Cut-and-Paste Editing of Multiresolution Surfaces

Henning Biermann†, Ioana Martin‡, Fausto Bernardini‡, Denis Zorin†

†Media Research Laboratory
New York University

‡IBM T. J. Watson Research Center

Abstract

Cutting and pasting to combine different elements into a common structure are widely used operations that have been successfully adapted to many media types. Surface design could also benefit from the availability of a general, robust, and efficient cut-and-paste tool, especially during the initial stages of design when a large space of alternatives needs to be explored. Techniques to support cut-and-paste operations for surfaces have been proposed in the past, but have been of limited usefulness due to constraints on the type of shapes supported and the lack of real-time interaction. In this paper, we describe a set of algorithms based on multiresolution subdivision surfaces that perform at interactive rates and enable intuitive cut-and-paste operations.

1 Introduction

Pasting and blending of images are among the most common operations implemented by image manipulation systems. Such operations are a natural way to build complex images out of individual pieces coming from different sources. For example, photographs can be easily combined with hand-drawn and computer-generated images. In contrast, pasting and blending tools are hardly available for surfaces. Most geometric modeling systems expect the user to manipulate control points of NURBS, individual mesh vertices and polygons, or use conventional, higher-level operations such as volume deformations and boolean operations. In an image processing system, vertex and control point manipulation would be equivalent to painting an image pixel-by-pixel. While it may be useful to have access to such low-level operations in certain cases, most image manipulations are done using higher-level tools.

In this paper we describe a technique for interactive cut-and-paste editing of surfaces, an important instance of a natural operation on a surface (see Figure 1 for an example). The algorithms we propose enable a number of useful design scenarios which are difficult to perform using existing technology. For example, in the design of automobile body parts, it is common to work in parallel on a digital mock-up and on a clay model. Using the cut-and-paste technique, a designer can paste a logo obtained by 3D scanning onto a digitally-modeled surface, import features from a library of predefined shapes, or copy parts of a design from a different project.

The basic idea of pasting is quite simple. The user selects an area of interest on the source surface. Both the source and the target surfaces are separated into *base* and *detail*, such that the detail surface represents a vector offset over the base surface. Next, the user specifies a location and an orientation on the target surface where the source feature is to be pasted and interactively adjusts the position, orientation, and size of the pasted feature. The main questions we address in this paper are:

- How to separate a surface into base and detail?
- How to identify an area on the target surface where the feature should be pasted and how to establish the necessary mappings between the source and the target?
- How to implement the process efficiently to allow for interactive pasting of complex features?

We use multiresolution subdivision surfaces as our underlying representation [28, 44]. The actual computer representation is a semi-regular control mesh for the surface and most operations are performed on this mesh. The associated limit surface is used for computing quantities such as tangents and normals, as well as for additional refinement when necessary for antialiasing. This is similar to pixel representations of images: when images are scaled or rotated, they are typically assumed to be sampled representations of smoothly varying continuous images (e.g., obtained by cubic interpolation).

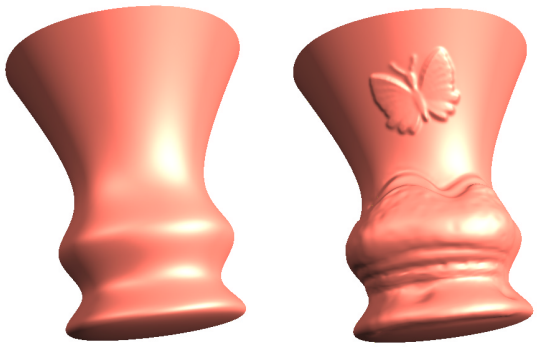


Figure 1: An example of a pasting operation.

The regular and hierarchical structure of this surface representation makes it possible to perform operations on detailed surfaces at interactive rates as discussed in Section 3.2. In many ways, using this representation makes surface manipulation similar to image manipulation: almost everywhere the connectivity of the mesh approximating the surface is locally regular. At the same time, many common problems specific to geometry have to be addressed: the lack of a common parameterization domain for separate surfaces, the lack of a unique best parameterization domain for the surfaces, the separation of surface features. While our algorithms can be applied to a broad class of surfaces, there are limitations on the source and target geometry for which the pasting paradigm is appropriate (see Section 9 for a discussion).

2 Previous Work

The concept of surface pasting was introduced in the work of Bartels, Mann and co-workers in the context of hierarchical splines [3, 7, 8, 29, 42]. We are using similar ideas, but most of the technical details are different. Most importantly, we consider more general surface types and we do not assume that separate detail and base surfaces are given: they need to be extracted from the input surfaces.

Moving existing features on a mesh was explored by Suzuki et al. [41]. The advantage of their approach is that no resampling of the repositioned feature is performed. However, continuous remeshing is required, which limits the complexity of the objects and features that can be handled. Issues such as pasting features between surfaces and separation into base and detail surfaces are not considered by these authors.

The task of base/detail separation is similar to the construction of displaced subdivision surfaces [22]. One of the elements of our

approach, i.e., mesh smoothing to extract a base surface, was described by Kobbelt et al. [18] in the more general context of arbitrary meshes. An alternative approach was proposed by Guskov et al.[14]. We discuss the advantages and disadvantages of restricting the class of surfaces to semi-regular meshes in Section 3.2. Part of our construction of base surfaces is closely related to the work of Kobbelt et al. on variational subdivision [19, 17]. It also draws upon the work of Polthier et al. [34, 31].

Parameterization techniques are important in many geometric modeling and texturing applications and a variety of algorithms have been proposed, including general parameterization methods [10, 11] for reparameterization (i.e., changing connectivity to semi-regular) [9, 15, 20, 23] and texture mapping [25, 36, 30]. In [21], Kuriyama and Koneko use local parameterizations to add offsets to a surface. The work of Pedersen [32, 33] on interactively placing textures on implicit surfaces is also relevant as it requires dynamic reparameterization of surface areas similar to pasting.

However, the problem of parameterizing a surface area over a plane with minimal visual distortion is far from solved. As explained in Section 6, until recently no algorithms combining several crucial properties for our application were available. We use a variation of the remarkable algorithm by Sheffer and Sturler [39] which satisfies our requirements.

3 Pasting Surfaces

We begin with a formalized description of pasting operations on surfaces. At this point we discuss continuous surfaces and mappings without considering their discrete representations. This framework applies to a wide class of manifold surfaces, ranging from splines to implicit surfaces. Precise descriptions of all basic mathematical concepts that we use can be found in any standard textbook (e.g., [43]).

3.1 Formulation of the Problem

For simplicity, we restrict our attention to parts of surfaces parameterized over planar domains: $M \subset \mathbf{R}^2$. Furthermore, we assume that these parameterizations are sufficiently smooth. Given two surfaces $(M_1, \mathbf{f}_1)$ and $(M_2, \mathbf{f}_2)$, where $\mathbf{f}_1$ and $\mathbf{f}_2$ are their parameterizations, we would like to paste a feature from one surface to the other (see Figure 1). Such an operation requires separating each surface into two parts: the *base* surface and the *detail* surface. The goal is to replace the detail part of the second surface with the detail part of the first. The key question is how to transfer correctly the details from one surface to the other.

