The Visual Computer

Sampling-sensitive multiresolution hierarchy for irregular meshes

Oscar Kin-Chung Au,
Chiew-Lan Tai

Department of Computer Science, Hong Kong University of Science & Technology, Kowloon, Hong Kong, China
E-mail: {oscarau, taicl} @cs.ust.hk

Published online: 4 August 2004
© Springer-Verlag 2004

Previous approaches of constructing multiresolution hierarchy for irregular meshes investigated how to overcome the connectivity and topology constraints during the decomposition, but did not consider the effects of sampling information on editing and signal processing operations. We propose a sampling-sensitive downsampling strategy and design a decomposition framework that produces a hierarchy of meshes with decreasing maximum sampling rates and increasingly regular vertex support sizes. The resulting mesh hierarchy has good quality triangles and enables more stable editing. The detail vectors better approximate the frequency spectrum of the mesh, thus making signal filtering more accurate.

Key words: Meshes – Multiresolution – Irregular connectivity – Sampling sensitive

1 Introduction

Traditionally, multiresolution representation and modeling were proposed in the context of parametric surfaces [4, 5] and subdivision surfaces [22] because these schemes provide well-structured refinement rules. With the help of wavelets techniques, the design of decomposition and reconstruction operations for these surfaces is made mathematically elegant.

For meshes, traditional multiresolution framework [22, 23] assumes that the input mesh has subdivision connectivity and uniform parameterization. However, many real world meshes have irregular connectivity and vertex distribution. Finding the scaling and wavelet functions for each resolution and determining the stencils are difficult for these meshes. One way to overcome these difficulties is by remeshing [3, 7, 18]. However, remeshing is a resampling process, thus it may cause artifacts and need many more vertices and triangles to recover the model shape. Also, if each triangle or vertex has some associated attributes (e.g., color or normal), finding the correct attribute value for the samples in the remeshed output is difficult.

Some researchers have built multiresolution representations directly for irregular meshes without remeshing. Kobbelt et al. [16] used progressive meshes (PM) [10] with the discrete umbrella smoothing to build a multiresolution representation for editing. Guskov et al. [8] employed a nonuniform smoothing operator and the progressive mesh to build a hierarchy for signal processing. The offsets (detail information) of the vertices that are removed by the simplification are interpreted as the frequency spectrum of the input mesh – the vertices removed earlier have their corresponding offsets taken as higher frequencies. Thus the derived frequency information is according to the order of vertex removal. After building the hierarchy, signal filtering is performed on the frequencies. Both these approaches use shape-sensitive simplification algorithms that focus on shape preservation in the decomposition process.

We argue that the computed offsets in a hierarchy built using a shape-sensitive simplification do not correspond to the actual frequency information. This leads to unexpected filtering output. We also contend that using shape-sensitive simplification in the decomposition process leads to irregular triangle areas and edge lengths in the lower resolution meshes, which is undesirable for low resolution edit-

The Visual Computer (2004) 20:479–493
Digital Object Identifier (DOI) 10.1007/s00371-004-0253-3

ing that results in large deformation of the original shape.

To achieve more accurate signal filtering and to facilitate editing in low resolution, we believe that the downsampling in the construction of mesh hierarchy should be based on the vertex distribution, rather than the geometry of the input mesh. In this paper, we propose a sampling-sensitive downsampling algorithm and use it to design a decomposition framework for irregular meshes to overcome these problems. We note that the idea of sampling-sensitive simplification is not new, but it has so far been used only for shape preservation and level-of-details rendering [6, 10]. In addition to adopting a new downsampling algorithm, we use the umbrella smoothing operator in our framework in order to produce a hierarchy of meshes with smoother geometry and more regular edge lengths and triangle shapes in the lower resolutions.

The remainder of this paper is organized as follows. We first state some assumptions and definitions in Sect. 2. In Sect. 3, we illustrate the weaknesses of the previous approaches with two examples and present an overview of the proposed solution. Section 4 presents the related previous work. Section 5 describes our sampling-sensitive downsampling algorithm and the decomposition framework. Some experimental data and comparison with alternative hierarchies are given in Sect. 6. Section 7 shows the advantages of our hierarchy in applications such as signal filtering, editing and remeshing. Finally the conclusions are offered in Sect. 8.

2 Assumptions and definitions

We assume that the model to be decomposed is a triangular mesh, which is a topological manifold. It can have any connectivity, topology and sampling distribution, and may contain boundary loops. The geometric realization of the mesh is considered as the union of points, which consists of the vertex positions and the linear interpolation of vertex positions such that the corresponding vertices form edges and triangles.

The *support size* of a vertex is defined as its one-ring triangles area. This is because only the positions of the one-ring triangles are modified when the vertex is moved. The *sampling rate* at a vertex is defined as the inverse of its support size.

3 Problems and proposed solution

We focus on two problems of previous decomposition frameworks. (1) Shape-sensitive simplification does not remove vertices in an order such that the extracted detail information corresponds to the correct frequency information. (2) Shape-sensitive simplification results in unequal vertex support sizes and unequal edge lengths in the low resolution meshes.

Figure 1 illustrates the above two problems with a 2D example. The top row shows the results when a shape-sensitive simplification is used in the decomposition process. The low-resolution version is obtained by first removing those points with zero offset distance since these points cause the least geometric change. These offsets are then taken as the high frequency information. Consequently, filtering (smoothing and enhancement) these offsets does not change the shape, even though the original structure clearly contains some high frequency details in the inner loop that should have been filtered. The rightmost column shows the result of repositioning the white points in the low-resolution version (and then adding the details back). It is observed that the modification of the vertices in the outer loop affects a larger region than the vertices in the inner loop. This is undesirable since users expect that editing different sample points in a low resolution mesh would affect regions of similar size in the finest mesh. This clearly cannot be achieved in a highly unequal sampling setting.

The top row of Fig. 2 illustrates the second problem in a 3D mesh setting. A low-resolution mesh with vertices of highly unequal support sizes is produced by a shape-sensitive simplification. The model is bent in low resolution and then the details are added back. It contains apparent artifacts because the bending is in the minimum curvature direction. Shape-sensitive simplification tends to produce edges that are longer in the minimal curvature direction because fewer vertices are needed to represent the shape under the same error tolerance. Since it is impossible to predict the editing direction intended by the user, meshes with edges of similar lengths in all directions are deemed to have a greater degree of freedom for editing without producing undesirable artifacts.

