Spectral Mesh Segmentation via ℓ_0 Gradient Minimization

Weihua Tong, Xiankang Yang, Maodong Pan, and Falai Chen

Abstract—Mesh segmentation is a process of partitioning a mesh model into meaningful parts — a fundamental problem in various disciplines. This paper introduces a novel mesh segmentation method inspired by sparsity pursuit. Based on the local geometric and topological information of a given mesh, we build a Laplacian matrix whose Fiedler vector is used to characterize the uniformity among elements of the same segment. By analyzing the Fiedler vector, we reformulate the mesh segmentation problem as a ℓ_0 gradient minimization problem. To solve this problem efficiently, we adopt a coarse-to-fine strategy. A fast heuristic algorithm is first devised to find a rational coarse segmentation, and then an optimization algorithm based on the alternating direction method of multiplier (ADMM) is proposed to refine the segment boundaries within their local regions. To extract the inherent hierarchical structure of the given mesh, our method performs segmentation in a recursive way. Experimental results demonstrate that the presented method outperforms the state-of-the-art segmentation methods when evaluated on the Princeton Segmentation Benchmark, the LIFL/LIRIS Segmentation Benchmark and a number of other complex meshes.

Index Terms—Mesh segmentation, spectral analysis, ℓ_0 gradient minimization, ADMM

1 Introduction

Mesh segmentation has become a key component of many applications in computer graphics and geometric modelling, such as texture mapping, parametrization, shape understanding and processing, component-based shape synthesis and retrieval. It provides a high-level representation of the underlying mesh, from which many applications can benefit. As a result, there has been an immense amount of effort dedicated to the problem of mesh segmentation over the past two decades. Recent surveys can be found in [1], [2].

However, the problem of mesh segmentation is challenging and remains open in a sense. The task of segmentation is to partition a given mesh into meaningful and disjoint sub-meshes that are connected in their interiors. But the concept of meaningful is highly subjective and often application dependent, e.g., semantic meaning, functional meaning, or human visual perception. Therefore, the mesh segmentation problem is not well posed and often driven by specific applications. In addition, several key aspects should be taken into account simultaneously to obtain a good segmentation, such as geometric and topological information, feature, salience, hierarchical structure, user interaction and even prior knowledge.

Currently, most popular methodologies [1] for mesh segmentation include hierarchical clustering, region growing, surface fitting, graph cut, boundary detection, spectral analysis, topological methods, variational methods, machine learning, etc. Although supervised methods such as conditional random field (CRF) [3], projective analysis [4] and deep convolutional neural networks (CNN) [5] have been proven to perform well, their performance heavily depends on the quality and size of training data. Among unsupervised methods, spectral analysis has achieved better performance and been investigated by many researchers recently [6], [7], [8], [9], [10]. Spectral methods first construct a proper matrix derived from the tasks at hand, which usually reflects the local geometric and topological attributes of the underlying mesh. By analyzing the eigenvalues and eigenvectors of the matrix, the global shape properties of the mesh model can be captured. Therefore spectral methods can be extensively applied in other tasks of mesh processing [11], [12]. The main differences between different spectral methods lie in the construction of the Laplacian matrix, the set of eigenvalues and eigenvectors employed, and the way of using eigenstructures to obtain a proper solution.

One of the most well-known techniques is spectral clustering [13], [14], which produces a k-way segmentation by performing a simple clustering (e.g., k-means) on the first k eigenvectors corresponding to the smallest nonzero eigenvalues of a Laplacian matrix. However, simple clustering method often leads to jaggy boundaries between different parts. In addition, some non-informative eigenvectors within the first k eigenvectors may result in over-segmentation as reported in [6]. Moreover, the resulting segmentation is closely related to the number of clusters which is often chosen heuristically.

In this paper, we propose a new method for unsupervised mesh segmentation inspired by sparsity pursuit. Based on the local geometric and topological information of a given mesh, our method first constructs an affinity matrix and converts it into a Laplacian matrix. Then we compute the Fiedler

• *The authors are with the School of Mathematical Sciences, University of Science and Technology of China, Hefei, Anhui 230026, P. R. China. E-mail: {tongwh, chenfl}@ustc.edu.cn, {yangxk, mdpan}@mail.ustc.edu.cn.*

Manuscript received 11 Mar. 2018; revised 5 Nov. 2018; accepted 11 Nov. 2018. Date of publication 19 Nov. 2018; date of current version 4 Mar. 2020. (Corresponding author: Weihua Tong)
Recommended for acceptance by S. Hu.
Digital Object Identifier no. 10.1109/TVCG.2018.2882212

Fig. 1. Mesh segmentation results by our method on some models of the Princeton Segmentation Benchmark [20].

vector of the Laplacian matrix, i.e., the eigenvector corresponding to the smallest nonzero eigenvalue. We observe that the coordinate values of the Fiedler vector can be used to characterize the uniformity among elements of the same part. If we are able to find a piecewise constant vector to approximate the Fielder vector, then we can obtain a segmentation of the given mesh. Based on these observations, we reformulate the mesh segmentation problem as a ℓ_0 gradient minimization problem. From the viewpoint of sparsity representation, the number of segment boundary edges compared to the whole number of edges in the given mesh are very sparse.

The core problem we address is how to solve the ℓ_0 gradient minimization problem efficiently. Due to the nonconvex and discontinuous nature of the ℓ_0 norm, our optimization problem is NP-hard. Although several algorithms [15], [16], [17], [18], [19] have been developed recently for solving the ℓ_0 gradient minimization problem on edge-preserving filtering in the literature, none of them can be used to find a satisfying solution for our problem. The reason is that the solution in the context of segmentation problem should be very sparse. In this paper, we propose an efficient method to solve the ℓ_0 gradient minimization problem. The key observation is that the coordinate values of the Fiedler vector change smoothly within each segment while they have sharp transitions on the boundaries between different segments. Furthermore, the Fiedler vector is capable of giving a linear ordering of mesh vertices or faces, i.e., a 1-dimensional embedding. Consequently, we can obtain a rational coarse segmentation by approximating the ordered Fiedler vector with a piecewise constant function. A fast heuristic algorithm is devised to find such a coarse solution. Since we don't take into account the topology of the given mesh, the segment boundaries of coarse segmentation may fail to align with the desired boundaries or be jaggy. To tackle this problem, we propose to solve the ℓ_0 gradient minimization problem individually within the local regions enclosing the segment boundaries. Using the re-weighting technique, we convert the ℓ_0 minimization problem into a series of weighted ℓ_1 optimization problem which can be easily solved by standard techniques such as the alternating direction method of multipliers (ADMM). Compared with the existing ℓ_0 gradient minimization algorithms, our method not only achieves better results but also is much faster.

In summary, the main contributions of our work are as follows:

1) From sparse representation point of view, a new mesh segmentation model is introduced. We use ℓ_0 norm to count the number of edges comprising segment boundaries, and reformulate mesh segmentation as an ℓ_0 gradient minimization problem. Our method employs the Fiedler vector to characterize the uniformity, and performs segmentation in a recursive way.
2) We propose an efficient algorithm to solve the ℓ_0 gradient minimization problem, which adopts a coarse-to-fine strategy. Our algorithm first finds a rational coarse segmentation by approximating the ordered Fiedler vector with a piecewise constant function, and then refines the segment boundaries by solving the ℓ_0 gradient minimization problem locally. To achieve these goals, a fast heuristic algorithm and an algorithm based on ADMM are developed respectively.
3) As shown by experimental results on standard datasets, our method outperforms the state-of-the-art mesh segmentation methods (see Fig. 1 for a snapshot). In addition, we demonstrate that the presented method is highly suitable for detecting sharp features on CAD models.

2 Related Work

In this section, we review some related work on spectral mesh segmentation and ℓ_0 gradient minimization. For a

comprehensive survey on these topics, the reader is referred to [1], [2] and references therein.

2.1 Spectral Mesh Segmentation

The seminal work of Shi and Malik [13] regards image segmentation as a graph partitioning problem, which can be solved by the normalized cut (NCut) algorithm. As reported in [21], the NCut method for mesh segmentation often misses the meaningful components or yields the jaggy boundaries no matter which strategy is used. The work of Liu and Zhang [14] uses the pairwise face distances proposed by Katz et al. [21] to define the affinity matrix, and performs k-means clustering on the k largest eigenvectors of normalized affinity matrix. In [22], the authors segmented a mesh into two parts by using 1-dimensional embedding spaces that are computed from distances between two chosen faces and the set of other faces. A linear search is applied to find an optimal cut, which is guided by part salience. To simplify segmentability analysis and sampling tasks, Liu and Zhang [23] further proposed to perform these tasks over a contour obtained from the 2D spectral embedding. However, these methods achieve good results only on relatively simple models, and a post-processing step for smoothing or shortening of the segment boundaries are usually required.

Lai et al. [24] presented a segmentation method based on random walks. By automatically choosing a set of seeds, they first oversegmented the given mesh and then merge the initial segments to obtain the final segmentation. Using a concavity-sensitive weighting scheme, Au et al. [10] constructed a set of scalar fields that can be used to locate concave creases and seams. They then sampled these fields to get the candidate isolines and found the final cuts by a greedy algorithm. Wang et al. [7] computed a single segmentation field by appropriately selecting and combining some sub-eigenvectors of a Laplacian operator. The segment boundaries are identified by a divide-merge algorithm using some cut scores. However, both of the methods are not suitable for detecting regions or parts with nonconcave boundaries. In [9], Zhang et al. presented a method by extending the Mumford-Shah image segmentation model to mesh segmentation. They utilized a data term defined by the first k eigenvectors of a dual Laplacian matrix to measure segment coherence, and employed a TV regularization term to approximate the length of the segment boundaries. From perspective of cognitive studies, Chahhou et al. [8] introduced a new adjacency matrix that encodes the connectivity of the mesh using the minima rule. They proposed to perform 1-spectral clustering for finding the optimal Cheeger cut instead of standard spectral clustering. Recently, Theologou et al. [6] devised a heterogeneous graph which encodes local geometric features and patch affinities together. In terms of the nodal-set and nodal-domain theory, they presented a new scheme for eigenvector analysis where each eigenvector is analyzed separately and the segmentation is carried out iteratively. Different from existing spectral methods, we reformulate mesh segmentation as an ℓ_0 gradient minimization problem, where the Fiedler vector is used to characterize the uniformity among elements of the same segment and the number of edges comprising segment boundaries is measured by ℓ_0 norm.

2.2 ℓ_0 Gradient Minimization

During the past two decades, sparse representation [25] and compressed sensing [26] have attracted considerable attention in areas of signal processing, applied mathematics, image processing and computer graphics. For a recent development of this technique in computer graphics and geometric processing, the reader is referred to [27]. In the following, we briefly review related work in edge-preserving smoothing through ℓ_0 minimization. To achieve edge-preserving smoothing of images, Xu et al. [15] proposed an optimization framework which uses ℓ_0 norm to measure the sparsity of gradients. Adopting the alternating optimization strategy and half quadratic splitting scheme, they found approximate solutions by iteratively solving two subproblems that have closed-form solutions. To extend edge-preserving smoothing from images to surfaces, He and Schaefer [16] introduced a discrete differential operator for capturing planarity of surfaces. A fairing term was integrated into the optimization framework to diminish folded triangles and improve the quality of mesh. Based on a fused coordinate descent framework, Cheng et al. [17] presented another algorithm to solve the ℓ_0 minimization problem. They solved a series of subproblems individually, and fused the neighboring variables into a single variable if their values are equal. By combining the coordinate descent step and the fusion step into a single step, Nguyen and Brown [18] proposed a region-based method that guarantees the objective function to monotonically decrease. Recently, Ono [19] reformulated the ℓ_0 minimization problem as a constrained optimization problem, which minimizes a quadratic data-fidelity term subject to the constraints that the ℓ_0 gradient is less than a user-specified threshold. He also presented an ADMM based algorithm for computing an approximate solution. However, these algorithms are mainly designed for edge-preserving smoothing, and are not suitable for mesh segmentation problem by our test. In this paper, we will propose a new algorithm to solve the ℓ_0 gradient minimization problem efficiently for mesh segmentation problem.

