L5 – Polynomial / Spline Curves

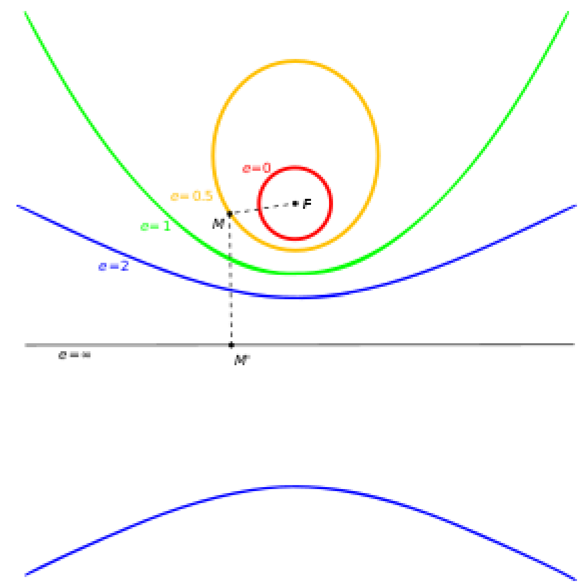
- Contents
  - Conic sections
  - Polynomial Curves
  - Hermite Curves
  - Bezier Curves
  - B-Splines
  - Non-Uniform Rational B-Splines (NURBS)
- Manipulation and Representation of Curves

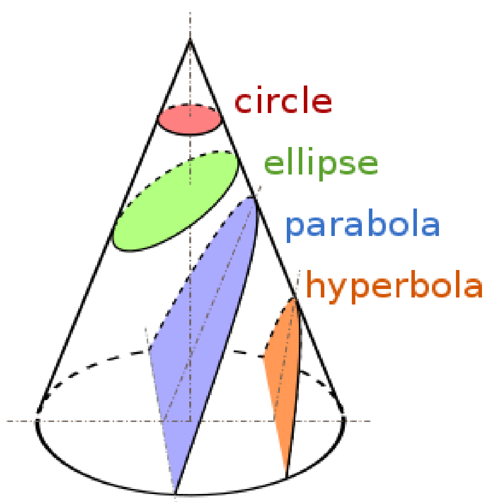
Types of Curve Equations

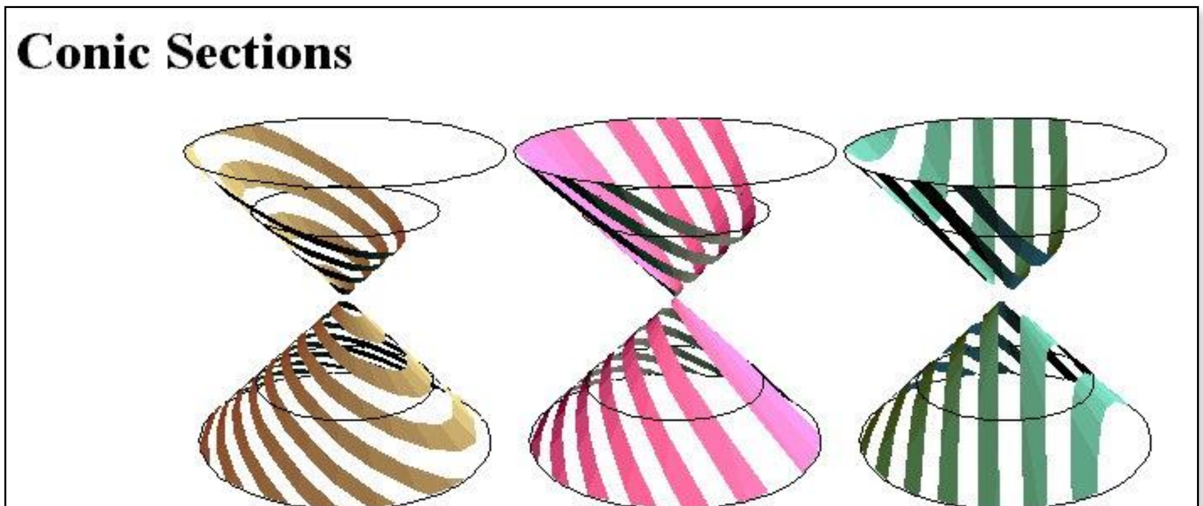
- Implicit: Describe a curve by a equation relating to (x,y,z) coordinates
  - Advantages: $x^2 + y^2 = R^2, \quad z = 0$
    - Compact; Easy to check if a point belongs to the curve
    - Easy to handle topological change
  - Disadvantages:
    - Difficult for curve evaluation
    - Difficult for partial curve definition
- Parametric: represent the (x,y,z) coordinates as a function of a single parameter (e.g., t as time for the trajectory of a moving path)
  - Advantages: $x = R\cos t, \quad y = R\sin t, \quad z = 0 \quad (0 \le t \le 2\pi)$
    - Easy for curve evaluation
    - Convenient for partial curve definition
    - Many others such as easy for manipulation, intersection.

Conic Sections

- Obtained by cutting a cone with a plane
  - Circle or Circular arc
  - Ellipse or Elliptic Arc
  - Parabola
  - Hyperbola
  - Arbitrary Conics (non-canonical form)



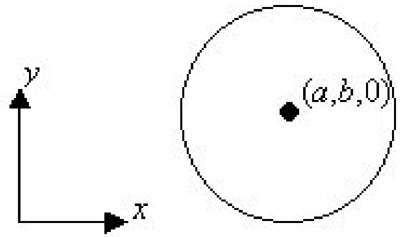




Intersections of parallel planes and a double cone, forming ellipses, parabolas, and hyperbolas respectively.

Circle, Ellipse, and Parabola

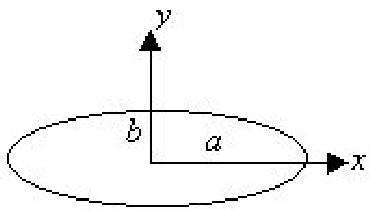
Circular arc:



$x = R\cos\theta + a$
$y = R\sin\theta + b$
$z = 0 \qquad (\theta 1 \le \theta \le \theta 2)$

The circular arc which lies on the xy plane with the center $(a, b, 0)$ and radius R. It becomes a full circle when $\theta 1 = 0$ and $\theta 2 = 2\pi$.

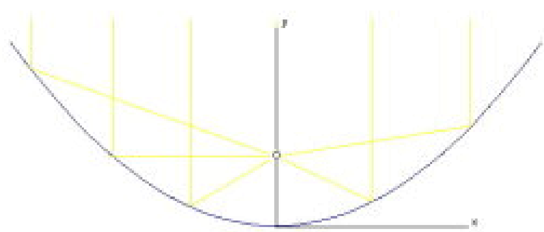
Elliptic arc:



$x = a\cos\theta$
$y = b\sin\theta$
$z = 0 \qquad (\theta 1 \le \theta \le \theta 2)$

The elliptic arc which lies on the xy plane with the center (0, 0, 0), x- and y-axis are the major and minor axes with length of a and b respectively. Becomes a full ellipse when $\theta 1 = 0$ and $\theta 2 = 2\pi$.