Broadly speaking, in order to solve the above two problems, we need to achieve the following three goals:

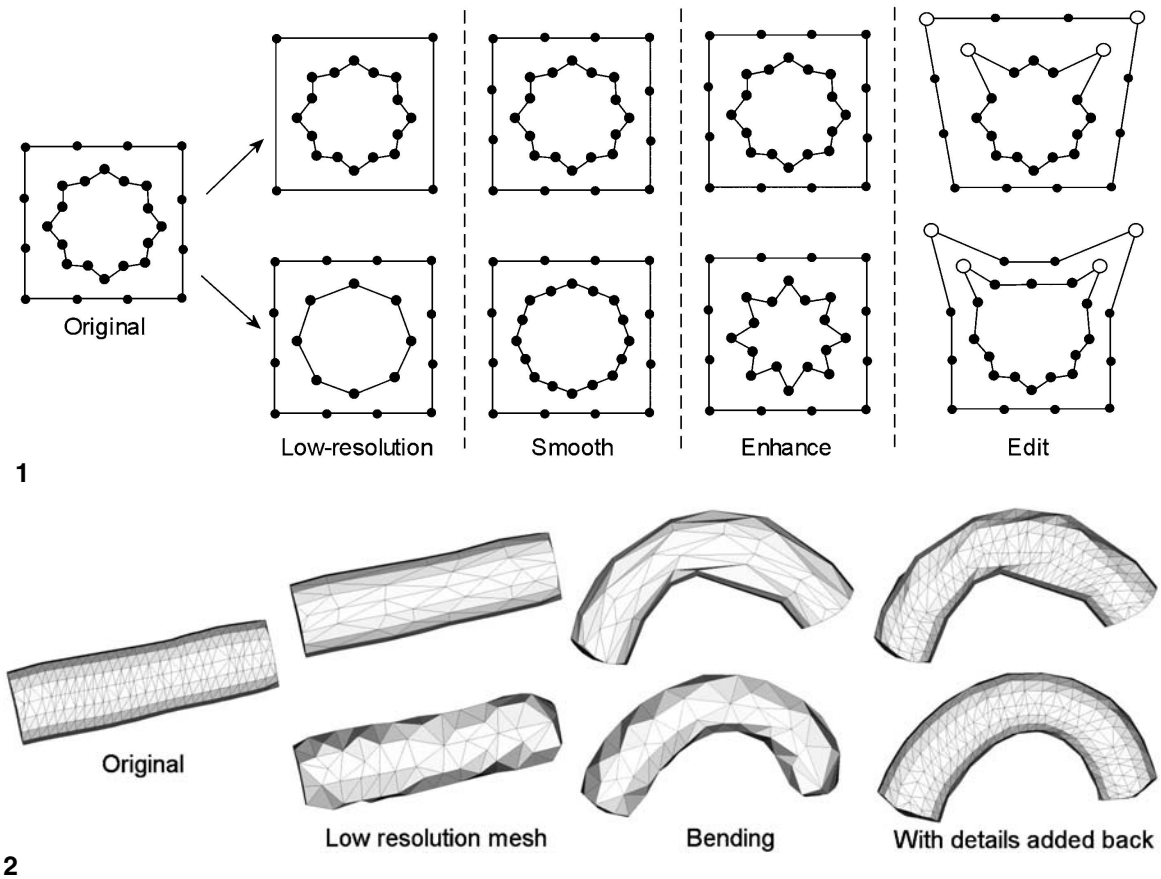


Fig. 1. Shape-sensitive downsampling (*top row*) and sampling-sensitive downsampling (*bottom row*) produce different low-resolution structures (*middle left column*) and thus have different effects on subsequent signal filtering (*middle and middle-right columns*) and multiresolution editing (far right column). Sampling-sensitive downsampling is able to remove vertices in an order corresponding to frequency, and thus produces the desired filtering results. It also produces a low-resolution version that contains vertices with a more regular support size, which is desirable for editing

Fig. 2. Shape-sensitive simplification produces longer edges in the minimum curvature direction (*top row*), while the sampling-sensitive method produces edges of similar length in all directions (*bottom row*), which gives better result when editing in low resolution (*center right* and *far right columns*)

(1) The downsampling algorithm used in the decomposition framework should depend on the sampling setting, and not the geometry of the mesh.
(2) The vertices of a lower resolution mesh should have similar support sizes. The edges along any direction should be of similar lengths, i.e., edge length is isotropic over the mesh.
(3) A lower-resolution mesh should have smoother geometry. This is required because high frequency information has to be removed from the higher resolution mesh.

To achieve these three goals, we propose an L2-norm downsampling algorithm, which is sampling-sensitive in the sense that it removes the vertices in the regions with the highest sampling rate first. It produces a hierarchy of meshes with decreasing maximum sampling rates. Since this downsampling algorithm depends heavily on the support size of vertices, the triangle sizes of the resulting downsampled meshes are more equal compared with the shape-sensitive downsampling methods. We also apply umbrella smoothing to the mesh during decomposition,

which improves the quality of the meshes in the hierarchy such that they have more similar edge lengths and smoother geometry.

The results of our decomposition framework are shown in the bottom rows of Figs. 1 and 2. In Fig. 1, the offsets of the vertex positions are stored in the normal fields of the low resolution. It is observed that the low and high frequency details are now correctly filtered. Additionally, the editing result is more intuitive due to the more uniform sampling distribution in the low-resolution version. The 3D example in Fig. 2 shows that the low-resolution mesh produced by our downsampling algorithm has edges of similar lengths in all directions, leading to fewer artifacts after a large deformation.

4 Previous work

Signal processing on meshes was first introduced by Taubin [20, 21] in the context of surface fairing. Eigenvectors of Laplacian operators are viewed as frequencies on meshes. Their filtering framework can be used to smooth and denoise meshes.

Our framework builds upon ideas from two previous approaches for constructing multiresolution hierarchies for irregular meshes. In [16], Kobbelt et al. proposed a multiresolution framework to smooth a region chosen for editing. The approach can be seen as single-level decomposition for editing. The simplification algorithm mainly considers the fairness of the resulting mesh and aims at minimizing the error with respect to the original mesh [15]. Guskov et al. were the first to generalize signal processing operations to irregular meshes [8]. They defined upsampling, downsampling and filtering operations and built mesh pyramids for irregular meshes. Editing and filtering operations are preformed efficiently on the mesh hierarchy just as the traditional (semi-) uniform subdivision surfaces. However, they employed quadric error metric (QEM) [6], which also measures error with respect to original mesh, as a priority criterion for mesh decimation. Both approaches employ shape-sensitive simplification, which can lead to problems in signal filtering and editing as mentioned before.

