A control of smooth deformations with topological change on a polyhedral mesh based on curves and loops

Anne Verroust and Matthieu Finiasz
INRIA Rocquencourt, Domaine de Voluceau, B.P. 105 78153 Le Chesnay Cedex, FRANCE
Anne.Verroust@inria.fr Matthieu.Finiasz@inria.fr

Abstract

We propose a method to model and control topological changes by a smooth deformation of a polyhedral mesh using curves and loops. As changing the genus of a surface is not a continuous transformation, the topological change is made when an intermediate shape between the two topologies has been obtained. The creation and the deletion of holes are studied. The deletion of a hole uses non null-homotopic loops to designate the hole to be deleted. A method computing two independent loops associated to a hole is presented.

1. Introduction

The problem of providing a simple and intuitive tool to control a smooth deformation of a polyhedral mesh has been the subject of numerous geometric studies (see for example [3, 24, 8, 21, 7, 5]). The general scheme of most of them is to embed the object in a deformable region of space (defined by a lattice, an axis or a set of points), to compute a parameterization of the object to define its position in the region, then to deform the region and compute the deformed object using the initial parameterization. Thus, if the object is a polyhedron, the underlying mesh is not modified during this process and the genus of the polyhedron does not change. Only Aubert and Bechmann [2], who introduce a 4th dimension representing time, embed the 3D objects in $\mathbb{R}^4$ in a volumic topology and model topological changes during the deformation process. This approach, however, raises two difficulties: firstly providing the user with a control tool that is really intuitive, and secondly computing the underlying mesh of the object's surface during the deformation process.

A control of topological changes for polyhedral meshes is proposed by Akleman et al. [1] but their aim is to introduce the possibility of creating handles or deleting holes in a shape modelling system. These operations produce discontinuities both on the shape and on the underlying mesh of the object during the modelling process, and thus a smooth deformation cannot be produced using their method.

Methods combining smooth deformations and modifications of the underlying meshes have been introduced in [12, 10, 11] to represent deformable models for 3D morphing or for 3D shape reconstruction purposes. In these papers the problem of providing a simple and explicit control of the topological transformation has not really been studied: in [10], the user controls the topological change by defining and mapping the same rough control mesh on both surfaces and in [12, 11], the topological change is automatically done during a physics-based deformation process. Our approach is based on a similar evolution scheme and we see in the following how curves and loops can be used to specify and model topological changes during the deformation.

Topological change

We focus here on a topological change which consists in adding (or deleting) a hole to (or from) a surface. This is not a continuous operation for the underlying mesh. If we want to obtain a smooth evolution process, the topological change cannot alter the geometry of the polyhedron: it must occur on a shape which can be represented by the two topologies and which is singular for these topologies. Thus, the problem is to define a limit shape between the two topologies and to build a continuous deformation between the polyhedron and this shape.

If we consider the case of the sphere and the torus, what types of limit shapes have we got? In his studies on morphogenesis [20], Koenderink presents two evolutionary sequences for a torus:

- The first one consists in pushing two points, "the north and the south poles", from the outside of a sphere toward each other. When the two points meet, he obtains a "pinched sphere". This shape can be a variety of torus, where the hole is reduced to a point and a variety of a sphere where two points meet together.
- The "strangled torus" is obtained by putting a wire loop round the torus and pulling it tight so as to reduce the diameter to zero. Here also, the strangled torus can be either a variety of torus or a variety of sphere.

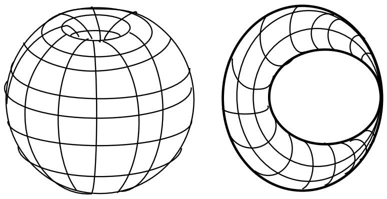


Figure 1. the two intermediate shapes. Left: the pinched sphere. Right: the strangled torus

These two shapes "the pinched sphere" and "the strangled torus" can both be obtained by a continuous deformation from a sphere or from a torus. Thus a smooth deformation from a sphere to a torus can be obtained by deforming the sphere into one of these two intermediate shapes, then changing the spherical underlying mesh into a toric one and finally deforming the shape into a torus. We can also transform a torus into a sphere by deforming it into pinched sphere or a strangled torus, cut the limit shape on its singular point to obtain a spherical surface and deform it again into a sphere. This transformation scheme has been used by DeCarlo et al. in a 3D morphing process [10] and in a reconstruction process [11].

