ELSEVIER

Computational Geometry 14 (1999) 5–24

Computational Geometry
Theory and Applications

www. elsevier.nl/locate/comgeo

Multiresolution hierarchies on unstructured triangle meshes

Leif Kobbelt *, Jens Vorsatz, Hans-Peter Seidel

Computer Graphics Group, Max-Planck-Institut für Informatik, Stadtwald, 66123 Saarbrücken, Germany

Abstract

The use of polygonal meshes for the representation of highly complex geometric objects has become the de facto standard in most computer graphics applications. Especially triangle meshes are preferred due to their algorithmic simplicity, numerical robustness, and efficient display. The possibility to decompose a given triangle mesh into a hierarchy of differently detailed approximations enables sophisticated modeling operations like the modification of the global shape under preservation of the detail features. So far, multiresolution hierarchies have been proposed mainly for meshes with subdivision connectivity. This type of connectivity results from iteratively applying a uniform split operator to an initially given coarse base mesh. In this paper we demonstrate how a similar hierarchical structure can be derived for arbitrary meshes with no restrictions on the connectivity. Since smooth (subdivision) basis functions are no longer available in this generalized context, we use constrained energy minimization to associate *smooth* geometry with *coarse* levels of detail. As the energy minimization requires one to solve a global sparse system, we investigate the effect of various parameters and boundary conditions in order to optimize the performance of iterative solving algorithms. Another crucial ingredient for an effective multiresolution decomposition of unstructured meshes is the flexible representation of detail information. We discuss several approaches. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Geometric modeling; Multiresolution hierarchies; Discrete fairing; Triangle meshes

1. Introduction

Subdivision techniques provide very efficient and flexible algorithms for the generation of free form surface geometry [2,5,6,18,25,39]. Starting with an arbitrary control mesh $\mathcal{M}_0$ we can apply the subdivision rules to compute finer and finer meshes $\mathcal{M}_m$ with control vertices $\boldsymbol{p}_i^m$ becoming more and more dense until the desired approximation tolerance required for a given application is reached. The result is a smooth surface having the same topology as the initial control mesh.

The distinct subdivision levels $\mathcal{M}_m$ give rise to powerful multiresolution semantics since we can consider a subdivision scheme as the low pass reconstruction operator in the filter bank algorithm for a wavelet-type decomposition of the geometric shape. The subdivision basis functions which are associated

[*] Corresponding author. E-mail: kobbelt@mpi-sb.mpg.de

PII: S0925-7721(99)00032-2

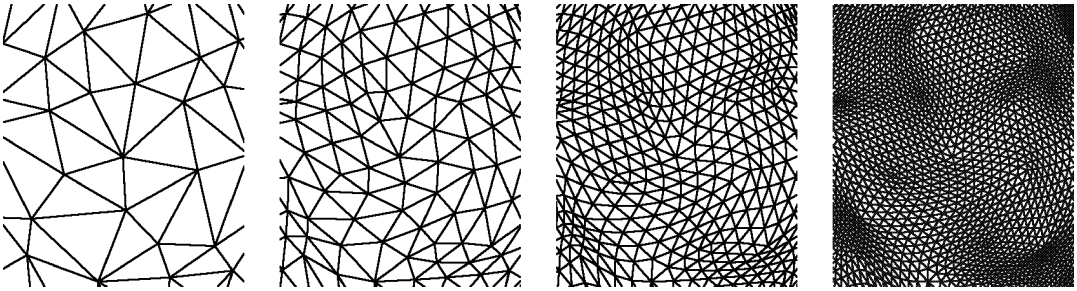

Fig. 1. Subdivision connectivity meshes result from iteratively applying a uniform split operation to the faces of an initial control mesh. Only a fixed number of isolated extraordinary vertices with valence $\neq 6$ remain in the mesh.

with the control vertices generalize the concept of dyadic scaling functions to polyhedral parameter domains [26,31,40].

However, subdivision techniques are genuinely based on the *coarse-to-fine* generation of hierarchical geometry representations: a coarse base mesh with only few faces is iteratively refined by introducing an exponentially increasing number of degrees of freedom for capturing finer and finer detail information. As a consequence, the control meshes must have so-called *subdivision connectivity* which means that sub-regions of the refined mesh $\mathcal{M}_m$ which correspond to one single face of the original base mesh $\mathcal{M}_0$, have the connectivity of regular grids (cf. Fig. 1).

It turns out that this restriction is not suitable for several standard application scenarios. In practice one is often given some *existing* geometric model which is to be modified by making local or global adjustments. Since such triangular meshes usually do not come with the rather special subdivision connectivity, we cannot apply subdivision techniques without preprocessing.

This preprocessing has to perform a global remeshing of the data. Although several flexible and robust algorithms have been proposed for this problem [7,22,24] there are still difficulties with automatically finding a suitable layout for the base mesh. Semi-automatic approaches like [23,24] with constraints set by the user only partially solve this problem. Moreover, the remeshing is *always* a resampling process and hence even an optimal remeshing algorithm cannot recover the original shape exactly. High frequency artifacts due to alias errors are rather likely to appear.

The rigidity of subdivision connectivity meshes emerges from the fact that the classification of the detail coefficients into predefined refinement levels is done topologically. The actual *size* or *geometric frequency* associated with a detail coefficient hence strongly depends on the size of the corresponding base triangle in the unrefined control mesh. As it is usually not possible to have all triangles in the base mesh of unit size, detail features on the same refinement level and their corresponding support can vary by one or more orders of magnitude. Avoiding this problem by using adaptive refinement strategies is not appropriate in some applications.

Another problem which is inherent to the multiresolution representation of free form geometry based on subdivision surfaces is the *fixed support* of the modifications. If control vertices are used as handles to modify the surface geometry on a certain level of detail then the region of the mesh which actually changes, is determined by the support of the associated basis function. We could simulate more flexibility in the definition of the support by moving several control vertices from some finer level simultaneously but this would diminish the advantages of a multiresolution representation.

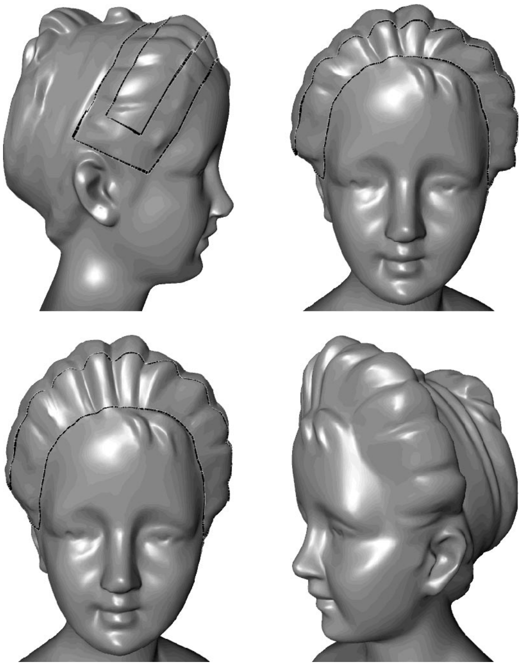

Fig. 2. In a multiresolution modeling environment, the support of the modification and its characteristic shape should adapt to the given geometry (here: the bust's hair). The low-frequency modification affects exactly the region defined by the designer. The high frequency detail is preserved in a natural way.

Moreover, the coarse scale control vertices in a subdivision representation are *aligned* to the coarse scale grid. This means that we lose spatial resolution if we modify a surface on a low frequency band. Consequently, we can apply modifications of the global shape only at a very limited number of locations. In fact, as every control vertex $\boldsymbol{c}$ in a subdivision connectivity mesh is introduced on a certain refinement level $l(\boldsymbol{c})$ the support of the modification when moving $\boldsymbol{c}$ is bounded by the size of the basis functions on that level.

