EUROGRAPHICS 2005 / M. Alexa and J. Marks
(Guest Editors)

Volume 24 (2005), Number 3

Automatic Generation of Structure Preserving Multiresolution Models

Martin Marinov Leif Kobbelt

Computer Graphics Group, RWTH Aachen, Germany

Abstract

We are proposing a multiresolution representation which uses a subdivision surface as a smooth base surface with respect to which a high resolution mesh is defined by normal displacement. While this basic representation is quite straightforward, our actual contribution lies in the automatic generation of such a representation. Given a high resolution mesh, our algorithm is designed to derive a subdivision control mesh whose structure is properly adjusted and aligned to the major geometric features. This implies that the control vertices of the subdivision surface not only control globally smooth deformations but in addition that these deformations are meaningful in the sense that their support and shape correspond to the characteristic structure of the input mesh. This is achieved by using a new decimation scheme for general polygonal meshes (not just triangles) that is based on face merging instead of edge collapsing. A face-based integral metric makes the decimation scheme very robust such that we can obtain extremely coarse control meshes which in turn allow for deformations with large support.

Categories and Subject Descriptors (according to ACM CCS):
I.3.5 [Computer Graphics]: Curve, surface, solid, and object representations. Modeling packages.

1. Introduction

Due to the availability of reasonably priced 3D scanning equipment the need for shape modeling techniques which enable the flexible modification of (unstructured) high resolution 3D models becomes more and more evident. Ideally such techniques should be independent from the actual tessellation of the mesh and they should provide handles through which the user can control the shape in a flexible and intuitive way. Due to the high complexity of the scanned meshes, necessary for representing the edited model with a certain precision, multiresolution modeling techniques resort to a coarse, low frequency representation (*base domain*) of the input geometry, so that the designer can efficiently describe the editing operations. Such an approach imposes a restriction on the used base-domain: it has to represent the structure of the model as close as possible, so that a modification expressed with respect to it propagates in an intuitive fashion to the edited mesh.

On the other hand, the ability to simplify the base domain such that just the very basic features of the object are present, allows large scale, almost global modifications to be performed. In addition, such modeling framework has to be flexible, permitting the designer to change the desired control resolution at any point, i.e., it needs to be progressive. Finally, the automatic generation of such a representation is highly desirable, especially in a production environment, since the costs associated with a reverse engineering/rapid prototyping cycle depend mostly on the amount of manual work (time) required. In this paper we propose a modeling framework in conjunction with a simplification algorithm, which conform to these requirements and provide an easy to use, flexible tool-set for editing polygonal meshes.

1.1. Contributions

Our first contribution is a new simplification scheme for polygonal meshes (Section 2) which generates a progressive sequence of coarser versions of the input mesh M adjusted and aligned to the major geometric features. The algorithm is specifically designed to overcome difficulties arising when M is decimated to extremely coarse resolutions and provides valid, non-folding, two-manifold polygonal meshes approximating the structure of the model (Fig. 2). The scheme is guided by two major principles: minimizing the integral $L^2/L^{2,1}$ error for all faces (this provides the relation to the input surface) and guaranteeing that each face is injectively

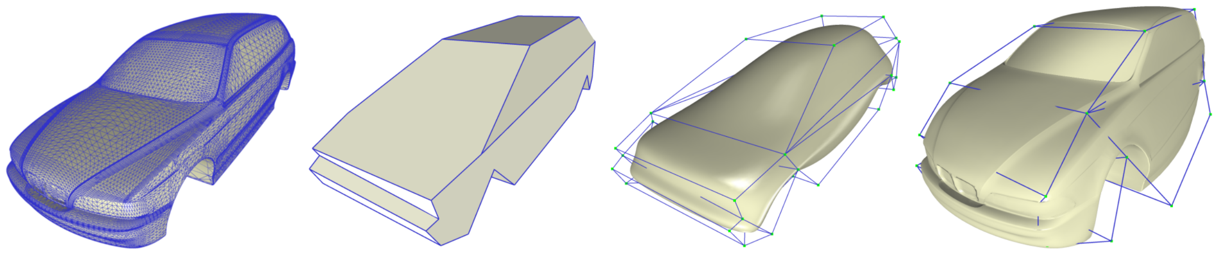


Figure 1: From left to right*: a) A tessellated CAD model as input. b) The model simplified to 24 faces. c) A corresponding Catmull-Clark surface with 44 control points. d) Using c) as a base domain of our multiresolution model allows us to perform several large-scale modifications on the original mesh (represented as a normal displacement field). Notice that moving a control point leads to a "meaningful" deformation because the coarse control mesh captures the global structure of the model.*

projectable to a plane (this weakly controls the quality of individual faces). As already observed in [CSAD04], the integral $L^2/L^{2,1}$ metric leads to a very robust and faithful alignment of the geometric clusters to the major anisotropic features of the surface. Even if we run a greedy procedure for the simplification instead of a global Lloyd-relaxation, we can still exploit the same effect and hence obtain polygonal meshes which reliably capture the shape and structure of the input geometry very well. The additional injectivity condition prevents the polygonal faces (= clusters) from degenerating while not affecting the alignment properties.