Here, our goal is to offer the user a simple and intuitive control of the whole process. For this purpose, let us return to Koenderink's description. He builds a "pinched sphere" from a sphere pushing two points together inside the sphere and a "strangled torus" from a torus tightening a loop of the torus. We can follow the same strategy to built a "pinched sphere" from a torus and a "strangled torus" from a sphere: for the pinched sphere, it suffices to tighten a loop surrounding the hole of the torus and for the strangled torus, two points of the sphere are joined together following a curve located outside the sphere.
Thus, these transformations can be specified in terms of curves: loops on the surface of the torus and 3D curves lying outside or inside the sphere. These features are sufficiently simple to be manipulated by non mathematicians and seems to be a solution for a simple and intuitive control of the whole process.
In the following, we explain in detail how curves and loops are used to build these two transformations. In section 2, the case of the deletion of a hole is studied. As the user specifies the transformation by giving a loop surrounding a hole or a handle, which can be a non trivial task, a method computing two independent loops associated to a hole is introduced in this section. In section 3, the creation of a hole is explained. Finally, we comment on our results in section 4 and conclude in section 5.
Notations: we consider an oriented h-genus triangulated surface $\mathcal{M}$. We call *loop* an oriented simple cycle of edges of $\mathcal{M}$ and *non trivial loop* a loop which cannot be continuously contracted on $\mathcal{M}$ to a point, i.e. which is not null-homotopic.

2. Deletion of a hole

To delete a hole during a smooth deformation of $\mathcal{M}$, a non trivial loop l associated to the hole will be tightened to reduce it to a point. The deformation of $\mathcal{M}$ is local: two regions surrounding l on $\mathcal{M}$ will be deformed.
To control the process, the user gives:

- A non trivial loop l on $\mathcal{M}$.
- Two distance values L_L and L_R defining the neighboring regions of l which will be deformed and two continuous functions (i.e two functions f_L and f_R C^0, $f_{I,I=L,R} : [0, L_I] \rightarrow [0, 1]$ such that $f_I(0) = 1$ and $f_I(d_i) = 0$) controlling the deformation strength on the two sides of l.

In the next section, we suppose a non trivial loop given and we describe the deletion process. Then, in section 2.5, we focus our attention on finding non trivial loops on $\mathcal{M}$: a method computing automatically two independent non trivial loops on $\mathcal{M}$ intersecting on a given vertex of $\mathcal{M}$ is presented.

2.1. The algorithm

The main steps of the algorithm are the following:

1. An axis a_l is computed to control the deformation of $\mathcal{M}$ when shrinking l. This step is described in section 2.2
2. $\mathcal{M}$ is locally deformed by a homothetic transformation along a_l until l is reduced to a point (see section 2.3 for more detail). The intermediate shape between the two topologies is then reached.
3. The topology of $\mathcal{M}$ is modified by acting on its underlying mesh: the neighboring region of l in $\mathcal{M}$ is cut along l and two new vertices v_R and v_L are created to replace the vertices of l (cf. section 2.4).
4. v_R and v_L their neighboring regions move aside each other following the direction given by the curve a_l.

One can notice in Figure 2 that the type of intermediate shape between the two topologies can be either a strangled torus or a pinched sphere : it depends on the loop l chosen for the transformation. If l surrounds the hole (resp. the handle), a pinched sphere (resp. a strangled torus) is obtained.

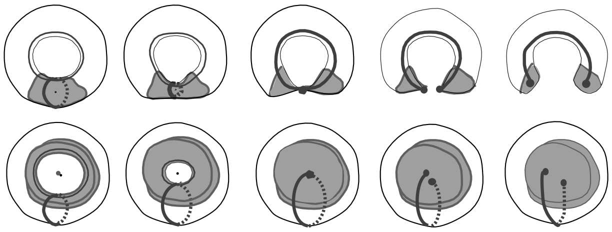


Figure 2. The deletion of a hole on a torus. The regions of $\mathcal{M}$ deformed during the process are drawn in grey. Up: the loop surrounding the handle is tightened; down: the loop surrounding the hole is tightened. The intermediate shapes between the two topologies are in the middle.

2.2. Computing a control axis

Given $\mathcal{M}$, a loop l on $\mathcal{M}$ and two values L_R and L_L, an axis will be computed, following a similar process than in [23] having a set of vertices instead of a source point as source for the distance function d_l on $\mathcal{M}$.

More precisely, d_l is computed as follows : we set $d_l(v) = 0$ for all the vertices belonging to l and we compute an approximation of the geodesic distance of the vertices of $\mathcal{M}$ to the loop l using a Dijkstra algorithm [9] on the graph induced by the edges of $\mathcal{M}$. Thus for any vertex v, $d_l(v)$ is the length of the shortest path of edges to l. This distance function is extended by a linear interpolation on the points lying on edges of $\mathcal{M}$.

We then compute the level sets of d_l on $\mathcal{M}$ and take as axis a_l the polygonal curve joining the successive centroids of the level sets until the distance L_L is reached for the left side of l and L_R for its right side (cf. Figure 3).

