Mesh Segmentation via Recursive and Visually Salient Spectral Cuts

Hao Zhang, Rong Liu

School of Computing Science, Simon Fraser University, Burnaby, Canada
Email: {haoz, lrong}@cs.sfu.ca

Abstract

We develop a new mesh segmentation algorithm via recursive spectral 2-way cut and Nyström approximation. The cut is performed on 1-D spectral embeddings, which are efficiently computed from appropriately defined distances between the set of mesh faces and only two sample faces. By using a novel sampling scheme based on shape context and a line search over the 1-D embeddings to locate the most perceptually salient cut, our algorithm achieves robust and intuitive segmentation results.

1 Introduction

Segmenting a 3D object, typically represented by a triangle mesh, into visually meaningful parts plays a key role in object recognition by humans [6]. It is also an important problem in geometry processing [17] with such applications as morphing [19], skeleton extraction [10], mesh parameterization [24], and compression [15]. Research on mesh segmentation seeks to find a computationally efficient procedure capable of producing results that are in close agreement with human shape perception, while requiring little or no user intervention.

We treat mesh segmentation as a clustering problem on mesh faces, where face distances are defined to respect the *minima rule* from cognitive studies [6]; it stipulates that cut boundaries should consist of surface points at negative minima of principal curvatures. We solve the clustering problem in the *spectral embedding space* derived from the faces distances. Spectral clustering [1, 3, 5, 13, 18, 20, 21] has received a great deal of attention recently in computer vision and machine learning. Although recent works on the topic have focused on using more eigenvectors and computing clusterings in a higher dimensional space, e.g., using k-means, we have found this to be non-robust and time-consuming for mesh segmentation. The non-robustness is mostly due to typical problems associated with the k-means approach, e.g., *chaining*[1] [4], existence of bad local minima, and the difficulty of choosing an appropriate k.

We recursively partition a mesh into two parts in 1-D embedding spaces, in the same spirit as normalized cuts [18]. An optimal cut, based on a quantification of perceptual part salience [7], can be easily found by a line search. Whereas for k-way partitions, it is unclear how part salience can be factored into k-means which would also allow an efficient search for a desirable clustering. To avoid computing all pairwise face distances, we apply Nyström method [5, 22] to approximate the eigenvectors of a matrix by subsampling only a small subset of its rows. We speed up our algorithm further by selecting the smallest possible number of samples, two.

One key observation [23] about the 1-D embeddings derived from Nyström approximation using two sample faces, say f_s and f_t, is that in general when a face f_i is closer to one of the samples, e.g., f_s, on the original mesh, it is also closer to f_s in the 1-D embedding. Thus to make line search work robustly, we would want to have the two sample faces chosen from perceptually separate parts of a shape; this is illustrated in Figure 1. We develop a novel sampling scheme based on shape context [2] that can efficiently and reliably extract such two sample faces without segmenting the mesh. The resulting mesh segmentation algorithm is efficient and produces robust and meaningful results.

1.1 Previous work

The majority of mesh segmentation works are geometry-based without the incorporation of prior knowledge, e.g., see recent survey by Shamir [17]. Li et al. [12] define a meaningful component of

[1] Chaining is a well known phenomenon arising from single-linkage based clustering [4], where elements are clustered based on a single link. In the context of mesh segmentation, chaining of faces from different parts of a shape typically occurs over a featureless region, where the minima rule cannot predict a cut boundary.

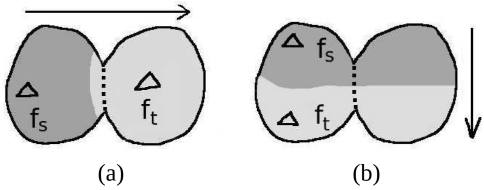


Figure 1: We show how the placement of sample faces may influence the result of 2-way cut via line search. The two sample faces are f_s and f_t and their Voronoi regions, in terms of face distances (Section 3), are shown in different shades of grey. A rough direction for the line search in the spectral domain is shown by the arrow. As we can see in (a), line search may be able to locate the best cut, marked by a dashed line. But for (b), where f_s and f_t come from the same part, line search will fail.

a mesh to be a volume swept by a shape-varying closed 2D region along a sweep path, between two suitably defined critical points. It works fairly well in practice but can produce counterintuitive results. Also, it does not conform to the minima rule and smoothing of the geometry function may miss meaningful segmentations [10].

Surface-based mesh segmentation algorithms can be divided into boundary-based [11] and region-based [14, 16, 13, 19, 10] approaches. Lee et al. [11] propose a recursive mesh scissoring operator, honoring the minima rule [6]. At each step, one of the extracted feature curves is selected for completion into a cut contour, based on its length and centricity. Although some contours may be rejected by a part salience [7] test, there is no global search for the best cut. The algorithm appears to work the best in a semi-automatic setting. With an effective contour completion procedure, snake movements, and a small amount of user intervention, excellent segmentation results have been reported.

