Computer Aided Ship Design

Part II. Curve and Surface Modeling

Ch. 3 B-Spline Curves

September, 2013
Prof. Myung-Il Roh

Department of Naval Architecture and Ocean Engineering,
Seoul National University of College of Engineering

SYstem Design Laboratory

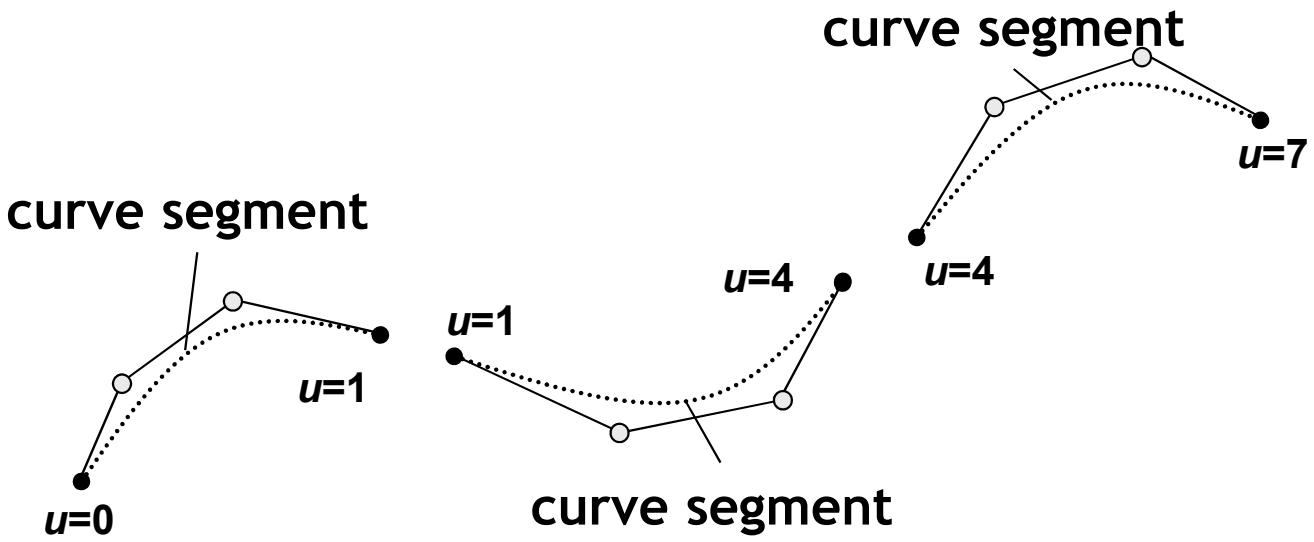
Ch. 3 B[asis]-Spline Curves

3.1 Introduction to B-Spline Curves
3.2 B-Spline Basis Function
3.3 C^1 and C^2 Continuity Condition
3.4 B-Spline Curve Interpolation
3.5 de Boor Algorithm
3.6 Cox-de Boor Algorithm

SYstem Design Laboratory

3.1 Introduction to B-Spline Curves

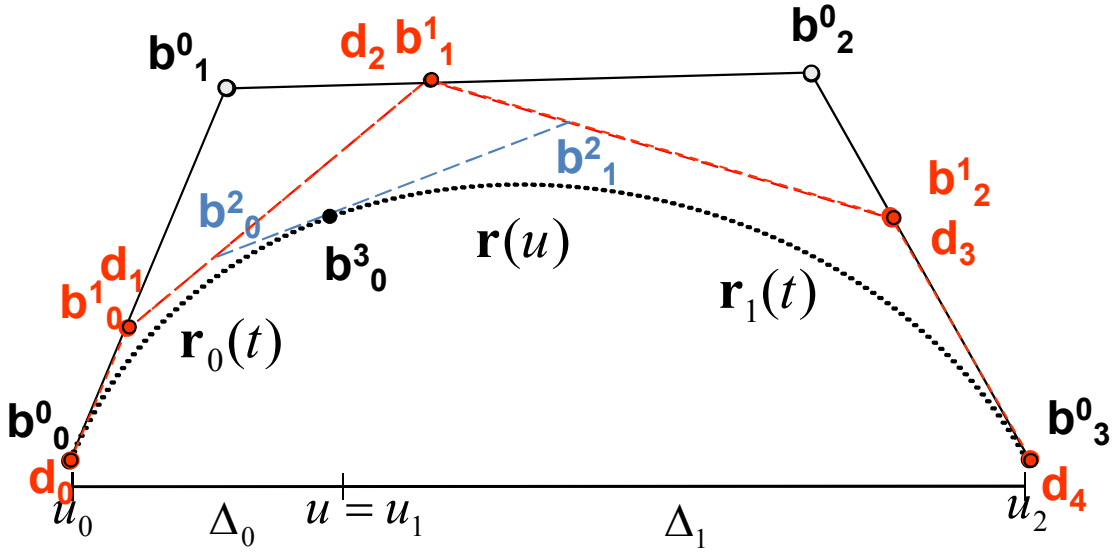
'Smooth' Connection of Separate Curve Segments at Knots - Spline Curve



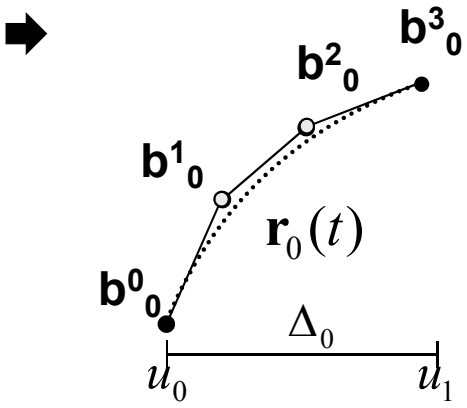
Knot = {..., 0, 1, 4, 7, ...}

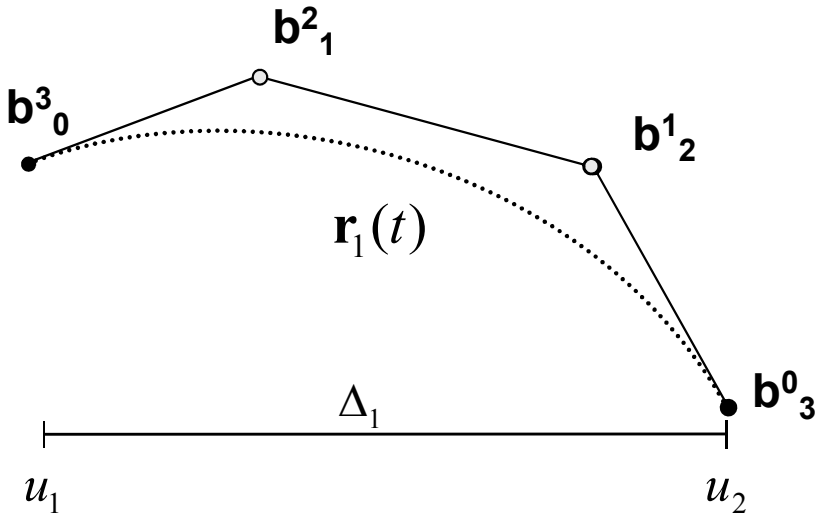
- A curve is "smoothly" connected with curve segments: spline curve
- Curve segments are tied by knots: knot

Point on the Bezier Curve
- Connected Two Bezier Curves at the 'Knot' (1/2)



In contrast with dividing one curve into two Bezier curves, we can imagine as if two Bezier curves $r_0(t)$ and $r_1(t)$ were connected at $u=u_1$. Here, u_1 is called 'knot', that means the knot ties the curves.





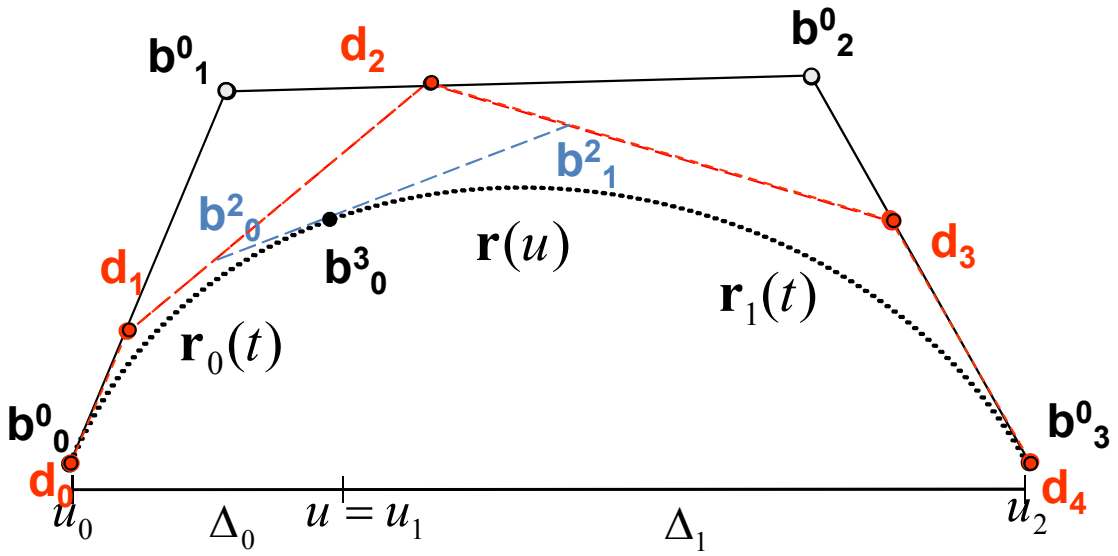
$$\mathbf{r}_0(t) = (1-t)^3\mathbf{b}_0^0 + 3(1-t)^2 t\mathbf{b}_0^1 + 3(1-t)t^2\mathbf{b}_0^2 + t^3\mathbf{b}_0^3,$$

$$t = \frac{u-u_0}{u_1-u_0}$$

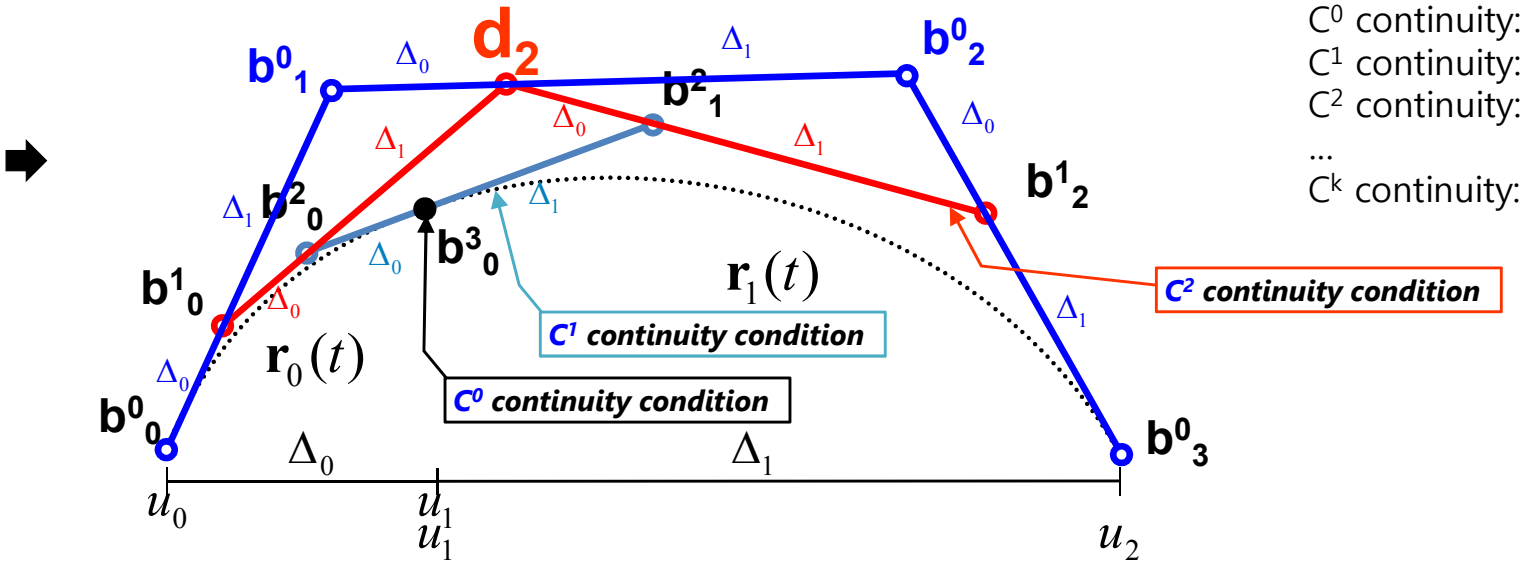
$$\mathbf{r}_1(t) = (1-t)^3\mathbf{b}_0^3 + 3(1-t)^2 t\mathbf{b}_1^2 + 3(1-t)t^2\mathbf{b}_2^1 + t^3\mathbf{b}_3^0,$$

$$t = \frac{u-u_1}{u_2-u_1}$$

Point on the Bezier Curve
- Connected Two Bezier Curves at the 'Knot' (2/2)



At $u=u_1$, the curves obviously satisfy C^0, C^1, and C^2 continuity conditions at least, because the curve r(u) was originally one curve.



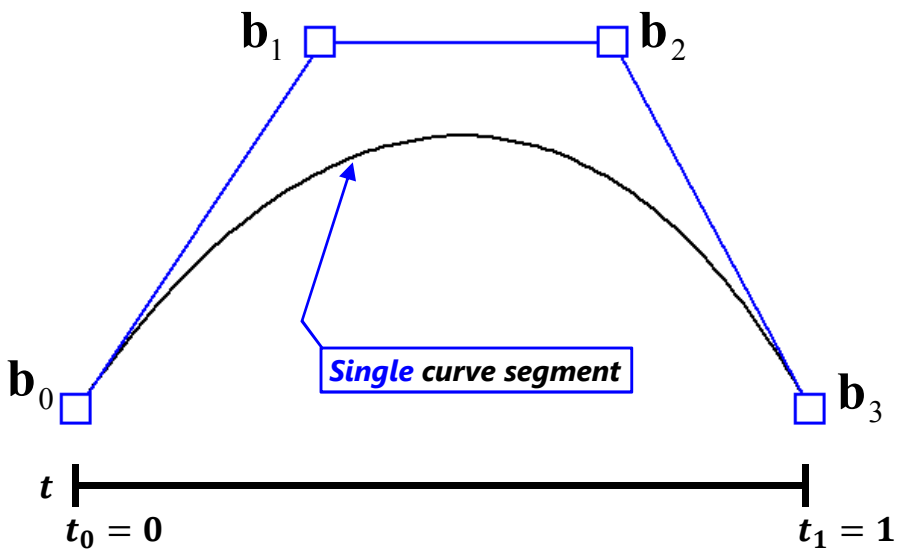
C^0 continuity: Continuity of the 0th derivatives
C^1 continuity: Continuity of the 1st derivatives
C^2 continuity: Continuity of the 2nd derivatives
...
C^k continuity: Continuity of the kth derivatives

$$\mathbf{r}_0(t) = (1-t)^3\mathbf{b}_0^0 + 3(1-t)^2 t\mathbf{b}_0^1 + 3(1-t)t^2\mathbf{b}_0^2 + t^3\mathbf{b}_0^3, \qquad t = \frac{u-u_0}{u_1-u_0}$$

$$\mathbf{r}_1(t) = (1-t)^3\mathbf{b}_0^3 + 3(1-t)^2 t\mathbf{b}_1^2 + 3(1-t)t^2\mathbf{b}_2^1 + t^3\mathbf{b}_3^0, \qquad t = \frac{u-u_1}{u_2-u_1}$$

Definition of B-Spline Curves

Cubic Bezier Curve



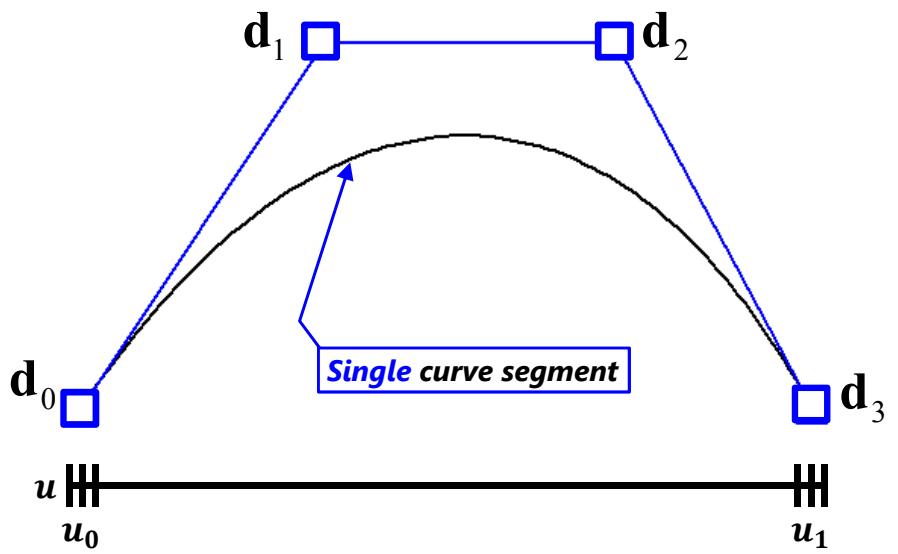
Given: $\mathbf{b}_0$, $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$, and t

Find: Points on the curve at parameter *t*

$$\mathbf{r}(t) = \mathbf{b}_0 B_0^3(t) + \mathbf{b}_1 B_1^3(t) + \mathbf{b}_2 B_2^3(t) + \mathbf{b}_3 B_3^3(t)$$

Bernstein basis function

Cubic B-spline Curve



Given: $\mathbf{d}_0$, $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, and u

Find: Points on the curve at parameter *u*

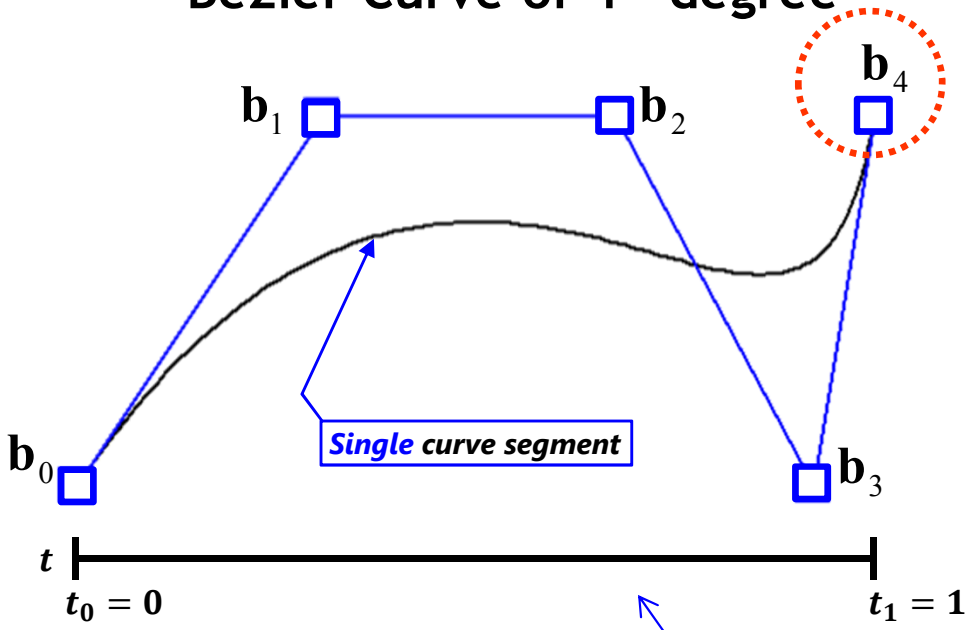
$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

B-spline basis function
(Cox-de Boor recurrence formula)

What happens if one more control point is included?

Property of Bezier Curves and B-Spline Curves in Increasing(Changing) the Number of the Control Points

Bezier Curve of 4th degree



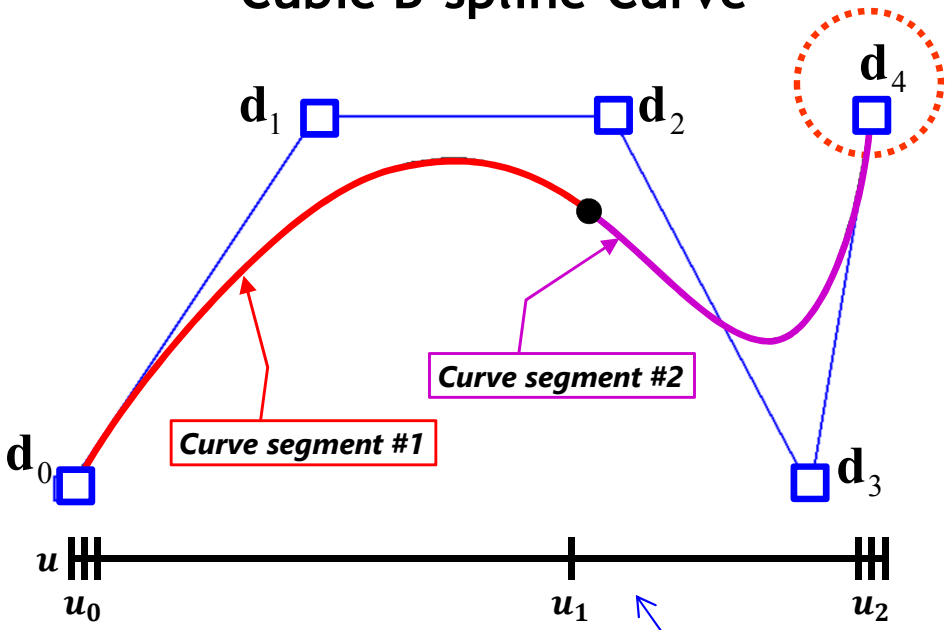
Given: $\mathbf{b}_0$, $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$, $\mathbf{b}_4$, and t

Find: Points on the curve at parameter t

$$\mathbf{r}(t) = \mathbf{b}_0 B_0^4(t) + \mathbf{b}_1 B_1^4(t) + \mathbf{b}_2 B_2^4(t) + \mathbf{b}_3 B_3^4(t) + \mathbf{b}_4 B_4^4(t)$$

For the Bezier curve, if the number of the control points increases, the degree of the Bezier curve will also increase.
(Cubic Bezier curve ➡ 4th-degree Curve)

Cubic B-spline Curve



Given: $\mathbf{d}_0$, $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, and u

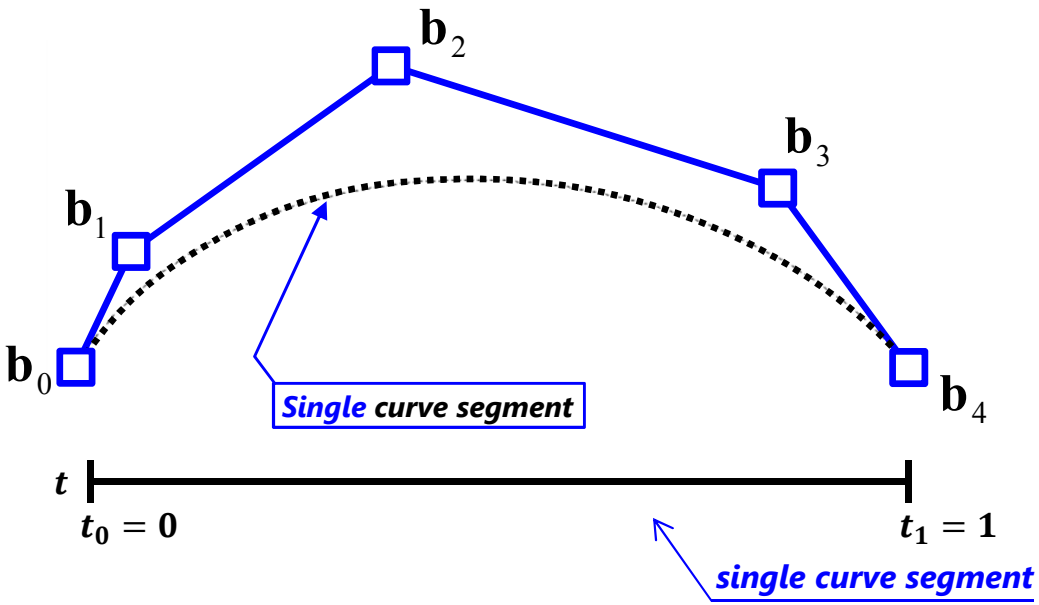
Find: Points on the curve at parameter u

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u)$$

For the B-spline curve, if the number of the control points increases,
the degree of the curve does not change but additional one Bezier curve of the 3rd degree is generated.

