Eurographics Symposium on Geometry Processing (2004)
R. Scopigno, D. Zorin (Editors)

A Remeshing Approach to Multiresolution Modeling

Mario Botsch Leif Kobbelt

Computer Graphics Group
RWTH Aachen University

Abstract

Providing a thorough mathematical foundation, multiresolution modeling is the standard approach for global surface deformations that preserve fine surface details in an intuitive and plausible manner. A given shape is decomposed into a smooth low-frequency base surface and high-frequency detail information. Adding these details back onto a deformed version of the base surface results in the desired modification. Using a suitable detail encoding, the connectivity of the base surface is not restricted to be the same as that of the original surface. We propose to exploit this degree of freedom to improve both robustness and efficiency of multiresolution shape editing.

In several approaches the modified base surface is computed by solving a linear system of discretized Laplacians. By remeshing the base surface such that the Voronoi areas of its vertices are equalized, we turn the unsymmetric surface-related linear system into a symmetric one, such that simpler, more robust, and more efficient solvers can be applied. The high regularity of the remeshed base surface further removes numerical problems caused by mesh degeneracies and results in a better discretization of the Laplacian operator.

The remeshing is performed on the low-frequency base surface only, while the connectivity of the original surface is kept fixed. Hence, this functionality can be encapsulated inside a multiresolution kernel and is thus completely hidden from the user.

1. Introduction

Caused by the steadily increasing availability and efficiency of both hardware and software for digital geometry processing, more and more physical stages in the industrial development and design process are assisted by or even replaced by methods from CAGD. Between those, one of the most challenging topics is shape editing, since complex mathematical frameworks have to be hidden behind an intuitive modeling metaphor, thus enabling even the non-expert user to perform sophisticated surface deformations.

Currently, such modeling tasks are typically done using CAD systems based on NURBS as the underlying surface representation. However, the topological restriction of NURBS to triangular or rectangular domains results in an extremely high number of (possibly trimmed) surface patches to be used for representing non-trivial geometries, making surface editing a complicated and tedious process.

During the last years, many techniques from NURBS modeling have been discretized and generalized to triangle meshes. While subdivision surfaces can be considered as a generalization of spline surfaces to arbitrary topology, they require the mesh to have semi-regular subdivision connectivity. Without the topological restrictions of NURBS and the connectivity constraints of subdivision surfaces, arbitrary triangle meshes gained increasing attention as a surface representation suitable for shape editing [KCVS98]. Recent approaches based on triangle mesh modeling provide an easy and intuitive modeling metaphor as well as real-time response times when interacting with the surface [BK04].

Freeform modeling approaches are mostly used to create and edit smooth surfaces. A combination with multiresolution modeling techniques is more relevant for real-world modeling tasks, since this provides both local and global shape deformations while additionally preserving fine surface details in an intuitive and plausible manner.

This is achieved by splitting a given surface — considered as a surface signal — into high and low frequencies. The low frequencies of the original surface are represented as a

smooth *base surface* and are typically constructed by low-pass filtering the original surface [DMSB99, Lév03]. The remaining high frequencies are encoded into *detail information*, such that the original surface can be reconstructed from the smooth base and this detail information. A general multiresolution modeling framework then consists of

1. a decomposition operator splitting the surface into a low-frequency base surface and high-frequency details,
2. a freeform modeling operator deforming the base surface,
3. a reconstruction operator adding the detail information back onto a modified version of the base surface.

Notice that the original surface and the base mesh play different roles in the modeling process. The former is the mesh the designer sees and interacts with, the latter is generally hidden from the user and is internally used for computing the deformations. As a consequence, the requirements on these two surfaces also differ. The original surface has to provide a high-quality approximation to the actual surface geometry and represents its fine details and sharp features by a possibly hand-crafted triangulation. On the other hand side, since all numerical computations are performed on the base mesh, its structure is mainly responsible for robustness and efficiency.

When a suitable detail encoding is used [KVS99, BK03], the tessellation of the base surface is not restricted to be the same as the tessellation of the original surface. As a consequence, the triangulation of the base surface can be considered as an additional degree of freedom that can be adjusted in order to improve the multiresolution modeling process. Note that by construction the base surface is smooth and contains no high-frequency details, hence a remeshing or resampling is possible without introducing aliasing artifacts. In contrast, the original surface generally does contain sharp features, therefore a remeshing is prohibitive, as it could destroy a feature-aligned triangulation.