Our decomposition framework closely resembles Guskov's [8], which is shown in Fig. 3. The multiresolution framework decomposes an irregular mesh $M = M^n$ into a sequence of meshes M^j, $j = n, n-1, \ldots, 0$, with decreasing resolution and sets of detail

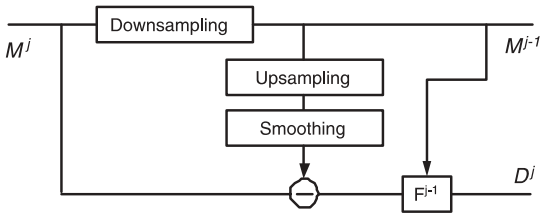
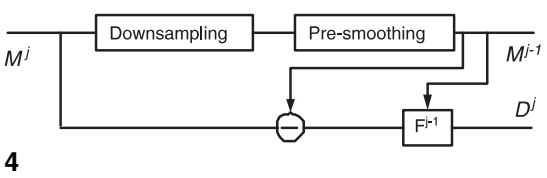


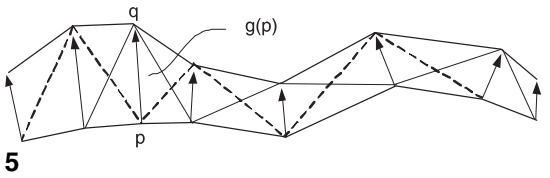
Fig. 3. Guskov's decomposition framework

vectors D^j, $j = n, n-1, \ldots, 0$. The detail vectors in D^j record the difference between the meshes M^j and M^{j-1}. The coarsest resolution mesh M^0 is the base mesh of the hierarchy, and the original mesh can be rebuilt from the base mesh M^0 and the detail vectors $D^1, D^2, \ldots, D^n$. In addition to the use of a new downsampling algorithm, our decomposition framework differs from that of Guskov et al. in two aspects. First, their framework is an interpolating one – the positions of the vertices in the lower resolutions are the same as those in the original mesh. We believe this restriction is not necessary for multiresolution modifications; it is more important that the lower resolution meshes have smoother geometry and sampling information. Second, due to the use of shape-preserving decimation, their framework does not require a pre-smoothing step when constructing a lower resolution mesh. Upsampling and smoothing steps are applied to predict a corresponding point for detail vector computation. Since they store detail vectors as 3D vectors capturing the difference between topologically corresponding points, the original parameter information has to be preserved in the lower meshes. Thus, they proposed a nonuniform operator for parameterization preserving smoothing, which requires storing the smoothing weights for reconstruction. In our framework, we employ the normal field detail encoding scheme in [17], which makes only the geometry, not the parameterization, of the lower meshes relevant. Therefore nonuniform smoothing and storing of weights for reconstruction become unnecessary.

Our mesh representation has some commonalities with the normal mesh [9]. Both representations comprise a base domain mesh and some sets of detail vectors, produce a sequence of multiresolution meshes, and represent the details in the normal field



4



5

Fig. 4. Our decomposition framework
Fig. 5. Surface function of a mesh

of the next lower resolution mesh. However, the construction of a normal mesh is from coarse to fine, whereas our method computes the details in a fine-to-coarse fashion. In addition, our setting is aimed at editing and filtering operations, while the normal mesh is designed for remeshing and compression purposes [12], thus their representation is optimized by removing the parameter information of the vertices as much as possible.

5 Sampling-sensitive framework

Figure 4 shows our sampling-sensitive decomposition framework. The decomposition of a mesh M^j into the lower resolution M^{j-1} and the set of detail vectors D^j involves three procedures: *downsampling, pre-smoothing*, and *detail encoding* (the lower processing line in Fig. 4). The downsampling procedure uses our selection criteria, which has the effect of removing a number of vertices from the high sampling regions of M^j. Next, a smoothing step is applied to remove the high-frequency content for constructing the smoother and coarser mesh M^{j-1}. Then we apply the umbrella operator, which does not only smooth the geometry, but also regulates the triangle shapes and edge lengths of the intermediate mesh. Finally, the detail vectors in D^j are computed for all the vertex positions in M^j with respect to the local frames F^{j-1} of M^{j-1}. To find a corresponding base point, we employ the Phong normal field [17]. We note that since most of the vertices in M^j are unchanged, we only need to encode the detail vectors for vertices that are removed in M^j and those correspond to the smoothed vertices in M^{j-1}.

5.1 L^2-norm error metric

To design a suitable decomposition framework for function analysis, we need to express meshes as continuous surface functions defined on a base domain. However, meshes with different topologies clearly do not have a common base domain. Hence we put all meshes of the same connectivity and topology into the same class of functions, and compute the difference of two meshes in terms of the norm difference. We use this mesh difference metric in our downsampling algorithm.

5.1.1 Surface function

We consider a mesh M as a piecewise linear function defined on a base domain mesh B of the same connectivity and topology. There is a continuous mapping between B and M that maps the corresponding vertices of the two surfaces and maps the interior of the corresponding triangles using barycentric coordinates. Given a base surface B, a mesh M can be represented as a surface function $g : B \to \mathrm{R}^3$ such that, for a point $p \in B$, the corresponding point on M is $q = p + g(p)$ (Fig. 5).

5.1.2 Mesh difference

Let M_1 and M_2 be two meshes of the same connectivity, and let g_1 and g_2 be their respective surface functions defined on a common base domain B. We define the inner product of M_1 and M_2 as the integration of the dot product of the surface functions over the base surface

$$\langle M_1, M_2 \rangle = \int_{p \in B} g_1(p) \cdot g_2(p)\, \mathrm{d}p.$$

The L^2 norm of a mesh M with surface function $g(p)$ is then

$$\|M\|_2 = \left(\int_{p \in B} |g(p)|^2 \, \mathrm{d}p \right)^{1/2}$$

and the difference between M_1 and M_2 in the L^2 norm sense becomes

$$\|M_1 - M_2\|_2 = \left(\int_{p \in B} |g_1(p) - g_2(p)|^2 \, \mathrm{d}p \right)^{1/2} = \left(\int_{p \in B} |p_1 - p_2|^2 \, \mathrm{d}p \right)^{1/2}$$

where $p_1 = p + g_1(p)$ and $p_2 = p + g_2(p)$ are the mapped points on the meshes M_1 and M_2, respectively.

We discretize the integration over each triangle T_i in B with its area as the measure, then the mesh difference becomes

$$\|M_1 - M_2\|_2 = \left(\sum_{T_i} A(T_i) \, IP(T_i) \right)^{1/2} \quad (1)$$

where $A(T)$ denotes the area of the triangle T in B and

$$IP(T) = \int_{p \in T} |p_1 - p_2|^2 \, \mathrm{d}p = \int_0^1 \int_0^{1-\beta} (\alpha \Delta p_a + \beta \Delta p_b + (1 - \alpha - \beta) \Delta p_c)^2 \, \mathrm{d}\alpha \, \mathrm{d}\beta \quad (2)$$

where p_a, p_b, p_c are the vertices forming the triangle T, Δp_a, Δp_b, Δp_c are the differences in positions of the corresponding vertices in M_1 and M_2, and $(\alpha, \beta, 1-\alpha-\beta)$ are the barycentric coordinates of an interior point.