Watershed was first proposed for mesh segmentation by Mangan and Whitaker [14]. Page [16] later uses fast marching watershed and hill-climbing that respects the minima rule. Although watershed is fast and there is no need to specify the number of segments, it is prone to over-segmentation. This may be corrected by region merging using a part salience measure [16], but it still does not resolve the “flooding” problem across featureless regions that should have been identified as cut boundaries.

Other region-based segmentation algorithms utilize k-means clustering. Shlafman et al. [19] apply original k-means in the spatial domain where both geodesic and angle distances between mesh faces are considered. Liu and Zhang [13] apply k-means in the spectral domain, where better segmentation results are predicted by the Polarization Theorem [3]. Katz and Tal [10] use a probabilistic k-means approach called fuzzy clustering. To optimize cut boundaries, a graph min-cut is computed over a fuzzy region of faces whose membership to the two patches is inconclusive. An important step is to iteratively find patch representatives that act as statistical cluster centers. Preferably, these representatives would reside on perceptually separate parts of a shape. Their locations determine the size and location of the fuzzy region, which in turn influences the segmentation result. Typical problems associated with k-means exist in all the above three algorithms. Meanwhile they all require distances between all face pairs, which are expensive to compute and store.

1.2 Our contributions

Our segmentation algorithm can be seen as an extension to the normalized cuts approach [18], where we utilize a novel sampling scheme to make effective use of Nyström approximation at a very low sample size, two in fact. Our algorithm also adopts a different optimization criterion, based on part salience [7], that is specific for mesh segmentation. The key features of our algorithm are:

- **Efficiency:** Our algorithm can handle highly dense meshes directly. It runs in $O(pn \log n)$ time, compared to $\Theta(n^2 \log n)$ by [10, 13], where n is the number of mesh faces and p is the number of recursions. Note that $p \ll n$, as it is no greater than the number of parts.
- **Visually meaningful segmentation:** Our algorithm quantifies the minima rule [6] and the part salience measures [7] resulting from cognitive studies. This appeals to human perception and ensures the segmentation quality.
- **Robustness:** This is achieved through a combination of recursive 2-way cut, salience-based global line search, and our sampling scheme based on shape context, which should also be useful for other shape analysis tasks.

The effectiveness of our approach can be demonstrated using several formal arguments and numerous examples. We believe our algorithm offers the best combination of speed, quality, and robustness among mesh segmentation algorithms to date.

1.3 Paper organization

The rest of the paper is organized as follows. Section 2 gives an algorithm overview and Section 3 defines the distance measure. The sampling scheme based on shape context is developed in Section 4. In Section 5, we describe normalized cuts, Nyström method, and line search. Section 6 quantifies part salience. Experimental results are given in Section 7 to demonstrate the effectiveness of our approach. Finally, we conclude in Section 8 and suggest possible future work.

2 Algorithm overview

Input and preprocessing: Input to our algorithm is a manifold triangle mesh with arbitrary topology and possibly boundary. Distances between adjacent mesh faces are calculated and serve as edge weights in the dual graph of the mesh. When the input mesh is noisy, it is first smoothed slightly using a few steps of Laplacian smoothing so that the face distances can capture more truthfully the surface variations, e.g., bending. Alternatively, we could use best fit polynomials [8] to measure bending without smoothing, but it is more time-consuming.

Recursive 2-way cut: We place all candidate parts, starting with the original mesh, in a priority queue ordered by their surface areas. At each step, the largest part is selected and the most salient cut is found using BESTCUT, given in Figure 2, where salience is measured relative to the candidate part. If the resulting salience is below a user-set threshold, then the cut is rejected and the part will no longer be considered. Otherwise, the resulting two subparts are inserted back into the priority queue.

Post-smoothing: Most of the segmentation results we report are obtained without explicit boundary smoothing, since the cuts are obtained through fine-grained line search using part salience and it prefers shorter cuts. We consider this an advantage of our algorithm. In the few cases where small jaggies persist, it is sufficient to use a simple smoothing procedure based on morphological processing, with a structuring element of size two, on mesh connectivity to remove any local artifacts in negligible time.

Termination: The user can specify a maximum number of segments to compute. The recursive algorithm stops when this number is reached or when no candidate part has a cut salience, computed via BESTCUT, above the given threshold.

