Interactive Multi-Resolution Modeling on Arbitrary Meshes

Leif Kobbelt* Swen Campagna Jens Vorsatz Hans-Peter Seidel

University of Erlangen–Nürnberg

Abstract

During the last years the concept of multi-resolution modeling has gained special attention in many fields of computer graphics and geometric modeling. In this paper we generalize powerful multi-resolution techniques to arbitrary triangle meshes without requiring subdivision connectivity. Our major observation is that the hierarchy of nested spaces which is the structural core element of most multi-resolution algorithms can be replaced by the sequence of intermediate meshes emerging from the application of incremental mesh decimation. Performing such schemes with local frame coding of the detail coefficients already provides effective and efficient algorithms to extract multi-resolution information from unstructured meshes. In combination with discrete fairing techniques, i.e., the constrained minimization of discrete energy functionals, we obtain very fast mesh smoothing algorithms which are able to reduce noise from a geometrically specified frequency band in a multi-resolution decomposition. Putting mesh hierarchies, local frame coding and multi-level smoothing together allows us to propose a flexible and intuitive paradigm for interactive detail-preserving mesh modification. We show examples generated by our mesh modeling tool implementation to demonstrate its functionality.

1 Introduction

Traditionally, geometric modeling is based on piecewise polynomial surface representations [8, 16]. However, while special polynomial basis functions are well suited for describing and modifying smooth triangular or quadrilateral *patches*, it turns out to be rather difficult to smoothly join several pieces of a composite surface along common (possibly trimmed) boundary curves. As flexible patch layout is crucial for the construction of non-trivial geometric shapes, spline-based modeling tools do spend much effort to maintain the global smoothness of a surface.

Subdivision schemes can be considered as an algorithmic generalization of classical spline techniques enabling control meshes with arbitrary topology [2, 5, 6, 18, 22, 39]. They provide easy access to globally smooth surfaces of arbitrary shape by iteratively applying simple refinement rules to the given control mesh. A sequence of meshes generated by this process quickly converges to a smooth limit surface. For most practical applications, the refined meshes are already sufficiently close to the smooth limit after only a few refinement steps.

Within a multi-resolution framework, subdivision schemes provide a set of basis functions $\phi_{i,j} = \phi(2^i \cdot - j)$ which are suitable to build a cascade of nested spaces $V_i = \mathrm{span}(\{\phi_{i,j}\}_j)$ [4, 33]. Since the functions $\phi_{i,j}$ are defined by uniform refinement of a given control mesh $\mathcal{M}_0 \cong V_0$, the spaces V_i have to be isomorphic to meshes $\mathcal{M}_i$ with *subdivision connectivity*.

While being much more flexible than classical (tensor-product) spline techniques, the multi-resolution representation based on the uniform refinement of a polygonal base mesh is still rather rigid. When analyzing a given mesh $\mathcal{M}_k$, i.e., when decomposing the mesh into disjoint frequency bands $W_i = V_{i+1} \setminus V_i$, we have to *invert* the uniform refinement operation $V_i \to V_{i+1}$. Hence, the input mesh always has to be topologically isomorphic to an iteratively refined base grid. In general this requires a global remeshing/resampling of the input data prior to the multi-resolution analysis [7]. Moreover, if we want to fuse several separately generated subdivision meshes (e.g. laser range scans) into one model, restrictive compatibility conditions have to be satisfied. Hence, subdivision schemes are able to deal with arbitrary *topology* but not with arbitrary *connectivity*!

The *scales* of subdivision based multi-resolution mesh representations are defined in terms of topological distances. Since every vertex $\mathbf{p}_{i,j}$ on each level of subdivision $\mathcal{M}_i$ represents the weight coefficient of a particular basis function $\phi_{i,j}$ with fixed support, its region of influence is determined by topological neighborhood in the mesh instead of geometric proximity. Being derived from the regular functional setting, the refinement rules of stationary subdivision schemes only depend on the valences of the vertices but not on the length of the adjacent edges. Hence, surface artifacts can occur when the given base mesh is locally strongly distorted.

Assume we have a subdivision connectivity mesh and want to apply modifications on a specific scale V_i. The usual way to implement this operation is to run a decomposition scheme several steps until the desired resolution level is reached. On this level the mesh $\mathcal{M}_i$ is modified and the reconstruction starting with $\mathcal{M}_i'$ yields the final result. The major draw-back of this procedure is the fact that coarse basis functions exist for the coarse-mesh vertices only and hence all low-frequency modifications have to be *aligned* to the grid imposed by the subdivision connectivity. Shifted low-frequency modifications can be faked by moving a *group* of vertices from a finer scale simultaneously but this annihilates the mathematical elegance of multi-resolution representations.

A standard demo example for multi-resolution modeling is pulling the nose tip of a human head model. Depending on the chosen scale either the whole face is affected or just the nose is elongated. On uniformly refined meshes this operation only works if a coarse-scale control vertex happens to be located right at the nose tip. However, for an *automatic* remeshing algorithm it is very difficult, if not impossible, to place the coarse-scale vertices at the semantically relevant features of an object.

In this paper we present an alternative approach to multi-resolution modeling which avoids these three major difficulties, i.e. the restriction to subdivision connectivity meshes, the restriction to basis functions with fixed support and the alignment of potential coarser-scale modifications.

*Computer Sciences Department (IMMD9), University of Erlangen-Nürnberg, Am Weichselgarten 9, 91058 Erlangen, Germany, Leif.Kobbelt@informatik.uni-erlangen.de

The first problem is solved by using mesh hierarchies which emerge from the application of a mesh decimation scheme. In Section 2 we derive the necessary equipment to extract multi-resolution information from arbitrary meshes and geometrically encode detail information with respect to local frames which adapt to the local geometry of the coarser approximation of the object.

To overcome the problems arising from the fixed support and aligned distribution of subdivision basis functions, we drop the structural concept of considering a surface in space to be a linear combination of scalar-valued basis functions. On each level of detail, the lower-frequency components of the geometric shape are simply characterized by energy minimization (*fairing*). In Section 3 we overview the discrete fairing technique [19, 38] and show how a combination with the non-uniform mesh hierarchy leads to highly efficient mesh optimization algorithms. Due to the local smoothing properties of the fairing operators, we are able to define a *geometric* threshold for the wavelength up to which a low-pass filter should remove noise.

With an efficient hierarchical mesh smoothing scheme available, we propose a flexible mesh modification paradigm in Section 4. The basic idea is to let the designer freely define the region of influence and the characteristics of the modification which both can be adapted to the surface geometry instead of being determined by the connectivity. The selected region defines the "frequency" of the modification since it provides the boundary conditions for a constrained energy minimization. Nevertheless the detail information within the selected region is preserved and does change according to the global modification. Exploiting the efficient schemes from Section 3 leads to interactive response times for moderately complex models.

Throughout the paper, we consider a modeling scenario where a triangle mesh $\mathcal{M}$ with arbitrary connectivity is *given* (no from-scratch design). All modifications just alter the position of the vertices but not their adjacency. In particular, we do not consider ad infinitum subdivision to establish infinitesimal smoothness. The given mesh $\mathcal{M} = \mathcal{M}_k$ represents per definition the finest level of detail.

2 Multi-resolution representations

Most schemes for the multi-resolution representation and modification of triangle meshes emerge from generalizing harmonic analysis techniques like the wavelet transform [1, 23, 30, 33]. Since the fundamentals have been derived in the scalar-valued functional setting $\mathbb{R}^d \to \mathbb{R}$, difficulties emerge from the fact that manifolds in space are in general not topologically equivalent to simply connected regions in $\mathbb{R}^d$.

The philosophy behind multi-resolution modeling on surfaces is hence to mimic the algorithmic structure of the related functional transforms and preserve some of the important properties like locality, smoothness, stability or polynomial precision which have related meaning in both settings [9, 12, 40]. Accordingly, the nested sequence of spaces underlying the decomposition into disjoint frequency bands is thought of being generated bottom-up from a coarse base mesh up to finer and finer resolutions. This implies that subdivision connectivity is mandatory on higher levels of detail. Not only the mesh has to consist of large regular regions with isolated extra-ordinary vertices in between. Additionally, we have to make sure that the topological distance between the singularities is the same for every pair of neighboring singularities and this topological distance has to be a power of 2.

Such special topological requirements prevent the schemes from being applicable to arbitrary input meshes. Global remeshing and resampling is necessary to obtain a proper hierarchy which gives rise to alias-errors and requires involved computations [7].

Luckily, the restricted topology is not necessary to define different levels of resolution or approximation for a triangle mesh. In the literature on mesh decimation we find many examples for hierarchies built on arbitrary meshes [11, 15, 20, 24, 27, 31, 35]. The key is always to build the hierarchy top-down by eliminating vertices from the current mesh (*incremental reduction, progressive meshes*). Running a mesh decimation algorithm, we can stop, e.g., every time a certain percentage of the vertices is removed. The intermediate meshes can be used as a level-of-detail representation [15, 23].

