Fiedler Trees for Multiscale Surface Analysis

Matt Berger[1,2], Luis Gustavo Nonato[1,3], Valerio Pascucci[1], Cláudio T. Silva[1]

[1]Scientific Computing and Imaging Institute, University of Utah, USA

[2]Air Force Research Laboratory [3]Departamento de Matemática Aplicada e Estatística, Universidade de São Paulo, Brazil

***Abstract*—In this work we introduce a new hierarchical surface decomposition method for multiscale analysis of surface meshes. In contrast to other multiresolution methods, our approach relies on spectral properties of the surface to build a binary hierarchical decomposition. Namely, we utilize the first nontrivial eigenfunction of the Laplace-Beltrami operator to recursively decompose the surface. For this reason we coin our surface decomposition the *Fiedler tree*. Using the Fiedler tree ensures a number of attractive properties, including: mesh-independent decomposition, well-formed and nearly equi-areal surface patches, and noise robustness. We show how the evenly distributed patches can be exploited for generating multiresolution high quality uniform meshes. Additionally, our decomposition permits a natural means for carrying out wavelet methods, resulting in an intuitive method for producing feature-sensitive meshes at multiple scales.**

***Keywords*—multiscale representation; multiresolution shape analysis; surface partition**

1. Introduction

Multiscale analysis has emerged as one of the most effective mechanisms for processing large and complex surface meshes. Broadly speaking, multiscale analysis may be broken down into regular and irregular analysis. Regular schemes assume uniform subdivision to construct the mesh connectivity and consequently the hierarchy, while irregular schemes are more general, being able to handle arbitrary connectivity during the hierarchy construction.

Although mathematically sound and flexible enough to be employed on models with arbitrary topology, construction of a uniform mesh from an existing irregular surface mesh is highly nontrivial, as it requires a mapping as-isometric-as-possible between the original and the regular surface, in order to succesfully leverage the uniform multiscale techniques [24], [19].

In contrast to uniform subdivision methods, irregular analysis typically relies on mesh simplification to build meshes of arbitrary connectivity. There are effectively two main classes of mesh simplification: bottom-up simplification and top-down simplification.

Bottom-up simplification directly works on the surface mesh, and is typified by local operations such as *vertex removal* and *edge collapse* [13], [9], [11]. Such techniques naturally allow for features to be preserved, and are particularly efficient. However, these methods typically produce poor quality triangles in their pursuit of preserving details, have a high dependence on the mesh tesselation, and are sensitive to noise.

Top-down simplification methods operate in the bounding volume of the surface, hierarchically refining the space from coarse to fine, stopping the refinement process when an error criterion is achieved for each node of the tree [32], [3]. A triangulation is produced by contracting the vertices in each node of the tree into one representative vertex, and retaining only nonzero-area triangles. These methods are fairly robust to noise and rather efficient, but suffer if the alignment of the spatial structure differs from that of the surface. This spatial dependence can be problematic: for instance, simplifications of varying triangle quality will result from a surface mesh and a rigid transformation of that surface.

Remeshing schemes [1] precisely address the issue of triangle quality, in producing a target surface with a given number of vertices which well approximates the original, while satisfying some measure of mesh quality. Unfortunately, the vast majority of remeshing schemes involve some notion of energy minimization, so for the purposes of multiresolution one would need to run the particular remeshing algorithm from scratch each time, for each target number of vertices. This also makes the notion of scale for multiresolution rather unclear.

In the spectrum between simplification and remeshing our approach lies somewhere in between, in that our goal is to generate *quality irregular multiresolution*. This is particularly important for applications such as multigrid for solving PDEs, where the quality of each mesh in the hierarchy should be acceptable.

We propose a top-down, binary, hierarchical surface decomposition to generate well-formed surface patches at every scale. Namely, we utilize the first nontrivial eigenfunction of the Laplace-Beltrami operator to drive the decomposition. This has a natural analogue in the area of graph theory, a process known as *spectral bisection* [2], where a combinatorial or weighted Laplacian is used. The first nontrivial eigenvector used to drive the decomposition is known as the *Fiedler vector*. We adapt this notation to coin our structure the *Fiedler tree*. By utilizing the Laplace-Beltrami operator instead of the combinatorial Laplacian, we obtain many nice properties: surface patches of uniform area, well-

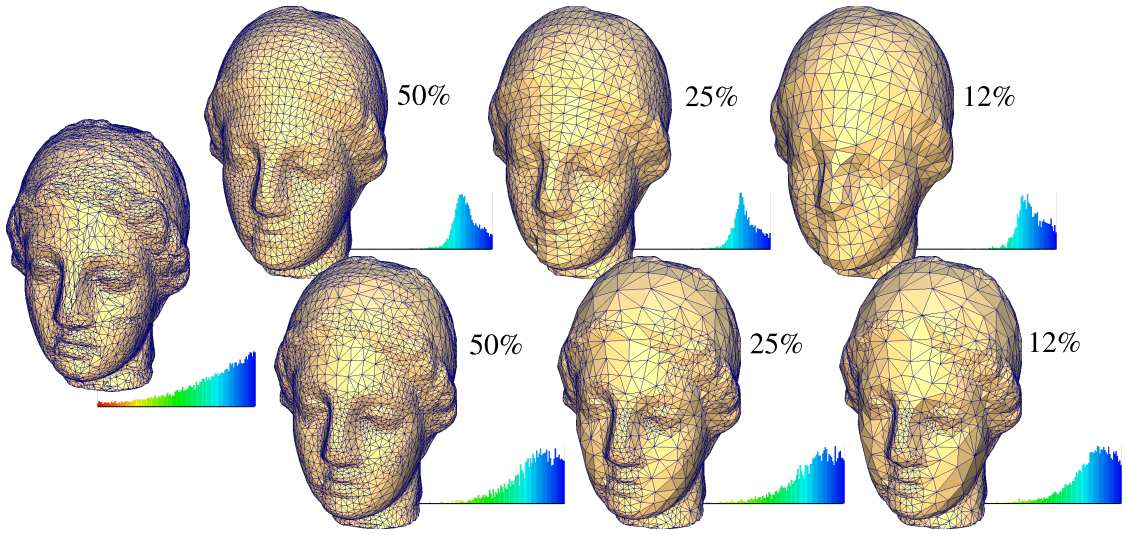


Fig. 1. Overview of our Fiedler tree approach. From the original egea model on the left, we are able to generate quality uniform meshes at different scales (top row). Due to the hierarchical nature, feature-sensitive meshes are also easily generated (bottom row)

shaped surface patches, mesh-independence, and noise robustness, among others. Moreover, we are able to generate high quality uniform meshes at multiple scales. Uniform in our case refers to uniform triangle areas and consistently good quality in the resulting triangles, measured by the triangle radius ratio metric.

Due to the properties of our construction, we argue that we have a well-defined notion of *scale* on the surface. This provides for a natural means of constructing wavelets on a surface, as scale is notoriously difficult to define on a sampled manifold [19], [11]. As an application, we illustrate the construction of a Haar wavelet basis, and from this wavelet basis, a trivial means of producing feature-sensitive meshes.

Figure 1 illustrates such flexibility, showing from left to right three different resolutions for the model on the left. Two different representations are presented for each resolution level, illustrating the capability of generating high quality uniform meshes (top) as well as adaptive meshes capturing surface features (bottom).

Main Contributions The main contribution of our work is a new hierarchical binary surface decomposition which generates high-quality, well-balanced surface patches, suitable for irregular multiresolution analysis. Specifically, our approach consists of the following contributions:

- *Quality Irregular Multiresolution:* We are able to generate a hierarchy of quality meshes, a task difficult to achieve with respect to current remeshing and simplification schemes.
- *Mesh Independence:* As our decomposition, and corresponding meshes, are completely determined by the Laplace-Beltrami operator, our approach is meshing-invariant.
- *Noise Robustness:* Utilizing the Fiedler vector, we are able to produce quality triangulations even in the presence of high-frequency noise.
- *Multiscale Analysis:* The binary hierarchy permits a multiscale analysis very similar to a Haar wavelet decomposition, making noise and feature identification quite natural.