Spectral clustering takes as input an *affinity matrix* A and typically acts on the *normalized* affinity matrix $N = D^{-1/2}AD^{-1/2}$ instead [21]. Here A_{ij} models the probability of data points i, j belonging to the same cluster and D is the diagonal matrix of A's row sums. In the procedure below, we only need partial blocks of A and N.

```
BESTCUT ($M$: a sub-mesh with $n$ faces)
  1. Select two sample faces $f_s$ and $f_t$ from $M$. — $\Theta(n \log n)$
  2. Compute $\hat{W} \in \mathbf{R}^{2 \times n}$, a partial distance matrix where $\hat{W}_{1j}$ (resp. $\hat{W}_{2j}$) encodes the distance between face $f_s$ (resp. $f_t$) and face $f_j$, $j = 1, \ldots, n$. — $\Theta(n \log n)$
  3. Convert $\hat{W}$ into an partial affinity matrix $\hat{A}$ using an exponential kernel. — $\Theta(n)$
  4. Use Nyström method and $\hat{A}$ to obtain $\vec{e}^{(1)}$ and $\vec{e}^{(2)}$, the approximate first two eigenvectors of the full matrix $N$. — $\Theta(n)$
  5. Construct the linear arrangement $\vec{z}$, where $z_i = e_i^{(2)}/e_i^{(1)}$, $i = 1, \ldots, n$. — $\Theta(n \log n)$
  6. Line search along $\vec{z}$ to locate the most salient cut using our part salience measure, which can be updated in constant time along the search. — $\Theta(n)$
  7. Insert the resulting two sub-meshes into the priority queue if the resulting salience is above a threshold. — $\Theta(\log p)$, where $p$ is the number of candidate parts so far.
```

Figure 2: BESTCUT(): Salience-based spectral 2-way cut. Also shown are the asymptotic time complexities of each step, where n is the face count.

3 Face distance computations

As in Shlafman et al. [19] and Katz and Tal [10], we consider both angle and geodesic distances between mesh faces, but with angle distances *histogram-equalized*. Specifically, the distance between a pair of *adjacent* faces f_i and f_j is defined as

$$d(f_i, f_j) = (1-\delta) \cdot \mathcal{H}(f_i, f_j) + \delta \cdot \mathcal{G}(f_i, f_j), \quad (1)$$

where $\mathcal{G}(f_i, f_j)$ is the geodesic distance between the centroids of faces f_i and f_j, normalized by the average geodesic distance, and $\mathcal{H}(f_i, f_j)$ is the histogram equalized angle distance.

Let θ be the angle formed by the normals of f_i and f_j, then one may define the angle distance between f_i and f_j as $h(f_i, f_j) = \eta(1 - \cos\theta)$, where η is a free parameter and h is subsequently normalized by its average over the whole mesh. However, for a smooth mesh, the values of h would typically be highly concentrated near zero. To increase the geometry contrast, we apply histogram equalization [9] to h to obtain $\mathcal{H}$, i.e., we sort the h's and map them to a set of equally spaced values in [0, 1]. To emphasize the minima rule, we set $\delta = 0.01$, $\eta = 1$ for concave angle θ, and $\eta = 0.1$ for convex θ. Thus faces separated by concave regions over the mesh surface are more likely to be clustered into different parts than those just geodesically far away.

Distances between non-adjacent faces are computed as shortest graph distances using Dijkstra's algorithm in $O(n \log n)$, where the graph is the dual of the mesh graph. We distinguish three cases for the edge weights: $\delta = 1$, $\delta = 0$, and $0 < \delta < 1$, referring to (1). In the rest of the paper, we refer to the resulting graph distances (respectively, the corresponding shortest paths) as geodesic, angle, or *combined* distances (paths). They will be used for sampling and face clustering, respectively. Note that we have chosen not to compute true geodesics since the graph distances given above are much simpler to compute and they provide sufficiently good approximations for our work. Finally, note that computing all-pair shortest distances is at least $\Theta(n^2 \log n)$, which would be too expensive for large meshes. We deal with this problem by carefully selecting two sample faces and using only distances originating from these two faces to construct the spectral embedding, relying on Nyström method. We describe our sampling scheme next.

4 Sampling based on shape context

The ability to select sample faces from perceptually separate parts of a shape without explicit segmentation is desirable for many tasks, e.g., in finding patch representatives for fuzzy clustering [10]. In our setting, as explained in Section 1 and Figure 1, we wish to locate samples from perceptually separate parts in order for line search to work robustly.

The simple strategy of choosing two faces furthest apart, e.g., geodesically, would not work in general, since these two faces may not lie on different parts of a sufficiently elongated shape. Besides, one cannot find such two faces precisely in sub-quadratic time. Even if we replace geodesic distance by the combined distance (geodesic plus angle), the chaining phenomenon [4] reveals that it is unreliable to judge whether two faces belong to different parts simply by measuring a single shortest path between them. This is illustrated in Figure 3, where faces from the fin are "chained" together with faces from the body through the marked flat region; f_p and f_q are the two faces furthest apart but belong to the same body part. Next, we propose a more robust sampling scheme based on shape context to alleviate this problem; the two triangles f_s and f_t are properly selected as samples by this scheme.