For example, if we move a control vertex $\boldsymbol{c}_0$ which topologically corresponds to a vertex in the base mesh then we can choose the basis function controlling the edit from any refinement level. However, moving a directly adjacent vertex $\boldsymbol{c}_m$ on refinement level m can only affect the finest scale since $\boldsymbol{c}_m$ does not have a representation on any coarser level. Hence, a coarse scale modification can be centered at $\boldsymbol{c}_0$ but not at $\boldsymbol{c}_m$ which lies only ε away. This is not intuitive for the designer to whom the actual surface representation should be opaque.

With all these difficulties enumerated, we understand that coarse-to-fine hierarchies emerging from subdivision techniques might certainly be the best way to effectively represent smooth free form geometry in applications like surface reconstruction, scattered data interpolation, or *ab initio* design where the face layout for the base mesh is defined by the designer. However, it does not appear to be the optimal solution for flexibly modifying *existing* models like the ones obtained from capturing real object geometry by laser scanning devices.

Our goal is to enable true free form multiresolution edits where the support and the characteristics of a modification can adapt to the surface geometry (cf. Fig. 2). In [21] we generalized the concept of

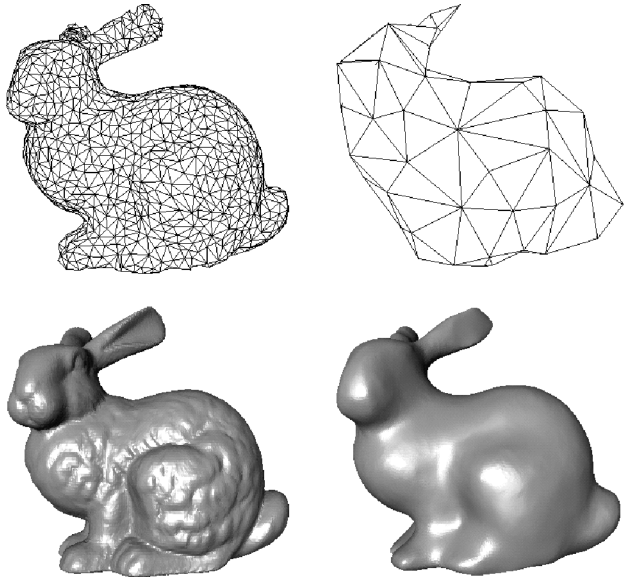

Fig. 3. For plain triangle meshes we have to distinguish coarse and smooth approximations (upper and lower row). If meshes are considered as control meshes with respect to scalar valued basis functions then the connection between the upper an the lower row is provided by evaluating the weighted superposition of the control vertices' influence.

multiresolution decomposition and modeling to meshes with arbitrary topology and connectivity. The key observation is that we can no longer stick to the notion of surface geometry being represented by the superposition of smooth scalar valued basis functions over a nested sequence of grids. The reason for this is that we cannot make any assumptions on the actual distribution of the mesh vertices a priori. Hence, imposing any kind of vector space structure would require us to construct explicitly a custom tailored basis function for each vertex.

Leaving the classical set-up, it turns out that for mere polygonal meshes (not control meshes), *coarseness* and *smoothness* are no longer synonyms. While in the subdivision framework the basis functions on the coarse scales are also smoother in the sense that they have less curvature, we find that for plain polygonal meshes the effect of shifting a control vertex on a coarse scale still causes a sharp feature. To speak about *smooth* polygonal meshes we need more degrees of freedom since smooth meshes are typically rather fine tesselations.

We have to solve two central problems in order to develop effective multiresolution algorithms for arbitrary meshes. First we have to construct a *topological hierarchy* of different resolutions with the finest resolution being the original mesh. This hierarchy must not rely on any assumptions about the connectivity of the given mesh.

Besides the topological levels of detail we need a *geometric hierarchy*, i.e., we need a proper characterization of *smooth* coarse-scale geometry. In the subdivision based multiresolution setting, we have the associated scaling functions which fill in the smooth geometry between the coarse scale control vertices. In the generalized setting we have to find an alternative definition since a priori defined scaling functions are no longer available. A possible solution to this problem is to use discrete energy minimization techniques to obtain smooth low-detail approximations to the original model.

While the basic principles of this approach have been presented in [21], we discuss more technical details in this paper. After explaining the generation of coarse-to-fine hierarchies and fine-to-coarse hierarchies, we compare different ways to represent the detail information between the resolution levels. The crucial issues are here how to define the local frames with respect to which the detail is encoded and how to chose the number of hierarchy levels. In the context of discrete energy minimization we investigate the effect of various parameters in the multi-level solving algorithm, namely the number of hierarchy levels and the number of Gauß–Seidel iterations on each level. We demonstrate that imposing interpolation constraints at the centers of the triangular faces accelerates the global convergence of the iterative solver compared to imposing the constraints at the vertices.

2. Multiresolution representations

Most schemes for the multiresolution representation and modification of triangle meshes emerge from generalizing harmonic analysis techniques like the wavelet transform [1,26,31,34]. Since the fundamentals are 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 multiresolution 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 [8,13,40]. 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, i.e., the mesh has to consist of large regular regions with isolated extra-ordinary vertices. 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. Obviously, sophisticated modeling operations like *boolean operations* necessarily require a complete restructuring of the resulting mesh to re-establish subdivision connectivity.

These special topological requirements prevent such techniques from being applicable to arbitrary input meshes. To obtain a proper hierarchy, global remeshing and resampling is necessary which gives rise to alias-errors and requires involved computations [7,24].

Luckily, the restricted connectivity 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 [12,17,20,27,29,32,36]. The key is always to build the hierarchy top-down by eliminating vertices from the current mesh (*incremental reduction*, cf. Fig. 4). 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 [17,26].

In both cases, i.e., the coarse-to-fine or the fine-to-coarse generation of nested (vertex-) grids, the multiresolution concept is rigidly attached to topological entities. This makes sense if hierarchies are merely used to adjust the complexity of the representation. We will exploit the sequence of nested grids emerging from this topological hierarchy to generalize the concept of multi-grid solvers for large sparse systems.

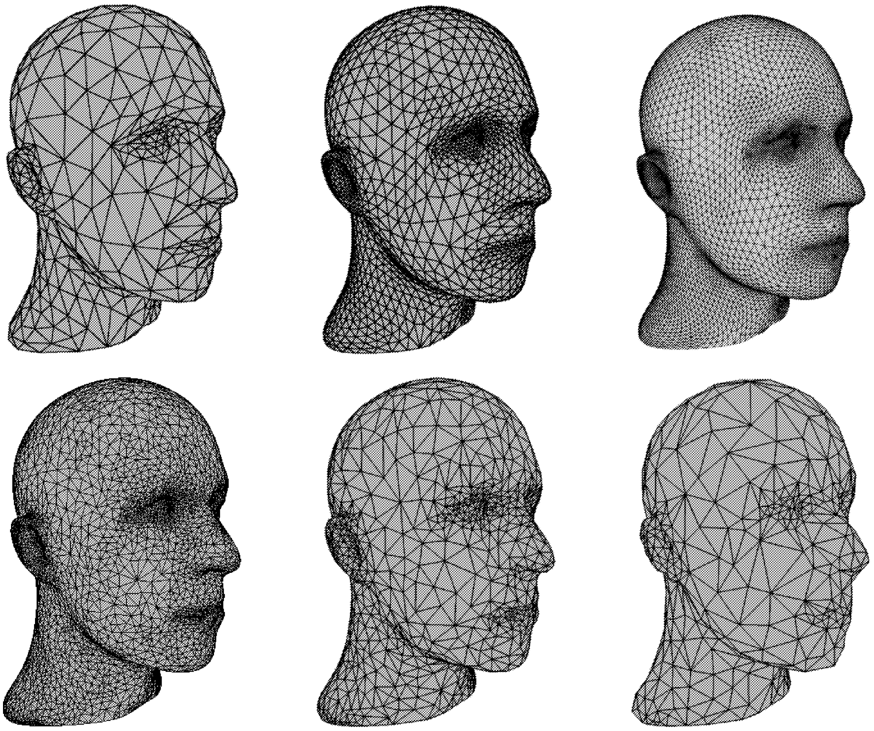

Fig. 4. For multiresolution representations based on subdivision techniques, the hierarchies are built from coarse to fine by applying a uniform subdivision operator (top row, left to right) while incremental mesh decimation generates hierarchies from fine to coarse by iteratively removing vertices (bottom row, left to right).