3 Preliminaries

Before going into the details of our method, we present some preliminary knowledge about mesh segmentation and spectral method.

3.1 Mesh Segmentation

In geometric modeling and processing, a *3D polygonal mesh* $\mathcal{M}$ is usually represented by a tuple (V, E, F) of vertices $V = \{v_i \,|\, v_i \in \mathbb{R}^3,\, i = 1, 2, \ldots, n\}$, edges $E \subset V \times V = \{e_{ij} = (v_i, v_j) \,|\, v_i, v_j \in V,\, i \neq j\}$, and faces F which are usually triangles $F \subset V \times V \times V = \{f_{ijk} = (v_i, v_j, v_k) \,|\, v_i, v_j,\, v_k \in V,\, i \neq j,\, i \neq k, j \neq k\}$ or other types of planar polygons. Let $F' \subset F$ be a sub-set of faces, and let V' be the set of all vertices that are included in F'. A *sub-mesh* $\mathcal{M}'$ is defined as the tuple $\mathcal{M}' = (V', E', F')$, where $E' = \{e_{ij} = (v_i, v_j) \in E \,|\, v_i, v_j \in V'\}$. Since disconnected components of a polygonal mesh can be processed individually, without loss of generality, we may assume $\mathcal{M}$ is connected for simplicity of presentations.

A *mesh segmentation problem* can be described as: Given a polygonal mesh $\mathcal{M} = (V, E, F)$, a segmentation of $\mathcal{M}$ is to

find a set of sub-meshes $\Sigma = \{\mathcal{M}_i \,|\, \mathcal{M}_i = (V_i, E_i, F_i),\ i = 1, 2, \ldots, k\}$, which is given by a partition of F into k sub-sets $\{F_i \,|\, F_i \subset F,\ i = 1, 2, \ldots, k\}$ such that

1) $F = F_1 \cup F_2 \cup \cdots \cup F_k$;
2) $\mathcal{M}_i^\circ \cap \mathcal{M}_j^\circ = \emptyset,\ \forall 1 \leqslant i \neq j \leqslant k$;
3) $\mathcal{M}_i$ is connected, $\forall i = 1, 2, \ldots, k$.

Here, $\mathcal{M}_i^\circ$ and $\mathcal{M}_j^\circ$ denote the interior of $\mathcal{M}_i$ and $\mathcal{M}_j$ respectively. In the past two decades, many segmentation methods have been developed and among them spectral method has gained in popularity.

3.2 Spectral Method

Spectral methods usually first introduce the *affinity matrix*, which defines its entries according to the similarity between adjacent nodes of graph. In order to partition faces instead of vertices, we consider the dual graph $G_d = (V_d, E_d)$ of the given mesh $\mathcal{M}$. Here the vertex set V_d is exactly the face set F of $\mathcal{M}$, and E_d is the set of dual edges, i.e., for every pair of adjacent faces in $\mathcal{M}$ there corresponds to an edge of G_d.

We now construct an affinity matrix with respect to G_d. For each pair of adjacent faces $f_i, f_j \in F$ with a common edge $e \in E$, their angular distance is defined as

$$\widetilde{d}(f_i, f_j) = \eta\big[1 + \cos(\alpha_{ij})\big] = \frac{\eta}{2}\|N(f_i) - N(f_j)\|^2, \quad (1)$$

where α_{ij} is the face angle (i.e., the internal angle with respect to the mesh) between f_i and f_j, $N(f_i)$ and $N(f_j)$ are the unit normal vectors of the faces f_i and f_j respectively, and η is a user-specified parameter related to the property of the edge e. In the context of mesh segmentation, we typically set $\eta = 1.0$ for concave edges and a comparatively small value (e.g., 0.1) for convex edges to obey the minima rule. Here an edge e is called a concave edge if the face angle α_{ij} is great than π; otherwise it is a convex edge. However, we may set $\eta = 1.0$ for convex edges and $\eta = 0.1$ for concave edges if we are aiming at detecting sharp features located at convex edges. The different choice of η provides us flexible control of mesh segmentation and sharp feature detection. Let $\overline{d}$ be the average angular distance over all interior edges in $\mathcal{M}$, then we normalize $\widetilde{d}(f_i, f_j)$ as

$$d(f_i, f_j) = \frac{\widetilde{d}(f_i, f_j)}{\overline{d}}.$$

The similarity factor between triangles f_i and f_j is now defined by

$$s_{ij} = |e| \exp\{-d(f_i, f_j)\},$$

where $|e|$ is the length of the edge e. Finally, we construct the affinity matrix $S \in \mathbb{R}^{|F|\times|F|}$ as follows

$$S_{ij} = \begin{cases} s_{ij}, & \text{if } i \neq j \text{ and } f_i, f_j \text{ share a common edge } e, \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

where $|F|$ denotes the number of faces in F.

The *Laplacian matrix* of the graph G_d is defined as

$$L \triangleq D - S \in \mathbb{R}^{|F|\times|F|},\ D = diag(d_1, d_2, \ldots, d_{|F|}), \quad (3)$$

where d_i is the sum of the ith row of S, $i = 1, 2, \ldots, |F|$. It is easy to show that L is symmetric and positive semi-definite. Furthermore, the smallest eigenvalue of L is 0 and the constant vector $\mathbb{1} = (1, 1, \ldots, 1)^{\mathrm{T}}$ is the corresponding eigenvector. Note that the construction of the Laplacian matrix will affect the segmentation results, and there are many other ways to build it (the reader is referred to [1], [28] for more details). The eigen-system of the Laplacian matrix is capable of describing many properties of the graph [29]. In this paper, we employ the Fiedler vector of the Laplacian matrix for mesh segmentation problem.

To begin with, we present a useful result for our spectral analysis framework.

Proposition 1. *[28] Let G be an undirected and non-negatively weighted graph with n vertices, and L be the Laplacian matrix of graph G. Then the eigenvalue 0 of L has multiplicity k which equals to the number of connected components $G_1, G_2, \ldots, G_k$ in G, and the eigenspace associated with 0 is spanned by the indicator vectors $\mathbb{1}_{G_1}, \mathbb{1}_{G_2}, \ldots, \mathbb{1}_{G_k}$ of those components, i.e.,*

$$\mathbb{1}_{G_i} \in \mathbb{R}^n \text{ and } \big(\mathbb{1}_{G_i}\big)_j = \begin{cases} 1, & \text{if vertex } j \in G_i, \\ 0, & \text{otherwise.} \end{cases}$$

Next we suppose that the given mesh $\mathcal{M}$ is divided into k sub-meshes $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_k$. Assume without loss of generality that the faces of $\mathcal{M}$ are ordered in accordance with the connected components they belong to. Let G_d^i be the dual graph of $\mathcal{M}_i$ and L_i be the Laplacian matrix of G_d^i, $i = 1, 2, \ldots, k$. Suppose $\widetilde{G}_d = \cup_{i=1}^k G_d^i$, then the Laplacian matrix of $\widetilde{G}_d$ is

$$\widetilde{L} = diag(L_1, L_2, \ldots, L_k).$$

Since the graph $\widetilde{G}_d$ comes from the graph G_d by removing some edges with low weights, the matrix L can be viewed as a perturbation of the matrix $\widetilde{L}$, that is,

$$L = \widetilde{L} + P,$$

where P is the perturbation matrix and is determined by the similarity factors between the boundaries of the sub-meshes $\mathcal{M}_i$, $i = 1, 2, \ldots, k$. Obviously, two adjacent faces along segment boundary with large angular distance contribute small value to the matrix P.

The following theorem from matrix perturbation theory lays the foundation for our method presented in the next section.

Theorem 1. *[28] (Davis-Kahan) Let L be a perturbation of a symmetric matrix $\widetilde{L} \in \mathbb{R}^{|F|\times|F|}$, and let $I_1 \subset \mathbb{R}$ be an interval. The set of eigenvalues of L contained in I_1 is denoted as $\lambda_{I_1}(L)$, and the eigenspace corresponding to $\lambda_{I_1}(L)$ is denoted as V_1. For matrix $\widetilde{L}$, $\lambda_{I_1}(\widetilde{L})$ and $\widetilde{V}_1$ are defined in a similar way. Let*

$$\delta = \min_{i,j}\Big\{|\lambda_i - \widetilde{\lambda}_j| : \lambda_i \in \lambda_{I_1}(L),\ \widetilde{\lambda}_j \in \lambda(\widetilde{L}) \setminus \lambda_{I_1}(\widetilde{L})\Big\} > 0,$$

then the distance $d(V_1, \widetilde{V}_1) \triangleq \|\sin\Theta(V_1, \widetilde{V}_1)\|_F$ between V_1 and $\widetilde{V}_1$ is bounded by

$$d(V_1, \widetilde{V}_1) \leqslant \frac{\|P\|_F}{\delta},\ P = L - \widetilde{L},$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, and $\Theta(V_1, \widetilde{V}_1)$ is the canonical angle.

4 METHOD

Now we describe our mathematical model and algorithms for mesh segmentation based on spectral analysis of the dual graph of a given mesh.

4.1 Spectral Analysis Framework via ℓ_0 Minimization

The Davis-Kahan theorem tells us that the eigenspace W_L corresponding to the k smallest eigenvalues of L is very close to the eigenspace $W_{\widetilde{L}}$ corresponding to the k smallest eigenvalues of $\widetilde{L}$ (i.e., the eigenvalue 0 with multiplicity k) when the perturbation $\|P\|_F$ is small. By Proposition 1, the eigenspace $W_{\widetilde{L}}$ is spanned by the indicator vectors $\mathbb{1}_{G_d^1}$, $\mathbb{1}_{G_d^2}, \ldots, \mathbb{1}_{G_d^k}$. Note that an arbitrary vector $\mathbf{v} = (v_1, v_2, \ldots, v_{|F|}) \in \mathbb{R}^{|F|}$ can be regarded as a piecewise constant function defined on the mesh $\mathcal{M}$, where the function value is v_i on the ith face of $\mathcal{M}$. Thus the indicator vector $\mathbb{1}_{G_d^i}$ can be looked at a piecewise constant function over $\mathcal{M}$ whose value is 1 on M_i and 0 otherwise.