Next we propose an elegant two-scale framework (Section 3), which allows for modeling of an arbitrary input mesh by simply moving control points as in [ZSS97]. The low-frequency, base domain for performing the modification is a smooth (almost everywhere) C^2 subdivision surface S (Fig. 1). We take advantage of several properties of our simplification scheme, which is used to produce the control mesh of the subdivision surface: Since the user is able to browse the sequence of fine-to-coarse representations of M, a desired domain surface S is available at any level of detail. The control points are placed at intuitive locations and the mesh faces are aligned with the anisotropy of the input surface, hence the support of the modification corresponding to a control point movement is naturally defined and adjusted with the structure of the model. Also, refining the control mesh is not constrained to a uniform refinement of a fixed base domain, but is completely irregular and new degrees of freedom are placed adaptively in accordance to the employed approximation metric. Since our simplification algorithm operates natively with general polygonal faces, both Catmull-Clark [CC78] and Loop [Loo87] surfaces are supported. We focus on the former because quad-dominant control lattices are more relevant for CAD applications.

1.2. Related work

Simplification: Numerous techniques exist for simplifying the complexity of a polygonal mesh, most of them are guided by a specific approximation metric, sometimes augmented by a quality measure for the produced elements. Greedy mesh decimation schemes, such as the one we present, are among the most popular methods. Partitioning through a repeated vertex collapsing was proposed by [SZL92, Hop96, GH97, KLS96]. The dual technique, face clustering, was employed for a high level description of the input geometry by [KT96, GWH01, LPRM02]. Face clustering in the context of mesh generation was specifically explored in [She01, CSAD04, BMRJ04]. A global optimization technique for mesh simplification was proposed in [HDD*93].

With the sole exception of [CSAD04, BMRJ04], all of these techniques target at triangle mesh generation. However, quad-dominant meshes are required for producing high-quality Catmull-Clark subdivision surfaces. Moreover, none of them is able to produce extremely coarse artifact-free meshes required for global, structure aligned deformations. Our simplification scheme is designed to overcome exactly these issues: it produces extremely coarse, structure-preserving, polygonal and artifacts-free meshes.

Modeling: We restrict our survey of free-form modeling techniques to approaches similar to ours, i.e., surface oriented methods. Direct polygonal mesh modeling was proposed by [WW94]. In [ZSS97], an input mesh is converted to a multiresolution Loop surface and deformations are performed interactively by transforming degrees of freedom on different levels of detail. In [LLS01], an arbitrary mesh is converted into a multiresolution Catmull-Clark surface, however, the user has to predefine the base control mesh. To represent the input mesh without resampling, these methods employ local frame encoding using multiple levels of detail.

In contrast to them, variational techniques for polygonal mesh modeling [KCVS98, BK04] use single level of detail to encode the fine scale geometry solely as normal displacements (without resampling). Our approach is positioned in-between these two paradigms: we use a subdivision surface to control the deformation, but a single fine scale is represented as normal displacements. However, unlike displaced subdivision surfaces [LMH00], we do not require resampling of the input geometry and hence avoid aliasing and other under-sampling artifacts. Recently, several alternative ways for performing modeling operations on polygonal meshes were proposed in [YZX*04, SLCO*04, LSCO*04].

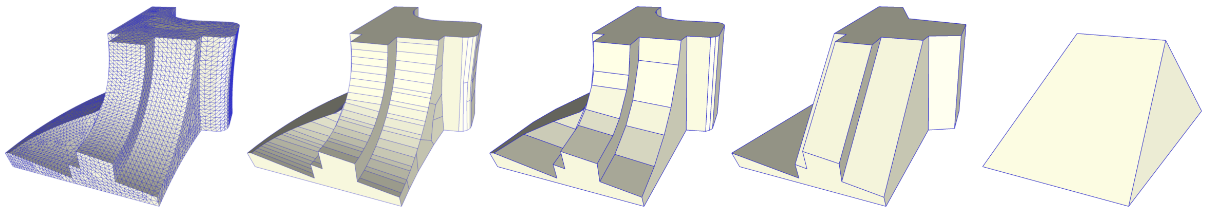


Figure 2: From left to right: *a) The fan model at full resolution, b) 200, c) 50, d) 17, e) 5 regions.*

2. Face-based polygonal mesh decimation

Given an arbitrary two-manifold mesh M_0 as input, our goal is to find coarse polygonal approximations M_i whose faces may be convex or non-convex regions with or without holes. Since we are not restricted to triangles, the faces of our polygonal meshes M_i are not necessarily planar. However, in order to preserve a proper mapping from the input mesh to the decimated ones, we have to guarantee at least a minimum quality of the decimated faces, i.e., we require that for each polygonal face $F_{i,j}$ of M_i, there has to be a plane $P_{i,j}$ such that projecting the boundary loops of $F_{i,j}$ into this plane is an injective mapping. Moreover, we require that inner loops (holes) project into the interior of the outer boundary loop and their projections are not nested.

Instead of using the well-established edge-collapse operation for the decimation, we use its dual, the *face-merge* operation. Although this operation is more complex than the edge collapse, especially for general polygonal faces, we can still define an inverse operation, the *face-split*, such that the hierarchy of coarser and coarser meshes $M_0 \dots M_n$ provides a (generalized) progressive mesh representation. In this hierarchy, we can browse up- and down-stream in constant time to choose the desired level of detail M_i for a given modeling operation.