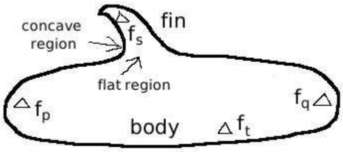


Figure 3: Faces f_p and f_q are samples furthest apart in terms of combined distance d, given in (1). f_s and f_t are chosen by our sampling scheme.

4.1 Shape context of a mesh face

One of the basic premises of shape context [2] is that similarity between data points should not be measured by distances between the absolute point coordinates in the original space. Rather, we should parameterize each point by how it is related to the rest of the points and measure distances in a new feature space. The relationship between a point and the rest of the points defines a *context* for the point.

The notion of shape context forms the basis of our sampling scheme. To save storage and processing times, we define the context of a face g with respect to k (k is small; we choose $k = 10$ in all our experiments) *reference faces*. Specifically, the *context vector* $\vec{c}(g) = [c_1 \dots c_k]$ of g is given by

$$c_j = \sum_{i=1}^{m-1} \mathcal{H}(f_i, f_{i+1}), j = 1, \dots, k$$

where the angle distance $\mathcal{H}$ is as defined in (1), $f_1 = g$, $f_m = h$ is one of the k reference faces, and $f_1, f_2, \dots, f_m$ form a *shortest geodesic path* between faces g and h. In other words, the context of face g is defined by angle distances accumulated along geodesic paths from g to k reference faces. The reference faces can be chosen randomly. But for more robust results, we uniformly sample in terms of geodesic distances, across the mesh. Once chosen, the k reference faces are used to compute context vectors for all the faces.

4.2 Our sampling scheme

To select k approximately uniformly distributed reference faces, we start by choosing one face at random and iteratively select a total of k faces that are mutually furthest apart with respect to geodesic distances. This would take time $O(kn \log n)$ using Dijkstra's algorithm. During this process, the context vector for each face with respect to the reference faces can be computed simultaneously; we simply need to accumulate the angle distances as the geodesic paths are being formed.

Let $\vec{c}^{(1)}, \vec{c}^{(2)}, \ldots, \vec{c}^{(n)}$ be the context vectors of faces on a part to segment, then the first sample f_s and second sample f_t are found in linear time by

$$f_s = \operatorname*{argmax}_i ||\vec{c}^{(i)}||, f_t = \operatorname*{argmax}_i ||\vec{c}^{(i)} - \vec{c}^{(s)}||.$$

Our sampling strategy implies that the first sample f_s is most "isolated", through angle accumulation, from the reference faces. Such a sample is most likely on a peripheral part, e.g., the sample face on the fin part in Figure 3. The second sample f_t is least similar to the first sample in terms of their contexts. The success of our sampling scheme relies on the premise that if the contexts of two faces differ significantly, then they do not belong to the same part of an object.

5 Spectral 2-way cut and line search

We now describe our spectral 2-way cut and line search procedure. This is inspired by previous work on normalized cut [18] and Nyström method [5, 22]. We first give a brief introduction to these topics.

5.1 Normalized cuts

Normalized cut was first introduced for image segmentation by Shi and Malik [18], where one seeks a graph cut minimizing the normalized cut criterion; the tendency is to locate a small edge cut that separates the graph into subgraphs of similar "strength of connectivity" to the whole graph. The algorithm recursively cuts a graph into two parts and at each step uses the second smallest eigenvector of the *normalized Laplacian matrix* $L = I - D^{-1}A$ to derive a 1-D embedding of the image pixels, where A and D are as defined in Figure 2 and one should view A as the weighted graph adjacency matrix. The original normalized cut relies on thresholding and line search along the embedding has also been suggested in [20] to locate the best cut.

It is known that [21] the second smallest eigenvector of the Laplacian matrix L is identical to the *component-wise ratio* between the second and first largest eigenvectors of the normalized affinity matrix $N = D^{-1/2}AD^{-1/2}$. We use component-wise ratio in this paper as it facilitates the use of Nyström method, as we explain in Section 5.3.