In both cases, i.e., the bottom-up or the top-down generation of nested (vertex-) grids, the multi-resolution concept is rigidly attached to topological entities. This makes sense if hierarchies are merely used to reduce the complexity of the representation. In the context of multi-resolution modeling, however, we want the hierarchy not necessarily to rate meshes according to their *coarseness* but rather according to their *smoothness* (cf. Fig 1).

We will use multi-resolution hierarchies for two purposes. First we want to derive highly efficient algorithms for mesh optimization. In Section 3 we will see that topologically reduced meshes are the key to significantly increase the performance (levels of coarseness). On the other hand, we want to avoid any restrictions that are imposed by topological peculiarities. In particular, when interactively modifying a triangle mesh, we do not want any alignment. The *support* of a modification should have no influence on *where* this modification can be applied (levels of smoothness).

To describe the different set-ups for multi-resolution representation uniformly, we define a generic decomposition scheme $\mathbf{A} = (\mathbf{A}_\Phi | \mathbf{A}_\Psi)^T$ (*analysis*) as a general procedure that transforms a given mesh $\mathcal{M}_i$ into a coarser/smoother one $\mathcal{M}_{i-1} = \mathbf{A}_\Phi \mathcal{M}_i$ plus detail coefficients $\mathcal{D}_{i-1} = \mathbf{A}_\Psi \mathcal{M}_i$. In the standard wavelet setting the cardinalities satisfy $\#\mathcal{D}_{i-1} + \#\mathcal{M}_{i-1} = \#\mathcal{M}_i$ since decomposition is a proper basis transform.

If a (bi-orthogonal) wavelet basis is not known, we have to store more detail information ($\#\mathcal{D}_{i-1} + \#\mathcal{M}_{i-1} > \#\mathcal{M}_i$) since the reconstruction operator $\mathbf{A}^{-1}$ might be computationally expensive or not even uniquely defined. Well-known examples for this kind of decomposition with extra detail coefficients are the Laplacian-pyramid type of representation in [40] and the progressive mesh representation [15].

When $\mathbf{A}_\Phi$ is merely a smoothing operator which does not change the topological mesh structure of $\mathcal{M}_i$ we have $\mathbf{A}_\Psi = \mathbf{I}d - \mathbf{A}_\Phi$ and $\#\mathcal{D}_{i-1} = \#\mathcal{M}_{i-1} = \#\mathcal{M}_i$.

2.1 Local Frames

In a multi-resolution representation of a geometric object $\mathcal{M} = \mathcal{M}_k$, the detail coefficients $\mathcal{D}_{i-1}$ describe the difference between two approximations $\mathcal{M}_{i-1}$ and $\mathcal{M}_i$ having different levels of detail. For parametric surfaces, the detail coefficients, i.e., the spatial location of the vertices in $\mathcal{M}_i$ have to be encoded relative to the local geometry of the coarser approximation $\mathcal{M}_{i-1}$. This is necessary since modifications on the coarser level should have an intuitive effect on the geometric features from finer scales.

First proposed by [10] it has become standard to derive local coordinate frames from the partial derivative information of the coarse representation $\mathcal{M}_{i-1}$. Since we do not assume the existence of any global structure or auxiliary information in the sequence of meshes $\mathcal{M}_i$, we have to rely on intrinsic geometric properties of the triangles themselves. Depending on the intended application we assign the local frames to the triangles or the vertices of $\mathcal{M}_{i-1}$. A detail vector is then defined by three coordinate values $[u, v, n]$ plus an index i identifying the affine frame $F_i = [\mathbf{p}_i, U_i, V_i, N_i]$ with respect to which the coordinates are given.

2.1.1 Vertex-based frames

We can use any heuristic to estimate the normal vector N_i at a vertex $\mathbf{p}_i$ in a polygonal mesh, e.g., taking the average of the adjacent triangle normals. The vector $U_i = E - (E^T N_i) N_i$ is obtained by pro-

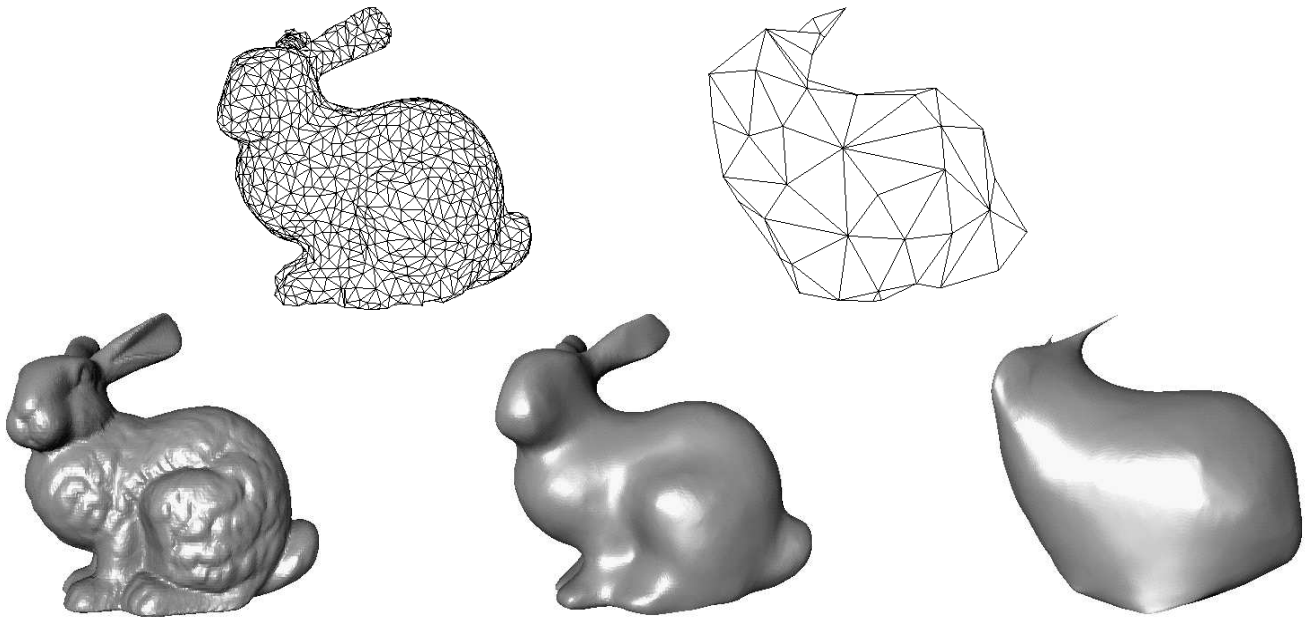

Figure 1: The well-known Stanford-Bunny. Although the original mesh does not have subdivision connectivity, mesh decimation algorithms easily generate a hierarchy of topologically simplified meshes. On the other hand, multi-resolution modeling also requires hierarchies of differently *smooth* approximations. Notice that the meshes in the lower row have identical connectivity.

jecting any adjacent edge E into the tangent plane and $V_i := N_i \times U_i$. The data structure for storing the mesh $\mathcal{M}_{i-1}$ has to make sure that E is uniquely defined, e.g. as the first member in a list of neighbors.

2.1.2 Face-based frames

It is tempting to simply use the local frame which is given by two triangle edges and their cross product. However, this will not lead to convincing detail reconstruction after modifying the coarser level. The reason for this is that the local frames would be rigidly attached to one coarse triangle. In fact, tracing the dependency over several levels of detail shows that the original mesh is implicitly partitioned into sub-meshes being assigned to the same coarse triangle T. Applying a transformation to T implies the same transformation for all vertices being defined relative to T. This obviously leads to artifacts between neighboring sub-meshes in the fine mesh.

A better choice is to use local low order polynomial interpolants or approximants that depend on more than one single triangle. Let $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_2$ be the vertices of a triangle $T \in \mathcal{M}_{i-1}$ and $\mathbf{p}_3$, $\mathbf{p}_4$, and $\mathbf{p}_5$ be the opposite vertices of the triangles adjacent to T (cf. Fig. 2). To construct a quadratic polynomial

$$\mathbf{F}(u,v) \;=\; \mathbf{f} + u\mathbf{f}_u + v\mathbf{f}_v + \frac{u^2}{2}\mathbf{f}_{uu} + uv\mathbf{f}_{uv} + \frac{v^2}{2}\mathbf{f}_{vv}$$

approximating the $\mathbf{p}_i$ we have to define a parameterization first. Note that the particular choice of this parameterization controls the quality of the approximant. Since we want to take the geometric constellation of the $\mathbf{p}_i$ into account, we define a parameterization by projecting the vertices into the supporting plane of T.

