On Generating Multi-resolution Representations of Polygonal Meshes

Dongmei Zhang, Andrew Johson, Martial Hebert and Yanxi Liu
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{dzhang, aej, hebert, yanxi}@ri.cmu.edu

Abstract[1]

Multi-resolution representation is one of the effective approaches to image processing in both computer vision and computer graphics. However, the generation of multi-resolution representation for complex 3D polygonal mesh is a difficult problem. In this paper, a number of tools developed in our research group for this purpose are described. These tools employ different techniques such as smoothing, edge length normalization and wavelets to create representations which describe 3D shape at multi-resolutions. Unlike the control of mesh resolution known as mesh simplification or mesh decimation in computer graphics, the tools we have developed are in the context of computer vision, in particular, object recognition and surface matching. Application examples are given for each of the representations generated by the above tools. Comparison of the representations is also discussed.

1. Introduction

With the development and improvement of new acquisition devices, range data have been much easier to obtain and more common in computer vision research. Range images are generally very large and contain highly complex non-polygonal structures. Therefore, processing range images at the finest level is expensive both in time and in storage. Multi-resolution representation is an approach to image processing in both the computer graphics and the computer vision fields. This methodology will greatly aid the interaction with complex range images.

Polygonal meshes are common representations in computer graphics because they are able to represent shapes of almost any topology and complexity. Therefore, polygonal meshes are direct and powerful ways to represent range data. However, the creation of multi-resolution polygonal meshes has a number of difficulties. The first difficulty derives from the complex nature of range images. The second difficulty lies in the trade-off between simplifying the mesh and preserving the original shape of the mesh. The third difficulty is the definition of multi-resolution. Although the meaning of multi-resolution is clear and definite in computer graphics, it may vary to some extent in different computer vision applications, as will be seen in the rest of the paper.

1. This work was supported in part by NSF Grant IRI-97-11853 and by a Research Grant from the Eastman Kodak Corporation.

Creating multi-resolution mesh has different applications, therefore requirements, in computer graphics and computer vision fields. In computer graphics the main motivation for control of mesh resolution is to reduce the number of faces describing an object while preserving the shape of the object. Some applications of this include: creation of mesh hierarchies for level-of-detail rendering, geometric compression for storage and transmission of polygonal mesh models and mesh editing, etc. In contrast, the control of mesh resolution in computer vision algorithms should consider the regularity of the mesh in addition to reducing the number of faces and preserving the shape of the object. Here, the meaning of regularity is to distribute the vertices of the mesh evenly over the surface of the object. Controlling regularity is very important because measuring local properties such as curvature and surface normal is very common in computer vision applications. Well defined locality is highly dependent on the regularity of the mesh.

The tools presented in this paper for generating multi-resolution meshes are developed in the context of computer vision. Although some techniques we used are similar to those in computer graphics, ours are tailored for the special needs of computer vision applications.

The first tool is used to generate multi-scale shape representation for 3D mesh in curvature space using smoothing technique. As opposed to the ordinary multi-resolution mesh, this multi-scale shape representation does not reduce the number of vertices on the mesh. Instead, it keeps the number of vertices on the mesh unchanged and gradually removes the details of the original shape using a smoothing technique. This representation is very similar to image pyramids in 2D. Three dimensional meshes can be reconstructed from the curvature-based multi-scale representations. Although the smoothing technique is applied on 3D meshes extensively, it is mainly used in range space. To our knowledge, it has not previously been used to create multi-scale shape representation in curvature space before.

The second tool is used to generate multi-resolution mesh using edge length normalization technique. Based on the mesh simplification algorithm used in computer graphics[10][13][15][16][17], our algorithm has a number of sig-

nificant modifications. In particular, our algorithm works on meshes representing curved and polygonal objects that contain holes and boundaries. Furthermore, our algorithm limits the maximum change in mesh shape and is able to adjust the spacing between vertices along ridges in the mesh. As a result, multi-resolution meshes with vertices evenly distributed on the surface are created.

The third tool is to create multi-resolution mesh using wavelets technique. Wavelets have been used extensively in signal processing for a long time. But the technique was introduced into computer graphics and computer vision fields less than a decade ago, especially for representing polygonal meshes. Early work of wavelets based multi-resolution analysis of polygonal meshes has been done by DeRose, Lounsbery and Eck[18][20]. The graphics applications for their multi-resolution meshes include compression, transmission and editing of mesh models, etc. However, computer vision applications such as model-based object recognition and classification can not benefit directly from the above representation because no special attention is paid to unify the domain meshes of objects which have the same topology, therefore making it very difficult to compare objects with different shapes. In our current research, we address this problem by starting with objects of spherical topology. The tool we developed is able to create multi-resolution representations for different objects over the same domain mesh.

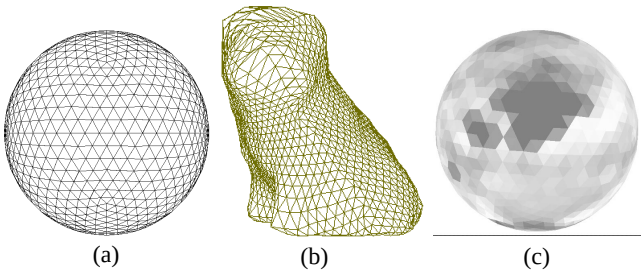
The paper is organized as follows: In Sections 2 to 4, three different tools for generating multi-resolution meshes are discussed. In Section 5, comparison of the representations generated by the three different tools is discussed. Conclusion and future work are discussed in Section 6.