In our multiresolution shape representation, the coarse meshes M_i serve as subdivision control meshes which are supposed to capture the global structure of the input geometry at a certain level of detail. Hence, we put more emphasis on the global orientation and alignment of the coarse edges and faces rather than minimizing the local geometric deviation between M_i and M_0. This is why we are using a combined *integral* $L^2/\,L^{2,1}$ measure as priority for a greedy decimation scheme: In each step we merge the two polygonal faces that cause the least increase of the $L^2/\,L^{2,1}$- error if the resulting face boundaries are still injectively projectable into a plane.

Polygonal face data structure: Each face $F_{i,j}$ has to store its topological as well as its geometrical information. The geometric information is specified by the mean (area-weighted) normal $N_{i,j}$, the centroid $B_{i,j}$ and the area $\sigma_{i,j}$ of the input surface's region that is associated with $F_{i,j}$. $N_{i,j}$ and $B_{i,j}$ define the plane $P_{i,j}$ which approximates this surface region.

The topological information consists of a set of polygonal loops, one outer loop and maybe one or several inner loops. There are two geometric embeddings for the vertices of these loops. One is a spatial position on the input surface, the second is a planar position in the plane $P_{i,j}$ which is obtained by orthogonal projection. We use the planar embedding to generate a (constrained Delaunay) triangulation $T'_{i,j}$ [BDP*02] of the corresponding planar polygon and by lifting this triangulation into the spatial embedding, we obtain a piecewise linear representation $T_{i,j}$ of the (non-planar) face $F_{i,j}$.

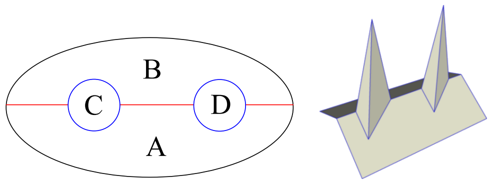


Figure 3: *Merging A and B produces a combined face with three loops: the outer (black) contour and the two inner (blue) holes corresponding to the adjacent faces C and D.*

Face merge: Since we are allowing for arbitrary polygonal faces with holes, the face merge operation can be quite complicated. However, when using a half-edge data structure, the merge operation can easily be implemented in terms of cutting and splicing boundary polygons. If we want to merge two faces $F_{i,j}$ and $F_{i,k}$ into a new face $F_{i+1,l}$ we first have to identify their common boundary segments. After removing these segments, we are left with a set of loops again. One of these loops consists of boundary segments that formerly belonged to the outer loops of $F_{i,j}$ and $F_{i,k}$. This is the new outer loop. All the other loops are inner loops of the merged face $F_{i+1,l}$ (Fig. 3).

As stated above, we are only accepting polygonal faces whose boundary loops project injectively into a plane. Hence, the face-merge operation also computes a new plane $P_{i+1,l}$ as an area weighted average of the two planes $P_{i,j}$ and $P_{i,k}$, i.e., $\sigma_{i+1,l} = \sigma_{i,j} + \sigma_{i,k}$ and

$$N_{i+1,l} = \frac{\sigma_{i,j}N_{i,j} + \sigma_{i,k}N_{i,k}}{\left\|\sigma_{i,j}N_{i,j} + \sigma_{i,k}N_{i,k}\right\|},\, B_{i+1,l} = \frac{\sigma_{i,j}B_{i,j} + \sigma_{i,k}B_{i,k}}{\sigma_{i,j} + \sigma_{i,k}}.$$

If the boundary loops of $F_{i+1,l}$ do not project injectively into this plane, we reject the merge operation.

Merge priorities: Our decimation scheme is a simple greedy algorithm which in each step performs the face merge operation with the highest priority, i.e., the lowest error, that

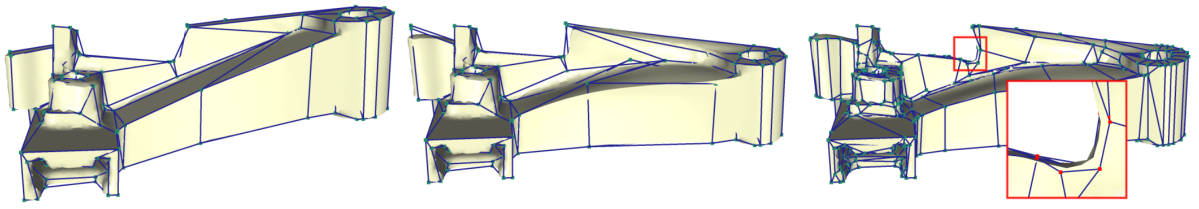


Figure 4: From left to right: *a) A scanned (high genus) technical part. b) On a coarse level the complete right side is pushed down and the left side is extended. c) Switching to a finer scale, we round a sharp corner using the additional controls.*

does not violate the injectivity criterion. Each merge operation is potentially followed by a number of valence two vertex removals. Then the priorities are updated and the next merge operation is selected.

