EUROGRAPHICS '94 / M. Daehlen and L. Kjelldahl (Guest Editors), Blackwell Publishers

Volume 13, (1994), number 3

AGRELs and BIPs: Metamorphosis as a Bezier curve in the space of polyhedra

Jarek Rossignac and Anil Kaul

IBM Research, P.O. Box 704, Yorktown Heights, New York 10598.
Phone: 914-784-7630, Fax: -7667, Email: jarek@watson.ibm.com

Abstract

The metamorphosis between two user-specified objects offers an intuitive metaphor for designing animations of deforming shapes. We present a new technique for interactively editing such deformations and for animating them in realtime. Besides the starting and ending shapes, our approach offers easy to use additional control over the deformations. The new Bezier Interpolating Polyhedron (BIP) provides a graphics representation of such a deforming object formulated mathematically as a point describing a Bezier curve in the space of all polyhedra. We replace, in the Bezier formulation, the traditional control points by arbitrary polyhedra and the vector addition by the Minkowski sum. BIPs are composed of Animated GRaphic ELement (AGRELs), which are faces with constant orientation, but with parametrized vertices represented by Bezier curves. AGRELs were designed to efficiently support smooth realtime animation on commercially available rendering hardware. We provide a tested algorithm for automatically computing BIPs from the sequence of control polyhedra and demonstrate its applications to animation design.

Keywords: Computational Geometry, Solid Representation, Deformations, Metamorphosis, Morphing, Polyhedra, Animation, Bezier curves.

1. Introduction

The advent of realtime graphics hardware opens new avenues for the interactive design of animated geometric models. Mechanisms of rigid bodies and parametrized geometric primitives supported by many CAD systems are inappropriate for designing deforming shapes. The behavior of more general deformable models [10] may be specified in terms of global parameters, such as forces and elasticity coefficients. However, the precise simulations of such physical models are computationally expensive and the resulting deformations difficult to adjust [11]. More local control is obtained by applying simple animations to free-form deformation techniques (FFD). (FFDs are based on trivariate polynomial mappings that deform simple volumes and thus all the objects imbedded in these volumes [2, 4, 8].

Deformations may be animated by prescribing motions to the control points or constraints defining the above mappings [3, 5, 11].) Because, free-form deformation techniques provide only indirect means for specifying the final shapes, more direct primitives for designing deformations are also needed. One such primitive is the metamorphosis between two user-specified shapes. Several techniques, restricted to polyhedra, have been proposed for automatically computing the intermediate shapes. These techniques find a matching between the boundary elements of both objects. Algorithms for computing such matchings for convex polyhedra [6], for star-shaped polyhedra, and for arbitrary polyhedra [7] are available. Animation is produced by linearly interpolating between matching elements. The starting and the ending shapes of a metamorphosis may be specified and later adjusted using conventional CAD techniques. They do not however completely define the deformation process, which largely depends on the chosen strategies for matching boundary elements.

We extend previous work on the computation and animation of polyhedral metamorphosis by: (a) providing a graphics datastructure, called AGRELs (short for Animated GRaphic ELements), for animated faces that interpolate boundary elements in a non-linear fashion; (b) defining smoothly deforming solids as points describing Bezier curves [1] in the space of all polyhedra (see Fig. 1);*(c) developing the BIP (short for Bezier Interpolating Polyhedron) representation for such deforming objects; and (d) presenting algorithms for computing BIPs from sequences of arbitrary *control* polyhedra.

2. Datastructure

Contemporary graphics adapters offer hardware support for rendering polygons, specified in terms of their vertices, and for tracing Bezier and other polynomial curves, specified in terms of control points. AGRELs combine these two concepts as follows. Each vertex of a face is defined as a point on a Bezier curve represented by its four control points[1]. For a given value of the time-parameter, the coordinates of each vertex are computed by evaluating the associated Bezier curve and used for rendering the face. The normal directions associated with the AGRELs used in this paper are independent of time, and thus need not be recomputed during animation.

