Topology-Free Cut-and-Paste Editing over Meshes

Hongbo Fu, Chiew-Lan Tai, Hongxin Zhang
Department of Computer Science
Hong Kong University of Science and Technology

Abstract

Existing cut-and-paste editing methods over meshes are inapplicable to regions with non-zero genus. To overcome this drawback, we propose a novel method in this paper. Firstly, a base surface passing through the boundary vertices of the selected region is constructed using the boundary triangulation technique. Considering the connectivity between the neighboring vertices, a new detail encoding technique is then presented based on surface parameterization. Finally, the detail representation is transferred onto the target surface via the base surface. This strategy of creating a base surface as a detail carrier allows us to paste features of non-zero genus onto the target surface. By taking the physical relationship of adjacent vertices into account, our detail encoding method produces more natural and less distorted results. Therefore, our elegant method not only can eliminate the dependence on the topology of the selected feature, but also reduces the distortion effectively during pasting.

1. Introduction

Cutting and pasting are two most common operations in text editing and image processing applications. They enable existing resources to be utilized repeatedly. Recently, research on cut-and-paste editing over 3D meshes has been carried out. These operations allow users to transfer geometry features from a source surface to a target surface. It is a natural way to build composite objects by making use of existing patches from different surfaces in 3D. For instance, we can paste two wings onto a fish to get an interesting flying fish.

Previous cut-and-paste editing methods can be classified into two groups. One uses mesh fusion to blend the source surface and the target surface directly [13, 18]. The other first extracts a base surface as a medium between the source surface and the target surface, and then transfers the details to the target surface via the base surface [3, 4, 7, 8, 11, 17]. The former pays more attention to the smoothness at the joint of the source surface and the target surface. The latter focuses on the global deformation of the source surface according to the target surface. However, most of these existing techniques are inapplicable to regions with non-zero genus [3, 4, 7, 8, 13, 17].

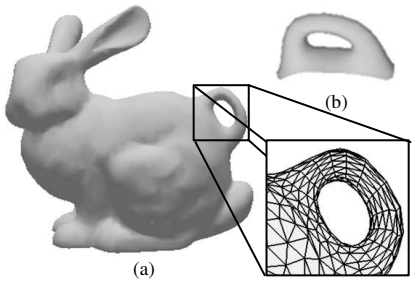

Figure 1. An output example of our topology-free pasting algorithm. (a) Stanford bunny with the pasted ring. (b) The original ring before pasting.

In this paper, we present an efficient topology-free cut-and-paste algorithm. Figure 1 shows an output of the algorithm. The algorithm is applicable to cut feature of non-zero genus, and it produces more natural pasting results than previous methods. Essentially, our cut-and-paste algorithm is a forward mapping from the source to the target: for each vertex of a source feature to be cut and pasted, we first find its corresponding base point on the source base surface, then we determine a corresponding position on the target base surface for the base point, finally the pasted position is computed with respect to that position on the target base surface. In our algorithm, the topology of the base surface is independent of the topology of the source feature; this guarantees that our cut-and-paste algorithm is topology-free.

Our main contributions are as follows:

- An efficient base surface extraction method based on boundary triangulation. Such extracted base surface facilitates the pasting of regions with non-zero genus or complicated features.

- A novel detail encoding technique based on surface parameterization. Our method can simulate the effect of adjacent vertices acting on each other when the mesh is distorted.

2. Related Work

The concept of surface pasting in a hierarchical fashion has been introduced by several researchers [3, 7, 8, 17]. These pasting techniques are only applicable to B-spline tensor product surfaces. Later, Biermann et al. choose semi-regular multiresolution subdivision surfaces as their surface pasting representation [4]. However, their method is essentially an inverse mapping and therefore cannot be applied to regions that are not homeomorphic to a disk (see Section 5 for details). Furukawa and Masuda [11] propose a cut-and-paste editing method based on constrained B-spline surface/volume fitting. It is applicable to regions containing handles, however their method lacks real-time interaction and easily leads to unnatural pasted results. All of the above pasting algorithms use the concept of base and detail. Details are built over the source base and transferred from the source to the target.

Another branch of cut-and-paste editing work makes use of blending techniques, instead of base/detail. Kanai et al. present a scheme based on local 3D metamorphosis, which works well for meshes that are isomorphic to a disk [13]. In a level set framework, Museth et al. implement a surface pasting technique with a union boolean operation and a subsequent blend operator [18].

Surface parameterization maps a 3D mesh isomorphic to a disk to a planar mesh. It builds a bridge between 3D space and 2D space and has been applied to many areas, such as texture mapping and surface reconstruction. Surface parameterization is usually achieved by minimizing the difference between the original surface and its associated parameterization, such as angle-based parameterization [19] and intrinsic parameterization [9].

Local detail representation is often used in multiresolution surface applications. It encodes the difference between successive multiresolution levels. Usually, a simple normal displacement scheme is adopted because it can be computed very efficiently [15]. To avoid negative barycentric coordinates for the base points of the normal displacement vectors, Kobbelt et al. propose the Phong-type normal field [16]. Botsch et al. present a detail representation method based on displacement volumes [5], which can effectively avoid local self-intersections in the reconstructed surface. In our application, the local detail representation is with respect to a local coordinate frame computed by surface parameterization for the detail surface over the source base surface.