Parabola:

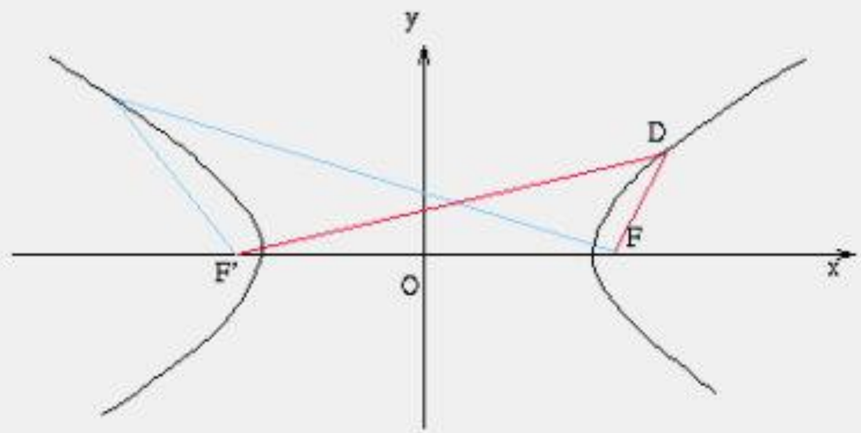


$x = u$
$y = cu^2$
$z = 0 \qquad (-\infty \le u \le \infty)$

A parabola symmetric with y-axis, c a constant. The two end points of a partial curve determines the range of parameter u.

Hyperbola

Take two points F and F' and a strictly positive value 2a.
The locus of all points D such that abs(|D,F| - |D,F'|) = 2a is a hyperbola.
Here abs(x) means absolute value of x.
We choose the line FF' as x-axis and the perpendicular bisector of the segment [F,F'] as y-axis.
We give F and F' resp. coordinates (c,0) and (-c,0).



Let $b^2 = c^2 - a^2$,

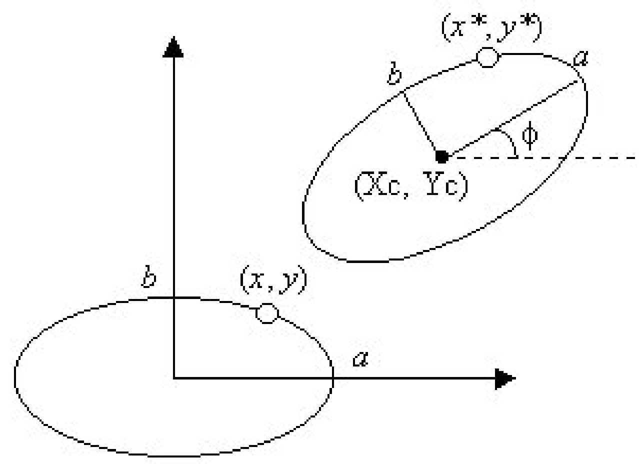
Implicit form: $$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$ $$z = 0$$

Parametric: $$x = a \bullet cosh(u)$$ $$y = b \bullet sinh(u)$$ $$z = 0 \qquad (u1 \le u \le u2)$$

$$(cosh(u) = (e^u + e^{-u})/2;\ sinh(u) = (e^u - e^{-u})/2)$$

Non-canonical conics

A general ellipse



The slanted ellipse can be obtained by rotating the reference ellipse at the origin ($x = a\cos\theta, y = b\sin\theta, z = 0$) by ϕ about the z-axis and then moved by Xc in x-direction and Yc in y-direction. After these two transformations, a point (x, y, 0) on the reference ellipse is moved to a new point (x*, y*, z*) on the new ellipse, related as:

$[x^*, y^*, z^*, 1]^T = \text{Trans}(Xc, Yc, 0) \bullet \text{Rot}(z, \phi) \bullet [x, y, 0, 1]^T$

$$= \begin{bmatrix} x\cos\phi - y\sin\phi + Xc \\ x\sin\phi + y\cos\phi + Yc \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} a\cos\theta\cos\phi - b\sin\theta\sin\phi + Xc \\ a\cos\theta\sin\phi + b\sin\theta\cos\phi + Yc \\ 0 \\ 1 \end{bmatrix}$$

Cubic Polynomial Curve

- Definition

$$P(u) = [x(u)\ y(u)\ z(u)]^T = a_0 + a_1u + a_2u^2 + a_3u^3 \qquad (0 \le u \le 1)$$

- Major Drawback:
  - a_0, a_1, a_2, a_3 are simply algebraic vector coefficients;
  - they do not reveal any relationship with the shape of the curve itself.
  - In other words, the change of the curve's shape cannot be intuitively anticipated from changes in their values.
- Data fitting? How?
  - Parameterization (Uniform, Chordal Length)
  - Least-Square Solution
- Why not quadric?

$$P(u) = a_0 + a_1u + a_2u^2 \qquad (0 \le u \le 1)$$

Cubic Polynomial Curve

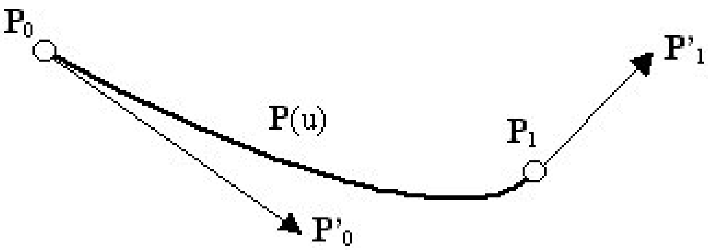
$$\mathbf{P}(u) = [x(u)\ y(u)\ z(u)]^T = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 + \mathbf{a}_3 u^3 \qquad (0 \le u \le 1)$$

An example: the parabola $\{x = cu^2, y = u, z = 0\}$

$$\underset{\mathbf{P}(u)}{\begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix}} = \underset{\mathbf{a}_0}{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}} + \underset{\mathbf{a}_1}{\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}} u + \underset{\mathbf{a}_2}{\begin{bmatrix} c \\ 0 \\ 0 \end{bmatrix}} u^2 + \underset{\mathbf{a}_3}{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}} u^3 \qquad (0 \le u \le 1)$$

Application:

Given two 3D points $\mathbf{P}_0$, $\mathbf{P}_1$, and their respective tangent vectors $\mathbf{P}'_0$ and $\mathbf{P}'_1$, find a cubic curve to interpolate them.



$$\mathbf{P}_0 = \mathbf{P}(0) = \mathbf{a}_0$$
$$\mathbf{P}_1 = \mathbf{P}(1) = \mathbf{a}_0 + \mathbf{a}_1 + \mathbf{a}_2 + \mathbf{a}_3$$
$$\mathbf{P}'_0 = \mathbf{P}'(0) = \mathbf{a}_1$$
$$\mathbf{P}'_1 = \mathbf{P}'(1) = \mathbf{a}_0 + 2\mathbf{a}_2 + 3\mathbf{a}_3$$

$\longrightarrow$

$$\mathbf{a}_0 = \mathbf{P}_0$$
$$\mathbf{a}_1 = \mathbf{P}'_0$$
$$\mathbf{a}_2 = -3\mathbf{P}_0 + 3\mathbf{P}_1 - 2\mathbf{P}'_0 - \mathbf{P}'_1$$
$$\mathbf{a}_3 = 2\mathbf{P}_0 - 2\mathbf{P}_1 + \mathbf{P}'_0 + \mathbf{P}'_1$$

$$\mathbf{P}(u) = \mathbf{P}_0 + \mathbf{P}'_0 u + (-3\mathbf{P}_0 + 3\mathbf{P}_1 - 2\mathbf{P}'_0 - \mathbf{P}'_1)u^2 + (2\mathbf{P}_0 - 2\mathbf{P}_1 + \mathbf{P}'_0 + \mathbf{P}'_1)u^3 \qquad (0 \le u \le 1)$$

Quadric Polynomial Curve

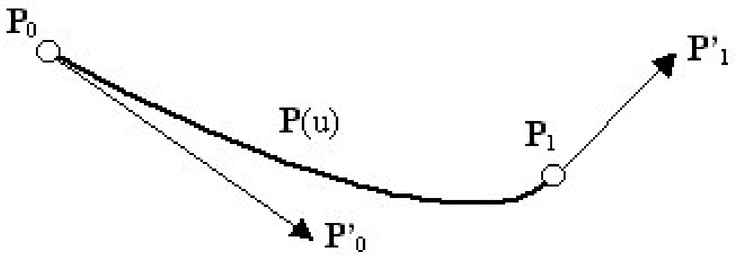
$$\mathbf{P}(u) = [x(u)\ y(u)\ z(u)]^T = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 \qquad (0 \le u \le 1)$$

An example: the parabola $\{x = cu^2, y = u, z = 0\}$

$$\underbrace{\begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix}}_{\mathbf{P}(u)} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{a}_0} + \underbrace{\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}}_{\mathbf{a}_1} u + \underbrace{\begin{bmatrix} c \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{a}_2} u^2 \qquad (0 \le u \le 1)$$

Why its use is limited:

Given two 3D points $\mathbf{P}_0$, $\mathbf{P}_1$, and their respective tangent vectors $\mathbf{P'}_0$ and $\mathbf{P'}_1$, find a cubic curve to interpolate them.