Boundary representations, which define solids by a list of their bounding faces, are well suited for hardware-supported rendering. To avoid data duplication, boundary representations are usually stored as a list of vertices and a list of faces, each defined by a normal direction and by one or several loops. A vertex is defined by three coordinates. A loop is an ordered list of references to vertices. Cubic BIPs extend such boundary representations to animated models as follows. In addition to the three coordinates, which will be recomputed during animation as the time-parameter is changing, we store with each vertex four references, each one into a different array of control points[2]. For performance improvements, two sets of coordinates are associated with each control point: the initial fixed coordinates and the scaled coordinates, which change with time.

3. Animation

For a given value of the time-parameter, the time-dependent coordinates of all of the vertices are obtained by: (a) evaluating the four weights of the cubic Bezier formulation for that specific time-parameter, (b) computing the scaled control points by multiplying the fixed coordinates in each array by the corresponding weights, and (c) summing the appropriate four scaled control points for each vertex.

The AGRELs of BIPs obtained by the construction process described in Section 6 evolve with time, but remain parallel to themselves, as will be shown later. Furthermore, for non-convex control polyhedra, the BIPs faces may intersect one another, allowing for topological changes in the boundaries of deforming objects. Surplus faces may only lie inside the objects and are thus hidden from the viewer. Realtime animation is possible because the geometric intersection operation usually involved in face trimming are unnecessary in this case.

---

1 For simplicity, we focus our presentation on the cubic Bezier formulation, but all the techniques described in this paper equally apply to Bezier curves of other degrees and to other parametric polynomial curves, such as non-uniform B-splines, that are unit-sum linear combinations of control points.

2 The same control point may be used in the definition of several vertices.

* **See page C–527 for Figure 1.**

4. Bezier metamorphosis

Given four arbitrary control polyhedra, A, B, C, and D, the deforming model implementing the metamorphosis of A into D along a deformation controlled by the intermediate shapes of B and C is obtained by computing and animating BIP(A,B,C,D).

The metamorphosis is formulated as a Bezier curve in the space of all polyhedra. In this formulation, four arbitrary polyhedra play the role of control points and the vector addition is replaced by the Minkowski sum of solids. The Minkowski sum of two pointsets, A and B, is defined as $A \oplus B = \{a + b : a \in A, b \in B\}$. Many of its properties are studied in [9]. A linear Minkowski combination of two solids is illustrated in Fig. 2.* The above formulation unambiguously defines the deforming shape, S(t), for all values of the time-parameter, t, between 0 and 1. Note that S implements the desired metamorphosis, since it interpolates the first and the last control polyhedra, $S(0) = A$ and $S(1) = D$. However, in general, S does not interpolate the intermediate polyhedra, B and C, which are mainly used to control the deformation path.

The technique developed here is based on the following important observation: the AGRELs of BIP(A,B,C,D), computed as described below and evaluated at t, are contained in the closed pointset, S(t), and contain the bounding faces of S(t). Consequently, as suggested earlier, BIP(A,B,C,D) may be directly used for animating S.

Note that only the shapes and orientations of the control polyhedra influence the shape of S. Translations applied to the control polyhedra are reflected in the position of S, but not in its shape.

A comparison of the cubic metamorphosis with a linear one is illustrated in Fig. 3."

6. Construction

The control polyhedra, A, B, C, and D, are represented in boundary form by a standard bi-directional incidence graph [12] which defines each face by the set of its bounding edges and defines each edge by its two vertices.

The algorithm computes the BIP's AGRELs by: (a) generating a sufficient set of candidate faces, (b) testing the associations of these faces with vertices of the appropriate control polyhedra, and (c) storing valid associations as AGRELs.

The set of tentative faces is easily obtainable, because each AGREL of the BIP is a scaled version either of a face of one of the control polyhedra or of a parallelogram that is the Minkowski sum of two edges belonging to two different control polyhedra (see Fig. 4).

A sufficient set of candidate face-vertex associations is obtained by combining each candidate face originated from one or two control polyhedra with all combinations of vertices of the other control polyhedra. For example, a candidate face coming from a face Fa of A generates one or several AGRELs having for vertices the Bezier curves defined in terms of four control points. The first one of these control points is a reference to the corresponding vertex of Fa. The second, third, and respectively fourth references are identical for all the vertices of the AGREL and refer to a vertex of B, of C, and respectively of D.