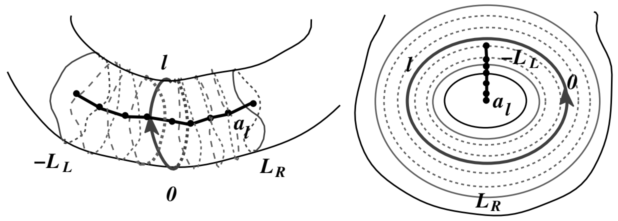


Figure 3. Are drawn in dark grey, the loop l; in grey, the limits of the deformation neighbor; in dashed lines, the level sets of the oriented distance function, and in black, the 3D polygonal curve a_l defining an axis.

One can note that the axis can fall outside the shape, as in the right shape of figure 3. In this case, the intermediate shape will be a pinched sphere.

2.3. Tightening the loop

The loop is tightened on its centroid and each vertex v belonging to the deformation neighbor of l is associated to a point p_v of a_l. Its deformation consists in a homothety of center p_v. The intensity of the homothety depends on the distance from l and the time t. It is given by two deformation functions f_L and f_R which are equal to 1 when the distance value d_l is equal to 0 and equal to 0 when the d_l is equal to L_L or L_R. These functions can define an assymetric deformation on the two sides of the loop , as in Figure 4.

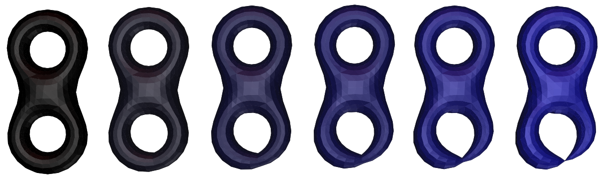


Figure 4. An assymetric deformation.

2.4. Changing the topology

This step does not change the geometry of $\mathcal{M}$ but only its topology. l is reduced to a point. A cut of $\mathcal{M}$ is done along l and two new vertices v_L and v_R are created. Faces of $\mathcal{M}$ adjacent to an edge of l are eliminated and faces adjacent to a vertex of l are replaced by faces adjacent to v_L or v_R according to their location w.r.t. l (cf. Figure 5).

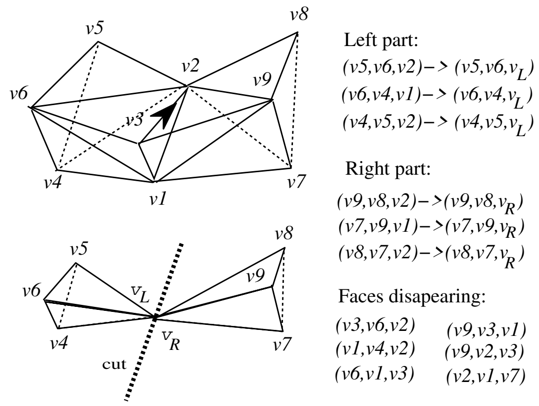


Figure 5. changing the topology

2.5. Non trivial loops

To delete a hole, the user has to specify a loop around where the surface will be deformed and cut. If the loop is trivial, then the operation will cut in two parts the surface. In the other case, a hole will be deleted from the surface.

We want to provide a tool computing non trivial loops of $\mathcal{M}$ to the user. Before going onto a detailed description of our method, let us examine existing approaches developed recently on topological computations on surface such as reducing the genus of a polyhedron or computing non trivial loops [15, 17, 16, 22]:

- Kartasheva [17] computes the cutting surface with a method based on the calculation of the Betti group of the polyhedron. The boundary of the cutting surface obtained by Kartasheva's algorithm is one of the two independent loops associated to a hole of the polyhedron.

- To cut a torus, Fujimura [15] computes two independent non trivial loops associated to a hole by sweeping a plane. By analyzing the successive cross-sections of the surface, holes are located and the loops are computed. For complex surfaces, the computation of the successive cross-sections may become unnecessary very expensive.

- To remove what they call "topological noise" on meshes, Guskov and Wood [16] use a local wave front traversal to locate small tunnels. When these features are located, they cut and seal the mesh to reduce its genus.

- To compute a set of loops forming a canonical fundamental polygon of $\mathcal{M}$ (cf. [14]), Lazarus et al. [22] propose two combinatorial algorithms: the first one is a modified version of [25] based on a wave front traversal of the polyhedron from a vertex of $\mathcal{M}$ and the second one is based on Brahana's reduction operations [6]. These two approaches are optimal in complexity but there is no geometric control of the resulting set of loops .

Our aim is to compute only two independent non trivial loops corresponding to a hole and, if the surface is a torus, to obtain a loop surrounding the hole and a loop surrounding the handle as in Figure 2. To indicate which hole of $\mathcal{M}$ is concerned, we ask the user to designate a vertex on $\mathcal{M}$ located near the hole where the two loops to be computed will intersect.
If we consider the three last approaches [15, 16, 22], a hole or a tunnel is detected by analyzing the boundary $\mathcal{B}$ of the visited part of the surface during a traversal of $\mathcal{M}$. For Fujimura, the traversal of $\mathcal{M}$ is a sweeping plane traversal and the boundary $\mathcal{B}$ is the cross-sectional plane and, for the others, a wave front traversal is made from a vertex v_s of $\mathcal{M}$, beginning with a face t_0 adjacent to v_s and growing the region t of visited faces by adding faces adjacent to t one by one. A hole is then detected when $\mathcal{B}$ can be split in two parts and $\mathcal{M} \setminus t$ is connected (cf. Figure 6).