Base and detail surfaces. The base surface $\mathbf{b}(x)$ is typically a smoothed or flattened version of the original surface (we discuss appropriate choices in Section 5). The detail surface $\mathbf{d}(x)$ can be defined as $\mathbf{f}(x) - \mathbf{b}(x)$. However, to ensure that the offset direction is at least invariant with respect to rigid transformations of the base, it must be represented in a local frame. The local frame is a triple of vectors $(\mathbf{n_b}, \partial_1\mathbf{b}, \partial_2\mathbf{b})$, including the normal and two tangents (two partial derivatives of the parameterization). It is convenient to think about these derivatives together as a map $D\mathbf{b}$ (differential of $\mathbf{b}$) which maps vectors in the plane to vectors in the tangent plane of the surface. The detail surface is thus defined by the triple d^n, d^{t1}, d^{t2}, which can also be thought of as a scalar displacement along the normal d_n and a tangential displacement in parametric coordinates $\mathbf{d}^t = (d^{t1}, d^{t2})$. The equation relating the original surface, the base, and the details can be written as: $\mathbf{f}(x) = \mathbf{b}(x) + D\mathbf{b}(x)\mathbf{d}^t(x) + \mathbf{n_b}(x)d^n(x)$, where x is a point in the domain.

Surface pasting. With both surfaces separated into base and detail parts, we can formulate a precise definition of pasting. All quantities with index 1 refer to the source surface from which we extract the details and all quantities with index 2 refer to the target surface on which the details are pasted.

Suppose the part of the surface we want to paste is defined over $G_1 \subset M_1$. Let p be a map from G_1 to M_2, which defines how the surface is pasted. We discuss separately how p is chosen (Section 6).

The result of a simple pasting operation is a new surface coinciding with $\mathbf{f}_2$ outside $p(G_1)$, which has the same base as $\mathbf{f}_2$ but for which the details are taken from the source surface:

$$\mathbf{f}^{pasted} = \mathbf{b}_2 + \left(D\mathbf{b}_2\, Dp\, \mathbf{d}_1^t + n_{\mathbf{b}_2} d_1^n\right) \circ p^{-1}$$

where all functions are evaluated at a point $x_2 \in p(G_1)$, and $\circ$ denotes function composition.

Note that we use the composition of differentials $D\mathbf{b}_2 \circ Dp$ to transform the tangential component of details. This establishes the natural map between the local frames on the source and target surfaces. Figure 2 illustrates the different maps involved.

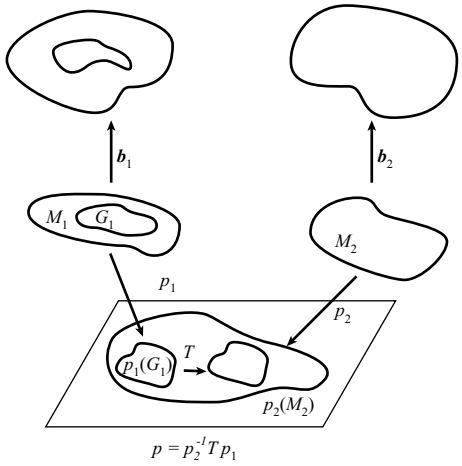


Figure 2: A diagram of surface maps involved in pasting.

Using this formulation, there are two main choices to be made: the separation of both source and target surfaces into base and detail and the definition of a pasting mapping p, identifying the domain G_1 with a part of the domain M_2.

The map p has to satisfy two conditions: it has to be one-to-one and it should minimize distortion of the mapped feature. An important consideration is whether the mapping p from the domain of one surface to the domain of the other surface is constructed directly or by using an intermediate planar domain. We favor the latter approach as it considerably simplifies three tasks: making sure that the mapping is visually smooth, minimizing distortion, and resampling the source over the target sampling pattern. To explain our choices, we need to be more specific about the surface representation we are using.

3.2 Multiresolution Subdivision Surfaces

The representation that we use was introduced in various forms in [28, 38, 45]. Subdivision defines a smooth surface recursively as the limit of a sequence of meshes.[1] Each finer mesh is obtained from a coarse mesh by using a set of fixed refinement rules, e.g., Loop [27] or Catmull-Clark [6] subdivision rules. In our implementation we

[1]To be more accurate, we should say that the limit surface is the pointwise limit of a sequence of piecewise linear functions defined on the initial control mesh.

use Catmull-Clark rules. Multiresolution surfaces extend subdivision surfaces by introducing *details* at each level. Each time a finer mesh is computed, it is obtained by adding detail offsets to the subdivided coarse mesh. If we are given a *semi-regular mesh*, i.e., a mesh with subdivision connectivity, we can easily convert it to a multiresolution surface if we define a smoothing operation to compute vertices on a coarse level from a finer level. The details are then computed as differences between levels (see Section 5).

An aspect of multiresolution surfaces important for modification operations is that details are represented in local coordinate frames, which are computed from the coarser level. This is analogous to representing the detail surface in the frame computed from the base surface.

For our purposes, it is important to interpret the multiresolution surface as a function on a domain. A multiresolution subdivision surface can be naturally viewed as a function on the initial mesh as shown in Figure 3.

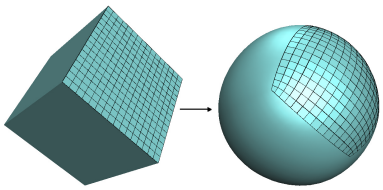


Figure 3: Natural parameterization of the subdivision surface. Each time we apply the subdivision rules to compute the finer control mesh we also apply midpoint subdivision to a copy of the initial control mesh. As we repeatedly subdivide, we get a mapping from a denser and denser subset of the control polygon to the control points of a finer and finer control mesh. In the limit we get a map from the control polygon to the surface.

Advantages and disadvantages of the representation. The main reason behind our choice of representation is efficiency. There are a number of reasons why semi-regular meshes allow for highly efficient algorithms:

• Connectivity information only needs to be stored for the coarsest level. Geometric data is stored in a regular and space-coherent manner. Both factors are important for computer architectures for which bad cache behavior results in poor performance. In addition, regularly sampled patches can be rendered very efficiently.
• Our meshes have a built-in natural hierarchy that can be exploited by numerical solvers. These are used, for example, to define a family of smooth surfaces for base surface selection by hierarchical fitting and for parameterization, when an initial approximation of the fine level solution can be obtained by solving the system on a coarse level, followed by refining the solution by subdivision.
• Compact representation for smooth surfaces: for example, the original vase in Figure 1 (left) is completely defined by the initial mesh and the smooth surface can be recomputed on-the-fly. When details are added, additional refinement is needed only in the regions with details.
• Local frames can be computed in a simple, fast, and reliable way, consistently across resolution levels (i.e., refining the mesh for the base surface does not change the frames computed for the surface at existing vertices).

Our experiments with solvers that take advantage of the regular structure and generic solvers that use a sparse matrix representation suitable for arbitrary meshes show that the former yield speedups by a factor of 2 to 4. Furthermore, using a hierarchical solver for an arbitrary mesh would require building a hierarchy by simplification, a step entirely omitted in our construction.

The main disadvantage of representing surfaces using semi-regular meshes is having to convert surfaces represented by arbitrary meshes to this format. Fortunately, considerably progress has been made in this area [20, 23, 15] and commercial software (e.g., Paraform [1], Geomagic Studio [2]) typically includes such conversion tools. All of the scanned models used in this paper were converted to semi-regular meshes using Geomagic Studio. We believe that, in all cases when surface data is extensively modified, conversion is the best approach, as reparameterization is almost inevitable if the surface is texture-mapped. Gu et al. provide a detailed study of the benefits of a conversion to a similar representation (i.e., the geometric image [12]).