2. Creating multi-scale shape representation in curvature space

The polygonal mesh models used in this section are built up using the deformable surface technique described in[1]. The object's surface is represented by a mesh in which each vertex is of degree 3. The number of vertices is determined by the frequency of tessellation. The vertices are distributed in a nearly uniform manner on the surface under some appropriate constraints. A local shape measure (referred to as local discrete curvature) is computed locally at each vertex of the mesh, as defined in[1]. A sphere of the same tessellation frequency as the deformable model with the local shape measure associated with each vertex is called the shape representation of the underlying object. Figure 1 shows a tessellated sphere, a deformable mesh model and its corresponding shape representation.

2.1. Smoothing

There has been extensive previous work on smoothing piece-wise linear shapes of arbitrary dimension and topology. The main disadvantage of mesh smoothing is the shrinkage of mesh that results from repeated averaging. Although several solutions to the shrinkage problem have been proposed[5][6][9], most of those approaches require careful design of the smoothing operator and, with the exception of [9], are hard to extend to 3D. In this section, mesh smoothing is studied from a new point of view. Instead of smoothing the mesh directly in 3D space, the shape representation of the mesh is smoothed in curvature space. By reconstructing a new mesh based on the smoothed shape representation, the mesh in 3D space is indirectly smoothed. This approach to mesh smoothing has two advantages in that it does not generate shrinkage on the mesh; and it does not involve direct computation on the mesh, which is an advantage when only the curvature based shape representation is needed.



(a) (b) (c)

Figure 1 (a) Tessellated sphere with frequency 7; (b) deformable model; (c) shape representation of (b).

A unit discrete sphere is represented as $S=\{V, E, C\}$, in which V is a list of vertices, E is a list of edges and C is a list of local discrete curvatures. The smoothed discrete curvature value c_i^{σ} at a node v_i is defined by:

$$c_i^{\sigma} = \alpha_{norm} \sum_{v_j \in W} e^{\frac{-1}{2}\frac{d_{ij}^2}{\sigma^2}} c(v_j) \qquad (1)$$

In (1), d_{ij} is the mesh distance, which is defined in [4], from node v_i to node v_j and W is the neighborhood of v_i, defined by $0 \le d_{ij} \le w$, where w is the size of the smoothing operator. $c(v_j)$ is the local discrete curvature at node v_j. c_i^{σ} is the smoothed local discrete curvature at v_i at scale σ. In practice, the size of the operator is selected as $w=4\sigma$ and the normalization factor α_{norm} is defined by:

$$\alpha_{norm} = 1 \Big/ \left(\sum_{v_j \in W} e^{\frac{-1}{2}\frac{d_{ij}^2}{\sigma^2}} \right) \qquad (2)$$

For a given vertex v_i, the vertices which are d mesh distances away from v_i can be found easily by traversing the edge list E.

Using the above definition of smoothing, the local discrete curvature distribution becomes a constant distribution as σ becomes large. In fact, this is true for all 3D objects which can be represented by the spherical discrete curvature func-

tion. This can be shown as follows: As indicated earlier, as σ increases, the neighborhood will reach its maximum size w_{max} due to the boundedness of the sphere. The corresponding neighborhood W is the entire sphere S. At that point, the smoothed value at node v_i is computed as:

$$c_i^{\sigma} = \left(\sum_{v_j \in S} e^{\frac{-1}{2}\frac{d_{ij}^2}{\sigma^2}} c(v_j) \right) \Big/ \left(\sum_{v_j \in S} e^{\frac{-1}{2}\frac{d_{ij}^2}{\sigma^2}} \right) \quad (3)$$

As σ increases further, the coefficients of the smoothing operator all converge to 1. Therefore, in the limit, c_i^{σ} converges to the average value $\bar{c}$ of discrete curvature over the entire sphere:

$$c_i^{\sigma} = \left(\sum_{v_j \in S} c(v_j) \right) / N = \bar{c} \quad (4)$$

A consequence of (4) is that, as σ increases, all objects converge to a shape of identical discrete curvature everywhere, namely a sphere. This behavior is consistent with the intuitive behavior of smoothing: as more and more details are eliminated from an object surface, that object should converge to the simplest object, a sphere. Therefore, by continuously smoothing the shape representation of an object, a sequence of shape representations at multi scales can be created. Here the meaning of scale is the value of the smoothing scale, σ. For example, Figure 2 shows such a sequence of the object shown in Figure 1(b). This sequence was obtained by smoothing the shape representation and by using the inverse mapping algorithm of [8] to construct the corresponding 3D shapes.

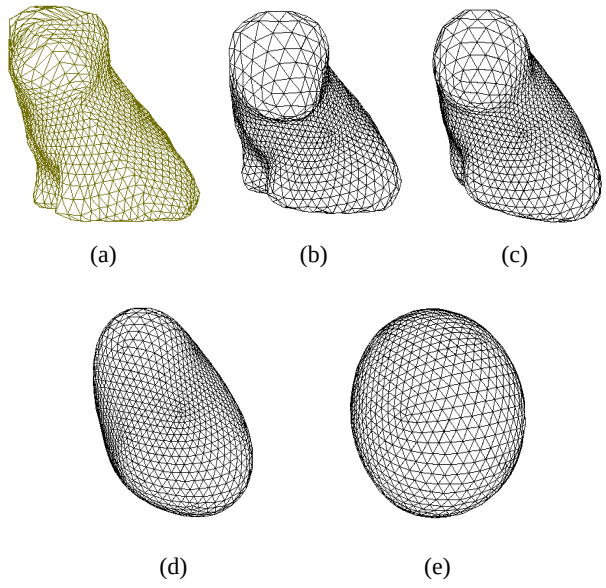


Figure 2 Reconstructed meshes of the object shown in Figure 1(b) from the smoothed shape representations. (a) σ=0; (b) σ=0.5, (c) σ=2.0; (d) σ=4.0; (e) σ=9.0.

In practice, different objects modeled by discrete meshes of the same frequency may converge to slightly different values of $\bar{c}$ due to round-off errors, and to the fact that the sampling is not perfectly uniform. However, the differences are small enough that they do not affect applications such as object classification. Example of this can be found in [4].