Similarly to [CSAD04] we base our error measure on a combined integral L^2/ $L^{2,1}$- metric. In order to estimate its value for a given (merged) face $F_{i+1,l}$, we integrate the L^2 deviation of the lifted CDT $T_{i+1,l}$ with respect to the proxy plane $P_{i+1,l}$: Let d_0, d_1, d_2 be the distances from the vertices of a triangle $\mathbf{t} \in T_{i,j} \cup T_{i,k}$ from the plane $P_{i+1,l}$. The integral L^2 error of $P_{i+1,l}$ with respect to $\mathbf{t}$ is:

$$L^2(\mathbf{t}, P_{i+1,l}) = \frac{1}{6}(d_0^2 + d_1^2 + d_2^2 + d_0 d_1 + d_0 d_2 + d_1 d_2)\ |\mathbf{t}|$$

as shown in [CSAD04]. Finally we estimate the total L^2 error for the face $F_{i,j}$ as:

$$L^2(F_{i+1,l}) = \sum_{t \in T_{i,j}} L^2(t, P_{i+1,l}) + \sum_{t \in T_{i,k}} L^2(t, P_{i+1,l}) .$$

The $L^{2,1}$ deviation is estimated by computing an area weighted sum of the deviation between the normals before and after the merge:

$$L^{2,1}(F_{i+1,l}) = \sigma_{i,j} \cdot \left\| N_{i,j} - N_{i+1,l} \right\|^2 + \sigma_{i,k} \cdot \left\| N_{i,k} - N_{i+1,l} \right\|^2 .$$

Notice that both error measures are only approximations to the true errors with respect to the original input mesh M_0. However, since the integral has a low-pass filtering effect anyway and since we never noticed a significant difference in the quality of the resulting coarse meshes M_i when using the true measure, we decided to use the approximate measure which can be evaluated much more efficiently, in constant time.

Since a weighted sum of the two error measures is not scale independent, we rather combine the two measures by multiplying them so that the $L^{2,1}$ metric acts as a weighting factor for the L^2 metric and vice-versa:

$$E(F_{i+1,l}) = \left[1 + L^2(F_{i+1,l})\right] \cdot \left[1 + L^{2,1}(F_{i+1,l})\right] . \qquad (1)$$

Contour decimation: The face-merge operation does not remove any vertices from M_i. Hence we need a second operation to actually reduce the geometrical complexity. In our algorithm, this operation is simply removing the valence two vertices that appear after several faces have been merged (Fig 5).

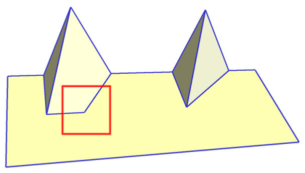


Figure 5: *To decimate the contours of the merged regions we remove all excess valence two vertices.*

For closed meshes, we decimate every valence two vertex v_k as soon as it appears. For meshes with boundary we have to take some error measure into account in order to preserve the shape of the input surface's boundary. As there is only one face incident to a boundary valence two vertex, we simply compute the corresponding L^2 error integral in closed form by

$$L^2(v_k) = \frac{1}{3} \left\| v_j - v_l \right\| d^2 ,$$

where v_j and v_l are the two adjacent vertices to v_k and d is the orthogonal distance of v_k to the edge $\overline{v_j v_l}$. The $L^{2,1}$ error vanishes because the face normal does not change after the vertex removal, hence $E(v_k) = 1 + L^2(v_k)$.

In any case, in order to prevent fold-overs in non-convex faces or in faces with holes, we allow removing any v_k only if the injectivity criterion is true for the adjacent faces after the removal. Moreover, in order to avoid inconsistencies in the mesh M_{i+1}, the removal operator is permitted only if the edge (v_j, v_l) did not already exist in the mesh M_i.

Performance considerations: The most time consuming step in the decimation procedure is the construction of the CDT $T_{i,j}$ after every merge or removal operation. The $T_{i,j}$ is necessary to evaluate the approximate $L^2/L^{2,1}$ error measures. The complexity of this step grows like $O(m \log m)$ with the valence m of the polygonal face $F_{i,j}$. This can lead to performance problems in flat surface regions where the error measure (1) is very small and faces with very high valence can occur. In order to avoid these pathological configurations, we add another criterion to the decimation scheme which prefers low valence faces. However, this criterion has to be designed in such a way that it does not influence the decimation scheme in regions where the error measure (1) has non-vanishing values.

Consequently, we add two more factors $(1+\varepsilon(m-4)^2)$ and $(1+\varepsilon\sum_q(\alpha_q-\pi/2)^2)$ with some small weight coefficient ε to the combined error measure (1) where m is the face valence and α_q are the inner angles at the vertices incident to the removed boundary segment(s) shared by $F_{i,j}$ and $F_{i,k}$. These two factors penalize high valences and non-regular inner angles. However, due to the small weight coefficient ε these factors can influence a greedy decision only if the other factors are close to constant and hence they act like a "tie-break". Note that these penalty terms are not meant to enforce the quality of the mesh elements in general — we need them only to prevent uncontrolled growth of the CDTs computation time in regions where the input mesh is nearly planar.

3. Multiresolution modeling

Subdivision control mesh generation: Once the input mesh M_0 is simplified to the coarsest possible level M_l, the user is able to browse the resulting hierarchy and choose a level of detail (an intermediate mesh) M_i which we use to define the base domain for subsequent modeling operations. We convert the selected M_i into a proper subdivision control mesh L_i defining a (limit) surface S_i. For Catmull-Clark surfaces, we have to construct L_i in a way that guarantees fair limit surfaces. In particular this means that we have to avoid non-convex faces and faces with high valence and that we prefer quad-dominant meshes. Since we do not want to insert additional vertices, we have to find a convex partitioning without Steiner points for each face $F_{i,j}$. This partitioning has to be applied to the planar embedding of $F_{i,j}$ and is then lifted to its spatial embedding. The convex partitioning algorithms available in the literature [HM83, KS98] do not provide the required solution since they do not take the valence of the generated convex faces into account.