$\mathbf{P}_0 = \mathbf{P}(0) = \mathbf{a}_0$ → $\mathbf{a}_0 = \mathbf{P}_0$

$\mathbf{P'}_0 = \mathbf{P'}(0) = \mathbf{a}_1$ → $\mathbf{a}_1 = \mathbf{P'}_0$

$\mathbf{P}_1 = \mathbf{P}(1) = \mathbf{P}_0 + \mathbf{P'}_0 + \mathbf{a}_2$ → $\mathbf{a}_2 = \mathbf{P}_1 - \mathbf{P}_0 - \mathbf{P'}_0$

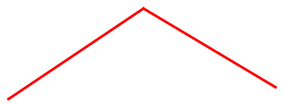
$\mathbf{P'}_1 = \mathbf{P'}(1) = \mathbf{P}_0 + 2\mathbf{a}_2$ → $\mathbf{a}_2 = (\mathbf{P'}_1 - \mathbf{P}_0)/2$

?

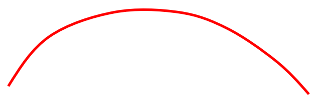
Continuity

Measures the degree of "smoothness" of a curve.

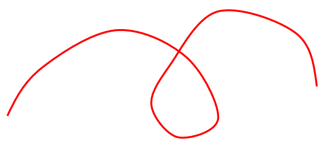
C^0 continuous – position continuous



C^1 continuous – slope continuous



C^2 continuous – curvature continuous



C^1 is the minimum acceptable curve for engineering design. Cubic polynomial is the lowest-degree polynomial that can guarantee the generation of C_0, C_1, and C_2 curves. Higher order curves tend to oscillate about control points.

That's reason why cubic polynomial is always used.

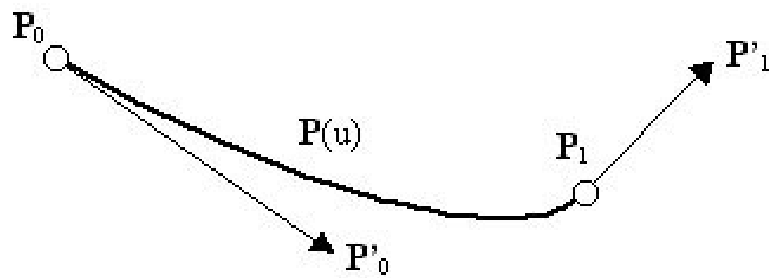
Hermite Curve

- Definition

$$P(u) = f0(u)P0 + f1(u)P1 + f2(u)P0' + f3(u)P1' \qquad (0 \leq u \leq 1)$$

- Benefits:
  - If the designer changes P0 or P1, he immediately knows what effect it will have on the shape – the end point moves.
  - Similarly, if he modifies P0' or P1', he knows at least the tangent direction at that end point will change accordingly.
- Deficiency
  - It is not easy and not intuitive to predict curve shape according to changes in magnitude of the tangents P0' or P1'.

Hermite Curve



The cubic curve that interpolates the two points $\mathbf{P}_0$, $\mathbf{P}_1$ and their tangents above is:

$$\mathbf{P}(u) = \mathbf{P}_0 + \mathbf{P}'_0 u + (-3\mathbf{P}_0 + 3\mathbf{P}_1 - 2\mathbf{P}'_0 - \mathbf{P}'_1)u^2 + (2\mathbf{P}_0 - 2\mathbf{P}_1 + \mathbf{P}'_0 + \mathbf{P}'_1)u^3 \quad (0 \le u \le 1)$$

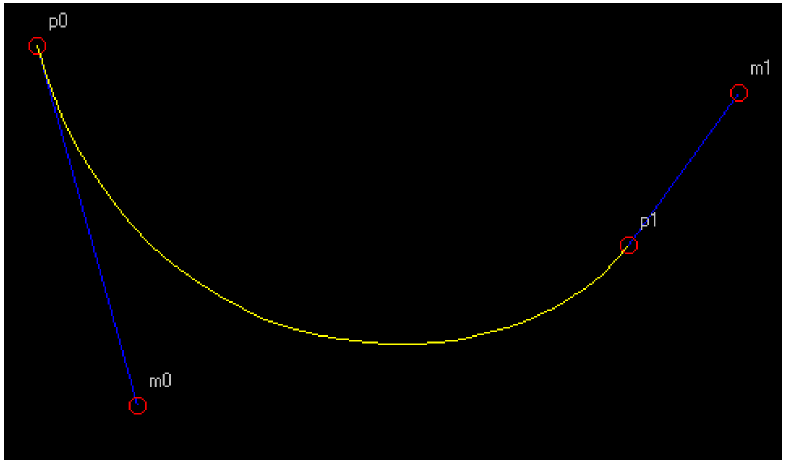
Rearrange the above equation around $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{P}'_0$, $\mathbf{P}'_1$, we have the Hermite curve:

$$\mathbf{P}(u) = \underbrace{(1-3u^2+2u^3)}_{f_0(u)} \mathbf{P}_0 + \underbrace{(3u^2-2u^3)}_{f_1(u)} \mathbf{P}_1 + \underbrace{(u-2u^2+u^3)}_{f_2(u)} \mathbf{P}'_0 + \underbrace{(-u^2+u^3)}_{f_3(u)} \mathbf{P}'_1$$

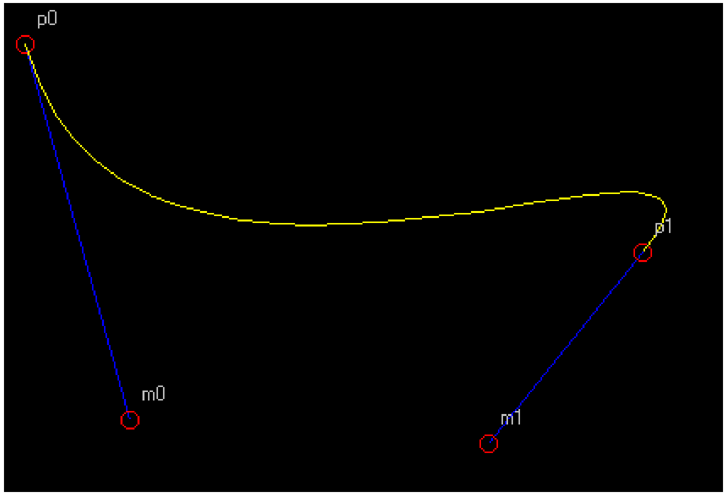
$$\boxed{\mathbf{P}(u) = f_0(u)\mathbf{P}_0 + f_1(u)\mathbf{P}_1 + f_2(u)\mathbf{P}'_0 + f_3(u)\mathbf{P}'_1} \qquad (0 \le u \le 1)$$

So now the curve is represented as linear combination of $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{P}'_0$, $\mathbf{P}'_1$, which are called ***geometric coefficients***. The four functions $f_0(u)$, $f_1(u)$, $f_2(u)$, $f_3(u)$ are called the ***blending functions***. That is, geometric coefficients are blended together by the blending functions.