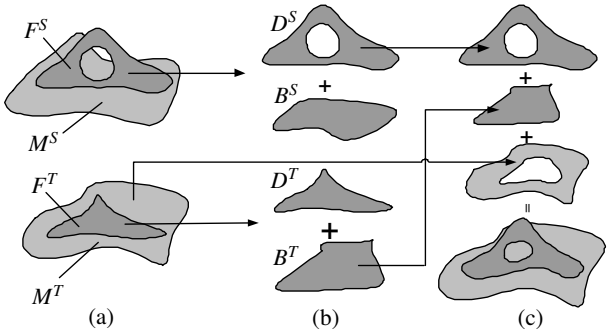

Figure 2. Essence of our cut-and-paste algorithm. (a) Source feature surface F^S and target feature surface F^T. (b) F^S is separated into the source detail surface D^S and the source base surface B^S; F^T is separated into the target detail surface D^T and the target base surface B^T. (c) D^S is pasted onto the target surface via B^T.

3. Essence of Cut-and-Paste Editing

In this section, we discuss how surface pasting is employed and what factors influence the pasted result.

Imagine that we are putting a soft object A onto another elastic object B and we want to combine A and B into one object seamlessly. One method to achieve this aim is to drag the boundary of A and make it cling to the surface of B. However, the features of A will be distorted. The force that causes the distortion actually comes from stretching the boundary of A. When a strain is applied at one point of an object, the force spreads out and distorts the neighboring parts around the point.

The above observation inspires us to investigate a cut-and-paste technique that produces natural results. We notice that only the contact surface between the source feature and the target object determines the distortion of the pasted feature. We call this contact surface the base surface.

Figure 2 illustrates this cut-and-paste approach. Given the source feature surface F^S selected from the source mesh M^S and the target feature surface F^T from the target mesh M^T, our aim is to paste F^S onto F^T (Figure 2 (a)). Like [4, 11], we separate both the source feature and the target feature into the detail surface D and the base surface B (Figure 2 (b)). Here, D simply denotes the feature surface F represented as encoded local detail representation over B. Note that the base surface must be homeomorphic to a disk. However, it is not necessary for the topology of D to be the same as that of B. Figure 2 (c) shows the final result: D^S is pasted onto the target mesh via B^T. The pasted detail on the target mesh can be either from D^S or as a blend of

D^S and D^T. We only implement the former option.

Pasted feature distortions are due to the difference between B^S and B^T. If B^S and B^T have the same shape, distortions will not occur; when they have different shapes, distortions are unavoidable. For example, if B^S has a convex shape and B^T has a concave shape, then the final result will be crushed together somewhere. Therefore, choosing the base surface is the most important step. Since a developable surface can be unfolded onto a plane without stretching, we choose an approximate developable surface as the base surface in our algorithm.

4. Algorithm Overview

Figure 3 illustrates the main steps of our cut-and-paste algorithm.

(a). A user specifies a region to be cut (called the source feature surface), which may have non-zero genus or areas with high curvature.

(b). A source base surface is constructed automatically for the cut source feature according to its boundary information (Section 5).

(c). Both the source feature and the source base surface are parameterized onto a common plane, and their parameterizations are enclosed within exactly the same region (Section 6.1).

(d). According to the correspondence established by the above two parameterizations, local frames are built over the base surface (Section 6.2). We call the feature surface that is defined with respect to the local frames the detail surface.

(e). The user specifies the position and orientation on the target surface for the pasting procedure. Then the target feature surface onto which the source feature surface will be pasted is determined automatically. Like the source feature surface, the target feature surface is separated into the detail surface and the base surface. By parameterizing and building a mapping between the source and the target bases, we first paste the source base onto the target base, and then transfer the detail surface onto the pasted source base according to the local frames (Section 7).

(f). The gap between the pasted source feature and the target surface is filled using the boundary triangulation technique. Smoothing is performed along the pasted boundary to obtain the final pasted version.

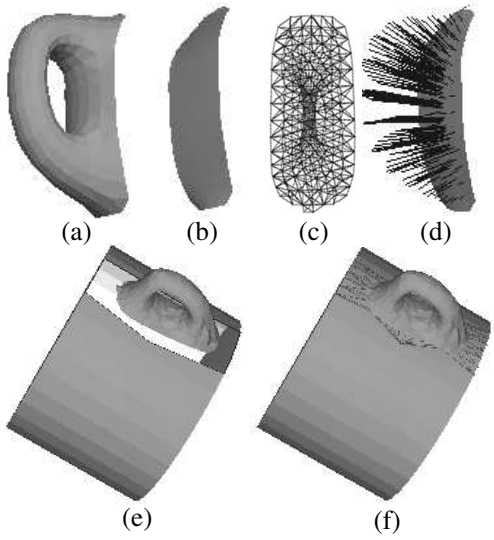

Figure 3. Overview of our topology-free cut-and-paste algorithm. (a) Source feature. (b) Source base surface. (c) Parameterizations of the source feature and the source base surface. (d) Local frames built over the base surface. (e) Source details are transferred onto the target surface. (f) Pasted result.

