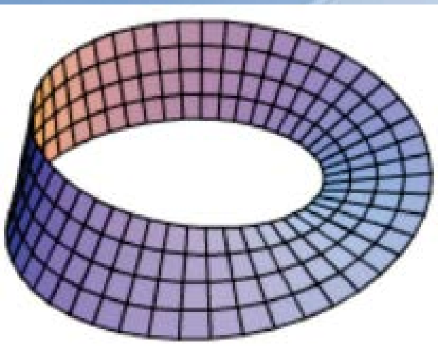
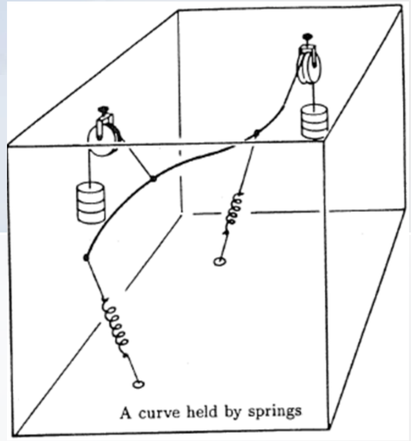
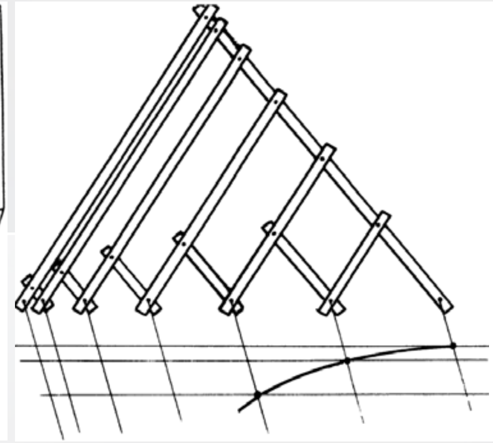
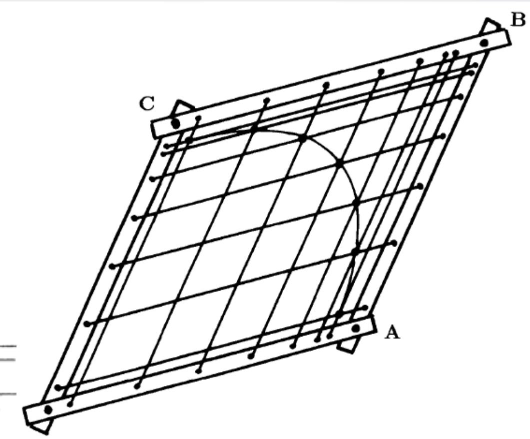
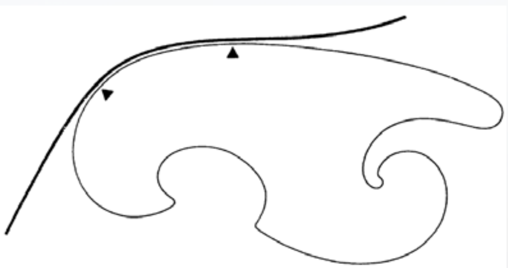
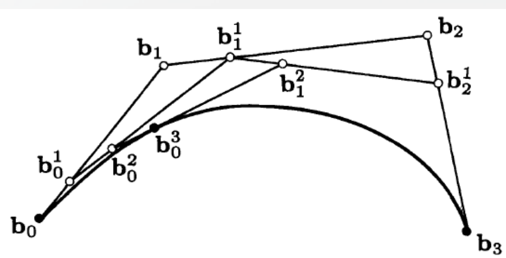


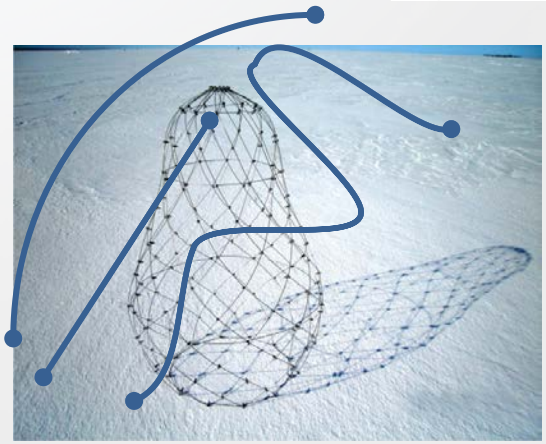






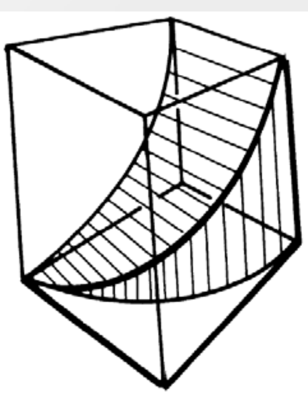






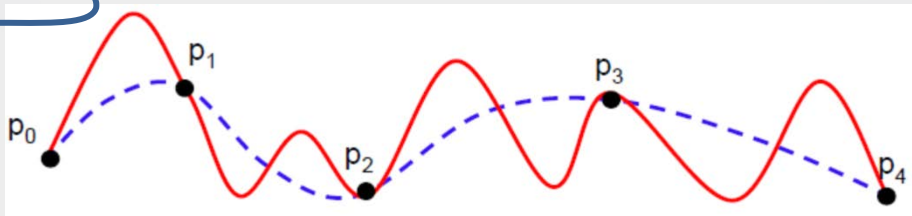
Advanced CAD Curves

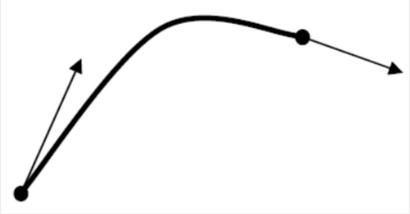
Hikmet Kocabas
Prof. PhD.
Istanbul Technical University



Lectures, Outline of the course

1 Advanced CAD Technologies, Hardwares, Softwares
2 Geometric Modeling
3 Transformations
4 Parametric Curves
5 Splines, NURBS
6 Parametric Surfaces
7 Solid Modeling
8 API programming





References

- Computer Aided Engineering Design, Anupam Saxena, Birendra Sahay, Springer, 2005

- CAD/CAM Theory and Practice , Ibrahim Zeid, McGraw Hill , 1991, Mastering CAD/CAM, ed. 2004

- The NURBS Book, Les Piegl, Springer-Verlag, 1997

- 3D CAD Principles and Applications, H Toriya, Springer-Verlag, 1991

References

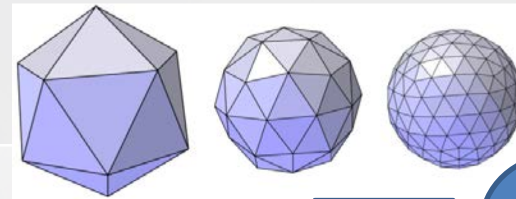
- *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide, Fourth Edition*, Gerald Farin, September 1996

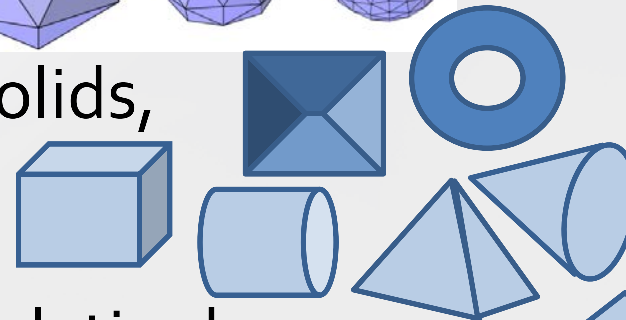
- *Geometric Modeling*, Second Edition, Michael E. Mortenson, John Wiley & Sons, January 1997

- *The NURBS Book*, Second Edition, Les A. Piegl and W. Tiller, Springer Verlay, January 1997

- *Mathematical Elements for Computer Graphics*, Rogers, D.F., Adams, J.A., McGraw Hill, 1990.

Geometric Modeling

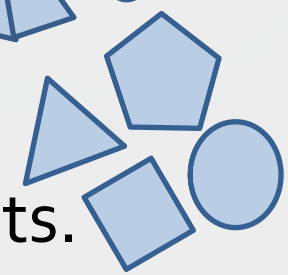




Geometric model is defined by solids, surfaces, curves, and points.

Solids are bounded by surfaces.

They represent solid objects. Analytic shape.

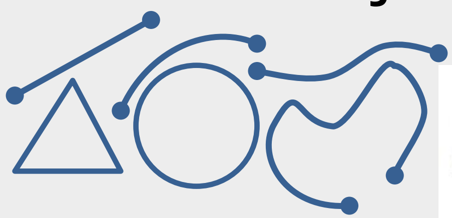


Surfaces are bounded by lines. They represent surfaces of solid objects, or planar or shell objects.

Curves are bounded by points.

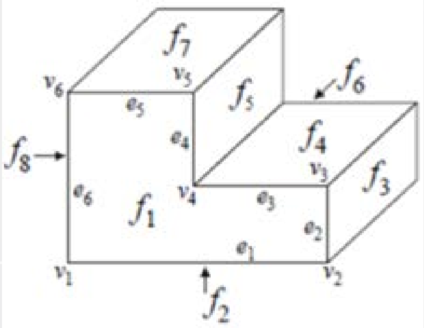
They represent edges of objects.

Lines, polylines, curve.



Points are locations in 3-D space. They represent vertices of objects. A set of points, point clouds.

Geometric Modeling



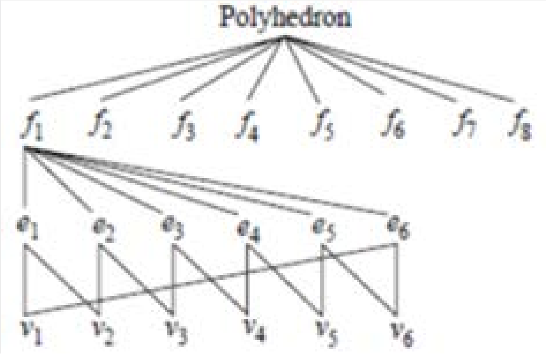
There is a built-in hierarchy among solid model entities.

Solids (volumes) from surfaces,

Surfaces (faces) from curves,

Curves (edges) are built from the points,

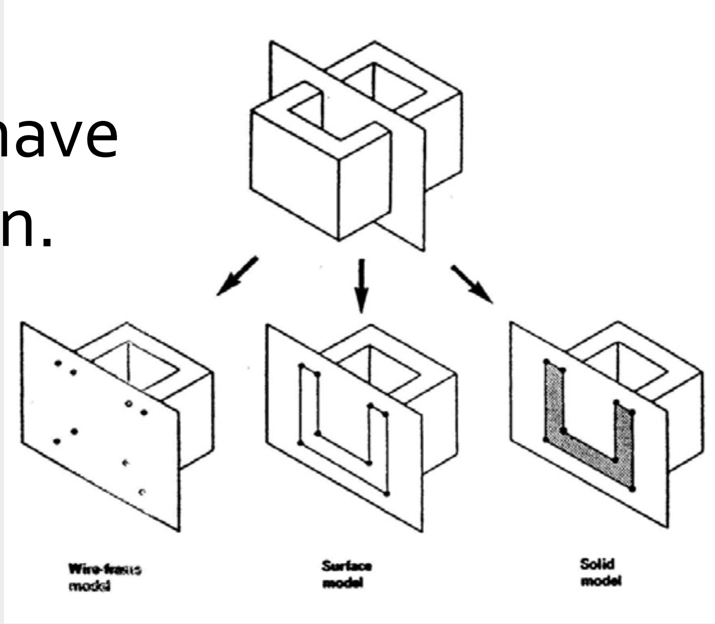
Points (vertices) are the foundation entities.



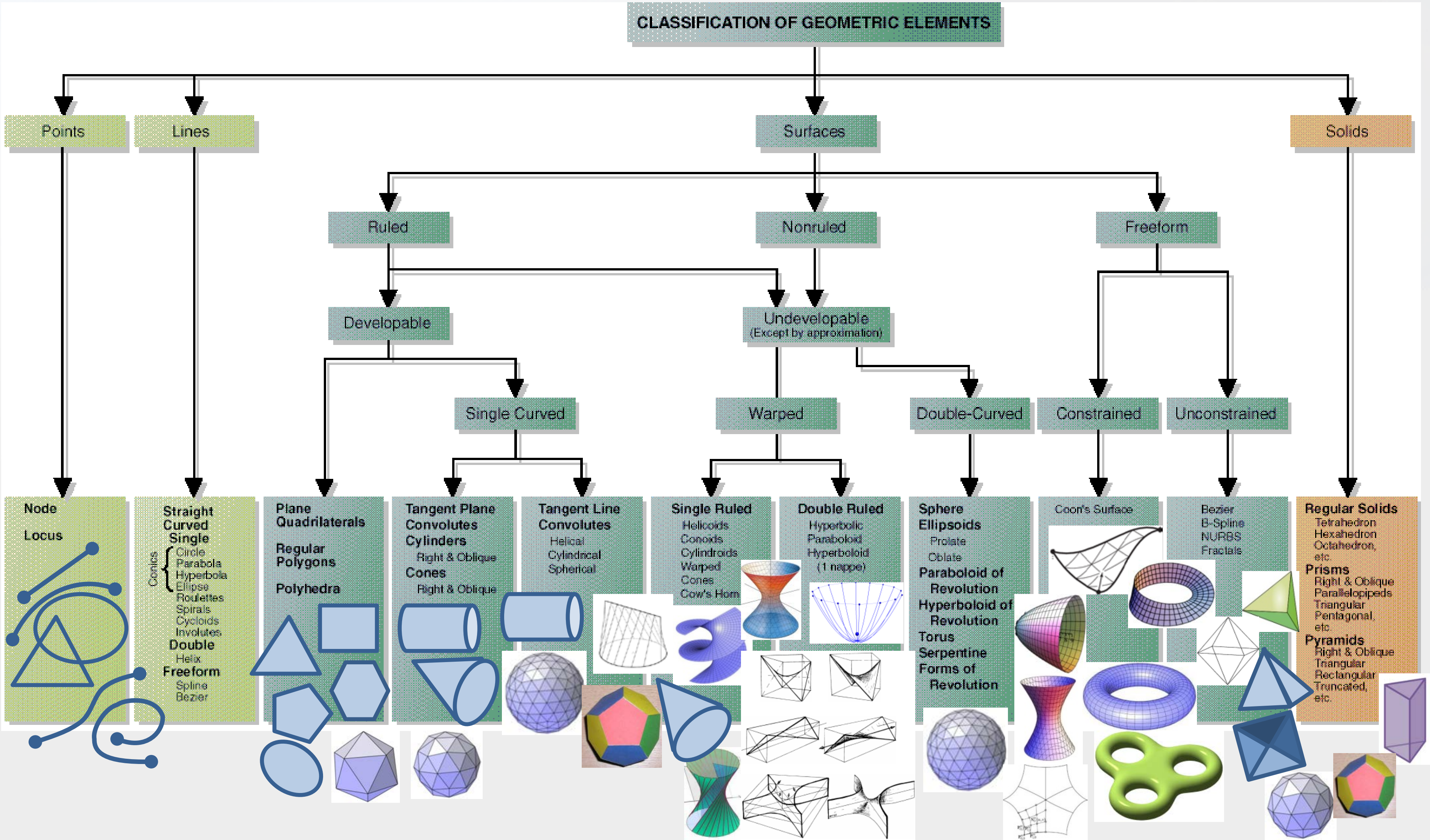
The wire frame models does'nt have the surface and volume definition.

Difference between

wire, surface and solid models :

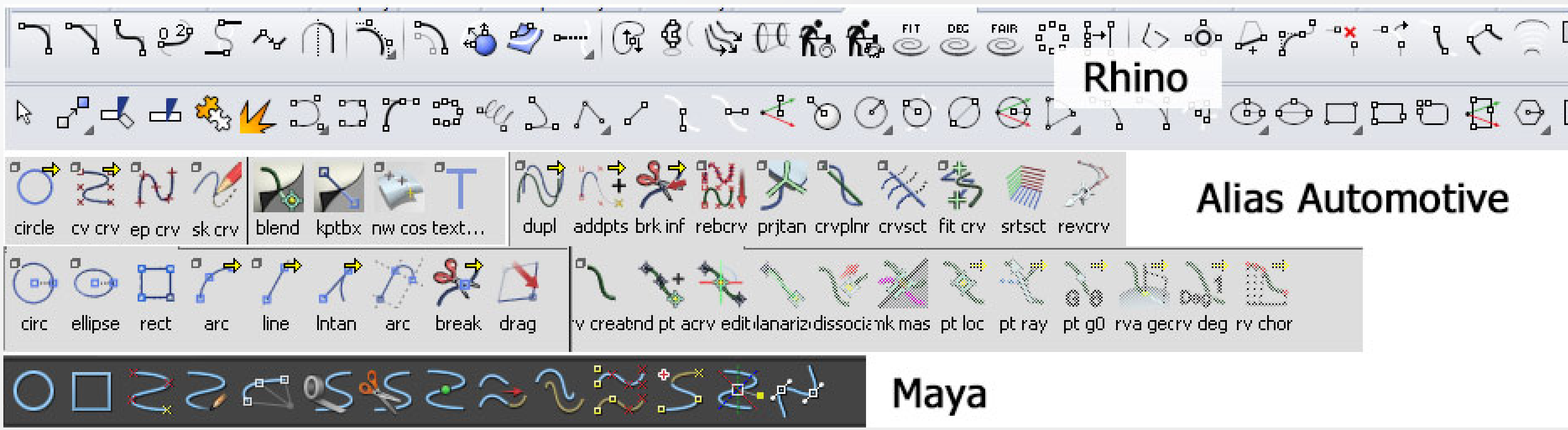


geometric elements



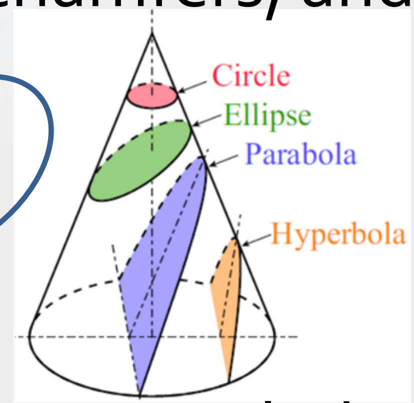
Sketching curve tools

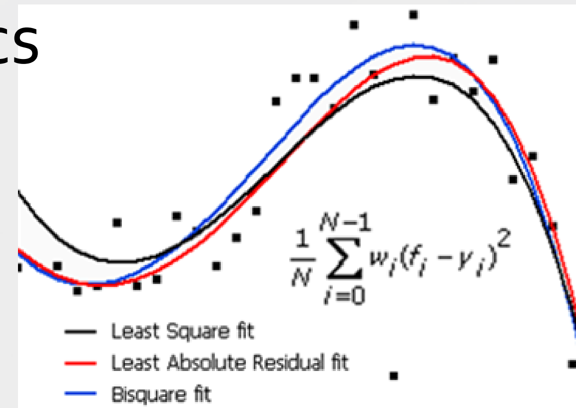
Curve tools in CAD programs



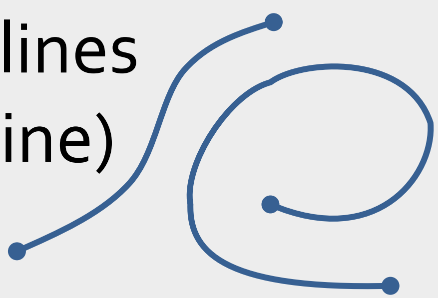
Analytic and Free Form Synthetic Curves

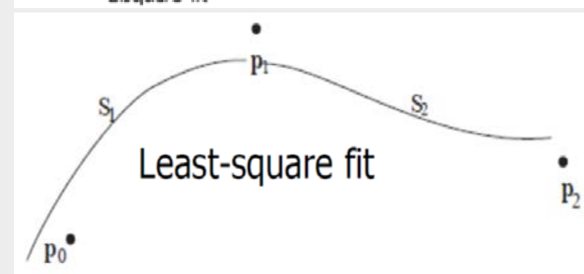
Analytic Curves are points, lines, arcs and circles, fillets and chamfers, and conics (ellipses, parabolas, and hyperbolas)

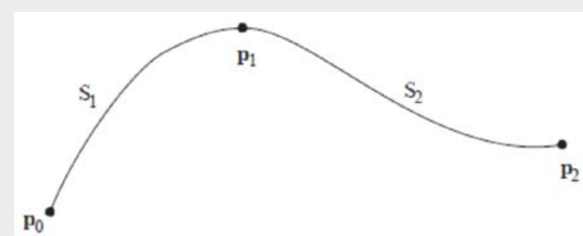




Free form synthetic curves include various types of splines (cubic spline, B-spline) and Bezier curves.



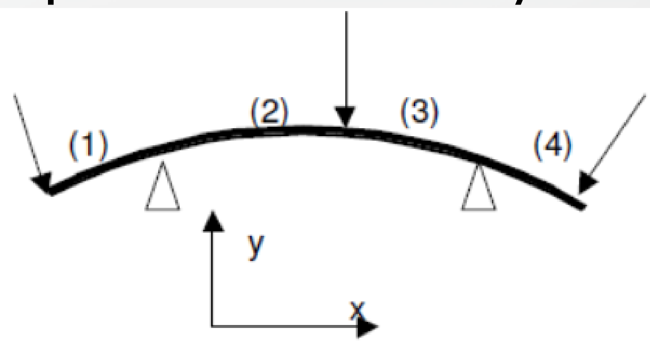




Splines

Splines -- a mechanical beam with bending deflections, or a smooth curve under multiple constraints.

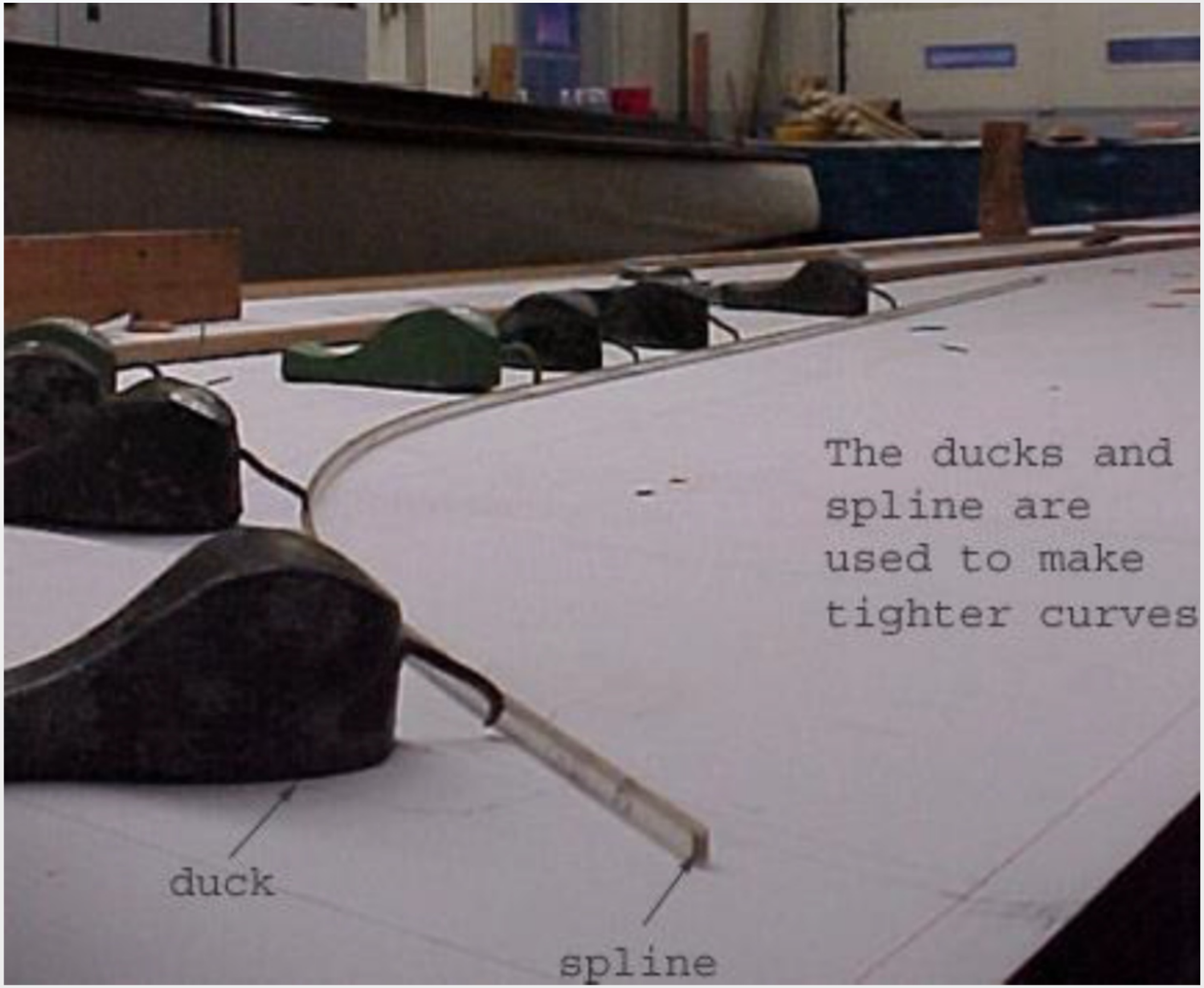
Natural Spline : mathematical approximation of the spline historically used in naval construction.



$$y''(x) = \frac{M(x)}{EI} = \frac{a_i x + b_i}{EI}$$

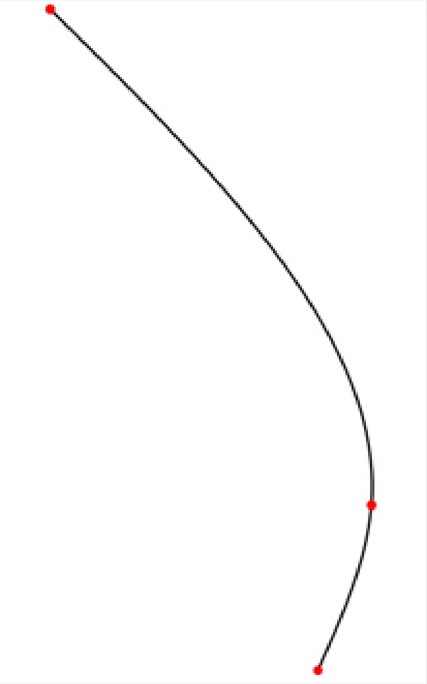
$$y(x) = \frac{1}{EI}\left[\frac{a_i}{6}x^3 + \frac{b_i}{2}x^2 + c_i x + d_i\right]$$

Physical Splines

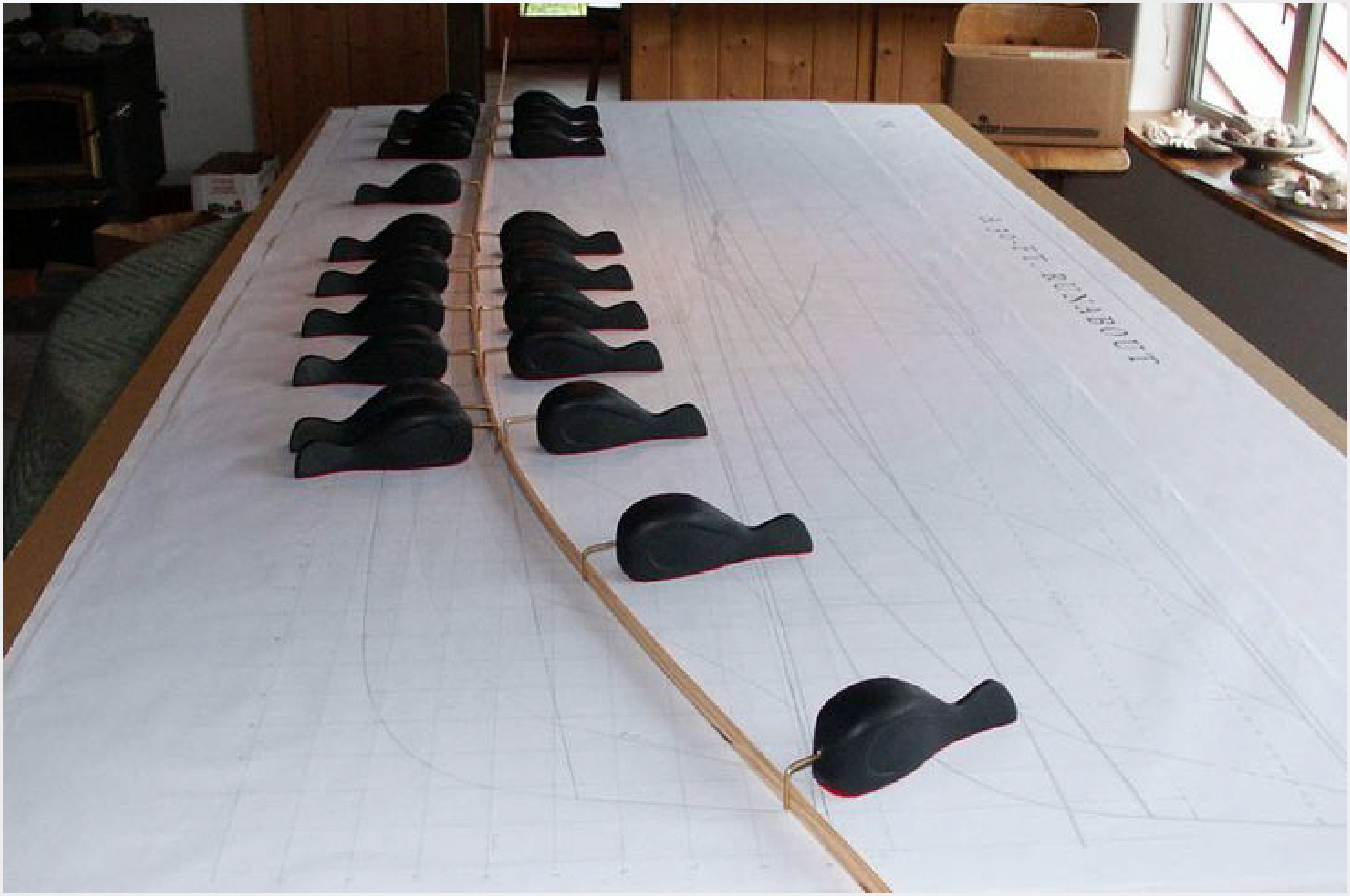


Physical Splines

The splines were once physical things. In an era prior to CAD and large-format printing, when draftsmen needed to lay out full-sized curves for boatbuilding, airplane manufacturing and the like this is how they did it

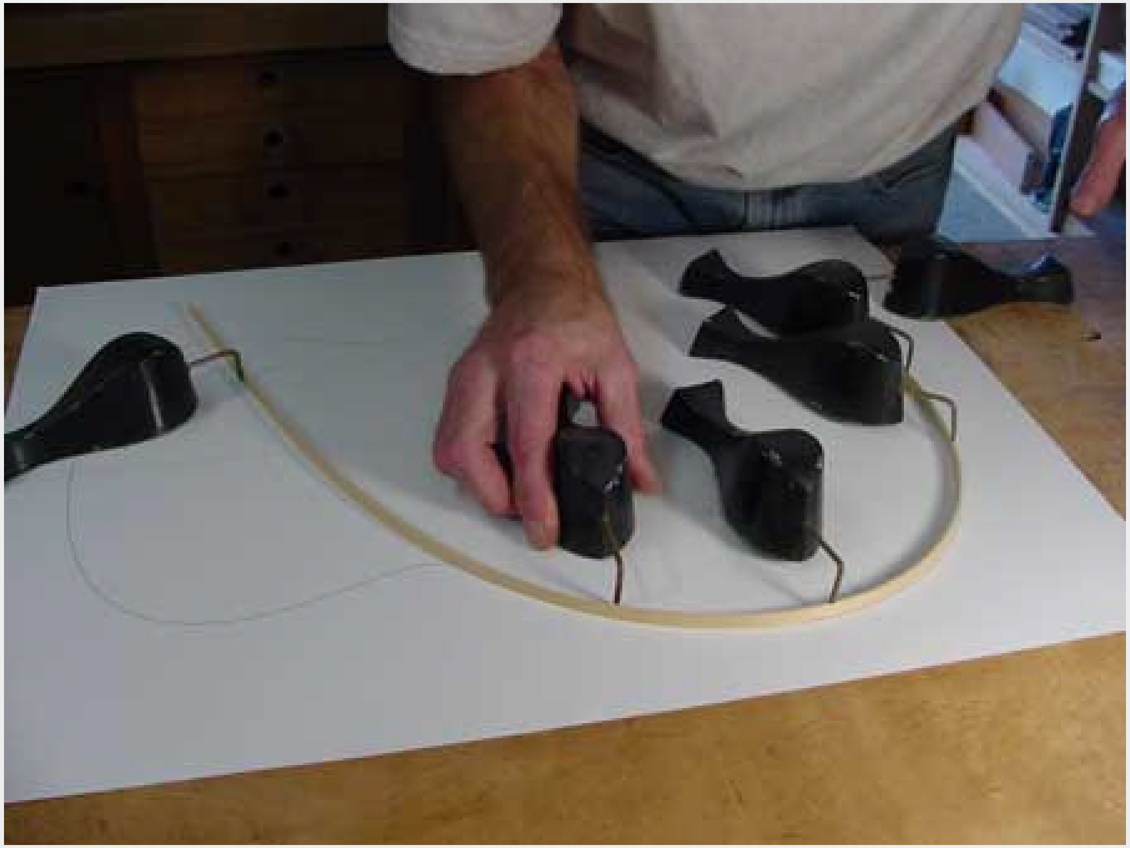


Physical Splines



Physical Splines

To be clear, the "spline" is the actual strip of wood being bent and held in place. The things holding it in place are called spline weights, or colloquially, "ducks" or "whales." They weigh about five pounds apiece.



Physical Splines

Spline weights were typically cast in lead, then painted to prevent the user from transferring lead smudges from his hands onto the drawing. The bottom was lined in felt, to prevent tearing the paper it sat on. A protruding hook was used to pin the spline itself down.

Physical Splines

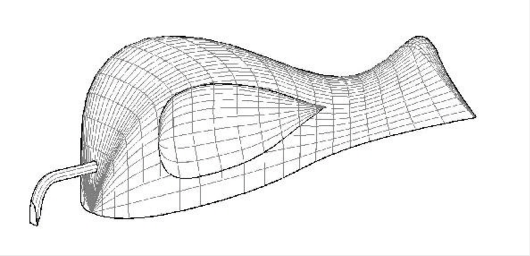
The surfaces of these heavy objects were purposely cast rough, which made them easier to pick up.

Physical Splines

In 2005 a company named Edson began manufacturing spline weights out of bronze rather than lead, to reduce the health risk. They go for $50 a pop.

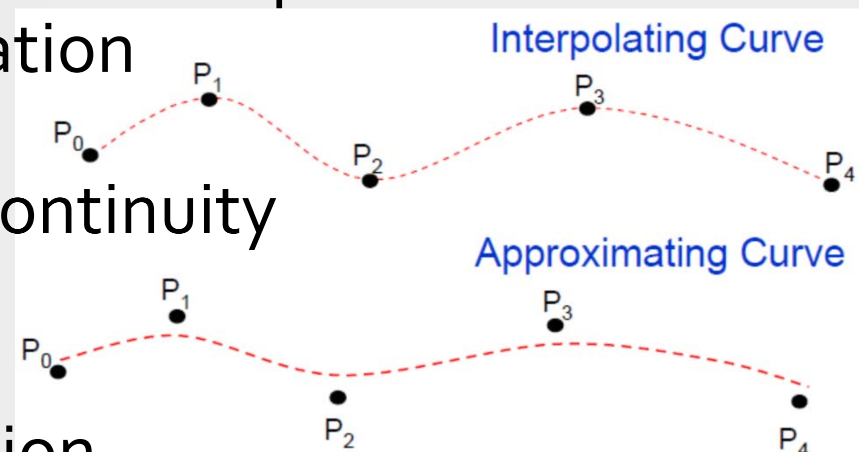
Physical Splines

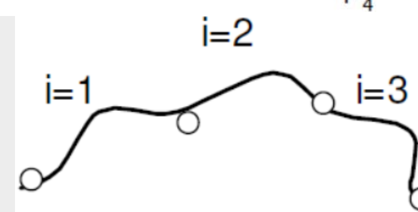
Anyways, next time you CAD jockeys are trying to massage the carpal tunnel out of your wrist, just be glad your mouse doesn't weigh five pounds.

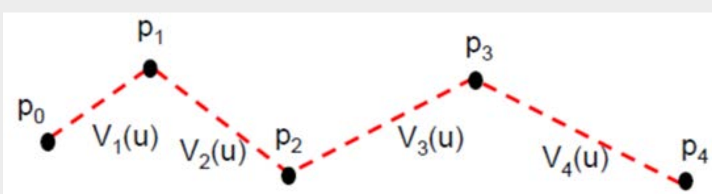


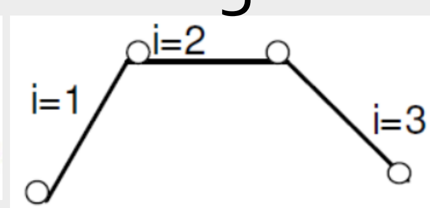
Curves Representations, Splines

Motivations
Techniques for Object Representation
Curves Representation, Free Form Representation
Approximation and Interpolation
Parametric Polynomials
Parametric and Geometric Continuity
Polynomial Splines
Hermite Interpolation
Primitive Based Representation,
Line segments: A curve is approximated by a collection of connected line segments

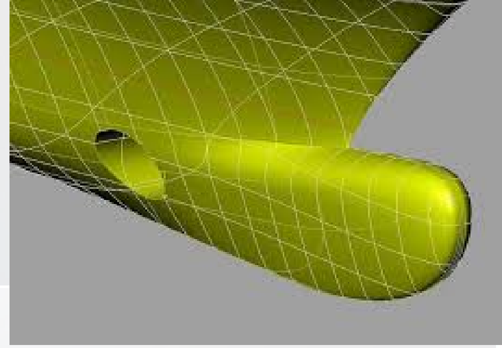








Parametric Curves



(1) Parametric vs. Nonparametric Curve Equation

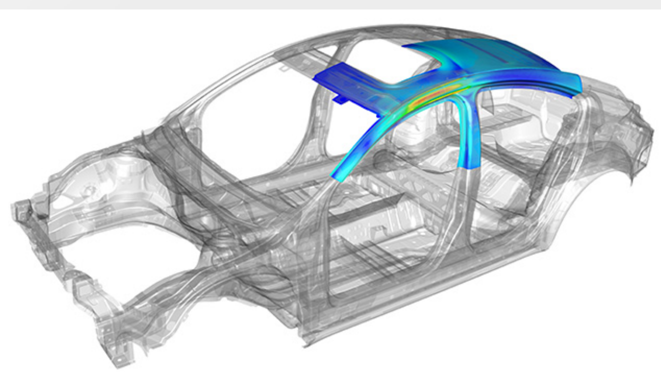
(2) Various free-form curves (some mathematics) :

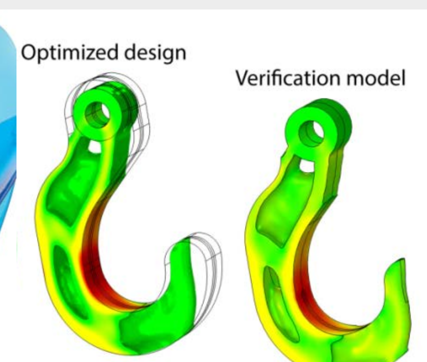
Hermite, Bezier, B-Spline Curve,

NURBS (Nonuniform Rational B-Spline) Curves

Applications: CAD, CAM, FEM, Design Optimization

Examples: car bodies, ship hulls, airplane fuselage and wings, propeller blades, shoe insoles, and bottles



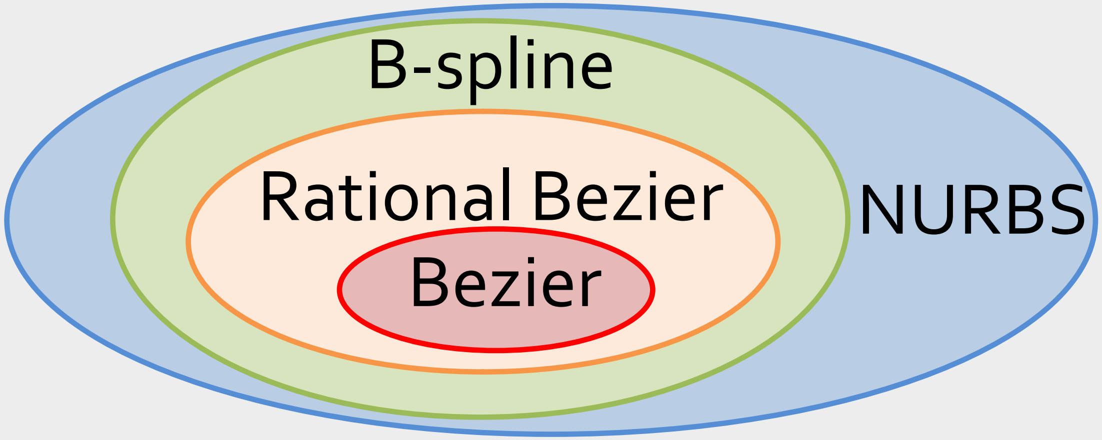


Bezier curves, Spline curves, NURBS

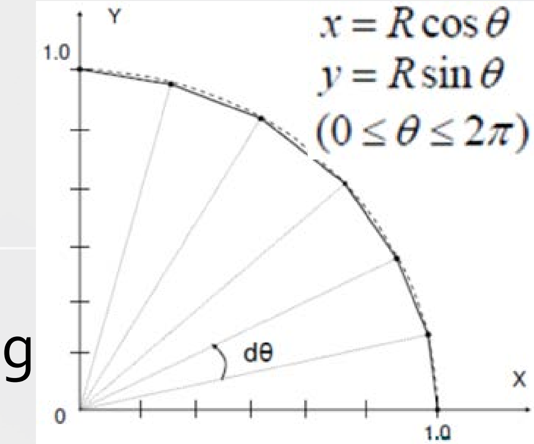
Bézier curves and B-splines are generalized to rational Bézier curves and Non-Uniform Rational B-splines (NURBS).

Rational Bézier curves are more powerful than Bézier curves since the former now can represent analitical curves, circles and ellipses.

Similarly, NURBS are more powerful than B-splines.

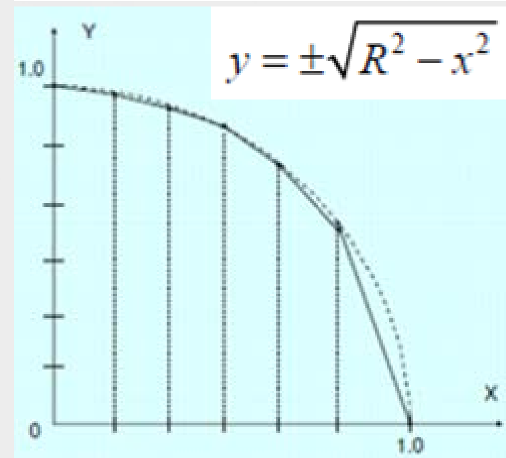


Parametric vs. Nonparametric Curve Representations



(1) Parametric equation is good for calculating the points at a certain interval along a curve.
x, y, z coordinates are related by a parametric and independent variable (*u* or θ)
Point on 2-D, 3-D curve : $\mathbf{p} = [x(u) \quad y(u)]$

$$\mathbf{p} = [x(u) \quad y(u) \quad z(u)]$$



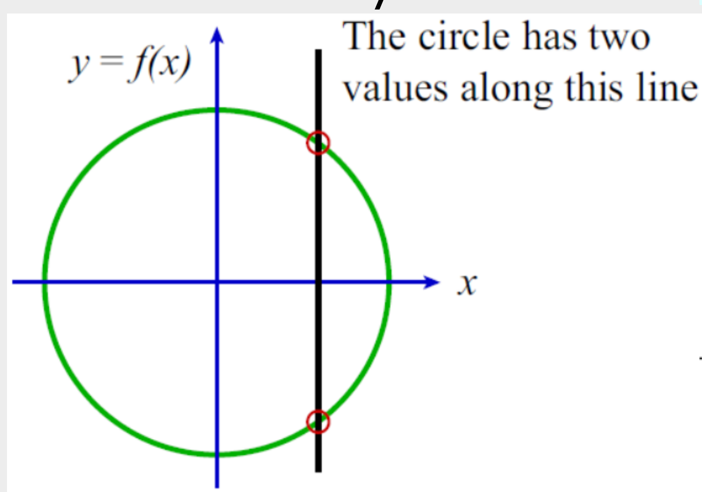
(2) Nonparametric equation
x, y, (z=0) coordinates are related by a function

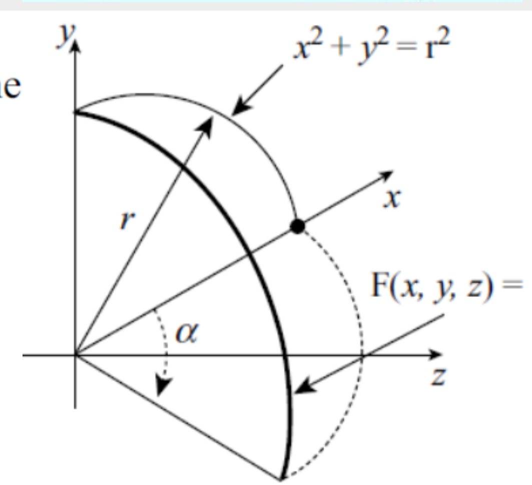
Explicit:

$$y = \pm\sqrt{R^2 - x^2}$$

Implicit:

$$x^2 + y^2 - R^2 = 0$$





Other Representations

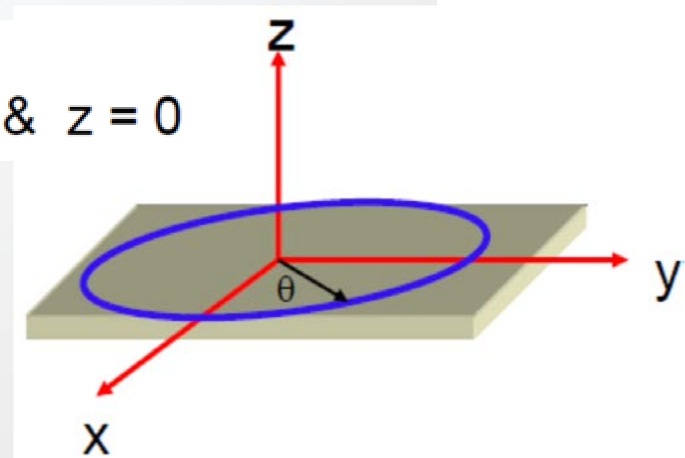
3D circle of radius R

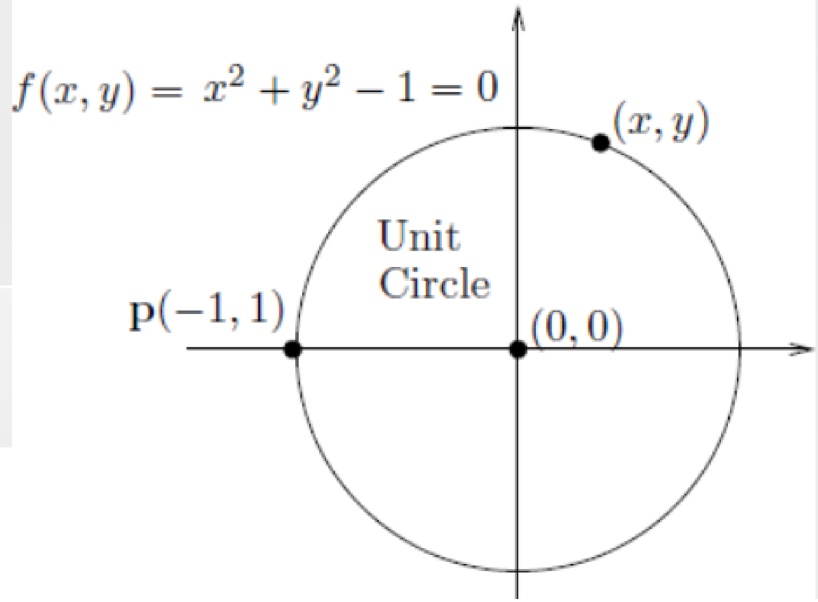
Implicit:

$x^2 + y^2 + z^2 - R^2 = 0$ & $z = 0$

Parametric:

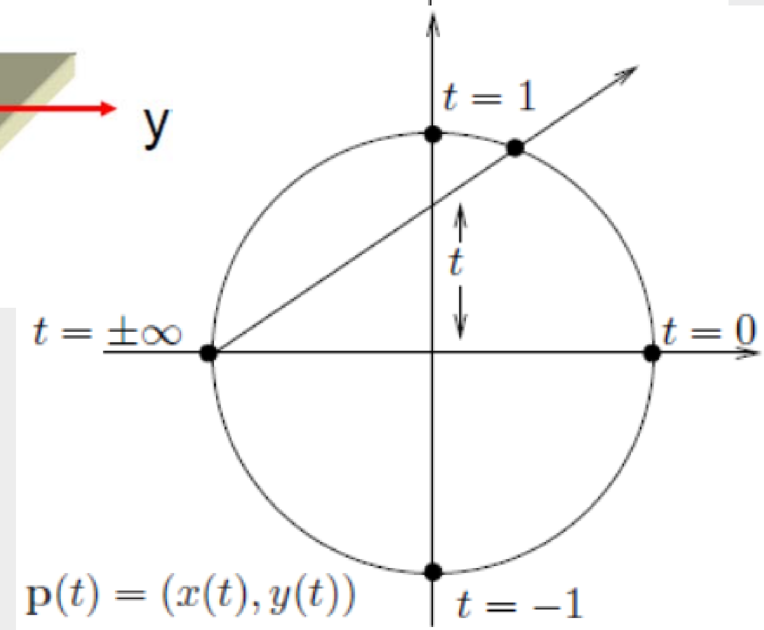
$x(\theta) = R\cos(\theta)$
$y(\theta) = R\sin(\theta)$
$z(\theta) = 0$





Other 2D representations of a circle is **p(t)=(x(t),y(t))** coordinate functions and homogenous coordinates.

$$(x(t), y(t), w(t)) = (1 - t^2, 2t, 1 + t^2)$$



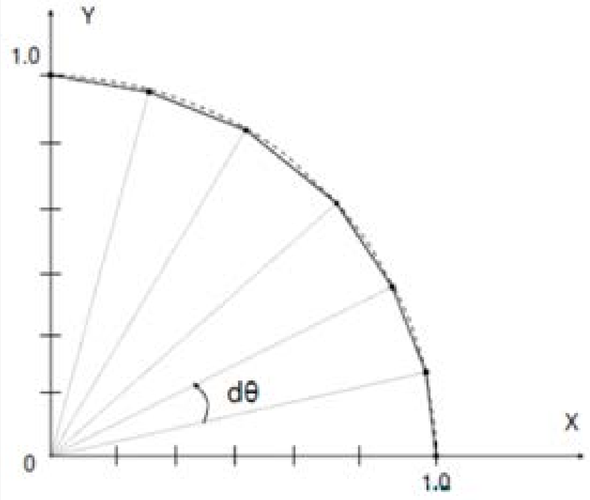
where $x(t) = \dfrac{1 - t^2}{1 + t^2}$, $y(t) = \dfrac{2t}{1 + t^2}$

Circle Represented in Recursive Fashion

$$\begin{cases} x_n = r\cos\theta \\ y_n = r\sin\theta \end{cases}$$

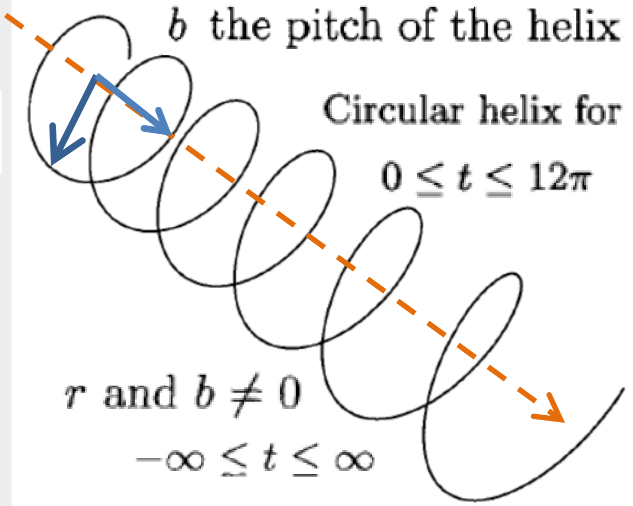
$$x_{n+1} = r\cos(\theta + d\theta) = r\cos\theta\cos d\theta - r\sin\theta\sin d\theta$$

$$\begin{cases} x_{n+1} = x_n \cos d\theta - y_n \sin d\theta \\ y_{n+1} = y_n \cos d\theta + x_n \sin d\theta \end{cases}$$



Circular helix parametric equation

$$P(t) = [\,x(t) \quad y(t) \quad z(t)\,]$$
$$= [\,r\cos t \quad r\sin t \quad bt\,]$$



Curve representation equations

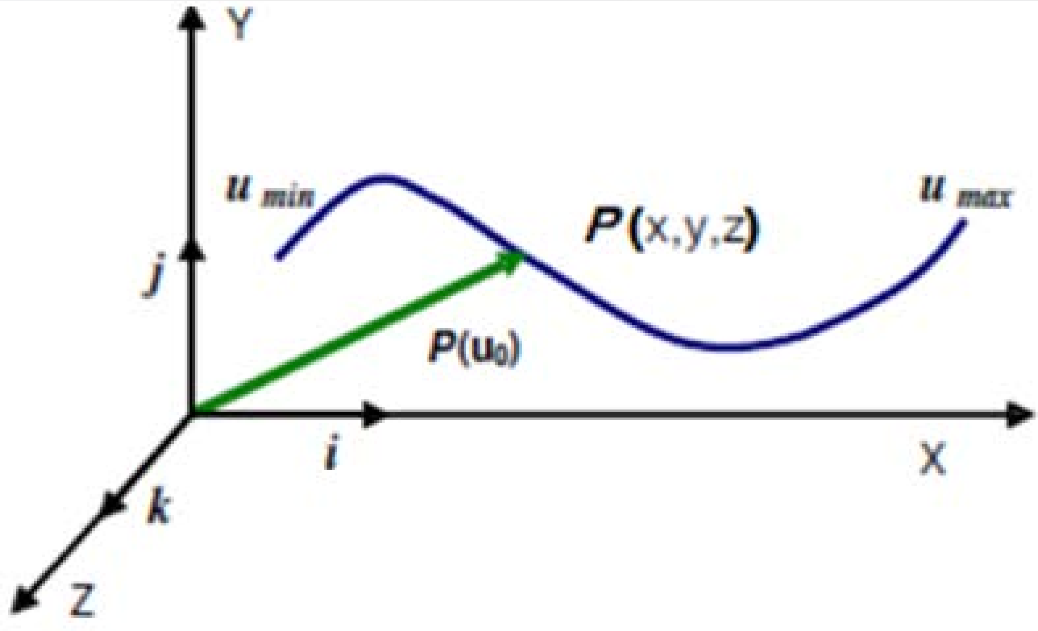
Nonparametric equation: related to *x*, *y*, *z*

Explicit $z = f(x, y)$

Implicit $f(x, y, z) = 0$

Parametric:

A **one-to-one mapping** from parametric space into the Euclidean space



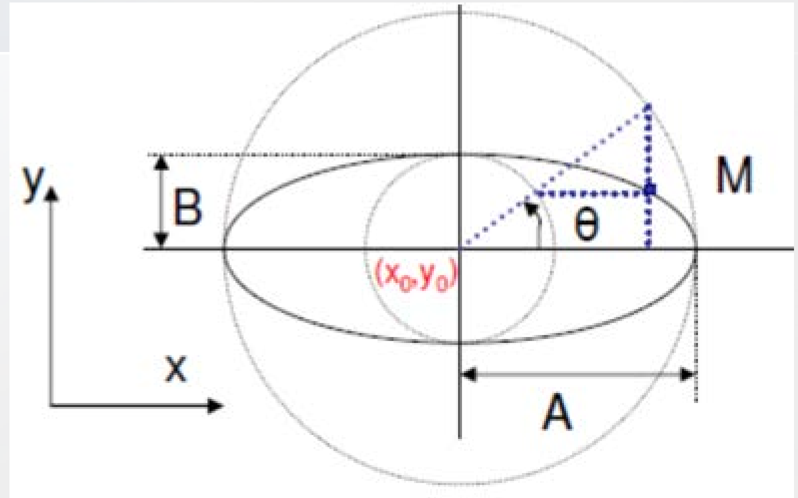
$x = X(u)$, $y = Y(u)$ and $z = Z(u)$, $\quad u_{min} \le u \le u_{max}$

$(X(u), Y(u), Z(u))$ -> $\vec{P}(u) = [X(u), Y(u), Y(u)]^T$

Other Parametric Curves

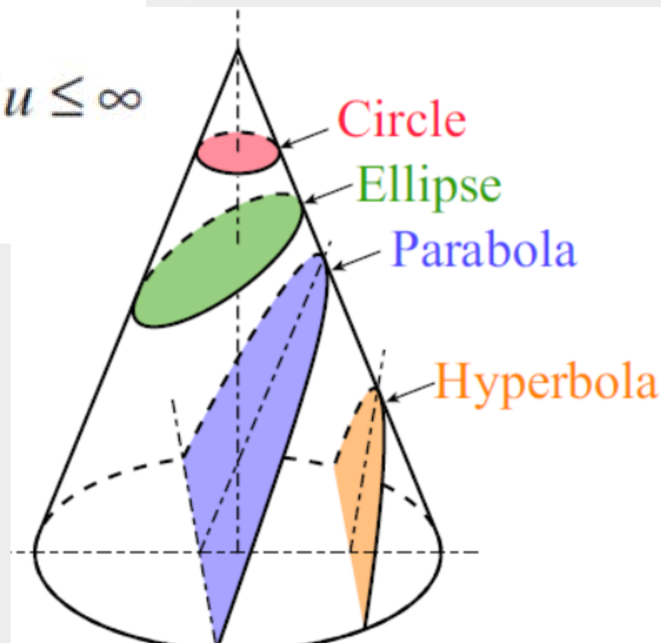
Ellipse

$$\begin{cases} x = x_o + A\cos\theta \\ y = y_o + B\sin\theta \qquad 0 \le \theta \le 2\pi \\ z = z_o \end{cases}$$



Parabola

$$\begin{cases} x = x_o + Au^2 \\ y = y_o + 2Au \qquad 0 \le u \le \infty \\ z = z_o \end{cases}$$

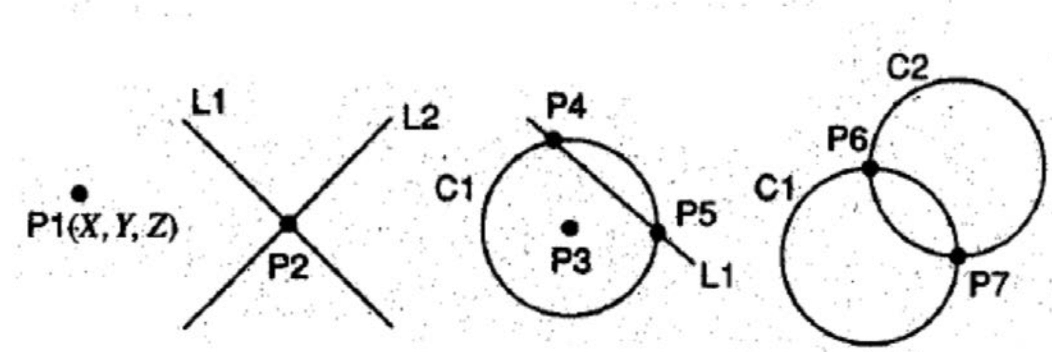


Conic section	Equation
Circle	$x^2 + y^2 = a^2$
Ellipse	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$
Parabola	$y^2 = 4ax$
Hyperbola	$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$

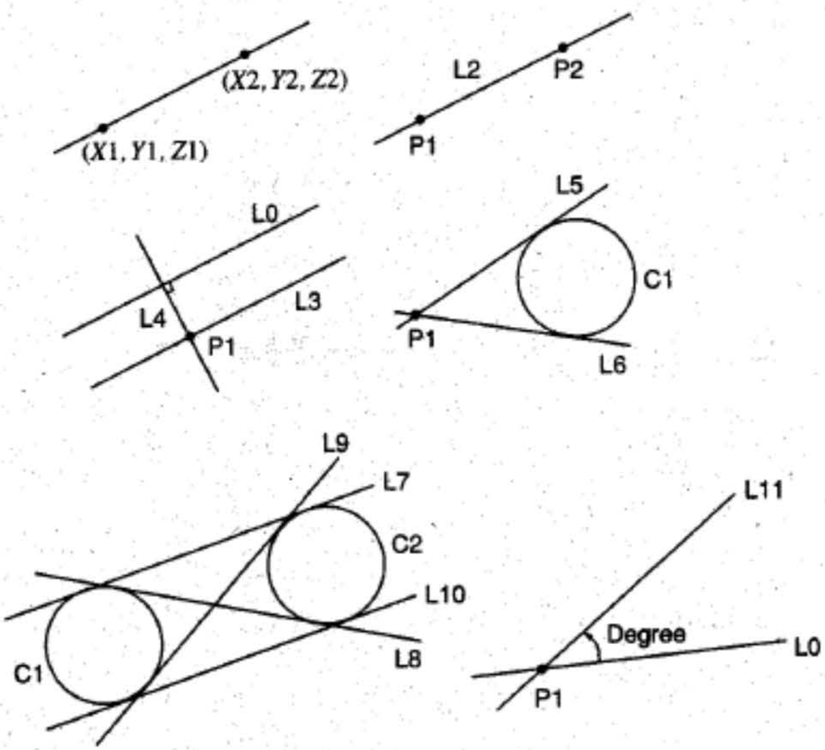
2D CAD

APT Statements

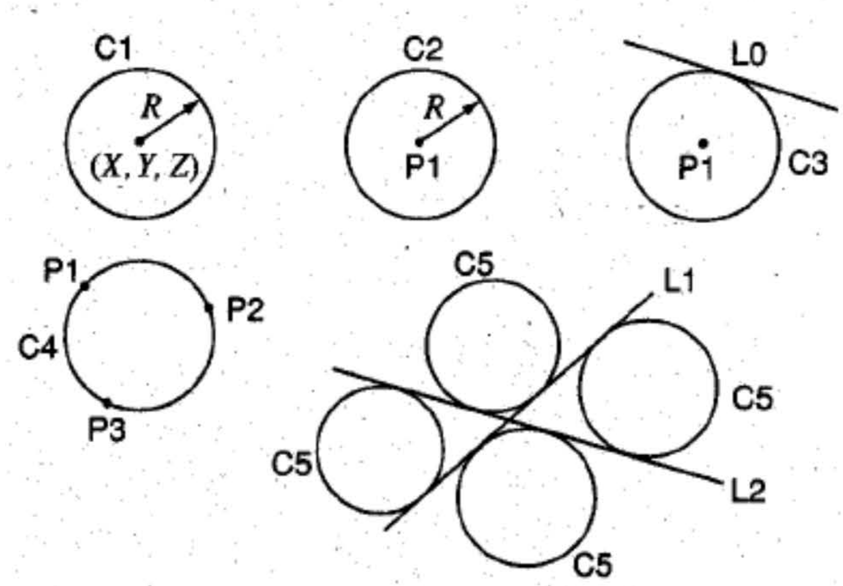
Points



Lines

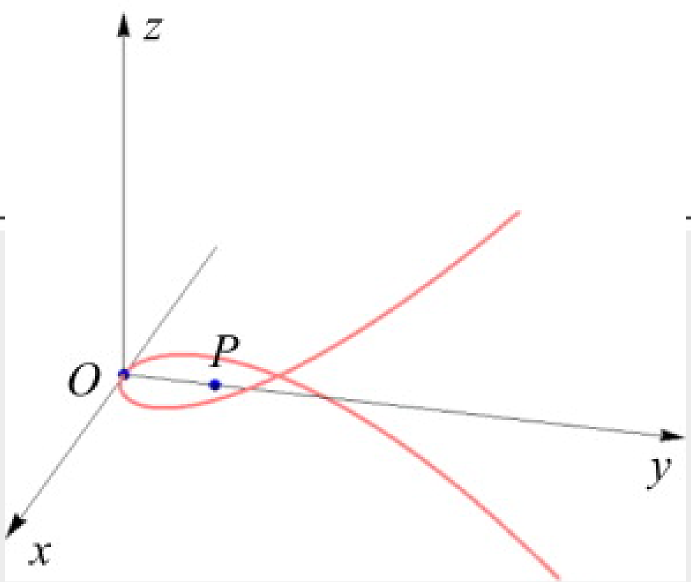


Circles

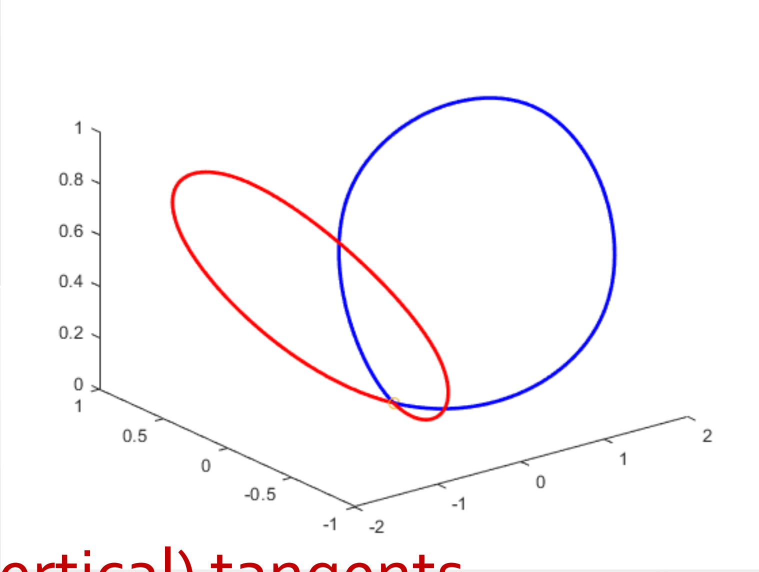


Representations of curves and surfaces

Geometry	Parametric	Implicit	Explicit
Plane curves	$x = x(t)$, $y = y(t)$ $t_1 \le t \le t_2$	$f(x, y) = 0$ or $\mathbf{r} = \mathbf{r}(u, v) \cap$ plane	$y = F(x)$
Space curves	$x = x(t)$, $y = y(t)$, $z = z(t)$, $t_1 \le t \le t_2$	$f(x, y, z) = 0 \cap g(x, y, z) = 0$ or $\mathbf{r} = \mathbf{r}(u, v) \cap f(x, y, z) = 0$ or $\mathbf{r} = \mathbf{p}(\sigma, t) \cap \mathbf{r} = \mathbf{q}(u, v)$	$y = Y(x) \cap$ $z = Z(x)$
Surfaces	$x = x(u, v)$, $y = y(u, v)$, $z = z(u, v)$, $u_1 \le u \le u_2$, $v_1 \le v \le v_2$	$f(x, y, z) = 0$	$z = F(x, y)$



Comparison of representations



Explicit Form $z = f(x, y)$

- Easy to render
- Unique representation
- Difficult to represent all (vertical) tangents

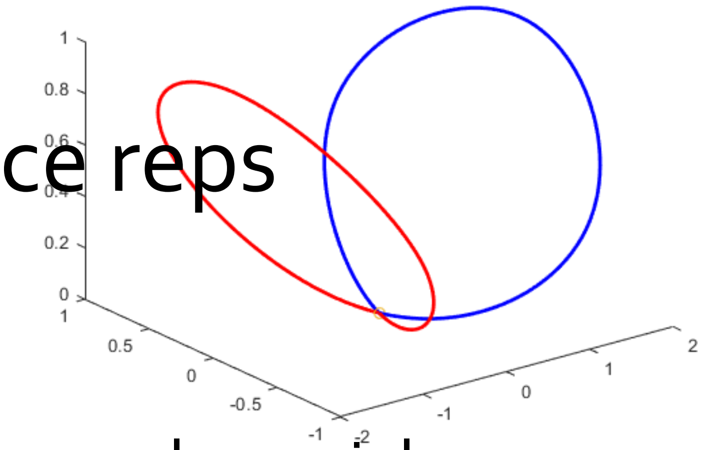
Implicit Form $f(x, y, z) = 0$

- Easy to test and determine if a point (object) lies on, inside, or outside a curve or surface
- Unique representation
- Difficult to connect two curves in a smooth manner
- Not efficient for drawing. Difficult to render.

Parametric Representation $p(u) = [x(u) \;\; y(u) \;\; z(u)]$

- Easy to render and common in modeling
- Representation is not unique

Comparison of curve, surface reps



Disadvantages

Explicit   z = f(x, y)

Infinite slopes are impossible if f(x,y) is a polynomial.

Axis dependent (difficult to transform).

Closed and multivalued curves are difficult to represent.

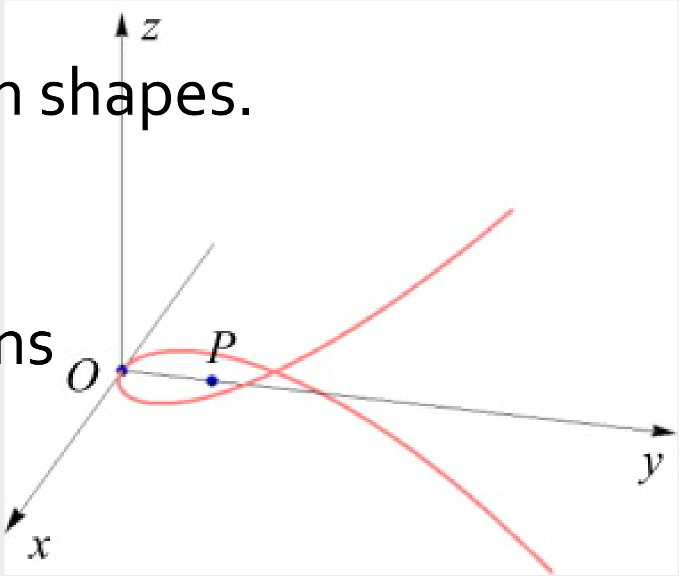
Implicit   f(x, y, z) = 0

Difficult to fit and manipulate free form shapes.

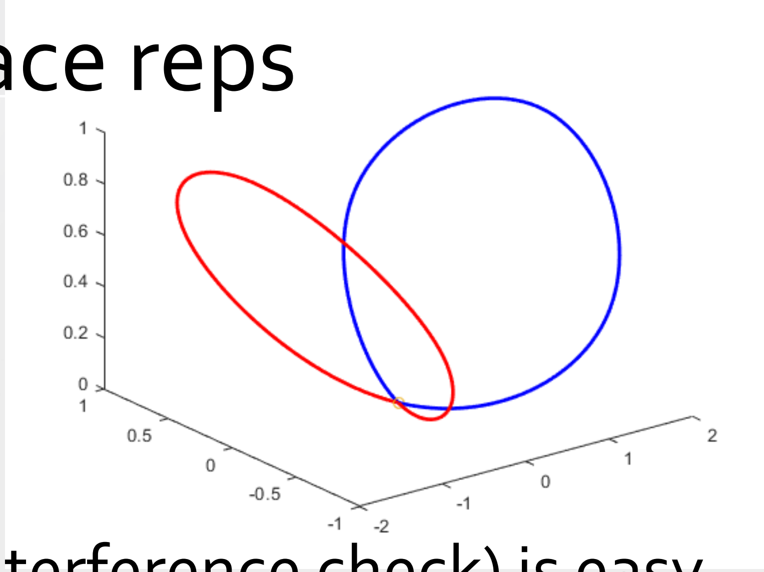
Axis dependent. Complex to trace.

Parametric   p(t)=(x(t), y(t), z(t))

High flexibility complicates intersections and point classification.



Comparison of curve, surface reps



Advantages

Explicit $z = f(x, y)$ Easy to trace.

Implicit $f(x, y, z) = 0$

Closed and multivalued curves and infinite slopes can be represented

Point classification (solid modeling, interference check) is easy

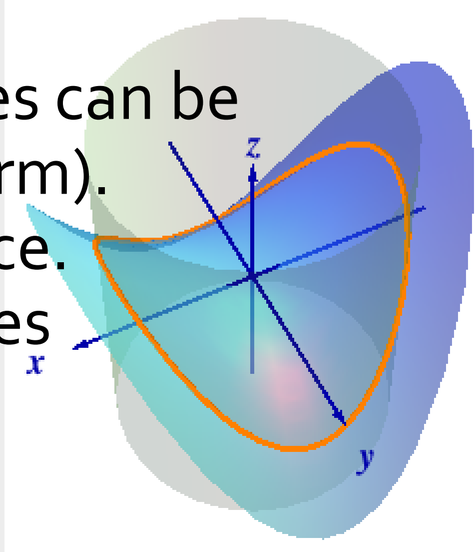
Intersections/offsets can be represented.

Parametric $p(t)=(x(t), y(t), z(t))$

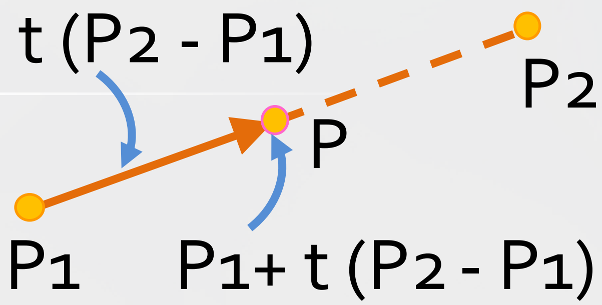
Closed and multivalued curves and infinite slopes can be represented. Axis independent (easy to transform).

Easy to generate composite curves. Easy to trace.