5.2 Nyström approximation

To avoid computing the full affinity matrix A, which is at least an $O(n^2)$ step, Fowlkes et al. [5] derive a matrix version of Nyström method that only requires values in a small sub-block of A. It approximates the k leading eigenvectors of A by using k randomly chosen data samples and extrapolating results from the eigenvectors of a $k \times k$ matrix. Specifically, let

$$A = \begin{bmatrix} X & Y \\ Y^T & Z \end{bmatrix} \tag{2}$$

with $X \in \mathbf{R}^{k \times k}$ and $Y \in \mathbf{R}^{k \times (n-k)}$. Let $X = U\Lambda U^T$ be an eigenvalue decomposition of X. Then the approximate eigenvectors, $\tilde{U}$ of A are given by

$$\tilde{U} = \begin{bmatrix} U \\ Y^T U \Lambda^{-1} \end{bmatrix}.$$

Thus only $\Theta(kn)$ pairs of affinities are needed and the complexity of computing k approximated eigenvectors of A is reduced to $O(k^3 + kn)$. In practice, $k \ll n$, and good image segmentation results using spectral embeddings have been reported [5].

5.3 Spectral embedding and line search

Given distance $d(f_i, f_j)$ between two faces f_i and f_j, defined in (1), we use an *exponential kernel* to define the affinity matrix A, $A_{ij} = e^{-d(f_i,f_j)/\sigma^2}$. The kernel width σ does not appear to have a great influence on the partition results, as long as it is not too small. Otherwise, small clusters can be formed that can clutter the spectral embedding. We simply choose σ to be the average of all distances available.

With only two samples f_s and f_t, Nyström method reduces to solving a 2×2 eigenvalue problem followed by $\Theta(n)$-time eigenvector extrapolation. Specifically, the sampled block of A is

$$\hat{A} = \begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \end{bmatrix} = \begin{bmatrix} 1 & u & \dots & x_j & \dots \\ u & 1 & \dots & y_j & \dots \end{bmatrix},$$

where $0 \leq u, x_i, y_i \leq 1$ are affinity values defined by the exponential kernel, u being the affinity between f_s and f_t. Since we use normalized affinity matrix $N = D^{-1/2}AD^{-1/2}$ rather than A, the corresponding block $\hat{N}$ has to be approximated without knowing all the rows of A. This can be done via an approximation to block Z of A [5], referring to (2). But with the use of component-wise ratios between the second and first eigenvectors of N as our 1-D embedding, we can avoid any errors resulting from this approximation since it can be shown that the unknown row sums of A cancel out [23].

Line search starts at one end of the 1-D embedding and visits one face at a time sequentially. During the search, we maintain a dynamic *connected meta patch* Q formed by faces visited so far. If a face a is encountered but it is disjoint from Q, then we give a special label to a but do not update Q. Later on in the line search, some faces with these special labels may be joined to Q via a newly encountered face b, at which time their labels are removed and Q is updated. The meta patch can be updated in $O(1)$ time after each face is added.

Our algorithm maintains the set of cut boundary edges and vertices of the meta patch Q, where any exterior edge of the original patch being segmented is excluded. Using a simple valence counting scheme, we can perform these updates in $O(1)$ time as well. Information about Q is used to compute a part salience measure. Our line search locates the best cut based on this measure and divides the mesh into two parts, the meta patch Q and the rest. In rare cases, Q might become a closed region with holes. This can be detected by our algorithm and the corresponding cut will be disregarded.

6 Part salience

We judge the "goodness" of a 2-way section by a visual salience measure of the resulting part that has a smaller size; this is often a peripheral part that we wish to cut away from the core body of a shape. Hoffman and Singh [7] have conducted a variety of cognitive studies about the salience of a visual part. They conclude that part salience should depend on (at least) three factors: its *size* ($\mathcal{V}_s$) relative to the whole object, the *strength* ($\mathcal{V}_c$) of its cut boundary, and its *protrusiveness* ($\mathcal{V}_p$), estimated by the ratio of the surface area of the part to its base area.

For a given 2-way section of a sub-mesh M, denote by Q the part with a smaller surface area and ∂Q its boundary. We define the visual salience of Q as a convex combination, similar to [16],

$$\mathcal{V}(Q) = \alpha\mathcal{V}_s(Q) + \beta\mathcal{V}_c(Q) + \gamma\mathcal{V}_p(Q), \quad (3)$$

where $\mathcal{V}_s(Q) = Area(Q)/Area(M)$, a ratio of surface areas. $\mathcal{V}_c(Q)$ measures the cut strength, accumulated over the m edges of ∂Q,

$$\mathcal{V}_c(Q) = \frac{1}{m}\sum_{e \in \partial Q} \frac{\tilde{\mathcal{H}}(e)}{\mathcal{H}_{max}(M)},$$

where $\mathcal{H}_{max}(M) = max_{e \in M}\mathcal{H}(e)$, $\tilde{\mathcal{H}}(e) = 0$ if the dihedral angle at edge e is convex, and otherwise $\tilde{\mathcal{H}}(e) = \mathcal{H}(e)$, the histogram equalized angle distance between two faces incident at edge e. Note that $\tilde{\mathcal{H}}$ is only defined for edges interior to the current patch being segmented. Finally, the protrusiveness $\mathcal{V}_p(Q) = 1 - 4\sqrt{\lambda_1\lambda_2}/Area(Q)$, where λ_1 and λ_2 are the leading eigenvalues of the covariance matrix for the mesh vertices along ∂Q. Note that both $\mathcal{V}_s$ and $\mathcal{V}_c$ can clearly be updated in $O(1)$. Constant time update for $\mathcal{V}_p$ is also possible since both the mean and the covariance matrix of a random sequence can be updated in $O(1)$ time after insertion or deletion of a variable (data point).