5. Base Surface Extraction

To specify the boundary of a feature surface to be cut, our current implementation supports the following interface. The user clicks on some faces of the source mesh one by one, demarcating a region of interest in counterclockwise. By connecting all the clicked points sequentially, we obtain a closed polygon curve and a set of faces that the polygon curve passes through. A flood fill algorithm is applied to get a complete region.

Next, we wish to derive a base surface from the selected region. Base surface extraction is the most important step of the whole cut-and-paste algorithm. On the one hand, it determines the geometry to be pasted [4] (for example, the geometric texture extracted from the selected feature surface), because the details to be pasted are defined as the difference between the feature surface and the base surface; on the other hand, it decides, to some extent, the type of surfaces that can be handled by the cut-and-paste algorithm. For example, in Biermann's method [4], the base surface is obtained by removing the multiresolution details from the mesh at the finer level. Since general subdivision rules do not change the topological structure, it is impossible to apply Biermann's method to features that are not homeomorphic to a disk.

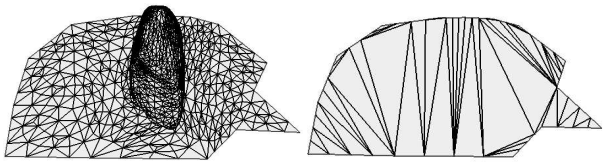

Figure 4. The optimal boundary triangulation (right) of the feature surface (left).

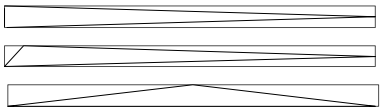

Figure 5. Degenerate triangles: needles (top and middle) and cap (bottom).

To resolve this problem, we construct a smooth surface passing through the boundary vertices as our desired base surface, regardless of the topological complexity of the feature structure. Since there exist infinite surfaces satisfying this basic requirement, we impose another constraint that the desirable surface is as planar as possible. Our base surface is generated on-the-fly, avoiding an exact mathematical expression. Section 5.1 describes an optimal boundary triangulation that approximates a developable surface. Section 5.2 gives a mesh optimization algorithm that removes degenerate triangles and adjusts the density of the approximate developable surface to obtain the final base surface.

5.1. Boundary Triangulation

Our base extraction problem becomes a geometric problem of constructing a smooth surface passing through a given closed space curve. One classical solution is using triangulation technique. However, the problem of whether a 3D polygon has a triangulation that is not self-intersecting and that defines a simply connected 2-manifold is NP-Complete [1]. Therefore we simply assume that the selected boundary specified by the user is triangulable. This assumption can easily be satisfied unless the user makes a weird selection.

We found that the optimal triangulation that minimizes the total edge length or triangle area [2] produces a good boundary triangulation for our application. The optimal triangulation approximates a developable surface, which can be unfolded onto a plane without any distortion. This characteristic reduces the unavoidable distortion generated by the surface pasting.

A formal description of the optimal triangulation problem is as follows [2]:

> Given a closed triangulable space curve C, find the triangulation of C that minimizes the objective function defined on all the triangles created by all possible triangulations.

In our case, the optimization can be stated mathematically as

$$\min_{\Gamma} \sum_{\triangle uvw \in \Gamma} Area(u, v, w) \qquad (1)$$

where Γ is a set of all the possible triangulations and $Area(u, v, w)$ is the area of $\triangle uvw$ with u, v, w denoting three different vertices of the given closed polygon $C = (v_0, v_1, ..., v_{n-1}, v_n = v_0)$.

The above optimal triangulation can be solved using dynamic programming (DP) technique [2, 14]. Figure 4 shows a feature surface and the corresponding optimal boundary triangulation representing the base surface. Note that the DP algorithm takes $O(n^3)$ running time and $O(n^2)$ space. Considering that the number of the boundary vertices is much less than that of all the mesh vertices, the boundary triangulation algorithm only constitutes a small portion of the total running time of the cut-and-paste algorithm.

5.2. Mesh Optimization

The resulting boundary triangulation may contain degenerate triangles, which have no well-defined normal vectors. The density of the base surface may also be very sparse. Therefore, we need to apply mesh optimization to remove the degenerate triangles and adjust the density of the base surface.

The aspect ratio of a triangle is defined as the ratio of the short edge to the long edge of the smallest rectangle containing this triangle. Degenerate triangles have an aspect ratio close to zero. There are two types of degenerate triangles [6].

Needle triangle whose shortest edge is much shorter than the longest one (Figure 5 top and middle).

Cap triangle with an angle close to 180^o, and is not a needle (Figure 5 bottom).

We propose a mesh optimization procedure consisting of the following four steps.

