Interactive Shape Control and Rapid Display of A-patches

Chandrajit L. Bajaj
Jindon Chen
Guoliang Xu

Report Number:
94-080

https://docs.lib.purdue.edu/cstech/1179
INTERACTIVE SHAPE CONTROL AND
RAPID DISPLAY OF A-PATCHES
(EXTENDED ABSTRACT)

Chandrajit L. Bajaj
Jindon Chen
Guoliang Xu

CSD-TR-94-080
December 1994
Interactive Shape Control and Rapid Display of A-patches*
(Extended Abstract)

Chandrajit L. Bajaj    Jindon Chen    Guoliang Xu

Department of Computer Sciences
Purdue University
West Lafayette, Indiana

{bajaj,jdc,xuguo}@cs.purdue.edu

Abstract

A-patches[BCX94a] are implicit surfaces in Bernstein-Bézier(BB) form that are smooth and single-sheeted. In this paper, we present algorithms to utilize the extra degrees of freedom of these patches for local shape control. A ray shooting scheme is also given to rapidly generate polygonal approximations of A-patches for graphic display. A distributed implementation of this scheme gives near “real time” performance on rendering the A-patches to support interactive shape modification.

1 Introduction

The A-patch is a smooth and single-sheeted zero-contour patch of a trivariate polynomial in Bernstein-Bézier(BB) form defined within a tetrahedron[BCX94a], where the “A” stands for algebraic. Solutions to the problem of constructing a $C^1$ mesh of implicit algebraic patches based on an input polyhedron $\mathcal{P}$ have been given by [Dah89, BCX94a, BCX94b, DTS93, Guo91, Guo93, BI92]. While papers [BI92, Dah89, DTS93, Guo91, Guo93] provide heuristics based on monotonicity and least square approximation to circumvent the multiple sheeted and singularity problems of implicit patches, [BCX94a] introduces new sufficiency conditions for the BB form of trivariate polynomials within a tetrahedron, such that the zero contour of the polynomial is a single sheeted non-singular surface within the tetrahedron (the A-patch) and guarantees that its cubic-mesh complex for $\mathcal{P}$ is both nonsingular and single sheeted. Figure 1.1 show two of the typical 3-sided and 4-sided patches that are used in [BCX94a]. A simplicial hull is then constructed so that a pair of 4-sided patches (called edge patches) connect two neighboring 3-sided patches (called face patches) (see Figure 1.2).

The geometry of implicit surfaces has been proven to be more difficult to specify, interactively control, or polygonize than those of their parametric counterpart. Literature that concerns these issues includes [BIW93, BW90, Pra87, WH94].

*This research was supported in part by NSF grants CCR 92-22467, AFOSR grants F49620-93-10138, F49620-94-1-0080 and ONR grant N00014-94-1-0370.
In the A-patch scheme proposed in [BCX94a], several degrees of freedom remain to modify the shape of the resulting $C^1$ piecewise surface. In this paper, we utilize these weights for local shape control. We also present a rapid display algorithm based on ray shooting, to polygonize A-patches in a distributed fashion. This algorithm is based on properties (c) and (d) of A-patches in [BCX94a].

The rest of this paper is as follows. Section 2 discusses shape control. Section 3 describes the ray shooting display algorithm.

2 Shape Control

In the A-patch scheme proposed in [BCX94a], weights $a_{110}^1$, $a_{2001}^1$, $a_{3021}^1$ and $b_{002}^1$ of $W_1$ are adjustable within some ranges. This freedom, on the one hand, allows us to locally change the shape of the surface, while on the other hand, burdens us with extra work to remove bumpy defects.

2.1 Default weights

One commonly used method is to keep the surface patch close to a lower degree patch ([Baj92, DTS93]), which, in our case, is quadric patch. Specifically, we determine a quadric that first least-squares approximates the known weights of the cubic and then selects the unknown weights of the cubic from this quadric using a degree raising formula.

However, the least-squares optimization is subject to the $C^1$ and single-sheeted linear constraints and therefore is a typical non-linear programming problem with a quadratic objective function and linear constraints. Here, we employ a simple heuristic to obtain an approximation of the optimal solution should the solution of the unconstrained counterpart fall outside the constraint domain.

Prior methods as given in [BCX94a] failed to consider the shape of neighboring face patches beyond the fact that they share common normals. Such neglect could lead to unwanted variation in the intermediate joining edge patches. Our current scheme sets the ideal weights of the edge patches first. Then the weights of the face patches "honor" the choice of the edge patches by making sure the ideal edge patch weights are changed the least when the $C^1$ conditions are set.
Figure 1.2: Adjacent tetrahedra, cubic functions and control points (weights) for two non-convex adjacent face patches (a complete case)

Figure 2.3: Changing $a_{1110}$ on a face patch affecting other face patches. From left, $a_{1110} = -0.5, -1.0, 0.5$

Considering edge patch $W_1$ and $W_2$. Let $\ell$ be the intersection of the two tangent plane at $p_0$ and $p_3$. The weights around the two vertices are set by the interpolatory and normal conditions. We set the rest of the weights so that $g_1 = 0$ is actually a swept surface parallel to $\ell$ and the cross section curve approximate a quadric in least square sense. This can be done by a few basis change of the polynomials. Set $g_2$ to be the same as $g_1$.

By $C^1$ conditions, the edge patch weights propose values for the neighboring face patches. A face patch, however, take weighted averages of the proposed values from different edge patches around it and set them as default values. In taking the weighted average, smaller edge patch weighs more as for the same BB-represented surfaces, smaller tetrahedra yields larger curvature and larger curvature change for the same amount of change in the weights.