Exploiting the invariance of the polynomial interpolant with respect to affine re-parameterizations, we can require $\mathbf{F}(0,0) := \mathbf{p}_0$, $\mathbf{F}(1,0) := \mathbf{p}_1$, and $\mathbf{F}(0,1) := \mathbf{p}_2$ which implies

$$\begin{aligned} \mathbf{f} &= \mathbf{p}_0 \\ \mathbf{f}_u &= \mathbf{p}_1 - \mathbf{p}_0 - \tfrac{1}{2}\mathbf{f}_{uu} \\ \mathbf{f}_v &= \mathbf{p}_2 - \mathbf{p}_0 - \tfrac{1}{2}\mathbf{f}_{vv}. \end{aligned} \qquad (1)$$

Let the vertices $\mathbf{p}_3$, $\mathbf{p}_4$, and $\mathbf{p}_5$ be projected to (u_3,v_3), (u_4,v_4), and (u_5,v_5) according to the frame $[\mathbf{p}_0,\mathbf{p}_1,\mathbf{p}_2]$. To additionally stabilize the interpolation scheme, we introduce a tension parameter $\tau \in [0,1]$ which trades approximation error at $\mathbf{p}_3$, $\mathbf{p}_4$, and $\mathbf{p}_5$ for minimizing the bending energy $\mathbf{f}_{uu}^2 + 2\mathbf{f}_{uv}^2 + \mathbf{f}_{vv}^2$. Using (1) we obtain

$$\begin{pmatrix} \frac{1}{2}u_3(u_3-1) & u_3v_3 & \frac{1}{2}v_3(v_3-1) \\ \frac{1}{2}u_4(u_4-1) & u_4v_4 & \frac{1}{2}v_4(v_4-1) \\ \frac{1}{2}u_5(u_5-1) & u_5v_5 & \frac{1}{2}v_5(v_5-1) \\ \tau & 0 & 0 \\ 0 & 2\tau & 0 \\ 0 & 0 & \tau \end{pmatrix} \begin{pmatrix} \mathbf{f}_{uu} \\ \mathbf{f}_{uv} \\ \mathbf{f}_{vv} \end{pmatrix} = \begin{pmatrix} (\mathbf{p}_3-\mathbf{p}_0) + u_3(\mathbf{p}_0-\mathbf{p}_1) + v_3(\mathbf{p}_0-\mathbf{p}_2) \\ (\mathbf{p}_4-\mathbf{p}_0) + u_4(\mathbf{p}_0-\mathbf{p}_1) + v_4(\mathbf{p}_0-\mathbf{p}_2) \\ (\mathbf{p}_5-\mathbf{p}_0) + u_5(\mathbf{p}_0-\mathbf{p}_1) + v_5(\mathbf{p}_0-\mathbf{p}_2) \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

which has to be solved in a least squares sense.

To compute the detail coefficients $[\hat{u},\hat{v},h]$ for a point $\mathbf{q}$ with respect to T, we start from the center $(u,v) = (\frac{1}{3},\frac{1}{3})$ and simple Newton iteration steps $(u,v) \leftarrow (u,v) + (\triangle u, \triangle v)$ with $\mathbf{d} = \mathbf{q} - \mathbf{F}(u,v)$ and

$$\begin{pmatrix} \mathbf{F}_u^T\mathbf{F}_u & \mathbf{F}_u^T\mathbf{F}_v \\ \mathbf{F}_u^T\mathbf{F}_v & \mathbf{F}_v^T\mathbf{F}_v \end{pmatrix} \begin{pmatrix} \triangle u \\ \triangle v \end{pmatrix} = \begin{pmatrix} \mathbf{F}_u^T\mathbf{d} \\ \mathbf{F}_u^T\mathbf{d} \end{pmatrix}$$

quickly converge to the point $\mathbf{F}(\hat{u},\hat{v})$ with the detail vector $\mathbf{d}$ perpendicular to the surface $\mathbf{F}(u,v)$. The third coefficient is then $h = \text{sign}(\mathbf{d}^T(\mathbf{F}_u \times \mathbf{F}_v))\,\|\mathbf{d}\|$.

Although the parameter values $(\hat{u},\hat{v})$ can lie outside the unit triangle (which occasionally occurs for extremely distorted configurations) the detail coefficient $[\hat{u},\hat{v},h]$ is still well-defined and reconstruction works. Notice that the scheme might produce counter-intuitive results if the maximum dihedral angle between T and one of its neighbors becomes larger than $\frac{\pi}{2}$. In this case the parameter-

ization for $\mathbf{p}_3$, $\mathbf{p}_4$, and $\mathbf{p}_5$ could be derived by rotation about T's edges instead of projection.

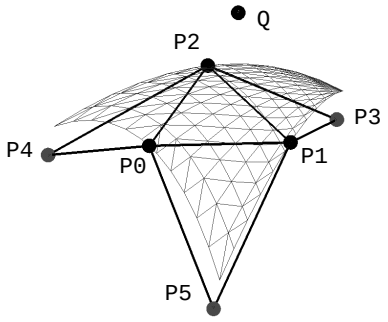

Figure 2: Vertex labeling for the construction of a local frame.

Obviously, the detail coefficient $[\hat{u}, \hat{v}, h]$ is not coded with respect to a local frame in the narrow sense. However, it has a similar semantics. Recovering the vertex position $\mathbf{q}'$ requires to construct the approximating polynomial $\mathbf{F}'(u,v)$ for the possibly modified vertices $\mathbf{p}'_i$, evaluate at $(\hat{u}, \hat{v})$ and move in normal direction by h. The distance h is a measure for the "size" of the detail.

In our current implementation on a SGI R10000/195 MHz workstation the analysis $\mathbf{q} \rightarrow [\hat{u}, \hat{v}, h]$ takes about $20\mu S$ while the reconstruction $[\hat{u}, \hat{v}, h] \rightarrow \mathbf{q}$ takes approximately $8\mu S$. Since a progressive mesh representation introduces two triangles per vertex split, this means that for the reconstruction of a mesh with 10^5 triangles, the computational overhead due to the local frame representation is less than half a second.

2.2 Decomposition and reconstruction

To complete our equipment for the multi-resolution set-up we have to define the decomposition and reconstruction operations which separate the high-frequency detail from the low-frequency shape and eventually recombine the two to recover the original mesh. We apply different strategies depending on whether decomposition generates a coarser approximation of the original geometry or a smoother approximation.

In either case the decomposition operator $\mathbf{A}$ is applied to the original mesh $\mathcal{M}_i$ and the details $\mathcal{D}_{i-1}$ are coded in local frame coordinates with respect to $\mathcal{M}_{i-1}$. Since the reconstruction is an extrapolation process, it is numerically unstable. To stabilize the operation we have to make the details as small as possible, i.e., when encoding the spatial position of a point $\mathbf{q} \in \mathbb{R}^3$ we pick that local frame on $\mathcal{M}_{i-1}$ which is closest to $\mathbf{q}$.

Usually the computational complexity of generating the detail coefficients is higher than the complexity of the evaluation during reconstruction. This is an important feature since for interactive modeling the (dynamic) reconstruction has to be done in real-time while the requirements for the (static) decomposition are not as demanding.

2.2.1 Mesh decimation based decomposition

When performing an incremental mesh decimation algorithm, each reduction step removes one vertex and retriangulates the remaining hole [15, 31]. We use a simplified version of the algorithm described in [20] that controls the reduction process in order to optimize the fairness of the coarse mesh while keeping the global approximation error below a prescribed tolerance.

The basic topological operation is the *half edge collapse* which shifts one vertex $\mathbf{p}$ into an adjacent vertex $\mathbf{q}$ and removes the two degenerate triangles. In [20] a fast algorithm is presented to determine that triangle T in the neighborhood of the collapse which lies closest to the removed vertex $\mathbf{p}$. The position of $\mathbf{p}$ is then coded with respect to the local frame associated with this triangle.

The inverse operation of an edge collapse is the *vertex split* [15]. Since during reconstruction the vertices are inserted in the reverse order of their removal, it is guaranteed that, when $\mathbf{p}$ is inserted, the topological neighborhood looks the same as when it was deleted and hence the local frame to transform the detail coefficient for $\mathbf{p}$ back into world coordinates is well-defined.

During the iterative decimation, each intermediate mesh could be considered as one individual level of detail approximation. However, if we want to define disjoint frequency bands, it is reasonable to consider a whole sub-sequence of edge collapses as one atomic transformation from one level $\mathcal{M}_i$ to $\mathcal{M}_{i-1}$.