In the context of multiresolution *modeling*, however, we want the hierarchy not necessarily to rate meshes according to their *coarseness* but rather according to their *smoothness*. For this we need a geometric hierarchy accompanying the topological one. To complete our basic equipment for the multiresolution set-up on unstructured meshes we hence need (besides the static levels of detail) 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. Here, the reconstruction operator has to generate the smooth low-frequency shape if the detail information is suppressed during reconstruction. This is where discrete fairing techniques come in. Further, we have to encode the detail information relative to the low-frequency shape in order to guarantee intuitive detail preservation after a global modification (*local frames*).

2.1. Coarse-to-fine hierarchies

For subdivision based multiresolution representation the reconstruction operator is given by the underlying subdivision scheme. We transform a given mesh $\mathcal{M}_m$ to the next refinement level $\mathcal{M}'_{m+1} = \boldsymbol{S}\mathcal{M}_m$ by applying the stationary subdivision operator $\boldsymbol{S}$ and move the obtained control vertices by adding the associated detail vectors: $\mathcal{M}_{m+1} = \mathcal{M}'_{m+1} + \mathcal{D}_m$. The support of the subdivision mask implies that each control vertex $\boldsymbol{p}_i^m$ in $\mathcal{M}_m$ has influence on several control vertices in $\mathcal{M}'_{m+1}$. Consequently, the modification of $\boldsymbol{p}_i^m$'s position eventually causes a smooth bump on the resulting surface. The actual shape of this bump can be computed by applying the subdivision operator $\boldsymbol{S}$ *without* detail reconstruction, i.e., $\mathcal{D}_m := 0$. Obviously, the support of the bump depends on the refinement level m on which the modification is applied.

The decomposition operator has to be an inverse of the subdivision operator, i.e., given a fine mesh $\mathcal{M}_{m+1}$ we have to find a mesh $\mathcal{M}_m$ such that $\mathcal{M}_{m+1} \approx \boldsymbol{S}\mathcal{M}_m$. In this case the detail vectors $\mathcal{D}_m := \mathcal{M}_{m+1} - \boldsymbol{S}\mathcal{M}_m$ become as small as possible [40]. Due to the uniform split which is part of the subdivision operator $\boldsymbol{S}$, it is obvious that this technique applies only if $\mathcal{M}_{m+1}$ has subdivision connectivity.

2.2. *Fine-to-coarse hierarchies*

If we build the hierarchy by using an incremental mesh decimation scheme, the decomposition operator $\boldsymbol{D}$ applies to arbitrary meshes. Given a fine mesh $\mathcal{M}_{m+1}$ we find $\mathcal{M}_m = \boldsymbol{D}\mathcal{M}_{m+1}$, e.g., by applying a number of edge collapse operations. However, it is not clear how to define the detail coefficients since inverse mesh decimation (*progressive meshes*) always reconstructs the original mesh and there is no canonical way to generate smooth low-frequency geometry by suppressing the detail information during reconstruction.

To solve this problem we split each step of the progressive mesh refinement into a topological operation (vertex insertion) and a geometric operation which places the re-inserted vertices at their original position. In analogy to the plain subdivision without detail reconstruction, we have to figure out a heuristic which places the new vertices such that they lie on a smooth surface (instead of their original position). The difference vector between this predicted position and the original location of the vertex can then be used as the associated detail vector.

Since we operate on unstructured meshes, we cannot use fixed (stationary) rules for the placement of the re-inserted vertices. Instead we use discrete energy minimization which means that the re-inserted vertices are placed such that some global bending energy is minimized. In Section 3 we review a simple technique for the effective generation of meshes with minimum bending energy without specific requirements on the connectivity.

2.3. *Detail encoding*

In order to guarantee intuitive detail preservation under modification of the global shape, we cannot simply store the detail vectors with respect to a global coordinate system but have to define them with respect to local frames which are aligned to the low-frequency geometry [10,11]. Usually, the associated local frame for each vertex has its origin at the location predicted by the reconstruction operator with suppressed detail. This is in analogy to decompositions based on a global parameterization of the surfaces.

However, in many cases this can lead to rather long detail vectors with a significant component within the local tangent plane (cf. Fig. 5). Since we prefer short detail vectors for stability reasons, it makes sense to use a different origin for the local frame. In fact, the optimal choice is to find that point on the low-frequency surface whose normal vector points directly to the original vertex. In this case, the detail is not given by a three dimensional vector $(\Delta x, \Delta y, \Delta z)^{\mathrm{T}}$ but rather by a base point $\boldsymbol{p} = \boldsymbol{p}(u, v)$ on the low-frequency geometry plus a scalar value h for the displacement in normal direction. If a local

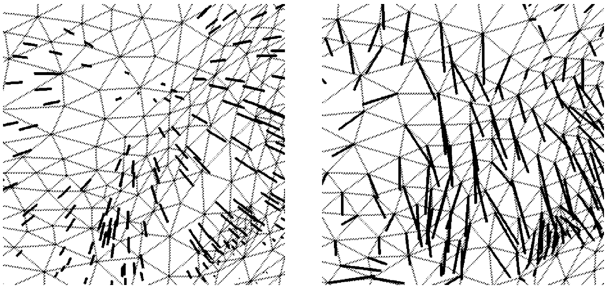

Fig. 5. The shortest detail vectors are obtained by representing the detail coefficients with respect to the nearest local frame (left) instead of attaching the detail vectors to the topologically corresponding original vertices.

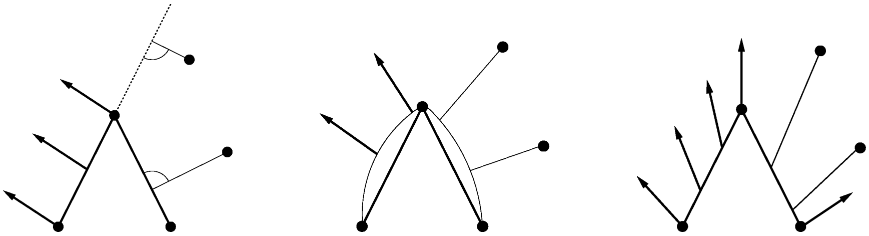

Fig. 6. The position of a vertex in the original mesh (high-frequency geometry) is given by a base point on the low-frequency geometry plus a displacement in normal direction. There are many ways to define a normal field on a triangle mesh. With piecewise constant normals (left) we do not cover the whole space and hence we sometimes have to use virtual base points with negative barycentric coordinates. The use of local quadratic patches and their normal fields (center) somewhat improves the situation but problems still occur since the overall normal field is not globally continuous. Such difficulties are completely avoided if we generate a Phong-type normal field by blending estimated vertex normals (right).

parameterization of the surface is available then the base point $\boldsymbol{p}$ can be specified by a two-dimensional parameter value (u, v).