2.2. Application: object Classification

One application of the multi-scale shape representation is object classification. The motivation for object classification is to speed up the process of object recognition. Ideally, we would like to be able to identify candidate objects (a certain object class) at a coarse scale, *i.e.*, based on their overall shapes, and then to identify the actual object present in the scene at a finer scale, *i.e.*, based on the detailed descriptions of their shapes. The multi-scale shape representation meets this requirement very well.

The basic idea of the classification algorithm is to compute the multi-scale shape representations of the objects in the library and to compute a similarity measure which integrates the shape information at all scales. Such a shape similarity measure is called *scaled shape similarity metric*, which has been proposed in [4]. In the following, shape similarity metric is first defined in (5) and (6), and then scaled shape similarity metric is defined in (7).

$$d_p(S_A, S_B, R) = \left(\int \left| k(S_A) - k_R(S_B) \right|^p ds \right)^{\frac{1}{p}} \quad (5)$$

$$D_p(A, B) = min_R d_p(S_A, S_B, R) \quad (6)$$

$$d_{\Omega}(A, B) = \sum_{\sigma \in \Omega} w_{\sigma} D_p^{\sigma}(A, B) \quad (7)$$

In (5), S_A and S_B are the mesh representations of object A and B, respectively. $k(S_A)$ and $k(S_B)$ are the shape distribution functions for S_A and S_B, respectively. We denote by $k_R(S_B)$ the shape distribution function of S_B after rotation by R. The L_p distance $d_p(S_A, S_B, R)$ between A and B for a given spherical rotation R is the sum of local discrete curvature differences over the sphere. In (6), the shape similarity measure $D_p(A, B)$ between A and B is the minimum of d_p over all possible rotations R. In (7), Ω is the set of scales σ; w_{σ} is the weight at scale σ; $D_p^{\sigma}(A,B)$ is shape similarity metric defined in (6) with σ indicating the comparison is done at scale σ.

The classification algorithm is illustrated by using a simple object library containing five synthetic objects: cube, tetrahedron, octahedron, octahedron with one indentation (referred to as octahedron1) and octahedron with two indentations (referred to as octahedron2). Figure 3 shows the mesh models and the corresponding shape representations. Due to space limitation, the multi-scale shape representations are not shown here for each object.

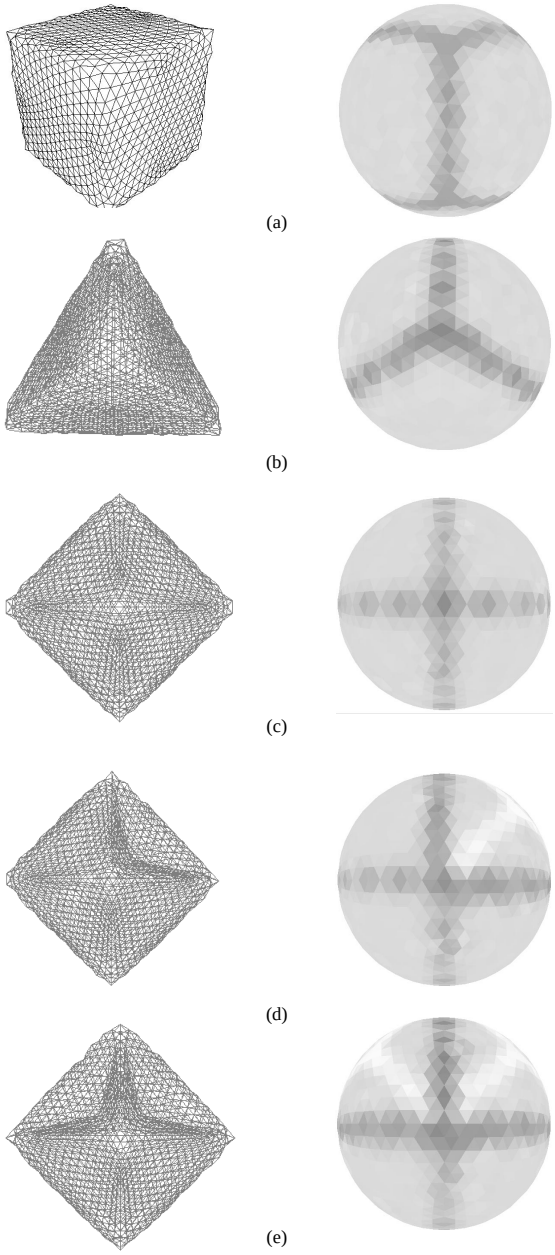


Figure 3 Meshes and their corresponding shape representations. (a) cube; (b) tetrahedron; (c) octahedron; (d) octahedron1; (e) octahedron2.

The octahedron is selected as the reference object. The scaled shape similarity metric for each of the four objects is shown in Table 1. As a result, octahedron1 and octahedron2 are classified into the same class as octahedrons.

Objects	cube	tetra	octa1	octa2
$d_{\Omega}(x, octa)$	6.13	9.15	1.31	2.27

Table 1 Scaled shape similarity values of four objects.

3. Creating multi-resolution mesh by normalizing edge lengths

In computer graphics the control of mesh resolution is termed mesh simplification [10][13][15][16][17] or mesh decimation. The main motivation for mesh simplification is to reduce the number of faces describing an object, while preserving the shape of the object, so that the object can be rendered as fast as possible. Heckbert and Garland [11] have written a comprehensive overview of mesh simplification algorithms for computer graphics.

In contrast, an algorithm that controls the resolution of a mesh for computer vision applications should, in addition to reducing the number of faces while preserving the shape of the object, distribute the vertices of the mesh evenly over the surface of the object. If a surface is uniformly sampled by the vertices of a mesh, then local surface properties can be computed using mesh connectivity. Examples of situations where well defined locality is needed in 3D computer vision are point based registration of surface meshes, establishment of control points for computation of spline surfaces on a surface mesh, clustering of points for segmentation of range images, and calculation of surface normal.