There are several criteria to determine which levels mark the boundaries between successive frequency bands. One possibility is to simply define $\mathcal{M}_i$ to be the coarsest mesh that still keeps a maximum tolerance of less than some ε_i to the original data. Alternatively we can require the number of vertices in $\mathcal{M}_{i-1}$ to be a fixed percentage of the number of vertices in $\mathcal{M}_i$. This simulates the geometric decrease of complexity known from classical multi-resolution schemes. We can also let the human user decide when a significant level of detail is reached by allowing her to browse through the sequence of incrementally reduced meshes.

In order to achieve optimal performance with the multi-level smoothing algorithm described in the next section, we decided in our implementation to distribute the collapses evenly over the mesh: When a collapse $\mathbf{p} \rightarrow \mathbf{q}$ is performed, all vertices adjacent to $\mathbf{q}$ are locked for further collapsing (independent set of collapses). If no more collapses are possible, the current mesh defines the next level of detail and all vertices are un-locked. One pass of this reduction scheme removes about 25% of the vertices in average.

2.2.2 Mesh smoothing based decomposition

For multi-resolution *modeling* we have to separate high frequency features from the global shape in order to modify both individually. Reducing the mesh complexity cannot help in this case since coarser meshes do no longer have enough degrees of freedom to be smooth, i.e., to have small angles between adjacent triangles. Hence, the decomposition operator $\mathbf{A}_\Phi$ becomes a mere smoothing operator that reduces the discrete bending energy in the mesh without changing the topology (cf. Section 3).

A natural way to define the detail coefficients would be to store the difference vectors between the original vertex position $\mathbf{q}$ and the shifted position $\mathbf{q}'$ with respect to the local frame defined at $\mathbf{q}'$. However, in view of numerical stability this choice is not optimal. Depending on the special type of smoothing operator $\mathbf{A}_\Phi$ the vertices can move "within" the surface such that another vertex $\mathbf{p}' \in \mathcal{M}_{i-1} = \mathbf{A}_\Phi \mathcal{M}_i$ could lie closer to $\mathbf{q}$ than $\mathbf{q}'$ (cf. Fig. 3).

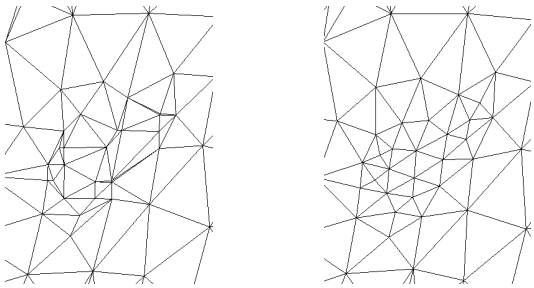

Figure 3: Although the bending energy minimizing smoothing operator $\mathbf{A}_\Phi$ is applied to a *plane* triangulation, the vertices are moved within the plane since linear operators always do the fairing with respect to a specific parameterization (cf. Section 3).

To stabilize the reconstruction, i.e., to minimize the length of the detail vectors, we apply a simple local search procedure to find the

nearest vertex $\mathbf{p}' \in \mathcal{M}_{i-1}$ to $\mathbf{q}$ and express the detail vector with respect to the local frame at $\mathbf{p}'$ or one of its adjacent triangles. This searching step does not noticeably increase the computation time (which is usually dominated by the smoothing operation $\mathbf{A}_\Phi$) but leads to much shorter detail vectors (cf. Fig 4).

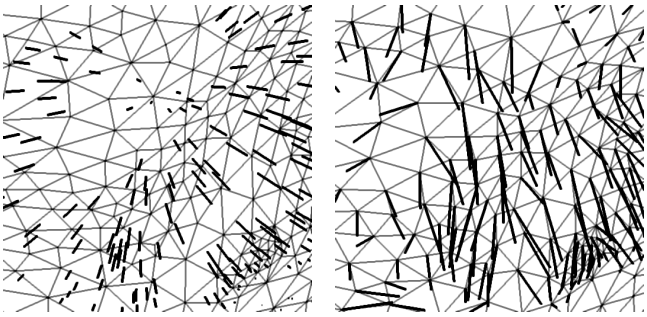

Figure 4: The shortest detail vectors are obtained by representing the detail coefficients with respect to the nearest local frame (left) instead of attaching the detail vectors to the topologically corresponding original vertices.

3 Discrete fairing

From CAGD it is well-known that constrained energy minimization is a very powerful technique to generate high quality surfaces [3, 13, 25, 28, 37]. For efficiency, one usually defines a simple quadratic energy functional $\mathcal{E}(f)$ and searches among the set of functions satisfying prescribed interpolation constraints for that function f which minimizes $\mathcal{E}$.

Transferring the continuous concept of energy minimization to the discrete setting of triangle mesh optimization leads to the discrete fairing approach [19, 38]. Local polynomial interpolants are used to estimate derivative information at each vertex by divided difference operators. Hence, the differential equation characterizing the functions with minimum energy is discretized into a linear system for the vertex positions.

Since this system is global and sparse, we apply iterative solving algorithms like the Gauß-Seidel-scheme. For such algorithms one iteration step merely consists in the application of a simple local averaging operator. This makes discrete fairing an easy accessible technique for mesh optimization.

3.1 The umbrella-algorithm

The most prominent energy functionals that are used in the theory and practice of surface design are the membrane energy

$$\mathcal{E}_M(f) := \int f_u^2 + f_v^2 \tag{2}$$

which prefers functions with smaller surface area and the thin plate energy

$$\mathcal{E}_{TP}(f) := \int f_{uu}^2 + 2\,f_{uv}^2 + f_{vv}^2 \tag{3}$$

which punishes strong bending. The variational calculus leads to simple characterizations of the corresponding minimum energy surfaces

$$\triangle f = f_{uu} + f_{vv} = 0 \tag{4}$$

or

$$\triangle^2 f = f_{uuuu} + 2\,f_{uuvv} + f_{vvvv} = 0 \tag{5}$$

respectively. Obviously, low degree polynomials satisfy both differential equations and hence appropriate (Dirichlet-) boundary conditions have to be imposed which make the semi-definite functionals $\mathcal{E}_M$ and $\mathcal{E}_{TP}$ positive-definite.

The discrete fairing approach discretizes either the energy functionals (2–3) [19, 38] or the corresponding Euler-Lagrange equations (4–5) [17, 36] by replacing the differential operators with divided difference operators. To construct these operators, we have to choose an appropriate parameterization in the vicinity of each vertex. In [38] for example a discrete analogon to the exponential map is chosen. In [17] the *umbrella-algorithm* is derived by choosing a symmetric parameterization

$$(u_i, v_i) := \Big(\cos(2\pi\frac{i}{n}), \sin(2\pi\frac{i}{n})\Big), \quad i = 0, \ldots, n-1 \tag{6}$$

with n being the valence of the center vertex $\mathbf{p}$ (cf. Fig 5). This parameterization does not adapt to the local geometric constellation but it simplifies the construction of the corresponding difference operators which are otherwise obtained by solving a Vandermonde system for local polynomial interpolation. With the special parameterization (6) the discrete analogon of the Laplacian $\triangle f$ turns out to be the umbrella-operator

$$\mathcal{U}(\mathbf{p}) = \frac{1}{n}\sum_{i=0}^{n-1} \mathbf{p}_i - \mathbf{p}$$

with $\mathbf{p}_i$ being the direct neighbors of $\mathbf{p}$ (cf. Fig. 5). The umbrella-operator can be applied recursively leading to

$$\mathcal{U}^2(\mathbf{p}) = \frac{1}{n}\sum_{i=0}^{n-1} \mathcal{U}(\mathbf{p}_i) - \mathcal{U}(\mathbf{p})$$

as a discretization of $\triangle^2 f$.

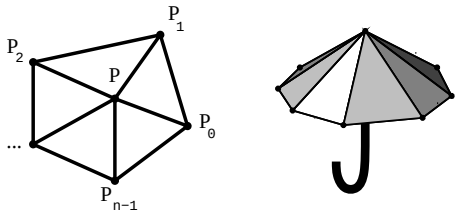

Figure 5: To compute the discrete Laplacian, we need the 1-neighborhood of a vertex $\mathbf{p}$ ($\rightarrow$ umbrella-operator).

The boundary conditions are imposed to the discrete problem by freezing certain vertices. When minimizing the discrete version of $\mathcal{E}_M$ we hold a closed boundary polygon fixed and compute the membrane that is spanned in between. For the minimization of $\mathcal{E}_{TP}$ we need two rings of boundary vertices, i.e., we keep a closed strip of triangles fixed. This imposes a (discrete) C^1 boundary condition to the optimization problem (cf. Fig 6). All internal vertices can be moved freely to minimize the global energy. The properly chosen boundary conditions imply positive-definiteness of the energy functional and guarantee the convergence of the iterative solving algorithm [29].