Edge splitting. In the detail encoding step (Section 6.2), the local frame for each vertex in the detail surface is built over a certain triangle in the base surface. The transformation of these base triangles causes the corresponding distortion of the detail surface. Thus, in order to minimize the distortion, a dense base mesh is desired. Since no new vertices are introduced in the boundary triangulation step, it is necessary to first increase the density of the base mesh. We perform

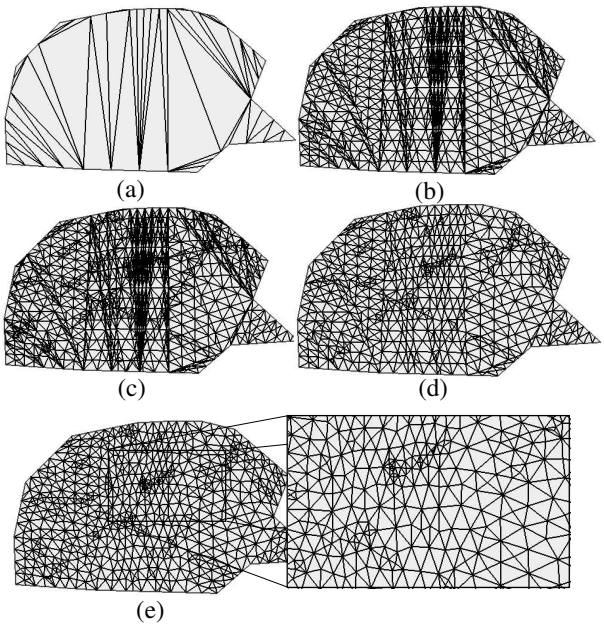

Figure 6. Steps of our mesh optimization algorithm. (a) Original mesh. (b) Edge splitting. (c) Mesh slicing. (d) Edge collapse. (e) Mesh smoothing.

an edge splitting operation to achieve this goal. A new vertex is inserted into the edge with the maximum length as a midpoint until all the edges satisfy the following inequality

$$length(e_i) < \varepsilon * l_{min}, \quad \text{for } 1 \leq i \leq n, \varepsilon > 0 \qquad (2)$$

where e_i is an edge in the current base surface, l_{min} is the minimum edge length of the original base surface, ε is a constant controlling the density of the base surface, and n is the total number of edges.

Edge splitting is likely to produce new caps or new needles, hence this operation is done as the first step of the mesh optimization algorithm (Figure 6 (b)).

Mesh slicing. Caps can be removed by mesh slicing [6]. In [6], before the mesh slicing is performed, a set of planes are manually or automatically constructed to intersect the mesh, and the intersection points and their associated faces are added to the original mesh to enhance the density of the mesh. In our algorithm, the desired density is attained by the edge splitting step prior to the mesh slicing step (Figure 6 (c)).

Edge collapse. This step eliminates all the needles. All the "bad" edges in the needles are collapsed until the aspect ratio of all the triangles is greater than a threshold specified by the user initially (Figure 6 (d)).

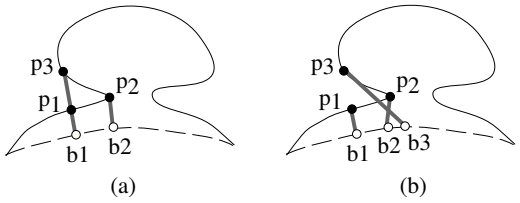

Figure 7. (a) Normal displacement. (b) Our detail encoding method. p_1, p_2 and p_3 are on the detail surface. b_1, b_2 and p_3 are the base points on the base surface (dashed curve).

Mesh smoothing. To stabilize the algorithm, we smooth the base surface as the last step. We use the umbrella operator [15], which successively moves each interior vertex to the barycenter of its neighboring vertices and needs only a few iterations to converge to a stable state (Figure 6 (e)). A drawback of the umbrella operator is that it causes shrinkage of the mesh. Alternative smoothing methods are Taubin's weighted Laplacian smoothing [20] or Desbrun's curvature flow operator [10].

6. Detail Encoding based on Surface Parameterization

After extracting the base surface, we need a method to transfer the source detail surface to the target surface. In general, this is solved by building local frames over the source base surface and reconstructing the local frames on the target base surface. Several detail encoding techniques have been proposed including normal displacement [15], Phong-type normal field [16] and displacement volumes [5]. All of these techniques compute the detail representation for each vertex of the detail surface, while ignoring the connectivity of the neighboring vertices.

We present a new detail encoding method based on surface parameterization, which can encode the detail of the source feature surface more physically and naturally, and consequently reduces the distortion.

Figure 7 illustrates the difference between our encoding method and the normal displacement [15]. We choose to illustrate the normal displacement method as an example of previous techniques, the other methods produce similar results. To compute a local coordinate for a vertex in the detail surface, the normal displacement method finds the nearest vertex on the base surface. For example, in Figure 7 (a), b_2 is the base point of p_2, and b_1 is the base point for both p_1 and p_3. If we wish to obtain a deformation in the neighborhood of p_1, we would deform the region containing b_1;

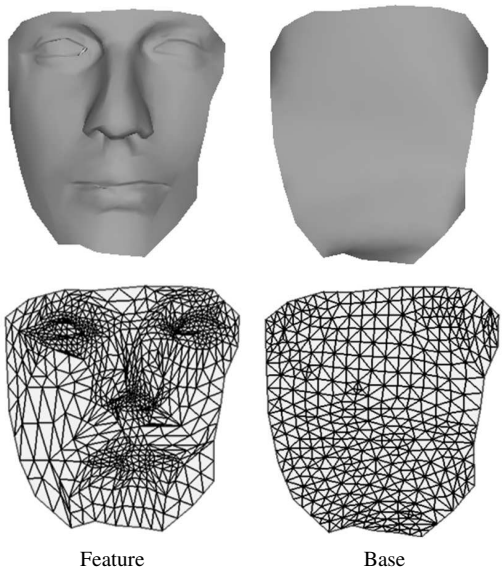

Figure 8. Parameterizations of the feature surface and its associated base surface.

however, this would cause the neighborhood of p_3 to be deformed too. This result is unnatural since the deformation would affect p_3 while bypassing p_2. Our detail encoding method (Section 6.2), which is based on surface parameterization, avoids this problem. Figure 7 (b) shows that the base points of p_1, p_2 and p_3 are b_1, b_2 and b_3 respectively. Although our encoding method is not physics-based, it simulates the effect of only neighboring vertices acting on each other.