2. Related Work

Our approach spans a variety of areas, ranging from: mesh decimation, multiscale surface representation, remeshing, and spectral geometry processing. It is beyond the scope of this paper to thoroughly cover each area. We will instead concentrate on the most relevant approaches in each respective field.

One of the first uses of mesh decimation for model simplification was presented in the work by Hoppe et al. [14], where edge-collapses and vertex-splits are used to simplify mesh connectivity while vertex positioning is set through a minimization procedure. Many works have built on that seminal idea toward creating multiresolution mesh representation schemes, with *progressive meshes* [13], *multiresolution signal processing for meshes* [11], and *surface simplification using quadric error metrics* [9] being good representatives of this class of algorithms. Indeed, the quadric error driven simplification scheme proposed by Garland and Heckbert [9] has become a reference in the context of decimation-based multiresolution representation, mainly due to its feature preservation, efficiency, and simplicity.

Multiresolution meshes may also be generated via partitioning of the ambient space for which the mesh lies in. Uniform [32] and adaptive [36], [22], [34] grids may be employed, where simplification is performed via collapsing vertices that belong to equivalent nodes. These approaches have the advantage of being fairly noise insensitive while still retaining a good computational performance. However, feature preservation and mesh quality are issues that must be carefully addressed with this kind of representation, as unbalanced partitions

are prone to generate bad quality meshes and poorly preserve features. The approach of [3], termed the *VS-tree*, addresses feature preservation by switching to a quad-tree structure at finer levels. However, building good quality meshes still remains an issue.

Generation of good quality, feature preserving meshes can be achieved via remeshing schemes [1]. These methods range from centroidal voronoi diagrams (CVD) [42], [46] to more general optimization methods [41], [39]. CVD methods, in particular, build a cellular complex which minimizes an energy functional, whose dual complex ensures a high-quality triangulation, while still sufficiently preserving features. These methods, however, are rather sensitive to noise and computationally costly, particularly for the purposes of generating a hierarchy of quality meshes, as any algorithm would need to be run from scratch each time for each resolution. These two issues can be alleviated via heuristic methods [4], albeit at the cost of good quality meshes.

Our tree construction is based on recent results in the areas of spectral graph theory and spectral geometry processing; we refer to [2] and [21] for comprehensive surveys on the subjects, respectively. Spectral bisection is well known in graph theory, and has been used for the purposes of dynamic load balancing [45], sparse matrix ordering [12], and partitioning finite element meshes [18].

Spectral methods have enjoyed much popularity in geometry processing the past decade, with applications ranging from registration [16], [25], segmentation [23], [6], and shape comparison [31], [38], amongst many others. Perhaps most relevant to our approach is the recent work in spectral surface quadrangulation [7], [15], in that we both utilize Laplace-Beltrami eigenfunctions for the purposes of remeshing. However, choosing a single eigenfunction to remesh from is difficult, as the number of critical points of a shape is highly dependent on the complexity of its geometry and topology. Additionally, it is nontrivial to build a nested hierarchy of meshes by choosing a single eigenfunction. We circumvent these issues by recursively choosing the Fiedler vector.

3. Fiedler Binary Tree Decomposition

The proposed framework relies on a binary hierarchical structure to carry out the multiscale decomposition. Once the hierarchical structure is established, a CW complex is constructed from which a triangulation can be built. Details on how to accomplish the tree construction follows in this section, while triangulation is handled in section 4.

3.1 Tree Construction

In order to construct a binary decomposition of the surface mesh, we require a mechanism to recursively split the mesh in two parts. Partitioning a surface into two surface patches amounts to finding a cut along the surface, or equivalently, finding a series of curves

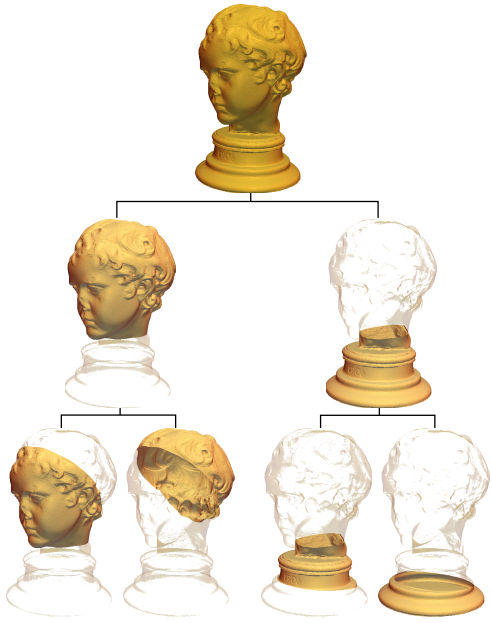


Fig. 2. Binary mesh decomposition: each patch (tree node) is recursively split on the Fiedler nodal line.

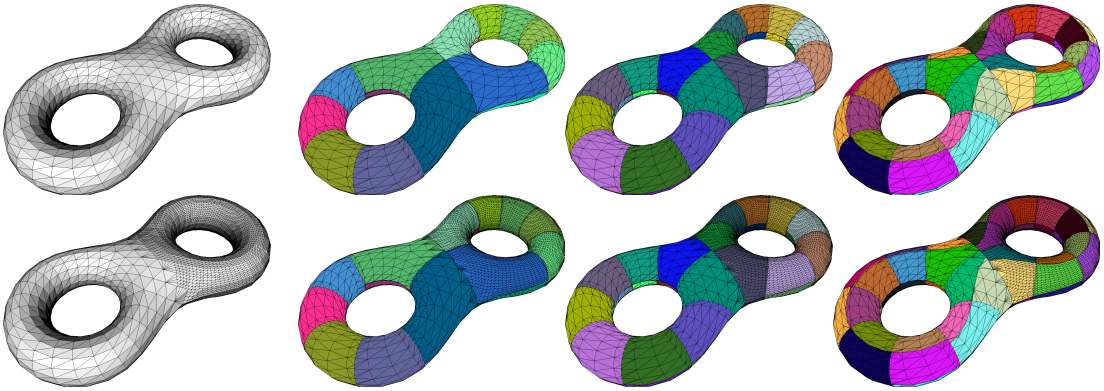
which splits the surface into two connected components. We utilize the nodal regions of the Laplace-Beltrami eigenfunctions to make these splits. Namely, we use the first nontrivial eigenfunction of the Laplace-Beltrami operator, which in graph theory circles is commonly referred to as the *Fiedler vector*, when considering the more general Laplacian. Splitting along the zero-set of the Fiedler vector ensures a split of the surface into exactly two connected components from the Courant Nodal Domain theorem [10], hence ensuring a binary decomposition.

To this end, we employ the discrete Laplace-Beltrami operator of [43], utilizing dual barycenter areas. In the computation of the Fiedler vector we also use the method of [43] in performing a spectral shift, in order to ensure a faster convergence in eigenvector computations.

Once we have computed the Fiedler vector on the original surface we isocontour the zero set, assuming linear interpolation, to split the mesh in two patches. From the two newly created surface patches, we simply recurse this process until a user-defined level of the tree is met. See Figure 2 for an illustration.

Note that we exactly isocontour the surface, rather than respect the original connectivity of the surface. By exactly cutting along the zero set, we are not inhibited by highly irregular meshes where portions of the surface may have large triangles, and others may have small triangles. Therefore, we are able to keep the notion of scale on a surface mesh independent.

For numerical robustness, we take care of instances where the Fiedler vector contains values *approximately* zero. If the value of the Fiedler vector in a vertex is very close to zero, then the resulting submesh may contain very poor triangles (i.e. skinny), posing numerical instability issues for the eigenvector computations. We assign



(a) Original meshes (b) Patches in depths 4, 5, and 6

Fig. 3. Intrinsic nature of the hierarchical decomposition. Almost identical decompositions are generated from meshes with varying refinement.

these vertices a small random value, within a range that will render the submesh numerically stable.

When splitting the mesh, every triangle along the nodal line is split into 3 separate triangles, where two triangles will be assigned to one of the submeshes, and the other triangle to the other submesh. This results in a significant amount of triangles being created at finer scales, though we suspect that symbolically cutting triangles similar to [46] is a viable alternative and would save much memory.

