NURBS: Non-Uniform Rational B-Splines
CS 525
©Denbigh Starkey

1. Background

The simplest B-splines have uniform knot vectors, and are not rational. NURBS (or sometimes NURBs)[1] are non-uniform rational curves, which means that they have rational weights associated with each control point. As we'll see, when the weights are all 1.0 then NURBS become the same at the non-rational B-splines that we studied earlier.

Rational polynomials were first described by Steven Coons in 1967, when he discussed how they could be used to represent conics, but most of the development of splines and NURBS wasn't done until Reisenfeld's PhD dissertation in 1973, based on the work of Cox, deBoor, and others.

One advantage of NURBS over the previous splines that we have studied is that they can represent additional objects. For example, without the weights it is impossible to accurately represent a circle, but with NURBS this can be done in a number of different ways. As a result they can also represent 3D conics like spheres. Another advantage is that they let the user locally reshape the generated curve without changing either the control points or the knot vector by changing point weights so that the curve moves towards (or away from) the affected points.

Basically a NURBS curve in 2D is the projection of a 3D curve onto the plane. Similarly, a NURBS curve in 3D is a projection of a curve from 4-space.

If you want to get more depth on NURBS (much more) than you'll be getting here, the best source in my view is *The NURBS Book*, by Les Piegl and Wayne Tiller, Springer-Verlag, 1995, where you'll get 646 pages on NURBS, including a lot of information on coding them efficiently. It is a nicely written book, and I recommend it.

In these notes I'll be looking at the basic definitions, and will use generating a circle as an example of a NURBS curve.

[1] The more common use of NURBS leads to a question of whether this is single or plural. I'll use it as a singular word when I have one of them and also as a plural word when I have more than one.

2. Definitions

A NURBS curve is defined using the Cox-deBoor blending functions with:

$$P(u) = \frac{\sum_{k=o}^{n} w_k p_k B_{k,d}(u)}{\sum_{k=0}^{n} w_k B_{k,d}(u)}$$

where the $B_{k,d}$ are the usual blending functions based on a knot vector, the p_k are the control points, and the w_k are weights assigned to each control point.

The blending functions for B-splines are non-negative for all u and sum to 1 (which gives the convex hull property for B-splines) and so if all of the weights are equal to 1 then this just reduces to the equation for regular B-splines. I.e., B-splines are a special case of NURBS.

It is also common to use *rational basis functions*, $R_{k,d}(u)$, instead of the blending functions by defining:

$$R_{k,d}(u) = \frac{w_k B_{k,d}(u)}{\sum_{i=0}^{n} w_i B_{i,d}(u)}$$

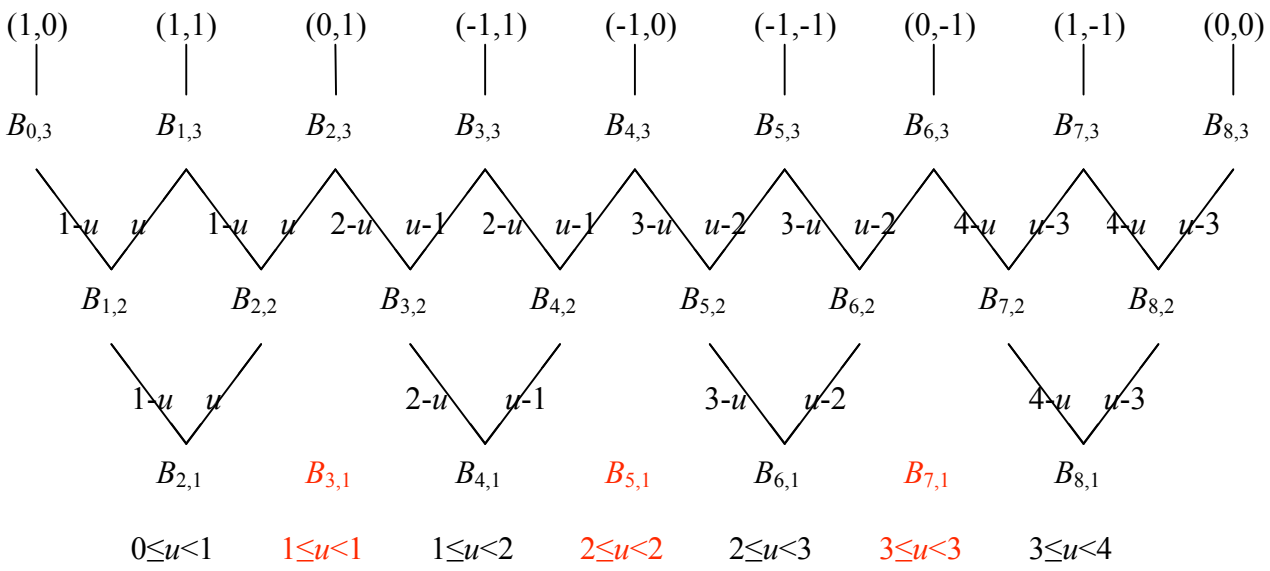
$$P(u) = \sum_{k=0}^{n} p_k R_{k,d}(u),$$