However, since the mesh can have any connectivity and topologically structure, the base function is unknown. So we have to relax the definition of $\|M_1 - M_2\|_2$ by using the area of triangles in M_1 instead of the area of triangles in B, as the measure. We denote this augmented mesh difference as $\|M_1, M_2\|$, and note that $\|M_1, M_2\|$ and $\|M_2, M_1\|$ are different in general. This derivation, though informal, produces good results in our implementation. This mesh difference metric considers the distances between all the corresponding points on the mesh, not just the vertices, and it produces different results compared with other distance functions such as the sum of the squared vertex displacements or the total Hausdorff distance. The sum of squared vertex displacements cannot distinguish the disparity in shape caused by different sampling rates of vertices. It is possible for two pairs of meshes to have the same mesh difference and yet the sampling setting for one pair of meshes to be totally different from the other pair. On the other hand, the total Hausdorff distance can distinguish the disparity in shape, but does not consider the difference in vertex positions. For example, moving the vertices in a planar region of a mesh does not affect its total Hausdorff distance with another mesh. Our difference metric considers both the shape and the vertex positions of the meshes, and can distinguish some cases that cannot be distinguished by the other measures.

5.2 Downsampling

We employ the half-edge collapse as the basic operation to remove vertices. For the purpose of deriving the mesh difference metric, we assume that a half-edge collapse produces a mesh of the same connectivity: collapsing a half-edge $e = \{s, t\}$ in a mesh M_1 produces a mesh M_2 in which the position of vertex s is set to be the same as vertex t, producing two degenerate triangles. Thus, only the one-ring neighbor faces of s, denoted $F_1(s)$, are changed by the edge collapse. By setting only one of the differences Δp_a, Δp_b, Δp_c as nonzero in Eq. 2, the mesh difference in Eq. 1 becomes

$$\|M_1, M_2\| = \left(\frac{1}{12} \sum_{T_i \in F_1(s)} A(T_i) \, \|p_s - p_t\|^2 \right)^{1/2}.$$

where $A(T_i)$ denotes the area of triangle T_i, and $\|p_s - p_t\|$ is the distance between the vertices s and t. We define the cost of collapsing an edge $e = \{s, t\}$ as $\mathrm{cost}(s, t) = \|M_1, M_2\|$. The edge with the minimum cost is chosen to be collapsed next. The summation of triangle areas in the cost function implies that vertices with small one-ring triangle areas are removed first. The edge-length term in the equation decides which of the surrounding vertices should vertex s be collapsed, given that vertex s is one of the endpoints of the edge collapse; it favors the removal of small triangles. We refer to this downsam-

pling algorithm as L^2-*norm downsampling* in this paper.

If the mesh has boundaries, we do not allow an edge $\{s, t\}$ to be collapsed if s is a boundary vertex and t is not. Collapses that result in closing a hole or changing the topology are also disallowed. An infinite cost is assigned to such forbidden edge collapses.

There are several possible criteria for determining the boundary between levels of resolution. The simplest one is to let M^j be the coarsest mesh that has an upper bound ε^j on the cost. Another way is to remove a fixed percentage of vertices at each level. For signal processing, a good choice is to define the threshold according to the one-ring triangles area of the collapsed vertex, and to quadruple the threshold at the next level. This is analogous to doubling the scale of the scaling functions in wavelet transform. In our implementation, we use the cost of edge collapse as the separator: the mesh is downsampled until the cost of collapse reaches a threshold, and the threshold is doubled at the next level. To begin, the threshold is initialized to be the cost when a quarter of the original vertices are removed.

The cost function of our downsampling algorithm is very simple compared to many simplification algorithms. There are two other main differences. First, most simplification algorithms (e.g., QEM, PM) select the element (vertex, edge or face) to be removed by comparing the resulting mesh with the input mesh. Our cost function compares the current mesh with the previous mesh; this is analogous to traditional signal analysis where the analysis of the next coarser resolution depends only on the current resolution. Second, we can view shape-sensitive simplification algorithms as removing samples in the order of increasing magnitude of error, whereas our downsampling algorithm removes samples in the order of their support size. Note that in our setting, the cost of an edge collapse is nonzero even if it does not change the geometry (e.g., in the planar regions). Figure 6 shows a close-up of the skull model at different levels of resolution (after applying the pre-smoothing described in the next subsection). Notice the increasingly smoother sampling rates in the different levels.

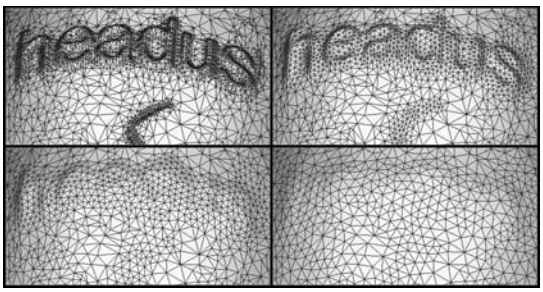


Fig. 6. Different levels of resolution of the skull model: the 1st level (original model) (*top left*), the 3rd (*top right*), 5th (*bottom left*) and 7th level (*bottom right*)

5.3 Pre-smoothing

To extract the high frequency details from the low frequency global features, after downsampling we smooth the mesh by re-positioning the surrounding vertices of the removed vertices to some weighted average of their one-ring neighbors.

Smoothing of nonhierarchical meshes usually requires the parameter information to be retained as much as possible [8]. However, in a multiresolution setting, we believe that it is desirable to produce intermediate meshes with smooth parameter information (i.e., to regulate the triangle areas and edge lengths) as they give better results in filtering and editing. Moreover, since we encode the detail vectors of the current mesh using the normal field of the next coarser mesh as in [17], which is independent of the parameterization of the coarser levels (see details in next subsection), parameterization preservation in intermediate meshes is not a necessary criterion.

To smooth the geometry and the parameter information of a current mesh, we employ the simplest and most efficient umbrella smoothing operator. Since the regions without edge collapses do not have details removed at this level, we only need to smooth the one-ring neighbors of the vertices removed at this level. Note that the smoothed vertices may not have been involved in any edge collapse; in fact, there is no specific correspondence between the smoothed vertices and the removed vertices in a level. So we use two data structures to store the information for reconstruction, one for the split records of the removed vertices and one for the original positions of the smoothed vertices. Figure 7 shows the different levels of resolution of the bunny model produced by our decomposition framework. Even though the original model contains obtuse and long triangles,

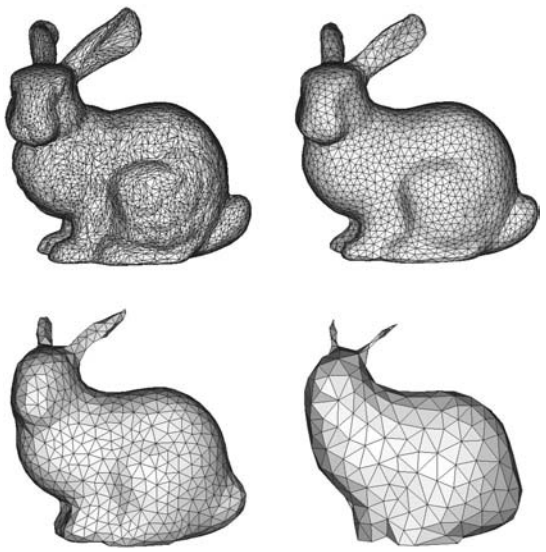


Fig. 7. Decomposition of the bunny model using our downsampling algorithm.

each of the lower resolution meshes produced by our framework is of almost uniform triangle size and sampling rate. In very low resolution meshes, some features may be collapsed or degenerated (e.g., Fig. 7, bottom right). This is mainly caused by the shrinkage effect of the smoothing. The umbrella $\lambda|\mu$ smoothing algorithm can be applied to eliminate this effect [21].