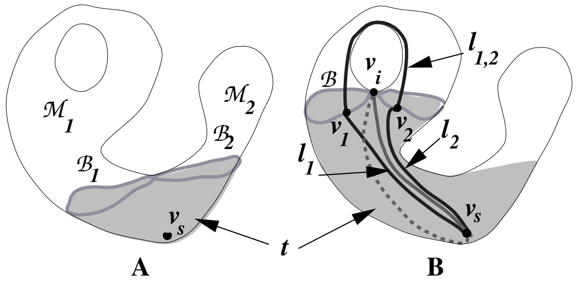


Figure 6. A: $\mathcal{B}$ presents a singularity but $\mathcal{M} \setminus t$ is not connected. $\mathcal{B}$ is split in two parts $\mathcal{B}_1$ and $\mathcal{B}_2$ and $\mathcal{M}$ in three pieces. B: $\mathcal{B}$ presents a singularity and $\mathcal{M} \setminus t$ is connected, thus a hole is detected and the two loops are created.

Here also, to locate a hole near a vertex v_s of $\mathcal{M}$ and to built two independent non trivial loops we follow a similar strategy:

1. From the vertex v_s given by the user, we make a "potato peeling" (cf. Figure 7) traversal[1] the faces of $\mathcal{M}$, maintaining the set of visited faces t and the boundary $\mathcal{B}$ of t. When $\mathcal{B}$ is split in two parts $\mathcal{B}_1$ and $\mathcal{B}_2$ having vertex v_i in common, we try to find a path connecting $\mathcal{B}_1$ and $\mathcal{B}_2$ in the unvisited part of $\mathcal{M}$. This is done by performing a tandem search traversing the edges of $\mathcal{M} \setminus t$ in parallel from $\mathcal{B}_1$ and $\mathcal{B}_2$.

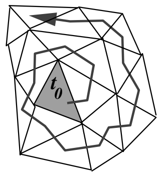


Figure 7. A potato peeling traversal of $\mathcal{M}$ from t_0.

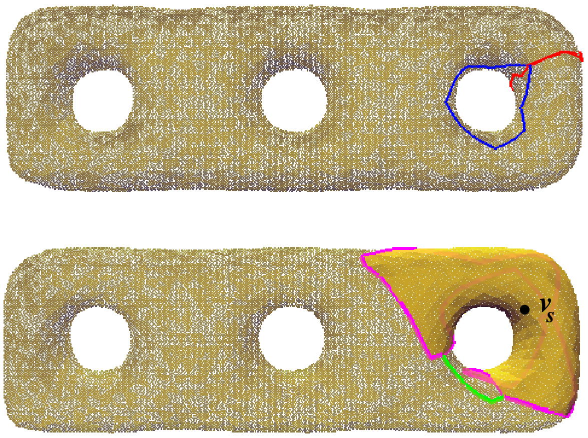
2. – If $\mathcal{M} \setminus t$ is not connected, $\mathcal{B}_1$ and $\mathcal{B}_2$ disconnect $\mathcal{M}$ in three pieces: t, $\mathcal{M}_1$ and $\mathcal{M}_2$. Suppose $\mathcal{M}_1$ is the part of $\mathcal{M} \setminus t$ entirely traversed while looking for a connecting path between $\mathcal{B}_1$ and $\mathcal{B}_2$. If the genus of $\mathcal{M}_1$ is positive, we return to step 1, taking $\mathcal{B}_1$ as boundary . If it is not the case, we return to step 1, with $\mathcal{B}_2$ as boundary.
   – In the other case, we consider a shortest edge path $l_{1,2} = e_1, ..e_k$ connecting $\mathcal{B}_1$ and $\mathcal{B}_2$ in $\mathcal{M} \setminus t$ (this path is found during the tandem search done in 1.). Let v_1 the vertex of e_1 belonging to $\mathcal{B}_1$ and v_2 the vertex of e_k belonging to $\mathcal{B}_2$. We

[1] in the potato peeling traversal, the order of the visited faces is similar than the order produced by a Dijkstra's algorithm on the dual graph used in [16] and it is less expensive to compute.

