Variational Mesh Decomposition

JUYONG ZHANG and JIANMIN ZHENG
Nanyang Technological University
CHUNLIN WU
National University of Singapore
and
JIANFEI CAI
Nanyang Technological University

The problem of decomposing a 3D mesh into meaningful segments (or parts) is of great practical importance in computer graphics. This article presents a variational mesh decomposition algorithm that can efficiently partition a mesh into a prescribed number of segments. The algorithm extends the Mumford-Shah model to 3D meshes that contains a data term measuring the variation within a segment using eigenvectors of a dual Laplacian matrix whose weights are related to the dihedral angle between adjacent triangles and a regularization term measuring the length of the boundary between segments. Such a formulation simultaneously handles segmentation and boundary smoothing, which are usually two separate processes in most previous work. The efficiency is achieved by solving the Mumford-Shah model through a saddle-point problem that is solved by a fast primal-dual method. A preprocess step is also proposed to determine the number of segments that the mesh should be decomposed into. By incorporating this preprocessing step, the proposed algorithm can automatically segment a mesh into meaningful parts. Furthermore, user interaction is allowed by incorporating the user's inputs into the variational model to reflect the user's special intention. Experimental results show that the proposed algorithm outperforms competitive segmentation methods when evaluated on the Princeton Segmentation Benchmark.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling

General Terms: Algorithms

Additional Key Words and Phrases: Shape analysis, mesh segmentation, Mumford-Shah model, Laplacian matrix, spectral attribute, primal-dual method

ACM Reference Format:

Zhang, J., Zheng, J., Wu, C., and Cai, J. 2012. Variational mesh decomposition. ACM Trans. Graph. 31, 3, Article 21 (May 2012), 14 pages.
DOI = 10.1145/2167076.2167079
http://doi.acm.org/10.1145/2167076.2167079

1. INTRODUCTION

This article considers the problem of decomposing a 3D mesh into a set of disjoint, meaningful parts. Figure 1 shows such decomposition examples where 19 mesh models taken from the Princeton Segmentation Benchmark [Chen et al. 2009], a benchmark for evaluating 3D mesh segmentation algorithms, are automatically decomposed into several parts that are depicted by various colors. Mesh decomposition is also known as mesh segmentation and mesh partitioning. It is a fundamental operation in geometry processing and computer graphics. It not only provides semantic information about the model for shape understanding and recognition, but also assists many geometric processing tasks such as skeleton extraction [Katz and Tal 2003], 3D morphing [Gregory et al. 1999], texture mapping [Lévy et al. 2002], and modeling by examples [Funkhouser et al. 2004].

The challenge with automatic mesh decomposition lies in the fact that the decomposition algorithm is expected to segment a mesh into meaningful parts which are consistent with user intention, geometric mesh attributes, and human shape perception, but the concept of "meaningful" and human perception are content dependent. In general, a good segmentation algorithm should at least be able to output the results that satisfy the following criteria. First, the elements within the same segment should have high similarity. Second, the association between different segments should be low. Geometrically, the segment boundary should be tight and smooth. Third, the segment boundary should match human perception. Based on cognitive science, the human visual system perceives segment boundaries at negative minima of the principal curvatures, which is known as the minima rule [Hoffman and Richards 1984]. Fourth, segmentation should reflect significant features and small-scale fluctuation should be ignored. In particular, the part salience theory provides three factors to determine the salience of a part: the relative size, the boundary strength, and the degree of protrusion [Hoffman and Singh 1997]. Many previous segmentation algorithms were designed to meet the criteria via two separate

This work is supported by the ARC 9/09 Grant (MOE2008-T2-1-075) of Singapore.

Authors' addresses: J. Zhang and J. Zheng (corresponding author), School of Computer Engineering, Nanyang Technological University, Singapore 639798; email: {s070051, asjmzheng}@ntu.edu.sg; C. Wu, Department of Mathematics, National University of Singapore, Singapore 119076; email: matwcl@nus.edu.sg; J. Cai, School of Computer Engineering, Nanyang Technological University, Singapore 639798; email: asjfcai@ntu.edu.sg.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2012 ACM 0730-0301/2012/05-ART21 $10.00
DOI 10.1145/2167076.2167079
http://doi.acm.org/10.1145/2167076.2167079

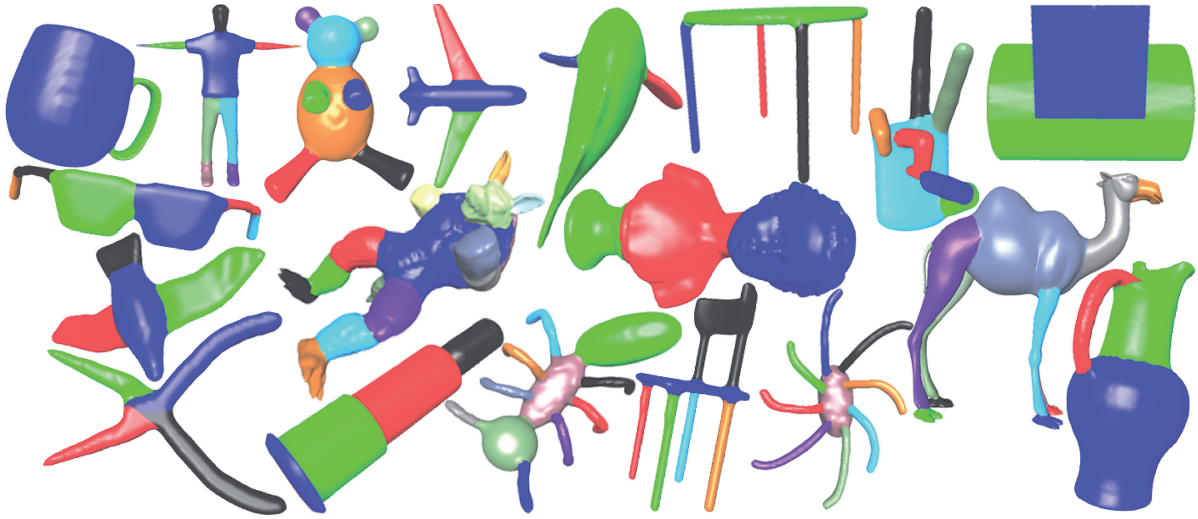


Fig. 1. Segmentation results collected from applying our proposed algorithm to the Princeton Segmentation Benchmark [Chen et al. 2009]. One mesh is shown for each category. The segmentation results match human perception well in not only the cutting boundaries but also the number of segments.

processes: segmentation and boundary smoothing, or partially satisfy the criteria through one process. For instance, the fuzzy clustering method [Katz and Tal 2003] first segments a mesh into components and then refines the boundaries in the fuzzy area, and the mesh scissoring [Lee et al. 2005] concentrates on the contours for cutting using the minima rule and part salience. For the best performance, however, all these criteria should be taken into account within one process.

The preceding observation motivates us to propose a new method for mesh decomposition. The mathematical tool at the heart of the new method is the Mumford-Shah model (M-S model) that has proven successful in image segmentation [Mumford and Shah 1989]. The piecewise constant Mumford-Shah model contains two terms: data term and regularization term. The data term measures the consistency of each segment and the regularization term measures the boundary length, which makes the model suitable for our purpose of unifying the two processes: segmentation and boundary smoothing. However, extending the Mumford-Shah model from image segmentation to 3D mesh segmentation is not trivial for two main reasons: (i) Unlike images, meshes are irregular in connectivity and sampling; and (ii) while the image intensity is directly used in the M-S model for image segmentation, it is not clear what should be used in the M-S model for mesh segmentation. The contributions of the article are as follows.

—We present a mesh segmentation algorithm using the convexified version of the Mumford-Shah model based on total variation. We extract spectral attributes from the eigenvectors of the dual graph Laplacian matrix for the mesh and adapt the Mumford-Shah model to the spectral attributes of the mesh. The spectral attributes reflect global information of the underlying mesh. The minimal rule is respected in constructing the Laplacian matrix. The Mumford-Shah model simultaneously handles the two processes: partitioning and boundary smoothing.

—We propose a fast numerical method to solve the convexified version of the Mumford-Shah model, which involves solving two subproblems. One has an explicit solution and the other is converted to a saddle-point problem that can be solved by the fast primal-dual method. Examples show that the numerical algorithm can quickly converge to the solution.

—We present a way to allow the user to interactively express his/her intention that some regions must be in the same segment. The Mumford-Shah model is modified to incorporate the user's intention.

—We also propose a method to determine the number of segments that the model should be decomposed into. By incorporating this method, the proposed Mumford-Shah mesh segmentation can automatically decompose a mesh into meaningful parts.

We test our method with various models and the experimental results show that our method is efficient and able to produce segmentation reflecting geometric attributes of the models and human perception. We also evaluate our method on the Princeton Segmentation Benchmark. Our method outperforms competitive geometry-based segmentation methods and is comparable to the learning-based segmentation [Kalogerakis et al. 2010] with a training size of 6. Figure 1 shows some segmentation results obtained automatically from our method requiring no prior information, no given number of segments, and no training.

The rest of the article is organized as follows. We review related work in Section 2 and briefly describe the Mumford-Shah image segmentation in Section 3. The main contribution of the article—the Mumford-Shah mesh decomposition algorithm—is presented in Section 4, followed by a discussion of how to incorporate users' inputs into the variational model to influence the segmentation in Section 5 and a method of determining the number of segments in Section 6. Section 7 provides experimental results and discussions. Section 8 concludes.

2. RELATED WORK

Mesh segmentation has become an active research topic in computer graphics. Many methods have been developed. Some of them borrow techniques or ideas from related fields such as image segmentation and data clustering. Comprehensive references can be found in Attene et al. [2006] and Shamir [2008].

There are different types of mesh segmentation. Our work basically focuses on segmenting a mesh into a set of disjoint parts whose union corresponds to the original mesh. Many algorithms in this category are based on clustering. For example, the K-means approach iteratively selects the representatives and performs clustering [Shlafman et al. 2002], but the boundaries between segments are often jagged or even not accurate; in Katz and Tal [2003], a fuzzy clustering is performed in a hierarchical way from coarse to fine to find fuzzy components and then the exact boundaries between the components are computed using graph cuts; unsupervised clustering techniques like the mean shift clustering are also used to segment meshes [Shamir et al. 2006; Yamauchi et al. 2005], but they often result in oversegmentation.