Based on mesh simplification techniques from computer graphics, we have developed an algorithm that normalizes the lengths of edges between vertices in a mesh while preserving object shape [14]. Our algorithm is widely applicable to real data sets because it can handle surface meshes with arbitrary shape (free-form and polyhedral), arbitrary topology, and boundaries. The increased generality of our control of mesh resolution algorithm comes with a cost; a parameterization of the surface is not available. Consequently, formal mathematical methods (e.g., wavelets) cannot be used to create or analyze the hierarchies created.

Our algorithm takes in a mesh with an uneven spacing of vertices over its surface and a desired resolution for the mesh. The output is a mesh of similar shape with vertices evenly spaced at the desired resolution. By creating meshes with decreasing resolutions, a hierarchy of multi-resolution meshes can be generated.

3.1. Algorithm

For an arbitrarily shaped object, it is generally impossible to make the distance between vertices exactly the same for all vertices in the mesh while still adequately describing the shape of the object. Therefore, we quantify the spacing between vertices using local and global statistics on the histogram of lengths of edges in the mesh. An example of an edge length histogram is given in Figure 4. We define mesh *resolution* to be the median of the lengths of all of the edges in the mesh, and we define the *edge length spread* to be the upper quartile of edge lengths minus the lower quartile of edge lengths (*i.e.*, the half width) in the edge length histogram. Given these definitions, the goal of our algorithm is to

adjust the resolution of the original mesh to a desired resolution, while minimizing the edge length spread of the histogram. We call this process *edge length normalization*.

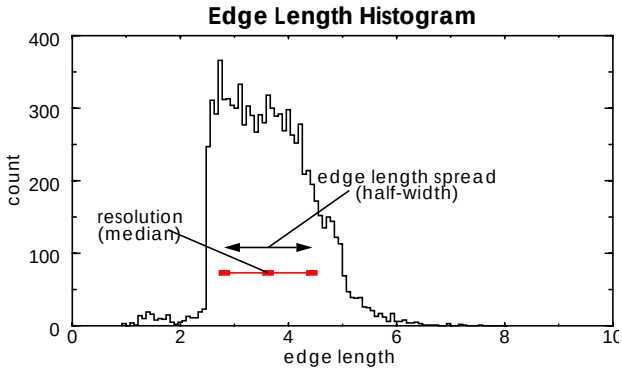


Figure 4 Edge length histogram. The resolution of a mesh is the median of the histogram; the edge length spread is the half-width of the histogram.

An additional constraint on edge length normalization is that the original shape of the object must be preserved; we assume that the original shape of the object is given by the mesh input into the algorithm. In our algorithm, two operations are iteratively applied to edges in the mesh to obtain the mesh of the desired resolution: edge-split is used to remove long edges, while edge-collapse is used to remove short edges. During edge-split, an edge is broken at its midpoint into two edges; the edge-split operation does not change the shape of the mesh. During edge-collapse, an edge is reduced to a point, so the local shape of the mesh is modified. In our algorithm, the position of the point resulting from edge-collapse is chosen to preserve the shape of the object, but some change in shape is unavoidable.The change in shape of the mesh is minimized at each iteration by ordering the edges in the mesh for operation based on the change in shape that results from application of each edge operation (*shape change measure*). Using this ordering, the edges that change the shape of the mesh the least are applied first, thus minimizing the change in shape at each iteration. Shape change measure *C(e)* of an edge *e* is defined as the maximum distance between the mesh before collapsing *e*, $M_b(e)$, and after collapsing *e*, $M_a(e)$

$$C(e) = d(M_b(e), M_a(e)) \tag{8}$$

Using the distance between meshes is an accurate way of measuring the shape change produced by an edge operation, and in contrast to less accurate shape change measures[16], it allows for simplification along ridges in surface data.

To strike a balance between preserving shape and normalizing the lengths of the edges in the mesh, the order of application of edge operations is also made a function of the length l of the edges. More specifically, an *edge length weight W(e)* is computed for each edge *e*

$$W(e) = \mathbf{exp}(-(l - L_0)/{L_D}^2) \tag{9}$$

Where L_0 is the desired resolution and L_D is the expected spread in edge lengths. The order *O(e)* in which an edge is operated on is decided by the product of *C(e)* and *W(e)*. With this ordering, short edges that do not greatly change the shape of the mesh upon removal will be collapsed first.

We prevent the shape of the mesh from changing too much by placing a bound on the maximum allowable change in mesh shape. Each time an edge operation is applied, the edge's shape change measure is added to the *accumulated shape change* (the accumulation of the change in shape that the edge has undergone in previous iterations) of all of the edges in the neighborhood of the edge. If the accumulated shape change of an edge is made greater than the maximum allowable change in shape, the edge is removed from consideration. This ensures that, over the entire surface of the mesh, the change in shape remains below a user-defined bound. Furthermore, if the length of an edge is within some user defined bound of the desired mesh resolution, the edge will not be decimated. The limit on the maximum allowable shape change and the length of edges being within some bounds of the desired resolution will eventually prevent operation on any edges in the mesh. This constitutes the stopping criterion for the algorithm. Figure 5 shows two mesh hierarchies created with our edge length normalization algorithm.

3.2. Application: surface registration

Figure 6 shows the use of multi-resolution meshes in surface registration when the transformation between views is completely unknown. First, two views of an object at a coarse resolution are registered using spin-image matching [14].The coarse version of the meshes is sufficient for determining the rough transformation between views. Next the alignment between views, given by matching the coarse meshes, is refined using an iterative closest point algorithm and finer resolution meshes. Having meshes at two resolutions allows the problem to be broken into two stages, making the overall registration algorithm more efficient.

