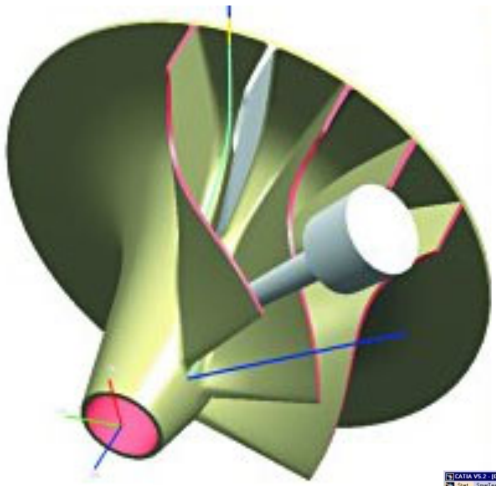
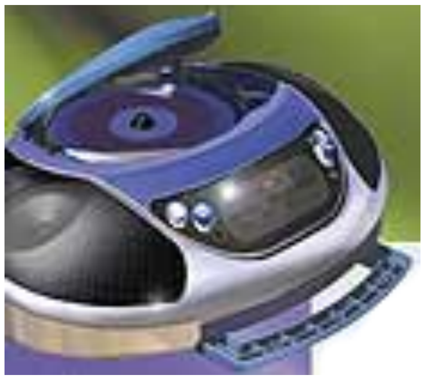
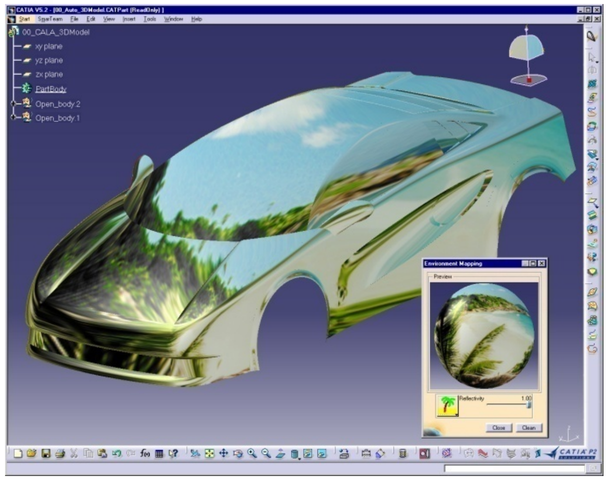
Curves

- Foundation of Free-form Surfaces



Why Not Simply Use a Point Matrix to Represent a Curve?

- Storage issue and limited resolution
- Computation and transformation
- Difficulties in calculating the intersections or curves and physical properties of objects
- Difficulties in design (e.g. control shapes of an existing object)
- Poor surface finish of manufactured parts

Advantages of Analytical Representation for Geometric Entities

- A few parameters to store
- Designers know the effect of data points on curve behavior, control, continuity, and curvature
- Facilitate calculations of intersections, object properties, etc.

Analytic Curves vs. Synthetic Curves

- Analytic Curves are points, lines, arcs and circles, fillets and chamfers, and conics (ellipses, parabolas, and hyperbolas)
- Synthetic curves include various types of splines (cubic spline, B-spline, Beta-spline) and Bezier curves.

Curved Surfaces

- In CAD, We want to find a math form for representing curved surfaces, that :

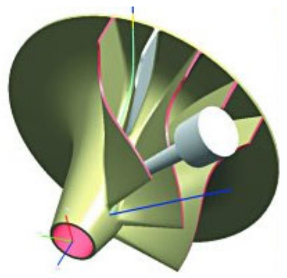
  (a) look nice (smooth contours)

  (b) is easy to manipulate and manufacture

  (c) follows prescribed shape (airfoil design)

To study the curved surface, we need to start from curves.

Parametric Representation

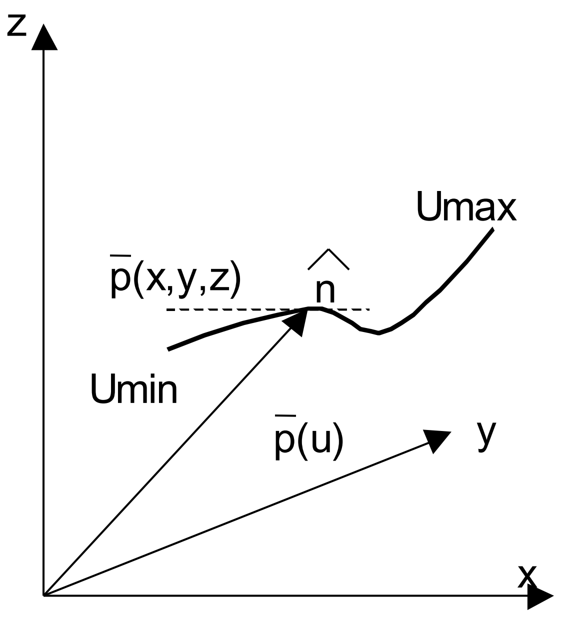


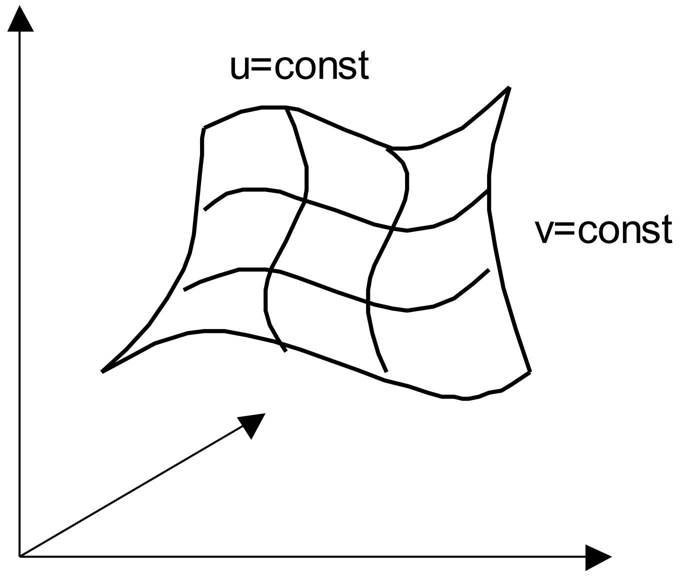
* a curve

$$\overline{P}(u) = [x(u), y(u), z(u)]^T$$

* a surface

$$\overline{P}(u,v) = [x(u,v), y(u,v), z(u,v)]^T$$

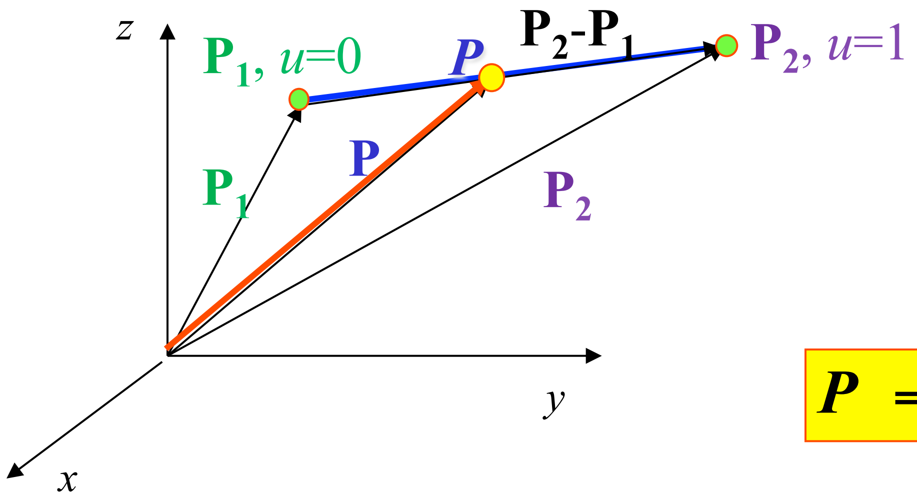




We can represent any functions of curve (curved surface) using parametric equation.

Parametric Representation of Lines

- How is a line equation converted by the CAD/ CAM software into the line database?
- How are the mathematical equation correlated to user commands to generate a line?

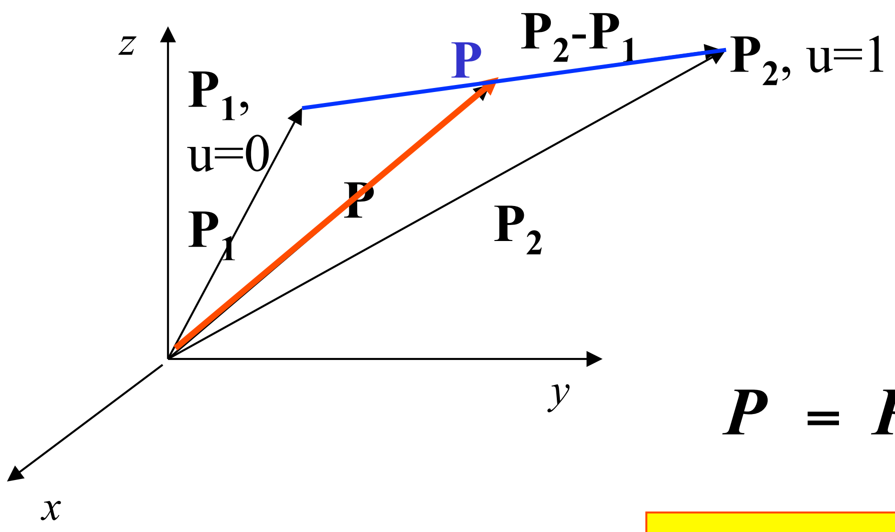


$$P = P_1 + (P - P_1)$$

$$P - P_1 = u\,(P_2 - P_1)$$

$$\boxed{P = P_1 + u(P_2 - P_1), \quad 0 \leq u \leq 1}$$

Lines



$$P = P_1 + u(P_2 - P_1), \quad 0 \le u \le 1$$

$$\begin{cases} x = x_1 + u(x_2 - x_1) \\ y = y_1 + u(y_2 - y_1) \quad 0 \le u \le 1 \\ z = z_1 + u(z_2 - z_1) \end{cases}$$

Circle

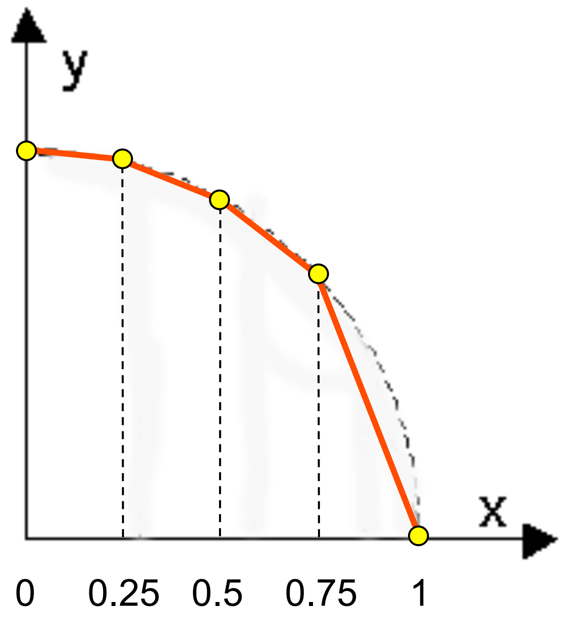
Representation 1 (Non-parametric)

$$x^2 + y^2 = 1$$

(a) $x = u$

$$y = \sqrt{1 - u^2}$$

- poor and non-uniform definition
- square root complicated to compute

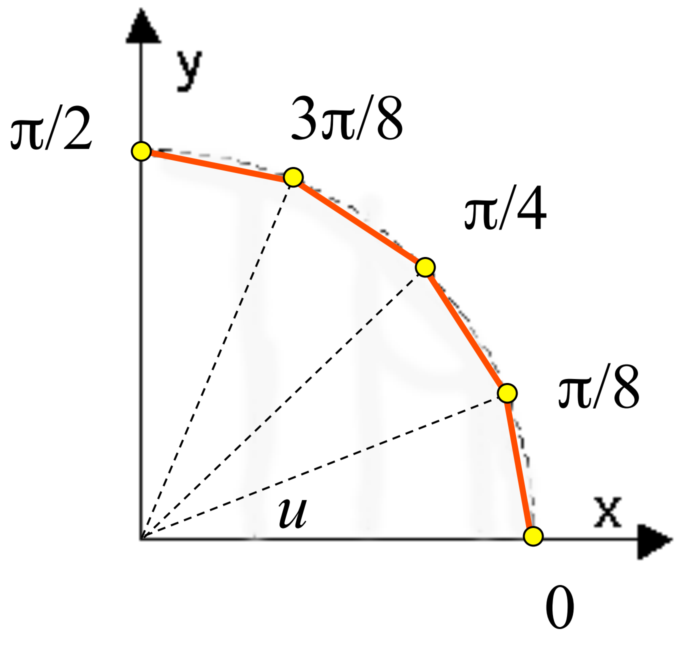


Circle

Representation 2 (parametric)