Two major design goals for a multiresolution modeling framework are robustness and efficiency. As both items are closely related to the base mesh, we propose to adjust its tessellation, i.e. to remesh the base surface, such that the shape editing process becomes both more stable and more efficient.

Numerical robustness on triangle meshes is mostly related to the triangles' shape. Close-to equilateral triangles enable stable computations, while for degenerate faces neither area nor normal vectors can be derived robustly, causing problems for the detail encoding as well as for the computation of the deformations. As a consequence, a remeshing of the base surface should aim at well-shaped triangles.

For the actual deformation of the base surface we have to solve a linear system based on Laplacians. This system is symmetric up to per-vertex normalization factors containing the Voronoi areas around the vertices (see Sec. 3). Fine-tuning the remeshing, such that the vertex areas are equalized, turns this matrix into a symmetric one. As a consequence, this enables us to use more robust solvers that additionally are faster by an order of magnitude. By exploiting this remeshing technique, our approach differs from previous ones, as we do not try to find a suitable solver fitting our problem, but we fit our problem to the best possible solvers instead.

2. Related Work

There is a large variety of shape editing approaches, being based either on NURBS, on subdivision surfaces or on arbitrary triangle meshes, and using surface related or volumetric techniques in order to compute shape deformations. However, we will focus on multiresolution modeling methods in this paper, since they provide the most intuitive preservation of fine details even for large scale modifications.

A key component is the encoding of high-frequency detail information. The standard method is to represent the original surface as a displacement of the smooth base surface, e.g. by associating a displacement vector with each vertex. To achieve a natural detail preservation, these vectors have to be stored in *local frames*, consisting of the surface normal and two perpendicular tangent vectors [FB95, FB88].

An approach based on subdivision surfaces is presented by Zorin et al. [ZSS97] that tightly couples the surfaces on different smoothness levels by the required subdivision connectivity. Guskov et al. [GSS99] propose a modeling system for arbitrary meshes and store details in vertex-based local frames. They build their hierarchy by progressive mesh decimation, such that the vertices on level i are a subset of the vertices of level $i+1$.

In order to base the detail encoding on purely geometric (instead of additional parametric) information, *normal displacements* represent vertices on one mesh as an offset in normal direction from another mesh. One method to generate normal displacements is to shoot rays from the base surface in normal direction and to place samples at the intersections of these rays with the original surface [GVSS00]. This, however, corresponds to a resampling of the high-frequency original surface and may therefore lead to aliasing problems.

In contrast, Kobbelt et al. [KCVS98, KVS99] resample the smooth base surface by finding base points on it, such that the vertices of the original surface are normal displacements of these base points. Since this technique resamples a low-frequency surface, it avoids aliasing and preserves sharp features of the original surface. As an additional advantage, it decouples the tessellation of the base surface from the original surface. Building on this work, Botsch and Kobbelt [BK03] present a volumetric detail encoding that provides a more natural behavior and avoids local self-intersections — at the expense of a more involved reconstruction operator.

The last two approaches offer the possibility to freely adjust the tessellation of the base mesh, while keeping the

triangulation of the original mesh fixed. We will shown in Sec. 4 that a special isotropic remeshing of the base surface greatly improves the multiresolution modeling process.

Isotropic remeshing of large meshes based on a global parameterization was proposed by Alliez et al. [ACdVDI03]. In order to avoid the expensive global parameterization, both Surazhsky et al. [SG03, SAG03] and Vorsatz et al. [VRS03] base the remeshing on local parameterizations instead, enhancing the general idea of dynamic meshes [KBS00].

The remeshing method we present in Sec. 4 is conceptually very similar to these techniques, but has the advantage of not having to respect a hard error bound or handling sharp features, therefore being considerably simpler and more efficient. Using this remeshing technique, the multiresolution modeling process described in the next section becomes both more robust and more efficient, as shown in Sec. 5.

3. Multiresolution Modeling

In this section we describe the multiresolution modeling framework and set up the notation used throughout the paper.

For our freeform modeling operator we use the *boundary constraint modeling* approach of [BK04]. The designer first specifies the support of the modification, i.e. the surface region that is allowed to change, by selecting an arbitrary region on the mesh using a painting or outlining tool. Within the support region, one or several control handle regions are chosen that act as generalized control points by providing 9 degrees of freedom (translation, rotation, scaling). Using a manipulator item, these affine transformations can be applied to the control handles in an intuitive manner. The support region then provides a smooth blend between the transformed handles and the fixed part of the surface (cf. Fig. 1).