Effect Tangents' Directions on a Hermite Curve



Change of tangent direction at point $\mathbf{p}_1$

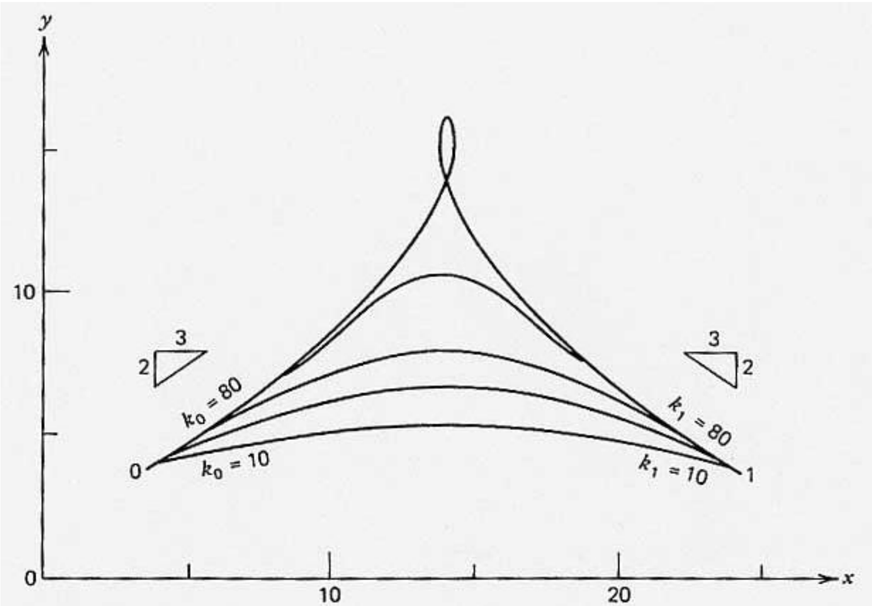


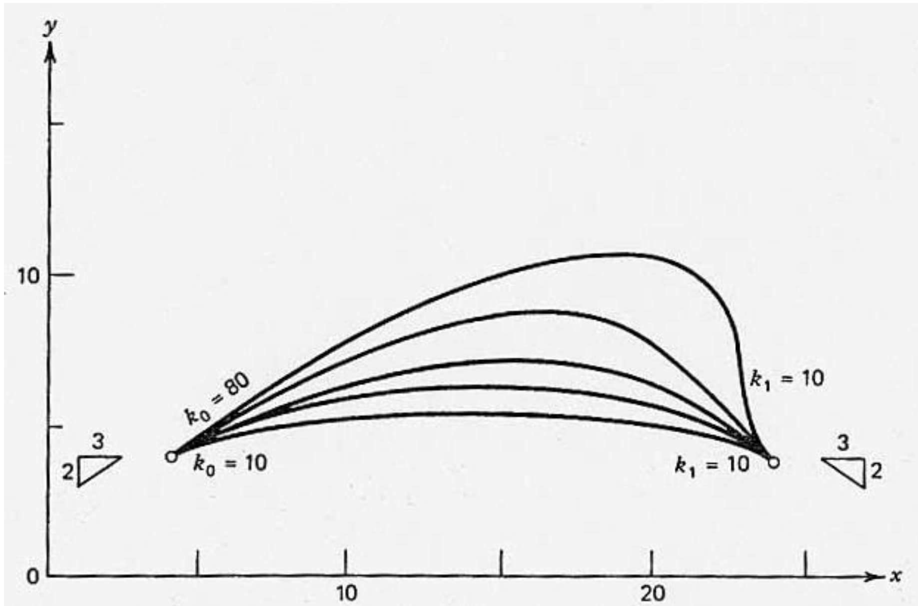
Effect Tangents' Magnitude on a Hermite Curve

k_0: magnitude of tangent at $\boldsymbol{p}_0$

k_1: magnitude of tangent at $\boldsymbol{p}_1$

The tangent directions at $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$ are fixed.





Bezier Curve

- Definition:

$$\mathbf{P}(u) = [x(u)\ y(u)\ z(u)]^T = f_0(u)\mathbf{P}_0 + f_1(u)\mathbf{P}_1 + \ldots + f_n(u)\mathbf{P}_n \qquad (0 \le u \le 1)$$

$$f_i(u) = B_{i,n}(u) = \binom{n}{i} u^i (1-u)^{n-i}$$

Where:

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

- How was Bezier curve discovered?
  1. Look at the desired properties
  2. Start from n = 1 to arbitrary

Road to the Discovery of Bezier Curve

Idea: Some kind of curve that can be controlled or manipulated by a polygon. Let this polygon have n+1 3D vertices $\{\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n\}$. The sought curve should be in the form:

$$\mathbf{P}(u) = [x(u)\ y(u)\ z(u)]^T = f_0(u)\mathbf{P}_0 + f_1(u)\mathbf{P}_1 + \ldots + f_n(u)\mathbf{P}_n \qquad (0 \le u \le 1)$$

where $f_0(u), f_1(u), \ldots, f_n(u)$ are the blending functions we are searching for.

Desired properties:

- The curve must passes through the first vertex $\mathbf{P}_0$ and last vertex $\mathbf{P}_n$ of the polygon. (Hermite property if n = 3.)

- The tangent vector at the start point $\mathbf{P}_0$ must have the same direction as the first segment of the polygon. Similarly, the last segment gives the direction of tangent at the last vertex $\mathbf{P}_n$. (Hermite property if n = 3.)

- The same curve is generated when the order of the vertices of the polygon is reversed. That is:

  $$f_0(u)\mathbf{P}_0 + f_1(u)\mathbf{P}_1 + \ldots + f_n(u)\mathbf{P}_n = f_0(u)\mathbf{P}_n + f_1(u)\mathbf{P}_{n-1} + \ldots + f_n(u)\mathbf{P}_0$$

  (Because to a designer, the curve should look exactly the same, as long as the polygon is the same, regardless in which direction the vertices are input.)

- The curve should be inside the convex hull of the polygon.

  (It gives the designer a safety zone. By controlling the convex hull of the control polygon, he knows the final curve will not go outside the convex hull.)

How would you discover Bezier Curve?

Step 1. Think of the simplest case, i.e., when n = 1, it has to be a line segment, and has to be in this form

$$\mathbf{P}(u) = (1-u)\mathbf{P}_0 + u\mathbf{P}_1 \qquad (0 \le u \le 1)$$

Step 2. Think of the desired convex hull property, using this classical affine theorem:

Partition of Unity

Given n+1 real and non-negative numbers $a_0, a_1, \ldots, a_n$, if $a_i \ge 0$ $(0 \le i \le n)$ and $\sum_{i=0}^{n} a_i = 1$, then the point $\mathbf{P} = a_0\mathbf{P}_0 + a_1\mathbf{P}_1 + \ldots + a_n\mathbf{P}_n$ is inside the convex hull of the n+1 3D points $\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n$.