3.2 Fiedler Tree Properties

By splitting along the Fiedler vector, we inherit several attractive properties in our decomposition. The Fiedler vector is known to be a good approximation to the normalized min-cut [37] in the segmentation literature. For the decomposition of a surface Ω into $\Omega = \Omega_1 \cup \Omega_2$, we recall the cut energy as:

$$Ncut(\Omega_1, \Omega_2) = \frac{cut(\Omega_1, \Omega_2)}{assoc(\Omega_1, \Omega)} + \frac{cut(\Omega_1, \Omega_2)}{assoc(\Omega_2, \Omega)} \quad (1)$$

where $assoc$ is a measure of similarity between two domains, and cut measures the dissimilarity in the boundary between Ω_1 and Ω_2.

In contrast to segmentation approaches, our measure of similarity is entirely uniform, in that by using the Laplace-Beltrami operator we are only considering the intrinsic geometry for the purposes of segmentation. Thus, if we denote $\xi = \Omega_1 \cap \Omega_2$ as the nodal set, we in fact have:

$$Ncut(\Omega_1, \Omega_2) = \frac{l(\xi)}{s(\Omega_1)} + \frac{l(\xi)}{s(\Omega_2)} \quad (2)$$

where l represents the length of a curve, and s represents surface area. Thus in our case the Fiedler vector approximates the minimization of the ratio of nodal set length to surface area [40]. As a result, for every split we are likely to obtain surface patches which are of similar surface area, while the split itself is of small length, and typically of small Gaussian curvature magnitude on the boundary. We argue that both of these properties give rise to a well-defined notion of *scale* in the decomposition.

Our tree construction is also mesh-independent. That is, for a given surface meshed in two different ways, our construction will produce identical decompositions. Seeing as the Laplace-Beltrami operator is isometry-invariant, this should come as no surprise. Only at very fine scales does the decomposition begin to differ, due to using linear interpolation in making the cuts. Figure 3 illustrates the mesh-independent property, showing patches in three different levels of the hierarchy. Notice that patches are practically indentical in the top and botton rows, even though the construction is performed with respect to completely different meshings (the left-most models).

Last, it has been illustrated in previous works [20], [10] that the Fiedler vector, in some sense, follows the "shape" of the surface. For the purposes of our construction, we find that for tubular and anisotropic surface patches, the zero set of the Fiedler vector consistently aligns with the maximum principal curvature directions. In other words, the cut tends to be along the minimum axis of the surface, and as a consequence, effectively removes the anisotropy of the surface. See Figure 4 for an illustration.

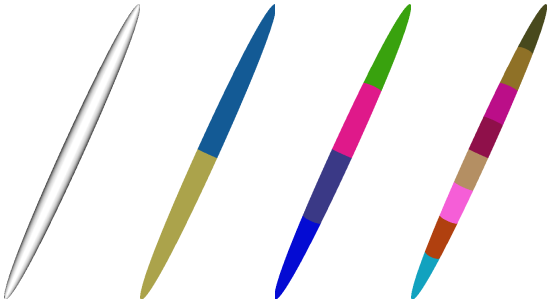


Fig. 4. Our decomposition tends to split along the minimum axis, and consequently along maximum principal curvatures, as illustrated for an ellipsoid.

4. Triangle Mesh Generation

Producing a triangulation from the tree construction involves topological and geometric considerations. We handle both in turn.

4.1 Topological Construction

At the end of the tree construction process, we are left with a set of surface patches at all scales. At some scale, each surface patch will become homeomorphic to a topological disk. At this scale we have in fact constructed a *cell complex*, or CW-complex. For the space of a 2-manifold, a CW-complex consists of a set of 0, 1, and 2-cells, where an n-cell is homeomorphic to an n-ball, and the boundary of an n-cell strictly consists of cells of dimension $m < n$ [27]. In our context 2-cells are the surface patches, 1-cells are arcs on the boundary of the patches whose ends are the 0-cells, or vertices.

The significance of the CW-complex for our purposes lies in the fact that, under certain circumstances, its dual complex is a valid triangulation. The dual complex of the CW-complex takes every n-cell and maps it to a unique $(2-n)$-cell, such that every 2-cell becomes a point, every 1-cell becomes an edge, and a 0-cell becomes a facet. Each 0-cell will map to a triangle if and only if the number of 1-cells which intersect to form the 0-cell is exactly 3. As our tree construction always cuts every edge the zero set crosses, open zero sets along the surface will always start/end at unique points, and consequently, we are always guaranteed triangle elements.

The only remaining issue is whether or not the dual complex is indeed a valid triangulation. There are three cases where zero set cuts will result in invalid triangulations, which correspond with violations of the *closed ball property* [8]:

- The zero set is closed.
- The zero set consists of multiple connected components.
- The zero set starts and ends at the same 1-cell.

The first case results in a dangling edge, the second case results in a degenerate triangle, and the third case results in the creation of duplicate triangles. Hierarchical space partitioning approaches [34], [3] suffer from similar problems; however, since we are partioning the surface directly, we may trivially detect these cases. We find that the first two cases only occur in coarse levels of the tree, as when we approach finer levels, the 2-cells begin to resemble developable, convex, topological disks, for which the zero set is known to be open and of a single component [26]. The third case, however, may still occur at any level, although in practice it is rare to occur at finer levels of the tree. In all examples throughout the paper, we have found that the closed ball property is first satisfied at a rather coarse level, and is consistently satisfied at all finer levels.

Care must be taken in the implementation of this hierarchical CW-complex for the purposes of memory efficiency. To this end we only store the triangles of the finest-scale CW-complex, that is, we label the triangles in the finest level in accordance with patches in that level. Moreover, ids are assigned such that the multiresolution structure is maintained. In other words, if a triangle has a label k in the finest level then it will be labeled in its father node as $\lfloor \frac{k}{2} \rfloor$, ensuring a consistent hierarchical labeling scheme. Therefore, a patch with id k at level j is labeled as $2k$ or $2k+1$ at level $j+1$ (the same being valid for the triangles representing these patches). Hence we are always able to process the CW-complex at any scale, strictly from the finest scale.

We next illustrate two mechanisms for generating meshes: multiresolution uniform meshes, and quadric error meshes.

1) Multiresolution Uniform Meshing: Generation of a uniform mesh amounts to reconstruction at a particular depth (i.e. scale) in the tree. Namely, for a prescribed resolution j, we identify the patches corresponding to depth j using the scheme as described above. This effectively corresponds to the CW-complex at scale j. From here, we identify the 0-cells to be the triangles in the dual triangulation, where a dual triangle's vertices are determined by the intersecting three 2-cells. This construction guarantees an oriented simplicial complex decomposition of the surface. Spatial partioning approaches [32], [34], on the other hand, encounter difficulty in ensuring a decomposition that guarantees a well defined simplicial complex as output, as issues may occur in clustering points which are close in Euclidean distance yet far apart in geodesic distance.

2) Quadric Error Meshing: Similar to previous approaches [34], [3], we may utilize our spatial decomposition for the purposes of applying quadric error-based decimation [9]. The primary difference here is that we have well-defined surface patches, both in terms of shape

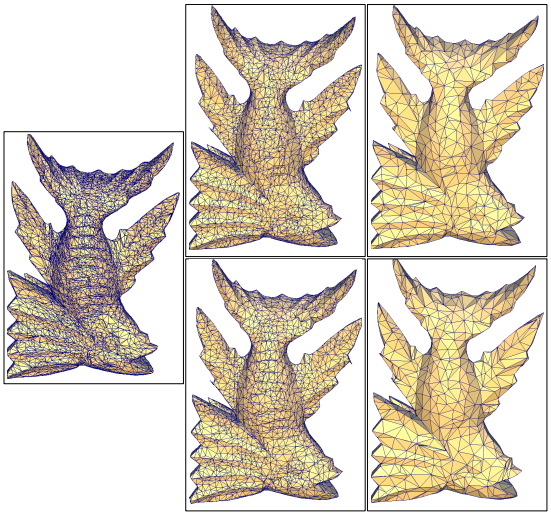


Fig. 5. QSlim decimation (top), compared to our quadric error meshing approach (bottom). Eigenvector computation time: 14s. Qslim timing for 4K and 1K vertex decimation, respectively: 44ms and 54ms. Our timing for 4K and 1K vertex decimation, respectively: 31ms and 57ms.

