C code for modeling smooth free-form surfaces of arbitrary patch-layout with linearly-trimmed bicubic B-splines (NURBS)

Jörg Peters *

April 7, 1997

Abstract

The routine `Pcp2Nurb` is a key building block in overcoming topological constraints in the mathematical modeling of smooth surfaces. On input of a nine-point subnet of a *planar-cut polyhedron*, this routine outputs bicubic NURBS patches. The bicubic patches join smoothly to form a surface following the outlines of the planar-cut polyhedron in the manner traditional tensor-product splines follow the outline of their rectilinear control polyhedron. A rectilinear polyhedron is a special case of a planar-cut polyhedron. Conversely, a planar-cut polyhedron is a generalization of the tensor control structure. The generalization allows the modeling of free-form surfaces of arbitrary topology and patch-layout. A planar-cut polyhedron can be obtained by cutting an unrestricted control polyhedron that may contain non-planar, n-sided panels and m-valent vertices. The generic output of `Pcp2Nurb` is a four-sided piece of a regular C^1 surface. The piece integrates seamlessly with biquadratic tensor-product surfaces and can be stored, transmitted and rendered according to the standard representation used in industry. Sharp features can be retained in this representation by using geometrically redundant edges in the planar-cut polyhedron.

*Supported by NSF National Young Investigator grant 9457806-CCR

1 Introduction

Polyhedra can be smoothed into free-form surfaces using a variety of approaches such as rational blends, generalized subdivision or simplex splines (see e.g. [3], [1], [2]). A major criticism leveled at these techniques is that they are incompatible, i.e. cannot be represented exactly or efficiently in the dominant patch representation, tensor-product B-splines. Tensor-product B-splines serve under the pseudonym NURBS as a standard for storage, transmission and high-level rendering. However, NURBS impose a rectilinear, checkerboard surface-layout unsuitable for modeling arbitrarily laid out facets of general free-form surfaces.

The incompatibility criticism seems also to apply to *surface splines* proposed in [6], because surface splines employ three-sided surface pieces. Yet, the routine `Pcp2Nurb` described in this paper allows surface splines to meet the B-spline standard by efficiently and exactly representing collections of surface-spline pieces as linearly-trimmed, regularly parametrized NURBS patches. These NURBS patches inherit a number of proven mathematical and shape properties and yield a low-degree polynomial representation of tangent continuous free-form surfaces with arbitrary patch-layout. As a proof of compatibility, `Pcp2Nurb` outputs Open Inventor `NurbsSurface` [8] structures on input of a polyhedron. The NURBS surface can be inspected using a standard display tool, `ivview`. It is clear that the surface can equally well be represented, say as an IGES structure.

In conjuntion with `Pcp2Nurb` and its two driver routines, the following software tools are useful.

1. A tool (graphics library) that renders linearly-trimmed tensor-product B-splines, such as `ivview` on a Silicon Graphics workstation. (See Figure 2 for a description of a linear trimming, i.e. restriction of evaluation to a subdomain of a standard domain).

2. A partial modeling system able to represent polyhedra and apply planar cuts. (See Section 3.1 for an explanation of planar cuts).