4. Creating wavelets based multi-resolution meshes over the same domain mesh

The basic idea behind wavelets-based representation is to represent a complicated function as a composition of simpler low-resolution function along with a group of perturbation functions at increasing levels of detail. The original function can be recovered without losing any information at the highest resolution. The wavelets representations in 1D and 2D have been well studied. However, the extension of wavelets to 3D polygonal mesh is not straight forward. DeRose and Lounsbery have done the fundamental work for

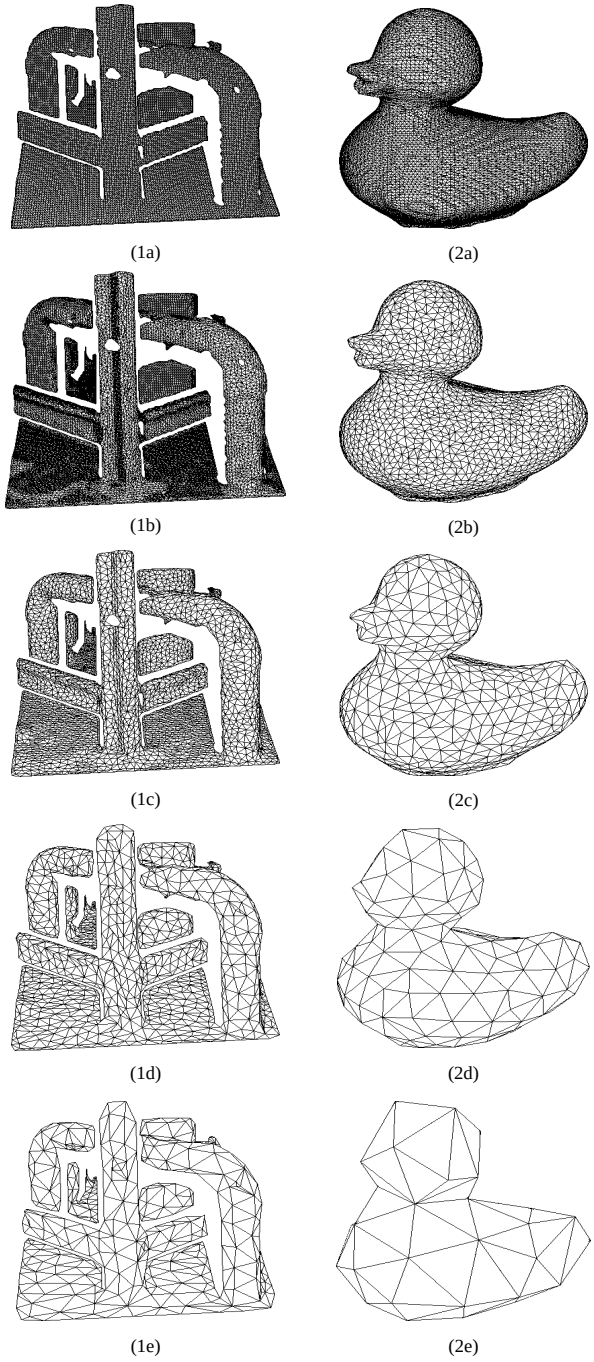


Figure 5 Two mesh hierarchies generated by normalizing edge lengths. The number of triangles in each image is: (1a) 21187; (1b) 34112; (1c) 6601; (1d) 1747; (1e) 615; (2a) 27246; (2b) 6182; (2c) 733; (2d) 182; (2e) 92.

extending wavelets representation to 3D meshes of arbitrary topologies[20].

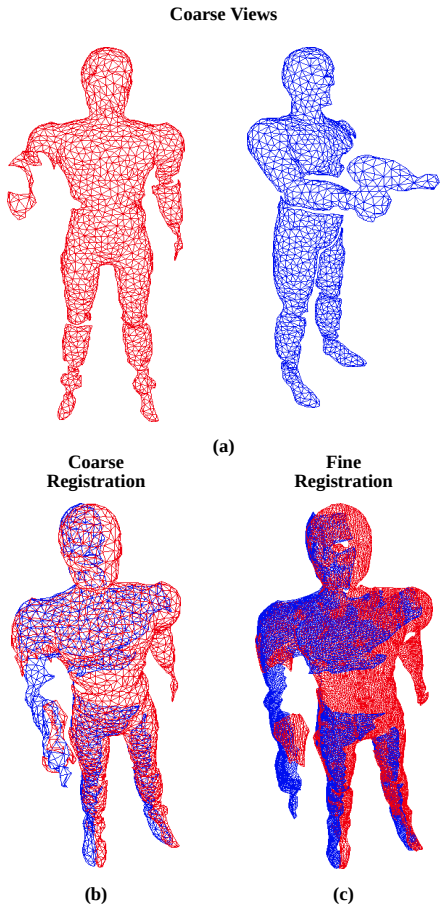


Figure 6 Application of multi-resolution meshes to surface registration. First, two coarse resolution views (a) of a toy robot are registered (b). The computed transformation is then refined using finer resolution meshes (c).

4.1. Reconstructing an object with spherical topology over the same domain mesh

As discussed in [18] and [20], the object mesh must have subdivision connectivity in order to create wavelets representations. Shortly speaking, for a triangular mesh, the simplest form of subdivision is to iteratively split each edge of the mesh so that each triangle is split into four smaller triangles (Figure 7). This requirement is so strong that very few polygonal meshes can meet it. In order for the wavelets construction to be possible, an algorithm has been proposed in [18] to first find a domain mesh for the input object and to then construct a parametrization of the input mesh over the domain mesh. Because the domain mesh has subdivision connectivity through subdivision, the input mesh will also have this property by mapping the vertices on the domain mesh to the input mesh through the parametrization constructed before. The details of the algorithm can be found in [18].