The deformed blue surface region is computed by a constrained energy minimization. The minimizer surfaces S of

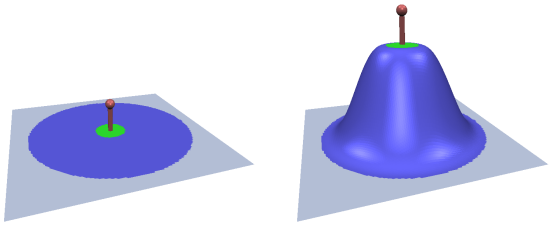

Figure 1: *In our freeform modeling metaphor, the designer selects a support region (blue) and a handle region (green). After transforming the handle using a manipulator widget, the blue region smoothly blends between the handle and the fixed part of the surface.*

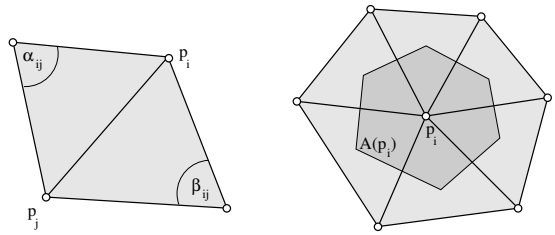

Figure 2: *The discretization of the Laplacian at a vertex p_i depends on cotangent values of α_{ij} and β_{ij} and on the Voronoi area $A(p_i)$.*

these special surface functionals are characterized by Euler-Lagrange PDEs of the form $\Delta^k S = 0$, where Δ denotes the Laplacian operator. Typical values for k lead to membrane surfaces ($k = 1$), thin-plate surfaces ($k = 2$), and surfaces minimizing curvature variation ($k = 3$).

In order to turn these PDEs into linear systems, we use the recursive discretization of the Laplacian operator proposed by [DMSB99, MDSB03]:

$$\Delta^{k+1} p_i := \sum_{p_j} \frac{2\left(cot\alpha_{ij} + cot\beta_{ij}\right)}{A(p_i)} \left(\Delta^k p_j - \Delta^k p_i\right) \quad (1)$$

$$\Delta^0 p_i := p_i \quad (2)$$

where $\alpha_{ij} = \angle(p_i, p_{j-1}, p_j)$ and $\beta_{ij} = \angle(p_i, p_{j+1}, p_j)$ for a vertex p_i and its one-ring neighbors $p_j \in N(p_i)$. The normalization factor $A(p_i)$ denotes the Voronoi area around the vertex p_i (cf. Fig. 2) and is built from the incident triangles' circum-centers (or edge midpoints for obtuse angles).

This leads to a linear system in the free (*blue*) vertices $\mathbf{p} = (p_1, \ldots, p_P)$, the (*green*) handle vertices $\mathbf{h} = (h_1, \ldots, h_H)$ and the (*grey*) fixed vertices $\mathbf{f} = (f_1, \ldots, f_F)$:

$$\left(\begin{array}{c|c} \Delta^k & \\ \hline 0 & I_{F+H} \end{array}\right) \begin{pmatrix} \mathbf{p} \\ \mathbf{f} \\ \mathbf{h} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{f} \\ \mathbf{h} \end{pmatrix}. \quad (3)$$

Note that $\mathbf{h}$ and $\mathbf{f}$ are boundary constraints and should be moved to the right-hand side for the actual solution of the system. They are used as pseudo-unknowns for clearer notation only. Each time the handle is moved, the right-hand side of this system is changed, such that the new blue surface is computed by solving the system again.

This freeform modeling operator is integrated into a multiresolution modeling framework as shown in Fig. 3. In a modeling session, the designer first selects the support region and control handles on the original mesh (top center). These regions are mapped to the smooth base surface and the linear system is solved in the undeformed state, removing further medium frequencies (bottom center). The difference between the surfaces shown at top center and bottom center are encoded into high-frequency detail information.