Properties of Bezier Curves and B-Spline Curves with Same Control Points

Bezier Curve of 4th degree

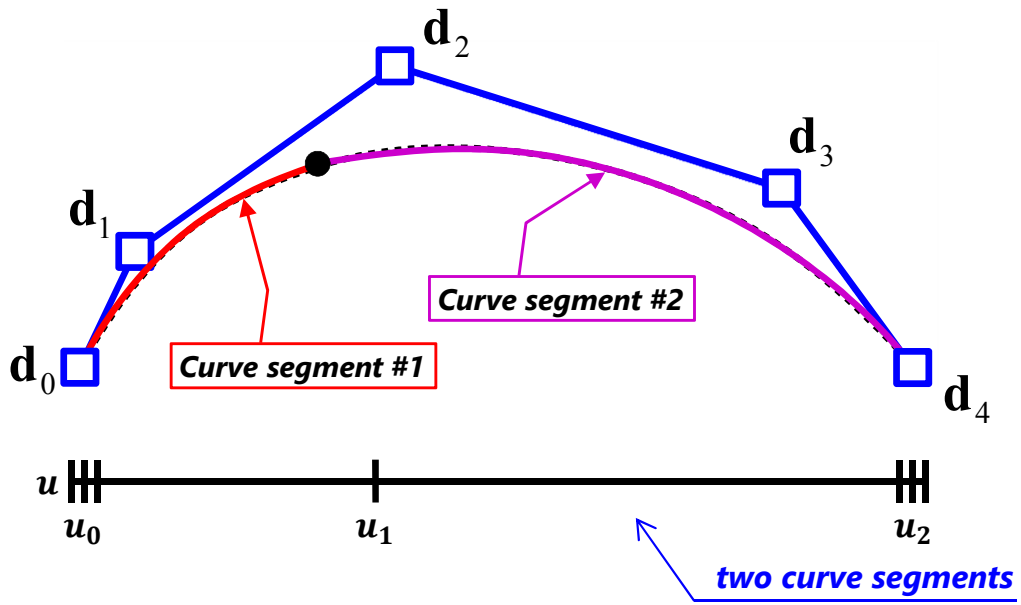


Given: $\mathbf{b}_0$, $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$, $\mathbf{b}_4$, and t

Find: Points on the curve at parameter *t*

$$\mathbf{r}(t) = \mathbf{b}_0 B_0^4(t) + \mathbf{b}_1 B_1^4(t) + \mathbf{b}_2 B_2^4(t) + \mathbf{b}_3 B_3^4(t) + \mathbf{b}_4 B_4^4(t)$$

Cubic B-spline Curve



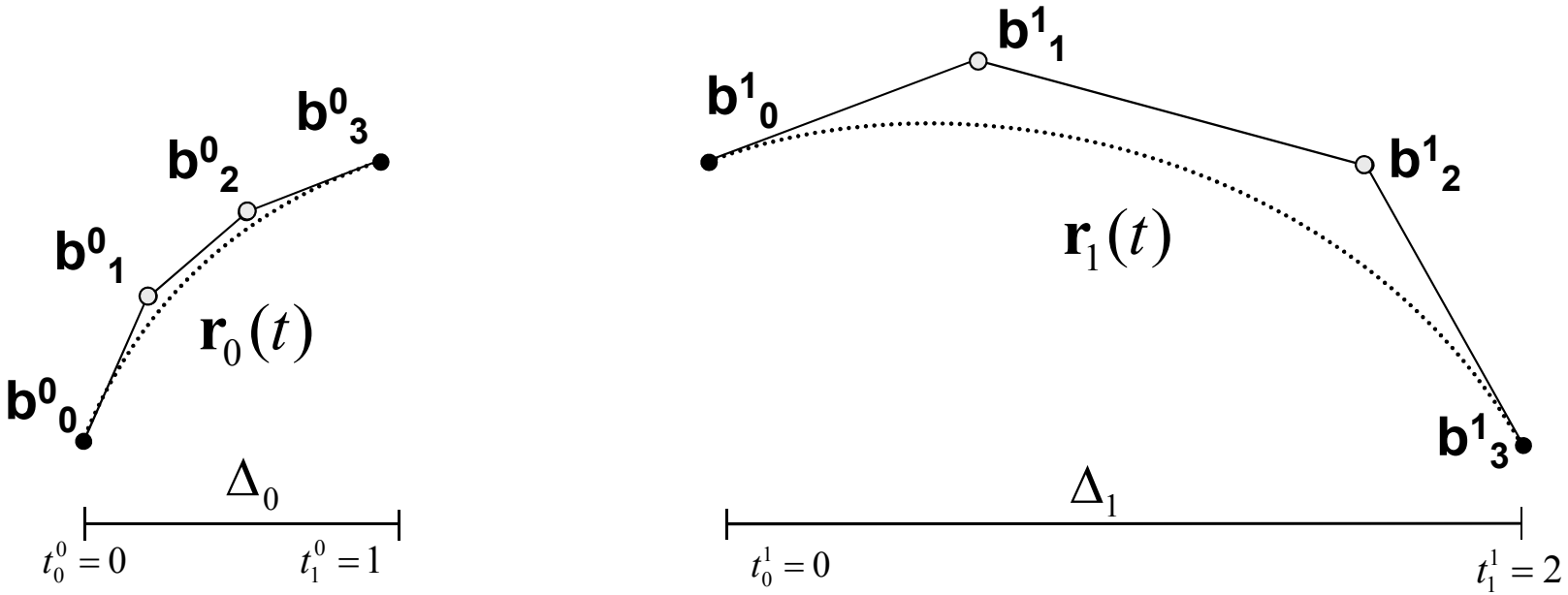
Given: $\mathbf{d}_0$, $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$, $\mathbf{d}_4$, and u

Find: Points on the curve at parameter *u*

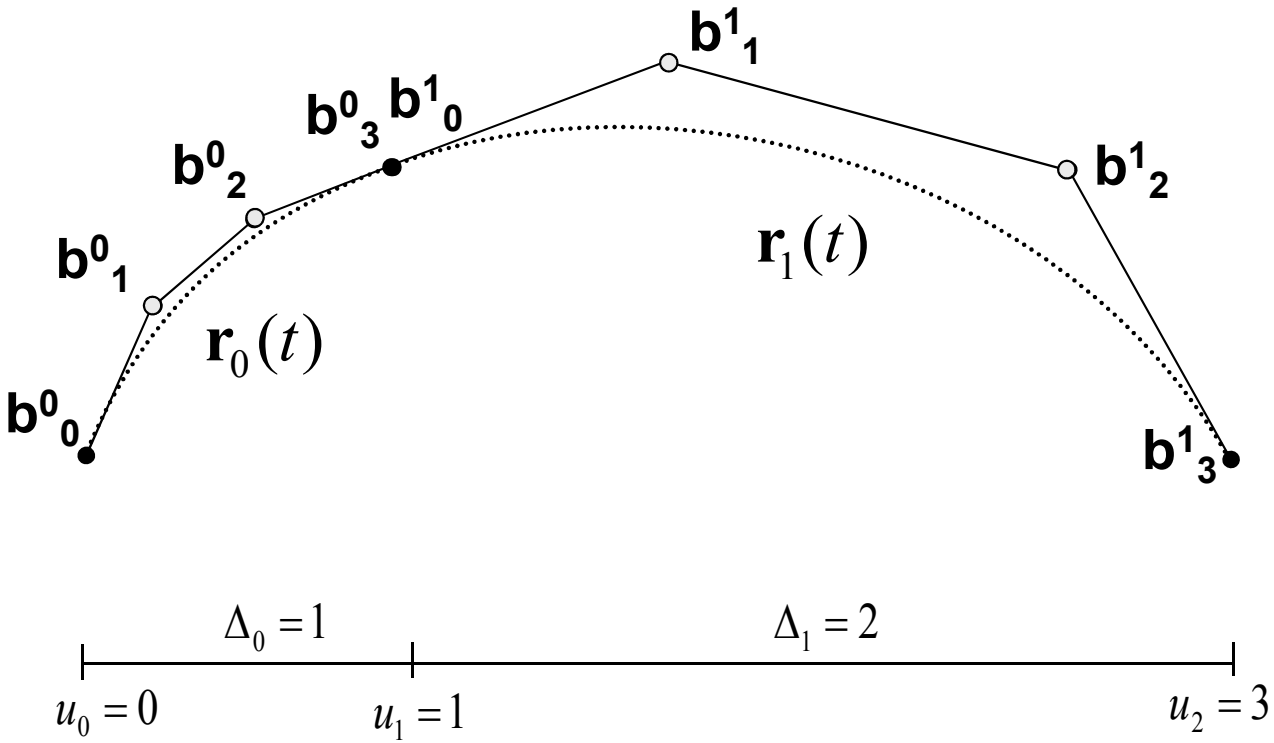
$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u)$$

Geometric Meanings of B-Spline Curve (1/4)

☑ A 'cubic' B-spline curve is composed of several 'cubic' Bezier curves, which are connected with the C^2 continuity condition (condition of continuous 2nd derivatives).



Geometric Meanings of B-Spline Curve (2/4)



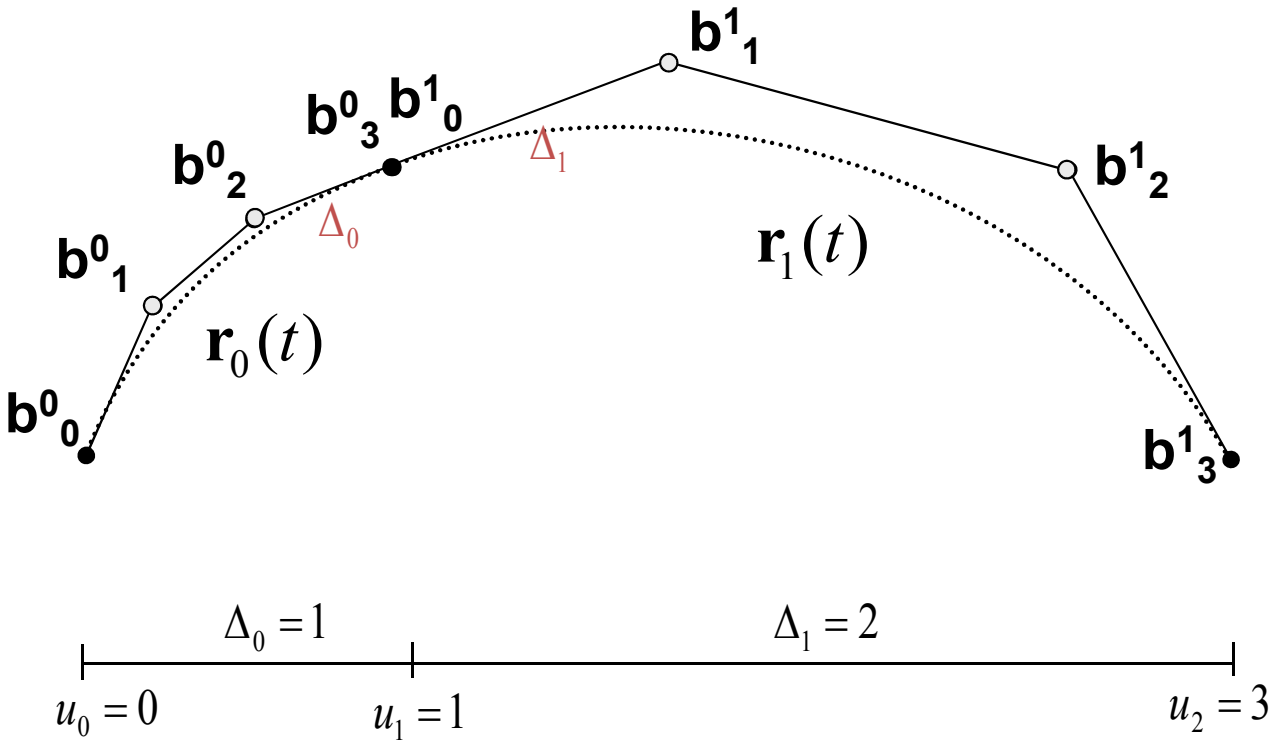
Assign new global parameter ***u*** to the connected curves.

C^0 continuity condition (coincident position)

$$\mathbf{b}_3^0 = \mathbf{b}_0^1$$

Geometric Meanings of B-Spline Curve (3/4)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Assign new global parameter *u* to the connected curves

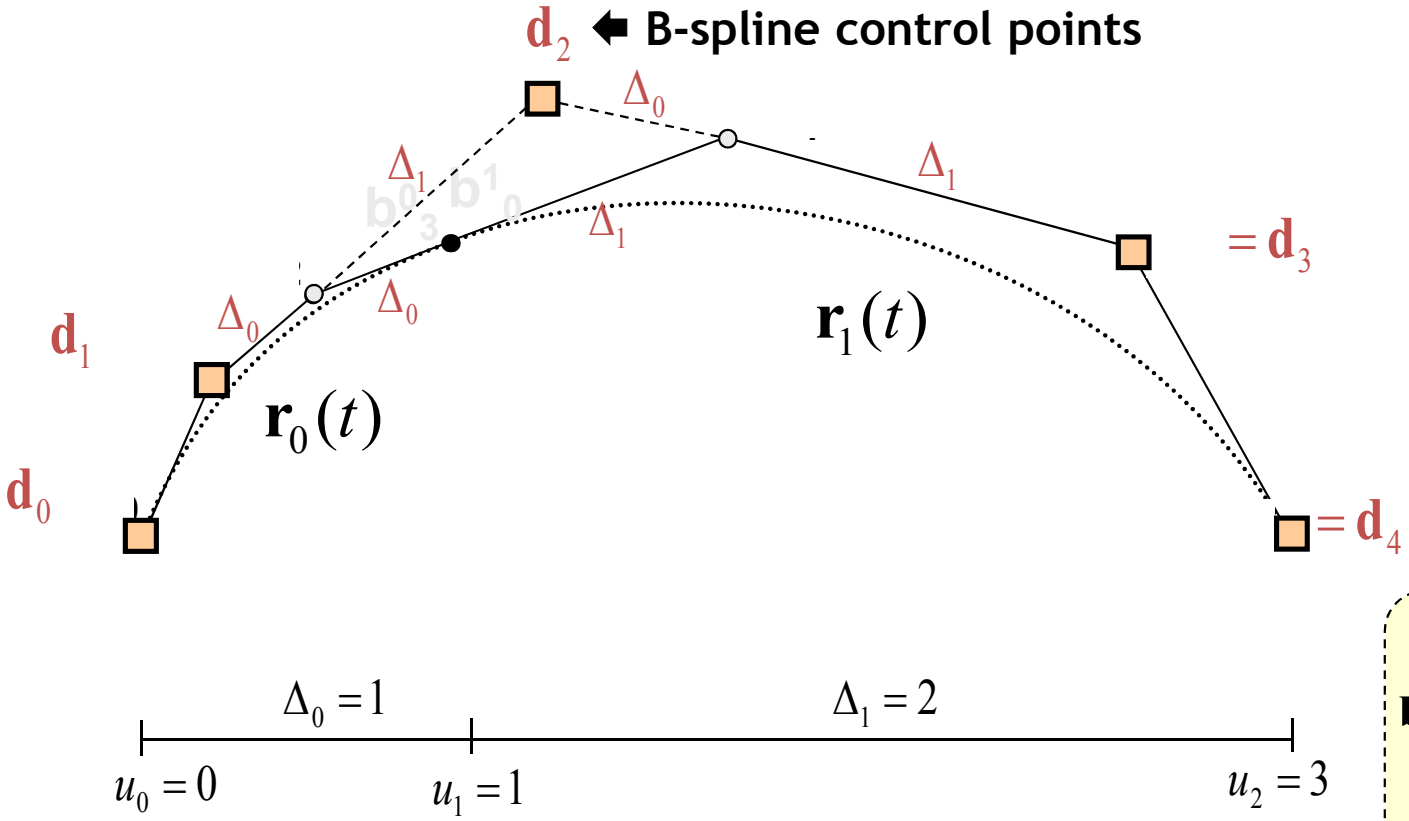
C^1 continuity condition

$$\mathbf{b}_3^0 = \mathbf{b}_0^1$$

$$\mathbf{b}_3^0 = \mathbf{b}_0^1 = \frac{\Delta_1}{\Delta_0 + \Delta_1}\mathbf{b}_2^0 + \frac{\Delta_0}{\Delta_0 + \Delta_1}\mathbf{b}_1^1$$

Geometric Meanings of B-Spline Curve (4/4)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Assign new global parameter *u* to the connected curves

C^2 continuity condition

$$\mathbf{b}_3^0 = \mathbf{b}_0^1$$

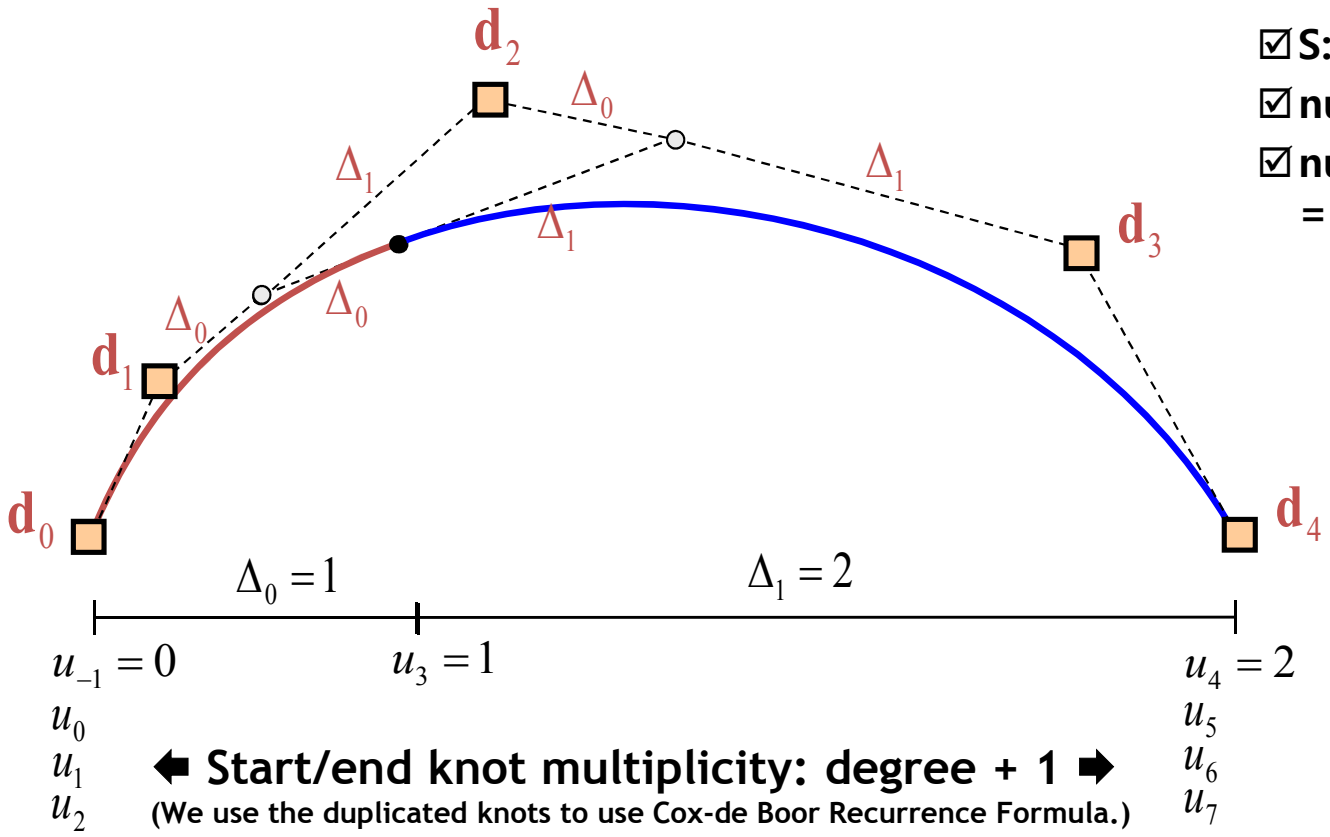
$$\mathbf{b}_3^0 = \mathbf{b}_0^1 = \frac{\Delta_1}{\Delta_0 + \Delta_1}\mathbf{b}_2^0 + \frac{\Delta_0}{\Delta_0 + \Delta_1}\mathbf{b}_1^1$$

$$\mathbf{b}_2^0 = \{\frac{\Delta_1}{\Delta_0 + \Delta_1}\}\mathbf{d}_1 + \{\frac{\Delta_0}{\Delta_0 + \Delta_1}\}\mathbf{d}_2$$

$$\mathbf{b}_1^1 = \{\frac{\Delta_1}{\Delta_0 + \Delta_1}\}\mathbf{d}_2 + \{\frac{\Delta_0}{\Delta_0 + \Delta_1}\}\mathbf{d}_3$$

Definition of B-Spline Curve
- Degree, Control Point, Knot, Curve Segments

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



- ☑ n: degree
- ☑ S: number of Bezier curve segments
- ☑ number of knot = (S-1)+2(n+1)
- ☑ number of control points = (n+1)+(S-1)

⬅ Start/end knot multiplicity: degree + 1 ➡

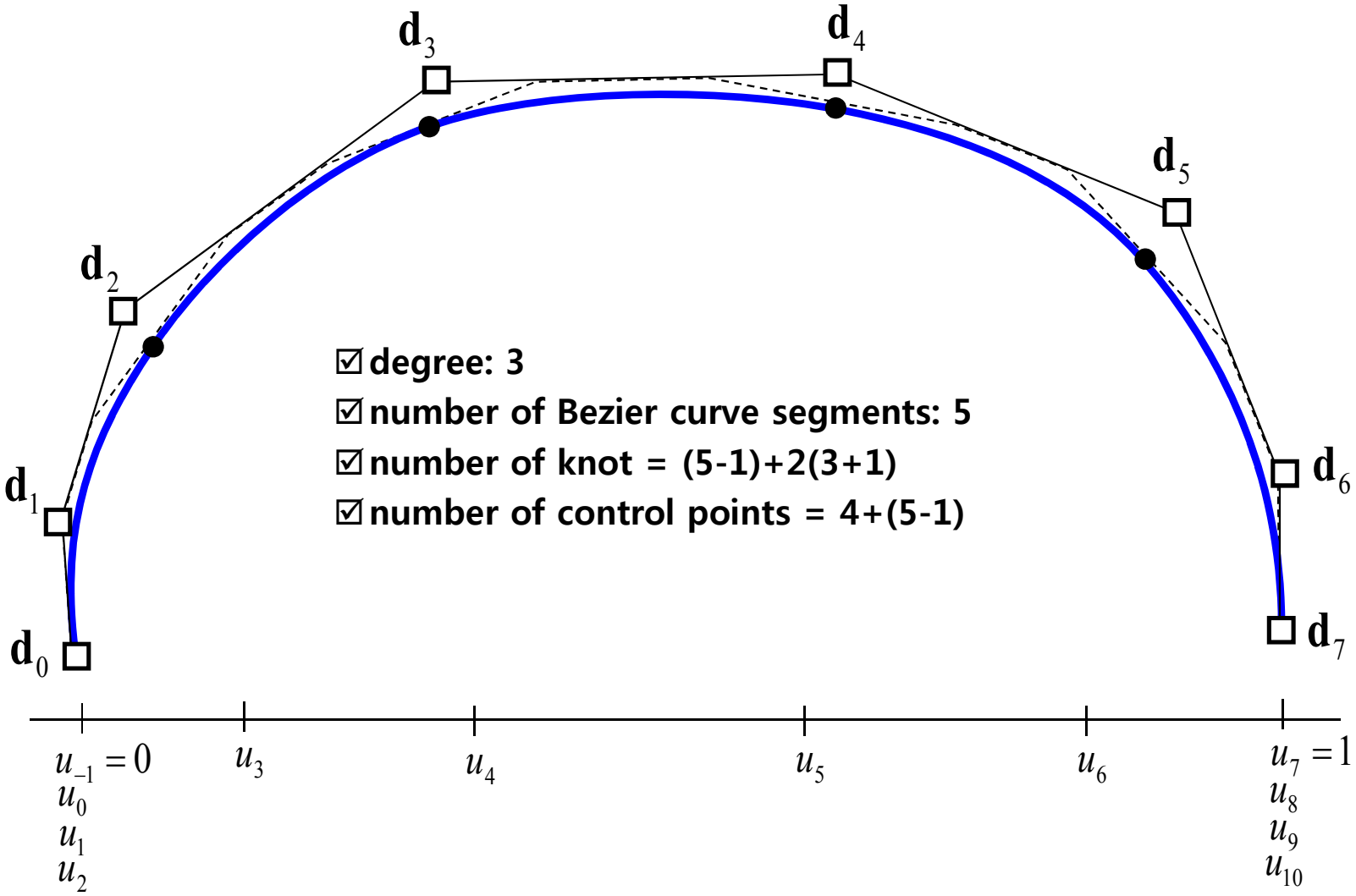
(We use the duplicated knots to use Cox-de Boor Recurrence Formula.)