The characteristic (linear) system for the corresponding unconstrained minimization problem has rows $\mathcal{U}(\mathbf{p}_i) = 0$ or $\mathcal{U}^2(\mathbf{p}_i) = 0$ respectively for the free vertices $\mathbf{p}_i$. An iterative solving scheme approaches the optimal solution by solving each row of the system separately and cycling through the list of free vertices until a stable solution is reached. In case of the membrane energy $\mathcal{E}_M$ this leads to the local update rule

$$\mathbf{p}_i \leftarrow \mathbf{p}_i + \mathcal{U}(\mathbf{p}_i) \tag{7}$$

and for the thin plate energy $\mathcal{E}_{TP}$, we obtain

$$\mathbf{p}_i \leftarrow \mathbf{p}_i - \frac{1}{\nu}\,\mathcal{U}^2(\mathbf{p}_i) \tag{8}$$

with the "diagonal element"

$$\nu = 1 + \frac{1}{n_i}\sum_j \frac{1}{n_{i,j}}$$

where n_i and $n_{i,j}$ are the valences of the center vertex $\mathbf{p}_i$ and its jth neighbor respectively.

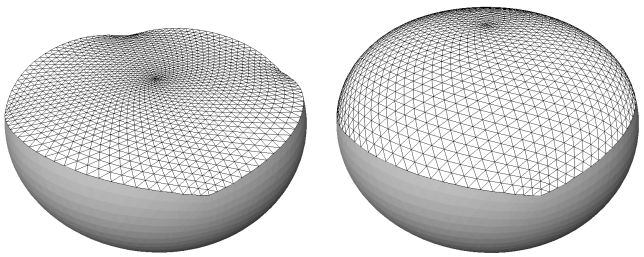

Figure 6: A closed polygon or a closed triangle strip provide C^0 or C^1 boundary conditions for the discrete fairing. On the left the triangle mesh minimizes $\mathcal{E}_M$ on the right it minimizes $\mathcal{E}_{TP}$.

Although the rule (8) can be implemented recursively, the performance is optimized if we use a two step process where all umbrella vectors $\mathcal{U}(\mathbf{p}_i)$ are computed in a first pass and $\mathcal{U}^2(\mathbf{p}_i)$ in the second. This avoids double computation but it also forces us to use in fact a plain Jacobi-solver since no intermediate updates from neighboring vertices can be used. However the $(n+2):2$ speed-up for a vertex with valence n amortizes the slower convergence of Jacobi compared to Gauß-Seidel.

3.2 Connection to Taubin's signal processing approach

The local update operator (7) in the iterative solving scheme for constrained energy minimization is exactly the Laplace smoothing operator proposed by Taubin in [34] where he derived it (also in the context of mesh smoothing) from a filter formalism based on generalized Fourier analysis for arbitrary polygonal meshes. In his paper, Taubin starts with a matrix version of the scaled update rule (7)

$$[\mathbf{p}_i'] := (I + \lambda\,\mathcal{U})\,[\mathbf{p}_i]$$

where λ is a damping factor and formally rewrites it by using a transfer function notation

$$f(k) := 1 - \lambda k$$

with respect to the eigenbasis of the (negative) Laplace operator. Since no proper boundary conditions are imposed, the continued filtering by $f(k)$ leads to severe shrinking and hence he proposes combined filters

$$f(k) := (1-\lambda k)\,(1-\mu k) \tag{9}$$

where λ and μ are set in order to minimize the shrinking. A feasible heuristic is to choose a *pass-band frequency*

$$k_{PB} = \frac{1}{\lambda} + \frac{1}{\mu} \in [0.01 \ldots 0.1]$$

and set λ and μ while observing the stability of the filter.

Obviously, the update rule for the difference equation $\mathcal{U}(\mathbf{p}_i) = 0$ which characterizes meshes with minimum membrane energy corresponds to a special low-pass filter with transfer function $f_{\mathcal{U}}(k) = (1-k)$. For the minimization of the total curvature, characterized by $\mathcal{U}^2(\mathbf{p}_i) = 0$, the iteration rule (8) can be re-written in transfer function notation as

$$f_{\mathcal{U}^2}(k) = (1 - \frac{1}{\nu}k^2) = (1 + \frac{1}{\sqrt{\nu}}k)\,(1 - \frac{1}{\sqrt{\nu}}k)$$

which corresponds to a combined Laplace filter of the form (9) with pass-band frequency $k_{PB} = 0$. Although this is not optimal for reducing the shrinking effect, we observe that the transfer function happens to have a vanishing derivative at $k = 0$. From signal processing theory it is known that this characterizes maximal smoothness [26], i.e., among the two step Laplace filters, the $\mathcal{U}^2$-filter achieves optimal smoothing properties. To stabilize the filter we might want to introduce a damping factor $0 < \sigma < \frac{1}{2}\nu$ into the update-rule

$$\mathbf{p}_i \leftarrow \mathbf{p}_i - \frac{\sigma}{\nu}\,\mathcal{U}^2(\mathbf{p}_i)$$

3.3 Multi-level smoothing

A well-known negative result from numerical analysis is that straight forward iterative solvers like the Gauß-Seidel scheme are not appropriate for large sparse problems [32]. More sophisticated solvers exploit knowledge about the *structure* of the problem. The important class of multi-grid solvers achieve linear running times in the number of degrees of freedom by solving the same problem on grids with different step sizes and combining the approximate solutions [14].

For difference (= discrete differential) equations of elliptic type the Gauß-Seidel iteration matrices have a special eigenstructure that causes high frequencies in the error to be attenuated very quickly while for lower frequencies no practically useful rate of convergence can be observed. Multi-level schemes hence solve a given problem on a very coarse scale first. This solution is used to predict initial values for a solution of the same problem on the next refinement level. If these predicted values have only small deviations from the true solution in low-frequency sub-spaces, then Gauß-Seidel performs well in reducing the high-frequency error. The alternating refinement and smoothing leads to highly efficient variational subdivision schemes [19] which generate fair high-resolution meshes with a rate of several thousand triangles per second (linear complexity!).

As we saw in Section 2, the bottom-up way to build multi-resolution hierarchies is just one of two possibilities. To get rid of the restriction that the uniform multi-level approach to fairing cannot be applied to arbitrary meshes, we generate the hierarchy top-down by incremental mesh decimation.

A complete V-cycle multi-grid solver recursively applies operators $\Phi_i = \Psi P \Phi_{i-1} R \Psi$ where the first (right) Ψ is a generic (pre-)smoothing operator — a Gauß-Seidel scheme in our case. R is a restriction operator to go one level coarser. This is where the mesh decimation comes in. On the coarser level, the same scheme is applied recursively, Φ_{i-1}, until on the coarsest level the number of degrees of freedom is small enough to solve the system directly (or any other stopping criterion is met). On the way back-up, the prolongation operator P inserts the previously removed vertices to go one level finer again. P can be considered as a non-regular subdivision operator which has to predict the positions of the vertices in the next level's solution. The re-subdivided mesh is an approximative solution with mostly high frequency error. (Post-)smoothing by some more iterations Ψ removes the noise and yields the final solution.

Fig 7 shows the effect of multi-level smoothing. On the left you see the original bunny with about 70K triangles. In the center left,

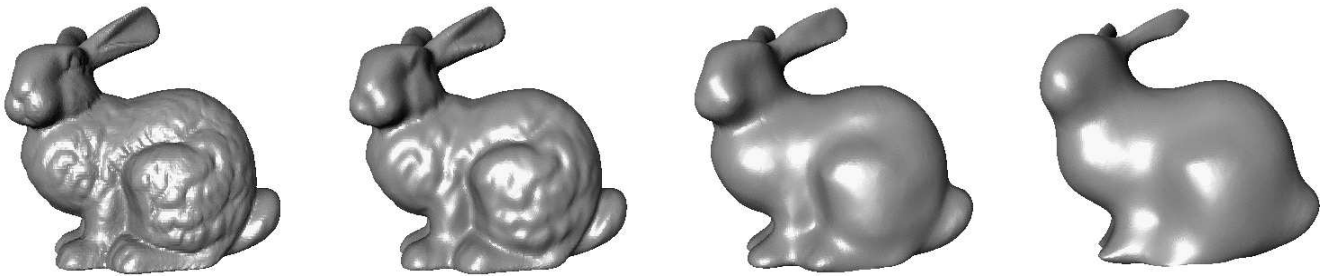

Figure 7: Four versions of the Stanford bunny. On the left the original data set. In the center left the same object after 200 iterations of the non-shrinking Laplace-filter. On the center right and far right the original data set after applying the multi-level umbrella filter with three or six levels respectively.

we applied the Laplace-filter proposed in [34] with $\lambda = 0.6307$ and $\mu = -0.6732$. The iterative application of the local smoothing operator

$$\mathbf{p}_i \leftarrow \mathbf{p}_i + [\lambda|\mu]\ \mathcal{U}(\mathbf{p}_i) \tag{10}$$

removes the highest frequency noise after a few iterations but does not have enough impact to flatten out the fur even after several hundred iterations. On the right you see the meshes after applying a multi-level smoothing with the following schedule: Hierarchy levels are generated by incremental mesh decimation where each level has about 50% of the next finer level's vertices. The pre-smoothing rule (8) is applied twice on every level before going downwards and five times after coming back up. On the center right model we computed a three level V-cycle and on the far right model a six level V-cycle. Notice that the computation time of the multi-level filters (excluding restriction and prolongation) corresponds to about $(2+5)(1+0.5+0.5^2+\ldots) < 14$ double-steps of the one-level Laplace-Filter (10).