3.3 Pasting with an Intermediate Plane

A direct construction of the pasting mapping p is difficult to achieve efficiently. Visual smoothness and minimization of distortion are typically obtained by minimizing appropriate functionals. In the case of a direct mapping of the source region to the target surface domain the values of the mapping are not a part of any affine space. Indeed, the domain of the surface is a collection of faces of the coarse-level control mesh, so each point needs to be characterized as (i, u, v) where i is the face id, and (u, v) are coordinates within the face. Unless the whole surface can be reparameterized on a plane, there is no simple way to compute linear combinations of two arbitrary points (e.g., the midpoint of the interval connecting the points), which makes the application of most common computational techniques very difficult. Even a simple operation such as computing angles of a triangle given three vertices becomes a complicated task, an important consideration for the angle-based flattening technique we consider.

To avoid these difficulties, we parameterize the corresponding areas of the source and target over the plane. The idea is to map each surface onto the plane as isometrically as possible and then align the two planar parameterizations, using a linear transformation to compensate for the first-order distortion. In this case, the pasting map is restricted to a simple class of maps (i.e., linear transformations T, see Figure 2), but new parameterizations p^1 and p^2 are constructed for the parts of surfaces of interest for every pasting operation.

There is a similarity between the idea of our approach and the method of Praun et al. [37] for establishing correspondences between different meshes. In [37] the correspondence is established by reparameterizing each mesh on the same base domain. Given our disk topology assumption, we can use the plane as the common domain.

Our approach has two main disadvantages. First, it makes it difficult to generalize our technique to pasting regions with topology different from that of a subset of a plane (e.g., pasting all details from one sphere to another). Second, it may result in higher distortion than a direct mapping from one surface to the other. The higher the Gaussian curvature of the base surface is, the more likely it is that additional distortion is introduced. A direct mapping method similar to the one used in [4] might produce better results in this case, but it would make pasting of complex surfaces at interactive rates difficult, if at all possible.

4 Overview of the Algorithm

The main steps of our algorithm are illustrated in Figure 4:

1. The user marks a region on the source surface and optionally specifies a *spine*. A spine of a region is a collection of curves which capture the general shape of the region. It approximates the medial axis of the region and it can be used by the system for mapping the source to the target (see also Figure 8).

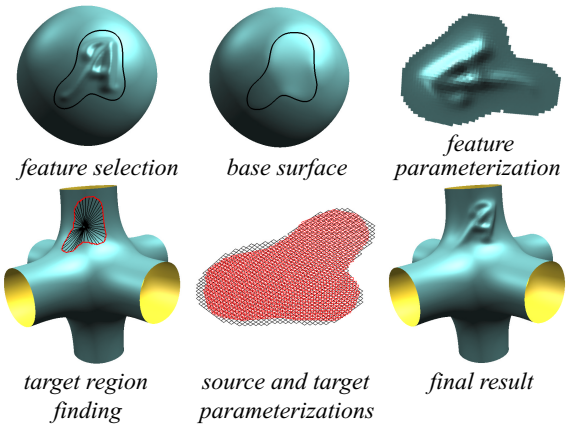


Figure 4: Main steps of the pasting algorithm, row-wise from top left: (a) selected feature on the source surface, (b) base source surface, (c) source parameterization onto the plane, (d) target region finding by geodesic walking, (e) source (black) and target (red) parameterizations superimposed in the plane, (f) source feature pasted onto the target surface.

2. The details are separated from the base for the source surface. The user interactively selects a base surface from a continuous range interpolating between a zero level given by a membrane surface and the actual surface (Section 5).

3. The source region is parameterized over the plane (Section 6).

4. The boundary of the source region is parameterized by distance and direction from the spine and a covering by disks is computed.

5. The user positions at least one point of the spine on the target, and specifies an orientation.

6. A target region for pasting is determined on the surface using geodesic disks (Section 7).

7. The target area is mapped to a common plane with the source and the source is resampled over the target sampling pattern (Section 8).

8. The resulting surface is computed by blending the target base surface, the source surface resampled details, and the target details. The user can specify different blending modes.

5 Separating Base Surface from Detail

An important step in the pasting process is the definition of which features of the source surface region constitute details that the user wants to paste over the target surface, as opposed to the larger-scale surface shape that should be ignored. Separating the base surface from the details depends on the semantics of the operation and has to be user-guided. For example, one may want to extract only the texture-like geometry of the skin on the nose of a head model, or to paste the entire nose onto a different model. Different choices for base-detail separation result in different pasting effects (Figure 6).

Our approach is to provide a continuum of base surface choices guided by a single parameter which can be thought of as the flatness of the base surface. A natural way to obtain a smooth base surface given our multiresolution data representation is to remove or reduce the multiresolution details present in the multiresolution hierarchy on the finer levels. The degree to which this approach works depends on the way the coarser levels were obtained when the hierarchy was constructed. By comparing several approaches (Taubin's smoothing, quasi-interpolation, and fitting), we found that fitting works best for pasting.

Least-Squares Fitting. The fitting procedure minimizes a functional that measures how well the smooth surface fits the vertices of the original mesh subdivided to the finest level M. While fitting of subdivision surfaces is not new (e.g., [26]), there appears to be no detailed description of it in the literature and we present it here for completeness. The minimization problem for level m of the smooth surface hierarchy can be stated as:

$$\min_{p} \sum_{w \in V^M} ||p_w^M - [S^{M-m} p^m]_w||^2 \qquad (1)$$

where the minimum is computed over all possible choices of control points p^m for the smooth mesh, V^M is the set of vertices of the finest-level mesh, p^M are the corresponding control points, S^{M-m} is the subdivision matrix for $M - m$ subdivision steps, and $[]_w$ means that the resulting smooth surface is evaluated at parameter values corresponding to vertices w of the control mesh. The minimization problem is equivalent to finding solutions for the linear system $A^T A x = A^T b$, with $A = S^{M-m}$, $b = p^M$ and $x = p^m$, and can be solved by using the Conjugate Gradient method. To apply this method, the only operations needed aside from linear combinations of vectors and dot products, are matrix-vector multiplications for the S^{M-m} matrix and its transpose. As the matrix is obtained by iterative application of the subdivision matrix, there is no need to represent or store it explicitly: applying A corresponds to the application of $M - m$ subdivision steps. Applying A^T to a vector can be interpreted in similar terms. More specifically, as shown in Figure 5, the mask for each vertex v on level $m - 1$ contains all vertices on level m which are affected by v when subdivision is performed. If vertex v has coefficient α in the subdivision rule used to compute the control point for vertex w, then vertex w has coefficient α in the transpose averaging rule for v.

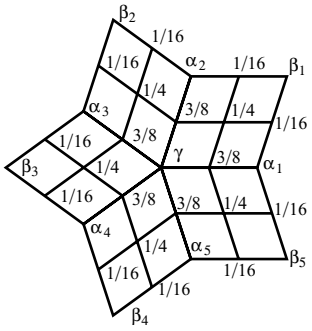


Figure 5: The mask for local averaging used for transpose subdivision. Coefficients α_i and β_i are the Catmull-Clark vertex rule coefficients for corresponding vertices. $\gamma = 1 - n\alpha - n\beta$, where n is the valence of the central vertex and α and β are the vertex rule coefficients.

Once the sequence of levels is computed, a continuum of base surfaces can be obtained by interpolation as shown in Figure 6. The user can select one interactively by moving a slider.