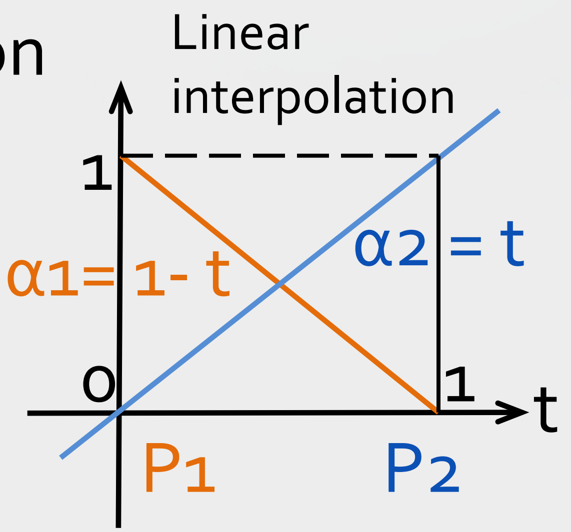
Easy in fitting and manipulating free-form shapes



Line
(Combinations of Points)



- Let P_1 and P_2 be points in space.
- if $0 \le t \le 1$ then P is somewhere on the line segment joining P_1 and P_2 .
- We may utilize the following notation
  $P = P(t) = (1 - t) P_1 + t P_2$

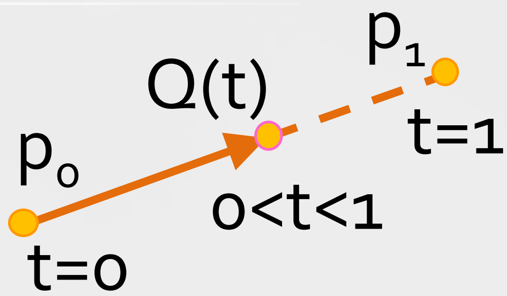


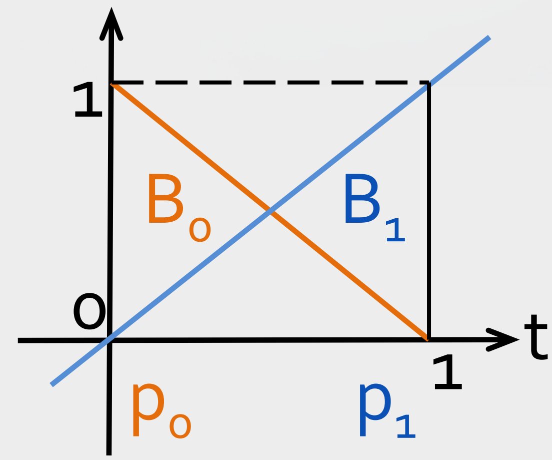
- We can then define a combination of two points P_1 and P_2 to be
- $P = \alpha_1 P_1 + \alpha_2 P_2$ where $\alpha_1 + \alpha_2 = 1$
- derive the transformation by setting $\alpha_2 = t$

Line (Linear Interpolation)

Three ways of writing a line segment:

1. Weighted average of the control points:
   $Q(t) = (1-t)p_0 + tp_1$
   Basis Blending Functions:
   $B_0(t)=1-t \qquad B_1(t)=t \qquad B_0(t)+B_1(t)=1$
   $Q(t) = B_0(t)p_0 + B_1(t)p_1$





2. Polynomial in *t:* $Q(t) = (p_1-p_0)t + p_0$

3. Matrix form: $$Q(t) = \begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \end{bmatrix}$$

Example: Linear Polynomial

The geometrical constraints for x(u) are:

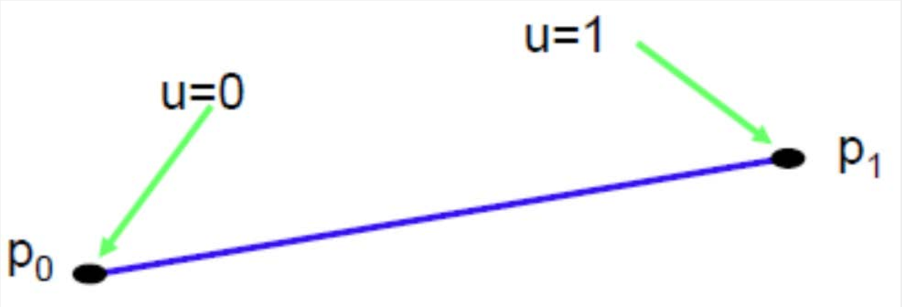
$$x(0) = a_x = P_0^x \quad ; \quad x(1) = a_x + b_x = P_1^x$$

Solving the coefficients for x(u) we get:

$$a_x = P_0^x \quad ; \quad b_x = P_1^x - P_0^x$$

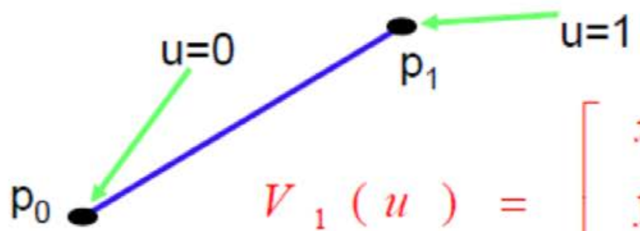
$$\Rightarrow \quad x(u) = P_0^x + (P_1^x - P_0^x) \quad u$$

Solving for [x(u) y(u) z(u)] we get:

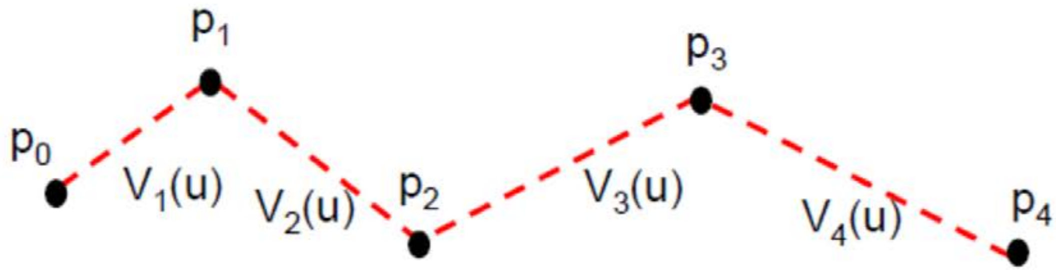


$$V(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} = P_0 + (P_1 - P_0) \quad u$$

Example: Linear Polynomial



$$V_1(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} = P_0 + (P_1 - P_0)\,u$$



$$V(u) = \begin{pmatrix} x(u) \\ y(u) \\ z(u) \end{pmatrix} = \sum_i \hat{V}_i(u)$$

$$\text{where} \qquad \hat{V}_i(u) = \begin{cases} V_i(u) & \text{if} \quad u \in [u_i, u_{i+1}] \\ 0 & \text{otherwise} \end{cases}$$

Line equation

Parametric representation

$$\mathbf{l}(\mathbf{p}_0, \mathbf{p}_1) = \mathbf{p}_0 + (\mathbf{p}_1 - \mathbf{p}_0)u \qquad u \in [0,1]$$

Parametric representation is not unique

$$\mathbf{l}(\mathbf{p}_0, \mathbf{p}_1) = \mathbf{0.5}(\mathbf{p}_1 + \mathbf{p}_0) + \mathbf{0.5}(\mathbf{p}_1 - \mathbf{p}_0)v \qquad v \in [-1,1]$$

In general $\mathbf{p}(u), \quad u \in [a, b]$

Re-parameterization (variable transformation)

$$v = (u - a)/(b - a)$$

$$u = (b - a)v + a$$

$$\mathbf{q}(v) = \mathbf{p}((b - a)v + a)$$

$$v \in [0,1]$$

Reparameterization

- Linear interpolation: $\mathbf{v} = \mathbf{v}_0(1-t) + \mathbf{v}_1(t)$

- Local coordinates: $\mathbf{v} \in [\mathbf{v}_0, \mathbf{v}_1], t \in [0,1]$

- Reparameterization:

$$f(u), u = g(v), f(g(v)) = h(v)$$

- Affine transformation:

$$f(ax+by) = af(x) + bf(y)$$
$$a + b = 1$$

Geometric Modeling - Points and Curves

- Points on 3D curves can be represented in a parametric form, as a function of a single variable $u \in [0, 1]$: $x = x(u),\ y = y(u),\ and\ z = z(u)$
- $x(u)$, $y(u)$, $z(u)$ are polynomials, usually cubic curves

- Any point on such a parametric curve is defined by the components of the vector $\mathbf{p}(u)$. Thus, the boundary conditions are defined by the vectors $[\mathbf{p}(0), \mathbf{p}(1), \mathbf{p}'(0), \mathbf{p}'(1)]$, where $\mathbf{p}'(u) = \dfrac{d\mathbf{p}(u)}{du}$

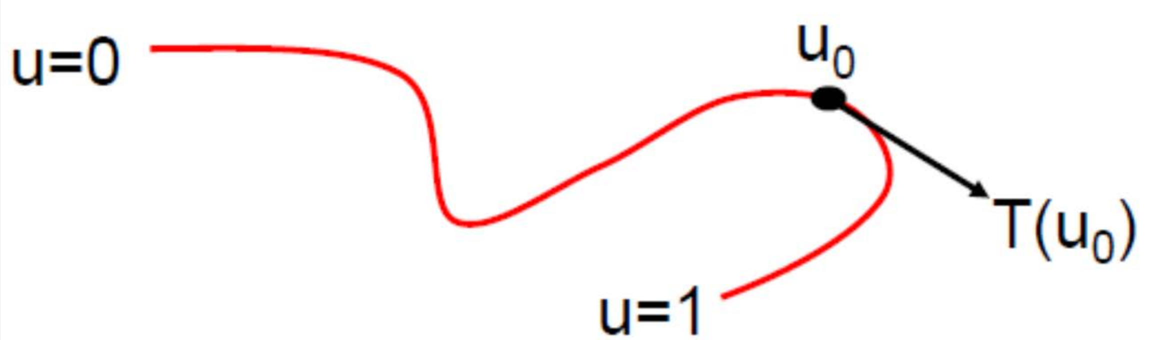
Tangent Vector

Let $V(u)=[x(u), y(u), z(u)]$, $u \rightarrow [0,1]$ be a continuous univariate parametric curve in $\mathbf{R}^3$

The tangent vector at $\mathbf{u_o}$, $\mathbf{T(u_o)}$, is:

$$\vec{T}(u_0) = V'(u_0) = \left.\frac{dV(u)}{du}\right|_{u=u_0} = \left[\frac{dx}{du}\ \frac{dy}{du}\ \frac{dz}{du}\right]_{u=u_0}$$

$V(u)$ may be thought of as the trajectory of a point in time. In this case, $\mathbf{T(u_o)}$ is the instantaneous velocity vector at time $\mathbf{u_o}$.



Parametric Continuity

Let $V1(u)$ and $V2(u)$, $u \rightarrow [0,1]$, be two parametric curves.

Level of parametric continuity of the curves at the joint between $V1(1)$ and $V2(0)$:

C^{-1}: The joint is discontinuous, $V1(1) \neq V2(0)$

C^0: Positional continuous, $V1(1)=V2(0)$

C^1: Tangent continuous, C^0 & $V'1(1)=V'2(0)$

C^k, $k>0$: Continuous up to the k-th derivative,

$V1^{(j)}(1)=V2^{(j)}(0)$, $0 \leq j \leq k$

Geometric Continuity

In computer aided geometry design, we also consider the notion of geometric continuity:

G^{-1}, G^0: Same as C^{-1} and C^0

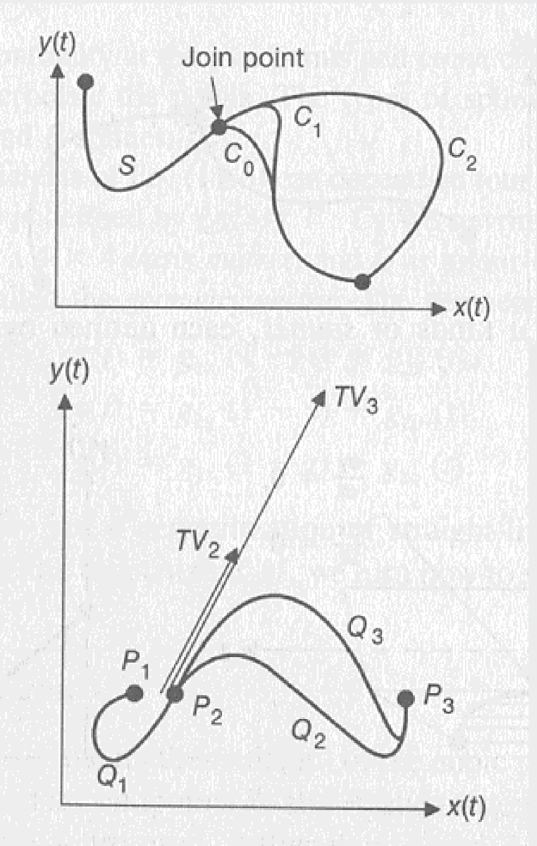
G^1: Same tangent direction:

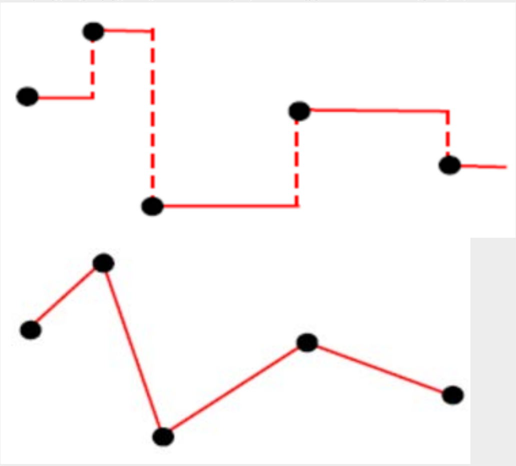
$V'1(1)=\alpha V'2(0)$

Given a set of points $\{p_i\}$:

A piecewise constant interpolant is C^{-1}

A piecewise linear interpolant is C^0





Parametric and Geometric Continuity

G^{-1}, G^0: Same as C^{-1} and C^0

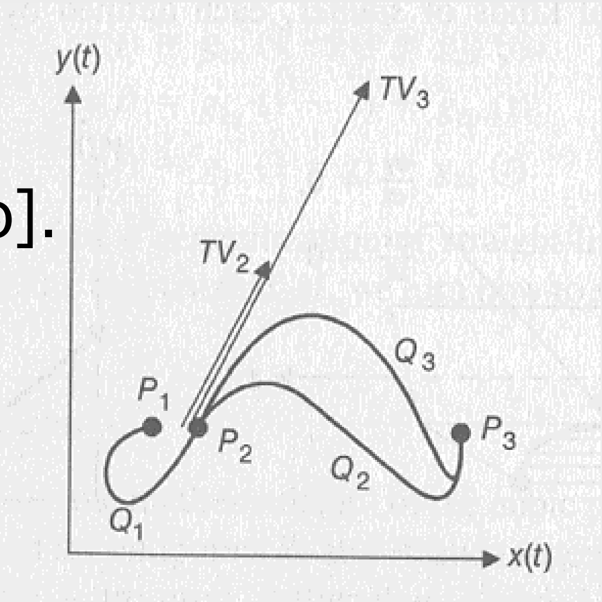
G^1: Same tangent direction: $V'1(1)=\alpha V'2(0)$

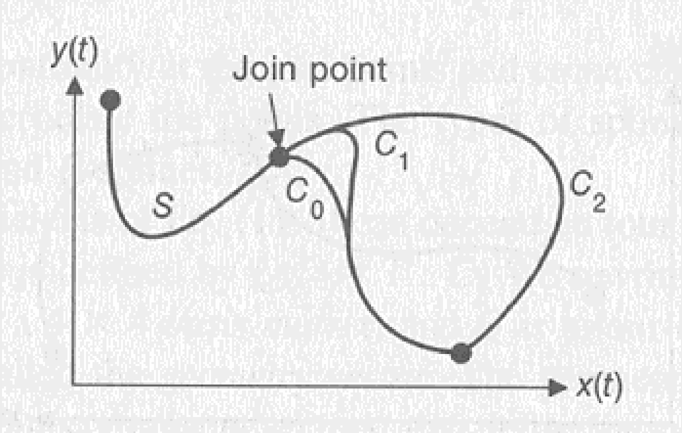
G^k: All derivatives up to the k-th order are proportional

In general, C^i implies G^i (not vice versa).

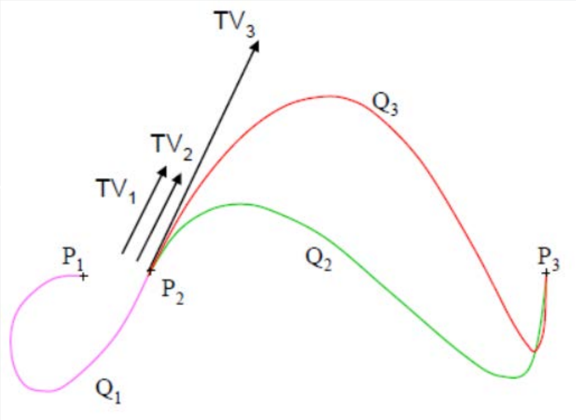
$C^1 \Rightarrow G^1$, unless tangent vector = [0, 0, 0].

Exception when the tangents are zero.





Parametric and Geometric Continuity



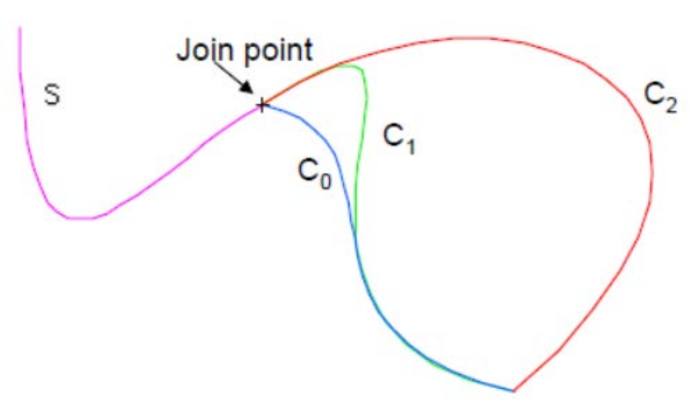
G^0 geometric continuity:
 Two curve segments join together.

G^1 geometric continuity: The direction of the two segments' tangent vectors are equal at the join point.

C^0 continuity: Curves share the same point where they join.

C^1 continuity: Tangent vectors of the two segments are equal in magnitude and direction (share the same parametric derivatives).

C^2 continuity: Curves share the same parametric second derivatives where they join.



Joining Curve Segments

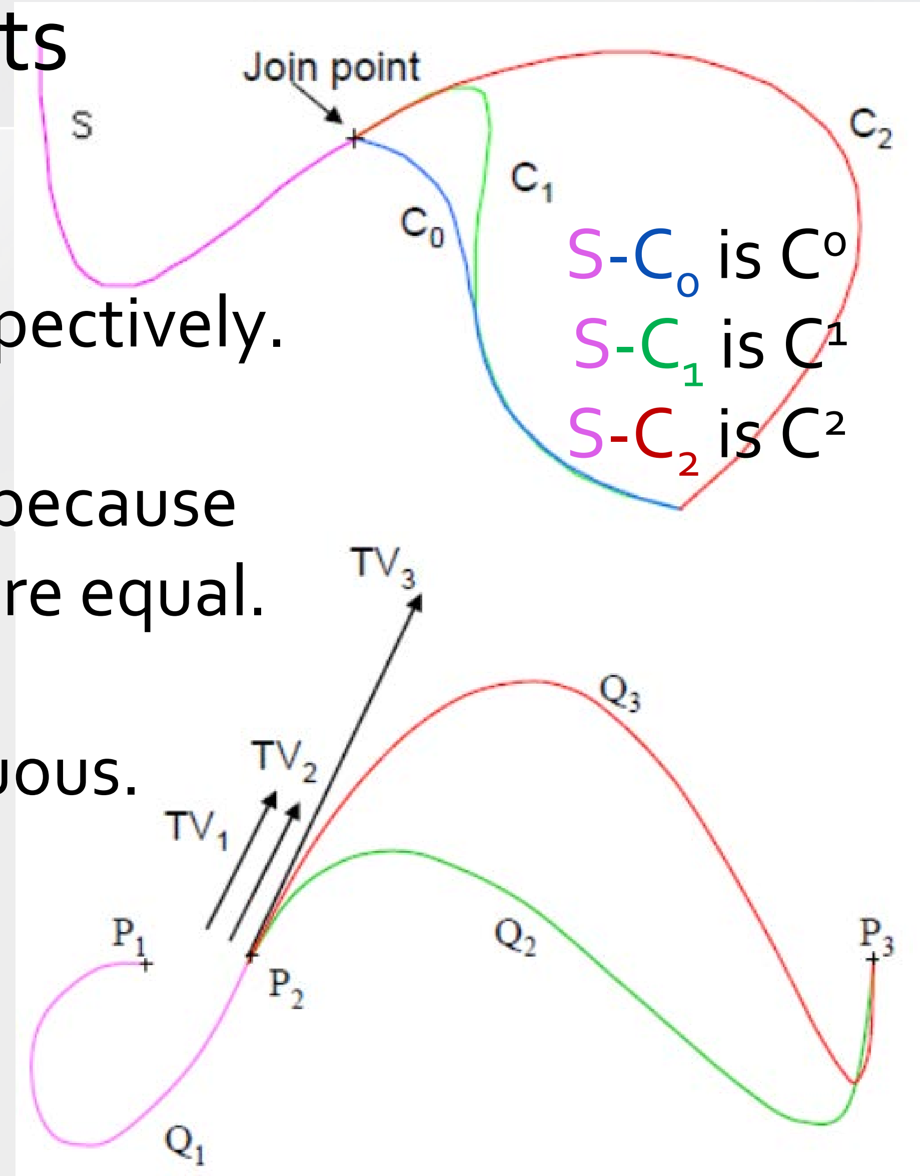
S joins C_0, C_1, and C_2 with C^0, C^1, and C^2 continuity, respectively.

Q_1 and Q_2 are C^1 continuous because their tangents, TV_1 and TV_2 are equal.

Q_1 and Q_3 are only G^1 continuous.

Q_1-Q_2 both C^1 and G^1

Q_1-Q_3 is G^1 but not C^1



S-C_0 is C^0

S-C_1 is C^1

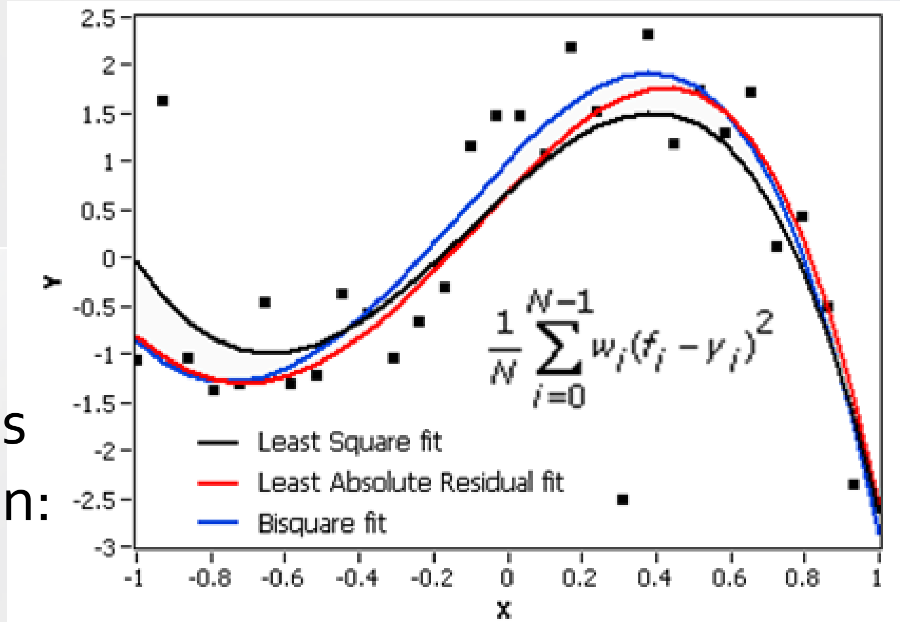
S-C_2 is C^2

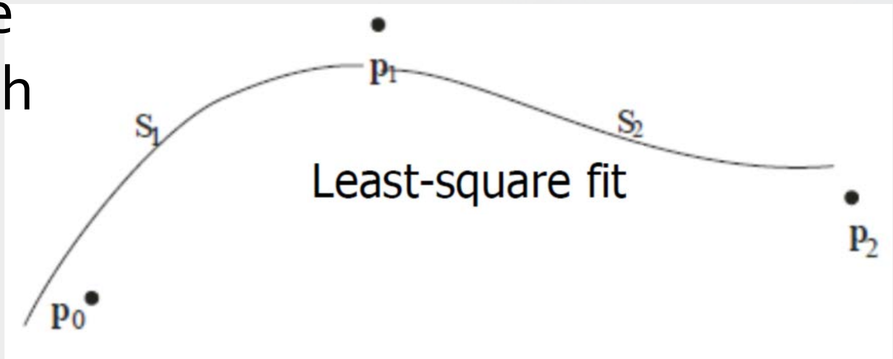
Curve fitting to points

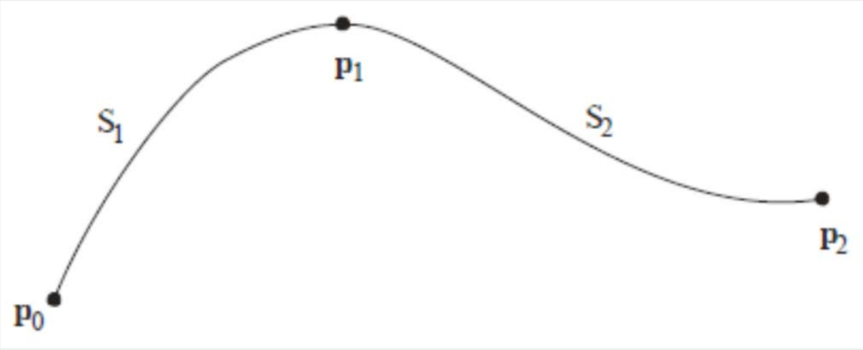
Two possible approaches to curve-fitting to a set of data points collected through experimentation:

Least-squares fit : The best curve would most likely not pass through any one of the points; and,

Spline fit : A set of curves pass through all the given points and provide any desired degree of continuity at meeting points.

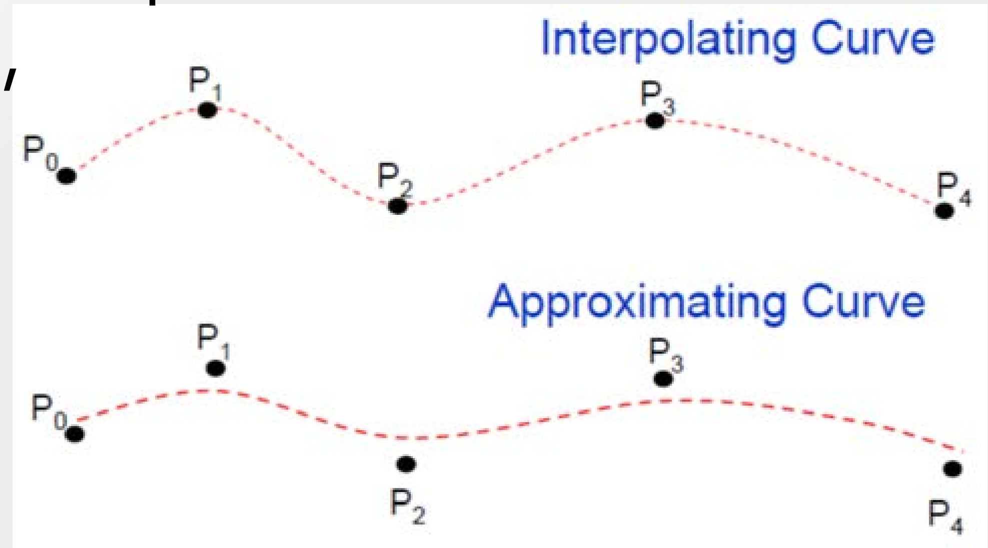


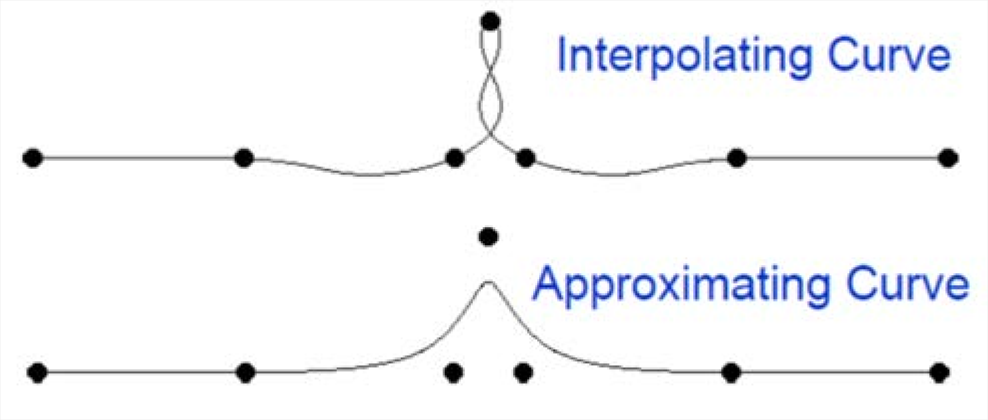




Interpolated vs. Approximated Curves

Given a set of control points $\mathbf{P}_i$ known to be on the curve, find a parametric curve that interpolates/ approximates the points

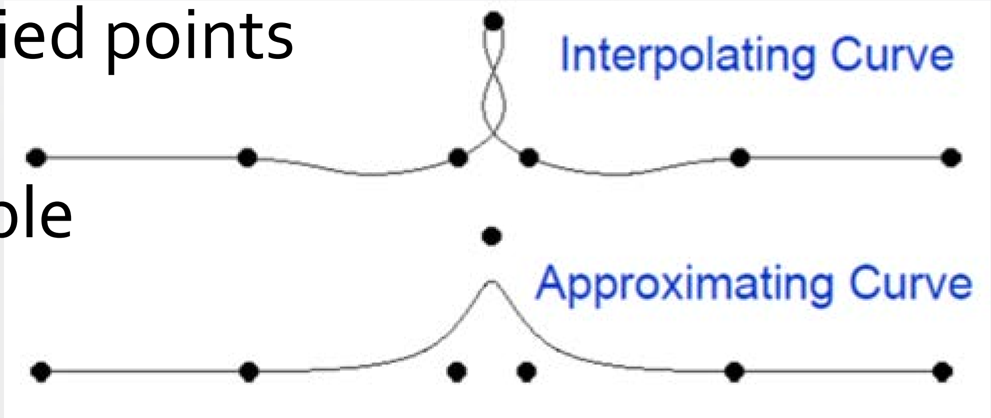




Interpolated vs. Approximated Curves

Interpolation

- Goes through all specified points
- Sounds more logical
- But can be more unstable

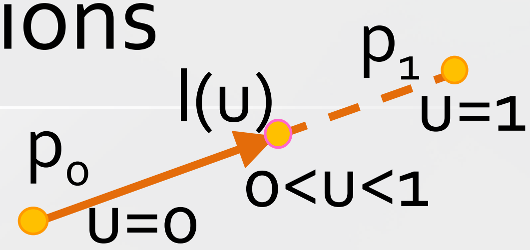


Approximation

- Does not go through all points
- Turns out to be convenient

We will do something in between.

Mathematical curve equations



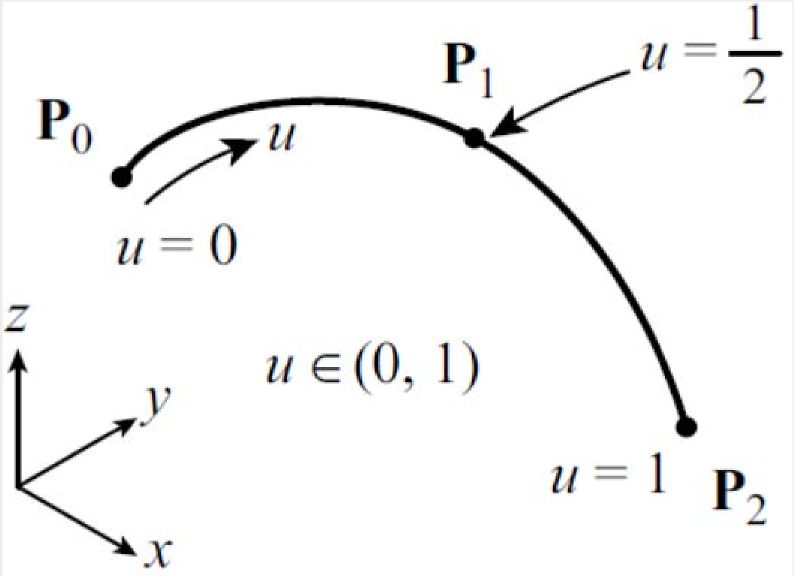
Point: $$\mathbf{p} = \begin{bmatrix} \mathbf{a}_x \\ \mathbf{a}_y \\ \mathbf{a}_z \end{bmatrix}$$

Line: $$\mathbf{l}(u) = \begin{bmatrix} \mathbf{a}_x & \mathbf{a}_y & \mathbf{a}_z \end{bmatrix}^T u + \begin{bmatrix} \mathbf{b}_x & \mathbf{b}_y & \mathbf{b}_z \end{bmatrix}^T$$

Quadratic curve: $$\mathbf{q}(u) = \begin{bmatrix} \mathbf{a}_x & \mathbf{a}_y & \mathbf{a}_z \end{bmatrix}^T u^2 + \begin{bmatrix} \mathbf{b}_x & \mathbf{b}_y & \mathbf{b}_z \end{bmatrix}^T u + \begin{bmatrix} \mathbf{c}_x & \mathbf{c}_y & \mathbf{c}_z \end{bmatrix}^T$$

$$u \in [u_s, u_e];\ v \in [0,1]$$

$$v = (u - u_s)/(u_e - u_s)$$



Cubic Polynomial Curve

Definition $(0 \le u \le 1)$

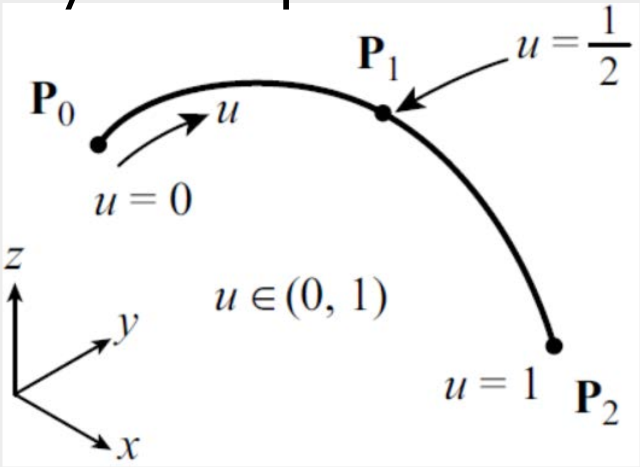
$\mathbf{P}(u) = [x(u)\ y(u)\ z(u)]^T = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 + \mathbf{a}_3 u^3$

Major Drawback

$\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ 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.

Why not quadric?

$\mathbf{P}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 \quad (0 \le u \le 1)$



Quadric Polynomial Curve

Definition quadric curve to interpolate

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

$(0 \le u \le 1)$ $\mathbf{P}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 + \mathbf{a}_3 u^3$ (cubic)

An example:

the parabola

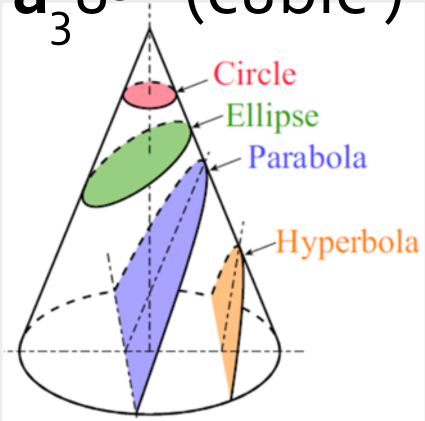
$(x = cu^2,\ y = u,\ z = 0)$

(quadric)

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

(cubic)

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



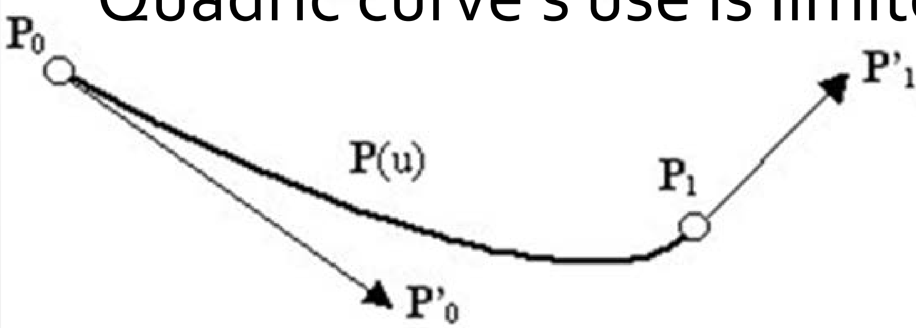
$\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ are algebraic vector coefficients.

Quadric Polynomial Curve

$\mathbf{P}(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 \quad (0 \le u \le 1)$

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

Quadric curve's use is limited.



$$
\begin{aligned}
P_0 &= P(0) = a_0 \\
P_1 &= P(1) = a_0 + a_1 + a_2 \\
P'_0 &= P'(0) = a_1 \\
P'_1 &= P'(1) = a_1 + 2a_2
\end{aligned}
$$

$$
\begin{aligned}
P_0 &= P(0) = a_0 \\
P'_0 &= P'(0) = a_1
\end{aligned}
\quad \longrightarrow \quad
\begin{aligned}
a_0 &= P_0 \\
a_1 &= P'_0
\end{aligned}
$$

$$
\begin{aligned}
P_1 &= P(1) = P_0 + P'_0 + a_2 \\
P'_1 &= P'(1) = P'_0 + 2a_2
\end{aligned}
\quad \longrightarrow \quad
\begin{aligned}
a_2 &= P_1 - P_0 - P'_0 \\
a_2 &= (P'_1 - P'_0)/2
\end{aligned}
\quad ?
$$

Parametric Cubic Curves

Cubic polynomials defining a curve in **R**^{3} have the form:

$$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x$$

$$y(u) = a_y u^3 + b_y u^2 + c_y u + d_y$$

$$z(u) = a_z u^3 + b_z u^2 + c_z u + d_z$$

where u is in [0,1]. Defining:

$$U^T(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \quad \text{and} \quad Q = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

The curve can be rewritten as:

$$\overrightarrow{p}(u) = [x(u) \;\; y(u) \;\; z(u)]^T = V^T(u) = U^T(u)\; Q$$

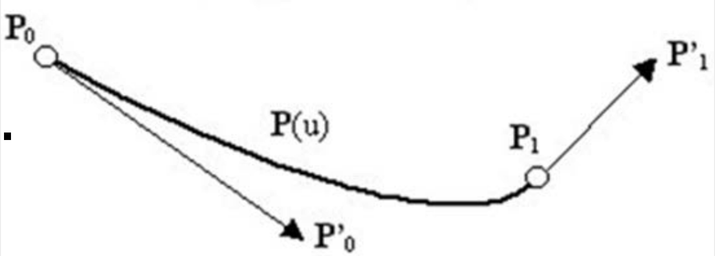
Parametric Cubic Curves

$$\overrightarrow{p}(u) = [x(u) \;\; y(u) \;\; z(u)]^T = V^T(u) = U^T(u)\; Q$$

$$U^T(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \qquad Q = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

The coefficients **Q** are unknown and should be determined. For this purpose, **we have to supply 4 geometrical constraints**.

Different types of constraints define different types of Splines.



Hermite Cubic Polynomial Curve

When the conditions are the positions and two tangent vectors of two end points, a Hermite cubic spline parametric equation results.

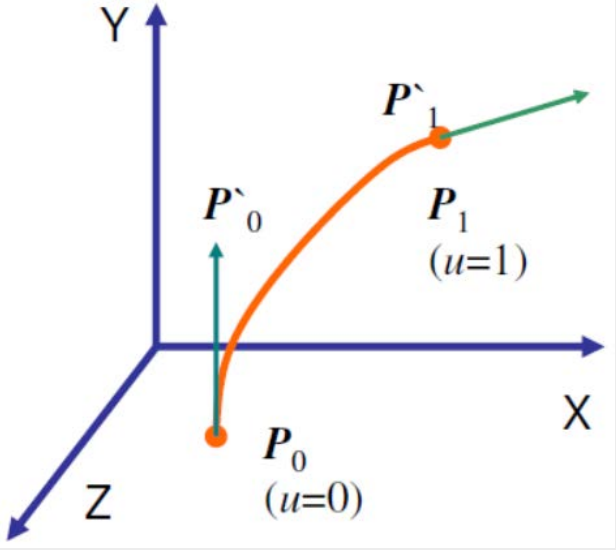
$$\vec{P}(u) = [x(u) \;\; y(u) \;\; z(u)]^T = [u^3 \;\; u^2 \;\; u \;\; 1] \begin{bmatrix} \vec{C}_3 \\ \vec{C}_2 \\ \vec{C}_1 \\ \vec{C}_0 \end{bmatrix} = U^T[C]$$

$$\vec{P}(u) = \vec{C}_3 u^3 + \vec{C}_2 u^2 + \vec{C}_1 u^1 + \vec{C}_0 = \sum_{i=0}^{3} \vec{C}_i u^i$$

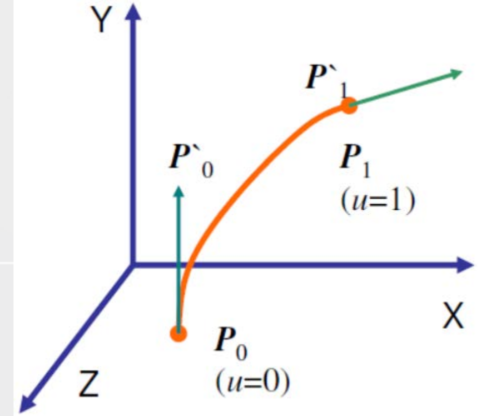
$$U = [u^3 \;\; u^2 \;\; u \;\; 1]^T \qquad (0 \le u \le 1)$$

$$C = \left[\vec{C}_3 \;\; \vec{C}_2 \;\; \vec{C}_1 \;\; \vec{C}_0\right]^T$$

$$\vec{C}_3 = \left[c_{3x} \;\; c_{3y} \;\; c_{3z}\right]^T$$



Hermite Cubic Polynomial Curve



$$\vec{P}(u)=[x(u)\ \ y(u)\ \ z(u)]^T=[u^3\ \ u^2\ \ u\ \ 1]\begin{bmatrix}\bar{C}_3\\ \bar{C}_2\\ \bar{C}_1\\ \bar{C}_0\end{bmatrix}=U^T[C]$$

$$\vec{P}(u)=\sum_{i=0}^{3}\bar{C}_i u^i=\bar{C}_3u^3+\bar{C}_2u^2+\bar{C}_1u^1+\bar{C}_0$$

$$\vec{P}`(u)=\sum_{i=0}^{3}\bar{C}_i i u^{i-1}=\begin{bmatrix}3u^2 & 2u & 1 & 0\end{bmatrix}[C]$$

$$\bar{P}(u)=U^T(u)\,C=U^T(u)\,M^{-1}\vec{V}$$

Algebraic coefficients

$$\underbrace{\begin{bmatrix}p_k\\ p_{k+1}\\ T_k\\ T_{k+1}\end{bmatrix}}_{V}=\underbrace{\begin{bmatrix}0&0&0&1\\ 1&1&1&1\\ 0&0&1&0\\ 3&2&1&0\end{bmatrix}}_{M}[C]$$

$$M^{-1}=\begin{bmatrix}2&-2&1&1\\ -3&3&-2&-1\\ 0&0&1&0\\ 1&0&0&0\end{bmatrix}$$

$$u=0\quad\begin{cases}\vec{P}_0=\bar{C}_0\\ \vec{P}_0'=\bar{C}_1\end{cases}$$

$$u=1\quad\begin{cases}\vec{P}_1=\bar{C}_3+\bar{C}_2+\bar{C}_1+\bar{C}_0\\ \vec{P}_1'=3\bar{C}_3+2\bar{C}_2+\bar{C}_1\end{cases}$$

$$\bar{C}_0=\vec{P}_0$$
$$\bar{C}_1=\vec{P}_0'$$
$$\bar{C}_2=3(\vec{P}_1-\vec{P}_0)-2\vec{P}_0'-\vec{P}_1'$$
$$\bar{C}_3=2(\vec{P}_0-\vec{P}_1)+\vec{P}_0'+\vec{P}_1'$$

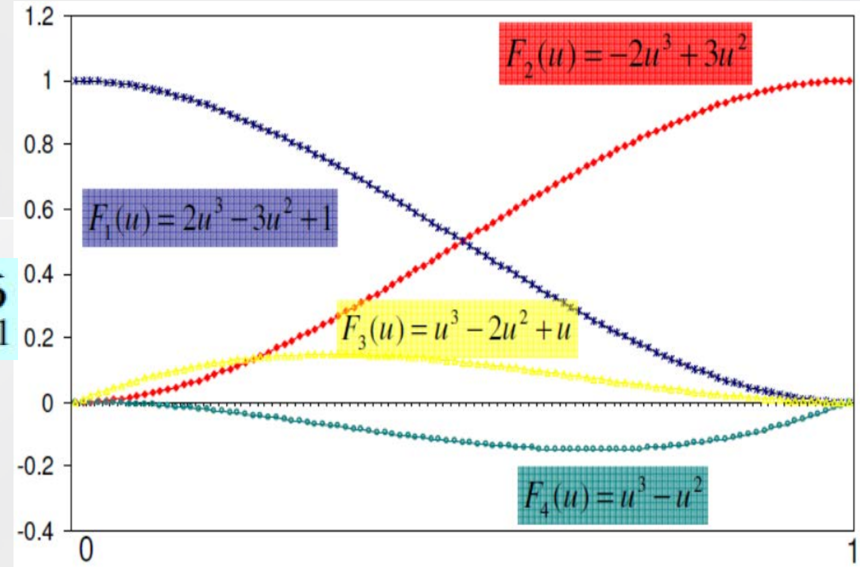
Rearrange the above cubic polynomial curve equation

$$\bar{P}(u)=(2u^3-3u^2+1)\vec{P}_0+(-2u^3+3u^2)\vec{P}_1+(u^3-2u^2+u)\vec{P}_0'+(u^3-u^2)\vec{P}_1'$$

Hermite Cubic Curve Blending Functions

$$\vec{P}(u) = (2u^3 - 3u^2 + 1)\vec{P}_0 + (-2u^3 + 3u^2)\vec{P}_1 + (u^3 - 2u^2 + u)\vec{P}_0' + (u^3 - u^2)\vec{P}_1'$$

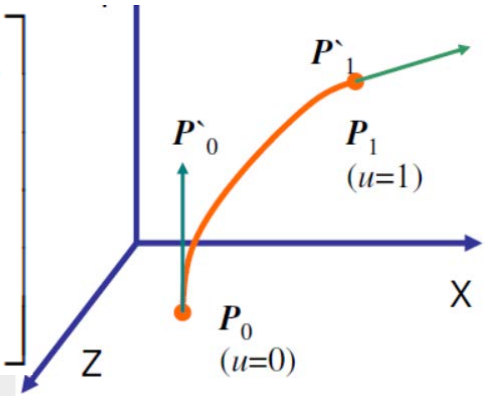
Blending (Basis) Functions



$$= F_1(u)\vec{P}_0 + F_2(u)\vec{P}_1 + F_3(u)\vec{P}`_0 + F_4(u)\vec{P}`_1$$

$$\vec{P}(u) = \left[(2u^3 - 3u^2 + 1)\ \ (-2u^3 + 3u^2)\ \ (u^3 - 2u^2 + u)\ \ (u^3 + u^2)\right] \begin{bmatrix} \vec{P}_0 \\ \vec{P}_1 \\ \vec{P}_0' \\ \vec{P}_1' \end{bmatrix}$$

Geometry Vector



$$\vec{P}(u) = [x(u)\ \ y(u)\ \ z(u)]^T = \left[u^3\ \ u^2\ \ u\ \ 1\right] \begin{bmatrix} \vec{C}_3 \\ \vec{C}_2 \\ \vec{C}_1 \\ \vec{C}_0 \end{bmatrix} = U^T[C]$$

$$\vec{P}(u) = \left[u^3\ \ u^2\ \ u\ \ 1\right] \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{P}_0 \\ \vec{P}_1 \\ \vec{P}_0' \\ \vec{P}_1' \end{bmatrix} = U^T\ [M_H]\ \vec{V}$$

Derivatives of Hermite Cubic Splines

Hermite curve

$$\vec{P}(u) = (2u^3 - 3u^2 + 1)\vec{P}_0 + (-2u^3 + 3u^2)\vec{P}_1 + (u^3 - 2u^2 + u)\vec{P}_0' + (u^3 - u^2)\vec{P}_1'$$

$$0 \le u \le 1$$

First derivative of curve, tangent of curve

$$\vec{P}'(u) = (6u^2 - 6u)\vec{P}_0 + (-6u^2 + 6u)\vec{P}_1 + (3u^2 - 4u + 1)\vec{P}_0' + (3u^2 - 2u)\vec{P}_1'$$

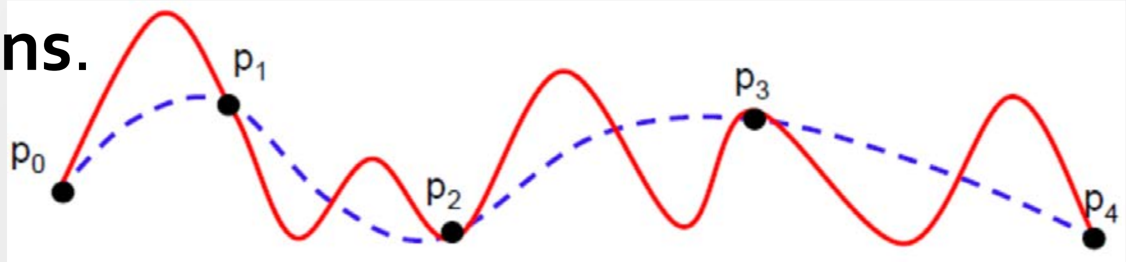
Second derivative of curve, inverse of radius curvature

$$\vec{P}''(u) = (12u - 6)\vec{P}_0 + (-12u + 6)\vec{P}_1 + (6u - 4)\vec{P}_0' + (6u - 2)\vec{P}_1'$$

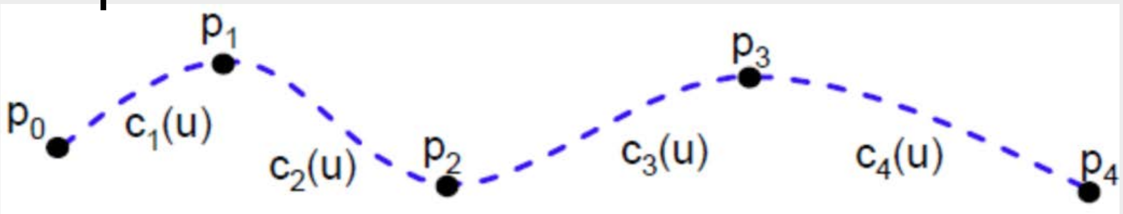
High-order Parametric Polynomials

Polynomial interpolation has several **disadvantages**:

Polynomial coefficients are geometrically meaningless.
Polynomials of high degree introduce unwanted **(wiggles) oscillations**.



Polynomials of **low degree** give **little flexibility**.
Solution: Polynomial Splines



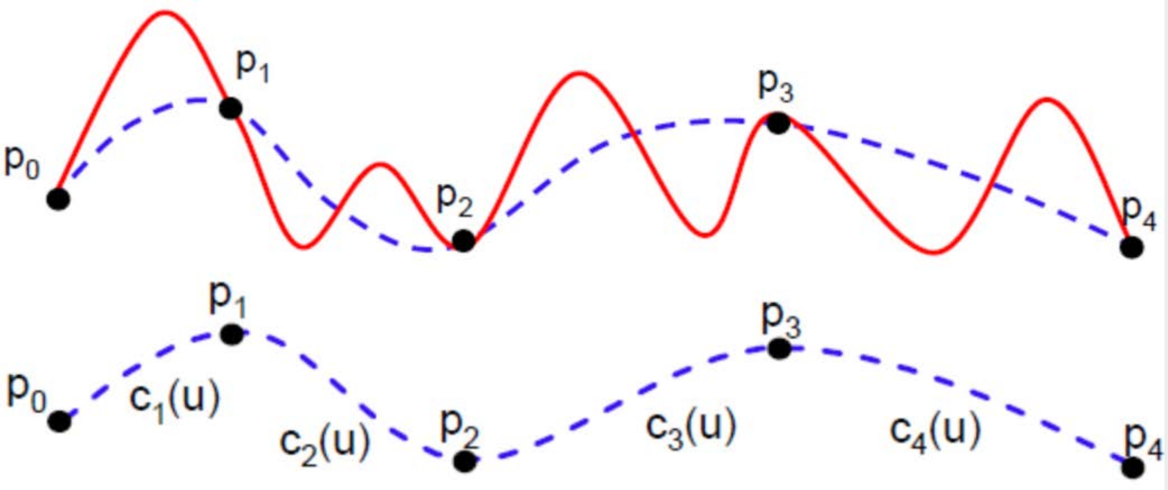
Parametric Polynomials

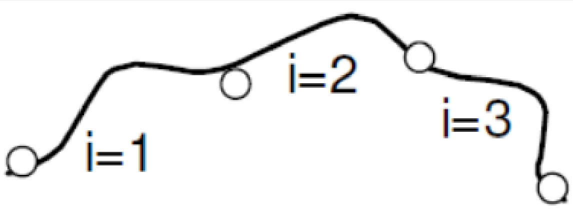
High-order polynomials

$$\mathrm{p}(u) = \begin{bmatrix} \mathbf{a}_{0,x} \\ \mathbf{a}_{0,y} \\ \mathbf{a}_{0,z} \end{bmatrix} + \dots + \begin{bmatrix} \mathbf{a}_{i,x} \\ \mathbf{a}_{i,y} \\ \mathbf{a}_{i,z} \end{bmatrix} u^i + \dots + \begin{bmatrix} \mathbf{a}_{n,x} \\ \mathbf{a}_{n,y} \\ \mathbf{a}_{n,z} \end{bmatrix} u^n$$

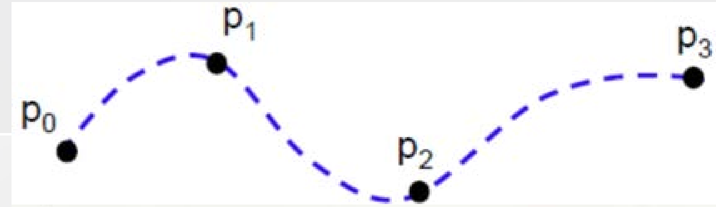
No intuitive insight for the curved shape.

Difficult for piecewise smooth curves.





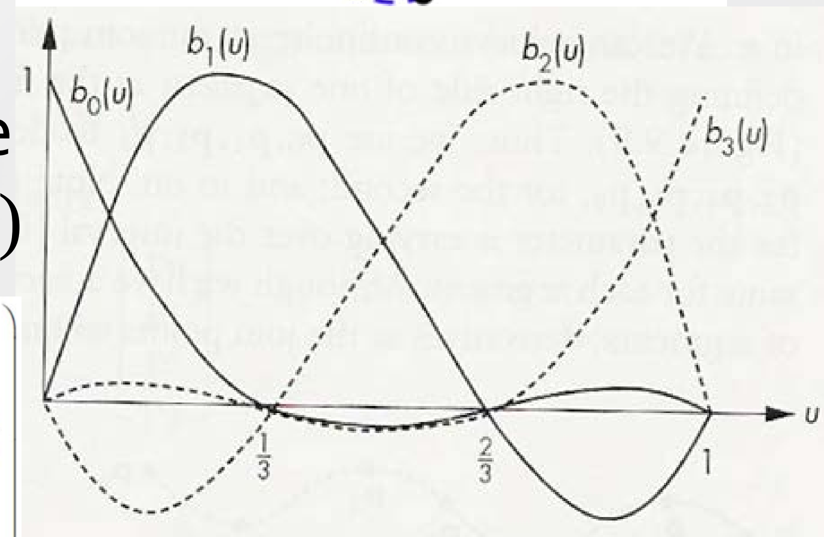
Parametric Polynomials



Polynomial interpolation curve using equal parameter distance

$u_i = i/n$, $0 \le u \le 1$, $i=0..n$ $\quad \mathbf{p_i} = \mathbf{p}(u_i)$

$$\mathbf{p}(u) = \sum_{i=0}^{n} b_i(u)\mathbf{p}_i$$



Blending polynomials for interpolation.

$$\mathrm{p(u)} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \mathrm{M_H} \begin{bmatrix} \mathbf{P_0} \\ \mathbf{P_1} \\ \mathbf{P_2} \\ \mathbf{P_3} \end{bmatrix}$$

$$\mathrm{M_H} = \begin{pmatrix} -\frac{9}{2} & \frac{27}{2} & -\frac{27}{2} & \frac{9}{2} \\ 9 & -\frac{45}{2} & 18 & -\frac{9}{2} \\ -\frac{11}{2} & 9 & -\frac{9}{2} & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{p(u)} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \cdot \begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \\ a_{01} & a_{02} & a_{03} \end{pmatrix}$$

$$b_0(u) = -\frac{9}{2}\left(u - \frac{1}{3}\right)\left(u - \frac{2}{3}\right)(u-1),$$

$$b_1(u) = \frac{27}{2}u\left(u - \frac{2}{3}\right)(u-1),$$

$$b_2(u) = -\frac{27}{2}u(u - \frac{1}{3})(u-1),$$

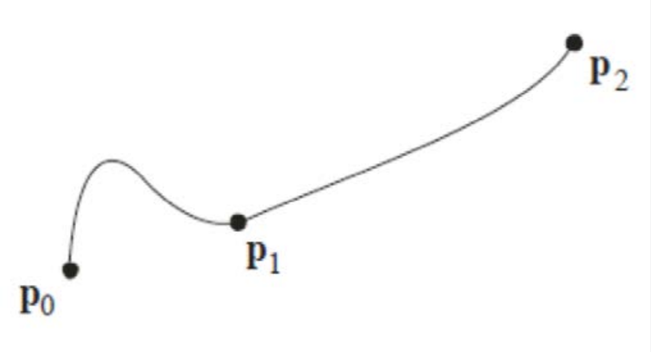
$$b_3(u) = \frac{9}{2}u\left(u - \frac{1}{3}\right)\left(u - \frac{2}{3}\right).$$

$$9 \cdot u^2 - \frac{9 \cdot u^3}{2} - \frac{11 \cdot u}{2} + 1$$

$$\frac{27 \cdot u^3}{2} - \frac{45 \cdot u^2}{2} + 9 \cdot u$$

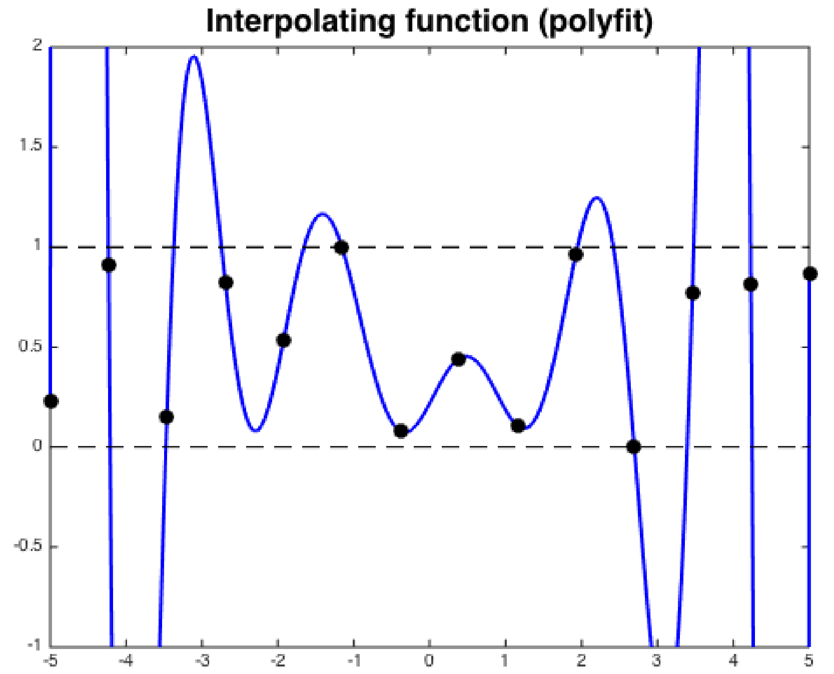
$$18 \cdot u^2 - \frac{27 \cdot u^3}{2} - \frac{9 \cdot u}{2}$$

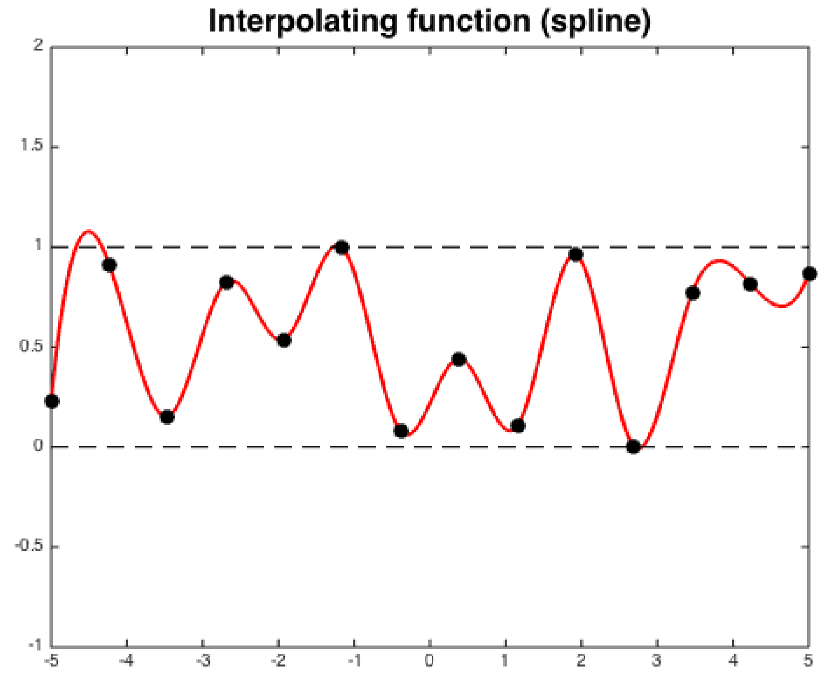
$$\frac{9 \cdot u^3}{2} - \frac{9 \cdot u^2}{2} + u$$



Motivation of Splines

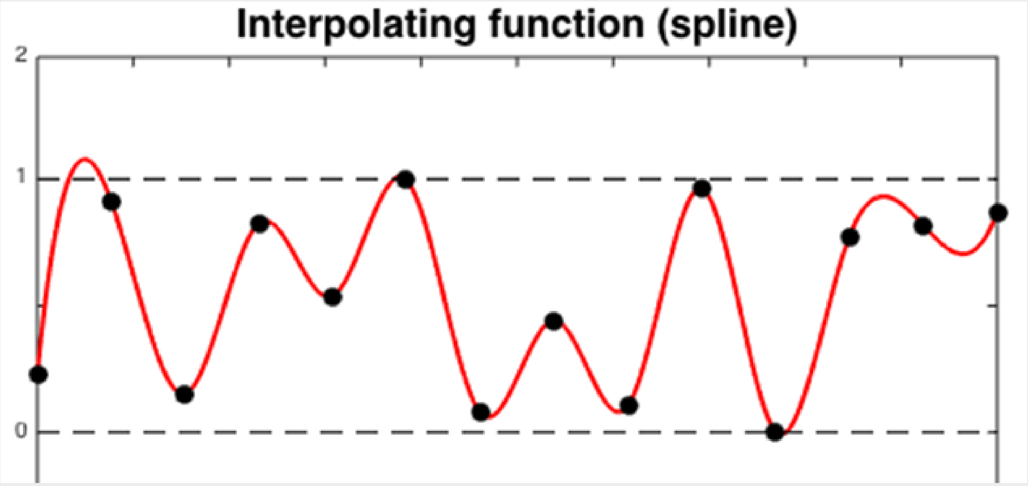
The main motivation for using splines instead of a single polynomial is **to avoid the oscillations in high-degree** interpolating polynomials that can occur between interpolation points. A spline-fit curve may have undesirable infliction points.

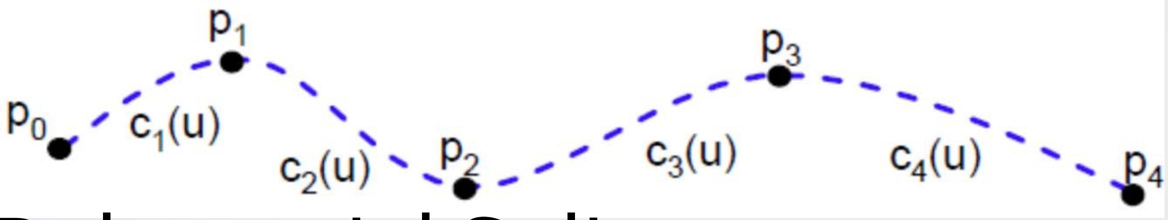




Why Cubic Polynomials

- Lowest degree for specifying curve in space
- Lowest degree for specifying points to interpolate and tangents to interpolate
- Commonly used in computer graphics
- Lower degree has too little flexibility
- Higher degree is unnecessarily complex, exhibit undesired wiggles





Piecewise Polynomial Splines

Piecewise, low degree polynomial curves for different parts of the curve with continuous joints

$$C(u) = \begin{pmatrix} x(u) \\ y(u) \\ z(u) \end{pmatrix} = \sum_i \hat{C}_i(u)$$

$$\text{where} \quad \hat{C}_i(u) = \begin{cases} C_i(u) & \text{if} \quad u \in [u_i, u_{i+1}] \\ 0 & \text{otherwise} \end{cases}$$

Advantages:

- Low-degree, Flexible, Rich representation
- Geometrically meaning coefficients
- Local effects
- Interactive sculpting capabilities

Disadvantages:

- How to ensure smoothness at the joints (continuity)

Hermite Curves

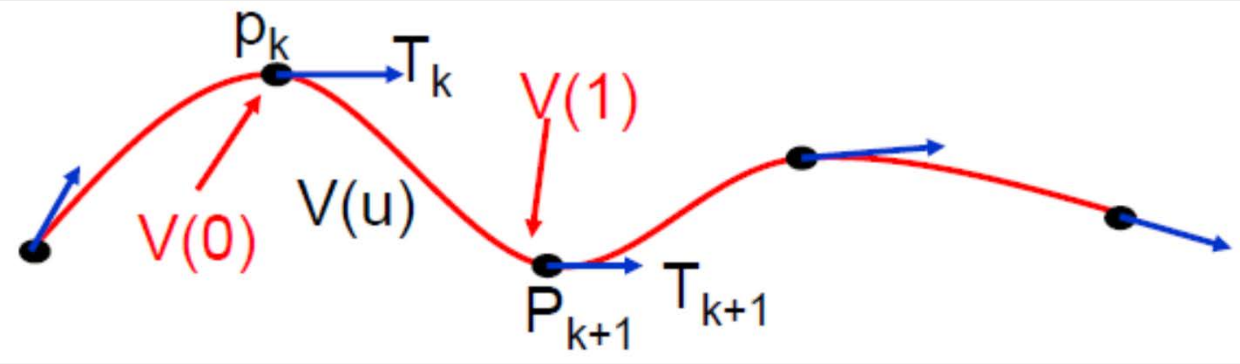
Assume we have n control points $\{p_k\}$ with their tangents $\{T_k\}$

V(u) represents a parametric cubic function for the section between p_k and p_{k+1}

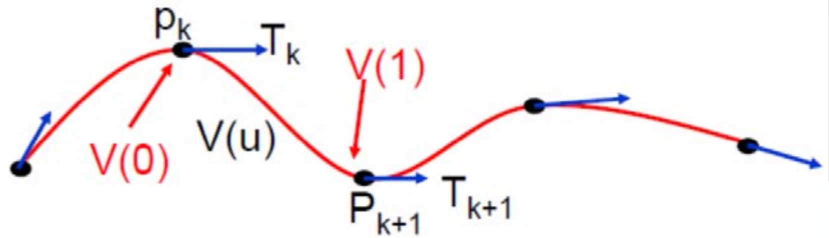
For V(u) we have the following geometric constraints:

$V(0) = p_k$; $V(1) = p_{k+1}$

$V'(0) = T_k$; $V'(1) = T_{k+1}$



Hermite Curves



$$\vec{p}(u) = [x(u) \quad y(u) \quad z(u)]^T = V^T(u) = U^T(u)\,Q \qquad Q = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

Since

$$V^T(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} Q$$

we have that

$$(V')^T(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} Q$$

We can write the constraints in a matrix form:

$$G = MQ \quad \Rightarrow \quad \underbrace{\begin{bmatrix} p_k \\ p_{k+1} \\ T_k \\ T_{k+1} \end{bmatrix}}_{G} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}}_{M} Q$$

And thus $\quad V^T(u) = U^T(u)Q = U^T(u)M^{-1}G$

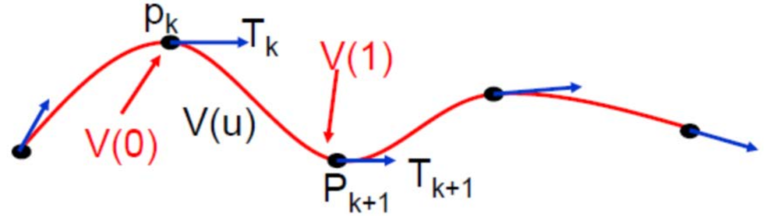
Where

$$M^{-1} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$[x(u) \quad y(u) \quad z(u)] = V^T(u) = U^T(u)\, Q = U^T(u)\, M^{-1} G$$

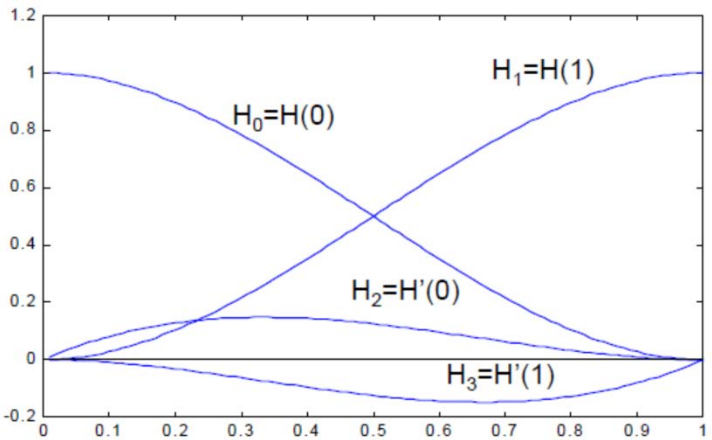
Hermite Curves

$$V(u) = [u^3 \quad u^2 \quad u \quad 1] \underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{} \underbrace{\begin{bmatrix} p_k \\ p_{k+1} \\ T_k \\ T_{k+1} \end{bmatrix}}_{\text{Geometry matrix}}$$



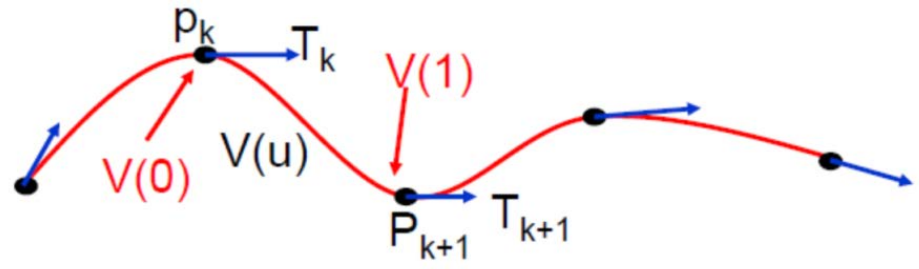
$$V(u) = \overbrace{\begin{bmatrix} 2u^3 - 3u^2 + 1 \\ -2u^3 + 3u^2 \\ u^3 - 2u^2 + u \\ u^3 - u^2 \end{bmatrix}}^{\text{Blending functions}}{}^{T} \begin{bmatrix} p_k \\ p_{k+1} \\ T_k \\ T_{k+1} \end{bmatrix} =$$

$$= \begin{bmatrix} H_0(u) \\ H_1(u) \\ H_2(u) \\ H_3(u) \end{bmatrix}^T \begin{bmatrix} p_k \\ p_{k+1} \\ T_k \\ T_{k+1} \end{bmatrix}$$



Hermite blending functions

Hermite Curves



Properties: The Hermite curve is composed of a linear combinations of tangents and locations (for each u). Alternatively, the curve is a linear combination of Hermite basis functions (the matrix M).

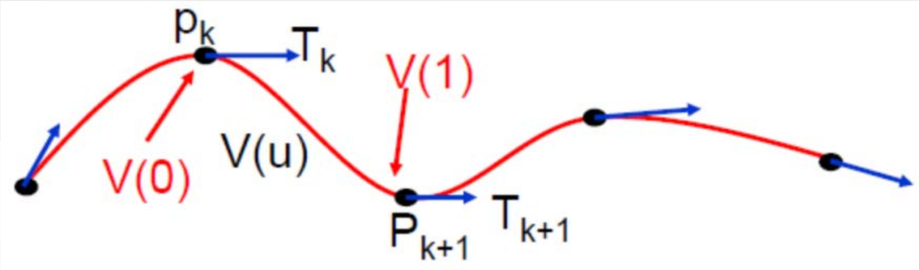
It can be used to create geometrically intuitive curves.

The piecewise interpolation scheme is C^1 continuous.

The blending functions have local support; changing a control point or a tangent vector, changes its local neighborhood while leaving the rest unchanged.

Main Drawback: Requires the specification of the tangents. This information is not always available.

Hermite Curves



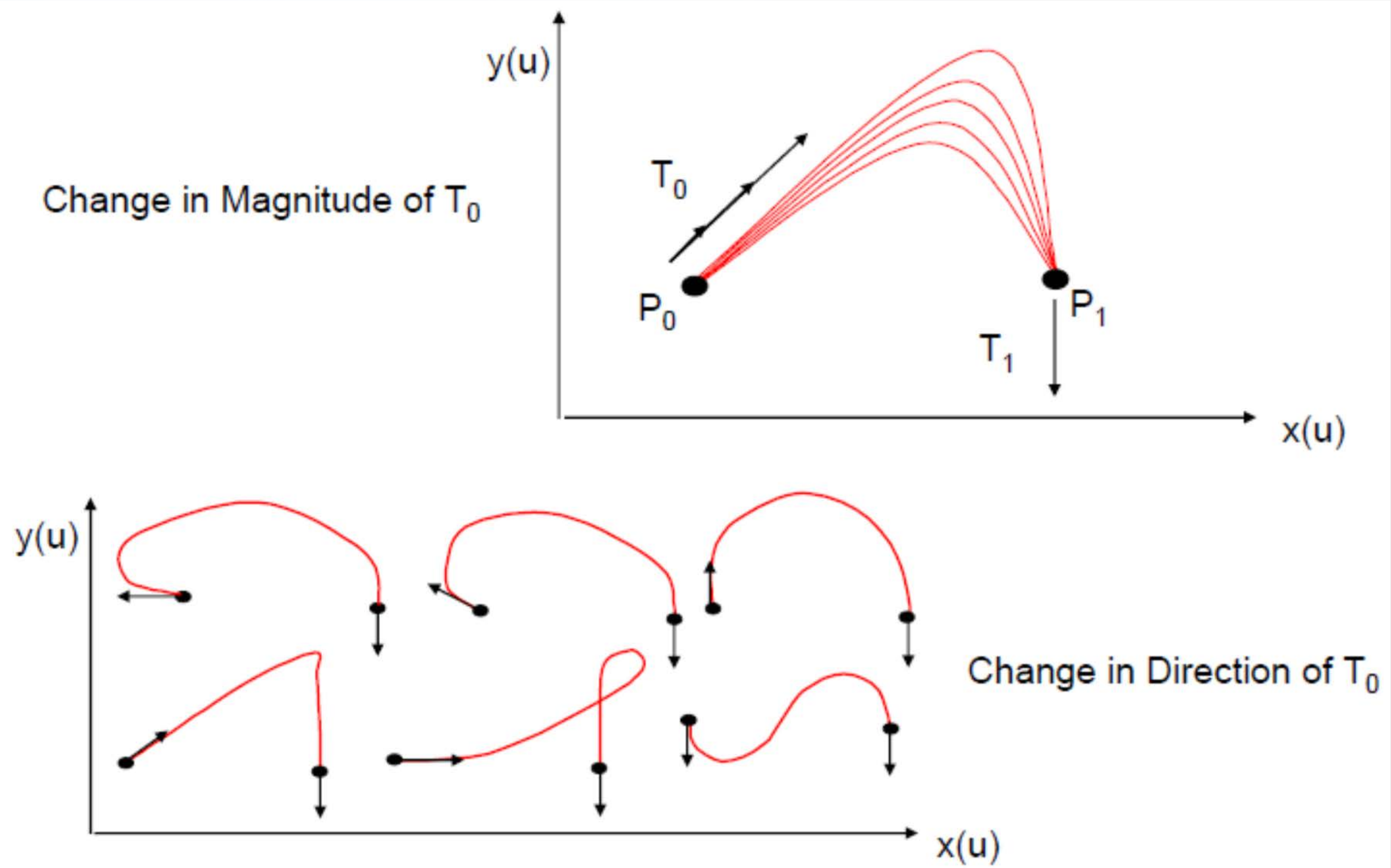
Benefits

If the designer changes $\mathbf{P}_0$ or $\mathbf{P}_1$, he/she immediately knows what effect it will have on the shape – the end point moves.

Similarly, if he modifies P_0' or P_1', he/she knows at least the tangent direction at that end point will change accordingly.

Deficiency It is not easy to predict curve shape according to changes in size of the tangents $\mathbf{P}_0'$ or $\mathbf{P}_1'$.

The effect of tangents on a Hermite Curve

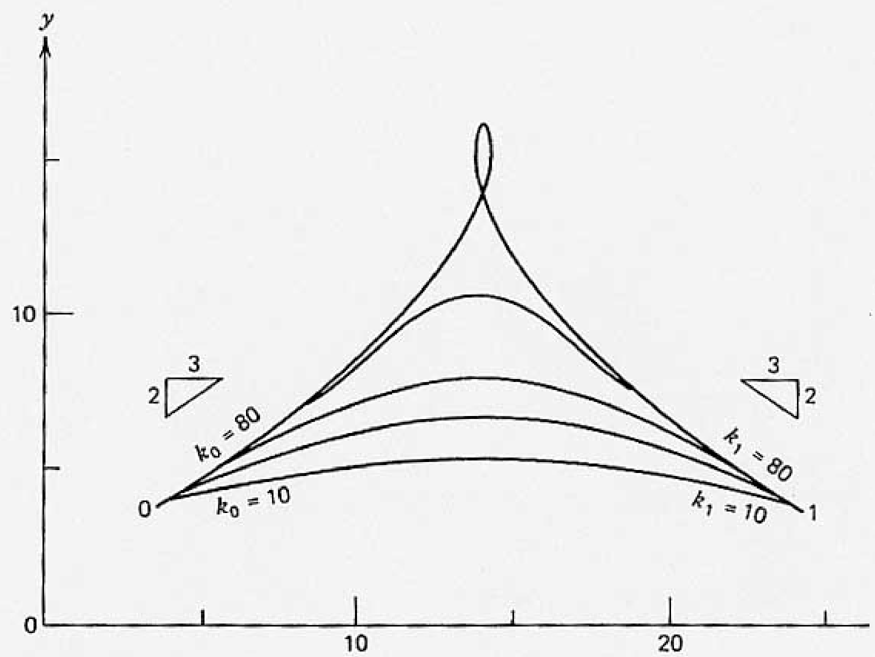


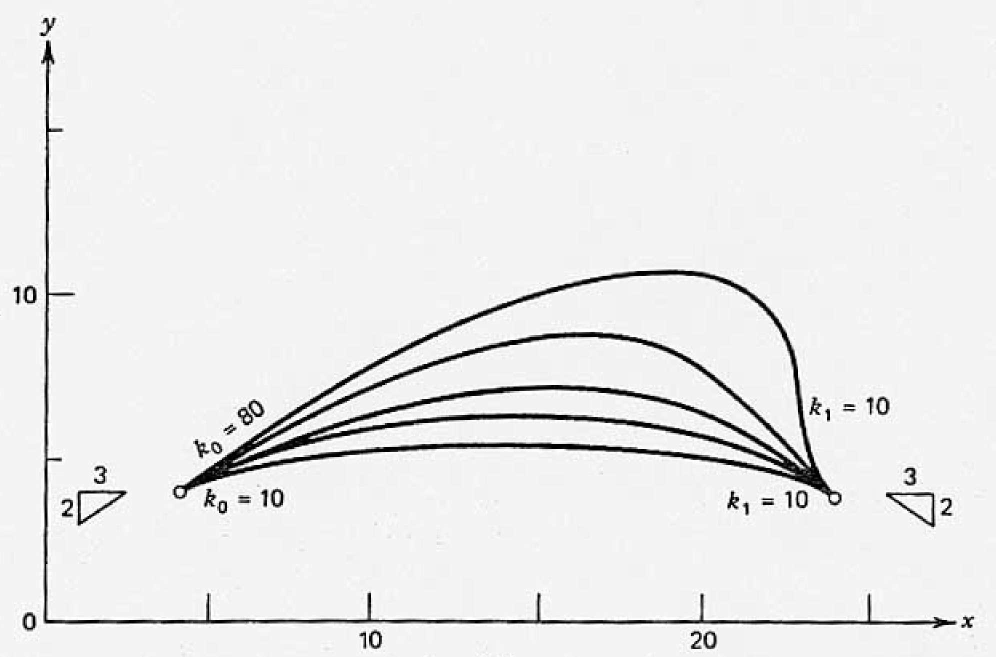
Effect tangents' Magnitudes on a Hermite

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.





Parametric polynomial Cubic Curve

$$x = a_{31} \cdot u^3 + a_{21} \cdot u^2 + a_{11} \cdot u + a_{01}$$

$$y = a_{32} \cdot u^3 + a_{22} \cdot u^2 + a_{12} \cdot u + a_{02}$$

$$z = a_{33} \cdot u^3 + a_{23} \cdot u^2 + a_{13} \cdot u + a_{03}$$

a_{ij} : 12 degrees of freedom (dof).