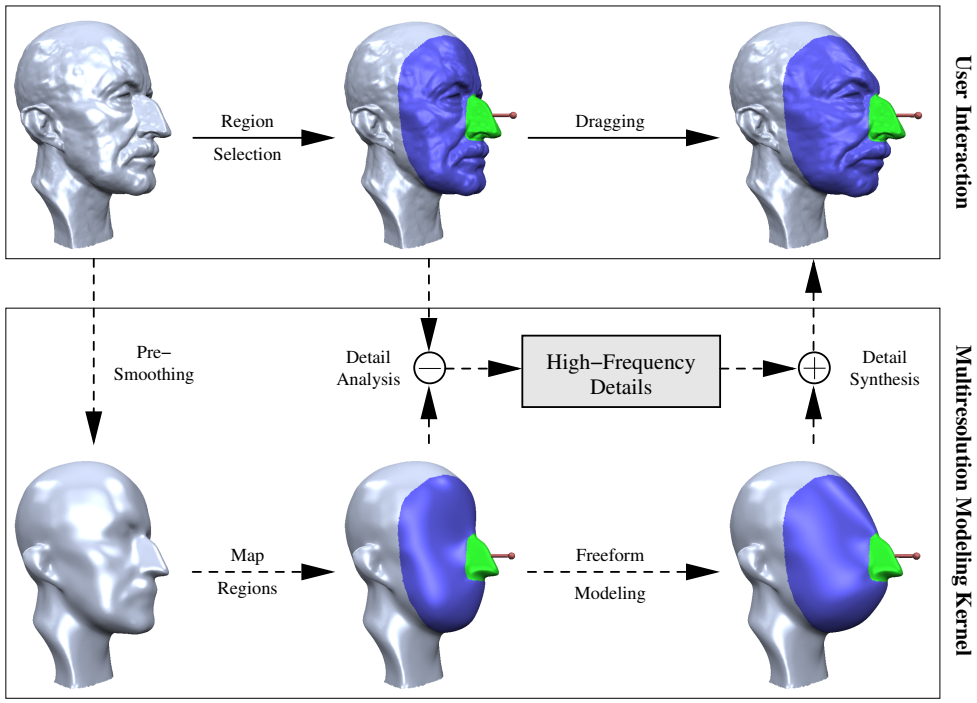

Figure 3: *Our multiresolution modeling framework initially computes a low-frequency base (bottom left) for the input surface (top left). After selecting support and control handles on the original surface (top center), these regions are mapped to the base surface and high-frequency details are encoded (bottom center). Moving the manipulator changes the base surface (bottom right), adding the detail information back results in the correct modification (top right).*

By dragging the manipulator the designer deforms the base surface using the freeform modeling operator (bottom right). Adding the details back onto this surface constructs the desired global deformation (top right). Notice that the user interacts with the original surface only, as shown in the top row. All computations depicted in the bottom row are hidden within the multiresolution modeling kernel.

For the detail encoding we use the Phong detail representation of [KVS99]. We first build a continuous normal field on the base surface and encode each vertex p of the original surface by a point b on the base surface and an offset λ in normal direction, i.e. $p = b + \lambda \cdot n(b)$. The base point b is not restricted to be a vertex of the base mesh, but is in general an interior point represented by a triangle index and barycentric coordinates. As a consequence, the tessellation of the base surface does not have to be the same as that of the original surface.

Taking a closer look at Fig. 3, we see that all deformations are computed by applying Eq. 3 to the base surface (bottom row). Notice that the linear system is not symmetric in general, caused by the normalization factors $A(p_i)$ of Eq. 1, denoting the Voronoi area around the vertex p_i. Since for a higher order Laplacian matrix Δ^k, $k > 1$, also Laplacians of degree $k-1$ of boundary vertices of $\mathbf{f}$ or $\mathbf{h}$ are involved, and since these again depend on some free vertices of $\mathbf{p}$, such a matrix cannot be made symmetric by a simple multiplication with a diagonal matrix.

We therefore propose to use the remeshing technique described in the next section to turn this matrix into a symmetric one, leading to both more robust and more efficient computations.

4. Area-Equalizing Remeshing

The matrix in Eq. 3 contains cotangent weights derived from the base surface geometry. They capture the surface metric and imitate a natural internal stiffness behavior, since a kind of conformal mapping between the surface before and after the modification is implicitly built by preserving the (relative) shape of triangles [DMA02].

Notice that these weights are computed only once after loading the mesh and are usually not updated during the modeling session. By remeshing the base surface such that the Voronoi areas of its vertices are equalized, a Laplacian matrix of this form becomes symmetric. As the weights are computed only once, this remeshing also is a preprocessing step that is performed after loading the model and before computing the cotangent weights.