An alternative approach to fitting is to use the quasi-interpolation approach of Litke et al.[26]. This approach is somewhat faster, but results in larger errors. We use the more accurate fitting approach as it does not constitute a bottleneck in our system.

Boundary constraints. The technique previously described explains how to produce smoother approximations of the surface

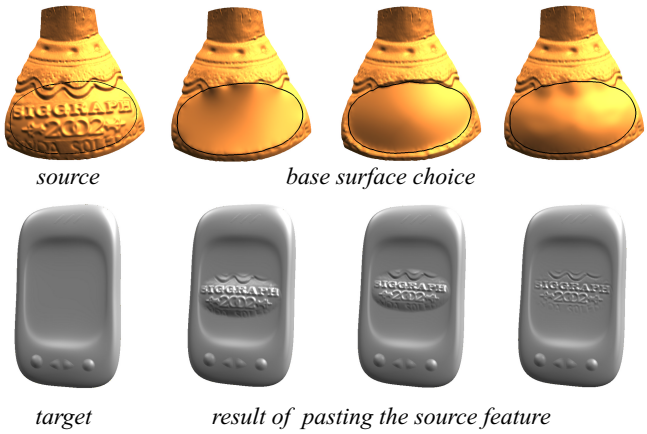


Figure 6: Depending on the choice of base surface, different scales of shape details are transferred to the target.

globally. This approach is quite fast as the base surfaces on different levels can be precomputed and only interpolation is required after that. However, when we separate the feature from the surface, we need a base surface only near the feature. Even more importantly, in most cases the details should gradually decay in magnitude as we approach the boundary of the feature. To adapt the global base surfaces to our needs, we use the following simple blending approach: the local base surface is computed as a blend of the source surface and a global base surface. The region in the interior of the feature is assigned alpha values 1 and all vertices outside the region are given values 0. Next, relaxation is applied for values in the interior while keeping the values outside constant. The amount of relaxation is user-controllable and allows to change the way features blend into the target. The resulting alpha values are used to interpolate between the global base and the source surface (Figure 11).

Minimal base surface. Base surfaces defined by fitting and blending cannot be flatter in the area of the feature than the base surface obtained by fitting on the coarsest level. This might be not appropriate for some applications where it is necessary to retain more of the feature shape (e.g., Figure 14). In such cases, it is best to define the base surface as a smooth, relatively flat surface that fills the hole remaining after the feature is cut off. To obtain such a surface, we optimize the membrane energy of the surface inside the feature curve while constraining its boundary to remain fixed. We use a multigrid-type approach [19], which is a natural choice in the context of our multiresolution representation. Similarly, the transition between the feature and the base is handled by assigning alpha values. This allows us to extend the range of possible base surfaces beyond the coarsest-level fitted surface. As a result the user has a choice of base surfaces varying from the minimal surface spanning the outline of the feature, all the way to the original surface.

6 Parameterization

Once we have separated the details from the base surface, we need to find a map from the source base surface to the target, to be able to transfer the details. As it was discussed in Section 3.3, we construct the map in two steps. First, we map the source surface to the plane. Second, we determine the region on the target surface where the feature will be pasted and we parameterize it onto the same plane.

Parameterization is needed both for the source and target base surfaces. The type of surface patches that we need to parameterize is relatively uncommon: while the surface is likely to be quite smooth, the shape of the patch can be relatively complex. The parameterization we construct should satisfy the following requirements:

- The parameterization region should not be chosen a priori. The need for this can be seen from the following simple example: the outline of a feature selected on the plane base surface can be arbitrarily complex, however the parameterization should not be different from the surface itself. Any algorithm that requires a fixed domain is not likely to perform well in this situation.
- The parameterization should be guaranteed to be one-to-one. As we need to resample, for each vertex on the target we need to identify a unique position on the source. This means that at least the map from the source to the plane has to be one-to-one.
- The parameterization should minimize a reasonable measure of distortion. Ideally, for developable surfaces it should be an isometry up to a scale factor. The algorithm that we use does not explicitly minimize a measure, but it appears to produce results with close to minimal shape distortion, as discussed below. It tends to produce better results than all other algorithms that we have tried in situations relevant for us.

Most of the existing parameterization algorithms do not determine the domain automatically: it is either determined using a heuristic approach or it has to be prescribed by the user. The parameterization described in [36] allows for free boundary evolution, but it requires a vector field defined over the surface and it does not provide a one-to-one guarantee. Until recently, algorithms that guaranteed a one-to-one parameterization required convex domains, like the many variations of Floater's algorithm ([10]). We use the algorithm of Sheffer and Sturler [39] which best meets our requirements.

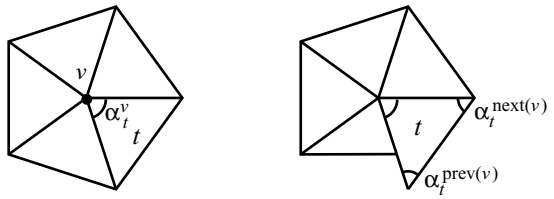
Angle-based flattening. For a given mesh a parameterization is defined by specifying the positions (parametric coordinates) of all vertices of the mesh in the plane. Without the loss of generality, we can assume a triangular mesh (we use quad subdivision surfaces, but each quad can be easily split into two triangles). The idea of angle-based flattening is to compute the parametric coordinates of the vertices indirectly: first, all angles are computed using an optimization procedure, then the planar mesh is reconstructed by fixing the length of one of the edges. The reason for computing the angles rather than vertex positions directly is that the one-to-one condition can be easily enforced and aspect ratios can be controlled explicitly. The disadvantage is that the reconstruction procedure is relatively unstable, as positions of vertices depend sequentially on each other. However, we found that for the relatively small numbers of triangles that we use (at most thousands), this is never a problem.

Next we describe the formulation of the optimization problem for angles mostly following [39]. Let t denote a triangle of the mesh, T the set of all triangles; let v be a vertex and V the set of all vertices in mesh. If v is a vertex of t then the angle α_t^v is the corresponding angle in the triangle t. Let $N(v)$ be the set of all triangles sharing a vertex v. The target value for angle α_t^v is defined as follows: $\varphi_t^v = 2\pi\alpha_s^v / \sum_{s\in N(v)} \alpha_s^v$, i.e., ideally all angles at a vertex should be rescaled by the same amount, so that their sum is equal to 2π. The resulting functional is

$$F(\alpha) = \sum_{t,v\in t} w_t^v (\alpha_t^v - \varphi_t^v)^2,$$

where the weights are chosen to be $1/(\varphi_t^v)^2$.

Minimization of this functional needs to be constrained for the results to correspond to a valid planar triangulation. The necessary constraints are as follows: (a) the angles should stay above some minimal value ϵ; (b) the sum of all angles at a vertex should be 2π; (c) the sum of all angles of each triangle should be π; (d) each 1-neighborhood of a vertex should be consistent. This means that



notation *consistency constraint violation*

Figure 7: The image on the right shows the situation prevented by the constraint $g_4(t)$ in the parameterization algorithm.

if we reconstruct a neighborhood triangle-by-triangle going around the vertex, the last edge of the last triangle should coincide with the first edge of the first triangle as shown in Figure 7.