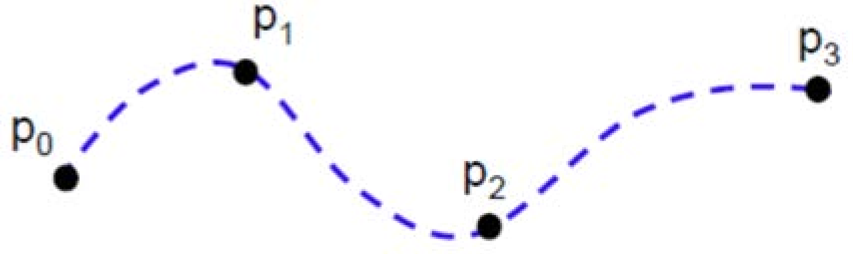
$$u = 0 .. 1$$

$$p(u) = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \cdot \begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \\ a_{01} & a_{02} & a_{03} \end{pmatrix}$$

$$p(u) = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \cdot A$$

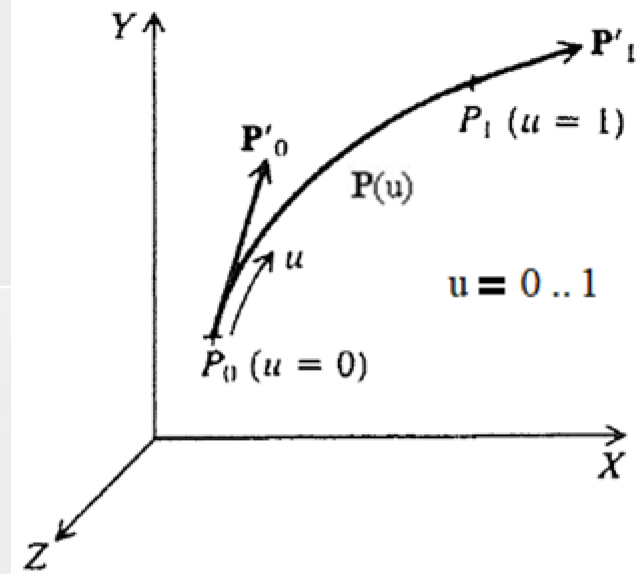
$$\frac{d}{du}\left(u^3 + u^2 + u + 1\right) \rightarrow 3 \cdot u^2 + 2 \cdot u + 1$$

$$\frac{d}{du} p(u) = p'(u) = \begin{pmatrix} 3 \cdot u^2 & 2 \cdot u & 1 & 0 \end{pmatrix} \cdot A$$



$$A := \begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \\ a_{01} & a_{02} & a_{03} \end{pmatrix}$$

Hermite Curve



$$p(u) = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \cdot \begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} p(0) \\ p(1) \\ p'(0) \\ p'(1) \end{pmatrix}$$

$$\mathbf{P}(u) = \mathbf{U}^T[M_H]\mathbf{V}$$

$$p(u) = \left(2 \cdot u^3 - 3 \cdot u^2 + 1\right) \cdot p(0) + \left(-2 \cdot u^3 + 3 \cdot u^2\right) \cdot p(1) + \left(u^3 - 2 \cdot u^2 + u\right) \cdot p'(0) + \left(u^3 - u^2\right) \cdot p'(1)$$

Hermite Blending functions

Hermite Curve

$$p(u) = \begin{pmatrix} F_1 & F_2 & F_3 & F_4 \end{pmatrix} \cdot \begin{pmatrix} p(0) \\ p(1) \\ p'(0) \\ p'(1) \end{pmatrix}$$

$$p'(u) := U^T \cdot M'_H \cdot V$$

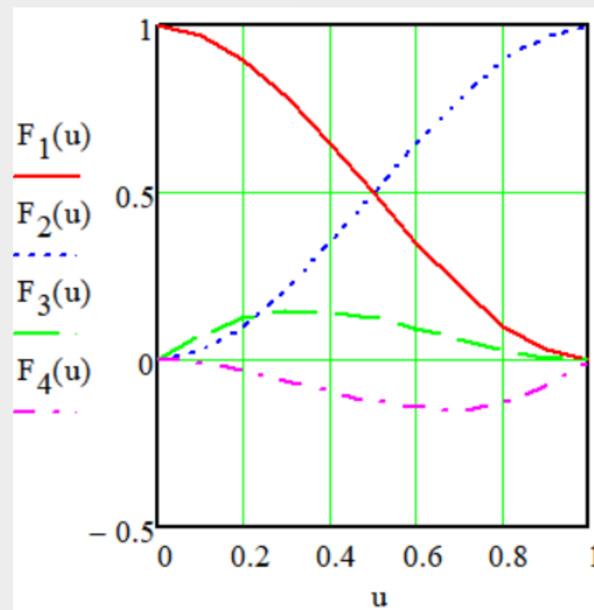
$$M'_H := \begin{pmatrix} 0 & 0 & 0 & 0 \\ 6 & -6 & 3 & 3 \\ -6 & 6 & -4 & -2 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$F_1 = 2 \cdot u^3 - 3 \cdot u^2 + 1$$

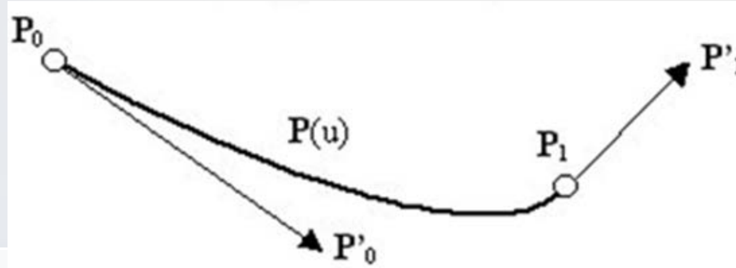
$$F_2 = -2 \cdot u^3 + 3 \cdot u^2$$

$$F_3 = u^3 - 2 \cdot u^2 + u$$

$$F_4 = u^3 - u^2$$



Hermite Curve

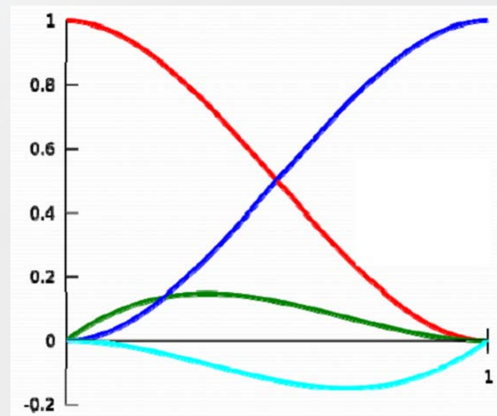


$$p(u) = \left(2\cdot u^3 - 3\cdot u^2 + 1\right)\cdot p(0) + \left(-2\cdot u^3 + 3\cdot u^2\right)\cdot p(1) + \left(u^3 - 2\cdot u^2 + u\right)\cdot p'(0) + \left(u^3 - u^2\right)\cdot p'(1)$$

$$\mathbf{V} = [\mathbf{P}_0 \quad \mathbf{P}_1 \quad \mathbf{P}'_0 \quad \mathbf{P}'_1]^T$$

$$p(u) = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \cdot M_H \cdot \begin{pmatrix} p(0) \\ p(1) \\ p'(0) \\ p'(1) \end{pmatrix}$$

$$\mathbf{P}(u) = \mathbf{U}^T[M_H]\mathbf{V}$$

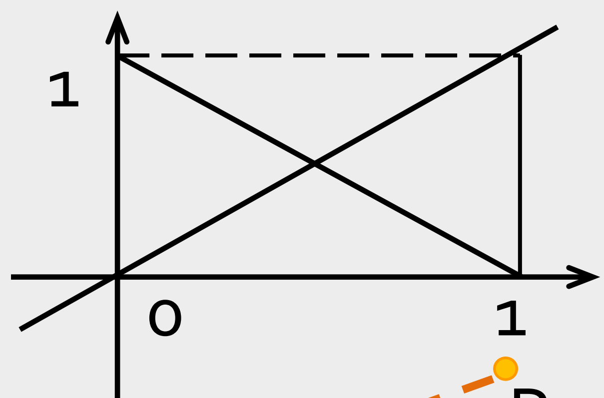


Hermite Basis Blending Functions

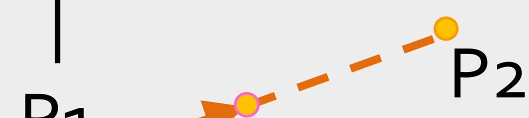
V is geometry vector (boundary conditions)

Hermite matrix

$$M_H = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

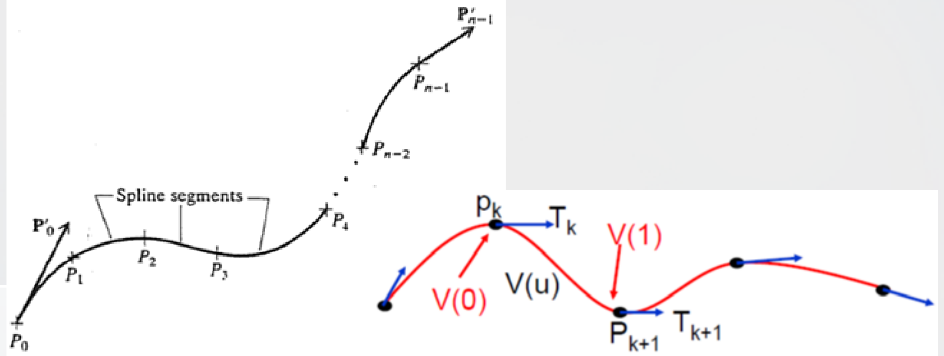


Linear blending functions for line



P(t) = (1 - t) P1+ t P2

Piecewise Hermite Cubic Spline Curve

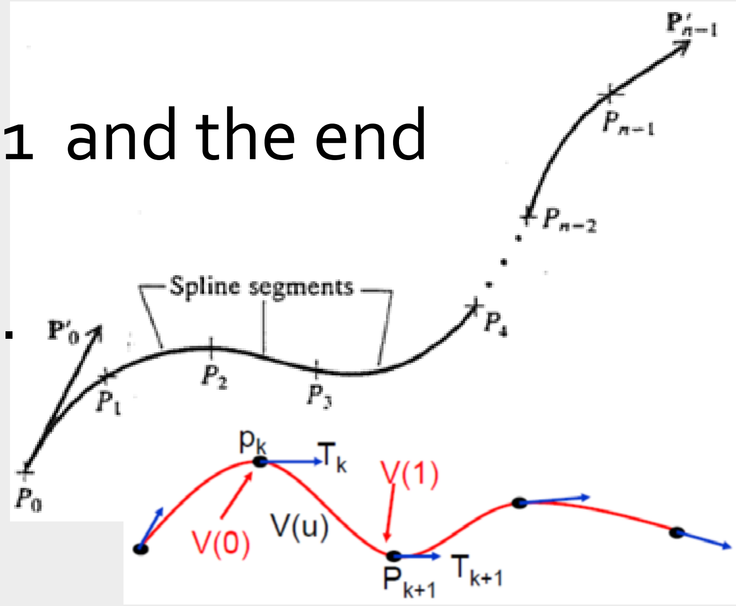


Rather than use a very high degree curve to interpolate a large number of points, it is more common to **break the curve** up into several simple curves. For example, a large complex curve could be broken into cubic curves, and would therefore be a **piecewise cubic curve**.

For the entire curve to look smooth and continuous, it is necessary to maintain **C^1 continuity** across segments, meaning that the **position** and **tangents** must match at the endpoints. For smoother looking curves, it is best to maintain the **C^2 continuity** as well.

Piecewise Hermite Cubic Spline Curve

The above equation is for one cubic spline segment.
It can be generalized for any two adjacent spline segments of a spline curve that are to fit a given number of data points.
This introduces the problem of blending or joining cubic spline segments.
Given a set of n points P0, P1,...Pn-1 and the end tangent vectors P'0, P'n-1 connect the points with a cubic spline curve.



Hermite Cubic Spline Curve

Fit a given number of data points and two end tangent vectors. $P_0 \;\; P_1 \; \cdots \; P_{n-1} \qquad P'_0 \qquad P'_{n-1}$

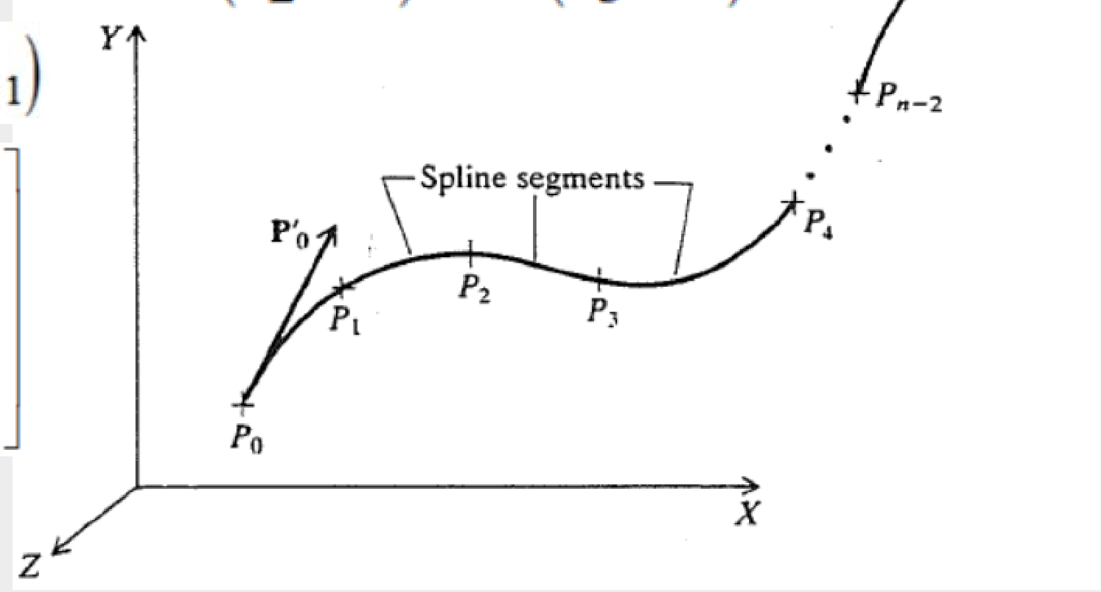
For curvature continuity between the first two segments, we can write $p''(u_1 = 1) = p''(u_2 = 0)$

For 2'nd and 3'rd segm. : $p''(u_2 = 1) = p''(u_3 = 0)$

$$P'_{i-1} + 4 \cdot P'_i + P'_{i+1} = 3 \cdot \left(P_{i+1} - P_{i-1}\right)$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} P'_0 \\ P'_{m-3} \\ P'_{m-2} \\ P'_{m-1} \end{pmatrix} = \begin{bmatrix} P'_0 \\ 3 \cdot \left(P_{m-2} - P_0\right) \\ 3 \cdot \left(P_{m-1} - P_{m-3}\right) \\ P'_{m-1} \end{bmatrix}$$

$$P'_i = M_{Gs}^{-1} \cdot G_{Cs}$$



$$p(u) = \left(2\cdot u^3 - 3\cdot u^2 + 1\right)\cdot p(0) + \left(-2\cdot u^3 + 3\cdot u^2\right)\cdot p(1) + \left(u^3 - 2\cdot u^2 + u\right)\cdot p'(0) + \left(u^3 - u^2\right)\cdot p'(1)$$

Hermite Cubic Spline Curve

$$p(u) = U^T \cdot M_H \cdot V$$

$$p'(u) = U^T \cdot M'_H \cdot V$$

$$p''(u) = U^T \cdot M''_H \cdot V$$

$$M'_H = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 6 & -6 & 3 & 3 \\ -6 & 6 & -4 & -2 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad V = \begin{pmatrix} P_0 \\ P_1 \\ P'_0 \\ P'_1 \end{pmatrix} \quad M''_H = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 12 & -12 & 6 & 6 \\ -6 & 6 & -4 & -2 \end{pmatrix}$$

$$p'(u) = \frac{d}{du}p(u) = -P0\cdot\left(6\cdot u - 6\cdot u^2\right) + P1\cdot\left(6\cdot u - 6\cdot u^2\right) + P'0\cdot\left(3\cdot u^2 - 4\cdot u + 1\right) - P'1\cdot\left(2\cdot u - 3\cdot u^2\right)$$

$$p''(u) = \frac{d^2}{du^2}p(u) = P0\cdot(12\cdot u - 6) - P1\cdot(12\cdot u - 6) + P'0\cdot(6\cdot u - 4) + P'1\cdot(6\cdot u - 2)$$

$$p''\left(u_1 = 1\right) = p''\left(u_2 = 0\right)$$

$$p''(1) = 6\cdot P0 - 6\cdot P1 + 2\cdot P'0 + 4\cdot P'1$$

$$p''(0) = -6\cdot P1 + 6\cdot P_2 - 4\cdot P'1 - 2\cdot P'_2$$

$$6\cdot P0 - 6\cdot P1 + 2\cdot P'0 + 4\cdot P'1 = -6\cdot P1 + 6\cdot P_2 - 4\cdot P'1 - 2\cdot P'_2$$

$$P'1 = \frac{-3}{4}\cdot P0 - \frac{1}{4}\cdot P'0 + \frac{3}{4}\cdot P_2 - \frac{1}{4}\cdot P'_2$$

$$P'1 = \frac{-3}{4} \cdot P0 - \frac{1}{4} \cdot P'0 + \frac{3}{4} \cdot P_2 - \frac{1}{4} \cdot P'_2$$

Hermite Cubic Spline Curve

for next cubic spline segments $p''(u_2 = 1) = p''(u_3 = 0)$

$$6 \cdot P1 - 6 \cdot P_2 + 2 \cdot P'1 + 4 \cdot P'_2 = -6 \cdot P_2 + 6 \cdot P_3 - 4 \cdot P'_2 - 2 \cdot P'_3$$

$$P'_2 = \frac{-3}{4} \cdot P1 - \frac{1}{4} \cdot P'1 + \frac{3}{4} \cdot P_3 - \frac{1}{4} \cdot P'_3 \qquad P'_2 = \frac{P0}{5} - \frac{4 \cdot P1}{5} + \frac{P'0}{15} - \frac{P_2}{5} + \frac{4 \cdot P_3}{5} - \frac{4 \cdot P'_3}{15}$$

If we have m-1 segments on cubic spline defined by Po . . . Pm-1 points.

a) if the end point tangents are known

b) and the second derivatives at Po . . . Pm-1 end points are equal to 0,

this curve is named as **Natural Cubic Spline**.

$$P'1 = \frac{-3}{4}\cdot P0 - \frac{1}{4}\cdot P'0 + \frac{3}{4}\cdot P_2 - \frac{1}{4}\cdot P'_2$$

Hermite Cubic Spline Curve

$$p''(u_1 = 1) = p''(u_2 = 0)$$

$$p''(u_2 = 1) = p''(u_3 = 0)$$

$$P'_{i-1} + 4\cdot P'_i + P'_{i+1} = 3\cdot(P_{i+1} - P_{i-1})$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} P'_0 \\ P'_1 \\ P'_2 \\ P'_3 \\ P'_{m-2} \\ P'_{m-1} \end{pmatrix} = \begin{bmatrix} P'_0 \\ 3\cdot(P_2 - P_0) \\ 3\cdot(P_3 - P_1) \\ 3\cdot(P_{m-2} - P_2) \\ 3\cdot(P_{m-1} - P_{m-3}) \\ P'_{m-1} \end{bmatrix}$$

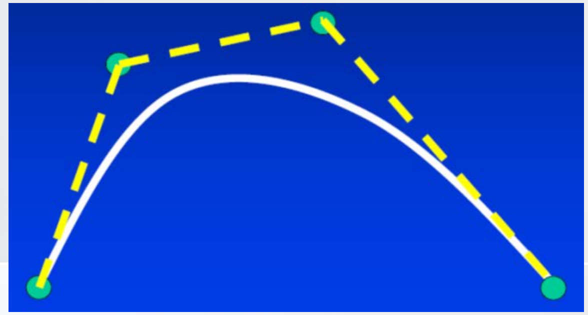
$$\begin{pmatrix} P'_0 \\ P'_{m-3} \\ P'_{m-2} \\ P'_{m-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} \cdot \begin{bmatrix} P'_0 \\ 3\cdot(P_{m-2} - P_0) \\ 3\cdot(P_{m-1} - P_{m-3}) \\ P'_{m-1} \end{bmatrix}$$

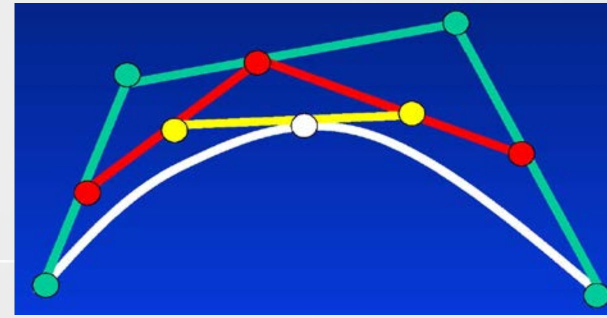
$$\begin{pmatrix} P'_0 \\ P'_1 \\ P'_2 \\ P'_3 \\ P'_{m-2} \\ P'_{m-1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}^{-1} \cdot \begin{bmatrix} P'_0 \\ 3\cdot(P_2 - P_0) \\ 3\cdot(P_3 - P_1) \\ 3\cdot(P_{m-2} - P_2) \\ 3\cdot(P_{m-1} - P_{m-3}) \\ P'_{m-1} \end{bmatrix}$$

$$P'_i = M_{Gs}^{-1}\cdot G_{Cs}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 15 & 0 & 0 & 0 \\ -4 & 4 & -1 & 1 \\ 1 & -1 & 4 & -4 \\ 0 & 0 & 0 & 15 \end{pmatrix} \frac{1}{15}$$

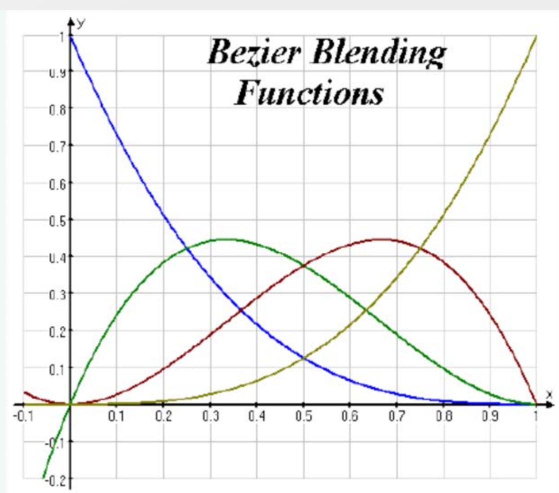
Bezier Curves





A Bézier curve only interpolates the first and last control points ($\mathbf{p}_0$, $\mathbf{p}_n$) and the order of the polynomial is defined by the number of control points considered, e.g.,

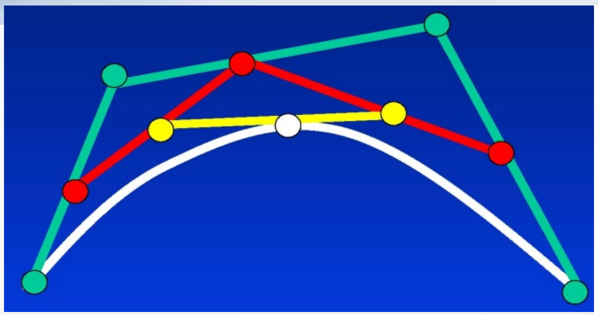
$$\mathbf{p}(u) = (1-u)^3\mathbf{p}_0 + 3u(1-u)^2\mathbf{p}_1 + 3u^2(1-u)\mathbf{p}_2 + u^3\mathbf{p}_3$$



$$\mathrm{p(u)} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \mathrm{M_B} \begin{bmatrix} \mathrm{P_1} \\ \mathrm{P_2} \\ \mathrm{P_3} \\ \mathrm{P_4} \end{bmatrix}$$

$$\mathrm{M_B} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Bezier Curves



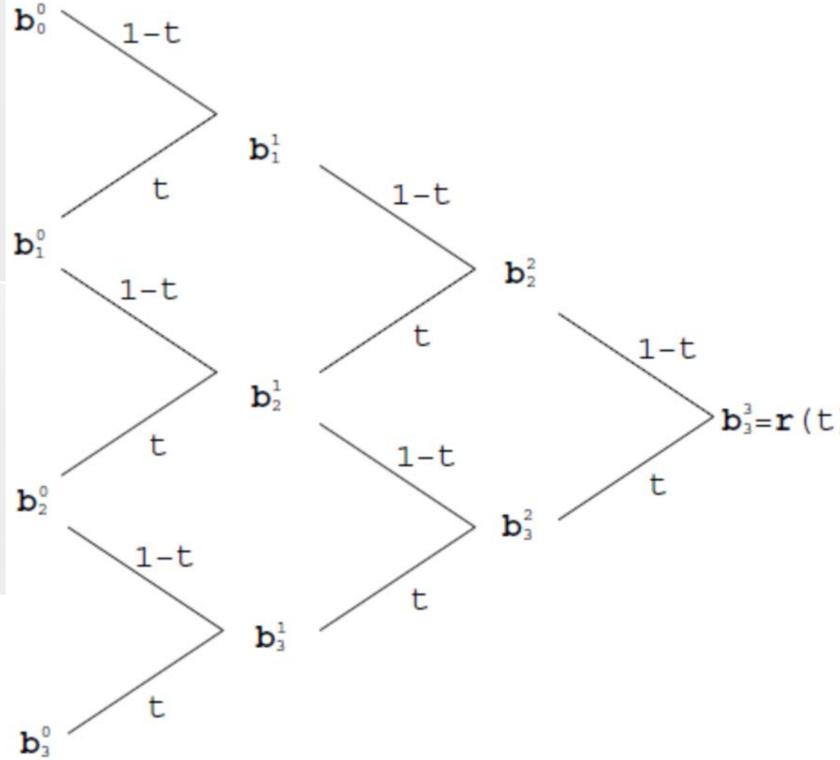
$$(1-u) \quad (u)$$

$$\mathbf{p}_0^0 \quad \mathbf{p}_1^0 \quad \mathbf{p}_2^0 \quad \mathbf{p}_3^0$$

$$\mathbf{p}_0^1 \quad \mathbf{p}_1^1 \quad \mathbf{p}_2^1$$

$$\mathbf{p}_0^2 \quad \mathbf{p}_1^2$$

$$\mathbf{p}_0^3 = \mathbf{c}(u)$$



$$p1(u) := (1-u)\cdot p_0 + u\cdot p_1$$

$$p2(u) := p_0\cdot(1-u)^2 + p_1\cdot 2\cdot u\cdot(1-u) + p_2\cdot u^2$$

$$p2(u) := (1-u)\left[(1-u)\cdot p_0 + u\cdot p_1\right] + u\cdot\left[(1-u)\cdot p_1 + u\cdot p_2\right]$$

$$p3(u) := (1-u)\cdot\left[(1-u)\cdot\left[(1-u)\cdot p_0 + u\cdot p_1\right] + u\cdot\left[(1-u)\cdot p_1 + u\cdot p_2\right]\right]$$
$$+ u\cdot\left[(1-u)\cdot\left[(1-u)\cdot p_1 + u\cdot p_2\right] + u\cdot\left[(1-u)\cdot p_2 + u\cdot p_3\right]\right]$$

$$p3(u) := (1-u)\cdot\left[(1-u)\cdot\left[(1-u)\cdot p_0 + u\cdot p_1\right] + u\cdot\left[(1-u)\cdot p_1 + u\cdot p_2\right]\right] + u\cdot\left[(1-u)\cdot\left[(1-u)\cdot p_1 + u\cdot p_2\right] + u\cdot\left[(1-u)\cdot p_2 + u\cdot p_3\right]\right]$$

$$p(u) := p_0\cdot(1-u)^3 + p_1\cdot 3\cdot u\cdot(1-u)^2 + p_2\cdot 3\cdot u^2\cdot(1-u) + p_3\cdot u^3$$

$$\mathbf{p}(u) = (1-u)^3\mathbf{p}_0 + 3u(1-u)^2\mathbf{p}_1 + 3u^2(1-u)\mathbf{p}_2 + u^3\mathbf{p}_3$$

Cubic Bezier Curve

$$\mathrm{p}(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \mathrm{M}_\mathrm{B} \begin{bmatrix} \mathrm{P}_1 \\ \mathrm{P}_2 \\ \mathrm{P}_3 \\ \mathrm{P}_4 \end{bmatrix}$$

$$\mathrm{M}_\mathrm{B} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

User specifies 4 control points P1 ... P4

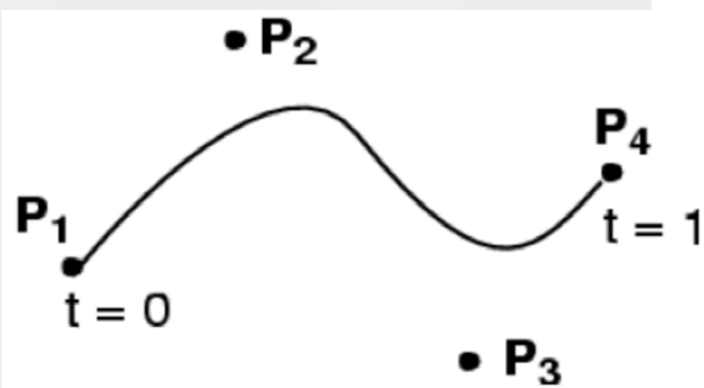
Curve goes through (interpolates) the ends P1, P4

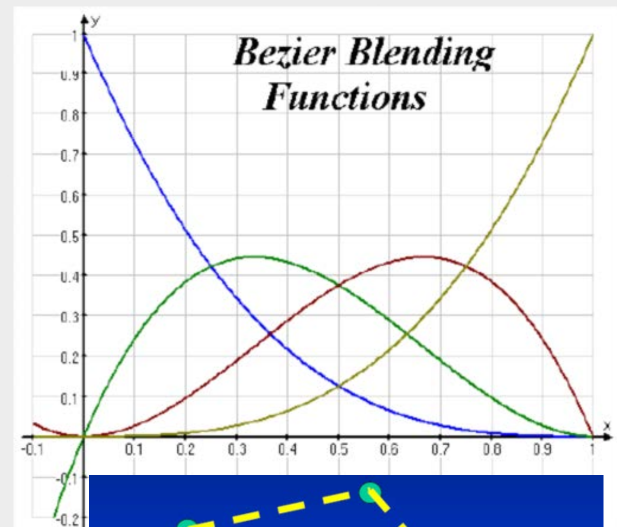
Approximates the two other ones

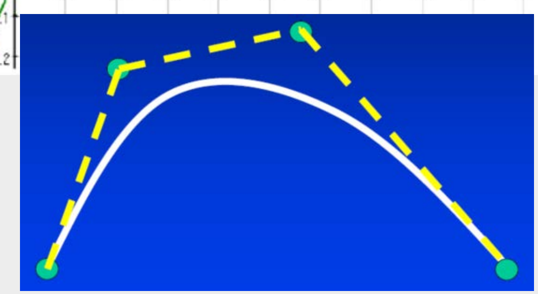
The weights describe the influence of each control point

Cubic polynomial :

$$\begin{aligned} \mathrm{P}(t) = {} & (1\text{-}t)^3 && \mathbf{P1} \\ + {} & 3t(1\text{-}t)^2 && \mathbf{P2} \\ + {} & 3t^2(1\text{-}t) && \mathbf{P3} \\ + {} & t^3 && \mathbf{P4} \end{aligned}$$





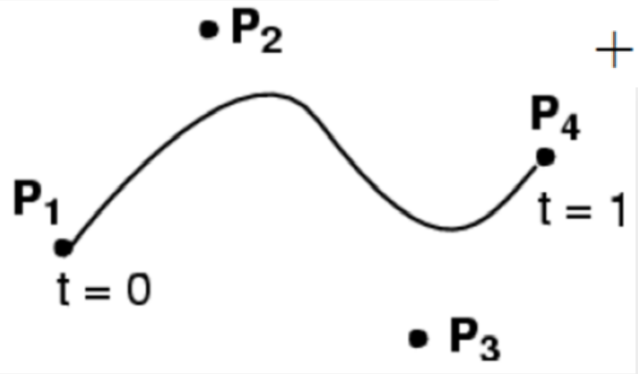


$$\mathbf{p}(u) = (1-u)^3\mathbf{p}_0 + 3u(1-u)^2\mathbf{p}_1 + 3u^2(1-u)\mathbf{p}_2 + u^3\mathbf{p}_3$$

Cubic Bezier Curve

Cubic polynomial :

$$\begin{aligned} P(t) = \ & (1\text{-}t)^3 && \mathbf{P1} \\ + \ & 3t(1\text{-}t)^2 && \mathbf{P2} \\ + \ & 3t^2(1\text{-}t) && \mathbf{P3} \\ + \ & t^3 && \mathbf{P4} \end{aligned}$$



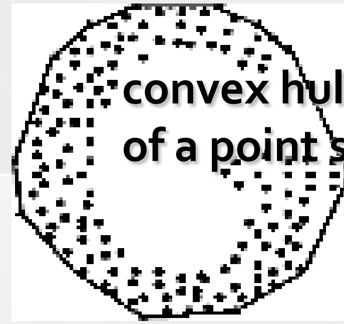
That is,

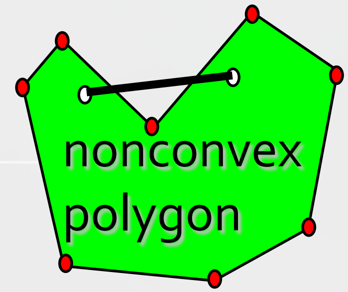
$$\begin{aligned} x(t) = (1-t)^3\ x_1 + \\ 3t(1-t)^2\ x_2 + \\ 3t^2(1-t)\ x_3 + \\ t^3\ x_4 \end{aligned}$$

$$\begin{aligned} y(t) = (1-t)^3\ y_1 + \\ 3t(1-t)^2\ y_2 + \\ 3t^2(1-t)\ y_3 + \\ t^3\ y_4 \end{aligned}$$

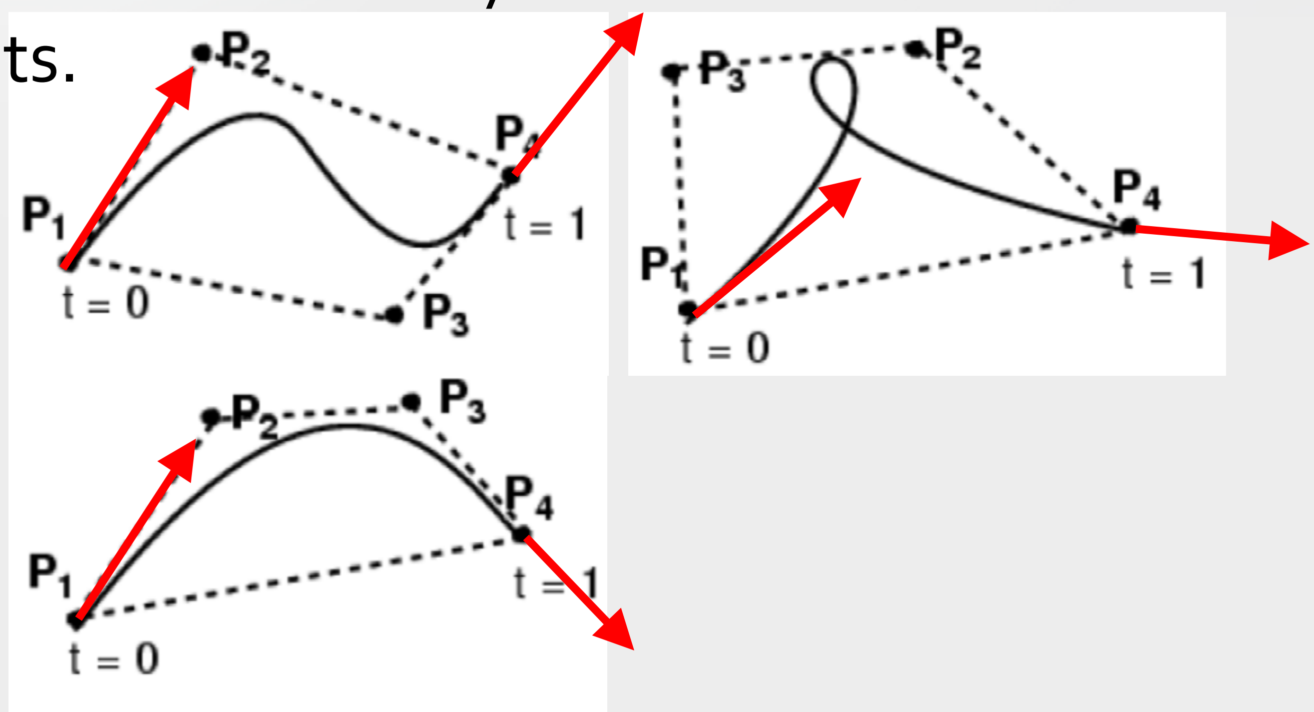
Verify what happens for t=0 and t=1

Cubic Bezier Curve





4 control points
Curve passes through first & last control point
Curve is tangent at P1 to (P1-P2) and at P4 to (P4-P3)
A Bézier curve is bounded by the convex hull of its control points.



$$\mathbf{p}(u) = (1-u)^3\mathbf{p}_0 + 3u(1-u)^2\mathbf{p}_1 + 3u^2(1-u)\mathbf{p}_2 + u^3\mathbf{p}_3$$

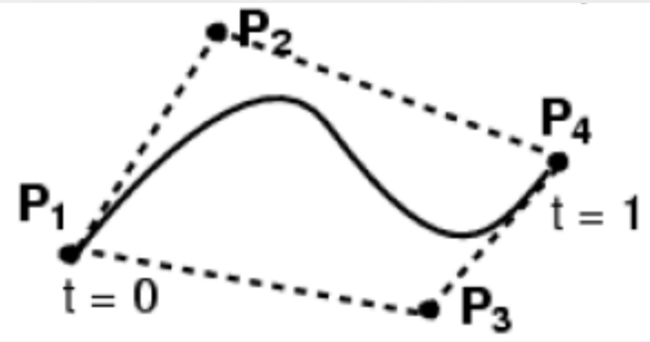
Bezier blending functions, weights

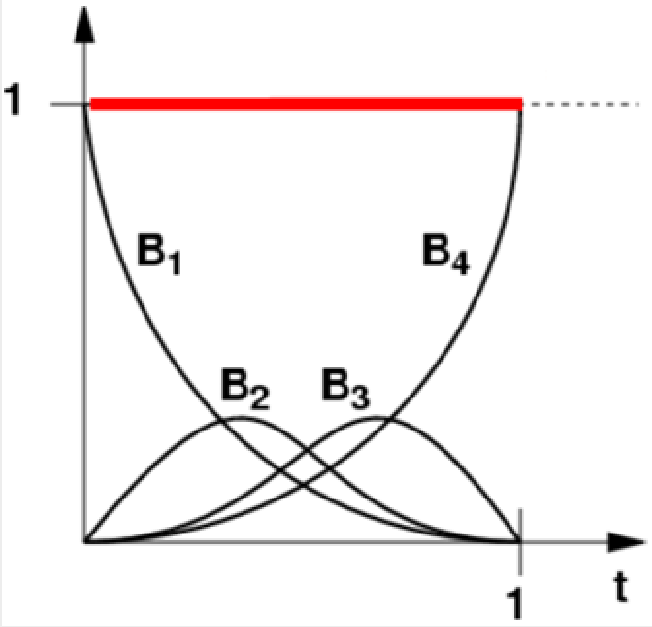
P(t) is a weighted linear combination of the 4 control points with weights:

$B1(t)=(1-t)^3$
$B2(t)=3t(1-t)^2$
$B3(t)=3t^2(1-t)$
$B4(t)=t^3$

$$
\begin{aligned}
P(t) = \; & (1-t)^3 && \mathbf{P1} \\
+ \; & 3t(1-t)^2 && \mathbf{P2} \\
+ \; & 3t^2(1-t) && \mathbf{P3} \\
+ \; & t^3 && \mathbf{P4}
\end{aligned}
$$

First, P1 is the most influential point, then P2, P3, and P4 .
P2 and P3 never have full influence.
Not interpolated!





Bezier, Bernstein basis polynomials

For Bézier curves, the basis polynomials/vectors are Bernstein polynomials

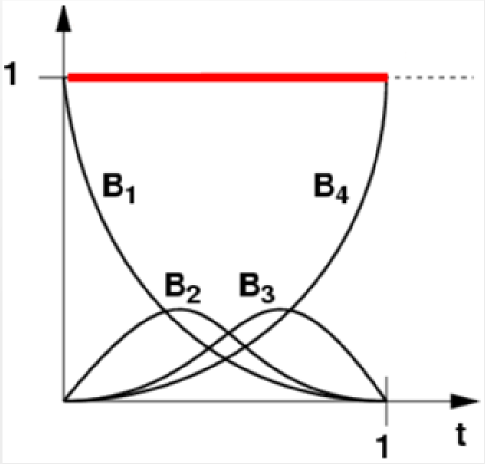
For cubic Bezier curve:

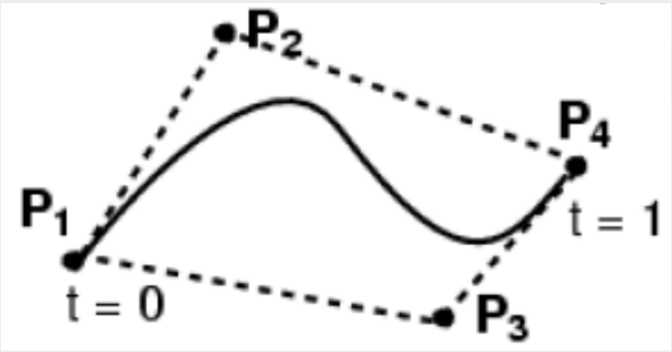
$B1(t)=(1-t)^3$

$B2(t)=3t(1-t)^2$

$B3(t)=3t^2(1-t)$

$B4(t)=t^3$





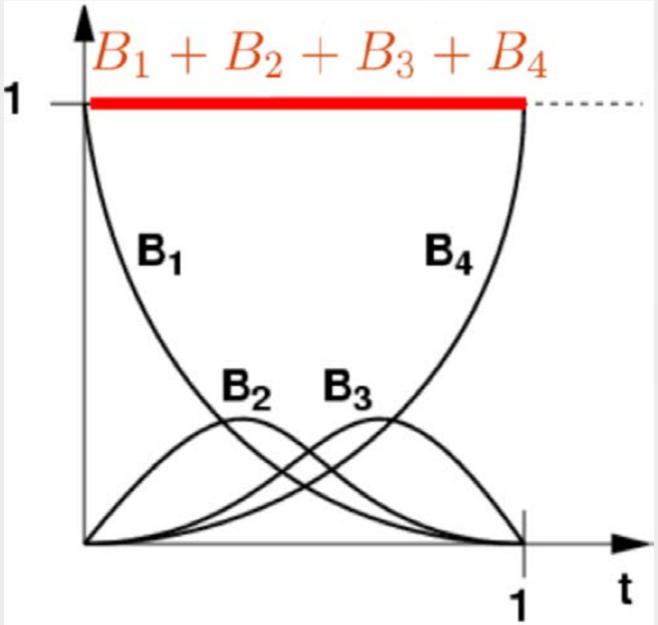
(careful with indices, many authors start at 0)

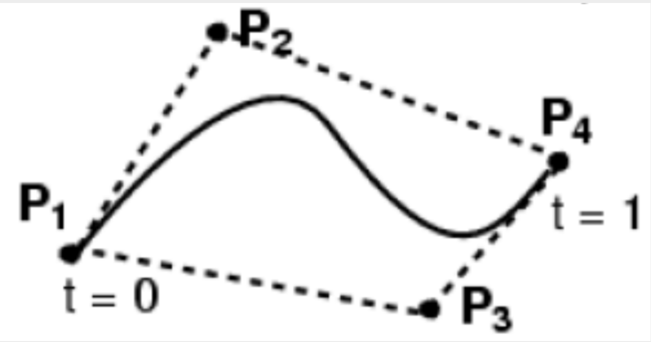
Defined for any degree

Properties of Bernstein Polynomials

≥ 0 for all $0 \leq t \leq 1$

- Sum to 1 for every t
- called partition of unity
- These two together are the reason why Bézier curves lie within convex hull
- B1(0) =1
- Bezier curve interpolates P1
- B4(1) =1
- Bezier curve interpolates P4





$$\mathbf{p}(u) = (1-u)^3\mathbf{p}_0 + 3u(1-u)^2\mathbf{p}_1 + 3u^2(1-u)\mathbf{p}_2 + u^3\mathbf{p}_3$$

Bézier Curves in Bernstein Basis

- P(t) = P1B1(t) + P2B2(t) + P3B3(t) + P4B4(t)
- P_i are 2D points (x_i , y_i)
- P(t) is a linear combination of the control points with weights equal to Bernstein polynomials at t
- But at the same time, the control points (P1, P2, P3, P4) are the "coordinates" of the curve in the Bernstein basis
- In this sense, specifying a Bézier curve with control points is exactly like specifying a 2D point with its x and y coordinates.

Change of Basis, How You Get the Matrix

$$p(u) = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \cdot \begin{pmatrix} a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \\ a_{01} & a_{02} & a_{03} \end{pmatrix} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} M_B \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$

$$M_B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

How do we go from Bernstein basis to the canonical monomial basis 1, t, t^2, t^3 and back?

- With a matrix!

$B1(t)=(1-t)^3$

$B2(t)=3t(1-t)^2$

$B3(t)=3t^2(1-t)$

$B4(t)=t^3$

$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{B} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Expand these out and collect powers of ***t***. The coefficients are the entries in the matrix B!

$$P(t) = \sum_{i=1}^{4} P_i\, B_i(t) = \sum_{i=1}^{4} \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} B_i(t) \right]$$

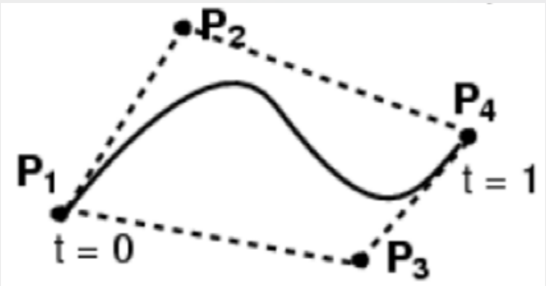
$$\underbrace{P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}}_{\text{point on curve (2x1 vector)}} = \underbrace{\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix}}_{\text{matrix of control points (2 x 4)}} \cdot \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{B} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Bernstein polynomials (4x1 vector)

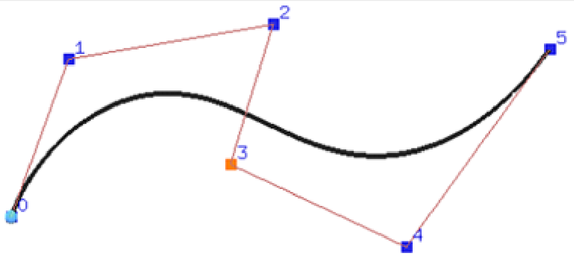
Change of Basis, Other Direction

Given B1…B4, how to get back to canonical 1, t, t^2, t^3 ?
That's right, with the inverse matrix!

$$\begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1/3 & 2/3 & 1 \\ 0 & 0 & 1/3 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{\boldsymbol{B}^{-1}} \begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix}$$



The cubic basis can be extended to higher-order polynomials. Higher-dimensional vector space, more control points



Change of Basis with matrices

- Cubic polynomials form a 4D vector space.
- Bernstein basis is canonical for Bézier.
- Can be seen as influence function of data points
- Or data points are coordinates of the curve in the Bernstein basis.
- We can change between basis with matrices.

$$P(t) = \sum_{i=1}^{4} P_i\, B_i(t) = \sum_{i=1}^{4} \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} B_i(t) \right]$$

$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \begin{pmatrix} (1-t)^3 \\ 3t(1-t)^2 \\ 3t^2(1-t) \\ t^3 \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{\boldsymbol{B}} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Bernstein polynomials (4x1 vector)

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix}$$

point on curve (2x1 vector)

matrix of control points (2 x 4)

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{\boldsymbol{B}} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

point on curve (2x1 vector)

matrix of control points (2 x 4)

More Matrix-Vector Notation

$$P(t) = \sum_{i=1}^{4} P_i \, B_i(t) = \sum_{i=1}^{4} \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} B_i(t) \right]$$

Bernstein polynomials (4x1 vector)

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix}$$

point on curve (2x1 vector)

matrix of control points (2 x 4)

Bernstein polynomials (4x1 vector)

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{\boldsymbol{B}} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

point on curve (2x1 vector)

matrix of control points (2 x 4)

Cubic Bézier in Matrix Notation

point on curve
(2x1 vector)

Canonical
monomial basis

$$\mathrm{P(t)} = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

"Geometry matrix"
of control points P1..P4
(2 x 4)

"Spline matrix"
(Bernstein)

$$\begin{pmatrix} B_1(t) \\ B_2(t) \\ B_3(t) \\ B_4(t) \end{pmatrix} = \overbrace{\begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}}^{\boldsymbol{B}} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

$$\mathbf{p}(u) = (1-u)^3\mathbf{p}_0 + 3u(1-u)^2\mathbf{p}_1 + 3u^2(1-u)\mathbf{p}_2 + u^3\mathbf{p}_3$$

Higher-Order Bézier Curves

$$P(t) = \sum_{i=1}^{4} P_i \, B_i(t) = \sum_{i=1}^{4} \left[\begin{pmatrix} x_i \\ y_i \end{pmatrix} B_i(t) \right]$$

> 4 control points

Bernstein Polynomials as the basis functions

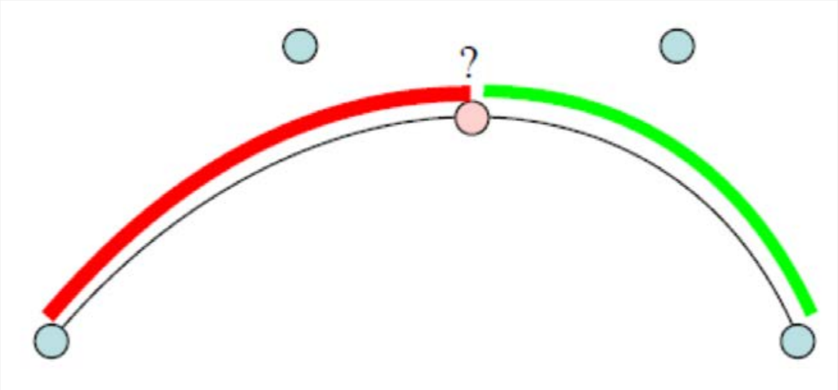
For polynomial of order *n*, the *ith* basis function is

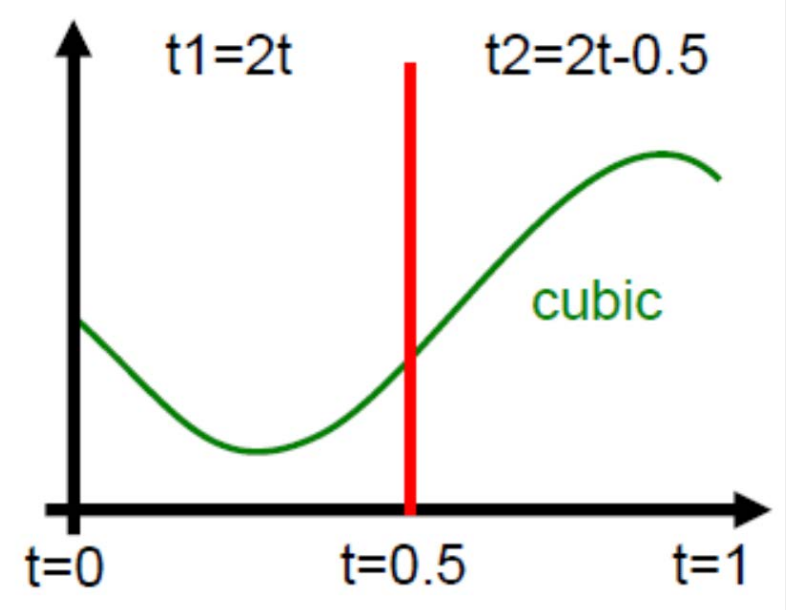
$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$$

- Every control point affects the entire curve
- Not simply a local effect
- More difficult to control for modeling
- You will not need this in this class

Subdivision of a Bezier Curve

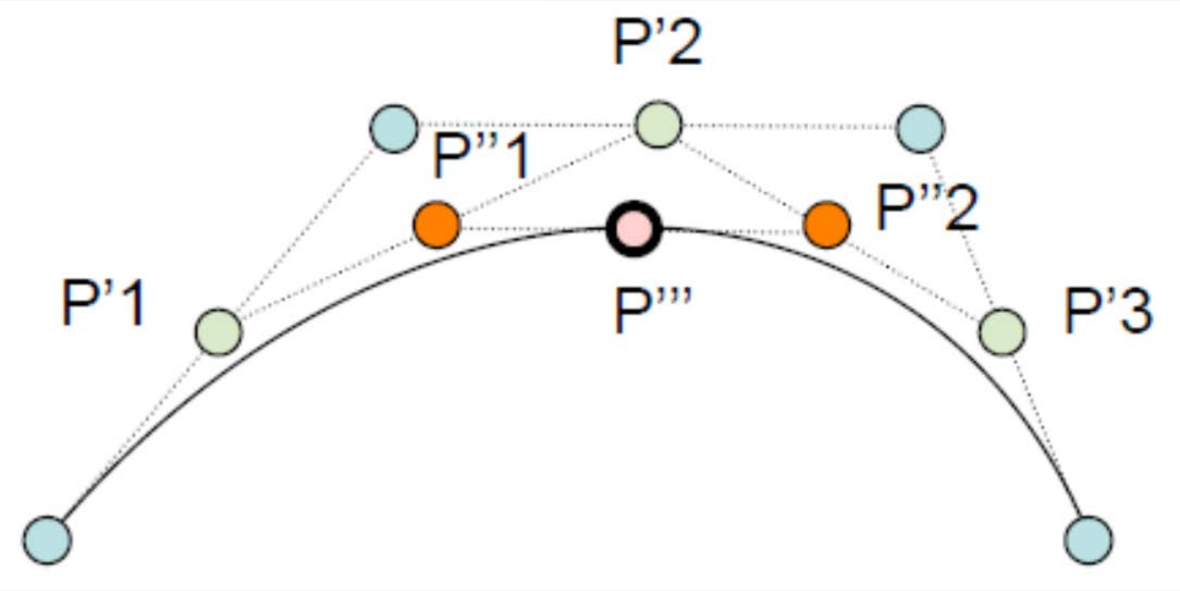
- Can we split a Bezier curve in the middle into two Bézier curves?
- This is useful for adding detail
- It avoids using nasty higher-order curves
- The resulting curves are again a cubic (Why? A cubic in t is also a cubic in 2t)



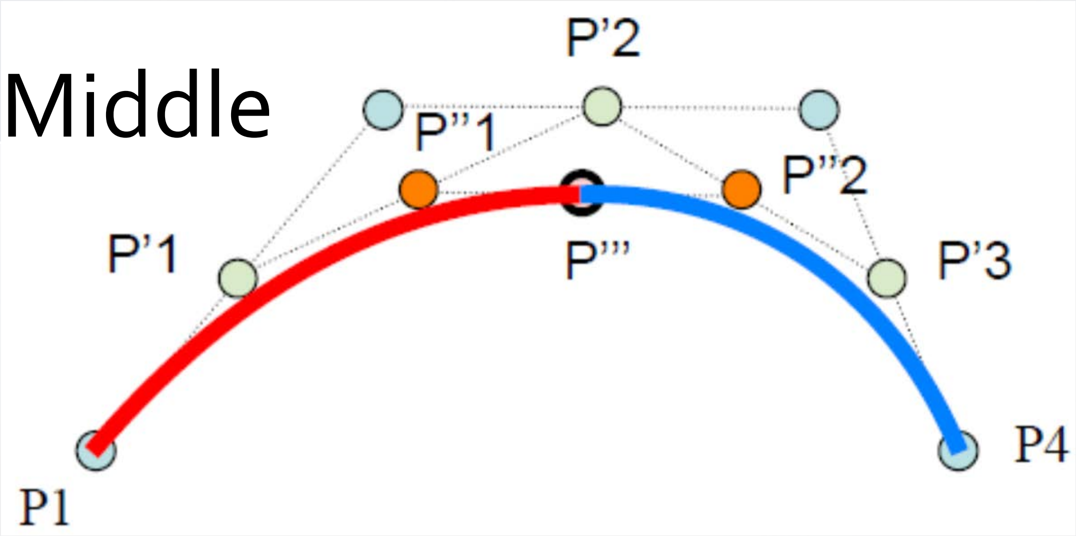


De Casteljau Construction

- Take the middle point of each of the 3 segments
- Construct the two segments joining them
- Take the middle of those two new segments
- Join them
- Take the middle point P'''



Result of Split in Middle

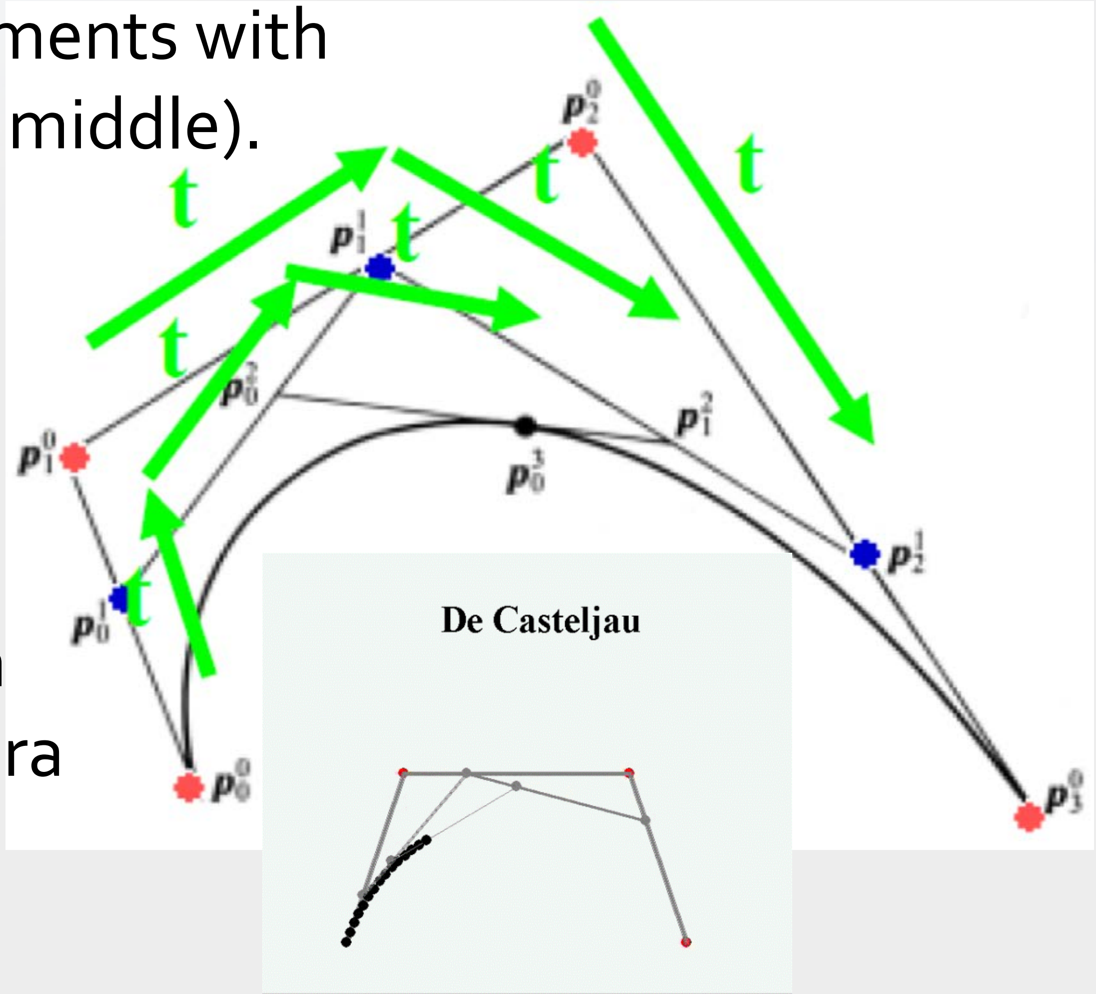


The two new curves are defined by
P1, P'1, P''1, and P'''
P''', P''2, P'3, and P4
Together they exactly replicate the original curve!
Originally 4 control points, now 7 (more control)
Ref. MIT OpenCourseWare http://ocw.mit.edu6.837 Computer Graphics Fall 2012 For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.

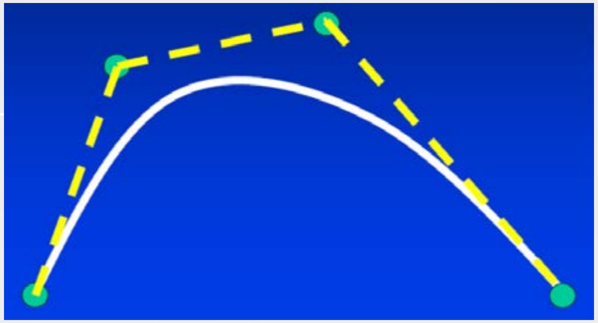
De Casteljau Construction

Actually works to construct a point at any t, not just 0.5
Just subdivide the segments with
ratio (1-t), t (not in the middle).
Linear combination
of basis functions.
Bernstein polynomials

Subdivision by
de Casteljau algorithm
All linear, matrix algebra



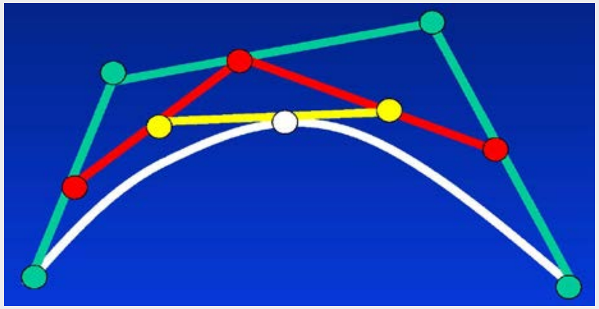
Cubic Bezier Curves



Four control points, Curve geometry

Cubic Bezier curve geometry

$$\mathbf{c}(u) = \sum_{i=0}^{3} \mathbf{p}_i B_i^3(u)$$



Control points and basis functions

Image and properties of basis functions

$$B_i^n(u) = \frac{n!}{(n-i)!\,i!}(1-u)^{n-i}u^i$$

$$B_0^3(u) = (1-u)^3$$

$$B_1^3(u) = 3u(1-u)^2$$

$$B_2^3(u) = 3u^2(1-u)$$

$$B_3^3(u) = u^3$$

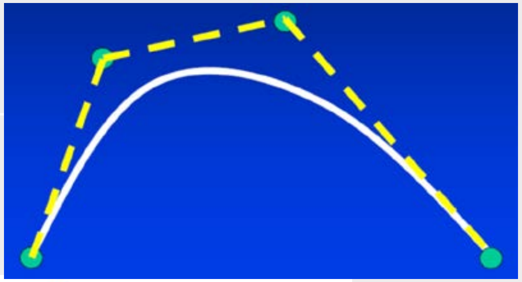
Recursive

Linear

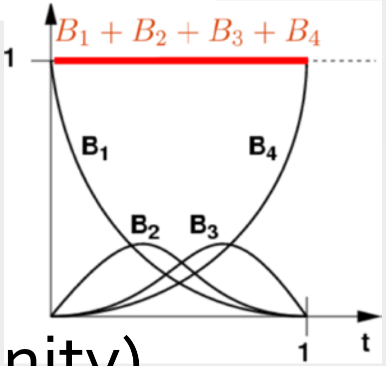
Evaluation

$$\begin{array}{ccccccc} & (1-u) & & (u) & & & \\ \mathbf{p}_0^0 & & \mathbf{p}_1^0 & & \mathbf{p}_2^0 & & \mathbf{p}_3^0 \\ & \mathbf{p}_0^1 & & \mathbf{p}_1^1 & & \mathbf{p}_2^1 & \\ & & \mathbf{p}_0^2 & & \mathbf{p}_1^2 & & \\ & & & \mathbf{p}_0^3 = \mathbf{c}(u) & & & \end{array}$$

Cubic Bezier Curve Properties



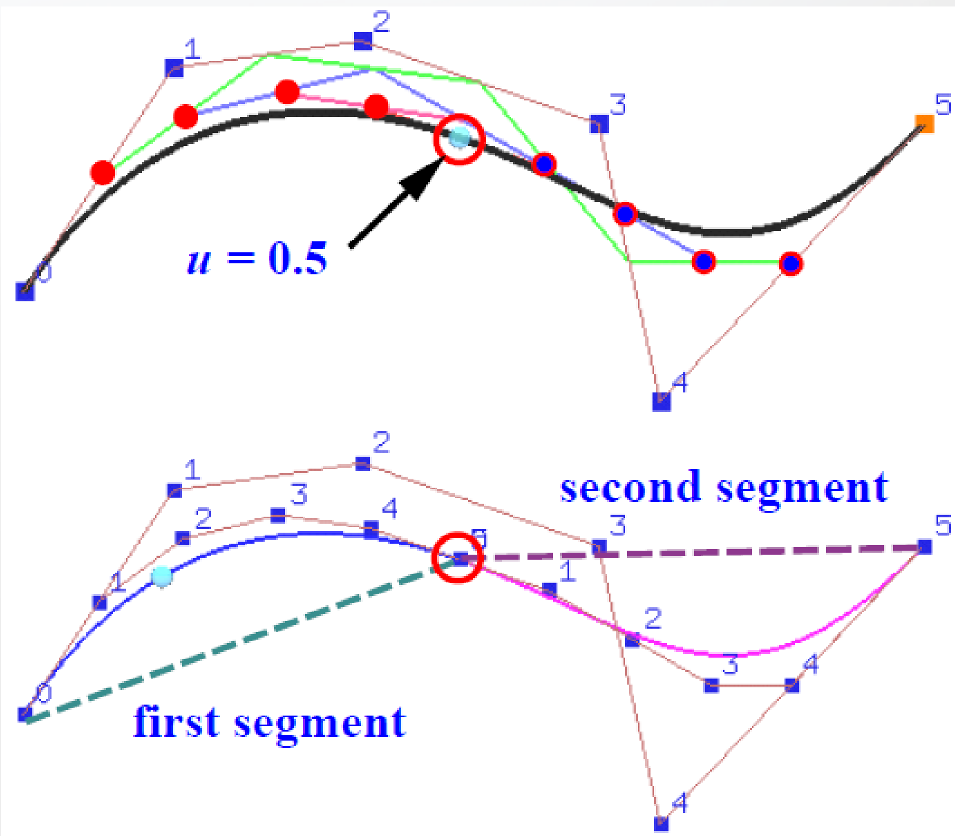
$$\mathbf{c}(u) = \sum_{i=0}^{3} \mathbf{p}_i B_i^3(u)$$

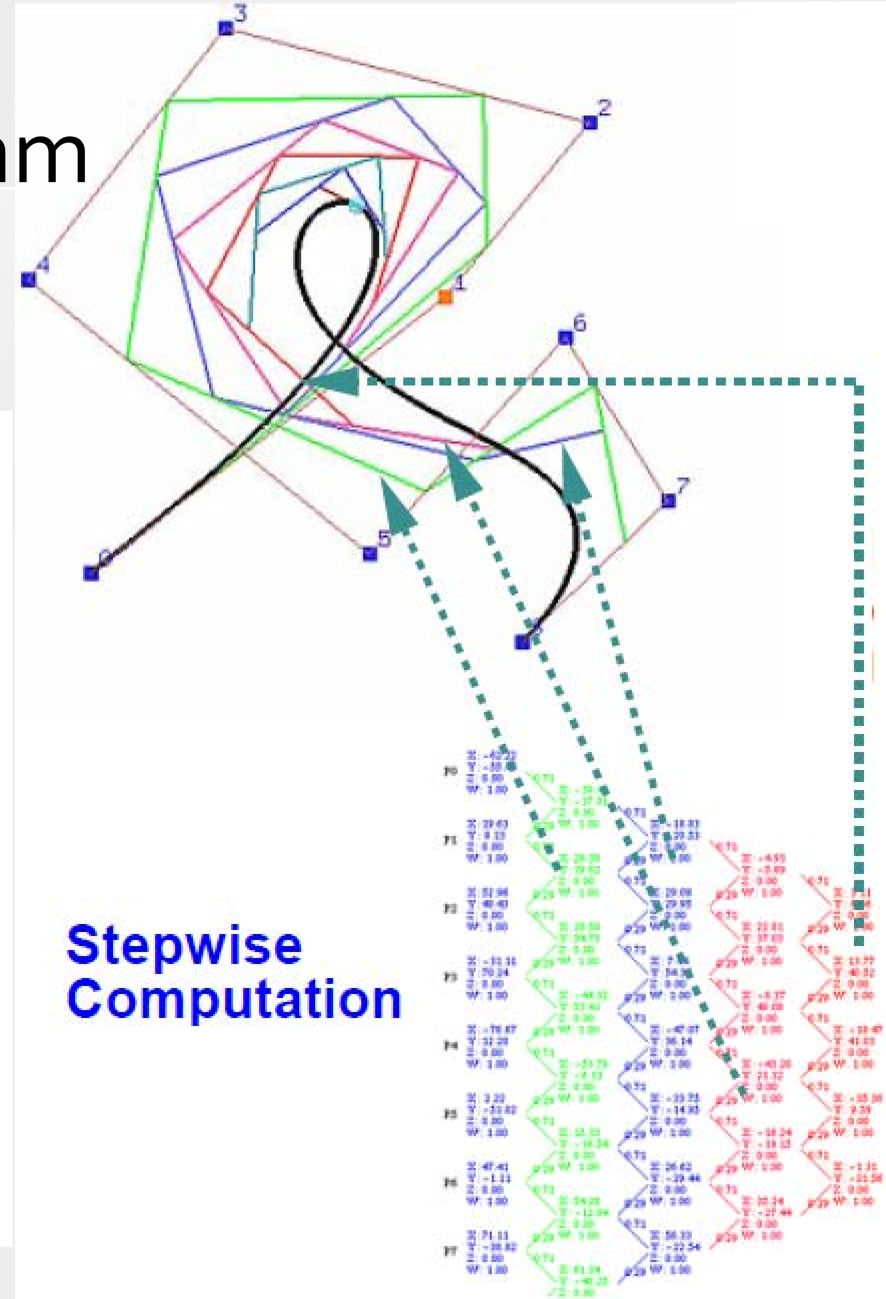


- The curve passes through the first and the last points (end-point interpolation).
- Linear combination of control points and basis functions.
- Basis functions are all polynomials
- Basis functions sum to one (partition of unity)
- All basis functions are non-negative
- Convex hull (both necessary and sufficient)
- Predictability

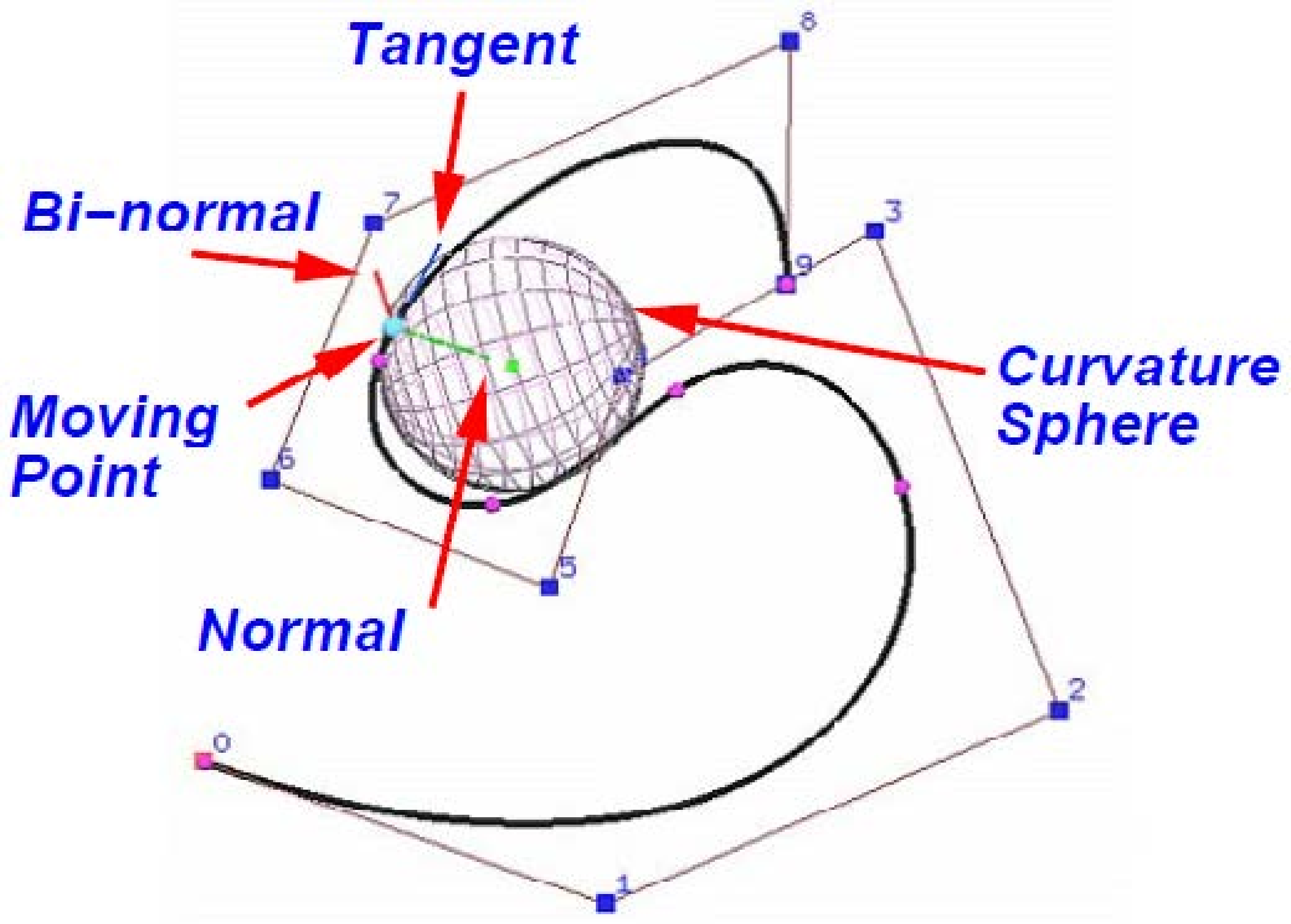
de Casteljau's Algorithm

- Curve Subdivision





Tangent, Normal, Curvature of a Curve



Continuity

Measures the degree of "smoothness" of a curve.

C^0 continuous – position continuous

C^1 continuous – slope continuous

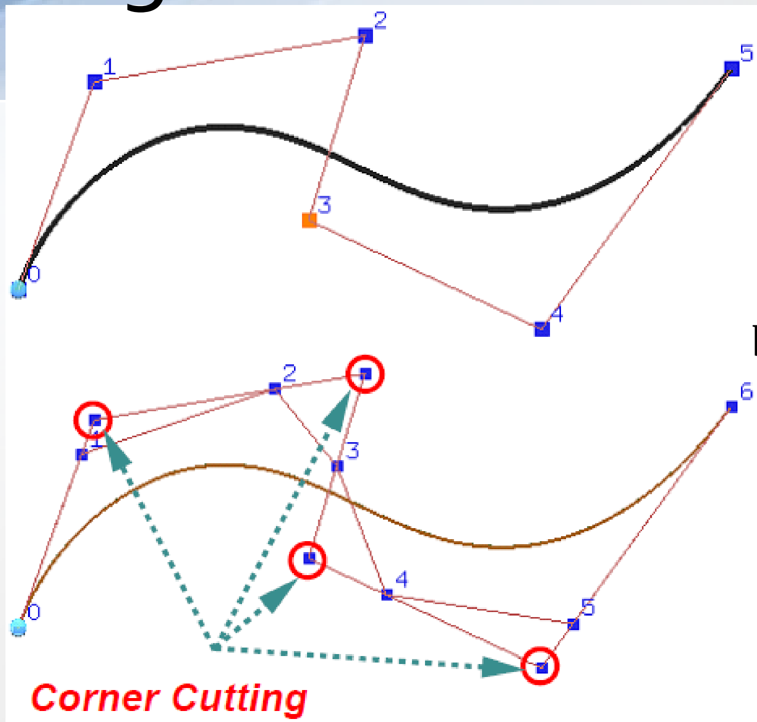
C^2 continuous – curvature continuous

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.

Continuity algorithm

Bezier curves can represent complex curves by **increasing the degree** and thus the number of control points.
Alternatively, complex curves can be represented using **composite curves**, which can be formed by joining several Bezier curves end to end.
If this method is adopted, the **continuity** between consecutive curves must be addressed.

Degree Elevation of a Curve

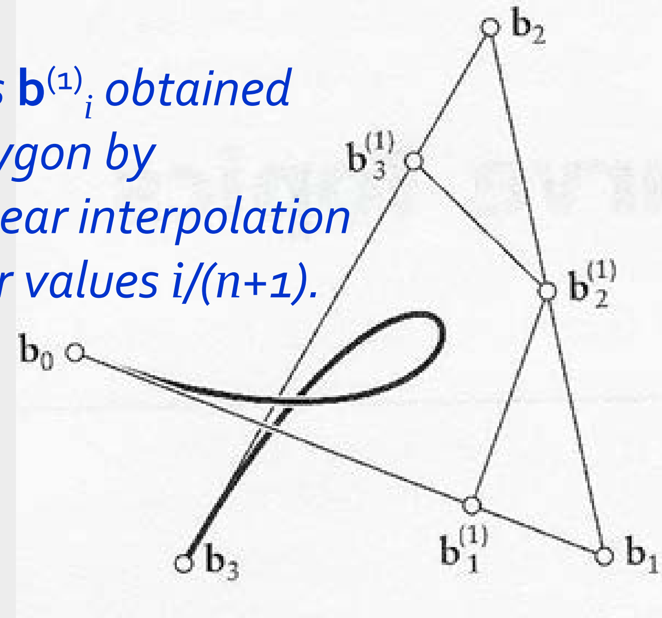


Adding a control point elevates degree by 1.