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u)$$

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u) \qquad N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}, \quad \sum_{i=0}^{D-1} N_i^n(u) = 1$$

Example of Cubic B-Spline Curve with Eight Control Points

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



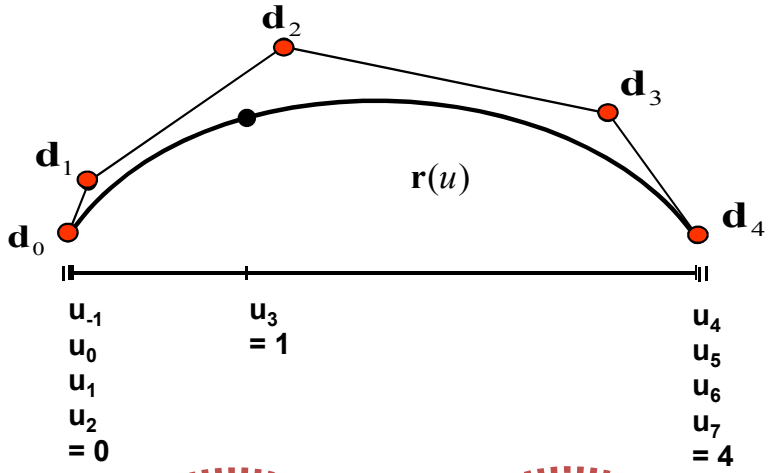
$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$

3.2 B-Spline Basis Function (Cox-de Boor Recurrence Formula)

Cox-de Boor Recurrence Formula (B-Spline Function)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$

☑ Example: Cubic B-spline curve



$$\mathbf{r}(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} = \sum_{i=0}^{D-1} \mathbf{d}_i N_i^n(u)$$

$$= \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u)$$

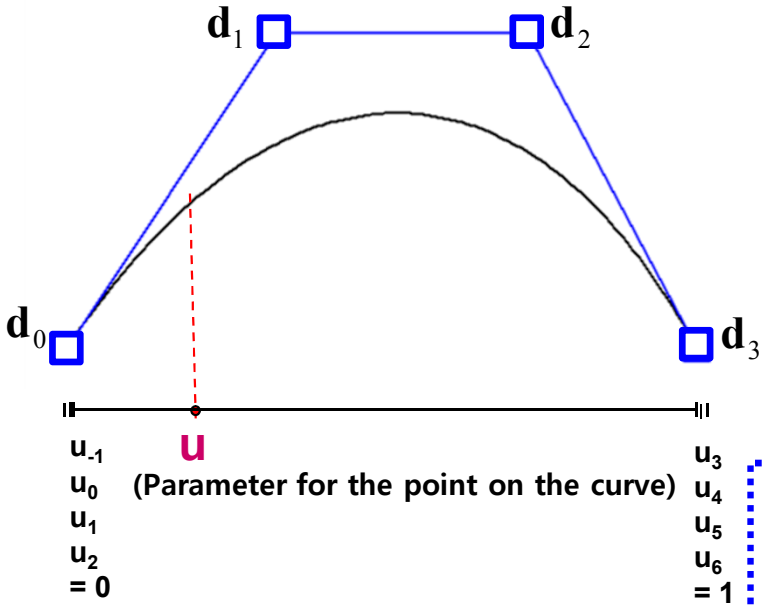
- Cox-de Boor Recurrence Formula (B-spline function)

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

[Example] B-Spline Function of One Curve Segment (1/10)

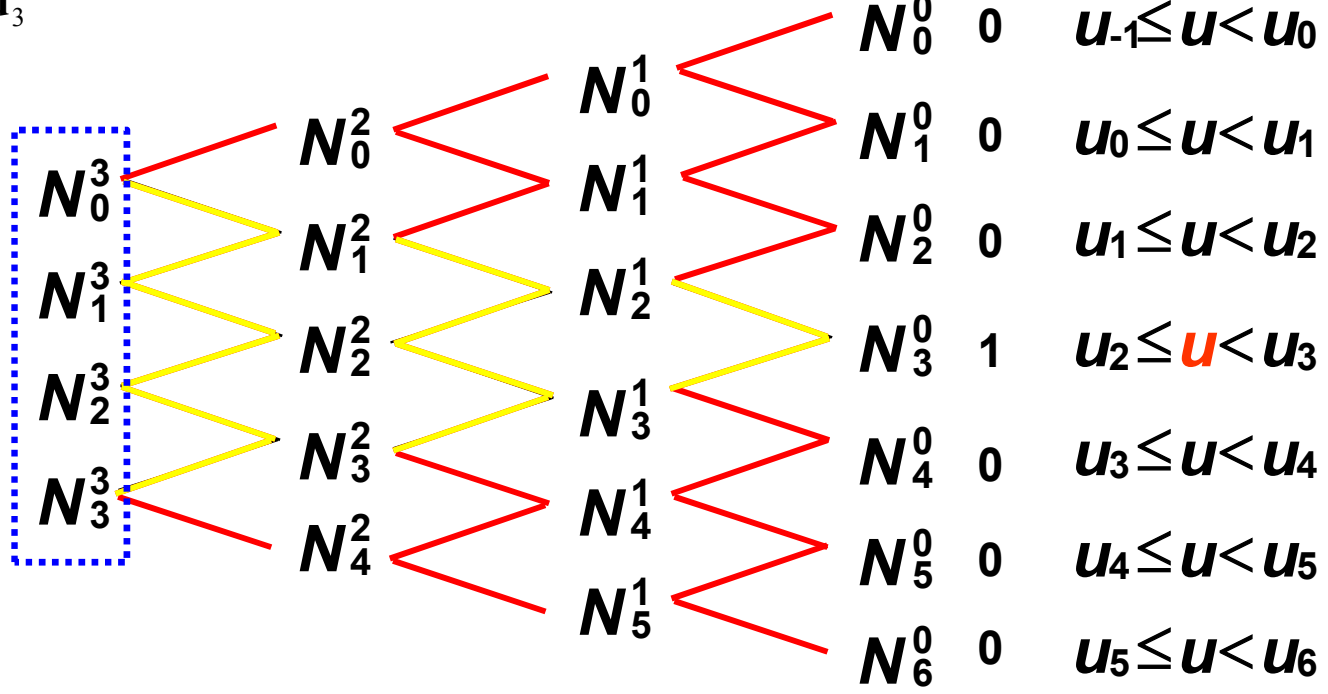
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

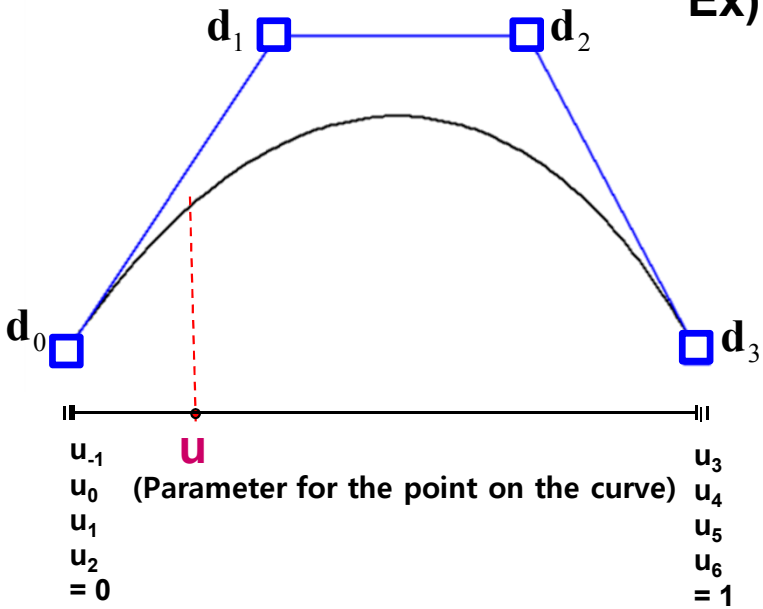
$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



[Example] B-Spline Function of One Curve Segment (2/10)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$

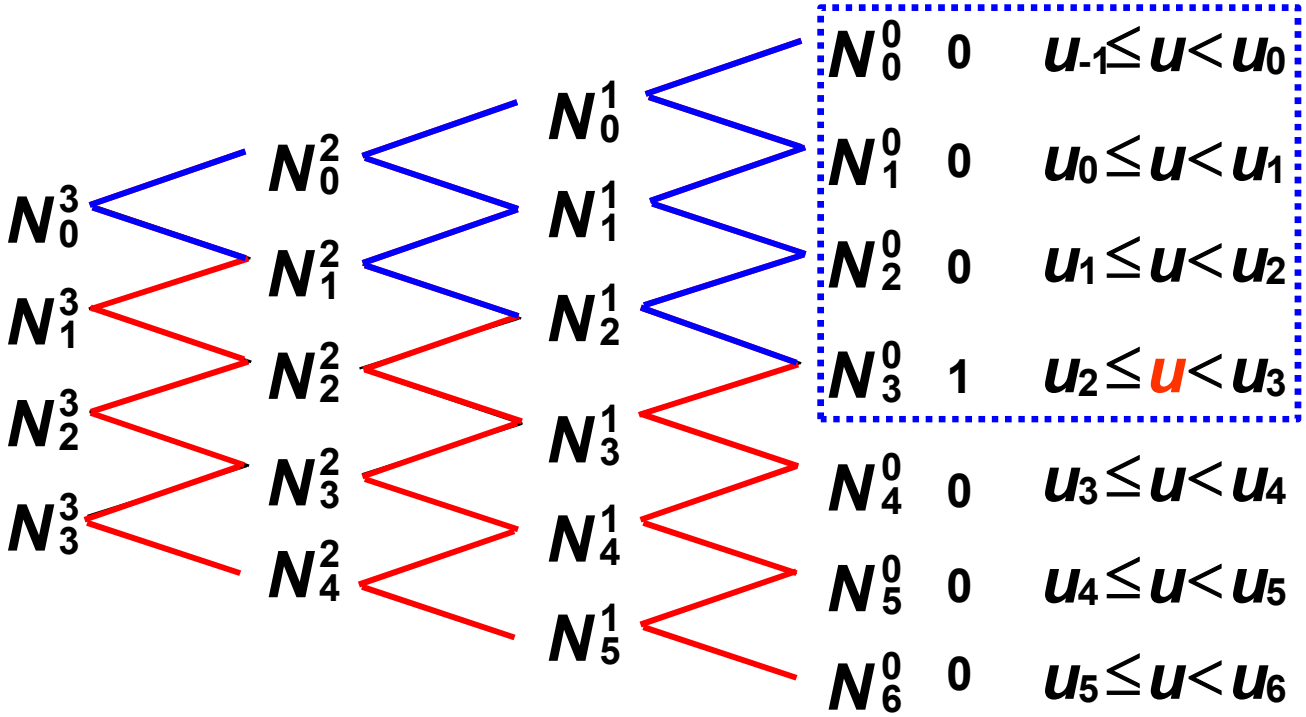


$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

Ex) Calculation of N_0^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



From $u_2 \le u < u_3$,

we can get

$$N_0^0(u) = 0$$

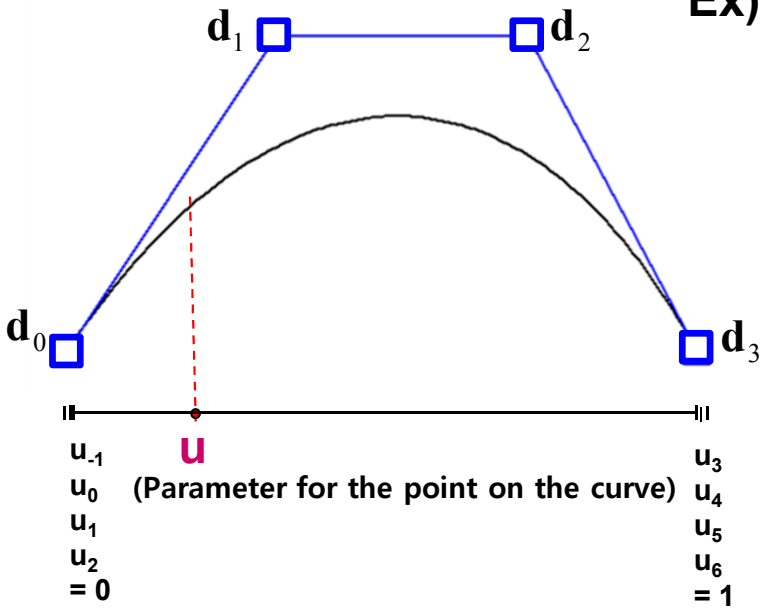
$$N_1^0(u) = 0$$

$$N_2^0(u) = 0$$

$$N_3^0(u) = 1$$

[Example] B-Spline Function of One Curve Segment (3/10)

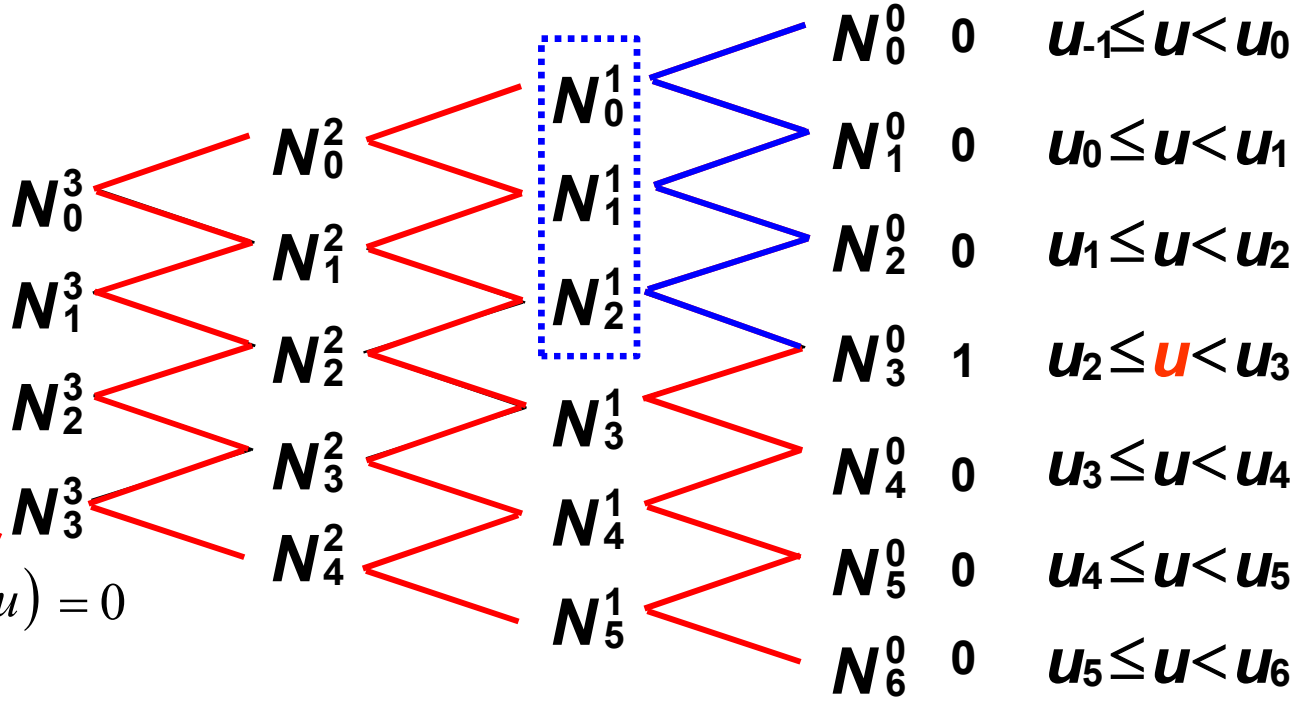
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Ex) Calculation of N_0^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$$N_0^0(u) = 0, \quad N_1^0(u) = 0$$

$$N_2^0(u) = 0, \quad N_3^0(u) = 1$$

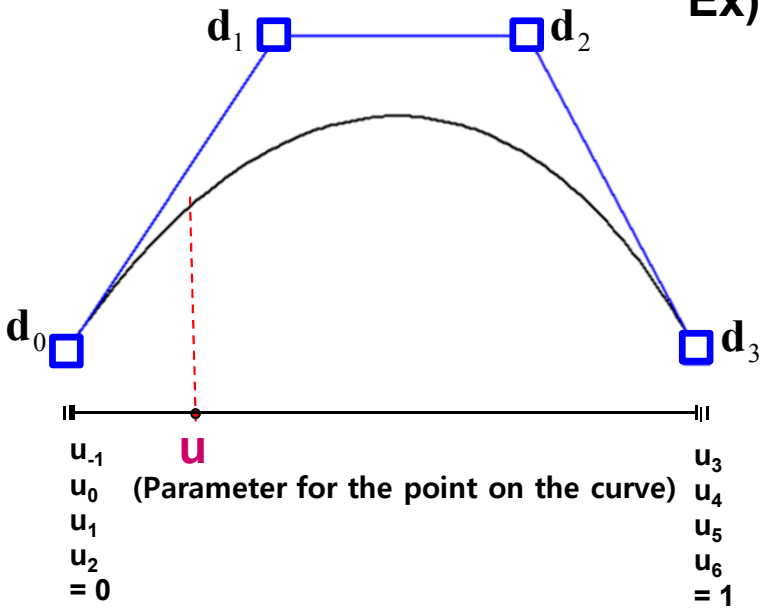
$$N_0^1(u) = \frac{u - u_{-1}}{u_0 - u_{-1}} N_0^0(u) + \frac{u_1 - u}{u_1 - u_0} N_0^0(u) = 0$$

$$N_1^1(u) = \frac{u - u_0}{u_1 - u_0} N_1^0(u) + \frac{u_2 - u}{u_2 - u_1} N_2^0(u) = 0$$

$$N_2^1(u) = \frac{u - u_1}{u_2 - u_1} N_2^0(u) + \frac{u_3 - u}{u_3 - u_2} N_3^0(u) = \frac{u_3 - u}{u_3 - u_2} = 1 - u$$

[Example] B-Spline Function of One Curve Segment (4/10)

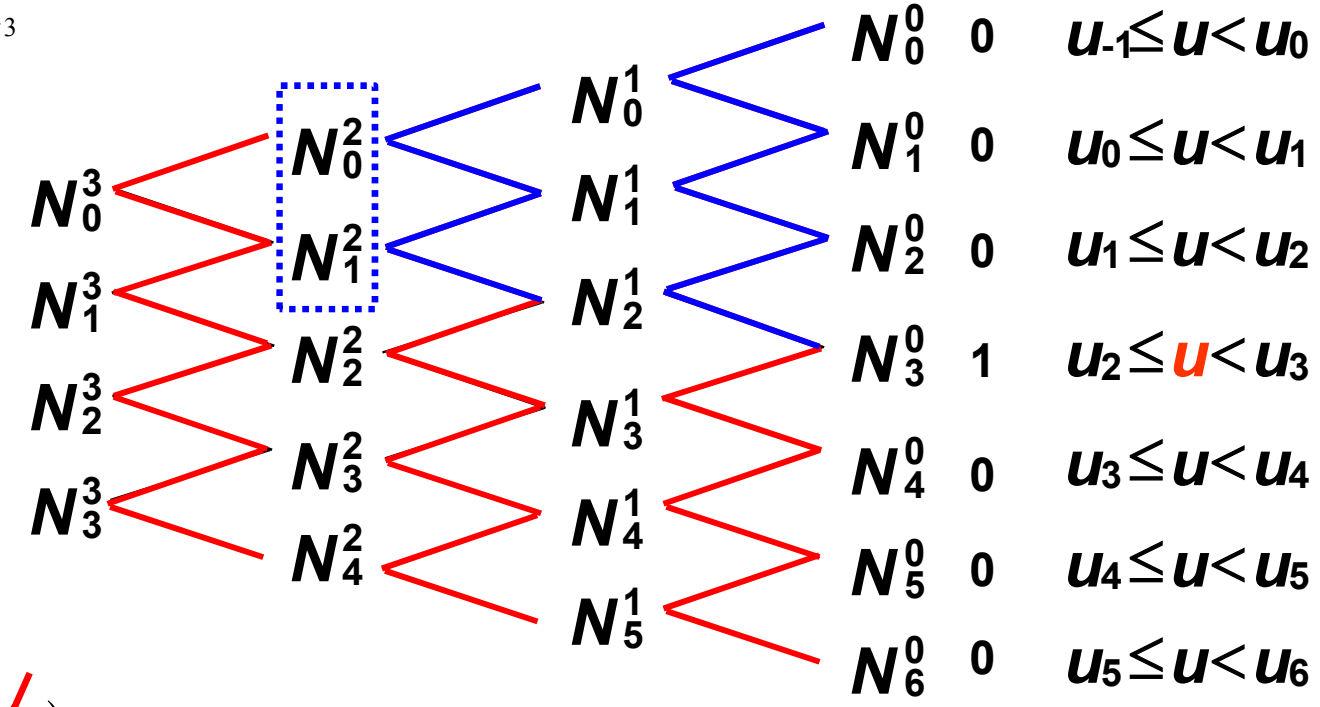
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Ex) Calculation of N_0^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$$N_0^0(u) = 0, \quad N_1^0(u) = 0$$

$$N_2^0(u) = 0, \quad N_3^0(u) = 1$$

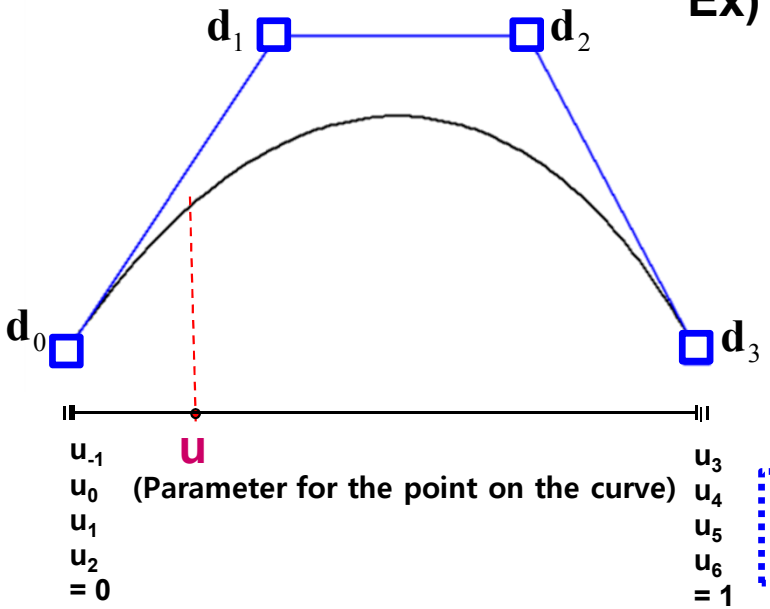
$$N_0^1(u) = 0, \quad N_1^1(u) = 0, \quad N_2^1(u) = 1 - u$$

$$N_0^2(u) = \frac{u - u_{-1}}{u_1 - u_{-1}} N_0^1(u) + \frac{u_2 - u}{u_2 - u_0} N_1^1(u) = 0$$

$$N_1^2(u) = \frac{u - u_0}{u_2 - u_0} N_1^1(u) + \frac{u_3 - u}{u_3 - u_1} N_2^1(u) = \frac{u_3 - u}{u_3 - u_1}(1 - u) = (1 - u)^2$$