3.4 Geometric filtering

The bunny example in Fig. 7 is well suited for demonstrating the effect of multi-level smoothing but we did not impose any boundary conditions and thus we applied the smoothing as a mere filter and not as a solving scheme for a well-posed optimization problem. This is the reason why we could use the number of levels to control the impact of the smoothing scheme on the final result. For constrained optimization, it does not make any sense to stop the recursion after a fixed number of decimation levels: we always reduce the mesh down to a small fixed number of triangles. Properly chosen boundary condition will ensure the convergence to the true solution and prevent the mesh from shrinking.

Nevertheless, we can exploit the effect observed in Fig. 7 to define more sophisticated geometric low-pass filters. Since the support of the Laplace-filters is controlled by the neighborhood relation in the underlying mesh, the smoothing characteristics are defined relative to a "topological wavelength". Noise which affects every other vertex is removed very quickly independent from the length of the edges while global distortions affecting a larger sub-mesh are hardly reduced. For *geometric* filters however we would like to set the pass-band frequency in terms of Euclidian distances by postulating that all geometric features being smaller than some threshold ε are considered as noise and should be removed.

Such filters can be implemented by using an appropriate mesh reduction scheme that tends to generate intermediate meshes with strong coherence in the length of the edges. In [20] we propose a mesh decimation scheme that rates the possible edge collapses according to some generic fairness functional. A suitable objective function for our application is to maximize the *roundness* of triangles, i.e., the ratio of its inner circle radius to its longest edge. If the mesh decimation scheme prefers those collapses that improve the global roundness, the resulting meshes tend to have only little variance in the lengths of the edges. For the bunny example, we can keep the standard deviation from the average edge length below one percent for incremental decimation down to about 5K triangles.

By selecting the lowest level $\mathcal{M}_0$ down to which the V-cycle multi-level filtering iterates, we set the threshold $\varepsilon = \varepsilon(\mathcal{M}_0)$ for detail being removed by the multi-level smoothing scheme. The thresholding works very well due to the strong local and poor global convergence of the iterative update rule (8). Fig. 8 shows the base meshes for the multi-level smoothing during the computation of the two right bunnies of Fig. 7.

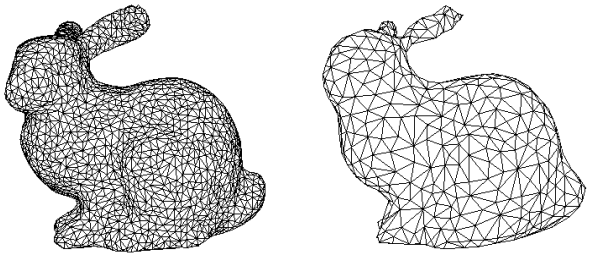

Figure 8: Base meshes where the V-cycle recursion stopped when generating the right models in Fig. 7. The final meshes do not loose significant detail (watch the silhouette). Notice how in the left example the fur is removed but the bunny's body preserved while in the right example the leg and the neck start to disappear.

4 Multi-resolution modeling on triangle meshes

In this section we describe a flexible and intuitive multi-resolution mesh modeling metaphor which exploits the techniques presented in the last two sections. As we discussed in the introduction, our goal is to get rid of topological restrictions for the mesh but also to get rid of difficulties emerging from the alignment of the basis functions in a hierarchical representation and the rigid shape of the basis function's support.

From a designer's point of view, we have to distinguish three *semantic* levels of detail. These levels are defined relative to a specific modeling operation since, of course, in a multi-resolution environment the features that are detail in a (global) modification become the global shape for a minute adjustment.

- The *global shape* is that part of the geometry that is the subject of the current modification. Intuitively, the designer selects a piece of the global shape and applies a transformation to it.
- The *structural detail* are features that are too small to be modified by hand but still represent actual geometry. This detail should follow the modification applied to the global shape in a

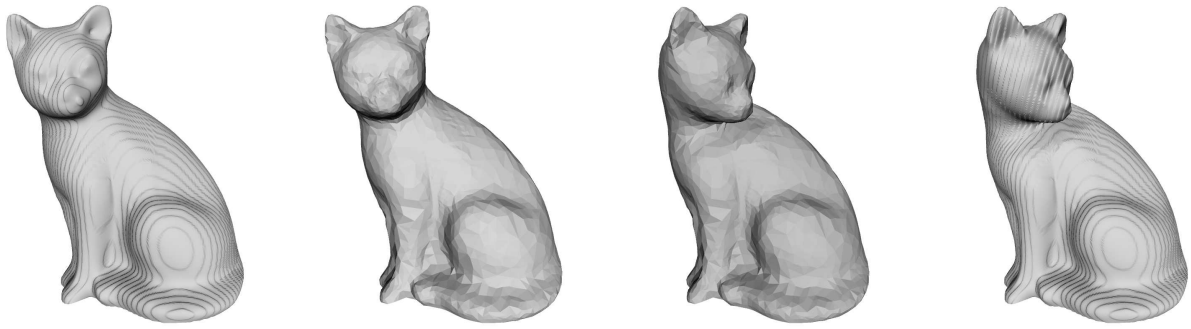

Figure 9: The wooden cat model $\mathcal{M}_k$ (178K triangles, left) is in progressive mesh representation. The high resolution is necessary to avoid alias errors in the displacement texture. The center left model $\mathcal{M}_i$ (23K triangles) is extracted by stopping the mesh reduction when a prescribed complexity is reached. On this level interactive mesh modification is done which yields the model $\mathcal{M}_i'$ (center right). The final result $\mathcal{M}_k'$ (right) is obtained by running the reconstruction on the modified mesh.

geometrically intuitive manner. The preservation of structural detail during interactive modeling is crucial for plausible visual feed-back (cf. the eyes and ears of the wooden cat model in Fig. 9).

- The *textural detail* does not really describe geometric shape. It is necessary to let the surface appear more realistic and is often represented by displacement maps [21]. In high quality mesh models it is the source for the explosive increase in complexity (cf. the wood texture in Fig. 9).

Having identified these three semantic levels of detail, we suggest a mesh modeling environment which provides flexible mesh modification functionality and allows the user to adapt the mesh complexity to the available hardware resources.

In an off-line preprocessing step, an incremental mesh decimation algorithm is applied and the detail coefficients are stored with respect to local frames as explained in Section 2.2.1. This transforms the highly complex input model into a progressive-mesh type multi-resolution representation without any remeshing or resampling. The representation allows the user to choose an appropriate number of triangles for generating a mesh model that is fine enough to contain at least all the structural detail but which is also coarse enough to be modified in realtime. This pre-process removes the textural detail prior to the actual interactive mesh modification.

Suppose the original mesh model $\mathcal{M}_k$ is transformed into the progressive mesh sequence $[\mathcal{M}_k, \ldots, \mathcal{M}_0]$ with $\mathcal{M}_0$ being the coarsest base mesh. If the user picks the mesh $\mathcal{M}_i$ and applies modifications then this invalidates the subsequence $[\mathcal{M}_{i-1}, \ldots, \mathcal{M}_0]$. If the working resolution is to be reduced afterwards to $\mathcal{M}_j$ ($j < i$) then the intermediate meshes have to be recomputed by online mesh decimation. The textural detail encoded in the subsequence $[\mathcal{M}_k, \ldots, \mathcal{M}_{i+1}]$ however remains unchanged since it is stored with respect to local frames such that reconstruction starting from a modified mesh $\mathcal{M}_i'$ leads to the intended result $\mathcal{M}_k'$. Fig. 9 shows an example of this procedure.

4.1 Interactive mesh modeling by discrete fairing

The most important feature in the proposed multi-resolution mesh modeling environment is the modification functionality itself (*modeling metaphor*) which hides the mesh connectivity to the designer.

The designer starts by marking an arbitrary region on the mesh $\mathcal{M}_i$. In fact, she picks a sequence of surface points (not necessarily vertices) on the triangle mesh and these points are connected either by geodesics or by projected lines. The strip of triangles $\mathcal{S}$ which are intersected by the geodesic (projected) boundary polygon separates an interior region $\mathcal{M}_*$ and an exterior region $\mathcal{M}_i \setminus (\mathcal{M}_* \cup \mathcal{S})$. The interior region $\mathcal{M}_*$ is to be affected by the following modification.