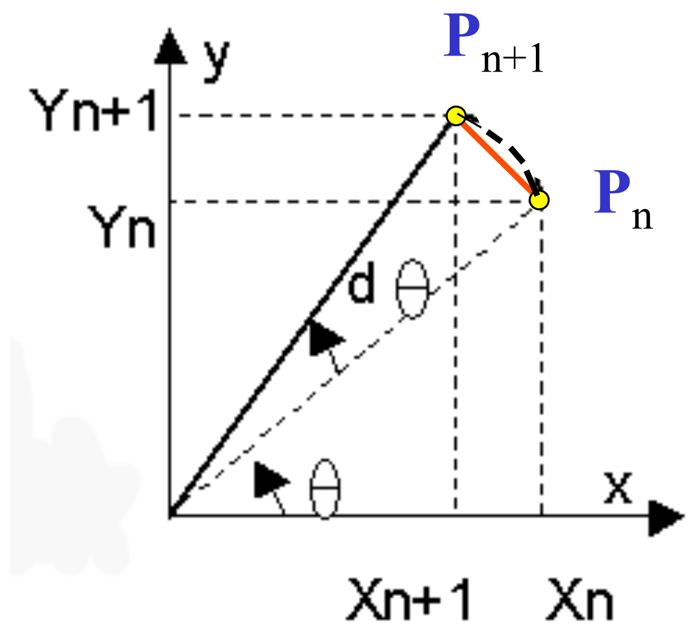
(b) $$\boxed{\begin{aligned} x &= \cos u \\ y &= \sin u \end{aligned}}$$

- better definition than (a)
- but still slow



Circle

Representation 3 (parametric)



Recursive approach

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

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

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

Observation: curves are represented by a series of line-segments

Similarly all conic sections can be represented.

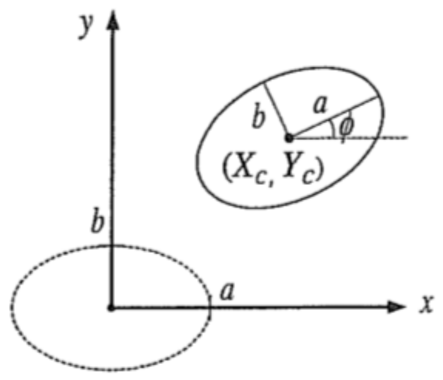
Ellipse

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

The computer uses the same method as in the Representation 3 of circle to reduce the amount of calculation.

Example

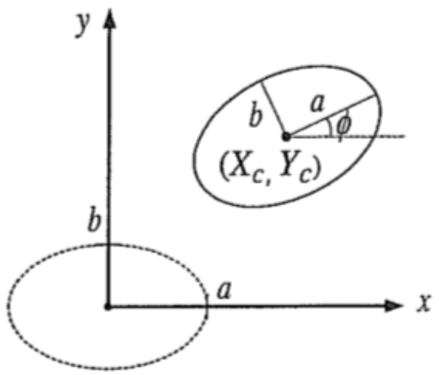
Derive the parametric equation of an ellipse in the *xy* plane, which has a center at (X_c, Y_c) and the major and the minor axes as illustrated in the accompanying figure.



ANSWER

The ellipse of interest can be obtained by rotating the reference ellipse at the origin by ϕ about the *z* axis and translating it by X_c in the *x* direction and by Y_c in the *y* direction. If the *x*, *y*, and *z* coordinates of the points on the ellipse of interest are denoted x^*, y^*, and z^* and those of the reference ellipse are denoted *x*, *y*, and *z*, the following equation will hold:

$$\left[x^*\ y^*\ 0\ 1\right]^T = Trans(X_c, Y_c, 0)Rot(z, \phi)(x\ y\ 0\ 1)^T$$



$$\left[x^* \ y^* \ 0 \ 1\right]^T = Trans(X_c, Y_c, 0)Rot(z,\phi)(x \ y \ 0 \ 1)^T$$

$$= \begin{bmatrix} 1 & 0 & 0 & X_c \\ 0 & 1 & 0 & Y_c \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix}$$

$$= (x\cos\phi - y\sin\phi + X_c \ \ x\sin\phi + y\cos\phi + Y_c \ \ 0 \ 1)$$

$$x^* = x\cos\phi - y\sin\phi + X_c = a\cos\theta\cos\phi - b\sin\theta\sin\phi + X_c$$
$$y^* = x\sin\phi + y\cos\phi + Y_c = a\cos\theta\sin\phi + b\sin\theta\cos\phi + Y_c$$
$$z^* = 0 \qquad (0 \le \theta \le 2\pi)$$

Parabola

explicit $x = cy^2$

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

Hyperbola

implicit

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

parametric

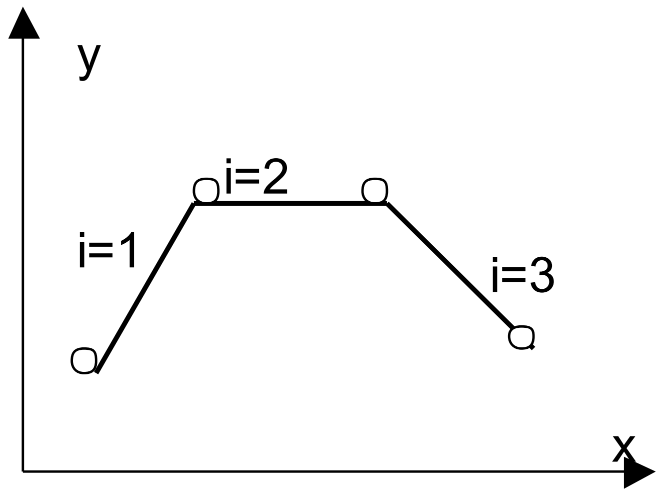
$$\begin{cases} x = a\cosh u \\ y = b\sinh u \\ z = z_o \end{cases} \qquad 0 \le u \le \infty$$

Parametric Representation of Synthetic Curves

- Analytic curves are usually not sufficient to meet geometric design requirements of mechanical parts.
- Many products need **free-form**, or **synthetic curved surfaces**.
- Examples: car bodies, ship hulls, airplane fuselage and wings, propeller blades, shoe insoles, and bottles
- The need for synthetic curves in design arises on occasions:
  - when a curve is represented by a collection of measured data points and (**generation**)
  - when a curve must change to meet new design requirements. (**modification**)

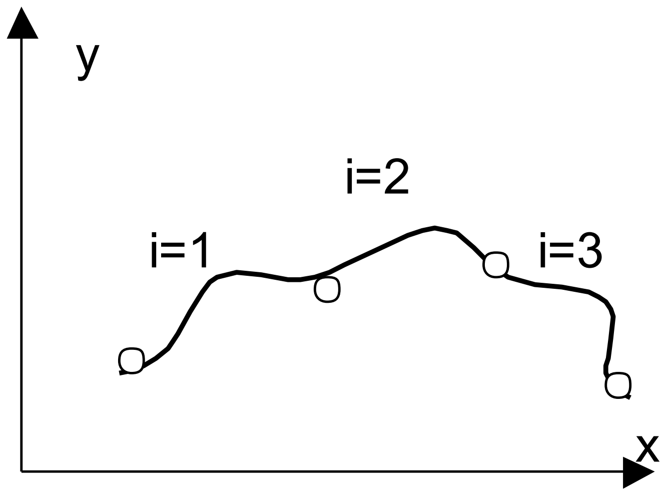
The Order of Continuity

The order of continuity is a term usually used to measure the degree of continuous derivatives (C^0 , C^1 , C^2).



Simplest Case Linear Segment

$$y_i = a_{i0} + a_{i1}x$$

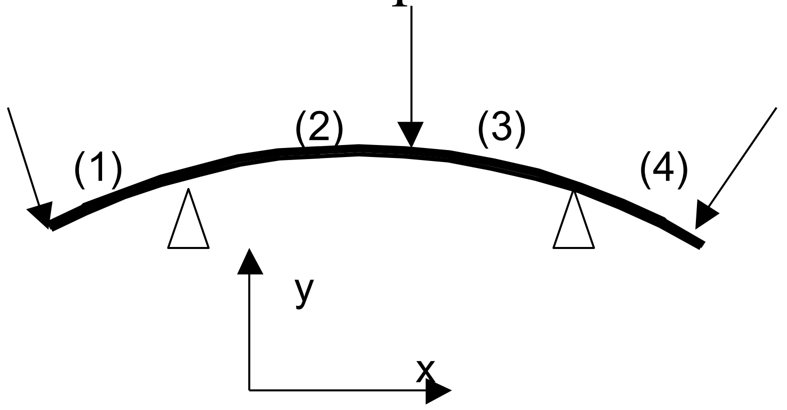


High order polynomial may lead to "ripples"

$$y_i = a_{i0} + a_{i1}x + \ldots + a_{in}x^n$$

Splines – Ideal Order

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

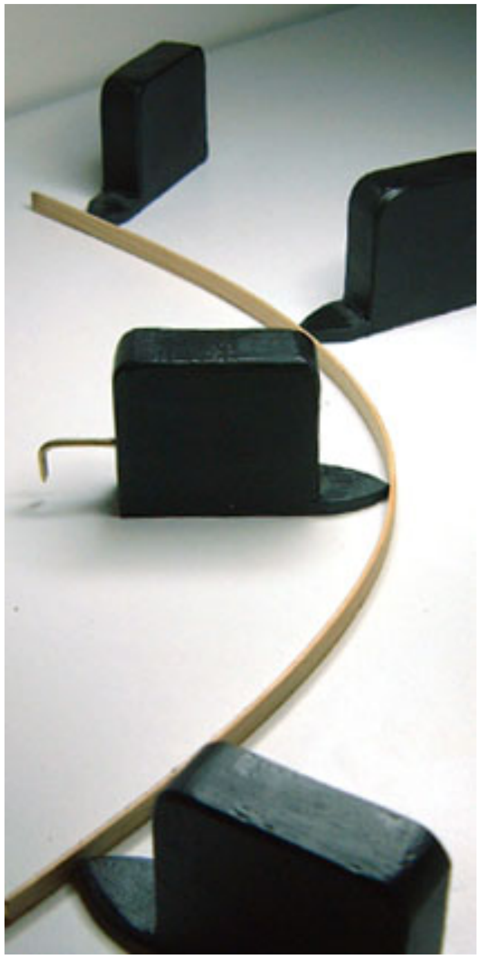


$$y''(x) = R(x) = \frac{M(x)}{EI} = \frac{a_i x + b_i}{EI}$$

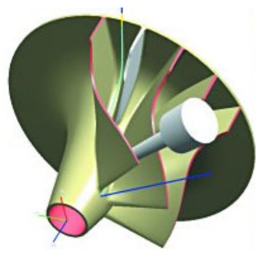
$$y(x) = \frac{1}{EI}\left[\frac{a_i}{6}x^3 + \frac{b_i}{2}x^2 + c_i x + d_i\right]$$