Although the above algorithm is applicable to meshes with arbitrary topology, it will result in a different domain mesh

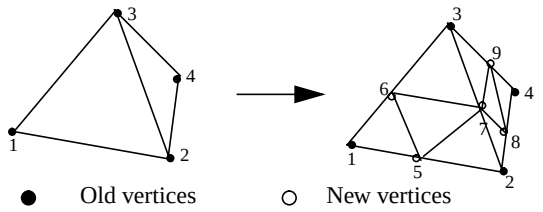


Figure 7 Polyhedron subdivision of tetrahedron.

for every different object. This is not a problem at all in computer graphics; however, it is one in the context of computer vision. Because our primary motivation for developing such multi-resolution representation is to provide models for object recognition and classification, we would like to have one unique domain mesh for objects with the same topology. In our current research, we are focusing on generating wavelets representations for objects having spherical topology. This is because shape similarity and shape classification are very difficult problems themselves and they will become even more challenging if object topologies are considered. In order to solve the problem step by step, we decompose the problem by considering objects with simple topologies such as spherical topology.

Because the objects we are dealing with have spherical topology, we select tetrahedron to be the domain mesh for all objects. Therefore, our algorithm does not need to look for a domain mesh for the input object. What our algorithm does is to partition the mesh surface into four correctly connected regions so that each region can be mapped onto its corresponding face of the domain tetrahedron.

Before presenting our region partition algorithm, we need to make some assumptions about the object mesh. These assumptions will help the region partition algorithm and they can be easily met by polygonal meshes.

First of all, we assume that the object mesh has a closed surface. If the mesh surface is not closed, we can do some preprocessing to make it close. The vertices which are added in the preprocessing step are labeled “virtual vertices”. The second assumption is that the variance of the areas of the triangles on the mesh surface are not very large. In fact, for dense range data sets, the triangles are generally of very small size. If the variance is large, we can use our second tool discussed in Section 3. to regularize the triangles.

There are two criteria for partitioning the mesh surface. The first one is that every region must be homeomorphic to a disk. This is to guarantee the mapping between each region and a corresponding face of the domain tetrahedron. The second one is that the four regions are of roughly equal size. This is to preserve the original shape of the mesh during the resampling process.

The region partition algorithm is as follows:

- Select an arbitrary starting triangle on the mesh surface to be the seed of region 0
- Grow region 0 until its area exceeds a threshold (one fourth of the total area of the mesh surface)
- Connect the triangles that are not labeled and are neighbors of the boundary triangles of region 0
- Select three triangles as seeds for regions 1 to 3 among the above boundary triangles so that the three triangles are roughly equally apart
- Grow regions 1 to 3 simultaneously until all the triangles on the mesh surface are labeled

Criterion 1 is checked when growing every region. If it is violated, the algorithm restarts by selecting a new seed for region 0. Because the mesh surface is assumed to be closed, all the triangles on the mesh surface are guaranteed to be labeled. Because the sizes of the triangles are assumed to be roughly equal, the variance of the four areas will not be large.

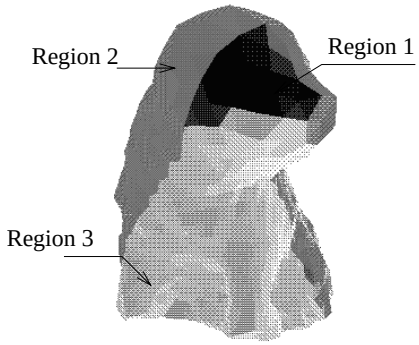


Figure 8 A mesh surface partitioned into four regions (the fourth region is in the back).

Figure 8 shows a mesh partitioned into four regions. Three regions can been seen in the figure. The fourth one is in the back. Figure 9 shows the histogram of the areas of the triangles on the mesh surface of the object shown in Figure 8. Table 2 lists the statistics for the four regions. Figure 10 shows the wavelets representation of the object.

4.2. Regularizing the reconstructed mesh

It can be noticed that on the reconstructed mesh surface in Figure 10, the vertices are not uniformly distributed. In fact, this is caused by the region partition algorithm. Although the partitioned four areas are of roughly equal size, they may have poor aspect ratios when mapped onto the triangular faces of the domain tetrahedron. Because each edge of the domain tetrahedron is subdivided at the same frequency, shorter edges will result in denser distribution of resampling vertices than longer edges. Although perfect uniform distribution of vertices is not feasible, the mesh should have some form of regularity. This is important for applications such as shape comparison.

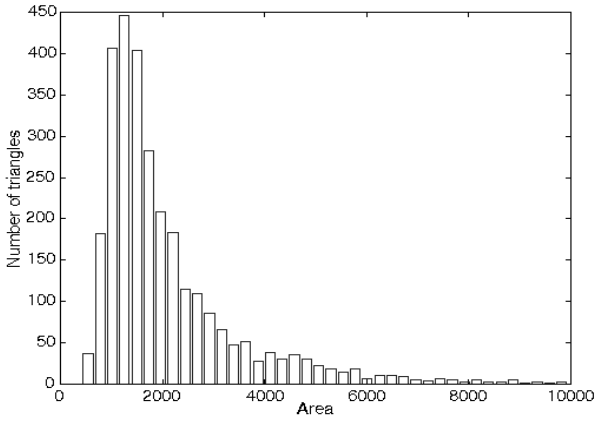
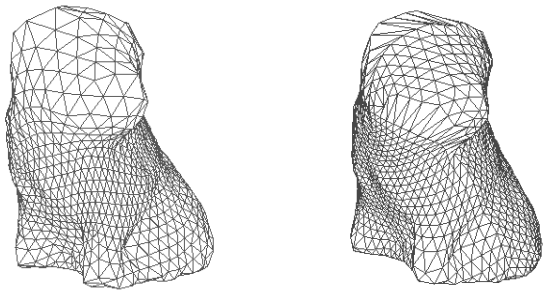


Figure 9 Histogram of the areas of the triangles on the object in Figure 8.