If an intermediate mesh of the decomposition is needed as an input to some applications that require the parameter information to be preserved, then a nonuniform smoothing operator can be employed [2, 8]. However, our experiments show that the umbrella smoothing operator gives more stable mesh hierarchy in terms of vertex position dependency and triangle quality (see Sect. 6 for details) than parameterization-preserving smoothing (curvature flow smoothing is used for comparison).

5.4 Detail encoding

In order to reconstruct the fine features of the mesh after signal processing or editing an intermediate mesh, the original positions of the removed vertices and smoothed vertices must be encoded with respect to the local frames at a lower resolution mesh. In general, we want to encode the vertex positions in M^j based only on the geometry and connectivity information of M^{j-1}. For greater stability of the reconstruction, the lengths of the detail vectors should be as short as possible. The optimal choice is to find the base point in M^{j-1} such that the vertex to be encoded is in the normal direction of this base point [16]. Following [17], we denote a detail vector by (d_1, d_2, h), where d_1 and d_2 are the barycentric coordinates of the base point on M^{j-1} and h is the offset distance in the normal direction. This encoding scheme means that the detail vectors are only dependent on the geometry of the coarser mesh M^{j-1}, i.e., only d_1 and d_2, neither the base point nor the normal offset, are affected by the parameterization of M^{j-1}.

We use the Phong normal field proposed by Kobbelt [17] to find the base point. This normal field is continuous for a closed mesh and covers all directions. So any point in space can be encoded with positive barycentric coordinates. Since we know in advance the order of edge collapses and the level in which a vertex is smoothed, we can very closely estimate in which region the corresponding base point is located. For each half-edge collapse $e = \{s, t\}$, to encode the removed vertex s, we search the base point within the one-ring triangles of t in the coarser mesh M^{j-1}. To encode a vertex in M^j corresponding to a smoothed one in M^{j-1}, we simply search for its base point within the one-ring triangles of the smoothed vertex in M^{j-1}. If no base point with positive coordinates is found in the triangles of the first searching region, we continue searching in the surrounding triangles in a breath-first-search fashion.

If the search for a base point is near a boundary, it is possible that no base point with positive coordinates exists (because the normal field ends at the boundary). In this case, we simply find a base point with the minimum sum of absolute barycentric coordinates within the first-guess region, and do not enter a recursive search.

The encoding scheme we use requires three floating-point numbers (for the two barycentric coordinates and the normal distance) and one integer (for the face index) per vertex. Comparatively, the weight-storing method in [8] needs three floats for detail encoding plus the storage space for the weight information. Storing the weight information explicitly (six floats per vertex in average) or storing the face areas and edge lengths of the mesh hierarchy both require more space than one integer per encoded vertex. Thus, we conclude that the weight-storing method needs more space than the barycentric-based method.

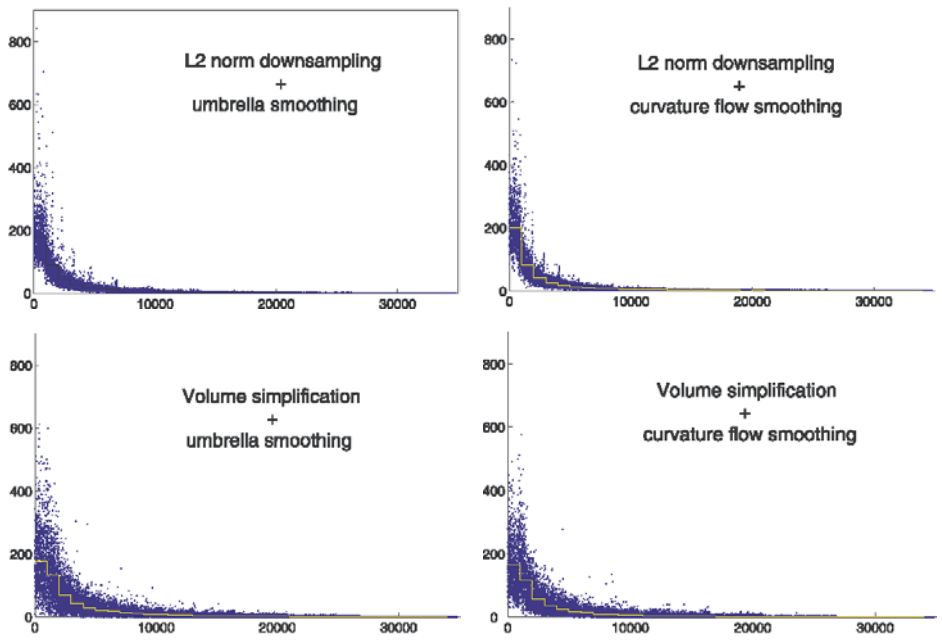


Fig. 8. Dependence of vertex positions for the bunny model using different decomposition and smoothing methods

6 Experimental data and comparisons with alternative methods

Table 1 shows the execution times of the decomposition and reconstruction of the input models using our framework. All data is generated on an Intel Pentium 4 2 GHz machine with 256 Mb of memory.

We compare our decomposition framework, which uses L^2-norm downsampling and umbrella smoothing, with frameworks that use alternative downsampling or smoothing methods. More specifically, we investigate the four possible combinations involving the L^2-norm downsampling versus volume-based simplification [19], and umbrella smoothing versus curvature flow smoothing [2]. Kim [13] proved that volume simplification is essentially a distance simplification (QEM) weighted by the area of triangles adjacent to the collapsed edge, thus our experimental results for volume-based simplification also apply to shape-preserving simplification, and we use these two terms interchangeably.