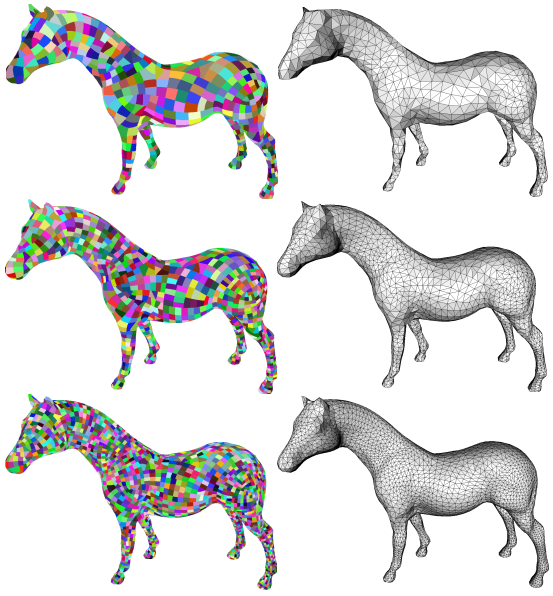


Fig. 6. The CW-complex and corresponding triangulations, for different scales. Note the consistency in the quality of the decomposition, as we go to finer scales.

and uniform area across all scales, whereas spatial partitioning approaches greatly suffer from nonuniformity as a result of axis-aligned spatial decompositions.

We prioritize nodes of the tree starting from the finest level, where the priority is the quadric error metric. When two neighboring nodes have both been removed, we may add their parent to the queue for processing. When adding parent nodes to the queue, we may simply add their child quadric error functions together; however note that since we have a binary tree structure, it is relatively inexpensive to compute the quadric error function from scratch. In fact it is $O(|V|log|V|)$ in the number of vertices $|V|$, whereas [9] rely on edge collapses, and consequently it would be quadratic in their approach.

Once we have selected the subset of nodes to be retained, we need to generate the dual triangulation. We associate each 2-cell with its scale and id, and then generate a unique id for each (scale, id) pairing. This gives us a consistent CW-complex representative of the quadric error decimation. Generation of the dual triangulation then proceeds in exactly the same manner as above.

See Figure 5 for a comparison between our approach and qslim. Note that the results are quite similar; however, the order of complexity of our approach is $\frac{|V|}{2}$, where $|V|$ is the number of vertices, whereas qslim works off of edge collapses, hence the complexity for a typical mesh with qslim is of the order $3|V|$, which is roughly the number of edges.

4.2 Geometric Embedding

In computing a representative vertex for every 2-cell, its center of mass is a logical choice. That is, for every 2-cell, we may take the area-weighted coordinate as the vertex position.

A disadvantage to using the center of mass is that we may miss features on the surface. If feature preservation is desired, we may position vertices according to the quadric error metric, taken with respect to the 2-cell. By doing so, however, our mesh quality suffers. To satisfy both ends, we opt to interpolate between the center of mass and the quadric error vertex, by a user-defined parameter α. This way, the user may choose between high-quality triangulations and feature preservation.

4.3 Triangulation Properties

If we are to use the center of mass for vertex positions, then our construction is able to produce high-quality triangulations. This is a consequence of the tree construction properties discussed in section 3.2. The fact that the nodal curves tend to follow the maximum principal curvature directions results in edges in the dual triangulation following the minimum principal curvature directions. This also accounts for the "quad-like" structure in our meshes, and consequently our triangles are slightly anistropic in the principal directions of the curvature tensor. As well, the property of surface patches being of almost uniform area for each level results in triangles containing very similar areas in the dual triangulation. See Figure 6 for an example illustrating these properties across several scales.

Simultaneously satisfying small-length nodal curves and equi-areal surface patches is rather difficult, and occasionally the Fiedler vector will favor one over the other. In the former case, this will result in nonuniform surface areas, and hence the dual triangulation will have triangles of varying areas. In most cases, however, we have noticed this to be desirable; for instance, the legs of the horse in Figure 6 should be meshed denser than the stomach. In the latter case, nodal curves may result in surface patches being non-convex, in which case

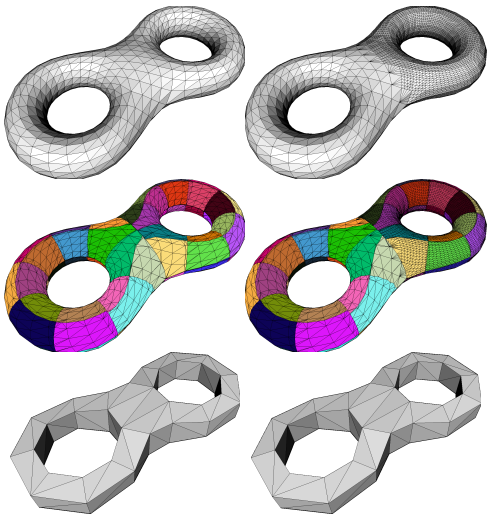


Fig. 7. Mesh generation for the eight model from two different meshings of the same surface.

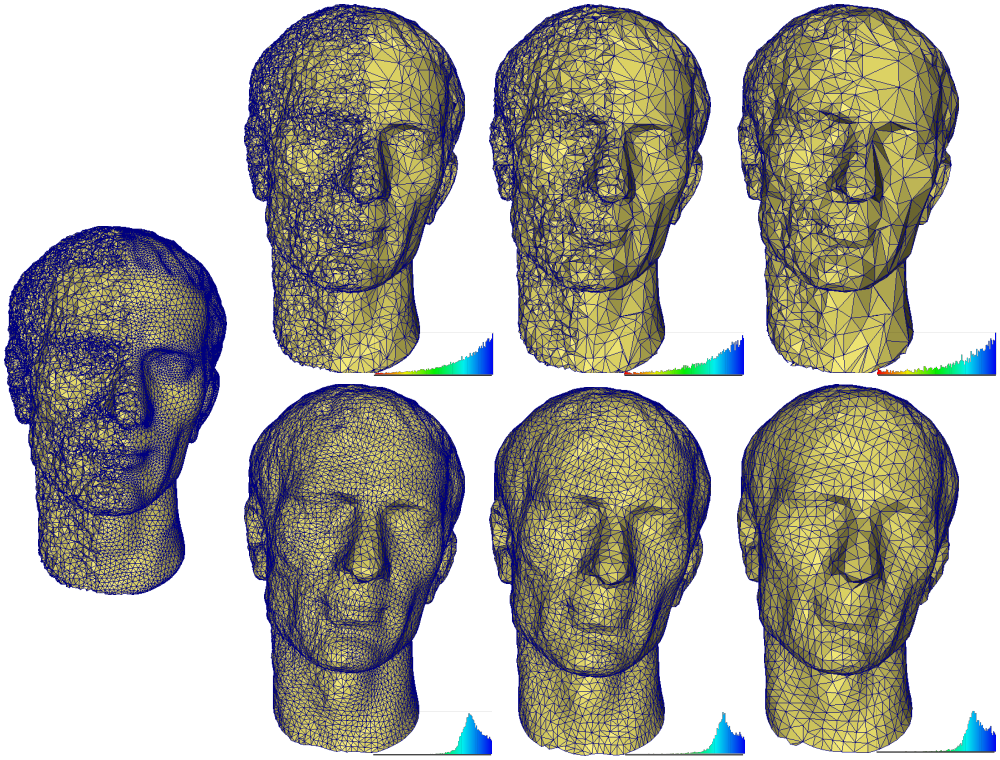


Fig. 8. Multiscale representation of half-noise Julius model (left most). Our approach (bottom row) can robustly smooth noise out while still producing good quality meshes in every level of the hierarchy. The noise remains prevalent when QEM is used as a simplification mechanism, thus preventing the generation of good meshes. Histograms on the bottom right of each model show the triangle radius ratio quality for each model.

skinny triangles and high-valence vertices are produced. In practice we have observed that this rarely occurs.