Step 2. What about an arbitrary n?

$$((1-u) + u)^n = 1 \qquad \text{(for any u)}$$

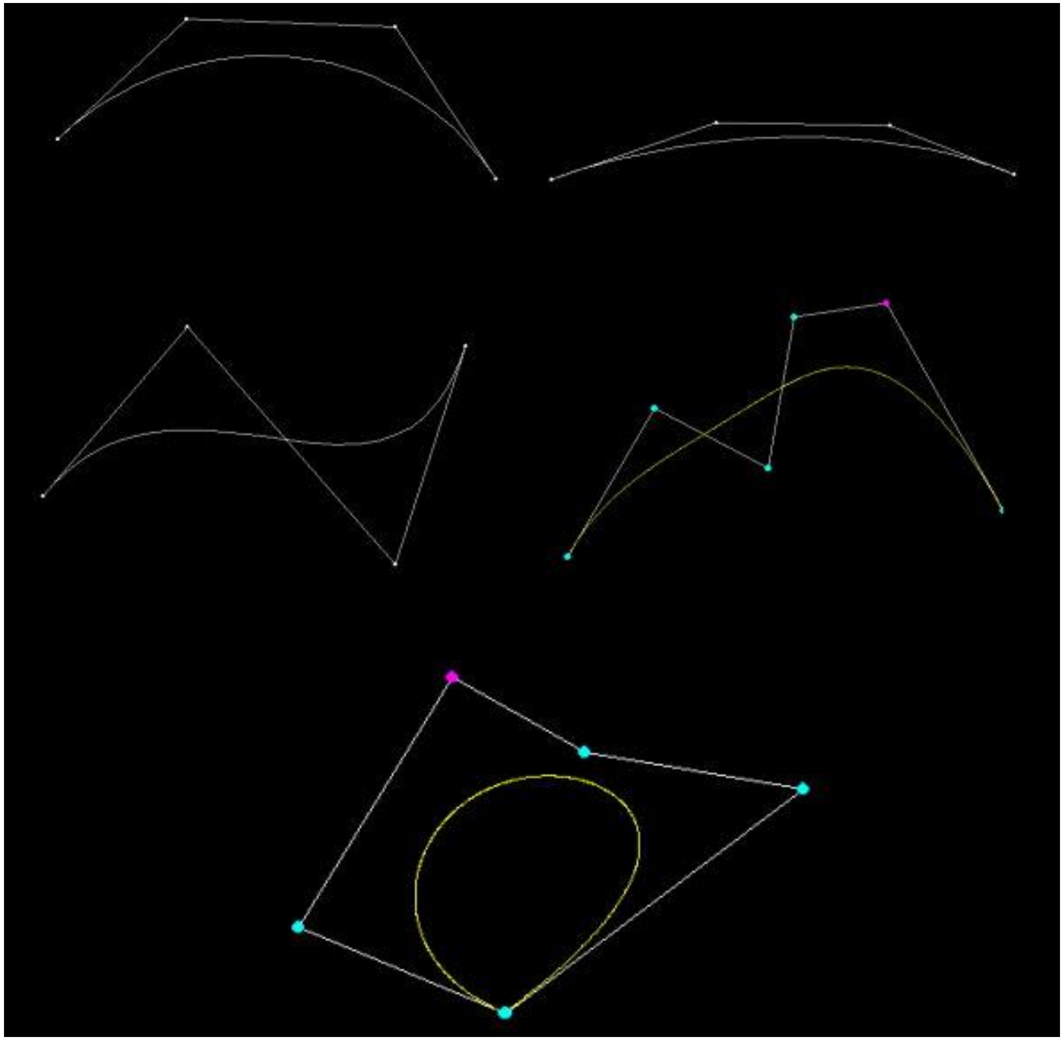
Expand into Binomial formula:

$$((1-u) + u)^n = \sum_{i=0}^{n} \binom{n}{i} u^i (1-u)^{n-i}$$

↑ $\mathbf{B}_{i,n}(\mathbf{u})$

The Bezier curve $\mathbf{P}(u) = B_{0,n}(u)\mathbf{P}_0 + B_{1,n}(u)\mathbf{P}_1 + \ldots + B_{n,n}(u)\mathbf{P}_n$ is inside the convex hull of $\{\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n\}$ for any u since $\sum_{i=0}^{n} B_{i,n}(u) = ((1-u) + u)^n = 1$ for any u.

Examples of Bezier Curves

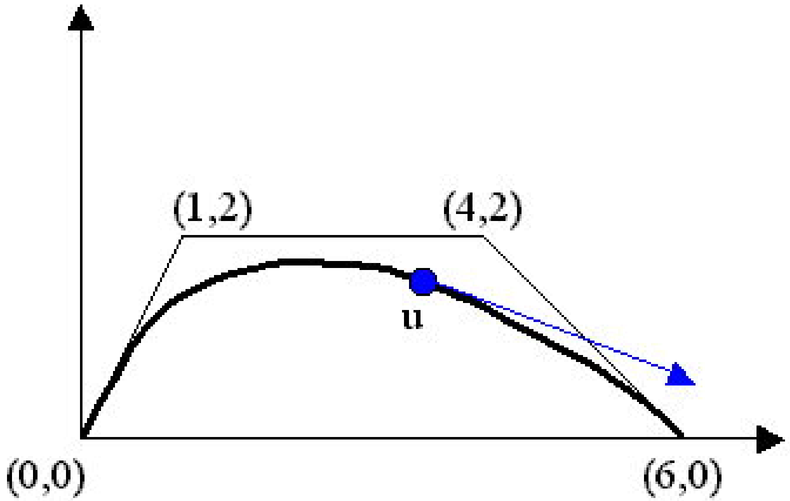


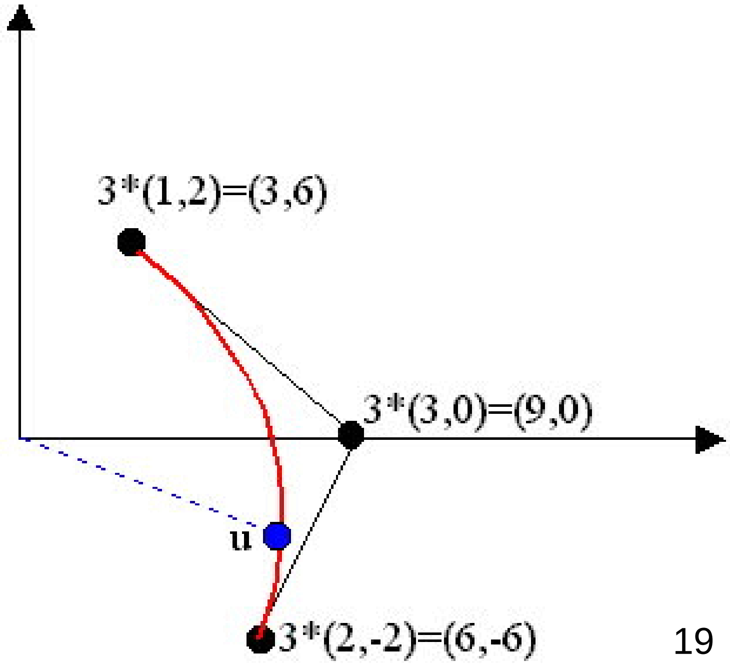
https://www.desmos.com/calculator/cahqdxeshd

Derivative of a Bezier curve

$$\frac{d\left(\sum_{i=0}^{n} B_{i,n}(u) P_i\right)}{du} = n \sum_{i=0}^{n-1} B_{i,n-1}(u)(P_{i+1} - P_i)$$

The right hand is a Bezier curve of degree (n-1), if you take $n(\mathbf{P}_{i+1} - \mathbf{P}_i)$ as a new control point. At u=0 and u=1, the two ends of the Bezier curve, their tangents are $n(\mathbf{P}_1 - \mathbf{P}_0)$ and $n(\mathbf{P}_n - \mathbf{P}_{n-1})$ respectively.





