NURBS, Spline Surfaces and Blossoming in Two-dimensions

Lecture #03: 23 September 2002
Lecturer: Prof. Denis Zorin
Scribe: Zhihua Wang

In this lecture we generalize the polar form and blossoming and introduce NURBS(non-uniform rational B-spline), then we extend the discussion to 2D about surfaces.

1 NURBS

NURBS is an industry standard tool for the representation and design of geometry. It is the non-uniform and rational extension of B-splines.

We will introduce these properties one by one using blossoming.

1.1 Polar forms: review

A cubic **polar form** is a multi-affine [1] and symmetric function $f(u_1, u_2, u_3)$ such that:

- **multi-affine**:

$$f(\alpha_1 t_1 + \alpha_2 t_2, u_2, u_3) = \alpha_1 f(t_1, u_2, u_3) + \alpha_2 f(t_2, u_2, u_3), if \alpha_1 + \alpha_2 = 1$$

- **symmetric**:

  $f(u_1, u_2, u_3)$ does not depend on arguments' order.

As described in last lecture, there exists a unique polar form $f(u_1, u_2, u_3)$ for every polynomial function $f(t)$ such that $f(t) = f(t, t, t)$.

1.2 Blossoming principle

Blossoming is a way to compute points on the curve or new control points by linear recursive interpolation of control points.

Blossoming for Bezier curves and B-splines is described in lecture 2. Here we describe how non-uniform B-splines can be constructed by blossoming.

For uniform B-splines, control point P_i corresponds to $f(i, i+1, i+2)$. So it is determined by 3 consecutive uniform-spaced parameter values (also called knots). However, we can use the multi-affine properties to build the blossom pyramid even if the initial control points are

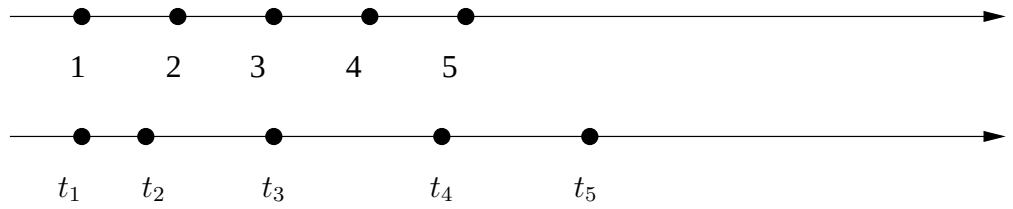


Figure 1: Comparison of parameter intervals between uniform and nonuniform B-splines.

associated with non-uniformly spaced knots: $t_0 < t_2 < t_3 ... < t_{n-1} < t_n$. See Figure 1. In fact some of the knots may coincide but we will not consider this case in detail.

The blossoming pyramid for non-uniform splines is shown in Figure 2.

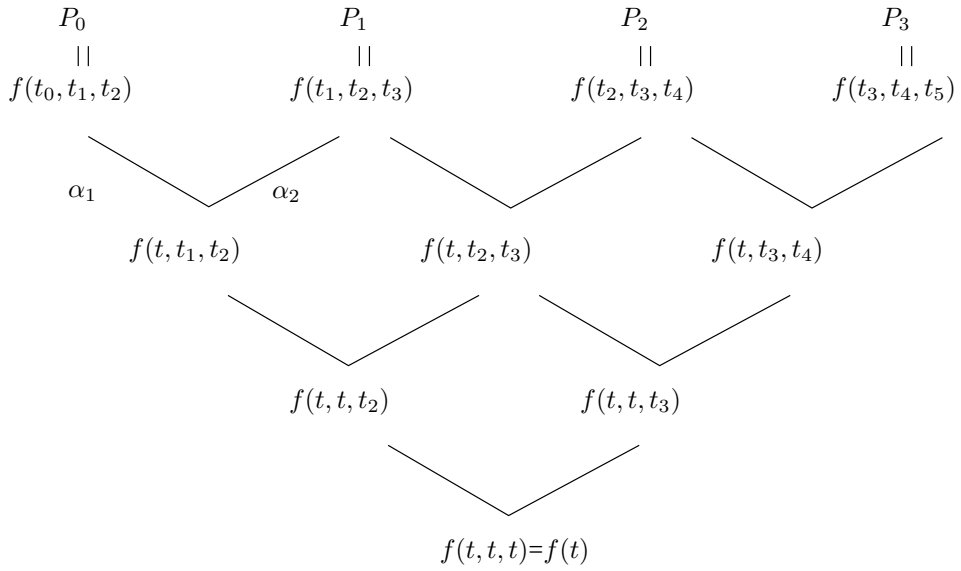


Figure 2: Blossoming for nonuniform B-splines.