Testing for valid association between a face and a vertex is performed using the outward normal N to the face and the tangents T, to the incident edges oriented away from the vertex. The association is valid if and only if the dot-product $N \bullet T_i < 0$ for all i. This test is derived from the following property of Minkowski sums: at every point P of the boundary of a linear Minkowski

* **See page C–527 for Figures 2 and 3.**

combination M of solids S_i there is a direction N such that N points away from M and there exist a set of points P_i, one on the boundaries of each Si, such that N points away from S_i at P_i for all i.

The following pseudo-code provides an outline for the organization of the algorithm:

```
Algorithm for computing BIP(A,B,C,D):
Reset Vs and Fs
Repeat 4 times using B, C, and D instead of A
  For each face F of A do
    N = outward normal for F
    Va = vertices of A bounding F
    Vb = vertices of B where N is pointing out of B
    Vc = vertices of C where N is pointing out of C
    Vd = vertices of D where N is pointing out of D
    For each combination of (b,c,d) in VbxVcxVd do
      Add face F to Fs
      For each vertex a of Va do
        V = (a,b,c,d)
        Add V to Vs
        Add to the vertices of F a reference to V
Repeat 6 times using all pairs instead of (A,B)
  For each pair of edges Ea of A and Eb of B do
    N = common orthogonal direction to Ea and Eb
    If N or -N points out of A at Ea and of B at Eb do
      N = -N if -N passed the above test
      Va = vertices of A bounding Ea
      Vb = vertices of B bounding Eb
      Vc = vertices of C where N is pointing out of C
      Vd = vertices of D where N is pointing out of D
      For each pair (c,d) in VcxVd do
        Add face F to Fs
        For each pair (a,b) in VaxVb do
          V = (a,b,c,d)
          Add V to Vs if
          Add to the vertices of F a reference to V
Return Vs and Fs
```

Let us assume that for each control polyhedron, the sum of the vertices, edges and faces is O(n). Then in the above algorithm, the complexity of generating the AGRELs consisting of a face of one polyhedron and combinations of vertices of the other polyhedra is $O(n^2)$, because the test involves comparing the normal to the chosen face with each vertex in all the other polyhedra. Updating the list Vs requires determining all combinations of the vertices found above and thus can be done in (On^4). time. The AGRELs obtained by summing edges of two polyhedra and vertices of the other can be found in $O(n^3)$ time (updating of Vs takes $O(n^4)$ time. Therefore, the overall complexity of the above algorithm is $O(n^4)$.

Note that when all control polyhedra are convex, each candidate face is used at most once and the topological line sweep technique of [6] may be adapted to reduce the computational complexity of the above algorithm.

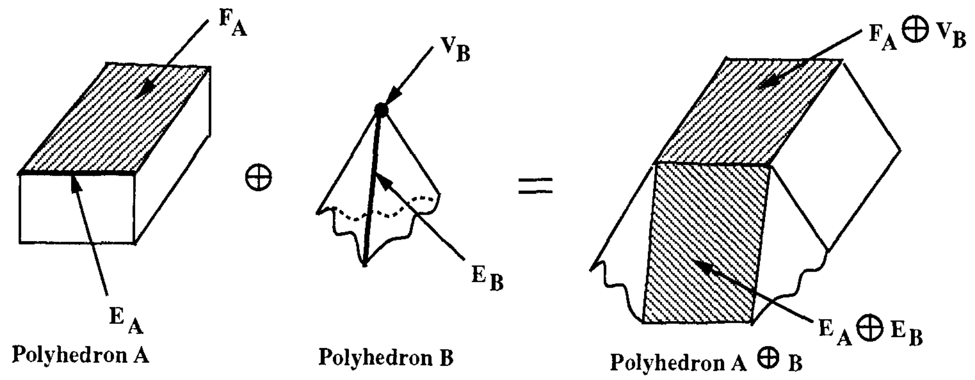

Figure 4: *Two types of candidate faces are shown for the Minkowski sum of two polyhedra: the top face of A matched with the top vertex of B and the Minkowski sum of an edge of A with an edge of B.*

From the above algorithm, we can now verify our earlier claim, that the AGRELs remain parallel to themselves during animation. For the AGRELs of both type described above, we note that at all points during the animation, the vertices of the face (from either a face of a control polyhedron, or the sum of two edges from two control polyhedra) are multiplied by a scalar which does not affect the face normal. This scaled face is subsequently translated by adding scaled vertices of the other polyhedra. Therefore the faces always remain parallel.