Other approaches include random walks [Lai et al. 2008], core extraction [Katz et al. 2005], and spectral clustering and embedding [Liu and Zhang 2004, 2007]. Using random walks on the dual graph of the mesh, Lai et al. [2008] first oversegments a mesh and then hierarchically merges segments in an order based on the relative lengths of the intersections and total perimeters of adjacent segments. Core extraction [Katz et al. 2005] transforms the mesh vertices into a pose-insensitive representation, extracts prominent feature points and then core components, and finally refines boundaries to follow the natural seams of the mesh. Spectral embedding [Liu and Zhang 2007] projects the mesh into a plane and then the outer contour of the 2D spectral embedding of the mesh is used to guide the segmentation. Spectral clustering [Liu and Zhang 2004] uses the K-means algorithm to cluster the eigenvectors of an adjacent matrix. As shown in Bardsley and Luttman [2009], the solution of this K-means segmentation algorithm is equivalent to the unregularized k-phase Mumford-Shah energy functional. Note that our work is based on the Mumford-Shah model with a regularization term which adds restrictions on the association between segments and thus produces better results in general. A very interesting work that also uses a variational approach is Variational Shape Approximation (VSA) [Cohen-Steiner et al. 2004]. VSA formulates the problem of faithful shape approximation into a variational geometric partitioning problem. While VSA uses the L^2- or $L^{2,1}$-based error functional and applies it to the mesh directly, our method uses the Mumford-Shah model and applies the model to the eigenvectors of a dual Laplacian matrix.

Recent progress includes Shape Diameter Function [Shapira et al. 2008], Randomized Cuts [Golovinskiy and Funkhouser 2008], and learning-based segmentation [Kalogerakis et al. 2010]. Shape Diameter Function [Shapira et al. 2008] is a measure of the diameter of an object's volume in the neighborhood of a point on the surface. It is computed to produce a vector for each face indicating the probability to be assigned to each of the clusters and then the graph-cut algorithm is used to refine the segmentation to minimize an energy function that combines the probabilistic vectors with boundary smoothness and concaveness. The general strategy of Randomized Cuts [Golovinskiy and Funkhouser 2008] is to randomize mesh segmentation algorithms to produce a function that captures the probability that an edge lies on segmentation boundary and to produce a ranked set of the most consistent cuts based on how much cuts overlap with others in a randomized set. Learning segmentation [Kalogerakis et al. 2010] is a data-driven approach. It formulates an objective function as a Conditional Random Field model with terms assessing the consistency of faces with labels and terms between labels of neighboring faces. The objective function is learned from a collection of labeled training meshes.

Most previous work segments a mesh based on some geometric criteria such as concavity, skeleton topology, curvature, geodesic distances, and shape diameter. These geometric quantities basically provide low-level, local geometric cues. In contrast, the learning-based segmentation method segments a mesh from a collection of labeled training meshes. It has an advantage of learning higher-level, global cues, but requires per-category training. Overall, the state-of-the-art of automatic segmentation is still far from satisfactory in terms of quality and speed, especially in the situation where the number of segments is required to be determined automatically. In fact, most of the previous methods require users to specify the segment number or training. Only a few such as Katz et al. [2005] and Shapira et al. [2008] can determine the number automatically. Accurately predicting the number of segments is not an easy task.

Our work is closely related to the Mumford-Shah image segmentation [Vese and Chan 2002] and multiclass labeling [Pock et al. 2009b; Lellmann and Schnörr 2011]. In the classic Mumford-Shah image segmentation model, the energy function is nonconvex, which causes difficulty in getting the global minima. Instead of resorting to sophisticated methods to find the minima of nonconvex problems, many efforts have been made to reformulate the energy based on TV-regularizers to produce a convex problem [Nikolova et al. 2006; Pock et al. 2009b; Lellmann and Schnörr 2011].

3. MUMFORD-SHAH IMAGE SEGMENTATION

To help understand the principle of our algorithm, this section gives a brief description of the Mumford-Shah image segmentation.

Given an image $I : \Omega \to R$ with bounded domain $\Omega \subset R^2$, the Mumford-Shah image segmentation problem is to find a partition $\Omega = \bigcup_{i=1}^{k} \Omega_i$ where Ω_i are pairwise disjoint and numbers c_i for Ω_i, which are the solution to the following optimization problem [Mumford and Shah 1989]

$$\inf_{\Omega_i, c_i} \sum_{i=1}^{k} \left(\int_{\Omega_i} (I(x) - c_i)^2 dx + \frac{\mu}{2} |\partial \Omega_i| \right), \tag{1}$$

where μ is a constant, and $\partial\Omega_i$ and $|\partial\Omega_i|$ represent the boundary and the boundary length of segment Ω_i, respectively. The basic idea of (1) is to minimize the variation within segments and the length of the boundary between segments as well. However, even if the optimal constants c_i and the number of segments have been known as a priori, in general this is still a difficult nonconvex problem.

When $k = 2$ and c_1, c_2 have been known, problem (1) can be rewritten as

$$\inf_{\Omega_1 \subset \Omega} \left\{ \int_{\Omega_1} (I(x) - c_1)^2 \, dx + \int_{\Omega \backslash \Omega_1} (I(x) - c_2)^2 \, dx + \frac{\mu}{2} (|\partial \Omega_1| + |\partial (\Omega \backslash \Omega_1)|) \right\}. \tag{2}$$

Nikolova et al. [2006] showed that problem (2) is equivalent to finding a scalar function $u(x)$ for the following convex minimization problem.

$$\min_{0 \le u \le 1} \left\{ \int_{\Omega} \{u(x)(I(x) - c_1)^2 + (1 - u(x))(I(x) - c_2)^2\} \, dx + \mu \int_{\Omega} |\nabla u| dx \right\} \tag{3}$$

Moreover, it has been shown that if $u(x)$ is a minimizer of (3), then for any $\eta \in (0, 1)$, the set $\{x \in \Omega : u(x) > \eta\}$ is the minimizer of (2), implying that the solution to (2) can be obtained by thresholding u at an arbitrary threshold between 0 and 1. While $u(x)$ can be considered to be the characteristic function for region Ω_1, $1 - u(x)$ is the one for region $\Omega \backslash \Omega_1$.

In practice, c_1 and c_2 are unknown. Note that if Ω_1 is fixed, the values of c_1 and c_2 that minimize (2) are

$$c_1 = \frac{1}{|\Omega_1|}\int_{\Omega_1} I(x)dx, \quad c_2 = \frac{1}{|\Omega\backslash\Omega_1|}\int_{\Omega\backslash\Omega_1} I(x)\,dx.$$

Thus a simple approach to finding the solution is a two-step scheme that first computes c_1 and c_2 according to these formulae and then updates Ω_1 by solving (3) [Nikolova et al. 2006].

Based on the convex formulation (3), several extensions have been developed for multiregion image segmentation [Pock et al. 2009a; Lellmann and Schnörr 2011].

4. MUMFORD-SHAH MESH DECOMPOSITION

We first introduce some notations. Assume that $M \subset \mathbb{R}^3$ is a compact triangulated surface of arbitrary topology with no degenerate triangles. The set of vertices, edges, and triangles of M are respectively denoted by $V = \{v_i : i = 0, 1, \ldots, |V| - 1\}$, $E = \{e_i : i = 0, 1, \ldots, |E| - 1\}$, and $T = \{\tau_i : i = 0, 1, \ldots, |T| - 1\}$ where $|V|$, $|E|$, and $|T|$ represent the numbers of vertices, edges, and triangles. For an edge e with two endpoints v_i and v_j, we explicitly denote it by $[v_i, v_j]$. Similarly a triangle τ whose vertices are v_i, v_j, v_k is explicitly denoted by $[v_i, v_j, v_k]$. If v is an endpoint of edge e, v is a vertex of triangle τ, or e is an edge of triangle τ, we denote these relationships by $v \prec e$, $v \prec \tau$, or $e \prec \tau$. Let $D_1(v_i)$ be the 1-disk of vertex v_i, which is the set of triangles having v_i as one of their vertices.

Our decomposition problem can be stated as follows: Given a triangular mesh M and a positive integer k, find a disjoint partitioning of M into $M_1, M_2, \ldots, M_k$ such that the elements within the same segment have high similarity and the association between different segments is low. We measure the similarity within each segment through the variance of some feature values and the association between different segments is measured by the weighted boundary length.

We now propose to solve the decomposition problem using the Mumford-Shah model. Suppose we have defined a multichannel function $\mathbf{f}(x)$ over mesh M, which, similar to the RGB function for an image, is a vector function representing some attributes of x over M and will be described in Section 4.1. The decomposition problem can be accomplished by solving the following minimization problem

$$\min_{\mathbf{u}\in K,\chi_i}\left\{\int_M \langle \mathbf{u}(x), \mathbf{s}(x)\rangle + \mu g(x)|\nabla_M \mathbf{u}(x)|d\sigma\right\}, \tag{4}$$

where K is the set of vector functions $\mathbf{u} = (u_1, \ldots, u_k)^T : M \to R^k$ satisfying that for all $x \in M$ and $i \in [1 \ldots k]$, $u_i(x) \geq 0$ and $\sum_{i=1}^k u_i(x) = 1$; $\mathbf{s}(x) = (s_1(x), \ldots, s_k(x))^T$ is a k-dimensional vector with $s_i(x) = (\mathbf{f}(x) - \chi_i)^T(\mathbf{f}(x) - \chi_i)$ indicating the affinity of x with segment M_i measured by the difference between $\mathbf{f}(x)$ and vector χ_i that is associated with M_i; $\langle , \rangle$ is the inner dot operator; the second term in (4) is a weighted multichannel total variation formulation with edge detection function $g(x)$; and μ is a trade-off factor balancing the two terms. This is the convexified version of the Mumford-Shah model. The first term of (4) is the data term that is to ensure the segmentation complying with segment coherence (i.e., small variance). If the affinity of x with segment M_i is large, $u_i(x)$ will tend to be small in order to minimize the energy functional of (4). Thus $u_i(x)$ can be viewed as the probability of x being assigned to segment M_i and $\mathbf{u}(x)$ can be used as a classification function for the segmentation. The second term of (4) is the regularization term that is to constrain the boundary between different segments to be as short as possible. We also introduce an edge detection function into the second term. The edge detection function is designed to return values between 0 and 1 (see Section 4.2 for details) and a small value corresponds to a likely edge. Including the edge detection function is to favor the segmentation along the curves where the edge detection function has small values.

The decomposition algorithm proceeds in three steps:

Step 1. Fix $\mathbf{u}$ and find the value of χ_i that minimizes (4). The value reads

$$\chi_i = \frac{\int_M u_i(x)\mathbf{f}(x)\,d\sigma}{\int_M u_i(x)\,d\sigma}$$

which corresponds to the mean of $\mathbf{f}(x)$ for segment M_i.