Table 1. Execution times for decomposition and reconstruction (in second)

	Bunny	Horse	Venus	Rocker arm
# Vertices	34 834	48 485	67 173	20 088
# Base vertices	about 1000			
Decomposition	17 s	24 s	41 s	13 s
Reconstruction	6 s	11 s	13 s	3 s

Except the vertices in the base mesh, the positions of all other vertices are encoded in the local frames of some mesh in the hierarchy. Thus a set of vertices at the finest level is moved when a vertex is edited in a lower resolution. This means that the position of a vertex depends on the positions of other vertices in the coarser resolutions. A mesh hierarchy is stable and useful for editing if the vertex position dependence has a small variance over all vertices. For example, in subdivision surfaces with uniform subdivision rules, the number of vertices that depend on a given vertex is fixed and is decided by the number of subdivision steps. Figure 8 plots the number of vertices whose positions depend on the ith vertex of the bunny model. The vertices are indexed according to their order of removal, with those removed earlier having higher indices. The X-axis is

the vertex indices and Y-axis is the number of vertices depending on ith vertex. The yellow line indicates the moving average of period 1000. The graphs show that vertex position dependency of decomposition using volume simplification coupled with curvature flow smoothing (similar to Guskov's framework) has a higher variance than our decomposition framework. This is because shape-sensitive simplification leads to higher variation in the one-ring triangles area (size of support) in the lower resolution meshes. Table 2 lists the variances, which are computed using the moving averages of the different downsampling and smoothing combinations for some input models. Tables 3 and 4 list the variances of the edge lengths and triangle areas, respectively, in their base meshes. Among these four combinations, the experimental data show that the L^2-norm downsampling coupled with the umbrella smoothing gives the lowest variance in vertex position dependence, as well as lowest variances in edge lengths and face areas in the base meshes. These data confirm that our framework produces a hierarchy that is more suitable for editing.

Table 2. Variance of vertex position dependence of input models

	Bunny	Horse	Venus	Rocker arm
# Vertices	34 834	48 485	67 173	20 088
# Base vertices		about 1000		
L2Norm + Umb	269.5	243.9	370.9	229.0
L2Norm + CF	266.2	278.3	485.9	212.4
Volume + Umb	561.2	507.6	1303.5	353.0
Volume + CF	393.6	372.8	961.3	248.1

Umb: umbrella operator CF: curvature flow smoothing

Table 3. Variance of edge lengths in base meshes of different input models

	Bunny	Horse	Venus	Rocker arm
L2Norm + Umb	0.034	0.048	0.029	0.026
L2Norm + CF	0.068	0.082	0.065	0.066
Volume + Umb	0.152	0.203	0.097	0.243
Volume + CF	0.255	0.273	0.197	0.359

Table 4. Variance of triangle areas in base meshes of different input models

	Bunny	Horse	Venus	Rocker arm
L2Norm + Umb	0.061	0.107	0.051	0.045
L2Norm + CF	0.122	0.168	0.111	0.114
Volume + Umb	0.195	0.179	0.144	0.328
Volume + CF	0.420	0.354	0.357	0.576

7 Applications

In this section, we illustrate the advantages of using our decomposition hierarchy in signal filtering and multiresolution editing. All the editing and filtering examples demonstrated in this section are done interactively within a few seconds. We also show that our downsampling algorithm can be used for isotropic remeshing.

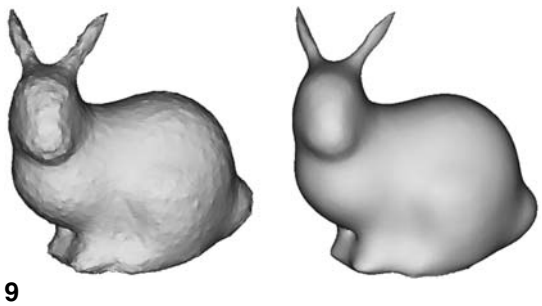
7.1 Signal filtering

Signal filtering should only affect the positions of the vertices in the normal direction, and not change the parameterization. This means that for each detail vector in D^j, only the normal component is multiplied by a scaling constant $\lambda^j \geq 0$. The choice of λ^j at different levels depends on the desired filtering operation: values close to one retain the details in the level, values less than one smooth out the details, and values greater than one enhance the details.

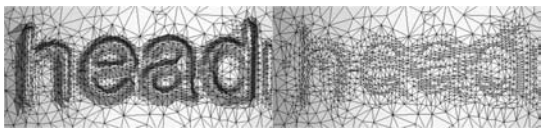
Since our decomposition framework is different from the previous frameworks, in particular, there is no smoothing step in the reconstruction pipeline, we need a slightly different filtering algorithm. More specifically, if we multiply the normal components of the details by zero at all the levels greater than j, instead of getting a smooth mesh, we would get a faceted surface (Fig. 9, left). This is because the added vertices at higher resolutions all fall on the chosen base surface. To achieve the expected smooth result, we have to apply a post-smoothing step to relocate the base positions.

Post-smoothing is performed as follows. Let the base point in the lower resolution mesh computed from the barycentric coordinates in a detail vector be b, and the position of the vertex indicated by the detail vector without filtering be p. We upsample the lower resolution mesh and initialize each new vertex to coincide with the base point b, and then apply the curvature flow smoothing with a small update factor (we use 0.3 in our implementation) on the point b to generate b'. The difference between p and b' is then multiplied by the factor λ^j to obtain the new vertex position p':

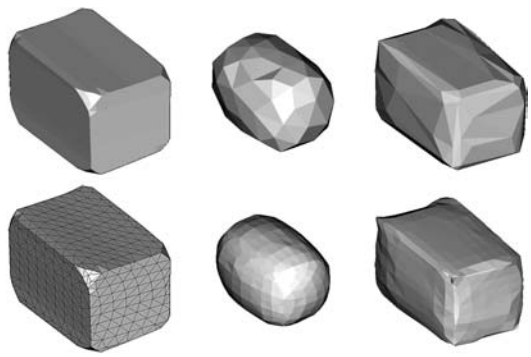
$$p' = b' + \lambda^j (p - b').$$



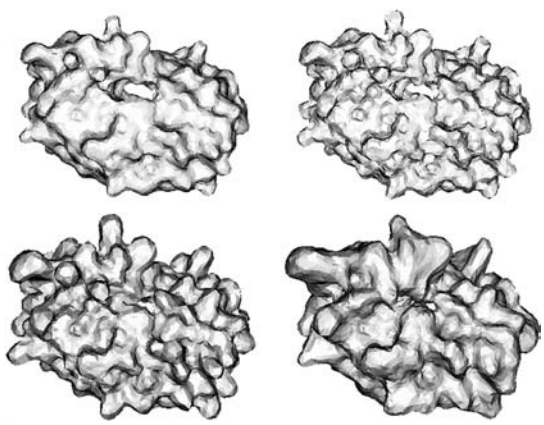
9



10



11



12

Fig. 9. Only filtering the normal components of detail vectors causes artifacts (*left*). Adding a post-smoothing operation improves the result and yet does not affect the sharp features (*right*)