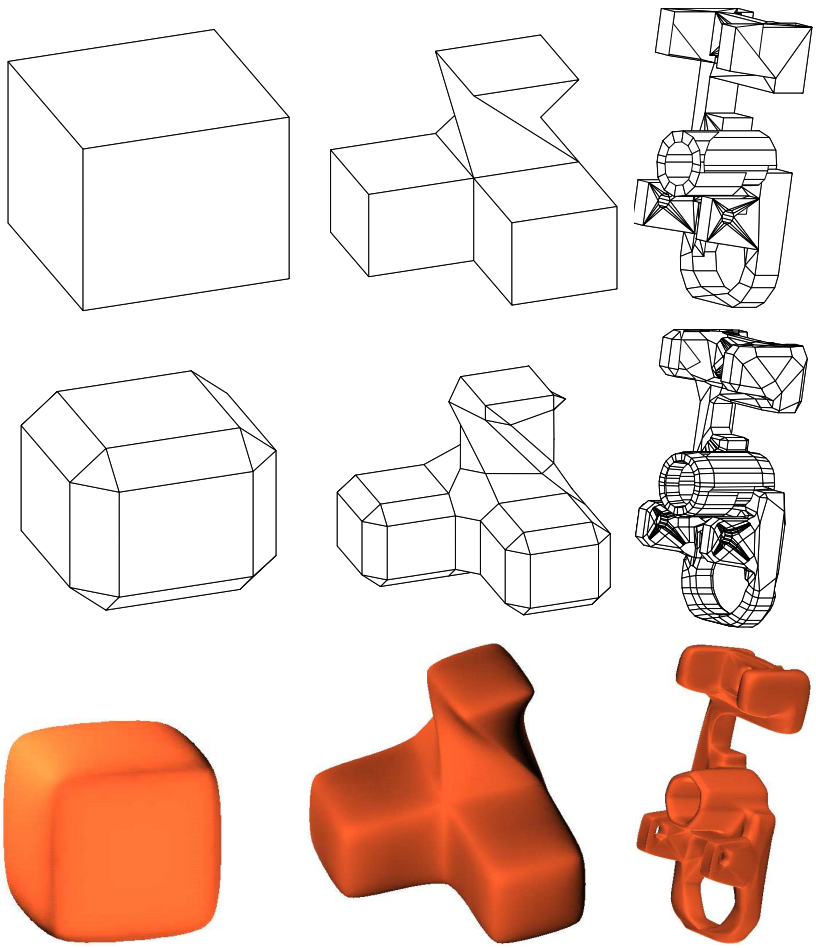

Figure 1: (Top) input polyhedron; (middle) planar-cut polyhedron; (bottom) NURBS surface.

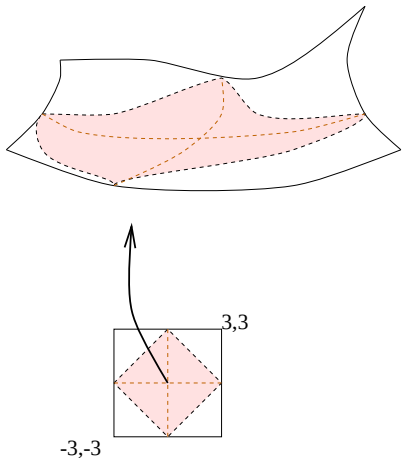

Figure 2: Trimmed NURBS patch with trim lines displayed in the domain.

2 Background

The principle underlying the algorithm and code is discussed in [7]: "Smoothing Polyhedra made Easy" where the coefficients of three-sided, cubic, C^1 connected patches are expressed as simple averages of a planar-cut polyhedron(see Section 3.1 for the definition of planar-cut polyhedron.) As a special case, the surface splines described in [6] always group four three-sided patches together as shown in Figure 2. By rotating and linearly clipping the domain as illustrated, each group can be represented as one linearly-trimmed, bicubic, tensor-product NURBS patch.

The increased flexibility provided by the internal second-order knot lines of the trimmed patches results in better surface parametrizations than bicubic or even biquartic Bernstein-Bézier patches (cf. Theorem 2 of [5]). In particular, this construction guarantees tangent plane continuity, the strong convex hull property, locality and affine invariance. The patches join seamlessly with tensor-product biquadratic patches obtained by interpreting nine points forming four quadrilaterals in the planar-cut polyhedron as a B-spline control net. The transition between the trimmed bicubic patches and the biquadratic patches is automatically tangent continuous (cf. [6], p 654).

3 Usage

Figure 1 illustrates the two stages of the algorithm for three objects of increasing complexity. A preprocessing step generates a planar-cut polyhedron from an arbitrary polyhedron, while the main step generates the spline coefficients from subnets of the planar-cut polyhedron.

3.1 Preprocessing: Generating the planar-cut polyhedron

The goal of the preprocessing step is to transform an arbitrary input polyhedron into a planar-cut polyhedron.

Definition 3.1 *A* planar-cut polyhedron *is a polyhedron with every interior vertex surrounded by four facets. The first and third facet are four-sided, the other two must be planar if they have more than four edges.*

This conforms with the intuitive notion of (edge and) corner cutting except that 4-sided facets need not be planar (cf. the twisted facets in Figure 1,middle). Any rectilinear control mesh is a particular planar-cut polyhedron.