Region	Area	Number of triangles	Area percentage
0	1.55e+06	444	25.31%
1	1.48e+06	861	24.12%
2	1.56e+06	789	25.46%
3	1.54e+06	846	25.10%

Table 2 Statistics for the partitioned regions.



(a) (b)

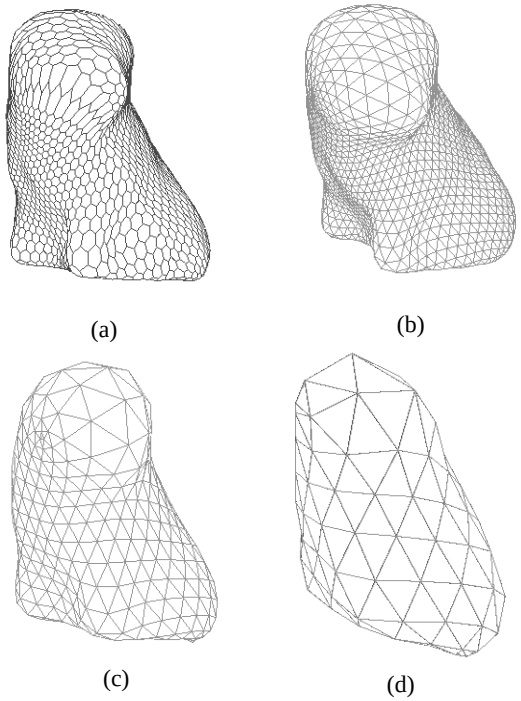
Figure 10 Wavelets-based multi-resolution representations of the object shown in Figure 8. (a) original mesh with 2940 triangles; (b) reconstructed mesh with 4096 triangles.

The regularization is done on the dual mesh of the reconstructed mesh. The reason for regularizing the dual mesh is that the regularity constraint, which will be explained shortly, can be applied easily to the dual mesh. If the dual mesh has some form of regularity, the reconstructed mesh will also have that regularity. Figure 11(a) shows the dual mesh of the reconstructed mesh in Figure 10(b). On the dual mesh, each vertex has exactly three neighboring vertices. The regularity constraint requires the projection of each vertex on the triangle formed by its three neighbors to be the center of that triangle as shown in Figure 11. The regularization of the dual mesh is thus an iterative process to adjust the positions of the vertices to satisfy the regularity constraint. The update rule of each vertex is defined as follows:

$$\vec{v}_j^{ideal} = \vec{v}_o + \vec{N} \bullet (\vec{v}_j - \vec{v}_o)\vec{N} \quad (10)$$

$$\vec{v}_j(t+1) = \vec{v}_j(t) + \alpha(\vec{v}_j(t) - \vec{v}_j^{ideal}(t)) + \beta(\vec{v}_j(t) - \vec{v}_j(t-1)) \quad (11)$$

In (10) and (11), $\vec{v}_j$, $\vec{v}_o$, $\vec{v}_j^{ideal}$ and $\vec{N}$ are defined in Figure 11. α and β are within the range of 0 and 1. In practice, we select α=0.05 and β=0.9. The regularization process finishes when the regularity constraint is satisfied. This can be checked for each vertex by computing the difference between its projection on the triangle and the center of the triangle. Figure 11(b), (c) and (d) show the regularized reconstructed meshes at different resolutions.



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

Figure 11 (a) The dual mesh of the reconstructed mesh shown in Figure 10(b); (b), (c), (d) normalized reconstructed meshes with 4096, 1024, 256 triangles respectively.

4.3. Constructing wavelets-based multi-resolution representation

In the domain of real line, multi-resolution representation is based on a sequence of nested linear spaces defined by dilating and translating a single function. In domains of objects with arbitrary topologies, however, it is difficult to generalize the concepts of translating and dilating a function. Therefore, DeRose and Lounsbery get around this problem by extracting only the main element - nested linear spaces of the multi-resolution representation in the real-line domain. As discussed in [20], by recursively subdividing

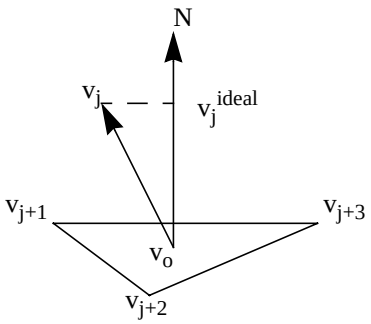


Figure 12 Illustration of the regularity constraint.

the domain mesh M^0, a sequence of nested linear function spaces (the functions in these spaces are termed scaling functions $\phi_j(x)$[20]*)* can be defined. The inner product for any two scaling functions is also defined. For any two continuous nested linear spaces $V_j(M^0)$ and $V_{j+1}(M^0)$, wavelets functions, $\psi_j(x)$, are functions that span the space $V_{\perp}^{j}(M^0)$, which is the orthogonal complement of $V_j(M^0)$[20]. Therefore, $S(x)$, the parametrization of the domain mesh M^0, can be written [20] in the form of

$$\begin{aligned} S(x) &= \sum_i v_i^j \bullet \phi_i^j(x) \\ &= \sum_i v_i^0 \bullet \phi_i^0(x) + \sum_j \sum_i w_i^j \bullet \psi_i^j(x) \end{aligned} \tag{12}$$

in which v_i^j denotes the coefficients of the scaling functions ϕ_i^j at the jth resolution, and $w^j{}_i$ denotes the coefficients of the wavelet functions ψ_i^j at the jth resolution. It can been seen from above that the meaning of resolution is the dimension of the linear space $V_j(M^0)$. For detailed mathematical background on 3-D wavelets functions, see [20].