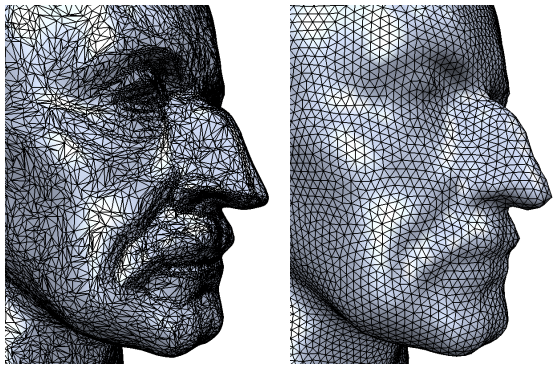

Figure 4: *Our remeshing technique yields very regular meshes and additionally equalizes the Voronoi areas of vertices.*

Similar to previous explicit remeshing approaches [SG03, SAG03, VRS03], we first generate a regularly remeshed surface by alternated equalization of edge lengths and vertex valences. Given a target edge length l, we perform the following steps:

1. Split all edges at their midpoint that are longer then $\frac{4}{3}l$.
2. Collapse all edges shorter than $\frac{4}{5}l$ into their midpoint.
3. Flip edges in order to minimize the deviation from valence 6 (or 4 on boundaries).
4. Relocate vertices on the surface by tangential smoothing.

After a few of these cycles (about 5) we obtain a triangulation whose edges have length close to l and whose vertices have valence close to 6. However, vertex areas may still differ, since a vertex of valence k is enclosed by a loop of k edges of length l. Hence the area of a vertex scales w.r.t. its valence, resulting in a "nesting" of vertices in some mesh regions.

To account for this we propose a fine-tuning by an area-based tangential smoothing. Each vertex p is assigned a gravity that equals its area $A(p)$. A tangential smoothing process moves each vertex p_i to its gravity-weighted centroid

$$g_i := \frac{1}{\sum_{p_j \in N(p_i)} A(p_j)} \sum_{p_j \in N(p_i)} A(p_j) p_j .$$

To ensure a tangential smoothing *on* the surface, the update vector is projected back into the tangent plane of p_i, such that the update rule becomes:

$$p_i \leftarrow p_i + \lambda \left(I - n_i n_i^T \right) (g_i - p_i) ,$$

where n_i is the normal vector of p_i and λ is a damping factor used to avoid oscillations. Vertices with large (relative) Voronoi area have a higher gravity and attract other vertices, thereby reducing their own area. Usually very few (< 20) iterations suffice to reduce the total variance of vertex areas by a factor of about 5, resulting in a mesh that provides (almost) equal vertex areas (cf. Fig. 4). In all our experiments the relative mean area error was below 5% and the average inner angle deviated from 60° by at most 0.1°.

Notice that in comparison to existing sophisticated remeshing approaches [SG03, SAG03, VRS03], our problem setting is much simpler, since we do not have to take care of an exact approximation error or normal deviation. Slight deviations from the original surface are acceptable, as long as the remeshed surface roughly captures the surface metric. As the base surface that is to be remeshed does not contain high frequencies, there is also no need to handle sharp features.

The target edge length l for the remeshing is chosen to be slightly less than the average edge length of the original surface. Because the sampling rate stays about the same and since we do not have to guarantee an error bound, we do not even have to base the vertex relocation steps of our remeshing on global or local parameterizations. Projecting the update vectors into the respective tangent planes turned out to be sufficient. Since parameterizations would be the most expensive part of the remeshing procedure, our scheme can be implemented quite efficiently, such that we can process an input base mesh of 100k triangles in less than 5 seconds (cf. Fig. 4).

For the mapping of support and handle regions from the original surface to the remeshed version, we implicitly construct a mutual parameterization between both surfaces during the pre-smoothing and remeshing phases.

5. Numerical Consequences

The first obvious consequence of the remeshing is that degenerate triangles are removed. Those would otherwise cause numerical problems, since the matrix would no longer be positive definite [PP93]. Because of the high regularity of the resulting mesh, all inner triangle angles (up to a few outliers) are very close to 60 degrees. This causes the respective cotangent weights to be positive, removing stability and convergence problems caused by negative weights. Additionally, the discretization of the Laplacian gets more reliable if the triangulation is lacking obtuse angles [MDSB03].

While the better approximation of the Laplacian operator and the improved conditioning of the resulting matrix are clearly preferable, the main benefit is that the matrix gets symmetric up to small errors. By replacing all vertex areas $A(p_i)$ by their mean value, we enforce an exactly symmetric matrix, but also introduce slight errors in the discretization.