6.1. Surface Parameterization

Surface parameterization not only builds a mapping from the source surface to the target surface, but also allows us to encode the detail representation.

By parameterizing both the source feature surface F and the source base surface B onto the same plane, we build a mapping between them for encoding the details. Since the boundary vertices of the feature surface should have zero detail vectors, we want to map them to the corresponding boundary vertices of the base surface. In other words, the feature surface parameterization P^F and the base surface parameterization P^B should be within the same bounded region. Therefore, we need a parameterization method that allows us to control the boundary vertices of the parameterization. We adopt the intrinsic parameterization method [9] because it not only allows the user to define the parameterization boundary, but also minimizes both the Dirichlet energy and Chi energy on triangulations.

First, we project the boundary vertices onto a plane that best fits the boundary vertices. Next, using the intrinsic parameterization algorithm, we build and solve a linear system with the 2D parameterization coordinates of the internal vertices as the unknowns and the boundary parameterization coordinates as the known quantities. The bottom row of Figure 8 shows the parameterization results of the feature F and base B shown in the top row.

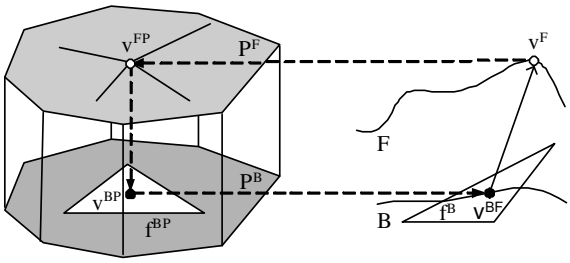

Figure 9. Local frame encoding based on surface parameterization.

We choose the plane projection of the boundary vertices as the boundary condition of the parameterization because the selection region usually has a planar disk-type boundary. For arbitrary boundaries, we first compute the parameterization P^B with a natural boundary using intrinsic parameterization. Next, we compute P^F with the fixed boundary of P^B. This order of steps guarantees that P^F and P^B can be exactly enclosed within the same region.

In addition, some internal vertices of P^F may be outside the parameterization boundary for meshes with non-zero genus. This undesirable result only occurs when the parameterization boundary is concave. Therefore, we can first parameterize the base surface B to get an approximating unfolded shape. Next, we compute the convex hull of the boundary vertices of P^B, and finally, we parameterize both F and B onto this convex hull. Adopting this scheme guarantees that our cut-and-paste algorithm is topologically free.

6.2. Local Frame Encoding

Although the boundary triangulation is likely to create a few new boundary points, it does not change the original boundary vertices, which guarantees that both F and B can be parameterized onto the same region in the plane. Putting these two parameterizations together (Figure 3 (c)), we can easily find a corresponding base point on the base for each vertex of the feature surface.

Firstly, for each vertex v^F of the feature surface F, we find the corresponding vertex v^{FP} on the feature surface parameterization P^F. Secondly, we find a face f^{BP} containing v^{FP} in P^B. Let v^{BP} be the barycentric coordinates of v^{FP} in f^{BP}, and let f^B be the corresponding face of

f^{BP} on B. According to v^{BP}, the barycentric coordinates of the base point v^{BF} in f^B is computed. Finally, the local detail vector $\overrightarrow{d^F} = v^F - v^{BF}$ is obtained and it is associated with the base point v^{BF} and the base face f^B (see Figure 9).

7. Detail Transferring and Gap Filling

Once the details are encoded with respect to the source base B^S, the next step is to paste B^S onto the target surface and then automatically reconstruct the detail information onto the target surface. An area on the target, within which B^S is to be pasted, must be determined before the pasting operation. To relieve the user of the task of specifying a target area and to retain the source base shape as much as possible, we utilize the parameterization of the source base surface to approximate a target area, as in [4] (Section 7.1). Let P^{SB} and P^{TB} denote the parameterizations of the source base surface and the target base surface, respectively. We extract the target base B^T from the approximated area, and then build a mapping between the parameterization P^{SB} of B^S and the parameterization P^{TB} of B^T. The detail surface is reconstructed on B^S after B^S is pasted onto B^T (Section 7.2). Finally, the resulting gap is filled and the pasting operation is completed (Section 7.3).

7.1. Approximating Target Area

The basic idea of the target area approximation algorithm is to do a geodesic walk on the original target surface (not the target base surface) according to the boundary information of P^{SB}.

Biermann et al. [4] proposed the following approach to identify the target region, within which the source feature will be pasted. First, the boundary of the source base surface parameterization is represented in a radial form. Line segments are constructed connecting the centroid of the boundary to each vertex of the boundary. When the user specifies a position and a direction on the target surface, the centroid is placed at the specified position. Each boundary point is then mapped to the target by performing geodesic walking according to its corresponding line segment. Finally, all sequential points are chained and the interior region is filled. This method requires determining the position of the centroid on the target mesh. Therefore, the user cannot easily specify the position and direction when the pasted area contains holes or highly curved region.