Step 2. Fix χ_i and solve the minimization problem (4) for $\mathbf{u}$. If the update of $\mathbf{u}$ is less than a prescribed value, go to step 3; otherwise, go to step 1.

Step 3. Once $\mathbf{u}$ is finally obtained, each vertex v is labeled by $\ell(v) = \arg\min_{i\in\{1,\ldots,k\}} \|\mathbf{u}(v) - \mathbf{e}^i\|$, where $\mathbf{e}^i = [0, \ldots, 1, \ldots, 0]^T$ with the i-th entry being 1, and is then classified into $M_{\ell(v)}$.

For completeness of the algorithm, we next need to discuss how to define an appropriate multichannel function $\mathbf{f}(x)$ for mesh M, how to discretize the continuous Mumford-Shah model (4), and how to numerically solve (4), which are elaborated next.

4.1 Multichannel Function $\mathbf{f}(x)$

Note that the success of the M-S model relies on the assumption that the underlying data are approximately piecewise constant. Also, it is revealed in Chen et al. [2009] that algorithms based on nonlocal shape features produce segmentations that most closely resemble ones made by humans. However, for 3D mesh models, the commonly used geometric quantities like curvature, vertex normal, face normal, and geodesic distance are local features and they are not necessarily piecewise constant either. As shown in Dey et al. [2003], appropriate shape features should be defined. Our approach is to define the multichannel function from the eigenvectors of a dual graph Laplacian matrix. The Laplacian matrix encodes the similarity between adjacent nodes into its entries, and its eigenvectors reflect some global information (such as spectral attributes) of the underlying mesh model [Lévy and Zhang 2010; Dey et al. 2010].

Now we define the Laplacian matrix. We choose the weight of two adjacent triangles based on their dihedral angle and whether the edge shared by the two triangles is concave or convex. Let τ_i and τ_j be two adjacent triangles of M, sharing an edge e. Their dihedral angle is denoted by $dih(\tau_i, \tau_j)$. We define $d_1(\tau_i, \tau_j)$ to measure the difference of τ_i and τ_j

$$d_1(\tau_i, \tau_j) = \eta[1 - \cos(dih(\tau_i, \tau_j))] = \frac{\eta}{2}\|N(\tau_i) - N(\tau_j)\|^2, \tag{5}$$

where $N(\tau_l)$ is the normal vector of triangle τ_l, and η is a constant used to give higher priority to a concave edge. We set $\eta = 1.0$ for a concave edge and a relatively small number (e.g., 0.2) for a convex edge, to follow the minima rule. The difference $d_1(\tau_i, \tau_j)$ is then normalized by the average over all edges, $\overline{d_1}$, which gives the normalized difference $d(\tau_i, \tau_j) = \frac{d_1(\tau_i,\tau_j)}{\overline{d_1}}$. Then a weight is heuristically defined to describe the similarity of triangles τ_i and τ_j and is assigned to their common edge e

$$w_{ij} = |e| \exp\{-d(\tau_i, \tau_j)\}, \tag{6}$$

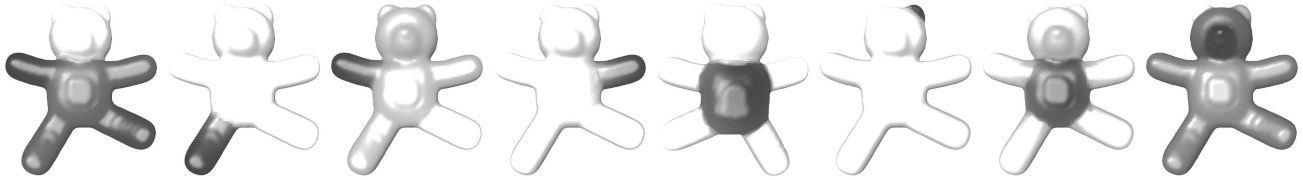


Fig. 2. Plots of the eigenvectors of the proposed Laplacian matrix corresponding to the first eight nonzero eigenvalues. Note that for the fifth model, we purposely render the back side of the model to make the dark part visible. The model is courtesy of the Princeton Segmentation Benchmark.

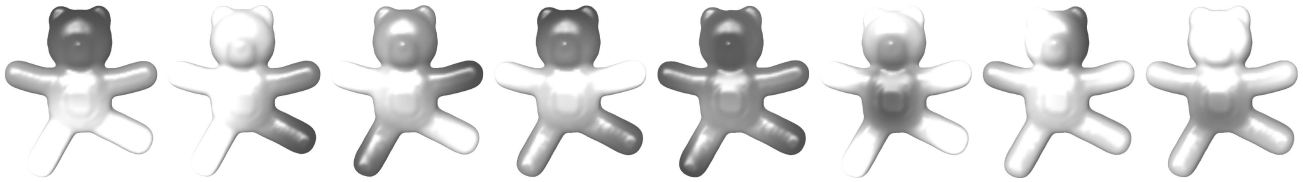


Fig. 3. Plots of the eigenvectors of the simple Laplacian matrix corresponding to the first eight nonzero eigenvalues. The model is courtesy of the Princeton Segmentation Benchmark.

where $|e|$ is the edge length. Finally the Laplacian matrix $L = [L_{ij}]$ is defined, where

$$L_{ij} = \begin{cases} -w_{ij}, & i \neq j \text{ and } \tau_i \text{ and } \tau_j \text{ share a common edge} \\ \sum_k w_{ik}, & j = i \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Actually L is the Laplacian matrix of the dual graph of M.

It is easy to check that Laplacian matrix L is symmetric and its smallest eigenvalue is 0. Assume that the eigenvectors of L are $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_{|T|-1}$ corresponding to the eigenvalues in ascending order. Here we select eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_{k-1}$ corresponding to the $k-1$ smallest nonzero eigenvalues for k-partitioning. These eigenvectors form a $|T| \times (k-1)$ matrix $[\mathbf{v}_1, \ldots, \mathbf{v}_{k-1}]$. Each row of the matrix corresponds to a triangle of M. This suggests that we define the multichannel function

$$\mathbf{f}(\tau_i) = [\mathbf{v}_1(i), \mathbf{v}_2(i), \ldots, \mathbf{v}_{k-1}(i)],$$

where $\mathbf{v}_j(i)$ represents the i-th entry of $\mathbf{v}_j$. In this way, we introduce $(k-1)$-channel data to each triangle on the mesh. Since the data are from the eigenvectors of the Laplacian matrix, $\mathbf{f}$ can be viewed as a spectral attribute of the mesh.

Remark 1. We chose to use the dual graph Laplacian matrix, rather than the mesh Laplacian matrix itself. This results in the multichannel function $\mathbf{f}$ to be actually a dual function, which benefits our numerical solver presented later in Section 4.3.

Remark 2. As pointed out in von Luxburg [2007], the eigenvectors are an approximation of the characteristic functions for each component of the mesh. Figure 2 shows plots of the eigenvectors of the Laplacian matrix L corresponding to the first eight nonzero eigenvalues for a bear model. Each eigenvector is linearly mapped such that the maximum and minimum values of the entries are 1 and 0. The value of the i-th entry is interpreted as the grey value assigned to the i-th triangle. We can see that each of these eigenvectors tends to binary segment the bear model. This feature implies that our definition of the multichannel function is feasible. It is worth pointing out that our Laplacian matrix is different from the simple cotangent-based Laplacian. Our expression of the Laplacian better measures the similarity between adjacent triangles and has been used in random walks-based segmentation algorithms [Lai et al. 2008]. As a comparison, Figure 3 provides plots of the eigenvectors of the simple Laplacian matrix corresponding to the first eight nonzero eigenvalues.

Remark 3. In graph partitioning, the spectral decomposition of the Laplacian matrix is often used to approximately minimize the RatioCut model [Hagen and Kahng 1992]

$$RatioCut(M_1, \ldots, M_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(M_i, \overline{M_i})}{|M_i|}, \tag{8}$$

where $M_1, \ldots, M_k$ are a k-partition of a graph, $\overline{M_i}$ is the complement of M_i, $W(M_i, \overline{M_i})$ is an association value of sets M_i and $\overline{M_i}$, and $|M_i|$ is the size of M_i. The objective function (8) tries to achieve a "balance" of the clusters in terms of the size. This implies that the use of eigenvectors might avoid the influence of small-scale fluctuation.

4.2 Discretization of the Mumford-Shah Model

Let φ_i denote a hat function that is linear on each triangle of M and $\varphi_i(v_j) = \delta_{ij}$ for $v_j \in V$, where δ_{ij} is the Kronecker delta. The functions $\{\varphi_i : i = 0, \ldots, |V|-1\}$ have three properties: local support, nonnegativity, and partition of unity. Any piecewise linear function $f(x)$ defined over M, which has value f_i at vertex v_i and is linear on each triangle, can be written as $f(x) = \sum_{0 \le i \le |V|-1} f_i \varphi_i(x)$ for any $x \in M$. To discretize the M-S segmentation energy functional of (4), we restrict $\mathbf{u}$ to be a piecewise linear function over M: $\mathbf{u}(x) = \sum_{v_i \in V} \mathbf{u}(v_i)\varphi_i(x)$. Then the M-S segmentation energy functional of (4) can be discretized into

$$\sum_{\tau \in T} \left(\sum_{v_i \prec \tau} \left\langle \mathbf{u}(v_i), \int_\tau \varphi_i(x)\mathbf{s}(x)d\sigma \right\rangle + \mu \int_\tau g_\tau |\nabla_M \mathbf{u}(x)| d\sigma \right). \tag{9}$$

For triangle τ, the edge detection function is set to be $g_\tau = \frac{1}{1+\sum_{i=1}^{3} \lambda_i ||N(\tau)-N(\tau_i)||^2}$ where τ_i, $i = 1, 2, 3$, are the triangles sharing edges with τ, and λ_i is a scaling factor, which is set respecting the minima rule: $\lambda_i = 5$ if the edge shared by τ and τ_i is a concave edge; otherwise, $\lambda_i = 1$.

Moreover, the gradient of $\mathbf{u}(x)$ is

$$\nabla_M \mathbf{u}(x) = \sum_{v_i \in V} \mathbf{u}(v_i) \nabla_M \varphi_i(x), \tag{10}$$

where $\nabla_M \varphi_i(x)$ is regarded as a row vector, that is, a 1×3 matrix, which can be calculated using the natural piecewise parametrization (see Wu et al. [2009] for details). It follows that $\nabla_M \mathbf{u}(x)$ is a constant $k \times 3$ matrix on each triangle. Thus (9) becomes

$$\sum_{\tau \in T} \left(\sum_{v_i \prec \tau} \left\langle \mathbf{u}(v_i), \int_\tau \varphi_i(x) \mathbf{s}(x) d\sigma \right\rangle + \mu g_\tau a_\tau |\nabla_M \mathbf{u}(\tau)| \right), \tag{11}$$

where a_τ is the area of triangle τ.