The multiaffine property leads to the following equations for linear interpolation weights:

$$\begin{cases} \alpha_1 t_1 + \alpha_2 t_2 = t \\ \alpha_1 + \alpha_2 = 1 \end{cases}$$

from which we get:

$$\begin{cases} \alpha_1 = \frac{t_4 - t}{t_4 - t_1} \\ \alpha_2 = \frac{t - t_1}{t_4 - t_1} \end{cases}$$

Generally, for $0 \leq i < n$,

$$\begin{cases} \alpha_1 = \frac{t_{i+3} - t}{t_{i+3} - t_i} \\ \alpha_2 = \frac{t - t_i}{t_{i+3} - t_i} \end{cases}$$

We observe that the indices of base parameters differ by the degree of B-spline.

[1]"Multi-" here means that the property holds for every argument.

1.3 Usage of non-uniform splines

One of the most important advantages of non-uniform splines is possibility of knot insertion. Suppose we want to add a control point without changing the curve; in this way, we can add small features to a curve gradually while having complete control over the shape.

For a non-uniform B-spline, the problem can be turned into replacing the old sequence of parameters $t_0, t_1, ...t_{n-1}$ by new sequence of parameters $t_0, t_1, ...t_i, s, t_{i+1}, ...t_{n-1}$ such that: the $f(u_1, u_2, u_3)$ which was evaluated on $[t_i, t_{i+1}]$ is exactly the same as polynomials on $[t_i, s]$ and $[s, t_{i+1}]$ after insertion; see Figure 3. The user either defines s directly, or, more commonly clicks on the point of the curve and the value of s is computed by solving an equation $f(s, s, s) = p$, where p is the point selected by the user.

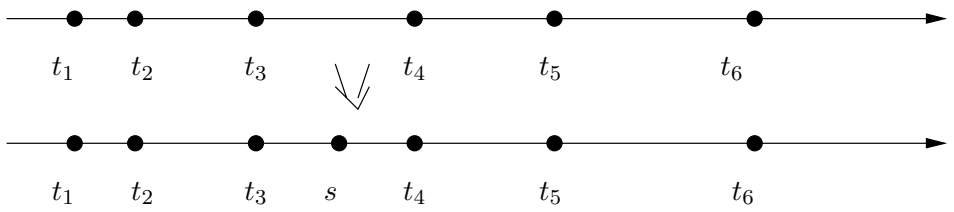


Figure 3: Adding new knot.

As a result of knot insertion, one new control point is added, and 2 old are replaced by new; the two new polynomial segments are defined by the following two groups of control points:

$$f(t_1, t_2, t_3), f(t_2, t_3, s), f(t_3, s, t_4), f(s, t_4, t_5)$$

and

$$f(t_2, t_3, s), f(t_3, s, f_4), f(s, t_4, t_5), f(t_4, t_5, t_6)$$

The new control points are obtained using blossoming as before. Each new control point is a linear combination of two old control points; see Figure 4

1.4 Rational splines

The "R" in NURBS stands for rational. Rational functions are ratios of two polynomials. The main reason for having rational splines is the need to represent quadrics, i.e. curves defined as solutions of quadratic equations, e.g. the circle $x^2 + y^2 = 1$. More generally, rational representation adds weights to the control points, so that some control points "attract" the curve more than others.

Now we discuss how rational functions help to represent quadrics. For example, an arc of a circle cannot be represented in parametric form $[x(t), y(t)]$ if polynomials are used for $x(t)$ and $y(t)$. Indeed, then we have $x(t)^2 + y(t)^2 = 1$, and solving for coefficients, for any degree of $x(t)$ and $y(t)$ we can easily get by induction that $x(t)$ and $y(t)$ should be constants.

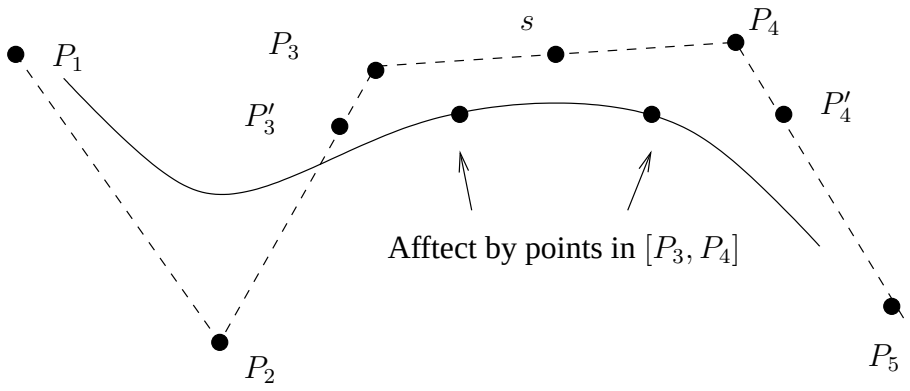


Figure 4: New control points.

Rational functions however allow us to represent a circular arc: Consider the most obvious parameterization of a circle $(\sin(t), \cos(t))$. This can be written as

$$\begin{cases} x(t) = \sin(t) = \frac{2s}{1+s^2} \\ y(t) = \cos(t) = \frac{1-s^2}{1+s^2} \end{cases}$$

, where we reparameterize using $s = \tan^2(\frac{t}{2})$.