Cubic Spline

Drafting Spline



Hermite Cubic Splines



Cubic Spline

$$\vec{P}(u) = [x(u), y(u), z(u)]^T$$

$$\begin{cases} x(u) = c_{3x}u^3 + c_{2x}u^2 + c_{1x}u + c_{0x} \\ y(u) = c_{3y}u^3 + c_{2y}u^2 + c_{1y}u + c_{0y} \\ z(u) = c_{3z}u^3 + c_{2z}u^2 + c_{1z}u + c_{0z} \end{cases}$$

$3\times4 = 12$ coefficients to be determined

$$\vec{p}(u) = [x(u) \;\; y(u) \;\; z(u)]^T = \sum_{i=0}^{3} \vec{C}_i u^i \qquad (0 \le u \le 1)$$

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

Hermite Cubic Splines

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

$$\vec{P}' = 3\vec{C}_3 u^2 + 2\vec{C}_2 u + \vec{C}_1$$

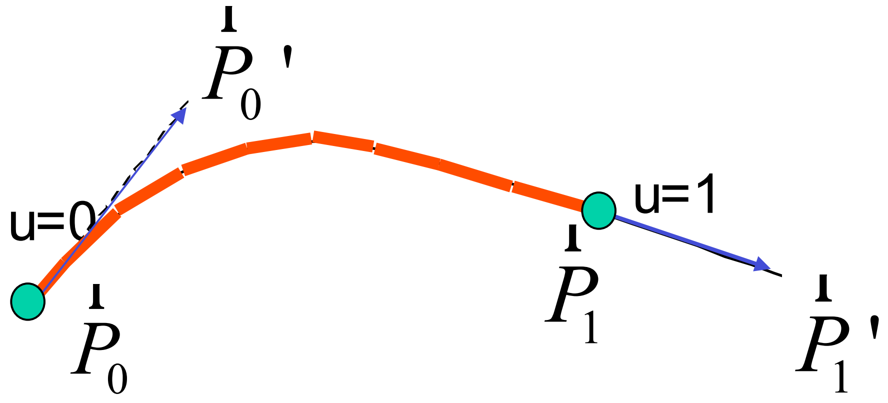
Two End Points

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

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

4×3 equations from two control points

Boundary Conditions: Location of the two end points and their slopes



Hermite Cubic Splines

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

$$\vec{C}_1 = \vec{P}_0'$$

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

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

12 unknowns
and
12 equations

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

All parameters can be determined

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

Hermite Cubic curve in vector form

Hermite Cubic Splines Equation:

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

Hermite Cubic curve in vector form

In matrix form:

$$\begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{P}_0 \\ \vec{P}_1 \\ \vec{P}_0' \\ \vec{P}_1' \end{bmatrix}$$

Based on: Location of the two end points and their slopes

$$= U^T \left[M_H\right] \vec{V} \qquad 0 \le u \le 1$$

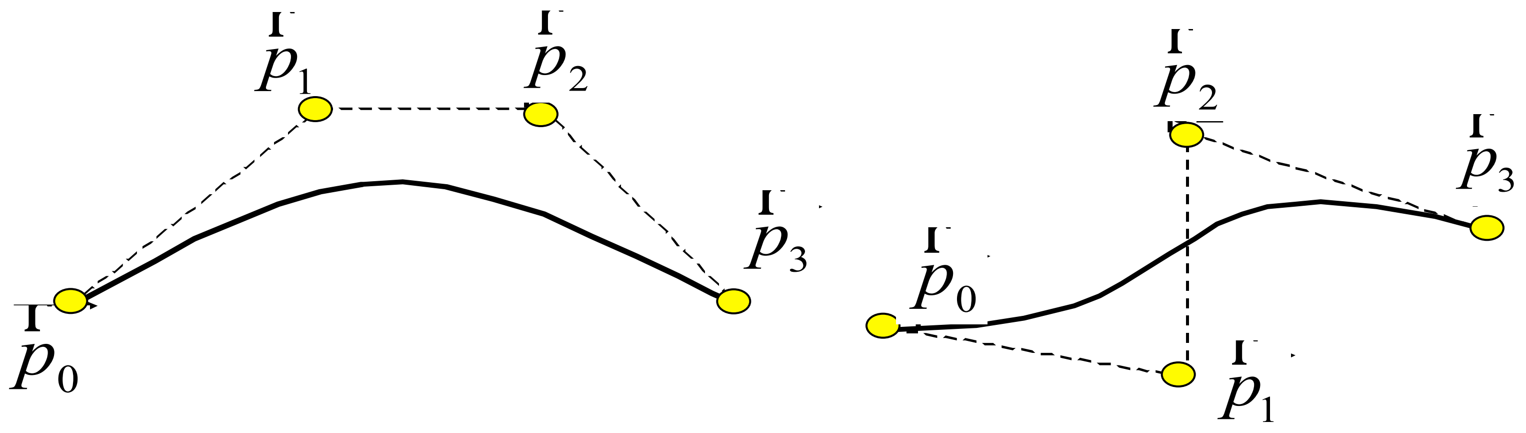
- $$\vec{P}'(u) = (6u^2 - 6u)\vec{P}_0 + (-6u^2 + 6u)\vec{P}_1 + (3u^2 - 4u + 1)\vec{P}_0' + (3u^2 - 2u)\vec{P}_1' \qquad 0 \le u \le 1$$

Limitations with Hermite Curves

- Hard to guess behavior between 2 defined points for arbitrary end point slopes

- Limited to 3rd degree polynomial therefore the curve is quite stiff

Bezier Curve



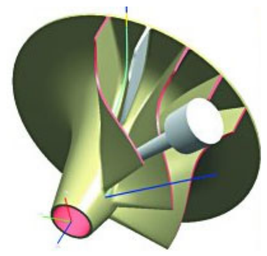
$$\vec{p}(u) = \sum_{i=0}^{n} \vec{p}_i B_{i,n}(u)$$

n —— segment(each polygon)

n+1 —— vertices (each polygon) and number of control points

$u \in [0, \quad 1]$

Bezier Curve



- P. Bezier of the French automobile company of Renault first introduced the Bezier curve.
- A system for designing sculptured surfaces of automobile bodies (based on the Bezier curve)

- passes $\overrightarrow{p_0}$ and $\overrightarrow{p_n}$, the two $\boxed{\text{end points.}}$

- has end point derivatives:

$$\vec{p}_0{}' = n(\vec{p}_1 - \vec{p}_0) \qquad \vec{p}_n{}' = n(\vec{p}_n - \vec{p}_{n-1})$$

- uses a vector of control points, representing the n+1 vertices of a "characteristic polygon".

Bernstein Polynomial

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

$B_{i,n}(u)$ is a function of the number of curve segments, n.

n=2

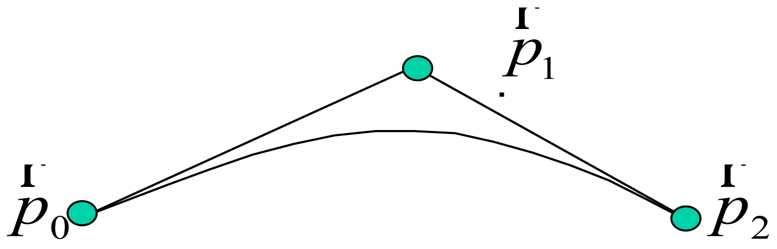
i	0	1	2
$\frac{n!}{i!(n-i)!}$	$\frac{2!}{0!\;\;2!} = 1$	$\frac{2!}{1!\;\;1!} = 2$	$\frac{2!}{2!\;\;0!} = 1$

Bernstein Polynomial

In the mathematical field of numerical analysis, a Bernstein polynomial, named after Sergei Natanovich Bernstein, is a polynomial in the Bernstein form, that is a linear combination of Bernstein basis polynomials.

A numerically stable way to evaluate polynomials in Bernstein form is de ***Casteljau's algorithm*** which reduces the computational demand caused by the factorials.

An Example: If $n = 2$,
then $n+1 = 3$ vertices



i	0	1	2
$\frac{n!}{i!(n-i)!}$	$\frac{2!}{0!\ 2!} = 1$	$\frac{2!}{1!\ 1!} = 2$	$\frac{2!}{2!\ 0!} = 1$

$$\vec{p}(u) = \sum_{i=0}^{n} \vec{p}_i B_{i,n}(u)$$

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

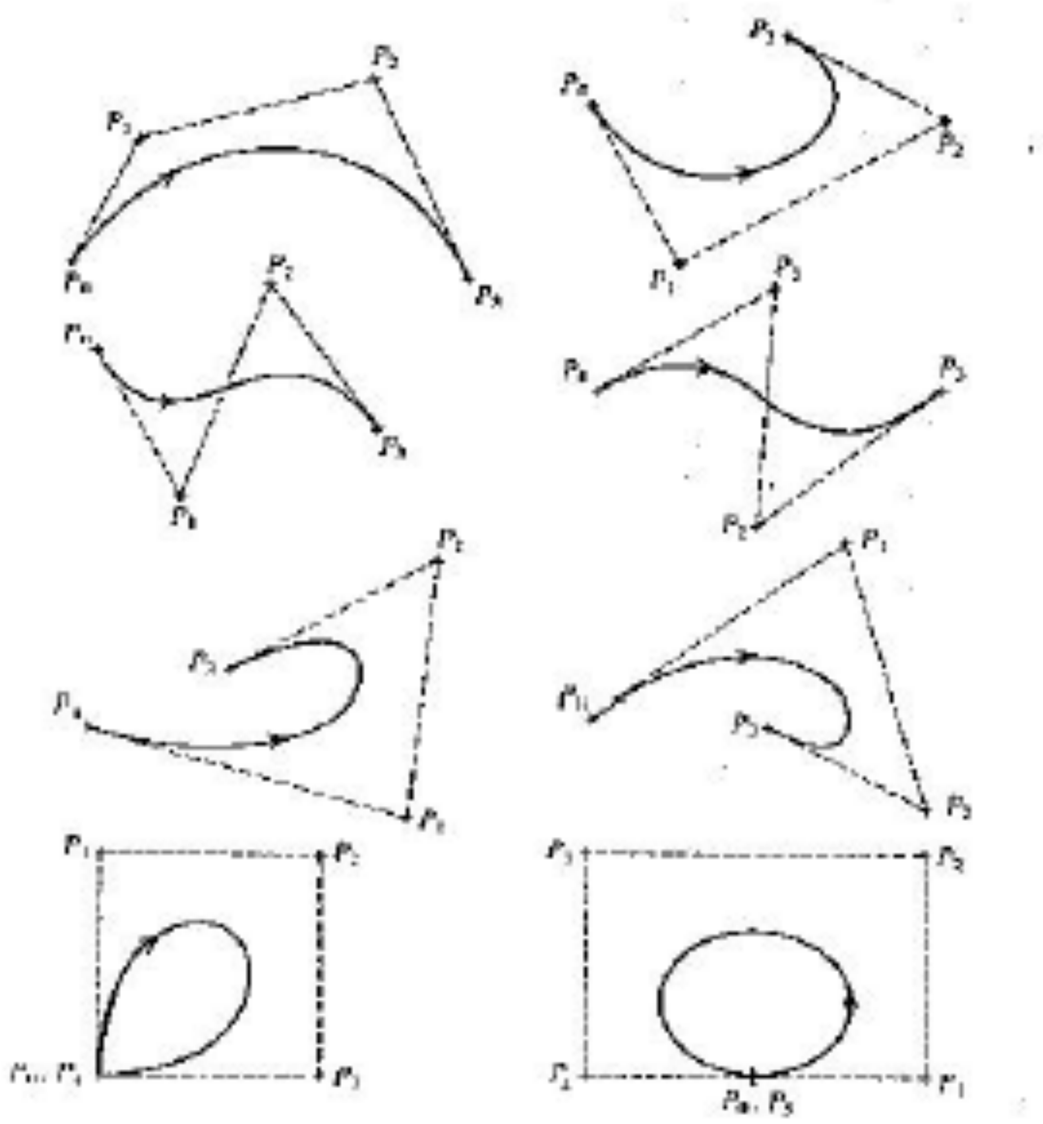
$$\boxed{\vec{p}(u) = 1 \times (1-u)^2 \vec{p}_0 + 2 \times u(1-u)\vec{p}_1 + 1 \times u^2 \vec{p}_2}$$

$$\vec{p}\,'(u) = -2(1-u)\vec{p}_0 + 2(1-2u)\vec{p}_1 + 2u\vec{p}_2$$

$$\vec{p}(0) = \vec{p}_0 \qquad \vec{p}\,'(0) = 2(\vec{p}_1 - \vec{p}_0)$$

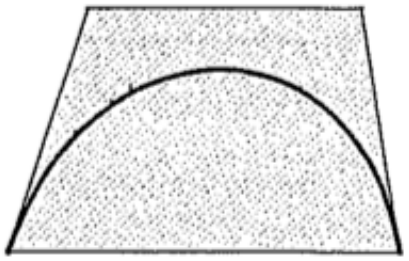
$$\vec{p}(1) = \vec{p}_2 \qquad \vec{p}\,'(1) = 2(\vec{p}_2 - \vec{p}_1)$$

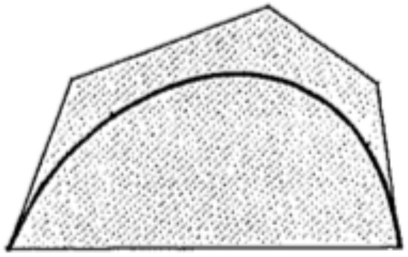
The order of Bezier curve is a function of the number of control points. Four control points (n=3) always produce a cubic Bezier curve.