The general setting for detail computation is that we have given two meshes $\mathcal{M}_{m+1}$ and $\mathcal{M}'_{m+1}$ where $\mathcal{M}_{m+1}$ is the original data while $\mathcal{M}'_{m+1}$ is reconstructed from the low-frequency approximation $\mathcal{M}_m$ with suppressed detail, i.e., for coarse-to-fine hierarchies, the mesh $\mathcal{M}'_{m+1}$ is generated by applying a stationary subdivision scheme and for fine-to-coarse hierarchies $\mathcal{M}'_{m+1}$ is optimal with respect to some global bending energy functional. Encoding the geometric difference between both meshes requires us to associate each vertex $\boldsymbol{p}$ of $\mathcal{M}_{m+1}$ with a corresponding base point $\boldsymbol{q}$ on the continuous (piecewise linear) surface $\mathcal{M}'_{m+1}$ such that the difference vector between the original point and the base point is parallel to the normal vector at the base point. Any point $\boldsymbol{q}$ on $\mathcal{M}'_{m+1}$ can be specified by a triangle index i and barycentric coordinates within the referred triangle.

To actually compute the detail coefficients, we have to define a normal field on the mesh $\mathcal{M}'_{m+1}$. The most simple way to do this is to use the normal vectors of the triangular faces for the definition of a piecewise constant normal field. However, since the orthogonal prisms spanned by a triangle mesh do not completely cover the vicinity of the mesh, we have to accept negative barycentric coordinates for the base points if an original vertex lies close to an edge of $\mathcal{M}'_{m+1}$ or if $\mathcal{M}'_{m+1}$ is not smooth enough (cf. Fig. 6). This leads to non-intuitive detail reconstruction if the low-frequency geometry is modified (cf. Fig. 7).

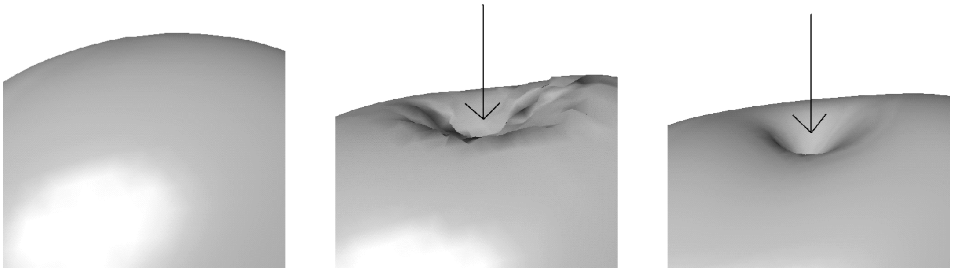

Fig. 7. We modified the original surface (left) by using a two-band multiresolution decomposition. Since in this particular experiment the low-frequency geometry was chosen not sufficiently smooth, many detail vectors have base points with negative barycentric coordinates when we use a piecewise constant normal field. Consequently, no proper detail reconstruction is possible after the modification (center). Representing the detail vectors with respect to the Phong normal field on the low-frequency mesh leads to the expected result (right).

A technique used in [21] is based on the construction of a local quadratic interpolant to the low-frequency geometry. The base point is found by Newton-iteration. Although this technique reduces the number of pathological configurations with negative barycentric coordinates for the base point, we still observe artifacts in the reconstructed high-frequency surface which are caused by the fact that the resulting global normal field of the combined local patches is not continuous.

We therefore propose a different approach which adapts the basic idea of Phong-shading [9] where normal vectors are estimated at the vertices of a triangle mesh and a continuous normal field for the interior of the triangular faces is computed by linearly blending the normal vectors at the corners.

Suppose we are given a triangle $\Delta(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ with the associated normal vectors $N_{\boldsymbol{a}}$, $N_{\boldsymbol{b}}$ and $N_{\boldsymbol{c}}$. For each interior point

$$\boldsymbol{q} = \alpha \boldsymbol{a} + \beta \boldsymbol{b} + \gamma \boldsymbol{c}$$

with $\alpha + \beta + \gamma = 1$ we find the associated normal vector $N_{\boldsymbol{q}}$ by

$$N_{\boldsymbol{q}} = \alpha N_{\boldsymbol{a}} + \beta N_{\boldsymbol{b}} + \gamma N_{\boldsymbol{c}}.$$

When computing the detail coefficients for a given point $\boldsymbol{p}$ we have to find the base point $\boldsymbol{q}$ such that

$$(\boldsymbol{p} - \boldsymbol{q}) \times N_{\boldsymbol{q}}$$

has all three coordinates vanishing. By plugging in the definition of $\boldsymbol{q}$ and $N_{\boldsymbol{q}}$ and eliminating $\gamma = 1 - \alpha - \beta$ we obtain a bivariate quadratic function

$$F : (u, v) \to \mathbb{R}^3$$

and we have to find the parameter value (α, β) such that $F(\alpha, \beta) = (0, 0, 0)^{\mathrm{T}}$. This can be accomplished by performing several steps of Newton-iteration. Notice that F can be interpreted as a quadratic surface patch in $\mathbb{R}^3$ which passes through the origin. The Taylor-coefficients of F can explicitly be given by

$$\begin{aligned}
F &= W + WW, \\
F_u &= U + UW - W - 2WW, \\
F_v &= V + VW - W - 2WW, \\
F_{uu} &= UU - UW + WW,
\end{aligned}$$

$$F_{uv} = UV - UW - VW + 2WW,$$
$$F_{vv} = VV - VW + WW,$$

where

$$U = \boldsymbol{p} \times N_{\boldsymbol{a}},$$
$$V = \boldsymbol{p} \times N_{\boldsymbol{b}},$$
$$W = \boldsymbol{p} \times N_{\boldsymbol{c}},$$
$$UU = N_{\boldsymbol{a}} \times \boldsymbol{a},$$
$$VV = N_{\boldsymbol{b}} \times \boldsymbol{b},$$
$$WW = N_{\boldsymbol{c}} \times \boldsymbol{c},$$
$$UV = (N_{\boldsymbol{b}} \times \boldsymbol{a}) + (N_{\boldsymbol{a}} \times \boldsymbol{b}),$$
$$UW = (N_{\boldsymbol{c}} \times \boldsymbol{a}) + (N_{\boldsymbol{a}} \times \boldsymbol{c}),$$
$$VW = (N_{\boldsymbol{c}} \times \boldsymbol{b}) + (N_{\boldsymbol{b}} \times \boldsymbol{c}).$$

In case one of the barycentric coordinates of the resulting point $\boldsymbol{q}$ is negative, we continue the search for a base point in the corresponding neighboring triangle. Since the Phong normal field is globally continuous we always find a base point with positive barycentric coordinates. Fig. 6 depicts the situation schematically and Fig. 7 shows an example edit where the piecewise constant normal field causes mesh artifacts which do not occur if the Phong normal field is used.

2.4. Hierarchy levels

For coarse-to-fine hierarchies the levels of detail are determined by the uniform refinement operator. Starting with the base mesh $\mathcal{M}_0$, the mth refinement level is reached after applying the refinement operator m times. For fine-to-coarse hierarchies there is no such canonical choice for the levels of resolution. Hence we have to figure out some heuristics to define such levels.