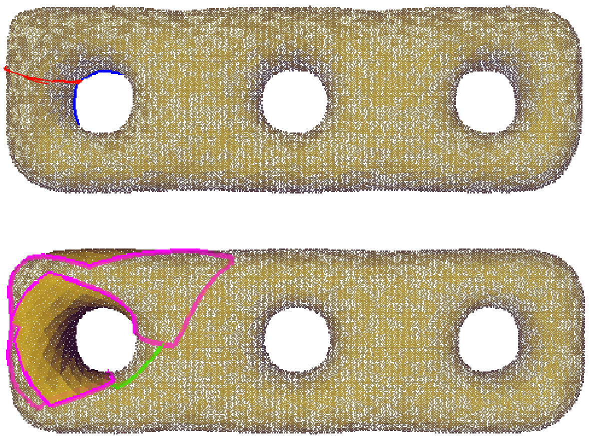
compute two shortest edge paths l_1 and l_2 in t, joining respectively v_s and v_1 and v_2 and v_s.

The first loop is composed of the paths of edges l_1, $l_{1,2}$ and l_2.

Two shortest paths of edges joining v_s and v_i computed on the two sides of t around v_i constitute the second loop (cf. Figures 6 and 8).



Front view



Back view

Figure 8. Two views of the same computation. Up: the two loops ; down: the boundary $\mathcal{B}$ presents a singularity and a join path is computed on $\mathcal{M} \setminus t$. The visited part t of $\mathcal{M}$ is darker.

Remark: if two holes are near the vertex, as it is the case for vertices v_1 and v_2 in figure 9, only one couple of loops will be computed, corresponding to one of the holes. Thus the user must select with attention the vertex to designate a specific hole.

Complexity: if n is the size of $\mathcal{M}$, the complexity of the algorithm reduces to the complexity of the traversal of $\mathcal{M}$ until a hole is detected (which is $\mathcal{O}(n)$ following the same arguments than in [22]) plus the time to compute shortest paths (we use Dijkstra's algorithm which requires time $\mathcal{O}(n \log n)$). Thus the overall complexity is $\mathcal{O}(n \log n)$.

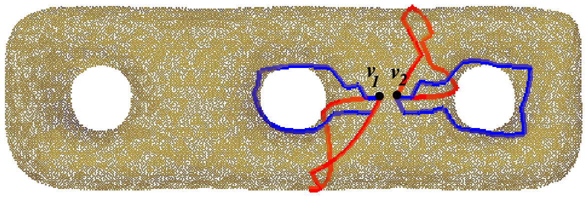


Figure 9. Two couples of independent loops computed from two different points. One can see that designating a point located on this handle may be ambiguous.

3. Creation of a handle or a hole

The creation of a handle or a hole will be made by the following process. A 3D curve $\mathcal{C}$ joining two vertices v_R and v_L of P is given by the user. This curve must be either entirely inside P or outside P. In the first case, a hole will be created and in the second case a handle will be created. A point on this curve is designated as the junction point v_j and a 2D closed polygonal curve $\mathcal{C}_J$ is given by the user. The surface is progressively deformed locally around v_R and v_L along $\mathcal{C}$ and the intermediate shape is obtained when the two vertices v_L and v_R have reached the junction point v_j. Then the mesh of $\mathcal{M}$ is modified, adding faces (geometrically reduced to an edge) and edges (reduced to a point) such that the genus of $\mathcal{M}$ increases by one. Finally, to obtain a non singular surface, $\mathcal{M}$ is deformed around v_j to reach the curve given by the user.
The input parameters of the process are:

- Two vertices v_L and v_R of $\mathcal{M}$, two distance values L_L and L_R defining the region of $\mathcal{M}$ which will be deformed around v_L and v_R and two continuous functions (i.e two functions f_L and f_R C^0, $f_{I,I=L,R}$: $[0, L_I] \to [0, 1]$ such that $f_I(0) = 1$ and $f_I(d_i) = 0$) controlling the deformation strength on the neighbors of v_L and v_R .

- A 3D curve $\mathcal{C}$ having v_L and v_R as extremities. $\mathcal{C}$ lies outside or inside $\mathcal{M}$.

- A point v_J on $\mathcal{C}$

- A closed 2D simple polygonal curve $\mathcal{C}_J = (e_1, ..., e_k)$ around v_J. The 3D embedding of $\mathcal{C}_J$ will be one of the two loops associated to the new hole.

3.1. The algorithm

The main steps of the process are :

1. Given v_L, v_R, the distance values L_L and L_R, the deformation functions f_L and f_R, the point v_J and a 3D polygonal curve $\mathcal{C}$ joining v_L and v_R, the surface $\mathcal{M}$ is progressively deformed along $\mathcal{C}$ until v_L and v_R reach v_J. The shape of $\mathcal{M}$ is then intermediate between the two topologies. This step is described in section 3.2.

2. The topology of $\mathcal{M}$ is modified: v_L and v_R are replaced by a closed sequence of edges $e_1, ...e_k$ having the same length than $\mathcal{C}_J$ and new faces are added around v_L and v_R. Then the genus of $\mathcal{M}$ is increased by one (see section 3.3 for details).