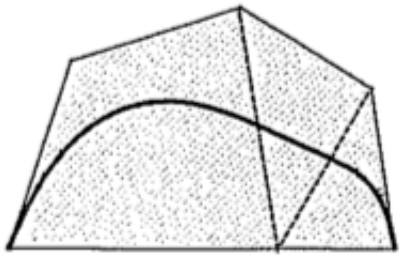


Convex Hull Property

In addition to the properties of a Bezier curve described previously, another important property of a Bezier curve is the *convex hull property*. A convex hull of a Bezier curve is a convex polygon formed by connecting the control points of the curve, as illustrated by the hatched areas in Figure Note that each of these Bezier curves resides completely inside its convex hull.


(a)


(b)


(c)

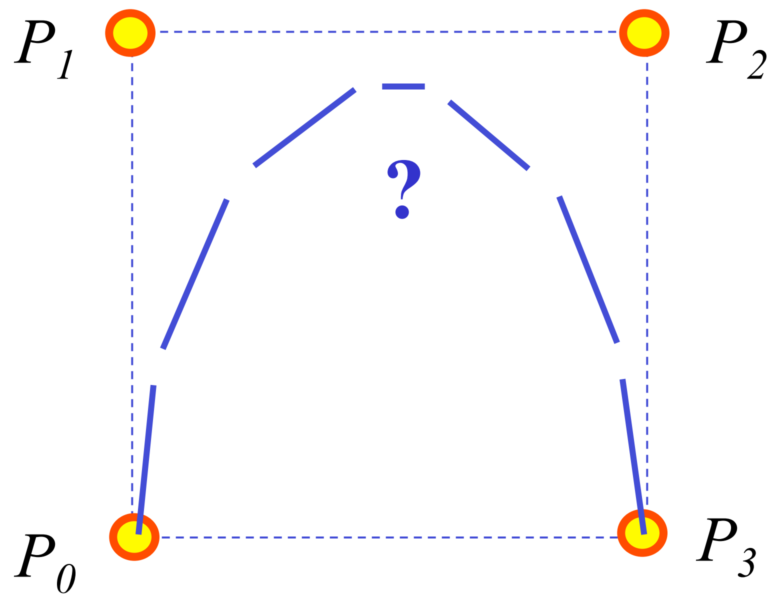
An Example

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

$$P_0 = \begin{bmatrix} 2 & 2 & 0 \end{bmatrix}^T, P_1 = \begin{bmatrix} 2 & 3 & 0 \end{bmatrix}^T, P_2 = \begin{bmatrix} 3 & 3 & 0 \end{bmatrix}^T, P_3 = \begin{bmatrix} 3 & 2 & 0 \end{bmatrix}^T$$

Find the equation of the resulting Bezier curve. Also find points on curve for

$u = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1$



Solution

$$\boxed{P(u) = P_0 B_{0,3} + P_1 B_{1,3} + P_2 B_{2,3} + P_3 B_{2,3} \qquad 0 \leq u \leq 1}$$

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

$$\boxed{P(u) = P_0 (1-u)^3 + 3P_1 u(1-u)^2 + 3P_2 u^2 (1-u) + P_3 u^3 \qquad 0 \leq u \leq 1}$$

Substituting the u values into his equation gives

u = 0, ¼, ½, ¾, 1

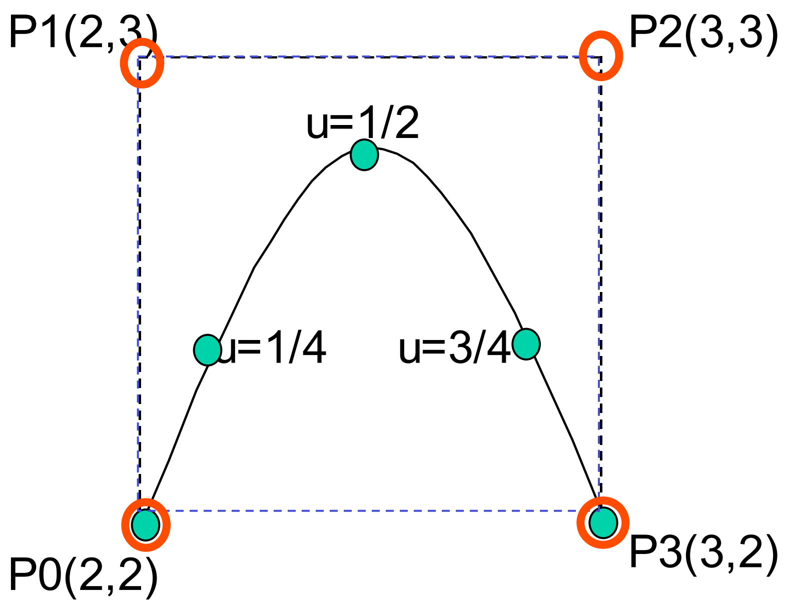
$$P(0) = P_0 = \begin{bmatrix}2 & 2 & 0\end{bmatrix}^T$$

$$P\left(\frac{1}{4}\right) = \frac{27}{64}P_0 + \frac{27}{64}P_1 + \frac{9}{64}P_2 + \frac{1}{64}P_3 = \begin{bmatrix}2.156 & 2.563 & 0\end{bmatrix}^T$$

$$P\left(\frac{1}{2}\right) = \frac{1}{8}P_0 + \frac{3}{8}P_1 + \frac{3}{8}P_2 + \frac{1}{8}P_3 = \begin{bmatrix}2.5 & 2.75 & 0\end{bmatrix}^T$$

$$P\left(\frac{3}{4}\right) = \frac{1}{64}P_0 + \frac{9}{64}P_1 + \frac{27}{64}P_2 + \frac{27}{64}P_3 = \begin{bmatrix}2.844 & 2.563 & 0\end{bmatrix}^T$$

$$P(1) = P_3 = \begin{bmatrix}3 & 2 & 0\end{bmatrix}^T$$



○ - control points, ***P₁, P₂, P₃, & P₄,***

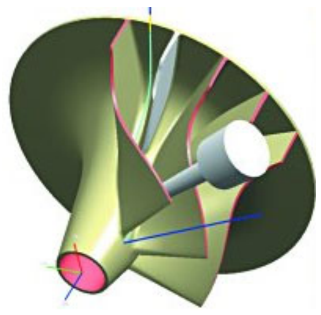
● - points on curve, ***P(u)***

Improvements of Bezier Curve Over the Cubic Spline

- The shape of Bezier curve is controlled only by its defining points (control points). First derivatives are not used in the curve development as in the cubic spline.
- The order or the degree of the Bezier curve is variable and is related to the number of points defining it; *n*+1 points define a *n*th degree curve.

  This is not the case for cubic splines where the degree is always cubic for a spline segment.
- The Bezier curve is smoother than the cubic splines because it has higher-order derivatives.

B-Spline



- A Generalization from Bezier Curve
- Better local control
- Degree of resulting curve is independent to the number of control points.

Math Representation

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

(k-1) degree of polynomial with ($n+1$) control points

— $\overline{P}_0, \overline{P}_1, \ldots, \overline{P}_n$ —— n+1 control points.

— $N_{i,k}(u)$ —— B-spline function (to be calculated in a recursive form)

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

Parametric Knots

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

u_j : parametric knots (or knot values), for an open curve B-spline:

$$\boxed{u_j = \begin{cases} 0 & j < k \\ j - k + 1 & k \le j \le n \\ n - k + 2 & j > n \end{cases}}$$

where, $0 \le j \le n+k$, thus if a curve with (k-1) degree and (n+1) control points is to be developed, (n+k+1) knots then are required with $0 \le u \le u_{max} = n - k + 2$

B-Spline (non-periodic, uniform)

The curve is expressed by:

$$P(u) = \sum_{i=0}^{n} P_i \, N_{i,k}(u) \qquad (u_{k-1} \le u \le u_{n+1}) \tag{1}$$

Where

$$N_{i,k} = \frac{(u-u_i)N_{i,k-1}}{u_{i+k-1}-u_i} + \frac{(u_{i+k}-u)N_{i+1,k-1}}{u_{i+k}-u_{i+1}} \tag{2}$$

Is the recursive expression ($N_{i,k}$depends on $N_{j,k-1}$ where j is a generic i) and the base case $N_{i,1}$ is defined as:

$$N_{i,1} = \begin{cases} 1 \; for \; u_i \le u_{i+1} \\ 0 \; otherwise \end{cases} \tag{3}$$

And the knots u_i are found based on the following:

$$u_i = \begin{cases} 0 & for \;\; 0 \le i < k \\ i-k+1 & for \; k \le i \le n \\ n-k+2 & for \; n < i \le n+k \end{cases} \tag{4}$$

B-Spline curve evaluation method:

1. Define k and n, where $k-1$ is the polynomial order and $n+1$ are the number of control points $P_0 \dots P_n$

2. Find the knots $u_i = u_0 \dots u_{n+k}$. This is done by applying rule (4) for each of the u_i

3. Find $N_{i,1} = N_{0,1} N_{1,1} \dots . N_{n-1,1}$ using the base case rule (3)

4. Find (note that you have to apply the rule that 0/0=0):
   a. $N_{i,2} = N_{0,2} N_{1,2} \dots . N_{n-2,2}$ using the base case rule (2) if you have k=2 stop here otherwise
   b. $N_{i,3} = N_{0,3} N_{1,3} \dots . N_{n-3,3}$ if you have k=3 stop here otherwise

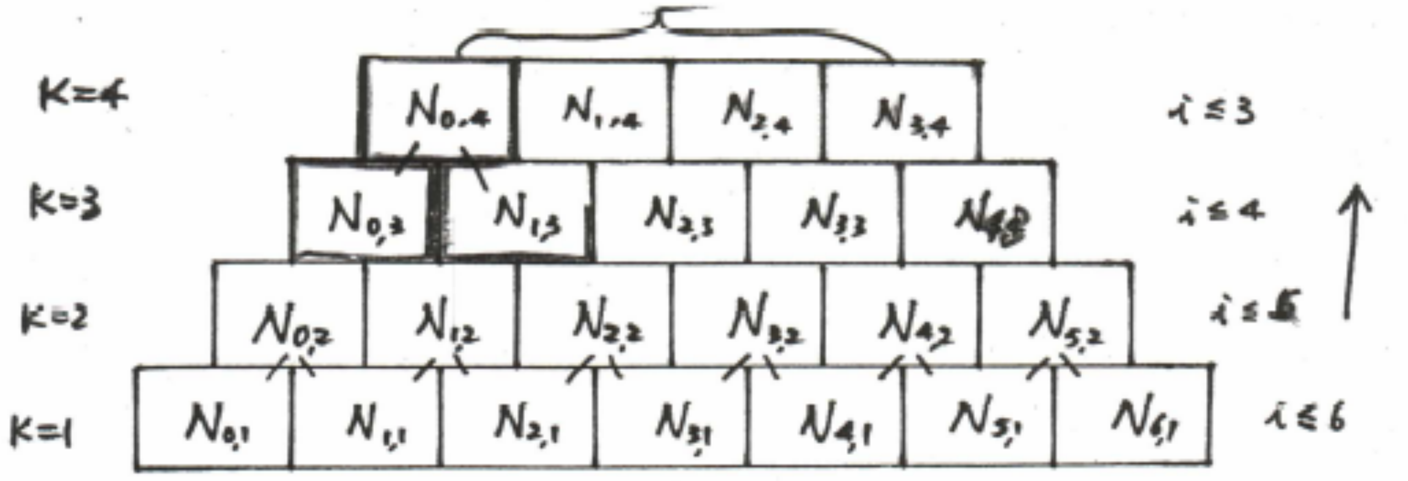
   keep repeating until you reach the defined order k

5. With the last step in 4. The $N_{i,k} = N_{0,k} N_{1,2} \dots . N_{n-k,k}$ have been obtained and the summation in (1) can be performed. Obviously in order to complete the curve expression the coordinates of the points P_i need to be known too.