Fig. 10. The filtered mesh (*right*) has the same connectivity and similar parameter information as the input one (*left*)

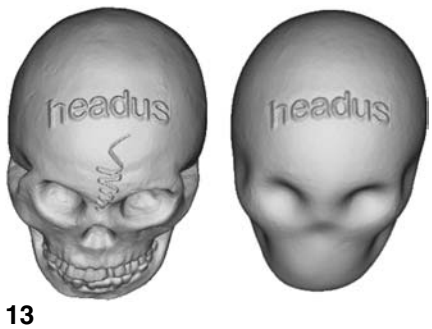
Fig. 11. Different downsampling strategies produce different effects on smoothing. On the left is the input model (# vertices = 521). The middle column is the base mesh (# vertices = 72) (*top*) and the filtered model (*bottom*) using our L^2-norm downsampling. The right column is the base (#vertices = 76) (*top*) and filtered (*bottom*) models using volume simplification

Fig. 12. Enhancement of different levels of details. The *upper left* is the original model, and the others are obtained by enhancing different sets of details, from high to low frequency

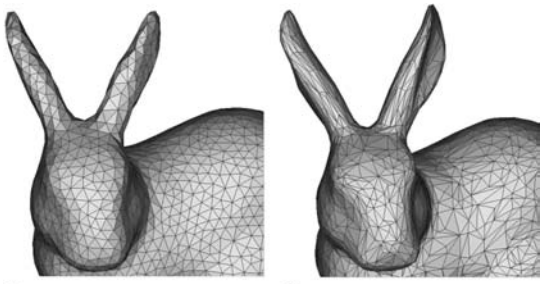
The bunny on the right in Fig. 9 is produced by adding the post-smoothing operation in the filtering process. Figure 10 shows that filtered meshes retain the connectivity and parameter information of the input mesh.

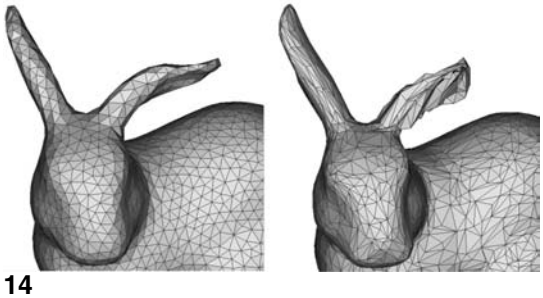
To illustrate the advantage of our framework in signal filtering, Fig. 11 compares the results of using the hierarchies produced by our L^2-norm downsampling and by volume-based simplification. Volume-based simplification removes the vertices in the planar regions first and the high frequency information remains in the base mesh. With the L^2-norm downsampling, high frequency details are removed first and a desired smoothed model is produced. Figure 12 shows the results of enhancing the details in a molecule example using our decomposition framework. It is observed that our framework can distinguish the frequency spectrum accurately, and thus smooth or enhance the relevant levels of geometric detail.

Selective filtering can be performed on different parts of a mesh (Fig. 13). The user specifies a region

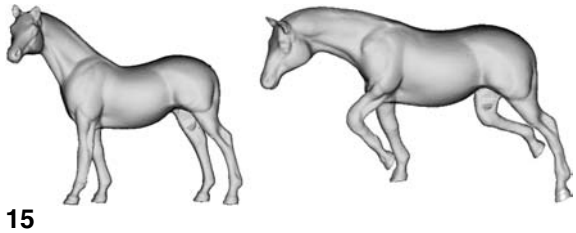


13





14



15

Fig. 13. Selective filtering of the skull model

Fig. 14. Bending the bunny's ear in low resolution. The top row shows the low-resolution meshes produced using our L2-norm downsampling (# vertices = 2310) (left) and volume simplification (# vertices = 2487) (*right*). The bottom row shows the corresponding edited models. The mesh generated by volume simplification contains artifacts after editing because the edges are longer in the minimum curvature direction

Fig. 15. Editing a horse model. The original model (*left*) and the modified model (*right*)

in a mesh at a specific level and only the detail vectors associated with the vertices within the region are filtered. For the vertices removed at the finer levels, we can use the barycentric coordinates in the detail vectors to determine whether or not such a vertex is in the selected region.

7.2 Multiresolution editing

As illustrated in Fig. 2, smooth parameterization in intermediate meshes provides more stable and uniform representational ability for editing. Figure 14 gives another example, comparing the results of bending the bunny's ear in two low resolution meshes, one created using our downsampling algorithm and the other using volume simplification.

Figure 15 shows the result of editing a horse model in low resolution. To perform this editing, we implemented a simple interface based on the transformation operation of the sketching interface in Teddy [11]. The user selects a region of the mesh to be modified, and sketches a reference stroke and a target stroke. The selected vertices are first encoded in the local frames of the reference stroke. The system then moves these vertices such that their new positions are offset from the target stroke, with the same offset distances as those encoded with respect to the reference stroke. The movements are only parallel to the screen and are restricted to be horizontal. Clearly, artifacts can appear at the boundary of the selected region due to a nonsmooth vertex displacement field there. To get around this problem, we extend the displacement field to the whole surface by

Fig. 16. Uniform remeshing of the Venus model. Input model (# vertices = 33k) (*left*), resampled model (*middle*) and final output (# vertices = 20k) (*right*)

setting the initial displacement of the vertices outside the selected region to be zero vectors, then diffuse the displacement field before moving the vertices, using a local smoothing filter.

7.3 Isotropic remeshing

We can also design a simple algorithm for isotropic remeshing using our downsampling algorithm. This application illustrates the ability of our algorithm to generate uniform sampling. In previous work by Pierre et al. [1], the resampling uses an error diffusion technique in either the spatial domain or the parameterization domain, and the meshing and optimization are done in the parameterization space. Our remeshing algorithm is performed totally in the spatial domain and no global parameterization is needed. The whole procedure involves four steps: resampling, connectivity improvement, parameterization smoothing and re-projection.

First, the user decides on the final sampling rate for the resultant mesh. The system computes the average triangle area and edge length under this sampling rate, then upsamples the input mesh by edge split (split the edge in the middle and connect the new vertex to the two vertices opposite the edge) until every edge in the mesh is shorter than half of the average edge. Next, the mesh is downsampled using the L^2-norm downsampling until the user's targeted number of vertices is reached (Fig. 16, middle).

The mesh now has a roughly uniform sampling rate. To improve the connectivity quality, we maximize the number of vertices with valence six by flipping an edge if it reduces the sum of the square of valences. This technique is used in the dynamic connectivity mesh in [14].