To address this problem, we propose the following method. Let $C = (v_1^P, v_2^P, ..., v_n^P, v_{n+1}^P = v_1^P)$ denote the boundary of P^{SB}. When the user first specifies the starting point v_1^M and the orientation $\overrightarrow{t_1}$, the point v_1^P is placed at v_1^M. Geodesically walk the distance of $\|v_1^P v_2^P\|$ along $\overrightarrow{t_1}$

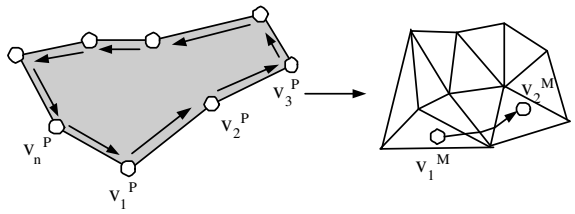

Figure 10. Geodesic walking

and arrive at the point v_2^M. According to the relative relationship between $\overrightarrow{v_3^P v_2^P}$ and $\overrightarrow{v_1^P v_2^P}$ and the direction at v_2^M, compute another direction $\overrightarrow{t_2}$. Repeat the above steps until all the positions of v_i^M $(1 \leq i \leq n)$ are computed (Figure 10). Finally, a flood fill algorithm is applied to obtain the target region.

7.2. Detail Reconstruction

Once the target region is found, we apply the same boundary triangulation technique used in the target base surface extraction to obtain the target base B^T. But there is a small difference in the edge splitting step. We use the average edge length of B^S instead of the minimum edge length to determine the density of B^T. This adjustment is necessary especially when the resolution of B^S is substantially higher than that of B^T. After that, both the target feature parameterization P^{TF} and target base parameterization P^{TB} are created within the same region and the details are encoded over B^T.

Note that our cut-and-paste algorithm can change the topology of the target mesh. This aim leads to the following forward mapping:

1. For each vertex v^{SBP} in P^{SB}, corresponding to v^{SB} in B^S, find a face f^{TBP} in P^{TB} containing v^{SBP}. According to the barycentric coordinates of v^{SBP} in f^{TBP}, compute the mapped coordinates of v^{SB} in B^T.

2. After all the source base vertices are mapped onto B^T, the local detail information is reconstructed on the pasted source base surface.

To ensure that the above steps work well for every vertex in P^{SB}, we should adjust P^{TB} to cover the entire P^{SB} by translation, rotation and scaling in the parameterization domain. Unfortunately, the region of P^{TB} not covered by P^{SB} causes a gap between the original part of the target surface and the newly pasted feature.

7.3. Gap Filling and Smoothing along Boundary

We fill the gap as in [2]. First, we find the closest vertex pair (u, v) between the boundary of the pasted source

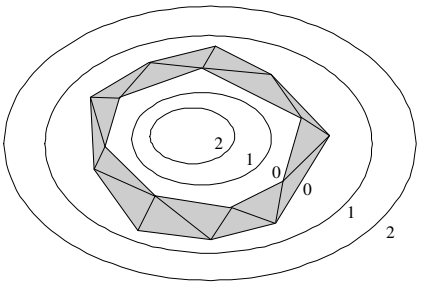

Figure 11. Smoothing around the gap boundary

feature and the boundary of the target surface and connect u and v. Next, we find a candidate from the neighboring vertices of these two vertices and connect it to u or v. This process terminates when the gap is triangulated.

After gap filling, we use a simple iterative smoothing technique to make the area smoother along the pasted patch boundary. In the first iteration, the umbrella operator [15] is applied on the vertices in the gap boundary $Ring0$; In the second iteration, the vertices of $Ring0$ and vertices of $Ring1$, which are adjacent to $Ring0$ and are on two sides of the gap, are smoothed, and so on (Figure 11). In general, 3 iterations are enough to get a smooth effect around the gap boundary.

8. Results and Discussions

Figure 12 shows an example of pasting the decorative texture cut from the Headus skull model [12] onto the Stanford bunny head. In this example, all the detail vectors of the detail surface are short, therefore it is less likely to cause high distortions as long as the target region is smooth.

Figure 14 illustrates an example that contains long detail vectors. In this case, a small deformation in the base surface will cause a large distortion of the detail surface if the previous detail encoding methods are adopted. Since our encoding algorithm guarantees that neighboring vertices have similar transformation behaviors when distortion is forced, the final result is smoother and more natural, even at the detail part furthest from the base surface.

It is always a challenge for the cut-and-paste editing operations to paste features onto highly curved regions because self-intersections are likely to occur. It is also the principal limitation of our algorithm. Figure 13 shows an example of pasting a teapot handle onto a concave surface. The base surface extracted from the teapot handle is convex, hence distortions, even self-intersections, inevitably occur after pasting. Adopting our detail encoding technique reduces the occurrence frequency of self-intersections and the degree of distortions. We note that the displacement volume method [5] solves the local self-intersection problem, and its iterative reconstruction scheme can be combined into our detail encoding method to completely avoid self-intersections.