4.4. Application of wavelets-based multi-resolution representation

As mentioned in Section 4.1., the main applications for wavelets-based multi-resolution representation in our current research are object recognition and classification. Having unique domain mesh for all the objects of spherical topology makes it easier to compare different shapes. From the modeling point of view, this representation can model very complex shapes compared with other modeling techniques such as deformable surface. Experiments of object recognition and classification using this representation are being conducted.

5. Comparison of the multi-scale representations

The edge length normalization-based representation is the most powerful one among the three in the sense of dealing with objects of arbitrary topologies. Although wavelets--based representation in general is not limited by object topologies, it is constrained in this case by the requirement of having unified domain mesh. Under wavelets-based representation, the original shape can be fully recovered by going up the resolution heirarchy. In other words, there is no information lost between any two continuous levels of resolution. On the contrary, no knowledge is available about how the shape changes when going from high resolution to low resolution under the edge length normalization-based representation.

Both the edge length normalization-based representation and the wavelets-based representation are in range space. In contrast, the smoothing-based representation is in curvature space. Of course smoothing can be done in range space directly. But special attention must be taken to avoid shrinkage of the mesh. Combining the smoothing-based representation with any one of the two representations in range space will give us a full description of how a certain shape changes at different resolutions in both curvature space and range space. This information is very important for object recognition and classification.

The three tools for creating multi-resolution representations are closely connected. For example, the edge length normalization algorithm can be used as a preprocessing step to regularize a mesh before generating wavelets-based representation. If curvature information can be computed from the wavelets-based representation, then the smoothing technique can be applied to generate multi-scale shape representations in curvature space. In summary, the three tools provide powerful representations for analyzing polygonal meshes.

6. Conclusion

Three different tools for creating multi-resolution polygonal meshes have been presented in the paper. The key techniques used are smoothing, edge length normalization and wavelets. The smoothing-based representations are multi-scale representations in curvature space, while the edge length normalization-based representation and the wavelets based representation are in range space. These three tools provide powerful multi-resolution representations for analyzing polygonal meshes in computer vision applications.

Future work will focus on using the multi-resolution representations created by the above tools in object recognition and classification.

Acknowledgments

The authors would like to thank Qinghui Zhou for providing the subdivision program, Marie Elm for proofreading the paper, and Andrew Mor for help on editing the paper.

References

[1] M. Hebert, K. Ikeuchi and H. Delingette, “A spherical representation for recognition of free-form surfaces”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(7): 681-689, July 1995.

[2] K. Higuchi, M. Hebert and K. Ikeuchi, “Building 3D models from unregistered range images”, CVGIP-Image Understanding. Vol. 57. No. 4. July 1995.

[3] H. Shum, M. Hebert and K. Ikeuchi, “On 3D shape similarity”, Proc. CVPR’96, pp. 526-531. June 1996.

[4] D. Zhang, M. Hebert, “Multi-scale classification of 3D objects”, Proc. CVPR’97, pp.864-869, July, 1997.

[5] J. Oliensis, “Local reproducible smoothing without shrinkage”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(3): pp.307-312, 1993.

[6] M. Wheeler and K. Ikeuchi, “Iterative smoothed residuals: a low-pass filter for smoothing with controlled shrinkage”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(3): pp. 334-337, March, 1996.

[7] H. Shum, “Modeling from reality: representation and integration”, Ph.D. thesis, Carnegie Mellon University, July 1996.

[8] H. Shum, M. Hebert and K. Ikeuchi, “On 3D shape synthesis”, In Object Representation in Computer Vision II. LNCS 1144. Springer-Verlag. April 1996.

[9] G. Taubin, “Curve and surface smoothing without shrinkage”, Proc. ICCV’95, pp. 852-857, June 1995.

[10]A. Guéziec, “Surface simplification with variable tolerance”, Proc. Medical Robotics and Computer Assisted Surgery(MRCAS’95), pp. 132-139, 1995.

[11]P. Heckbert and M. Garland, “Survey of polygonal surface simplification algorithms”, School of Computer Science, Carnegie Mellon University, Technical Report, CMU-CS-97-194, 1997.

[12]H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle. “Mesh optimization”, Proc. Computer Graphics 1993(SIGGRAPH ‘93), pp. 19-26, 1993.

[13]H. Hoppe, “Progressive meshes”, Proc. Computer Graphics 1996 (SIGGRAPH ‘96), pp. 99-108, 1996.

[14]A. Johnson. and M. Hebert, “Control of polygonal mesh resolution for 3D computer vision”, The Robotics Institute, Carnegie Mellon University, Technical Report, CMU-RI-TR-96-20, 1997.

[15]A. Kalvin and R. Taylor, “Superfaces: polyhedral approximation with bounded error”, SPIE Medical Imaging, vol. 2164, pp. 2-13, 1994.

[16]W. Schroeder, J. Zarge and W. Lorensen, “Decimation of triangular meshes”, Proc. Computer Graphics 1992 (SIGGRAPH ‘92), pp. 65-70, 1992.

[17]G. Turk, “Re-tiling polygonal surfaces”, Proc. Computer Graphics 1992 (SIGGRAPH ‘92), pp. 55-64, 1992.

[18]Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle, “Multi-resolution Analysis of Arbitrary Meshes”, University of Washington, Technical Report, 95-01-02, January 1995.

[19]Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin, “Wavelets for computer graphics: A primer”, part 1. IEEE Computer Graphics and Applications, 15(3):76-84, May 1995.

[20]Tony D. DeRose, Michael Lounsbery, Joe Warren, “Multi-resolution analysis for surfaces of arbitrary topological type”, Deptment of Computer Science and Engineering, University of Washington, Technical Report, 93-10-05, October 29, 1993.