[Example] B-Spline Function of One Curve Segment (5/10)

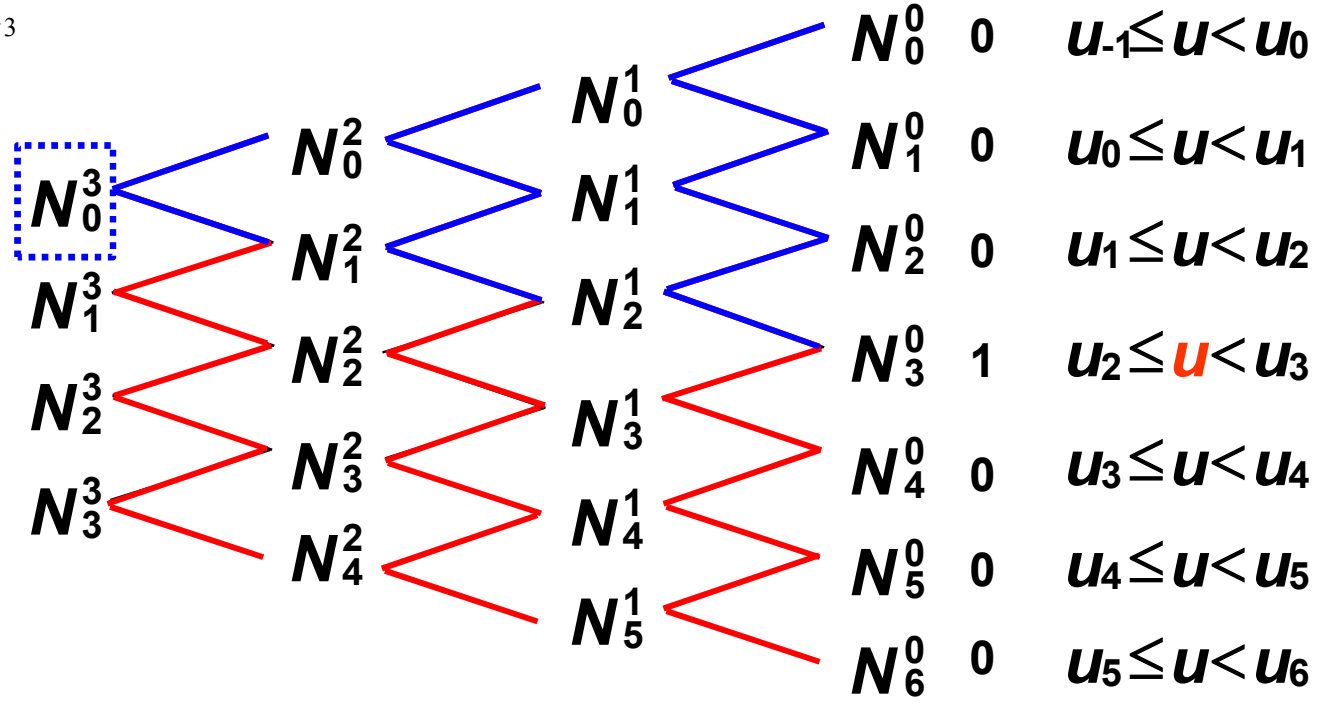
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Ex) Calculation of N_0^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$$N_0^0(u) = 0,\ N_1^0(u) = 0$$

$$N_2^0(u) = 0,\ N_3^0(u) = 1$$

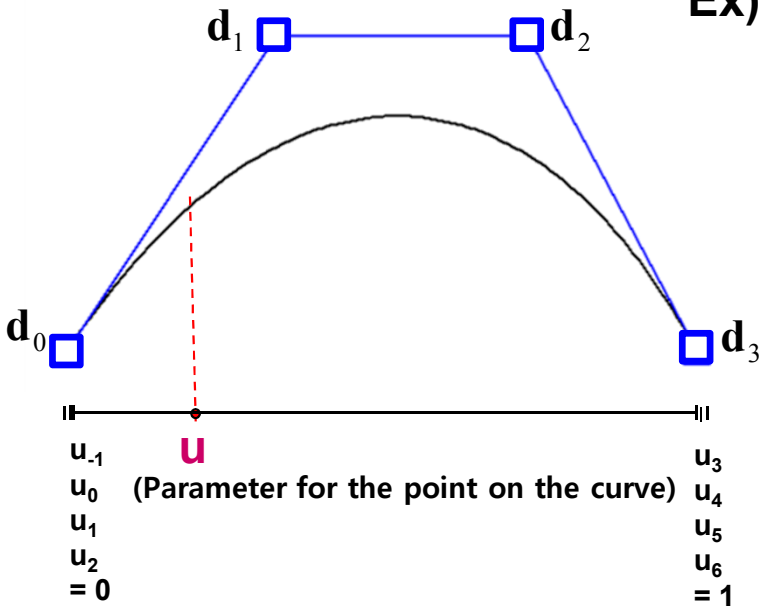
$$N_0^1(u) = 0,\ N_1^1(u) = 0,\ N_2^1(u) = 1 - u$$

$$N_0^2(u) = 0,\ N_1^2(u) = (1-u)^2$$

$$N_0^3(u) = \frac{u - u_{-1}}{u_2 - u_{-1}} N_0^2(u) + \frac{u_3 - u}{u_3 - u_0} N_1^2(u) = \frac{u_3 - u}{u_3 - u_0}(1-u)^2 = (1-u)^3$$

[Example] B-Spline Function of One Curve Segment (6/10)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$

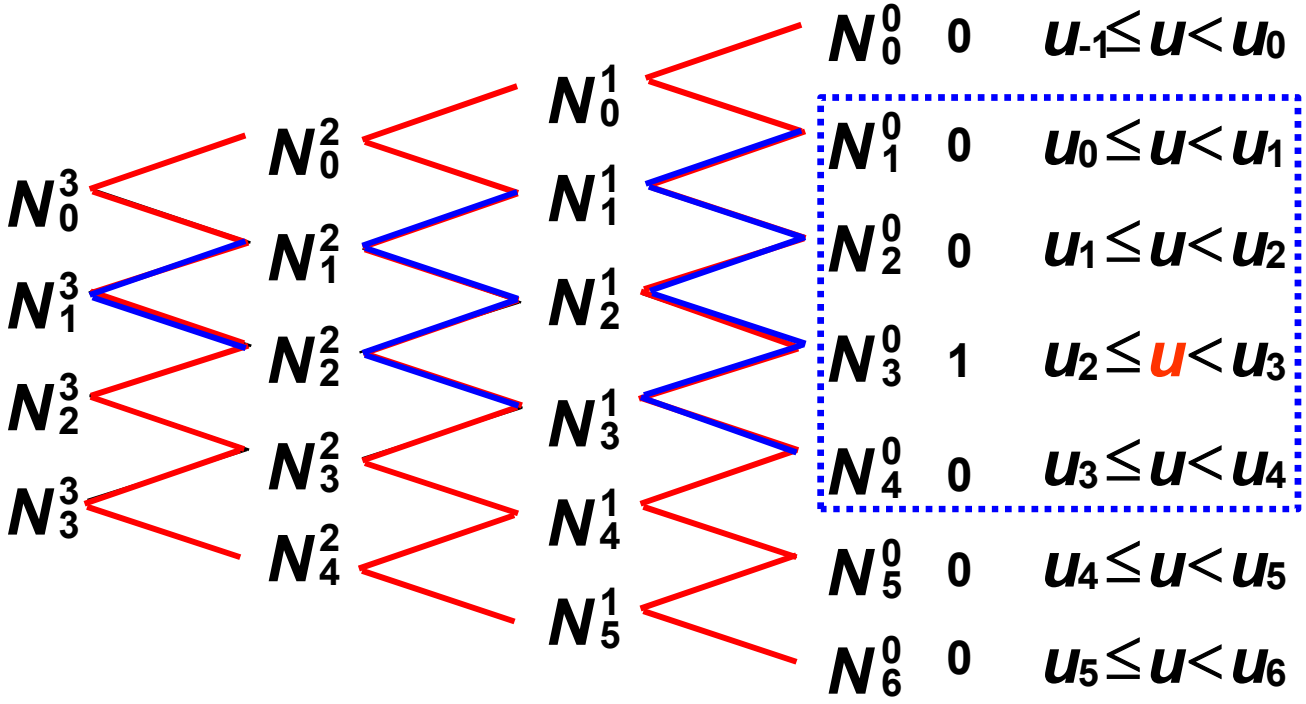


$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

Ex) Calculation of N_1^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



From $u_2 \le u < u_3$,

we can get

$$N_1^0(u) = 0$$

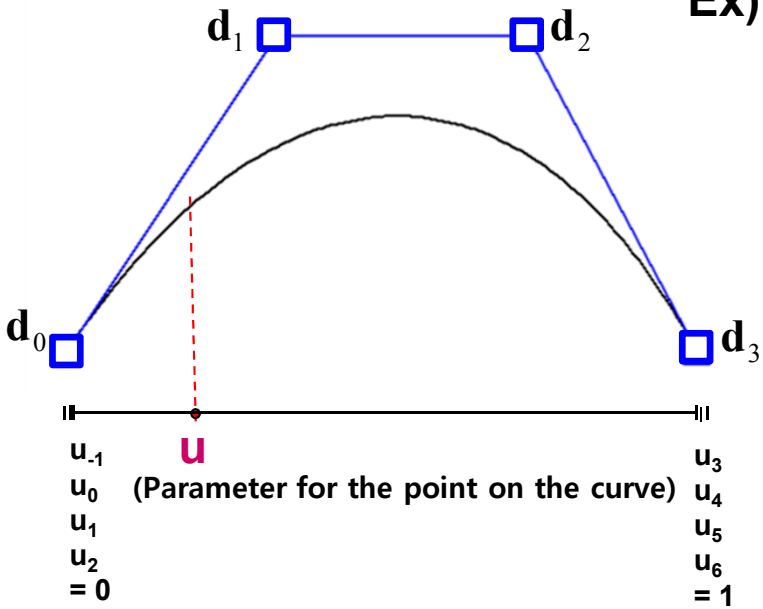
$$N_2^0(u) = 0$$

$$N_3^0(u) = 1$$

$$N_4^0(u) = 0$$

[Example] B-Spline Function of One Curve Segment (7/10)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Ex) Calculation of N_1^3

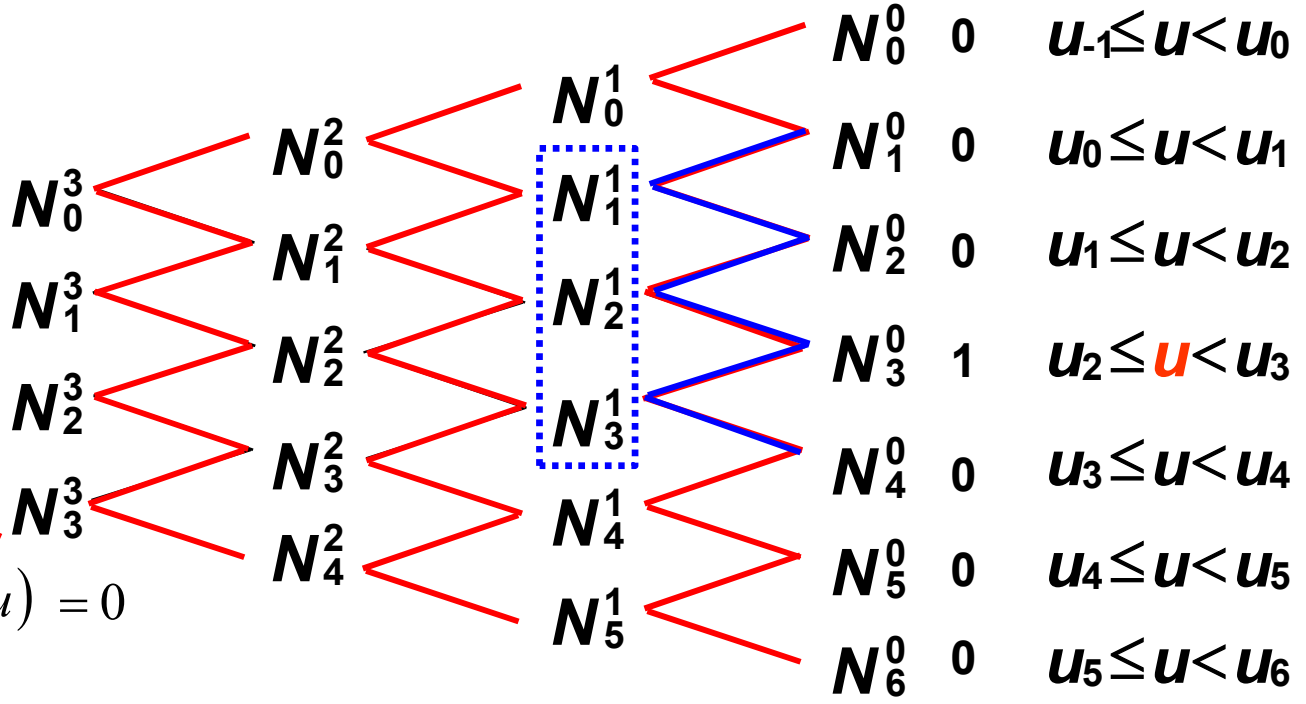
$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$N_1^0(u) = 0, \quad N_2^0(u) = 0$

$N_3^0(u) = 1, \quad N_4^0(u) = 0$



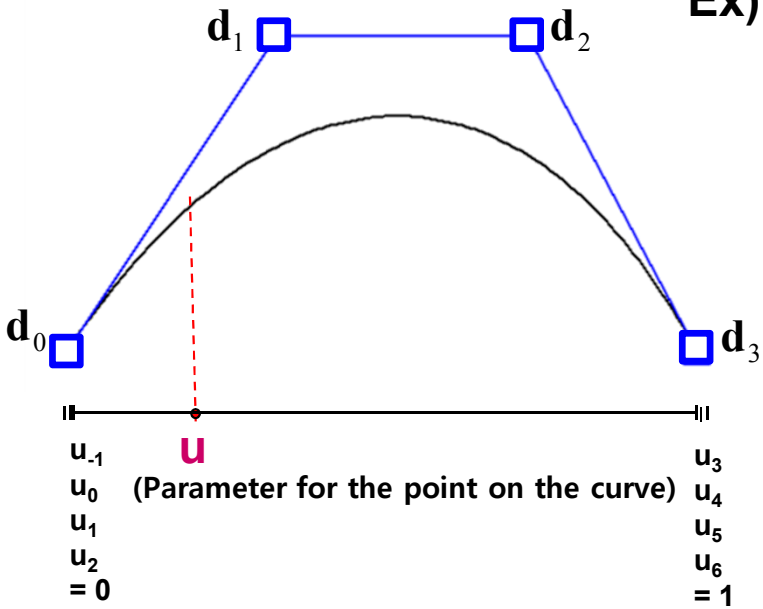
$$N_1^1(u) = \frac{u - u_0}{u_1 - u_0} N_1^0(u) + \frac{u_2 - u}{u_2 - u_0} N_2^0(u) = 0$$

$$N_2^1(u) = \frac{u - u_1}{u_2 - u_1} N_2^0(u) + \frac{u_3 - u}{u_3 - u_2} N_3^0(u) = \frac{u_3 - u}{u_3 - u_2} = 1 - u$$

$$N_3^1(u) = \frac{u - u_1}{u_3 - u_2} N_3^0(u) + \frac{u_4 - u}{u_4 - u_3} N_4^0(u) = \frac{u - u_1}{u_3 - u_2} = u$$

[Example] B-Spline Function of One Curve Segment (8/10)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Ex) Calculation of N_1^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

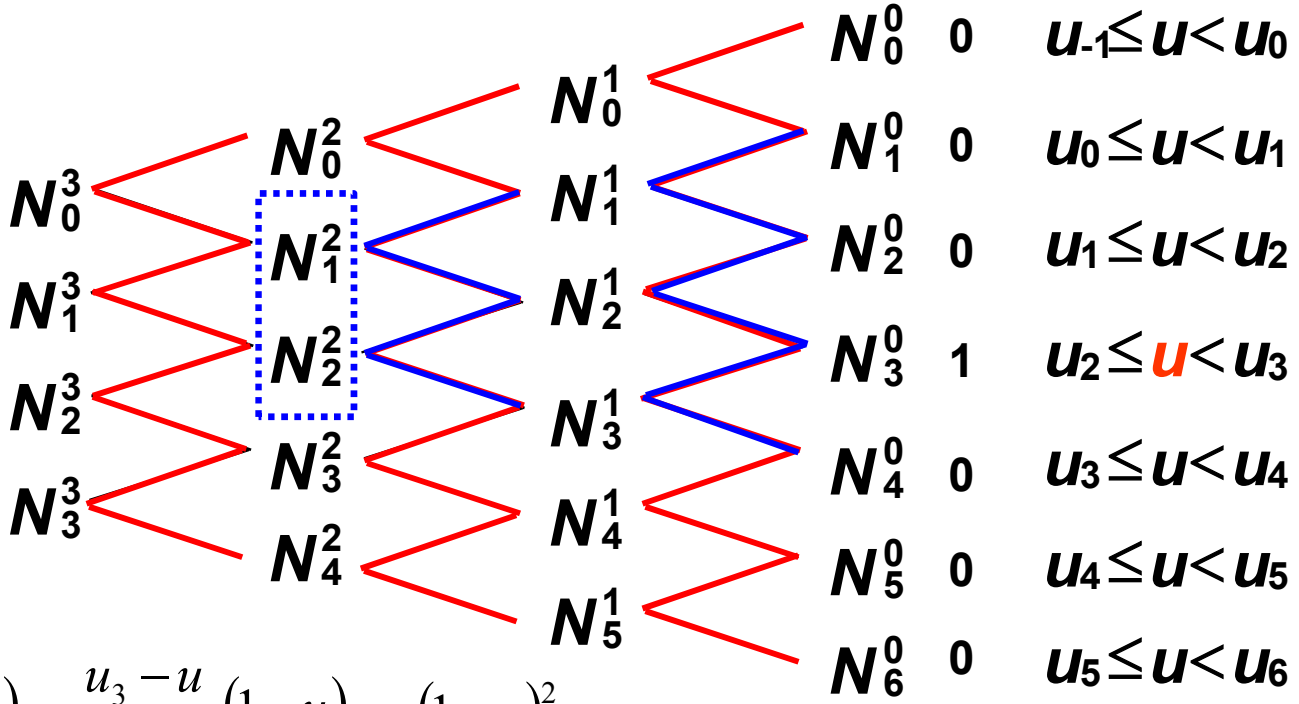
$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$$N_1^0(u) = 0,\ N_2^0(u) = 0$$

$$N_3^0(u) = 1,\ N_4^0(u) = 0$$

$$N_1^1(u) = 0,\ N_2^1(u) = 1 - u,\ N_3^1(u) = u$$

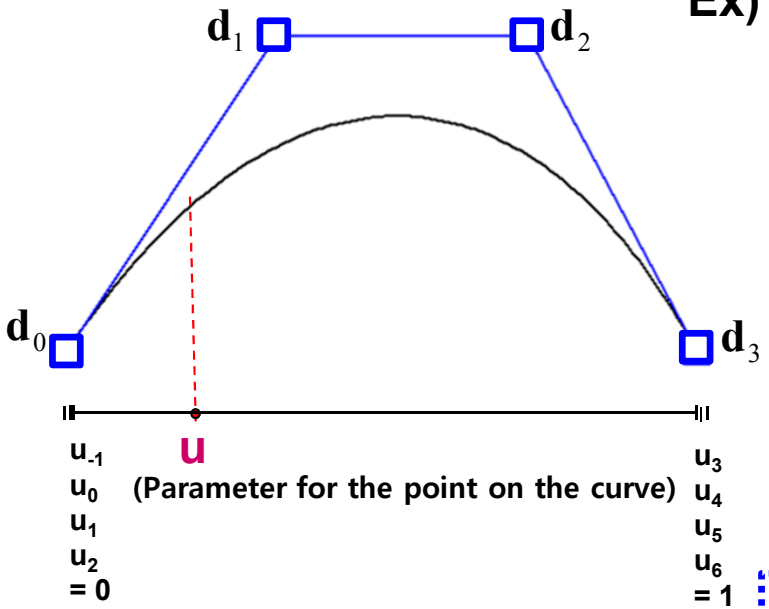


$$N_1^2(u) = \frac{u - u_0}{u_2 - u_0} N_1^1(u) + \frac{u_3 - u}{u_3 - u_1} N_2^1(u) = \frac{u_3 - u}{u_3 - u_1}(1 - u) = (1 - u)^2$$

$$N_2^2(u) = \frac{u - u_1}{u_3 - u_1} N_2^1(u) + \frac{u_4 - u}{u_4 - u_2} N_3^1(u) = \frac{u - u_1}{u_3 - u_1}(1 - u) + \frac{u_4 - u}{u_4 - u_2} u = 2u(1 - u)$$

[Example] B-Spline Function of One Curve Segment (9/10)