Suppose Γ is some convex partitioning of a polygonal face $F_{i,j}$. We define a score function ω for every cell $f\in\Gamma$: $\omega(f)=0$ if $val(f)=4$, $\omega(f)=1$ if $val(f)=5$, $\omega(f)=2$ if $val(f)=3$ and $\omega(f)=\infty$ otherwise. Valence 5 cells are preferred due to the better quality of the subdivision limit surface compared to valence 3 cells. The score of the complete partitioning Γ is defined as $\omega(\Gamma)=\sum_{f\in\Gamma}\omega(f)$.

We are interested in finding a partition Γ which minimizes $\omega(\Gamma)$. The already computed CDT $T_{i,j}$ of $F_{i,j}$ exhibits angle optimality with respect to the choice of available diagonals in the planar embedding of $F_{i,j}$. Hence, similarly to [HM83] we restrict our convex partitioning to diagonals existing in $T_{i,j}$. Since the number of vertices in $F_{i,j}$ is usually small, a brute force approach with branch caching rapidly finds the optimal Γ by simply checking all possible decompositions. If $F_{i,j}$ happens to have higher valence than a certain threshold (20 vertices), we fix several diagonals from $T_{i,j}$ in order to split $F_{i,j}$ into a few smaller polygons and run the brute-force algorithm on each of them.

Normal displacement: In order to represent the input mesh M_0 as a normal displacement of the subdivision limit surface S_i, we need to find for every vertex $v_k\in M_0$ a foot-point $S_i(t_k)$ on the surface S_i such that the displacement vector $v_k-S_i(t_k)$ is orthogonal to the tangent plane of S_i at $S_i(t_k)$. Since there is no analytical solution to this problem, we first have to generate some initial estimate r_k for t_k and then iteratively improve this estimate by an exact closest point search on S_i [MK04]. The convergence of the closest point search depends on the quality of the initial estimate r_k. Hence, we generate these estimates by computing a global parameterization of M_0 which uses the subdivision control mesh L_i as a domain. Notice that our conversion procedure from M_i to L_i is based on a convex partitioning of the planar embedding of each face $F_{i,j}$. As a consequence, all we have to do is to map this partitioning back onto the input mesh M_0 and then compute an individual parametrization for each of the resulting patches (Fig. 7).

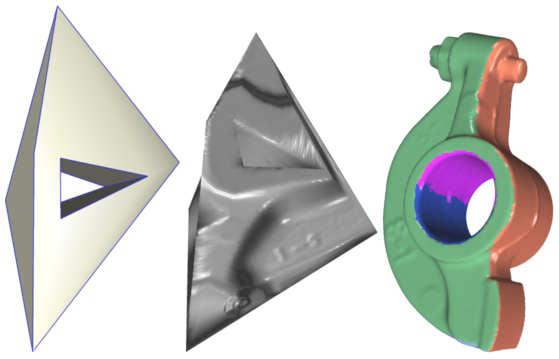


Figure 7: Left: *The Rocker Arm model simplified to 5 regions.* Middle: *The global map corresponding to this resolution (visualized as normal map).* Right: *The corresponding clusters.*

More precisely: Since the decimated faces $F_{i,j}$ have been generated by face-merging, there is a unique surface patch $C_{i,j}$ on M_0 which is associated with $F_{i,j}$. In particular, the boundary of the patch $C_{i,j}$ is associated with the planar embedding of the boundary of $F_{i,j}$. The diagonals used in the convex partitioning of $F_{i,j}$ can be mapped to shortest paths (inside $C_{i,j}$) between the corresponding vertices of M_0. These shortest paths are easily computed by Dijkstra's algorithm. If several diagonals are adjacent to the same boundary vertex of $F_{i,j}$ then we might have to locally refine M_0 in order to keep the shortest paths disjoint (Fig. 8). After we have fixed the parameterization of all vertices of $C_{i,j}$ mapped to the edges of the convex partitioning of the parameter domain Γ, we setup a sparse linear system for each cell to compute the parameterization of the remaining vertices using Floater's mean value weights [Flo03]. Then for every $v_k\in M_0$ we improve the initial parameter value r_k by

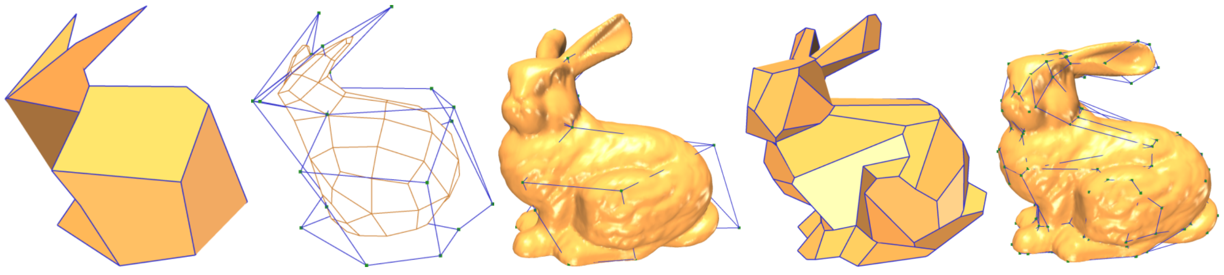


Figure 6: From left to right: *a) The bunny model simplified to* 14 *regions. b) The corresponding control mesh and its first level Catmull-Clark subdivision. c) Using b) large scale modifications are applied to the back and the head. d) A finer resolution (83 regions) is selected. e) The new control vertices are used to alter the ears.*

iteratively updating it towards the exact foot-point parameter value t_k [MK04].