The property of mesh independent tree constructions in fact translates to near identical triangulations. See Figure 7 for an example. Note that there are subtle differences in the meshes, as neighboring 2-cells may differ, corresponding to a difference of an edge flip in the triangulations.

Last, we note that our meshes are very robust to geometric noise. As pointed out in previous work [33], the low-frequency eigenfunctions of the Laplace-Beltrami operator are robust to even topological noise, in addition to geometric noise. The Fiedler vector being the lowest frequency nontrivial eigenfunction, it is most robust. This is a property inherited throughout our hierarchy, as Figure 8 illustrates. The noise in this example is generated by perturbing the per-vertex normals, and displacing the vertices a small amount along this perturbation. We are additionally able to produce high-quality triangles in the presence of noise, as our triangle radius ratio histograms demonstrate.

5. Fiedler Multiscale Analysis

Multiscale analysis usually relies on recursively decomposing a given signal into low-frequency and high-frequency components. Although different approaches can be used to compute low and high-frequency components of a signal in each resolution, such as expansion in a set of basis functions or prediction/updating schemes [17], all multiscale methods demand a splitting mechanism (also called up-sampling) in order to identify the subset of data that will be "shifted" to the next coarser level. Efficient splitting schemes are particularly difficult to be defined on unstructured data, as a biased choice might introduce artifacts in the multiscale decomposition. Our hierarchical scheme, however, provides for an intuitive notion of *scale*, and hence is an attractive starting point for many multiresolution methods. We illustrate such functionality by implementing a Haar-like multiscale analysis using our decomposition as a splitting mechanism.

Let γ_k^j be a surface patch with index k at scale j of the tree. Denoting by γ_{2k}^{j+1} and γ_{2k+1}^{j+1} the children nodes of γ_k^j, we can compute scaling and detail coefficients c_k^j, d_k^j in γ_k^j by simple averaging and differencing from scaling coefficients c_{2k}^{j+1} and c_{2k+1}^{j+1} in γ_{2k}^{j+1} and γ_{2k+1}^{j+1}. More specifically, scaling and detail coefficients in level j can be computed as [17]:

$$c_k^j = \frac{|\gamma_{2k}^{j+1}|}{|\gamma_k^j|} c_{2k}^{j+1} + \frac{|\gamma_{2k+1}^{j+1}|}{|\gamma_k^j|} c_{2k+1}^{j+1} \quad (3)$$

$$d_k^j = c_{2k}^{j+1} - c_{2k+1}^{j+1} \quad (4)$$

where $|\gamma_k^j|$ is the area of the surface patch k at scale j. At the finest scale J, we take the c_k^J to be the area-weighted average of the function values on that surface patch (assuming the function is constant in each patch of the finest level). Similarly, an inverse transform may

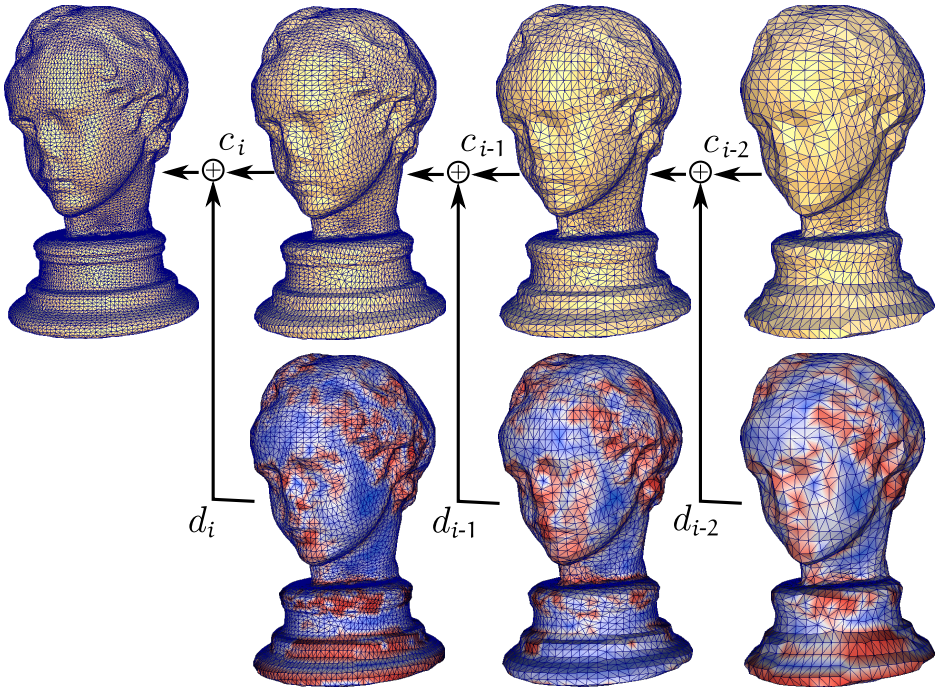


Fig. 9. Illustration of the multi-scale decomposition of the normals. From right to left, we are adding more details to the model, until we get the original surface back.

be applied as follows:

$$c_{2k}^{j+1} = c_k^j + \frac{|\gamma_{2k+1}^{j+1}|}{|\gamma_k^j|} d_k^j \tag{5}$$

$$c_{2k+1}^{j+1} = c_k^j - \frac{|\gamma_{2k}^{j+1}|}{|\gamma_k^j|} d_k^j \tag{6}$$

The capability of computing scaling and detail coefficients complements the binary hierarchical decomposition with a natural mechanism to detect features and surface details. In fact, we may utilize the Haar wavelet decomposition for the purposes of detecting multi-scale features in the mesh. To this end, we analyze the variation in per-vertex normals. If we denote the components of normal vectors as functions n_x, n_y, n_z over the surface, we may run our Haar decomposition, as described in equation (4) to obtain wavelet (detail) coefficients dx, dy, and dz for each coordinate function, respectively. By setting $\mathbf{d}_k^j = (dx_k^j, dy_k^j, dz_k^j)$ as a vector in every node k at scale j, we can take $\| \mathbf{d}_k^j \|$ as a feature measure at node k (and level j) of the tree. An example of such a Haar-like decomposition can be seen in Figure 9, where the warmer colors in the bottom models represent high values of detail coefficients. Notice that by going from right to left, more details are added in the model, characterizing the typical behavior of a multiscale scheme.

Scaling and detail coefficients may also be exploited for the purposes of feature detection and vertex positioning during the multiresolution process. In fact, we have exploited the Haar-like multiscale analysis for:

Feature-sensitive Meshing: The feature measure described above may be easily leveraged to produce adaptive meshes; that is, meshes where the sampling density is a function of the features of the mesh. This is achieved by culling nodes (i.e. 2-cells) from the tree in a greedy manner prioritized by $\| \mathbf{d}_k^j \|$.

Similar to the quadric error meshing, we first place all leaf nodes in the tree in a priority queue. A tree node is added to the queue only if its children have been removed. Additionally, in order to maintain nice triangulations and prevent high valence vertices, we do not allow the merging of two nodes n_{2k}^{j+1}, n_{2k+1}^{j+1} into n_k^j if a child of the neighbor node of n_k^j still exists. Once all nodes have been removed, the triangulation is generated in the exact same manner as section 4.12. This adaptive mechanism was used to generate the bottom models in Figure 1.

Multiresolution Embedding: In section 4.2 we demonstrated a means of computing the center of mass over every surface patch. This is unfortunately of complexity $O(|V|log|V|)$ to compute. However, we may make the computation linear by noting that the projection of the coordinate functions onto the Haar basis exactly corresponds to the center of masses at different scales. That is, the scaling coefficients of the coordinate functions at a particular scale correspond to the center of masses computed at that scale. Only the finest scale integration needs to be computed.