Figure 15 (b) and (c) demonstrate a knot pasted inside a sphere and outside a sphere, respectively. In this example, the knot, as a source feature, is slim and has many vertices far away from the base surface, so its pasted result is quite sensitive to the distortion of the base surface. Although the interior of the sphere is concave and the exterior of the sphere is convex, our robust cut-and-paste algorithm works well for both cases.

Since each vertex of the source base is mapped onto a particular triangle of the target base, the density of the base mesh must be high in order to preserve the local shape of the source base. Two adjacent detail vertices may have their corresponding base points in different triangles on the base mesh. When these two triangles are mapped onto the target surface, the different distortions of these two triangles cause the corresponding distortions in the pasted feature. Therefore, increasing the density of the meshes can reduce the unsmooth transition band between the pasted feature vertices. However, increasing the density of the source and target bases means increasing the time to compute their parameterizations and the mapping between them. Therefore, the user should strike a balance between efficiency and quality. In general, to obtain a smooth pasted result, the resolution of the target base should be higher than that of the source base.

9. Conclusions and Future Work

We have proposed an elegant cut-and-paste algorithm, which can paste features with non-zero genus or areas with high curvature onto a target mesh more naturally. The base surface derived from the boundary of the cut feature makes the algorithm independent of the topology of the selected region. We also present a new detail encoding method based on surface parameterization, which simulates the physical properties of only the neighboring vertices interacting with each other. Finally, a forward mapping transfers the details of the source surface to the base of the target surface. This algorithm is efficient and satisfies the need of interactive editing.

Our surface pasting scheme consists of two parts: pasting the base surface and transferring the detail information. It is obvious that if the base surface has a large distortion, the reconstructed detail will be influenced significantly. In the future, we will try to let the user control the shape of the base surface, or even create an adaptive scheme, which can determine the shape of the source base according to the shape of the target base.

References

[1] G. Barequet, M. Dickerson, and D. Eppstein. On triangulating three-dimensional polygons. *Proc. 12th Symp. Computational Geometry*, pages 38–47, May 1996.

[2] G. Barequete and M. Sharir. Filling gaps in the boundary of a polyhedron. *Computer Aided Geometric Design*, 12(2):207–229, March 1995.

[3] R. Bartels and D. Forsey. Spline overlay surfaces. *Technical Report CS-92-08, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1*, 1991.

[4] H. Biermann, I. Martin, F. Bernardini, and Z. Zorin. Cut-and-paste editing of multiresolution surfaces. *Proc. of ACM SIGGRAPH '02*, pages 312–321, July 2002.

[5] M. Botsch and L. Kobbelt. Multiresolution surface representation based on displacement volumes. *Proceedings of Eurographics '03*, 2003.

[6] M. Botsch and L. P. Kobbelt. A robust procedure to eliminate degenerate faces from triangle meshes. *Vision, Modeling and Visualization (VMV01)*, Nov. 2001.

[7] L. Chan, S. Mann, and R. Bartels. World space surface pasting. *Proceedings of Graphics Interface*, pages 146–154, May 1997.

[8] B. Conrad and S. Mann. Better pasting via quasi-interpolation. *Curve and Surface Design: Saint-Malo, 1999*, pages 27–36, 1999.

[9] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. *Proceedings of Eurographics '02*, pages 209–218, 2002.

[10] M. Desbrun, M. Meyer, P. Schroder, and A. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. *Proc. of ACM SIGGRAPH '99*, 1999.

[11] Y. Furukawa and H. Masuda. Cut-and-paste editing based on constrained b-spline volume fitting. *Computer Graphics International '03*, July 2003.

[12] http://www.headus.com/au/.

[13] T. Kanai, H. Suzuki, J. Mitani, and F. Kimura. Interactive mesh fusion based on local 3d metamorphosis. *Proc. of Graphics Interface '99*, pages 148–156, June 1999.

[14] Klincsek and G.T. Minimal triangulations of polygonal domains. *Annals of Discrete Mathemtatics 9*, pages 121–123, 1980.

[15] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. *Proc. of ACM SIGGRAPH '98*, pages 105–114, 1998.

[16] L. Kobbelt, J. Vorsatz, and H.-P. Seidel. Multiresolution hierarchies on unstructured triangle meshes. *Computational Geometry: Theory and Application*, 1999.

[17] S. Mann and T. Yeung. Cylindrical surface pasting. *Geometric Modeling '99*, pages 233–248, 1999.

[18] K. Museth, D. E. Breen, R. T. Whitaker, and A. H. Barr. Level set surface editing operators. *ACM Transactions on Graphics*, 21(3):330–338, 2002.

[19] A. Sheffer and E. de Sturler. Surface parameterization for meshing by triangulation flattening. *Proceedings of the 9th International Meshing Roundtable, Sandia National Laboratories*, pages 161–172, Oct 2000.

[20] G. Taubin. Curve and surface smoothing without shrinkage. *Fifth International Conference on Computer Vision*, June 1995.