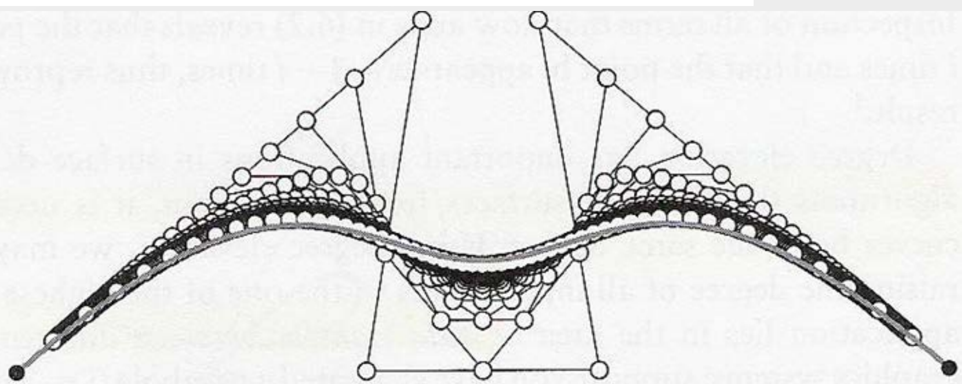
$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

$$\mathbf{b}_i^{(1)} = \frac{i}{(n+1)}\mathbf{b}_{i-1} + \left(1 - \frac{i}{(n+1)}\right)\mathbf{b}_i \qquad \text{for } i = 0, \ldots, n+1$$

New vertices $\mathbf{b}^{(1)}_i$ obtained from old polygon by piecewise linear interpolation at parameter values i/(n+1).

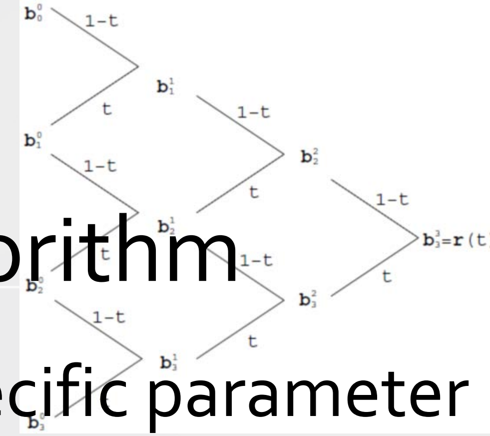


Degree elevation: both polygons define the same (degree three) curve.



Degree elevation: a sequence of polygons approaching the curve that is defined by each of them.

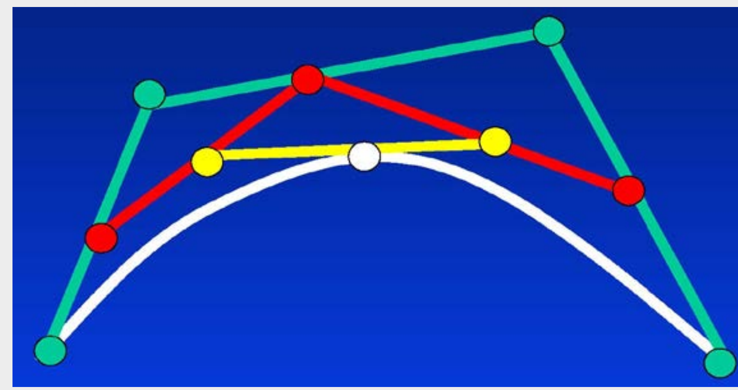
Evaluation and subdivision algorithm



A Bezier curve can be evaluated at a specific parameter value to and the curve can be split at that value using the **de Casteljau algorithm**

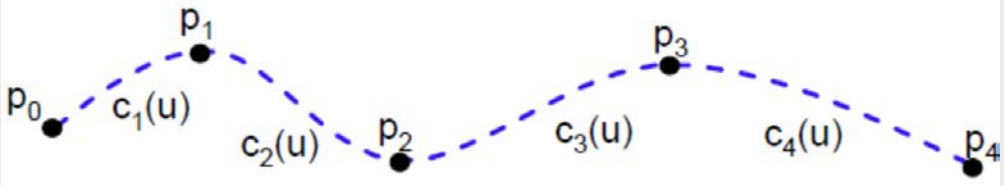
$$\mathrm{b}_i^k(t_0) = (1 - t_0)\mathrm{b}_{i-1}^{k-1} + t_0\mathrm{b}_i^{k-1}$$

$$k = 1, 2, \ldots, n, \quad i = k, \ldots, n$$



Symmetry property: If we renumber the control points as $\mathbf{b}^*_{n-i} = \mathbf{b}_i$,

$$\sum_{i=0}^{n} \mathrm{b}_i B_{i,n}(t) = \sum_{i=0}^{n} \mathrm{b}_i^* B_{i,n}(1-t)$$



Piecewise Polynomial Splines

Piecewise, low degree polynomial curves for different parts of the curve with continuous joints

$$C(u) = \begin{pmatrix} x(u) \\ y(u) \\ z(u) \end{pmatrix} = \sum_i \hat{C}_i(u)$$

$$\text{where} \quad \hat{C}_i(u) = \begin{cases} C_i(u) & \text{if} \quad u \in [u_i, u_{i+1}] \\ 0 & \text{otherwise} \end{cases}$$

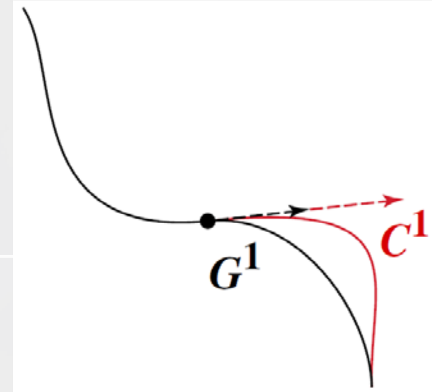
Advantages:

- Low-degree, Flexible, Rich representation
- Geometrically meaning coefficients
- Local effects
- Interactive sculpting capabilities

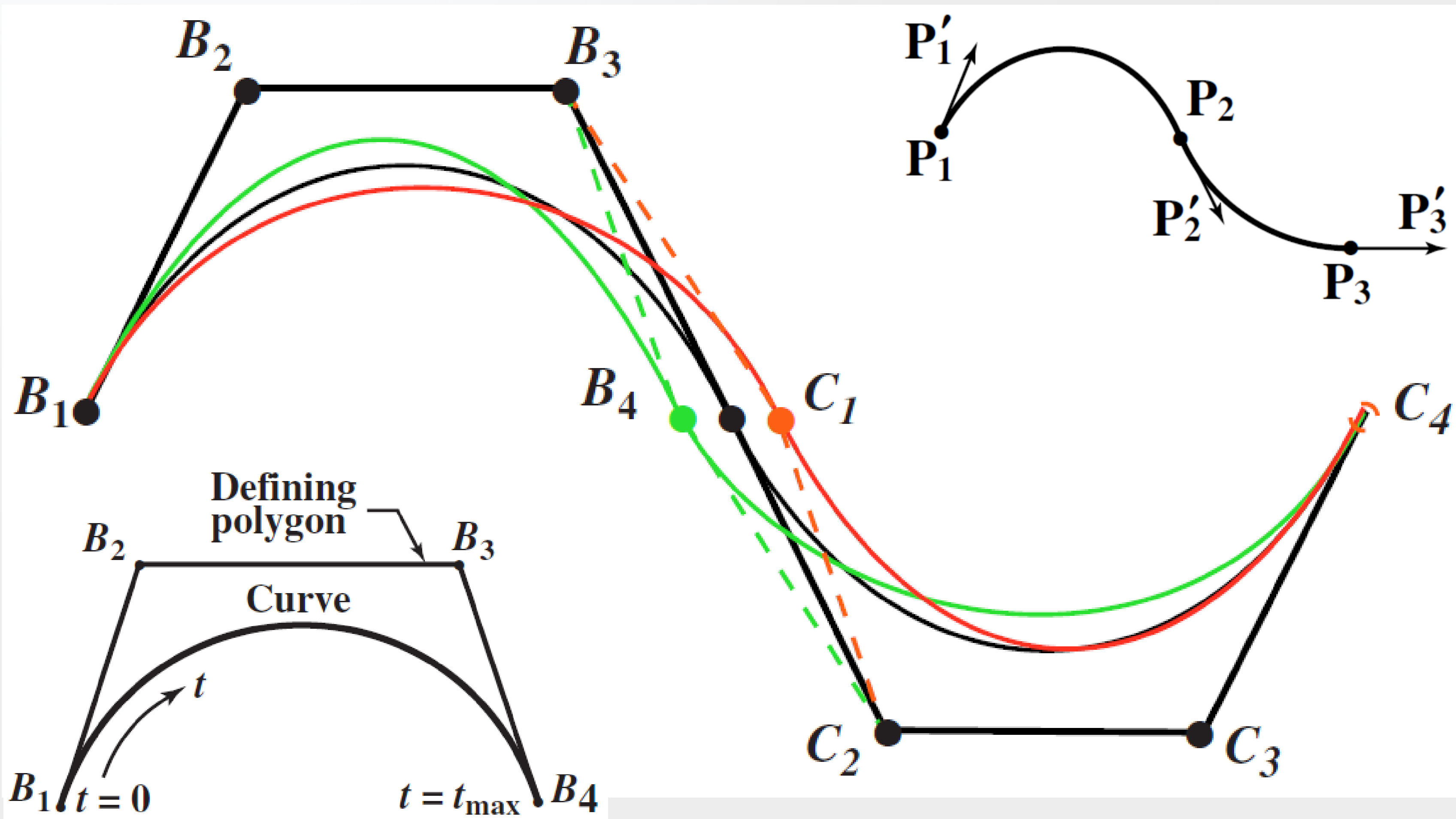
Disadvantages:

- How to ensure smoothness at the joints (continuity)

Bezier curves – Continuity



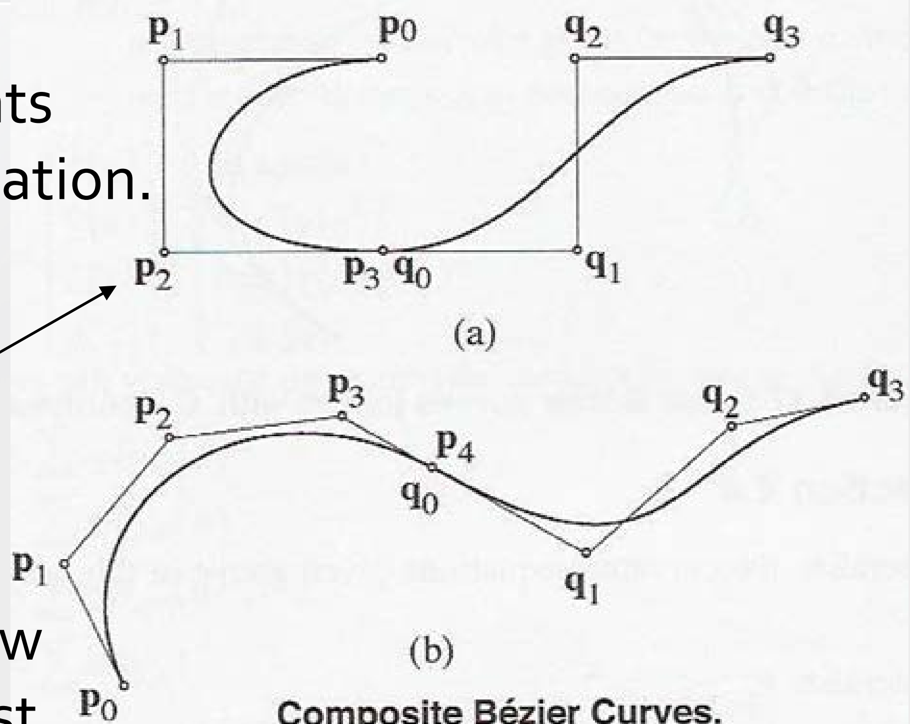
B3, B4/C1, C2 must lie in straight line



Composite Bezier Curves

Joining adjacent curve segments
is an alternative to degree elevation.

Collinearity of cubic Bezier
control points produces
G¹ continuity at join point:



Composite Bézier Curves.

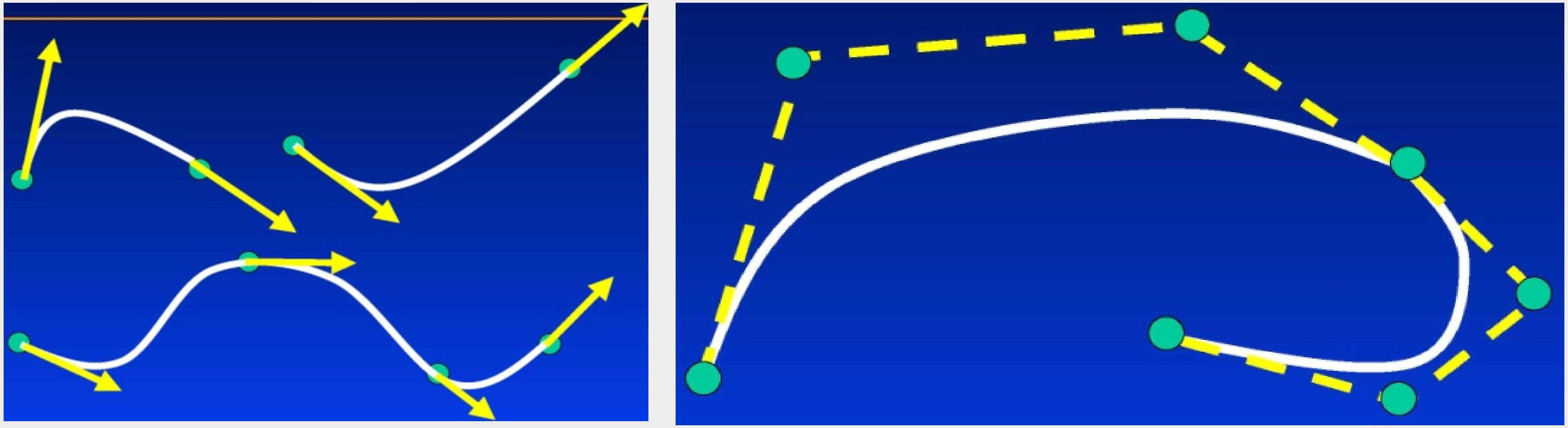
Evaluate at u=0 and u=1 to show
tangents related to first and last
control polygon line segment. $\mathbf{p}^u(0) = 3(\mathbf{p}_1 - \mathbf{p}_0)$ $\mathbf{p}^u(1) = 3(\mathbf{p}_3 - \mathbf{p}_2)$

For G² continuity at join point, 5 vertices must be coplanar.

Curves

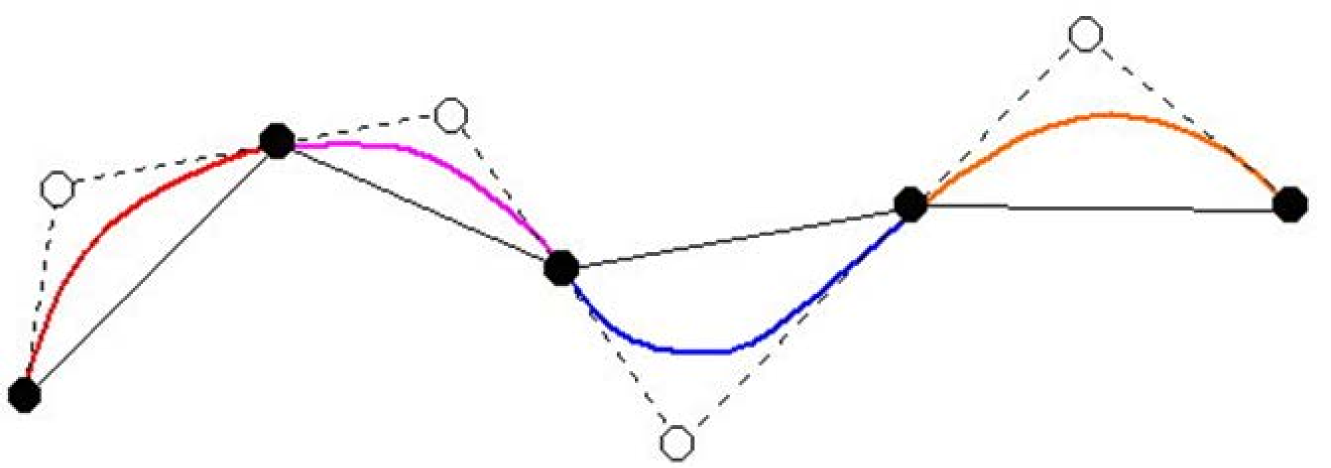
Both Bézier curves and B-splines are polynomial parametric curves. Polynomial parametric forms can not represent some simple analytic curves such as circles, conics (ellipses, parabolas, and hyperbolas).

Composite Curves by using Continuity

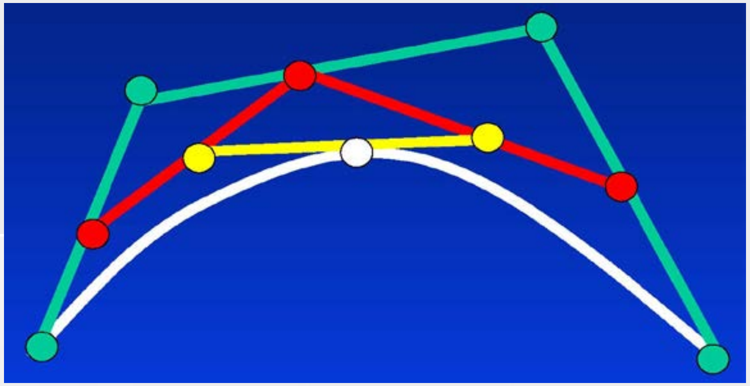


Interpolation using Bezier curve

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



Bezier Splines



Bezier curves of degree n

$$\mathbf{c}(u) = \sum_{i=0}^{n} \mathbf{p}_i \, B_i^n(u)$$

$$B_i^n(u) = \binom{n}{i}(1-u)^{n-i}u^i$$

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

Control points and basis functions (Bernstein polynomials of degree n):

$$B_i^n(u) = \frac{n!}{(n-i)!\,i!}(1-u)^{n-i}u^i$$

Recursive Computation:

$$\mathbf{p}_i^0 = \mathbf{p}_i, i = 0,1,2,\ldots n$$

$$\mathbf{p}_i^j = (1-u)\mathbf{p}_i^{j-1} + u\mathbf{p}_{i+1}^{j-1}$$

$$\mathbf{c}(u) = \mathbf{p}_0^n(u)$$

Recursive Computation

N+1 levels

$$\mathbf{p}_i^0 = \mathbf{p}_i, i = 0,1,2,\ldots n$$
$$\mathbf{p}_i^j = (1-u)\mathbf{p}_i^{j-1} + u\mathbf{p}_{i+1}^{j-1}$$
$$\mathbf{c}(u) = \mathbf{p}_0^n(u)$$

$$\begin{array}{ccccccc}
 & (1-u) & & & (u) & & \\
\mathbf{p}_0^0 & & \ldots & & \ldots & & \mathbf{p}_n^0 \\
 & \mathbf{p}_0^1 & & \ldots & & \mathbf{p}_{n-1}^1 & \\
 & \ldots & & \ldots & & \ldots & \\
 & & \mathbf{p}_0^{n-1} & & \mathbf{p}_1^{n-1} & & \\
 & & & \mathbf{p}_0^n = \mathbf{c}(u) & & &
\end{array}$$

Properties of Bezier Splines

- Basis functions are non-negative
- The summation of all basis functions is unity
- End-point interpolation $\mathbf{c}(0) = \mathbf{p}_0 \ , \ \mathbf{c}(1) = \mathbf{p}_n$
- Binomial expansion theorem
- Convex hull: the curve is bounded by the convex hull defined by control points
- Recursive subdivision and evaluation

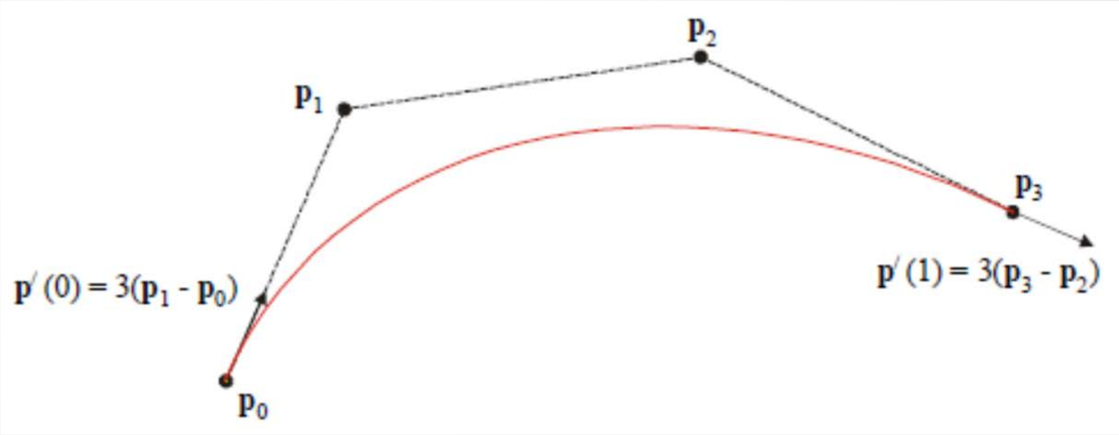
$$((1-u)+u)^n = \sum_{i=0}^{n} \binom{n}{i} u^i (1-u)^{n-i}$$

- Symmetry: c(u) and c(1-u) are defined by the same set of points, but different ordering

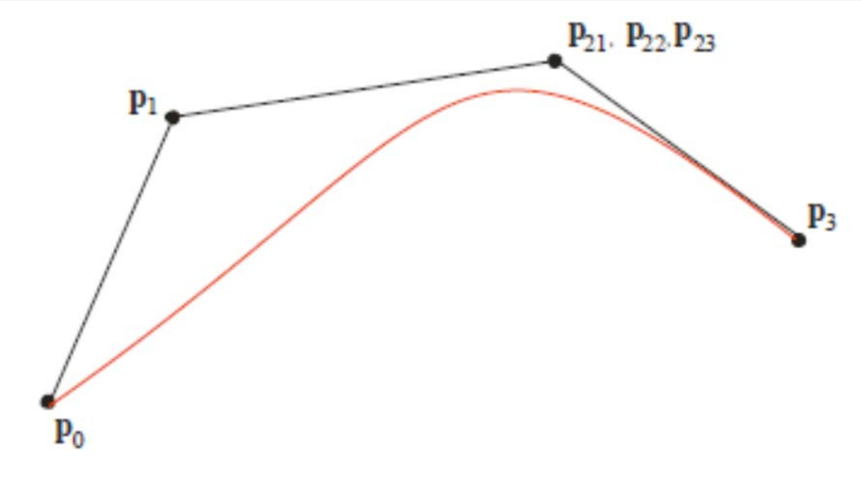
Bezier Curve Rendering

- Use its control polygon to approximate the curve
- Recursive subdivision till the tolerance is satisfied
- Algorithm go here
- If the current control polygon is flat (with tolerance), then output the line segments, else subdivide the curve at $u=0.5$
- Compute control points for the left half and the right half, respectively
- Recursively call the same procedure for the left one and the right one

Points and Curves (cont.)

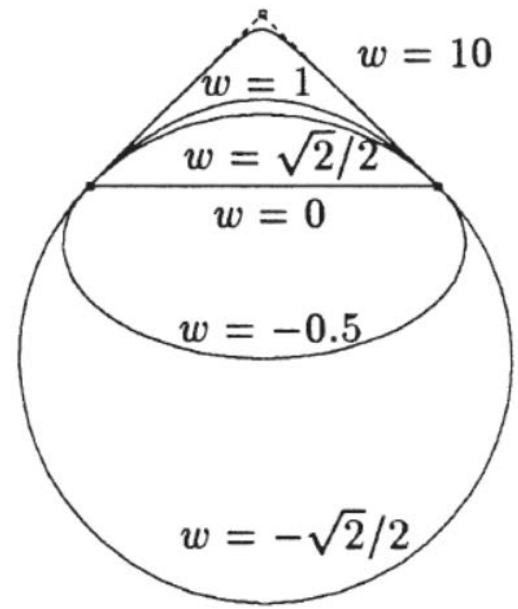


The shape of the curve can be changed by emphasizing certain desired points by creating pseudo-points coinciding at the same location:

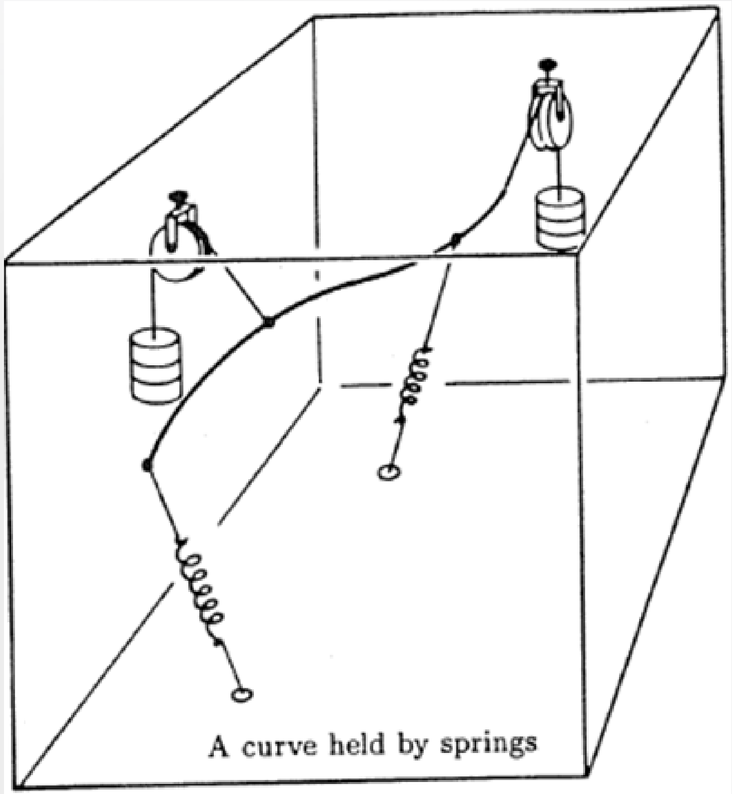


Rational Bezier curve

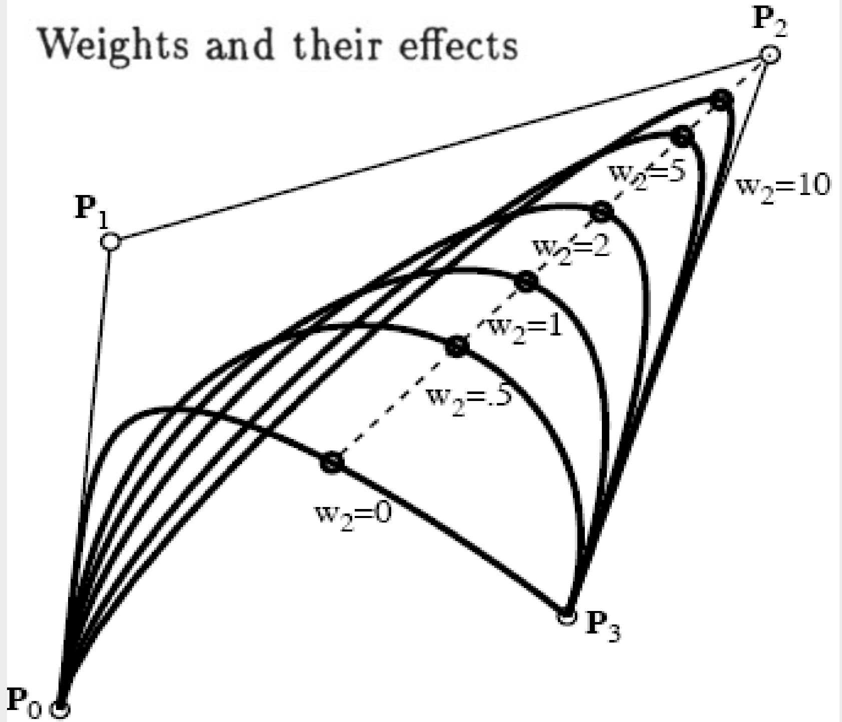
Effect of the weights of
the control points
of a Rational Bezier curve



Weights and their effects



A curve held by springs



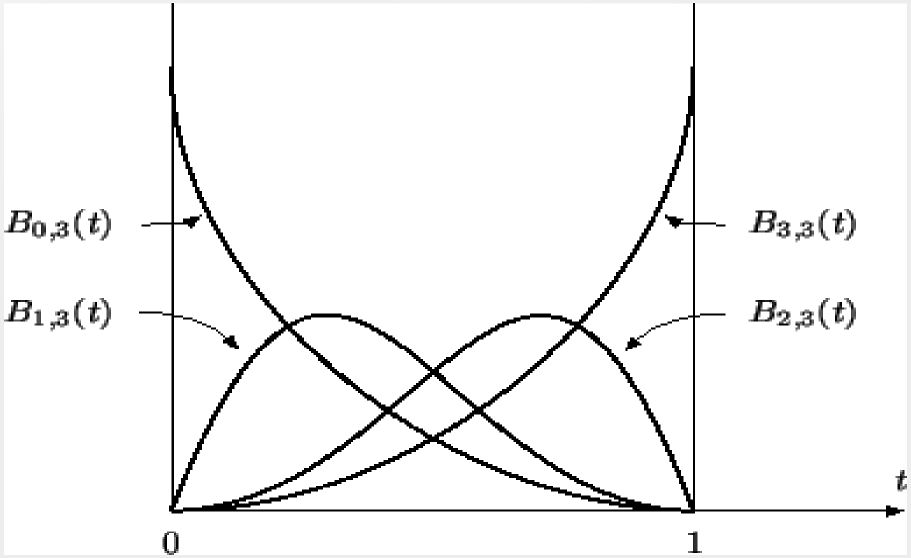
Bezier curves by Bernstein polynomials

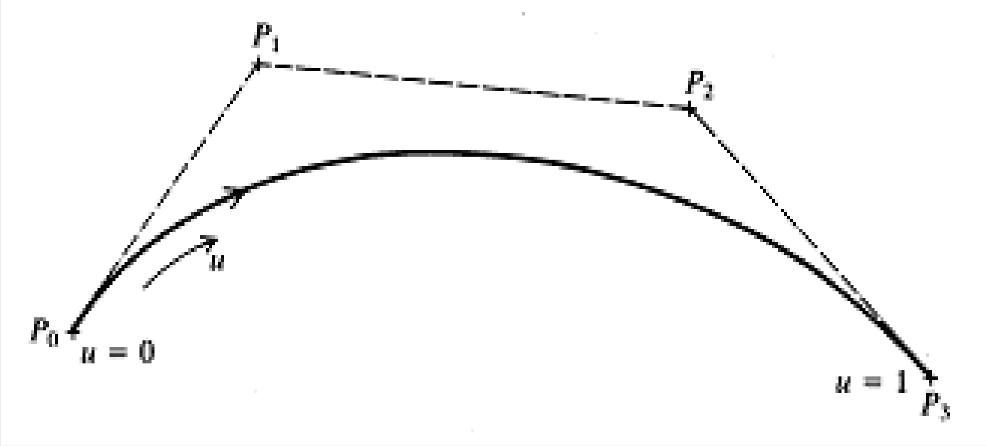
n=3 i=0,…,n binomial const. C(n,i) = n! / (i! (n-i)!)

$B(i,n,u) = C(n,i)\, u^i\, (1-u)^{n-i}$ Bezier weights

$$B_{i,n}(t) = \frac{n!}{i!(n-i)!}(1-t)^{n-i}t^i, \qquad i = 0, \ldots, n$$

$$P(u) = \sum_{i=0}^{n} B(i,n,u)\, P_i$$

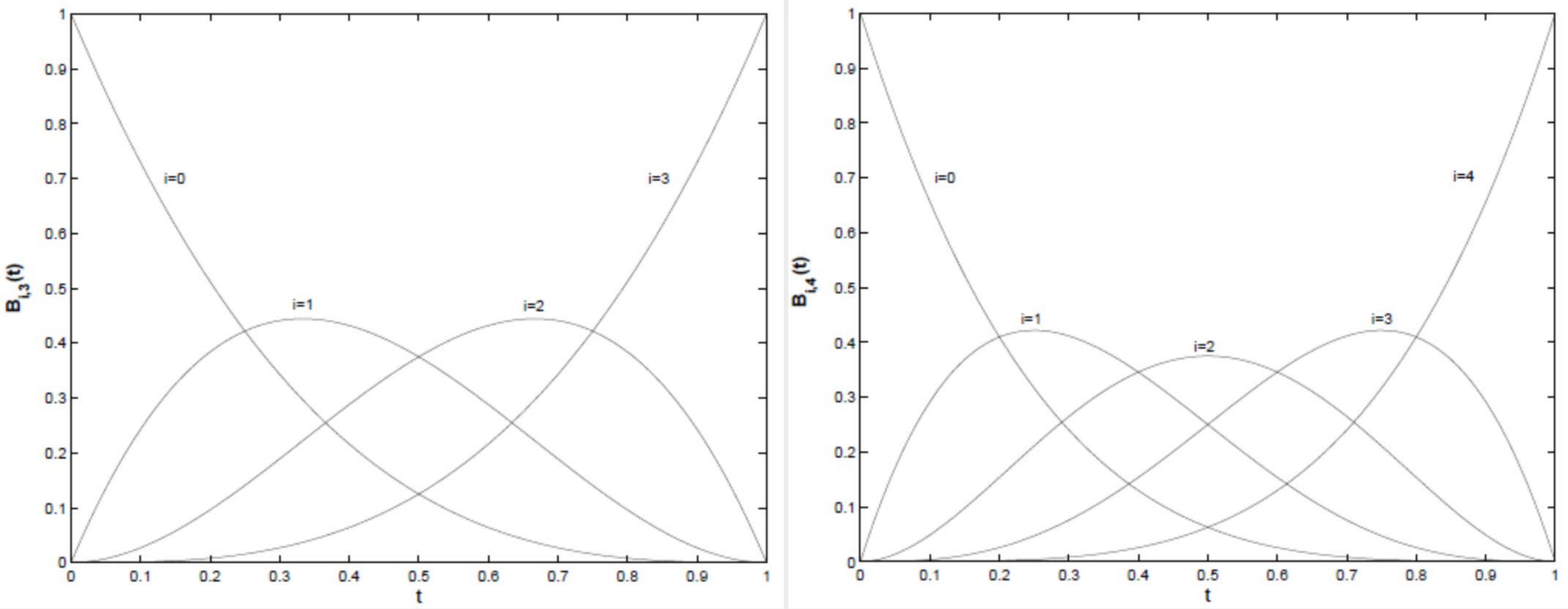




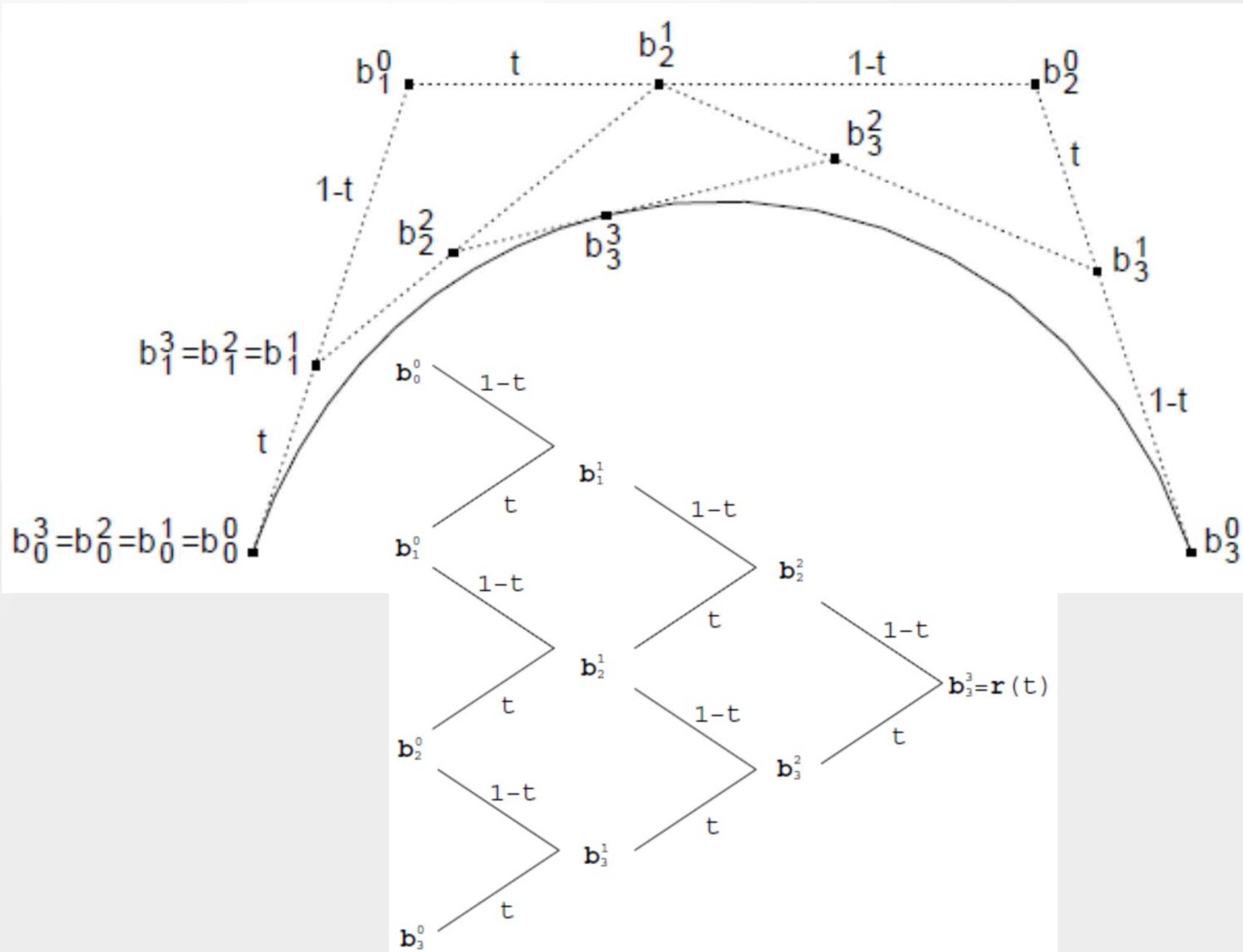
Bernstein polynomials

$$B_{i,n}(t) = \frac{n!}{i!(n-i)!}(1-t)^{n-i}t^i, \qquad i = 0, \ldots, n$$

(a) degree three, (b) degree four



The de Casteljau algorithm



Properties of the Berntein polynomials

- *Non-negativity*: $B_{i,n}(t) \geq 0, \qquad 0 \leq t \leq 1, \quad i = 0, \ldots, n$.
- *Partition of unity*: $\sum_{i=0}^{n} B_{i,n}(t) = (1 - t + t)^n = 1$ (by the *binomial* theorem).
- *Symmetry*:

$$B_{i,n}(t) = B_{n-i,n}(1-t) \; .$$

- *Recursion*: $B_{i,n}(t) = (1-t)B_{i,n-1}(t) + tB_{i-1,n-1}(t)$ with $B_{i,n}(t) = 0$ for $i < 0$, $i > n$ and $B_{0,0}(t) = 1$.
- *Linear precision*:

$$t = \sum_{i=0}^{n} \frac{i}{n} B_{i,n}(t) \; ,$$

which implies that the monomial t can be expressed as the weighted sum

Derivative of a Bernstein polynomial

$$\frac{dB_{i,n}(t)}{dt} = n[B_{i-1,n-1}(t) - B_{i,n-1}(t)]$$

where $B_{-1,n-1}(t) = B_{n,n-1}(t) = 0$

Degree elevation: The basis functions of degree n can be expressed in terms of those of degree $n+1$ as:

$$B_{i,n}(t) = \left(1 - \frac{i}{n+1}\right) B_{i,n+1}(t) + \frac{i+1}{n+1} B_{i+1,n+1}(t) \ ,$$

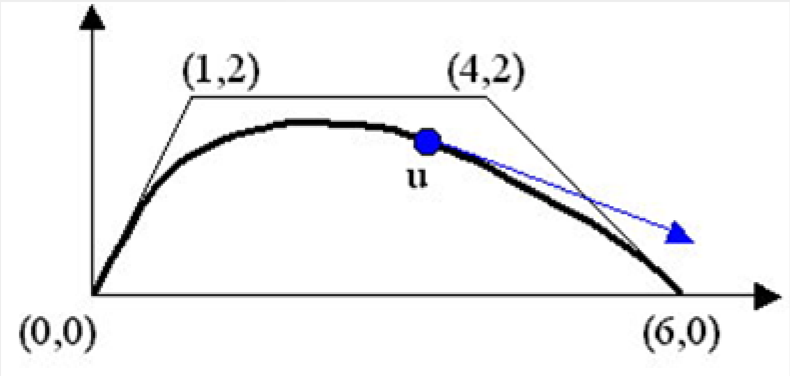
where $i = 0, 1, \cdots, n$. Or more generally in terms of basis functions of degree $n+r$ as:

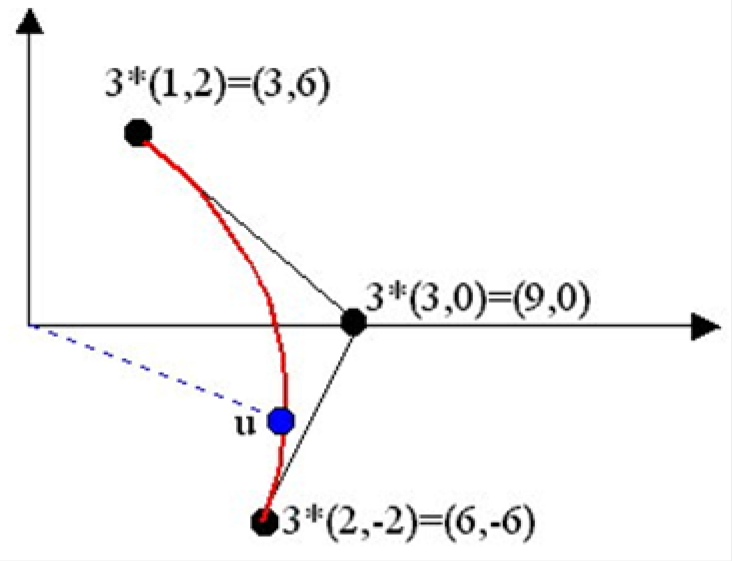
$$B_{i,n}(t) = \sum_{j=i}^{i+r} \frac{\binom{n}{i}\binom{r}{j-i}}{\binom{n+r}{j}} B_{j,n+r}(t), \quad i = 0, 1, \cdots, n \ .$$

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 **$n(P_{i+1}-P_i)$** as a new control point. At **u=0** and **u=1**, the two ends of the Bezier curve, their tangents are **$n(P_1-P_0)$** and **$n(P_n-P_{n-1})$** respectively.





Evaluation of Bezier Curves

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)

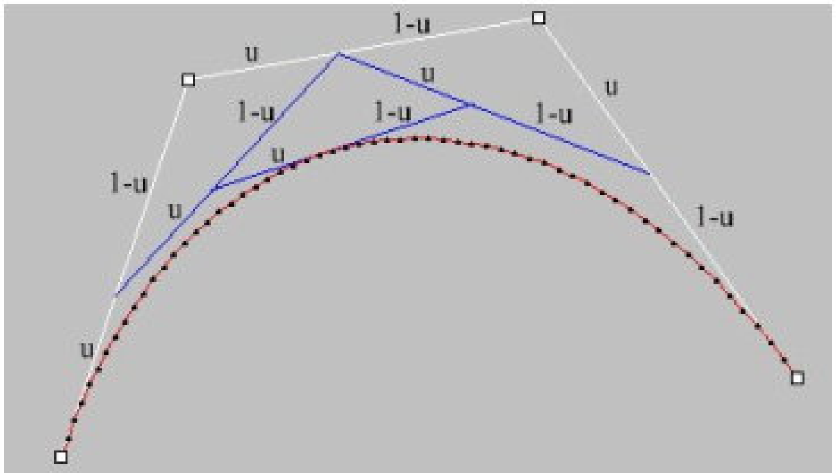
To draw the Bezier curve with 100 points

Number of multiplications and divisions:

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

De Casteljau Algorithm

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

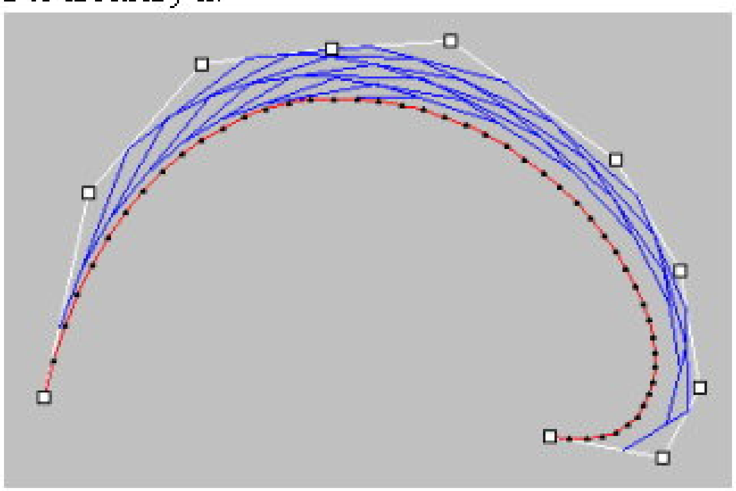


Number of multiplications of a single point:

- 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. 2700 of naïve way)

For arbitrary n:

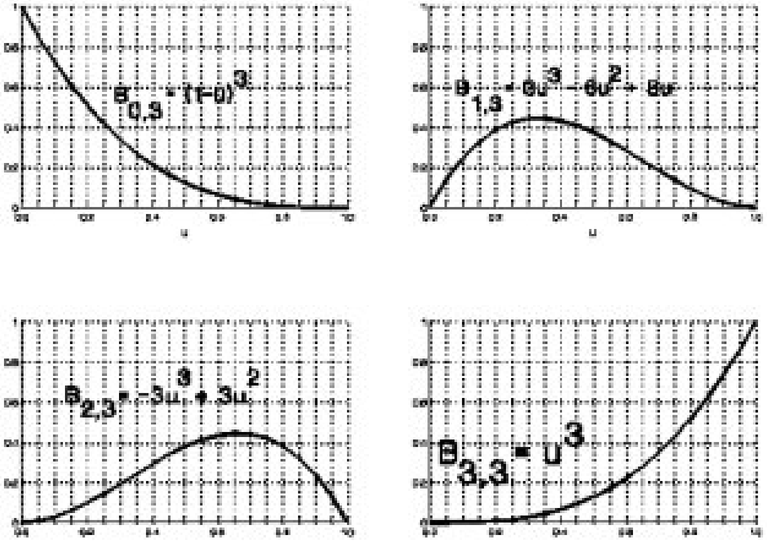


Number of multiplications of a single point:

- 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

Bezier Blending Functions $f_i(u) = B_{i,n}(u)$



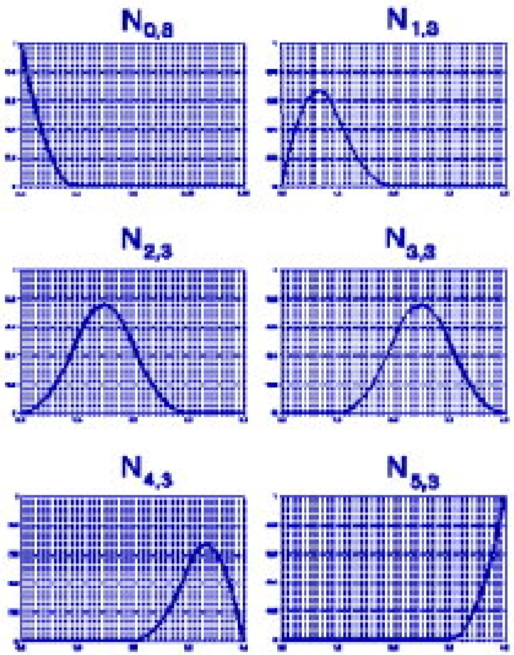
The blending function corresponding to a given point influences the curve maximally near that point.

The Blend for each point is active along the entire length.

The sum of blends anywhere on the curve is 100%.

The end points influence by 100% alone in their respective ends, i.e., the curve interpolated the end points.

Desired Blending Functions $f_i(u) = N_{i,k}(u)$



B-spline blending functions have local control, i.e., they do not influence the whole interval.

This makes it easier to control the curve locally because movement of one control point only propagates to the neighbourhood of the point and not to the whole curve.

Addition and subtraction

If the degrees of the two polynomials are the same, i.e. m = n, we simply add or subtract the coefficients

$$f(t) + g(t) = \sum_{i=0}^{m} (f_i^m \pm g_i^m) B_{i,m}(t)$$

If m > n, we need to first degree elevate g(t), m-n times and then add or subtract the coefficients

$$f(t) + g(t) = \sum_{i=0}^{m} \left(f_i^m \pm \sum_{j=max(0,i-m+n)}^{min(n,i)} \frac{\binom{n}{j}\binom{m-n}{i-j}}{\binom{m}{i}} g_j^n \right) B_{i,m}(t)$$

Multiplication

Multiplication of two polynomials of degree **m** and **n** yields a degree **m+n** polynomial

$$f(t)g(t) = \sum_{i=0}^{m+n} \left(\sum_{j=max(0,i-n)}^{min(m,i)} \frac{\binom{m}{j}\binom{n}{i-j}}{\binom{m+n}{i}} f_j^m g_{i-j}^n \right) B_{i,m+n}(t)$$

Geometry invariance property: Partition of unity property of the Bernstein polynomial assures the invariance of the shape of the Bezier curve **under translation and rotation** of its control points.

End points geometric property:

The first and last control points are the endpoints of the curve. In other words, $\mathbf{b}_0 = \mathbf{r}(0)$ and $\mathbf{b}_n = \mathbf{r}(1)$.
The curve is tangent to the control polygon at the endpoints. This can be easily observed by taking the first derivative of a Bezier curve

$$\dot{\mathbf{r}}(t) = \frac{d\mathbf{r}(t)}{dt} = n\sum_{i=0}^{n-1}(\mathbf{b}_{i+1} - \mathbf{b}_i)B_{i,n-1}(t), \qquad 0 \le t \le 1$$

In particular we have $\dot{\mathbf{r}}(0) = n(\mathbf{b}_1 - \mathbf{b}_0)$ and $\dot{\mathbf{r}}(1) = n(\mathbf{b}_n - \mathbf{b}_{n-1})$. Equation can be simplified by setting $\Delta\mathbf{b}_i = \mathbf{b}_{i+1} - \mathbf{b}_i$:

$$\dot{\mathbf{r}}(t) = n\sum_{i=0}^{n-1}\Delta\mathbf{b}_i B_{i,n-1}(t), \qquad 0 \le t \le 1 \,.$$

Bezier curves

- The order of the degree of curve is variable and related to the number of points defining it. **n+1 points** define an **n'th degree curve** which permit higher-order continuity.
- Control points form the vertices of what is called characteristic polygon. Curve lies entirely within the **convex hull** defined by the polygon vertices.
- Only the **first and the last control points** or vertices of the polygon actually **lie on the curve**. The other vertices define the order, derivatives, and the shape of the curve.
- **The curve is also always tangent to the first and last polygon segments**. In addition, the curve shape tends to follow the polygon shape.

Continuity conditions

One set of continuity conditions are the geometric continuity conditions, designated by the letter **G** with an integer exponent.

Position continuity, or **G^0** continuity, requires the endpoints of the two curves to coincide.

$$\mathbf{r}^a(1) = \mathbf{r}^b(0)$$

The superscripts denote the first and second curves.

G^1 tangent continuity

Tangent continuity, or **G^1** continuity, requires **G^0** continuity and in addition the tangents of the curves to be in the same direction,

$$\dot{\mathbf{r}}^a(1) = \alpha_1 \mathbf{t} \qquad \dot{\mathbf{r}}^b(0) = \alpha_2 \mathbf{t}$$

where **t** is the common unit tangent vector and α_1, α_2 are the magnitude of vectors. **G^1** continuity is important in **minimizing stress concentrations in physical solids** loaded with external forces and in helping **prevent flow separation** in fluids.

G^2 Curvature continuity

Curvature continuity, or **G^2** continuity, requires **G^1** continuity and in addition the center of curvature to move continuously past the connection point.

$$\ddot{\mathbf{r}}^b(0) = \left(\frac{\alpha_2}{\alpha_1}\right)^2 \ddot{\mathbf{r}}^a(1) + \mu\dot{\mathbf{r}}^a(1)$$

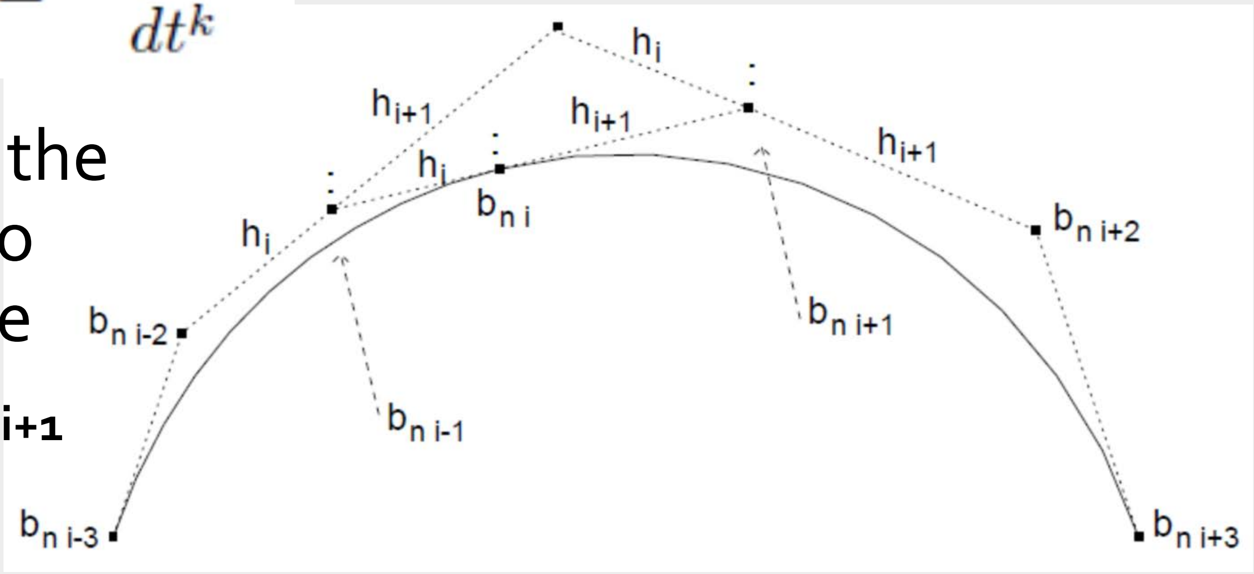
where **μ** is an arbitrary constant. **G^2** continuity is **important for aesthetic reasons** and also for helping prevent fluid flow separation.

C^k continuity (all lower derivatives)

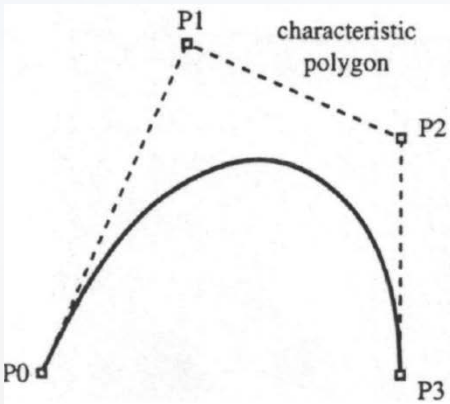
More stringent continuity conditions are the parametric continuity conditions, where $\mathbf{C^k}$ continuity requires the ***k***th derivative (and all lower derivatives) of each curve to be equal at the joining point. In other words,

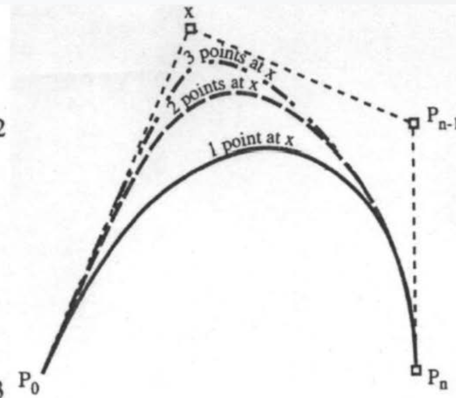
$$\frac{d^k\mathbf{r}^a(1)}{dt^k} = \frac{d^k\mathbf{r}^b(0)}{dt^k}$$

Figure illustrates the connection of two cubic Bezier curve segments at $\mathbf{t = t_{i+1}}$

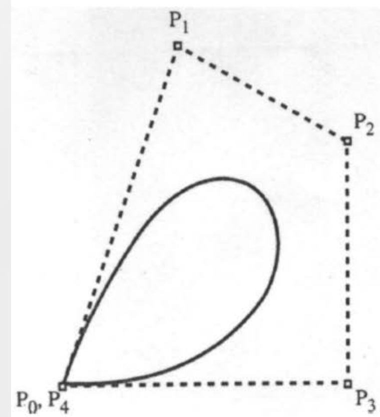


Examples of Bezier Curves

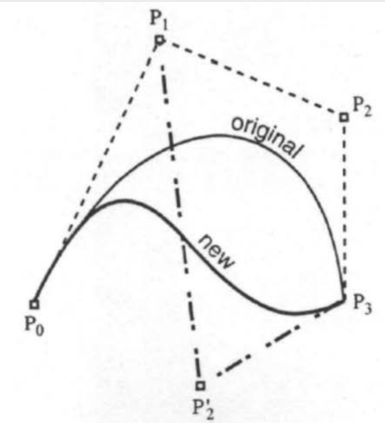




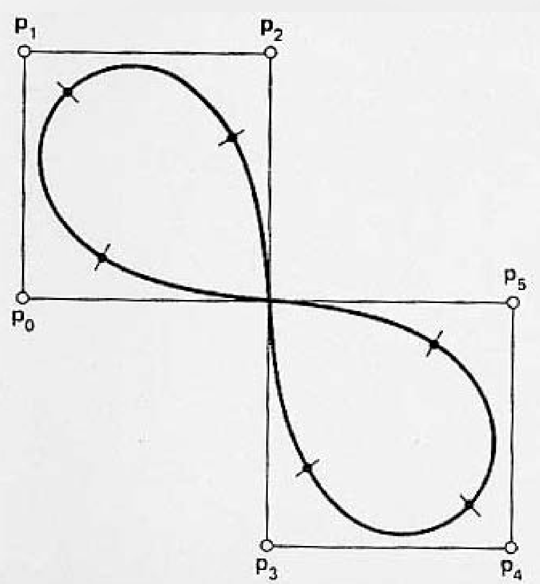
Pull by coincident control points

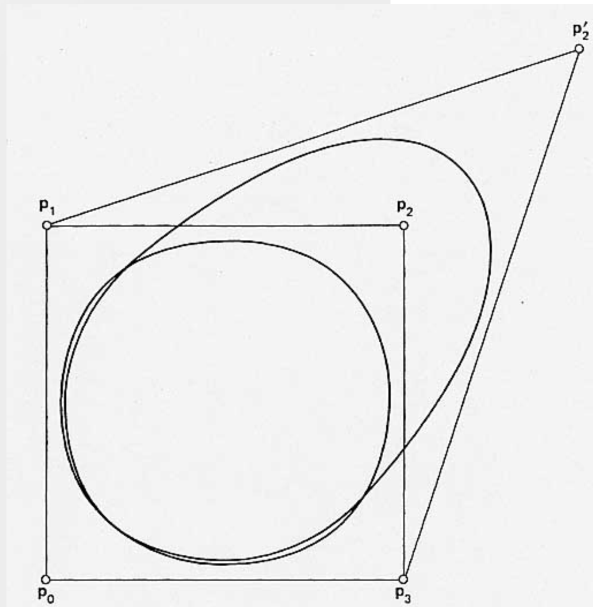


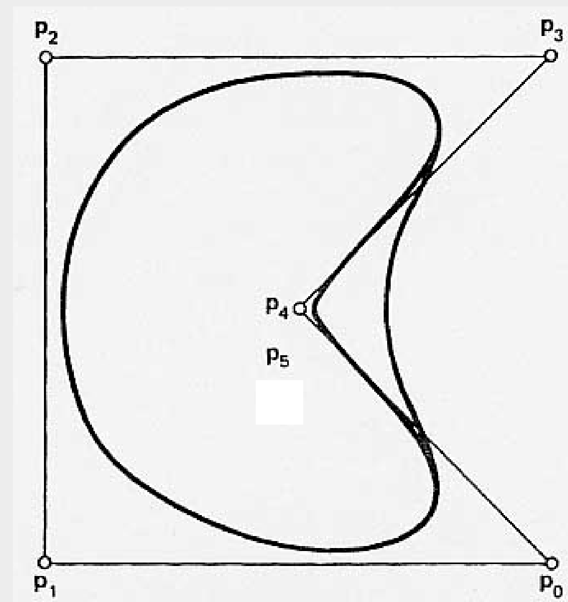
Closed Bezier curve



Influence of point position







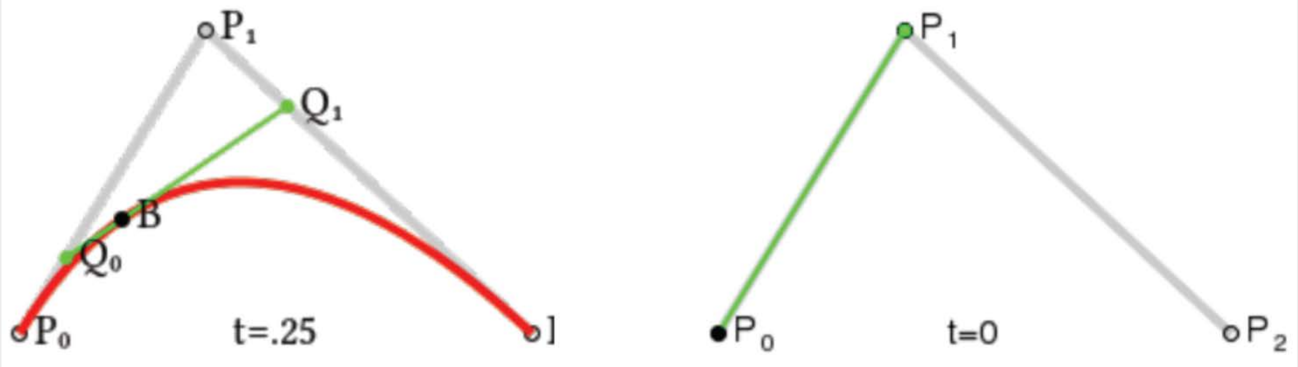
Quadratic Bézier curves

A quadratic Bézier curve is the path traced by the function **B**(*t*), given points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$,

$$\mathbf{B}(t) = (1-t)[(1-t)\mathbf{P}_0 + t\mathbf{P}_1] + t[(1-t)\mathbf{P}_1 + t\mathbf{P}_2] \ , \ t \in [0,1]$$

which can be interpreted as the linear interpolant of corresponding points on the linear Bézier curves from $\mathbf{P}_0$ to $\mathbf{P}_1$ and from $\mathbf{P}_1$ to $\mathbf{P}_2$ respectively. Rearranging the preceding equation yields:

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2(1-t)t\mathbf{P}_1 + t^2\mathbf{P}_2 \ , \ t \in [0,1].$$



Cubic Bézier curves

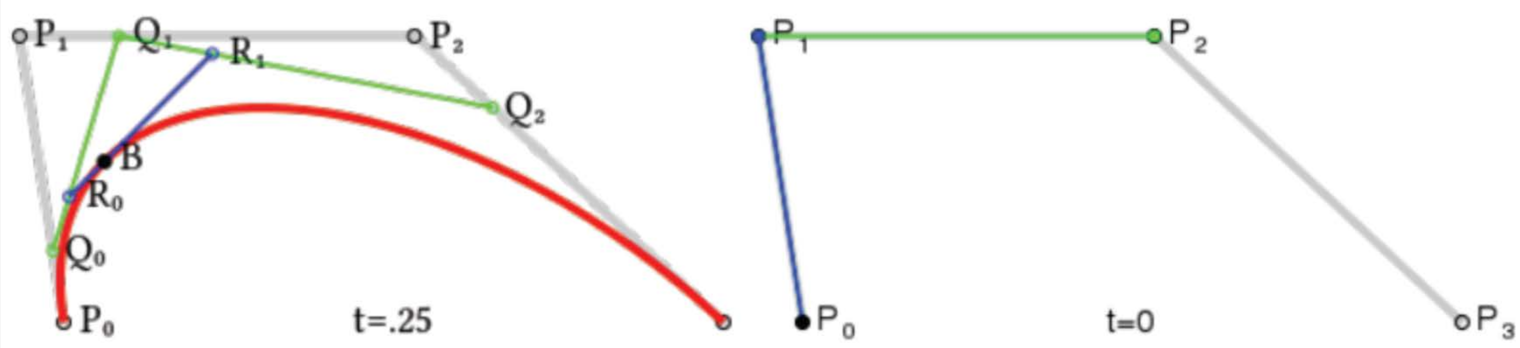
Writing $\mathbf{B}_{\mathbf{P}i,\mathbf{P}j,\mathbf{P}k}(t)$ for the quadratic Bézier curve defined by points $\mathbf{P}_i$, $\mathbf{P}_j$, and $\mathbf{P}_k$, the cubic Bézier curve can be defined as a linear combination of two quadratic Bézier curves:

$$\mathbf{B}(t) = (1-t)\mathbf{B}_{\mathbf{P}_0,\mathbf{P}_1,\mathbf{P}_2}(t) + t\mathbf{B}_{\mathbf{P}_1,\mathbf{P}_2,\mathbf{P}_3}(t) \ , \ t \in [0,1].$$

The explicit form of the curve is:

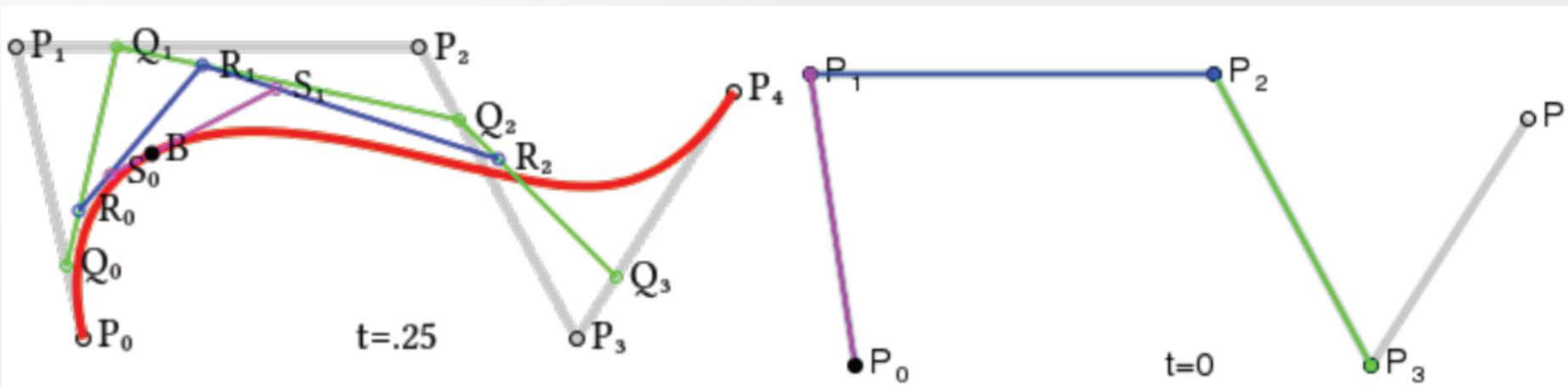
$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3(1-t)^2 t\mathbf{P}_1 + 3(1-t)t^2\mathbf{P}_2 + t^3\mathbf{P}_3 \ , \ t \in [0,1]$$

For some choices of $\mathbf{P}_1$ and $\mathbf{P}_2$ the curve may intersect itself, or contain a cusp.



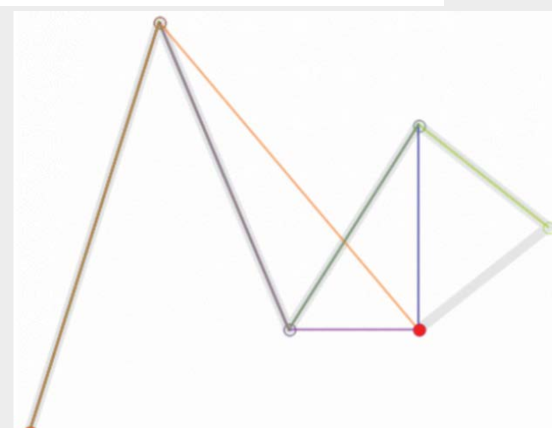
Quartic Bezier Curve

For fourth-order curves one can construct intermediate points $\mathbf{Q}_0$, $\mathbf{Q}_1$, $\mathbf{Q}_2$ & $\mathbf{Q}_3$ that describe linear Bézier curves, points $\mathbf{R}_0$, $\mathbf{R}_1$ & $\mathbf{R}_2$ that describe quadratic Bézier curves, and points $\mathbf{S}_0$ & $\mathbf{S}_1$ that describe cubic Bézier curves:

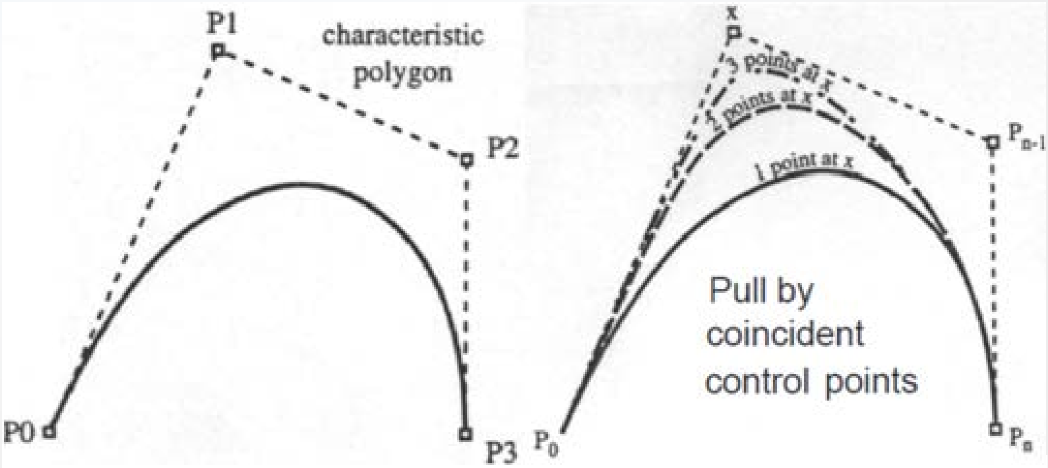


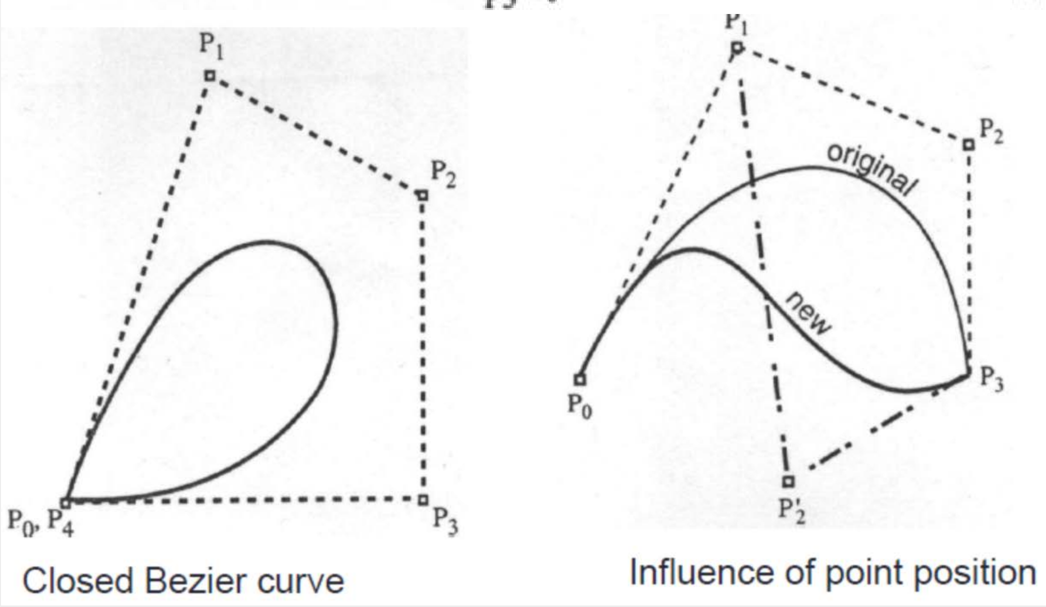
Fifth Order Bezier Curve

For fifth-order curves, one can construct similar intermediate points.

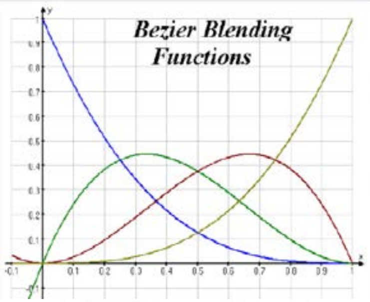


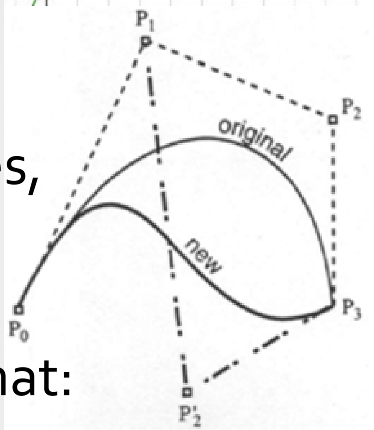
Bezier curve defined by 4 points





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.

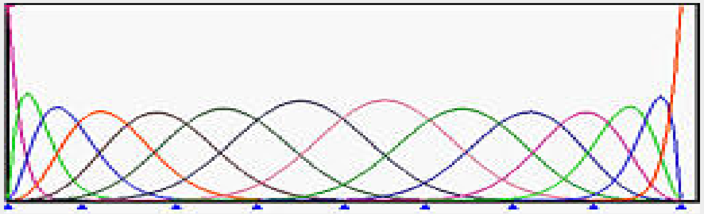
What we need is some blending function $f_i(u)$ such that:

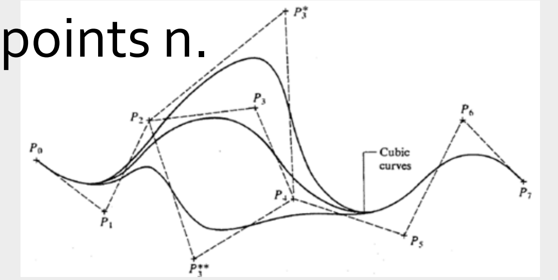
1. It is non-zero over only a limited portion of the parameter interval of the entire curve, and this limited portion is different for each blending function. (Therefore, when $\mathbf{P}_i$ moves, it only affects a limited portion of the curve.)

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

Answer:

B-Splines

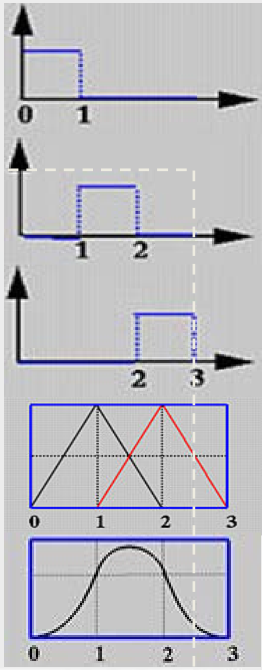




Let's Search for a Good Blending Function $f_i(u)$

$\mathbf{P}(u) = \mathbf{f}_0(u)\mathbf{P}_0 + f_1(u)\mathbf{P}_1 + \mathbf{f}_2(u)\mathbf{P}_2 + \ldots + \mathbf{f}_n(u)\mathbf{P}_n \quad (u_{min} \leq u \leq u_{max})$

Two properties that $f_i(u)$ must have: (1) $f_i(u) \geq 0$ for all *i* and (2) $\sum_{i=0}^{n-1} f_i(u) = 1$ for any $u \in [0,1]$



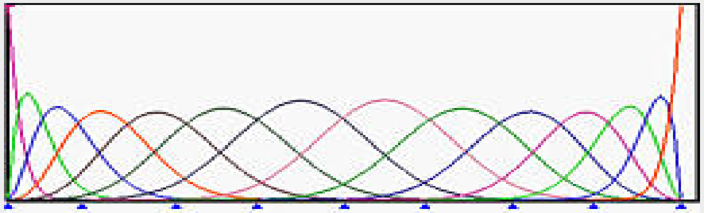
Step 1. Allocate an interval of u for each $f_i(u)$,
say we let $u_{min}=0$, $u_{max}=n+1$, and interval $[i,i+1]$ is for $f_i(u)$.

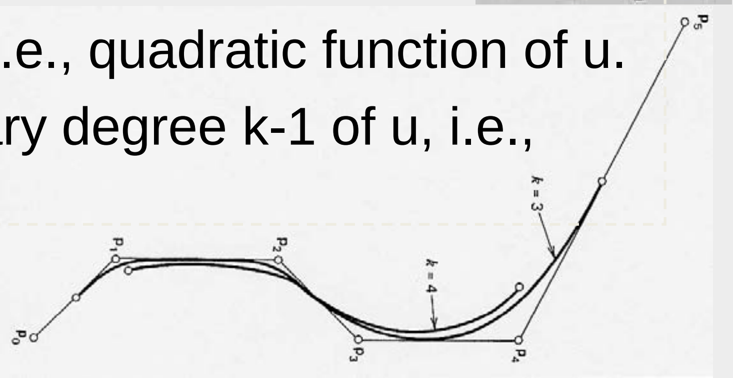
Step 2. Look at the simplest case of $f_i(u)$, i.e., $f_i(u)$ is of degree 0, a constant. Let this special blending function be called $N_{i,1}(u)$ (where 1 indicates the *order* of $f_i(u)$).

Step 3. Look at the 2nd simplest case of $f_i(u)$, i.e., $f_i(u)$ is linear; it is now denoted as $N_{i,2}(u)$.