In [21] a simple two-band decomposition has been proposed for the modeling, i.e., the high frequency geometry is given by the original mesh and the low-frequency geometry is the solution of some constrained optimization problem. This simple decomposition performs well if the original geometry can be projected onto the low-frequency geometry without self-intersections. Fig. 8 schematically shows a configuration where this requirement is not satisfied and consequently the detail feature does not deform intuitively with the change of the global shape.

This effect can be avoided by introducing several intermediate levels of detail, i.e., by using a true multi-band decomposition. The number of hierarchy levels has to be chosen such that the $(i + 1)$st level can be projected onto level i without self-intersection. Detail information has to be computed for every intermediate level.

The intermediate levels can be generated by the following algorithm. We start with the original mesh and apply an incremental mesh decimation algorithm which performs a sequence of edge collapse operations. When a certain mesh complexity is reached, we perform the reverse sequence of vertex split operations which reconstructs the original mesh connectivity. The position of the reinserted vertices is found by solving a global bending energy minimization problem (discrete fairing). The mesh that results from this procedure is a smoothed version of the original mesh where the degree by which detail information has been removed depends on the target complexity of the decimation algorithm (cf. Fig. 10).

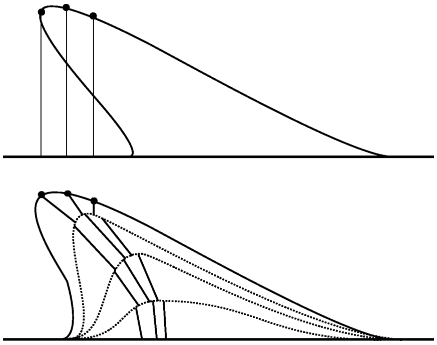

Fig. 8. When the difference between two geometric levels of detail is too big, the high-frequency geometry cannot be projected directly onto the low-frequency geometry without self-intersections. In order to guarantee correct detail reconstruction, we have to generate intermediate levels such that the mapping between two successive levels is one-to-one.

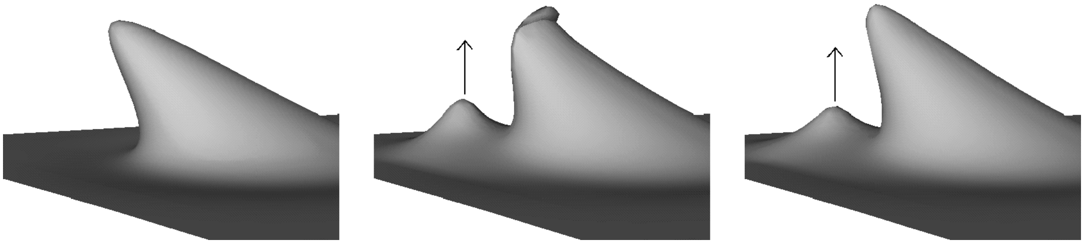

Fig. 9. Non-projectable detail features are not reconstructed correctly. The original geometry (left) is modified by using a two-band decomposition in the center and a multi-band decomposition with five intermediate levels on the right.

Suppose the original mesh has n_m vertices, where m is the number of intermediate levels that we want to generate. We can compute the meshes $\mathcal{M}_m, \ldots, \mathcal{M}_0$ with fewer detail by applying the above procedure where the decimation algorithm stops at a target resolution of $n_m, \ldots, n_0$ remaining vertices respectively. The resulting meshes yield a multi-band decomposition of the original data. When a modeling operation changes the shape of $\mathcal{M}_0$ we first reconstruct the next level $\mathcal{M}'_1$ by adding the stored detail vectors and then proceed by successively reconstructing $\mathcal{M}'_{i+1}$ from $\mathcal{M}'_i$.

The remaining question is how to determine the numbers n_i. A simple way to do this is to build a geometric sequence with $n_{i+1}/n_i = \text{const}$. This mimics the exponential complexity growth of the coarse-to-fine hierarchies. Another approach is to stop the decimation every time a certain average edge length $\bar{l}_i$ in the remaining mesh is reached.

A more complicated heuristic tries to equalize the sizes of the differences between levels, i.e., the sizes of the detail vectors. We first compute a multi-band decomposition with, say, 100 levels of detail where we choose $\sqrt[i]{\overline{n}_i} = \text{const}$. For every pair of successive levels we can compute the average length of the detail vectors (displacement values). From this information we can easily choose appropriate values $n_j = \overline{n}_{i_j}$ such that the geometric difference is distributed evenly among the detail levels.

In practice it turned out that about five intermediate levels is usually enough to guarantee correct detail reconstruction. Fig. 9 compares the results of a modeling operation based on a two-band and a multi-band decomposition.

3. Constrained discrete fairing

In the previous section we explained how to generate topological hierarchies for meshes with arbitrary connectivity by incremental mesh decimation. An associated geometric hierarchy can be obtained by re-inserting the removed vertices and moving them to a new position such that a global bending energy functional is minimized. The idea is to compute a mesh which is as smooth as possible while still containing a controllable amount of geometric detail. Fig. 10 shows an example.

From CAGD it is well known that constrained energy minimization is a very powerful technique to generate high quality surfaces [3,14,28,30,37]. 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.

For the most popular fairing functional, the *thin-plate energy*, this approach leads to a simple update-rule [21]

$$\boldsymbol{p} \leftarrow \boldsymbol{p} - \frac{1}{v}\mathcal{U}^2(\boldsymbol{p}), \tag{1}$$

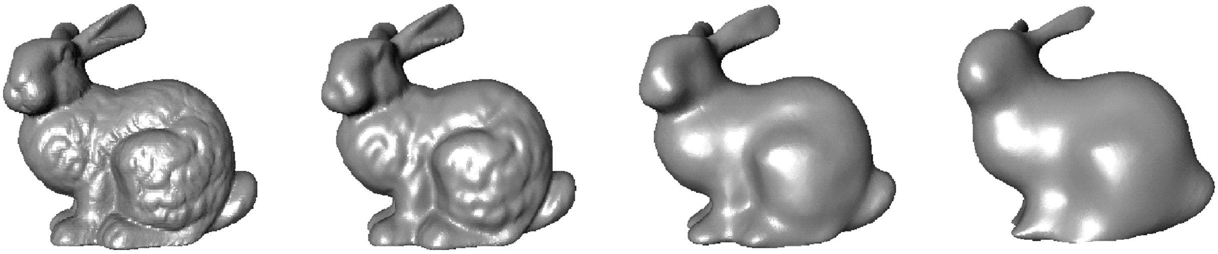

Fig. 10. Four versions of the Stanford bunny. The smoother versions are generated by applying mesh decimation down to a certain target complexity and then re-inserting the vertices under minimization of some discrete fairness functional. The degree by which geometric detail is removed depends on the coarseness of the base mesh. Notice that all shown meshes have exactly the same connectivity.

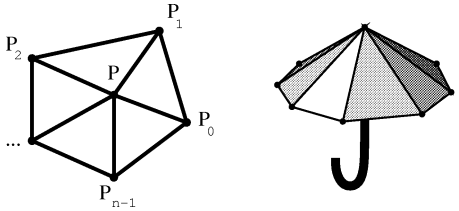

Fig. 11. To compute the discrete Laplacian, we need the 1-neighborhood of a vertex $\boldsymbol{p}$ (→ *umbrella-operator*).

which has to be applied to all vertices of the mesh. Here, the umbrella-operator $\mathcal{U}$ is a discretization of the Laplace-operator [35]

$$\mathcal{U}(\boldsymbol{p}) = \frac{1}{n}\sum_{j=0}^{n-1} \boldsymbol{p}_j - \boldsymbol{p},$$

with $\boldsymbol{p}_j$ being the directly adjacent neighbor vertices of $\boldsymbol{p}$ (cf. Fig. 11). The umbrella-operator can be applied recursively leading to

$$\mathcal{U}^2(\boldsymbol{p}) = \frac{1}{n}\sum_{j=0}^{n-1} \mathcal{U}(\boldsymbol{p}_j) - \mathcal{U}(\boldsymbol{p})$$

as a discretization of the squared Laplacian. The coefficient v in (1) is given by

$$v = 1 + \frac{1}{n}\sum_j \frac{1}{n_j},$$

where n and n_j are the valences of the center vertex $\boldsymbol{p}$ and its jth neighbor $\boldsymbol{p}_j$, respectively.