Naïve evaluation of a Bezier curve

To evaluate a point at $u = u_0$ on a Bezier curve of degree n

$$\mathbf{P}(u) = \sum_{i=0}^{n} \binom{n}{i} u^i (1-u)^{n-i} \mathbf{P}_i$$

Number of multiplications and divisions:

$\binom{n}{i} = \frac{n!}{i!(n-i)!}$: $(n-1)+(n-1)+1 = 2n - 1$

$u^i(1-u)^{n-i}$: $n - 1$

Total : $(3n - 2 + 2)*n = \mathbf{3n^2}$ (32 if for cubic n = 3)

⇨ (2n-1)+(n-1)+1+1

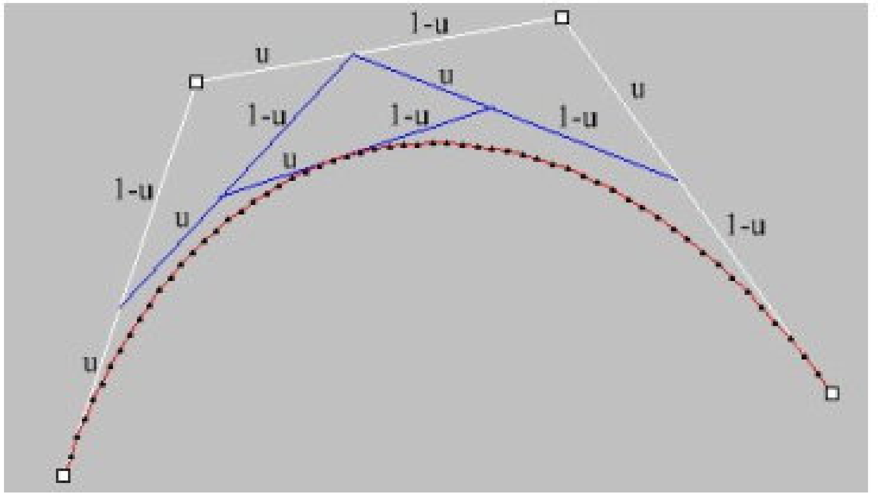
To draw the Bezier curve with 100 points

Number of miltiplications and divisions:

$3n^2*100 = \mathbf{300n^2}$ (3200 if for cubic n = 3)

The de Casteljau algorithm

An example of cubic Bezier curve (n = 3):

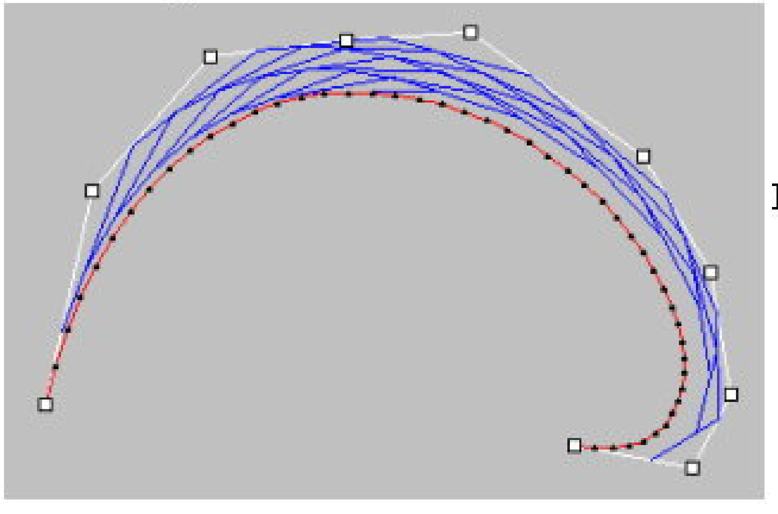


Number of multiplications of a single point:

1st iteration: 2*3 = 6
2nd iteration: 2*2 = 4
3rd iteration: 2*1 = 2
Total: 12 (vs. 32 of naïve way)

For 100 points: 12*100 = 1200 (vs. 3200 of naïve way)

For arbitrary n:



Number of multiplications of a single point:

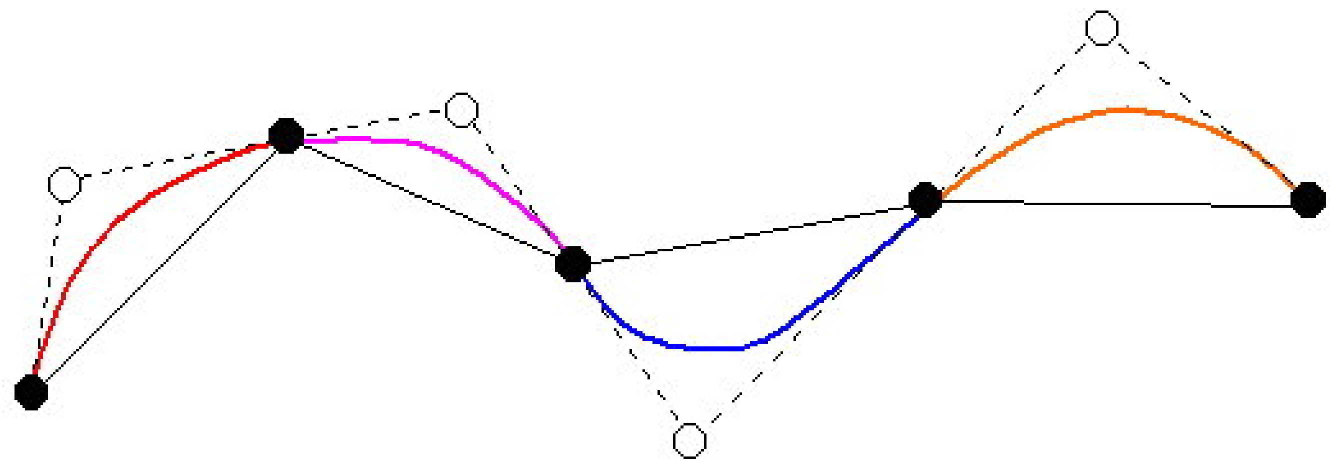
1st iteration: 2*n
2nd iteration: 2*(n-1)
3rd iteration: 2*(n-2)
.
nth iteration: 2
Total: $2*(1+2+\ldots+n) = n(n+1)$ (vs. $300n^2$ of naïve way)

Save 2/3 time compared to the naïve way of evaluation

Interpolation Using Multiple Bezier Curves

Smoothly interpolate an ordered list of points by many Bezier curves

To interpolate points ●, first construct temporary points ○, the two sets of points induce 4 quadric Bezier curves that meet smoothly at the points to be interpolated.



Problem: 1. How to determine these temporary points ○ ? By what criteria?
2. Is quadric Bezier enough?

Interpolation Using A Single Bezier Curve

Given points $Q_1, Q_2, \ldots, Q_m$, find a Bezier curve P(u) with control points $P_0, P_1, \ldots, P_n$ ($n < m$) so that P(u) interpolates all the Q_i.

1. $P_0 = Q_0$; $P_n = Q_m$.

2. $$u_j = \frac{\sum_{i=2}^{j} \|Q_i - Q_{i-1}\|}{\sum_{i=2}^{m} \|Q_i - Q_{i-1}\|} \quad ; \; j = 2,3,\ldots,m-1$$

3. $$e_j = \left\| Q_j - \sum_{i=0}^{n} B_{i,n}(u_j) P_i \right\| ; \; j = 2,3,\ldots,m-1$$

4. $$S_r(P_1, P_2, \cdots, P_{n-1}) = S_r(x_1, y_1, z_1, x_2, y_2, z_2, \cdots x_{n-1}, y_{n-1}, z_{n-1}) = \sum_{j=2}^{m-1} e_j^2$$

5. $$\frac{\partial S_r}{\partial x_i} = 0; \quad \frac{\partial S_r}{\partial y_i} = 0; \quad \frac{\partial S_r}{\partial z_i} = 0 \qquad i = 1,2,\ldots,n-1$$

These are 3 independent sets of (n-1) linear equations; solve them we get control points $P_0, P_1, \ldots, P_n$ whose corresponding Bezier best fit the points $Q_1, Q_2, \ldots, Q_m$ in the **least-squares** sense.