The mathematical expressions for these four constraints are:

$$g_1(v,t) = \alpha_t^v - \epsilon \geq 0; \quad g_2(v) = \sum_{t \in N(v)} \alpha_t^v - 2\pi = 0;$$

$$g_3(t) = \sum_{v \in t} \alpha_t^v - \pi = 0$$

$$g_4(v) = \quad \Pi_{t \in N(v)} \frac{\sin \alpha_t^{\text{prev}(t)}}{\sin \alpha_t^{\text{next}(v)}} - 1 = 0.$$

The inequality constraints are best enforced in an iterative procedure for minimizing the functional by rejecting values that violate the constraints. The equality constraints are included into the functional by means of Lagrange multipliers: $L(\alpha) = F(\alpha) + \sum_v \lambda(v) g_2(v) + \sum_t \mu(t) g_3(v) + \sum_v g_4(t)$.

This is a nonlinear optimization problem which is solved using Newton's method: $x_{n+1} = x_n - H(L)^{-1} \nabla L$, where ∇L and $H(L)$ are the gradient and the hessian of L, and x is the vector of all angles and Lagrange multipliers. At each step, we need to solve a linear system to invert $H(L)$. In [39] a direct solver is used. We observe that the system does not change much from one iteration of the Newton method to the next. Furthermore, for smooth surfaces it is likely that the initial guess for the angles is quite close to the solution (e.g. for the developable surface it is already a solution). This indicates that iterative solvers are likely to perform quite well. Although the system is symmetric, it is not positive definite, and the Conjugate Gradient method cannot be used. However, the Conjugate Residuals method applies. For fast convergence rates, preconditioning is required, i.e., an approximate sparse inverse of the matrix needs to be computed. To avoid an expensive preconditioner computation, we add a small negative constant equal to the inverse of the number of triangles on the diagonal which makes it possible to avoid pivoting when calculating an incomplete factorization (ILU) preconditioner. Our experience was that a small number of iterations of the solver were sufficient to obtain a reasonable parameterization.

It should be noted that, while the constraints guarantee that the resulting mesh is one-to-one locally (no flipped triangles), the boundary of the image may self-intersect and globally the map is still not one-to-one. A technique for eliminating such self intersections is described in [39].

7 Determining a Target Region

Before pasting can be performed, an area on the target surface corresponding to the feature has to be identified and parameterized. It is a chicken-and-egg problem: to determine the region covered by the pasted feature, we need to map it to the target; however, mapping the feature to the target requires parameterizing the corresponding part of the target surface over the plane. As parameterizing the whole target is generally not an option, we use the following approach: we observe that initially we need to identify only an approximate boundary region where the feature will fit, rather than to establish a one-to-one mapping of the interior. Once the region is identified, it can be parameterized over the plane and a mapping is computed as the composition of the two parameterizations.

The algorithm that we use for identifying the region proceeds in several steps: first, we represent the boundary of the source region in a generalized radial form, constructing line segments (planar geodesics) connecting the spine to the boundary. Then we map the one-dimensional spine to the target, and use geodesics on the target to map the boundary points to the target. Finally, we connect the points on the target and fill in the interior region (Figure 8). The computation of geodesics passing through a point is a central tool in the algorithm and is discussed in greater detail.

Parameterizing the source boundary. The user has the option to draw a curve on the surface, possibly with several branches, which serves as the spine of the feature. It is our main intention to help the system map the feature to the target surface with the least distortion. If the user does not define a spine, a single point (the centroid of the boundary of the feature) is automatically selected to serve as the spine.

The following algorithm is used to parameterize the source boundary. First, the spine is mapped to a curve in the plane by the parameterization. Let $c_0, \ldots c_{m-1}$ be equispaced points on the spine in the parametric domain. The number of points can be adjusted to trade speed for quality.

For each vertex w_j on the boundary of the source parameterization find the closest point c_i. Let n_i be the number of points closest to the point c_i, d_j be the distance from c_i to w_j and γ_j be the angle between the direction from c_i to w_i and the spine. If the spine consists of a single point, an arbitrary fixed direction is used as the direction of the spine.

The boundary of the source region can be characterized by the set of triples (c_i, d_j, γ_j), where $i = 0 \ldots m-1$, and $j = 0 \ldots n_i - 1$. This collection of triples can be thought of as a discrete parameterization of the boundary with respect to the spine generalizing the radial parameterization. In the case of a single-point spine, this is just the radial parameterization.

Mapping the spine to the target. Mapping the spine to the target is straightforward: the user specifies an initial position and orientation for a point on the spine. The other points on the spine are obtained sequentially by walking as follows. Suppose the positions $T(c_0) \ldots T(c_i)$ are known. If the angle between the intervals (c_{i-1}, c_i) and (c_i, c_{i+1}) is β_i, then the next point on the spine is obtained by walking on the target surface at an angle β_i to the previous segment for a distance equal to $|c_i, c_{i+1}|$.

Finding the target region. Once the positions of all points $T(c_0) \ldots T(c_{m-1})$ are found on the target, we find the positions of each boundary point $T(w_j)$ using the corresponding triple (c_i, d_j, γ_j). Specifically, we walk starting from c_i along a geodesic direction forming the angle γ_j on the target for a distance d_j to obtain $T(w_j)$.

Once all the points on the boundary are found, they need to be connected. We do this by using a plane which passes through the two points $T(w_j)$ and $T(w_{j+1})$ and the normal at one of the points. If both normals happen to be aligned with the direction between the points, an additional boundary point is inserted midway between them and a corresponding radial representation is generated for it by adding an extra geodesic path.

We traverse the triangles along the intersection of the plane with the surface starting from $T(w_j)$ in the direction of $T(w_{j+1})$. There are three possible outcomes: either we reach $T(w_{j+1})$ (both points are on the same continuous segment of the plane surface intersection), we return to $T(w_j)$, or we reach a boundary. In the last two cases, we add a new point on the boundary of the source region and we repeat the procedure for each pair of points.

Once all sequential points $T(w_j)$ on the target surface are connected, we use a fill algorithm to mark the complete region.

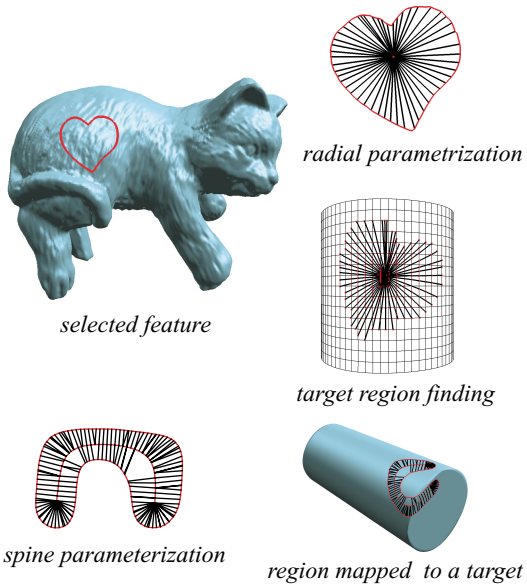


Figure 8: Finding the target region.

It is important to note that the algorithm may produce an area which is not topologically equivalent to a disk. For example, any large enough region mapped to a sphere can cover the entire sphere. The algorithm described above should be followed by a test checking the topology of the resulting area. This can be done by computing the genus assuming that there is a face attached to the boundary loop of the region. The genus computed in this way should be zero, and the region should have exactly one boundary loop. If the test fails, pasting at this scale is not possible. The user should decrease the scale for the pasted feature or place it at a different location.