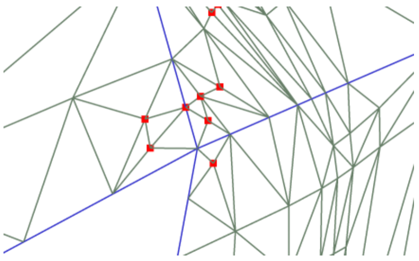


Figure 8: *Close-up view on a parameterization around the hole of the Rocker arm model. The map constraints are setup on the blue lines. Note the vertices (red points) refining M_0 so that disjoint vertex paths corresponding to the diagonals of the domain partition are possible.*

Two-scale model: We define the *two-scale* model Q_i which represents the vertex positions of the original input mesh M_0 as normal displacements D_i with respect to S_i. More precisely, Q_i is the collection of the following entities: the subdivision surface S_i (*base scale*), the parameter values t_k of the foot-points of the vertices of M_0 on S_i (*parameterization*) and the normal displacements d_k such that

$$v_k = S_i(t_k) + d_k \cdot N_i(t_i)\,, \tag{2}$$

where $v_k \in M_0$ and $N_i(t_k)$ is the normal of S_i at t_k (*fine scale*).

Editing: By moving control vertices of the control mesh L_i, the user deforms the two-scale model Q_i in a region defined by the resolution of L_i, i.e., in the region corresponding to the support of the respective subdivision basis functions. Since our decimation scheme produces structure-aligned meshes M_i, the distribution of subdivision control vertices as well as the shape of the corresponding basis function's support are well-adapted to the shape features of the underlying input surface. In this sense, our automatically generated multiresolution representation allows modifications which align to the geometric structure of the input model M_0.

Note that while initially the two-scale model Q_i represents the geometry of the of the input mesh M_0, any subsequent deformations are implicitly encoded by the (modified) positions of the control points of L_i. Hence, the deformed positions of the fine scale vertices can be always computed by evaluating (2).

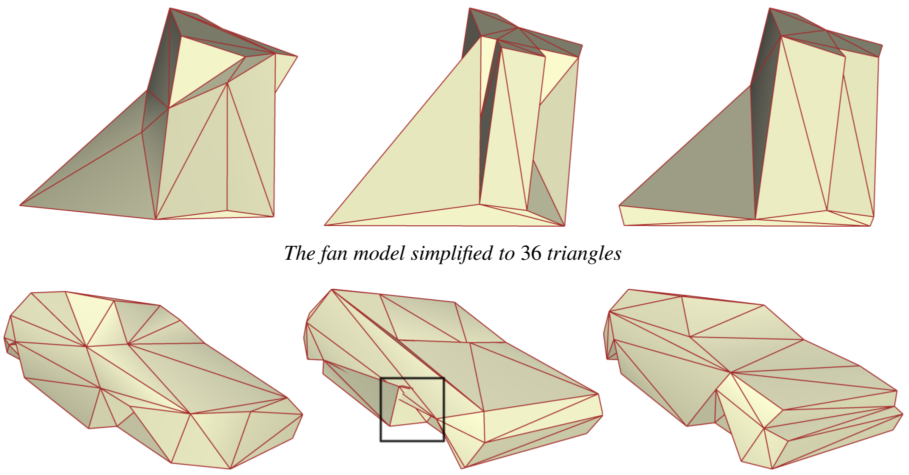
Resolution change: If another modification should be made on a different scale, i.e., another level of detail M_j is chosen by the designer, we first convert M_j to a subdivision control mesh L_j as before. Since the current two-scale model Q_i might have been deformed, we first propagate this deformation on the new model Q_j. To do that, we simply obtain the vertex positions of the fine scale vertices v_k by evaluating (2). Note that since every control vertex of L_j corresponds to a fine scale vertex (our decimation procedure is a subsampling process) the control vertices of L_j are implicitly shifted to their new (possibly deformed) positions. Accordingly the new normal displacements D_j are computed with respect to the new base surface S_j (Fig. 4, 6).

4. Results

We tested our algorithm on various models, with emphasis on CAD objects: Fig. 1, 2, 4. Recall that our simplification scheme discards the best possible merge operation in terms of approximation error, if its result does not meet the minimum requirement of injective projection as specified in Section 2. Not surprisingly, this approach leads to a slightly suboptimal approximation quality when compared to established methods such as [GH97] and [CSAD04]. However, we trade a small increase of the approximation error for the possibility to generate much coarser meshes which are still consistent (Fig. 9).