Knot Value Calculation

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

$$\boxed{\underline{N_{i,k}} = f\,(N_{i,k-1}, N_{i+1,k-1})}$$



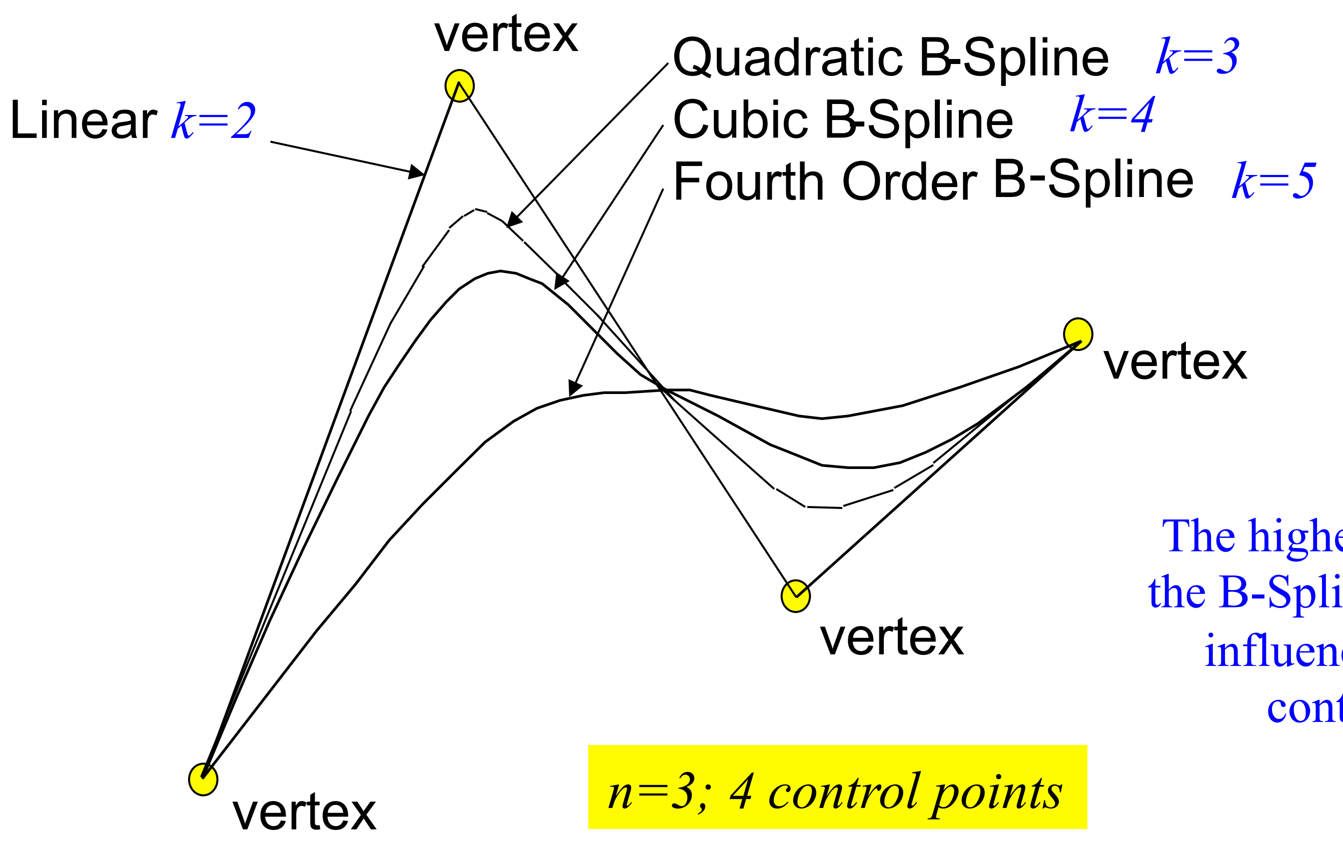
K=4
$N_{0,4}$ $N_{1,4}$ $N_{2,4}$ $N_{3,4}$
$i \le 3$
K=3
$N_{0,3}$ $N_{1,3}$ $N_{2,3}$ $N_{3,3}$ $N_{4,3}$
$i \le 4$
K=2
$N_{0,2}$ $N_{1,2}$ $N_{2,2}$ $N_{3,2}$ $N_{4,2}$ $N_{5,2}$
$i \le 5$
K=1
$N_{0,1}$ $N_{1,1}$ $N_{2,1}$ $N_{3,1}$ $N_{4,1}$ $N_{5,1}$ $N_{6,1}$
$i \le 6$

n = 3;   4 control points
k = 4;   4-1=3 cubic polynomial
$0 \le j \le n+k = 7$

n increases – wider base
k increases – wider & taller

Properties of B-Spline

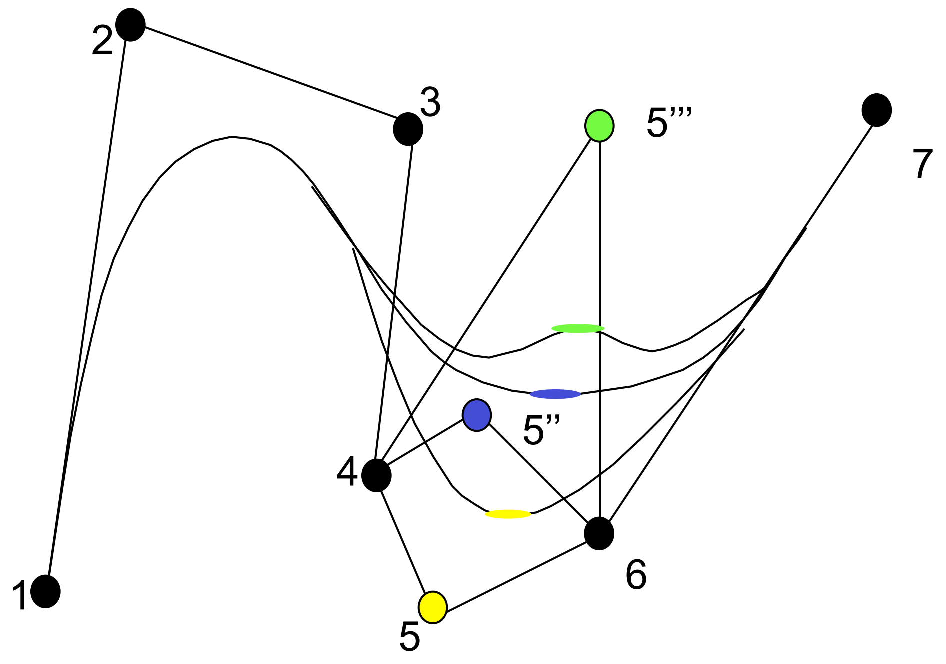
- Number of control points independent of degree of polynomial



The higher the order of the B-Spline, the less the influence the close control point

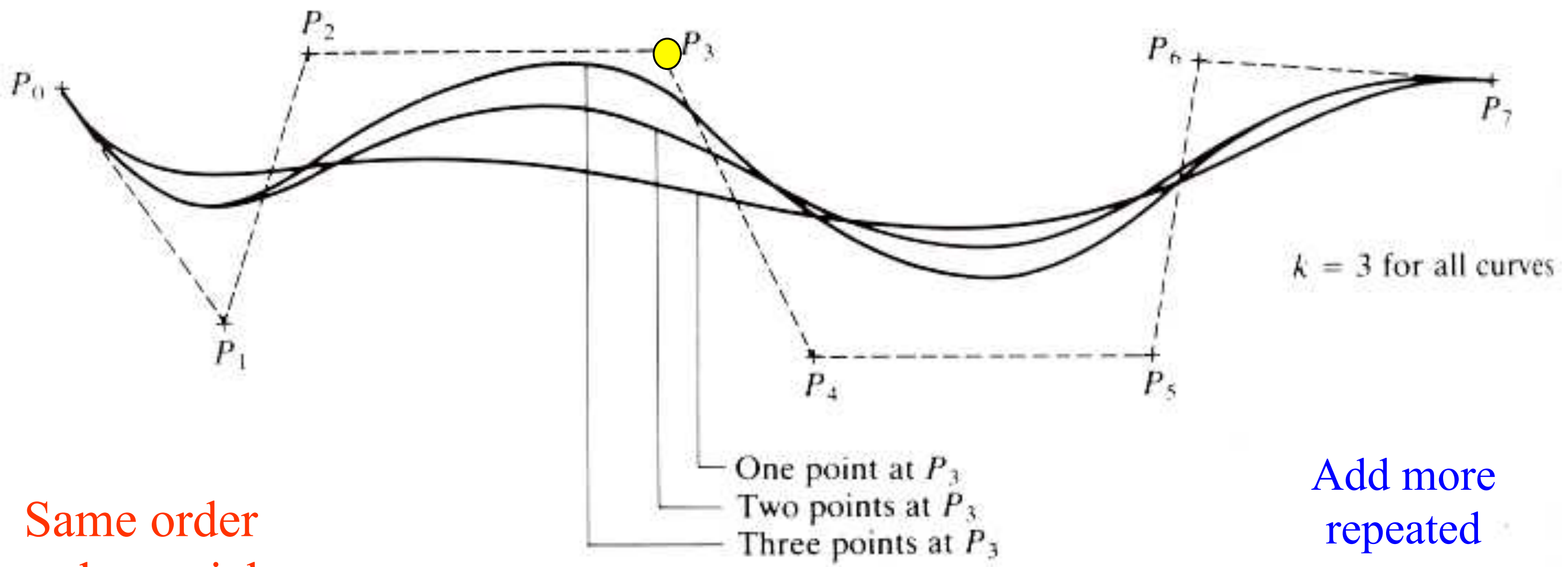
Properties of B-Spline

- B-spline allows better local control. Shape of the curve can be adjusted by moving the control points.
- Local control: a control point only influences *k* segments.



Properties of B-Spline

Repeated values of a control point can pull a B-spline curve forward to vertex. ("Interactive curve control")



Same order polynomial

Add more repeated control points to pull the curve

An Example

Find the equation of a cubic B-spline curve defined by the same control points as in the last example.

How does the curve compare with the Bezier curve?