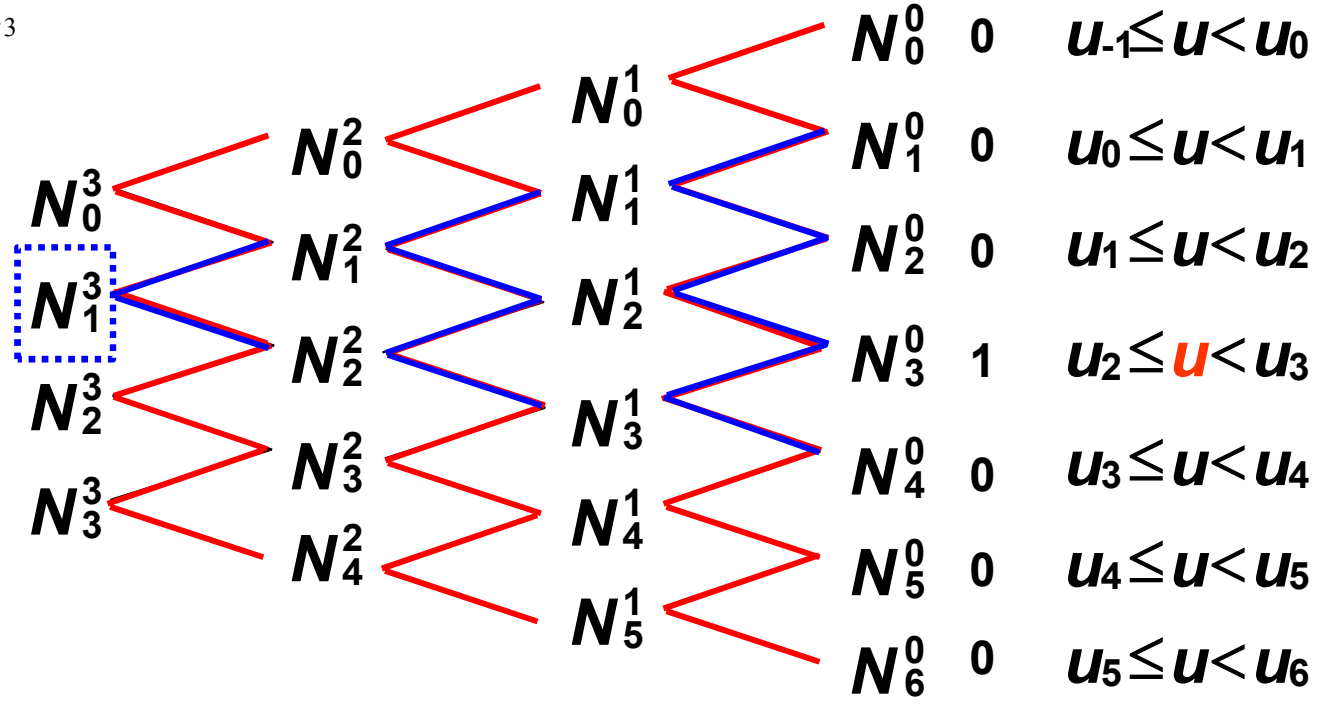
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



Ex) Calculation of N_1^3

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$$N_1^0(u) = 0, \quad N_2^0(u) = 0$$

$$N_3^0(u) = 1, \quad N_4^0(u) = 0$$

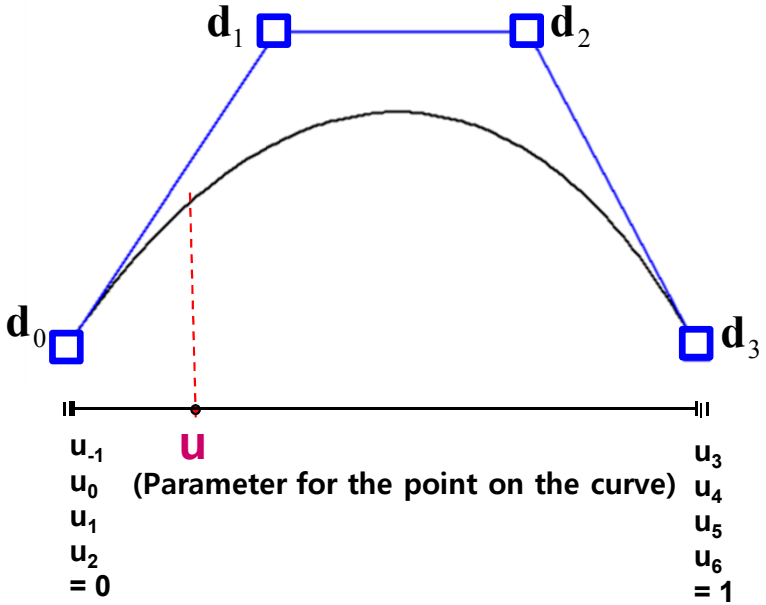
$$N_0^1(u) = 0, \quad N_1^1(u) = 0, \quad N_2^1(u) = 1 - u$$

$$N_1^2(u) = (1-u)^2, \quad N_2^2(u) = 2u(1-u)$$

$$N_1^3(u) = \frac{u - u_0}{u_3 - u_0} N_1^2(u) + \frac{u_4 - u}{u_4 - u_1} N_2^2(u) = \frac{u - u_0}{u_3 - u_0}(1-u)^2 + \frac{u_4 - u}{u_4 - u_1} \cdot 2u(1-u) = 3u(1-u)^2$$

[Example] B-Spline Function of One Curve Segment (10/10)

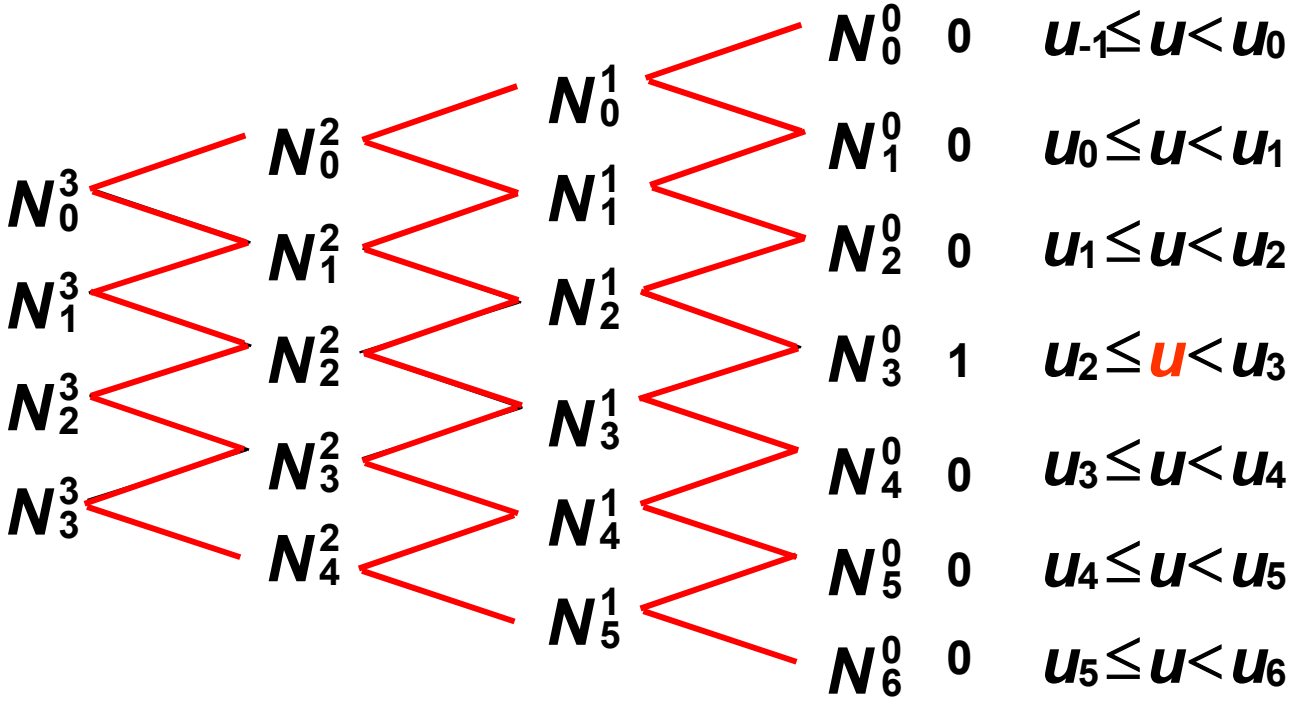
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$



Calculate $N_2^3(u)$, $N_2^3(u)$ in the same manner.

$$N_0^3(u) = (1-u)^3$$

$$N_1^3(u) = 3u(1-u)^2$$

$$N_2^3(u) = 3u^2(1-u)$$

$$N_3^3(u) = 3u^3$$

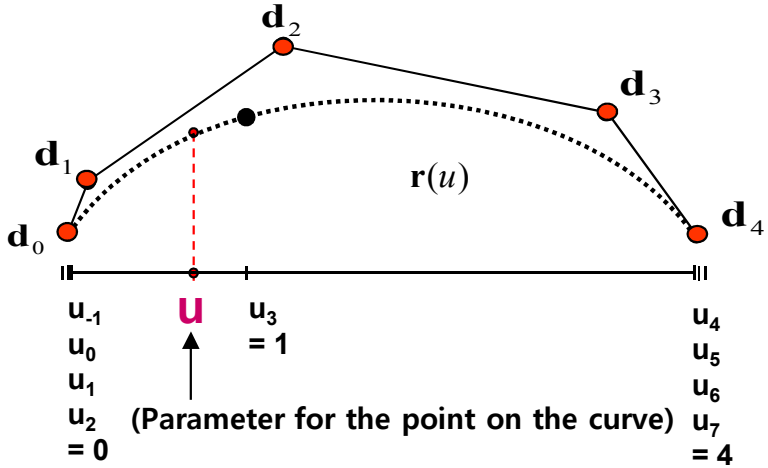
$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

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

➡ The B-spline curve is identical with the 3rd-degree Bezier curves four given control points.

[Example] B-Spline Function of Two Curve Segments (1/4)

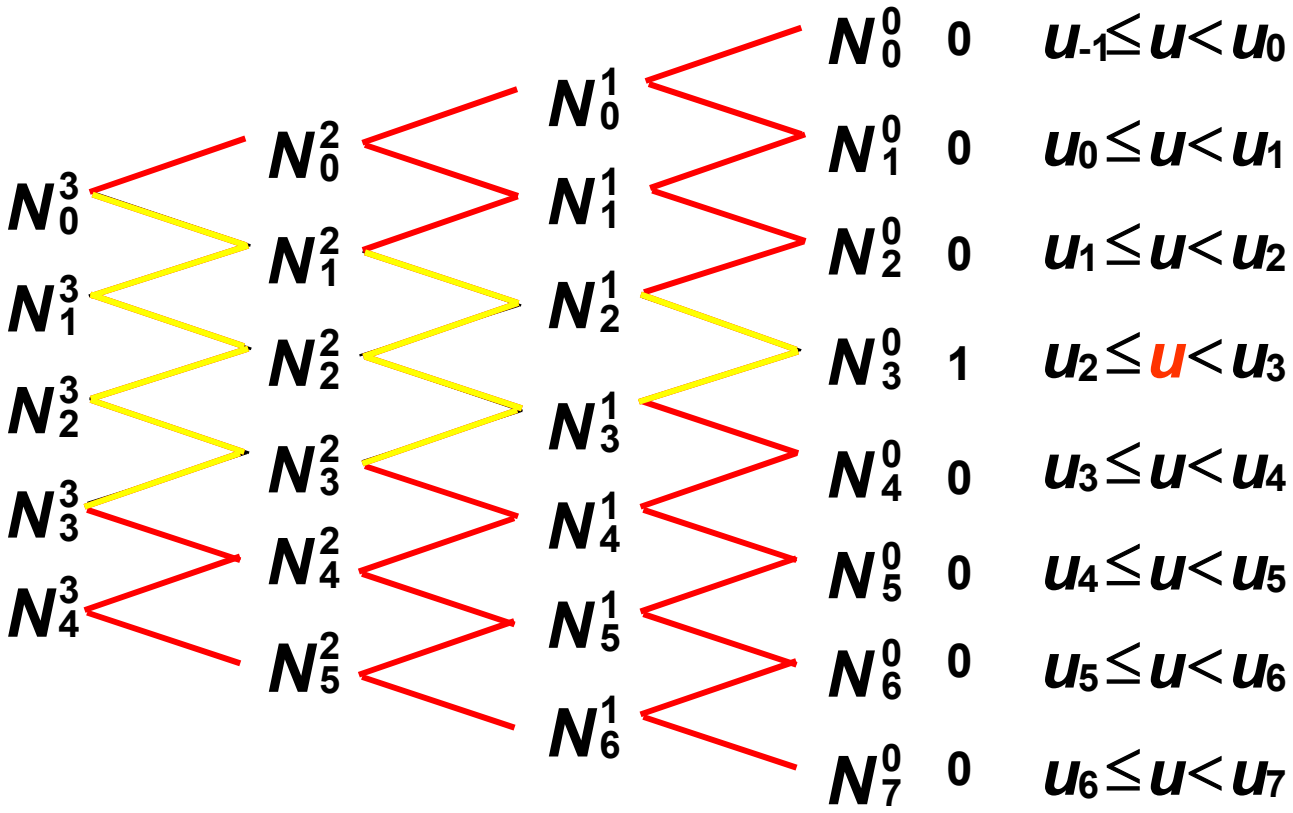
Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$



$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

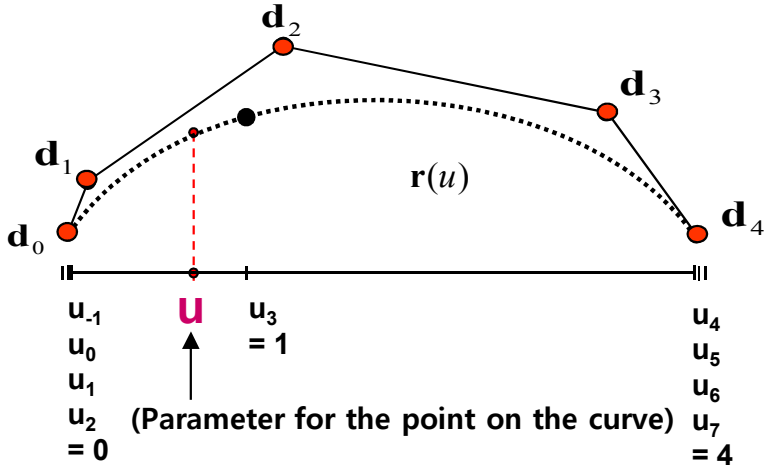
$$\mathbf{r}(u) = \sum_{i=0}^{D-1} \mathbf{d}_i N_i^n(u)$$



[Example] B-Spline Function of Two Curve Segments (2/4)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$

Ex) Calculation of N_1^3

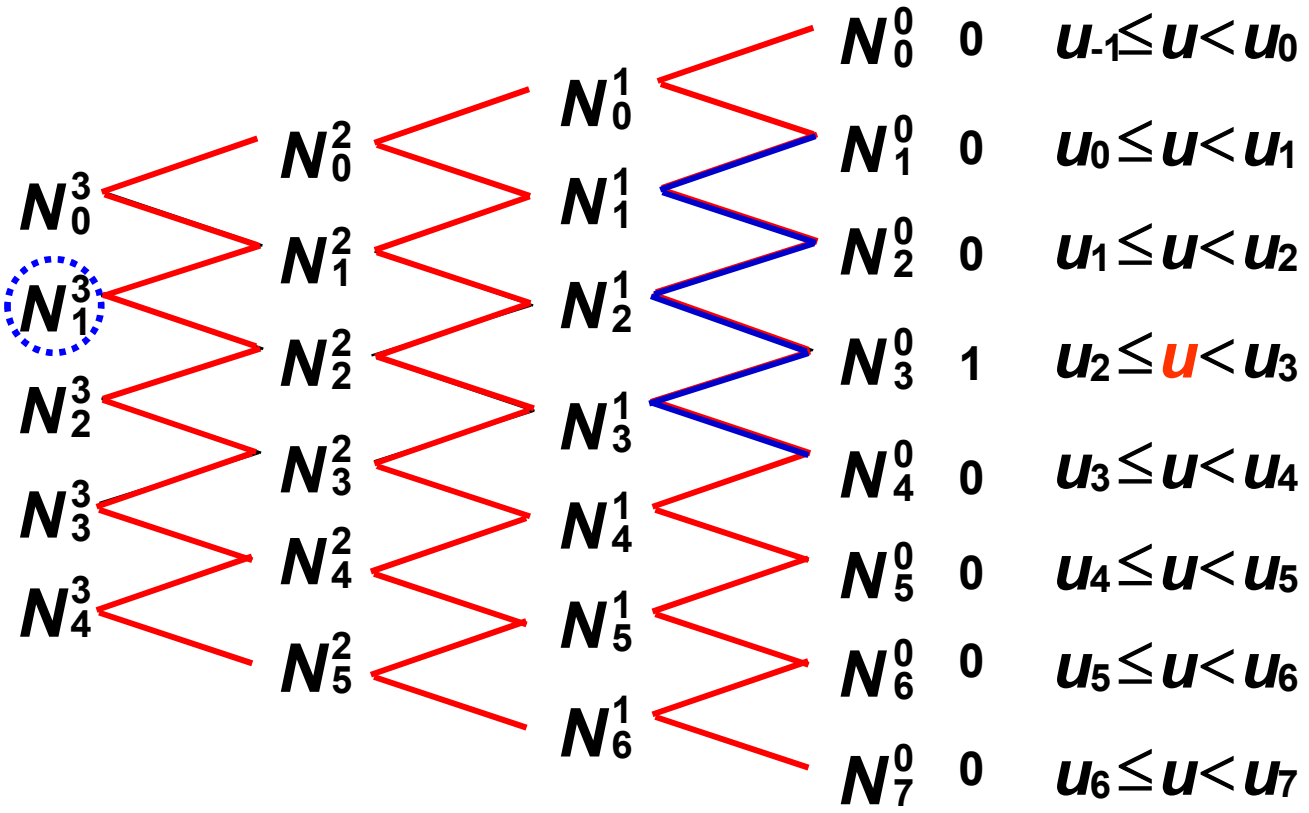


$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

$$N_1^1 = 0 \qquad N_2^1 = \frac{u_3 - u}{u_3 - u_2} \qquad N_3^1 = \frac{u - u_2}{u_3 - u_2}$$

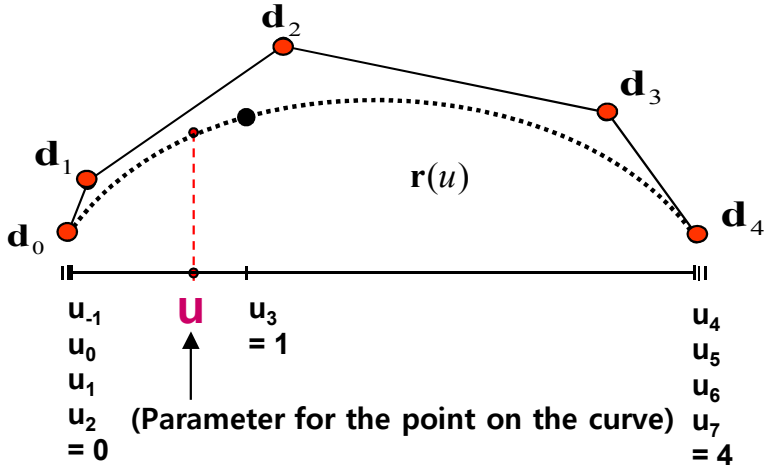
$$\mathbf{r}(u) = \sum_{i=0}^{D-1} \mathbf{d}_i N_i^n(u)$$



[Example] B-Spline Function of Two Curve Segments (3/4)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$

Ex) Calculation of N_1^3



$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

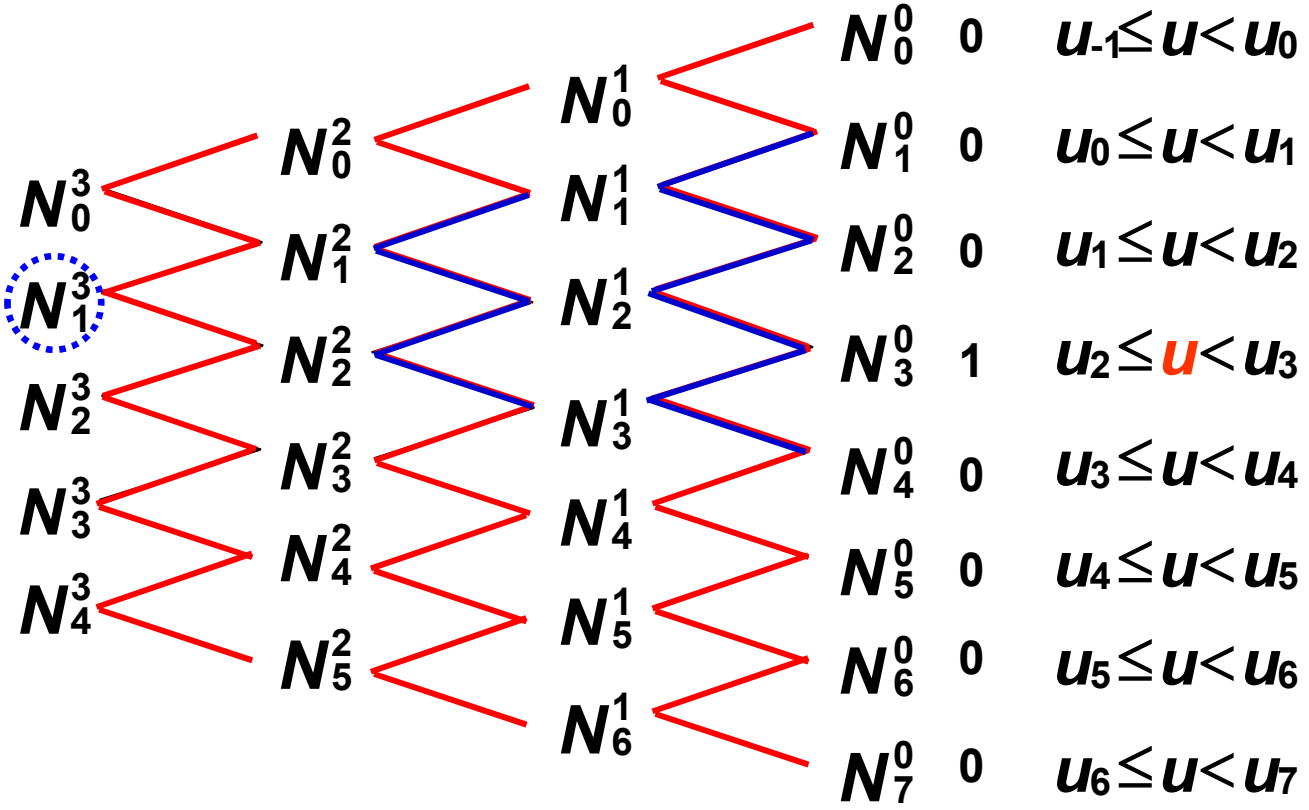
$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

$$N_1^1 = 0 \quad N_2^1 = \frac{u_3 - u}{u_3 - u_2} \quad N_3^1 = \frac{u - u_2}{u_3 - u_2}$$

$$N_1^2 = \frac{u_3 - u}{u_3 - u_1} N_2^1 = \frac{u_3 - u}{u_3 - u_1} \cdot \frac{u_3 - u}{u_3 - u_2}$$

$$N_2^2 = \frac{u - u_1}{u_3 - u_1} N_2^1 + \frac{u_4 - u}{u_4 - u_2} N_3^1$$

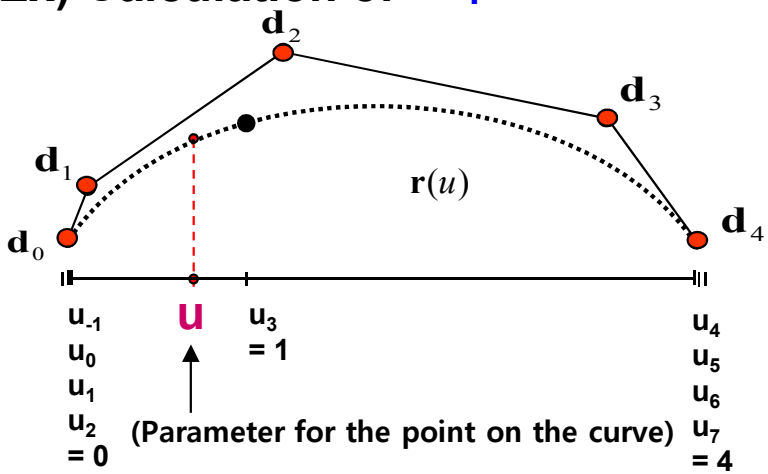
$$= \frac{u - u_1}{u_3 - u_1} \cdot \frac{u_3 - u}{u_3 - u_2} + \frac{u_4 - u}{u_4 - u_2} \cdot \frac{u - u_2}{u_3 - u_2}$$



[Example] B-Spline Function of Two Curve Segments (4/4)

Given	B-spline control point $\mathbf{d}_i$ Parameter u B-spline basis function $N_i^n(u)$
Find	B-spline curve $\mathbf{r}(u)$