In the context of discrete energy minimization, the iterative application of the update-rule (1) implements a Gauß–Seidel solver for the underlying linear system. From a more abstract point of view, the rule can also be considered as a mere relaxation operator that effectively filters out high frequency noise from the mesh [35].

3.1. 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 [33]. 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 [16].

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 remaining 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!).

We can apply the same principle to hierarchical mesh structures which are generated from fine-to-coarse. Instead of iteratively solving the discretized optimization problem on the finest level, we solve it on coarser intermediate levels first and then use the coarse solutions to estimate better starting values for the iterative solver on the finer levels.

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.

In our particular setting of thin-plate optimization on fine-to-coarse hierarchies, the Ψ-operator is simply the update-rule (1) and the restriction operator is a sequence of edge-collapse or vertex removal steps which are performed by the mesh decimation algorithm. The prolongation operator re-inserts the vertices. Since the prolongation operator can be designed to insert the new vertices to a locally optimal position, i.e., the center of gravity of its direct neighbors such that $\mathcal{U}(\boldsymbol{p}) = 0$, there is no need to actually perform any pre-smoothing. In fact, the whole multi-level smoothing algorithm reduces to mesh decimation down to a certain resolution and then alternating the re-inserting and Gauß–Seidel smoothing. Another consequence is that more sophisticated W-cycle schedules are very unlikely to improve the convergence of the algorithm.

There are several algorithmic parameters in this generic multi-level scheme. First, we have to choose the number of Gauß–Seidel steps which are performed on every level. As this is the most time consuming step of the algorithm and since our goal is to run the optimization in real-time with a prescribed number of frames per second, we cannot allow the iteration to proceed until the residuum drops below some given threshold. We rather perform a fixed number of iterations on each level. By adjusting that number we directly trade the quality of the resulting mesh for the speed of the algorithm.

Another algorithmic parameter is the number of hierarchy levels. The two extreme positions are either to re-insert all vertices and then perform Gauß–Seidel on the finest level only or to apply (1) after the insertion of every single vertex. From a practical point of view, the upper bound for the granulatity of hierarchy levels is reached if the vertices which are inserted when going from level $\mathcal{M}_i$ to $\mathcal{M}_{i+1}$, are *independent* from each other, i.e., their topological distance is larger than some threshold. This is because the local update operation (1) propagates geometric changes very slowly. An alternative to combining a sequence of independent vertex splits (or edge collapses) is proposed in [15] where the local smoothing operator is applied only in the vicinity of the newly inserted vertex.

Since the eigenstructure of the Gauß–Seidel iteration matrix and hence the convergence behavior of the generalized multi-level scheme strongly depends on the actual connectivity of the mesh, we cannot derive general estimates for the convergence rates. Nevertheless we can analyze the typical behavior of the multi-level smoothing on fine-to-coarse hierarchies by numerical experiments. We made some experiments where we performed the multi-level smoothing with a varying number of hierarchy levels and Gauß–Seidel iterations per level. The results are shown in Fig. 12.

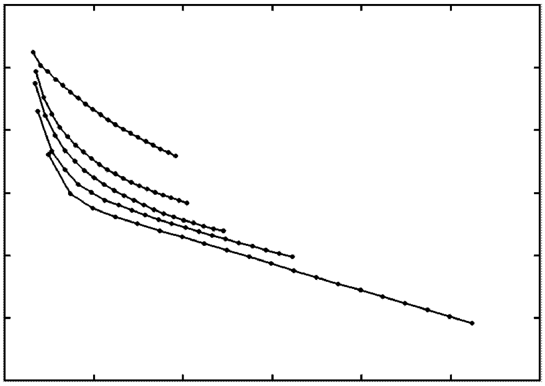

Fig. 12. This diagram shows the logarithm of the approximation error (vertical axis) versus the computation time (horizontal axis). The knots on each polygon mark the measurements for a different number of Gauß–Seidel iterations $(1,\ldots,20)$. The different polygons connect the measurements for the same number of hierarchy levels (from bottom to top: 27, 14, 9, 7, 6 levels). The monotony of the curves shows that for a fixed amount of computation time (vertical line) or a prescribed approximation error (horizontal line) the multi-level smoothing schedule with the higher number of levels always outperforms the others.

Obviously the approximation error decreases with increasing number of Gauß–Seidel steps and with increasing number of levels but also the computational costs become higher. When using the multi-level smoothing in practical applications we typically prescribe the maximum time or the maximum approximation error, i.e., we want to find the best approximation within a given period of time or we want to find a solution with a prescribed approximation error as fast as possible. In Fig. 12 these constraints correspond to vertical or horizontal lines respectively.

As a general rule of thumb it turned out that more Gauß–Seidel iterations per level only marginally improve the final result. This is due to the bad convergence on each individual level. Better results can be achieved if more hierarchy levels are used but with fewer iterations per level.

Notice that the number of *topological* hierarchy levels as one algorithmic parameter in the multi-level smoothing scheme has nothing to do with the number of *geometric* hierarchy levels in the geometric multi-band decomposition (topological versus geometric hierarchy). One is used to make the detail reconstruction more robust while the other is used to accelerate the global optimization procedure.

3.2. Boundary constraints

In order to enable intuitive modeling functionality we have to implement a simple and effective interaction metaphor. As the shape of the mesh is controlled by discrete curvature minimization, the most simple way to influence the result is by imposing appropriate boundary constraints. These constraints determine the support and the shape of the modification.

In [21] we proposed a simple metaphor where the designer starts by marking an arbitrary region on the mesh $\mathcal{M}_m$. 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.

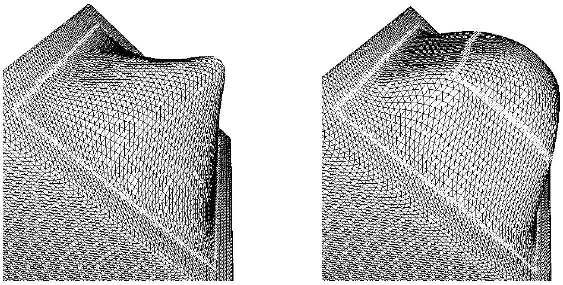

Fig. 13. Controlling the characteristics of the modification by the size of a closed handle polygon.