which is clearly equivalent to the original definition. I'll mainly use the first definition. One advantage of the use of the rational basis functions is that they make it clear that if $w_k \geq 0, \ \forall k$, then the convex hull property still holds.

3. Using NURBS to Define a Circle

Before getting into the circle code I'll look at a traditional B-spline, which I'll then modify by making it a rational spline to get a circle.

The curve has nine control points, $p_0 = p_8 = (1, 0)$, $p_1 = (1, 1)$, $p_2 = (0, 1)$, $p_3 = (-1, 1)$, $p_4 = (-1, 0)$, $p_5 = (-1, -1)$, $p_6 = (0, -1)$, and $p_7 = (1, -1)$, has the knot vector (0, 0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 4), and is quadratic ($d = 3$). This gives the blending function tree shown below:



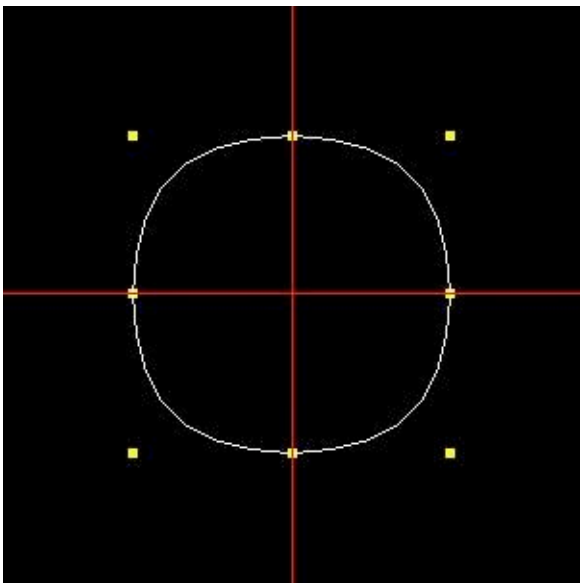
I've made some changes here. To simplify things I've converted from the $u_k \le u \le u_{k+1}$ form that I used before to the $u_k \le u < u_{k+1}$ form which is more traditional. This lets me ignore paths leading to the blending functions shown in red since they are now attached to empty ranges.

The tree gives the curve definition below:

$$P(u) = \begin{cases} (1-u)^2 p_0 + 2u(1-u)p_1 + u^2 p_2 & 0 \le u < 1 \\ (2-u)^2 p_2 + 2(u-1)(2-u)p_3 + (u-1)^2 p_4 & 1 \le u < 2 \\ (3-u)^2 p_4 + 2(u-2)(3-u)p_5 + (u-2)^2 p_6 & 2 \le 3 < 4 \\ (4-u)^2 p_6 + 2(u-3)(4-u)p_7 + (u-3)^2 p_8 & 3 \le u < 4 \end{cases}$$

The curve values at the five knot values are: $P(0) = p_0$, $P(1) = p_2$, $P(2) = p_4$, $P(3) = p_6$, and $P(4) = p_8$ (assuming that we define $P(4)$ based on the limit).

I.e., when we set up the curve this way it interpolates through the five values. The reason for showing this now is that I want to build a unit circle, which will need to pass through the five points (1, 0), (0, 1), (-1, 0), (0, -1), and (1, 0) to close, which are the five points p_0, p_2, p_4, p_6, and p_8. Unfortunately, although our current curve passes through these points, it isn't quite a circle, as the output below shows.



Obviously it resembles a circle, but sticks out too much towards the extra control points on the diagonals. We could get closer to the circle by adding more control points and forcing interpolation through them, but it would never be an accurate circle and as we add more points the computational costs increase.

Using NURBS, where we can assign smaller weights to the diagonal points, we can reduce the effect of these intermediate control points and so reduce this to a mathematically accurate circle. As I'll prove below, if we set the weights for the diagonal points (with odd subscripts) to $\frac{1}{\sqrt{2}}$ and leave the weights on the even subscript points at 1, then we'll get a circle.