Ex) Calculation of N_1^3



$$N_i^n(u) = \frac{u-u_{i-1}}{u_{i+n-1}-u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n}-u}{u_{i+n}-u_i} N_{i+1}^{n-1}(u)$$

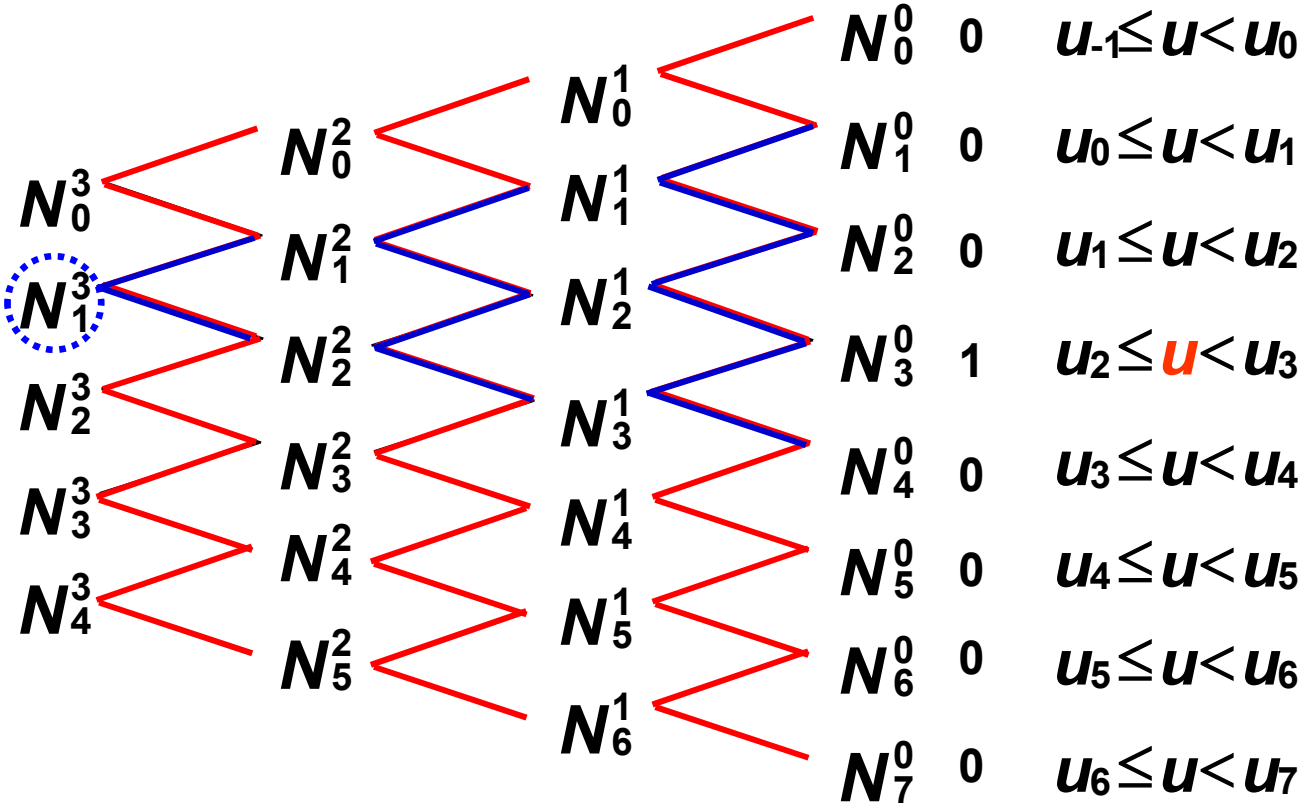
$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

$$N_1^1 = 0 \quad N_2^1 = \frac{u_2-u}{u_3-u_2} \quad N_3^1 = \frac{u-u_2}{u_3-u_2}$$

$$N_1^2 = \frac{u_3-u}{u_3-u_1} N_2^1 = \frac{u_3-u}{u_3-u_1} \cdot \frac{u_2-u}{u_3-u_2}$$

$$N_2^2 = \frac{u-u_1}{u_3-u_1} N_2^1 + \frac{u_4-u}{u_4-u_2} N_3^1$$

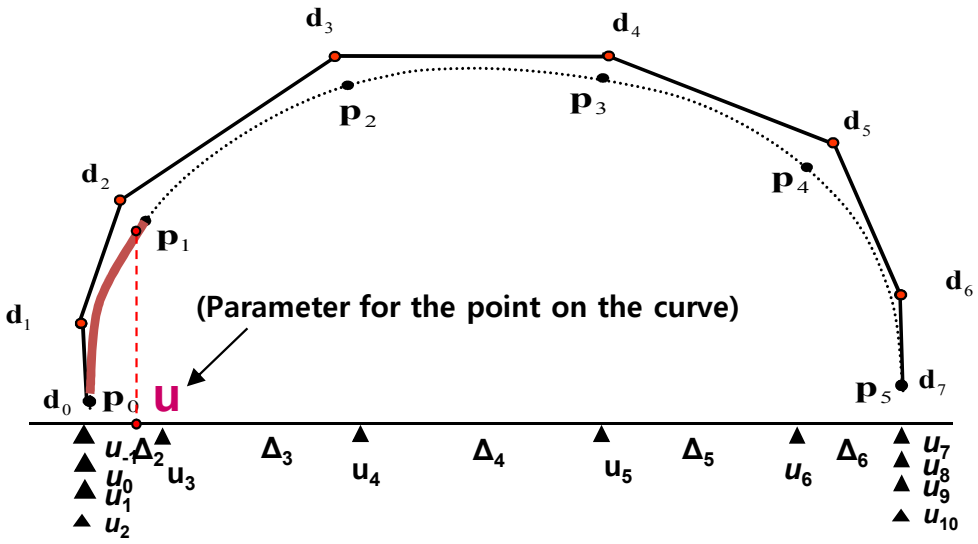
$$= \frac{u-u_1}{u_3-u_1} \cdot \frac{u_2-u}{u_3-u_2} + \frac{u_4-u}{u_4-u_2} \cdot \frac{u-u_2}{u_3-u_2}$$



$$\therefore N_1^3 = \frac{u-u_0}{u_3-u_2} N_1^2 + \frac{u_4-u}{u_4-u_1} N_2^2 = \frac{u-u_0}{u_3-u_2} \cdot \frac{u_3-u}{u_3-u_1} \cdot \frac{u_2-u}{u_3-u_2} + \frac{u_4-u}{u_4-u_1} \cdot \frac{u-u_1}{u_3-u_1} \cdot \frac{u_2-u}{u_3-u_2} + \frac{u_4-u}{u_4-u_1} \cdot \frac{u_4-u}{u_4-u_2} \cdot \frac{u-u_2}{u_3-u_2}$$

[Example] B-Spline Function of 5 Curve Segments (1/5)

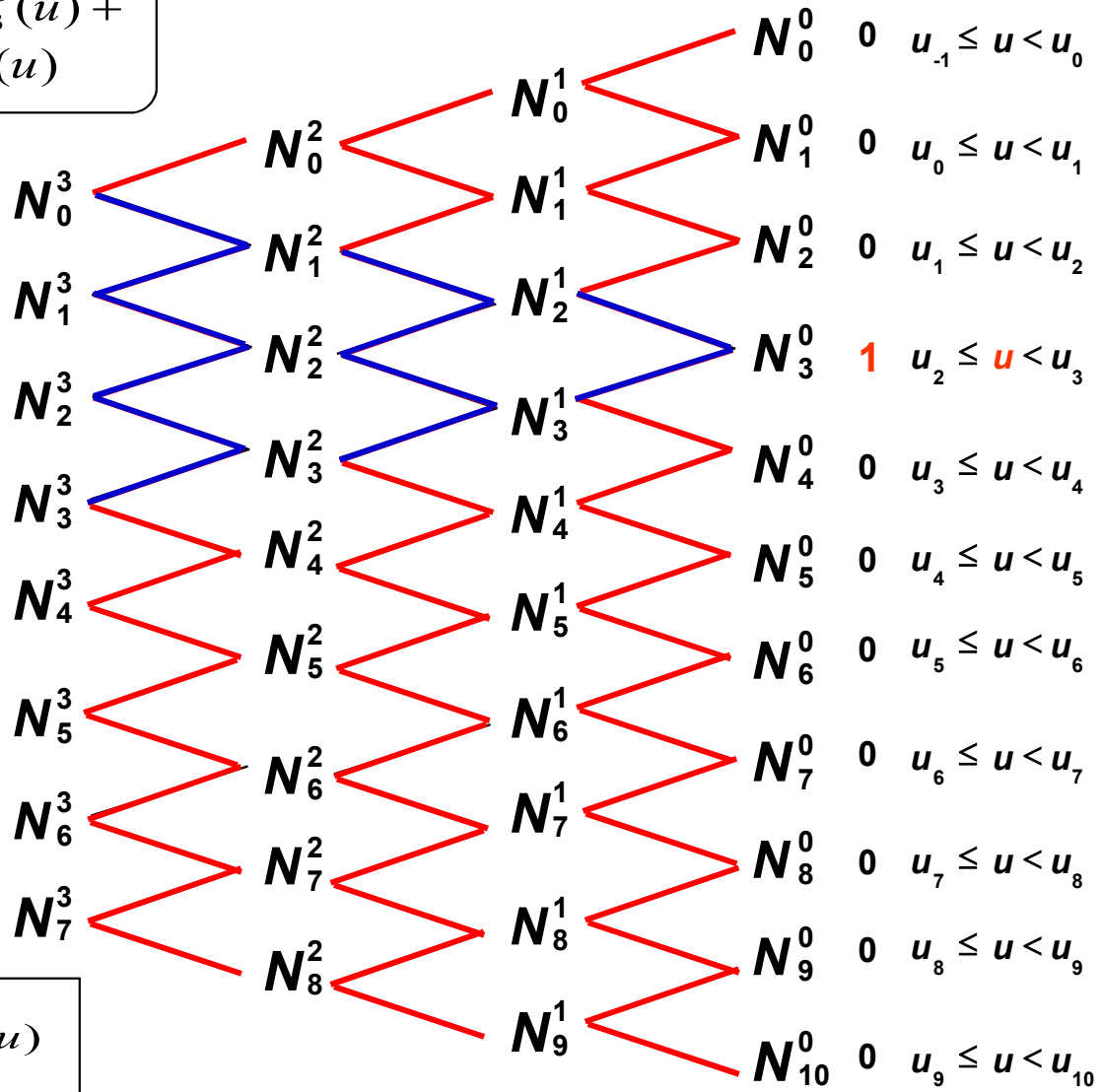
$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$

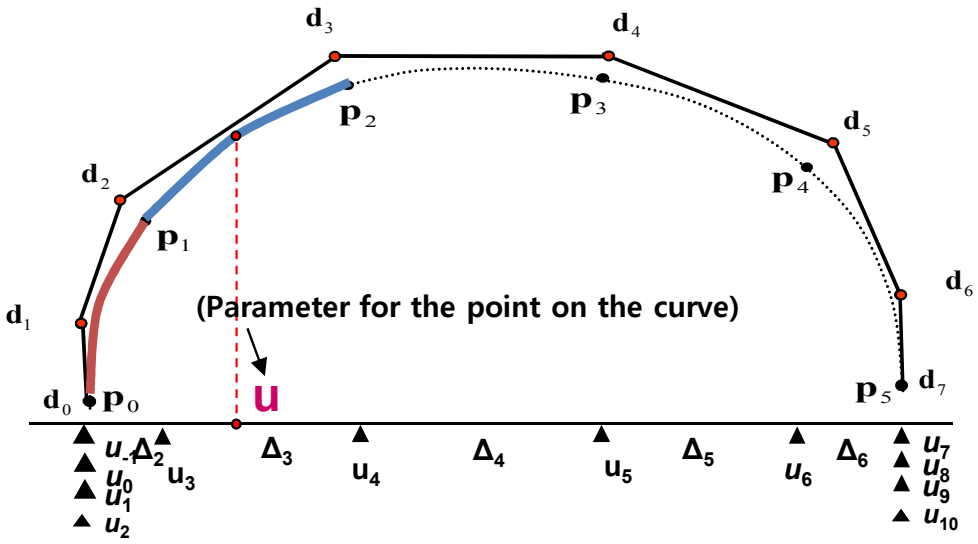
$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

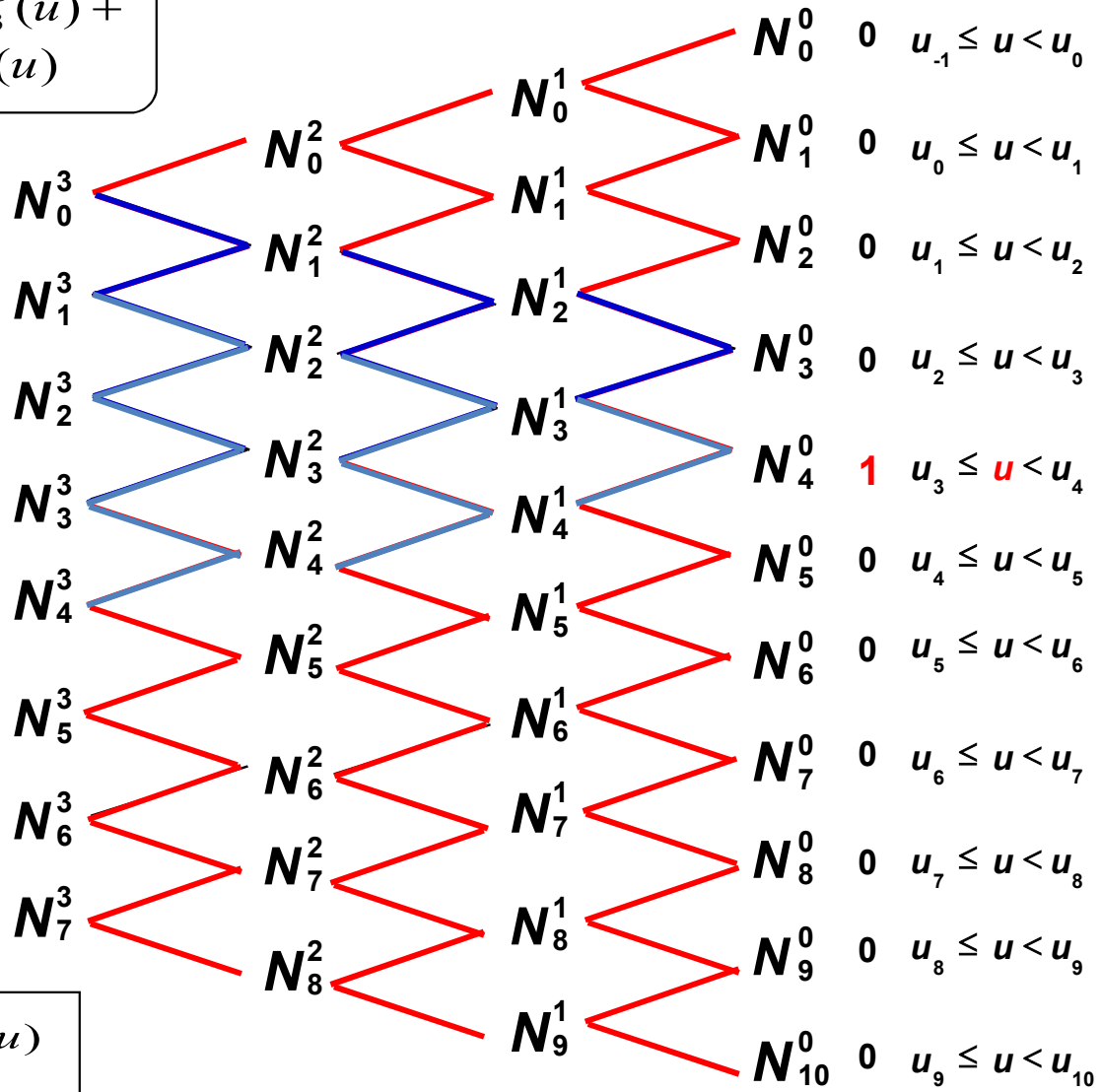


[Example] B-Spline Function of 5 Curve Segments (2/5)

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$

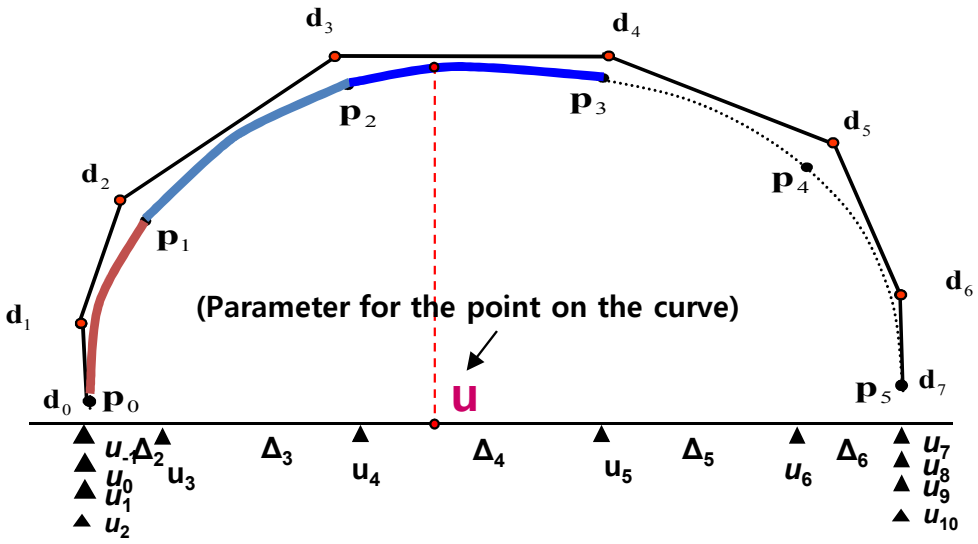


$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

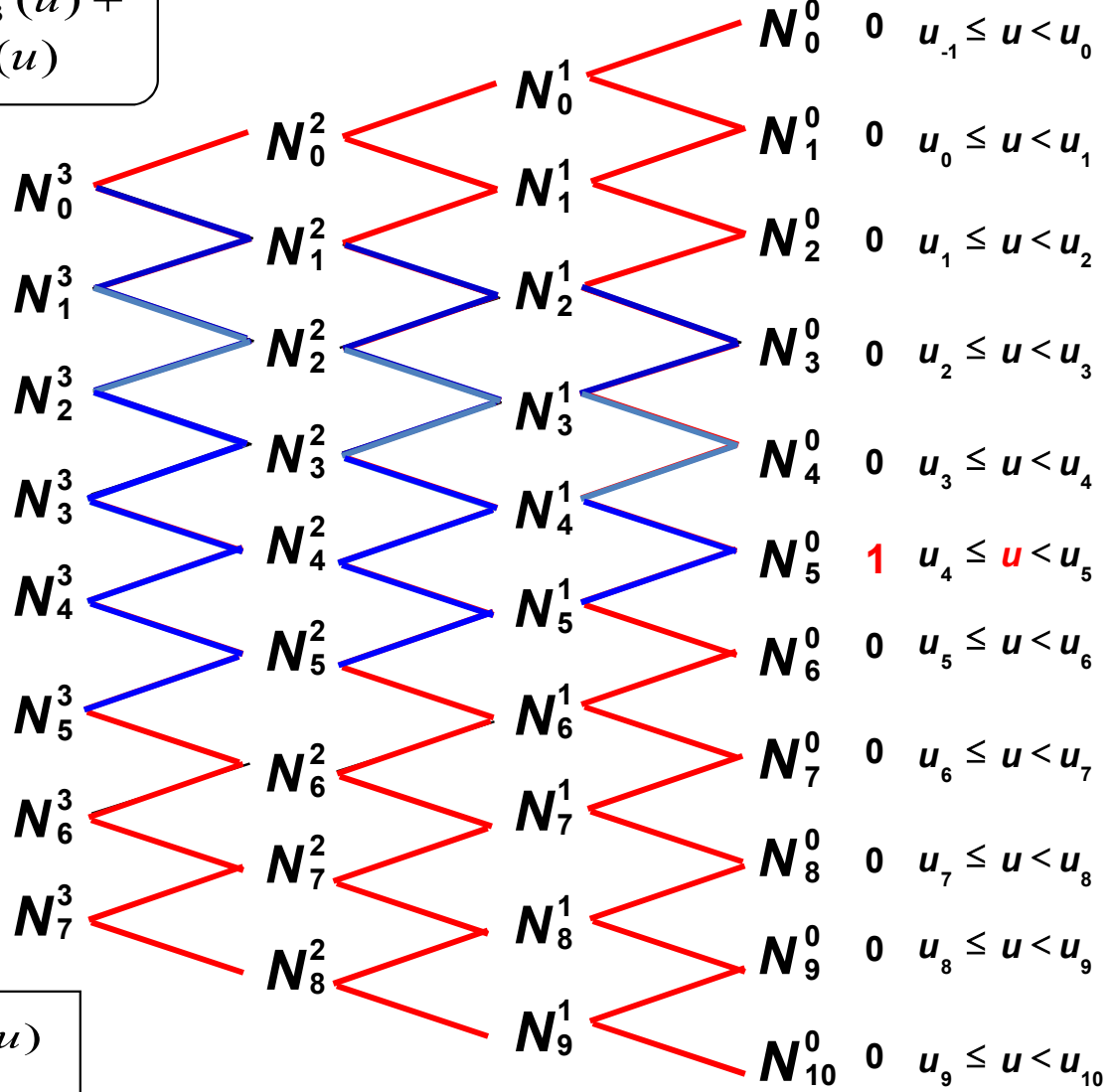
$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

[Example] B-Spline Function of 5 Curve Segments (3/5)