Target parameterization. Once the target is determined, it is mapped to the plane. Generally, we use the same relatively expensive angle-based flattening algorithm for target parameterization each time a pasting operation is performed. This allows us to achieve maximum flexibility in feature placement and lowest distortion. While this approach still permits interactive manipulation rates, the frame rate is much better if a larger area of the target can be parameterized and the feature is moved inside this area. In this case, the most expensive part, i.e., target area finding and reparameterization is completely excluded, and only resampling has to be done at most steps.

Geodesic walking. One of the key ingredients of the algorithm for determining the target region is the algorithm for computing a geodesic emanating from a given point in a specified direction. While a number of algorithms for this or similar problems have been proposed [35, 16, 24], our application has specific requirements.

- We need the algorithm to be fast, as the target region has to be found at interactive rates. This makes it difficult to use methods based on front propagation.
- Even more importantly, we need a *continuity property*. Note that termination of the algorithm for finding the target region depends on our ability to make the distance between points $T(w_j)$ on the target arbitrarily small by increasing the density of the points w_j on the source boundary. Such continuity means that as we decrease the angle between two outgoing geodesics for a point, the distance between their endpoints can be made arbitrarily small. It is known however that straightest geodesics on meshes may violate this condition ("the saddle point problem").
- Accuracy of the result is of secondary importance, as the mapping process is approximate. Also "the swallow tail problem", i.e., the fact that geodesics may intersect near an elliptic point, is not relevant for us as we only determine the the boundary of the region and we do not construct a one-to-one map.

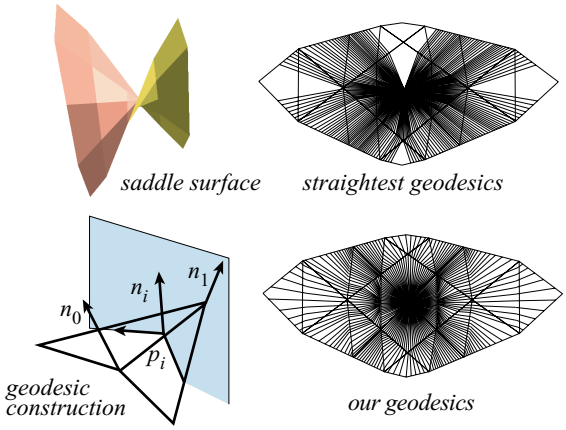


Figure 9: Comparison of straightest geodesics and our geodesics. Note the empty regions for the straightest geodesics: no matter how densely the directions are sampled, no geodesic passes through a part of the region.

Our procedure is based on the fact that the geodesic $g(t)$ is always a locally normal curve, i.e., its second derivative $g''(t)$ is pointing along the normal to the surface. By interpolating the normals, we approximate a smooth surface with continuously changing normal. The elementary step remains going from triangle to triangle, but the angles are computed differently.

Suppose we start at a point p_i at the edge e_0 of a triangle T_i, and v_0 and v_1 are the vertices of the edge e_0. Let n_0 and n_1 be the normals at the vertices v_0 and v_1. We compute the normal n_i at the point p_i linearly interpolating between normals n_0 and n_1 and renormalizing. If n_0 and n_1 point in opposite directions, arbitrary choice is made. Suppose there is an initial direction vector t_i defined at p_i. The procedure described next defines the direction vector at a sequence of points p_j of the discrete geodesic to be perpendicular to the interpolated normal vector at the point (if the initial one is not, we project it to the plane perpendicular to the normal). To obtain the point p_{i+1} and the new direction t_{i+1} at that point we perform the following steps:

1. Intersect the plane spanned by n_i and t_i with T_i to get a direction $P(t_i)$. If the plane coincides with the plane of T_i, t_i itself is used.

2. Intersect the line along $P(t_i)$ in the triangle T_i with its edges to get the point p_{i+1}. Suppose the intersected edge is e_1 with endpoints v_1 and v_2. The next triangle T_{i+1} is the triangle across the edge e_1.

3. Compute the normal n_{i+1} at p_{i+1} by linearly interpolating bewtween n_1 and n_2 at vertices v_1 and v_2. Project the direction $P(t_i)$ onto the plane perpendicular to n_{i+1} to obtain t_{i+1}. If $P(t_i)$ is parallel to n_{i+1}, we use the average of the projections obtained for two small perturbations of position of the point p_{i+1}.

It can be verified that this procedure satisfies the continuity requirement if the mesh approximating the surface is smooth enough, i.e., the projection of the ring of triangles around any vertex onto the plane perpendicular to the normal is one-to-one.

8 Mapping and Resampling

Once the mappings from the source and target to the plane are established, their planar images are aligned using the point and orientation correspondences specified by the user when the target area was chosen. The final step in the pasting algorithm is resampling and combining the details from the source with the details and base surface of the target.

For every vertex v of the parameterization of the target which is inside the parameterization domain of the source, we find the corresponding quad of the source parameterization. Next, u, v coordinates are computed in this quad and the source is evaluated. Evaluation can be done in two ways: for fast resampling, the values of the source at the vertices of the quad are interpolated. For higher quality, subdivision surface evaluation [40] should be used. This is similar to using bilinear filters for fast image editing and bi-cubic filtering for a higher-quality final result.

Adaptive refinement and sampling. The further away the geometry of the feature is from a displacement map, the less suitable pasting for surface operations is. However, in some cases it is desirable to use the pasting paradigm to place objects which cannot be reparameterized over the plane without considerable distortion (Figure 14 left). In other cases, the resolution of the source surface is substantially higher than the resolution of the target. In these cases, uniform sampling of the target is not adequate and a form of adaptivity is needed. Hybrid meshes [13] offer the maximal degree of flexibility, as it is possible to perform irregular refinement in some spots and align mesh edges exactly with pasted feature edges. We use a more conventional approach where only regular refinement of individual faces is allowed. However, rather than quadrisecting individual faces recursively according to a criterion, we estimate the local density of the source samples over a target face. We use this estimate to directly compute the subdivision level required for a given face and we refine faces to that level uniformly.

9 Results

A number of models created using our system are shown in Figures 12 to 14. Figure 12 shows how details from a scanned object are pasted on a simple vase model. In this case, the target object itself serves as the base surface. Similarly, Figure 13 demonstrates how details from a scanned model can be combined with a different computer-generated model. Figure 14 (right) demonstrates how a medium scale detail can be pasted on a surface while preserving small-scale surface details. Figure 14 (left) shows examples of feature manipulation on the surface.

In all cases the operations were performed interactively, but the frame rate varied greatly depending on the complexity of the feature, the complexity of the target region, and the sampling density in the target region. If the target is a simple smooth object, a large area can be parameterized at once without significant distortion and no dynamic parameterization is required. In such cases, sufficiently complex models permit manipulation at high frame rates. However, if no large region can be parameterized without distortion, the frame rate varies in the range 5-0.5 frames per second.

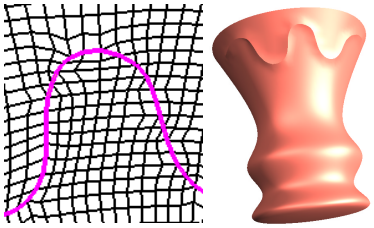


Figure 10: Adding a sharp crease to a multiresolution surface without changing the connectivity.