The blending function definitions aren't affected by the weights, and so adding in the weights and using the NURBS formula changes the curve definition to:

$$P(u) = \begin{cases} \dfrac{(1-u)^2 p_0 + \sqrt{2}u(1-u)p_1 + u^2 p_2}{(1-u)^2 + \sqrt{2}u(1-u) + u^2} & 0 \le u < 1 \\ \dfrac{(2-u)^2 p_2 + \sqrt{2}(u-1)(2-u)p_3 + (u-1)^2 p_4}{(2-u)^2 + \sqrt{2}(u-1)(2-u) + (u-1)^2} & 1 \le u < 2 \\ \dfrac{(3-u)^2 p_4 + \sqrt{2}(u-2)(3-u)p_5 + (u-2)^2 p_6}{(3-u)^2 + \sqrt{2}(u-2)(3-u) + (u-2)^2} & 2 \le u < 3 \\ \dfrac{(4-u)^2 p_4 + \sqrt{2}(u-3)(4-u)p_6 + (u-3)^2 p_8}{(4-u)^2 + \sqrt{2}(u-3)(4-u) + (u-3)^2} & 3 \le u < 4 \end{cases}$$

Consider the range $0 \le u < 1$. Since $p_0 = (1, 0)$, $p_1 = (1, 1)$, and $p_2 = (0, 1)$, then for this range:

$$x(u) = \frac{(1-u)^2 + \sqrt{2}u(1-u)}{(1-u)^2 + \sqrt{2}u(1-u) + u^2}$$
$$y(u) = \frac{\sqrt{2}u(1-u) + u^2}{(1-u)^2 + \sqrt{2}u(1-u) + u^2}$$

Now for this to be on a unit circle, we need $x^2(u) + y^2(u) = 1$, for all u in the range. This gets messy with the formulae here, so I'll use $a = (1-u)^2$, etc., to get

$$x(u) = \frac{a+b}{a+b+c} \text{ and } y(u) = \frac{b+c}{a+b+c}.$$

$$x^2(u) + y^2(u) = \frac{a^2 + 2ab + 2b^2 + 2bc + c^2}{a^2 + b^2 + c^2 + 2ab + 2ac + 2bc}$$
$$= 1 + \frac{b^2 - 2ac}{a^2 + b^2 + c^2 + 2ab + 2ac + 2bc},$$

which defines a circle if and only if $b^2 = 2ac$. In this case $b = \sqrt{2}\,u(1-u)$, $a = (1-u)^2$, and $c = u^2$, so the condition is satisfied and throughout the first u range the points lie on a circle.

The other three cases work out similarly to this one.

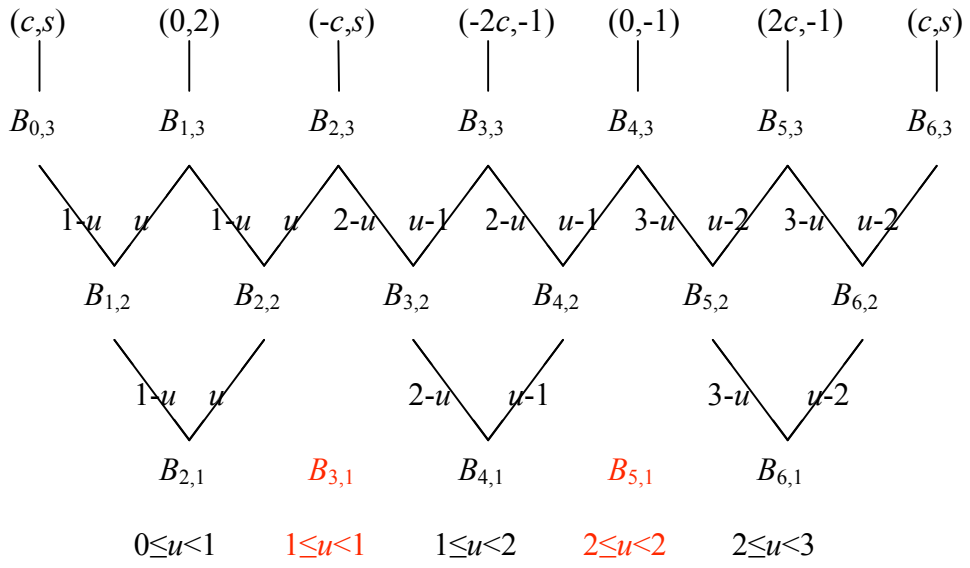
Since the four u ranges give quarter circles, which join at the p_{2i} values, this defines a circle.