In-class exercise: verify it for the case of n = 3 and planar.

Drawbacks of Bezier curve

- High degree
  - The degree is determined by the number of control points which tend to be large for complicated curves. This causes oscillation as well as increases the computation burden.
- Non-local modification Property
  - When modifying a control point, the designer wants to see the shape change locally around the moved control point. In Bezier curve case, moving a control point affects the shape of the entire curve, and thus the portions on the curve not intended to change.
  - http://www.mat.dtu.dk/people/J.Gravesen/cagd/bez9-3.html
- Intractable linear equations
  - If we are interested in interpolation rather than just approximating a shape, we will have to compute control points from points on the curve. This leads to systems of linear equations, and solving such systems can be impractical when the degree of the curve is large.

Why and What To Do?

The culprit is the Bezier curve's blending functions $f_i(u) = B_{i,n}(u)$: because $B_{i,n}(u)$ is non-zero in the entire parameter domain [0,1], if the control point $\mathbf{P}_i$ moves, it will also affect the entire curve.

What we need is some blending function $f_i(u)$ such that:

1. It is non-zero over only a limited portion of the parameter interval of the entire curve, and this limited portion is different for each blending function. (Therefore, when $\mathbf{P}_i$ moves, it only affects a limited portion of the curve.)

2. It is ***independent*** of the number of control points n.

Answer: B-Splines

Definition of B-Spline Curve

Given $\{\mathbf{P}_0, \mathbf{P}_1, \ldots, \mathbf{P}_n\}$, a ***non-periodic*** and ***uniform*** B-spline curve is constructed according to the following four steps.

- Step 1. Select an integer k, called the ***order*** of the B-spline curve, usually k=4
- Step 2. Define (n + k +1) numbers t_0 to t_{n+k}, called ***knot values***:

$$t_i = \begin{cases} 0 & 0 \le i < k \\ i - k + 1 & k \le i \le n \\ n - k + 2 & n < i \le n + k \end{cases}$$

- Step 3. Compute the n+1 blending function $N_{i,k}(u)$ recursively:

$$N_{i,1}(u) = \begin{cases} 1 & t_i \le u \le t_{i+1} \\ 0 & otherwise \end{cases}$$

$$N_{i,k}(u) = \frac{(u - t_i) N_{i,k-1}(u)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - u) N_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}}$$

- Step 4. Put together:

$$\mathbf{P}(u) = \sum_{i=0}^{n} N_{i,k}(u)\, \mathbf{P}_i \qquad (t_{k-1} \le u \le t_{n+1})$$

Properties of B-Spline Curves

- The order k determines the degree of the blending functions $N_{i,k}(u)$: the highest degree p of u^p in $N_{i,k}(u)$ is $p = k-1$, independent of the number of control points n.

- ***All*** the properties enjoyed by Bezier curves. For example, the convex hull property ($\sum_{i=0}^{n} N_{i,k}(u) = 1$ for any u $(t_{k-1} \le u \le t_{n+1})$).

- The derivative of a B-spline is still a B-spline

$$\frac{d\left(\sum_{i=0}^{n} N_{i,k}(u) P_i\right)}{du} = \sum_{i=0}^{n-1} N_{i,k-1}(u) Q_i$$

where

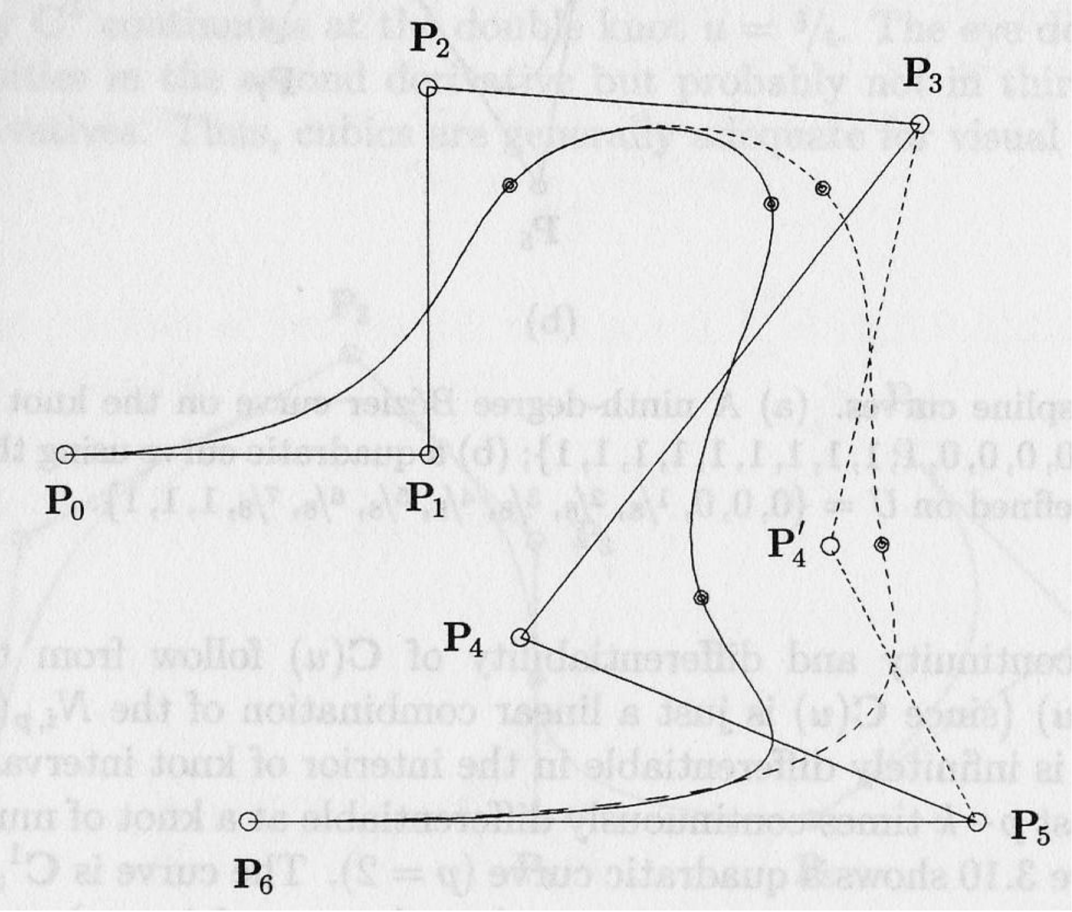
$$Q_i = (k-1)\frac{P_{i+1} - P_i}{t_{i+k} - t_{i+1}} .$$

- The most important feature of B-spline only

$N_{i,k}(u)$ is non-zero only in the interval $[t_i \le u \le t_{i+k})$.

Change of P_i therefore only affects that portion of curve.

Local Control on B-Spline Curves



Control point P_4 moves to a new position P_4'; only a portion of the original curve has changed.

Nonuniform Rational B-Splines (NURBS)

$$\mathbf{P}(u) = \frac{\sum_{i=0}^{n} w_i \, \mathbf{p}_i \, N_{i,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)}$$

for $0 \leq u < n\text{-}K+2$

with knots vector $\{t_0, t_1, \ldots, t_{n+K}\}$.

If weights $w_i = 1$ for all i, then it reduces to a standard B-Spline.