3. The shape of $\mathcal{M}$ is enlarged around the junction and at the end of the process the sequence $e_1, ..., e_k$ forms the curve $\mathcal{C}_J$. As in section 2.3, this is done by a homothetic transformation of the neighboring regions R_R and R_L along the curve $\mathcal{C}$.

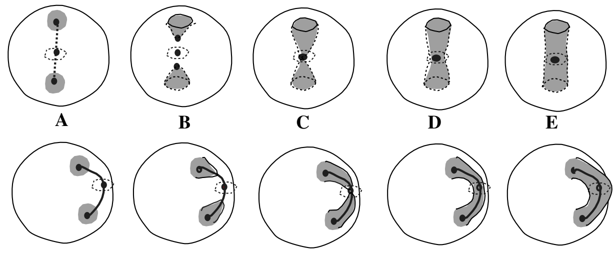


Figure 10. Up : the 3D curve $\mathcal{C}_J$ is inside $\mathcal{M}$. Down: $\mathcal{C}_J$ is outside $\mathcal{M}$. A: $\mathcal{M}$; C: the intermediate shape ; E: $\mathcal{M}$ at the end of the deformation process

3.2. Obtaining the limit shape

Given (v_L, L_L) and (v_R, L_R), we compute the neighboring regions R_L and R_R to be deformed : for $i = R, L$, R_i is composed of the vertices having a distance d_i to v_i on $\mathcal{M}$ less than L_i, where $d_i(v)$ is the length of the shortest path of edges from v to v_i.

- The curve $\mathcal{C}$ is discretized and a 3D frame is associated to each discretization point. As in [21], we have chosen a rotation minimizing orthogonal frame along $\mathcal{C}$ [4, 19].

- Then, using the moving frame along $\mathcal{C}$, each vertex belonging to R_i is translated along the part of $\mathcal{C}$ joining v_i and v_J. The length of the translation is determined by the distance $d_i(v)$, the deformation function f_i, the length of the part of $\mathcal{C}$ joining v_i and v_J and the time t.
At the end of this step, the limit shape is reached, v_R and v_L coincide on v_j and each vertex v belonging to R_R and R_L is associated to a point p_v of $\mathcal{C}$ as in section 2.3.

3.3. Changing the topology

In this step, the two vertices v_L and v_R will be replaced by a sequence of edges $e_1, ...e_k$ geometrically reduced to v_J.
To add properly new faces around $e_1, ...e_k$, we proceed in a similar way around v_L and v_R as follows:

- The 2D curve $\mathcal{C}_J$ is embedded in the reference plane P_J associated to v_J on $\mathcal{C}$.
- The faces of $\mathcal{M}$ adjacent to $v_{i,i=L,R}$ are projected on P_J (case A of Figure 11).
- A homothety of center v_J is applied on $\mathcal{C}_J$ so that $\mathcal{C}_J$ stay inside the set of faces adjacent to $v_{i,i=L,R}$ (case B of Figure 11)
- We use the half lines joining v_J and the extremities of the segments forming $\mathcal{C}_J$ and the half lines joining v_J and the middle of the segments forming $\mathcal{C}_J$ to compute the new faces to be created, as in case C of Figure 11.

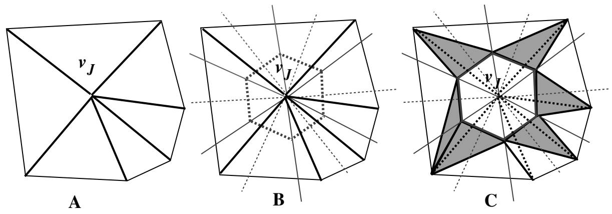


Figure 11. A: the original faces in the projected plane; B: $\mathcal{C}_J$ is drawn in dashed lines ; C: six new faces are created (drawn in grey)

4. Results

All the algorithms have been implemented using the polyhedral objects [18] of the `C++` library CGAL (`http://www.cgal.org/`). The results are interactive (no more than a minute in an O2 Silicon Graphics).

• The two loops of the genus 1 torus of Figure 12 have been computed by our method. One can see that taking, as described in section 2.5, the shortest paths of edges to build the loops leads to a correct set of non trivial loops : the first loop surrounds once the handle and does not surround the hole and the second loop surrounds once the hole and does not surround the handle. The hole is deleted by tightening the first loop . Figure 13 shows how the shape is deformed around the first loop .