Part salience is used in several components of our algorithm. When selecting a candidate part to segment, we only use size salience, relative to the original mesh. When finding the best cut, appropriate values of α, β, and γ need to be chosen. In our current work, we do not develop an automatic mechanism for their selection. We have found setting $\alpha = 0.1$, $\beta = 0.6$, and $\gamma = 0.3$ to work generally well. To determine whether a part should be segmented further, the part salience returned is tested against a user-set threshold. The choice of the threshold and weights α, β, and γ is model-dependent and would require further study.

7 Experimental results

In this section, we first evaluate several components of our segmentation algorithm using isolated tests. We then show our segmentation results.

7.1 2-way cut and line search vs. k-means

Most k-way partitioning schemes use a variant of k-means clustering [1, 10, 13, 19, 21]. The common pitfalls of k-means, e.g., bad local minima, linkage, and the difficulty of choosing k, have also been well-documented [4]. Our algorithm relieves the burden of having to choose a proper k by relying on recursive 2-way cut. The robust sampling scheme and line search illuminated by part salience neutralize the chaining problem. Here we compare our algorithm to spectral clustering using k-means [13], where full affinity matrix is used and the dimensionality of the embedding space is the same as the number of desirable segments set by user. The results are shown in Figure 4, where k-way, (a), which partitions a mesh into a desirable number of parts all at once, and recursive 2-means, (b)-(c), are compared with recursive 2-way cut via line search, (d). The advantage of our approach is quite evident.

7.2 Two samples vs. more samples

We have also experimented with using more samples in Nyström approximation for computing the 1-D embedding. Although less L_2 error is introduced with the use of more samples, where we compare the approximate eigenvectors with the eigenvectors of the full normalized affinity matrix N, the resulting segmentation is no better and can sometimes be even worse. Note that we have conducted this test using several sampling schemes, including random and uniform sampling based on geodesic, angle, combined, and shape context distances; the outcomes are consistent. Apparently, the clustering structure in the 1-D embedding resulting from using two sample faces only is more favorable, but this issue requires further study.

7.3 Mesh segmentation results

Our segmentation algorithm has been tested on various mesh models having varying size (see Table 1), genus, boundary type, and geometric complexity in terms of part count, size and shape. In Table 1, we report timing statistics recorded on a Xeon 2.2 GHz machine with 1GB RAM. Due to sub-sampling, our algorithm is much more efficient than those requiring all-pair face distances [10, 13].

Figure 5 displays the segmentation results using different colors for different segments. Note that our algorithm only requires a mesh to be a 2-manifold; it applies to open meshes, (a) and (i), and meshes with genus greater than zero, (g). It can be seen that parts of various sizes and shapes are obtained as long as they pass our part salient test and the results generally appeal to our intuition. For most cut boundaries, post-smoothing are not necessary at all and only few of them are smoothed using morphological processing to remove local artifacts.

The quality of our segmentation results attests to the robustness of our sampling scheme and the effectiveness of salience-based line search. Chaining, for example, which would have been a problem for single-linkage based clustering on several models, e.g., the bird tail in (c), the "snake" in (e), and the fin on the back of the dolphin in (b), has been gracefully handled by our algorithm.

Finally, note that the search space we use at each recursion is restricted by a linear ordering, thus the cut returned is not guaranteed to be the most salient among all possible 2-way sections (there are exponentially many of them). However, our experimental results demonstrate remarkable robustness of the line search approach for mesh segmentation.

8 Conclusion and future work

We present a mesh segmentation algorithm based on recursive spectral 2-way section and Nyström approximation. A novel sampling scheme inspired by shape context is designed to place two sample faces on perceptually different parts of a shape. Our study of the effect of Nyström approximation on spectral embeddings using only two sample faces suggests that the negative impact of distance distortion in low dimensional embeddings can be countered by an appropriate sampling scheme. This allows the combination of Nyström method and a line search based on part salience to produce high-quality mesh segmentations efficiently and robustly.

A number of ways to automate the segmentation process have been experimented with. In our current implementation, recursion order is determined by part size, which is one of the three factors of part salience [7]. Our stopping criterion is still rather primitive however. Indeed, the question of whether a segmentation is sufficiently salient is a difficult one. We believe this is model-dependent and plan to investigate this issue further. At the same time, a study of the relative importance between the three part salience factors would also be interesting.