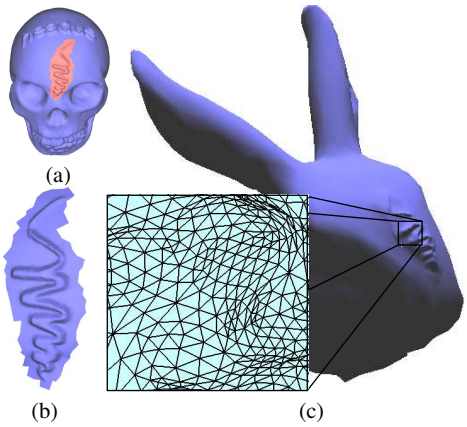

Figure 12. Bunny with decoration. (a) Headus skull model. (b) Feature cut from the skull model. (c) Pasted result.

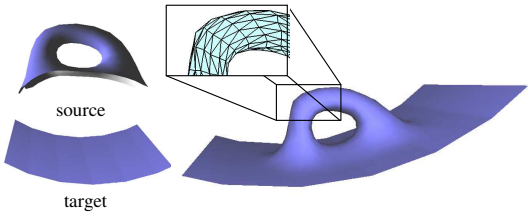

Figure 13. Pasting a teapot handle onto a concave region.

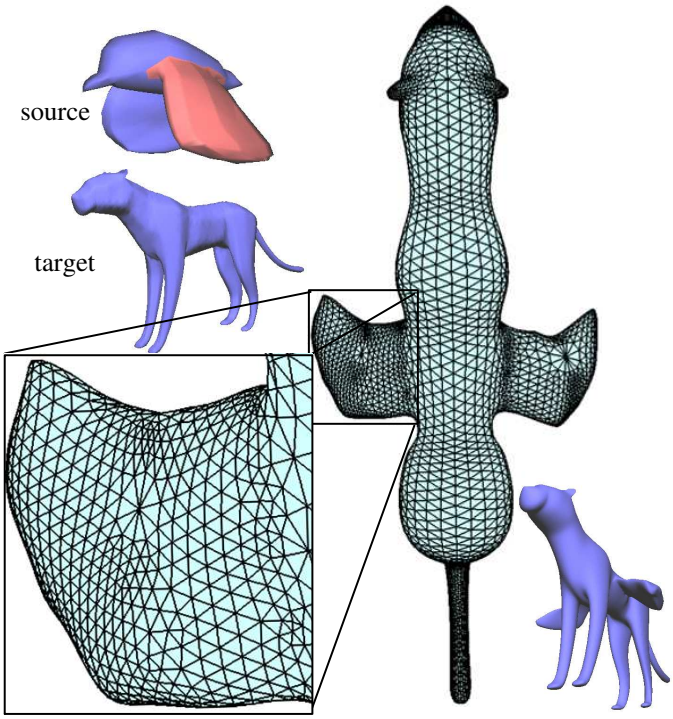

Figure 14. Jaguar with pasted wings.

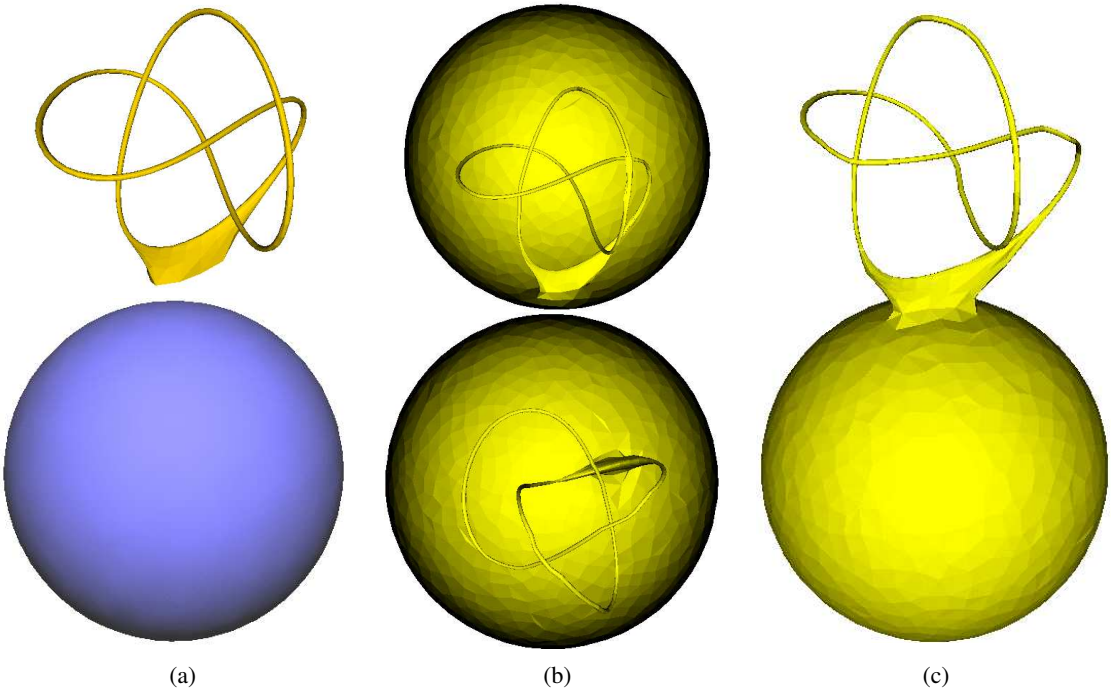

Figure 15. (a) Knot to be pasted (top) and target mesh (bottom). (b) Pasting a knot inside a sphere (two different views). (c) Pasting a knot outside a sphere.