There are many strategies for generating a planar-cut polyhedron. The most efficient strategy will depend on the particular class of surfaces modeled. A general algorithm for generating a planar-cut polyhedron can be found in [6] pp 649–650. The code provided with this paper is an independent module and does not require the data structures for maintaining polyhedra. Such data structures, e.g. the half-edge data structure, can be found in [4]. Also many commercially available modeling environments provide the necessary functionality for maintaining a planar-cut polyhedron.

When generating the planar-cut polyhedron, interpolation and curvature properties of the surface can be controlled. First, note that the depth of the cuts can be chosen to determine in a natural way, the sharpness of features. Variation of the extent of the cuts between 0% and 100% results in a continuous change of the distribution of curvature. In particular, as Figure 3 illustrates, **sharp features** can be produced by zero-extent cuts which amount to placing vertices or edges of the planar-cut polyhedron on top of one another. **Interpolation of points and normals** of the input polyhedron can be achieved without solving a global system of equations. The key

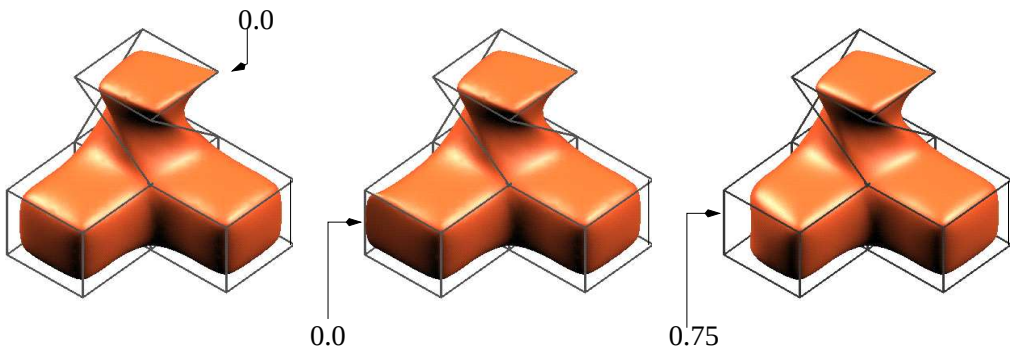

Figure 3: The effect of locally changing the ratio for planar cuts from a default setting of 0.35.

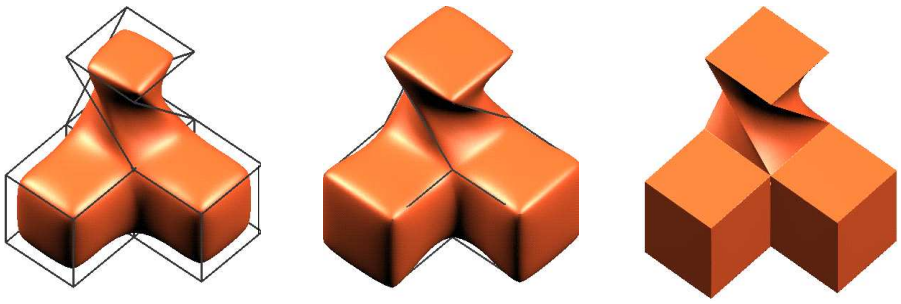

Figure 4: (left) The default case: smoothness and containment in the local convex hull of the polyhedron; (middle) smoothness and interpolation by moving the planes of the planar-cut polyhedron; (right) requiring the convex hull property and interpolation forces sharp edges in any representation. The surface-spline representation degenerates continuously to this extreme case.

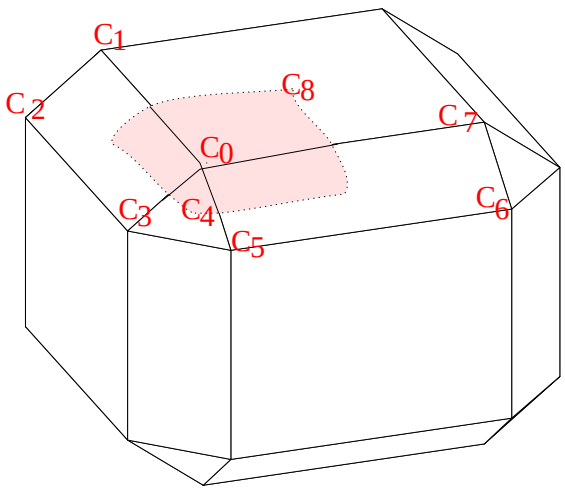

Figure 5: Vertices of the planar-cut polyhedron that determine one surface patch.

observation is that the surface interpolates face centroids and face normals of the planar-cut polyhedron. Hence, it suffices to place the centroids and normals of the planar-cut polyhedron so that the input points and normals are matched (see Figure 4(middle)).