Now let $\mathbf{u}_0 \in \mathbb{R}^{|F|}$ be the Fiedler vector of L, i.e., the eigenvector corresponding to the smallest nonzero eigenvalue of L. By Theorem 1 we have

$$\mathbf{u}_0 = c_1 \mathbb{1}_{G_d^1} + c_2 \mathbb{1}_{G_d^2} + \cdots + c_k \mathbb{1}_{G_d^k} + \mathbf{e} \triangleq \mathbf{u} + \mathbf{e},$$

where $\mathbf{u} \in \mathbb{R}^{|F|}$ is a linear combination of the indicator vectors $\mathbb{1}_{G_d^1}, \mathbb{1}_{G_d^2}, \ldots, \mathbb{1}_{G_d^k}$, and $\mathbf{e} \in \mathbb{R}^{|F|}$ is an error vector. We call $\mathbf{u}$ a *piecewise constant vector*, since it can be regarded as a piecewise function over $\mathcal{M}$ which has value c_i over $\mathcal{M}_i$. If $\mathbf{u}$ is computed, then a segmentation $\Sigma = \cup_{i=1}^k \mathcal{M}_i$ of the mesh $\mathcal{M}$ is readily obtained by checking the coordinate values of $\mathbf{u}$, i.e.,

$$f_j \in \mathcal{M}_i \Longleftrightarrow u_j = c_i,\ i = 1, 2, \ldots, k.$$

Hence, mesh segmentation problem can be solved by finding a piecewise constant vector $\mathbf{u}$ to approximate the Fiedler vector $\mathbf{u}_0$. In the following, we reformulate this problem as a sparse representation problem using the ℓ_0 gradient minimization.

Clearly, we would like to minimize the approximation error between $\mathbf{u}_0$ and $\mathbf{u}$, i.e.,

$$\min_{\mathbf{u}} \frac{1}{2} \|\mathbf{u} - \mathbf{u}_0\|_2^2. \tag{4}$$

On the other hand, we wish $\mathbf{u}$ be a piecewise constant vector, that is, $\mathbf{u}$ remains unchanged over the most parts of the mesh $\mathcal{M}$, and only has obvious changes near the boundaries of the segmentation of $\mathcal{M}$. This leads to the following minimization

$$\min_{\mathbf{u}} \|\nabla \mathbf{u}\|_0, \tag{5}$$

where $\|\cdot\|_0$ denotes the ℓ_0 norm and is used to measure the sparsity of vectors (i.e., the number of nonzero entries in a vector), and ∇ is the discrete gradient operator which is defined as follows.

For each pair of adjacent faces $f_i, f_j \in F$ with a common edge $e \in E$, we define the jump of $\mathbf{u} = (u_1, u_2, \ldots, u_{|F|})^{\mathrm{T}}$ over e as

$$[\mathbf{u}]_e \triangleq u_i - u_j \text{ or } [\mathbf{u}]_e \triangleq u_j - u_i,$$

where either of them can be randomly chosen and then remains unchanged. The discrete gradient operator $\nabla : \mathbf{u} \rightarrow \nabla \mathbf{u}$ is defined as

$$\nabla \mathbf{u}|_e = \begin{cases} [\mathbf{u}]_e, & \text{if } e \in E \text{ and } e \notin \partial\mathcal{M}, \\ 0, & \text{otherwise,} \end{cases}$$

where $\partial\mathcal{M}$ denotes the set of boundary edges. We rewrite it in matrix form as

$$\nabla \mathbf{u} = (\nabla \mathbf{u}|_1, \nabla \mathbf{u}|_2, \ldots, \nabla \mathbf{u}|_{|E|})^{\mathrm{T}} \triangleq A\mathbf{u},\ A \in \mathbb{R}^{|E| \times |F|},$$

where a row of A takes the form

$$\begin{gathered} \overset{i}{\downarrow} \qquad\qquad \overset{j}{\downarrow} \\ (0, 0, \ldots, 0, \underset{}{1}, 0, \ldots, 0, -1, 0, \ldots, 0) \\ \overset{i}{\downarrow} \qquad\qquad \overset{j}{\downarrow} \\ \text{or } (0, 0, \ldots, 0, -1, 0, \ldots, 0, 1, 0, \ldots, 0) \\ \text{or } (0, 0, \ldots, 0). \end{gathered}$$

Combining the the two objective functions (4) and (5) into one equation arrives at the following ℓ_0 gradient minimization problem

$$\min_{\mathbf{u}} \frac{1}{2} \|\mathbf{u} - \mathbf{u}_0\|_2^2 + \lambda \|A\mathbf{u}\|_0, \tag{6}$$

where λ is a positive number to balance the two objective functions.

In summary, our spectral analysis framework first constructs the Laplacian matrix L as (3) from a given mesh $\mathcal{M}$, and then computes the Fiedler vector $\mathbf{u}_0$. By solving the optimization problem (6), we find a piecewise constant vector $\mathbf{u}$ which produces a segmentation $\Sigma = \{\mathcal{M}_i,\ i = 1, 2, \ldots, k\}$ of $\mathcal{M}$ by checking its coordinate values. Aiming at aligning with human perception and extracting the inherent hierarchical structure of the object, we perform our spectral analysis framework in a recursive way. By doing this, we not only avoid several drawbacks resulting from using the first k eigenvectors of L as explained in Section 1, but also gain more fine control over segmentation by tuning the parameters (e.g., η, λ) at each level.

4.2 Overview

A brief sketch of our method is illustrated in Fig. 2. We first perform our spectral analysis framework on $\mathcal{M}$ and obtain a set of sub-meshes $\Sigma = \{\mathcal{M}_i,\ i = 1, 2, \ldots, k\}$. Then our method extracts the sub-mesh with the largest area from Σ, conducts segmentation, and updates the set Σ accordingly. Here, a priority queue is employed to accomplish this mission. The above procedure repeats until no sub-mesh needs to be segmented further, which can be judged by user or part salience.

The main issue raised by our method is how to solve the ℓ_0 gradient minimization problem (6) efficiently. This problem is quite challenging because the objective function is nonconvex and discontinuous, and hence it is NP-hard. Although several algorithms have been developed to find approximate solutions for edge-preserving filtering, none of them is suited for our mesh segmentation purpose. The underlying reason is that a very sparse solution is demanded for our application. In addition, these algorithms are slow to converge, especially for problems of large sizes.

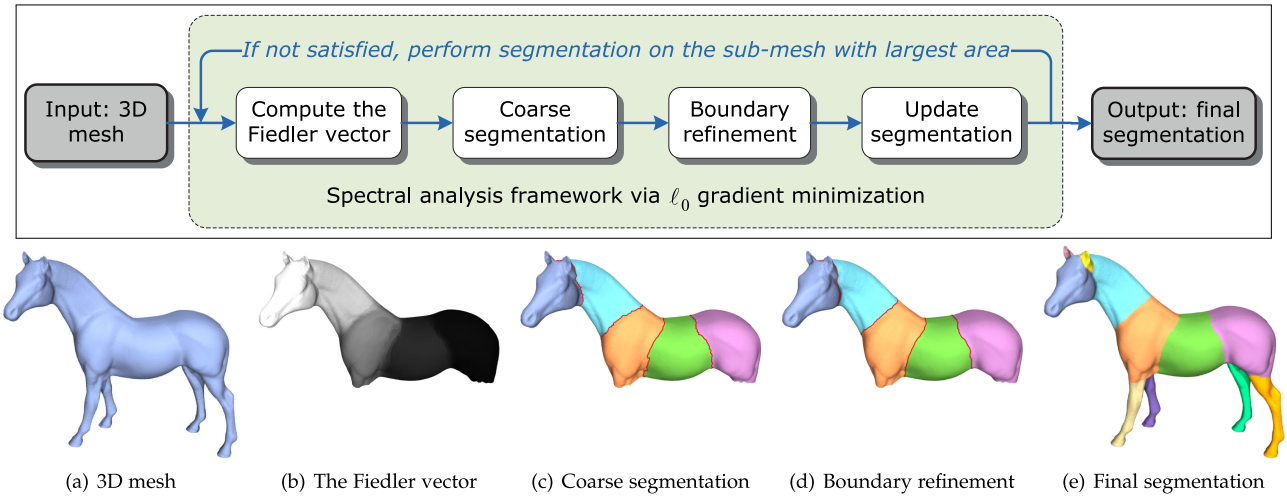


Fig. 2. The framework of our method. (b)-(d) illustrate the key ingredients of our spectral analysis framework on the sub-mesh (the four legs have been segmented) selected from an intermediate segmentation. (a) and (e) show the input and the output respectively.

In the following, we propose a very effective algorithm for solving the ℓ_0 gradient minimization problem, which proceeds from coarse to fine. Our algorithm first finds a rational coarse segmentation by approximating the ordered Fiedler vector with a piecewise constant function, and then refines the coarse segmentation boundaries by solving the ℓ_0 gradient minimization problem individually within their local regions. A re-weighting technique is further applied to convert the ℓ_0 problem into a series of weighted ℓ_1 optimization problem. Since only some convex optimization problems of relatively small sizes are solved, the computation is greatly sped up.

4.3 Coarse Segmentation

Let $\mathbf{u}_0 = (u_1^0, u_2^0, \ldots, u_{|F|}^0)^{\mathrm{T}}$ be the Fiedler vector computed from the Laplacian matrix L, our goal is to find a piecewise constant vector $\mathbf{u}$ to approximate $\mathbf{u}_0$. Instead of solving the ℓ_0 gradient minimization problem (6) directly, we propose a simple algorithm to compute a coarse approximation solution beforehand. By spectral graph theory [12], [30], we know that the Fiedler vector provides an approximate solution to the minimum linear arrangement (MLA) problem. Given the dual graph $G_d = (F, E_d)$, it seeks a permutation $\phi : F \to \{1, 2, \ldots, |F|\}$ of the nodes in F so as to minimize

$$\sum_{i,j=1}^{|F|} S_{ij} |\phi(f_i) - \phi(f_j)|,$$

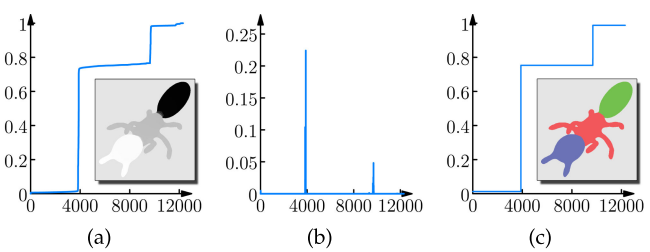


Fig. 3. An example of coarse segmentation: (a) presents the ordered Fiedler vector $\widehat{\mathbf{u}}_0$ of the Ant model; (b) plots the first-order difference of $\widehat{\mathbf{u}}_0$; (c) shows the piecewise constant vector $\widetilde{\mathbf{u}}$ and the result of coarse segmentation ($k = 3$).

where S_{ij} is the similarity factor defined by (2). Let

$$\widehat{\mathbf{u}}_0 = (u^0_{\phi^{-1}(1)}, u^0_{\phi^{-1}(2)}, \ldots, u^0_{\phi^{-1}(|F|)})^{\mathrm{T}} \triangleq (\widehat{u}_1^0, \widehat{u}_2^0, \ldots, \widehat{u}_{|F|}^0)$$

be the vector resulting from sorting the elements of $\mathbf{u}_0$ in increasing order, an approximation to the optimal permutation ϕ is given by

$$\phi(\phi^{-1}(i)) = i, \ i = 1, 2, \ldots, |F|.$$

In other words, the ordered Fielder vector can induce a 1-dimensional embedding of the graph on a line that tries to respect the weights of the graph. This idea has been successfully used to construct streaming meshes [31]. An example of $\widehat{\mathbf{u}}_0$ is illustrated in Fig. 3a. Through the embedding technique, we can approximate the ℓ_0 gradient minimization problem (6) by the following 1-dimensional optimization problem

$$\min_{\widetilde{\mathbf{u}}} \frac{1}{2} \|\widetilde{\mathbf{u}} - \widehat{\mathbf{u}}_0\|_2^2 + \lambda \sum_{i=1}^{|F|-1} |\widetilde{u}_{i+1} - \widetilde{u}_i|_0, \tag{7}$$

where $\widetilde{\mathbf{u}} = (\widetilde{u}_1, \widetilde{u}_2, \ldots, \widetilde{u}_{|F|})^{\mathrm{T}}$. However, this problem is still difficult to solve due to the ℓ_0 norm and the large size of the problem by existing algorithms, e.g., [15], [16], [17], [18], [19].