Limitations of the approach. The principal limitations of our approach include:

- The algorithm fails to produce a valid surface when the identified target region is not homeomorphic to a disk. This may occur, for example, if it completely covers a handle.
- The approach is useful for transferring features from one surface to another when the curvature of the chosen target base surface does not deviate radically from the curvature of the source base surface at corresponding points. While the algorithm will produce a valid surface for any situation when the identified target region has disc topology, when the target and source base surfaces are radically different the resulting surface may exhibit distortion of features and self-intersections.
- The resulting surfaces may exhibit geometric aliasing near sharp features as the sampling pattern of the target is used to resample the source. Possible straightforward solutions include adaptive refinement near sharp features which does not eliminate the problem but reduces the scale of artifacts and smoothing which eliminates the artifacts at the expense of detail. A more promising approach is mentioned in Section 10.

Except for the first one, all of the above limitations are "soft" in the sense that the algorithms we have described still produce a formally valid result.

10 Conclusion and Future Work

We have described an approach to surface editing that can be extended in many ways. One can imagine a variety of blending modes, combinations of pasting and texture generation, as well as other enhancements. One of the important advantages of the approach is that the structure of the target mesh is not changed by pasting (except for possible adaptive refinement). This means that the complexity of the object is not likely to increase quickly each time a feature is added as in the case of boolean operations. This is also a disadvantage, as pasting features with complex shapes may result in strong mesh distortion.

While applicable to a broad range of surfaces, pasting is primarily intended for displacement-map-like features. In its current implementation, the further away a feature is from a displacement map, the more likely self-intersections are to appear especially when a feature is pasted on a highly curved surface. We believe that the applicability of the approach can be extended if hierarchal pasting is used, i.e., the feature is decomposed into details and each level is pasted onto the previous. In this case, more complex features can be pasted more robustly.

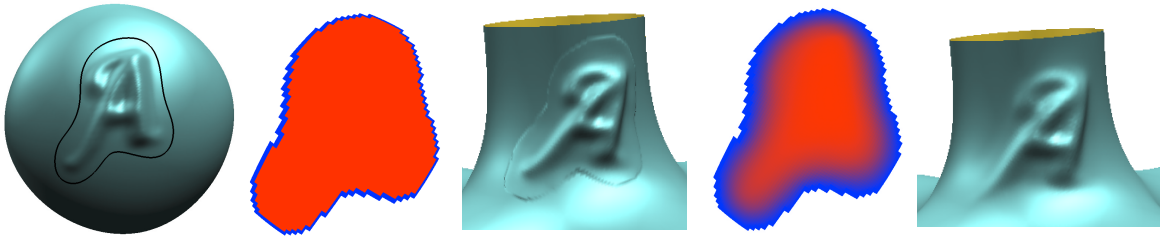


Figure 11: Blend region computation for eliminating boundary artifacts.

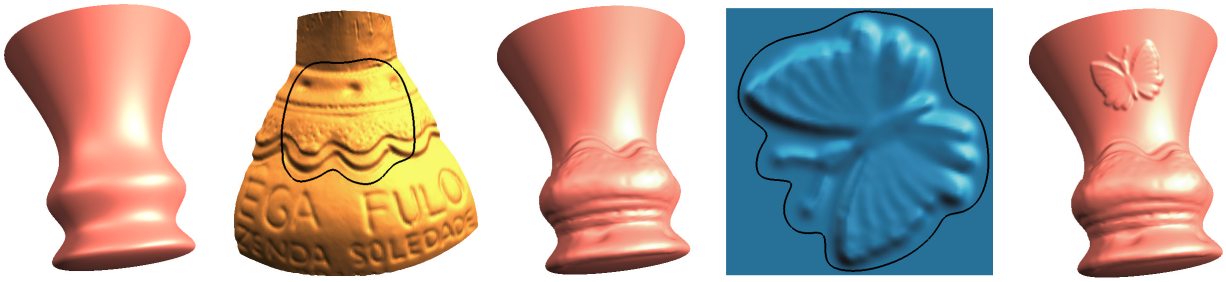


Figure 12: Combining a feature obtained by scanning a wine bottle with a displacement map created from a photograph onto a simple vase model.

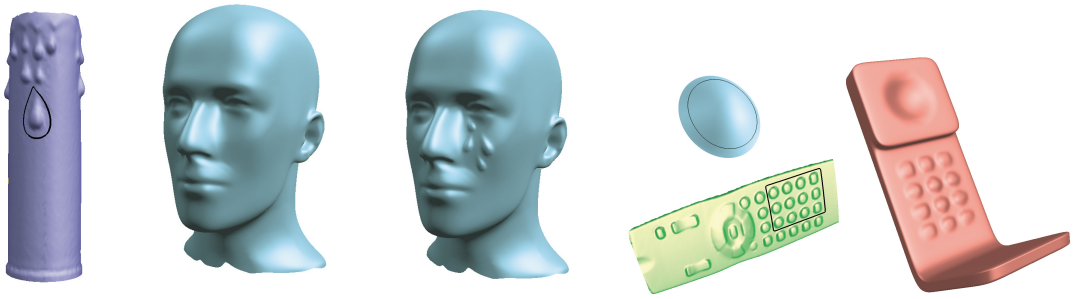


Figure 13: Left: details from a scanned candle pasted on the mannequin head. Right: features of different scales from other models added to a phone model.

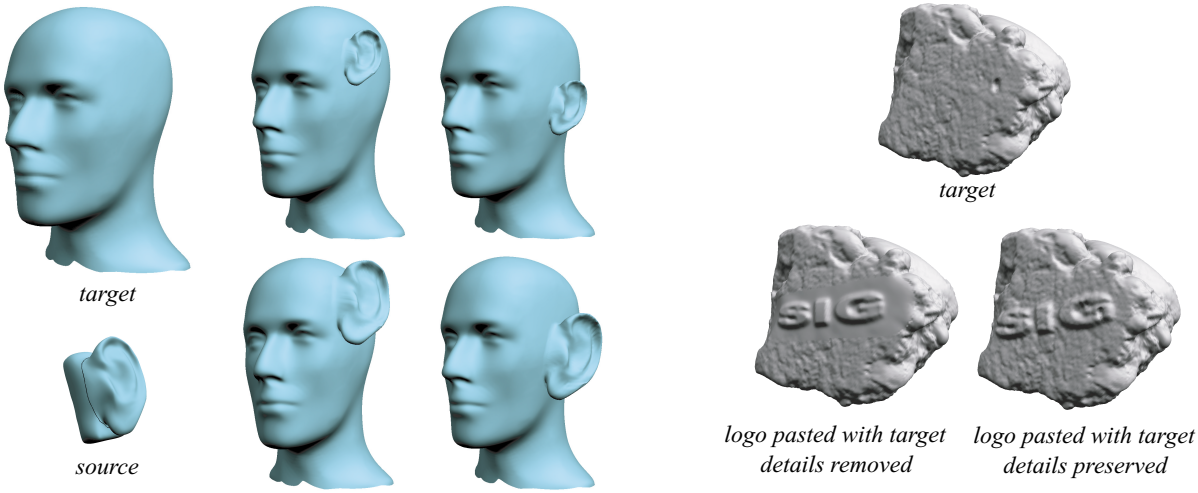


Figure 14: Left: pasting a complex feature (an ear) onto the mannequin head. Right: blending of source and target details. The object is a scanned rock.

Many CAD models have sharp creases. While a multiresolution surface can approximate sharp creases arbitrarily well, the approximate creases are never perfectly sharp and often exhibit aliasing. Furthermore, using details on all levels to introduce a simple corner is wasteful. The representation can be extended [5] to introduce such features by tagging some of the edges but without changing connectivity. Note that the parameterization in this approach (Figure 10) must conform to the sharp feature. An important future enhancement of our system is the ability to paste sharp features.