7. Implementation

The above algorithm was implemented and integrated in the LAMBADA experimental system developed at IBM Research. LAMBADA provides simple mechanisms for hierarchically applying motions and space deformations to static or parameterized objects. To create a BIP as a parametric object, the user selects and positions the four control polyhedra. LAMBADA computes the corresponding BIP. Animation is performed by changing the value of the time-parameter, either manually or automatically. The deformation may be synchronized and combined with rigid body motions.

8. Conclusion

We have presented a new technique for designing smooth deformations of solids. It provides control over the deformation in a manner analogous to the control of Bezier curves. The user designs a sequence of arbitrary polyhedra containing the initial and the final shape as well as any number of intermediate control polyhedra. A BIP representation is automatically computed and may be animated in realtime for simple objects. The BIP is defined in terms of AGRELs, which are animated faces whose vertices are moving along Bezier curves. The resulting BIPs are parameterized objects. Their deformations may be synchronized and combined with rigid body motions and other animated deformation techniques. The deformation represented by a BIP may be edited by using tranditional CAD tools to alter the shape or the orientation of the control polyhedra.

9. Acknowledgements

The authors would like to thank P. Bezier for his encouragements to pursue the non-linear extensions of the work on the linear metamorphosis presented at Eurographics'91 in Vienna. We are also deeply indebted to J. Mastrogiulio for integrating the BIP-generation algorithms in LAMBADA and for providing a user interface and the rendering mechanism.

References

[1] Boehm, W., Farin, G. and Kahmann, J., "A Survey of Curve and Surface Methods in CAGD," Computer Aided Geometric Design, vol. 1, no. 1, pp. 1-60, July 1984.

[2] Borrel, P. and Bechmann, D., "Deformations of n-dimensional Objects," in J. Rossignac and J. Turner, editors, ACM Symp. on Solid Modeling Foundations and CAD/CAM Applic., pp. 351-369, Austin, TX: ACM Press, Order number 429912, June 5-7 1991.

[3] Borrel, P. and Rappoport, A., "Simple constrained deformations (Scod) for geometric modeling and design," IBM Research Report, no. 17189, September 1991.

[4] Coquillart, S., "Extended Free-Form Deformation: a Sculpturing Tool for 3D Geometric Modeling," ACM Computer Graphics (Proc. SIGGRAPH), vol. 24, no. 4, pp. 187-196, 1990.

[5] Coquillart, S. and Jancene, P., "Animated Free-Form Deformation:," ACM Computer Graphics (Proc. SIGGRAPH), vol. 25, no. 4, pp. 23-26, July 1991.

[6] Guibas, L. and Seidel, R., "Computing convolutions by reciprocal search," Discrete and Computational Geometry, vol. 2, pp. 175-193, 1987.

[7] Kaul, A. and Rossignac, J., "Solid-Interpolating Deformations: Constructions and Animation of PIPs," Proceedings *of* EUROGRAPHICS 91, pp. 493-505, Vienna, September 1991.

[8] Sedeberg, T.W. and Parry, S.R., "Free-form deformation of solid geometric models," ACM *Computer* Graphics (Proc. SIGGRAPH), vol. 20, no. 4, pp. 151-160, Dallas, August 1986.

[9] Serra, J., Image Analysis and Mathematical Morphology, New York: Academic Press, 1982.

[10] Terzopoulos, D., Platt, J., Barr, A. and Fleischer, K., "Elastically Deformable Models," ACM *Computer* Graphics, vol. 21, pp. 205-214, 1987.

[11] Witkin, A. and Welch, W., "Fast Animation and Control of Nonrigid Structures," Computer Graphics (Proc. SIGGRAPH), vol. 24, no. 4, pp. 243-250, Dallas, August 1990.

[12] Woo, T.C. and Wolter, J.D., "A Constant Expected Time, Linear Storage Data Structure for Representing Three-Dimensional Objects," IEEE Trans. Systems, Man and Cybernetics, vol. SMC-14, no. 3, pp. 510-515, May/June 1984.