In fact the initial prototype of our modeling framework used [CSAD04] to generate low complexity (control) meshes. However, since we did not find a way to produce unconditionally artifact-free meshes using this method, we developed the presented simplification algorithm. We intentionally avoided dependence of any user-specified parameters, hence our decimation is fully automatic and not a "trial and error" experience. The decimation stops when there are no more possible valid simplification steps to be performed.

When compared to previous approaches, our modeling


The fan model simplified to 36 triangles

The car model simplified to 92 triangles

Figure 9: From left to right: *a)* [GH97], *b)* [CSAD04], *c) our result. The QEM decimation produces acceptable meshes, but important features are smoothed out. On the other side the VSA meshes capture precisely the geometry structure, however at so coarse level of detail, exhibit artifacts and fold-overs. These issues can be resolved using more proxies, but at the expense of additional triangles. Our algorithm preserves the structure of the original models, without introducing any artifacts, even on such low resolutions.*

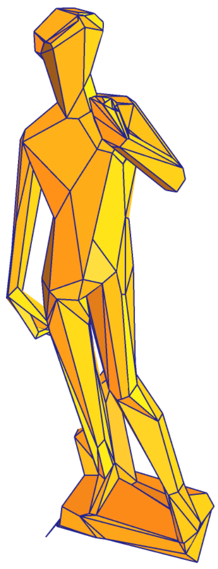


Figure 10: *Michelangelo's David simplified by our method (395 vertices, 481 convex faces, 235 quads).*

framework exhibits several advantages. Switching to finer levels in the hierarchy is not constrained to dyadic points as for multiresolution subdivision surfaces [ZSS97]. Therefore less degrees of freedom are necessary to control the surface locally, which in turn allows for more natural modifications using fewer controls (Fig. 11). Note that a dyadic refinement of a current (sub) set of control vertices is still straightforward, in case one needs it at all. Also the support of the modification is intuitive, since the control faces are adjusted to the structure of the mesh. In addition, our framework is conceptually simpler since only plain subdivision surfaces are used, instead of the more complex multiresolution subdivision surfaces [Zor97]. Compared to [LMH00], the advantage of our approach is that we do not have to re-sample the input mesh and hence avoid alias artifacts. The supplemental video provided with the paper demonstrates editing operations on the fan model and the car model using our framework.

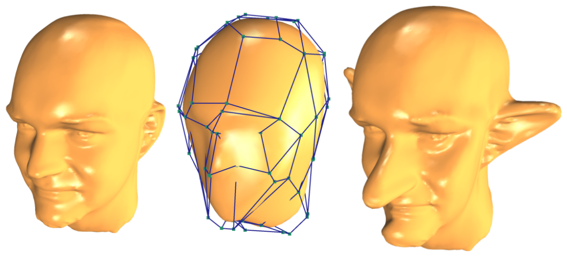


Figure 11: From left to right: *a) A scanned male head. b) An intermediate base Catmull-Clark surface. c) The final result after several modifications. Note the new position of the left ear - it was moved by just translating a few control vertices of b). On a finer resolution, both of the ears were bended.*

Beside modeling, the coarse meshes produced by our simplification technique can be exploited in a number of different applications, e.g., re-sampling and approximation. For example the control mesh on Fig. 6b has 25 vertices and 23 faces (9 quads, 7 pentagons, 7 triangles) and still captures the global shape of the bunny including the most problematic region: its ears. The mesh subdivided once using Catmull-Clark subdivision consists of 92 faces, all quads, i.e., its complexity is 60% of the coarsest control mesh of the bunny model presented in [BMRJ04]. On the other side, the finer control meshes which our method produces are not particularly optimal with respect to the number of extraordinary control vertices. However, this was never a problem in practice, since for our purposes C^1 continuity is fully sufficient. Also, with the availability of techniques improving the surface smoothness in vicinity of extraordinary vertices [Loo04], this issue becomes less and less relevant for other applications as well.

Timings: For most models our simplification algorithm performs in-between the QEM simplification and the VSA approach. Despite the greedy nature of the algorithm, the construction of a CDT in every simplification step increases the computation cost considerably. Our "prototype" implementation takes about 3*min* for producing the simplification of Fig. 10, starting with a mesh consisting of 413*K* triangles. The simplification times for the Rocker Arm (80*K* faces), the car model (60*K* faces) and the fan model (13*K* faces) are respectively 34*sec*, 26*sec* and 6*sec*. The computation time required for the parameterization of the input mesh on a base domain (including the normal displacement computation) depends not only on the complexity of the input model, but also on the coarseness of the base mesh. Hence, it varies between several seconds for low complexity input meshes (4*sec* for the technical part in Fig. 4 — 18*K* faces) and a few minutes for dense models parameterized over very coarse base domains, e.g., 1*min* for the bunny (74*K* faces) over the domain in Fig. 6b. Timing are taken on P4, 2.8*GHz* system.

5. Future work

One of the possible directions for improving our work is combining a uniform and an irregular refinement [GKSS02] to build automatically a hierarchy which more closely conforms to the standard for a CAD model, i.e., large regular patches cover feature-less regions of the surface. Another interesting route is topology removal. In fact, by introducing an additional operator, our decimation is able to reduce the input topology in certain cases, however for now we did not investigate that direction any further.

