Computer Graphics Notes-Parametric Curves and Surfaces

Parametric Representation

eg. $C(t) = (x(t), y(t))$

Continuity

Parametric Continuity

- If the first derivative of a curve is continuous, we say it has C^1 continuity.

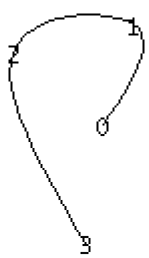
Geometric Continuity

- If the magnitude of the first derivative of a curve changes but the direction doesn't then, we say it has G^1 continuity.
- Curves need G2 continuity in order for a car to drive along them. (ie. not instantly change steering wheel angle at any points).

Control Points

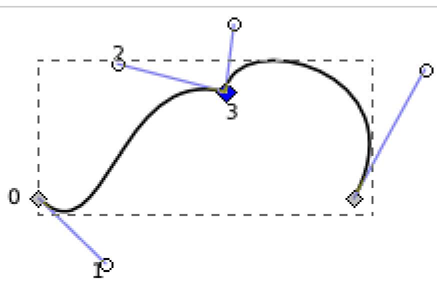
Control points allow us to shape/define curves visually. A curve will either interpolate or approximate control points.

Natural Cubic Splines



- Interpolate control points.
- A cubic curve between each pair of control points
- Four unknowns,
    -
        - interpolating the two control points gives two,
        - requiring that derivatives match at end of points of these curves gives the other two.
- Moving one control point changes the whole curve (ie. no local control over the shape of the curve)

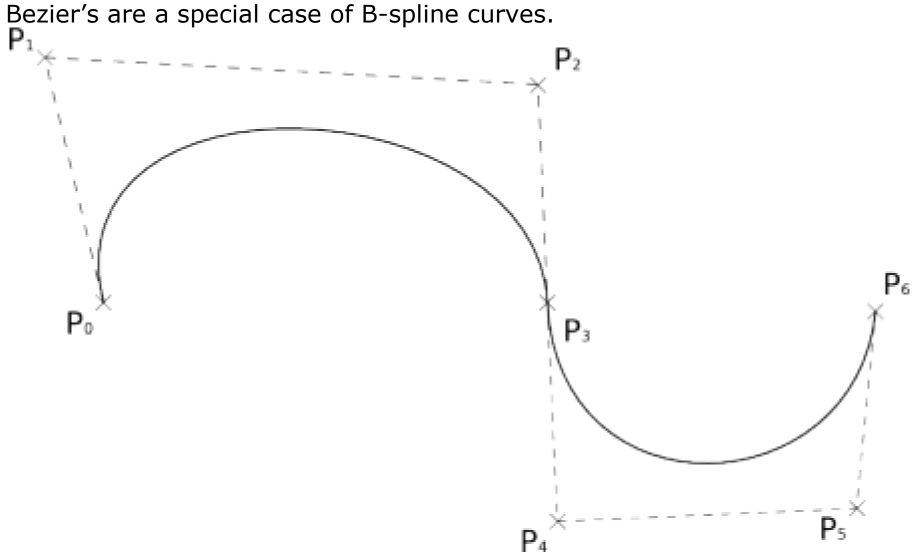
Bezier Curve



This Bezier curve shown has two segments, where each segment is defined by 4 control points. The curve interpolates two points and approximates the other two. The curve is defined by a Bernstein polynomial. In the diagram changing points 1 and 2 only affects that segment. Changing the corner points (0 and 3) each only affect the two segments that they boarder.

Some properties of Bezier Curves:

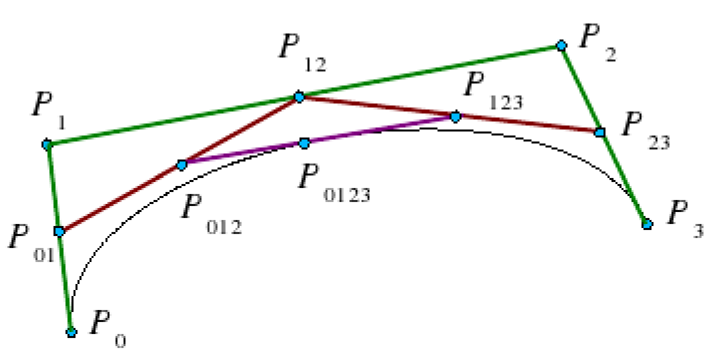
- Tangent Property. Tangent at point 0 is line 0 to 1, similarly for point 3.
- Convex Hull Property. The curve lies inside the convex hull of the control points. (The corollary of this is if the control points are colinear, the curve is a line.)
- They have affine invariance.
- Can't fluctuate more than their control polygon does.
- Bezier's are a special case of B-spline curves.



We can join two Bezier curves together to have C^1 continuity (where $B_1(P_0, P_1, P_2, P_3)$ and $B_2(P_0, P_1, P_2, P_3)$) if $P_3 - P_2 = P_4 - P_3$. That is P_2, P_3, and P_4 are colinear and P_3 is the midpoint of P_2 and P_4. To get G^1 continuity we just need P_2, P_3, and P_4 to be colinear. If we have G^1 continuity but not C^1 continuity the curve still won't have any corners but you will notice a "corner" if your using the curve for something else such as some cases in animation. [Also if the curve defined a road without G^1 continuity there would be points where you must change the steering wheel from one rotation to another instantly in order to stay on the path.]

De Casteljau Algorithm

De Casteljau Algorithm is a recursive method to evaluate points on a Bezier curve.



To calculate the point halfway on the curve, that is t = 0.5 using De Casteljau's algorithm we (as shown above) find the midpoints on each of the lines shown in green, then join the midpoints of the lines shown in red, then the midpoint of the resulting line is a point on the curve. To find the points for different values of *t*, just use that ratio to split the lines instead of using the midpoints. Also note that we have actually split the Bezier curve into two. The first defined by P_0, P_{01}, P_{012}, P_{0123} and the second by P_{0123}, P_{123}, P_{23}, P_3.

Curvature

The curvature of a circle is $\frac{1}{r}$.

The curvature of a curve at any point is the curvature of the osculating circle at that point. The osculating circle for a point on a curve is the circle that "just touches" the curve at that point. The curvature of a curve corresponds to the position of the steering wheel of a car going around that curve.

Uniform B Splines

Join with C2 continuity.

Any of the B splines don't interpolate any points, just approximate the control points.

Non-Uniform B Splines

Only invariant under affine transformations, not projective transformations.

Rational B Splines

Rational means that they are invariant under projective and affine transformations.

NURBS

Non-Uniform Rational B Splines

Can be used to model any of the conic sections (circle, ellipse, hyperbola)

Source: http://andrewharvey4.wordpress.com/2009/12/02/computer-graphics-notes/