In order to characterize a piecewise constant vector, it is sufficient to provide the information about the number of segments, the positions of jumps and the value associated with each segment. However, these unknowns are interrelated and interact on each other in (7). Thus it is difficult to solve them simultaneously. Instead we employ a step-by-step approach to tackle the problem based on the following observation. Obviously, if the number of segments is fixed to k and the indices of jumps are fixed to $i_1, i_2, \ldots, i_{k-1}$, then the value associated with each segment can be explicitly determined by minimizing the 1-dimensional optimization problem (7) as follows

$$c_l = \frac{\sum_{j=i_{l-1}}^{i_l - 1} \widehat{u}_j^0}{i_l - i_{l-1}}, \ l = 1, 2, \ldots, k, \tag{8}$$

with $i_0 = 1$ and $i_k = |F| + 1$. Hence, our problem is reduced to finding the number of segments and the indices of jumps.

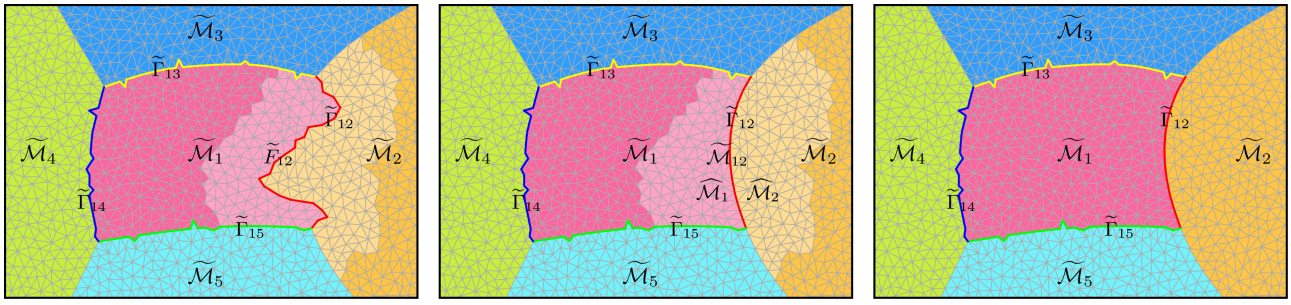


Fig. 4. Illustration of boundary refinement: (a) shows the boundary segment $\widetilde{\Gamma}_{12}$ with its r-ring neighborhood $\widetilde{F}_{12}$ ($r = 6$); (b) shows the new $\widetilde{\Gamma}_{12}$ resulting from Algorithm 2 with the new segmentation $\widehat{\mathcal{M}}_1$ and $\widehat{\mathcal{M}}_2$ of the sub-mesh $\widetilde{\mathcal{M}}_{12}$; (c) shows the result of boundary refinement for $\widetilde{\Gamma}_{12}$.

If we further fix the number of segments k, the second term in (7) will equal to a constant (i.e., $\lambda k - \lambda$) and can be omitted. By substituting (8) into (7), we have

$$\min_{i_1,i_2,\ldots,i_{k-1}} \sum_{l=1}^{k} \sum_{j=i_{l-1}}^{i_l-1} (c_l - \widehat{u}_j^0)^2 = \min_{i_1,i_2,\ldots,i_{k-1}} \sum_{l=1}^{k} (i_l - i_{l-1}) \mathrm{Var}(\{\widehat{u}_{i_{l-1}}^0, \widehat{u}_{i_{l-1}+1}^0, \ldots, \widehat{u}_{i_l-1}^0\}).$$

In addition, we observe that the coordinate values of the Fiedler vector $\mathbf{u}_0$ change smoothly within each segment while they have sharp transitions on the boundaries between different segments. This suggests that the jumps of $\widetilde{\mathbf{u}}$ should locate at the indices where the coordinate values of $\widehat{\mathbf{u}}_0$ have a significant change. Since $\widehat{\mathbf{u}}_0$ is a vector with ordered elements, the indices of jumps can be chosen in a straightforward manner by computing the first-order difference of its elements, i.e.,

$$\widehat{u}_2^0 - \widehat{u}_1^0, \widehat{u}_3^0 - \widehat{u}_2^0, \ldots, \widehat{u}_{|F|}^0 - \widehat{u}_{|F|-1}^0,$$

and sorting them in decreasing order as follows

$$\widehat{u}_{i_1}^0 - \widehat{u}_{i_1-1}^0, \widehat{u}_{i_2}^0 - \widehat{u}_{i_2-1}^0, \ldots, \widehat{u}_{i_{k-1}}^0 - \widehat{u}_{i_{k-1}-1}^0, \ldots, \widehat{u}_{i_{|F|}}^0 - \widehat{u}_{i_{|F|}-1}^0.$$

The first $k-1$ indices, i.e., $i_1, i_2, \ldots, i_{k-1}$, are candidates where jumps occur. Let $\widetilde{\Sigma} = \{\widetilde{\mathcal{M}}_i,\ i = 1, 2, \ldots, k\}$ be a coarse segmentation of the mesh $\mathcal{M}$ determined by the set of indices. To avoid producing very small segments, we also require that the number of faces within each segment satisfies

$$\frac{|F_i|}{|F|} > \delta,\ i = 1, 2, \ldots, k,$$

where $|F_i|$ is the number of faces in $\widetilde{\mathcal{M}}_i$, and δ is a threshold that is set to 0.02 by default.

Now the remaining thing is how to determine the number of segments, i.e., k. This is a common problem for most mesh segmentation methods. In this work, we provide two simple ways for selecting k. One is an interactive manner, that is, plot the first-order difference of $\widehat{\mathbf{u}}_0$ and let the user to specify the number k. Figs. 3b and 3c present an example, where it is apparent $k = 3$ is a proper choice. The other way is to set k as a constant, e.g., $k = 2$. In practice, we find that they are sufficient for our method, since we perform segmentation in a recursive way and therefore are less sensitive to the choice of k. Our coarse segmentation algorithm is summarized in Algorithm 1.

Algorithm 1. Coarse Segmentation Algorithm

Input: the Fiedler vector $\mathbf{u}_0$
Output: a piecewise constant vector $\widetilde{\mathbf{u}}$
1: obtain $\widehat{\mathbf{u}}_0$ and ϕ by sorting the elements of $\mathbf{u}_0$ in increasing order;
2: compute the first-order difference of $\widehat{\mathbf{u}}_0$'s elements and plot it as a graph (optional);
3: select k by the user (optional) or set $k = 2$ by default;
4: seek $i_1, i_2, \ldots, i_{k-1}$ through sorting the first-order difference of $\widehat{\mathbf{u}}_0$'s elements in decreasing order;
5: compute the average value associated with each segment by (8);
6: determine $\widetilde{\mathbf{u}}$ by $k, i_1, i_2, \ldots, i_{k-1}, c_1, c_2, \ldots, c_k$ and ϕ;
7: **return** $\widetilde{\mathbf{u}}$;

4.4 Boundary Refinement

Using the piecewise constant vector $\widetilde{\mathbf{u}}$ computed from Algorithm 1, we obtain a coarse segmentation of mesh $\mathcal{M}$. Although the meaningful parts are detected, the boundaries between different parts may fail to align with the desired boundaries or be jaggy as shown in Figs. 2c and 8e. The primary reason is that the 1-dimensional optimization problem (7) is only a rough approximation of the original ℓ_0 gradient minimization problem (6). To obtain satisfactory boundaries, we solve the ℓ_0 minimization problem (6) again within the local regions enclosing the segment boundaries. In this way, the problem size is greatly reduced.

Let $\widetilde{\Sigma} = \{\widetilde{\mathcal{M}}_i,\ i = 1, 2, \ldots, k\}$ be a coarse segmentation of $\mathcal{M}$ according to the piecewise constant vector $\widetilde{\mathbf{u}}$. Denote the boundary of the sub-mesh $\widetilde{\mathcal{M}}_i$ by $\widetilde{\Gamma}_i$. We divide $\widetilde{\Gamma}_i$ into distinct segments $\{\widetilde{\Gamma}_{ij}\}$ such that

$$\widetilde{\Gamma}_i = \widetilde{\Gamma}_{ij_1} \cup \widetilde{\Gamma}_{ij_2} \cup \cdots \cup \widetilde{\Gamma}_{ij_d},$$

with $\widetilde{\Gamma}_{ij} = \widetilde{\mathcal{M}}_i \cap \widetilde{\mathcal{M}}_j,\ j = j_1, j_2, \ldots, j_d$, as shown in Fig. 4a.

Then for each boundary segment $\widetilde{\Gamma}_{ij}$, we construct a sub-mesh $\widetilde{\mathcal{M}}_{ij}$ whose faces are

$$\widetilde{F}_{ij} = \left\{ f \in NF_r(v) \cap (\widetilde{F}_i \cup \widetilde{F}_j) \mid \forall v \in \widetilde{\Gamma}_{ij} \right\},$$

where $NF_r(v)$ denotes the r-ring neighborhood of a given vertex v, $\widetilde{F}_i$ and $\widetilde{F}_j$ are the set of faces associated with $\widetilde{\mathcal{M}}_i$ and $\widetilde{\mathcal{M}}_j$ respectively, and r is a user-specified parameter

and is used to adjust the width of the local region defined by $\widetilde{\mathcal{M}}_{ij}$.

Next we solve the ℓ_0 gradient minimization problem (6) by restricting the domain to the sub-mesh $\widetilde{\mathcal{M}}_{ij}$, i.e.,

$$\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{u}-\widetilde{\mathbf{u}}_0\|_2^2 + \lambda\|\widetilde{A}\mathbf{u}\|_0, \tag{9}$$

where $\widetilde{\mathbf{u}}_0 = \mathbf{u}_0|_{\widetilde{\mathcal{M}}_{ij}} \in \mathbb{R}^{|\widetilde{F}_{ij}|}$, $\widetilde{A} = A|_{\widetilde{\mathcal{M}}_{ij}} \in \mathbb{R}^{|\widetilde{E}_{ij}|\times|\widetilde{F}_{ij}|}$, and $\widetilde{E}_{ij}$ denotes the set of edges in $\widetilde{\mathcal{M}}_{ij}$. The above minimization problem can be rewritten as

$$\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{u}-\widetilde{\mathbf{u}}_0\|_2^2 + \lambda\|\mathbf{y}\|_0 \quad \text{subject to } \mathbf{y} = \widetilde{A}\mathbf{u},$$

where $\mathbf{y}$ is an auxiliary variable. Using the iterative scheme [32] for approximating ℓ_0 norm and taking the length of edge into consideration, we convert it into a sequence of weighted ℓ_1 minimization problems:

$$\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{x}-\widetilde{\mathbf{u}}_0\|_2^2 + \lambda\sum_{i=1}^{|\widetilde{E}_{ij}|} w_i^l \cdot \widetilde{r}_i|y_i| \quad \text{subject to } \mathbf{y} = \widetilde{A}\mathbf{u}, \tag{10}$$

with $\widetilde{r}_i = \frac{|\widetilde{e}_i|}{\bar{e}}$, $|\widetilde{e}_i|$ is the length of edge $\widetilde{e}_i$, $\bar{e}$ is the average length of all edges, $\mathbf{y} = (y_1, y_2, \ldots, y_{|\widetilde{E}_{ij}|})^{\mathrm{T}}$ and $l = 1, 2, \ldots, L$. The weights $\{w_i^l\}$ are successively updated as

$$w_i^l = \frac{1}{\rho + \left|(\widetilde{A}\mathbf{x}^{(l-1)})_i\right|},\ i = 1, 2, \ldots, |\widetilde{E}_{ij}|, \tag{11}$$

where ρ is a small positive constant for enhancing numerical stability ($\rho = 1.0\mathrm{E}-7$ by default), and $\mathbf{x}^{(l-1)}$ is the solution to the previous step (starting with $\mathbf{x}^{(0)} = \widetilde{\mathbf{u}}_0$).