References

[BDP*02] Boissonnat J.-D., Devillers O., Pion S., Teillaud M., Yvinec M.: Triangulations in CGAL. *Comput. Geom. Theory Appl. 22* (2002), 5–19.

[BK04] Botsch M., Kobbelt L.: An intuitive framework for real-time freeform modeling. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 630–634.

[BMRJ04] Boier-Martin I., Rushmeier H., Jin J.: Parameterization of triangle meshes over quadrilateral domains. In *Proc. of the Symposium on Geometry Processing* (2004).

[CC78] Catmull E., Clark J.: Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Geometric Design 10*, 6 (Sep 1978), 350–355.

[CSAD04] Cohen-Steiner D., Alliez P., Desbrun M.: Variational shape approximation. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 905–914.

[Flo03] Floater M. S.: Mean value coordinates. *Computer Aided Geometric Design 20*, 1 (Mar. 2003), 19–27.

[GH97] Garland M., Heckbert P. S.: Surface simplification using quadric error metrics. In *Proc. of SIGGRAPH 97* (Aug. 1997), pp. 209–216.

[GKSS02] Guskov I., Khodakovsky A., Schröder P., Sweldens W.: Hybrid meshes: multiresolution using regular and irregular refinement. In *Proc. of the Symposium on Comput. Geometry* (2002), pp. 264–272.

[GWH01] Garland M., Willmott A., Heckbert P. S.: Hierarchical face clustering on polygonal surfaces. In *2001 ACM Symposium on Interactive 3D Graphics* (Mar. 2001), pp. 49–58.

[HDD*93] Hoppe H., DeRose T., Duchamp T., McDonald J., Stuetzle W.: Mesh optimization. In *Proc. of SIGGRAPH 93* (Aug. 1993), pp. 19–26.

[HM83] Hertel S., Mehlhorn K.: Fast triangulation of simple polygons. In *Proc. of the FCT-Conference on Fundamentals of Computation Theory* (1983), pp. 207–218.

[Hop96] Hoppe H.: Progressive meshes. In *Proc. of SIGGRAPH 96* (Aug. 1996), pp. 99–108.

[KCVS98] Kobbelt L., Campagna S., Vorsatz J., Seidel H.-P.: Interactive multi-resolution modeling on arbitrary meshes. In *Proc. of SIGGRAPH 98* (July 1998), pp. 105–114.

[KLS96] Klein R., Liebich G., Strasser W.: Mesh reduction with error control. In *IEEE Visualization '96* (Oct. 1996), pp. 311–318.

[KS98] Keil M., Snoeyink J.: On the time bound for convex decomposition of simple polygons. In *Proc. of the Canadian Conference on Comput. Geometry* (1998).

[KT96] Kalvin A. D., Taylor R. H.: Superfaces: Polygonal mesh simplification with bounded error. *IEEE Comput. Graph. Appl. 16*, 3 (1996), 64–77.

[LLS01] Litke N., Levin A., Schröder P.: Fitting subdivision surfaces. In *IEEE Visualization 2001* (October 2001), pp. 319–324.

[LMH00] Lee A., Moreton H., Hoppe H.: Displaced subdivision surfaces. In *Proc. of SIGGRAPH 00* (2000), pp. 85–94.

[Loo87] Loop C.: *Smooth subdivision surfaces based on triangles*. Master's thesis, University of Utah, 1987.

[Loo04] Loop C.: Second order smoothness over extraordinary vertices. In *Proc. of the Symposium on Geometry Processing* (2004).

[LPRM02] Lévy B., Petitjean S., Ray N., Maillot J.: Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics 21*, 3 (July 2002), 362–371.

[LSCO*04] Lipman Y., Sorkine O., Cohen-Or D., Levin D., Rössl C., Seidel H.-P.: Differential coordinates for interactive mesh editing. In *Proc. of Shape Modeling International* (2004), pp. 181–190.

[MK04] Marinov M., Kobbelt L.: Optimization techniques for approximation with subdivision surfaces. In *ACM Symposium on Solid Modeling and Applications* (2004), pp. 113–122.

[She01] Sheffer A.: Model simplification for meshing using face clustering. *Computer Aided Design 33* (2001), 925–934.

[SLCO*04] Sorkine O., Lipman Y., Cohen-Or D., Alexa M., Rössl C., Seidel H.-P.: Laplacian surface editing. In *Proc. of the Symposium on Geometry processing* (2004), pp. 179–188.

[SZL92] Schroeder W. J., Zarge J. A., Lorensen W. E.: Decimation of triangle meshes. In *Proc. of SIGGRAPH 92* (July 1992), vol. 26, pp. 65–70.

[WW94] Welch W., Witkin A.: Free-form shape design using triangulated surfaces. In *Proc. of SIGGRAPH 94* (July 1994), pp. 247–256.

[YZX*04] Yu Y., Zhou K., Xu D., Shi X., Bao H., Guo B., Shum H.-Y.: Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 644–651.

[Zor97] Zorin D.: *Subdivision and Multiresolution Surface Representations*. PhD thesis, Caltech, Pasadena, California, 1997.

[ZSS97] Zorin D., Schröder P., Sweldens W.: Interactive multiresolution mesh editing. In *Proc. of SIGGRAPH 97* (Aug. 1997), pp. 259–268.