A second polygon (not necessarily closed) is marked within the first one to define the *handle*. The semantics of this arbitrarily shaped handle is quite similar to the handle metaphor in [37]: when the designer moves or scales the virtual tool, the same geometric transformation is applied to the rigid handle and the surrounding mesh $\mathcal{M}_*$ follows according to a constrained energy minimization principle.

The freedom to define the boundary strip $\mathcal{S}$ and the handle geometry allows the designer to build "custom tailored" basis functions for the intended modification. Particularly interesting is the definition of a *closed* handle polygon which allows to control the characteristics of a bell-shaped dent: For the same region $\mathcal{M}_*$, a tiny ring-shaped handle in the middle causes a rather sharp peak while a bigger ring causes a wider bubble (cf. Fig 10). Notice that the mesh vertices in the interior of the handle polygon move according to the energy minimization.

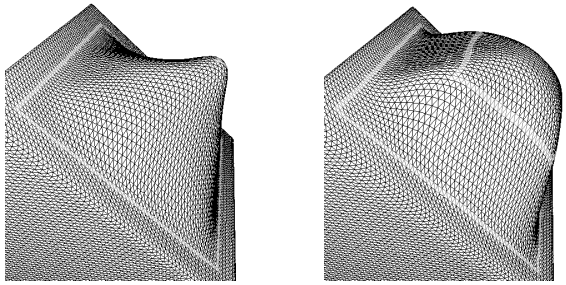

Figure 10: Controlling the characteristics of the modification by the size of a closed handle polygon.

Since we are working on triangle meshes, the energy minimization on $\mathcal{M}_*$ is done by discrete fairing techniques (cf. Section 3). The boundary triangles $\mathcal{S}$ provide the correct C^1 boundary conditions for minimizing the thin plate energy functional (3).

The handle imposes additional interpolatory constraints on the location only — derivatives should not be affect by the handle. Hence, we cannot have triangles being part of the handle geometry. We implemented the handle constraint in the following way: like the boundary polygon, the handle polygon defines a strip of triangles being intersected by it. Whether the handle polygon is open or closed, we find two polygons of mesh edges on either side of that strip. We take any one of the two polygons and collect every other mesh vertex in a set of *handle vertices*. Keeping these handle vertices fixed during the mesh optimization is the additional interpolatory constraint.

The reason for freezing only every other handle vertex is that three fixed vertices directly connected by two edges build a rigid constellation leaving no freedom to adjust the *angle* between them. During discrete optimization this would be the source of undesired artifacts in the smooth mesh.

With the boundary conditions properly set we perform the thin plate energy minimization by using the umbrella algorithm described in Section 3.1. To obtain interactive response times, we exploit the multi-level technique: a mesh decimation algorithm is applied to the mesh $\mathcal{M}_* \cup \mathcal{S}$ to build up a hierarchy. Then starting from the coarsest level, we apply the $\mathcal{U}^2$ smoothing filter alternating with mesh refinement. This process is fast enough to obtain several frames per second when modeling with meshes of $\#\mathcal{M}_* \approx 5K$ triangles (SGI R10000/195MHz). Typically, we set the ratio of the complexities between successive meshes in the hierarchy to $1:2$ or $1:4$ and iterate the smoothing filter 3 to 5 times on each level.

During the interactive mesh smoothing we do not compute the full V-cycle algorithm of Sect. 3.3. In fact, we omit the pre-smoothing and always start from the coarsest level. When a vertex is inserted during a mesh refinement step we place it initially at its neighbor's center of gravity unless the vertex is a handle vertex. Handle vertices are placed at the location prescribed by the designer's interaction (*handle interpolation constraint*). Hence the mesh is computed from scratch in every iteration instead of just updating the last position. This is very important for the modeling dialog since only the current position, orientation and scale of the handle determines the smoothed mesh and not the whole history of movements.

For the fast convergence of the optimization procedure it turns out to be important that the interpolation constraints imposed by the handle vertices show up already on rather coarse levels in the mesh hierarchy. Otherwise their impact cannot propagate far enough through the mesh such that cusps remain in the smoothed mesh which can only be removed by an excessive number of smoothing iterations. This additional requirement can easily be included into the mesh reduction scheme by lowering the priority ranking of collapses involving handle vertices.

4.2 Detail preservation

If the modified mesh $\mathcal{M}_*'$ is merely defined by constrained energy minimization, we obviously loose all the detail of the originally selected submesh $\mathcal{M}_*$. Since only the boundary and the handle vertices put constraints on the mesh, all other geometric features are smoothed out.

To preserve the detail, we use the multi-resolution representation explained in Section 2.2.2. After the boundary $\mathcal{S}$ and the handle polygon are defined but before the handle is moved by the designer, we apply the multi-level smoothing scheme once. Although the original mesh $\mathcal{M}_*$ and the smoothed mesh $\widetilde{\mathcal{M}_*}$ are topologically equivalent, they do have different levels of (geometric) resolution and hence constitute a two-scale decomposition based on varying levels of smoothness. We encode the difference $\mathcal{D}_*$ between the two meshes, i.e., the detail coefficients for the vertices $\mathbf{p}_i \in \mathcal{M}_*$ by storing the displacement vectors with respect to the local frame associated with the nearest triangle in $\widetilde{\mathcal{M}_*}$.

When the designer moves the handle, the bottom-up mesh smoothing is performed to re-adjust the mesh to the new interpolation conditions. On the resulting smooth mesh $\widetilde{\mathcal{M}_*'}$, the detail $\mathcal{D}_*$ is added and the final mesh $\mathcal{M}_*'$ is rendered on the screen. Due to the geometric coding of the detail information, this leads to intuitive changes in the surface shape (cf. Figs. 11, 12). The "frequency" of the modification is determined by the size of the area, i.e., by the boundary conditions and the fact that the *supporting mesh* is optimal with respect to the thin-plate functional.

5 Conclusions and future work

We presented a new approach to multi-resolution mesh representation and modeling which does not require the underlying triangle mesh to have subdivision connectivity. By adapting multi-level techniques known from numerical analysis to the non-regular setting of arbitrary mesh hierarchies, we are able to approximately solve constrained mesh optimization in realtime. Combining the two results allows us to present a flexible metaphor for interactive mesh modeling where the shape of the modification is controlled by energy minimization while the geometric detail is preserved and updated according to the change of the global shape.

Our current implementation of an experimental mesh modeling tool already provides sufficient functionality to apply sophisticated realtime modifications to arbitrary input meshes with up to 100K triangles. However, all changes do affect the *geometry* of the meshes only. So far we did not consider *topological* modifications of triangle meshes. In the future, when modifying a given mesh, we would like new vertices to be inserted where the mesh is locally stretched too much and, on the other hand, we would like vertices to be removed when strong global modification causes local self-intersection of the reconstructed detail.

References

[1] BONNEAU, G., HAHMANN, S., AND NIELSON, G. Blac-wavelets: a multiresolution analysis with non-nested spaces. In *Visualization Proceedings* (1996), pp. 43–48.

[2] CATMULL, E., AND CLARK, J. Rcursively Generated B-Spline Surfaces on Arbitrary Topological Meshes. *Computer Aided Design 10*, 6 (Nov. 1978), 239–248.

[3] CELNIKER, G., AND GOSSARD, D. Deformable curve and surface finite elements for free-form shape design. In *Computer Graphics (SIGGRAPH 91 Proceedings)* (July 1991), pp. 257–265.

[4] DAUBECHIES, I. *Ten Lectures on Wavelets*. CBMS-NSF Regional Conf. Series in Appl. Math., Vol. 61. SIAM, Philadelphia, PA, 1992.

[5] DOO, D., AND SABIN, M. Behaviour of recursive division surfaces near extraordinary points. *CAD* (1978).

[6] DYN, N., LEVIN, D., AND GREGORY, J. A. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Transactions on Graphics 9*, 2 (April 1990), 160–169.

[7] ECK, M., DEROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. Multiresolution Analysis of Arbitrary Meshes. In *Computer Graphics (SIGGRAPH 95 Proceedings)* (1995), pp. 173–182.

[8] FARIN, G. *Curves and Surfaces for CAGD*, 3rd ed. Academic Press, 1993.

[9] FINKELSTEIN, A., AND SALESIN, D. H. Multiresolution Curves. In *Computer Graphics (SIGGRAPH 94 Proceedings)* (July 1994), pp. 261–268.

[10] FORSEY, D. R., AND BARTELS, R. H. Hierarchical B-spline refinement. In *Computer Graphics (SIGGRAPH 88 Proceedings)* (1988), pp. 205–212.