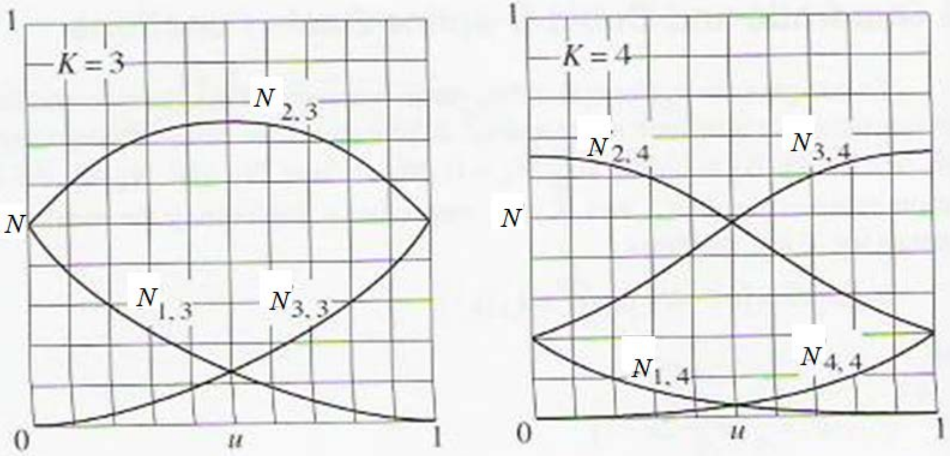
Step 4. One more step further, $f_i(u) = N_{i,3}(u)$, i.e., quadratic function of u.

Step 5. Finally, how about $f_i(u)$ is of an arbitrary degree k-1 of u, i.e., $N_{i,k}(u)$?





B-Spline $\mathbf{p}(u) = \sum_{i=0}^{n} N_{i,K}(u)\mathbf{p}_i$



Uniform B-spline basis functions on the unit interval for $K = 3$ and $K = 4$.

Geometric form (non-uniform, non-rational case), where K controls degree (K -1) of basis functions:

$$\sum_{i=0}^{n} N_{i,K}(u) = 1$$

Convex combination, so B-spline curve points all lie within convex hull of control polygon.

$$N_{i,1}(u) = 1 \quad \text{if} \quad t_i \le u \le t_{i+1}$$
$$N_{i,1}(u) = 0 \quad \text{otherwise}$$

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

t_i are n+1+K knot values that relate u to the control points.

Uniform case: space knots at equal intervals of u. Repeated knots move curve closer to control points. Cubic B-splines can provide C^2 continuity at curve segment join points.

Rational form (NURBS) where h_i are weights.

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

Non-periodic and uniform B-Spline

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***:

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

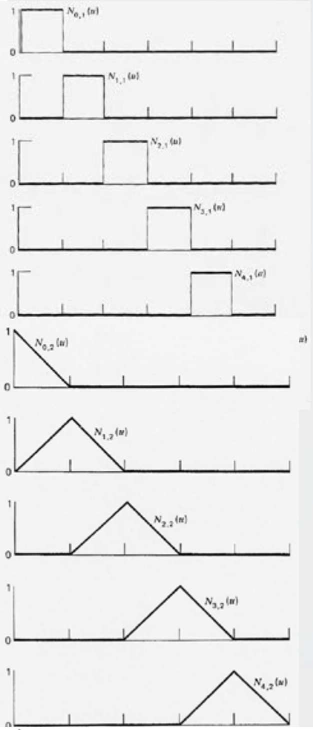
- Step 3. Compute the n+1 blending function $N_{i,k}(u)$ recursively:

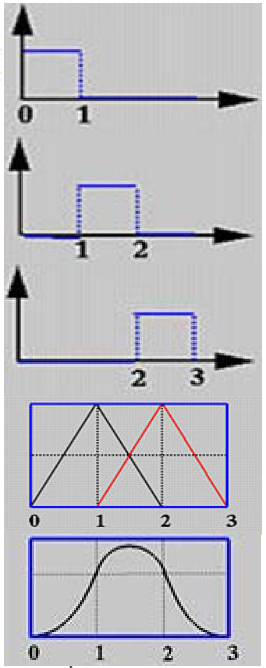
$$N_{i,1}(u) = \begin{cases} 1 & t_i \le u \le t_{i+1} \\ 0 & otherwise \end{cases}$$

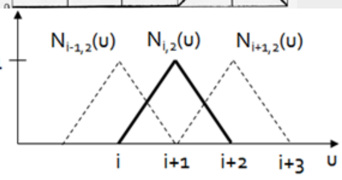
$$N_{i,k}(u) = \frac{(u-t_i)N_{i,k-1}(u)}{t_{i+k-1}-t_i} + \frac{(t_{i+k}-u)N_{i+1,k-1}(u)}{t_{i+k}-t_{i+1}}$$

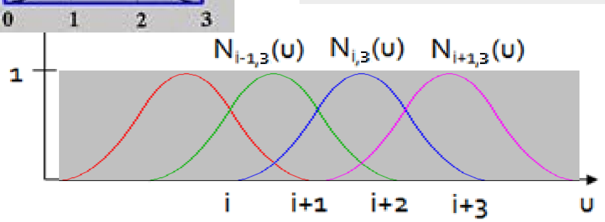
- Step 4. Put together:

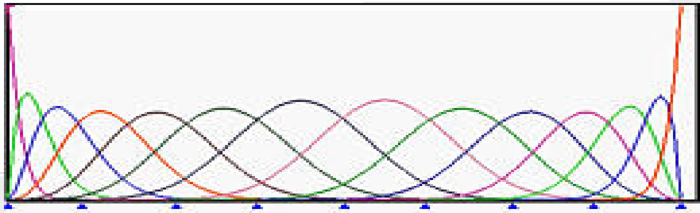
$$\mathbf{P}(u) = \sum_{i=0}^{n} N_{i,k}(u)\,\mathbf{P}_i \qquad (t_{k-1} \le u \le t_{n+1})$$











Properties of B-Spline

- 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

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

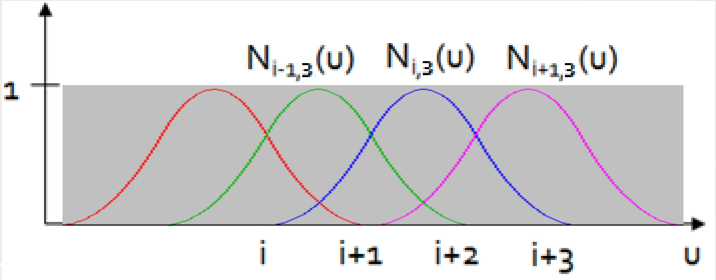
where

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

- The most important feature of B-spline only

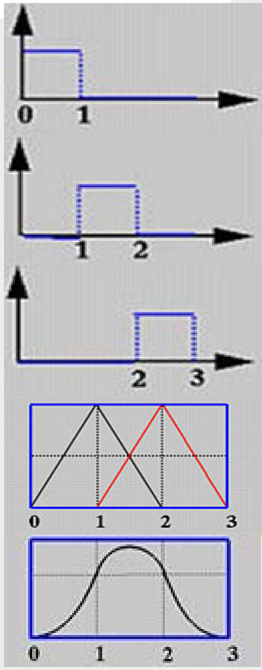
> **$N_{i,k}(u)$ is non-zero only in the interval** $[t_i \le u \le t_{i+k})$.
>
> **Change of P_i therefore only affects that portion of curve.**

B-Spline Curve



Similar to Bezier curves, the B-spline curve defined by $n + 1$ control points P_i First, the parameter k controls the degree $(k - 1)$ of the resulting B-spline curve and is usually independent of the number of control points except as restricted as shown below. Second, the maximum limit of the parameter u is no longer unity as it was so chosen arbitrarily for Bezier curves. The B-spline functions have the following properties:

$$\mathbf{P}(u) = \sum_{i=0}^{n} \mathbf{P}_i N_{i,k}(u), \qquad 0 \le u \le u_{\max} \qquad (5.103)$$



Partition of unity: $\sum_{i=0}^{n} N_{i,k}(u) = 1$

Positivity: $N_{i,k}(u) \ge 0$

Local support: $N_{i,k}(u) = 0 \qquad \text{if } u \notin [u_i, u_{i+k+1}]$

Continuity: $N_{i,k}(u)$ is $(k-2)$ times continuously differentiable

B-Spline Curve

The first property ensures that the relationship between the curve and its defining control points is invariant under affine transformations. The second property guarantees that the curve segment lies completely within the convex hull of P_i. The third property indicates that each segment of a B-spline curve is influenced by only k control points or each control point affects only k curve segments. It is useful to notice that the Bernstein polynomial, $B_{i,n}(u)$, has the same first two properties mentioned above.

The B-spline function also has the property of recursion which is defined as

$$N_{i,k}(u) = (u - u_i)\frac{N_{i,k-1}(u)}{u_{i+k-1} - u_i} + (u_{i+k} - u)\frac{N_{i+1,k-1}(u)}{u_{i+k} - u_{i+1}} \qquad (5.104)$$

where $$N_{i,1} = \begin{cases} 1, & u_i \leq u \leq u_{i+1} \\ 0, & \text{otherwise} \end{cases} \qquad (5.105)$$

$$N_{i,k}(u) = (u - u_i)\frac{N_{i,k-1}(u)}{u_{i+k-1} - u_i} + (u_{i+k} - u)\frac{N_{i+1,k-1}(u)}{u_{i+k} - u_{i+1}} \quad (5.104)$$

B-Spline Curve

$$N_{i,1} = \begin{cases} 1, & u_i \leq u \leq u_{i+1} \\ 0, & \text{otherwise} \end{cases} \quad (5.105)$$

Choose $0/0 = 0$ if the denominators in Eq. (5.104) become zero. Equation (5.105) shows that $N_{i,1}$ is a unit step function.

Because $N_{i,1}$ is constant for $k = 1$, a general value of k produces a polynomial in u of degree $(k - 1)$ [see Eq. (5.104)] and therefore a curve of order k and degree $(k - 1)$. The u_i are called parametric knots or knot values. These values form a sequence of nondecreasing integers called the knot vector. The values of the u_i depend on whether the B-spline curve is an open (nonperiodic) or closed (periodic) curve. For an open curve, they are given by

$$u_j = \begin{cases} 0, & j < k \\ j - k + 1, & k \leq j \leq n \\ n - k + 2, & j > n \end{cases} \quad (5.106)$$

where $\quad 0 \leq j \leq n + k \quad (5.107)$

and the range of u is

$$0 \leq u \leq n - k + 2 \quad (5.108)$$

B-Spline Curve

$$u_j = \begin{cases} 0, & j < k \\ j - k + 1, & k \le j \le n \\ n - k + 2, & j > n \end{cases} \qquad (5.106)$$

where $$0 \le j \le n + k \qquad (5.107)$$

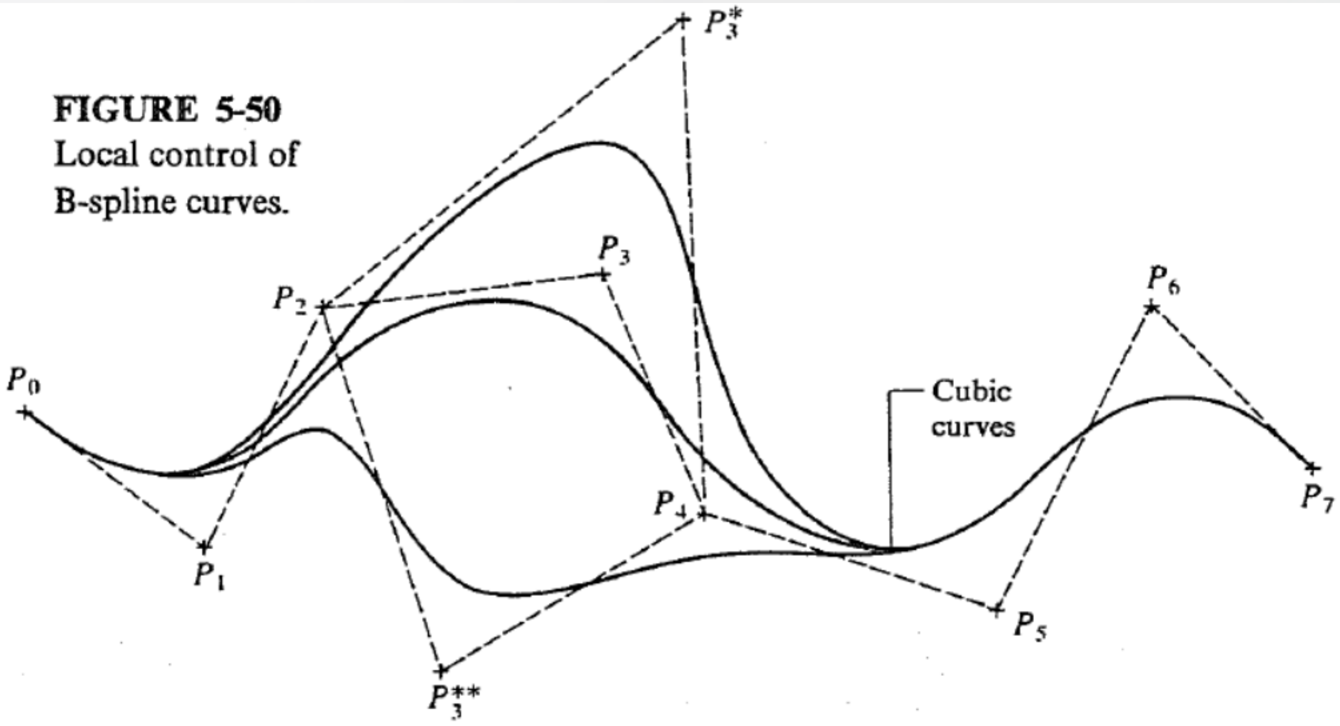
and the range of u is

$$0 \le u \le n - k + 2 \qquad (5.108)$$

$$\mathbf{P}(u) = \sum_{i=0}^{n} \mathbf{P}_i N_{i,k}(u), \qquad 0 \le u \le u_{\max}$$

Relation (5.107) shows that $(n + k + 1)$ knots are needed to create a $(k - 1)$ degree curve defined by $(n + 1)$ control points. These knots are evenly spaced over the range of u with unit separation ($\Delta u = 1$) between noncoincident knots. Multiple (coincident) knots for certain values of u may exist.

FIGURE 5-50
Local control of B-spline curves.



B-Spline Curve

where $\quad 0 \leq j \leq n + k \qquad (5.107)$

and the range of u is

$$0 \leq u \leq n - k + 2 \qquad (5.108)$$

While the degree of the resulting B-spline curve is controlled by k, the range of the parameter u as given by Eq. (5.108) implies that there is a limit on k that is determined by the number of the given control points. This limit is found by requiring the upper bound in Eq. (5.108) to be greater than the lower bound for the u range to be valid, that is,

$$n - k + 2 > 0 \qquad (5.109)$$

This relation shows that a minimum of two, three, and four control points are required to define a linear, quadratic, and cubic B-spline curve respectively.

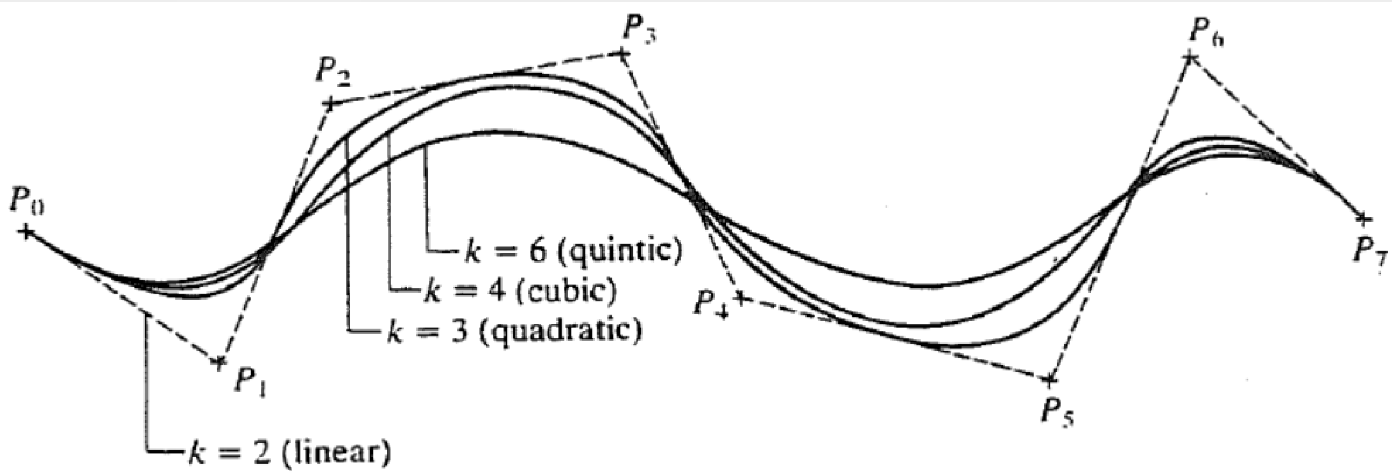
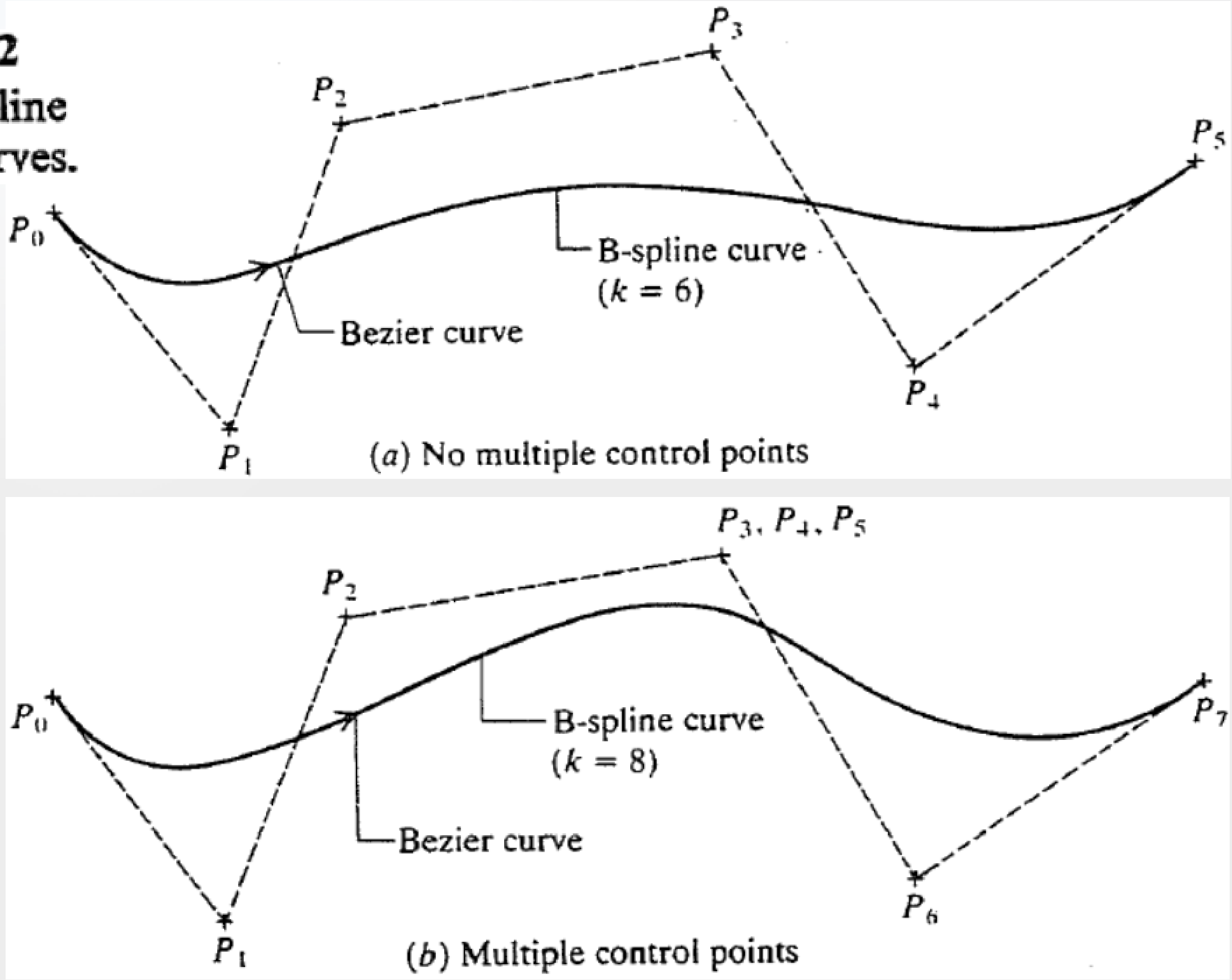


FIGURE 5-51 Effect of the degree of B-spline curve on its shape.

B-Spline Curve

FIGURE 5-52
Identical B-spline and Bezier curves.



B-Spline Curve

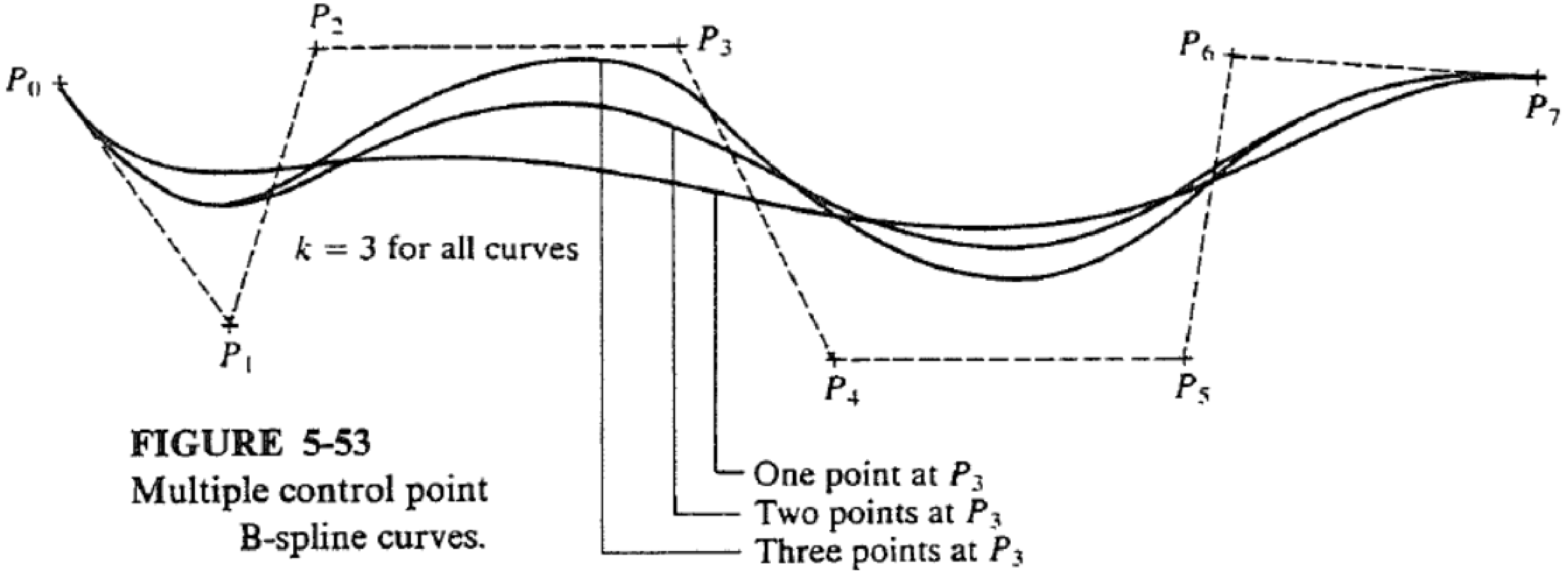


FIGURE 5-53
Multiple control point B-spline curves.

B-Spline

The characteristics of B-spline curves that are useful in design can be summarized as follows:

1. The local control of the curve can be achieved by changing the position of a control point(s), using multiple control points by placing several points at the same location, or by choosing a different degree $(k - 1)$. As mentioned earlier, changing one control point affects only k segments. Figure 5-50 shows the local control for a cubic B-spline curve by moving P_3 to P_3^* and P_3^{**}. The four curve segments surrounding P_3 change only.
2. A nonperiodic B-spline curve passes through the first and last control points P_0 and P_{n+1} and is tangent to the first $(P_1 - P_0)$ and last $(P_{n+1} - P_n)$ segments of the control polygon, similar to the Bezier curve, as shown in Fig. 5-50.
3. Increasing the degree of the curve tightens it. In general, the less the degree, the closer the curve gets to the control points, as shown in Fig. 5-51. When $k = 1$, a zero-degree curve results. The curve then becomes the control points themselves. When $k = 2$, the curve becomes the polygon segments themselves.
4. A second-degree curve is always tangent to the midpoints of all the internal polygon segments (see Fig. 5-51). This is not the case for other degrees.
5. If k equals the number of control points $(n + 1)$, then the resulting B-spline curve becomes a Bezier curve (see Fig. 5-52). In this case the range of u becomes zero to one [see Eq. (5.108)] as expected.
6. Multiple control points induce regions of high curvature of a B-spline curve. This is useful when creating sharp corners in the curve (see Fig. 5-53). This effect is equivalent to saying that the curve is pulled more towards a control point by increasing its multiplicity.
7. Increasing the degree of the curve makes it more difficult to control and to calculate accurately. Therefore, a cubic B-spline is sufficient for a large number of applications.

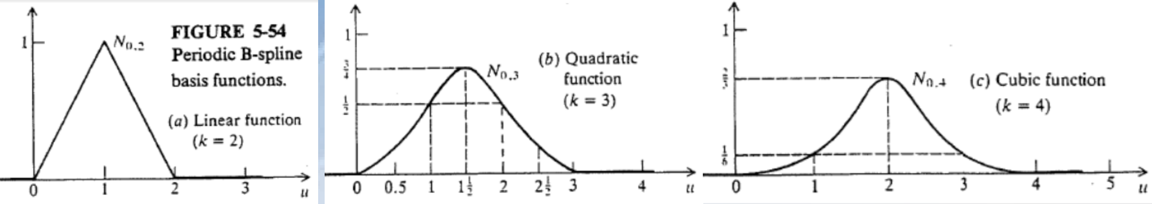


FIGURE 5-54 Periodic B-spline basis functions.

where $$0 \le j \le n + k \qquad (5.107)$$

and the range of u is

$$0 \le u \le n - k + 2 \qquad (5.108)$$

Closed B-Spline Curve

Thus far, open or nonperiodic B-spline curves have been discussed. The same theory can be extended to cover closed or periodic B-spline curves. The only difference between open and closed curves is in the choice of the knots and the basic functions. Equations (5.106) to (5.108) determine the knots and the spacing between them for open curves. Closed curves utilize periodic B-spline functions as their basis with knots at the integers. These basis functions are cyclic translates of a single canonical function with a period (interval) of k for support. For example, for a closed B-spline curve of order 2 ($k = 2$) or a degree 1 ($k - 1$), the basis function is linear, has a nonzero value in the interval (0, 2) only, and has a maximum value of one at $u = 1$, as shown in Fig. 5-54. The knot vector in this case is $[0 \quad 1 \quad 2]$. Quadratic and cubic closed curves have quadratic and cubic basis functions with intervals of (0, 3) and (0, 4) and knot vectors of $[0 \quad 1 \quad 2 \quad 3]$ and $[0 \quad 1 \quad 2 \quad 3 \quad 4]$ respectively.

The closed B-spline curve of degree $(k - 1)$ or order k defined by $(n + 1)$ control points is given by Eq. (5.103) as the open curve. However, for closed curves Eqs. (5.104) to (5.108) become

$$N_{i,1} = \begin{cases} 1, & u_i \le u \le u_{i+1} \\ 0, & \text{otherwise} \end{cases} \quad (5.105)$$

$$u_j = \begin{cases} 0, & j < k \\ j - k + 1, & k \le j \le n \\ n - k + 2, & j > n \end{cases} \quad \text{where} \quad 0 \le j \le n + k \quad (5.107)$$

and the range of u is

$$0 \le u \le n - k + 2 \quad (5.108)$$

Closed B-Spline Curve

$$\mathbf{P}(u) = \sum_{i=0}^{n} \mathbf{P}_i N_{i,k}(u), \qquad 0 \le u \le u_{\max} \quad (5.103)$$

$$N_{i,k}(u) = (u - u_i)\frac{N_{i,k-1}(u)}{u_{i+k-1} - u_i} + (u_{i+k} - u)\frac{N_{i+1,k-1}(u)}{u_{i+k} - u_{i+1}} \quad (5.104)$$

The closed B-spline curve of degree $(k - 1)$ or order k defined by $(n + 1)$ control points is given by Eq. (5.103) as the open curve. However, for closed curves Eqs. (5.104) to (5.108) become

$$N_{i,k}(u) = N_{0,k}((u - i + n + 1) \bmod (n + 1)) \quad (5.110)$$

$$u_j = j, \qquad 0 \le j \le n + 1 \quad (5.111)$$

$$0 < j \le n + 1 \quad (5.112)$$

and the range of u is

$$0 \le u \le n + 1 \quad (5.113)$$

The mod $(n + 1)$ in Eq. (5.110) is the modulo function. It is defined as

$$A \bmod n = \begin{cases} A, & A < n \\ 0, & A = n \\ \text{remainder of } A/n, & A > n \end{cases} \quad (5.114)$$

For example, $3.5 \bmod 6 = 3.5$, $6 \bmod 6 = 0$, and $7 \bmod 6 = 1$. The mod function enables the periodic (cyclic) translation [mod $(n + 1)$] of the canonical basis function $N_{0,k}$. $N_{0,k}$ is the same as for open curves and can be calculated using Eqs. (5.104) and (5.105).

Closed B-Spline Curve

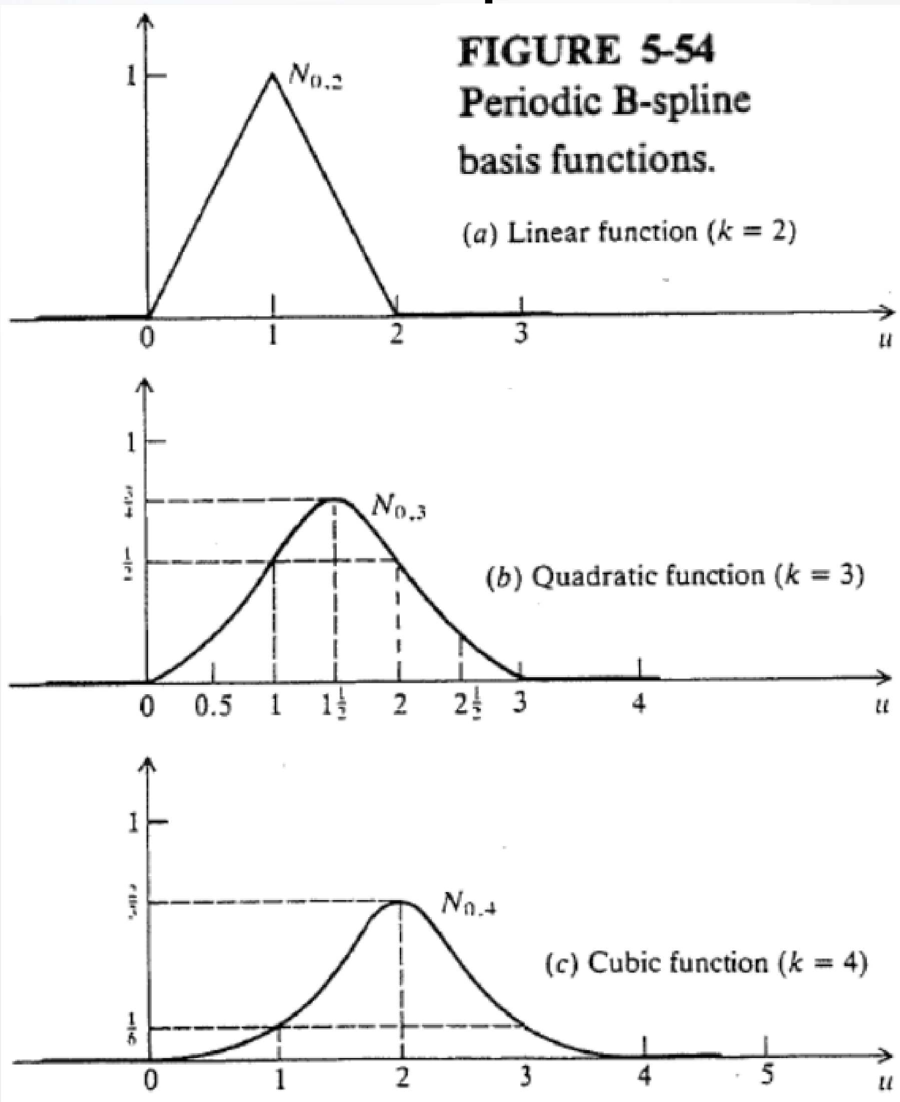


FIGURE 5-54 Periodic B-spline basis functions.

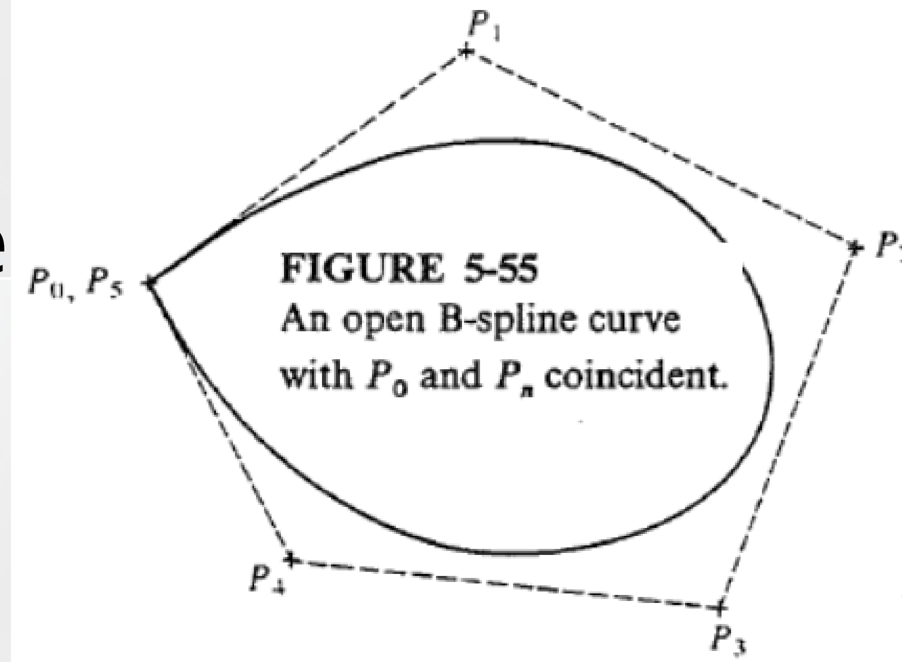
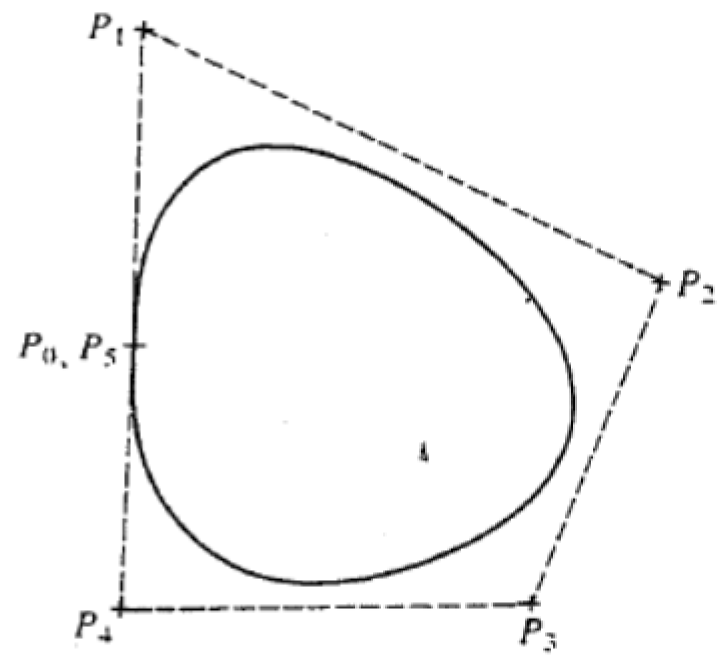


FIGURE 5-55 An open B-spline curve with P_0 and P_n coincident.



(b) C^1 continuity

Closed B-Spline Curve

Like open curves, closed B-spline curves enjoy the properties of partition of unity, positivity, local support, and continuity. They also share the same characteristics of the open curves except that they do not pass through the first and last control points and therefore are not tangent to the first and last segments of the control polygon. In representing closed curves, closed polygons are used where the first and last control points are connected by a polygon segment. It should be noticed that a closed B-spline curve can not be generated by simply using an open curve with the first and last control points being the same (coincident). The resulting curve is only C^0 continuous, as shown in Fig. 5-55. Only if the first and last segments of the polygon are colinear does a C^1 continuous curve result as in a Bezier curve.

Based on the above theory, the database of a B-spline curve includes the type of curve (open or closed), its order k or degree $(k - 1)$, and the coordinates of the control points defining its polygon stored in the same order as input by the user. Other information such as layer, color, name, font, and line width of the curve may be stored.

Example 5.19. The coordinates of four control points relative to a current WCS are given by

$$\mathbf{P}_0 = [2 \quad 2 \quad 0]^T, \quad \mathbf{P}_1 = [2 \quad 3 \quad 0]^T, \quad \mathbf{P}_2 = [3 \quad 3 \quad 0]^T, \quad \text{and} \quad \mathbf{P}_3 = [3 \quad 2 \quad 0]^T$$

Find the equation of the resulting Bezier curve. Also find points on the curve for $u = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$, and 1.

Solution. Equation (5.91) gives

$$\mathbf{P}(u) = \mathbf{P}_0 B_{0,3} + \mathbf{P}_1 B_{1,3} + \mathbf{P}_2 B_{2,3} + \mathbf{P}_3 B_{3,3}, \qquad 0 \le u \le 1$$

Using Eqs. (5.92) and (5.93), the above equation becomes

$$\mathbf{P}(u) = \mathbf{P}_0(1-u)^3 + 3\mathbf{P}_1 u(1-u)^2 + 3\mathbf{P}_2 u^2(1-u) + \mathbf{P}_3 u^3, \qquad 0 \le u \le 1$$

Substituting the u values into this equation gives

$$\mathbf{P}(0) = \mathbf{P}_0 = [2 \quad 2 \quad 0]^T$$

$$\mathbf{P}\left(\frac{1}{4}\right) = \frac{27}{64}\mathbf{P}_0 + \frac{27}{64}\mathbf{P}_1 + \frac{9}{64}\mathbf{P}_2 + \frac{1}{64}\mathbf{P}_3 = [2.156 \quad 2.563 \quad 0]^T$$

$$\mathbf{P}\left(\frac{1}{2}\right) = \frac{1}{8}\mathbf{P}_0 + \frac{3}{8}\mathbf{P}_1 + \frac{3}{8}\mathbf{P}_2 + \frac{1}{8}\mathbf{P}_3 = [2.5 \quad 2.75 \quad 0]^T$$

$$\mathbf{P}\left(\frac{3}{4}\right) = \frac{1}{64}\mathbf{P}_0 + \frac{9}{64}\mathbf{P}_1 + \frac{27}{64}\mathbf{P}_2 + \frac{27}{64}\mathbf{P}_3 = [2.844 \quad 2.563 \quad 0]^T$$

$$\mathbf{P}(1) = \mathbf{P}_3 = [3 \quad 2 \quad 0]^T$$

Observe that $\sum_{i=0}^{3} B_{i,3}$ is always equal to unity for any u value. Figure 5-49 shows the curve and the points.

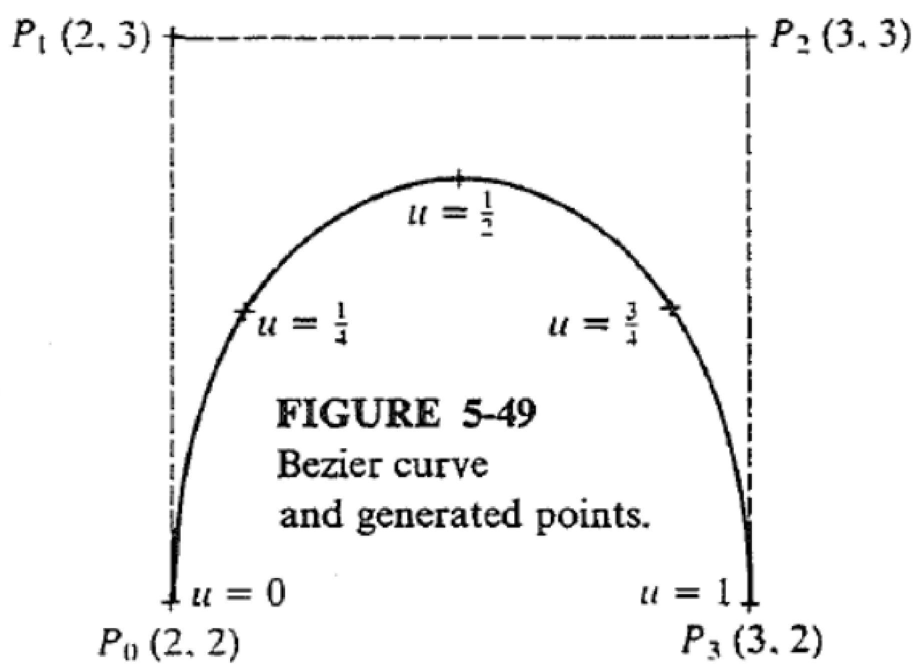


FIGURE 5-49
Bezier curve
and generated points.

Example 5.20. A cubic spline curve is defined by the equation

$$\mathbf{P}(u) = \mathbf{C}_3 u^3 + \mathbf{C}_2 u^2 + \mathbf{C}_1 u + \mathbf{C}_0, \qquad 0 \le u \le 1 \tag{5.100}$$

where $\mathbf{C}_3$, $\mathbf{C}_2$, $\mathbf{C}_1$, and $\mathbf{C}_0$ are the polynomial coefficients [see Eq. (5.76)]. Assuming these coefficients are known, find the four control points that define an identical Bezier curve.

Solution. The Bezier equation is

$$\mathbf{P}(u) = \mathbf{P}_0 B_{0,3} + \mathbf{P}_1 B_{1,3} + \mathbf{P}_2 B_{2,3} + \mathbf{P}_3 B_{3,3} \tag{5.101}$$

where

$$B_{0,3} = 1 - 3u + 3u^2 - u^3 \qquad B_{1,3} = 3u - 6u^2 + 3u^3$$

$$B_{2,3} = 3u^2 - 3u^3 \qquad B_{3,3} = u^3$$

Substituting all these functions into Eq. (5.101) and rearranging, we obtain

$$\mathbf{P}(u) = (-\mathbf{P}_0 + 3\mathbf{P}_1 - 3\mathbf{P}_2 + \mathbf{P}_3)u^3 + (3\mathbf{P}_0 - 6\mathbf{P}_1 + 3\mathbf{P}_2)u^2 + (-3\mathbf{P}_0 + 3\mathbf{P}_1)u + \mathbf{P}_0 \tag{5.102}$$

Comparing the coefficients of Eqs. (5.100) and (5.102) gives

$$\mathbf{P}_0 = \mathbf{C}_0$$

$$\mathbf{P}_1 = \tfrac{1}{3}\mathbf{C}_1 + \mathbf{C}_0$$

$$\mathbf{P}_2 = \tfrac{1}{3}(\mathbf{C}_2 + 2\mathbf{C}_1 + 3\mathbf{C}_0)$$

$$\mathbf{P}_3 = \mathbf{C}_3 + \mathbf{C}_2 + \mathbf{C}_1 + \mathbf{C}_0$$

Example 5.21. Find the equation of a cubic B-spline curve defined by the same control points as in Example 5.19. How does the curve compare with the Bezier curve?

Solution. This cubic spline has $k = 4$ and $n = 3$. Eight knots are needed to calculate the B-spline functions. Equation (5.106) gives the knot vector

$$[u_0 \quad u_1 \quad u_2 \quad u_3 \quad u_4 \quad u_5 \quad u_6 \quad u_7] \quad \text{as} \quad [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1]$$

The range of u [Eq. (5.108)] is $0 \leq u \leq 1$. Equation (5.103) gives

$$\mathbf{P}(u) = \mathbf{P}_0 N_{0,4} + \mathbf{P}_1 N_{1,4} + \mathbf{P}_2 N_{2,4} + \mathbf{P}_3 N_{3,4}, \qquad 0 \leq u \leq 1 \tag{5.115}$$

To calculate the above B-spline functions, use Eqs. (5.104) and (5.105) together with the knot vector as follows:

$$N_{0,1} = N_{1,1} = N_{2,1} = \begin{cases} 1, & u = 0 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{3,1} = \begin{cases} 1, & 0 \leq u \leq 1 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{4,1} = N_{5,1} = N_{6,1} = \begin{cases} 1, & u = 1 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{0,2} = (u - u_0)\frac{N_{0,1}}{u_1 - u_0} + (u_2 - u)\frac{N_{1,1}}{u_2 - u_1} = \frac{uN_{0,1}}{0} + \frac{(-u)N_{1,1}}{0} = 0$$

$$N_{1,2} = (u - u_1)\frac{N_{1,1}}{u_2 - u_1} + (u_3 - u)\frac{N_{2,1}}{u_3 - u_2} = \frac{uN_{1,1}}{0} + \frac{(-u)N_{2,1}}{0} = 0$$

$$N_{2,2} = (u - u_2)\frac{N_{2,1}}{u_3 - u_2} + (u_4 - u)\frac{N_{3,1}}{u_4 - u_3} = \frac{uN_{2,1}}{0} + \frac{(1 - u)N_{3,1}}{1} = (1 - u)N_{3,1}$$

$$N_{3,2} = (u - u_3)\frac{N_{3,1}}{u_4 - u_3} + (u_5 - u)\frac{N_{4,1}}{u_5 - u_4} = uN_{3,1} + \frac{(1 - u)N_{4,1}}{0} = uN_{3,1}$$

$$N_{4,2} = (u - u_4)\frac{N_{4,1}}{u_5 - u_4} + (u_6 - u)\frac{N_{5,1}}{u_6 - u_5} = (u-1)\frac{N_{4,1}}{0} + \frac{(1-u)N_{5,1}}{0} = 0$$

$$N_{5,2} = (u - u_5)\frac{N_{5,1}}{u_6 - u_5} + (u_7 - u)\frac{N_{6,1}}{u_7 - u_6} = \frac{(u-1)N_{5,1}}{0} + \frac{(1-u)N_{6,1}}{0} = 0$$

$$N_{0,3} = (u - u_0)\frac{N_{0,2}}{u_2 - u_0} + (u_3 - u)\frac{N_{1,2}}{u_3 - u_1} = u\frac{0}{0} + (-u)\frac{0}{0} = 0$$

$$N_{1,3} = (u - u_1)\frac{N_{1,2}}{u_3 - u_1} + (u_4 - u)\frac{N_{2,2}}{u_4 - u_2} = u\frac{N_{1,2}}{0} + \frac{(1-u)N_{2,2}}{1} = (1-u)^2 N_{3,1}$$

$$N_{2,3} = (u - u_2)\frac{N_{2,2}}{u_4 - u_2} + (u_5 - u)\frac{N_{3,2}}{u_5 - u_3} = uN_{2,2} + (1-u)N_{3,2} = 2u(1-u)N_{3,1}$$

$$N_{3,3} = (u - u_3)\frac{N_{3,2}}{u_5 - u_3} + (u_6 - u)\frac{N_{4,2}}{u_6 - u_4} = u^2 N_{3,1} + (1-u)\frac{N_{4,2}}{0} = u^2 N_{3,1}$$

$$N_{4,3} = (u - u_4)\frac{N_{4,2}}{u_6 - u_4} + (u_7 - u)\frac{N_{5,2}}{u_7 - u_5} = (u-1)\frac{N_{4,2}}{0} + (1-u)\frac{N_{5,2}}{0} = 0$$

$$N_{0,4} = (u - u_0)\frac{N_{0,3}}{u_3 - u_0} + (u_4 - u)\frac{N_{1,3}}{u_4 - u_1} = (1-u)^3 N_{3,1}$$

$$N_{1,4} = (u - u_1)\frac{N_{1,3}}{u_4 - u_1} + (u_5 - u)\frac{N_{2,3}}{u_5 - u_2} = 3u(1-u)^2 N_{3,1}$$

$$N_{2,4} = (u - u_2)\frac{N_{2,3}}{u_5 - u_2} + (u_6 - u)\frac{N_{3,3}}{u_6 - u_3} = 3u^2(1-u)N_{3,1}$$

$$N_{3,4} = (u - u_3)\frac{N_{3,3}}{u_6 - u_3} + (u_7 - u)\frac{N_{4,3}}{u_7 - u_4} = u^3 N_{3,1}$$

Substituting $N_{i,4}$ into Eq. (5.115) gives $\qquad 0 \le u \le 1$

$$\mathbf{P}(u) = [\mathbf{P}_0(1-u)^3 + 3\mathbf{P}_1 u(1-u)^2 + 3\mathbf{P}_2 u^2(1-u) + \mathbf{P}_3 u^3]N_{3,1}$$

Substituting $N_{3,1}$ into this equation gives the curve equation as

This equation is the same as the one for the Bezier curve in Example 5.19. Thus the cubic B-spline curve defined by four control points is identical to the cubic Bezier curve defined by the same points. This fact can be generalized for a $(k-1)$-degree curve as mentioned earlier.

B-Spline Curve

Substituting $N_{i,4}$ into Eq. (5.115) gives

$$\mathbf{P}(u) = [\mathbf{P}_0(1-u)^3 + 3\mathbf{P}_1 u(1-u)^2 + 3\mathbf{P}_2 u^2(1-u) + \mathbf{P}_3 u^3]N_{3,1}, \quad 0 \le u \le 1$$

Substituting $N_{3,1}$ into this equation gives the curve equation as

$$\mathbf{P}(u) = \mathbf{P}_0(1-u)^3 + 3\mathbf{P}_1 u(1-u)^2 + 3\mathbf{P}_2 u^2(1-u) + \mathbf{P}_3 u^3, \qquad 0 \le u \le 1$$

This equation is the same as the one for the Bezier curve in Example 5.19. Thus the cubic B-spline curve defined by four control points is identical to the cubic Bezier curve defined by the same points. This fact can be generalized for a $(k-1)$-degree curve as mentioned earlier.

There are two observations that are worth mentioning here. First, the sum of the two subscripts (i, k) of any B-spline function $N_{i,k}$ cannot exceed $(n + k)$. This gives a control on how far to go to calculate $N_{i,k}$. In this example six functions of $N_{i,1}$, five of $N_{i,2}$, and four of $N_{i,3}$ were needed such that $(6 + 1)$ for the first, $(5 + 2)$ for the second, and $(4 + 3)$ for the last are always equal to 7 $(n + k)$. Second, whenever the limits of u for any $N_{i,1}$ are equal, the u range becomes one point.

Example 5.22. Find the equation of a closed (periodic) B-spline curve defined by four control points.

Solution. This closed cubic spline has $k = 4$, $n = 3$. Using Eqs. (5.111) to (5.113), the knot vector $[u_0 \quad u_1 \quad u_2 \quad u_3 \quad u_4]$ is the integers $[0 \quad 1 \quad 2 \quad 3 \quad 4]$ and the range of u is $0 \leq u \leq 4$. Equation (5.103) gives the curve equation as

$$\mathbf{P}(u) = \mathbf{P}_0 N_{0,4} + \mathbf{P}_1 N_{1,4} + \mathbf{P}_2 N_{2,4} + \mathbf{P}_3 N_{3,4}, \qquad 0 \leq u \leq 4 \qquad (5.116)$$

To calculate the above B-spline functions, use Eq. (5.110) to obtain

$$N_{0,4}(u) = N_{0,4}((u+4) \bmod 4)$$

$$N_{1,4}(u) = N_{0,4}((u+3) \bmod 4)$$

$$N_{2,4}(u) = N_{0,4}((u+2) \bmod 4)$$

$$N_{3,4}(u) = N_{0,4}((u+1) \bmod 4)$$

In the above equations, $N_{0,4}$ on the right-hand side is the function for the open curve and on the left-hand side is the periodic function for the closed curve. Substituting these equations into Eq. (5.116) we get

$$\mathbf{P}(u) = \mathbf{P}_0 N_{0,4}((u+4) \bmod 4) + \mathbf{P}_1 N_{0,4}((u+3) \bmod 4)$$
$$+ \mathbf{P}_2 N_{0,4}((u+2) \bmod 4) + \mathbf{P}_3 N_{0,4}((u+1) \bmod 4), \qquad 0 \leq u \leq 4 \quad (5.117)$$

In Eq. (5.117), the function $N_{0,4}$ has various arguments, which can be found if specific values of u are used. To find $N_{0,4}$, similar calculations to the previous example 5.21 are performed using the above knot vector as follows:

$$N_{0,1} = \begin{cases} 1, & 0 \leq u \leq 1 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{1,1} = \begin{cases} 1, & 1 \leq u \leq 2 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{2,1} = \begin{cases} 1, & 2 \leq u \leq 3 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{3,1} = \begin{cases} 1, & 3 \leq u \leq 4 \\ 0, & \text{elsewhere} \end{cases}$$

$$N_{0,2} = (u - u_0)\frac{N_{0,1}}{u_1 - u_0} + (u_2 - u)\frac{N_{1,1}}{u_2 - u_1} = uN_{0,1} + (2 - u)N_{1,1}$$

$$N_{1,2} = (u - u_1)\frac{N_{1,1}}{u_2 - u_1} + (u_3 - u)\frac{N_{2,1}}{u_3 - u_2} = (u - 1)N_{1,1} + (3 - u)N_{2,1}$$

$$N_{2,2} = (u - u_2)\frac{N_{2,1}}{u_3 - u_2} + (u_4 - u)\frac{N_{3,1}}{u_4 - u_3} = (u - 2)N_{2,1} + (4 - u)N_{3,1}$$

$$N_{0,3} = (u - u_0)\frac{N_{0,2}}{u_2 - u_0} + (u_3 - u)\frac{N_{1,2}}{u_3 - u_1} = \frac{1}{2}uN_{0,2} + \frac{1}{2}(3 - u)N_{1,2}$$

$$= \tfrac{1}{2}u^2N_{0,1} + \tfrac{1}{2}[u(2 - u) + (3 - u)(u - 1)]N_{1,1} + \tfrac{1}{2}(3 - u)^2N_{2,1}$$

$$N_{1,3} = (u - u_1)\frac{N_{1,2}}{u_3 - u_1} + (u_4 - u)\frac{N_{2,2}}{u_4 - u_2} = \frac{1}{2}(u - 1)N_{1,2} + \frac{1}{2}(4 - u)N_{2,2}$$

$$= \tfrac{1}{2}(u - 1)^2N_{1,1} + \tfrac{1}{2}[(u - 1)(3 - u) + (u - 2)(4 - u)]N_{2,1} + \tfrac{1}{2}(4 - u)^2N_{3,1}$$

$$N_{0,4} = (u - u_0)\frac{N_{0,3}}{u_3 - u_0} + (u_4 - u)\frac{N_{1,3}}{u_4 - u_1} = \frac{1}{3}uN_{0,3} + \frac{1}{3}(4 - u)N_{1,3}$$

$$= \tfrac{1}{6}\{u^3N_{0,1} + [u^2(2 - u) + u(3 - u)(u - 1) + (4 - u)(u - 1)^2]N_{1,1}$$
$$+ [u(3 - u)^2 + (4 - u)(u - 1)(3 - u) + (4 - u)^2(u - 2)]N_{2,1} + (4 - u)^3N_{3,1}\}$$

or

$$N_{0,4} = \tfrac{1}{6}[u^3N_{0,1} + (-3u^3 + 12u^2 - 12u + 4)N_{1,1} + (3u^3 - 24u^2 + 60u - 44)N_{2,1}$$
$$+ (-u^3 + 12u^2 - 48u + 64)N_{3,1}]$$

Due to the non-zero values of the functions $N_{i,1}$ for various intervals of u, the above equation can be written as

$$N_{0,4}(u) = \begin{cases} \tfrac{1}{6}u^3, & 0 \le u \le 1 \\ \tfrac{1}{6}(-3u^3 + 12u^2 - 12u + 4), & 1 \le u \le 2 \\ \tfrac{1}{6}(3u^3 - 24u^2 + 60u - 44), & 2 \le u \le 3 \\ \tfrac{1}{6}(-u^3 + 12u^2 - 48u + 64), & 3 \le u \le 4 \end{cases} \qquad (5.118)$$

To check the correctness of the above expression of $N_{0,4}(u)$, one would expect to obtain Fig. 5-54c if this function is plotted. Indeed, this figure is the plot of $N_{0,4}$. If $u = 0, 1, 2, 3$, and 4 are substituted into this function, the corresponding values of $N_{0,4}$ that are shown in the figure are obtained.

Equations (5.117) and (5.118) together can be used to evaluate points on the closed B-spline curve for display or plotting purposes. As an illustration, consider the following points:

$$\begin{aligned}
\mathbf{P}(0) &= \mathbf{P}_0 N_{0,4}(4 \bmod 4) + \mathbf{P}_1 N_{0,4}(3 \bmod 4) \\
&\quad + \mathbf{P}_2 N_{0,4}(2 \bmod 4) + \mathbf{P}_3 N_{0,4}(1 \bmod 4) \\
&= \mathbf{P}_0 N_{0,4}(0) + \mathbf{P}_1 N_{0,4}(3) + \mathbf{P}_2 N_{0,4}(2) + \mathbf{P}_3 N_{0,4}(1) \\
&= \tfrac{1}{6}\mathbf{P}_1 + \tfrac{2}{3}\mathbf{P}_2 + \tfrac{1}{6}\mathbf{P}_3
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathbf{P}(0.5) &= \mathbf{P}_0 N_{0,4}(0.5) + \mathbf{P}_1 N_{0,4}(3.5) + \mathbf{P}_2 N_{0,4}(2.5) + \mathbf{P}_3 N_{0,4}(1.5) \\
&= \frac{1}{48}\mathbf{P}_0 + \frac{1}{48}\mathbf{P}_1 + \frac{23}{48}\mathbf{P}_2 + \frac{23}{48}\mathbf{P}_3
\end{aligned}$$

$$\begin{aligned}
\mathbf{P}(1) &= \mathbf{P}_0 N_{0,4}(1) + \mathbf{P}_1 N_{0,4}(0) + \mathbf{P}_2 N_{0,4}(3) + \mathbf{P}_3 N_{0,4}(2) \\
&= \tfrac{1}{6}\mathbf{P}_0 + \tfrac{1}{6}\mathbf{P}_2 + \tfrac{2}{3}\mathbf{P}_3
\end{aligned}$$

$$\mathbf{P}(2) = \mathbf{P}_0 N_{0,4}(2) + \mathbf{P}_1 N_{0,4}(1) + \mathbf{P}_2 N_{0,4}(0) + \mathbf{P}_3 N_{0,4}(3) = \tfrac{2}{3}\mathbf{P}_0 + \tfrac{1}{6}\mathbf{P}_1 + \tfrac{1}{6}\mathbf{P}_3$$

$$\mathbf{P}(3) = \mathbf{P}_0 N_{0,4}(3) + \mathbf{P}_1 N_{0,4}(2) + \mathbf{P}_2 N_{0,4}(1) + \mathbf{P}_3 N_{0,4}(0) = \tfrac{1}{6}\mathbf{P}_0 + \tfrac{2}{3}\mathbf{P}_1 + \tfrac{1}{6}\mathbf{P}_2$$

$$\mathbf{P}(4) = \mathbf{P}_0 N_{0,4}(0) + \mathbf{P}_1 N_{0,4}(3) + \mathbf{P}_2 N_{0,4}(2) + \mathbf{P}_3 N_{0,4}(1) = \tfrac{1}{6}\mathbf{P}_1 + \tfrac{2}{3}\mathbf{P}_2 + \tfrac{1}{6}\mathbf{P}_3$$

In the above calculations, notice the cyclic rotation of the $N_{0,4}$ coefficients of the control points for the various values of u excluding $u = 0.5$. Notice also the effect of the canonical (symmetric) form of $N_{0,4}$ on the coefficients of the control points. If the u values are 0.5, 1.5, 2.5, and 3.5, or other values separated by unity, a similar cyclic rotation of the coefficients is expected. Finally, notice that $P(0)$ and $P(4)$ are equal, which ensures obtaining a closed B-spline curve.

β-Spline Curve

The theory of the B-spline has been extended further to allow more control of the curve shape and continuity. For example, β-spline (beta-spline) and ν-spline (nu-spline) curves provide manipulation of the curve shape and maintain its geometric continuity rather than its parametric continuity as provided by B-spline curves. The β-spline (sometimes called the spline in tension) curve is a generalization of the uniform cubic B-spline curve. The β-spline curve provides the designer with two additional parameters: the bias and the tension to control the shape of the curve. Therefore, the control points and the degree of the β-spline curve can remain fixed and yet the curve shape can be manipulated.

Although the β-spline curve is capable of applying tension at each control point, its formulation as piecewise hyperbolic sines and cosines makes its computation expensive. The ν-spline curve is therefore developed as a piecewise polynomial alternative to the spline in tension.

Rational Curves

5.6.4 Rational Curves

A rational curve is defined by the algebraic ratio of two polynomials while a nonrational curve [Eq. (5.103) gives an example] is defined by one polynomial. Rational curves draw their theories from projective geometry. They are important because of their invariance under projective transformation; that is, the perspective image of a rational curve is a rational curve. Rational Bezier curves, rational B-spline and β-spline curves, rational conic sections, rational cubics, and rational surfaces have been formulated. The most widely used rational curves are NURBS (nonuniform rational B-splines). A brief description of rational B-spline curves is given below.

The formulation of rational curves requires the introduction of homogeneous space and the homogeneous coordinates. This subject is covered in detail in Chap. 9 (Sec. 9.2.5). The homogeneous space is four-dimensional space. A point in E^3 with coordinates (x, y, z) is represented in the homogeneous space by the coordinates (x^*, y^*, z^*, h), where h is a scalar factor. The relationship between the two types of coordinates is given by Eq. (9.59).

A rational B-spline curve defined by $n + 1$ control points P_i is given by

$$\mathbf{P}(u) = \sum_{i=0}^{n} \mathbf{P}_i R_{i,k}(u), \qquad 0 \leq u \leq u_{\max} \tag{5.119}$$

$R_{i,k}(u)$ are the rational B-spline basis functions and are given by

$$R_{i,k}(u) = \frac{h_i N_{i,k}(u)}{\sum_{i=0}^{n} h_i N_{i,k}(u)} \tag{5.120}$$

The above equation shows that $R_{i,k}(u)$ are a generalization of the nonrational basis functions $N_{i,k}(u)$. If we substitute $h_i = 1$ in the equation, $R_{i,k}(u) = N_{i,k}(u)$. The rational basis functions $R_{i,k}(u)$ have nearly all the analytic and geometric characteristics of their nonrational B-spline counterparts. All the discussions covered in Sec. 5.6.3 apply here.

Conversion between curve representation

Hermite	Bezier	B-Spline

$$M_H := \begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad M_B := \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad M_S := \frac{1}{6}\cdot\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}$$

$$p(u) := U\cdot M_H\cdot V \qquad p(u) := U\cdot M_B\cdot P_B \qquad P_i(t) := U\cdot M_S\cdot P_S$$

$$HtoB := \frac{1}{3}\cdot\begin{pmatrix} -3 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 \\ 0 & 3 & 0 & -1 \\ 0 & 3 & 0 & 0 \end{pmatrix} \qquad BtoH := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{pmatrix} \qquad BStoH := \begin{pmatrix} 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ -3 & 0 & 3 & 0 \\ 0 & -3 & 0 & 3 \end{pmatrix}$$

$$HtoBS := \frac{1}{3}\cdot\begin{pmatrix} -3 & 6 & -7 & -2 \\ 6 & -3 & 2 & 1 \\ -3 & 6 & -1 & -2 \\ 6 & -3 & 2 & 7 \end{pmatrix} \qquad BtoBS := \begin{pmatrix} 6 & -7 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -7 & 6 \end{pmatrix} \qquad BStoB := \begin{pmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 0 \\ 0 & 1 & 4 & 1 \end{pmatrix}$$

Cubic Bezier representation

Conversion to B-Spline representation

$$P := \begin{pmatrix} -6 & 0 & 0 & 1 \\ -3 & 4 & 0 & 1 \\ 3 & -4 & 0 & 1 \\ 6 & 0 & 0 & 1 \end{pmatrix} \qquad V_S := BtoBS\cdot P \qquad V_S = \begin{pmatrix} -9 & -36 & 0 & 1 \\ -9 & 12 & 0 & 1 \\ 9 & -12 & 0 & 1 \\ 9 & 36 & 0 & 1 \end{pmatrix}$$

Conversion between curve representation

Hermite <— Bezier

$$V = M_H^{-1} \cdot M_B \cdot P_B \qquad V := \begin{pmatrix} P_0 \\ P_1 \\ P'_0 \\ P'_1 \end{pmatrix} \qquad V := \begin{bmatrix} P_0 \\ P_1 \\ 3 \cdot (P_1 - P_0) \\ 3 \cdot (P_3 - P_2) \end{bmatrix}$$

$$BtoH := M_H^{-1} \cdot M_B$$

BtoH

$$M_H^{-1} \cdot M_B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{pmatrix} \qquad P := \begin{pmatrix} P_3 \\ P_2 \\ P_1 \\ P_0 \end{pmatrix}$$

Bezier <— Hermite

$$P = M_B^{-1} \cdot M_H \cdot V \qquad P := \begin{pmatrix} P_3 \\ P_2 \\ P_1 \\ P_0 \end{pmatrix} \qquad P = \begin{pmatrix} P_0 \\ P_0 + \frac{1}{3} \cdot P'_0 \\ P_1 - \frac{1}{3} \cdot P'_1 \\ P_1 \end{pmatrix}$$

$$HtoB = M_B^{-1} \cdot M_H$$

HtoB

$$M_B^{-1} \cdot M_H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & \frac{1}{3} & 0 \\ 0 & 1 & 0 & -\frac{1}{3} \\ 0 & 1 & 0 & 0 \end{pmatrix} \qquad V := \begin{pmatrix} P_0 \\ P_1 \\ P'_0 \\ P'_1 \end{pmatrix}$$

$$U \cdot M_B \cdot P_B = U \cdot M_S \cdot P_S$$

$$M_B \cdot P_B = M_S \cdot P_S$$

$$P_B := M_B^{-1} \cdot M_S \cdot P_S$$

BStoB

$$P_B = \frac{1}{6} \cdot \begin{pmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 0 \\ 0 & 1 & 4 & 1 \end{pmatrix} \cdot P_S$$

$$P_S := M_S^{-1} \cdot M_B \cdot P_B$$

$$p(u) := U \cdot B \cdot P$$

BtoBS

$$P_S = \begin{pmatrix} 6 & -7 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -7 & 6 \end{pmatrix} \cdot P_B$$

BStoH

$$V = M_H^{-1} \cdot M_S \cdot P_S \qquad V = \frac{1}{6} \cdot \begin{pmatrix} 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ -3 & 0 & 3 & 0 \\ 0 & -3 & 0 & 3 \end{pmatrix} \cdot P_S$$

HtoBS

$$P_S := M_S^{-1} \cdot M_H \cdot V \qquad P_S = \frac{1}{3} \cdot \begin{pmatrix} -3 & 6 & -7 & -2 \\ 6 & -3 & 2 & 1 \\ -3 & 6 & -1 & -2 \\ 6 & -3 & 2 & 7 \end{pmatrix} \cdot V$$

Comparison of curve/surface rep. methods

Property	Ferguson	Hermite-Coons	Bézier	Lagrange	Composite Bézier	Cardinal Spline	B-Splines	NURBS
Easy geometric representation	low	med	high	med	high	Medium	high	high
Convex hull	no	no	yes	no	yes	no	yes	yes
Variation diminishing *	no	no	yes	no	yes	no	yes	yes
Easiness for creation	low	med	med	inappr.	med	high	high	high
Local control	no	no	no	no	yes but complex	no	yes	yes
Approximation ease	med	med	high	low	high	medium	high	high
Interpolation ease	med	med	med	high but inappr.	med	high	high	high
Generality	med	med	med	med	med	med	med	high
Popularity **	low	low	med	low	med	med	high	very high

Ferguson : cubic (or higher order) polynomials

Lagrange : interpolation polynomial curve

B-Spline Curve

B-Spline Curve

B-Spline Curve

Motivation Bezier representation is good, but has these drawbacks:

1. High degree

The degree is determined by the number of control points which tends to be large for complex curves. This causes oscillation and other computational problems.

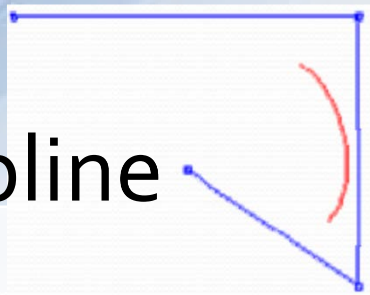
2. Global propagation of local change

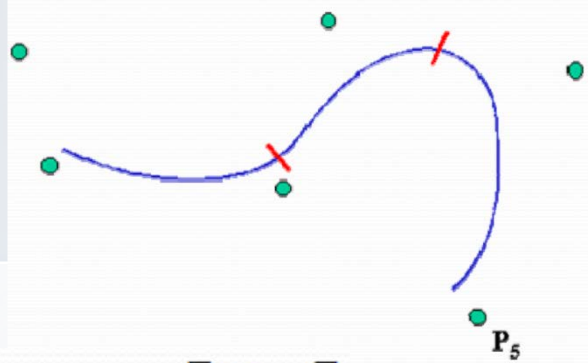
When modifying a control point, the designer wants to see the shape change locally around the moved control point. In Bezier case, the change however tends to be strongly propagated throughout the entire curve.

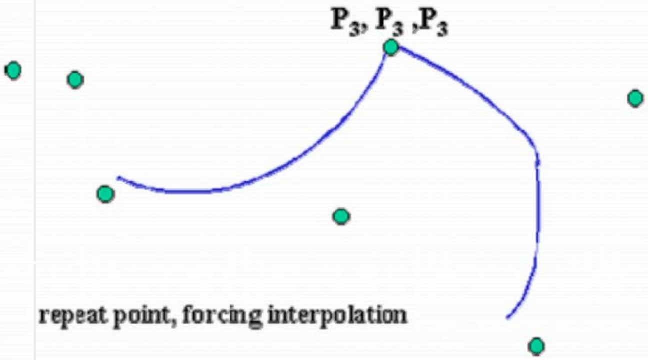
3. Intractable linear equations

If we are interested in interpolation rather than just approximating a shape, we need to compute the control points from the 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.

B-Spline



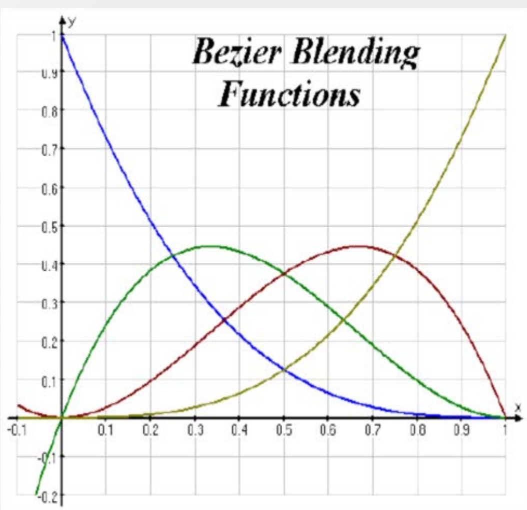


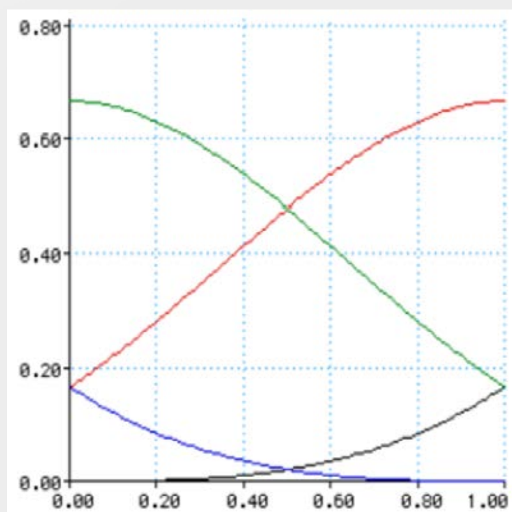


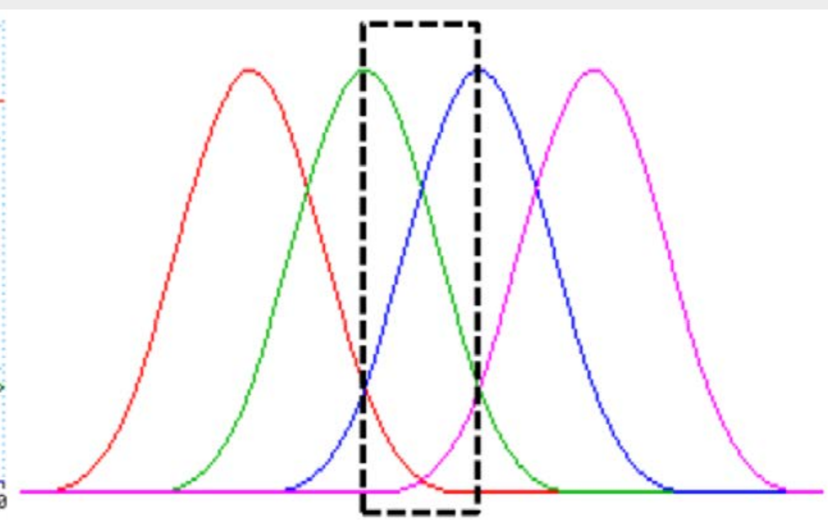
$$f(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} M_{Bspline} \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix}$$

$$M_{Bspline} = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 3 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

B-Spline Blending

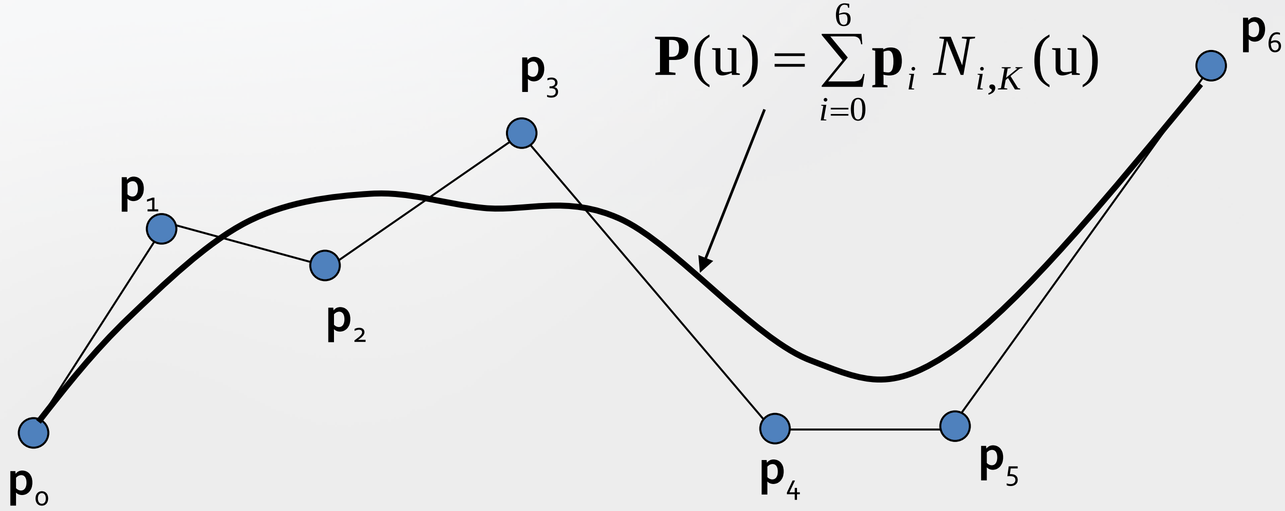






B-Spline Curve

A pictorial illustration



Unlike Bezier basis function $B_{i,n}(u)$, the degree of the B-Spline basis function $N_{i,K}(u)$ is NOT dependent on the number of control points n; instead, it is controlled by another integer K independent of n.

B-Spline Curve

Definition of B-Spline basis function $N_{i,K}(u)$

Step 1. Choose an appropriate K (usually 4); K-1 will be the degree of $N_{i,K}(u)$.

Step 2. Choose n+K+1 real numbers called knots
$\{t_0, t_1, \ldots, t_{n+K}\}$ $(t_j \leq t_{j+1})$.