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$

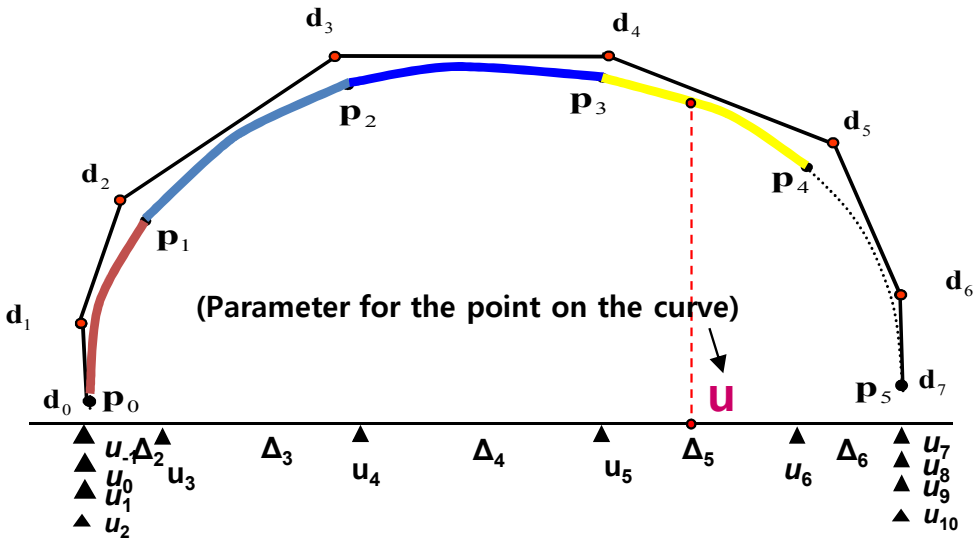


$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

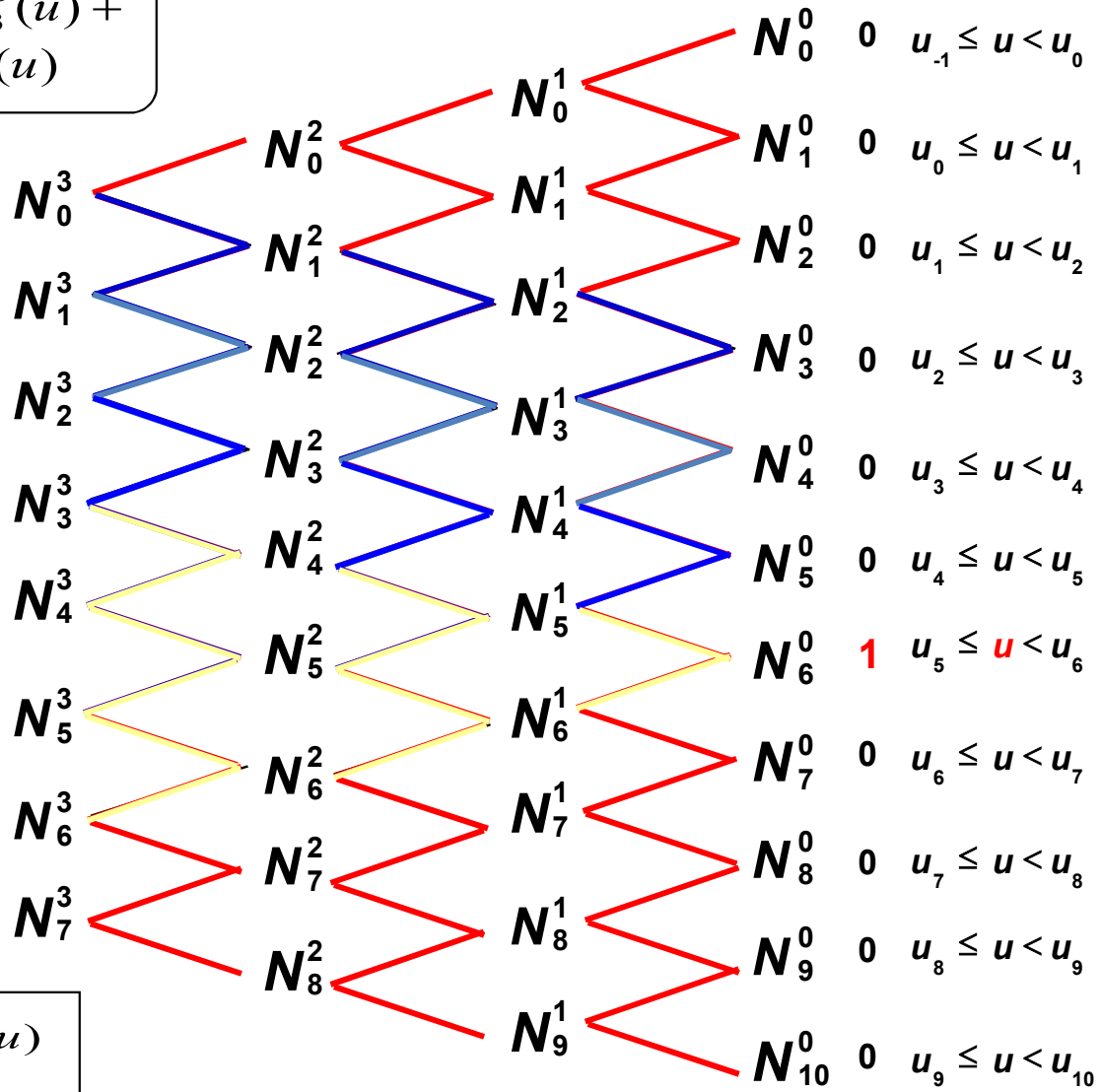
$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

[Example] B-Spline Function of 5 Curve Segments (4/5)

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$

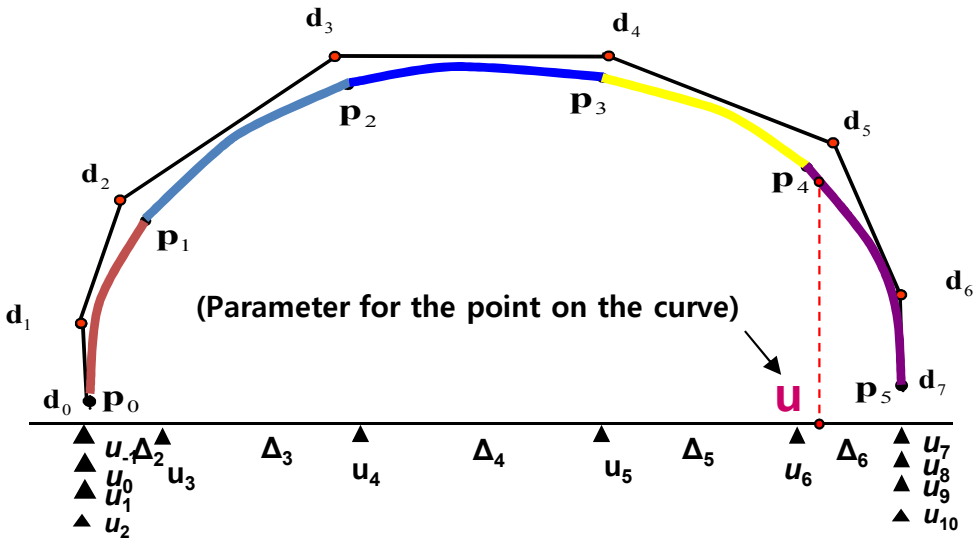


$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

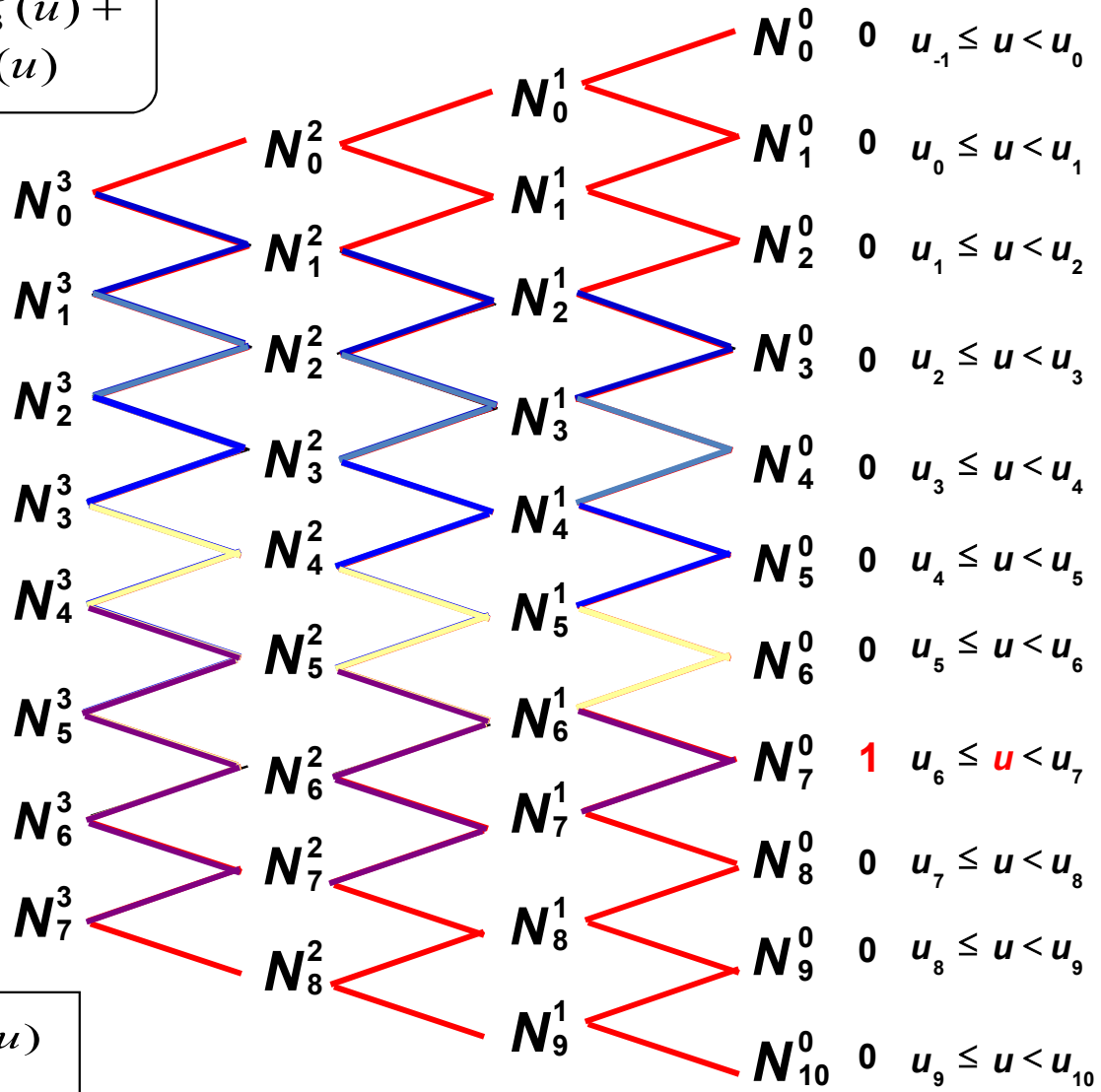
$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

[Example] B-Spline Function of 5 Curve Segments (5/5)

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u) + \mathbf{d}_4 N_4^3(u) + \mathbf{d}_5 N_5^3(u) + \mathbf{d}_6 N_6^3(u) + \mathbf{d}_7 N_7^3(u)$$



$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

[Summary] Cubic B-Spline Curves

Cf: Cubic Bezier Curve

Given: $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, t$

Find

$$\mathbf{r}(t) = \mathbf{b}_0 B_0^3(t) + \mathbf{b}_1 B_1^3(t) + \mathbf{b}_2 B_2^3(t) + \mathbf{b}_3 B_3^3(t)$$

Bernstein polynomial function

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

$$\binom{n}{i} = {}_nC_i = \begin{cases} \dfrac{n!}{i!(n-i)!} & \text{if } 0 \le i \le n \\ 0 & \text{else} \end{cases}$$

☑ Cubic B-spline curves

- Given: d_i, u_j
- Find: r(u) (Points on curve at parameter u)

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \cdots + \mathbf{d}_{D-1} N_{D-1}^3(u)$$

$\mathbf{d}_i$: Control points (de Boor points), i = 0, 1, … , D-1

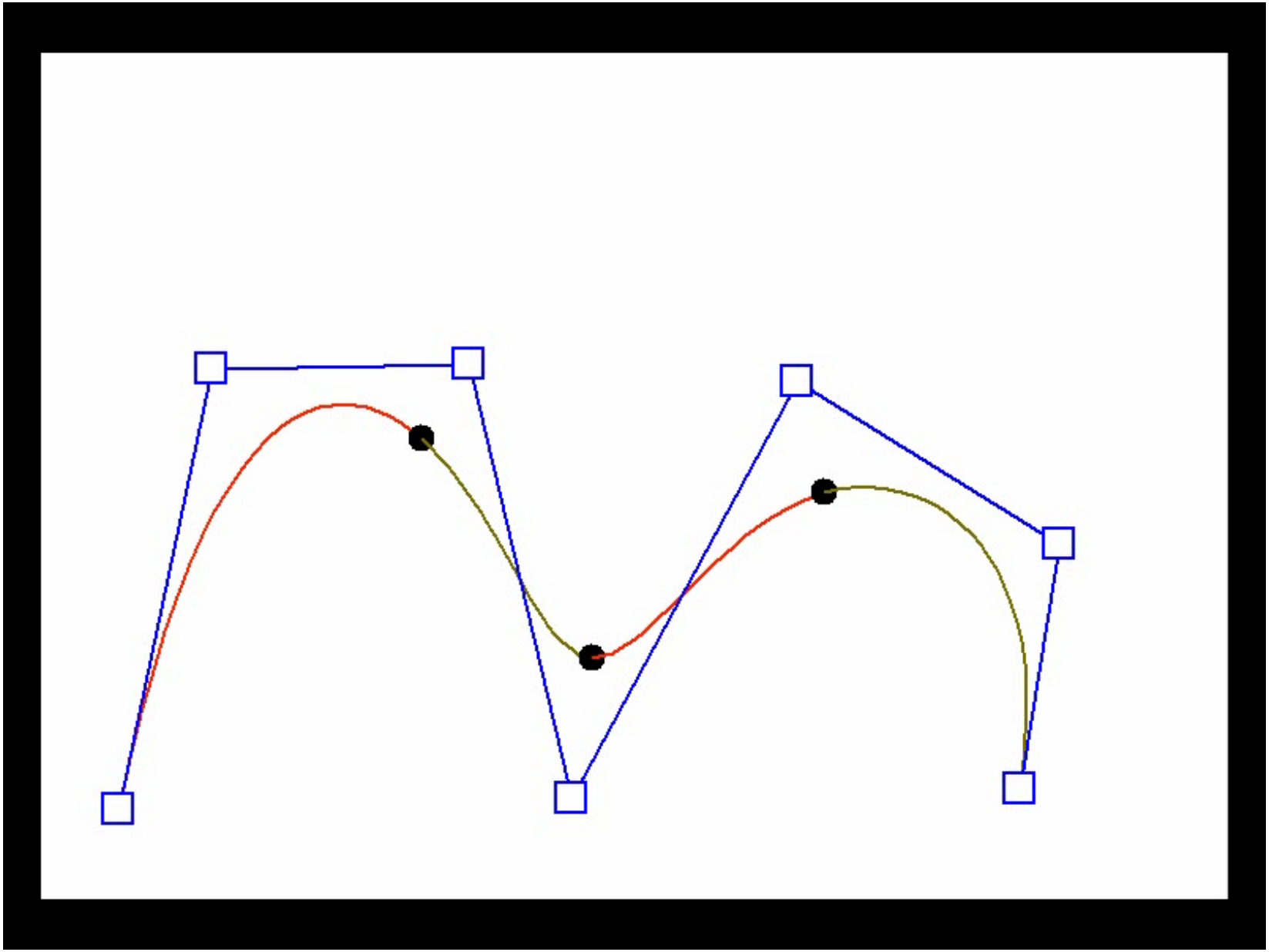
$N_i^n(u)$: B-spline basis function of degree n(=3)

u_j : Knots, j = 0, 1, …, K-1

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}, \quad \sum_{i=0}^{D-1} N_i^n(u) = 1$$

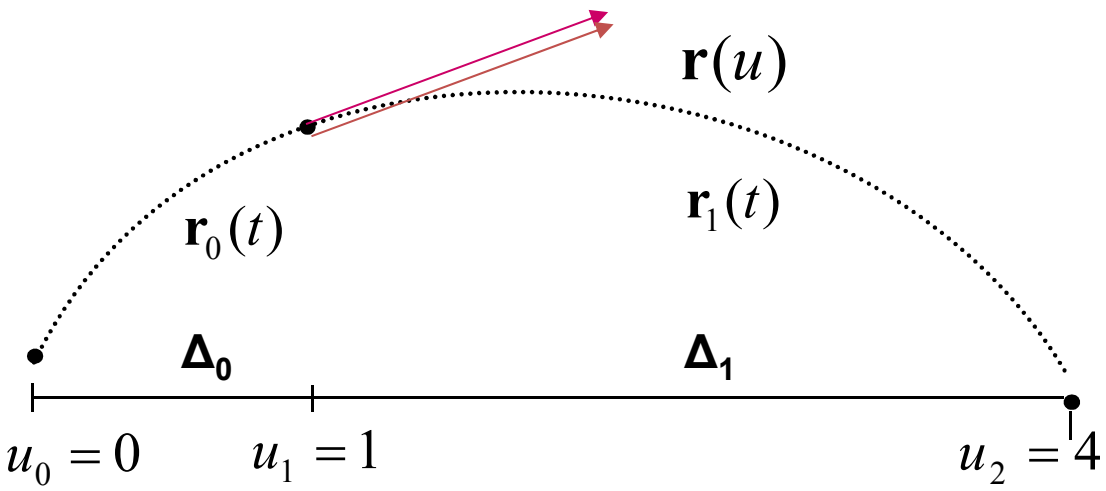
Computer Implementation of B-Spline Curve



3.3 C^1 and C^2 Continuity Condition

(1) 1st Derivatives of Cubic Bezier Curves at Junction Point
(2) C^1 Continuity Condition of Composite Curves
(3) 2nd Derivatives of Cubic Bezier Curves
(4) C^2 Continuity Condition of Composite Curves

1st Derivatives of Cubic Bezier Curves at Junction Point



Parameter transformation: u ➡ t
- u is a global parameter.
- t is a local parameter defined in [0, 1] section

The derivative value of $\frac{d\mathbf{r}(u)}{du}$ **in** $u_0 \le u \le u_1$

$$t = \frac{u - u_0}{u_1 - u_0} = \frac{u - u_0}{\Delta_0}$$

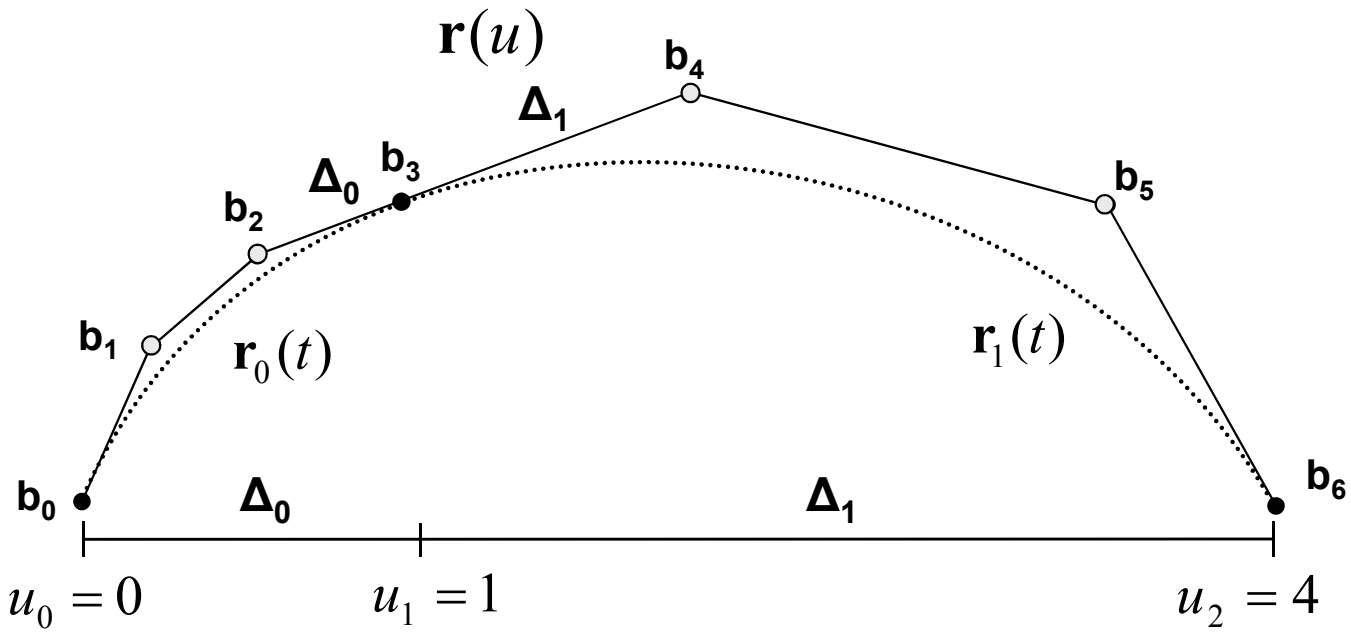
$$\frac{d\mathbf{r}(u)}{du} = \frac{d\,\mathbf{r}_0(u(t))}{dt}\frac{dt}{du} = \frac{1}{\Delta_0}\frac{d\,\mathbf{r}_0(t)}{dt}$$

The derivative value of $\frac{d\mathbf{r}(u)}{du}$ **in** $u_1 \le u \le u_2$

$$t = \frac{u - u_1}{u_2 - u_1} = \frac{u - u_1}{\Delta_1}$$

$$\frac{d\mathbf{r}(u)}{du} = \frac{d\,\mathbf{r}_1(t)}{dt}\frac{dt}{du} = \frac{1}{\Delta_1}\frac{d\,\mathbf{r}_1(t)}{dt}$$

C^1 Continuity Condition of Composite Curves at Junction Point



$\mathbf{r}(u = u_1) = \mathbf{r}_0(t = 1) = \mathbf{r}_1(t = 0)$. C^1 condition must satisfy at junction point.

$$\left.\frac{d\mathbf{r}(u)}{du}\right|_{u_1=1} = \frac{1}{\Delta_0}\left.\frac{d\mathbf{r}_0(t)}{dt}\right|_{t=1} = \frac{1}{\Delta_0}3(\mathbf{b}_3-\mathbf{b}_2)$$

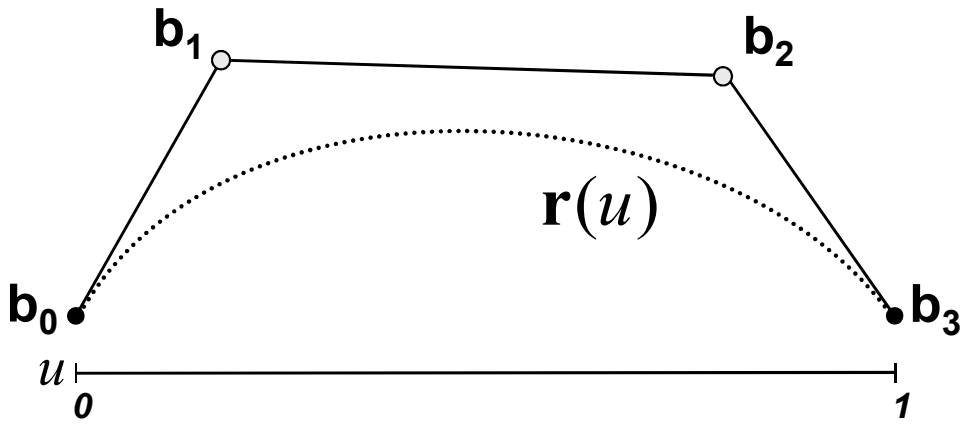
$$= \frac{1}{\Delta_1}\left.\frac{d\mathbf{r}_1(t)}{dt}\right|_{t=0} = \frac{1}{\Delta_1}\cdot 3(\mathbf{b}_4-\mathbf{b}_3)$$

$$(\mathbf{b}_3-\mathbf{b}_2):(\mathbf{b}_4-\mathbf{b}_3) = \Delta_0 : \Delta_1$$

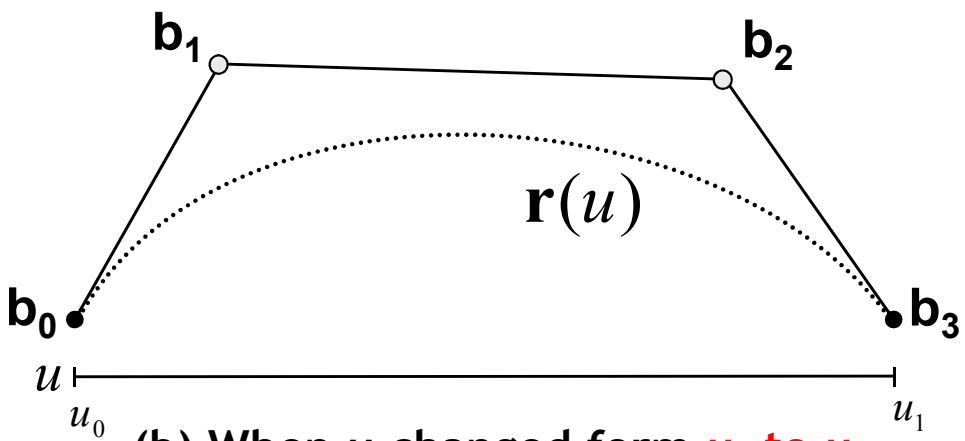
$$\mathbf{b}_3 = \frac{\Delta_1}{\Delta}\mathbf{b}_2 + \frac{\Delta_0}{\Delta}\mathbf{b}_4$$

- Suppose the parameter u is time, then, 1st derivative is velocity of the point which passes through the curve.
- If 1st derivative of the curve is continuous at the junction point b_3, then the velocity must be continuous.
- Accordingly, if the time interval is changed from $\triangle_0$ to $\triangle_1$. the distance must be changed proportionally, because the velocity is continuous at the junction point.