4.3 Fast Primal-Dual Method

To find $\mathbf{u}$ that minimizes the functional (11), a gradient descent method [Delaunoy et al. 2009] may be used. However, explicit schemes often lead to numerical instability, slow convergence for large meshes, and insufficient control over global behavior. In this section, we adapt a fast primal-dual algorithm to meshes to quickly and stably minimize the objective functional. The primal-dual algorithm can be highly paralleled and thus the GPU technology can be used to speed up the computation.

Using the Cauchy-Schwartz inequality, we can reformulate the second term of (11) to be

$$\begin{aligned}
&\sum_{\tau \in T} \mu g_\tau a_\tau |\nabla_M \mathbf{u}(\tau)| = \max_{\mathbf{d}(\tau) \in D, \forall\, \tau} \mu \sum_{\tau \in T} g_\tau a_\tau \langle \nabla_M \mathbf{u}, \mathbf{d}(\tau) \rangle \\
&= \max_{\mathbf{d}(\tau) \in D} \mu \sum_{\tau = [v_i, v_j, v_k] \in T} g_\tau a_\tau (\langle \mathbf{u}(v_i) \nabla_M \varphi_i(\tau), \mathbf{d}(\tau) \rangle \\
&\quad + \langle \mathbf{u}(v_j) \nabla_M \varphi_j(\tau), \mathbf{d}(\tau) \rangle + \langle \mathbf{u}(v_k) \nabla_M \varphi_k(\tau), \mathbf{d}(\tau) \rangle) \\
&= \max_{\mathbf{d}(\tau) \in D} \mu \sum_{v_i \in V} \left\langle \mathbf{u}(v_i), \sum_{\tau \in D_1(v_i)} g_\tau a_\tau \langle \nabla_M \varphi_i(\tau), \mathbf{d}(\tau) \rangle \right\rangle
\end{aligned}$$

where $D = \{\mathbf{d}(\tau) = (\mathbf{d}_1, \ldots, \mathbf{d}_k)^T \in R^{k \times 3} : \|\mathbf{d}(\tau)\|_F \leq 1, \tau \in T\}$, and $\mathbf{d}(\tau)$ is a dual variable defined for each triangle of the mesh. Here the operations $\langle \nabla_M \mathbf{u}, \mathbf{d}(\tau) \rangle$ and $\langle \mathbf{u}(v_i) \nabla_M \varphi_i(\tau), \mathbf{d}(\tau) \rangle$ are understood as follows.

$$\begin{aligned}
\langle A, B \rangle &= \sum_{1 \leq m \leq k, 1 \leq n \leq 3} a_{mn} b_{mn}, \\
\langle a, B \rangle &= \left(\sum_{1 \leq n \leq 3} a_n b_{1n}, \sum_{1 \leq n \leq 3} a_n b_{2n}, \ldots, \sum_{1 \leq n \leq 3} a_n b_{kn} \right)^T
\end{aligned}$$

for $a \in R^{1 \times 3}$ and $A, B \in R^{k \times 3}$, respectively, and no confusion will be caused by writing out the arguments in the bracket $\langle , \rangle$.

Let $C_i = \sum_{\tau \in D_1(v_i)} \int_\tau \varphi_i(x) \mathbf{s}(x) d\sigma$. Minimizing (11) becomes a saddle-point problem

$$\min_{\mathbf{u} \in K} \max_{\mathbf{d}(\tau) \in D} g(\mathbf{u}, \mathbf{d}), \tag{12}$$

where

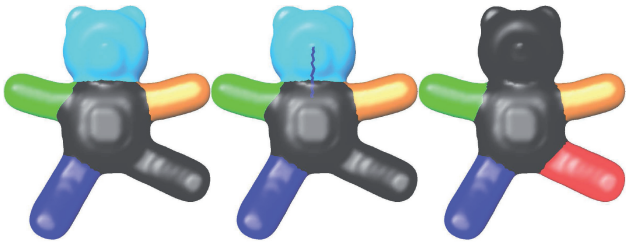
$$g(\mathbf{u}, \mathbf{d}) = \sum_{v_i \in V} \left\langle \mathbf{u}(v_i), C_i + \mu \sum_{\tau \in D_1(v_i)} g_\tau a_\tau \langle \nabla_M \varphi_i(\tau), \mathbf{d}(\tau) \rangle \right\rangle.$$

Since $\mathbf{u}$ is uniquely defined by its values at vertices, it is considered as a primal variable. Thus the saddle-point problem has its primal objective $p(\mathbf{u}) := \max_{\mathbf{d} \in D} g(\mathbf{u}, \mathbf{d})$ and its dual objective $p_d(\mathbf{d}) := \min_{\mathbf{u} \in K} g(\mathbf{u}, \mathbf{d})$. Rockafellar [1970] has shown that g has a saddle-point $(\mathbf{u}^*, \mathbf{d}^*)$ and

$$\min_{\mathbf{u} \in K} p(\mathbf{u}) = p(\mathbf{u}^*) = g(\mathbf{u}^*, \mathbf{d}^*) = p_d(\mathbf{d}^*) = \max_{\mathbf{d} \in D} p_d(\mathbf{d}). \tag{13}$$

ALGORITHM 1: FPD method

Choose initial values $\mathbf{u}^0$, $\overline{\mathbf{u}}^0$ and $\mathbf{d}^0$
Choose primal step $\tau_p > 0$, dual step $\tau_d > 0$, and termination tolerance $\epsilon > 0$
while $\frac{p(\mathbf{u}^k) - p_d(\mathbf{d}^k)}{p_d(\mathbf{d}^k)} > \epsilon$ **do**
  $\mathbf{d}^{k+1}(\tau) \leftarrow \Pi_D(\mathbf{d}^k(\tau) + \tau_d(\mu g_\tau a_\tau \nabla_M \overline{\mathbf{u}}^k))$ for $\tau \in T$
  $\mathbf{u}^{k+1}(v_i) \leftarrow \Pi_K(\mathbf{u}^k(v_i) - \tau_p(C_i$
    $+\mu \sum_{\tau \in D_1(v_i)} g_\tau a_\tau \langle \nabla_M \varphi_i(\tau), \mathbf{d}^{k+1}(\tau) \rangle))$ for $v_i \in V$
  $\overline{\mathbf{u}}^{k+1} \leftarrow 2\mathbf{u}^{k+1} - \mathbf{u}^k$
  $k \leftarrow k + 1$
end while



(a) Initial result (b) Must-link input (c) Updated result

Fig. 4. User's input changes the segmentation result. The model is courtesy of the Princeton Segmentation Benchmark.

Based on the preceding analysis, we can see that a straightforward approach to solving (12) is to alternatingly apply the projected gradient decent method on the primal variable $\mathbf{u}$ and the projected gradient ascent method on the dual variable $\mathbf{d}$ [Lellmann and Schnörr 2011]. In fact, Pock et al. [2009b] has presented the Fast Primal-Dual (FPD) method, which is a variant of Popov's saddle-point method [Popov 1980] with provable convergence. We adapt it to our problem, which is outlined in Algorithm 1.

In Algorithm 1, the initial primal variable $\mathbf{u}^0$ is randomly chosen, $\overline{\mathbf{u}}^0$ is the same as $\mathbf{u}^0$, and the initial dual variable $\mathbf{d}^0$ is set to the divergence of the initial primal variable. $\Pi_K(\cdot)$ is the operator that projects the primal variable $\mathbf{u}$ on the set K and $\Pi_D(\cdot)$ is the operator that projects the dual variable $\mathbf{d}$ on the dual constraint set D.

5. USER INTERACTION

Though automatic decomposition is preferred, it is also desirable to allow users to express their intention or their preference in some situations, especially when there exist several decomposition possibilities. For example, if we decompose a bear model into five parts, our algorithm generates the result shown in Figure 4(a). However, if we want the head to be connected to the body, user's input such as a stroke passing through the head and the body shown in Figure 4(b) would help. Therefore we introduce constraints into the Mumford-Shah model to incorporate such user's inputs and the new result is shown in Figure 4(c).

5.1 Mumford-Shah Model with Constraints

If the user draws a stroke to express the intention that some vertices must be in the same segment, we can achieve this "must-link" constraint by enforcing $\mathbf{u}$ at those vertices to be the same. Then the Mumford-Shah model becomes a minimization problem with some equality constraints. For example, assume that m vertices are

required to be in the same segment. This will introduce $m-1$ independent constraints. If we let $\mathbf{U}=[\mathbf{u}(v_0),\ldots,\mathbf{u}(v_{|V|-1})]^T$, the k-th constraint that v_i and v_j belong to the same segment can be expressed as $\mathbf{z}_k^T\mathbf{U}=0$ where $\mathbf{z}_k$ is a $|V|$-dimensional vector with only two nonzero entries: $\mathbf{z}_k(i)=1$ and $\mathbf{z}_k(j)=-1$. Grouping all the $\mathbf{z}_k$ gives $\mathbf{Z}=[\mathbf{z}_1,\ldots,\mathbf{z}_{m-1}]$ and all the constraints become

$$\mathbf{Z}^T\mathbf{U}=0. \tag{14}$$

Incorporating these constraints into the Mumford-Shah model of (9) leads to a constrained Mumford-Shah model. Using Lagrange multipliers, we can change the constrained model into an unconstrained optimization problem

$$\min_{\mathbf{u}\in K}\max_{\mathbf{w}}\left\{\sum_{\tau\in T}\left(\sum_{v_i\prec\tau}\left\langle\mathbf{u}(v_i),\int_\tau\varphi_i(x)\mathbf{s}(x)d\sigma\right\rangle+\mu\int_\tau g_\tau|\nabla_M\mathbf{u}(x)|d\sigma\right)+\langle\mathbf{w},\mathbf{Z}^T\mathbf{U}\rangle\right\}, \tag{15}$$

where $\mathbf{w}$ are the Lagrangian multipliers. This minimization problem can also be solved using the fast primal-dual method described in Section 4.3. If there are several sets of such requirements, we can express all these constraints in a similar way.