Next, a local smoothing operation is applied to smooth the parameterization of the vertices over the mesh surface. The vertices are shifted to the weighted average density of its adjacent triangles, in the tangential direction, that is, for a vertex i with adjacent triangles $T_j \in F_1(i)$, which is formed with vertices i, a_j, b_j, we apply the following update rule:

$$v = \frac{\sum_{T_j \in F_1(i)} A(T_j)(p_i + p_{a_j} + p_{b_j})/3}{\sum_{T_j \in F_1(i)} A(T_j)},$$

$$p_i \leftarrow p_i + (v - (v \cdot n_i)n_i)$$

where $A(T_j)$ is the area of T_j and n_i is the normal of the vertex i. We apply several steps of smoothing until the vertices stabilize.

Now the vertices may not be on the surface of the input mesh. Hence the final step is to re-project the vertices onto the original surface. We use the operator described in Sect. 8 of [13] for this purpose. Figure 16 shows a remeshed Venus model.

8 Conclusions

We have developed a decomposition framework for constructing a multiresolution hierarchy for meshes with arbitrary connectivity and topology. By devising a sampling-sensitive mesh difference error measure, we design a simple and efficient downsampling algorithm that attacks the high sampling density regions, and use the umbrella operator as the smoothing operator to filter out high frequency and also to regulate the triangles of the intermediate meshes in the hierarchy. This gives a better approximation of the frequency spectrum, according to the size of support of the vertices. Our approach has advantages in multiresolution editing and signal filtering.

The proposed framework should also be useful in other applications. Future research may explore its use in applications such as texture mapping, feature cut-and-paste editing, inter-model morphing for irregular meshes, and curvature-sensitive remeshing.

Acknowledgements. We wish to thank an anonymous reviewer for the detailed comments. We also thank Hongxin Zhang of Zhejiang University for a helpful discussion regarding the mesh difference error measurement. The datasets are courtesy of Cuberware, Stanford University, Headus, Inc., and Arthur Olson, The Scripps Research Institute.

References

1. Alliez P, Meyer M, Desbrun M (2002) Interactive geometry remeshing. In: Proceedings SIGGRAPH '02. Comput Graph 21(3):347–354
2. Desbrun M, Meyer M, Schröder P, Barr AH (1999) Implicit fairing of irregular meshes using diffusion and curvature flow. In: Proceedings SIGGRAPH '99. Comput Graph Proc, Annual Conference Series, pp 317–324
3. Eck M, DeRose T, Duchamp T, Hoppe H, Lounsbery M, Stuetzle W (1995) Multiresolution analysis of arbitrary meshes. In: Proceedings SIGGRAPH '95. Comput Graph Proc, Annual Conference Series, pp 173–182
4. Finkelstein A, Salesin D (1994) Multiresolution curves. In: Proceedings SIGGRAPH '94. Comput Graph Proc, Annual Conference Series, pp 261–268
5. Forsey DR, Bartels RH (1988) Hierarchical B-spline refinement. In: Proceedings SIGGRAPH '88. Comput Graph 22:205–212
6. Garland M, Heckbert PS (1997) Surface simplification using quadric error metrics. In: Proceedings SIGGRAPH '97. Comput Graph Proc, Annual Conference Series, pp 209–216
7. Gu X, Gortler SJ, Hoppe H (2002) Geometry images. In: Proceedings SIGGRAPH '02. ACM Trans Graph 21(3):355–361
8. Guskov I, Sweldens W, Schröder P (1999) Multiresolution signal processing for meshes. In: Proceedings SIGGRAPH '99. Comput Graph Proc, Annual Conference Series, pp 325–334
9. Guskov I, Vidimce K, Sweldens W, Schröder P (2000) Normal meshes. In: Proceedings SIGGRAPH '00. Comput Graph Proc, Annual Conference Series, pp 95–102
10. Hoppe H (1996) Progressive meshes. In: Proceedings SIGGRAPH '96. Comput Graph Proc, Annual Conference Series, pp 99–108
11. Igarashi T, Matsuoka S, Tanaka H (1999) Teddy: a sketching interface for 3D freeform design. In: Proceedings SIGGRAPH '99, Comput Graph Proc, Annual Conference Series, pp 409–416
12. Khodakovsky A, Schröder P, Sweldens W (2000) Progressive geometry compression. In: Proceedings SIGGRAPH '00. Comput Graph Proc, Annual Conference Series, pp 271–278
13. Kim D, Kim J, Ko H-S (1999) Unification of distance and volume optimization in surface simplification. J Graph Models Image Process 61:363–367
14. Kobbelt L, Bareuther T, Seidel H-P (2000) Multiresolution shape deformations for meshes with dynamic vertex connectivity. Comput Graph Forum 19(3):249–259
15. Kobbelt L, Campagna S, Seidel H-P (1998) A general framework for mesh decimation. In: Proceedings of Graphics Interface '98, pp 43–50
16. Kobbelt L, Campagna S, Vorsatz J, Seidel H-P (1998) Interactive multiresolution modeling on arbitrary meshes. In: Proceedings SIGGRAPH '98. Comput Graph Proc, Annual Conference Series, pp 105–114
17. Kobbelt L, Vorsatz J, Seidel H-P (1999) Multiresolution hierarchies on unstructured triangle meshes. Comput Geom 14(1–3):5–24
18. Lee A, Sweldens W, Schröder P, Cowsar L, Dobkin D (1998) MAPS: Multiresolution adaptive parameterization of surfaces. In: Proceedings SIGGRAPH '98. Comput Graph Proc, Annual Conference Series, pp 95–104
19. Lindstrom P, Turk G (1998) Fast and memory efficient polygonal simplification. In: Proceedings IEEE Visualization '98, pp 279–286
20. Taubin G (1995) A signal processing approach to fair surface design. In: Proceedings SIGGRAPH '95. Comput Graph Proc, Annual Conference Series, pp 351–358
21. Taubin G (2000) Geometric signal processing on polygonal meshes. In: EUROGRAPHICS '2000
22. Zorin D, Schröder P (2000) Subdivision for modeling and animation. In: SIGGRAPH 2000 course notes
23. Zorin D, Schröder P, Sweldens W (1997) Interactive multiresolution mesh editing. In: Proceedings SIGGRAPH '97. Comput Graph Proc, Annual Conference Series, pp 259–469

Photographs of the authors and their biographies are given on the next page.

OSCAR KIN-CHUNG AU is a Ph.D. student at the Hong Kong University of Science & Technology, from which he received his B.Sc. and M.Phil. degrees in computer science in 2001 and 2003, respectively. His research interests include computer graphics and polygonal mesh modeling and processing.

CHIEW-LAN TAI is an assistant professor of computer science at the Hong Kong University of Science and Technology. She received her B.Sc. and M.Sc. in mathematics from the University of Malaya, her M.Sc. in computer & information sciences from the National University of Singapore, and D.Sc. in information science from the University of Tokyo. Her research interests include geometric modeling and computer graphics.