• On the double torus of Figure 14, the two holes are successively deleted. From steps 1 to 4, the surface is tightened around the loop drawn in 0, as a strangled torus. Then,

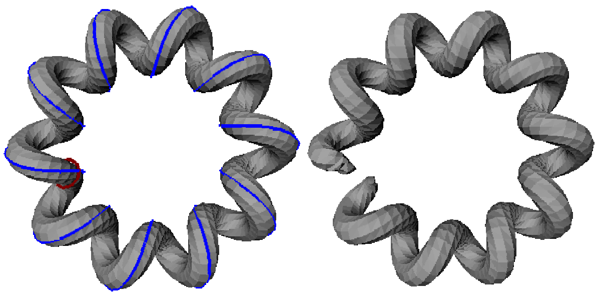


Figure 12. The two loops and the final result of the deletion process using the loop which surrounds the handle.

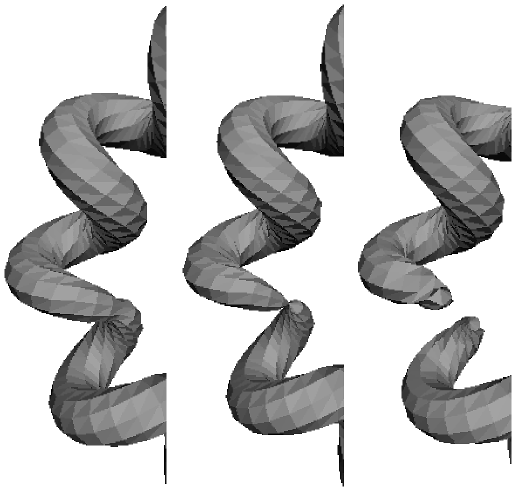


Figure 13. Intermediate shapes of the deformation process.

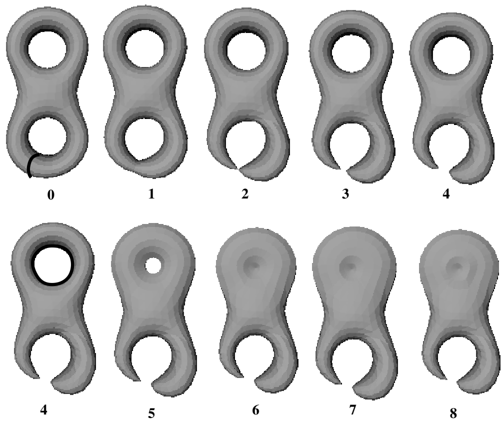


Figure 14. The two holes of the double torus are deleted.

from steps 4 to 8, the loop drawn in 4 is tightened. The two intermediate shapes are on steps 2 and 6.

• On Figure 15, we add a handle to a toric shape joining the same points varying the other input parameters. The leftmost handle has a deformation neighbor smaller than the others and the rightmost handle was defined with a circular curve joining the two points.

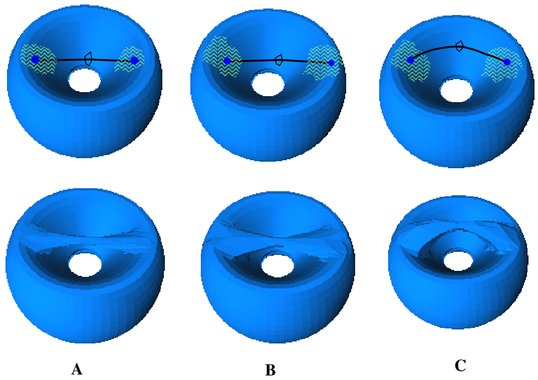


Figure 15. Influence of the parameters on the result.

• On the genus 3 torus of Figure 16, first a hole is deleted, tightening the loop and then a handle is created, joining the two points which replaced the first loop .

5. Conclusion

We have shown how curves and loops can be used to control and model smooth deformations with topological changes on polyhedral shapes. The control is simple and intuitive and can be proposed to any user. To improve the smoothness of the result when creating or deleting holes, the part of the mesh to be deformed may be subdivided before performing the deformation. This deformation tool can be enhanced by composing it with other geometric deformation models.
We have presented a method computing non trivial loops which provides the user two candidate loops to perform a hole deletion. As Kartasheva [17] and Fujimura [15] notice, these loops can also be used to cut the surface in parts for other purposes such as texture mapping h genus polyhedral shapes.
These methods can easily be extended to any surface having an underling triangular mesh such as surfaces defined by triangular patches.

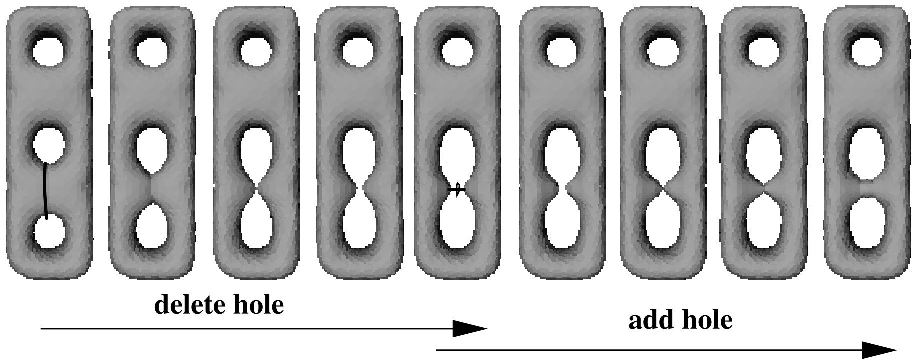


Figure 16. Two successive topological changes.