Step 3.

$$N_{i,1}(u) = 1 \quad \text{if } t_i \leq u \leq t_{i+1}$$
$$= 0 \quad \text{otherwise}$$

and

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

B-Spline Curve

Knots vectors

- Uniform (U)

$t_0 = 0,\ t_1 = 1,\ . . . ,\ t_{n+K} = n+K.$

- Nonuniform and clamped (NU&C)

$$t_j = 0 \qquad \text{if } j < K$$
$$t_j = j - K + 1 \qquad \text{if } K \le j \le n$$
$$t_j = n - K + 2 \qquad \text{if } j > n$$

for $j = 0, \ldots, n + K.$

- Nonuniform (NU)

Any, as long as $t_j \le t_{j+1}$ for all $j=0,1,\ldots, n+K.$

Non-periodic and uniform B-Spline

- Step 1. Select an integer k, called the ***order*** of the B-spline curve,
- 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: $\mathbf{P}(u) = \sum_{i=0}^{n} N_{i,k}(u)\,\mathbf{P}_i \qquad (t_{k-1} \le u \le t_{n+1})$

B-Spline Curve

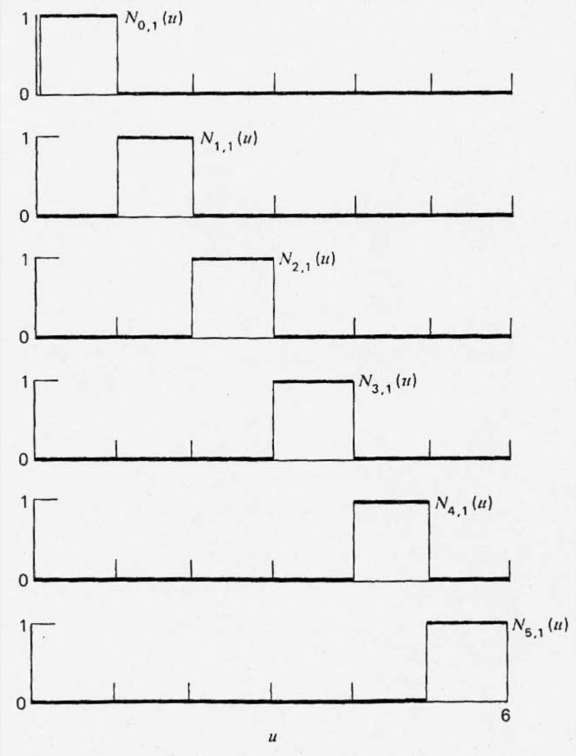
NU&C example 1 ($n = 5$, $K = 1$)

1. Since $n+K = 6$, there are seven knots and their values are:

$t_0 = 0 \quad t_1 = 1 \quad t_2 = 2 \quad t_3 = 3 \quad t_4 = 4 \quad t_5 = 5 \quad t_6 = 6$

2. By recursive definition of $N_{i,1}(u)$

$N_{0,1}(u)$	$= 1$	if $0 \le u < 1$
	$= 0$	otherwise
$N_{1,1}(u)$	$= 1$	if $1 \le u < 2$
	$= 0$	otherwise
$N_{2,1}(u)$	$= 1$	if $2 \le u < 3$
	$= 0$	otherwise
$N_{3,1}(u)$	$= 1$	if $3 \le u < 4$
	$= 0$	otherwise
$N_{4,1}(u)$	$= 1$	if $4 \le u < 5$
	$= 0$	otherwise
$N_{5,1}(u)$	$= 1$	if $5 \le u < 6$
	$= 0$	otherwise

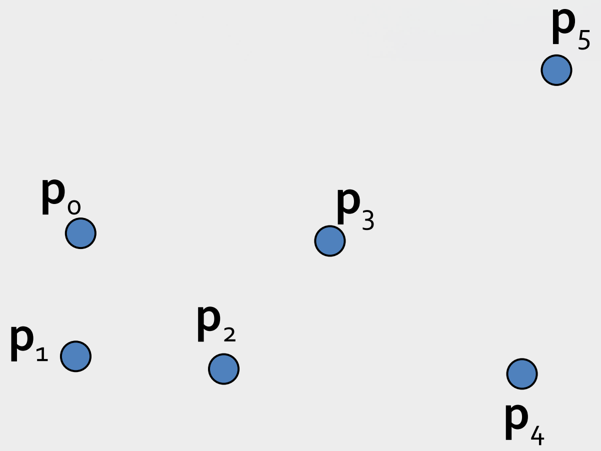


B-Spline Curve

NU&C example 1 (n = 5, K =1) (cont'd)

3. What is the curve ? $$\mathbf{P}(\mathrm{u}) = \sum_{i=0}^{5} \mathbf{p}_i \, N_{i,1}(\mathrm{u})$$

$\mathbf{P}(\mathrm{u}) = \mathbf{p}_0$ for $0 \leq \mathrm{u} < 1$
$\mathbf{P}(\mathrm{u}) = \mathbf{p}_1$ for $1 \leq \mathrm{u} < 2$
$\mathbf{P}(\mathrm{u}) = \mathbf{p}_2$ for $2 \leq \mathrm{u} < 3$
$\mathbf{P}(\mathrm{u}) = \mathbf{p}_3$ for $3 \leq \mathrm{u} < 4$
$\mathbf{P}(\mathrm{u}) = \mathbf{p}_4$ for $4 \leq \mathrm{u} < 5$
$\mathbf{P}(\mathrm{u}) = \mathbf{p}_5$ for $5 \leq \mathrm{u} < 6$



B-Spline Curve

NU&C example 2 ($n = 5$, $K = 2$)

1. Since $n+K = 7$, there are eight knots and their values are:

$$t_0 = 0 \quad t_1 = 0 \quad t_2 = 1 \quad t_3 = 2 \quad t_4 = 3 \quad t_5 = 4 \quad t_6 = 5 \quad t_7 = 5$$

2. By definition of $N_{i,1}(u)$

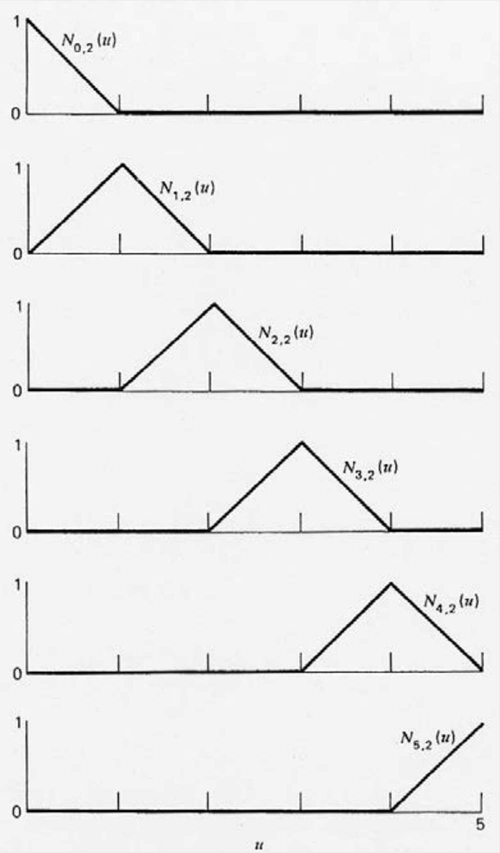
$N_{0,1}(u)$	$= 1$	if $u = 0$
	$= 0$	otherwise
$N_{1,1}(u)$	$= 1$	if $0 \le u < 1$
	$= 0$	otherwise
$N_{2,1}(u)$	$= 1$	if $1 \le u < 2$
	$= 0$	otherwise
$N_{3,1}(u)$	$= 1$	if $2 \le u < 3$
	$= 0$	otherwise
$N_{4,1}(u)$	$= 1$	if $3 \le u < 4$
	$= 0$	otherwise
$N_{5,1}(u)$	$= 1$	if $4 \le u < 5$
	$= 0$	otherwise

B-Spline Curve

NU&C example 2 (n = 5, K =2) (cont'd)

3. By definition of $N_{i,2}(u)$

$$
\begin{aligned}
N_{0,2}(u) &= (1 - u)N_{0,1}(u) \\
N_{1,2}(u) &= uN_{1,1}(u) + (2 - u)N_{2,1}(u) \\
N_{2,2}(u) &= (u - 1)N_{2,1}(u) + (3 - u)N_{3,1}(u) \\
N_{3,2}(u) &= (u - 2)N_{3,1}(u) + (4 - u)N_{4,1}(u) \\
N_{4,2}(u) &= (u - 3)N_{4,1}(u) + (5 - u)N_{5,1}(u) \\
N_{5,2}(u) &= (u - 4)N_{5,1}(u)
\end{aligned}
$$

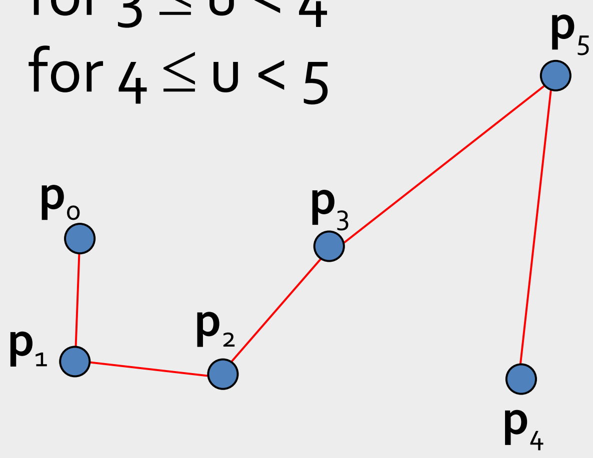


B-Spline Curve

NU&C example 2 (n = 5, K =2) (cont'd)

4. What is the curve ? $\mathbf{P}(\mathrm{u}) = \sum_{i=0}^{5} \mathbf{p}_i \, N_{i,2}(\mathrm{u})$

$\mathbf{P}(u) = \mathbf{P}_1(u) = (1-u)\mathbf{p}_0 + u\mathbf{p}_1$ for $0 \le u < 1$
$\mathbf{P}(u) = \mathbf{P}_2(u) = (2-u)\mathbf{p}_1 + (u-1)\mathbf{p}_2$ for $1 \le u < 2$
$\mathbf{P}(u) = \mathbf{P}_3(u) = (3-u)\mathbf{p}_2 + (u-2)\mathbf{p}_3$ for $2 \le u < 3$
$\mathbf{P}(u) = \mathbf{P}_4(u) = (4-u)\mathbf{p}_3 + (u-3)\mathbf{p}_4$ for $3 \le u < 4$
$\mathbf{P}(u) = \mathbf{P}_5(u) = (5-u)\mathbf{p}_4 + (u-4)\mathbf{p}_5$ for $4 \le u < 5$



B-Spline Curve

NU&C example 3 ($n = 5$, $K = 3$)

1. Since $n+K = 8$, there are nine knots and their values are:

   $t_0 = 0$ $t_1 = 0$ $t_2 = 0$ $t_3 = 1$ $t_4 = 2$ $t_5 = 3$ $t_6 = 4$ $t_7 = 4$ $t_8 = 4$

2. By definition of $N_{i,1}(u)$

$N_{0,1}(u)$	$= 1$	if $u = 0$
	$= 0$	otherwise
$N_{1,1}(u)$	$= 1$	if $u = 0$
	$= 0$	otherwise
$N_{2,1}(u)$	$= 1$	if $0 \leq u < 1$
	$= 0$	otherwise
$N_{3,1}(u)$	$= 1$	if $1 \leq u < 2$
	$= 0$	otherwise
$N_{4,1}(u)$	$= 1$	if $2 \leq u < 3$
	$= 0$	otherwise
$N_{5,1}(u)$	$= 1$	if $3 \leq u < 4$
	$= 0$	otherwise

B-Spline Curve

NU&C example 3 (n = 5, K = 3) (cont'd)

3. By definition of $N_{i,2}(u)$

$$
\begin{aligned}
N_{0,2}(u) &= 0 \\
N_{1,2}(u) &= (1-u)N_{2,1}(u) \\
N_{2,2}(u) &= uN_{2,1}(u) + (2-u)N_{3,1}(u) \\
N_{3,2}(u) &= (u-1)N_{3,1}(u) + (3-u)N_{4,1}(u) \\
N_{4,2}(u) &= (u-2)N_{4,1}(u) + (4-u)N_{5,1}(u) \\
N_{5,2}(u) &= (u-3)N_{5,1}(u)
\end{aligned}
$$

The u-range for $N_{i,2}(u)$ is determined by that of $N_{i,1}(u)$ and $N_{i+1,1}(u)$. For example, $N_{3,2}(u)$ is non-zero only in $1 \le u < 3$, since $N_{3,1}(u)$ non-zero only in $1 \le u < 2$ and $N_{4,1}(u)$ is defined only in $2 \le u < 3$.

B-Spline Curve

NU&C example 3 (n = 5, K = 3) (cont'd)

4. By definition of $N_{i,3}(u)$

$$
\begin{aligned}
N_{0,3}(u) &= (1-u)^2 N_{2,1}(u) \\
N_{1,3}(u) &= 0.5u(4-3u)N_{2,1}(u) + 0.5(2-u)^2 N_{3,1}(u) \\
N_{2,3}(u) &= 0.5u^2 N_{2,1}(u) + 0.5(-2u^2 + 6u - 3)N_{3,1}(u) + 0.5(3-u)^2 N_{4,1}(u) \\
N_{3,3}(u) &= 0.5(u-1)^2 N_{3,1}(u) + 0.5(-2u^2 + 10u - 11)N_{4,1}(u) + 0.5(4-u)^2 N_{5,1}(u) \\
N_{4,3}(u) &= 0.5(u-2)^2 N_{4,1}(u) + 0.5(-3u^2 + 20u - 32)N_{5,1}(u) \\
N_{5,3}(u) &= (u-3)^3 N_{5,1}(u)
\end{aligned}
$$

The u-range for $N_{i,3}(u)$ is determined by that of $N_{i,1}(u)$, $N_{i+1,1}(u)$, and $N_{i+2,1}(u)$.

For example, $N_{2,3}(u)$ is non-zero only in $0 \leq u < 3$, since outside $0 \leq u < 3$ all $N_{2,1}(u)$, $N_{3,1}(u)$, and $N_{4,1}(u)$ are zero.

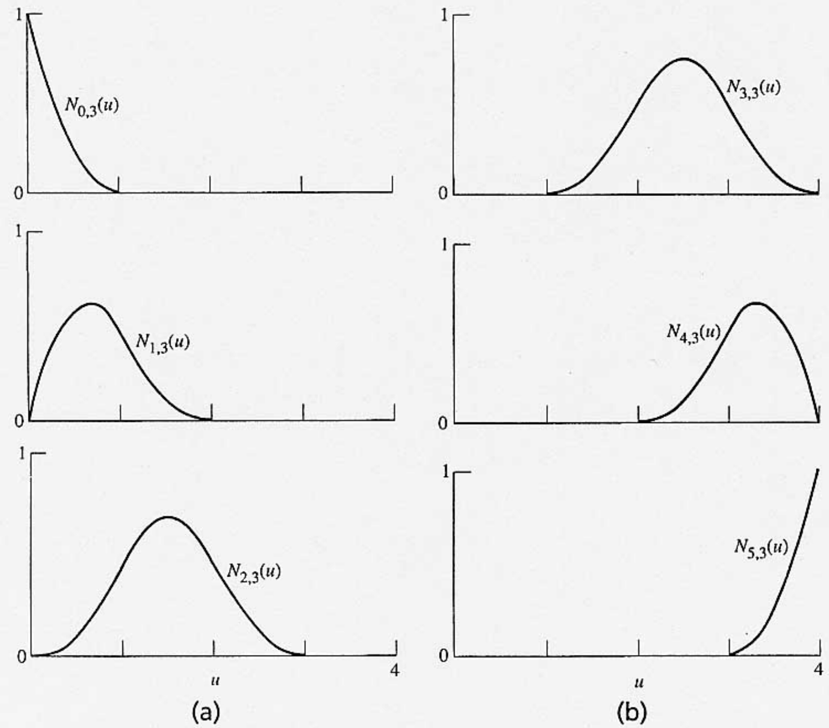
Pictures of $N_{i,3}(u)$.

B-Spline Curve

NU&C example 3 (n = 5, K = 3) (cont'd)

5. Pictures of $N_{i,3}(u)$

Example: On interval $1 \leq u < 2$, only $N_{1,3}(u)$, $N_{2,3}(u)$, and $N_{3,3}(u)$ are non-zero; therefore, only the three control points $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$ will affect on this interval.



Nonuniform B-Spline basis functions for $n = 5$, $K = 3$.

B-Spline Curve

NU&C example 3 (n = 5, K = 3) (cont'd)

6. What is the curve ?

$$\mathbf{P}(u) = \sum_{i=0}^{5} \mathbf{p}_i \, N_{i,3}(u)$$

$\mathbf{P}(u) = \mathbf{P}_1(u) = (1-u)^2\mathbf{p}_0 + 0.5u(4-3u)\mathbf{p}_1 + 0.5u^2\mathbf{p}_2$ for $0 \le u < 1$

$\mathbf{P}(u) = \mathbf{P}_2(u) = 0.5(2-u)^2\mathbf{p}_1 + 0.5(-2u^2 + 6u - 3)\mathbf{p}_2 + 0.5(u-1)^2\mathbf{p}_3$ for $1 \le u < 2$

$\mathbf{P}(u) = \mathbf{P}_3(u) = 0.5(3-u)^2\mathbf{p}_2 + 0.5(-2u^2 + 10u - 11)\mathbf{p}_3 + 0.5(u-2)^2\mathbf{p}_4$ for $2 \le u < 3$

$\mathbf{P}(u) = \mathbf{P}_4(u) = 0.5(4-u)^2\mathbf{p}_3 + 0.5(-3u^2 + 20u - 32)\mathbf{p}_4 + (u-3)^2\mathbf{p}_5$ for $3 \le u < 4$

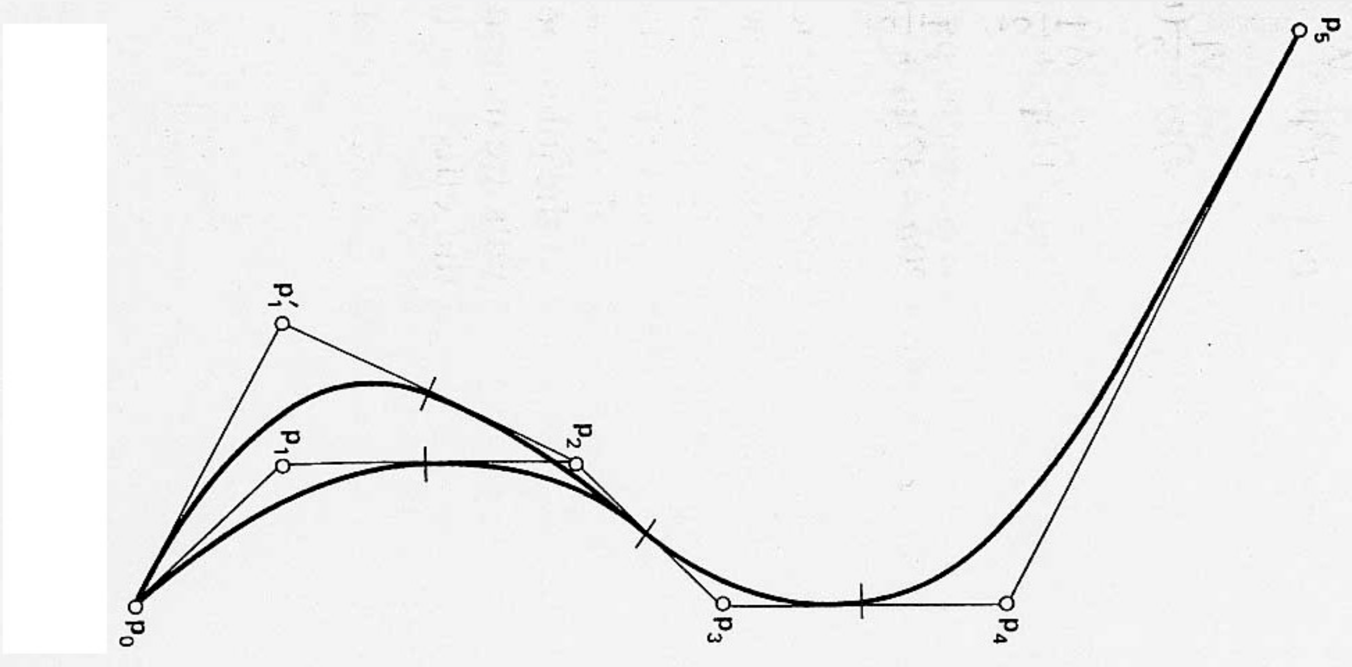
This is a C^1 curve with:

$\mathbf{P}(1) = \mathbf{P}_1(1) = \mathbf{P}_2(1)$;	$\mathbf{P}'(1) = \mathbf{P}'_1(1) = \mathbf{P}'_2(1)$
$\mathbf{P}(2) = \mathbf{P}_2(2) = \mathbf{P}_3(2)$;	$\mathbf{P}'(2) = \mathbf{P}'_2(2) = \mathbf{P}'_3(2)$
$\mathbf{P}(3) = \mathbf{P}_3(3) = \mathbf{P}_4(3)$;	$\mathbf{P}'(3) = \mathbf{P}'_3(3) = \mathbf{P}'_4(3)$

B-Spline Curve

NU&C example 3 (n = 5, K = 3) (cont'd)

7. Picture of $\mathbf{P}(u) = \sum_{i=0}^{5} \mathbf{p}_i \, N_{i,3}(u)$



B-Spline Curve

Properties of NU&C (n, K)

1. It is a C^{K-2} curve in $[0 \le u \le n - K+2]$.

2. It consists of n-K+2 independent C^{K-1} curves (segments) defined respectively on [0,1), [1,2), …, [n-K+1, n-K+2].

3. The curve passes the first and last control points $\mathbf{p}_0$ and $\mathbf{p}_n$.

4. The tangents at the two ends, $\mathbf{p}_0$ and $\mathbf{p}_n$, are parallel to $\mathbf{p}_1 - \mathbf{p}_0$ and $\mathbf{p}_n - \mathbf{p}_{n-1}$ respectively.

5. Each segment is influenced by only K control points, and, conversely, each control point influences only K curve segments.
   http://theory.lcs.mit.edu/~boyko/classes/b-spline.html
   http://www.cs.berkeley.edu/~j-yen/splines/bsplinecurve/

B-Spline Curve

Properties of NU&C (n, K) (cont'd)

6. Except for the very few segments near the end points (depending on K), the segment $\mathbf{P}_i(u)$ is only dependent on K, not n.

   K = 2: $\mathbf{P}_{i+1}(u) = (i + 1 - u)\mathbf{p}_i + (u - i)\mathbf{p}_{i+1}$ $\quad (i \le u < i+1)$

   K = 3:
   $\mathbf{P}_{i+1}(u) = 0.5(i+1-u)^2\mathbf{p}_i + 0.5[(u-i+1)(i+1-u) + (i+2-u)(u-i)]\mathbf{p}_{i+1} + 0.5(u-i)^2\mathbf{p}_{i+2}$

   $(i \le u < i+1)$

   K = 4: ???

7. Each segment $\mathbf{P}_i(u)$ is a C^{K-1} continuous curve.

8. The B-Spline curve itself is C^{K-2} continuous at the knots t_i, i = 1, …, n–K+1, i.e., the end points of the segments.

B-Spline Curve

Normalized B-Spline segment (NBS)

Segment

$\mathbf{P}_{i+1}(u) = 0.5(i+1-u)^2\mathbf{p}_i + 0.5[(u-i+1)(i+1-u) + (i+2-u)(u-i)]\mathbf{p}_{i+1} + 0.5(u-i)^2\mathbf{p}_{i+2}$

is defined on the interval ($i \leq u < i+1$). Re-parameterize the domain by replacing u with $u + i$, the same segment now is defined on the interval ($0 \leq u < 1$) as:

$\mathbf{P}_{i+1}(u) = 0.5[(1-u)^2\mathbf{p}_i + (-2u^2 + 2u + 1)\mathbf{p}_{i+1} + u^2\mathbf{p}_{i+2}]$

B-Spline Curve

Matrix form of NBS

K = 3:

$\mathbf{P}_i(u) = 0.5(1-u)^2\mathbf{p}_{i-1} + 0.5(-2u^2 + 2u + 1)\mathbf{p}_i + 0.5u^2\mathbf{p}_{i+1}]$

$$\mathbf{P}_i(u) = \underbrace{\begin{bmatrix} u^2 & u & 1 \end{bmatrix}}_{\mathbf{U}} \underbrace{\frac{1}{2}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 1 & 0 \end{bmatrix}}_{\mathbf{M}_s} \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \end{bmatrix} = \mathbf{U}\mathbf{M}_s \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \end{bmatrix}$$

B-Spline Curve

Matrix form of NBS

K = 4:

$$\mathbf{P}_i(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & 6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \end{bmatrix} = \mathbf{U}\mathbf{M}_s \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \end{bmatrix}$$

$\mathbf{U}$ $\mathbf{M}_s$

B-Spline Curve

Uniform B-Spline curve (n=5, K=3)

- For a NU&C B-Spline (n = 5, K = 3)

$\mathbf{P}_1(u) = (1 - u)^2\mathbf{p}_0 + 0.5u(4\text{-}3u)\mathbf{p}_1 + 0.5u^2\mathbf{p}_2$

$\mathbf{P}_2(u) = UM_s[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]^T$

$\mathbf{P}_3(u) = UM_s[\mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4]^T$

$\mathbf{P}_4(u) = 0.5(4 - u)^2\mathbf{p}_3 + 0.5(-3u^2 + 20u - 32)\mathbf{p}_4 + (u\text{-}3)^2\mathbf{p}_5$

- Represent both $\mathbf{P}_1(u)$ and $\mathbf{P}_4(u)$ same as $UM_s[\mathbf{p}_{i-1}\ \mathbf{p}_i\ \mathbf{p}_{i+1}]^T$

$\mathbf{P}_1(u) = UM_s[\mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2]^T$

$\mathbf{P}_2(u) = UM_s[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]^T$

$\mathbf{P}_3(u) = UM_s[\mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4]^T$

$\mathbf{P}_4(u) = UM_s[\mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_5]^T$

This is called a Uniform or Periodic B-Spline. It is “periodic” because the basis function UM_s repeats itself ***identically*** over successive intervals of the parameter variable u, whereas a NU&C only does it for some interior intervals.

B-Spline Curve

Uniform B-Spline curve (n, K)

- Knots vector

  $\{t_0, \ldots, t_{n+K}\} = \{0, 1, \ldots, n+K\}$

- Parameter u range [0, n-K+2]

- n-K+2 normalized B-Spline segment $\mathbf{P}_1(u), \ldots \mathbf{P}_{n-K+2}(u)$

$$\mathbf{P}(u) = \mathbf{P}_1(u) \oplus \mathbf{P}_2(u) \oplus \ldots \oplus \mathbf{P}_{n-K+2}(u) \qquad 0 \le u < n-K+2$$

$$\mathbf{P}(u) = \mathbf{P}_1(u) = UM_s[\mathbf{p}_0\ \mathbf{p}_1 \ldots \mathbf{p}_{K-1}]^T \qquad \text{for } 0 \le u < 1$$

$$\vdots$$

$$\mathbf{P}(u) = \mathbf{P}_i(u-i+1) = UM_s[\mathbf{p}_{i-1}\ \mathbf{p}_i \ldots \mathbf{p}_{K+i-2}]^T \qquad \text{for } i-1 \le u < i$$

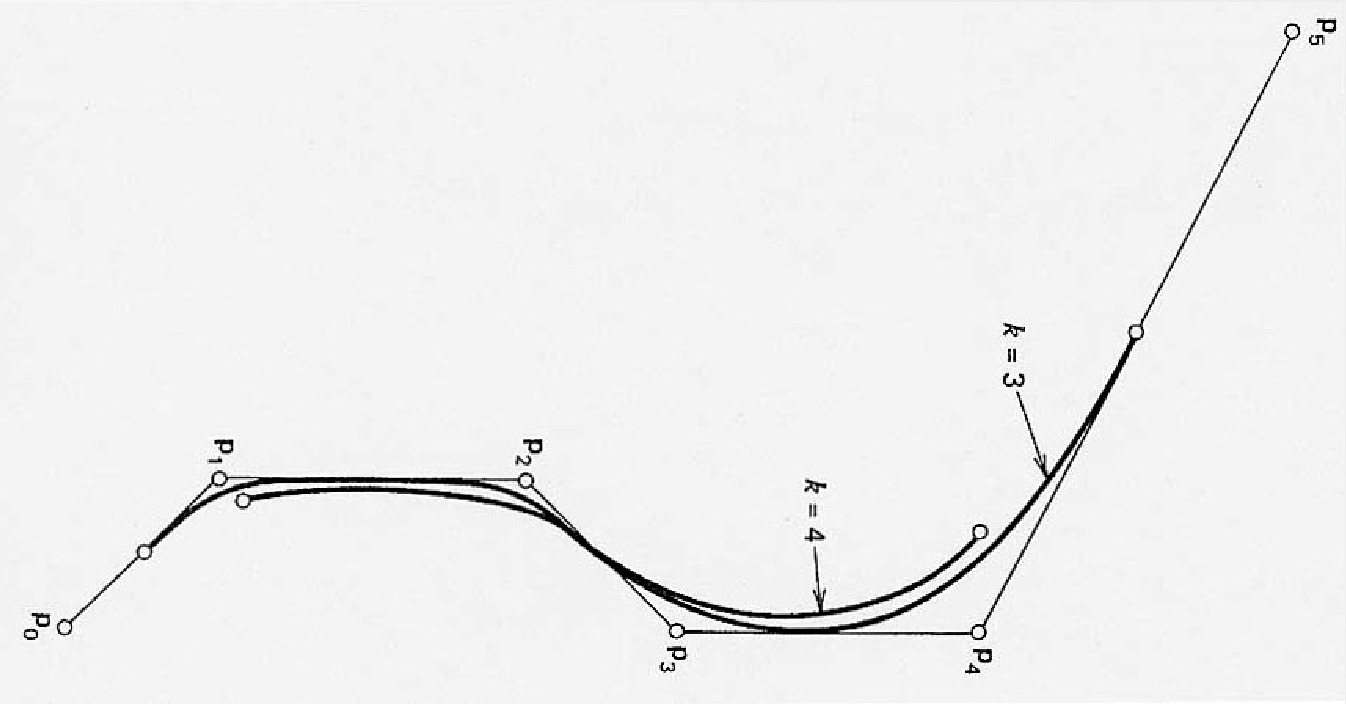
$$\vdots$$

$$\mathbf{P}(u) = \mathbf{P}_{n-K+2}(u-n+K-1) = UM_s[\mathbf{p}_{n-K+1}\ \mathbf{p}_{n-K+2} \ldots \mathbf{p}_n]^T \qquad \text{for } n-K+1 \le u < n-K+2$$

- Question: Why the knots range from 0 to n+K, but the parameter u range is only from 0 to n-K+2?

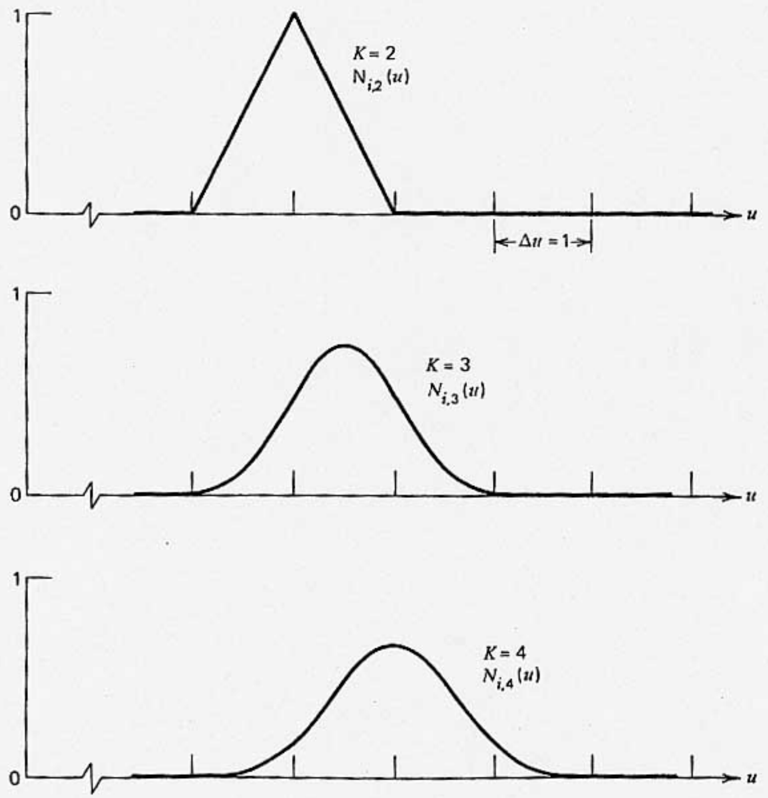
B-Spline Curve

Uniform B-Spline curve examples



B-Spline Curve

- B-Spline basis functions UM$_s$



Uniform B-Spline basis functions: $N_{i,2}(u)$, $N_{i,3}(u)$, and $N_{i,4}(u)$.

B-Spline Curve

- Quadratic and cubic B-Spline basis functions

- Quadratic (K = 3)

$N_3 = U_3 M_3 = [N_{1,3}(u) \quad N_{2,3}(u) \quad N_{3,3}(u)]$

$N_{1,3}(u) = 0.5(u^2 - 2u + 1)$

$N_{2,3}(u) = 0.5(-2u^2 + 2u + 1)$

$N_{3,3}(u) = 0.5u^2$

- Cubic (K = 4)

$N_4 = U_4 M_4 = [N_{1,4}(u) \quad N_{2,4}(u) \quad N_{3,4}(u) \quad N_{4,4}(u)]$

$N_{1,4}(u) = (1/6)(-u^3 + 3u^2 - 3u + 1)$

$N_{2,4}(u) = (1/6)(3u^3 - 6u^2 + 4)$

$N_{3,4}(u) = (1/6)(-3u^3 + 3u^2 + 3u + 1)$

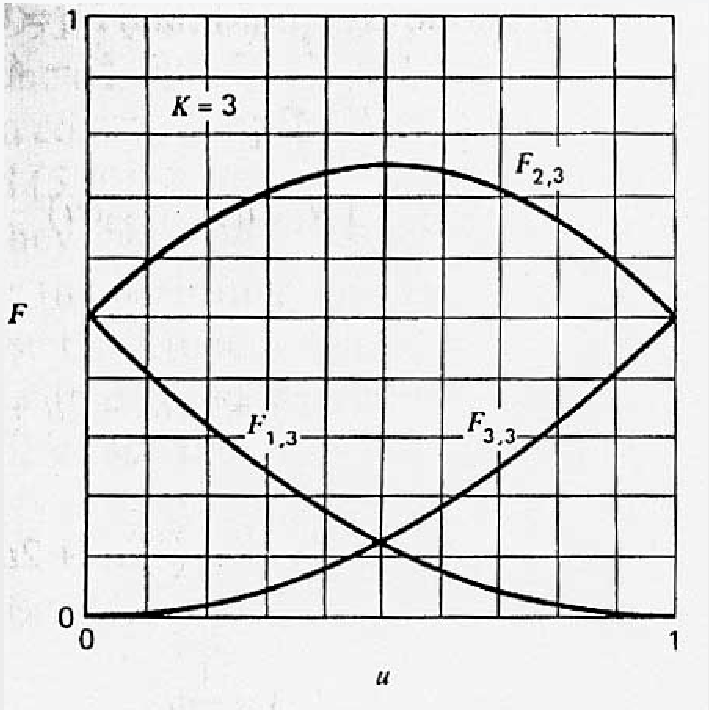
$N_{4,4}(u) = (1/6)u^3$

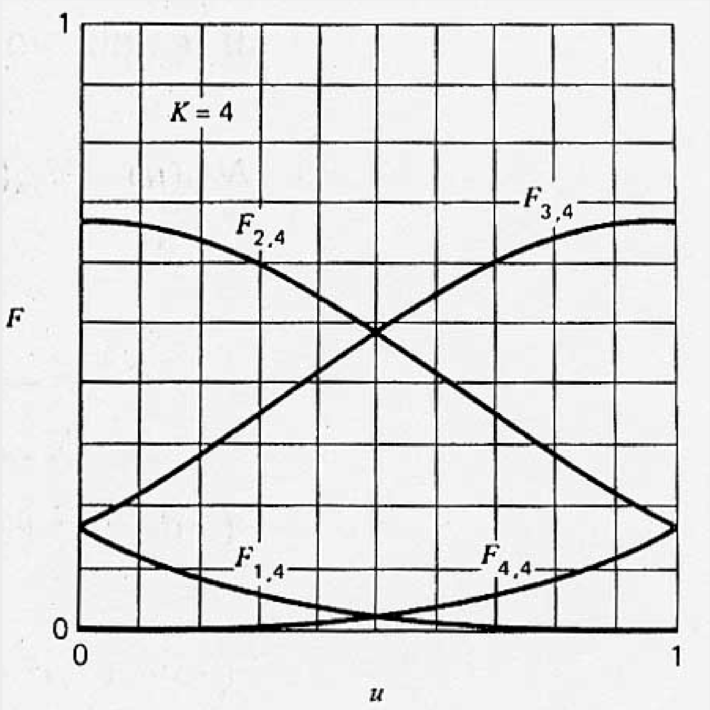
- Partition of unity

$$\sum_{i=1}^{K} N_{i,K}(u) = 1$$ for any u (proof!)

B-Spline Curve

- Quadratic and cubic B-Spline basis functions





Quadratic basis functions: $F_{i,3} = N_{i,3}$

Cubic basis functions: $F_{i,4} = N_{i,4}$

B-Spline Curve

Closed uniform B-Spline curves

- For open uniform B-Spline curves (n, K)

$$\mathbf{P}_i(u) = UM_s[\mathbf{p}_{i-1}\ \mathbf{p}_i\ \ldots\ \mathbf{p}_{K+i-2}]^T; \qquad i=1,2,\ldots,n-K+2$$

- For closed uniform B-Spline curves (n, K)

$$\mathbf{P}_i(u) = UM_s[\mathbf{p}_{(i-1)\bmod(n+1)}\ \ \mathbf{p}_{i\bmod(n+1)}\ \cdots\ \ \mathbf{p}_{(K+i-2)\bmod(n+1)}]^T;$$

$i=1,2,\ldots,n+1$

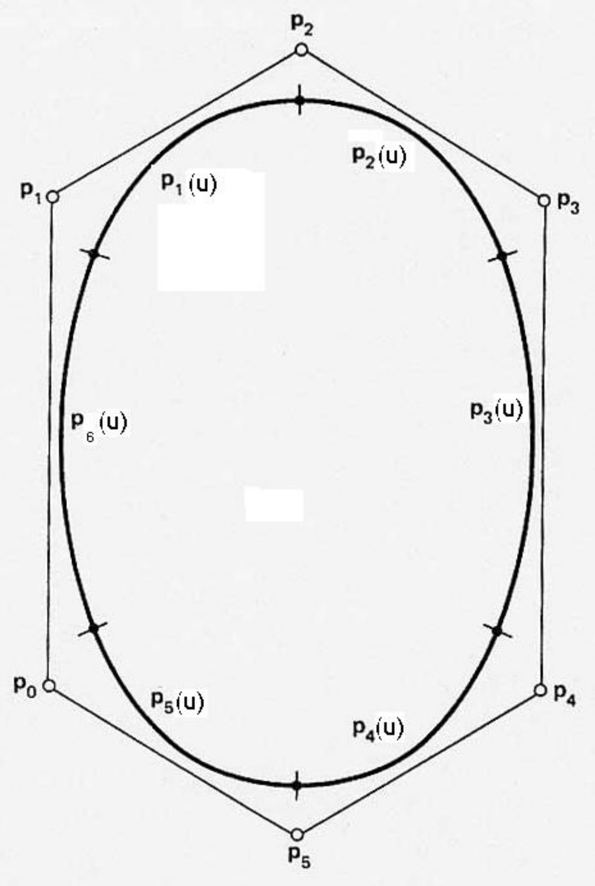
Example: $(n = 5, K = 4)$

$$\mathbf{P}_1(u) = U_4M_4[\mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]^T$$
$$\mathbf{P}_2(u) = U_4M_4[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4]^T$$
$$\mathbf{P}_3(u) = U_4M_4[\mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_5]^T$$

$$\mathbf{P}_4(u) = U_4M_4[\mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_5\ \mathbf{p}_0]^T$$
$$\mathbf{P}_5(u) = U_4M_4[\mathbf{p}_4\ \mathbf{p}_5\ \mathbf{p}_0\ \mathbf{p}_1]^T$$
$$\mathbf{P}_6(u) = U_4M_4[\mathbf{p}_5\ \mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2]^T$$

B-Spline Curve

Closed uniform B-Spline curves (n=5, K=4)

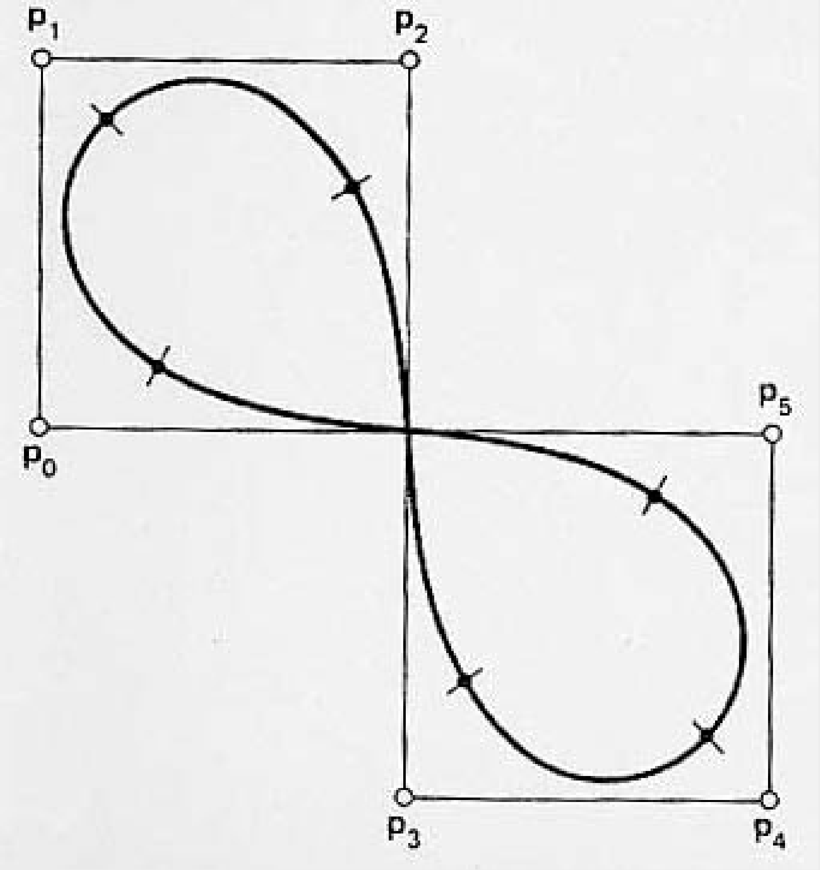


To the open curve, only three segments $\mathbf{P}_1(u)$, $\mathbf{P}_2(u)$ and $\mathbf{P}_3(u)$ are defined.
http://theory.lcs.mit.edu/~boyko/classes/b-spline.html

But to the closed curve, three more segments $\mathbf{P}_4(u)$, $\mathbf{P}_5(u)$ and $\mathbf{P}_6(u)$ are defined to form a closed loop.

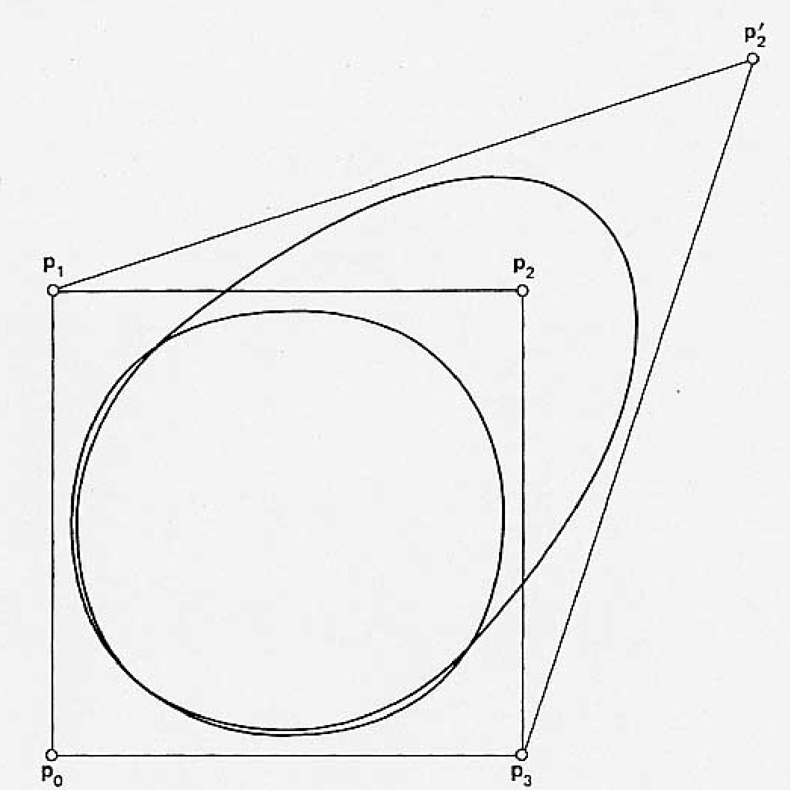
B-Spline Curve

A self-intersecting closed uniform B-Spline



B-Spline Curve

Effect of moving the control points



Closed B-Spline curve with $n = 3$, $K = 4$.

Example (n=3, K=4):

$\mathbf{P}$(U) is a closed B-Spline consisting of:

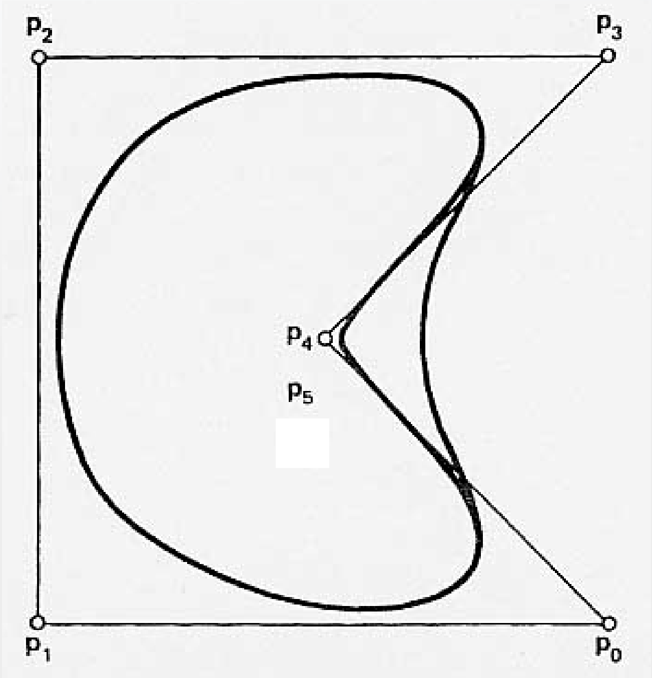
$$\mathbf{P}_1(u) = U_4 M_4[\mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]^T$$
$$\mathbf{P}_2(u) = U_4 M_4[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_0]^T$$
$$\mathbf{P}_3(u) = U_4 M_4[\mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_0\ \mathbf{p}_1]^T$$
$$\mathbf{P}_4(u) = U_4 M_4[\mathbf{p}_3\ \mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2]^T$$

Pulling $\mathbf{p}_2$ affects the curve mostly near $\mathbf{p}_2$.

Question: Does the perturbation of $\mathbf{p}_2$ affects the entire curve? How about when n=4?

B-Spline Curve

- Effect of multiply coincident control points



Example (K=4):

n = 4:

$\mathbf{P}_1(u) = U_4M_4[\mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]^T$
$\mathbf{P}_2(u) = U_4M_4[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4]^T$
$\mathbf{P}_3(u) = U_4M_4[\mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_0]^T$
$\mathbf{P}_4(u) = U_4M_4[\mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_0\ \mathbf{p}_1]^T$
$\mathbf{P}_4(u) = U_4M_4[\mathbf{p}_4\ \mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2]^T$

n = 5 and $\mathbf{p}_5 = \mathbf{p}_4$:

$\mathbf{P}_1(u) = U_4M_4[\mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]^T$
$\mathbf{P}_2(u) = U_4M_4[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4]^T$
$\mathbf{P}_3(u) = U_4M_4[\mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_5]^T$
$\mathbf{P}_4(u) = U_4M_4[\mathbf{p}_3\ \mathbf{p}_4\ \mathbf{p}_5\ \mathbf{p}_0]^T$
$\mathbf{P}_5(u) = U_4M_4[\mathbf{p}_4\ \mathbf{p}_5\ \mathbf{p}_0\ \mathbf{p}_1]^T$
$\mathbf{P}_6(u) = U_4M_4[\mathbf{p}_5\ \mathbf{p}_0\ \mathbf{p}_1\ \mathbf{p}_2]^T$

B-Spline Curve

Continuity of uniform B-Spline curves

- The basis function $N_{i,K}(u)$ is a polynomial of degree K-1

  Example: $N_{2,4}(u) = (1/6)(3u^3 – 6u^2 + 4)$ $(K = 4)$

- The segment $\mathbf{P}_i(u) = N_{1,K}(u)\mathbf{p}_{i-1} + N_{2,K}(u)\mathbf{p}_i + \ldots + N_{K,K}(u)\mathbf{p}_{i+K-2}$ is a simple summation of the $N_{i,K}(u)$ and hence is of class C^{K-1}

- The B-Spline curve $\mathbf{P}(u) = \mathbf{P}_1(u) \oplus \mathbf{P}_2(u) \oplus \ldots \oplus \mathbf{P}_{n-K+2}(u)$ is a curve of class C^{K-2}, i.e., the $(K-2)^{th}$-derivative exists and is continuous on the entire $\mathbf{P}(u)$.

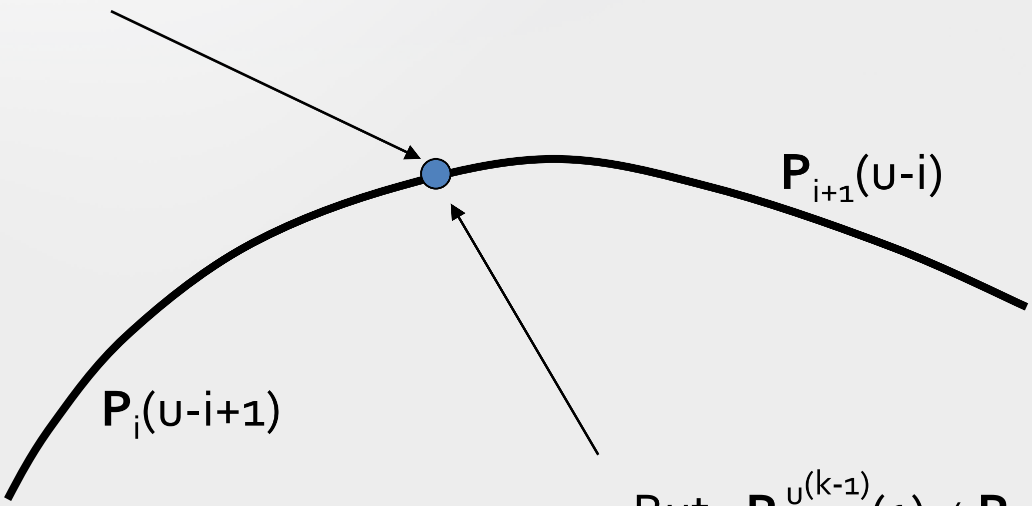
  Example (K=4):

$$\mathbf{P}(i) = \mathbf{P}_i(1) = (1/6)(\mathbf{p}_i + 4\mathbf{p}_{i+1} + \mathbf{p}_{i+2}) = \mathbf{P}_{i+1}(0)$$
$$\mathbf{P}^u(i) = \mathbf{P}_i^u(1) = 0.5(-\mathbf{p}_i + \mathbf{p}_{i+2}) = \mathbf{P}_{i+1}^u(0)$$
$$\mathbf{P}^{uu}(i) = \mathbf{P}_i^{uu}(1) = \mathbf{p}_i - 2\mathbf{p}_{i+1} + \mathbf{p}_{i+2} = \mathbf{P}_{i+1}^{uu}(0)$$
$$\mathbf{P}^{uuu}(i) = ???$$

B-Spline Curve

Continuity of B-Spline curves

At u=i: $\mathbf{P}^{u^{(K-2)}}(i) = \mathbf{P}_i^{u^{(K-2)}}(1) = \mathbf{P}_{i+1}^{u^{(K-2)}}(0)$



But, $\mathbf{P}_i^{u^{(k-1)}}(1) \neq \mathbf{P}_{i+1}^{u^{(k-1)}}(0)$

B-Spline Curve

Conversion between Bezier and B-Spline (K = 4)

Bezier: $\mathbf{P}(u) = UM_B\mathbf{P}_B$

B-Spline: $\mathbf{P}(u) = UM_S\mathbf{P}_S$

B-Spline → Bezier:

$$UM_B\mathbf{P}_B = UM_S\mathbf{P}_S \rightarrow \mathbf{P}_B = M^{-1}{}_B M_S\mathbf{P}_S \rightarrow \mathbf{P}_B = \frac{1}{6}\begin{bmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 0 \\ 0 & 1 & 4 & 1 \end{bmatrix}\mathbf{P}_S$$

Bezier → B-Spline:

$$UM_S\mathbf{P}_S = UM_B\mathbf{P}_B \rightarrow \mathbf{P}_S = M^{-1}{}_S M_B\mathbf{P}_B \rightarrow \mathbf{P}_S = \begin{bmatrix} 6 & -7 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -7 & 6 \end{bmatrix}\mathbf{P}_B$$

B-Spline Curve

Bezier curve as a special NU&C B-spline

- A nonuniform and clamped B-Spline curve

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

with knots vector {0 … 0   1 2 … n-K+1   n-K+2 … n-K+2}

K   n-K   K

- If n = K, then it becomes a Bezier curve of order K, as now

$N_{i,K}(u) = B_{i,K}(u)$ for i=1,2,…, K.

B-Spline Curve

Knot insertion

Old B-spline:

n+K+1 knots: $U = \{u_0, u_1, \ldots, u_{n+K+1}\}$;

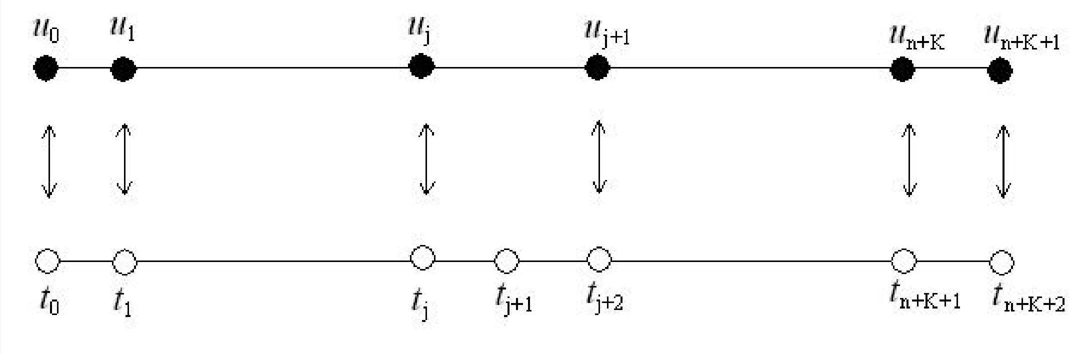
$$\mathbf{P}(\mathrm{u}) = \sum_{i=0}^{n} \mathbf{p}_i \, N_{i,K}(\mathrm{u})$$

$$||$$

New B-spline:

n+K+2 knots: $\{t_0, t_1, \ldots, t_{n+K+2}\}$;

$$\boldsymbol{Q}(\mathrm{u}) = \sum_{i=0}^{n+1} \boldsymbol{q}_i \, \overline{N}_{i,K}(\mathrm{u})$$



B-Spline Curve

Knot insertion

For i=0,1,…,j-K: $\overline{N}_{i,K}(u) = N_{i,K}(u)$

For i=j+1,…,n: $\overline{N}_{i+1,K}(u) = N_{i,K}(u)$

For $u \in [u_j, u_{j+1}]$: $\sum_{i=j-K+1}^{j} \boldsymbol{p}_i N_{i,K}(u) = \sum_{i=j-K+1}^{j+1} \boldsymbol{q}_i \overline{N}_{i,K}(u)$

$\Downarrow$

$\boldsymbol{q}_i = \boldsymbol{p}_i$: i=0,1,…,j-K

$\boldsymbol{q}_{i+1} = \boldsymbol{p}_i$: i=j+1,…,n

How about the other K+1 control points $\boldsymbol{q}_{j-K+1}$, $\boldsymbol{q}_{j-k+2}$, …, $\boldsymbol{q}_{j+1}$?

B-Spline Curve

Knot insertion

$\boldsymbol{q}_{j-K+1} = \boldsymbol{p}_{j-K+1}$

$\boldsymbol{q}_i = \alpha_i \boldsymbol{p}_i + (1 - \alpha_i)\boldsymbol{p}_{i-1}$ $\qquad$ $j-K+2 \leq i \leq j$

$\boldsymbol{q}_{j+1} = \boldsymbol{p}_j$

where:

$$\alpha_i = \frac{t_{j+1} - u_i}{u_{i+K-1} - u_i}$$

B-Spline Curve

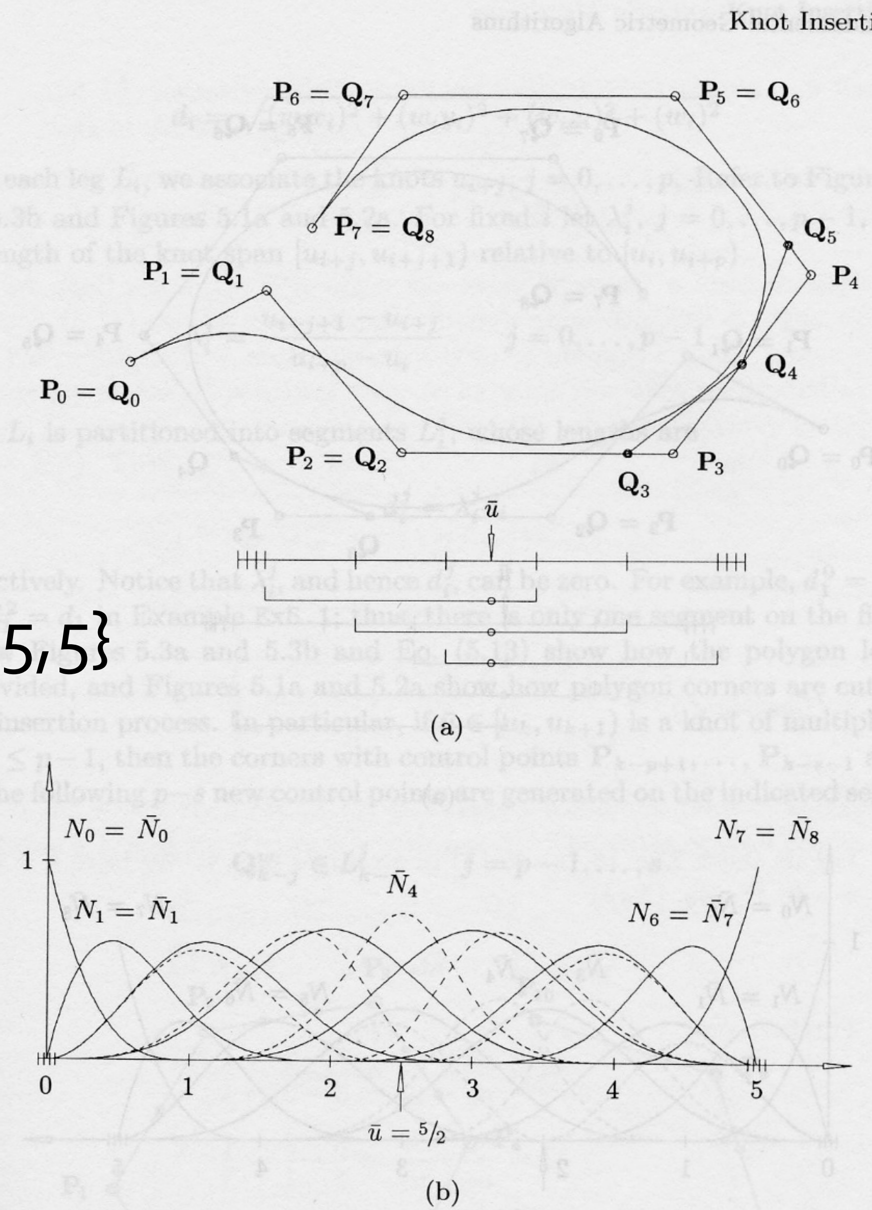
Knot insertion

An example:

K = 4

U = {0,0,0,0,1,2,3,4,5,5,5,5}

j = 5, t_{j+1} = 5/2



B-Spline Curve

Non**U**niform **R**ational **B**-**S**pline (NURBS)

$$\mathbf{P}(u) = \frac{\sum_{i=0}^{n} w_i \, \mathbf{p}_i \, N_{i,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} \qquad \text{for } 0 \le u < n-K+2$$

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

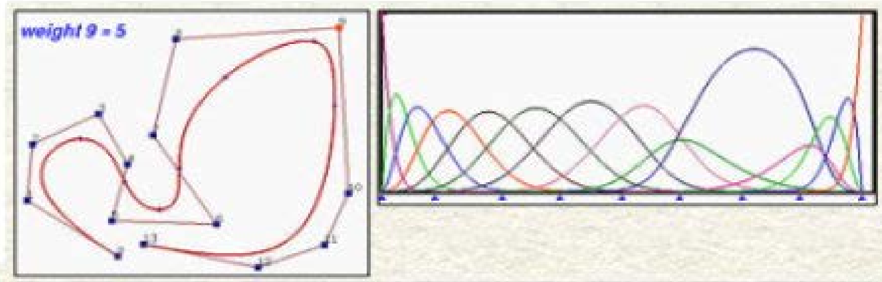
If weights $w_i = 1$ for all i, then it reduces to a nonrational B-Spline.

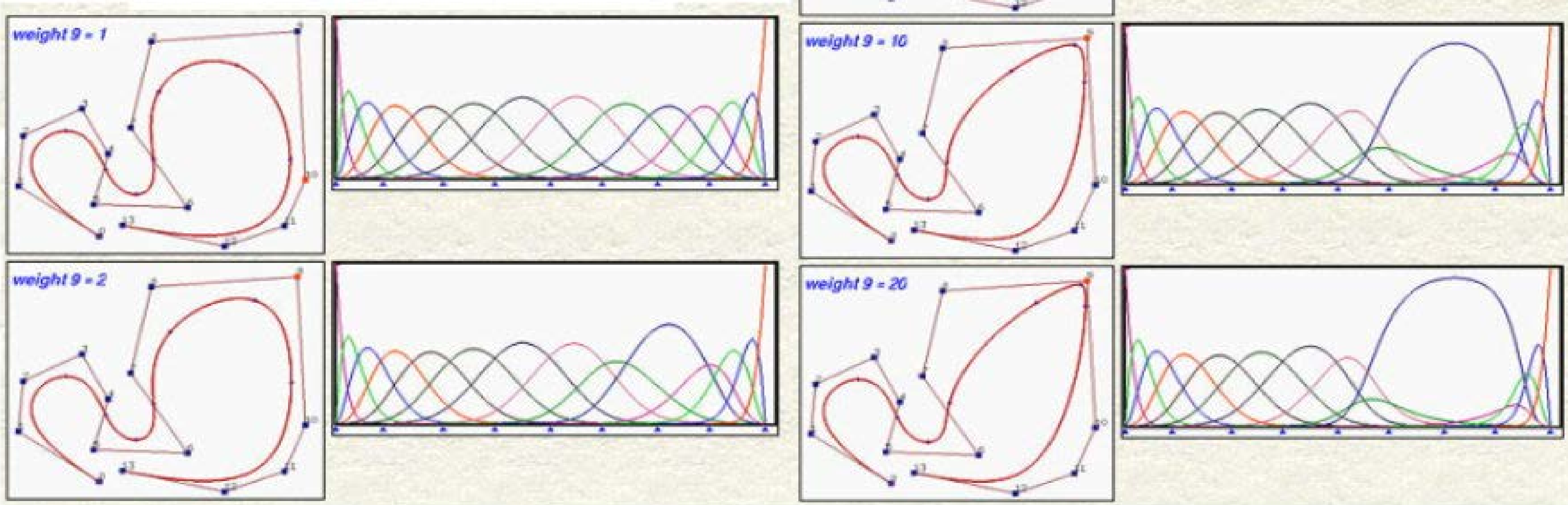
NURBS is the most general and popular representation.
1. All Hermite, Bezier, and B-spline are special cases of NURBS.
2. It can represent exactly conics and other special curves.
3. The weights w_i add one more degree of freedom of curve manipulation.
4. It enjoys all the nice properties of nonrational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

Effect of Weights

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.

B-Spline Curve

Unit partitioning property of NURBS basis functions

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

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

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

Exercise

1. Consider a cubic uniform B-Spline curve (n=5, K=4). Please give the representation of its first derivative.

2. Given a Bezier curve of order 10 (i.e., it has 10 control points), can you represent it by a cubic B-Spline (n, K=4)? Why?

3. Nonuniform and nonclamped B-Spline

   Consider a (n=5, K=3) B-Spline $\mathbf{P}(\mathrm{u}) = \sum_{i=0}^{5} \mathbf{p}_i N_{i,3}(\mathrm{u})$ with knots vector

   $\{t_0 = 0 \quad t_1 = 0 \quad t_2 = 1 \quad t_3 = 2 \quad t_4 = 3 \quad t_5 = 4 \quad t_6 = 5 \quad t_7 = 5 \quad t_8 = 5\}$

   Please derive the basis functions $N_{i,3}(u)$ and give the equations of P(u) on different intervals ($i-1 \le u \le i$), i=1,2,…,

5. What are P(0) and P(5)? What are the derivatives of P(u) at u=0 and u=5?

B-Spline Synthetic Curves

- B-spline curves are specified by giving set of coordinates, called control points, which indicates the general shape of the curve.

- B-splines can be either interpolating or approximating curves. Interpolation splines used for construction and to display the results of engineering.

B-Spline Synthetic Curves

- Approximation B-spines defined as linear and **2nd degree** and the flexibility is provided by the basic functions with **(n+1) control points**, B-splines are defined as

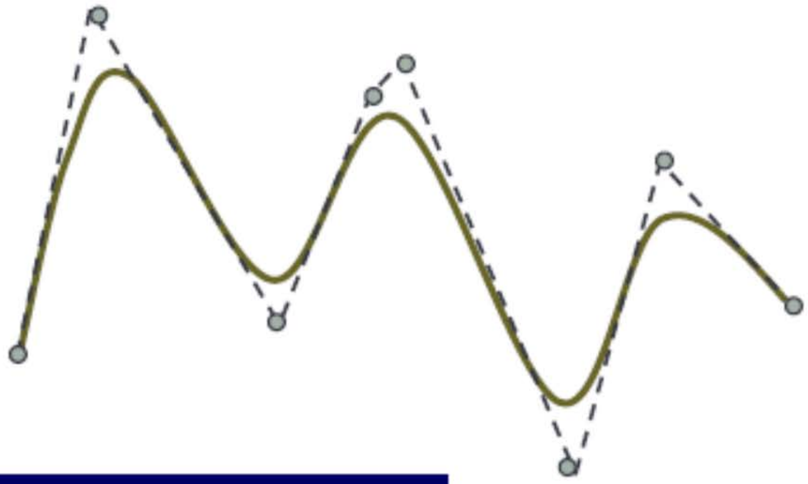
$$C(u) = \sum_{i=0}^{n} P_i N_{i,k}(u) \qquad 0 \le u < u_{max} \qquad 2 \le k \le n+1$$

P : set of control points
u : knot vector
k : spline's degree

B-Spline Synthetic Curves

P is the set of control points as shown in Figure, **N** is the B-spline blending functions and they are defined as,

$$N_{i,0} = \begin{cases} 1 & u_i \le u \le u_{i+1} \\ 0 & otherwise \end{cases}$$



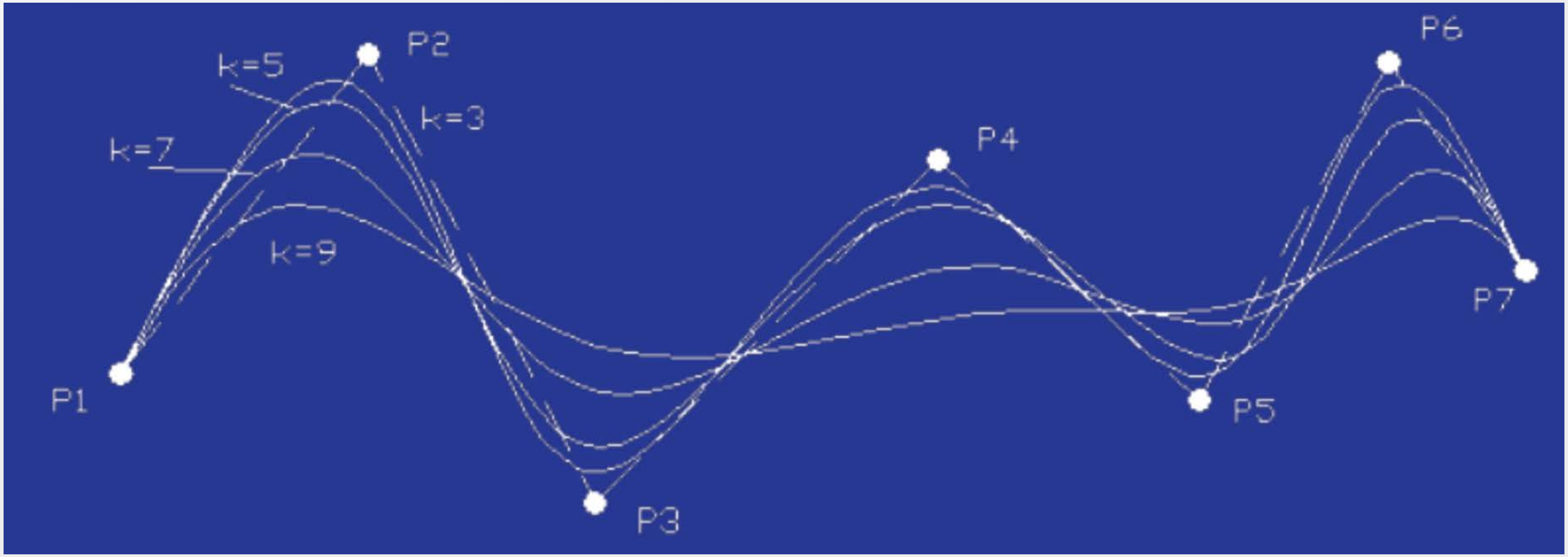
$$N_{i,p}(u) = (u - u_i)\frac{N_{i,p-1}(u)}{(u_{i+p} - u_i)} + (u_{i+p} - u)\frac{N_{i+1,p-1}(u)}{(u_{i+p} - u_{i+1})}$$

Effect of curve order

$$C(u) = \sum_{i=0}^{n} P_i N_{i,k}(u)$$

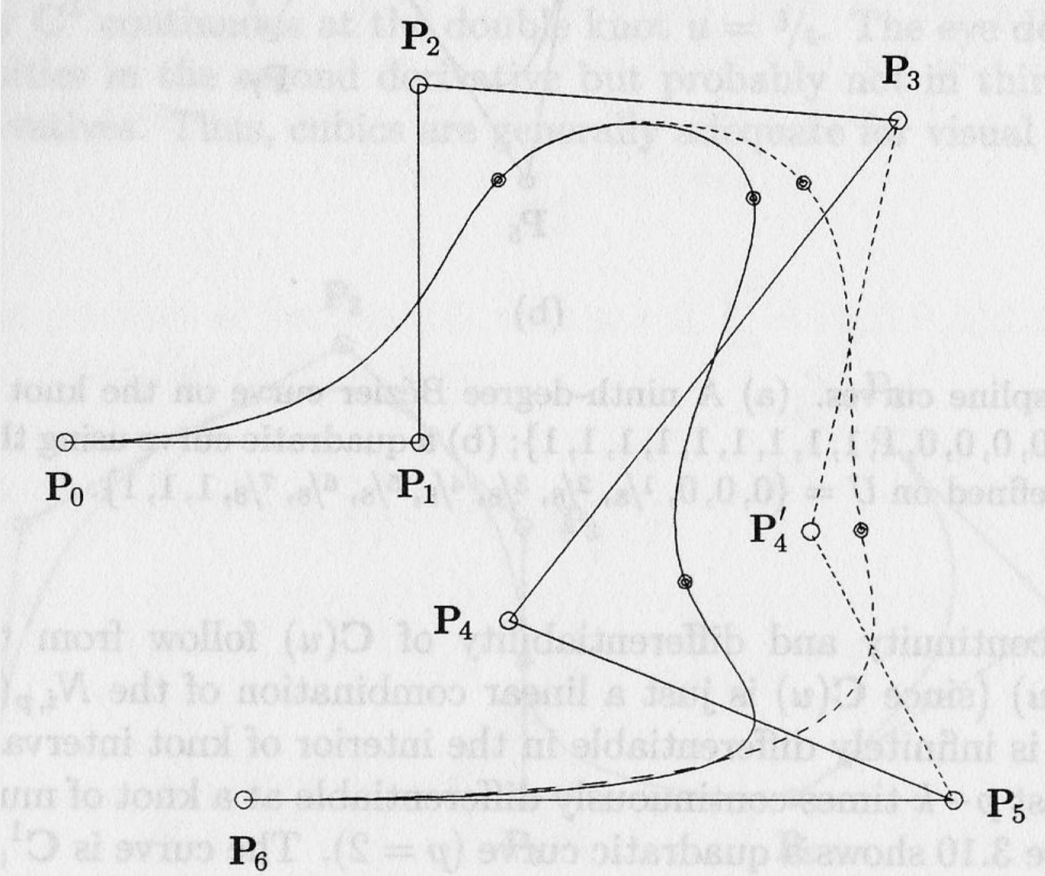
$$0 \le u < u_{max} \qquad 2 \le k \le n+1$$

The range of u is related to the number of control points and the knot vectors. The effect of the degree of the B-spline curves on the shape of the curve is shown in Figure. Bspline curve lays in the control polygon.



Local control property of B-Spline curve

Control point P4 moves to a new position P4' ,
only a portion of the original curve has changed.



The derivative of the B-spline curve

Basic functions defined from the knot vector $U=\{u_0 \ldots u_m\}$ and for $i=0,\ldots m-1$, $u_i \leq u_{i+1}$. u_i knot vector, U is the set of knot vectors. Specifically, a B-spline order p (or degree $p-1$) basic functions are defined as $N_{i,p}(u)$.

The derivative of the B-spline curve

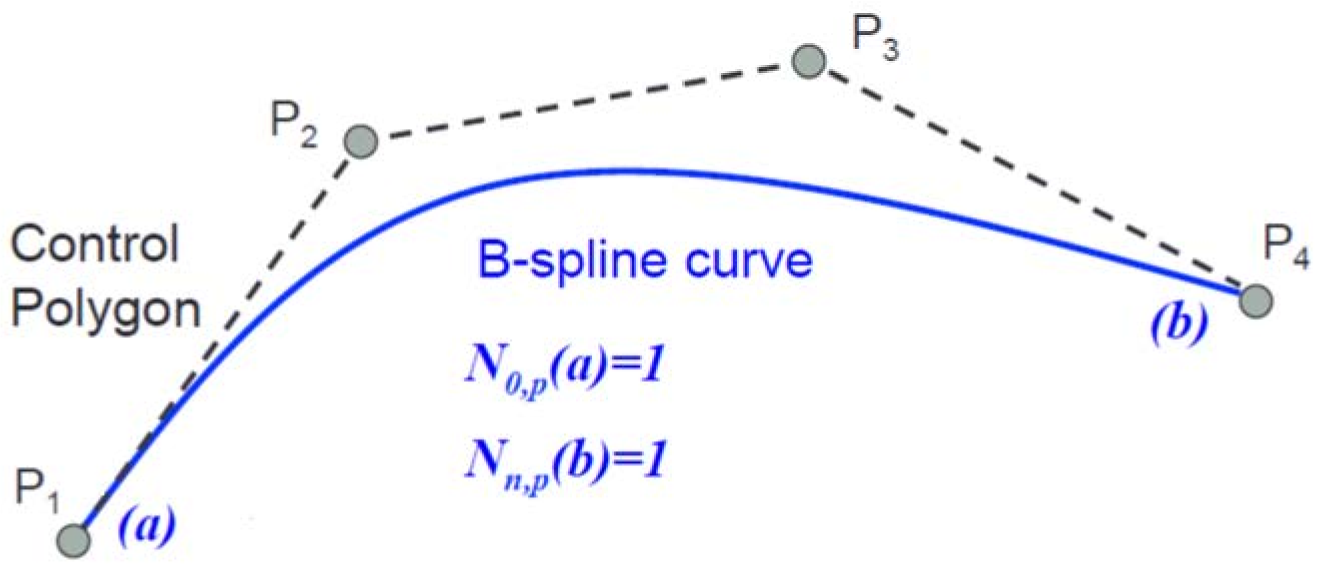
$$N^{1}_{i,p}(u) = \frac{N_{i,p-1}(u)}{(u_{i+p} - u_i)}p + \frac{N_{i+1,p-1}(u)}{(u_{i+p+1} - u_{i+1})}p$$

and the k^{th} derivative of the function is,

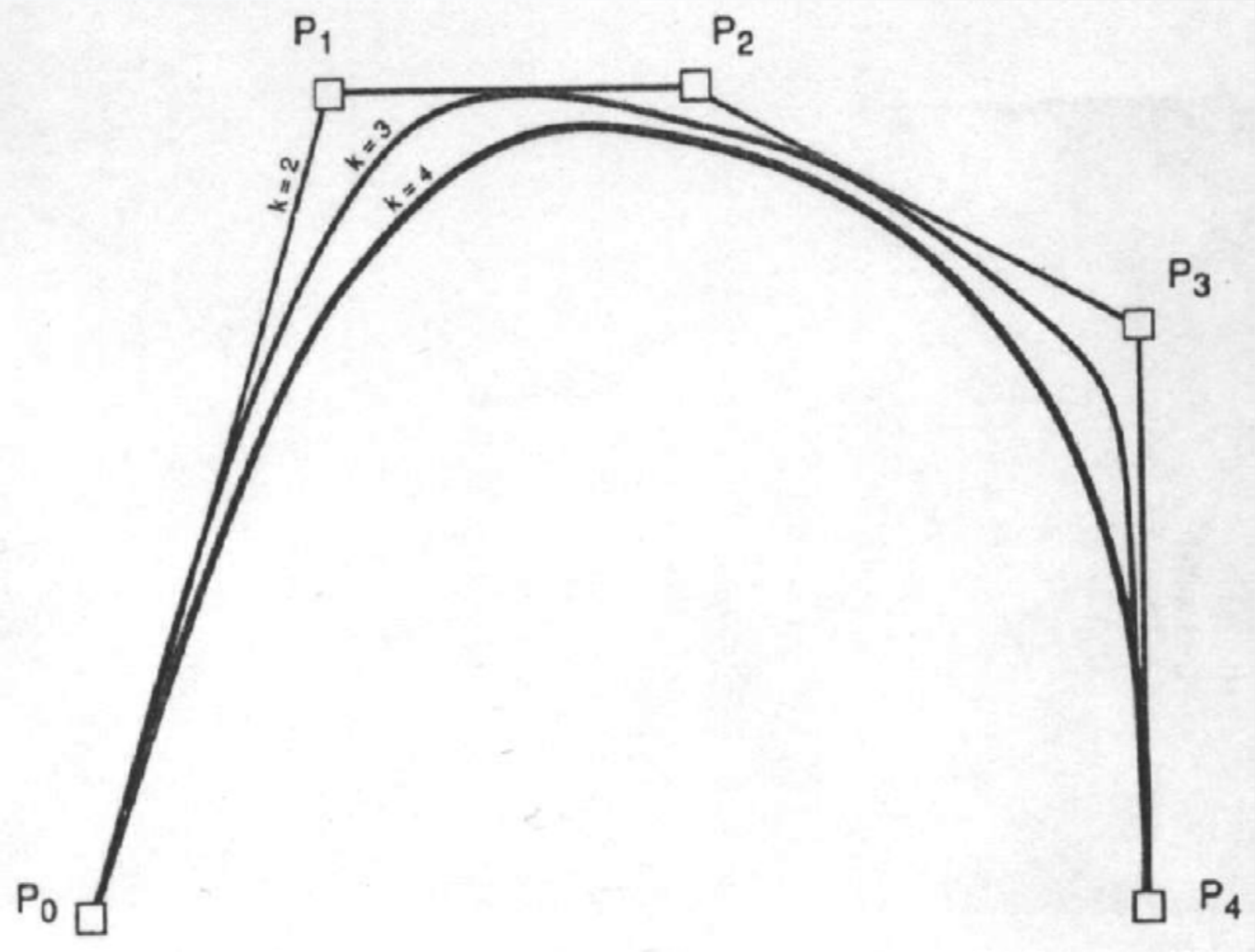
$$N^{k}_{i,p}(u) = p\left[\frac{N^{k-1}_{i-1}}{u_{i+p} - u_i} - \frac{N^{k-1}_{i+1,p-1}}{u_{i+p+1} - u_{i+1}}\right]$$

B-Spline Synthetic Curves

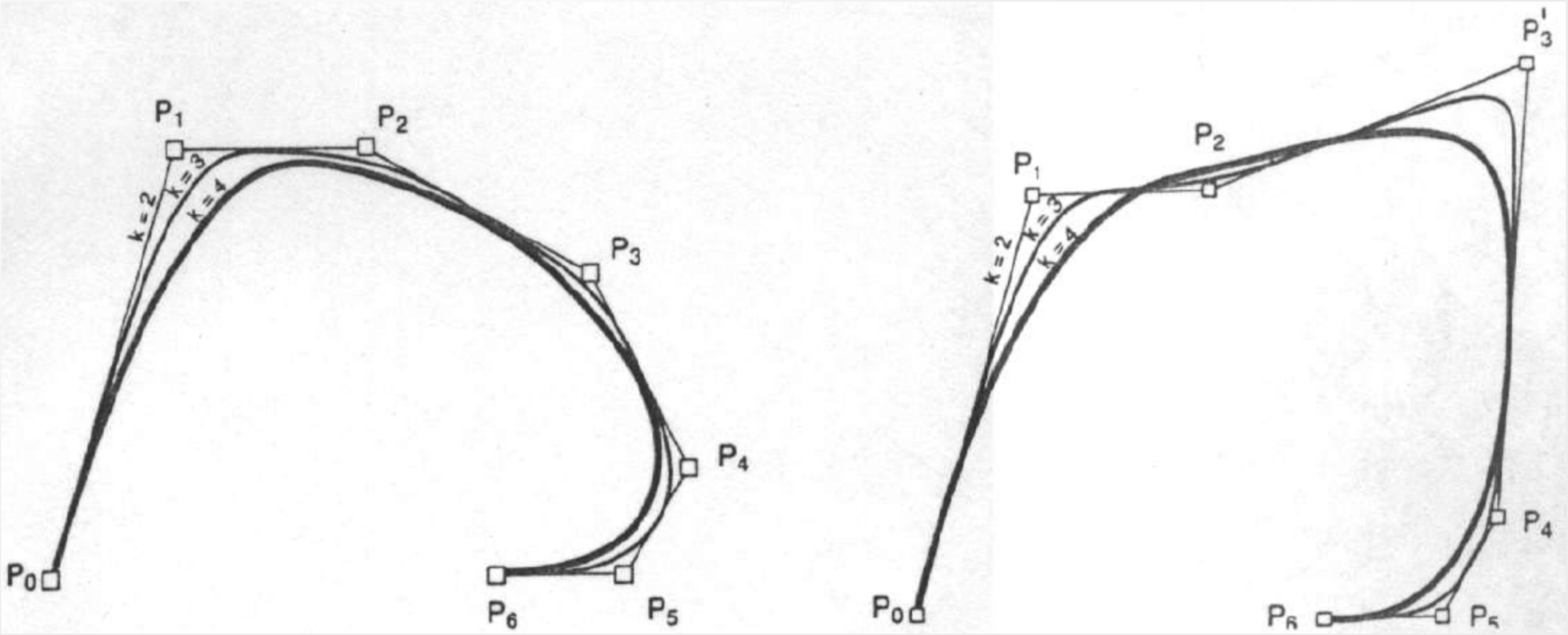
With (m+1) control points, there are always (n=m+p-1) basic functions. The basis functions are 1, at the end points of the curve defined as a and b. If there's no other definition, then a=0 and b=1. $\{P_i\}$ the set of the control points forms the control polygon from Figure.