6. Experimental Results

In this section we present the results of applying the described methodology for the purposes of generating multiresolution uniform meshes and feature-sensitive meshes. All the models presented in the following

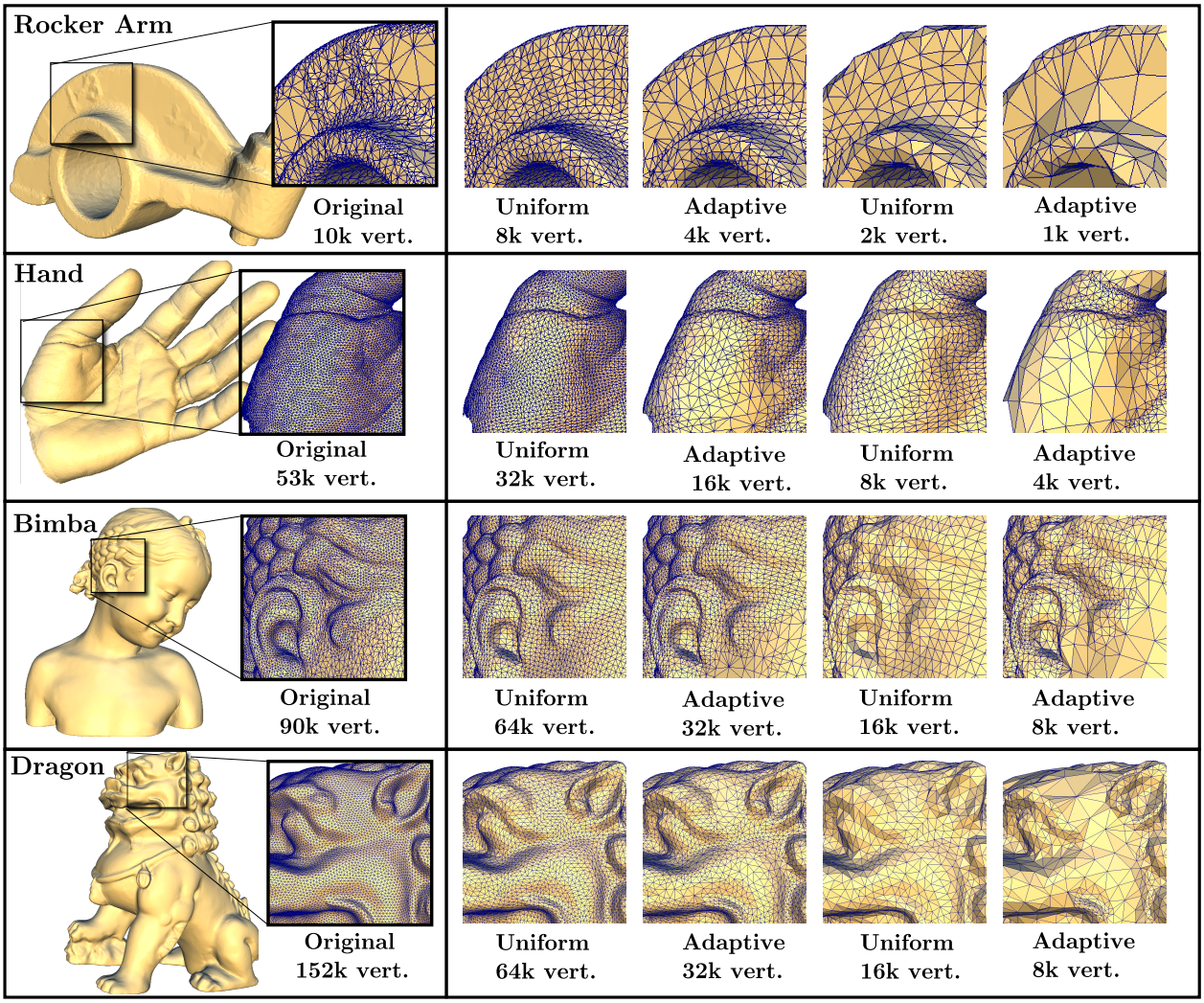


Fig. 10. Uniform and adaptive meshing results for a variety of surface meshes.

applications were generated on a MacBook with a dual-core processor of 2 GHz and 2 GB of memory.

While minimum angle in a triangle is a common quality measure in the remeshing literature, we find that our meshes are slightly anisotropic in the curvature tensor; see section 4.3 for a discussion on this matter. Hence minimum angle is not a fair measure of quality for our meshes. For this reason, we measure mesh quality by the incircle to circumcircle ratio, commonly referred to as the radius ratio.

Figure 10 demonstrates our results for a variety of surface meshes, uniform and adaptive meshing alike. The rocker arm mesh demonstrates our method's robustness to meshes with highly irregular geometry and connectivity, where discrete variational methods face problems [42].

Figure 11 shows our multiresolution scheme applied to the fertility model, decimated to 16K and 8K vertices from 240K vertices. Note the drastic improvement in mesh quality (top part), and our method's resilience to the input triangulation. The mesh independence of our construction ensures a high-quality triangulation, regardless of how the input surface is meshed.

Table 1 shows quality statistics for these meshes. We note that for the uniform meshes, and the other uniform meshing results shown throughout, we obtain very consistent triangle radius ratio histograms, *independent*

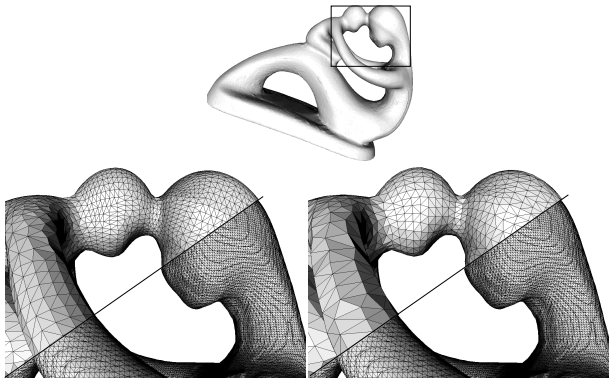


Fig. 11. Fertility model, 240K vertices, uniformly decimated to 16K vertices, 8K vertices. Our Fiedler approach is shown in the top-half image.

TABLE 1
MESH QUALITY OF MULTIRESOLUTION MODELS PRESENTED IN FIGURE 10. NUMBERS IN EACH ENTRY CORRESPOND TO AVERAGE QUALITY, WORST CASE, AND PERCENTAGE OF TRIANGLES WITHIN THE INTERVAL $[0.5, 1.0]$, WHICH COMPRISES THE GOOD QUALITY TRIANGLES.

Rocker Av Wt [%]	Un 8K 0.84 0.06 99.0%	Ad 4K 0.82 0.02 98%	Un 2K 0.84 0.15 99.3%	Ad 1K 0.80 0.18 94.4%
Hand Av Wt [%]	Un 32K 0.82 0.13 99.7%	Ad 16K 0.82 0.15 98.8%	Un 8K 0.83 0.25 99.8%	Ad 4K 0.81 0.01 96.2%
Bimba Av Wt [%]	Un 64K 0.83 0.02 99.5%	Ad 32K 0.83 0.01 98.5%	Un 16K 0.83 0.16 99.7%	Ad 8K 0.80 0.05 95.2%
Dragon Av Wt [%]	Un 64K 0.83 0.01 99.7%	Ad 32K 0.83 0.01 98.5%	Un 16K 0.84 0.15 99.7%	Ad 8K 0.81 0.04 95.3%
Fertility Av Wt [%]	Un 16K 0.82 0.10 99.8%	Ad 8K 0.82 0.10 98.2%	Un 4K 0.83 0.17 99.6%	Ad 2K 0.80 0.11 95.1%

of the particular mesh, in a similar manner to [35]. Indeed, the vast majority of the triangles produced with our method tend to have $90\,^\circ$ angles, and consequently we produce many triangles with angles approximately $< 30\,^\circ, 60\,^\circ, 90\,^\circ >$, due to the slight anistropy of our method. This is reflected by the peaks in the histograms. It is worth pointing out that for uniform meshing, our approach resulted in more than 99% of good quality triangles, where the notion of a good quality triangle is such that its radius ratio is greater than 0.5 [35]. This reinforces our method's capability to generate quality multiresolution meshes.