Acknowledgments

A part of this work was the subject of Matthieu Finiasz's DEA [13]. We would like to thank Francis Lazarus for many useful discussions at the beginning of this work.

References

[1] E. Akleman, J. Chen, and V. Srinivasan. A new paradigm for changing topology of 2-manifold polygonal meshes. In *Pacific Graphics'2000*, pages 192–201, 2000.

[2] F. Aubert and D. Bechmann. Animation by deformation of space-time objects. *Computer Graphics Forum*, 16(3):57–66, Aug. 1997.

[3] A. H. Barr. Global and local deformations of solid primitives. *Computer Graphics (SIGGRAPH'84)*, 18(3):21–29, 1984.

[4] R. Bishop. There is more than a way to frame a curve. *American Mathematical Monthly*, 82:246–251, 1975.

[5] P. Borrel and D. Bechmann. Deformation of n-dimensional objects. *International Journal of Computational Geometry and Applications (Special Issue : Solid Modeling II)*, 1(4):427–453, Sept. 1991.

[6] T. Brahana. Systems of circuits on 2-dimensional manifolds. *Ann. Math.*, 23:144–168, 1921.

[7] Y. Chang and A. Rockwood. A generalized de Casteljau approach to 3D free form deformation. *Computer Graphics (SIGGRAPH '94)*, 28:257–260, Aug. 1994.

[8] S. Coquillart. Extended free-form deformation: A sculpturing tool for 3D geometric modeling. *Computer Graphics (SIGGRAPH '90)*, 24:187–196, Aug. 1990.

[9] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.

[10] D. DeCarlo and J. Gallier. Topological evolution of surfaces. In *Graphics Interface'96*, pages 194–203, Toronto, Ontario, Canada, 1996. Canadian Information Processing Society.

[11] D. DeCarlo and D. Metaxas. Shape evolution with structural and topological changes using blending. *IEEE PAMI*, 20(11):1186–1205, Nov. 1998.

[12] H. Delingette, Y. Watanabe, and Y. Suenaga. Simplex based animation. In *Computer Animation'93, Models and Techniques in Computer Animation*, pages 13–28, Geneva, Switzerland, 1993. Springer Verlag.

[13] M. Finiasz. Déformation de polyèdre avec changements de topologie. Rapport de DEA, DEA Algorithmique, Paris, June 2000.

[14] A. T. Fomenko and T. L. Kunii. *Topological Modeling for Visualization*. Springer-Verlag, 1997.

[15] K. Fujimura. On cutting a torus. In *International Conference on Shape Modelling and Applications*, Aizu-Wakamatsu, Mar. 1997. IEEE Computer Society.

[16] I. Guskov and Z. Wood. Topological noise removal. In *Graphic Interface'2001*, Ottawa, Ontario, Canada, June 2001.

[17] H. Kartasheva. The algorithm for automatic cutting od three-dimensional polyhedron of h-genus. In *International Conference of Shape Modelling and Applications*, pages 26–33, Aizu, Japan, Mar. 1999. IEEE Computer Society.

[18] L. Kettner. Designing a data structure for polyhedral surfaces. In *Proc. 14th Annual ACM Symposium on Computational Geometry*, pages 146–154, 1998.

[19] F. Klok. Two moving coordinate frames for sweeping along a 3D trajectory. *Computer Aided Geometric Design*, 3:217–229, 1986.

[20] J. Koenderink. *Solid Shape*. MIT Press, 1989.

[21] F. Lazarus, S. Coquillart, and P. Jancène. Axial deformations: an intuitive deformation technique. *Computer Aided Design*, 26(8):607–613, Aug. 1994.

[22] F. Lazarus, M. Pocchiola, G. Vegter, and A. Verroust. Computing a canonical polygonal schema of an orientable triangulated surface. In *Proc. 17th Annual ACM Symposium on Computational Geometry*, pages 80–89, June 2001.

[23] F. Lazarus and A. Verroust. Level set diagrams of polyhedral objets. In *SMA'99 (Fifth ACM Symposium on Solid Modeling and Applications)*, Ann Arbor, June 1999. ACM Press.

[24] T. Sedeberg and S. Parry. Free-form deformations of solid geometric models. *Computer Graphics (SIGGRAPH'86)*, 20(4):151–160, 1986.

[25] G. Vegter and C. K. Yap. Computational complexity of combinatorial surfaces. In *Proc. 6th Annual ACM Symposium on Computational Geometry*, pages 102–111, 1990.