Table 1: Execution times (in seconds) for our mesh segmentation algorithm.

Model (# faces)	**Initialization**	**Sampling**	**Embedding**	**Line Search**	**Total**	**# parts**
Heart (1.6K)	0.04	0.03	0.01	0.09	0.17	5
Dolphin (2K)	0.03	0.14	0.03	0.25	0.45	8
Bird (3K)	0.07	0.15	0.02	0.24	0.48	5
Dino-pet (4K)	0.08	0.45	0.15	0.79	1.47	29
"Snake" (12K)	0.24	0.40	0.12	0.61	1.37	4
Bowl (13K)	0.25	0.30	0.07	0.32	0.94	3
Machine part (20K)	0.46	1.82	0.47	2.1	4.85	6
Horse (40K)	0.54	5.3	1.79	5.13	12.76	19
Bunny (70K)	1.02	7.2	1.86	7.29	17.37	14
Isis (200K)	2.92	19.7	4.54	13.48	40.64	5

There is still room to improve the efficiency of our algorithm, e.g., through reuse of the reference faces. We would also like to extend our current sampling scheme to include more parts. From a theoretical perspective, we would like to study the polarization phenomenon on the samples used by Nyström approximation further, especially in a higher dimensional embedding space. Finally, we believe that to obtain truly intuitive shape segmentation, the incorporation of prior human knowledge is necessary. Thus we plan to look into that issue as well.

References

[1] C. J. Alpert and S. Z. Yao, "Spectral Partitioning: The More Eigenvectors, The Better", *Discrete Applied Math*, No. 90, pp. 3-26, 1995.

[2] S. Belongie, J. Malik and J. Puzicha, "Shape Context: A New Descriptor for Shape Matching and Object Recognition", *NIPS*, 2000.

[3] M. Brand and K. Huang, "A Unifying Theorem for Spectral Embedding and Clustering", *Proc. 9th Int. Conf. on Artificial Intelligence and Statistics*, 2003.

[4] B. S. Everitt, S. Landau, and M. Leese, *Cluster Analysis*, Oxford University Press, 2001.

[5] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral Grouping Using the Nyström Method", *IEEE PAMI*, **26**(2), pp. 214–225, 2004.

[6] D. D. Hoffman and W. A. Richards, "Parts of Recognition", *Cognition*, **18**, pp. 65–96, 1984.

[7] D. D. Hoffman and M. Singh, "Salience of visual parts", *Cognition*, **63**, pp. 29-78, 1997.

[8] A. Hubeli, and M. Gross, "Multiresolution Feature Extraction for Unstructured Meshes," *IEEE Visualization 2001*, pp. 287–294, 2001.

[9] A. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[10] S. Katz and A. Tal, "Hierarchical Mesh Decomposition Using Fuzzy Clustering and Cuts", *ACM Transactions on Graphics*, **22**(3), pp. 954–961, 2003.

[11] Y. Lee, S. Lee, A. Shamir, D. Cohen-Or, and H.-P. Seidel, "Mesh Scissoring with Minima Rule and Part Salience", *CAGD*, to appear.

[12] X. Li, T. Toon, T. Tan, and Z. Huang, "Decomposing Polygon Meshes for Interactive Applications", *Proc. Symposium on Interactive 3D Graphics*, pp. 35–42, 2001.

[13] R. Liu and H. Zhang, "Segmentation of 3D Meshes through Spectral Clustering", *Proc. Pacific Graphics 2004*, pp. 298–305, 2004.

[14] A. Mangan and R. Whitaker, "Partitioning 3D Surface Meshes using Watershed Segmentation", *IEEE TVCG*, **5**(4), pp. 309–321, 1999.

[15] D. Cohen-Or, R. Cohen and R. Irony, "Multiway Geometry Coding", *preprint*, 2002.

[16] D. L. Page, *Part Decomposition of 3D Surfaces*, Ph. D. Dissertation, May, 2003.

[17] A. Shamir, "A Formalization of Boundary Mesh Segmentation", *Proc. Second International Symposium on 3D Data Processing, Visualization, and Transmission*, 2004.

[18] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation", *ICCV 1997*, pp. 731–737.

[19] S. Shlafman, A. Tal, and S. Katz, "Metamorphosis of Polyhedral Surfaces using Decomposition", *Eurographics*, pp. 219–228, 2002.