TABLE 2
COMPUTATIONAL TIMES TO COMPUTE THE FIEDLER VECTOR DURING THE TREE CONSTRUCTION.

Model	Rocker	Hand	Bimba	Dragon	Fertility
Size	10K vert.	53K vert.	90K vert.	152K vert.	240K vert.
# Levels	13	15	16	16	14
Eigen Calc.	8s	49s	1m40s	2m48s	4m44s

Table 2 shows the computational time involved in the Fiedler vector computation. Times refer to the total time, that is, the 8 seconds shown in the column of the rocker arm model is the time to carry out the eigen decomposion in the $2^{13} - 1 = 8,191$ nodes (the Fiedler vector is not computed in the tree leaves).

Figure 8 demonstrates qslim's inherent limitation in mistaking noise as features. Space decomposition-based methods tend to be more robust to noise, so we have compared our approach to that of the VS-tree [3] in Figure 12. Although the VS-tree has the capability to construct a decomposition on the surface at a fine-enough level, utilizing a height field indicator in the presence of high-frequency noise results in unreliable analysis. The Fiedler tree, however, remains invariant to this high-frequency noise, sufficiently smoothing the mesh. We note that the VS-tree and qslim have the advantage of being computationally efficient, whereas our method is significantly more time consuming. However, our comparisons illustrate flaws in these approaches, resulting from the lack of a proper analysis of the surface at multiple scales, which is precisely what our method excels at.

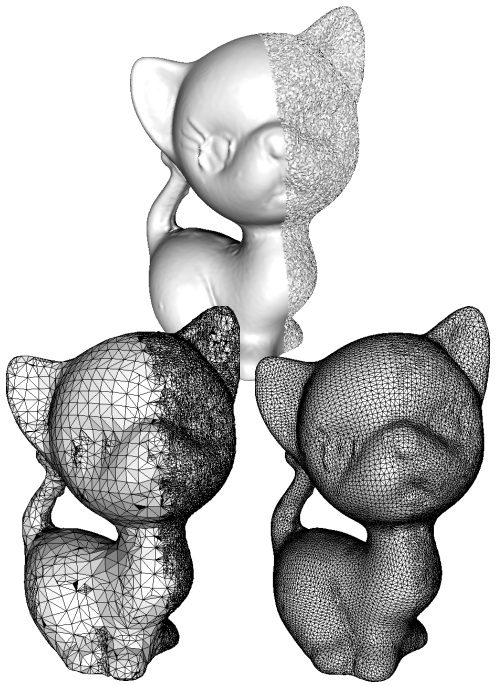


Fig. 12. Comparison of VS-tree [3] (left) to our approach (right), for simplification of a noisy surface. The original surface (135K vertices) is decimated to 21K vertices for both approaches. Timing for VS-tree: 70ms, timing for our method: 2m30s for eigenvector computations, and 900ms to generate the mesh

Last, we have compared the quality of our meshes to that of a state of the art remeshing algorithm, delpsc [5]. See Figure 13 for a comparison of the egea model, remeshed to approximately 4K vertices. Our results are competitive in terms of triangle radius ratio, albeit not quite as good; however we are able to construct a multiresolution hierarchy of quality meshes, whereas delpsc operates with respect to a target number of vertices.

7. DISCUSSION AND LIMITATIONS

Examples and comparisons presented in Sections 3–6 support that our multiresolution scheme gathers a set of properties not present in any other approach devoted to represent meshes in multiresolution. Table 3 exemplifies this fact, in comparing our approach to the various methodologies. As can be observed (the symbol ✓ means a property is present), only the Fiedler tree

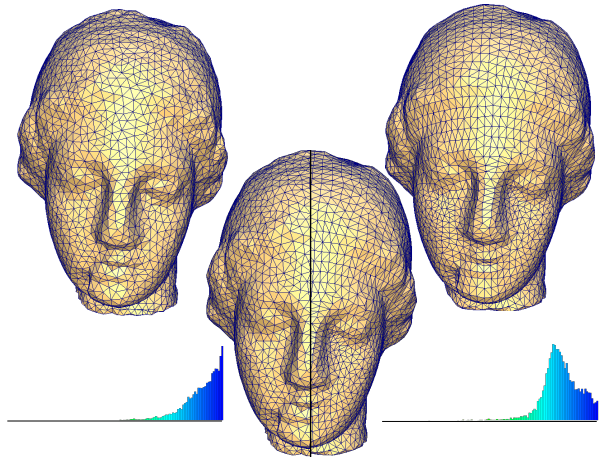


Fig. 13. Comparison of delpsc [5] on the left, to our method on the right, with corresponding triangle radius ratio histograms. Timing for delpsc: 7.4s, timing for our method: 7.5s for eigenvector computations, and 109ms to generate the mesh

endows the intrinsic properties of mesh independence, noise robustness, mesh quality, feature detection, and multiresolution. The symbol • indicates a property is not intrinsic, but can be somehow approximated through tuned implementation.

TABLE 3
COMPARISON OF OUR APPROACH TO OTHER METHODOLOGIES. THE SYMBOL ✓MEANS THE PROPERTY IS PRESENT WHILE THE SYMBOL •INDICATES THE PROPERTY CAN BE SOMEHOW INCORPORATED.

Method / Property	Mesh Independence	Noise Robustness	Mesh Quality	Feature Detection	Multi-resolution	Comput. Efficiency
Decimation Methods				✓	✓	•
Space/Tree Partition		•		•	✓	✓
Fiedler Decomposition	✓	✓	✓	✓	✓	
Remeshing Methods			✓	✓		

Table 3 also suggests that the proposed Fiedler tree represents a methodology that stands between hierarchical space decomposition and remeshing approaches. Our approach shares the conceptual simplicity of space decomposition techniques, as we are merely performing a top-down hierarchical partitioning of the surface, instead of the volume in which the surface resides. We are able to produce meshes which are of competitive quality to that of remeshing schemes, yet at the same time, our approach is much simpler in comparison to most remeshing schemes.

Another interesting aspect of our approach is the ability to analyze features at multiple scales. The intrinsic hierarchical structure provided by the Fiedler tree makes multiscale analysis quite natural. In fact, the Haar-like implementation described in section 5 is only the simplest mechanism in carrying out multiscale feature analysis. We believe that more sophisticated and accurate schemes can be derived on top of our decomposition.

Our binary hierarchical mesh decomposition is only one way of decomposing a mesh, and many hierarchical segmentation methods, including spectral methods, exist in the literature [23], [6], [30], [29]. However, recall that the advantage of utilizing the Fiedler vector is in generating patches which have small boundary length, and consistent surface areas. As segmentation methods assume some notion of part saliency, they are unlikely to satisfy these properties, especially in the absence of saliency, which is common at finer depths in the decomposition. We note that a possible extension to our decomposition is choosing a different eigenfunction which still splits the mesh into two connected components, while satisfying other properties such as reflectional symmetries [28], [44]. This could lead to a method for intrinsically symmetric remeshing, and we leave this for future work.

The main limitation of our approach is the computational burden, including processing time and memory consumption. Althouth Table 2 shows our technique could be applied to process fairly big meshes on a conventional laptop, massive meshes would demand out-of-core eigenvector computation methods, especially in the first levels of the hierarchy, increasing computational times considerably. Moreover, by cutting exactly along the surface, we are encumbered by an increasing number of triangles being produced at every scale. This hinders the performance and memory efficiency of our method.

8. Conclusion and Future Work

We have presented a new method for multiresolution analysis by utilizing spectral surface methods for a multiresolution construction. We have demonstrated applications to quality uniform and adaptive mesh generation, and the inherent robustness to noise.

For future work, we intend to improve on the efficiency of the construction. More sophisticated methods for cutting the mesh would lead to more efficient, robust means of constructing the domains. Additionally, we feel that the decompositions produced may provide an effective initialization for Voronoi-based methods to start from, leading to higher quality meshes.