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

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

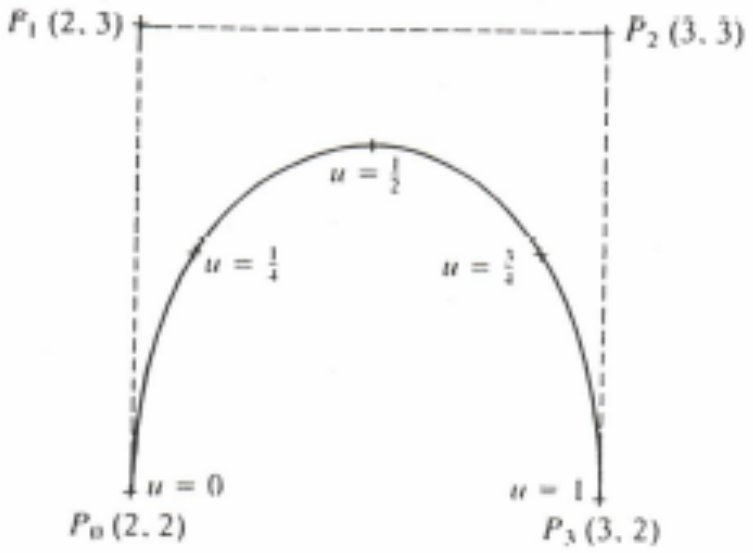
Find the equation of the resulting Bezier curve. Also find points on the curve for $u = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$, and 1.

Solution. Equation (5.91) gives

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

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

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



Example Problem for Finding the Bezier Curve

FIGURE 5-49
Bezier curve and generated points.

Example Problem for Finding the Bezier Curve

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

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

Substituting the u values into this equation gives

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

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

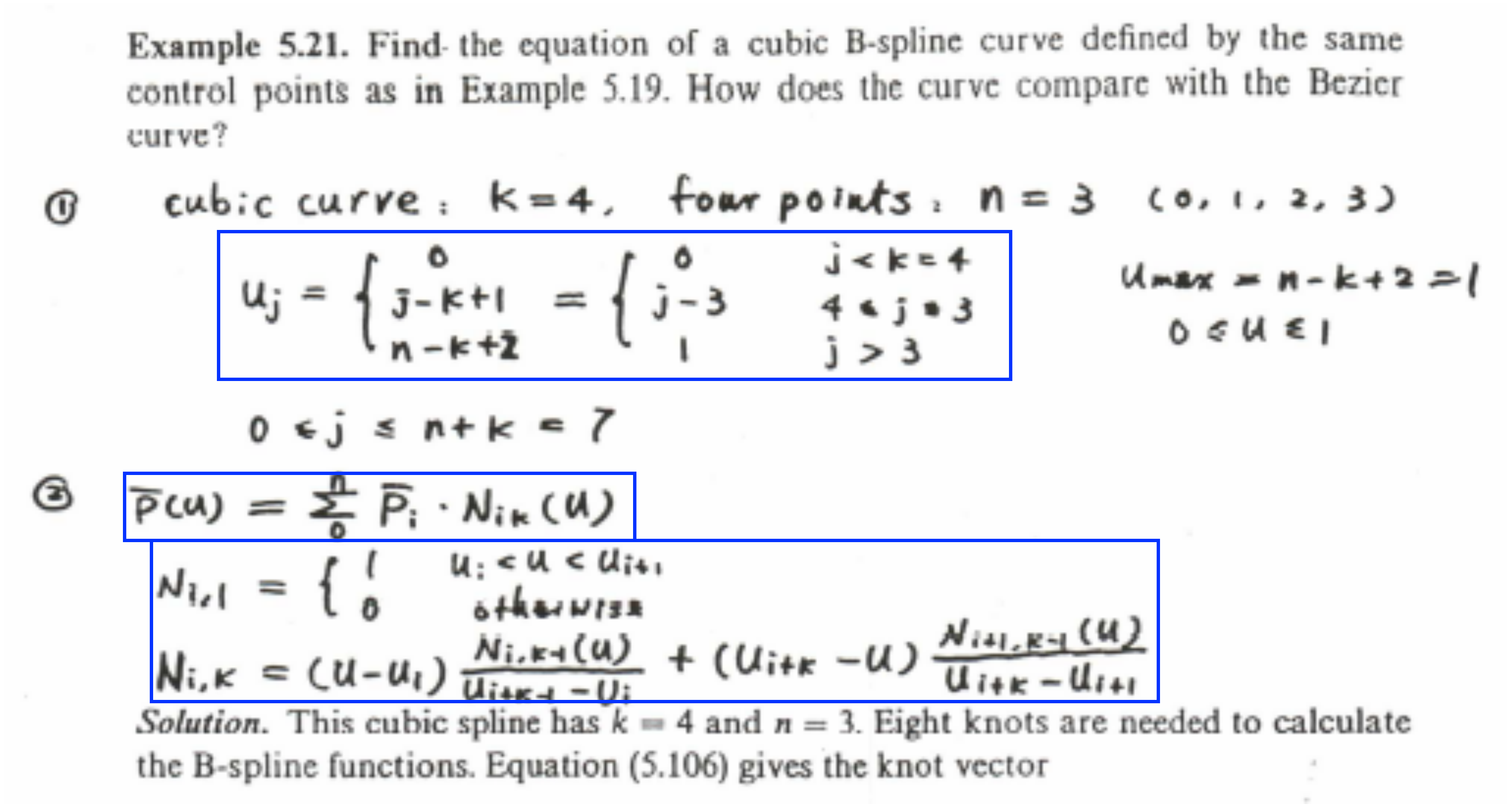
$$\mathbf{P}\left(\frac{1}{2}\right) = \frac{1}{8}\mathbf{P}_0 + \frac{3}{8}\mathbf{P}_1 + \frac{3}{8}\mathbf{P}_2 + \frac{1}{8}\mathbf{P}_3 = [2.5 \quad 2.75 \quad 0]^T$$

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

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

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.

Finding the B-Spline Curve for the Same Example Problem



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?

① cubic curve : $k=4$, four points : $n=3$ $(0, 1, 2, 3)$

$$u_j = \begin{cases} 0 \\ j-k+1 \\ n-k+2 \end{cases} = \begin{cases} 0 & j<k=4 \\ j-3 & 4\le j\le 3 \\ 1 & j>3 \end{cases}$$

$u_{max} = n-k+2 = 1$

$0 \le u \le 1$

$0 \le j \le n+k = 7$

③ $\bar{P}(u) = \sum_0^n \bar{P}_i \cdot N_{ik}(u)$

$$N_{i,1} = \begin{cases} 1 & u_i < u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

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

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

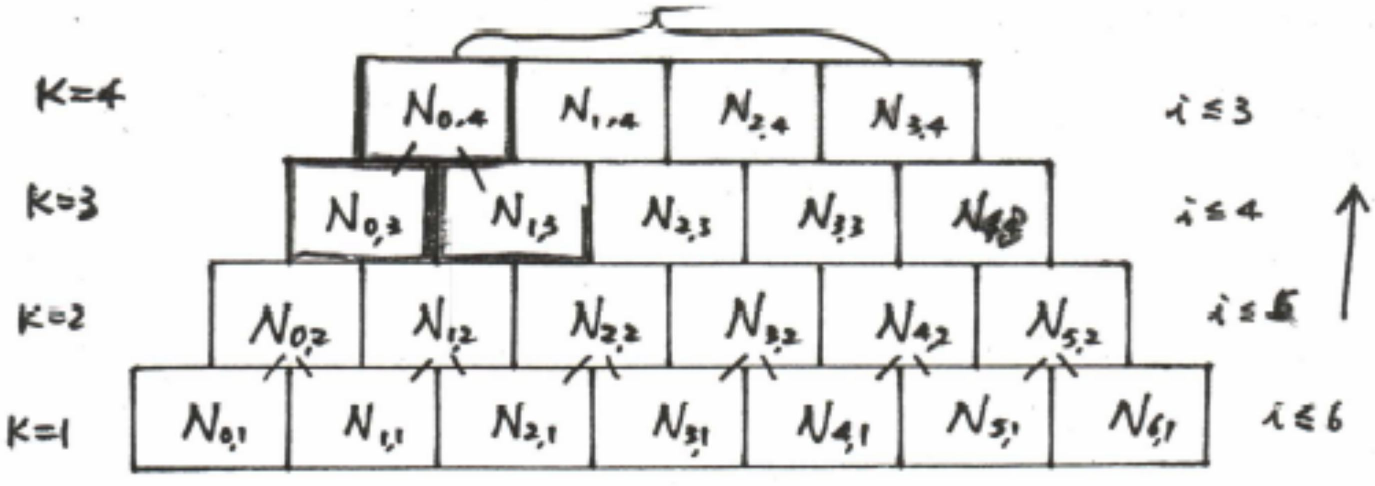
Values to be Calculated

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

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

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

$$\boxed{\underline{N_{i,k}} = f(N_{i,k-1}, N_{i+1,k-1})}$$



K=4
$N_{0,4}$ $N_{1,4}$ $N_{2,4}$ $N_{3,4}$
$i \le 3$
K=3
$N_{0,3}$ $N_{1,3}$ $N_{2,3}$ $N_{3,3}$ $N_{4,3}$
$i \le 4$
K=2
$N_{0,2}$ $N_{1,2}$ $N_{2,2}$ $N_{3,2}$ $N_{4,2}$ $N_{5,2}$
$i \le 5$
K=1
$N_{0,1}$ $N_{1,1}$ $N_{2,1}$ $N_{3,1}$ $N_{4,1}$ $N_{5,1}$ $N_{6,1}$
$i \le 6$

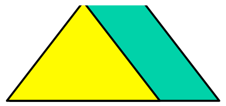
$n = 3$;   4 control points

$k = 4$; $4-1=3$ cubic polynomial

u_j: $0 \le j \le n+k = 7$

n increases – wider base

k increases – wider & taller



Calculating the Knots, u_j

cubic curve: $k=4$, four points: $n = 3$ $(0, 1, 2, 3)$

$$u_j = \begin{cases} 0 \\ j-k+1 \\ n-k+2 \end{cases} = \begin{cases} 0 \\ j-3 \\ 1 \end{cases} \quad \begin{array}{l} \underline{j < k = 4} \\ 4 \le j \le 3 \\ \underline{j > 3} \end{array}$$