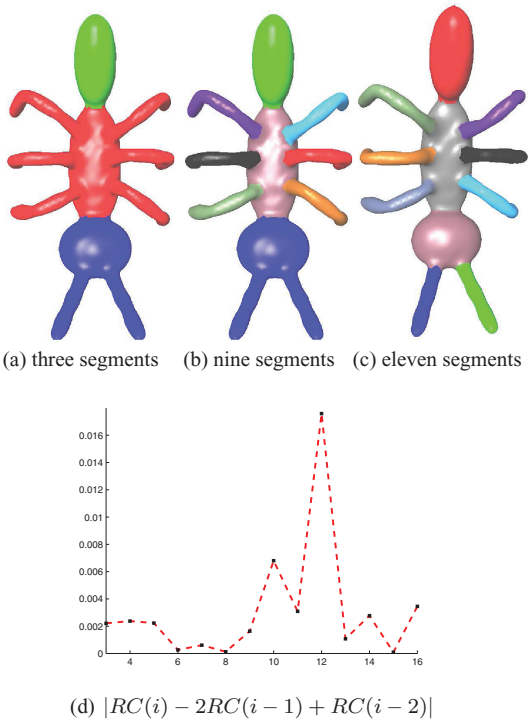
5.2 Constraint Propagation

It is observed that constraints (14) only guarantee that those m vertices specified by user's inputs are in the same segment and they have little influence on the neighborhood of the user's inputs. Moreover, we wish that the user's inputs could be reflected in the two terms of the Mumford-Shah model. However, in our method, the data term is determined by the Laplacian matrix's eigenvectors, which are fixed if the edge weights w_{ij} are fixed. To overcome this problem, we propose to increase the weights for those edges near the user's inputs in order to propagate the influence of the user's inputs. Specifically, for each indicated vertex, we get its farthest vertex on the mesh. The user-indicated vertices are treated as foreground seeds and those farthest vertices are treated as background seeds. Then we use random walks algorithm [Grady 2006] to compute a probability value $p(v_k)$ for each vertex v_k on the mesh. For edge e bounded by vertices v_i and v_j, if $\frac{p(v_i)+p(v_j)}{2}>0.9$, which means that v_i and v_j are very close to the user-indicated vertices, we reset the edge weight of e to 1. In this way, a small region around the user-indicated vertices will be strongly connected because the similarity weights between them are set to the largest value 1.

6. DETERMINATION OF THE NUMBER OF SEGMENTS

The Mumford-Shah mesh decomposition presented in preceding sections depends on the number of segments. This section proposes a heuristic method to compute a number based on the stability of the RatioCut values for the number of segments. It can be done in a preprocessing stage and could be added to the Mumford-Shah mesh decomposition to provide an automatic algorithm. The experiments show that the number computed by the method matches the one given by users fairly well.

Our observation is that if a good k-partitioning has been done and one more segment is to be added, the existing segments have to be split and merged to form a new segmentation, which usually causes the association between segments to increase rapidly and the sizes of segments to lose balance. As a result, the RatioCut value of (8) where the association value $W(M_i,\overline{M_i})$ is computed as the length of M_i's boundary will have a sudden change. This motivates us to propose a brute-force approach: perform segmentation for various numbers of segments, compute the RatioCut value for the segmentation results, and choose $h-1$ as the number of segments if h-partitioning causes a sudden change in RatioCut values.



(a) three segments (b) nine segments (c) eleven segments

(d) $|RC(i)-2RC(i-1)+RC(i-2)|$

Fig. 5. Numbers of segments and the second-order differences of RatioCut values. The model is courtesy of the Princeton Segmentation Benchmark.

Note that producing good segmentations for every possible number of segments is a time-consuming process. Considering our goal here is to find the number of segments, instead of using sophisticated segmentation methods for high-quality segmentation, we employ a fast algorithm to obtain a reasonable segmentation, based on which we perform RatioCut value stability analysis. As explained in Section 4.1, the Laplacian matrix L contains global information of the underlying mesh. Ng et al. [2001] and Polito and Perona [2001] determine the cluster number by searching for a drop in the magnitude of the eigenvalues of the Laplacian matrix. Zelnik-Manor and Perona [2004] rotates the eigenvectors of the Laplacian matrix and finds the number of segments which provides the best alignment with the canonical coordinate system. We notice that each eigenvector is roughly considered to be an indicator for some segment of mesh M. Therefore we use the K-means algorithm to cluster all the triangles based on their multichannel data from the $k-1$ smallest (nonzero) eigenvectors to obtain a rough k segments, which can be done in a very fast speed. Then the RatioCut value is computed for each k, which is denoted by $RC(k)$. Finally the number of segments is chosen to be $\arg\max_i\{|RC(i)-2RC(i-1)+RC(i-2)|\}-1$, which maximizes the second-order difference.

Figure 5 shows three possible numbers of segments and their corresponding second-order differences of RatioCut values.

7. EXPERIMENTAL RESULTS AND DISCUSSIONS

This section provides experimental results to validate our proposed algorithm. The experiments are conducted in two aspects. Since

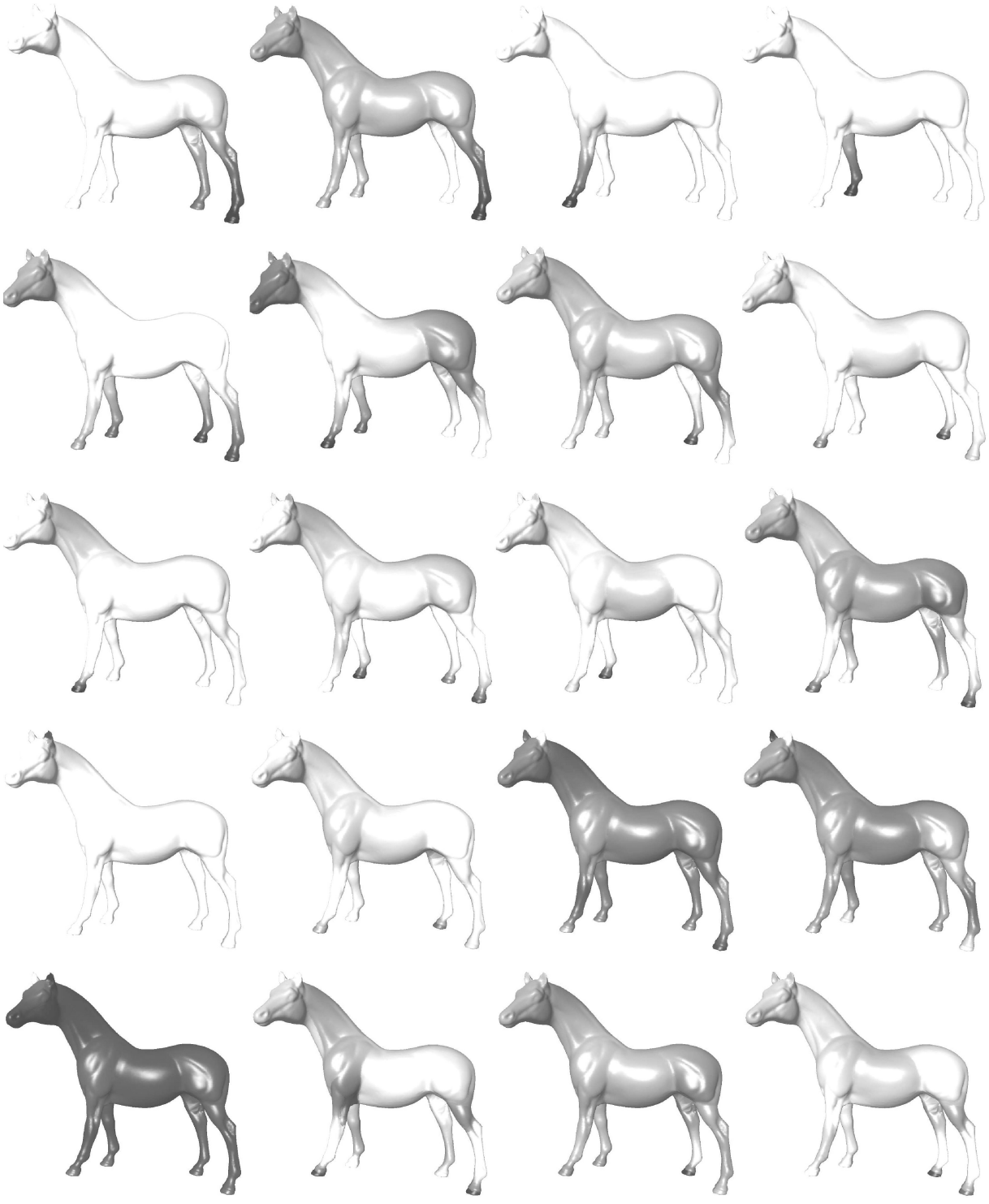
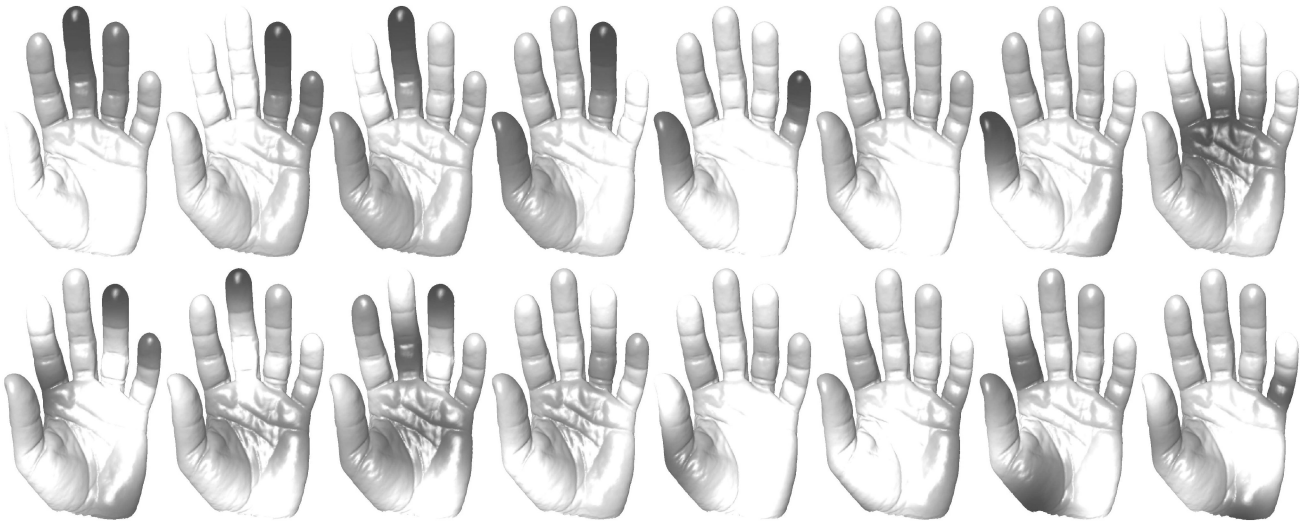