Acknowledgements. The authors thank the staff and students of NYU Media Research Lab for their help. Special thanks go to Xin Zhang for his help with writing and debugging parts of the code. This work was partially supported by funds from the NYU Center for Advanced Technology, IBM Faculty Partnership award, Sloan Foundation Fellowship, NSF award ACI-9978147, CCR-9900528, CCR-0093390, and NYU Dean's fellowship.

References

[1] www.paraform.com.

[2] www.geomagic.com.

[3] C. Barghiel, R. Bartels, and D. Forsey. Pasting spline surfaces. In *Mathematical Methods for Curves and Surfaces: Ulvik, Norway*, pages 31–40. Vanderbilt University Press, 1994. Available at ftp://cgl.uwaterloo.ca/pub/users/rhbartel/Paste.ps.gz.

[4] H. Biermann, D. Kristjansson, and D. Zorin. Approximate boolean operations on free-form solids. In *Proceedings of SIGGRAPH 01*, pages 185–194, August 2001.

[5] H. Biermann, I. Martin, D. Zorin, and F. Bernardini. Sharp features on multiresolution subdivision surfaces. In *Proceedings of Pacific Graphics 2001*, 2001.

[6] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. 10(6):350–355, 1978.

[7] L. K. Y. Chan, S. Mann, and R. Bartels. World space surface pasting. In W. Davis, M. Mantei, and V. Klassen, editors, *Proceedings of Graphics Interface*, pages 146–154, May 1997.

[8] B. Conrad and S. Mann. Better pasting via quasi-interpolation. In P.-J. Laurent, P. Sablonnière, and L. L. Schumaker, editors, *Curve and Surface Design: Saint-Malo, 1999*, pages 27–36, Nashville, TN, 2000. Vanderbilt University Press.

[9] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. *Proceedings of SIGGRAPH 95*, pages 173–182, August 1995.

[10] M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997.

[11] L. Freitag, M. Jones, and P. Plassmann. A parallel algorithm for mesh smoothing. *SIAM J. Sci. Comput.*, 20(6):2023–2040 (electronic), 1999.

[12] X. Gu, S. Gortler, and H. Hoppe. Geometry images. In *Proceedings of SIGGRAPH 02*, July 2002.

[13] I. Guskov, A. Khodakovsky, and P. Schröoder. Hybrid meshes. submitted, 2001.

[14] I. Guskov, W. Sweldens, and P. Schröder. Multiresolution signal processing for meshes. In *Proceedings of SIGGRAPH 99*, pages 325–334, August 1999.

[15] I. Guskov, K. Vidimce, W. Sweldens, and P. Schrder. Normal meshes. In *Proceedings of SIGGRAPH 00*, pages 95–102, July 2000.

[16] R. Kimmel and J. A. Sethian. Computing geodesic paths on manifolds. *Proc. Natl. Acad. Sci. USA*, 95(15):8431–8435 (electronic), 1998.

[17] L. Kobbelt. A variational approach to subdivision. *Comput. Aided Geom. Design*, 13(8):743–761, 1996.

[18] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multiresolution modeling on arbitrary meshes. In *Proceedings of SIGGRAPH 98*, pages 105–114, July 1998.

[19] L. P. Kobbelt. Discrete fairing and variational subdivision for freeform surface design. *The Visual Computer*, 16(3-4):142–150, 2000.

[20] V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In *Proceedings of SIGGRAPH 96*, pages 313–324, August 1996.

[21] S. Kuriyama and T. Kaneko. Discrete parameterization for deforming arbitrary meshes. In *Proceedings of Graphics Interface '99*, pages 132–139, June 1999.

[22] A. Lee, H. Moreton, and H. Hoppe. Displaced subdivision surfaces. In *Proceedings of SIGGRAPH 00*, pages 85–94, July 2000.

[23] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. Maps: Multiresolution adaptive parameterization of surfaces. In *Proceedings of SIGGRAPH 98*, pages 95–104, July 1998.

[24] H. Lee, L. Kim, M. Meyer, and M. Desbrun. Meshes on fire. In *EG Workshop on Computer Animation and Simulation*, 2001.

[25] B. Lévy and J.-L. Mallet. Non-distorted texture mapping for sheared triangulated meshes. In M. Cohen, editor, *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 343–352. Addison Wesley, July 1998.

[26] N. Litke, A. Levin, and P. Schröder. Fitting subdivision surfaces. In *Proceedings of IEEE Visualization 2001*, pages 319–324, October 2001.

[27] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, Department of Mathematics, 1987.

[28] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *Transactions on Graphics*, 16(1):34–73, January 1997.

[29] M. Ma. The direct manipulation of pasted surfaces. Master's thesis, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1, 2000. Available on WWW as ftp://cs-archive.uwaterloo.ca/cs-archive/CS-2000-15/.

[30] J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping. In *Proceedings of SIGGRAPH 93*, pages 27–34, August 1993.

[31] B. Oberknapp and K. Polthier. An algorithm for discrete constant mean curvature surfaces. In *Visualization and mathematics (Berlin-Dahlem, 1995)*, pages 141–161. Springer, Berlin, 1997.

[32] H. Køhling Pedersen. Decorating implicit surfaces. In *Proceedings of SIGGRAPH 95*, pages 291–300, August 1995.

[33] H. Køhling Pedersen. A framework for interactive texturing operations on curved surfaces. In *Proceedings of SIGGRAPH 96*, pages 295–302, August 1996.

[34] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experiment. Math.*, 2(1):15–36, 1993.

[35] K. Polthier and M. Schmies. Straightest geodesics on polyhedral surfaces. In H.C. Hege and K. Polthier, editors, *Mathematical Visualization*. Springer Verlag, 1998.

[36] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. In *Proceedings of SIGGRAPH 00*, pages 465–470, July 2000.

[37] E. Praun, W. Sweldens, and P. Schröder. Consistent mesh parameterizations. In *Proceedings of ACM SIGGRAPH 01*, pages 179–184, August 2001.

[38] K. Pulli and M. Lounsbery. Hierarchical editing and rendering of subdivision surfaces. Technical Report UW-CSE-97-04-07, Dept. of CS&E, University of Washington, Seattle, WA, 1997.

[39] A. Sheffer and E. de Sturler. Surface parameterization for meshing by triangulation flattening. In *Proc. 9th International Meshing Roundtable*, pages 161–172, 2000.

[40] J. Stam. Exact evaluation of catmull-clark subdivision surfaces at arbitrary parameter values. In *Proceedings of SIGGRAPH 98*, pages 395–404, July 1998.

[41] H. Suzuki, Y. Sakurai, T. Kanai, and F. Kimura. Interactive mesh dragging with an adaptive remeshing technique. *The Visual Computer*, 16(3-4):159–176, 2000.

[42] C. L. F. Tsang. Animated surface pasting. Master's thesis, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1, 1998. Available at ftp://cs-archive.uwaterloo.ca/cs-archive/CS-98-19/.

[43] F. W. Warner. *Foundations of differentiable manifolds and Lie groups*. Springer-Verlag, New York, 1983. Corrected reprint of the 1971 edition.

[44] D. Zorin, P. Schröder, and W. Sweldens. Interpolating subdivision for meshes with arbitrary topology. In *Proceedings of SIGGRAPH 96*, pages 189–192, August 1996.

[45] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. In *Proceedings of SIGGRAPH 97*, pages 259–268, August 1997.