models into 3D-printable parts*, ACM Trans Graph (Proc SIGGRAPH Asia) 31, 2012, 129: 1-9

[23] A P Mangan, R T Whitaker. *Partitioning 3d surface meshes using watershed segmentation*, Vis Comput Graph, IEEE Trans 5, 1999, 308-321.

[24] D Martin, C Fowlkes, D Tal, and J Malik. *A database of human segmented natural images and its application to evaluating seg-mentation algorithms and measuring ecological statistics*, In: *Proc 8th Int Conf Comput Vision*, 2001, 416-423.

[25] W M Rand. *Objective criteria for the evaluation of clustering methods*, J Amer Statist Assoc 66, 1971, 846-850.

[26] A Shamir. *A survey on mesh segmentation techniques*, Comput Graph Forum 27, 2008, 1539-1556.

[27] L Shapira, A Shamir, and D Cohen-Or. *Consistent mesh partitioning and skeletonisation using the shape diameter function*, Visual Comput 24, 2008, 249-259.

[28] S Shlafman, A Tal, and S Katz. *Metamorphosis of polyhedral surfaces using decomposition*, Comput Graph Forum 21, 2002, 219-228.

[29] H Yamauchi, S Y Lee, Y J Lee, Y Ohtake, A Belyaev, and H P Seidel. *Feature sensitive mesh segmentation with mean shift*, In: *Proc Int Conf Shape Model Appl 2005*, SMI '05, 2005, 238-245.

[30] Y Zhang, J Paik, A Koschan, M A Abidi, and D Gorsich. *Simple and efficient algorithm for part decomposition of 3-d triangulated models based on curvature analysis*, In: *Image Process Proc 2002 Int Conf*, 2002, 3: 273-276.

[31] Z Emanoil, T Ayellet, and S Shymon. *Polyhedral surface decomposition with applications*, Comput Graph 26, 2002, 733-743.

[1] School of Information Science and Technology, Sun Yat-sen University, Guangzhou 510006, China.
[2] National Engineering Research Center of Digital Life, Guangzhou 510006, China.
Email: isszf@mail.sysu.edu.cn; isswrm@mail.sysu.edu.cn