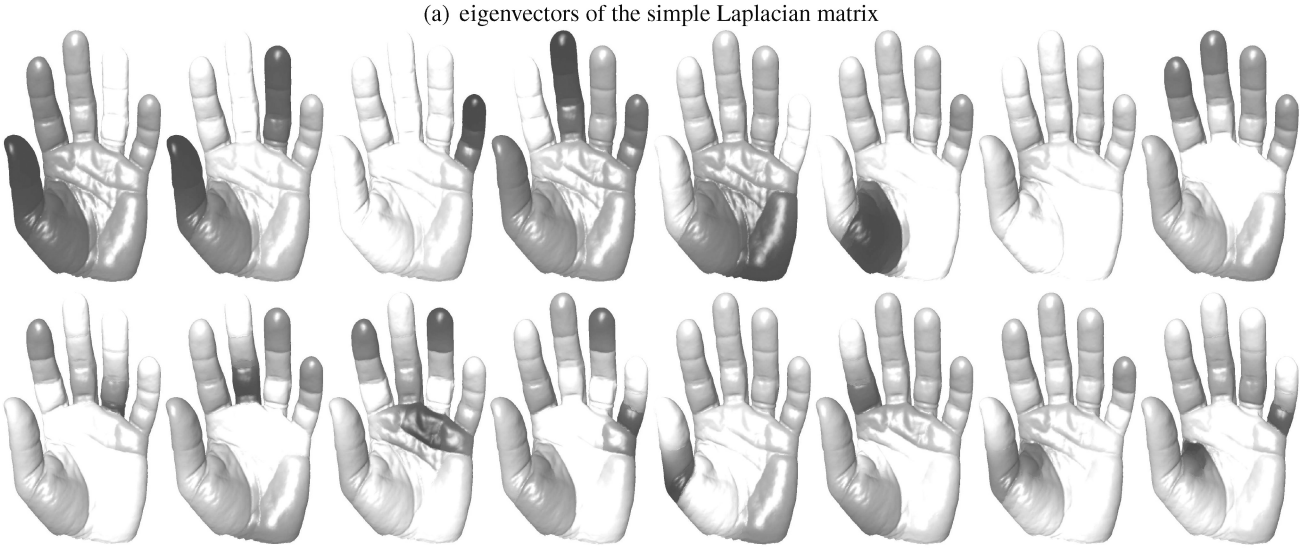
Fig. 6. Plots of the eigenvectors of our proposed Laplacian matrix corresponding to the first twenty nonzero eigenvalues. The model is provided courtesy of the AIM@SHAPE Shape Repository.

the proposed Mumford-Shah mesh segmentation algorithm contains various technical components, the first aspect is to test the effects of the key ingredients. The second aspect is to evaluate the performance of the algorithm as a whole applied to various 3D mesh models and to perform comparison with other methods as well.

Our proposed algorithm requires to select a value for trade-off parameter μ in Eq. (4). The larger μ is, the more significant the



(a) eigenvectors of the simple Laplacian matrix



(b) eigenvectors of our proposed Laplacian matrix

Fig. 7. Comparison of the eigenvectors of the simple Laplacian matrix and our proposed Laplacian matrix. We can see that our proposed Laplacian matrix better reflects the structure of the underlying models. The model is provided courtesy of INRIA by the AIM@SHAPE Shape Repository.

weighted boundary term. Different μ values could result in different segmentation results. As the data and regularization terms are affected by the mesh model, the number of segments, and the magnitude of geometric features of the model, it is usually difficult to select a value that is suitable for all situations. However, considering that μ is expected to be independent of the model scale and also noticing that when the number of segments increases, the value of the regularization term increases and the value of the data term decreases, we here provide an empirical formula to compute μ. We have

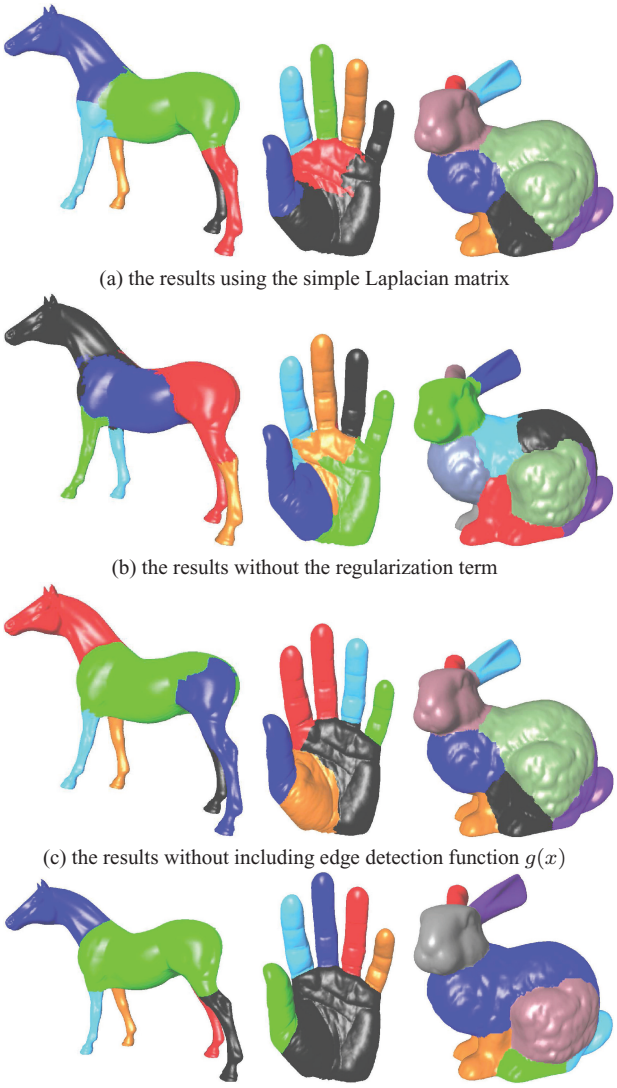
$$\mu = \bar{\mu} \times \text{Number-of-segments} \times \frac{E_f}{E_r},$$

where $E_f = \int_M \langle \mathbf{u}(x), \mathbf{s}(x) \rangle d\sigma$ is the data term and $E_r = \int_M g(x) |\nabla_M \mathbf{u}(x)| d\sigma$ is the regularization term given an initial $\mathbf{u}$; and $\bar{\mu}$ is a "normalized" parameter. Our empirical value for $\bar{\mu}$ is 0.02, which works well for many situations. In fact, we used this choice for all the models in the Princeton Segmentation Benchmark. If needed, users can still adjust $\bar{\mu}$ around 0.02, which is more convenient than tuning μ in a wide range.

It is worth pointing out that the Mumford-Shah segmentation does not guarantee the connectedness of the segmentation. That is, though the segmentation produces k segments for k-partition, the resulting regions belonging to the same segment may not be connected geometrically, which results in more than k disjoint regions. In our experiments, in case this situation occurs, we recursively merge the segment with the smallest number of the vertices to its neighboring segment that has the largest number of vertices until the number of the disjoint regions is equal to k.

7.1 Effects of Key Ingredients

Our proposed algorithm contains several key ingredients. First, we use the eigenvectors of a dual Laplacian matrix to define the



(a) the results using the simple Laplacian matrix

(b) the results without the regularization term

(c) the results without including edge detection function $g(x)$

(d) the results with our proposed M-S model.

Fig. 8. Comparison of segmentation with various choices. The horse and hand models are provided courtesy of the AIM@SHAPE Shape Repository. The bunny model is courtesy of Stanford Computer Graphics Laboratory.

attributes for the mesh elements instead of using purely local feature descriptions. Remark 2 of Section 4.1 has mentioned that the eigenvectors approximate the characteristic functions of components of a mesh. Figures 2, 6, and 7(b) visualize the eigenvectors of the proposed Laplacian matrix corresponding to the first few nonzero eigenvalues, from which it can be seen that the eigenvectors convey segment information. It is also observed that usually the first few smallest (nonzero) eigenvectors correspond to relatively large structures of a mesh. Considering the computational cost, we choose only the first $(k-1)$ nonzero eigenvectors to define our multichannel function if the target number of segments is k, which is similar to the approach of Liu and Zhang [2004] that chooses k eigenvectors of a normalized affinity matrix.

Second, different from the simple Laplacian matrix, our Laplacian matrix is constructed from the weights that consider the coplanarity between adjacent triangles and the local convexity or concavity of the edge shared by the triangles as well. As a result, each eigenvector of our Laplacian matrix tends to be more suitable for binary segmentation and concentrate more on meaningful parts of the model. Figures 2, 3, and 7 show such comparison.

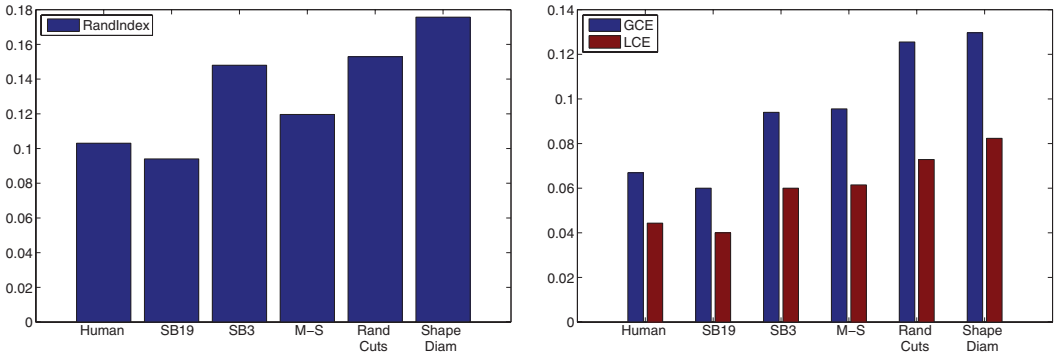
Third, the Mumford-Shah model contains a regularization term that constrains the boundary between segments to be as short as possible. This has an effect of smoothing. Furthermore, we introduce an edge detection function into the regularization term to encourage the segmentation to align with feature edges. Figure 8(b) shows some results of segmentation without the regularization term. The boundaries of segments are apparently not smooth. Adding the regularization term improves the smoothness of the boundaries, but without the edge detection function the resulting boundaries of segments may not align well with the geometric features of the models, as depicted in Figure 8(c). For a comparison, we also display the segmentation results using the M-S model applied to the eigenvectors of the simple Laplacian matrix in Figure 8(a). Obviously, by integrating all the key ingredients mentioned earlier, our proposed algorithm produces excellent segmentation results as shown in Figure 8(d).