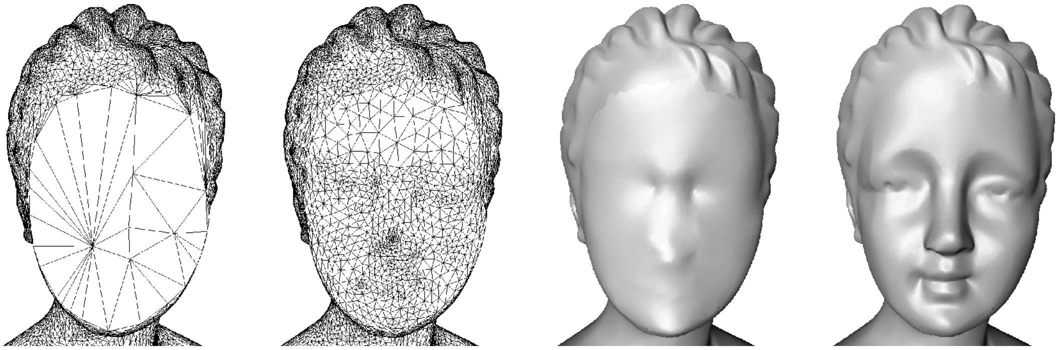

Fig. 14. During the real-time modeling, the multi-level smoothing always starts on the coarses level down to which $\mathcal{M}_*$ is reduced (left). We alternate vertex re-insertion and Gauß–Seidel smoothing (center left) until the mesh with minimum thin plate energy with respect to the current interpolation constraints is found (center right). To this smooth mesh, we add the detail coefficients to reconstruct the modified surface (right).

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. 13). Notice that the mesh vertices in the interior of the handle polygon also move according to the energy minimization.

Since we are working on triangle meshes, the energy minimization on $\mathcal{M}_*$ is done by discrete fairing techniques. To enable real-time editing we use the multi-level smoothing approach (cf. Fig. 14). While Fig. 15 depicts the general modeling set-up for a geometric two-band decomposition, more intermediate levels can be used for the detail reconstruction if the original geometry cannot be projected onto the optimized mesh without self-intersections. The boundary triangles $\mathcal{S}$ provide the correct C^1 boundary conditions for minimizing the thin plate energy functional. The handle imposes additional interpolatory constraints on the location only – derivatives should not be affected by the handle.

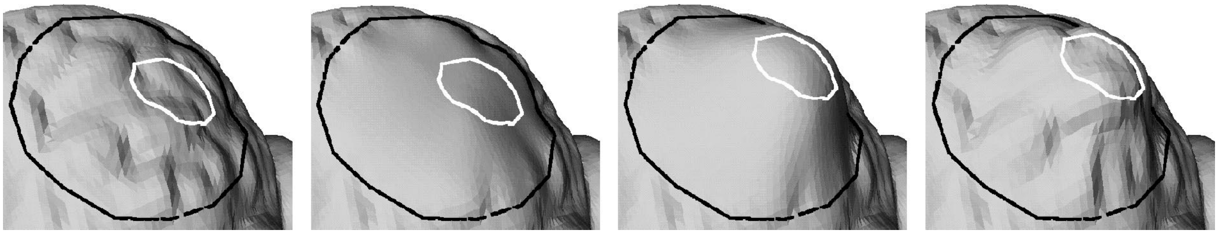

Fig. 15. A flexible metaphor for multiresolution edits. On the left, the original mesh is shown. The black line defines the region of the mesh which is subject to the modification. The white line defines the handle geometry which can be moved by the designer. Both boundaries can have an arbitrary shape and hence they can, e.g., be aligned to geometric features in the mesh. The boundary and the handle impose C^1 and C^0 boundary conditions to the mesh and the smooth version of the original mesh is found by applying discrete fairing while observing these boundary constraints. The center left shows the result of the curvature minimization (the boundary and the handle are interpolated). The geometric difference between the two left meshes is stored as detail information with respect to local frames. Now the designer can move the handle polygon and this changes the boundary constraints for the curvature minimization. Hence the discrete fairing generates a modified smooth mesh (center right). Adding the previously stored detail information yields the final result on the right. Since we can apply fast multi-level smoothing when solving the optimization problem, the modified mesh can be updated with several frames per second during the modeling operation. Notice that all four meshes have the same connectivity.

In [21] we proposed to impose the handle interpolation constraints to the optimization problem by simply freezing every other vertex of the handle polygon. On the one hand, this is a simple way to implement interpolation constraints, on the other hand, it prevents any influence on the tangent plane.

Another way to impose interpolation constraints is to prescribe them for *centers* of triangles. Such constraints can easily be embedded into the iterative energy minimization by allowing Gauß–Seidel updates for *all* vertices and re-enforcing the constraints after each iteration. This means that we shift the constrained triangles such that their centers coincide with the interpolation points after every smoothing cycle. By *shifting* the triangles without rotation we allow the tangent at the interpolation point to be controlled by the optimization process (and hence we do not impose a C^1 constraint). Fig. 16 demonstrates that the convergence behavior is much better for this kind of interpolation constraint compared to freezing vertices.

4. Conclusions and future work

We explained how to address various technical problems when using the fine-to-coarse multiresolution mesh representation which has been proposed in [21]. We presented a new way to encode the geometric detail information by using a continuous normal field on the low-frequency geometry. This makes the detail reconstruction more robust than other local frame based techniques. We also showed how the use of several intermediate levels of detail enables the handling of geometric configurations which cannot be processed correctly with a plain two-band decomposition. We further investigated the influence of various algorithmic parameters onto the overall performance of multi-level smoothing schemes when applied to a fine-to-coarse hierarchy on arbitrary meshes.

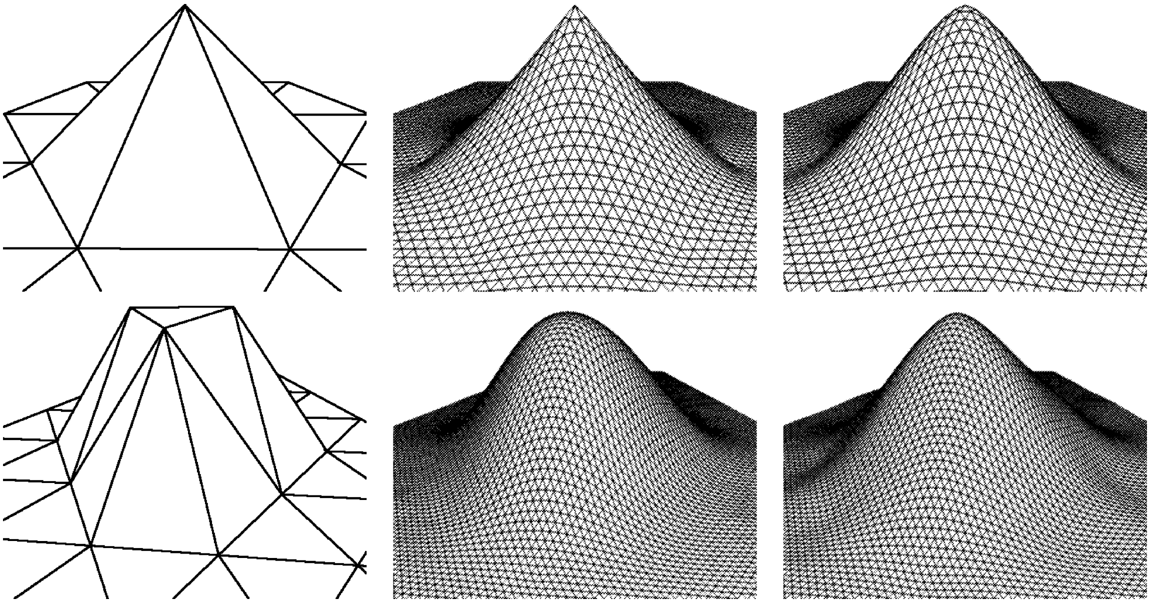

Fig. 16. In the top row discrete thin plate energy is minimized while imposing point interpolation constraint at the vertices of the original mesh (left: original, center: after 5 multi-level Gauß–Seidel iterations; right: exact result). In the bottom row the interpolation constraints are imposed at the centers of the triangles (left: original; center: after 5 multi-level Gauß–Seidel iterations; right: exact result).