Now we turn to the algorithm for solving the weighted ℓ_1 minimization problem (10). Since it is a convex optimization problem with equality constraints and its objective function is separable, the alternating direction method of multipliers (ADMM) is a good choice for solving the problem [33]. Let the augmented Lagrangian function be

$$\begin{aligned} L_\gamma(\mathbf{x},\mathbf{y},\mathbf{z}) = &\frac{1}{2}\|\mathbf{x}-\widetilde{\mathbf{u}}_0\|_2^2 + \lambda\sum_{i=1}^{|\widetilde{E}_{ij}|} w_i^l \cdot \widetilde{r}_i|y_i| \\ &+ \frac{1}{\gamma}\mathbf{z}^{\mathrm{T}}(\widetilde{A}\mathbf{x}-\mathbf{y}) + \frac{1}{2\gamma}\|\widetilde{A}\mathbf{x}-\mathbf{y}\|_2^2, \end{aligned}$$

ADMM consists of the following iterations

$$\begin{cases} \mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} L_\gamma(\mathbf{x}, \mathbf{y}^{(n)}, \mathbf{z}^{(n)}) \\ = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\widetilde{\mathbf{u}}_0\|_2^2 + \frac{1}{2\gamma}\|\widetilde{A}\mathbf{x} - \mathbf{y}^{(n)} + \mathbf{z}^{(n)}\|_2^2, & \text{(12a)} \\ \mathbf{y}^{(n+1)} = \arg\min_{\mathbf{y}} L_\gamma(\mathbf{x}^{(n+1)}, \mathbf{y}, \mathbf{z}^{(n)}) \\ = \arg\min_{\mathbf{y}} \lambda\sum_{i=1}^{|\widetilde{E}_{ij}|} w_i^l \cdot \widetilde{r}_i|y_i| + \frac{1}{2\gamma}\|\widetilde{A}\mathbf{x}^{(n+1)} - \mathbf{y} + \mathbf{z}^{(n)}\|_2^2, & \text{(12b)} \\ \mathbf{z}^{(n+1)} = \mathbf{z}^{(n)} + \widetilde{A}\mathbf{x}^{(n+1)} - \mathbf{y}^{(n+1)}, & \text{(12c)} \end{cases}$$

where $\mathbf{z}$ is the scaled dual variable, and γ is a positive real number. The first subproblem (12a) is a strictly convex quadratic minimization, and it boils down to solving a linear system of equations:

$$\left(I + \gamma^{-1}\widetilde{A}^{\mathrm{T}}\widetilde{A}\right)\mathbf{x}^{(n+1)} = \widetilde{\mathbf{u}}_0 + \gamma^{-1}\widetilde{A}^{\mathrm{T}}\left(\mathbf{y}^{(n)} - \mathbf{z}^{(n)}\right), \tag{13}$$

where I is the identity matrix. Denote $\mathbf{p} = \widetilde{A}\mathbf{x}^{(n+1)} + \mathbf{z}^{(n)}$, the second subproblem (12b) can be written as

$$\begin{aligned} &\min_{\mathbf{y}} \lambda\sum_{i=1}^{|\widetilde{E}_{ij}|} w_i^l \cdot \widetilde{r}_i|y_i| + \frac{1}{2\gamma}\|\mathbf{y}-\mathbf{p}\|_2^2 \\ &= \sum_{i=1}^{|\widetilde{E}_{ij}|} \min_{y_i}\left[\lambda w_i^l \cdot \widetilde{r}_i|y_i| + \frac{1}{2\gamma}(y_i - p_i)^2\right], \end{aligned}$$

with $\mathbf{p} = (p_1, p_2, \ldots, p_{|\widetilde{E}_{ij}|})^{\mathrm{T}}$. The objective function is decomposed into many simple piecewise quadratic functions, which can be solved individually. With some calculation (see Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2018.2882212 for details), the solution to the second subproblem (12b) is given by

$$y_i = \begin{cases} p_i - \mathrm{sgn}(p_i)\lambda w_i^l \cdot \widetilde{r}_i\gamma, & \text{if } \lambda w_i^l \cdot \widetilde{r}_i\gamma > |p_i|, \\ 0, & \text{otherwise}, \end{cases} \tag{14}$$

where $\mathrm{sgn}(x)$ denotes the sign function of a real number x, and $i = 1, 2, \ldots, |\widetilde{E}_{ij}|$. The parameter γ can be used to balance the primal residual and the dual residual, and there are several ways to choose it (please refer to [33] for details). In this work, we empirically set $\gamma = 0.01$ as a constant for the sake of simplicity and speedup, which is proven to work well in practice. The ADMM iterations run until the sequence of $\{\mathbf{x}^{(n)}\}$ has almost no change or the maximum number of iterations reaches. A detailed description of our ADMM is given in Algorithm 2. In our experiments, we choose $\varepsilon = 1.0\mathrm{E}-7$, $L = 5$ and $N = 200$, which are enough to achieve satisfactory results.

Algorithm 2. The ADMM Algorithm

Input: the Fiedler vector $\widetilde{\mathbf{u}}_0$
Output: a piecewise constant vector $\mathbf{x}$

$\mathbf{x}^{(0)} \leftarrow \widetilde{\mathbf{u}}_0$;
for $l = 1$ to L **do**
  compute the weights $\{w_i^l\}$ as (11);
  $\mathbf{y}^{(0)} \leftarrow \widetilde{A}\mathbf{x}^{(0)}$, $\mathbf{z}^{(0)} \leftarrow \mathbf{y}^{(0)}$, $n \leftarrow 0$;
  repeat
    $\mathbf{x}^{(n+1)} \leftarrow \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\widetilde{\mathbf{u}}_0\|_2^2 + \frac{1}{2\gamma}\|\widetilde{A}\mathbf{x} - \mathbf{y}^{(n)} + \mathbf{z}^{(n)}\|_2^2$, i.e., (13);
    $\mathbf{y}^{(n+1)} \leftarrow \arg\min_{\mathbf{y}} \lambda\sum_{i=1}^{|\widetilde{E}_{ij}|} w_i^l \cdot \widetilde{r}_i|y_i| + \frac{1}{2\gamma}\|\widetilde{A}\mathbf{x}^{(n+1)} - \mathbf{y} + \mathbf{z}^{(n)}\|_2^2$, i.e., (14);
    $\mathbf{z}^{(n+1)} \leftarrow \mathbf{z}^{(n)} + \widetilde{A}\mathbf{x}^{(n+1)} - \mathbf{y}^{(n+1)}$;
    $n \leftarrow n + 1$;
  until $\|\mathbf{x}^{(n)} - \mathbf{x}^{(n-1)}\|_2 < \varepsilon|F|$ or $n > N$;
end for
return $\mathbf{x} \leftarrow \mathbf{x}^{(n)}$;

Using the piecewise constant vector $\mathbf{x}$, we partition $\widetilde{\mathcal{M}}_{ij}$ into two sub-meshes such that

$$\widetilde{\mathcal{M}}_{ij} = \widehat{\mathcal{M}}_i \cup \widehat{\mathcal{M}}_j,\ A(\widehat{\mathcal{M}}_i \cap \widetilde{\mathcal{M}}_i) > A(\widehat{\mathcal{M}}_j \cap \widetilde{\mathcal{M}}_j),$$

where $A(\widehat{\mathcal{M}}_i \cap \widetilde{\mathcal{M}}_i)$ denotes the area of the intersection of $\widehat{\mathcal{M}}_i$ and $\widetilde{\mathcal{M}}_i$, and $A(\widehat{\mathcal{M}}_j \cap \widetilde{\mathcal{M}}_j)$ is defined in a similar way. Then the sub-meshes $\widehat{\mathcal{M}}_i$ and $\widetilde{\mathcal{M}}_j$ is updated by

$$\widetilde{\mathcal{M}}_i \leftarrow (\widetilde{\mathcal{M}}_i \setminus \widetilde{\mathcal{M}}_{ij}) \cup \widehat{\mathcal{M}}_i,\ \widetilde{\mathcal{M}}_j \leftarrow (\widetilde{\mathcal{M}}_j \setminus \widetilde{\mathcal{M}}_{ij}) \cup \widehat{\mathcal{M}}_j.$$

Figs. 4b and 4c give an illustration. Finally, the piecewise constant vector $\mathbf{u}$ is determined by

$$\mathbf{u}|_{\widetilde{\mathcal{M}}_i} = c_i \mathbb{1}_{\widetilde{\mathcal{M}}_i},\ \mathbf{u}|_{\widetilde{\mathcal{M}}_j} = c_j \mathbb{1}_{\widetilde{\mathcal{M}}_j},$$

where $c_i = \sum_{f_l \in \widetilde{\mathcal{M}}_i} (\mathbf{u}_0)_{f_l} / |\widetilde{\mathcal{M}}_i|$ and $c_j = \sum_{f_l \in \widetilde{\mathcal{M}}_j} (\mathbf{u}_0)_{f_l} / |\widetilde{\mathcal{M}}_j|$, i.e., the average value of $\mathbf{u}_0$ within $\widetilde{\mathcal{M}}_i$ and $\widetilde{\mathcal{M}}_j$ respectively. An outline of our boundary refinement algorithm is depicted in Algorithm 3.

Algorithm 3. The Boundary Refinement Algorithm

Input: the Fiedler vector $\mathbf{u}_0$ and its coarse approximation $\widetilde{\mathbf{u}}$
Output: a piecewise constant vector $\mathbf{u}$

```
partition M into a set of sub-meshes Σ̃ = {M̃_i} according to ũ;
u ← ũ;
for each M̃_i in Σ̃ do
    find out its boundary Γ̃_i;
    divide Γ̃_i into distinct segments {Γ̃_ij};
    for each Γ̃_ij in Γ̃_i do
        construct sub-mesh M̃_ij;
        ũ_0 ← u_0|_{M̃_ij}, Ã ← A|_{M̃_ij}, R̃ ← R|_{M̃_ij};
        solve the local ℓ0 gradient minimization problem (9) by Algorithm 2 to obtain x;
        partition M̃_ij into M̃_i and M̃_j using x;
        update M̃_i and M̃_j;
        u|_{M̃_i} ← c_i 1_{M̃_i}, u|_{M̃_j} ← c_j 1_{M̃_j};
    end for
end for
return u;
```

4.5 Implementation

The presented algorithms are implemented in C++, and the Eigen [34] is employed to perform matrix computations. In our method, we need to compute the Fiedler vector $\mathbf{u}_0$ of the graph Laplacian matrix L, which is a sparse, symmetric, and positive semi-definite matrix. Consequently, we call the ARPACK [35] through the interface provided by the Eigen for solving large scale eigenvalue problems.

