Geodesic Bézier Curves on Triangle Meshes

Dimas Martínez Morera, IMPA - Brazil / ICIMAF - Cuba
Paulo Cezar Carvalho, IMPA - Brazil
Luiz Velho, IMPA - Brazil

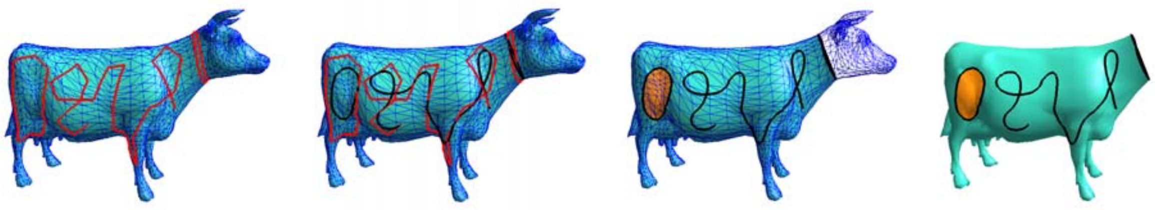
Figure 1: Modeling on the surface of a cow. From left: the control polygons of two closed curves and a C^1 spline; the corresponding curves; a region is filled with a different color and other is trimmed; final result.

Designing free-form curves is a basic operation in Geometric Modeling. Doing so in Euclidean space is a widely studied problem. The problem becomes harder if we wish to design on a curved geometry, such as triangulated surfaces.

In this paper we introduce a new class of curves that are suitable for free-form modeling directly on the geometry of a manifold mesh. Defined by means of the de Casteljau Algorithm, they are a natural generalization of Bézier curves. Thus, we call them “Geodesic Bézier Curves”.

Geodesic Bézier curves are defined by means of the subdivision algorithm for classical Bézier curves. Given a control polygon $P_0, P_1, \ldots, P_n$ on a surface $\mathcal{S}$, we want to compute a curve $\mathcal{C}$ on $\mathcal{S}$ interpolating P_0 and P_n, whose shape is controlled by the position of the interior points $P_1, P_2, \ldots, P_{n-1}$. The curve $\mathcal{C}$ is defined as the limit of the Bézier subdivision scheme. The de Casteljau algorithm provides a geometric procedure to evaluate a Bézier curve at any parameter $u \in [0,1]$, using repeated linear interpolation: The main difference in a polygonal surface from the planar case is that the sides of the control polygon are no longer line segments, but geodesics connecting the control points. This imposes the necessity of modifying the interpolation step of the de Casteljau Algorithm. The equivalent to linear interpolation in the geometry of the surface is the interpolation along geodesic lines. Algorithm 1 describes the interpolation step in the case of manifold triangulations:

Algorithm 1 *Interpolation on Manifold Triangulations*

```
Input: A manifold triangulation $\mathcal{S}$, two points $Q_1$ and $Q_2$
       on it and a parameter $u \in [0,1]$
Output: A point $Q$ on $\mathcal{S}$ interpolating $Q_1$ and $Q_2$.
  step 1. $\gamma$ = ComputeGeodesic($Q_1, Q_2$).
  step 2. $Q$ = the point of $\gamma$ such that
              $d_\gamma(Q_1, Q) = u d_\gamma(Q_1, Q_2)$
```

In algorithm 1, $d_\gamma(A,B)$ computes the distance between A and B along γ. Note that since γ is a polygonal line, it is very simple to perform step 2.

To compute an approximation of $\mathcal{C}$, we can use the subdivision adaptive algorithm. It stops at some prescribed level of subdivision or when the control polygon can be considered as a geodesic segment; i.e., when all of its control vertices have error smaller than a prescribed tolerance.

The use of de Casteljau's algorithm in the definition of geodesic Bézier curves makes them a generalization of planar Bézier curves. Since a geodesic on a plane is a straight line, when the triangulation is planar both concepts coincide.

Modeling : In order to model with geodesic Bézier curves we must be able to perform the usual modeling operations, such as interactive editing, region fill and trimming.

Fast user interaction is very important in free-form design. The user should be able to modify any previously defined curve by changing the position of some of its control points. Efficient curve manipulation is achieved because we employ an incremental algorithm for geodesic computation [1]. Every time the position of a control point is changed, its neighboring sides in the control polygon should be recomputed. These (at most two) sides are geodesic lines. Each new (recomputed) geodesic is very close to the old (original) one, since one of their extremes remain fixed while the other one is very close to the corresponding extreme in the old curve. Therefore this update process runs very fast using our incremental algorithm. Note that, during interaction only the shape of the control polygon is shown. When the user releases the mouse, the curve is recomputed and displayed.

Region Fill and Trimming operations involve the problem of identifying a piece of a surface $\mathcal{S}$ limited by one or more curves defined on it. Solving this problem allows us to trim (cut) a piece of $\mathcal{S}$, to paint it with a certain color, or to map a texture to it. Given a point P in $\mathcal{S}$, typically obtained by a mouse click, the idea is to use a flood-fill algorithm, propagating a wavefront from this point until it reaches the boundary curves. We developed a simple algorithm to identify the faces in the region $\mathcal{R}$ that contains the point P and to split the faces bounding $\mathcal{R}$. Figure 1 shows an example of modeling operations using geodesic Bézier curves .

Future Work : There remain some theoretical issues associated with geodesic Bézier curves; it will be very interesting to see which other properties of classical Bézier curves hold for the new curves and also which concepts can be generalized to the geometry of manifold triangulations. The continuity of the curves at mesh vertices has to be studied. Is there something equivalent to affine invariance of Bézier curves in the case of geodesic Bézier curves?

[1] Martínez, D., Velho, L., Carvalho, P. 2005. Computing geodesics on triangular meshes. *Computer and Graphics 29:5*,p.667–675.