Measuring the error between surfaces computed using either the correct or the mean vertex areas revealed very slight and low-frequency differences. As the same slight errors are done when the system is solved before and during the modification (Fig. 3, bottom center and bottom right), they are diminished by the detail encoding. These observations proved that we can safely use the symmetric approximation to the

Laplacian matrix. This, however, provides significant benefits for the linear system solvers to be used, as shown in the next sections.

5.1. Iterative Solvers

The standard method for solving a sparse and unsymmetric system is probably the preconditioned biconjugate gradients method (BiCG) [PFTV92]. Although working well for most cases, BiCG does not provide theoretical convergence guarantees and therefore shows a very irregularly (non-monotonic) decreasing residual for ill-conditioned systems. From an implementation point of view this method is not optimal, because a multiplication with both the matrix A and its transposed A^T have to be provided. This prevents the use of just one of either compressed row or compressed column format for sparse matrices [BBC*94].

An improvement of this situation is provided by the stabilized version Bi-CGSTAB, being a mixture of BiCG and GMRES [BBC*94]. For this method, only the multiplication with the matrix A has to be provided and the convergence behavior is smoother. However, there are still no guarantees that the method converges at all.

Using our remeshing technique, the matrix of Eq. 3 becomes symmetric and positive definite (SPD). For these systems the simple conjugate gradients (CG) is the preferable method [AMS90], providing both guaranteed and monotonic convergence and a simple and more efficient implementation (up to a factor of 2).

5.2. Multigrid Solvers

For large systems iterative solvers rapidly remove the high frequencies of the residual, but converge very slowly to the final solution. If very large systems are to be solved efficiently, multi-grid methods should be employed [Hac86].

After pre-computing a mesh hierarchy based on mesh decimation techniques, the system is first solved on the coarsest hierarchy level. This solution is prolongated to the next finer level and used as starting value for the iterations on this level. The same process is repeated until the finest level has been reached. The logarithmic number of levels leads to a total complexity of $O(n)$ for multi-grid solvers, as opposed to $O(n^2)$ for non-multi-grid iterative methods, n being the number of unknowns, i.e. the free vertices $\mathbf{p}$.

In order to avoid the remeshing of all hierarchy levels, we propose to build the mesh hierarchy using the remeshing of Sec. 4 with growing target edge lengths. This leads to symmetric matrices on each multi-grid level, providing the benefits for the single-level iterative solvers also to the multi-grid methods. As the vertices of the resulting hierarchy do not form a nested sequence, the correspondences and the prolongation operator should be based on barycentric interpolation on the piecewise linear triangle meshes.

The main problem of multi-grid solvers is that it can be quite complicated to implement them, since special care has to be taken for the hierarchy building, the preconditioning scheme and the prolongation operator [AKS]. However, if iterative solvers are to be used, multi-grid methods are the only way to achieve rapid solution times [KCVS98, RL03, AKS].

5.3. Direct Solvers

The use of direct solvers for linear systems is often underestimated, since naïve direct methods have complexity $O(n^3)$. Additionally, Gaussian elimination (LU factorization) is know to be numerically unstable unless proper pivoting is used. For symmetric systems, however, Cholesky factorization provably is backward stable, but it still has cubic complexity.

As the system we are considering is sparse, but not band-limited, the first step is to generate a band-limitation. For this task several standard methods based on symmetric row and column permutations are available [GL81], e.g. the reverse Cuthill-McKee [CM69] or the minimal degree algorithm. They result in a concentration of the non-zero elements around the diagonal, leading to a band-limit b.

If a matrix A is band-limited by b, so is its Cholesky factor L with $A = LL^T$ [GL89]. Computing such a band-limited Cholesky factorization has complexity $O(bn^2)$ instead of $O(n^3)$ for the unbanded case. The actual solution based on this factorization has complexity $O(bn)$ and is therefore comparable to multi-grid methods. Although having the same asymptotic complexity, band-limited Cholesky solvers turned out to be faster by an order of magnitude compared even to sophisticated multi-grid solvers.

5.4. Comparison

Comparing direct solvers to multi-grid iterative methods, direct solvers seem to be preferable for sparse SPD matrices. For both the pre-computation phase and the actual solution of linear systems they are faster than multi-grid iterative methods.