As described in [33], the parameter γ in (12) can be varying or be fixed for each iteration. In order to speed up computation, we prefer to choose γ as a constant. In this situation, the coefficient matrix in (13) remains unchanged for each iteration while the right-hand side changes constantly. Moreover, it is easy to show that the matrix $(I + \gamma^{-1}\widetilde{A}^{\mathrm{T}}\widetilde{A})$ is symmetric, positive definite and sparse. Thus, we first compute the sparse $\mathrm{LDL}^{\mathrm{T}}$ decomposition of the matrix $(I + \gamma^{-1}\widetilde{A}^{\mathrm{T}}\widetilde{A})$ and only execute the back substitution step for solving (13). Our experiments indicate that this simple scheme can achieve a several-fold speedup.

For further details of our implementation, please refer to Appendix B (i.e., http://doi.ieeecomputersociety.org/10.1109/TVCG.2018.2882212), available in the online supplemental material.

5 Results and Discussions

We have extensively tested our segmentation method on a variety of meshes, including all the models from the Princeton Segmentation Benchmark [20], the LIFL/LIRIS Segmentation Benchmark [36], and some complex models downloaded from the Internet. All experiments were performed on a PC with a quad-core Intel i7-3770 @3.40 GHz processor and 8GB RAM.

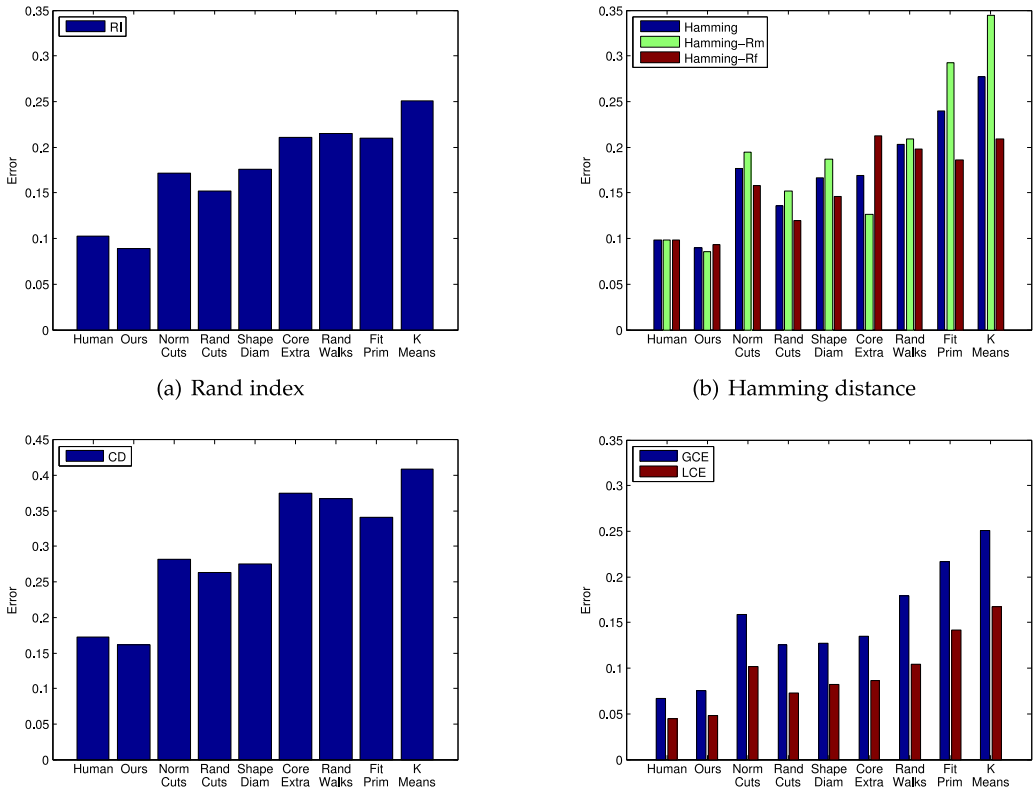
5.1 Quantitative Evaluations

The Princeton Segmentation Benchmark has 4,300 manual segmentations over 380 meshes of 19 categories (20 meshes per category), and it is the most popular one for evaluating mesh segmentation algorithms. The segmentation results of K-Means [37], CoreExtra [38] FitPrim [39], NormCuts [40], RandCuts [40], ShapeDiam [41] and RandWalks [24] are also included. Based on the assumptions that people tend to segment objects consistently and that segmentation algorithms ought to mimic what people to, the human-generated segmentations (11 segmentations for each mesh on average) are regarded as the ground truth. This benchmark offers four quantitative metrics, i.e., Cut Discrepancy, Hamming Distance, Rand Index and Consistency Error to measure the similarity between computer-generated segmentations and human-generated segmentations together with source code for computing these metrics. The smaller values suggest the better segmentation results. We conducted evaluation of our method on the benchmark by the automatic mode and using the default parameters, and a snapshot of our segmentation results in each category is illustrated in Fig. 1. The detailed quantitative results are shown in Fig. 5. To provide comparisons with top performing state-of-the-art methods, e.g., RandCuts [40], CRF-SB19 [3], ConcAware [10], MS [9], SSF [7], WCSeg [42], CNN-SB19 [5] and HG [6], we present the detailed Rand Index score statistics of each category in Table 1. Our method achieves an average Rand Index error 8.9 which clearly outperforms other state-of-the-art methods. It is worth noting that the limited amount of training data and the absence of effective learning models for mesh segmentation lead to a rapid decline in the performance of learning based methods. In addition, we observe that if the training dataset contains quite dissimilar segmentations over a mesh, these learning based methods usually generate fuzzy segmentations (e.g., the bust model in Fig. 6). Therefore, our unsupervised method obtains even better results than the supervised methods [3] and [5].

In addition, we also evaluated our method on the LIFL/LIRIS Segmentation Benchmark which contains 112 manual segmentations over 28 meshes of 5 categories. This benchmark introduces the 3D Normalized Probabilistic Rand Index (3DNPRI), which has high discriminative power. The larger values indicate the better performance. A comparison on 3DNPRI with several methods, e.g., FitPrim [39], ShapeDiam [41], BoundLearn [43] and HG [6], is shown in Table 2. Our method achieves an average 3DNPRI of 0.66 which outperforms other methods again.

5.2 Qualitative Evaluations

Visual Comparisons. Fig. 6 shows the visual comparison results between our method and RandCuts [40], CRF-SB19 [3] and CNN-SB19 [5], where the segmentation results of these methods are provided directly by the original inventors. For the sake of fairness, our segmentation results are produced by choosing the number of segments defined as the median number of segments in the human-generated segmentations



(a) Rand index (b) Hamming distance

(c) Cut discrepancy (d) Global and local consistency error

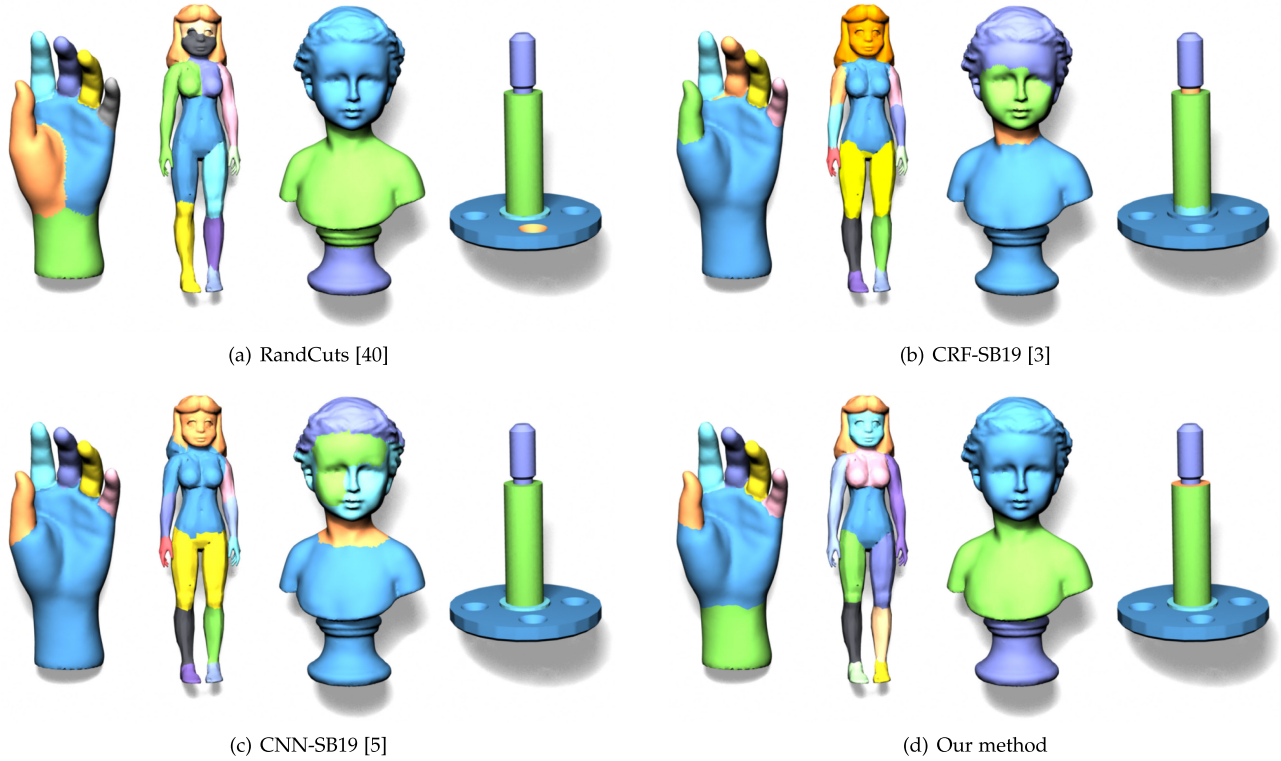
Fig. 5. Quantitative evaluation of our method on the Princeton Segmentation Benchmark [20] using the following metrics: Rand Index, Hamming Distance, Cut Discrepancy and Consistency Error.