2.2 Interactive Shape Control

At a vertex, if the normal becomes longer then the surface becomes flatter around this vertex. The change of normal length is equivalent to a scaling of all the weights around the vertex by the same ratio. The direction of a normal also changes the surface shape around the vertex. However, one needs
Figure 2.4: Shape control by adjusting $a_{1002}$, $a_{0102}$, $a_{0012}$ and $a_{0003}$

Figure 2.5: Ray shooting A-patches

to ensure that the direction change does not violate the tangent containment constraints ([BCX94a]).

In a face tetrahedra $V$, $a_{0003}$, $a_{1002}$, $a_{0102}$ and $a_{0012}$ raise or lower the surface patch without altering its variation (see Figure 2.4); $a_{1110}$ alters the variation of the surface. Please note that by $C^1$ condition, $a_{1110}$ is related to $a_{0111}$, $a_{1011}$ and $a_{1101}$, other weights that could alter variation (see Figure 2.3). Hence the effect of changing $a_{1110}$ for the shape variation could be exaggerated or lessened, depending on the geometric relationship between the tetrahedra. Also, as $a_{1110}$ are related to neighboring patches in a linear equation, change of this weight propagates further into neighboring patches, while that of $a_{1002}$, $a_{0102}$ and $a_{0012}$ do not affect neighboring face patches and that of $a_{0003}$ affects only incident edge patches.

In general, a desirable modification involves collaboration of several adjustable weights rather than a single one. Hence an alternative way is to specify some additional data points in the tetrahedra, then approximate these points in the least-squares sense.

3 Rapid Display Scheme

Algorithms to generate polygonal approximations of a three-sided or four-sided patch are suggested by properties (d) and (e) of A-patches in [BCX94a].
For two based sharing 3-sided 4-patch \([p_1p_2p_3q_4]\) and \([p_1p_2p_3q_4]\), (see Figure 1.2) any point \(V\) on the surface defined in them is one to one mapping to a point \(\alpha^*\) on face \([p_1p_2p_3]\) as polyline \(p_4a^*q_4\) intersects with the union of two patches exactly once (see Figure 2.5). Hence the barycentric coordinate of points on \([p_1p_2p_3]\) can be used as parameterization of the union of the two 3-side apatches. Similarly, for two edge sharing four-sided 14-patch \([p^\prime_1p_2p_3q_4]\) and \([q^\prime_2p_2p_3q_4]\) (see Figure 1.2), the union of the two 4-sided A-patches can be parameterized into a quadrilateral domain \((s, t)\), where \((1 - s, s)\) is the barycentric coordinate of both a point \(\beta^* \in [p^\prime_1p_4]\) and a point \(\gamma^* \in [q^\prime_2q_4]\), \((1 - t, t)\) be the coordinate of a point \(\alpha^* \in [p_2p_3]\) as polyline \(\beta^*a^*\gamma^*\) intersection the union of apatches exactly once (see Figure 2.5). We call such a pair of 3-sided or 4-sided patches a double A-patch and call the parametrizations ray coordinate.

A simple polygonization algorithm can be described as follows, for both three-sided and four-sided patches. The algorithm works in an adaptive fashion. Beginning with some initial triangles, we keep subdividing them until the shape is desirable to some criterion.

Algorithm 1 Ray-shooting

1. Initialization: (i) 3-sided double apatch: Compute vertices \(A, B\) and \(C\) whose ray coordinates are \((0, 0), (1, 0)\) and \((0, 1)\), respectively. Enter \([ABC]\) as the first cell in the polygon list.
   (ii) 4-sided double apatch: Compute vertices \(A, B, C\) and \(D\) whose ray coordinates are \((0, 0), (0, 1), (1, 1)\) and \((1, 0)\), respectively. Enter \([ABC]\) and \([ACD]\) as the first two cells in the polygon list.

2. For each edge \([A = (s_0, t_0), B = (s_1, t_1)]\), if it is too long, Compute

   \[
   C = \begin{pmatrix} \frac{s}{t} \\ \frac{s_0}{t_0} \end{pmatrix} = (1 - \alpha) \begin{pmatrix} \frac{s_0}{t_0} \\ \frac{s_1}{t_1} \end{pmatrix} + \alpha \begin{pmatrix} \frac{s_1}{t_1} \end{pmatrix}
   \]

   for some \(\alpha \in [0, 1]\), weighted by the normals at \(A\) and \(B\). Replace \([AB]\) by \([AC]\) and \([CB]\). Exit if no edge is broken; go to 3 otherwise.

3. Triangulate every cell that has broken edges due to step 2. Go to 2.

Please note that a 4-sided double patch may have some special points where the surface passes through edge \(p_2p_3\). At those points, \(s\), the first component of its ray coordinate, is undefined. Or you may think of this situation as all the points with the same \(t\) values coinciding at one point.

We observe that, in a simplicial hull, a large portion of edge tetrahedra are thin compared to their neighboring face tetrahedra. If we ray-shoot each double patch separately, the polygonal mesh of an edge tetrahedron could be rather skew and dense compared to that of its neighboring face, which is not desirable for display or further processing based on the polygonal representation. To obtain a more uniform polygonal mesh, we instead ray-shoot a group of A-patches collectively, namely, a double face patch and the double edge patches around it. The algorithm is essentially the same.

The two algorithms can be speed up dramatically by distributing independent patch display computations to independent machines in a network cluster. Our distributed implementation was achieved in our client-server SplineX toolkit, part of our distributed modeling and visualization environment SHASTRA [AB94]. For the distributed display computation we need ensure that the boundary curves of two neighboring partitions are approximated by the same polyline. This is achieved by enforcing the same rayshooting criterion on both sides for a common boundary face.

References