An example of a planar-cut polyhedronis provided with the driver routine `nurb_iv1.c`. Consider the cube with vertex coordinates ± 2 as displayed in Figure 1 upper left. Cutting all corners at depth 0.5 yields $6 * 4$ new vertices with coordinates $(1, 1, 2)$, $(-1, 1, 2)$, etc.. The resulting planar-cut polyhedron is displayed in Figure 1 (middle-left) and Figure 5.

3.2 The routine `Pcp2Nurb`: Generating the spline coefficients from the planar-cut polyhedron

Each vertex of the planar-cut polyhedron gives rise to one linearly-trimmed bicubic NURBS patch (cf. Figure 5). To generate the coefficients of the patch anchored at a vertex C_0, a nine-point subnet of the planar-cut polyhedron

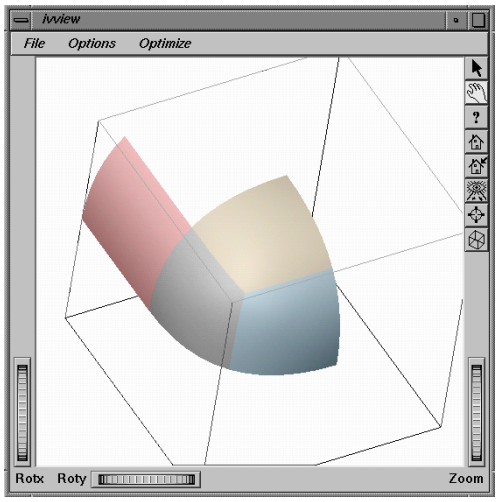

Figure 6: An `ivview` of the surface pieces generated by `Pcp2Nurb`. The three patches around the central point are bicubic. The fourth, attached patch is biquadratic.

with vertices $C_i, i = 0..8$ serves as input. As shown in Figure 5, the vertex sequences C_0, C_1, C_2, C_3 and C_0, C_5, C_6, C_7 each form a quadrilateral face of the planar-cut polyhedron. The vertices C_4 and C_8 are centroids of faces with edge counts or valencies recorded as n_0 and n_1. The nine vectors C_i and the two integers n_i are the input to the routine `Pcp2Nurb`. `Pcp2Nurb` returns the knot sequence and coefficients of a bicubic tensor-product spline patch in B-spline representation. Note the intended similarity of this nine-point subnet to the generic nine-point subnet defining a biquadratic tensor-product patch.

The example `nurb_iv1.c` (cf. Section 3.1) illustrates the usage of `Pcp2Nurb` by generating four `Inventor V2.0 ascii NurbsSurfaces` that can be rendered by `ivview` as shown in Figure 6. Three of these NurbsSurfaces are linearly-trimmed, the fourth is a regular biquadratic B-spline patch. This patch is added to demonstrate the smooth integration of both patch types. To illustrate the use of varying depth cuts when generating the planar-cut polyhedron, the driver routine accepts a command line parameter which varies the sharpness of the blend. A second driver routine, `nurb_iv2.c` generates the spline surface shown in the central column of Figure 1(middle).

References

[1] Catmull, E., and Clark, J. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Design 10* (1978), 350–355.

[2] Fong, P., and Seidel, H.-P. An implementation of triangular b-spline surface over arbitrary triangulations. *CAGD 10* (1993), 267–275.

[3] Loop, C., and DeRose, T. Generalized b-spline surfaces of arbitrary topology. *Computer Graphics 24*, 4 (1990), 347–356.

[4] Mantyläa, M. *An Introduction to Solid Modeling.* Computer Science Press, 1988.

[5] Peters, J. Biquartic C^1-surface splines over irregular meshes. *Computer Aided Design 27*, 12 (December 1995), 895–903.

[6] Peters, J. C^1-surface splines. *SIAM J. of Numer. Anal. 32*, 2 (1995), 645–666.

[7] Peters, J. Smoothing polyhedra made easy. *ACM Transactions on Graphics 14*, 2 (April 1995), 161–169.

[8] Wernecke, J. *The Inventor Mentor.* Addison Wesley, New York, 1993.