The idea A point $P = [P^x, P^y]$ on the plane can be represented by $P' = [P^{x\prime}, P^{y\prime}, P^w]$ where P^w is the weight and $P^x P^w = P^{x\prime}$ and $P^y P^w = P^{y\prime}$.

Now we can compute a three-dimensional curve $\sum P_i' B(t-i)$ with P_i's as control points and represent the circle in the plane as:

$$\frac{\sum P_i B(t-i)}{\sum P_i^w B(t-i)}$$

.

2 Spline Surface

Spline Surfaces can be divided into two categories: tensor patches and triangular patches, according to the domain on which it is defined; see Figure 5

2.1 Tensor patch surfaces

Tensor patches surface is the simplest extension of curves. Consider two curve functions with interval domains V and U, respectively, $f(v)$, $g(u)$, and a surface formed by $h(v,u) = \sum j \sum i p_{ij} f(v) g(u)$. It is called a tensor product surface with domain $V \times U$.

For example, a tensor Bezier patch is represented as:

$$f(u,v) = \sum_{i=0}^{3} \sum_{j=0}^{3} P_{ij} B_{ij}(u,v)$$

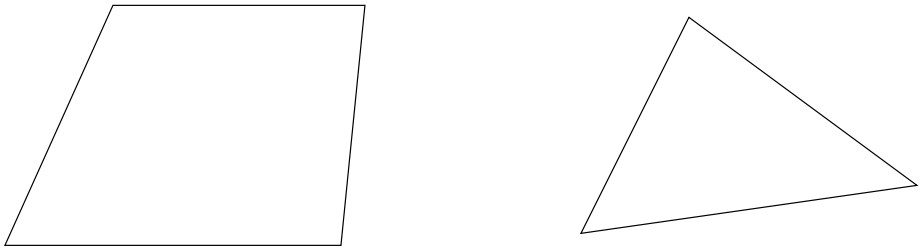


Figure 5: Tensor patches and triangular patches.

So there is 16 control points and 16 basis functions. We can do blossoming in a similar way. See Figure 6

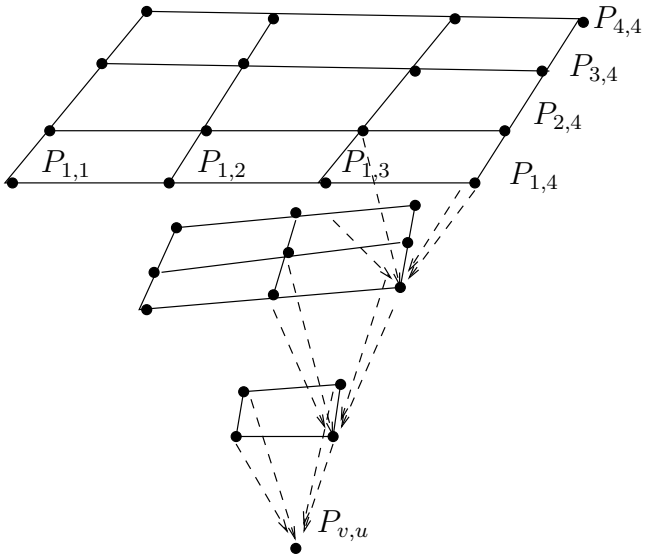


Figure 6: Blossoming of tensor Bezier patch.

Although combining Bezier curve segments into longer curves is easy (just make sure that the control points are aligned), it is more difficult to match Bezier patches in 3D. Tensor-product B-spline surface can have arbitrary number of control points and their polynomial patches are automatically jointed smoothly (for example for bicubic surfaces, (i.e. cubic in u and v) the patchees are joined with C^2 continuity. We use $(n+2)*(n+2)$ control points to get $n*n$ matched patches.

2.2 Triangular patches

For triangular polynomial patches the polar form can be represented as $f(t_1, t_2, t_3)$ where $t_i = (\beta_1, \beta_2, \beta_3)$ and $\beta_1 + \beta_2 + \beta_3 = 1$ for each i; see Figure 7

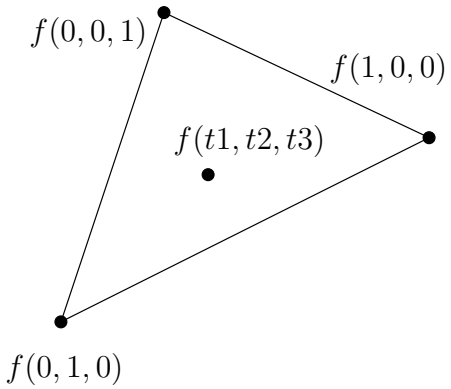


Figure 7: Interpolation of triangular patch.

One of the advantages of triangular is its lower degree for given smoothness. For example, to get C^2 continuity, for a tensor-product spline surface we must use bidegree 3 (total degree 6), while we only need total degree 4 for triangular patches.