Besides higher efficiency, direct solvers are additionally much easier to use. Packages based on highly optimized lapack and BLAS libraries are publicly available, some of them also including the band-limitation, e.g. the TAUCS library [TCR]. In contrast, hierarchical multi-grid solvers on triangle meshes are quite sophisticated and several key components and parameter values have to chosen by experience.

There are also similar methods for solving unsymmetric problems based on a band-limited LU factorization [DEG*99]. In this case, two types of permutations are required: symbolic permutations for band-limitation and pivoting permutations ensuring numerical robustness. As these

Method	Complex.	Precomp.	3 Solutions
Iterative	15k	7.2s	7.4s
Multi-grid	15k	4.5s	0.8s
Direct	15k	2.4s	0.07s
Iterative	31k	15.2s	25.5s
Multi-grid	31k	8.8s	2.0s
Direct	31k	6.7s	0.16s

Table 1: *The timings for computing a thin-plate surface consisting of 15k resp. 31k free vertices using different solvers. The respective linear system is solved three times for the x, y and z coordinate, respectively. The corresponding pre-computations are preconditioning (*Iterative*), building the multigrid hierarchy (*Multigrid*), and band-limiting and factorizing (*Direct*), respectively.*

permutations cannot be handled separately, the overall factorization process is more involved. Since no matrix symmetries can be exploited, these methods are generally not as efficient as symmetric matrix solvers.

6. Results

Our remeshing method effectively removes degenerate triangles and results in a very regular tessellation consisting of almost equilateral faces, thereby avoiding numerical degeneracies and improving the Laplacian discretization and general conditioning of the matrix.

Since for our remeshing we do not have to take care of approximation errors or sharp features, our method is significantly faster than previous (more sophisticated) remeshing techniques, such that we can process meshes of 100k triangles in less than 5s on a Pentium 4 3.0GHz.

The resulting matrices are symmetric and positive definite, hence we can use a direct Cholesky-based solver, that provides the same linear asymptotic time complexity as multi-grid methods, but is significantly faster compared to those. As a representative example, Table 1 lists the timings for computing a thin-plate surface of 15k or 31k free vertices (solving the system three times for x, y and z), using single-level iterative methods, multi-grid iterative methods and a direct solver, respectively. As the same linear system has to be solved each time the control handle is interactively moved (i.e. the right-hand side is changed), the direct methods are faster by an order of magnitude.

Exploiting the fact that the control handles are affinely transformed, [BK04] propose the pre-computation of basis functions associated with the control handle. This shifts the per-frame solution of the linear system to a pre-computation by solving the same system $3 + 4 \cdot h$ times, h being the number of individual control handles. In this setting, our approach can efficiently be used to speed up this pre-computation phase significantly.

7. Conclusions

In this paper we pointed out that using state-of-the-art multiresolution surface representations, the connectivity of the low-frequency base surface is not restricted to be the same as that of the original surface. We exploit this degree of freedom by remeshing the base surface, such that the problem becomes numerically better conditioned and the linear system to be solved turns into a symmetric one.

We showed that for symmetric positive definite sparse linear systems direct solvers based on a band-limited Cholesky factorization are clearly preferable to iterative methods, as they are significantly faster as well as much easier to use. This observation should also be useful in other applications where this type of matrix problems is to be solved.

For our multiresolution modeling framework the proposed remeshing technique provides the possibility to keep the tessellation of the input mesh fixed, even if it contains numerically critical skinny triangles. This makes multiresolution modeling more robust, more efficient and also applicable to a wider range of input geometries, at the cost of just a small increment of the pre-computation phase.

References

[ACdVDI03] ALLIEZ P., COLIN DE VERDIÈRE É., DEVILLERS O., ISENBURG M.: Isotropic surface remeshing. In *Proceedings of Shape Modeling International* (2003), pp. 49–58.

[AKS] AKSOYLU B., KHODAKOVSKY A., SCHRÖDER P.: Multilevel Solvers for Unstructured Surface Meshes. In review, SIAM J. Sci. Comput.

[AMS90] ASHBY S. F., MANTEUFFEL T. A., SAYLOR P. E.: A taxonomy for conjugate gradient methods. *SIAM J. Numer. Anal. 27*, 6 (1990), 1542–1568.

[BBC*94] BARRETT R., BERRY M., CHAN T. F., DEMMEL J., DONATO J., DONGARRA J., EIJKHOUT V., POZO R., ROMINE C., DER VORST H. V.: *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*. SIAM, Philadelphia, PA, 1994.