Acknowledgements

We thank the anonymous reviewers for their detailed feedback in improving the paper. We also thank Tiago Etiene Queiroz for help in creating surface meshes. This work was supported in part by grants from the National Science Foundation (grants IIS-0905385, CNS-0855167, IIS-0844546, ATM-0835821, CNS-0751152, OCE-0424602, CNS-0514485, IIS-0513692, CNS-0524096, CCF-0401498, OISE-0405402, CCF-0528201, CNS-0551724), the Department of Energy, and Fapesp-Brazil (#2008/03349-6).

References

[1] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene. Recent advances in remeshing of surfaces. *Shape Analysis and Structuring*, pages 53–82, 2008.

[2] T. Biyikoglu, J. Leydold, and P.F. Stadler. *Laplacian eigenvectors of graphs: Perron-Frobenius and Faber-Krahn type theorems*. Springer, 2007.

[3] T. Boubekeur, W. Heidrich, X. Granier, and C. Schlick. Volume-surface tree. *Computer Graphics Forum*, 25:399–406, 2006.

[4] Tamy Boubekeur and Marc Alexa. Mesh simplification by stochastic sampling and topological clustering. *Comput. Graph.*, 33(3):241–249, 2009.

[5] S.W. Cheng, T.K. Dey, and J.A. Levine. A practical Delaunay meshing algorithm for a large class of domains. In *Proc. 16th Internat. Meshing Roundtable*, pages 477–494. Springer, 2007.

[6] F. de Goes, S. Goldenstein, and L. Velho. A hierarchical segmentation of articulated bodies. In *Computer Graphics Forum*, volume 27, pages 1349–1356. Blackwell Science Ltd, Osney Mead, Oxford, OX 2 0 EL, UK,, 2008.

[7] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J.C. Hart. Spectral surface quadrangulation. *ACM Trans. Graph.*, 25(3):1057–1066, 2006.

[8] H. Edelsbrunner and N.R. Shah. Triangulating Topological Spaces. *International Journal of Computational Geometry & Applications*, 7(4):365–378, 1997.

[9] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[10] K. Gebal, JA Bærentzen, H. Aanæs, and R. Larsen. Shape Analysis Using the Auto Diffusion Function. In *Symposium on Geometry Processing*, pages 1405–1413, 2009.

[11] I. Guskov, W. Sweldens, and P. Schroder. Multiresolution signal processing for meshes. In *Computer Graphics (SIGGRAPH'99)*, pages 325–334, 1999.

[12] B. Hendrickson and R. Leland. A Multi-Level Algorithm For Partitioning Graphs. In *Supercomputing, 1995. Proceedings of the IEEE/ACM SC95 Conference*, pages 28–28, 1995.

[13] H. Hoppe. Progressive meshes. In *ACM SIGGRAPH*, pages 99–108, 1996.

[14] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 19–26, New York, NY, USA, 1993. ACM.

[15] J. Huang, M. Zhang, J. Ma, X. Liu, L. Kobbelt, and H. Bao. Spectral quadrangulation with orientation and alignment control. *ACM Transactions on Graphics*, 27(5):147, 2008.

[16] V. Jain and H. Zhang. Robust 3D shape correspondence in the spectral domain. *Proc. of Shape Modeling International (SMI)*, pages 118–129, 2006.

[17] M. Jansen and P. Oonincx. *Second generation wavelets and applications*. Springer, 2005.

[18] A. Kaveh and A. Davaran. Spectral bisection of adaptive finite element meshes for parallel processing. *Computers and Structures*, 70(3):315–323, 1999.

[19] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multiresolution modeling on arbitrary meshes. In *Computer Graphics (SIGGRAPH 98)*, pages 105–114, 1998.

[20] B. Levy. Laplace-Beltrami Eigenfunctions Towards an Algorithm That Understands Geometry. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006*, page 13. IEEE Computer Society, 2006.

[21] Bruno Levy and Richard Hao Zhang. Spectral geometry processing. In *ACM SIGGRAPH ASIA Course Notes*, 2009.

[22] Peter Lindstrom. Out-of-core simplification of large polygonal models. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 259–262, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[23] R. Liu and H. Zhang. Mesh segmentation via spectral embedding and contour analysis. In *Computer Graphics Forum*, volume 26, pages 385–394, 2007.

[24] M Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics*, 16(1):34–73, 1997.

[25] D. Mateus, R. Horaud, D. Knossow, F. Cuzzolin, and E. Boyer. Articulated shape matching using Laplacian eigenfunctions and unsupervised point registration. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.

[26] A. Melas. On the nodal line of the second eigenfunction of the Laplacian in R2. *J. Differential Geom*, 35(1):255–263, 1992.

[27] J.R. Munkres. *Elements of algebraic topology*. Perseus Books, 1993.

[28] M. Ovsjanikov, J. Sun, and L. Guibas. Global intrinsic symmetries of shapes. In *Computer Graphics Forum*, volume 27, pages 1341–1348. Blackwell Science Ltd, Osney Mead, Oxford, OX 2 0 EL, UK,, 2008.

[29] Martin Reuter. Hierarchical shape segmentation and registration via topological features of laplace-beltrami eigenfunctions. *International Journal of Computer Vision*, doi:10.1007/s11263-009-0278-1, 2009.

[30] Martin Reuter, Silvia Biasotti, Daniela Giorgi, Giuseppe Patanè, and Michela Spagnuolo. Discrete laplace-beltrami operators for shape analysis and segmentation. *Computers & Graphics*, 33:381–390, 2009.

[31] Martin Reuter, Franz-Erich Wolter, Martha Shenton, and Marc Niethammer. Laplace-beltrami eigenvalues and topological features of eigenfunctions for statistical shape analysis. *Computer-Aided Design*, 41(10):739–755, 2009. doi:10.1016/j.cad.2009.02.007.

[32] J. Rossignac and P. Borrel. Multi-resolution 3d approximations for rendering complex scenes. In B. Falcidieno and T.L. Kunii, editors, *Geometric Modeling and Computer Graphics*, pages 455–465. Springer-Verlag, 1993.

[33] R.M. Rustamov. Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, page 233. Eurographics Association, 2007.

[34] Scott Schaefer and Joe Warren. Adaptive vertex clustering using octrees. In *SIAM Geometric Design and Computing*, 2003.

[35] J. Schreiner, C.E. Scheidegger, S. Fleishman, and C.T. Silva. Direct (re) meshing for efficient surface processing. In *Computer Graphics Forum*, volume 25, pages 527–536. Amsterdam: North Holland, 1982-, 2006.

[36] Eric Shaffer and Michael Garland. Efficient adaptive simplification of massive meshes. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 127–134, Washington, DC, USA, 2001. IEEE Computer Society.

[37] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[38] Y. Shi, R. Lai, S. Krishna, N. Sicotte, I. Dinov, and A.W. Toga. Anisotropic Laplace-Beltrami eigenmaps: Bridging reeb graphs and skeletons. In *Proc. MMBIA*, pages 1–7, 2008.

[39] V. Surazhsky and C. Gotsman. Explicit surface remeshing. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, page 30. Eurographics Association, 2003.

[40] A.D. Szlam, M. Maggioni, R.R. Coifman, and J.C. Bremer Jr. Diffusion-driven multiscale analysis on manifolds and graphs: Top-down and bottom-up constructions. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 5914, pages 445–455, 2005.

[41] Greg Turk. Re-tiling polygonal surfaces. *SIGGRAPH Comput. Graph.*, 26(2):55–64, 1992.

[42] S. Valette, J.M. Chassery, and R. Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Trans Visu Comp Grap*, 14(2):369–381, 2008.

[43] B. Vallet and B. Levy. Spectral geometry processing with manifold harmonics,. *Computer Graphics Forum (EUROGRAPHICS)*, 22(2):251–260, 2008.

[44] Bruno Vallet and Bruno Lvy. What you seam is what you get. Technical report, INRIA - ALICE Project Team, 2009.

[45] R. Van Driessche and D. Roose. An improved spectral bisection algorithm and its application to dynamic load balancing. *Parallel Computing*, 21(1):29–48, 1995.

[46] D.M. Yan, B. Lévy, Y. Liu, F. Sun, and W. Wang. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. In *Computer Graphics Forum*, volume 28, pages 1445–1454. Blackwell Publishing Ltd, 2009.