7.2 Overall Performance of the Algorithm

Overall performance on benchmark dataset. We apply our proposed algorithm on the entire 3D Segmentation Benchmark dataset [Chen et al. 2009], which contains 19 categories of meshes (20 models per category), segmentation results by human users, source code for computing evaluation scores, and the results of many existing segmentation methods. One snapshot of the visual results in each category produced by our algorithm is shown in Figure 1. Note that our algorithm is general and fully automatic, requiring no prior information, no given number of segments, and no training. From Figure 1, we can see that the results of our algorithm match human perception well in not only the cutting boundaries but also the number of segments. In addition, the cutting contours are along geometric features.

We further compare our algorithm with the two state-of-the-art geometry-based methods: Randomized Cuts [Golovinskiy and Funkhouser 2008] and Shape Diameter Function [Shapira et al. 2008], where Randomized Cuts requires given number of segments and Shape Diameter Function determines the number of segments automatically. We also include the latest learning-based method [Kalogerakis et al. 2010] for reference, although it is a different type of approach. For all the methods, we perform evaluations according to the protocols of Chen et al. [2009], using all human segmentations in the Princeton Segmentation Benchmark. Following the comparison presentation in Kalogerakis et al. [2010], we show the scores of two evaluation metrics, Rand Index and Consistency Error, in Figure 9 and list the detailed Rand Index scores for each category in Table I, which also include the segmentation results of human users. The definitions of Rand Index and Consistency Error can be referred to [Chen et al. 2009], and smaller values suggest better segmentation results. Note that Rand Cuts is given the dominant number of segments in the human segmentations for each model.

Among the three geometry-based methods, M-S, Randomized Cuts and Shape Diameter Function, from Figure 9 and Table I, we can see that our proposed M-S algorithm achieves the best performance, significantly outperforming the other two. Note that the scores of human segmentations are not perfect. This is due to the variations of the segmentation results among different users. Figure 10 further shows the visual comparisons of the three



(a) Rand Index (b) Consistency Error

Fig. 9. Quantitative evaluation of segmentation results in terms of Rand Index and Consistency Error (smaller values stand for better results). For all the methods, we perform evaluations according to the protocols of Chen et al. [2009], using all human segmentations in the Princeton Segmentation Benchmark. "SB19" and "SB3" stand for the learning-based method with training set sizes of 19 and 3 (out of 20 models for each category), respectively. "M-S", "Rand Cuts" and "Shape Diam" represent our proposed method, Randomized Cuts and Shape Diameter Function, respectively. Note that Rand Cuts is given the dominant number of segments in the human segmentations for each model. Our method significantly outperforms the two state-of-the-art geometry-based methods while worse than the learning-based method with large training size.

Table I. Rand Index Scores for Human Segmentations, SB19, SB3, M-S (our method), Randomized Cuts, and Shape Diameter Function

Object Categories	Bench Mark	SB19	SB3	M-S	Rand Cuts	Shape Diam
Human	13.5	11.9	14.7	11.1	13.1	17.9
Cup	13.6	9.9	10.0	20.4	21.9	35.8
Glasses	10.1	13.6	14.2	9.4	10.1	20.4
Airplane	9.2	7.9	10.2	11.1	12.2	9.2
Ant	3.0	1.9	2.6	2.2	2.5	2.2
Chair	8.9	5.4	6.6	10.9	18.4	11.1
Octopus	2.4	1.8	2.2	2.5	6.3	4.5
Table	9.3	6.2	11.1	10.3	38.3	18.4
Teddy	4.9	3.1	5.6	3.2	4.5	5.7
Hand	9.1	10.4	15.8	7.9	9.0	20.2
Plier	7.1	5.4	10.5	8.9	11.0	37.5
Fish	15.5	12.9	13.5	29.6	29.7	24.8
Bird	6.2	10.4	18.6	9.4	10.7	11.5
Armadillo	8.3	8.0	8.6	8.7	9.2	9.0
Bust	22.0	21.4	39.3	25.1	23.2	29.9
Mech	13.1	10.0	24.0	13.1	27.7	23.8
Bearing	10.4	9.7	32.7	16.6	12.4	11.9
Vase	14.4	16.0	25.3	12.5	13.3	23.9
FourLeg	14.9	13.3	16.3	14.4	17.4	16.1
Average	10.3	9.4	14.8	12.0	15.3	17.6

The Rand Index scores are measured against all human segmentations in the Princeton Benchmark and smaller scores suggest better segmentation results.

geometry-based methods, which further demonstrate the superior performance of our proposed method.

As for the learning-based method [Kalogerakis et al. 2010], we do not intend to have a comprehensive comparison with our method since they are different types of approaches. The learning-based method is an excellent work, which can perform not only segmentation but also labeling while requiring category-specified training. In terms of the segmentation results, in general the average performance of the learning-based method is better than ours when the training set size is large. However, when the training set size is small, say no more than 30% of the entire category size, the performance of our algorithm is comparable or better on average. Note that the human model in Figure 10 is actually a failure case for the learning-based method [Kalogerakis et al. 2010] while our method can segment it well.

Table II. Mesh Information and Running Time Statistics

Model (Figure)	# of vertices	# of triangles	# of segments	CPU (s)	GPU (s)
Horse (8(d))	48485	96966	6	50.47	4.16
Hand (8(d))	53054	105860	6	71.45	4.41
Santa (8(d))	75781	151558	6	80.48	6.54
Bunny (8(d))	34834	69451	12	91.68	8.63
Octahedron (12(a))	16386	32768	8	21.78	2.21
Fandisk (12(b))	6475	12946	3	4.81	0.60

The GPU implementation improves the processing speed by around ten times.

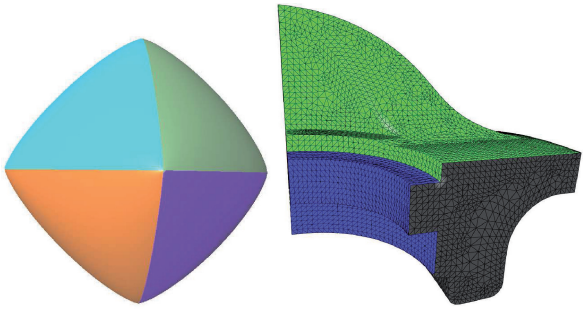
Cutting contour smoothness. The visual results in Figures 1 and 10 already show that our algorithm produces smooth cutting contour, along geometry features. This is mainly because of the boundary term in the M-S model, which essentially pulls the cutting contours toward the geodesic curves [Zhang et al. 2010]. In addition to the benchmark dataset, we also test our algorithm on other models in Figures 8(d) and 11 and two CAD models in Figure 12, which contain sharp edges. We can see that our algorithm can cut along the sharp edges and produce smooth cutting contour.

Segmentation efficiency. As described in Section 4.3, a fast primal-dual algorithm has been devised to efficiently solve the M-S model for mesh decomposition. To further accelerate the processing speed, we implement our algorithm on GPU using the NVIDIA CUDA framework with Quadro FX 4600 graphics card. All the experiments are run on a PC with Intel Core 2.66GHz CPU and 2GB RAM. Table II lists the computing time for the models in Figures 8(d) and 12. It can be seen that with our GPU implementation, the processing speed is improved by around ten times and the segmentation of a middle-size model can be done in a few seconds even though our GPU implementation is not optimized yet. Further speed improvement might be possible since over thirty times acceleration has been reported in Pock et al. [2009b].

Fig. 10. Visual comparisons to other segmentation methods for Table, Octopus, Hand, Human, Fourleg, Chair, Mech. Top row: results of our method. Middle row: results of Randomized Cuts, with number of segments defined as the dominant number of segments in the human segmentations for each model. Bottom row: results of Shape Diameter Function. These models are courtesy of the Princeton Segmentation Benchmark.

Fig. 11. Segmentation results of the proposed algorithm on four nonbenchmark models, which are provided courtesy of UU, VCG-ISTI, INRIA, and MPII by the AIM@SHAPE Shape Repository. The cutting contours are along geometry features.

Number of segments. All the visual results shown in the previous figures have demonstrated that the number of segments selected by our method proposed in Section 6 matches human perception well.



(a) Octahedron (b) Fandisk

Fig. 12. Segmentation results of the proposed algorithm on two CAD models. Our method cuts the models along the sharp edges.

Considering that the method of Shape Diameter Function [Shapira et al. 2008] can also automatically choose the number of segments, we further conduct the following experiments for comparison. In particular, for each mesh model we compare the selected number of segments with each of the human segmentations. If the predicted number is equal to a human segmentation, we count it as one; otherwise, we count it as zero. For the entire 4300 human segmentations on 380 different meshes, the number of segments we choose coincides with 1087 human segmentations while Shape Diameter Function scores 741. In fact, the number of matches for our method is quite high, considering that there exist significant variations in the number of segments among different human segmentations for each model.

7.3 Limitations

Despite its superior average performance over the entire benchmark dataset, our method fails at some cases. Figure 13 gives one example, where our results do not match the human segmentations. This

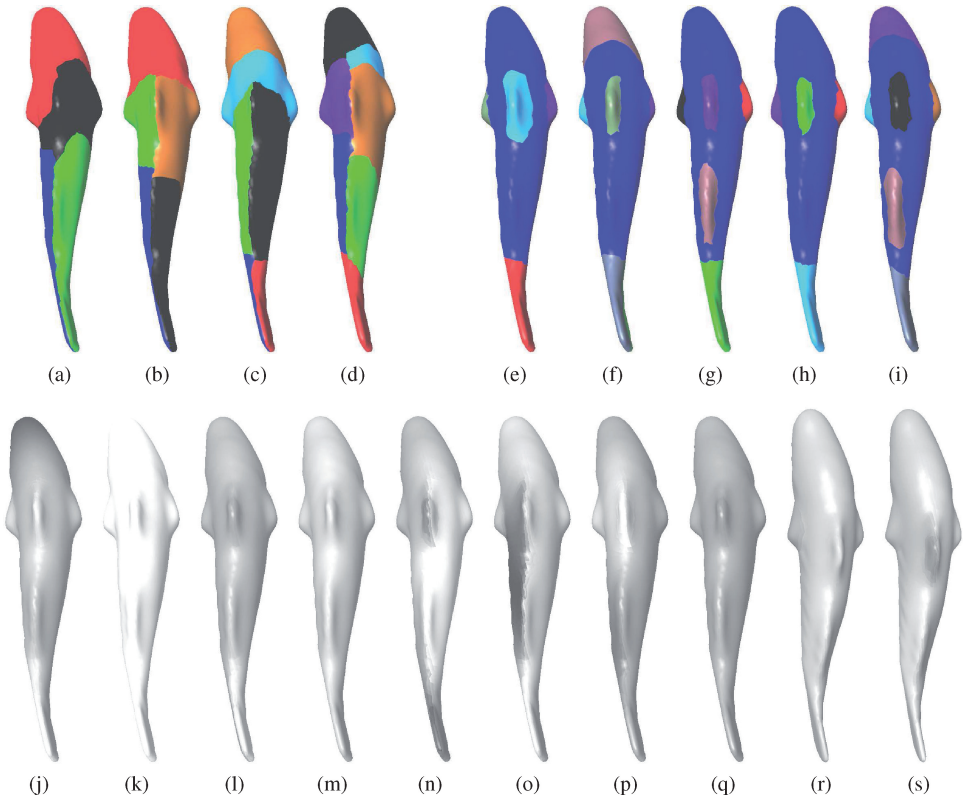


Fig. 13. (a)-(d): The segmentation results of our method with the number of segments being 4, 5, 6, 7, respectively. (e)-(i): samples of human segmentations. (j)-(s): Plots of the eigenvectors of our proposed Laplacian matrix corresponding to the first ten nonzero eigenvalues. Our method fails at this smooth model which contains some tiny segments (fins) and lacks clear geometry edges. The model is courtesy of the Princeton Segmentation Benchmark.

is mainly because the fish model is very smooth, lacking clear geometry edges and some expected segments (the fins) are of small size, while our method assumes that the size of each segment should be considerable. By observing the first ten nonzero eigenvectors of the Laplacian matrix for the fish model, we find that the eigenvectors themselves do not contain much expected segment information. We believe high-level cues are needed to well segment such type of mesh models.