Although the approach above is, in my view, the most intuitive way to define a circle, it isn't the simplest in terms of number of control points. Instead of nine control points, including the first repeated at the end, there are a number of ways to build a circle with seven control points, including the first repeated at the end. E.g., both of the following seven point quadratic NURBS define a circle.

$KV = (0, 0, 0, 1, 2, 2, 3, 4, 4, 4)$
$p = ((1, 0), (1, 1), (-1, 1), (-1, 0), (-1, -1), (1, -1), (1, 0))$
$w = (1, 0.5, 0.5, 1, 0.5, 0.5, 1)$

$KV = (0, 0, 0, 1, 1, 2, 2, 3, 3, 3)$
$p = ((c, s), (0, 2), (-c, s), (-2c, -1), (0, -1), (2c, -1), (c, s))$
where $c = \cos(30°) = \frac{\sqrt{3}}{2}$, and $s = \sin(30°) = \frac{1}{2}$.
$w = (1, 0.5, 1, 0.5, 1, 0.5, 1)$

I'll leave the first as an exercise and prove the second.



which, before including weights, gives:

$$P(u) = \begin{cases} (1-u)^2 p_0 + 2u(1-u)p_1 + u^2 p_2 & 0 \le u < 1 \\ (2-u)^2 p_2 + 2(u-1)(2-u)p_3 + (u-1)^2 p_4 & 1 \le u < 2 \\ (3-u)^2 p_4 + 2(u-2)(3-u)p_5 + (u-2)^2 p_6 & 2 \le u < 3 \end{cases}$$

Adding in the weights and substituting values for the control points gives, for the range $0 \le u \le 1$,

$$x(u) = \frac{(1-u)^2 c - u^2 c}{(1-u)^2 + u(1-u) + u^2}$$
$$y(u) = \frac{(1-u)^2 s + 2u(1-u) + u^2 s}{(1-u)^2 + u(1-u) + u^2}.$$

Replacing *c* and *s* and simplifying gives:

$$x(u) = \frac{(1-2u)\frac{\sqrt{3}}{2}}{u^2 - u + 1}$$
$$y(u) = \frac{-u^2 + u + \frac{1}{2}}{u^2 - u + 1}.$$

Finally

$$x^2(u) + y^2(u) = \frac{\frac{3}{4}(1 - 4u + 4u^2) + u^4 + u^2 + \frac{1}{4} - 2u^3 - u^2 + u}{u^4 - 2u^3 + 3u^2 - 2u + 1} = 1$$

Since (x(0), y(0)) = $(\frac{\sqrt{3}}{2}, \frac{1}{2})$ and (x(1), y(1)) = $(-\frac{\sqrt{3}}{2}, \frac{1}{2})$, this 120° arc lies on the unit circle. The proof of the other two arcs is similar.

Note that in the first example only two of the control points are interpolated, (1, 0) at the beginning and end, and (-1, 0) in the middle. The second example uses a equilateral triangular control system, with three of the control points interpolated (including one twice). As can be seen from the knot vectors, the first uses four 90° quarter-circle sweeps, and the second uses three 120° third-circle sweeps to define the circle.

4. Homogeneous Coordinates & Control Points at Infinity

A common approach to NURBS is to consider them as curves with an extra dimension, which are projected into the desired dimension. A point in a 3D curve will be represented as a four-tuple (x, y, z, w) where w is the weight associated with the control point. I.e., we are using a system which is very similar to homogeneous coordinates.

One big difference as compared to homogeneous coordinates, and a place where the last coordinate isn't really the weight, is that some NURBS systems will let the last coordinate be zero and treat it in a special way, by effectively saying that it gives us a control point at infinity, $(\frac{x}{0}, \frac{y}{0}, \frac{z}{0})$, which lets us pull of some neat effects.

The first thing to note is that if we go back to the original NURBS equations on page 3 of these notes, having a zero weight effectively means that the associated control point will be ignored since it is multiplied by zero. So we need a new definition for $P(u)$ if we are going to allow zero as a last coordinate, which is different from setting a weight to zero. The usual distinction that is given is that a zero weight nullifies that control point while a zero last coordinate defines a direction vector (x, y, z) as a control vector as compared to a control point.