$u_{max} = n-k+2 = 1$

$0 \le u \le 1$

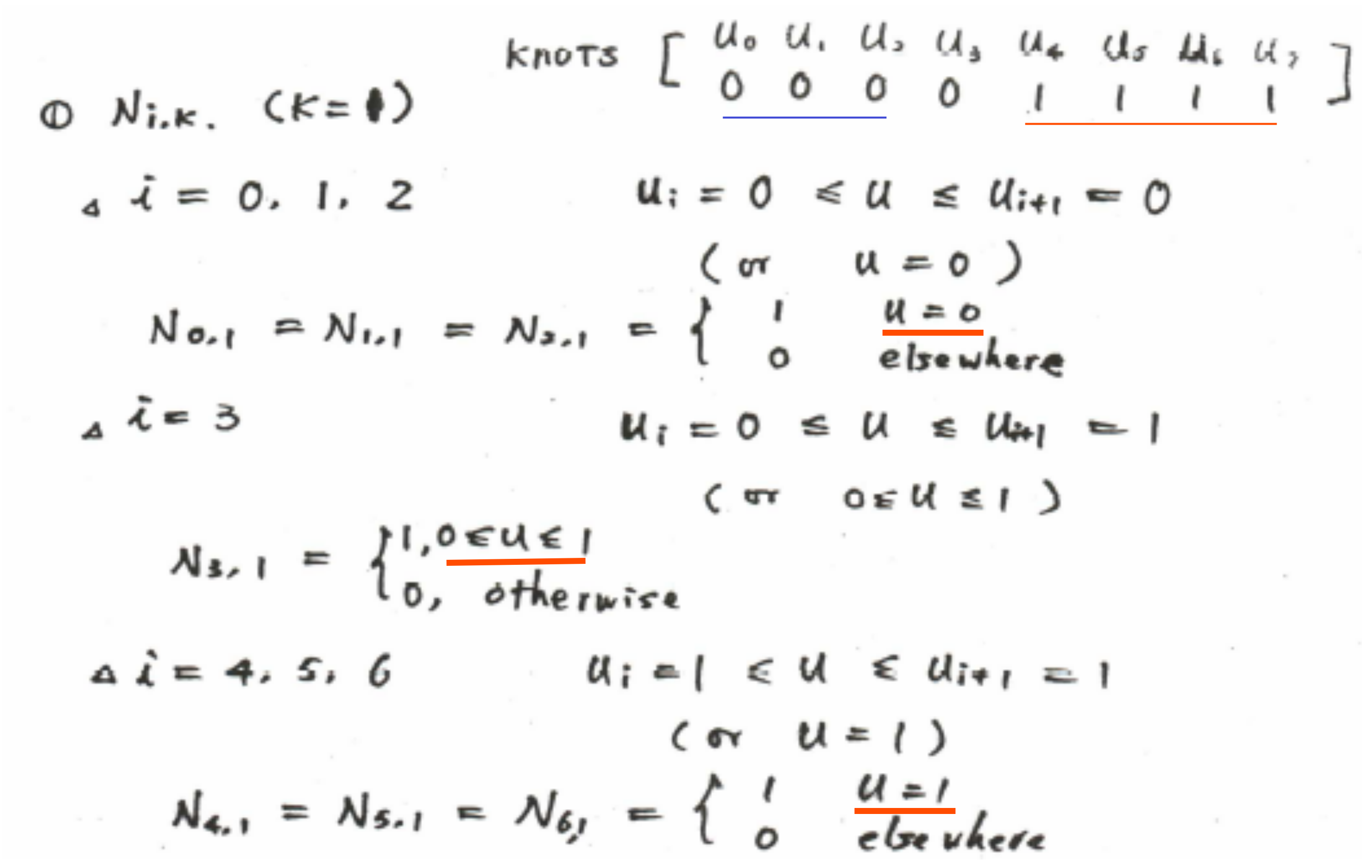
$0 \le j \le n+k = 7$

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

Calculating $N_{i,1}$

$$N_{i,1} = \begin{cases} 1 & u_i \le u < u_{i+1} \\ 1 & u = u_{\max} \text{ and } u \le u_{i+1} \text{ and } u - u_i = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{0}{0} = 0$$



KNOTS $\left[\begin{array}{cccccccc} u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 \\ \underline{0} & \underline{0} & \underline{0} & 0 & \underline{1} & \underline{1} & \underline{1} & \underline{1} \end{array}\right]$

① $N_{i,k}$. $(k=1)$

Δ $i = 0, 1, 2$ $\quad u_i = 0 \le u \le u_{i+1} = 0$

(or $u = 0$)

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

Δ $i = 3$ $\quad u_i = 0 \le u \le u_{i+1} = 1$

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

$$N_{3,1} = \begin{cases} 1, \underline{0 \le u \le 1} \\ 0, \text{ otherwise} \end{cases}$$

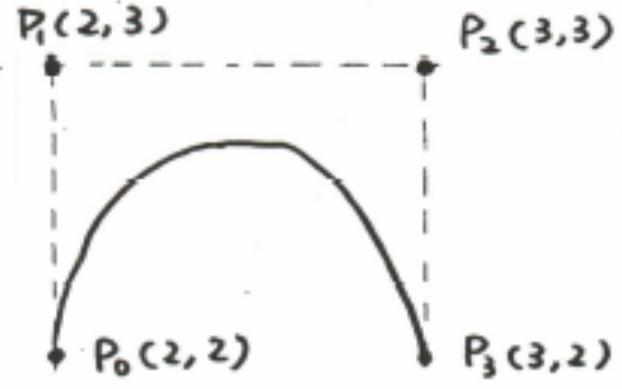
Δ $i = 4, 5, 6$ $\quad u_i = 1 \le u \le u_{i+1} = 1$

(or $u = 1$)

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

Calculating $N_{i,k}$

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



K=1
i+K≤7
(6)

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

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

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

K=2
i+K≤7
(5)

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

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

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

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

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

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

Calculating $N_{i,k}$

K=3

$i+K\le 7$

(4)

$$\left\{\begin{aligned}
N_{0,3} &= (u-u_0)\frac{N_{0,2}}{u_2-u_0} + (u_3-u)\frac{N_{1,2}}{u_3-u_1} = u\frac{0}{0} + (-u)\frac{0}{0} = 0\\
N_{1,3} &= (u-u_1)\frac{N_{1,2}}{u_3-u_1} + (u_4-u)\frac{N_{2,2}}{u_4-u_2} = u\frac{N_{1,2}}{0} + \frac{(1-u)N_{2,2}}{1} = (1-u)^2N_{3,1}\\
N_{2,3} &= (u-u_2)\frac{N_{2,2}}{u_4-u_2} + (u_5-u)\frac{N_{3,2}}{u_5-u_3} = uN_{2,2} + (1-u)N_{3,2} = 2u(1-u)N_{3,1}\\
N_{3,3} &= (u-u_3)\frac{N_{3,2}}{u_5-u_3} + (u_6-u)\frac{N_{4,2}}{u_6-u_4} = u^2N_{3,1} + (1-u)\frac{N_{4,2}}{0} = u^2N_{3,1}\\
N_{4,3} &= (u-u_4)\frac{N_{4,2}}{u_6-u_4} + (u_7-u)\frac{N_{5,2}}{u_7-u_5} = (u-1)\frac{N_{4,2}}{0} + (1-u)\frac{N_{5,2}}{0} = 0
\end{aligned}\right.$$

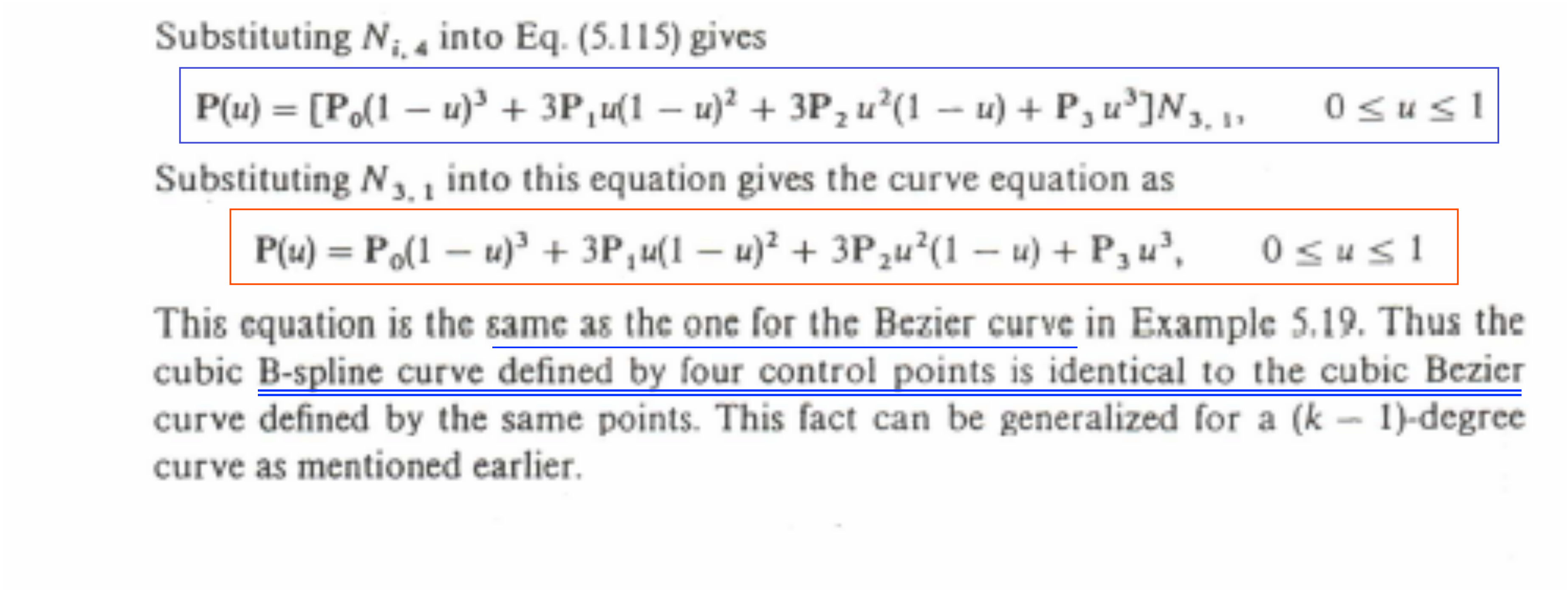
K=4

$i+K\le 7$

(3)

$$\left\{\begin{aligned}
N_{0,4} &= (u-u_0)\frac{N_{0,3}}{u_3-u_0} + (u_4-u)\frac{N_{1,3}}{u_4-u_1} = (1-u)^3N_{3,1}\\
N_{1,4} &= (u-u_1)\frac{N_{1,3}}{u_4-u_1} + (u_5-u)\frac{N_{2,3}}{u_5-u_2} = 3u(1-u)^2N_{3,1}\\
N_{2,4} &= (u-u_2)\frac{N_{2,3}}{u_5-u_2} + (u_6-u)\frac{N_{3,3}}{u_6-u_3} = 3u^2(1-u)N_{3,1}\\
N_{3,4} &= (u-u_3)\frac{N_{3,3}}{u_6-u_3} + (u_7-u)\frac{N_{4,3}}{u_7-u_4} = u^3N_{3,1}
\end{aligned}\right.$$

Result



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

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

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

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

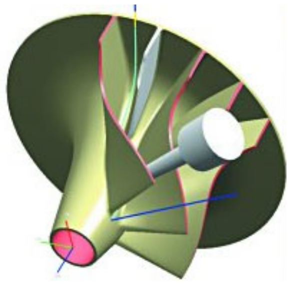
This equation is the same as the one for the Bezier curve in Example 5.19. Thus the cubic B-spline curve defined by four control points is identical to the cubic Bezier curve defined by the same points. This fact can be generalized for a $(k-1)$-degree curve as mentioned earlier.

n + 1 control points: 3+1=4

k – 1 degree curve: 4-1=3

4 control points – cubic polynomial

Non-Uniform Rational B-Spline Curve (NURBS)



Rational B-Spline

$$\boxed{\overline{P}(u)=\sum_{i=0}^{n}\overline{P}_i\times R_{i,k}(u) \qquad 0\le u\le u_{\max}}$$

$$R_{i,k}=\frac{h_i N_{i,k}(u)}{\sum_{i=0}^{n}h_i N_{i,k}(u)} \qquad (h_i - scalar)$$

If $h_i=1$, then $R_{i,k}(u)=N_{i,k}(u)$,it is the representation of a B-Spline curve.

Industry Standard Today!

h adds a degree of freedom to the curve, allowing to give more or less weight to each control point

$$x \cdot h = \sum_{i=0}^{n} (h_i \cdot x_i) N_{i,k}(u)$$

$$y \cdot h = \sum_{i=0}^{n} (h_i \cdot y_i) N_{i,k}(u)$$

$$z \cdot h = \sum_{i=0}^{n} (h_i \cdot z_i) N_{i,k}(u)$$

$$h = \sum_{i=0}^{n} h_i N_{i,k}(u)$$

then

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

What is NURBS ?

rational and B-spline.

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

Basis fun:

$$R_{i,k}(u) = \frac{h_i \, N_{i,k}(u)}{\sum_{i=0}^{n} h_i \, N_{i,k}(u)} \qquad (h_i \text{ - scalar},$$

a homogeneous coordinate vector $H = [h_0, h_1, \cdots h_n]^T$ is introduced and the basis function is defined by the algebraic ratio of two polynomials.

The curves are defined in the homogeneous space using homogeneous coordinates $(x^*, y^*, z^*, h)^T$

If $h_i = 1$, $R_{i,k}(u) = N_{i,k}(u)$

Development of NURBS

- Boeing: Tiger System in 1979
- SDRC: Geomod in 1993
- University of Utah: Alpha-1 in 1981
- Industry Standard: IGES, PHIGS, PDES, Pro/E, etc.