[BK03] BOTSCH M., KOBBELT L.: Multiresolution surface representation based on displacement volumes. In *Proceedings of Eurographics 03* (2003), pp. 483–492.

[BK04] BOTSCH M., KOBBELT L.: An intuitive framework for real-time freeform modeling. In *Proceedings of ACM SIGGRAPH 04* (2004).

[CM69] CUTHILL E., MCKEE J.: Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 24th National Conference ACM* (1969), pp. 157–172.

[DEG*99] DEMMEL J. W., EISENSTAT S. C., GILBERT J. R., LI X. S., LIU J. W. H.: A supernodal approach to sparse partial pivoting. *SIAM Journal on Matrix Analysis and Applications 20*, 3 (1999), 720–755.

[DMA02] DESBRUN M., MEYER M., ALLIEZ P.: Intrinsic parameterizations of surface meshes. In *Proceedings of Eurographics 02* (2002), pp. 209–218.

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of ACG SIGGRAPH 99* (1999), pp. 317–324.

[FB88] FORSEY D. R., BARTELS R. H.: Hierarchical B-spline refinement. In *Proceedings of ACM SIGGRAPH 88* (1988), vol. 22, pp. 205–212.

[FB95] FORSEY D., BARTELS R. H.: Surface fitting with hierarchical splines. *ACM Transactions on Graphics 14*, 2 (1995), 134–161.

[GL81] GEORGE A., LIU J. W.: *Computer Solution of Large Sparse Positive Definite Matrices,*. Prentice Hall, 1981.

[GL89] GOLUB G. H., LOAN C. F. V.: *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1989.

[GSS99] GUSKOV I., SWELDENS W., SCHRÖDER P.: Multiresolution signal processing for meshes. In *Proceedings of ACM SIGGRAPH 99* (1999), pp. 325–334.

[GVSS00] GUSKOV I., VIDIMCE K., SWELDENS W., SCHRÖDER P.: Normal meshes. In *Proceedings of ACM SIGGRAPH 00* (2000), pp. 95–102.

[Hac86] HACKBUSCH W.: *Multi-Grid Methods and Applications*. Springer Verlag, 1986.

[KBS00] KOBBELT L., BAREUTHER T., SEIDEL H.-P.: Multiresolution shape deformations for meshes with dynamic vertex connectivity. In *Proceedings of Eurographics 00* (2000).

[KCVS98] KOBBELT L., CAMPAGNA S., VORSATZ J., SEIDEL H.-P.: Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of ACM SIGGRAPH 98* (1998), pp. 105–114.

[KVS99] KOBBELT L., VORSATZ J., SEIDEL H.-P.: Multiresolution hierarchies on unstructured triangle meshes. *Computational Geometry: Theory and Applications 14* (1999).

[Lév03] LÉVY B.: Dual domain extrapolation. In *Proceedings of ACM SIGGRAPH 03* (2003), pp. 364–369.

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, Hege H.-C., Polthier K., (Eds.). Springer-Verlag, Heidelberg, 2003, pp. 35–57.

[PFTV92] PRESS W. H., FLANNERY B. P., TEUKOLSKY S. A., VETTERLING W. T.: *Numerical Recipes: The Art of Scientific Computing*, 2nd ed. Cambridge University Press, 1992.

[PP93] PINKALL U., POLTHIER K.: Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2*, 1 (1993), 15–36.

[RL03] RAY N., LEVY B.: Hierarchical Least Squares Conformal Map. In *Proceedings of Pacific Graphics 03* (2003), pp. 263–270.

[SAG03] SURAZHSKY V., ALLIEZ P., GOTSMAN C.: Isotropic remeshing of surfaces: a local parameterization approach. In *Proceedings of 12th International Meshing Roundtable* (2003).

[SG03] SURAZHSKY V., GOTSMAN C.: Explicit surface remeshing. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing* (2003), pp. 20–30.

[TCR] TOLEDO S., CHEN D., ROTKIN V.: Taucs: A library of sparse linear solvers. http://www.tau.ac.il/ stoledo/taucs.

[VRS03] VORSATZ J., RÖSSL C., SEIDEL H.-P.: Dynamic remeshing and applications. In *Proceedings of Solid Modeling and Applications* (2003), pp. 167–175.

[ZSS97] ZORIN D., SCHRÖDER P., SWELDENS W.: Interactive multiresolution mesh editing. In *Proceedings of ACM SIGGRAPH 97* (1997), pp. 259–268.