TABLE 1
Rand Index Values of our Method and Some State-of-the-Art Methods Across all 19 Categories in the Princeton Segmentation Benchmark [20], where Benchmark Denotes the Error Included in the Manual Segmentations

Method	Benchmark	RandCuts	CRF-SB19	ConcAware	MS	WCSeg	SSF	CNN-SB19	HG	**Ours**
Human	13.8	13.1	11.9	12.3	11.1	12.8	12.8	12.1	12.4	12.8
Cup	13.6	21.9	9.9	21.1	20.4	17.1	14.6	10.0	11.0	11.1
Glasses	10.1	10.1	13.6	9.8	9.4	17.3	11.3	13.7	9.9	7.0
Airplane	9.2	12.2	7.9	12.7	11.1	8.9	13.2	7.6	11.2	8.4
Ant	3.0	2.5	1.9	3.9	2.2	2.1	2.8	1.9	2.0	1.9
Chair	8.9	18.4	5.4	12.1	10.9	10.3	8.4	5.5	7.4	5.7
Octopus	2.4	6.3	1.8	4.1	2.5	2.9	2.6	4.1	2.6	2.2
Table	9.3	38.3	6.2	6.5	10.3	9.1	6.1	6.2	7.0	6.0
Teddy	4.9	4.5	3.1	5.3	3.2	5.6	3.6	3.4	3.9	3.7
Hand	9.1	9.0	10.4	11.5	7.9	11.6	11.0	9.8	10.7	7.5
Plier	7.1	10.9	5.4	7.3	8.9	8.7	8.5	7.2	5.7	5.5
Fish	15.5	29.7	12.9	24.3	29.6	20.3	21.5	14.7	18.6	16.1
Bird	6.2	10.7	10.4	9.7	9.4	10.1	7.8	5.8	7.8	5.7
Armadillo	8.3	9.2	8.0	10.6	8.7	8.1	9.1	6.9	10.3	7.6
Bust	22.0	23.2	21.4	24.4	25.1	26.6	28.6	27.3	25.8	22.0
Mech	13.1	27.7	10.0	12.2	13.1	18.2	12.6	11.4	10.5	9.0
Bearing	10.4	12.4	9.7	17.7	16.6	11.9	14.8	12.1	9.5	8.6
Vase	14.4	13.3	16.0	16.8	12.5	16.1	15.4	15.4	12.1	10.9
Fourleg	14.9	17.4	13.3	18.1	14.4	15.2	16.5	13.7	15.7	17.8
Average	10.3	15.3	9.4	12.7	12.0	12.3	11.6	9.9	10.2	8.9

[20] for each model. It is evident that our method yields more natural segmentation results than the other methods, which further demonstrates the superior performance of our method. Moreover, since we take into account the exactness and smoothness of segment boundaries simultaneously, our method usually obtains more pleasing boundaries.

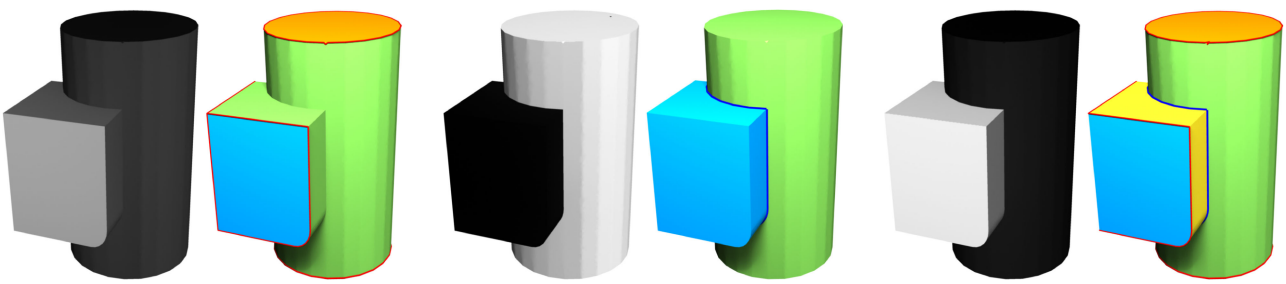
Parameter Settings. In our method, there are several parameters that can be used to adjust the segmentation results. To illustrate the impact of parameter η on segmentation results, we provide an example on the mechanical model as shown in Fig. 7. By choosing appropriate η in (1) for convex edges and concave edges, our method is able to



(a) RandCuts [40] (b) CRF-SB19 [3]

(c) CNN-SB19 [5] (d) Our method

Fig. 6. Visual comparisons of the segmentation results of our method with some state-of-the-art methods. These models are courtesy of the Princeton Segmentation Benchmark [20].



(a) (b) (c)

Fig. 7. Fiedler vectors and segmentation results of our method using various parameters η for building Laplacian matrix L: (a) $\eta = 0.1$ for convex edges and $\eta = 1.0$ for concave edges; (b) $\eta = 1.0$ for convex edges and $\eta = 0.1$ for concave edges; (c) $\eta = 1.0$ for convex edges and $\eta = 1.0$ for concave edges.

partition the given mesh along the desired boundaries. Figs. 7a and 7b present the segmentation results along convex edges and concave edges respectively, and Fig. 7c provides results along both cases. Another influence parameter λ in (6) affects the result of boundary refinement. Fig. 8 gives an example. The larger λ is, the more significant the weighted ℓ_0 term in (9) is, which leads to more tight boundary. However, it is obvious that overlarge λ will result in degenerate solutions. In practice, we typically set $\lambda \in [0.01, 5]$. Other parameters, such as k, r, may affect the final result too. Fortunately, we observe that using the default values works well in our experiments.

TABLE 2
3DNPRI Values of our Method and Some State-of-the-Art Methods Across all 5 Categories in the LIFL/LIRIS Segmentation Benchmark [36]

Method	FitPrim	ShapeDiam	BoundLearn	HG	**Ours**
Animal	0.45	0.62	0.68	0.60	0.68
Bust	0.09	0.24	0.41	0.20	0.48
Furniture	0.56	0.85	0.79	0.69	0.79
Hand	0.52	0.19	0.68	0.63	0.67
Human	0.61	0.66	0.69	0.62	0.66
Average	0.45	0.51	0.65	0.55	0.66

Note that we only compare our method with those methods, whose scores of 3DNPRI have been reported.

5.3 Noisy Data and Sharp Feature Detection

To evaluate the robustness of our method resistent to noise, we have run it on noisy mesh models by adding random Gaussian noise with zero mean and the standard deviation σ on each vertex along the direction of their normals. Fig. 9 presents the segmentation results with different levels of noise. It is not surprising that our method can still obtain satisfying segmentation results on noisy data, thanks to the

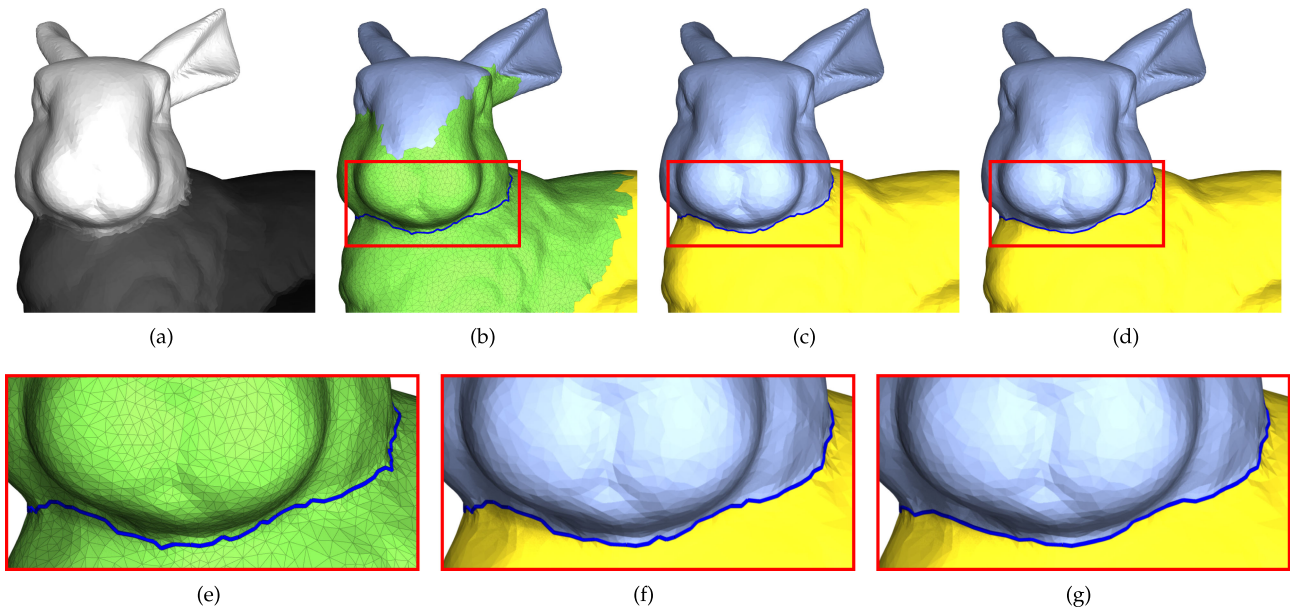


Fig. 8. Boundary refinement results of the Bunny model using various parameters λ: (a) The Fiedler vector; (b) coarse segmentation (the local region is marked as green); (c) $\lambda = 0.5$; (d) $\lambda = 4.0$. The red regions in (b), (c) and (d) are shown in (e), (f) and (g) respectively.

ℓ_0 regularization term in our spectral analysis framework. Moreover, since segmentation naturally induces boundaries at which sharp features are located, our method can be used to detect them, especially for CAD models. Fig. 10 shows an example. All sharp feature lines are perfectly detected.

5.4 Running Time

The most time-consuming task in our method is to solve the ℓ_0 gradient minimization problem (6). Since we adopt a coarse-to-fine strategy and solve the problem locally, the computation is greatly sped up. Our method partitions the whole Princeton Segmentation Benchmark in 592 seconds, which is about 1.56 seconds for each mesh on average. Clearly, the presented method is relatively fast compared with some state-of-the-art spectral methods, e.g., MS [9] and HG [6]. To further evaluate the performance, we also run our segmentation method on some complex models by the interactive mode. Fig. 11 presents the final results, and Table 3 summarizes the computing time for these models. Similar to other spectral methods, one of the bottlenecks of our method is the computation of eigenvalues and eigenvectors for large meshes.

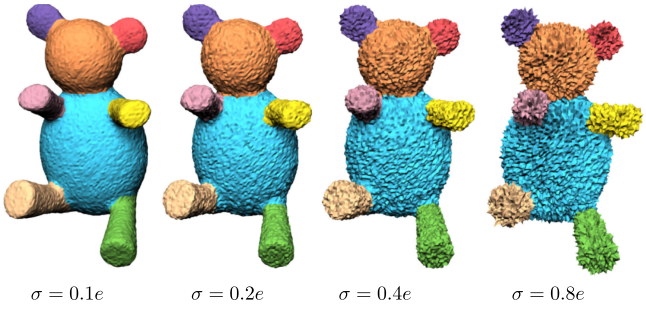


Fig. 9. The effect of noise on the segmentation results of the proposed method. The Teddy model is corrupted by four different levels of Gaussian noise ($\sigma = 0.1\overline{e}, 0.2\overline{e}, 0.4\overline{e}, 0.8\overline{e}$, where $\overline{e}$ is the average length of all edges).

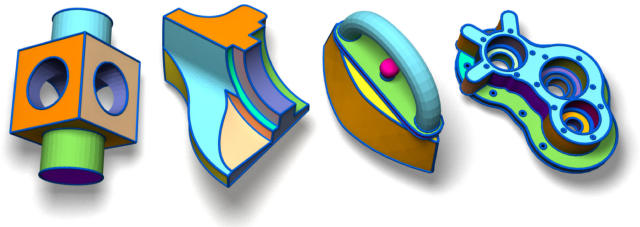


Fig. 10. Sharp feature lines (the lines are colored in blue) and segmentation results by applying the proposed method to CAD models.