In our current implementation of the multiresolution mesh modeling technique, the supporting mesh which is controlled by constrained optimization during the interactive modeling has the same connectivity as the original mesh. In the future it might be promising to drop this restriction. We could improve the stability and convergence speed of the multi-level scheme by using regularly connected meshes instead. Moreover, this could provide the possibility to use “better” local parameterizations for the discrete Laplace operator with reasonable computational effort [4,15]. However, imposing the boundary conditions into the optimization would become more involved.

References

[1] G. Bonneau, S. Hahmann, G. Nielson, Blac-wavelets: a multiresolution analysis with non-nested spaces, in: Visualization Proceedings, 1996, pp. 43–48.

[2] E. Catmull, J. Clark, Recursively generated B-spline surfaces on arbitrary topological meshes, Comput. Aided Design 10 (6) (1978) 239–248.

[3] G. Celniker, D. Gossard, Deformable curve and surface finite elements for free-form shape design, in: Computer Graphics (SIGGRAPH 91 Proceedings), July 1991, pp. 257–265.

[4] M. Desbrun, M. Meyer, P. Schröder, A. Barr, Implicit fairing of irregular meshes using diffusion and curvature flow, in: Computer Graphics (SIGGRAPH 99 Proceedings), 1999, pp. 317–324.

[5] D. Doo, M. Sabin, Behaviour of recursive division surfaces near extraordinary points, Comput. Aided Design (1978).

[6] N. Dyn, D. Levin, J.A. Gregory, A butterfly subdivision scheme for surface interpolation with tension control, ACM Trans. Graphics 9 (2) (1990) 160–169.

[7] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, W. Stuetzle, Multiresolution analysis of arbitrary meshes, in: Computer Graphics (SIGGRAPH 95 Proceedings), 1995, pp. 173–182.

[8] A. Finkelstein, D.H. Salesin, Multiresolution curves, in: Computer Graphics (SIGGRAPH 94 Proceedings), July 1994, pp. 261–268.

[9] Foley, van Dam, Feiner, Hughes, Computer Graphics, Addison-Wesley, Reading, MA, 1990.

[10] D.R. Forsey, R.H. Bartels, Hierarchical B-spline refinement, in: Computer Graphics (SIGGRAPH 88 Proceedings), 1988, pp. 205–212.

[11] D.R. Forsey, R.H. Bartels, Surface fitting with hierarchical splines, ACM Trans. Graphics 14 (2) (1995) 134–161.

[12] M. Garland, P.S. Heckbert, Surface simplification using quadric error metrics, in: Computer Graphics (SIGGRAPH 97 Proceedings), 1997, pp. 209–218.

[13] S.J. Gortler, M.F. Cohen, Hierarchical and variational geometric modeling with wavelets, in: Proceedings Symposium on Interactive 3D Graphics (May 1995).

[14] G. Greiner, Variational design and fairing of spline surfaces, Comput. Graphics Forum 13 (1994) 143–154.

[15] I. Guskov, W. Sweldens, P. Schröder, Multiresolution signal processing for meshes, in: Computer Graphics (SIGGRAPH 99 Proceedings), 1999, pp. 325–334.

[16] W. Hackbusch, Multi-Grid Methods and Applications, Springer, Berlin, 1985.

[17] H. Hoppe, Progressive meshes, in: Computer Graphics (SIGGRAPH 96 Proceedings), 1996, pp. 99–108.

[18] L. Kobbelt, Interpolatory subdivision on open quadrilateral nets with arbitrary topology, in: Computer Graphics Forum, Proceedings of Eurographics ’96, 1996, pp. C407–C420.

[19] L. Kobbelt, Discrete fairing, in: Proceedings of the Seventh IMA Conference on the Mathematics of Surfaces, 1997, pp. 101–131.

[20] L. Kobbelt, S. Campagna, H.-P. Seidel, A general framework for mesh decimation, in: Proceedings of the Graphics Interface Conference ’98, 1998.

[21] L. Kobbelt, S. Campagna, J. Vorsatz, H.-P. Seidel, Interactive multi-resolution modeling on arbitrary meshes, in: Computer Graphics (SIGGRAPH 98 Proceedings), 1998, pp. 105–114.

[22] L. Kobbelt, J. Vorsatz, U. Labsik, H.-P. Seidel, A shrink wrapping approach to remeshing polygonal surfaces, in: Computer Graphics Forum, Proceedings of Eurographics ’99, 1999.

[23] V. Krishnamurthy, M. Levoy, Fitting smooth surfaces to dense polygon meshes, in: Computer Graphics (SIGGRAPH 96 Proceedings), 1996, pp. 313–324.

[24] A. Lee, W. Sweldens, P. Schröder, L. Cowsar, D. Dobkin, Multiresolution adaptive parameterization of surfaces, in: Computer Graphics (SIGGRAPH 98 Proceedings), 1998, pp. 95–104.

[25] C. Loop, Smooth spline surfaces over irregular meshes, in: Computer Graphics Proceedings (1994), Annual Conference Series, ACM SIGGRAPH, pp. 303–310.

[26] M. Lounsbery, T. DeRose, J. Warren, Multiresolution analysis for surfaces of arbitrary topological type, ACM Trans. Graphics 16 (1) (1997) 34–73.

[27] D. Luebke, C. Erikson, View-dependent simplification of arbitrary polygonal environments, in: Computer Graphics (SIGGRAPH 97 Proceedings), 1997, pp. 199–208.

[28] H. Moreton, C. Séquin, Functional optimization for fair surface design, in: Computer Graphics (SIGGRAPH 92 Proceedings), 1992, pp. 167–176.

[29] J. Rossignac, Simplification and compression of 3D scenes, in: Tutorial Eurographics ’97, 1997.

[30] N.E. Sapidis, Designing Fair Curves and Surfaces, SIAM, Philadelphia, PA, 1994.

[31] P. Schröder, W. Sweldens, Spherical wavelets: Efficiently representing functions on the sphere, in: Computer Graphics (SIGGRAPH 95 Proceedings), 1995, pp. 161–172.

[32] W.J. Schroeder, J.A. Zarge, W.E. Lorensen, Decimation of triangle meshes, in: Computer Graphics (SIGGRAPH 92 Proceedings), 1992, pp. 65–70.

[33] J. Stoer, Einführung in die Numerische Mathematik I, Springer, Berlin, 1983.

[34] E. Stollnitz, T. DeRose, D. Salesin, Wavelets for computer graphics, Morgan Kaufmann, San Mateo, CA, 1996.

[35] G. Taubin, A signal processing approach to fair surface design, in: Computer Graphics (SIGGRAPH 95 Proceedings), 1995, pp. 351–358.

[36] G. Turk, Re-tiling polygonal surfaces, in: Computer Graphics (SIGGRAPH 92 Proceedings), 1992, pp. 55–64.

[37] W. Welch, A. Witkin, Variational surface modeling, in: Computer Graphics (SIGGRAPH 92 Proceedings), 1992, pp. 157–166.

[38] W. Welch, A. Witkin, Free-form shape design using triangulated surfaces, in: A. Glassner (Ed.), Computer Graphics (SIGGRAPH 94 Proceedings), 1994, pp. 247–256.

[39] D. Zorin, P. Schröder, W. Sweldens, Interpolating subdivision for meshes with arbitrary topology, in: Computer Graphics (SIGGRAPH 96 Proceedings), 1996, pp. 189–192.

[40] D. Zorin, P. Schröder, W. Sweldens, Interactive multiresolution mesh editing, in: Computer Graphics (SIGGRAPH 97 Proceedings), 1997, pp. 259–268.