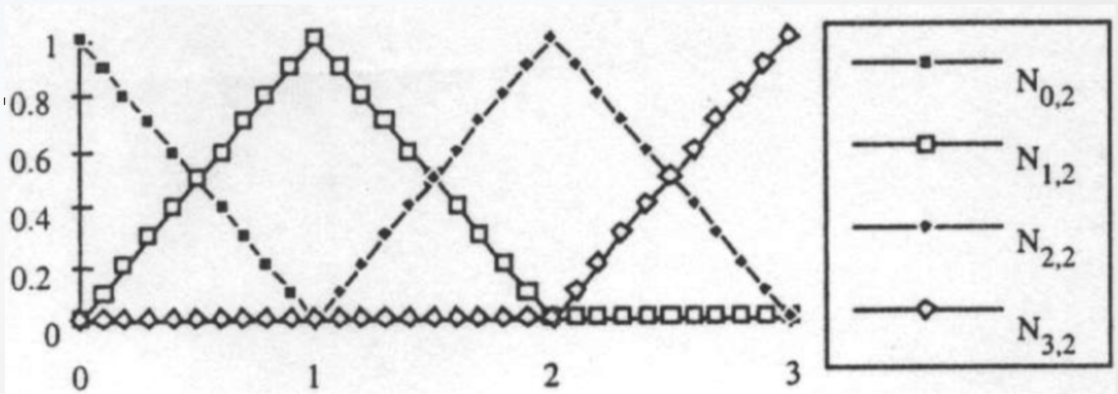
Linear, quadratic, cubic B-spline

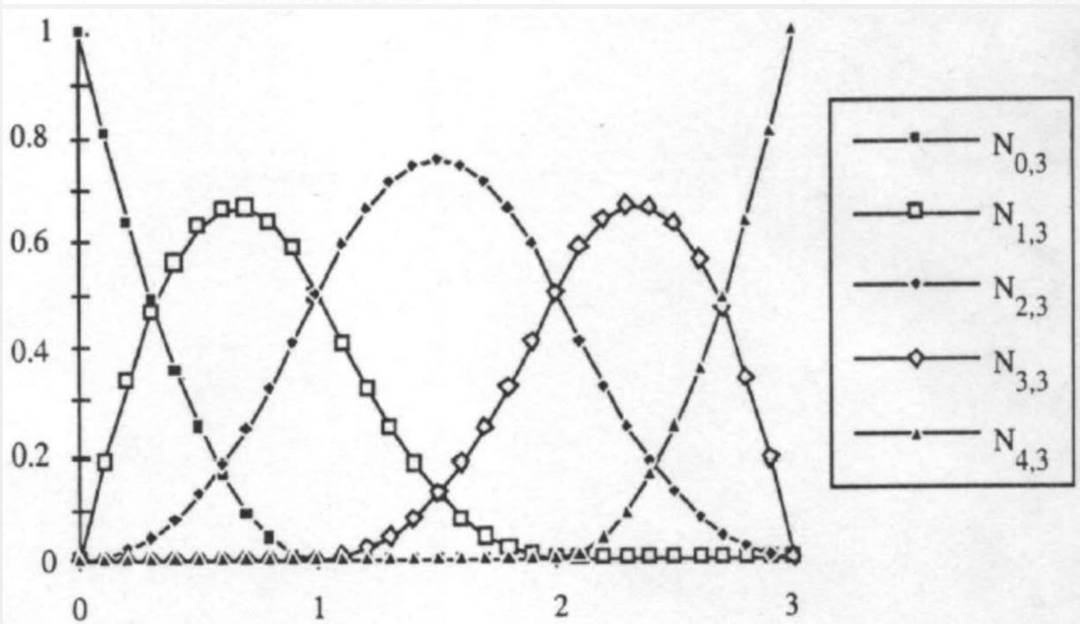


Influence of control point position



Blending functions for linear B-spline

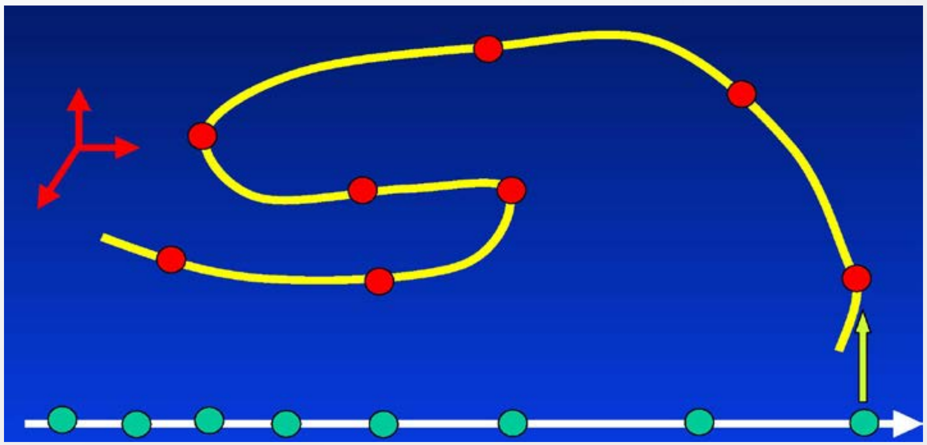




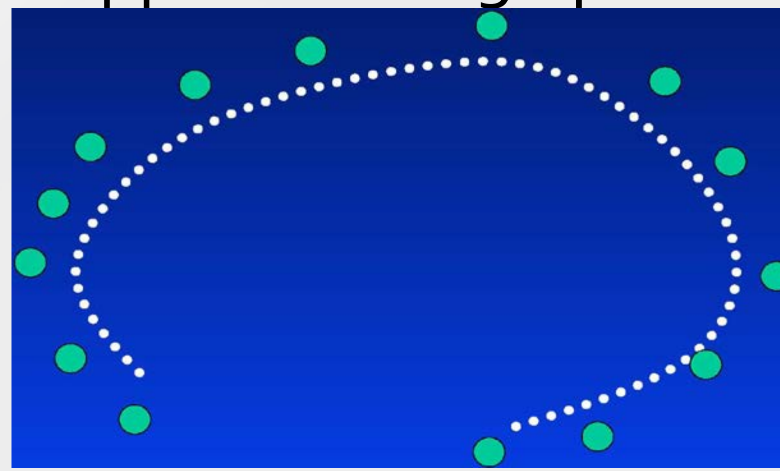
Quadratic B-spline blending fn (k=3)

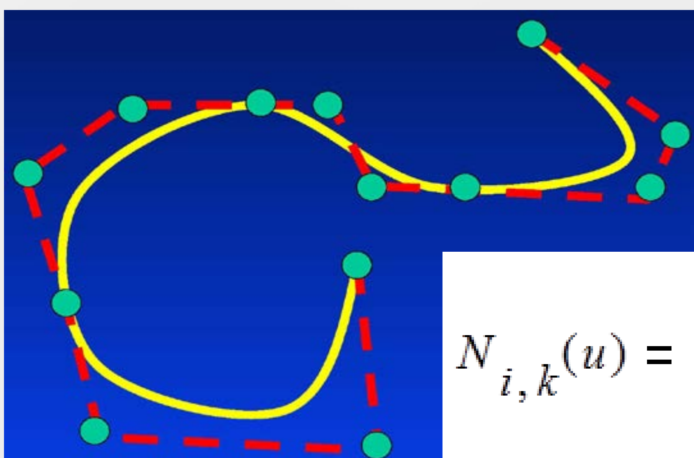
B-spline Curves

Piecewise Polynomials



Approximating Splines



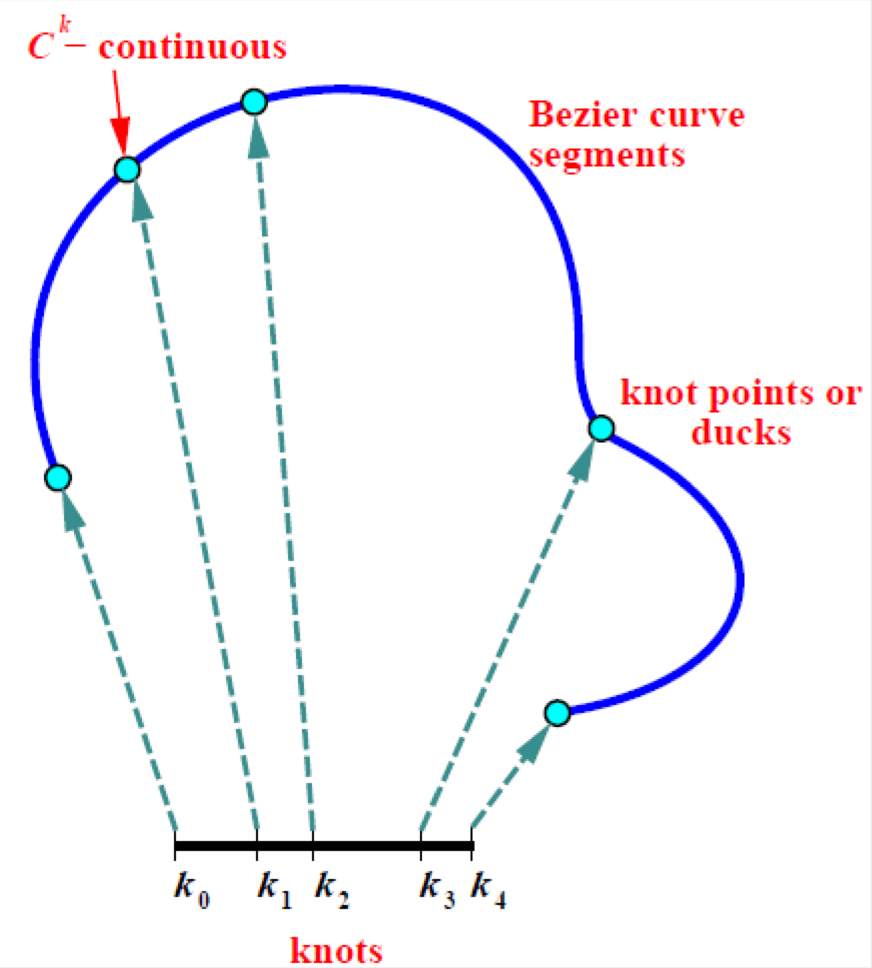


Spline eq. $$P(u) = \sum_{i=0}^{n} N(i,k,u)\, P_i$$

$$N_{i,k}(u) = \left(u - u_i\right)\cdot\frac{N_{i,k-1}}{u_{i+k-1} - u_i} + \left(u_{i+k} - u\right)\cdot\frac{N_{i+1,k-1}}{u_{i+k} - u_{i+1}}$$

B-spline Curves

a B–spline curve is the union of a number of Bezier curves joining together with
C – continuity.

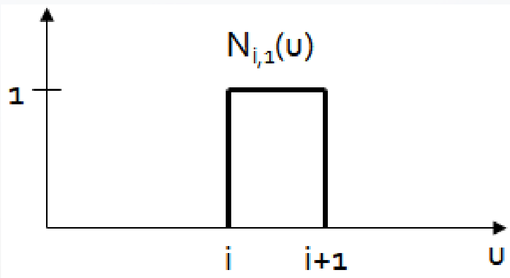


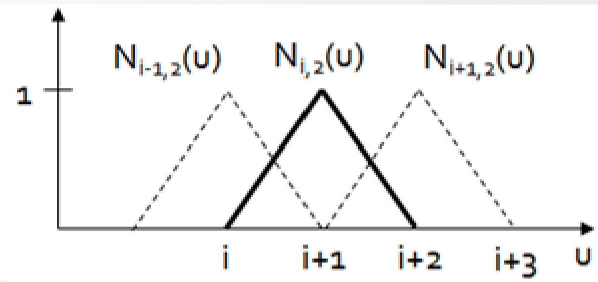
B-spline Curve

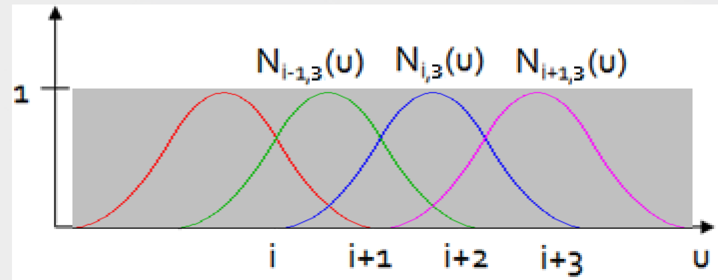
Blending functions

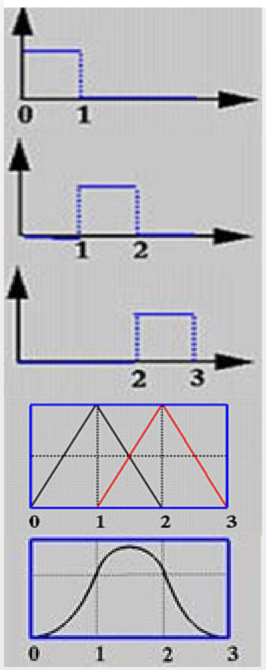
$$N_{i,1}(u) = \begin{cases} 1, & u_i \le u \le u_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

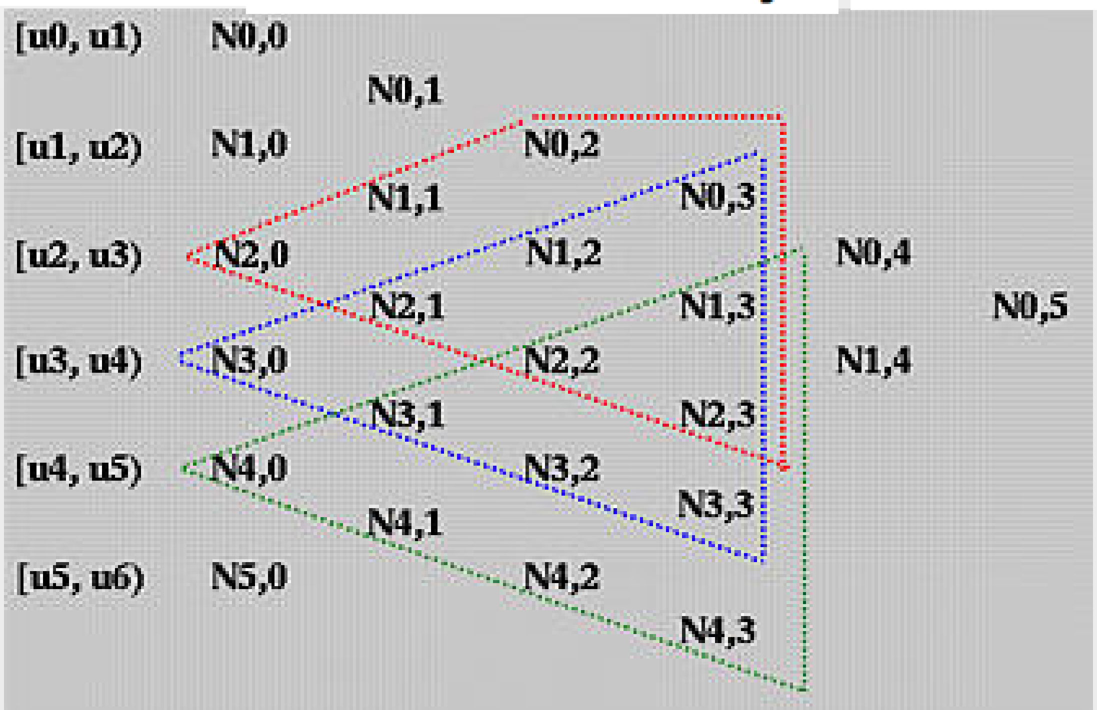
$$N_{i,k}(u) = (u - u_i)\frac{N_{i,k-1}(u)}{u_{i+k-1} - u_i} + (u_{i+k} - u)\frac{N_{i+1,k-1}(u)}{u_{i+k} - u_{i+1}}$$

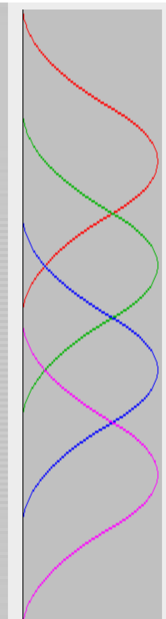


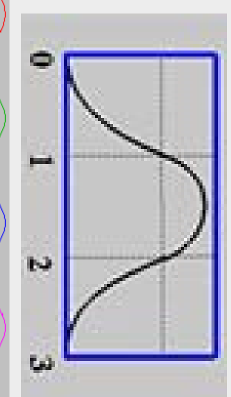






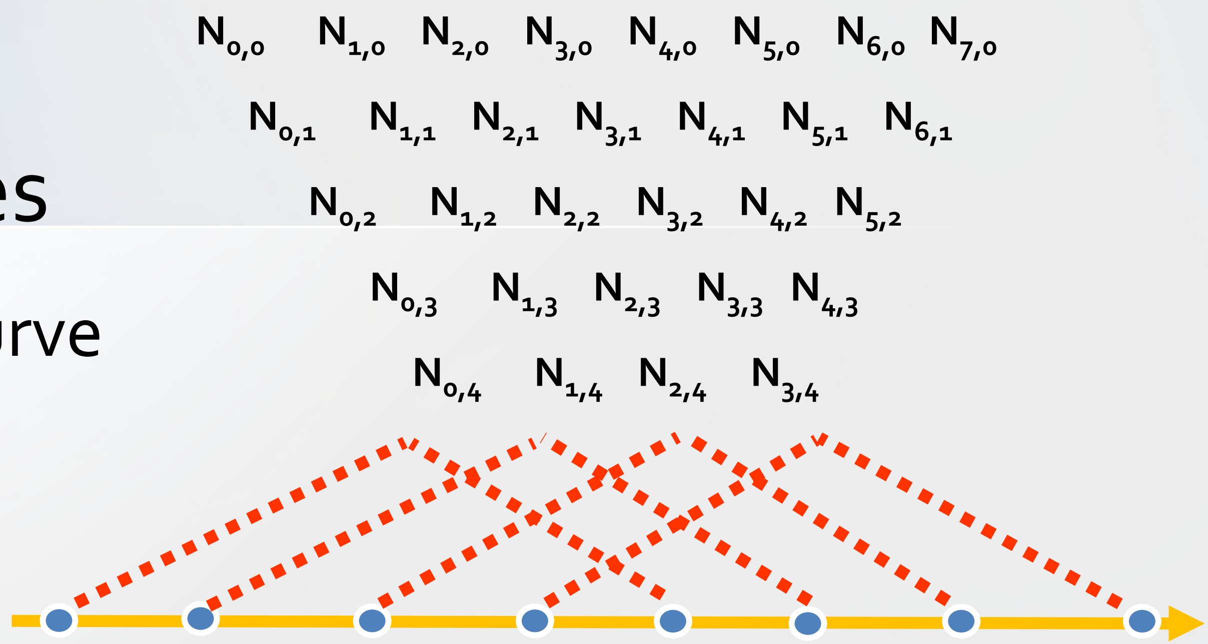






B-spline curves

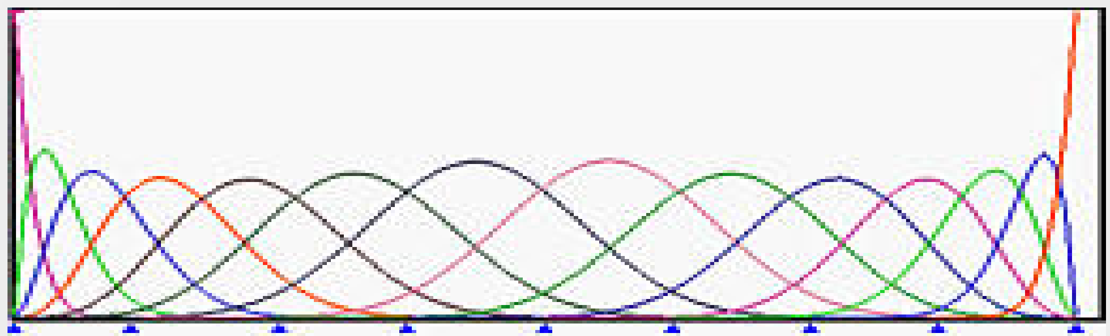
Since a NURBS curve is rational, circles, ellipses and hyperbolas can be represented

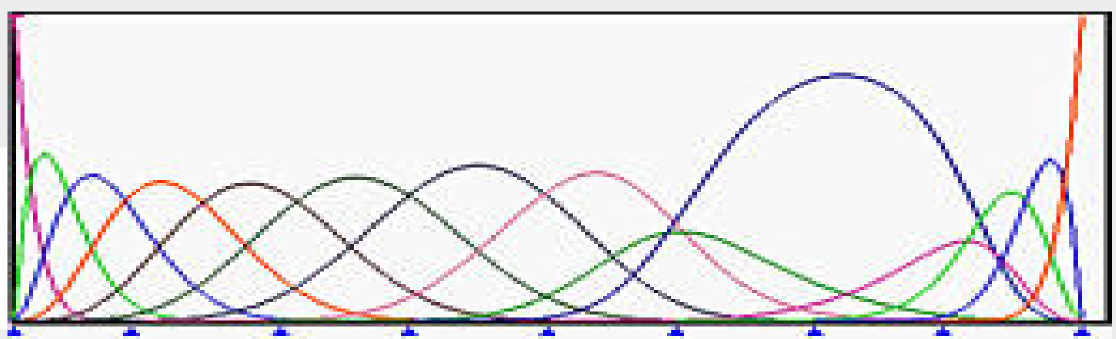


NURBS

$$\mathbf{C}(u) = \sum_{i=0}^{n} R_{i,p}(u)\mathbf{P}_i$$

$$R_{i,p}(u) = \frac{N_{i,p}(u)w_i}{\sum_{j=0}^{n} N_{j,p}(u)w_j}$$





Bezier curves

Bezier curve difficulties

Bernstein basis is global, No local control

Order (degree) fixed, Equal to number of control vertices

High order (degree) required for flexibility

Wiggles

Difficult to maintain continuity

ref. Sigg2002 Course57 NURBS.pdf

B-spline curves – Definition

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t) \quad t_{\min} \le t < t_{\max}, \quad 2 \le k \le n+1$$

B_i s are the polygon control vertices

$N_{i,k}(t)$ are the normalized B-spline basis functions of order k

$n+1$ is the number of control vertices

B-spline curves – Basis functions

$$N_{i,1}(t) = \begin{cases} 1 & \textbf{if } x_i \le t < x_{i+1} \\ 0 & \textbf{otherwise} \end{cases}$$

$$N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

x_i **s** **are the elements of a knot vector**

Note $0/0 \equiv 0$

B-spline curves – Properties

$\sum N_{i,k}(t) \equiv 1$ for all t

$N_{i,k}(t) \geq 0$ for all t

Maximum order $k_{max} = n + 1$

Maximum degree, n, is one less than the order

Exhibits the variation diminishing property

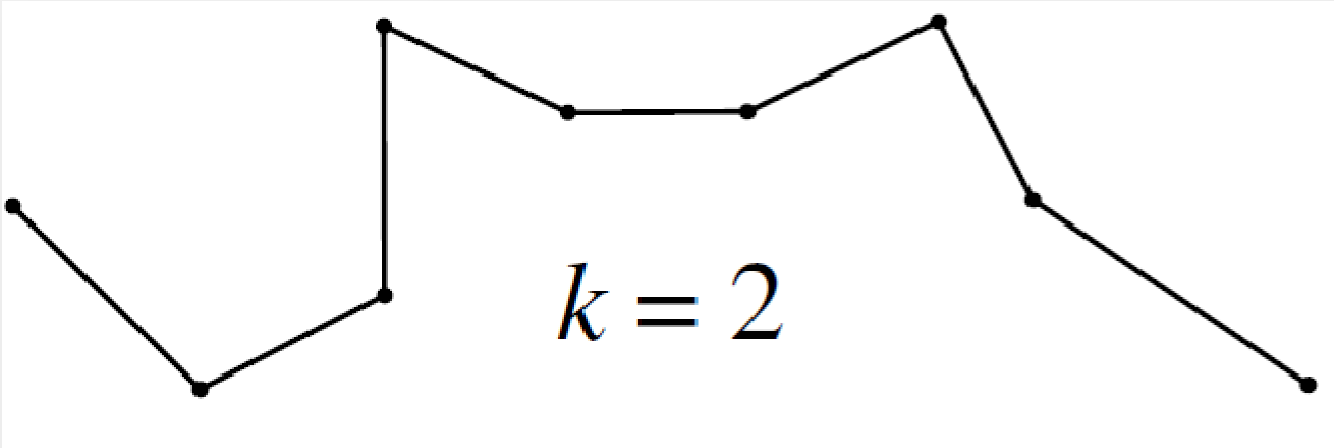
Follows shape of the control polygon

Transform curve – transform control polygon

Everywhere C^{k-2} continuous

B-spline curves – Convex hulls

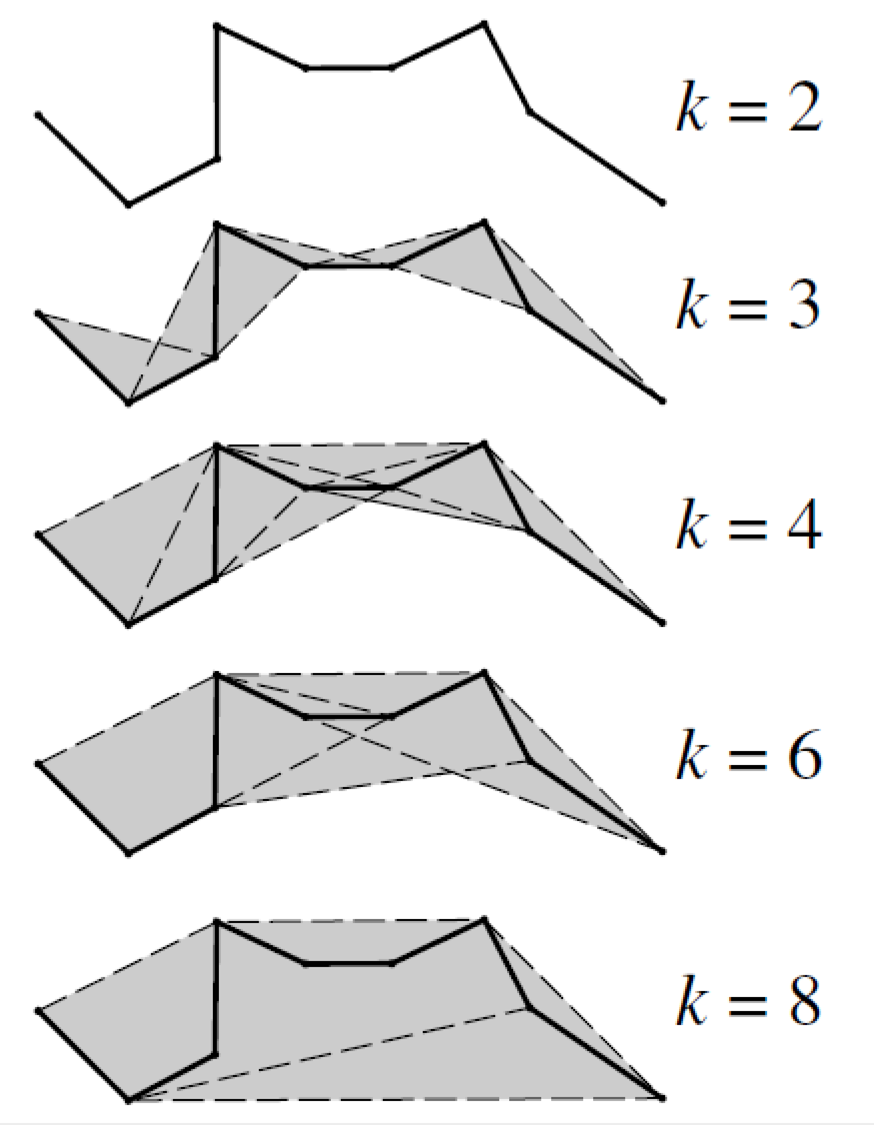
B-spline curves stronger than for Bezier curves.
A point on the curve **P(t)** lies within the convex hull of **k** neighboring control vertices.
Notice for order, **k = 2** the degree is one – **a straight line**. The B-spline curve is the control polygon.



B-spline curves – Convex hulls

For **k = 3** a larger region may contain the curve.
The B-spline curve will not exactly follow polygon.

The higher the order
the less closely the B-spline curve follows the control polygon.



Non-uniform rational B-spline

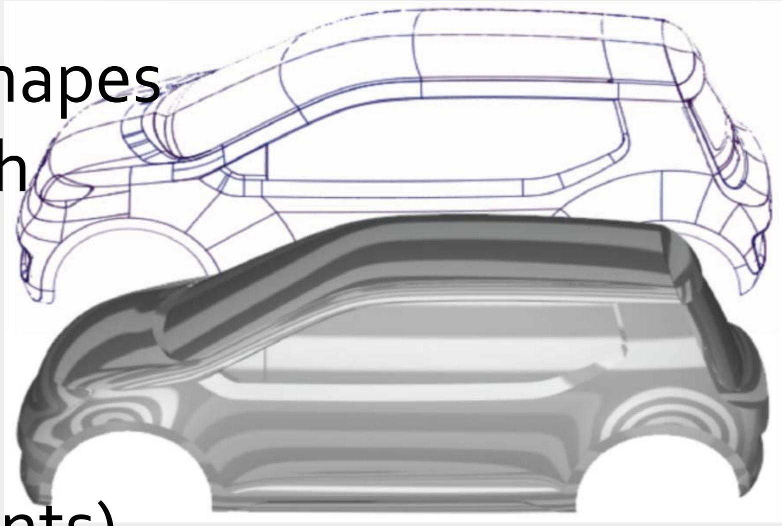
- NURBS is a mathematical model commonly used in computer graphics for generating and representing curves and surfaces. **It offers great flexibility** and precision for handling both analytic and modeled shapes.
- **NURBS** are commonly used in CAD, CAM and CAE and are part of numerous industry standards, such as **IGES**, **STEP**, **ACIS**, and **PHIGS**.

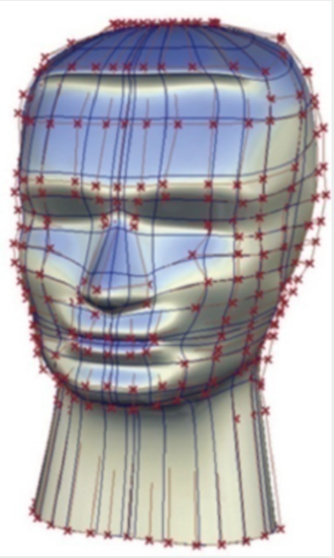
Non-uniform rational B-spline

- Most modern CAD systems use the NURBS curve representation scheme.
- Uniformity deals with the spacing of control points.
- **Rational functions** include a **weighting value** at each control point for effect of control point.
- very popular due to their flexibility in curve generation.
- **NURBS permit definition of surfaces from ratios of polynomials.** (Rational functions permit much better control over the derivatives of curves, hence the surface curvature, than polynomials alone.)

NURBS modelling

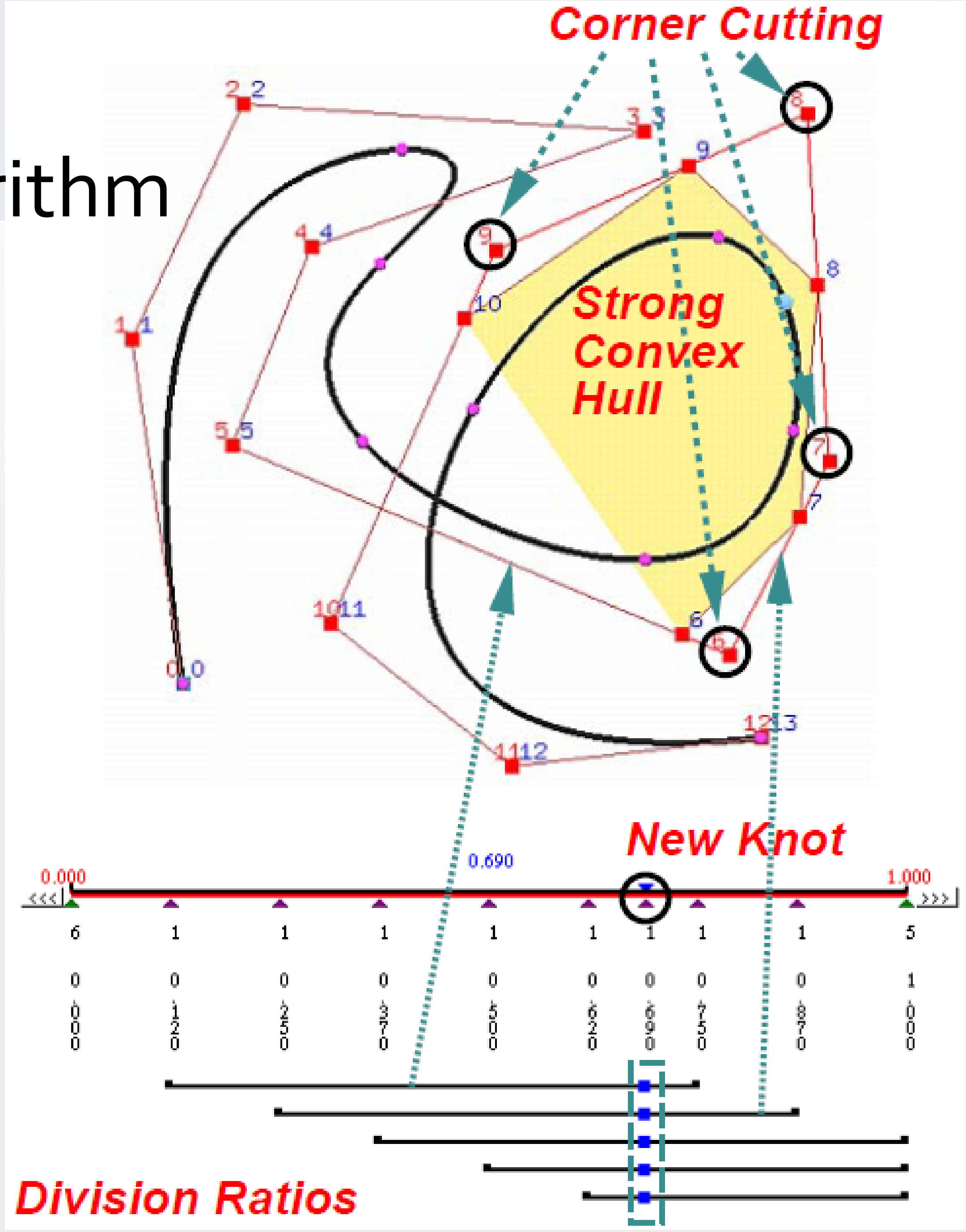
Flexibility to create sculptural shapes
Tension to keep surfaces smooth and taught
Alignment to create smooth, invisible joins
Control Vertices (**CV**, control points), Degree and Spans control the **flexibility** and **shape** of the curve.
the **smoothness** determined by the **Degree** and **Spans** of the NURBS.





De Boor's Algorithm

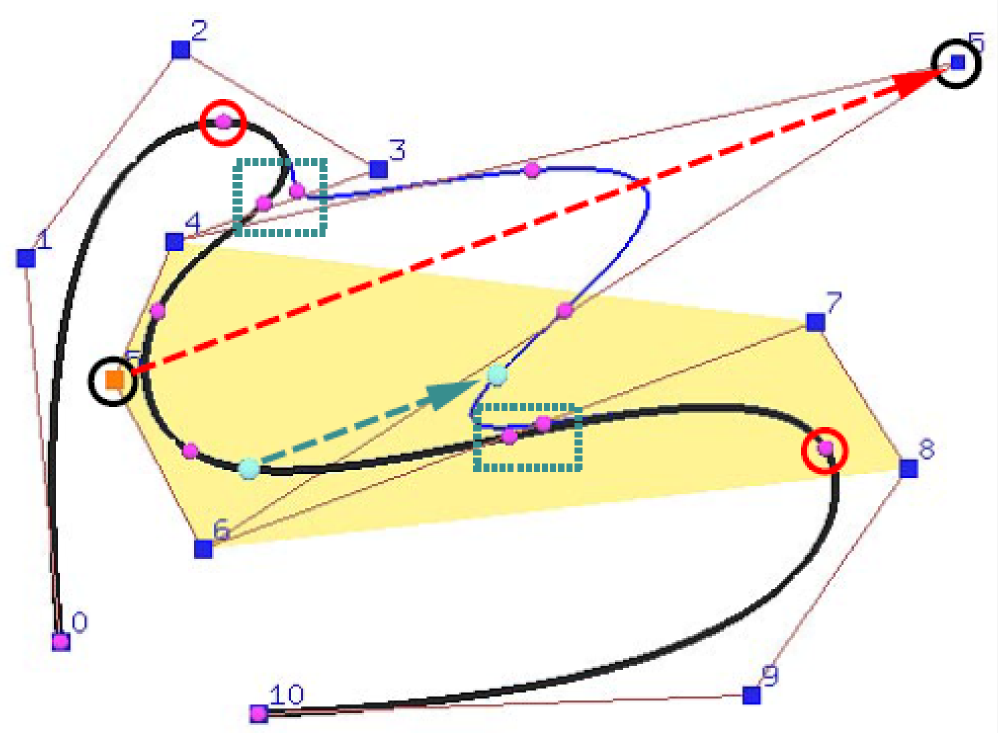
De Boor's algorithm is implemented by repeated knot insertion.



Shape Editing

The local modification property guarantees that only part of the curve will be affected when a control point changes its position.

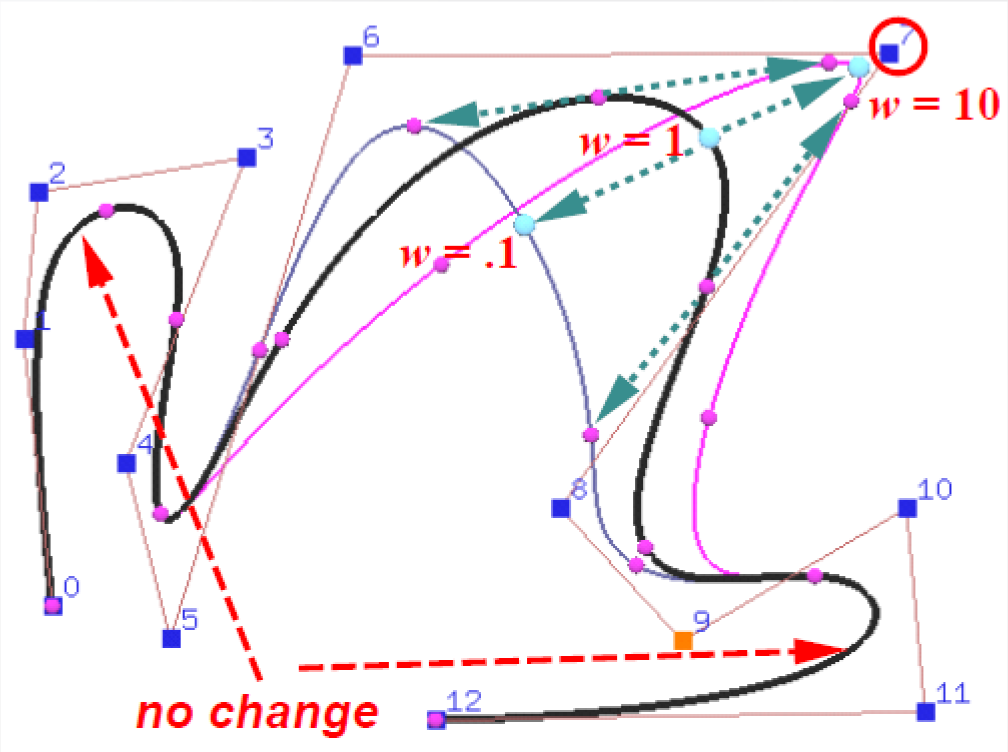
In fact, position change is parallel to the Movement of the control point.



Weight Change

Increasing weight pulls the curve toward the selected control point.

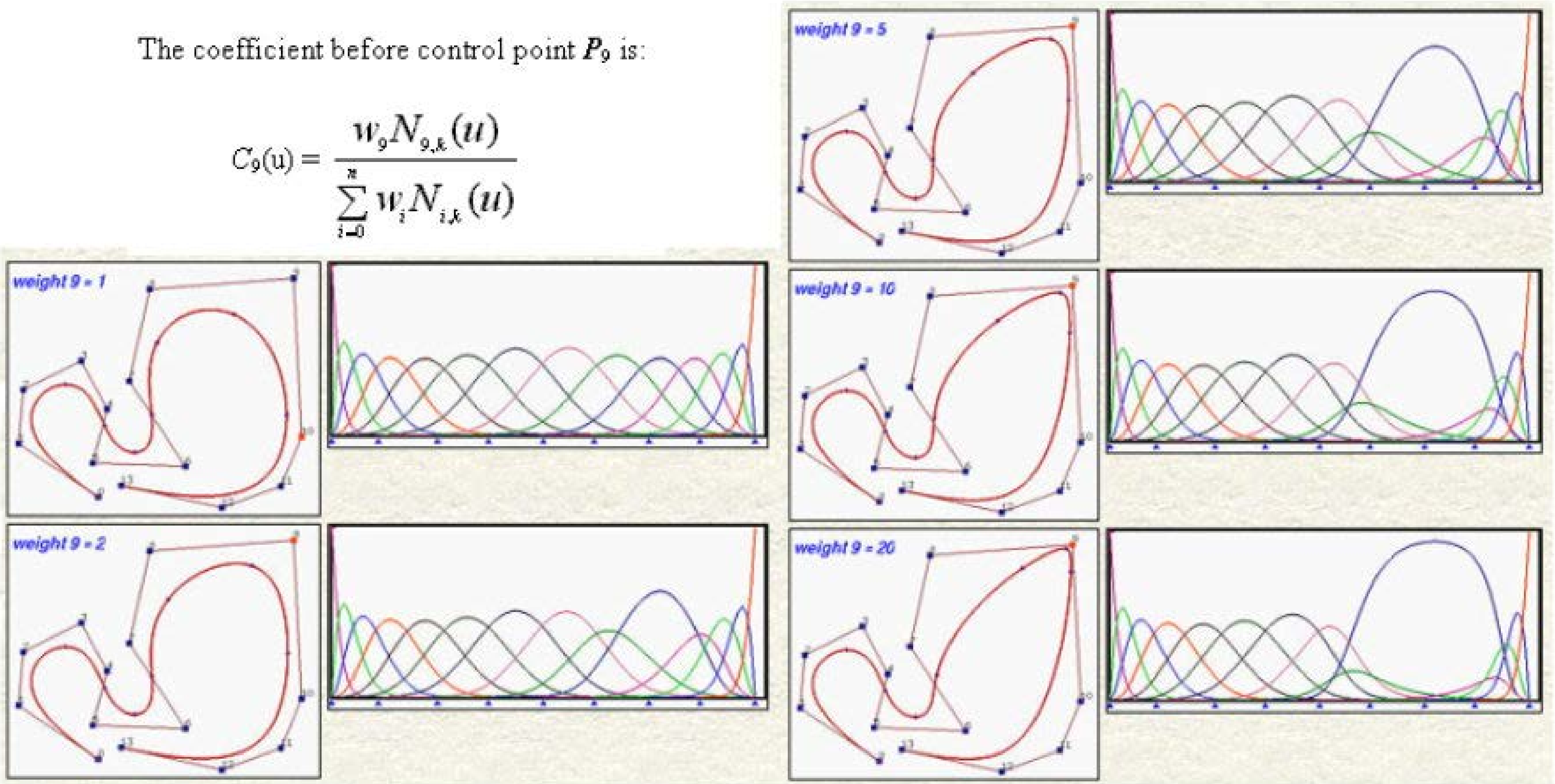
Decreasing weight pushes the curve away from the selected control point.



Effect of Weights

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.

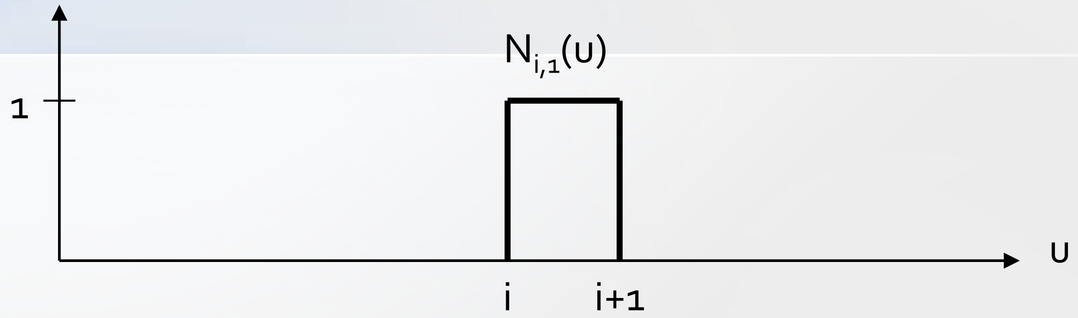
Unit partitioning property of NURBS basis functions

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

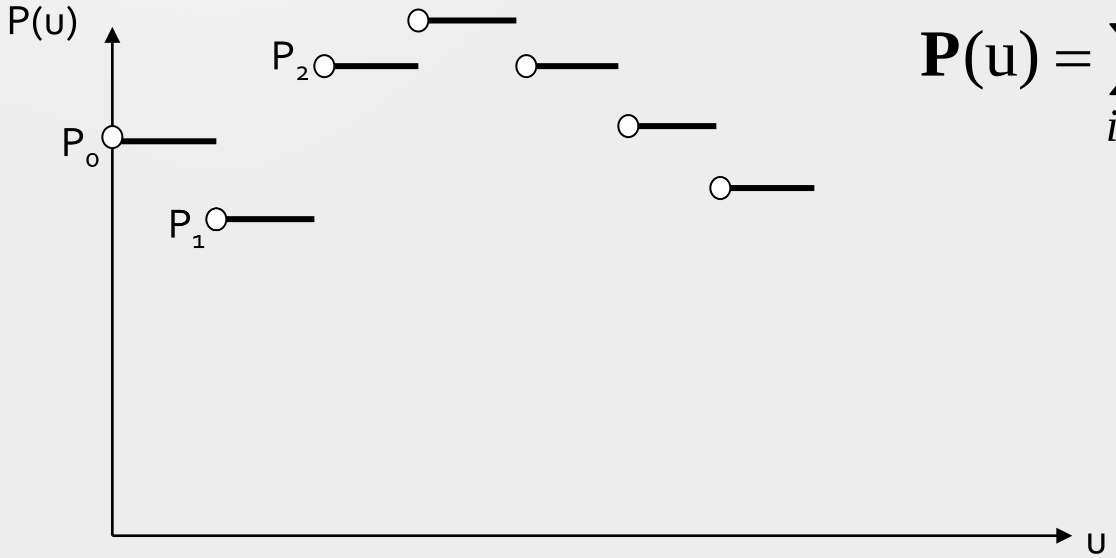
$$= \sum_{i=0}^{n} \left(\frac{w_i N_{i,K}(u)}{\sum_{i=0}^{n} w_i N_{i,K}(u)} \right) \mathbf{p}_i$$

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

Degree 0 (order 1) B-Spline Function $N_{i,1}(u)$

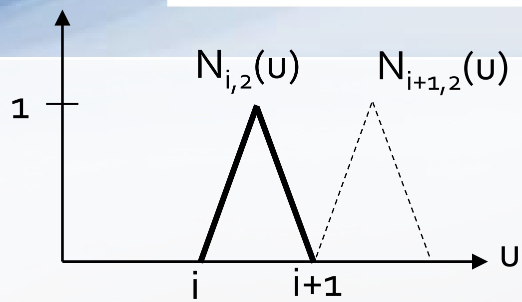


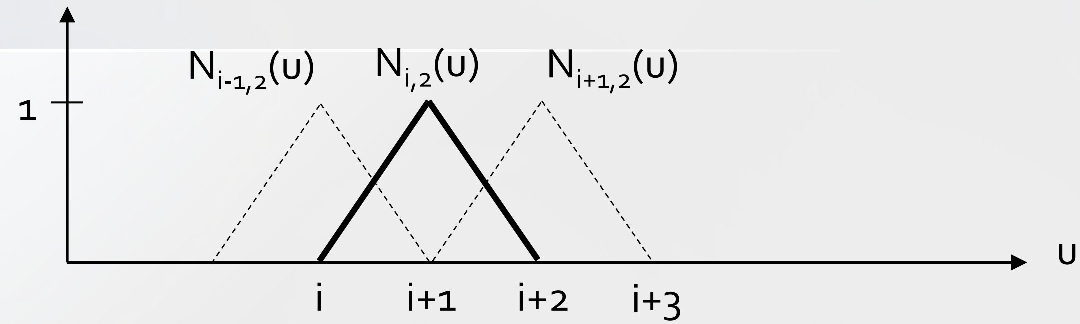
Problem (serious): discontinuity of the curve



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

Degree 1 (order 2) B-Spline Function $N_{i,2}(u)$



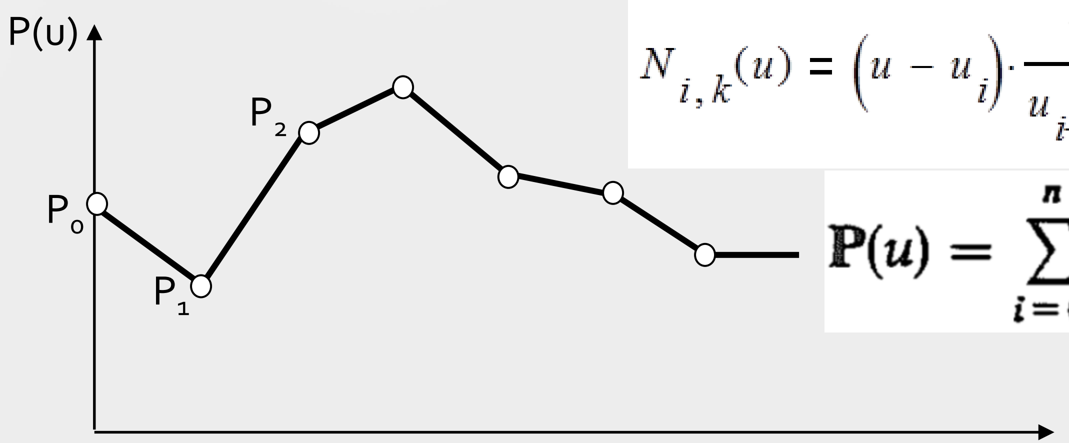


This can't be allowed; it would violate

$$\sum_{i=0}^{n-1} f_i(u) = 1 \;\text{ for any } u \in [0,1]$$

$$N_{i,2}(u) = \begin{cases} u-i & i \le u < i+1 \\ (i+2)-u & i+1 \le u < i+2 \\ 0 & \textit{otherwise} \end{cases} = \frac{u-i}{(i+1)-i} N_{i,1}(u) + \frac{(i+2)-u}{(i+2)-(i+1)} N_{i+1,1}(u)$$

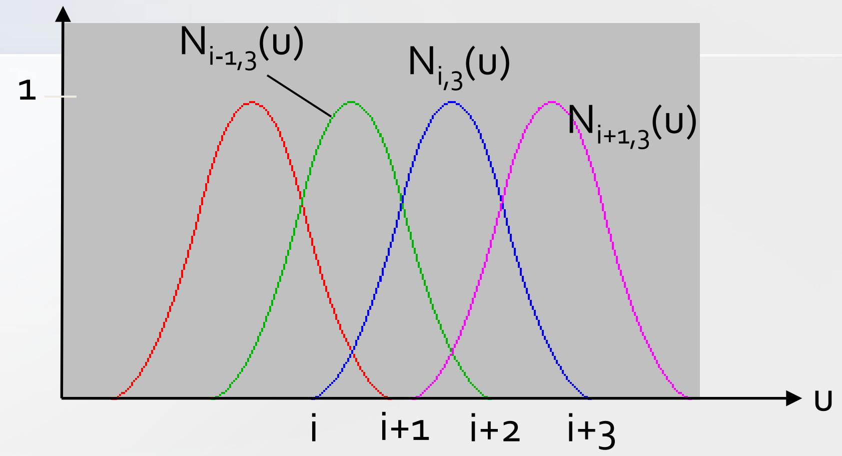
Problem: discontinuity of tangents



$$N_{i,k}(u) = (u - u_i) \cdot \frac{N_{i,k-1}}{u_{i+k-1} - u_i} + (u_{i+k} - u) \cdot \frac{N_{i+1,k-1}}{u_{i+k} - u_{i+1}}$$

$$\mathbf{P}(u) = \sum_{i=0}^{n} \mathbf{P}_i N_{i,k}(u), \qquad 0 \le u \le u_{\max}$$

Degree 2 (order 3) B-Spline Function $N_{i,3}(u)$



$$N_{i,3}(u) = \frac{u-i}{(i+2)-i} N_{i,2}(u) + \frac{(i+3)-u}{(i+3)-(i+1)} N_{i+1,2}(u) = \begin{cases} & i \le u < i+1 \\ & i+1 \le u < i+2 \\ & i+2 \le u \le i+3 \\ & otherwise \end{cases}$$

(What are the u-domains for $N_{i,2}(u)$ and $N_{i+1,2}(u)$?)

Curve now has continuous tangents!

Degree k-1 (order k) B-Spline Function $N_{i,k}(u)$

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

Change partitioning of u-domain from integers to arbitrary real values (so to have more flexibility):

$\{0, 1, 2, 3, \ldots, i, \ldots\} \rightarrow \{t_0, t_1, t_2, t_3, \ldots, t_i, \ldots\}$

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

[Demonstrations](http://www.people.nnov.ru/fractal/Splines/Basis.htm)

(http://www.people.nnov.ru/fractal/Splines/Basis.htm)

Representing Curves and Surfaces

We need smooth curves and surfaces in many applications:

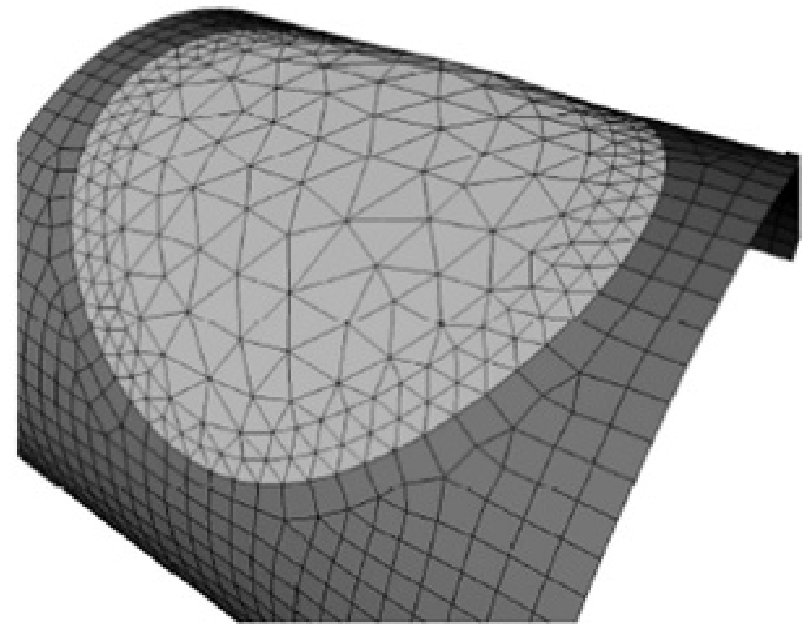
- computer-aided design (CAD)
- model real world objects
- high quality fonts, data plots, artists sketches

Most common representation for surfaces

- polygon mesh
- parametric surfaces
- quadric surfaces

Polygon mesh surfaces

- set of connected planar surfaces bounded by polygons
- good for boxes, cabinets, building exteriors
- bad for curved surfaces
- errors can be made arbitrarily small at the cost of space and execution time
- enlarged images show geometric aliasing



Mesh with gradual adjustment of edge length.

Parametric polynomial surface patches

Parametric bivariate (two-variable) surface patches:

- point on 3D surface = (x(u,v), y(u,v), z(u,v))
- boundaries of the patches are parametric polynomial curves
- many fewer parametric patches than polynomial patches are needed to approximate a curved surface to a given accuracy
- more complex algorithms though

Parametric cubic curves

Polylines and polygons:

- large amounts of data to achieve good accuracy
- interactive manipulation of the data is tedious

Higher-order curves:

- more compact (use less storage)
- easier to manipulate interactively

Possible representations of curves:

- explicit, implicit, and parametric

Parametric cubic curves

Explicit functions: $y = f(x)$, $z = g(x)$

- impossible to get multiple values for a single x
- break curves like circles and ellipses into segments

- not invariant with rotation
- rotation might require further segment breaking

- problem with curves with vertical tangents
- infinite slope is difficult to represent

Parametric cubic curves

Implicit equations: $f(x,y,z) = 0$

- equation may have more solutions than we want
- circle: $x^2 + y^2 = 1$, half circle: ?

- problem to join curve segments together
- difficult to determine if their tangent directions agree at their joint point

Parametric cubic curves

Parametric representation:
$x = x(t)$, $y = y(t)$, $z = z(t)$

- overcomes problems with explicit and implicit forms
- no geometric slopes (which may be infinite)
- parametric tangent vectors instead (never infinite)
- a curve is approximated by a piecewise polynomial curve

Parametric cubic curves

- Why cubic?

- lower-degree polynomials give too little flexibility in controlling the shape of the curve
- higher-degree polynomials can introduce unwanted wiggles and require more computation
- lowest degree that allows specification of endpoints and their derivatives
- lowest degree that is not planar in 3D

Parametric cubic curves

Kinds of continuity:

- G^0: two curve segments join together
- G^1: directions of tangents are equal at the joint
- C^1: directions and magnitudes of tangents are equal at the joint
- C^n: directions and magnitudes of n-th derivative are equal at the joint

Parametric cubic curves

Major types of curves:

- **Hermit**
- defined by two endpoints and two tangent vectors

- **Bezier**
- defined by two endpoints and two other points that control the endpoint tangent vectors

- **Splines**
- several kinds, each defined by four points
- uniform B-splines, non-uniform B-splines, ß-splines

Parametric cubic curves

- General form:

$$x(t) = a_x t^3 + b_x t^2 + c_x t \quad + d_x$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t \quad + d_y$$

$$z(t) = a_z t^3 + b_z t^2 + c_z t \quad + d_z$$

$$C = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix} \qquad T = [t^3 \quad t^2 \quad t \quad 1]$$

$$Q(t) = [x(t) \quad y(t) \quad z(t)] = T \cdot C = T \cdot M \cdot G$$

Parametric cubic curves

- It is not necessary to choose a single representation, since it is possible to convert between them.
- Interactive editors provide several choices, but internally they usually use NURBS, which is the most general.

- Generalization of parametric cubic curves.

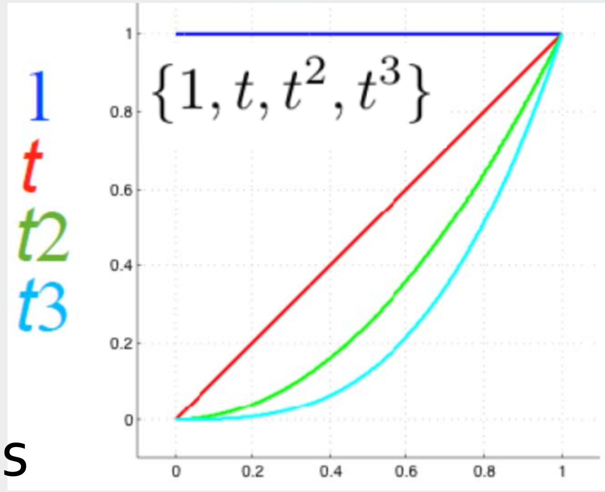
$$Q(s,t) = T \cdot C(t) = T \cdot M \cdot G(t)$$

- For each value of *s* there is a family of curves in *t*.

General Spline Formulation

Q(t) = **GBT(t)** = Geometry **G** . Spline Basis **B** . Power Basis **T(t)**

- Geometry: control points coordinates assembled into a matrix (P1, P2, …, Pn+1)
- Spline matrix: defines the type of spline
- Bernstein for Bézier
- Power basis: the monomials (1, t, …, tn)
- Advantage of general formulation
- Compact expression
- Easy to convert between types of splines
- Dimensionality (plane or space) does not really matter

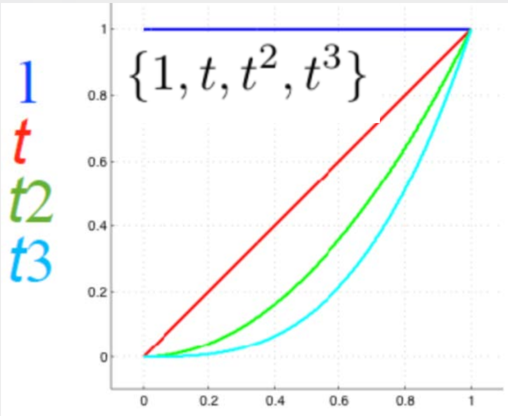


$$y(t) = a_0 + a_1\, t + a_2\, t^2 + a_3\, t^3$$

Polynomials as a Vector Space

- Polynomials $y(t) = a_0 + a_1 t + a_2 t^2 + \ldots + a_n t^n$
- Can be added: just add the coefficients

$$(y+z)(t) = (a_0 + b_0) + (a_1 + b_1)t + (a_2 + b_2)t^2 + \ldots + (a_n + b_n)t^n$$



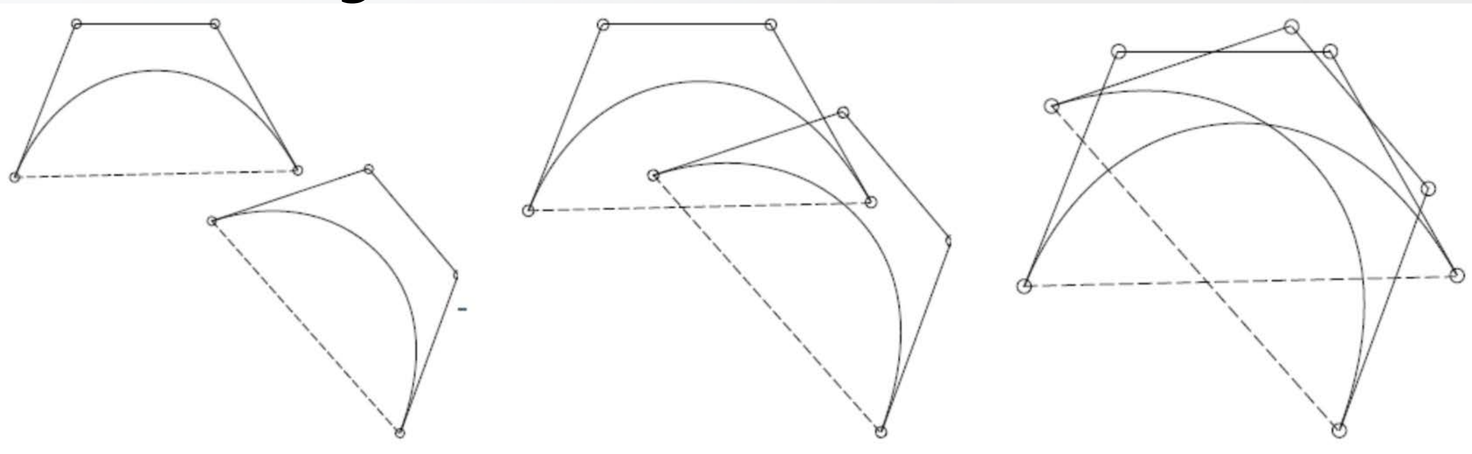
- Can be multiplied by a scalar: multiply the coefficients

$$s \cdot y(t) = (s \cdot a_0) + (s \cdot a_1)t + (s \cdot a_2)t^2 + \ldots + (s \cdot a_n)t^n$$

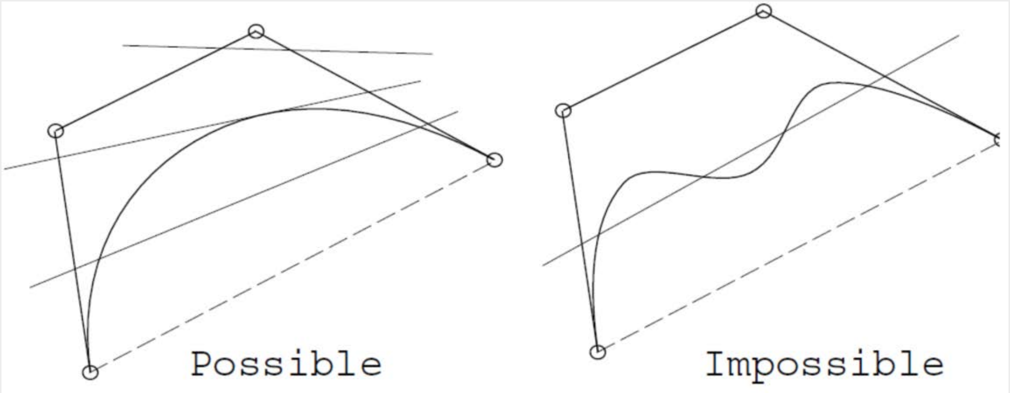
- In the polynomial vector space, {1, t, …, tn} are the basis vectors, a0, a1, …, an are the components

Detecting intersection by convex hulls

Comparison of convex hulls of Bezier curves as means of detecting intersection



Variation diminishing property of a cubic Bezier

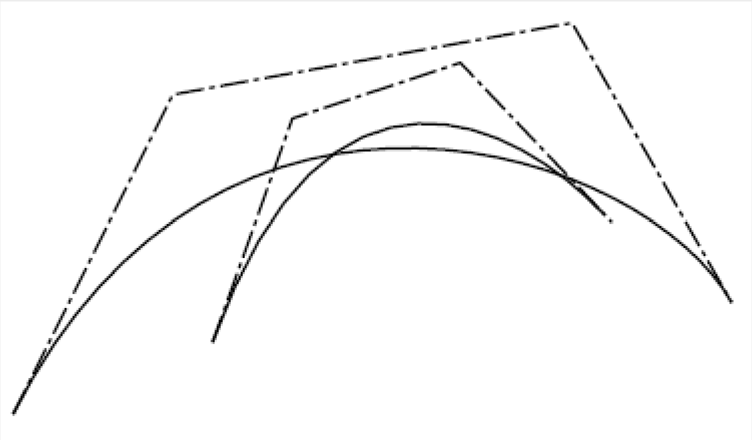


Detecting intersection by polygon

Linear approximation of curves in finding intersections

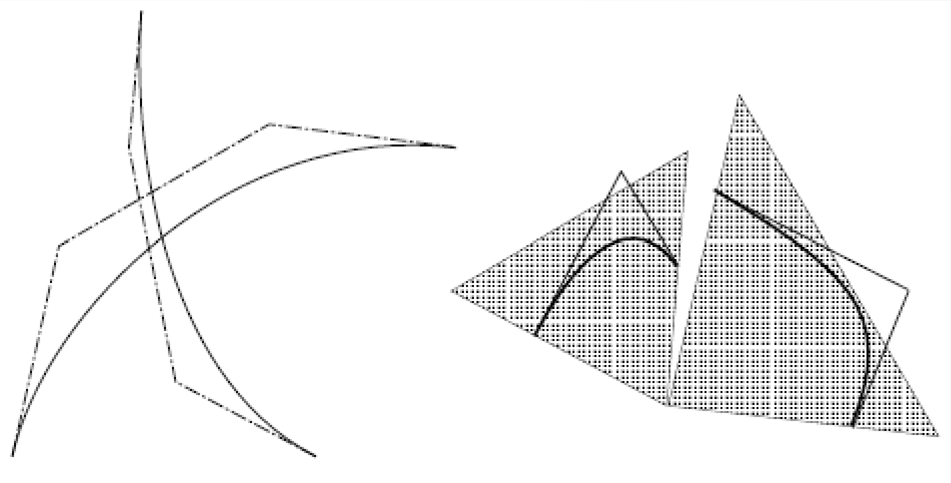
(a) Approximation by linear segments

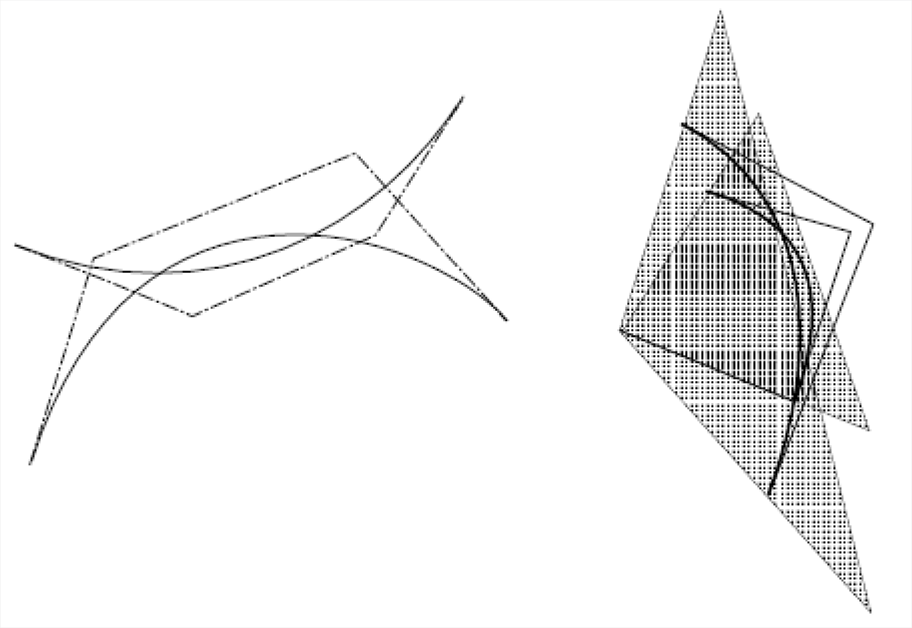
(b) approximation by polygon



Detecting intersection by polygon

Bounding wedges of curves in finding intersections





Bezier Clipping Method

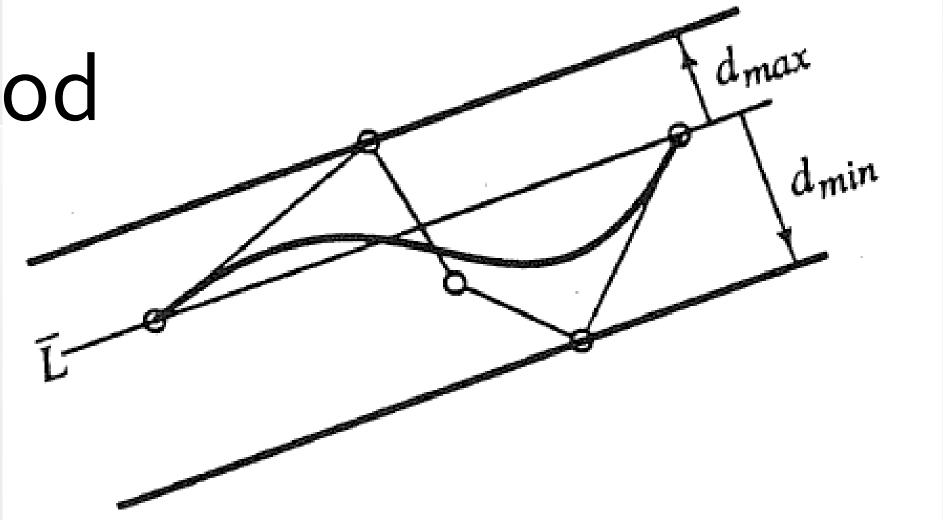
Fat line bounding

a quartic curve

refs.

cad methods - me 554-farouki.pdf

surface intersection algorithms.pdf



If $\bar{L}$ is defined in its normalized implicit equation

$$ax + by + c = 0 \quad (a^2 + b^2 = 1)$$

then, the distance $d(x, y)$ from any point (x, y) to $\bar{L}$ is

$$d(x, y) = ax + by + c$$

$$\{(x, y) | d_{min} \leq d(x, y) \leq d_{max}\}$$

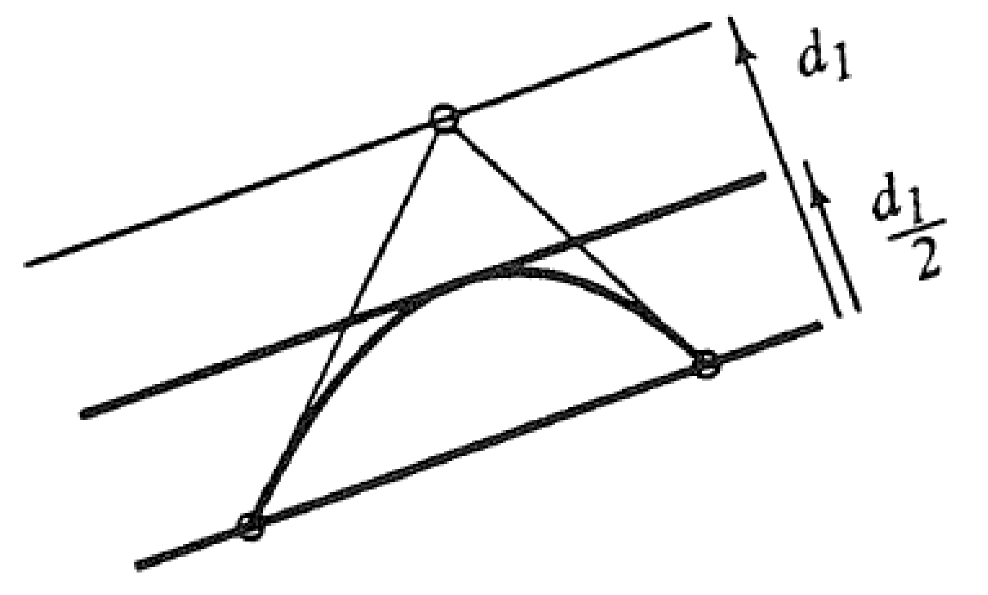
Bezier Clipping Method

Fat line bounding for a polynomial quartic curve

$$d(x,y) = ax + by + c$$

$$d(t) = 2t(1-t)d_1$$

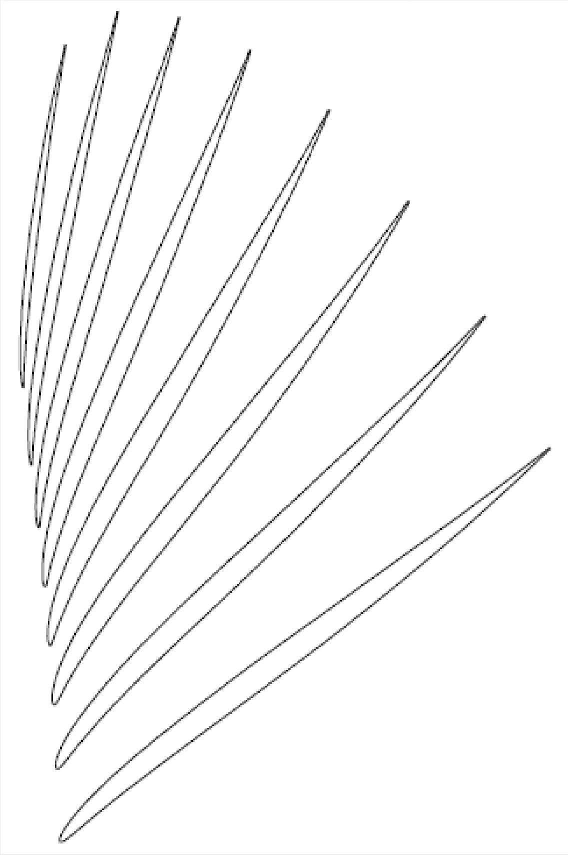
$$d_{min} = \min\{0, \frac{d_1}{2}\}, \quad d_{max} = \max\{0, \frac{d_1}{2}\}.$$



Examples of intersection problems

Contouring of surfaces through intersection with a series of parallel planes or coaxial circular cylinders or cones for visualization

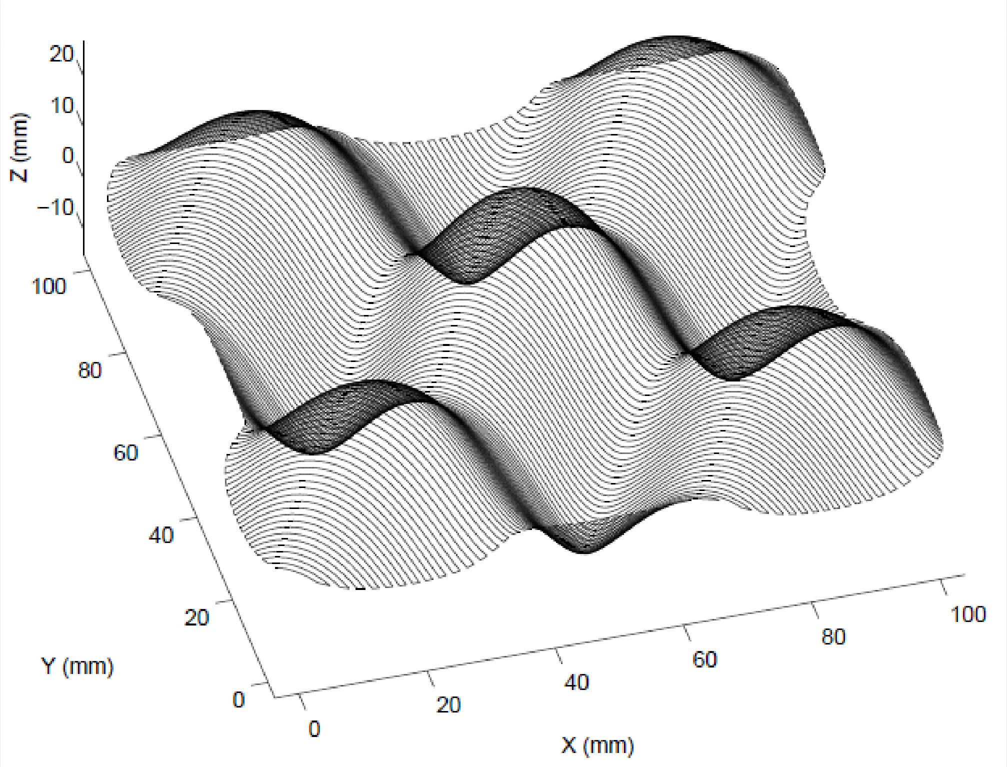
A marine propeller is visualized through intersection with a series of parallel planes



Examples of intersection problems

Numerical control machining (milling) involving intersection of offset surfaces with a series of parallel planes, to create machining paths for ball (spherical) cutters

Offset surface is intersected with a series of parallel planes to generate a tool path for 3-D NC machining

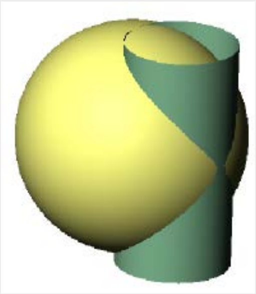


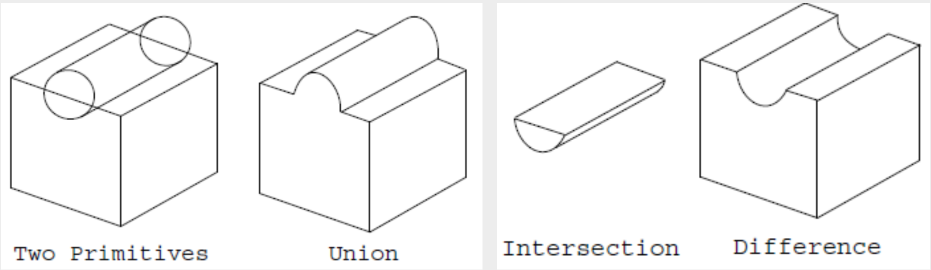
Intersection Problems

Representation of complex geometries in the Boundary Representation (B-rep) scheme; for example, the description of the internal geometry and of structural members of automobiles, airplanes, and ships involves

- Intersections of free-form parametric surfaces with low order algebraic surfaces (planes, quadrics, torii,cyclides);
- Intersections of low order algebraic surfaces;

in a process called boundary evaluation, in which the Boundary Representation is created by evaluating a Constructive Solid Geometry (CSG) model of the object





Intersection Problems

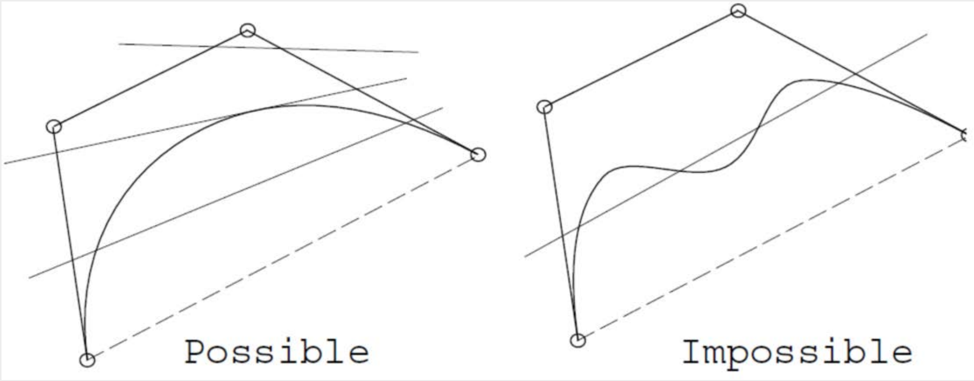
When studying intersection problems, the type of curves and surfaces can be classified as follows:
Rational polynomial parametric (**RPP**)
Procedural parametric (**PP**),
Implicit algebraic (**IA**), Implicit procedural (**IP**)

In order to solve general **surface to surface** (**S/S**) intersection problems, the following auxiliary intersection problems need to be considered:

1. point/point (P/P)
2. point/curve (P/C)
3. point/surface (P/S)
4. curve/curve (C/C)
5. curve/surface (C/S)

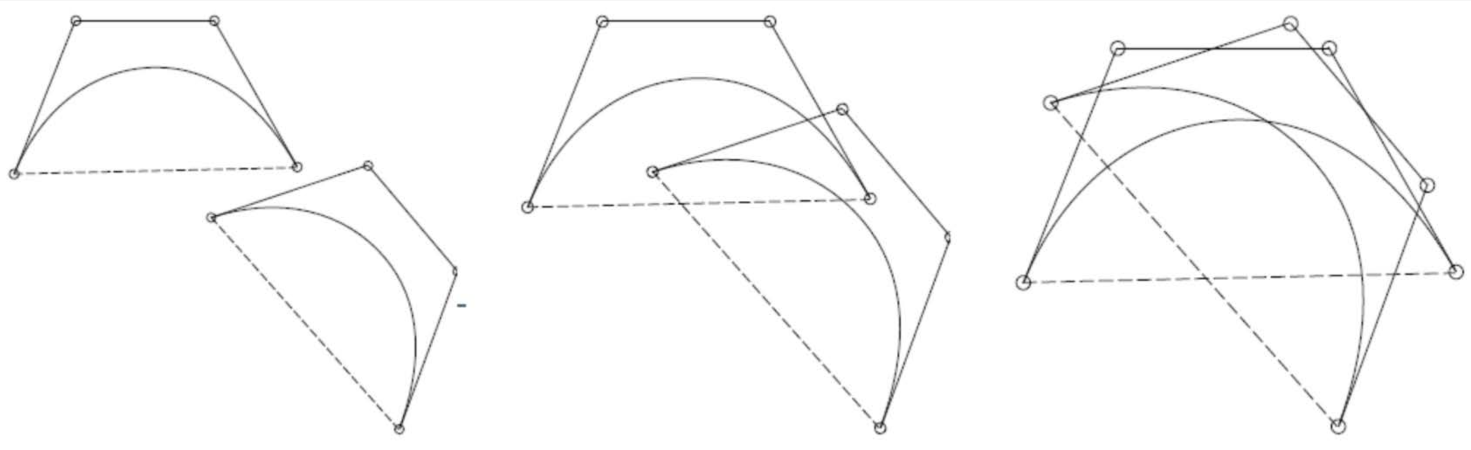
Variation diminishing property

2-D: The number of intersections of a straight line with a planar Bezier curve is no greater than the number of intersections of the line with the control polygon.
A **line** intersecting the convex hull of a planar Bezier curve may **intersect the curve transversally**, be **tangent** to the curve, or **not intersect** the curve at all. **It may not**, however, **intersect the curve more times** than it intersects the control polygon.
3-D: The same relation holds true for a plane with a space Bezier curve.