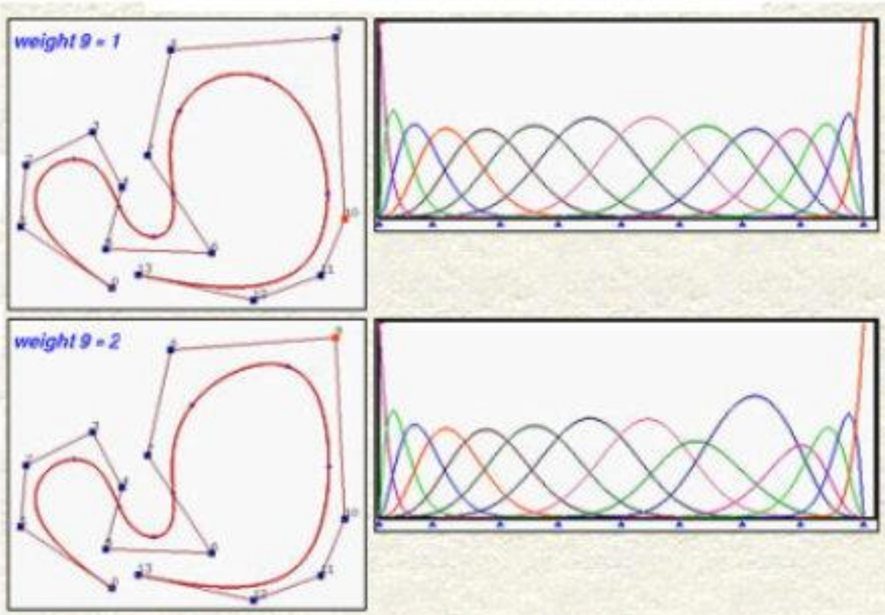
NURBS is the most general and popular representation.

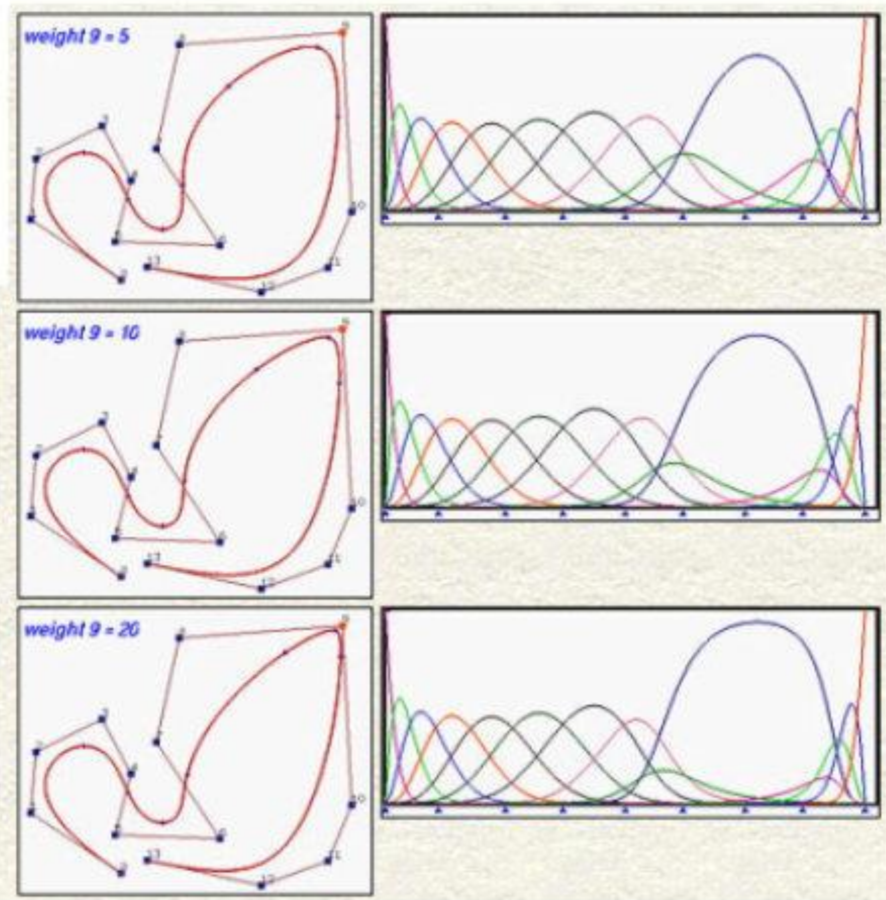
1. All Hermite, Bezier, and B-spline are special cases of NURBS.
2. It can represent exactly conics and other special curves.
3. The weights w_i add one more degree of freedom of **curve manipulation**.
4. It enjoys all the nice properties of standard (nonrational) B-splines (such as **affine transformation invariant and convex hull property**).

Effect of Weights on NURBS Curve

The coefficient before control point P_9 is:

$$C_9(u) = \frac{w_9 N_{9,k}(u)}{\sum_{i=0}^{n} w_i N_{i,k}(u)}$$





The larger w_9 is, the closer curve is pulled toward P_9. When w_9 is infinite, the curve passes through P_9; on the other hand, when w_9 is 0, P_9 does not effect the curve at all.

Partitioning of Unity Property on NURBS Basis Functions

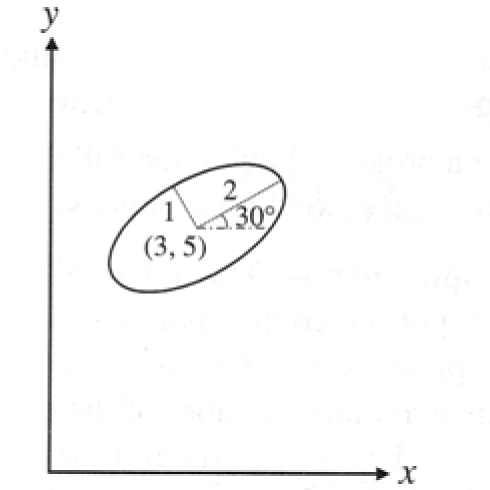
$$\mathbf{P}(u) = \frac{\sum_{i=0}^{n} w_i \, \mathbf{p}_i \, N_{i,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)}$$

$$= \sum_{i=0}^{n} \left(\frac{w_i N_{i,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} \right) \mathbf{p}_i$$

$$\frac{w_0 N_{0,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} + \frac{w_1 N_{1,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} + \ldots + \frac{w_n N_{n,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} = \frac{\sum_{i=0}^{n} w_i N_{i,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} = 1$$

http://fab.cba.mit.edu/classes/S62.12/docs/Piegl_NURBS.pdf

Take Home Questions (1)

1. 

Give the parametric curve representation of the ellipse shown left. (Hint: utilize transformations.)

2. Consider the Hermite curve defined in the plane with P(0) = (2,3), P(1) = (4, 0), P'(0) = (3,2), and P'(1) = (3, -4).
   a. Find a Bezier curve of degree 3 that represents this Hermite curve as exactly as possible, i.e., decide the four control points of the Bezier curve.
   b. Expand both of the curve equations into polynomial form and compare them. Are they identical?

Take Home Questions (2)

3. Determine a Bezier curve of degree 3 that approximates a quarter circle centered at (0,0). The two end points of the quarter circle are (1,0) and (0,1). Calculate the X and Y coordinates of the middle point of your Bezier curve and compare them with that of the quarter circle.

4. Answer the following questions for a non-periodical and uniform B-spline of order 3 defined by the control points P0, P1, P2, and P3:
   a. What are the knots values?
   b. There are two independent curves comprising this B-spline, each defined on the parameter range $u \in [0,1]$ and $u \in [1,2]$ respectively. Expand the B-spline curve equation to get the separate equations of these two curves.
   c. The two curve equations of b have different parameter u-ranges, i.e., for the first curve $C_1(u)$, its parameter u-range is [0, 1], for the 2nd curve $C_2(u)$, it is [1,2]. Please do:

      (1) Show that curve $C_1(u)$ is a Bezier curve. What are its control points?

      (2) Let $s = u - 1$. Show that $C_2(s+1)$: $s \in [0,1]$ is also a Bezier curve. What are its control points?