8. CONCLUSION

This article has studied three fundamental issues in mesh decomposition: how to accurately and efficiently decompose a mesh into meaningful parts, how to incorporate users' inputs to influence the segmentation, and how to automatically determine the number of segments that a mesh should be decomposed into. First, a Mumford-Shah formulation for mesh segmentation is presented, which minimizes a functional with two terms: one measuring the variation within a segment and the other measuring the length of the boundary between segments. To solve the minimization, an alternating strategy is proposed, which involves solving two subproblems: one with an explicit solution and the other that is converted to a saddle-point problem solved by the fast primal-dual method. This formulation simultaneously handles segmentation and boundary smoothing and is able to efficiently partition a mesh into a prescribed number of components. Second, a constrained Mumford-Shah model is formulated to incorporate users' inputs and an approach is presented to compute the segmentation that reflects users' intention. Users can draw constraint curves to force the area around the curves to belong to the same segment. Third, an automatic approach is proposed to compute the best number of segments for a given model. These works extensively utilize the spectral information of the mesh represented by the eigenvectors of the dual Laplacian matrix of the mesh, aim at segmentation with high similarity within each segment and low association among different segments, and take human perception into consideration. Consequently, the proposed algorithm outperforms most existing geometry-based segmentation algorithms in terms of quality and speed when evaluated on the Princeton Segmentation Benchmark. Extensive experiments show that our algorithm is able to produce segmentation results that match human perception.

There are a few issues worth further investigation. In the article only $k-1$ eigenvectors are used for k-partition. It is not clear whether using more eigenvectors will help improve the accuracy of segmentation or what is the optimal number of eigenvectors that should be used to construct the multichannel function. Second, it is certainly valuable to develop deep understanding of the conditions under which the eigenvectors of the Laplacian matrix can effectively describe the attributes for mesh elements. Moreover, regarding k-partition, the method proposed in the article for determining the number of segments is heuristic. It is interesting to study how to devise an efficient way directly from the Mumford-Shah model to determine the number of segments.

ACKNOWLEDGMENTS

We are grateful to the reviewers for their helpful comments and suggestions.

REFERENCES

ATTENE, M., KATZ, S., MORTARA, M., PATANÈ, G., SPAGNUOLO, M., AND TAL, A. 2006. Mesh segmentation - A comparative study. In *Proceedings of the International Conference on Shape Modeling and Applications*, 14–25.

BARDSLEY, J. M. AND LUTTMAN, A. 2009. A fixed point formulation of the k-means algorithm and a connection to Mumford-Shah. *Appl. Math. E-Notes 9*, 274–276.

CHEN, X., GOLOVINSKIY, A., AND FUNKHOUSER, T. 2009. A benchmark for 3D mesh segmentation. *ACM Trans. Graph. 28,* 3.

COHEN-STEINER, D., ALLIEZ, P., AND DESBRUN, M. 2004. Variational shape approximation. *ACM Trans. Graph. 23,* 3, 905–914.

DELAUNOY, A., FUNDANA, K., PRADOS, E., AND HEYDEN, A. 2009. Convex multi-region segmentation on manifolds. In *Proceedings of the International Conference on Computer Vision (ICCV)*.

DEY, T. K., GIESEN, J., AND GOSWAMI, S. 2003. Shape segmentation and matching with flow discretization. In *Proceedings of 8th International Workshop on Algorithms and Data Structures (WADS)*. 25–36.

DEY, T. K., RANJAN, P., AND WANG, Y. 2010. Convergence, stability, and discrete approximation of Laplace spectra. In *Proceedings of ACM/SIAM Symposium on Discreate Algorithms (SODA)*. 650–663.

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S., AND DOBKIN, D. 2004. Modeling by example. *ACM Trans. Graph.* 652–663.

GOLOVINSKIY, A. AND FUNKHOUSER, T. A. 2008. Randomized cuts for 3D mesh analysis. *ACM Trans. Graph. 27,* 5.

GRADY, L. 2006. Random walks for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. 28,* 11, 1768–1783.

GREGORY, A., STATE, A., LIN, M. C., MANOCHA, D., AND LIVINGSTON, M. A. 1999. Interactive surface decomposition for polyhedral morphing. *Vis. Comput.* 9, 453–470.

HAGEN, L. W. AND KAHNG, A. B. 1992. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Comput. Aid. Des. Integr. Circ. Syst. 11,* 9, 1074–1085.

HOFFMAN, D. AND SINGH, M. 1997. Salience of visual parts. *Cognition*, 29–78.

HOFFMAN, D. D. AND RICHARDS, W. 1984. Parts of recognition. *Cognition*, 65–96.

KALOGERAKIS, E., HERTZMANN, A., AND SINGH, K. 2010. Learning 3D mesh segmentation and labeling. *ACM Trans. Graph. 29,* 4.

KATZ, S., LEIFMAN, G., AND TAL, A. 2005. Mesh segmentation using feature point and core extraction. *Vis. Comput. 21,* 8-10, 649–658.

KATZ, S. AND TAL, A. 2003. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM Trans. Graph. 22,* 3, 954–961.

LAI, Y.-K., HU, S.-M., MARTIN, R. R., AND ROSIN, P. L. 2008. Fast mesh segmentation using random walks. In *Proceedings of the ACM Symposium on Solid and Physical Modeling*. 183–191.

LEE, Y., LEE, S., SHAMIR, A., COHEN-OR, D., AND SEIDEL, H.-P. 2005. Mesh scissoring with minima rule and part salience. *Comput. Aid. Geom. Des. 22,* 5, 444–465.

LELLMANN, J. AND SCHNÖRR, C. 2011. Continuous multiclass labeling approaches and algorithms. CoRR abs/1102.5448.

LÉVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph. 21*, 362–371.

LÉVY, B. AND ZHANG, R. H. 2010. Spectral geometry processing. In *ACM SIGGRAPH Course Notes*.

LIU, R. AND ZHANG, H. 2004. Segmentation of 3D meshes through spectral clustering. In *Proceedings of the Pacific Conference on Computer Graphics and Applications*. 298–305.

LIU, R. AND ZHANG, H. 2007. Mesh segmentation via spectral embedding and contour analysis. *Comput. Graph. Forum 26,* 3, 385–394.

MUMFORD, D. AND SHAH, J. 1989. Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math. 42,* 5, 577–685.

NG, A. Y., JORDAN, M. I., AND WEISS, Y. 2001. On spectral clustering: Analysis and an algorithm. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 849–856.

NIKOLOVA, M., ESEDOGLU, S., AND CHAN, T. F. 2006. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J. Appl. Math. 66,* 5, 1632–1648.

POCK, T., CHAMBOLLE, A., CREMERS, D., AND BISCHOF, H. 2009a. A convex relaxation approach for computing minimal partitions. In *Computer Vision and Pattern Recognition*, 810–817.

POCK, T., CREMERS, D., BISCHOF, H., AND CHAMBOLLE, A. 2009b. An algorithm for minimizing the piecewise smooth Mumford-Shah functional. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

POLITO, M. AND PERONA, P. 2001. Grouping and dimensionality reduction by locally linear embedding. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 1255–1262.

POPOV, L. D. 1980. A modification of the Arrow-Hurwicz method for search of saddle points. *Math. Notes 28,* 5, 845–848.

ROCKAFELLAR, R. T. 1970. *Convex Analysis*. Princeton University Press.

SHAMIR, A. 2008. A survey on mesh segmentation techniques. *Comput. Graph. Forum 27,* 6, 1539–1556.

SHAMIR, A., SHAPIRA, L., AND COHEN-OR, D. 2006. Mesh analysis using geodesic mean-shift. *Vis. Comput. 22*, 99–108.

SHAPIRA, L., SHAMIR, A., AND COHEN-OR, D. 2008. Consistent mesh partitioning and skeletonisation using the shape diameter function. *Vis. Comput. 24,* 4, 249–259.

SHLAFMAN, S., TAL, A., AND KATZ, S. 2002. Metamorphosis of polyhedral surfaces using decomposition. *Comput. Graph. Forum 21*, 219–228.

VESE, L. A. AND CHAN, T. F. 2002. A multiphase level set framework for image segmentation using the mumford and shah model. *Int. J. Comput. Vis. 50,* 3, 271–293.

VON LUXBURG, U. 2007. A tutorial on spectral clustering. *Statist. Comput. 17,* 4, 395–416.

WU, C., DENG, J., CHEN, F., AND TAI, X. 2009. Scale-Space analysis of discrete filtering over arbitrary triangulated surfaces. *SIAM J. Imaging Sci. 2,* 2, 670–709.

YAMAUCHI, H., LEE, S., LEE, Y., OHTAKE, Y., BELYAEV, A., AND SEIDEL, H.-P. 2005. Feature sensitive mesh segmentation with mean shift. In *Proceedings of the International Conference on Shape Modeling and Applications*. 238–245.

ZELNIK-MANOR, L. AND PERONA, P. 2004. Self-Tuning Spectral Clustering. In *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*.

ZHANG, J., WU, C., CAI, J., ZHENG, J., AND TAI, X.-C. 2010. Mesh snapping: Robust interactive mesh cutting using fast geodesic curvature flow. *Comput. Graph. Forum 29,* 2, 517–526.

Received September 2011; accepted December 2011