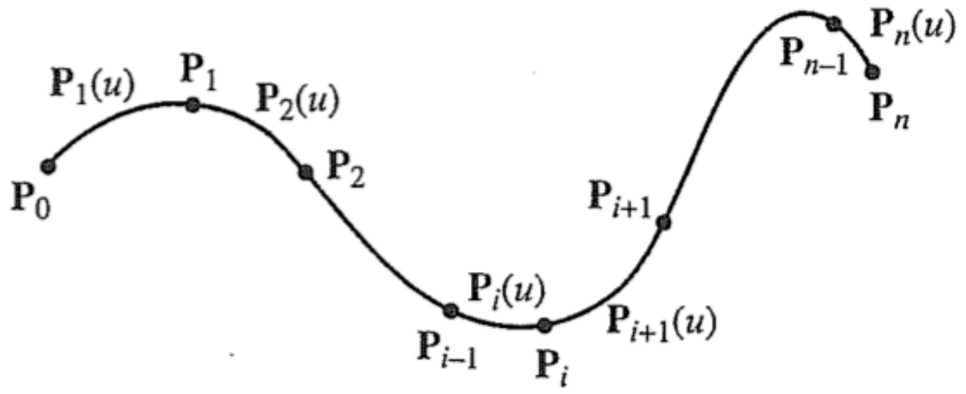
Advantages of NURBS

- Serve as a genuine generalizations of non-rational B-spline forms as well as rational and non-rational Bezier curves and surfaces
- Offer a common mathematical form for representing both standard analytic shapes (conics, quadratics, surface of revolution, etc) and free-from curves and surfaces precisely. B-splines can only approximate conic curves.
- Provide the flexibility to design a large variety of shapes by using control points and weights. increasing the weights has the effect of drawing a curve toward the control point.
- Have a powerful tool kit (knot insertion/refinement/removal, degree elevation, splitting, etc.
- Invariant under scaling, rotation, translation, and projections.
- Reasonably fast and computationally stable.
- Clear geometric interpretations

Interpolation Using Hermite Curves

The ith Hermite curve, $\mathbf{P}_i(u)$, can be expressed by using Equations (6.10) and (6.12) as follows:

$$\mathbf{P}_i(u) = \mathbf{P}_{i-1} + \mathbf{P}'_{i-1}u + [3(\mathbf{P}_i - \mathbf{P}_{i-1}) - 2\mathbf{P}'_{i-1} - \mathbf{P}'_i]u^2 + [2(\mathbf{P}_{i-1} - \mathbf{P}_i) + \mathbf{P}'_{i-1} + \mathbf{P}'_i]u^3 \tag{6.55}$$



where $\mathbf{P}'_{i-1}$ and $\mathbf{P}'_i$ are the tangent vectors at data points $\mathbf{P}_{i-1}$ and $\mathbf{P}_i$, respectively. We can obtain the same form of the Hermite curve equation for all $\mathbf{P}_i(u)$ by substituting specific values of i in the equation. The parameter value for every segment ranges from 0 to 1.

There is one problem with the form of Equation (6.55)—the coefficients $\mathbf{P}'_{i-1}$ and $\mathbf{P}'_i$ are not usually provided. Thus we must modify Equation (6.55) somehow so that $\mathbf{P}'_{i-1}$ and $\mathbf{P}'_i$ do not appear. To be able to evaluate the values of $\mathbf{P}'_{i-1}$ and $\mathbf{P}'_i$ from the given data points, we have to impose the following constraint equation to guarantee second-order continuity across the curve segments:

$$\left.\frac{d^2\mathbf{P}_i(u)}{du^2}\right]_{u=1} = \left.\frac{d^2\mathbf{P}_{i+1}(u)}{du^2}\right]_{u=0} \qquad (i = 1, 2, \ldots, n-1) \tag{6.56}$$

Substituting Equation (6.55) in Equation (6.56) gives

$$\begin{aligned}2(-3\mathbf{P}_{i-1} + 3\mathbf{P}_i - 2\mathbf{P}'_{i-1} - \mathbf{P}'_i) + 6(2\mathbf{P}_{i-1} - 2\mathbf{P}_i + \mathbf{P}'_{i-1} + \mathbf{P}'_{i+1}) &= \\ 2(-3\mathbf{P}_i + 3\mathbf{P}_{i+1} - 2\mathbf{P}'_i - \mathbf{P}'_{i+1})&\end{aligned} \tag{6.57}$$

Note that the first line in Equation (6.57) is obtained by differentiating Equation (6.55) twice and substituting $u = 1$ in the resulting equation. The second line is obtained by a similar procedure after we derive $\mathbf{P}_{i+1}(u)$ by substituting $i + 1$ for i in Equation (6.55). Rearranging Equation (6.57) gives

$$\mathbf{P}'_{i-1} + 4\mathbf{P}'_i + \mathbf{P}'_{i+1} = 3\mathbf{P}_{i+1} - 3\mathbf{P}_{i-1} \qquad (i = 1, 2, \ldots, n-1) \tag{6.58}$$

Then we can derive the following matrix equation by substituting the values of i from 1 to $(n-1)$ in Equation (6.58):

$$\begin{bmatrix} 4 & 1 & 0 & \cdot & \cdot & \cdot & & 0 & 0 \\ 1 & 4 & 1 & 0 & \cdot & \cdot & & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & \cdot & \cdot & & \\ & & & & & & & & \\ 0 & 0 & \cdot & \cdot & & & 1 & 4 & 1 \\ 0 & 0 & \cdot & \cdot & & & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} \mathbf{P}_1' \\ \mathbf{P}_2' \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{P}_{n-1}' \end{bmatrix} = \begin{bmatrix} 3\mathbf{P}_2 - 3\mathbf{P}_0 - \mathbf{P}_0' \\ 3\mathbf{P}_3 - 3\mathbf{P}_1 \\ 3\mathbf{P}_4 - 3\mathbf{P}_2 \\ \cdot \\ \cdot \\ 3\mathbf{P}_n - 3\mathbf{P}_{n-2} - \mathbf{P}_n' \end{bmatrix} \tag{6.59}$$

assuming that $\mathbf{P}_0''$ and $\mathbf{P}_n''$ are zeros at the ends because the second-order derivative is proportional to the bending moment in a simply supported beam. Thus the following extra constraint equations can be derived:

$$\left.\frac{d^2\mathbf{P}_1(u)}{du^2}\right]_{u=0} = -3\mathbf{P}_0 + 3\mathbf{P}_1 - 2\mathbf{P}_0' - \mathbf{P}_1' = 0 \tag{6.60}$$

$$\begin{aligned} \left.\frac{d^2\mathbf{P}_n(u)}{du^2}\right]_{u=1} &= 2[3(\mathbf{P}_n - \mathbf{P}_{n-1}) - 2\mathbf{P}_{n-1}' - \mathbf{P}_n'] \\ &\quad + 6[2(\mathbf{P}_{n-1} - \mathbf{P}_n) + \mathbf{P}_{n-1}' + \mathbf{P}_n'] = 0 \end{aligned} \tag{6.61}$$

Rearranging Equations (6.60) and (6.61) yields

$$2\mathbf{P}'_0 + \mathbf{P}'_1 = 3\mathbf{P}_1 - 3\mathbf{P}_0 \tag{6.62}$$

$$2\mathbf{P}'_n + \mathbf{P}'_{n-1} = 3\mathbf{P}_n - 3\mathbf{P}_{n-1} \tag{6.63}$$

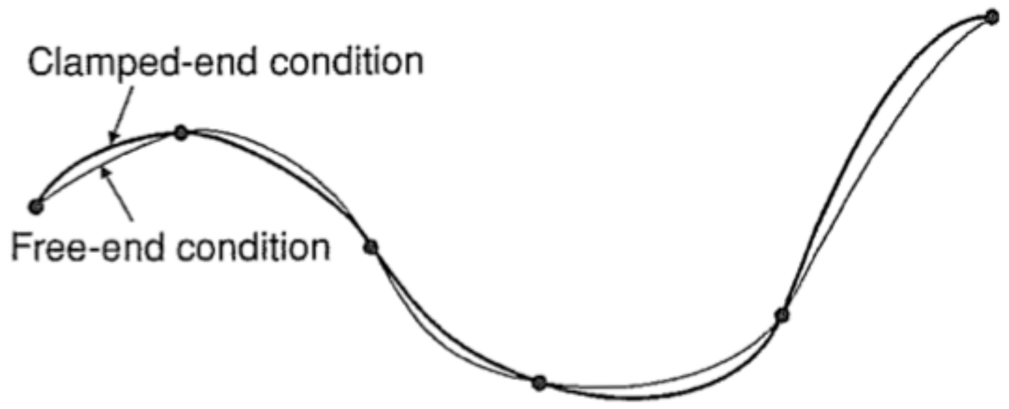
We rewrite Equation (6.59), moving $\mathbf{P}'_0$ and $\mathbf{P}'_n$ from the right-hand side into the variable vector on the left-hand side and adding Equations (6.62) and (6.63) at the beginning and end of the equations in Equation (6.59). Then the following equation is derived.

$$\begin{bmatrix} 2 & 1 & 0 & \cdot & \cdot & \cdot & & 0 & 0 \\ 1 & 4 & 1 & 0 & \cdot & \cdot & & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & \cdot & \cdot & & \\ & & & & & & & & \\ 0 & 0 & \cdot & \cdot & & 0 & 1 & 4 & 1 \\ 0 & 0 & \cdot & \cdot & & & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} \mathbf{P}'_0 \\ \mathbf{P}'_1 \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{P}'_n \end{bmatrix} = \begin{bmatrix} 3\mathbf{P}_1 - 3\mathbf{P}_0 \\ 3\mathbf{P}_2 - 3\mathbf{P}_0 \\ 3\mathbf{P}_3 - 3\mathbf{P}_1 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \tag{6.64}$$

We can solve Equation (6.64) for $(n + 1)$ unknowns, $\mathbf{P}'_0$, $\mathbf{P}'_1$, . . ., $\mathbf{P}'_n$, from $(n + 1)$ equations.

Clamped or Free Ends

Figure 6.12 illustrates two different interpolation curves obtained from the same data points: one under the clamped-end condition and the other under the free-end condition. Note that the interpolation curve under the free-end condition tends to be flat near the ends.



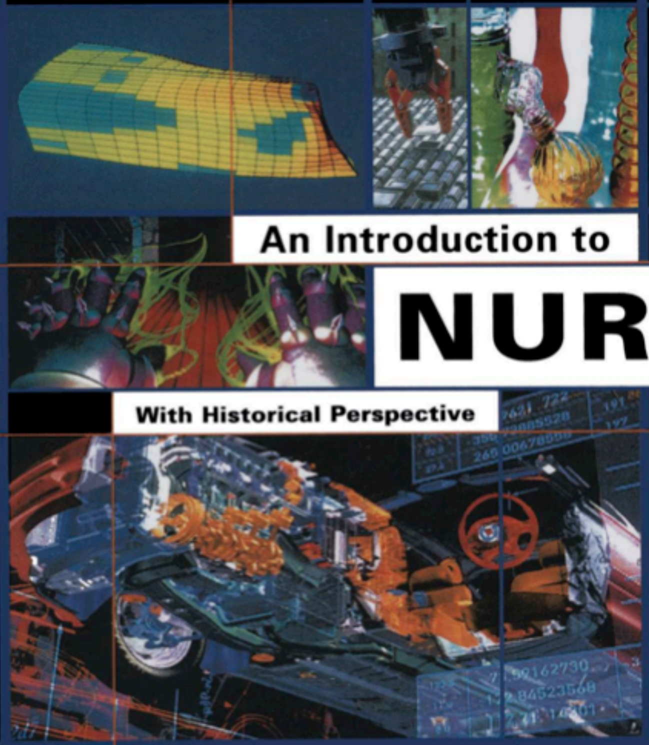
Disadvantages of Cubic Splines

- The order of the curve is always constant regardless of the number of data points. In order to increase the flexibility of the curve, more points must be provided, thus creating more spline segments which are still of cubic order.
- The control of the curve is through the change of the positions of data points or the end slope change. The global control characteristics is not intuitive.

An Introduction to

NURBS

With Historical Perspective



David F. Rogers