Variation diminishing property

From this property, we can roughly say that a Bezier curve oscillates less than its control polygon,
or in other words, the control polygon's segments exaggerate the oscillation of the curve.
This property is important in intersection algorithms and in detecting the fairness of Bezier curves.

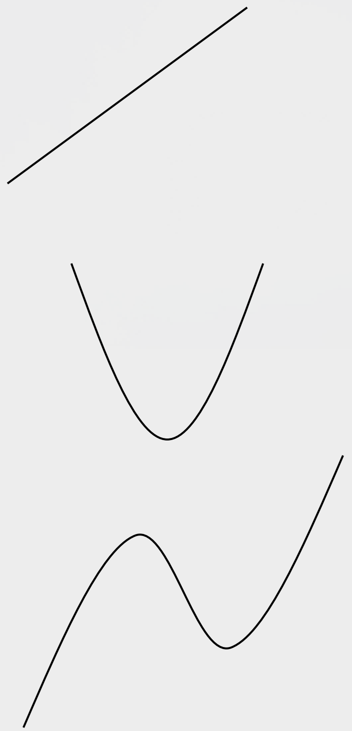


Cubic Curves

CSE167: Computer Graphics
Instructor: Steve Rotenberg
UCSD, Fall 2006

Cubic Curves
Polynomial Functions

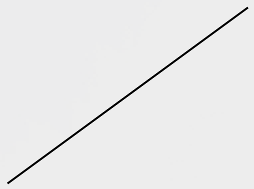
- Linear: $f(t) = at + b$
- Quadratic: $f(t) = at^2 + bt + c$
- Cubic: $f(t) = at^3 + bt^2 + ct + d$



Ref.
CSE167: Computer Graphics
Instructor: Steve Rotenberg,UCSD, Fall 2006

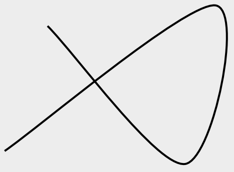
Vector Polynomials (Curves)

- Linear: $\mathbf{f}(t) = \mathbf{a}t + \mathbf{b}$



- Quadratic: $\mathbf{f}(t) = \mathbf{a}t^2 + \mathbf{b}t + \mathbf{c}$

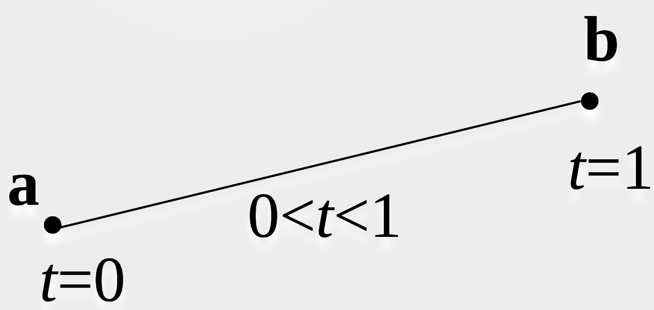
- Cubic: $\mathbf{f}(t) = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d}$



We usually define the curve for $0 \le t \le 1$

Linear Interpolation

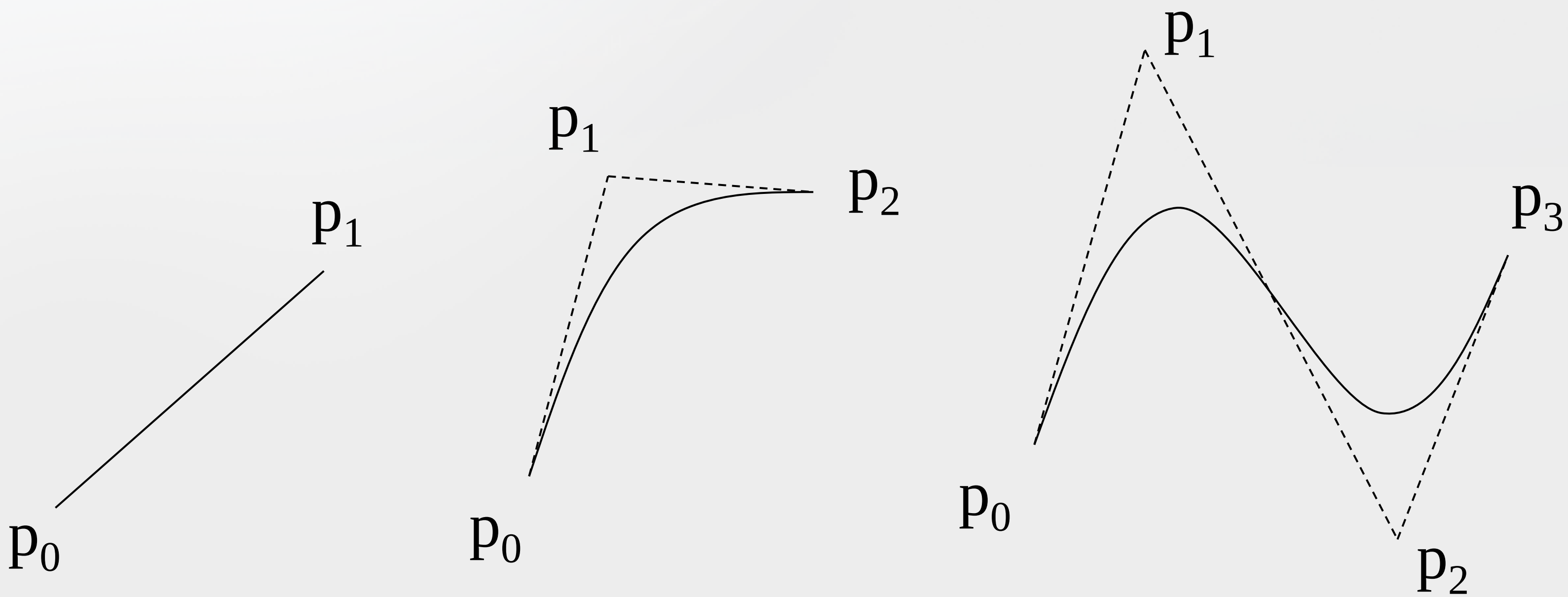
- Linear interpolation (Lerp) is a common technique for generating a new value that is somewhere in between two other values
- A 'value' could be a number, vector, color, or even something more complex like an entire 3D object…
- Consider interpolating between two points **a** and **b** by some parameter *t*



$$Lerp(t, \mathbf{a}, \mathbf{b}) = (1 - t)\mathbf{a} + t\mathbf{b}$$

Bezier Curves

- Bezier curves can be thought of as a higher order extension of linear interpolation



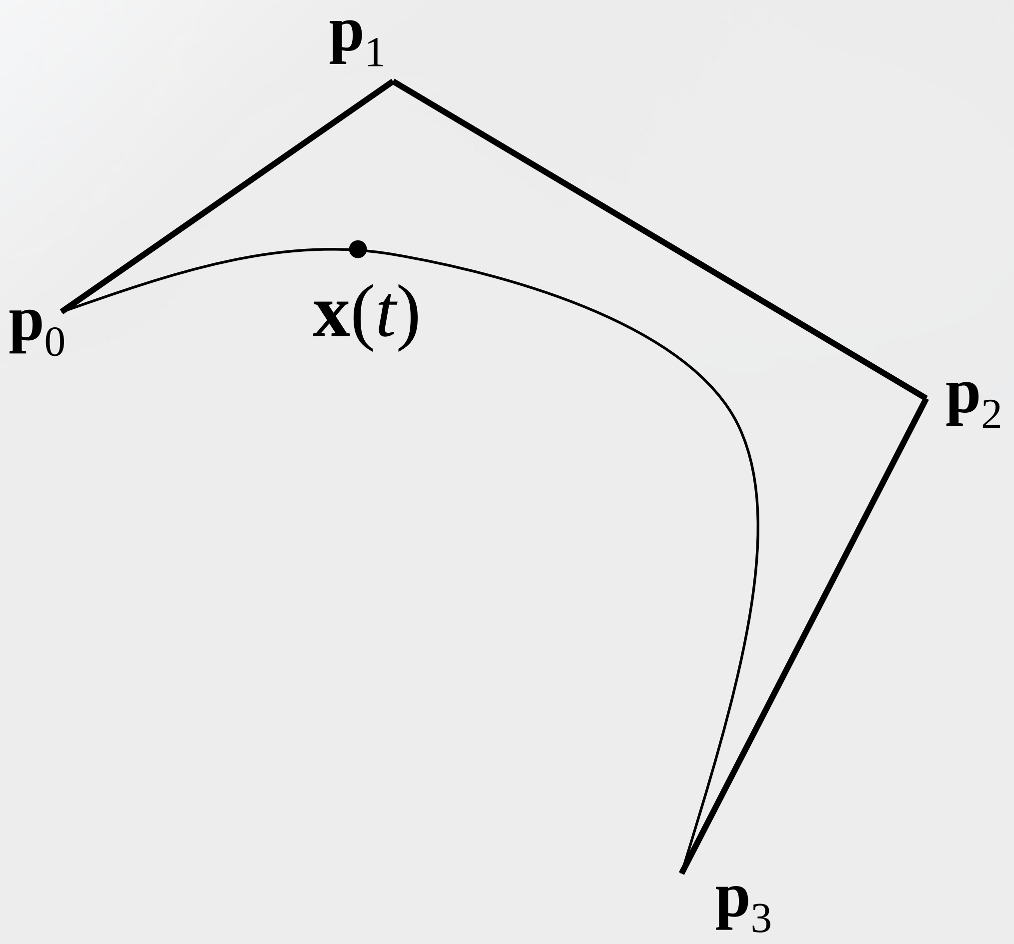
Linear Quadratic Cubic

Bezier Curve Formulation

- There are lots of ways to formulate Bezier curves mathematically. Some of these include:
  - de Casteljau (recursive linear interpolations)
  - Bernstein polynomials (functions that define the influence of each control point as a function of t)
  - Cubic equations (general cubic equation of t)
  - Matrix form
- We will briefly examine each of these

Bezier Curve

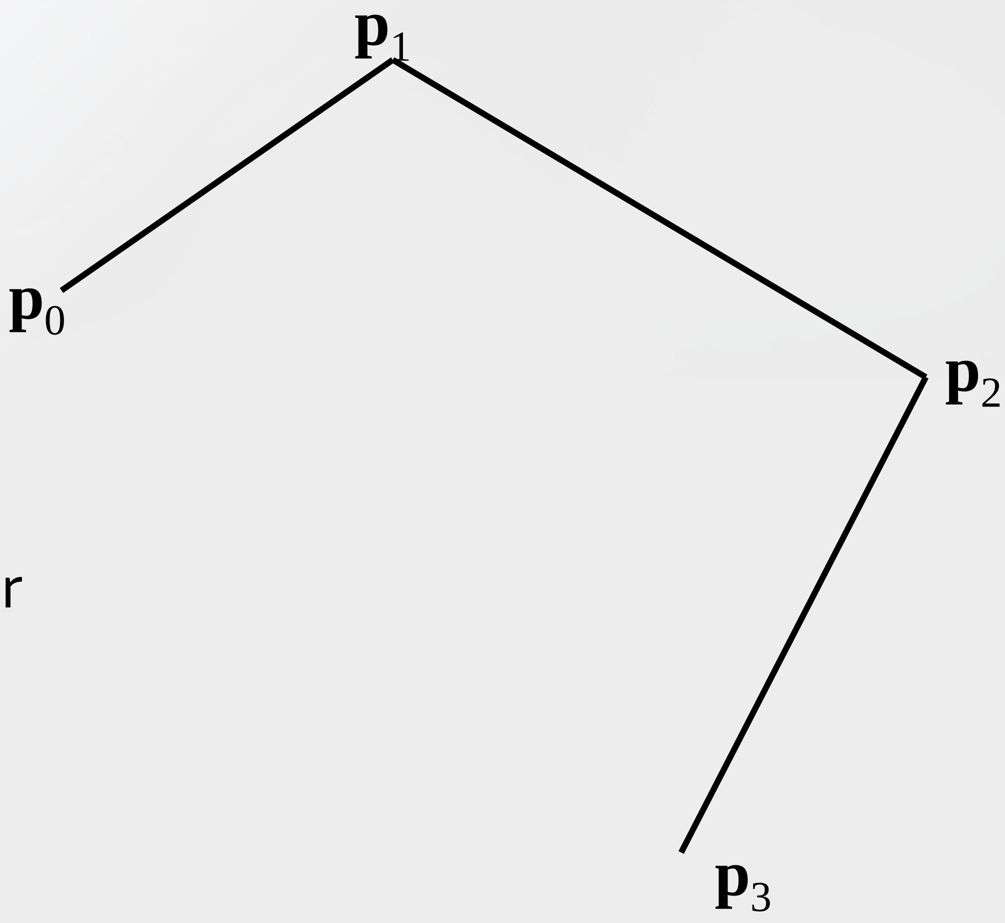
- Find the point **x** on the curve as a function of parameter *t*:



de Casteljau Algorithm

- The de Casteljau algorithm describes the curve as a recursive series of linear interpolations
- This form is useful for providing an intuitive understanding of the geometry involved, but it is not the most efficient form

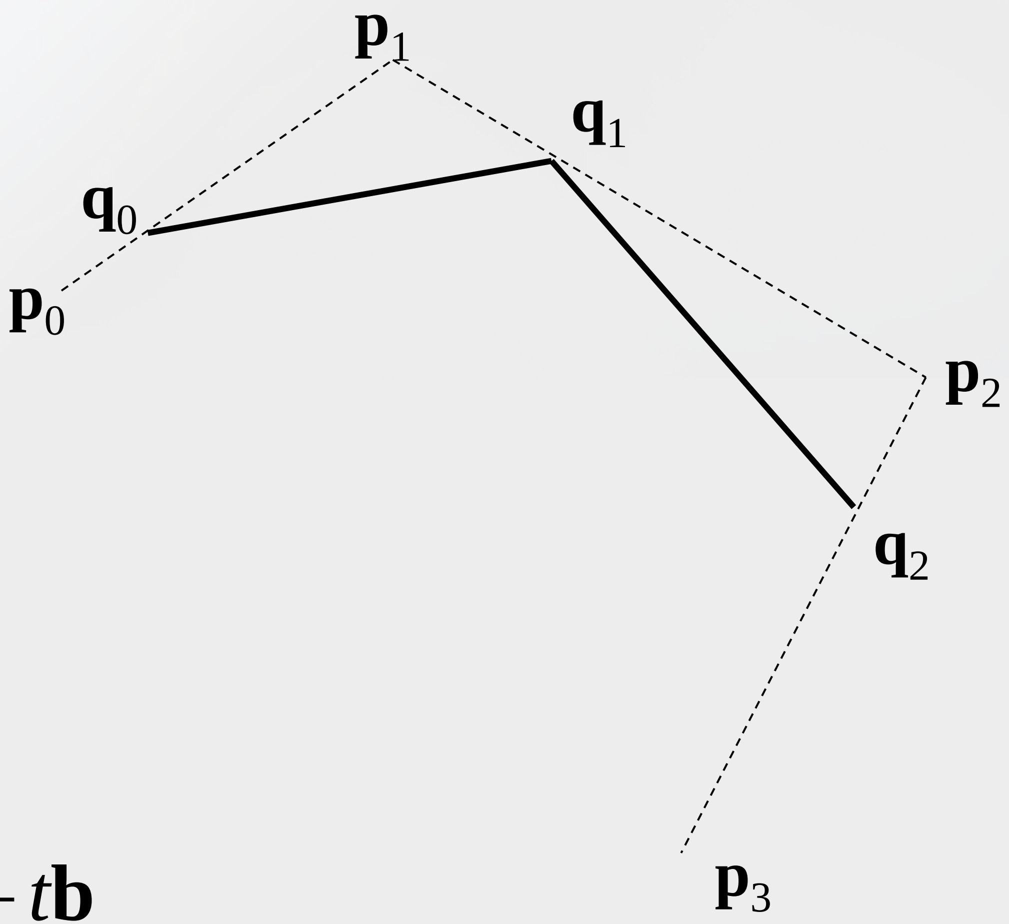
de Casteljau Algorithm



- We start with our original set of points
- In the case of a cubic Bezier curve, we start with four points

de Casteljau Algorithm

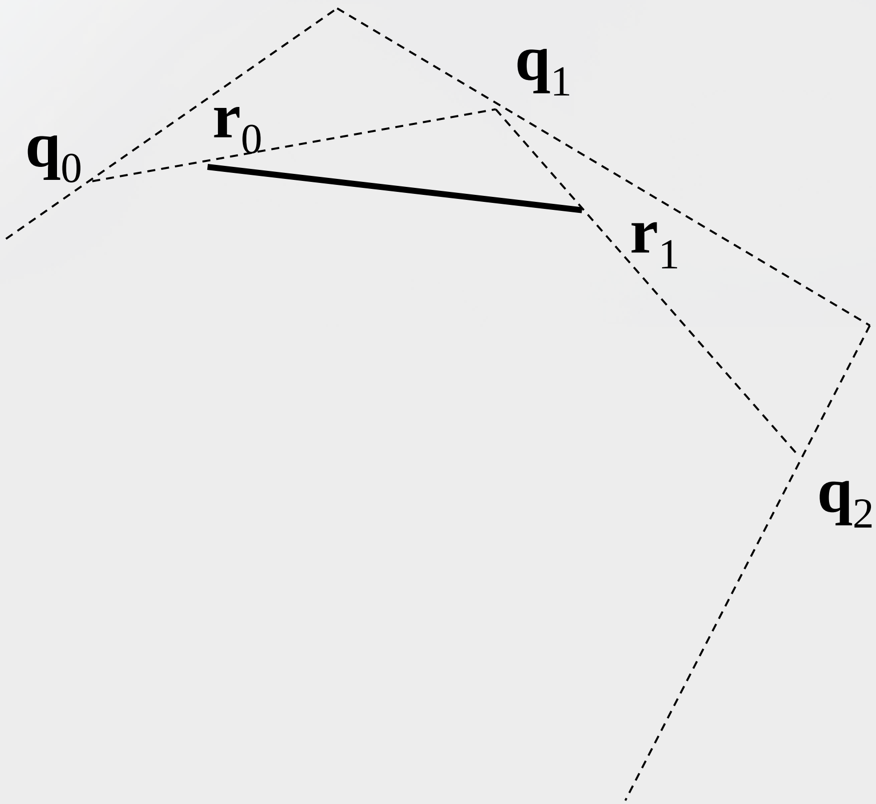
$$\mathbf{q}_0 = Lerp(t, \mathbf{p}_0, \mathbf{p}_1)$$
$$\mathbf{q}_1 = Lerp(t, \mathbf{p}_1, \mathbf{p}_2)$$
$$\mathbf{q}_2 = Lerp(t, \mathbf{p}_2, \mathbf{p}_3)$$



$$Lerp(t, \mathbf{a}, \mathbf{b}) = (1 - t)\mathbf{a} + t\mathbf{b}$$

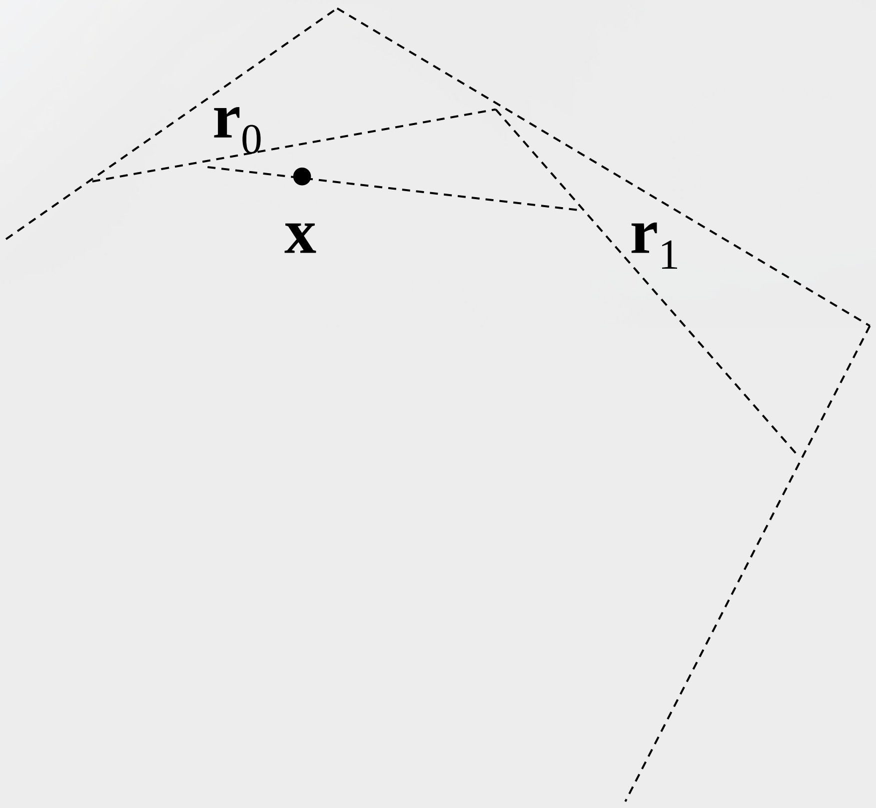
de Casteljau Algorithm

$$\mathbf{r}_0 = Lerp(t, \mathbf{q}_0, \mathbf{q}_1)$$
$$\mathbf{r}_1 = Lerp(t, \mathbf{q}_1, \mathbf{q}_2)$$

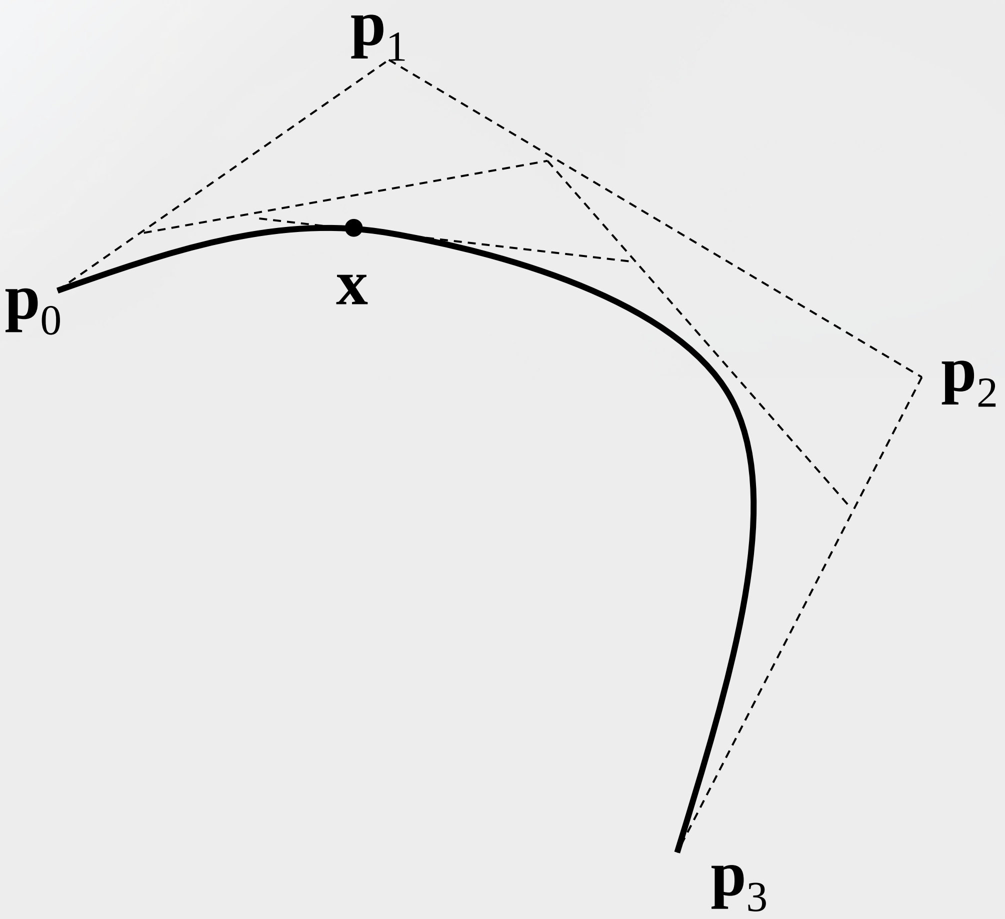


de Casteljau Algorithm

$$\mathbf{x} = Lerp\left(t, \mathbf{r}_0, \mathbf{r}_1\right)$$



Bezier Curve



Recursive Linear Interpolation

$$\mathbf{x} = Lerp(t, \mathbf{r}_0, \mathbf{r}_1)$$

$$\mathbf{r}_0 = Lerp(t, \mathbf{q}_0, \mathbf{q}_1)$$
$$\mathbf{r}_1 = Lerp(t, \mathbf{q}_1, \mathbf{q}_2)$$

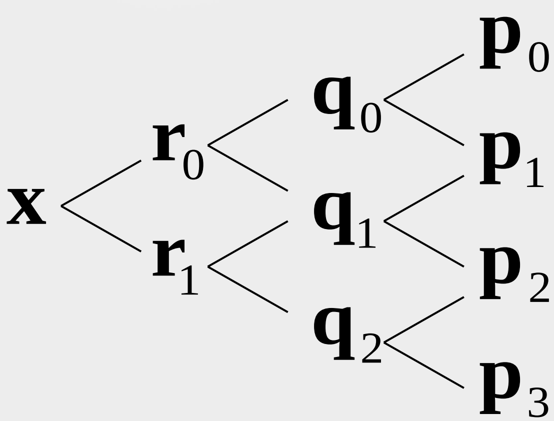
$$\mathbf{q}_0 = Lerp(t, \mathbf{p}_0, \mathbf{p}_1)$$
$$\mathbf{q}_1 = Lerp(t, \mathbf{p}_1, \mathbf{p}_2)$$
$$\mathbf{q}_2 = Lerp(t, \mathbf{p}_2, \mathbf{p}_3)$$

$\mathbf{p}_0$

$\mathbf{p}_1$

$\mathbf{p}_2$

$\mathbf{p}_3$



Expanding the Lerps

$$\mathbf{q}_0 = Lerp(t, \mathbf{p}_0, \mathbf{p}_1) = (1-t)\mathbf{p}_0 + t\mathbf{p}_1$$
$$\mathbf{q}_1 = Lerp(t, \mathbf{p}_1, \mathbf{p}_2) = (1-t)\mathbf{p}_1 + t\mathbf{p}_2$$
$$\mathbf{q}_2 = Lerp(t, \mathbf{p}_2, \mathbf{p}_3) = (1-t)\mathbf{p}_2 + t\mathbf{p}_3$$

$$\mathbf{r}_0 = Lerp(t, \mathbf{q}_0, \mathbf{q}_1) = (1-t)((1-t)\mathbf{p}_0 + t\mathbf{p}_1) + t((1-t)\mathbf{p}_1 + t\mathbf{p}_2)$$
$$\mathbf{r}_1 = Lerp(t, \mathbf{q}_1, \mathbf{q}_2) = (1-t)((1-t)\mathbf{p}_1 + t\mathbf{p}_2) + t((1-t)\mathbf{p}_2 + t\mathbf{p}_3)$$

$$\mathbf{x} = Lerp(t, \mathbf{r}_0, \mathbf{r}_1) = (1-t)((1-t)((1-t)\mathbf{p}_0 + t\mathbf{p}_1) + t((1-t)\mathbf{p}_1 + t\mathbf{p}_2))$$
$$+ t((1-t)((1-t)\mathbf{p}_1 + t\mathbf{p}_2) + t((1-t)\mathbf{p}_2 + t\mathbf{p}_3))$$

Bernstein Polynomial Form

$$\mathbf{x} = (1-t)\Big((1-t)\big((1-t)\mathbf{p}_0 + t\mathbf{p}_1\big) + t\big((1-t)\mathbf{p}_1 + t\mathbf{p}_2\big)\Big)$$
$$+ t\Big((1-t)\big((1-t)\mathbf{p}_1 + t\mathbf{p}_2\big) + t\big((1-t)\mathbf{p}_2 + t\mathbf{p}_3\big)\Big)$$

$$\mathbf{x} = (1-t)^3\mathbf{p}_0 + 3(1-t)^2 t\mathbf{p}_1 + 3(1-t)t^2\mathbf{p}_2 + t^3\mathbf{p}_3$$

$$\mathbf{x} = \left(-t^3 + 3t^2 - 3t + 1\right)\mathbf{p}_0 + \left(3t^3 - 6t^2 + 3t\right)\mathbf{p}_1$$
$$+ \left(-3t^3 + 3t^2\right)\mathbf{p}_2 + \left(t^3\right)\mathbf{p}_3$$

Cubic Bernstein Polynomials

$$\mathbf{x}(t) = \left(-t^3 + 3t^2 - 3t + 1\right)\mathbf{p}_0 + \left(3t^3 - 6t^2 + 3t\right)\mathbf{p}_1 + \left(3t^3 - 6t^2 + 3t\right)\mathbf{p}_2 + \left(t^3\right)\mathbf{p}_3$$

$$\mathbf{x}(t) = \sum B_i^3(t)\mathbf{p}_i$$

$$\mathbf{x} = B_0^3(t)\mathbf{p}_0 + B_1^3(t)\mathbf{p}_1 + B_2^3(t)\mathbf{p}_2 + B_3^3(t)\mathbf{p}_3$$

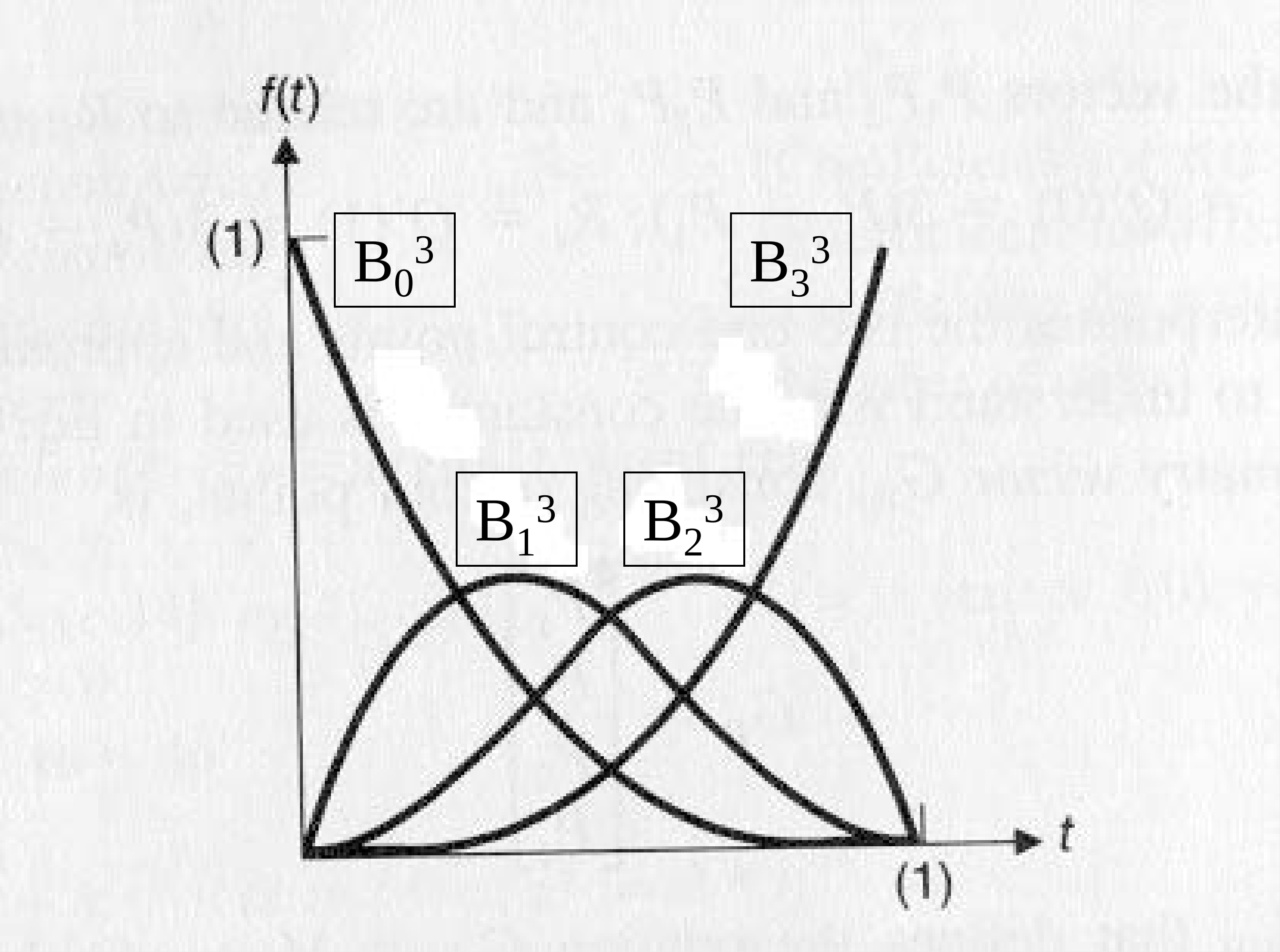
$$B_0^3(t) = -t^3 + 3t^2 - 3t + 1$$

$$B_1^3(t) = 3t^3 - 6t^2 + 3t$$

$$B_2^3(t) = -3t^3 + 3t^2$$

$$B_3^3(t) = t^3$$

Bernstein Polynomials



Bernstein Polynomials

$$B_0^3(t) = -t^3 + 3t^2 - 3t + 1$$

$$B_1^3(t) = 3t^3 - 6t^2 + 3t$$

$$B_2^3(t) = -3t^3 + 3t^2$$

$$B_3^3(t) = t^3$$

$$B_0^2(t) = t^2 - 2t + 1$$

$$B_1^2(t) = -2t^2 + 2t$$

$$B_2^2(t) = t^2$$

$$B_0^1(t) = -t + 1$$

$$B_1^1(t) = t$$

$$B_i^n(t) = \binom{n}{i}(1-t)^{n-i}(t)^i$$

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

$$\sum B_i^n(t) = 1$$

Bernstein Polynomials

- Bernstein polynomial form of a Bezier curve:

$$B_i^n(t) = \binom{n}{i}(1-t)^{n-i}(t)^i$$

$$\mathbf{x}(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{p}_i$$

Bernstein Polynomials

- We start with the de Casteljau algorithm, expand out the math, and group it into polynomial functions of t multiplied by points in the control mesh
- The generalization of this gives us the Bernstein form of the Bezier curve
- This gives us further understanding of what is happening in the curve:
  - We can see the influence of each point in the control mesh as a function of t
  - We see that the basis functions add up to 1, indicating that the Bezier curve is a convex average of the control points

Cubic Equation Form

$$\mathbf{x} = \left(-t^3 + 3t^2 - 3t + 1\right)\mathbf{p}_0 + \left(3t^3 - 6t^2 + 3t\right)\mathbf{p}_1 + \left(-3t^3 + 3t^2\right)\mathbf{p}_2 + \left(t^3\right)\mathbf{p}_3$$

$$\mathbf{x} = \left(-\mathbf{p}_0 + 3\mathbf{p}_1 - 3\mathbf{p}_2 + \mathbf{p}_3\right)t^3 + \left(3\mathbf{p}_0 - 6\mathbf{p}_1 + 3\mathbf{p}_2\right)t^2 + \left(-3\mathbf{p}_0 + 3\mathbf{p}_1\right)t + \left(\mathbf{p}_0\right)1$$

Cubic Equation Form

$$\mathbf{x} = \left(-\mathbf{p}_0 + 3\mathbf{p}_1 - 3\mathbf{p}_2 + \mathbf{p}_3\right)t^3 + \left(3\mathbf{p}_0 - 6\mathbf{p}_1 + 3\mathbf{p}_2\right)t^2$$
$$+ \left(-3\mathbf{p}_0 + 3\mathbf{p}_1\right)t + \left(\mathbf{p}_0\right)1$$

$$\mathbf{x} = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d}$$

$$\mathbf{a} = \left(-\mathbf{p}_0 + 3\mathbf{p}_1 - 3\mathbf{p}_2 + \mathbf{p}_3\right)$$
$$\mathbf{b} = \left(3\mathbf{p}_0 - 6\mathbf{p}_1 + 3\mathbf{p}_2\right)$$
$$\mathbf{c} = \left(-3\mathbf{p}_0 + 3\mathbf{p}_1\right)$$
$$\mathbf{d} = \left(\mathbf{p}_0\right)$$

Cubic Equation Form

- If we regroup the equation by terms of exponents of *t*, we get it in the standard cubic form
- This form is very good for fast evaluation, as all of the constant terms (**a**,**b**,**c**,**d**) can be precomputed
- The cubic equation form obscures the input geometry, but there is a one-to-one mapping between the two and so the geometry can always be extracted out of the cubic coefficients

Cubic Matrix Form

$$\mathbf{x} = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d}$$

$$\mathbf{a} = \left(-\mathbf{p}_0 + 3\mathbf{p}_1 - 3\mathbf{p}_2 + \mathbf{p}_3\right)$$
$$\mathbf{b} = \left(3\mathbf{p}_0 - 6\mathbf{p}_1 + 3\mathbf{p}_2\right)$$
$$\mathbf{c} = \left(-3\mathbf{p}_0 + 3\mathbf{p}_1\right)$$
$$\mathbf{d} = \left(\mathbf{p}_0\right)$$

$$\mathbf{x} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

Cubic Matrix Form

$$\mathbf{x} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} p_{0x} & p_{0y} & p_{0z} \\ p_{1x} & p_{1y} & p_{1z} \\ p_{2x} & p_{2y} & p_{2z} \\ p_{3x} & p_{3y} & p_{3z} \end{bmatrix}$$

Matrix Form

$$\mathbf{x} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} p_{0x} & p_{0y} & p_{0z} \\ p_{1x} & p_{1y} & p_{1z} \\ p_{2x} & p_{2y} & p_{2z} \\ p_{3x} & p_{3y} & p_{3z} \end{bmatrix}$$

$$\boxed{\begin{aligned} \mathbf{x} &= \mathbf{t} \cdot \mathbf{B}_{Bez} \cdot \mathbf{G}_{Bez} \\ \mathbf{x} &= \mathbf{t} \cdot \mathbf{C} \end{aligned}}$$

Matrix Form

- We can rewrite the equations in matrix form
- This gives us a compact notation and shows how different forms of cubic curves can be related
- It also is a very efficient form as it can take advantage of existing 4x4 matrix hardware support…

Bezier Curves & Cubic Curves

- By adjusting the 4 control points of a cubic Bezier curve, we can represent any cubic curve
- Likewise, any cubic curve can be represented uniquely by a cubic Bezier curve
- There is a one-to-one mapping between the 4 Bezier control points ($\mathbf{p}_0,\mathbf{p}_1,\mathbf{p}_2,\mathbf{p}_3$) and the pure cubic coefficients ($\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}$)
- The Bezier basis matrix $\mathbf{B}_{Bez}$ (and it's inverse) perform this mapping
- There are other common forms of cubic curves that also retain this property (Hermite, Catmull-Rom, B-Spline)

Derivatives

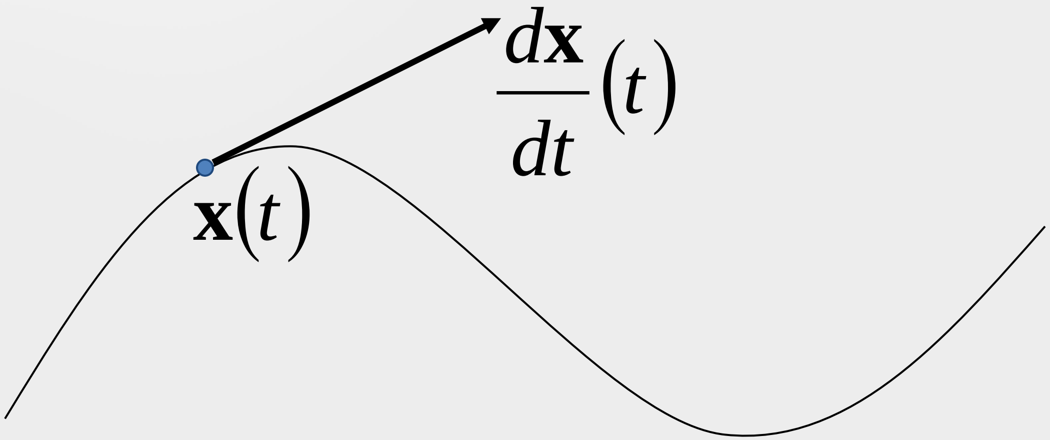
- Finding the derivative (tangent) of a curve is easy:

$$\mathbf{x} = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d} \qquad \frac{d\mathbf{x}}{dt} = 3\mathbf{a}t^2 + 2\mathbf{b}t + \mathbf{c}$$

$$\mathbf{x} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix} \qquad \frac{d\mathbf{x}}{dt} = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \end{bmatrix}$$

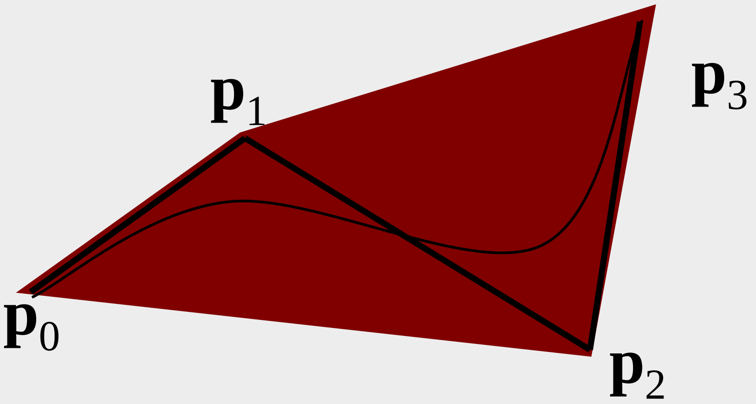
Tangents

- The derivative of a curve represents the tangent vector to the curve at some point



Convex Hull Property

- If we take all of the control points for a Bezier curve and construct a convex polygon around them, we have the *convex hull* of the curve
- An important property of Bezier curves is that every point on the curve itself will be somewhere within the convex hull of the control points

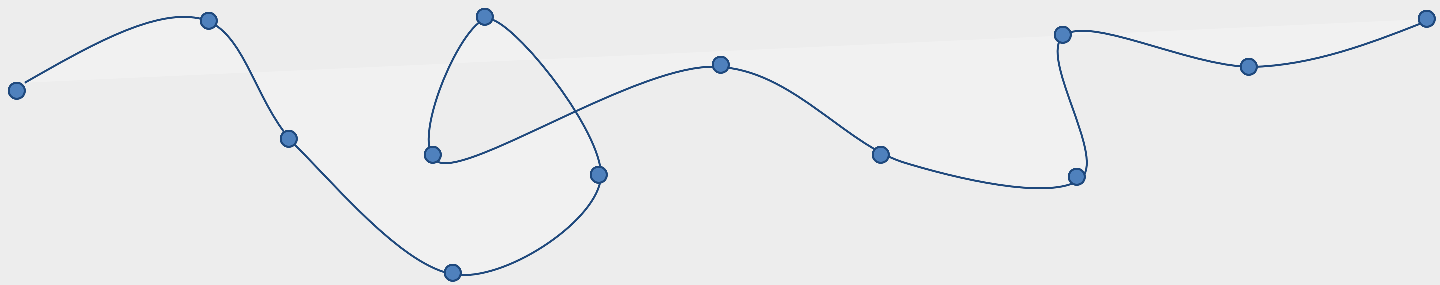


Continuity

- A cubic curve defined for *t* ranging from 0 to 1 will form a single continuous curve and not have any gaps
- We say that it has *geometric continuity*, or C^0 *continuity*
- We can easily see that the first derivative will be a continuous quadratic function and the second derivative will be a continuous linear function
- The third derivative (and all others) are continuous as well, but in a trivial way (constant), so we generally just say that a cubic curve has *second derivative continuity* or C^2 *continuity*
- In general, the higher the continuity value, the 'smoother' the curve will be, although it's actually a little more complicated than that…

Interpolation / Approximation

- We say that cubic Bezier curves *interpolate* the two endpoints ($\mathbf{p}_0$ & $\mathbf{p}_3$), but only *approximate* the interior points ($\mathbf{p}_1$ & $\mathbf{p}_2$)
- In geometric design applications, it is often desirable to be able to make a single curve that interpolates through several points

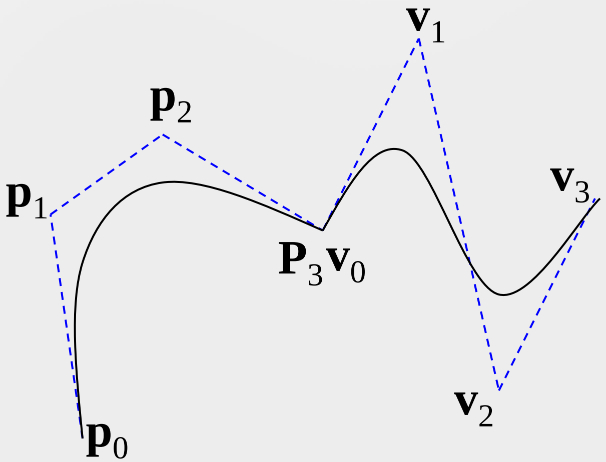


Piecewise Curves

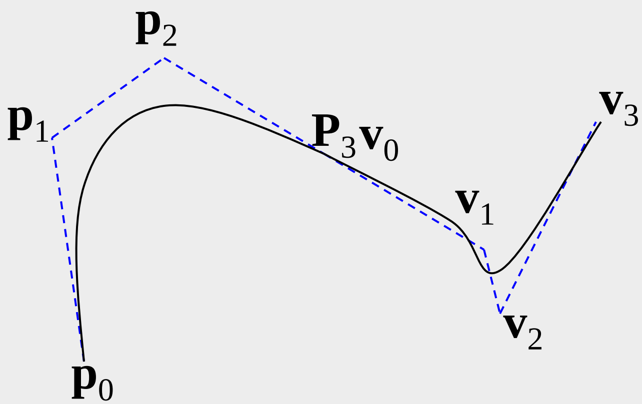
- Rather than use a very high degree curve to interpolate a large number of points, it is more common to break the curve up into several simple curves
- For example, a large complex curve could be broken into cubic curves, and would therefore be a *piecewise cubic curve*
- For the entire curve to look smooth and continuous, it is necessary to maintain C^1 continuity across segments, meaning that the position and tangents must match at the endpoints
- For smoother looking curves, it is best to maintain the C^2 continuity as well

Connecting Bezier Curves

- A simple way to make larger curves is to connect up Bezier curves
- Consider two Bezier curves defined by $\mathbf{p}_0 \ldots \mathbf{p}_3$ and $\mathbf{v}_0 \ldots \mathbf{v}_3$
- If $\mathbf{p}_3 = \mathbf{v}_0$, then they will have C^0 continuity
- If $(\mathbf{p}_3 - \mathbf{p}_2) = (\mathbf{v}_1 - \mathbf{v}_0)$, then they will have C^1 continuity
- C^2 continuity is more difficult…



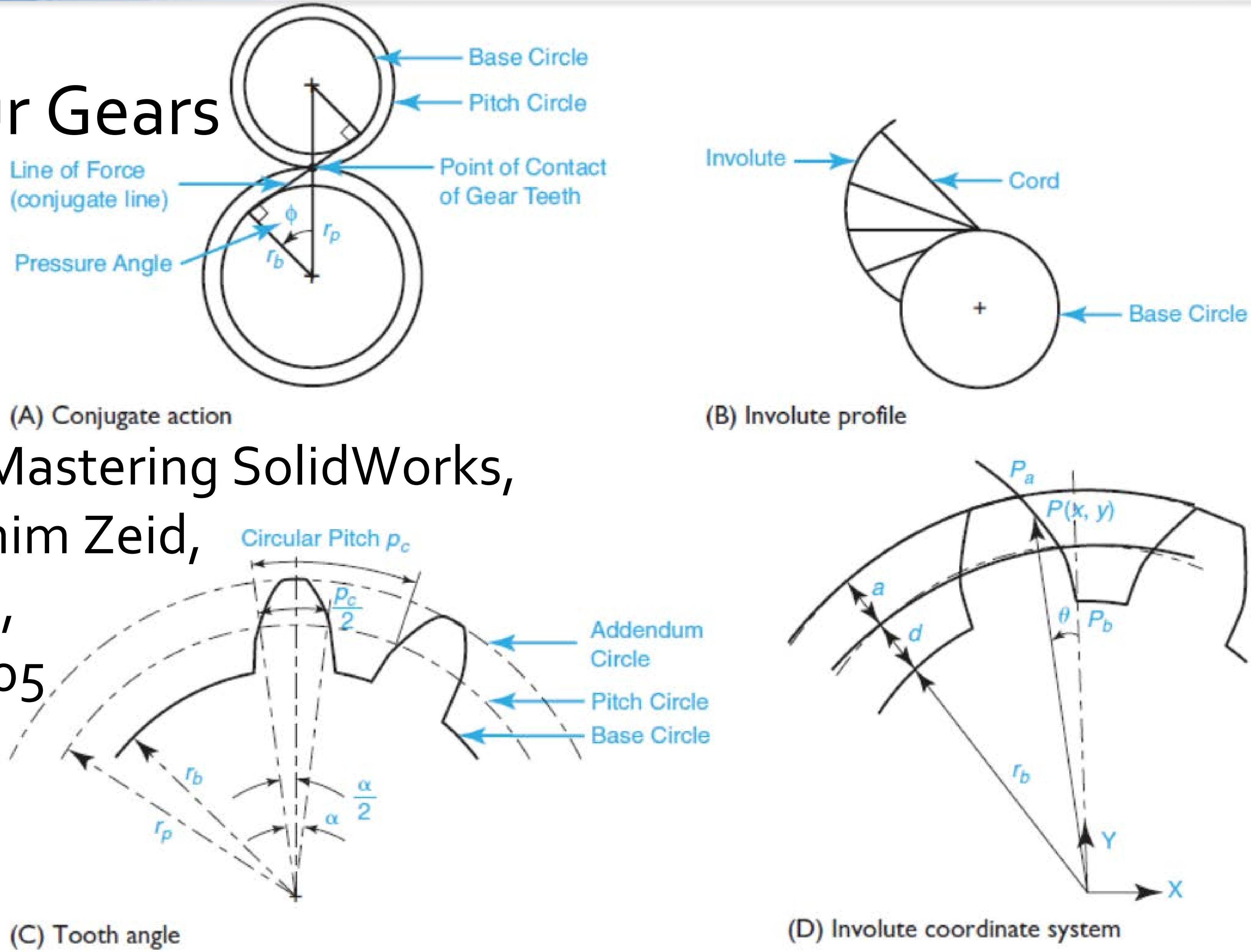
C^0 continuity



C^1 continuity

Spur Gears

ref. Mastering SolidWorks,
Ibrahim Zeid,
2014,
pg.105



(A) Conjugate action

(B) Involute profile

(C) Tooth angle

(D) Involute coordinate system

FIGURE 4.4
Details of a gear tooth

Spur Gears with involute profile

The circular pitch , p_c

Where $d_p = 2r_p$ is the pitch circle diameter,

N is the number of gear teeth and α is the tooth angle.

The pitch circle radius

$$p_c = \frac{\pi d_p}{N} \qquad \frac{p_c}{2} = r_p \alpha$$

$$\alpha = \frac{\pi}{N} \text{ radians or } \alpha = \frac{180}{N} \text{ degrees}$$

We align the involute of one tooth with the XY coordinate system as shown in Figure 4.4D.

$$x = -r_b(\sin\theta - \theta\cos\theta) \qquad 0 \le \theta \le \theta_{\max}$$
$$y = r_b(\cos\theta + \theta\sin\theta)$$

(4.4)

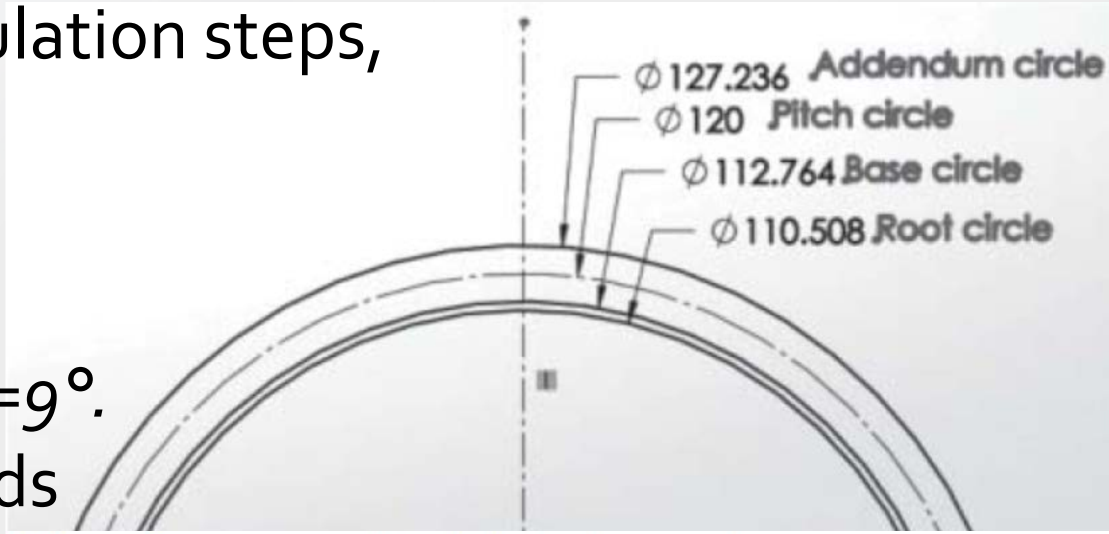
$$r_b = r_p \cos\phi$$

$$d = r_p - r_b$$

$$r_r = 0.98 r_b \qquad r_a = r_p + a = r_p + d$$

Spur Gears with involute profile

Example 4.2 Create the CAD model of a spur gear with r_p = *60 mm*, ϕ=20°, and *N=20*.
Using the above calculation steps,
we get r_b=*56.382 mm*,
d = *a* = *3.618 mm*,
r_a = *63.618 mm*,
r_r= 55.254 mm, and α=9°.
There are two methods
to create the tooth
involute curve.



Step 1: Create *Sketch1* circles and axes:

File > **New** > **Part** > **OK** > **Front Plane** > **Circle** on **Sketch** tab > sketch four circles and dimension as shown > **Centerline** on the **Sketch** tab > sketch vertical line > **File** > **Save As** > *example4.2* > **Save.**

Involute profile curve as spline

We can arbitrarily select a large enough value for θ_{max}
so that the involute crosses the addendum circle and then trim it to that circle. Therefore, we create the involute profile by generating points on it using Eq. (4.4) and connecting them with a spline curve, or we input Eq. (4.4) into a CAD/CAM system.

Note: Set the part units to mm before you start. The vertical centerline serves as a validation that the involute bottom endpoint passes through it when we create it in Step 2. Also, you will not close the sketch until you finish Step 5.

Involute profile curve as spline

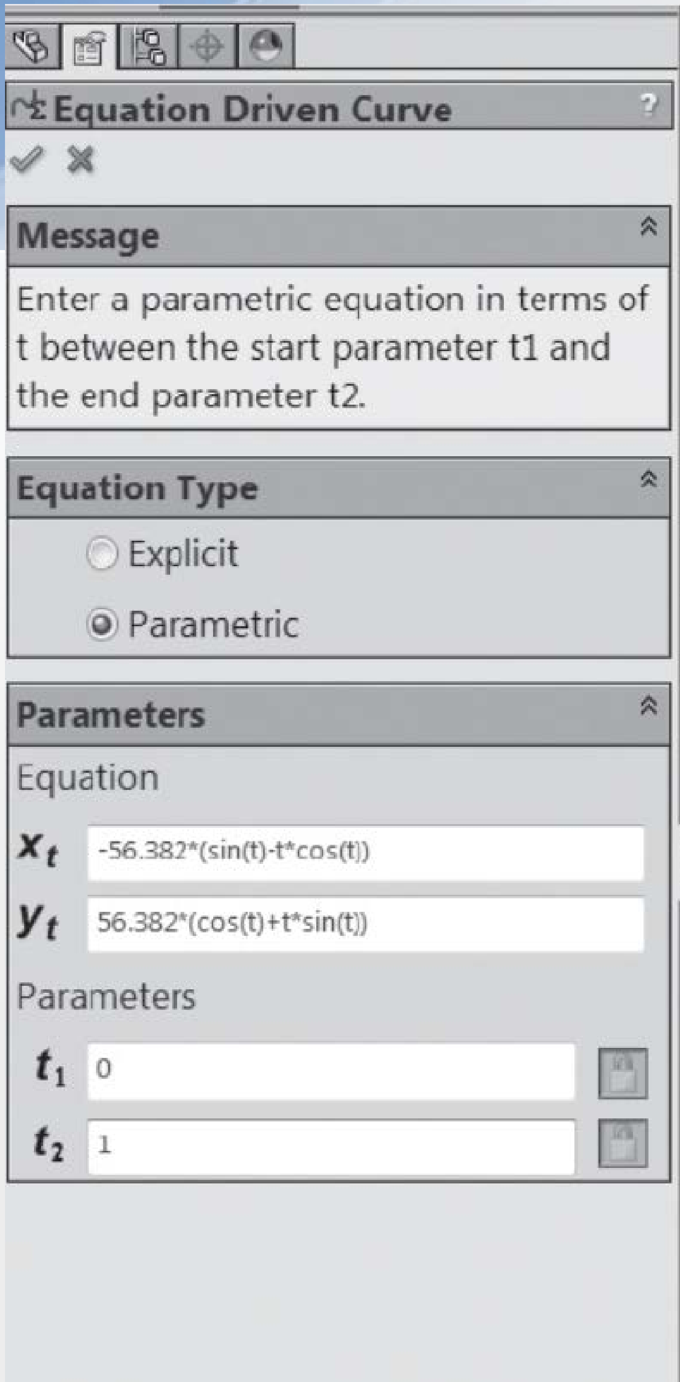
In the **first method**, we use Eq. (4.4) with $\Delta\theta$=5°. We generate 11 points on the involute, for a θ_{max} = 50°.

We generate the points on the involute curve. We then use **Insert > Curve > Curve Through XYZ Points.**

A better method is to input Eq. (4.4) to SolidWorks and let it generate the curve. Enter the **involute parametric equation** given by Eq. (4.4) into a CAD/CAM system to sketch the involute curve as a spline. SolidWorks uses the parameter *t*, requiring us to replace θ by *t* when we input the equation.

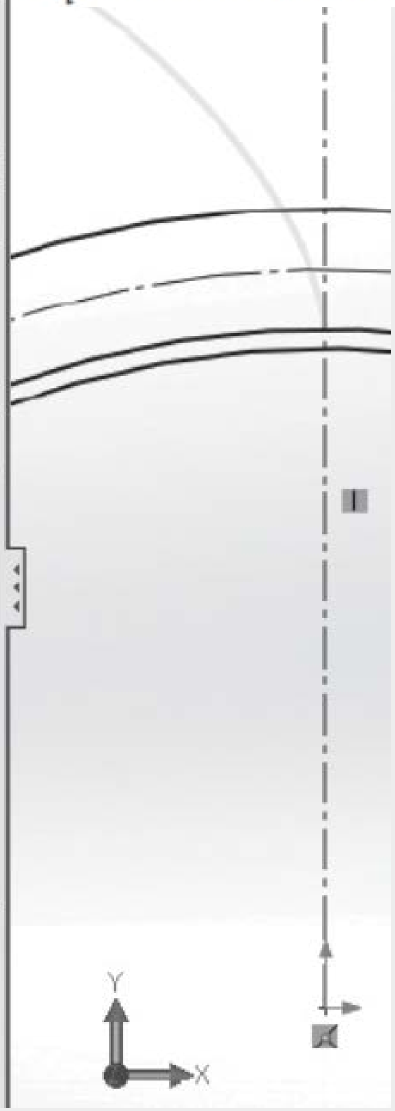
Create one gear tooth and use sketch circular pattern to pattern it to create all gear teeth.

$$\begin{aligned} x &= -r_b(\sin\theta - \theta\cos\theta) \\ y &= r_b(\cos\theta + \theta\sin\theta) \end{aligned} \qquad 0 \le \theta \le \theta_{\max} \qquad (4.4)$$

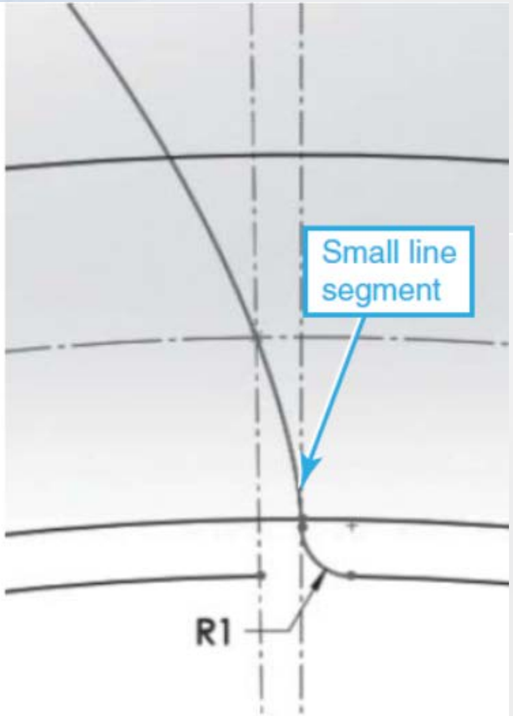


Step 2: Create *Sketch1*-tooth involute:

Sketch tab > **Spline** dropdown on the **Sketch** tab > **Equation Driven Curve** > **Parametric** > enter x and y equations and limits as shown > ✔.

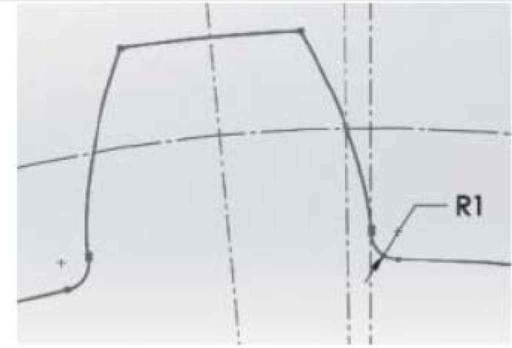


Step 3: Create *Sketch1*-tooth bottom:



Line on **Sketch** tab > sketch a line passing through bottom end of involute curve and crossing the root circle > **Esc** on keyboard > select the line + **Ctrl** on keyboard + involute curve > **Tangent** from **Add Relations** options on left pane > ✔ > **Point** on **Sketch** tab > create a point at intersection of involute and pitch circle (turn relations on: **View** > **Sketch Relations** to see all) > **Center line** on **Sketch** tab > sketch a line passing through origin and crossing involute at any point > **Esc** key > select centerline just created + **Ctrl** + point > **Coincident** from **Add Relations** options on left pane > ✔ > **Trim Entities** on **Sketch** tab > **Trim to closest** > select line below root circle and select root circle between two centerlines > ✔ > **Fillet** on **Sketch** tab > enter 1 mm for radius > select line and root circle > **Yes** to continue > ✔ > select base circle > **Delete** key on keyboard.

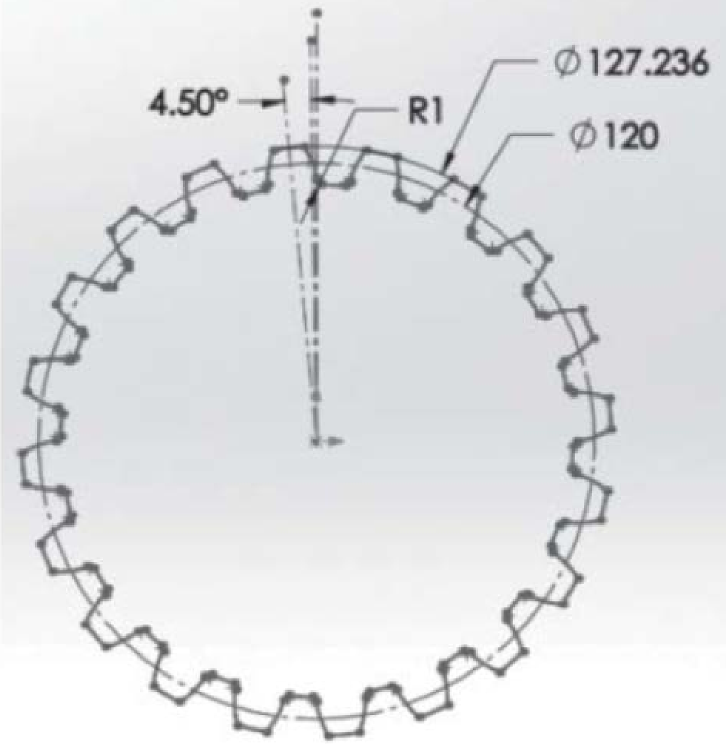
Step 4: Create *Sketch1*-tooth other half: **Trim Entities** on **Sketch** tab > **Trim to closest** > select involute top part > **Centerline** on **Sketch** tab > sketch a line passing through origin and to left of involute > **Smart Dimension** on **Sketch** tab > select the centerline just created and the other centerline to the right of it > enter 4.5 > ✔ > **Mirror Entities** on **Sketch** tab > select involute + **Ctrl** key + line segment connected to involute + fillet created in Step 3 > click **Mirror about** box on left of screen > select the far left centerline > ✔ > **Trim Entities** on **Sketch** tab > **Trim to closest** > click addendum circle outside tooth > click root circle inside tooth twice to delete its two segments inside the tooth > ✔.



Spur Gear

.

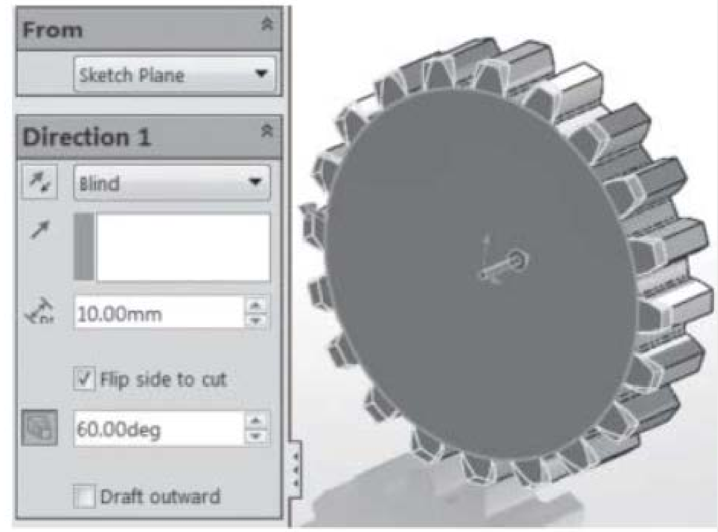
Step 5: Create *Sketch1*-all gear teeth: **Linear Sketch Pattern** dropdown on **Sketch** tab > **Circular Sketch Pattern >** click first box under **Parameters** on left pane > select origin to define axis of pattern > click **Entities to Pattern** box > select the tooth profile 7 entities > enter 20 for the number of instances to create > ✔ > the sketch becomes over defined when you pattern the tooth because of the profile mirror of first tooth. Click this sequence to resolve it: **Over Defined** (shown red in status bar) > **Diagnose** > **Accept** > **Trim Entities** > **Trim to closest** > trim all excess form root circle (segments inside teeth) > ✔ > exit sketch.



Spur Gear chamfer

Step 6: Create *Gear* feature: Select *Sketch1* > **Features** tab > **Extruded Boss/Base** > Enter 25 for thickness (D1) > reverse extrusion direction > ✔.

Step 7: Create *Sketch2* and *Cut-Extrude1-Chamfer*: Select *Gear* front face > **Features tab** > **Extruded Cut** > **Circle** on **Sketch** tab > click origin and snap to teeth root circle > exit sketch > enter 10 for thickness (**D1**) > check **Flip side cut** as shown above > click **Draft** icon as shown above > enter 60 for draft angle > ✔.



Step 8: Create *Sketch3* and *Cut-Extrude2-Chamfer*: Repeat Step 7, but use the back face of *Gear*.

Spline curve

.