Assuming that the p_i now all have the form (x_i, y_i, z_i, w_i), and that we define $q_i = (x_i, y_i, z_i)$ then the new curve definition is:

$$P(u) = \frac{\sum_{w_k \neq 0} w_k q_k B_{k,d}(u) + \sum_{w_k = 0} q_k B_{k,d}(u)}{\sum_{w_k \neq 0} w_k B_{k,d}(u)}$$

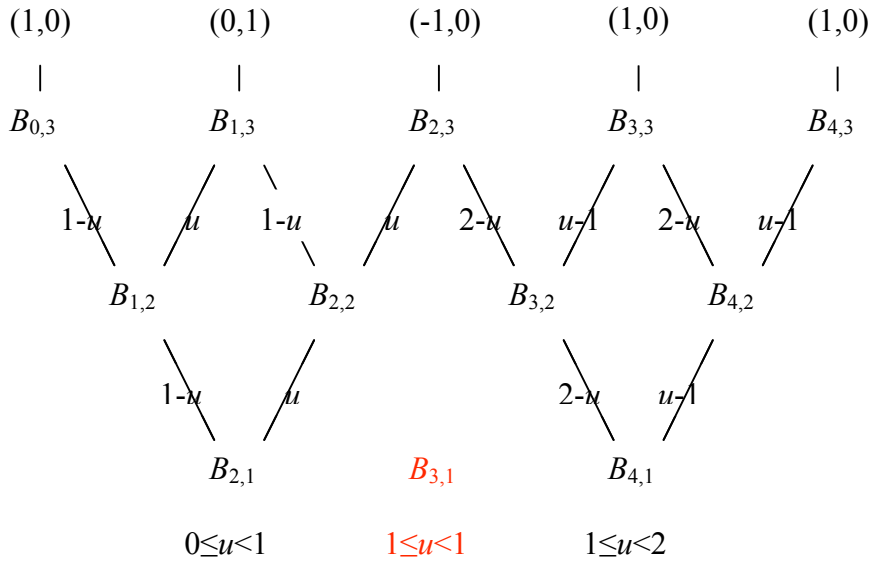
If none of the w_k are zero then this matches the original definition.

The question, of course, is how this has helped us. It turns out to be surprisingly powerful for defining some shapes, although there is an associated loss of intuition. E.g., we can use this, as I'll show below, to construct a circle using only four p values, where the first is repeated as a fifth point to close the circle. Computationally this gives a significant

advantage over the six-point and eight-point solutions. Consider the quadratic 2D curve defined by:

p = ((1, 0, 1), (0, 1, 0), (-1, 0, 1), (0, -1, 0), (1, 0, 1))
kv = (0, 0, 0, 1, 1, 2, 2, 2)

We need the blending functions, as usual, which are given by:



and this gives:

$$P(u) = \begin{cases} \dfrac{(1-u)^2(1,0) + u^2(-1,0) + 2u(1-u)(0,1)}{(1-u)^2 + u^2} & 0 \le u < 1 \\ \dfrac{(2-u)^2(-1,0) + (u-1)^2(1,0) + 2(u-1)(2-u)(0,-1)}{(2-u)^2 + (u-1)^2} & 1 \le u < 2 \end{cases}$$

To prove that this is a unit circle, I need to show that $(x(u))^2 + (y(u))^2 = 1$ for all u. Since the second range is (apart from a 180° rotation) just a parameter shift from the first, I only need to show this for the first range. For $0 \le u \le 1$ we get:

$$x(u) = \frac{(1-u)^2 - u^2}{(1-u)^2 + u^2} = \frac{1-2u}{1-2u+2u^2}$$
$$y(u) = \frac{2u(1-u)}{1-2u+2u^2}$$

and simple algebra shows that $(1-2u)^2 + 4u^2(1-u)^2 = (1-2u+2u^2)^2$.

An interesting feature of this approach is that the speed of the sweep is not uniform. By this I mean that even though $u = \frac{1}{2}$ will put us at the top of the circle at (0, 1), as expected, a value like $\frac{1}{4}$ or $\frac{5}{4}$ will give a point which is on the circle but is not on the diagonal. E.g., for $u = \frac{5}{4}$ substitution gives the point $(-\frac{4}{5}, -\frac{3}{5})$.

5. Constructing Other Conics

There are three types of conic arcs, parabolas, hyperbolas, and ellipses. All can be constructed in similar ways using quadratic NURBS. This is a relatively common need in CAD applications, but I won't get deeper into this topic. For details on how to generate any conic, including code, see the *The NURBS Book* by Piegl and Tiller that I referenced earlier.