[11] GARLAND, M., AND HECKBERT, P. S. Surface Simplification Using Quadric Error Metrics. In *Computer Graphics (SIGGRAPH 97 Proceedings)* (1997), pp. 209–218.

[12] GORTLER, S. J., AND COHEN, M. F. Hierarchical and Variational Geometric Modeling with Wavelets. In *Proceedings Symposium on Interactive 3D Graphics* (May 1995).

[13] GREINER, G. Variational design and fairing of spline surfaces. *Computer Graphics Forum 13* (1994), 143–154.

[14] HACKBUSCH, W. *Multi-Grid Methods and Applications*. Springer Verlag, Berlin, 1985.

[15] HOPPE, H. Progressive Meshes. In *Computer Graphics (SIGGRAPH 96 Proceedings)* (1996), pp. 99–108.

[16] HOSCHEK, J., AND LASSER, D. *Fundamentals of Computer Aided Geometric Design*. AK Peters, 1993.

[17] KOBBELT, L. *Iterative Erzeugung glatter Interpolanten*. Shaker Verlag, ISBN 3-8265-0540-9, 1995.

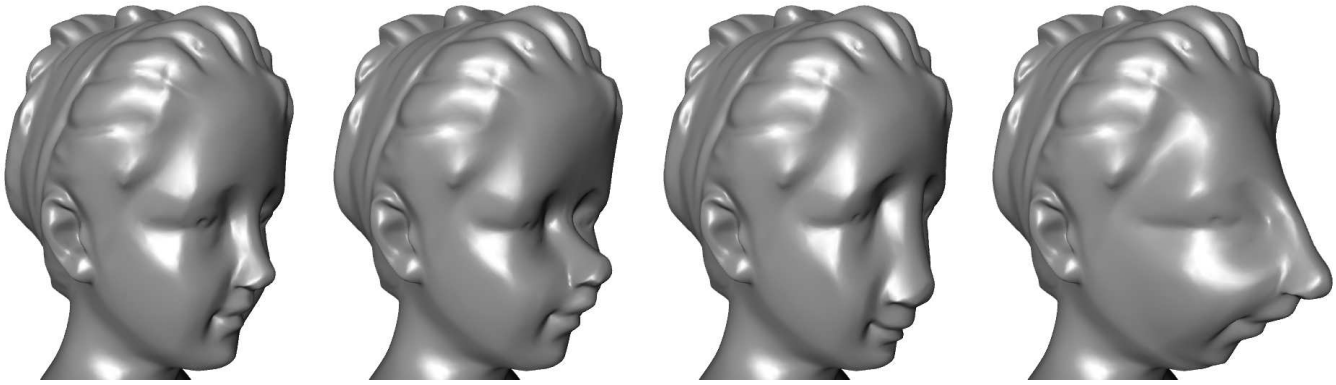

Figure 11: The mesh model of a bust (62K triangles, left, courtesy Stefan Karbacher) is modified by multi-resolution edits. The modified area $\mathcal{M}_*$ is the bust's face while the handle polygon lies around the nose. From left to right, we apply rotation, scaling and translation.

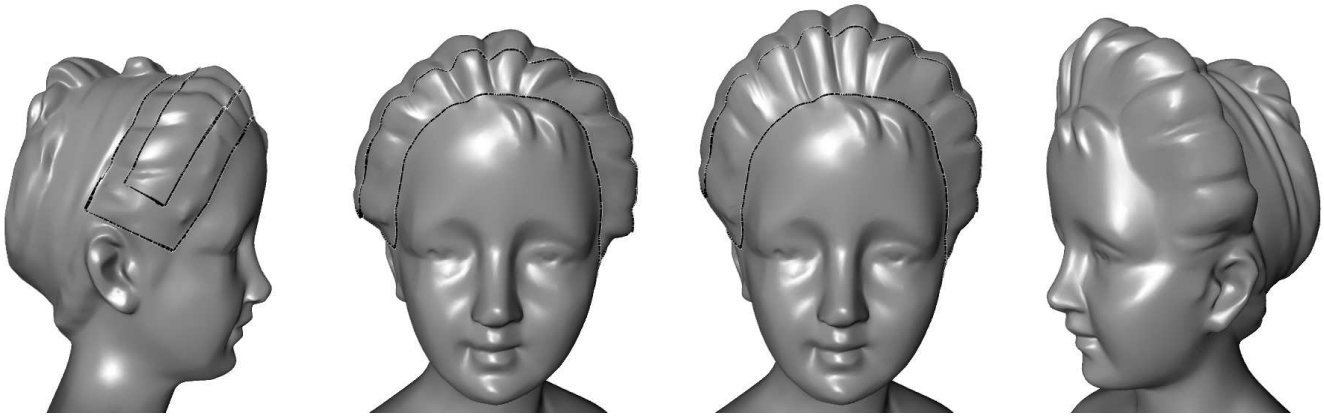

Figure 12: Some more modifications on the bust model. The support of the modification and the handle geometry adapt to the intended design operation. The detail is preserved while the global modification is controlled by discrete fairing.

[18] KOBBELT, L. Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology. In *Computer Graphics Forum, Proceedings of Eurographics '96* (1996), pp. C407–C420.

[19] KOBBELT, L. Discrete fairing. In *Proceedings of the Seventh IMA Conference on the Mathematics of Surfaces* (1997), pp. 101–131.

[20] KOBBELT, L., CAMPAGNA, S., AND SEIDEL, H.-P. A general framework for mesh decimation. In *Proceedings of the Graphics Interface conference '98* (1998).

[21] KRISHNAMURTHY, V., AND LEVOY, M. Fitting smooth surfaces to dense polygon meshes. In *Computer Graphics (SIGGRAPH 96 Proceedings)* (1996), pp. 313–324.

[22] LOOP, C. Smooth spline surfaces over irregular meshes. In *Computer Graphics Proceedings* (1994), Annual Conference Series, ACM Siggraph, pp. 303–310.

[23] LOUNSBERY, M., DEROSE, T., AND WARREN, J. Multiresolution Analysis for Surfaces of Arbitrary Topological Type. *ACM Transactions on Graphics 16*, 1 (January 1997), 34–73.

[24] LUEBKE, D., AND ERIKSON, C. View-Dependent Simplification of Arbitrary Polygonal Environments. In *Computer Graphics (SIGGRAPH 97 Proceedings)* (1997), pp. 199–208.

[25] MORETON, H., AND SÉQUIN, C. Functional optimization for fair surface design. In *Computer Graphics (SIGGRAPH 92 Proceedings)* (1992), pp. 167–176.

[26] OPPENHEIM, A., AND SCHAFER, R. *Discrete-Time Signal Processing*. Prentice Hall, 1989.

[27] ROSSIGNAC, J. Simplification and Compression of 3D Scenes, 1997. Tutorial Eurographics '97.

[28] SAPIDIS, N. E. *Designing Fair Curves and Surfaces*. SIAM, 1994.

[29] SCHABAK, R., AND WERNER, H. *Numerische Mathematik*. Springer Verlag, 1993.

[30] SCHRÖDER, P., AND SWELDENS, W. Spherical wavelets: Efficiently representing functions on the sphere. In *Computer Graphics (SIGGRAPH 95 Proceedings)* (1995), pp. 161–172.

[31] SCHROEDER, W. J., ZARGE, J. A., AND LORENSEN, W. E. Decimation of Triangle Meshes. In *Computer Graphics (SIGGRAPH 92 Proceedings)* (1992), pp. 65–70.

[32] STOER, J. *Einführung in die Numerische Mathematik I*. Springer Verlag, 1983.

[33] STOLLNITZ, E., DEROSE, T., AND SALESIN, D. *Wavelets for Computer Graphics*. Morgan Kaufmann Publishers, 1996.

[34] TAUBIN, G. A signal processing approach to fair surface design. In *Computer Graphics (SIGGRAPH 95 Proceedings)* (1995), pp. 351–358.

[35] TURK, G. Re-Tiling Polygonal Surfaces. In *Computer Graphics (SIGGRAPH 92 Proceedings)* (1992), pp. 55–64.

[36] WARREN, J. Subdivision schemes for variational splines, 1997. Preprint.

[37] WELCH, W., AND WITKIN, A. Variational surface modeling. In *Computer Graphics (SIGGRAPH 92 Proceedings)* (1992), pp. 157–166.

[38] WELCH, W., AND WITKIN, A. Free–Form shape design using triangulated surfaces. In *Computer Graphics (SIGGRAPH 94 Proceedings)* (1994), A. Glassner, Ed., pp. 247–256.

[39] ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. Interpolating subdivision for meshes with arbitrary topology. In *Computer Graphics (SIGGRAPH 96 Proceedings)* (1996), pp. 189–192.

[40] ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. Interactive multiresolution mesh editing. In *Computer Graphics (SIGGRAPH 97 Proceedings)* (1997), pp. 259–268.