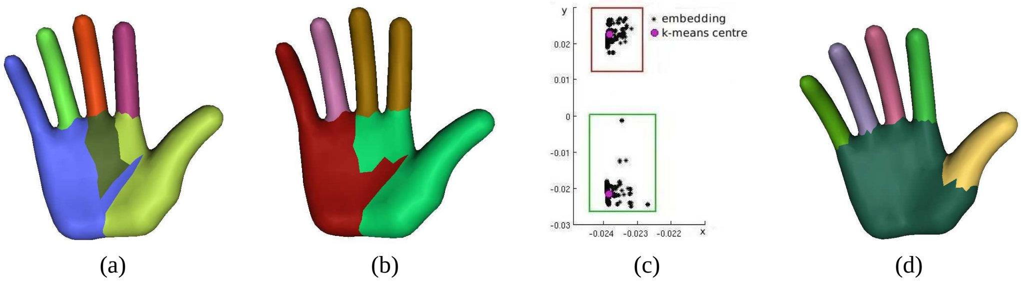
[20] D. A. Spielman and S-H. Teng, "Spectral Partitioning Works: Planar Graphs and Finite Element Meshes", *IEEE Symposium on Foundation of Computer Science*, pp. 96-105, 1996.

[21] Y. Weiss, "Segmentation Using Eigenvectors: A Unifying View", *ICCV 1999*, pp. 975-982.

[22] C. K. I. Williams and M. Seeger, "Using the Nyström Method to Speed Up Kernel Machines", *NIPS 13*, pp. 682–688, 2001.

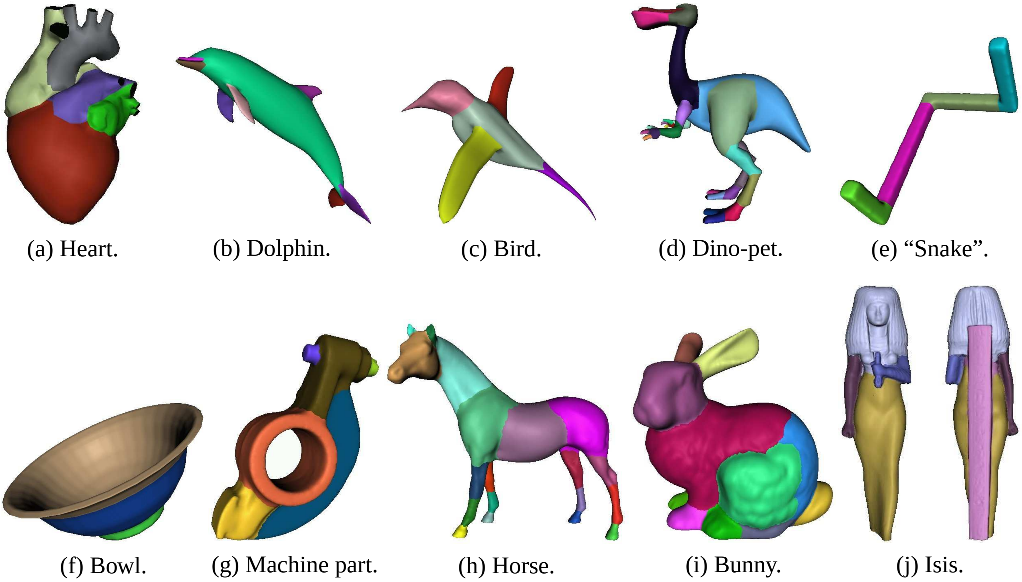
[23] H. Zhang and R. Liu, "An Analysis of Nyström Method for Spectral Clustering", *preprint*, 2005.

[24] K. Zhou, J. Snyder, B. Guo, and H-Y. Shum, "Iso-Charts: Stretch-Driven Mesh Parameterization using Spectral Analysis", *Eurographics Symposium on Geometry Processing*, 2004.



(a) (b) (c) (d)

Figure 4: Comparison between k-means spectral clustering [13], (a)-(c), and our algorithm, (d), on a hand model with 4000 faces. (a) Result of spectral clustering using k-means, $k = 6$. (b) Result of recursive k-means, $k = 2$. The first recursion groups the forefinger and the middle finger together and the third recursion gives the red and green parts. (c) Embedding of the faces, at the third recursion, that are separated into the red and green parts in (b), by 2-means; x and y axes represent spectral embedding coordinates given by the first and second eigenvectors. We see that although k-means ($k = 2$) finds the global minima in embedding space, the segmentation is still counterintuitive. This is due to the concavities present on the palm and at the back, which do not give rise to salient parts, but provide a separation between the red and green parts. (d) Result of our algorithm, which solves the problem shown in (c). The jagged boundaries are tessellation artifacts, as our segmentation boundary does not cut across faces.



(a) Heart. (b) Dolphin. (c) Bird. (d) Dino-pet. (e) “Snake”.

(f) Bowl. (g) Machine part. (h) Horse. (i) Bunny. (j) Isis.

Figure 5: Some of our segmentation results with running times shown in Table 1.