6 Conclusion

In this work, we present a novel spectral mesh segmentation method inspired by sparsity pursuit. Our method treats mesh segmentation as an ℓ_0 gradient minimization problem, where the Fiedler vector is used to characterize the uniformity among the elements in the same segment and ℓ_0 norm is applied to measure the number of edges comprising segment boundaries. To solve this optimization problem effectively, we devise a fast algorithm to find a rational coarse segmentation and propose an optimization algorithm based

TABLE 3
Runtime Performance of our Method on a Variety of Models which are Shown in Fig. 11

Model	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert F\rvert$	K	T_1	T_2	T
Fertility	13,971	41,931	27,953	11	0.33	2.08	2.41
Dancer	24,998	74,994	49,996	16	0.78	4.01	4.79
Bunny	27,843	83,523	55,682	12	0.81	10.78	11.59
Memento	49,968	149,898	99,932	17	1.97	7.93	9.90
Neptune	52,511	157,545	105,030	27	2.27	7.71	9.98
Santa	75,781	227,337	151,558	18	2.59	14.98	17.57
Dancing children	724,645	2,173,976	1,449,316	22	65.22	93.35	158.57

K is the number of segments, T_1 is the amount of time for computing eigenvalues and eigenvectors, and T_2 counts the time of other procedures, e.g., coarse segmentation, boundary refinement, updating segmentation. The total running time is reported in the last column T, where only the time spent on user interactions is not included. All timings are measured in seconds.

Fig. 11. More segmentation results of our method on some complex meshes downloaded from the Internet.

on ADMM to refine the segment boundaries within their local regions. Our method is computationally efficient, insensitivity to noise and straightforward to implement. As demonstrated by experimental results on the Princeton Segmentation Benchmark, our method outperforms the state-of-the-art segmentation methods and is comparable even to those produced by human beings.

Regarding future work, our method will greatly benefit from the parallelization of our method on GPU, since boundary refinement is constrained to local regions and it can be carried out simultaneously. Furthermore, the computation of the Fiedler vector can be sped up by using GPU technique. Another interesting direction is to incorporate multi-resolution technique into our method for partitioning very large meshes. More intuitive and automatic ways to choose parameters are also worthy of further study.

ACKNOWLEDGMENTS

This work is supported by the NSF of China (No. 11571338, 61877056), the Fundamental Research Funds for the Central Universities (WK0010000051), the China Postdoctoral Science Foundation (No. 2018M632548) and the Open Project Program of the State Key Lab of CAD&CG (No. A1819). Weihua Tong, Xiankang Yang Joint first authors.

REFERENCES

[1] P. Theologou, L. Pratikakis, and T. Theoharis, "A comprehensive overview of methodologies and performance evaluation frameworks in 3D mesh segmentation," *Comput. Vis. Image Understanding*, vol. 135, pp. 49–82, 2015.

[2] A. Shamir, "A survey on mesh segmentation techniques," *Comput. Graph. Forum*, vol. 27, no. 6, pp. 1539–1556, 2008.

[3] E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3D mesh segmentation and labeling," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 102:1–102:12, 2010.

[4] Y. Wang, M. Gong, T. Wang, D. Cohen-Or, H. Zhang, and B. Chen, "Projective analysis for 3D shape segmentation," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 192:1–192:12, 2013.

[5] K. Guo, D. Zou, and X. Chen, "3D mesh labeling via deep convolutional neural networks," *ACM Trans. Graph.*, vol. 35, no. 1, pp. 3:1–3:12, 2015.

[6] P. Theologou, I. Pratikakis, and T. Theoharis, "Unsupervised spectral mesh segmentation driven by heterogeneous graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 2, pp. 397–410, Feb. 2017.

[7] H. Wang, T. Lu, O. K. C. Au, and C. L. Tai, "Spectral 3D mesh segmentation with a novel single segmentation field," *Graphical Models*, vol. 76, pp. 440–456, 2014.

[8] M. Chahhou, L. Moumoun, M. El Far, and T. Gadi, "Segmentation of 3D meshes using p-spectral clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1687–1693, Aug. 2014.

[9] J. Y. Zhang, J. M. Zheng, C. L. Wu, and J. F. Cai, "Variational mesh decomposition," *ACM Trans. Graph.*, vol. 31, no. 3, pp. 21:1–21:14, 2012.

[10] O. K. C. Au, Y. Y. Zheng, M. L. Chen, P. F. Xu, and C. L. Tai, "Mesh segmentation with concavity-aware fields," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 7, pp. 1125–1134, Jul. 2012.

[11] B. Lévy and H. R. Zhang, "Spectral mesh processing," in *Proc. ACM SIGGRAPH Courses*, 2010, pp. 8:1–8:312.

[12] H. Zhang, O. Van Kaick, and R. Dyer, "Spectral mesh processing," *Comput. Graph. Forum*, vol. 29, no. 6, pp. 1865–1894, 2010.

[13] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[14] R. Liu and H. Zhang, "Segmentation of 3D meshes through spectral clustering," in *Proc. 12th Pacific Conf. Comput. Graph. Appl.*, 2004, pp. 298–305.

[15] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via L_0 gradient minimization," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 174:1–174:12, 2011.

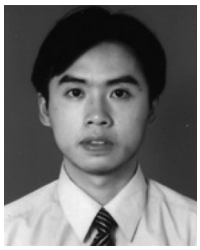
[16] L. He and S. Schaefer, "Mesh denoising via L_0 minimization," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 64:1–64:8, 2013.

[17] X. Cheng, M. Zeng, and X. G. Liu, "Feature-preserving filtering with L_0 gradient minimization," *Comput. Graph.*, vol. 38, pp. 150–157, 2014.

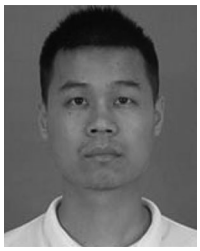
[18] R. M. H. Nguyen and M. S. Brown, "Fast and effective L_0 gradient minimization by region fusion," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 208–216.

[19] S. Ono, "L_0 gradient projection," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1554–1564, Apr. 2017.

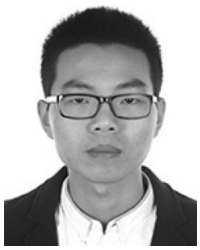
[20] X. B. Chen, A. Golovinskiy, and T. Funkhouser, "A benchmark for 3D mesh segmentation," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 73:1–73:12, 2009.
[21] S. Katz and A. Tal, "Hierarchical mesh decomposition using fuzzy clustering and cuts," in *Proc. ACM SIGGRAPH Papers*, 2003, pp. 954–961.
[22] H. Zhang and R. Liu, "Mesh segmentation via recursive and visually salient spectral cuts," in *Proc. Vis. Model., Vis.*, 2005, pp. 429–436.
[23] R. Liu and H. Zhang, "Mesh segmentation via spectral embedding and contour analysis," *Comput. Graph. Forum*, vol. 26, no. 3, pp. 385–394, 2007.
[24] Y. K. Lai, S. M. Hu, R. R. Martin, and P. L. Rosin, "Rapid and effective segmentation of 3D models using random walks," *Comput. Aided Geometric Des.*, vol. 26, no. 6, pp. 665–679, 2009.
[25] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Berlin, Germany: Springer, 2010.
[26] Y. C. Eldar and G. Kutyniok, *Compressed Sensing: Theory and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2012.
[27] L. Xu, R. Wang, J. Zhang, Z. Yang, J. Deng, F. Chen, and L. Liu, "Survey on sparsity in geometric modeling and processing," *Graphical Models*, vol. 82, pp. 160–180, 2015.
[28] U. von Luxburg, "A tutorial on spectral clustering," *Statist. and Comput.*, vol. 17, no. 4, pp. 395–416, 2007.
[29] F. R. K. Chung, *Spectral Graph Theory*. Providence, RI, USA: American Mathematical Society, 1997.
[30] M. Juvan and B. Mohar, "Optimal linear labelings and eigenvalues of graphs," *Discrete Appl. Math.*, vol. 36, no. 2, pp. 153–168, 1992.
[31] M. Isenburg and P. Lindstrom, "Streaming meshes," in *Proc. IEEE Vis.*, 2005, pp. 231–238.
[32] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, no. 5, pp. 877–905, 2008.
[33] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[34] G. Guennebaud, B. Jacob, et al., "Eigen," (2018). [Online]. Available: http://eigen.tuxfamily.org/
[35] R. Lehoucq, K. Maschhoff, D. Sorensen, and C. Yang, "ARPACK," (2018). [Online]. Available: https://github.com/opencollab/arpack-ng
[36] H. Benhabiles, J.-P. Vandeborre, G. Lavoué, and M. Daoudi, "A comparative study of existing metrics for 3D-mesh segmentation evaluation," *Visual Comput.*, vol. 26, no. 12, pp. 1451–1466, 2010.
[37] S. Shlafman, A. Tal, and S. Katz, "Metamorphosis of polyhedral surfaces using decomposition," *Comput. Graph. Forum*, vol. 21, no. 3, pp. 219–228, 2002.
[38] S. Katz, G. Leifman, and A. Tal, "Mesh segmentation using feature point and core extraction," *Visual Comput.*, vol. 21, no. 8, pp. 649–658, 2005.
[39] M. Attene, B. Falcidieno, and M. Spagnuolo, "Hierarchical mesh segmentation based on fitting primitives," *Visual Comput.*, vol. 22, no. 3, pp. 181–193, 2006.
[40] A. Golovinskiy and T. Funkhouser, "Randomized cuts for 3D mesh analysis," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 145:1–145:12, 2008.
[41] L. Shapira, A. Shamir, and D. Cohen-Or, "Consistent mesh partitioning and skeletonisation using the shape diameter function," *Visual Comput.*, vol. 24, no. 4, pp. 249–259, 2008.
[42] O. V. Kaick, N. Fish, Y. Kleiman, S. Asafi, and D. Cohen-OR, "Shape segmentation by approximate convexity analysis," *ACM Trans. Graph.*, vol. 34, no. 1, pp. 4:1–4:11, 2014.
[43] H. Benhabiles, G. Lavoué, J.-P. Vandeborre, and M. Daoudi, "Learning boundary edges for 3D-mesh segmentation," *Comput. Graph. Forum*, vol. 30, no. 8, pp. 2170–2182, 2011.



Weihua Tong received the BS and PhD degrees in computational mathematics from the University of Science and Technology of China, in 1999 and 2005. Currently, he is an associate professor with the School of Mathematical Sciences, University of Science and Technology of China. His research interests include computer graphics, computer aided geometric design and mesh processing.



Xiankang Yang is working toward the graduate degree in the School of Mathematical Sciences, University of Science and Technology of China. His research interests include computer graphics and image processing.



Maodong Pan received the PhD degree in computational mathematics from the University of Science and Technology of China, in 2017. Currently, he is a postdoctoral researcher with the School of Mathematical Sciences, University of Science and Technology of China. His research interests include computer aided geometric design and computer graphics.

Falai Chen received the BS and PhD degrees in computational mathematics from the University of Science and Technology of China, in 1987 and 1994, respectively. He has been a professor with the School of Mathematical Sciences, University of Science and Technology of China since 1998. His research interests include computer aided geometric design, geometric modeling and computer graphics. He is on the editorial boards of several international journals such as Computer Aided Geometric Design, Visual Computer and Numerical Mathematics: Theory, Methods and Applications.