2nd Derivatives of Cubic Bezier Curves at Junction Point



(a) When u changed from 0 to 1



(b) When u changed form u_0 to u_1

The second derivative of cubic Bezier curve *at* u=1

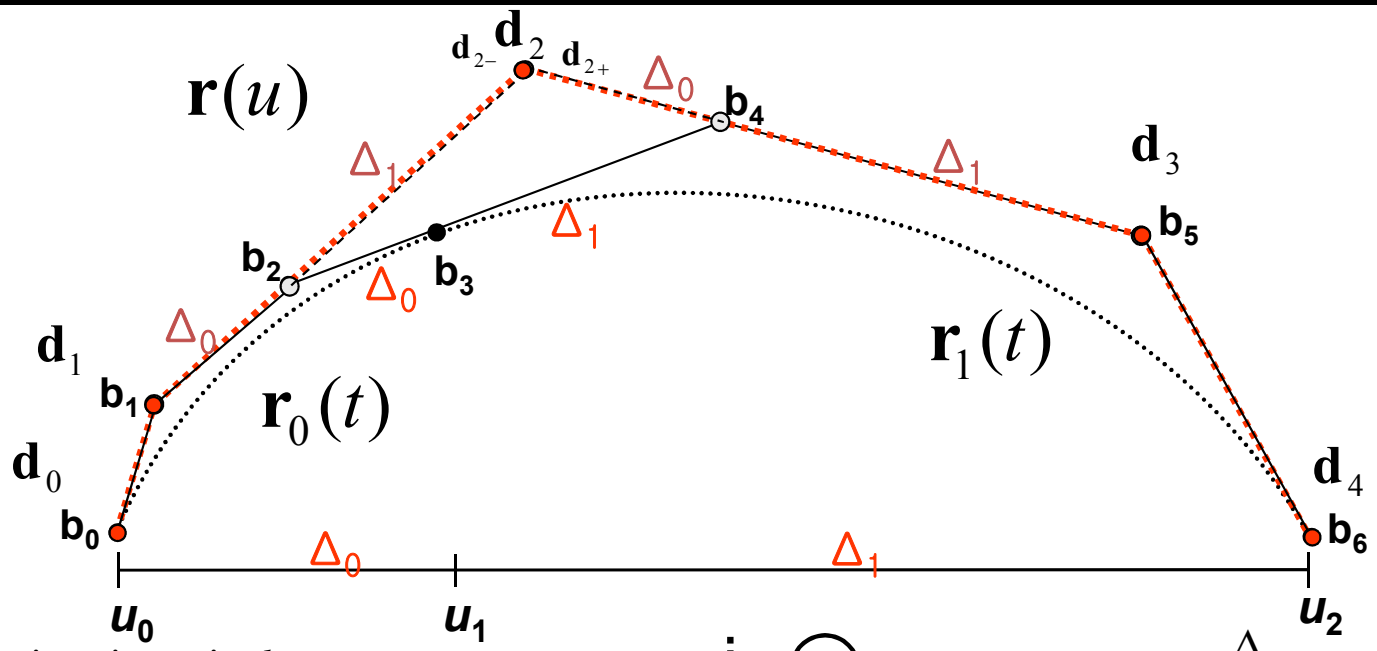
$$\frac{d^2\mathbf{r}(1)}{du^2} = 3(3-1)(\mathbf{b}_3 - 2\mathbf{b}_2 + \mathbf{b}_1)$$

The second derivative of cubic Bezier curve *at* u=1

$$\frac{d^2\,\mathbf{r}(u(t))}{du^2} = \frac{1}{(\Delta)^2}\frac{d^2\,\mathbf{r}(t)}{dt^2} \qquad (\Delta = u_1 - u_0)$$

$$\frac{d^2\mathbf{r}(u_1)}{du^2} = \frac{1}{(\Delta)^2}\frac{d^2\mathbf{r}(1)}{dt^2} = \frac{1}{(\Delta)^2}3(3-1)(\mathbf{b}_3 - 2\mathbf{b}_2 + \mathbf{b}_1)$$

C^2 Continuity Condition of Composite Curves at Junction Point



① C^2 condition at junction point b_3

$$\frac{d^2\mathbf{r}(u_{1-})}{du^2}=\frac{1}{(\Delta_0)^2}\frac{d^2\mathbf{r}_0(1)}{dt^2}=\frac{1}{(\Delta_0)^2}3(3-1)(\mathbf{b}_3-2\mathbf{b}_2+\mathbf{b}_1)$$

$$\frac{d^2\mathbf{r}(u_{1+})}{du^2}=\frac{1}{(\Delta_1)^2}\frac{d^2\mathbf{r}_1(0)}{dt^2}=\frac{1}{(\Delta_1)^2}3(3-1)(\mathbf{b}_5-2\mathbf{b}_4+\mathbf{b}_3)$$

② Since $\frac{d^2\mathbf{r}(u_{1-})}{du^2}=\frac{d^2\mathbf{r}(u_{1+})}{du^2}$

$$\frac{6}{(\Delta_0)^2}(\mathbf{b}_3-2\mathbf{b}_2+\mathbf{b}_1)=\frac{6}{(\Delta_1)^2}(\mathbf{b}_5-2\mathbf{b}_4+\mathbf{b}_3)$$

③ And substitute C^1 condition$(\mathbf{b}_3=\frac{\Delta_1}{\Delta}\mathbf{b}_2+\frac{\Delta_0}{\Delta}\mathbf{b}_4)$.

To summarize is

$$\Rightarrow -\frac{\Delta_1}{\Delta_0}\mathbf{b}_1+\frac{\Delta}{\Delta_0}\mathbf{b}_2=\frac{\Delta}{\Delta_1}\mathbf{b}_4-\frac{\Delta_0}{\Delta_1}\mathbf{b}_5$$

④ If LHS is $\mathbf{d}_{2-}=-\frac{\Delta_1}{\Delta_0}\mathbf{b}_1+\frac{\Delta}{\Delta_0}\mathbf{b}_2$

$$\mathbf{b}_2=\frac{\Delta_1}{\Delta}\mathbf{b}_1+\frac{\Delta_0}{\Delta}\mathbf{d}_{2-}$$

⑤ If RHS is $\mathbf{d}_{2+}=\frac{\Delta}{\Delta_1}\mathbf{b}_4-\frac{\Delta_0}{\Delta_1}\mathbf{b}_5$

$$\mathbf{b}_4=\frac{\Delta_1}{\Delta}\mathbf{d}_{2+}+\frac{\Delta_1}{\Delta}\mathbf{b}_5$$

⑥ Thus, if the points $\mathbf{d}_{2-}=\mathbf{d}_{2+}=\mathbf{d}_2$ exist, C^2 condition is satisfied.

⑦ C^2 condition at junction point

$$-\frac{\Delta_1}{\Delta_0}\mathbf{b}_1+\frac{\Delta}{\Delta_0}\mathbf{b}_2=\frac{\Delta}{\Delta_1}\mathbf{b}_4-\frac{\Delta_0}{\Delta_1}\mathbf{b}_5$$

$$ratio(\mathbf{b}_1,\mathbf{b}_2,d_2)=ratio(d_2,\mathbf{b}_4,\mathbf{b}_5)=\frac{\Delta_0}{\Delta_1}$$

3.4 B-Spline Curve Interpolation

(1) Determination of the Number of Curve Segments and Knots Values

(2) Problem Definition of B-Spline Curve Interpolation

(3) Determination of Bezier End Control Points by End Tangent Vectors

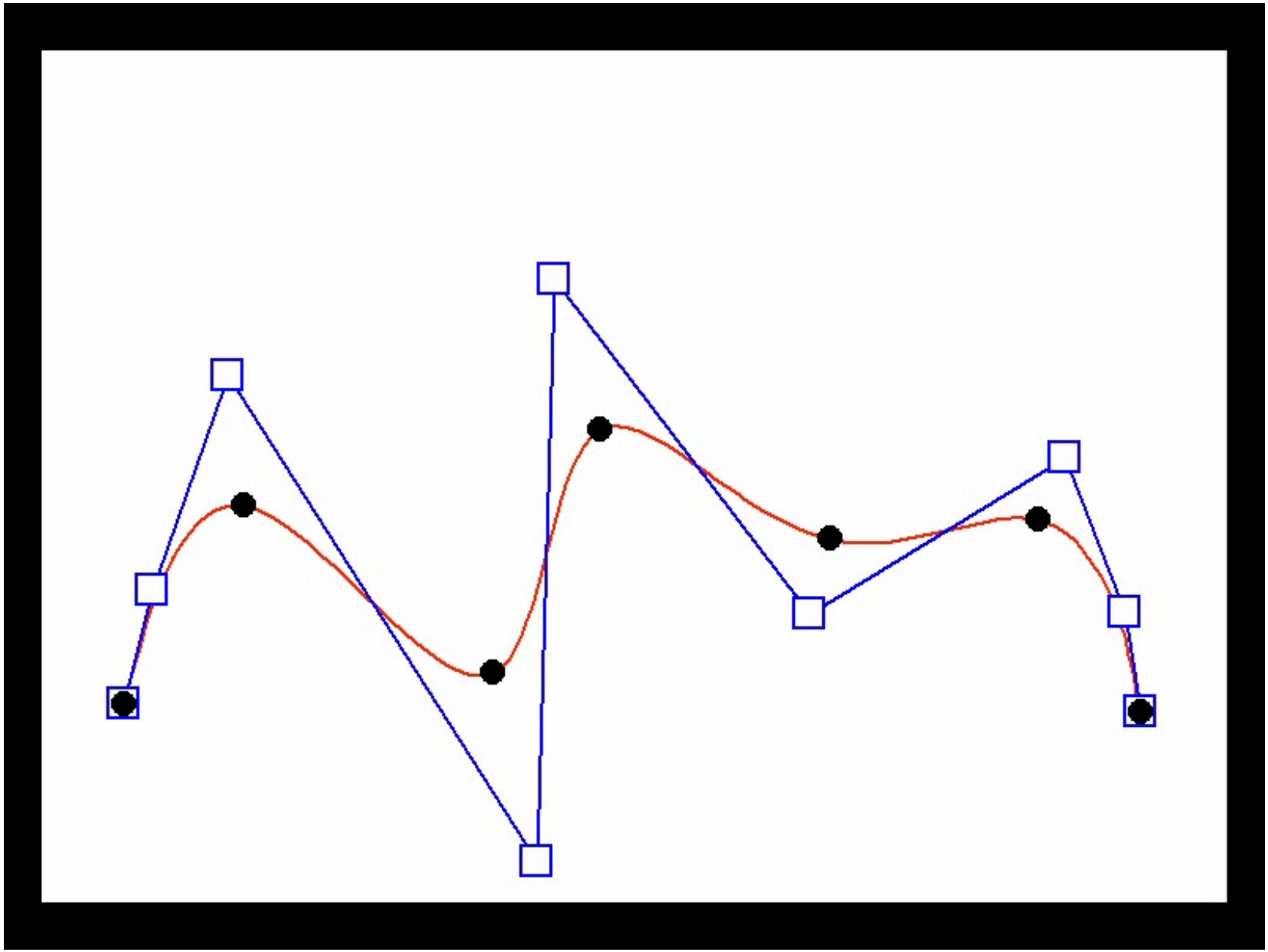
(4) Determination of Bezier Control Points Satisfying C^1 Continuity Condition

(5) Determination of B-Spline Control Points Satisfying C^2 Continuity Condition

(6) Calculation of B-Spline Control Points by Using Tri-diagonal Matrix Solution

(7) Bessel End Condition

Example of B-Spline Curve Interpolation



Determination of the Number of Bezier Curve Segments and Knot Values

☑ Given: Fitting points $\mathbf{P}_i$ and corresponding parameter t_i

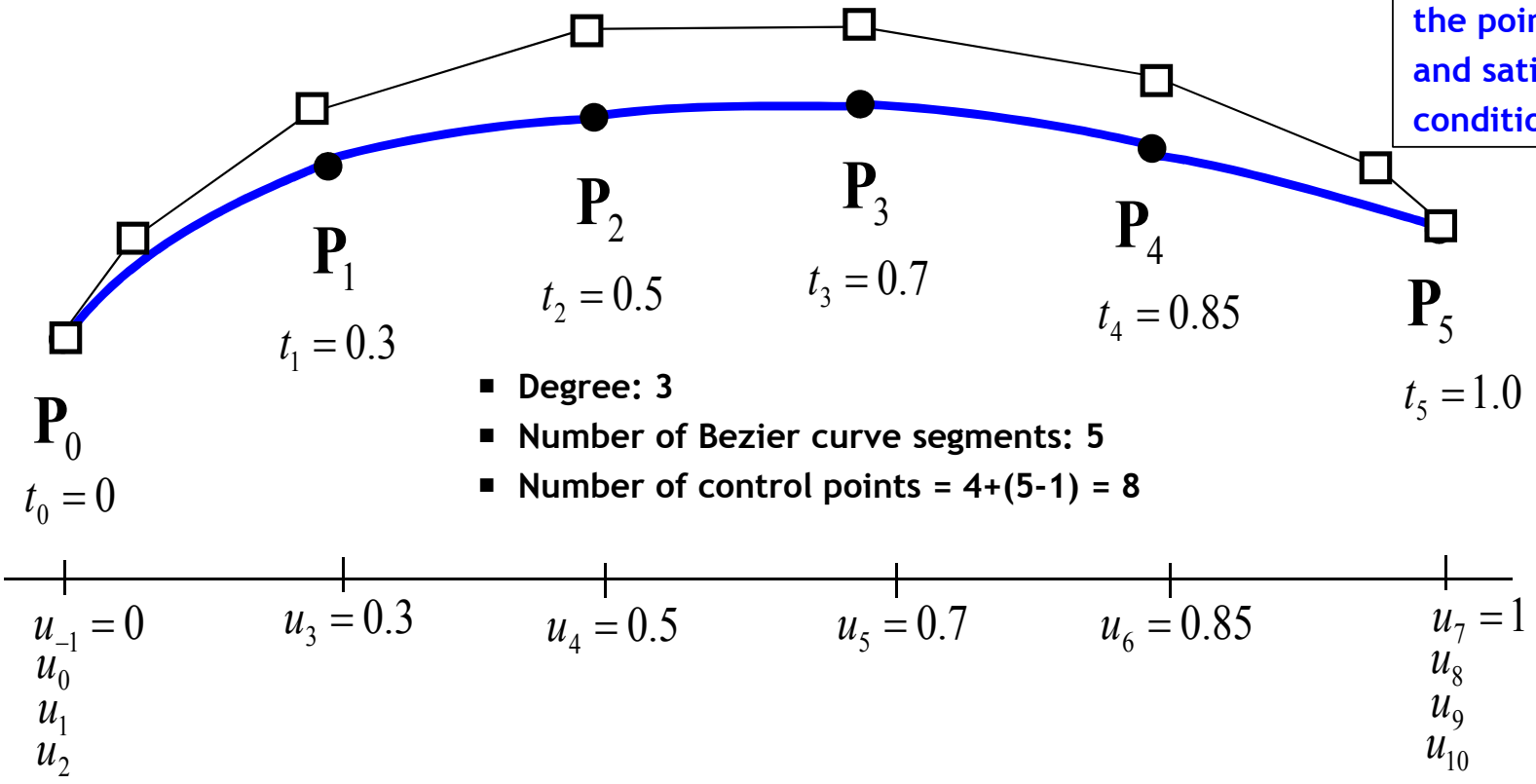
1. Determine the number of Bezier curve segments
(= Number of fitting points - 1)

2. We can determine knots to be same as the parameters t_i

3. How can we determine the B-spline control points?

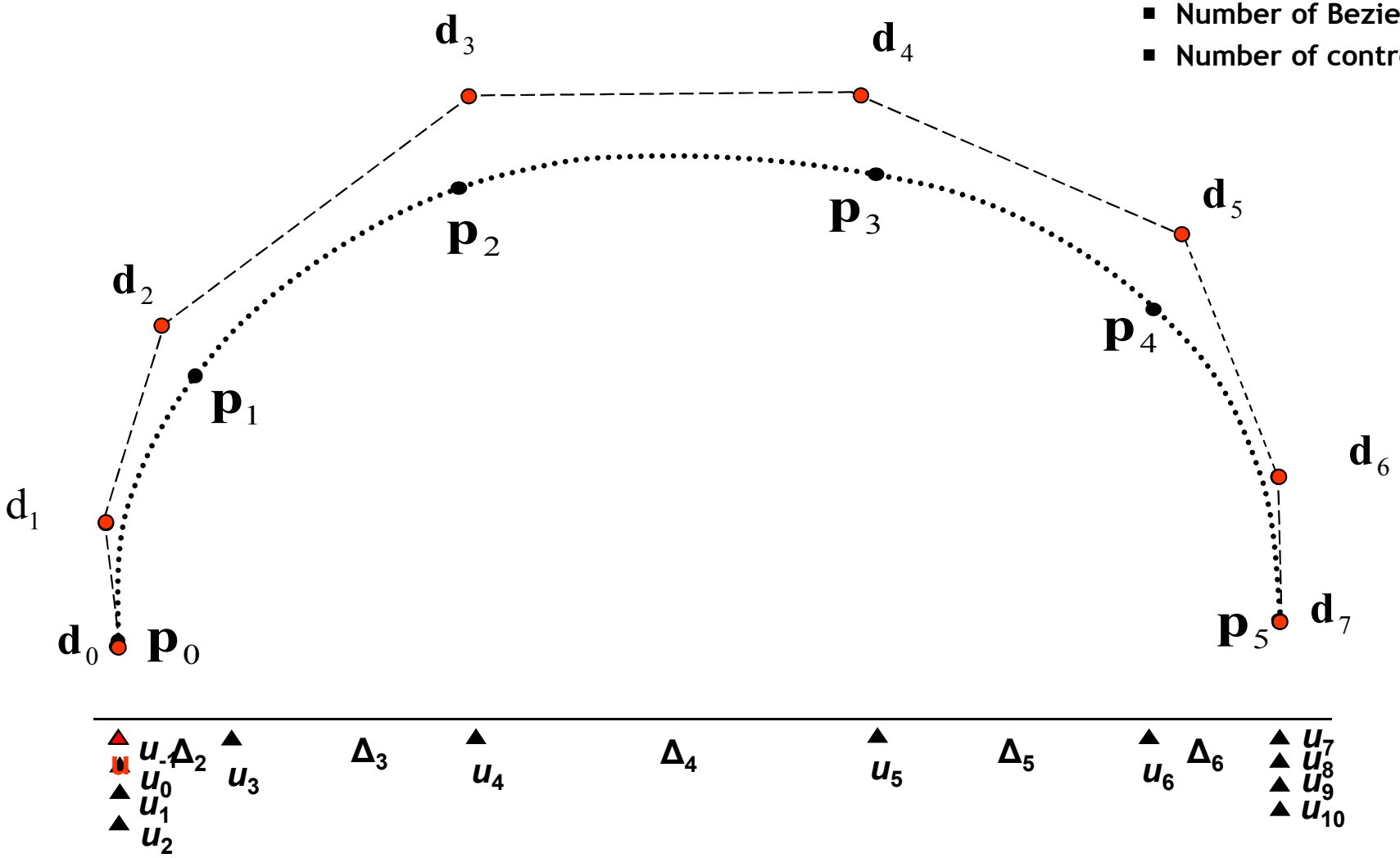
Given:
- Points p_i on the curve
- Knots u_j of the given points on the curve
- Tangent vectors t_0, t_1 at both ends

Find:
1. B-spline control point d_i
2. Cubic B-spline curve $r(u)$ passing through the points p_i on the curve and satisfying C^2 continuity condition.



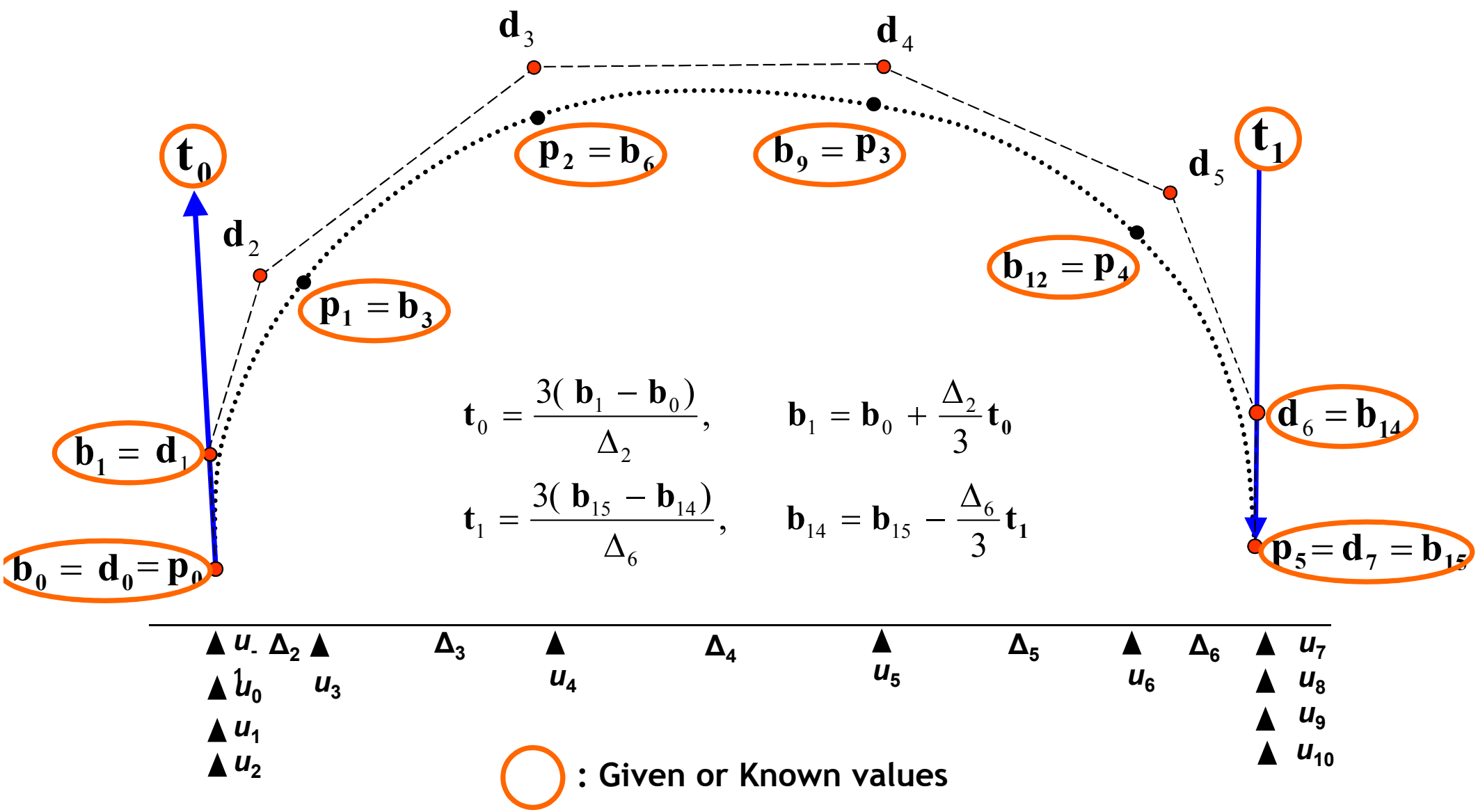
Problem Statement of an Example of a Cubic B-Spline Curve Interpolation

- Degree: 3
- Number of Bezier curve segments: 5
- Number of control points = 4+(5-1) = 8

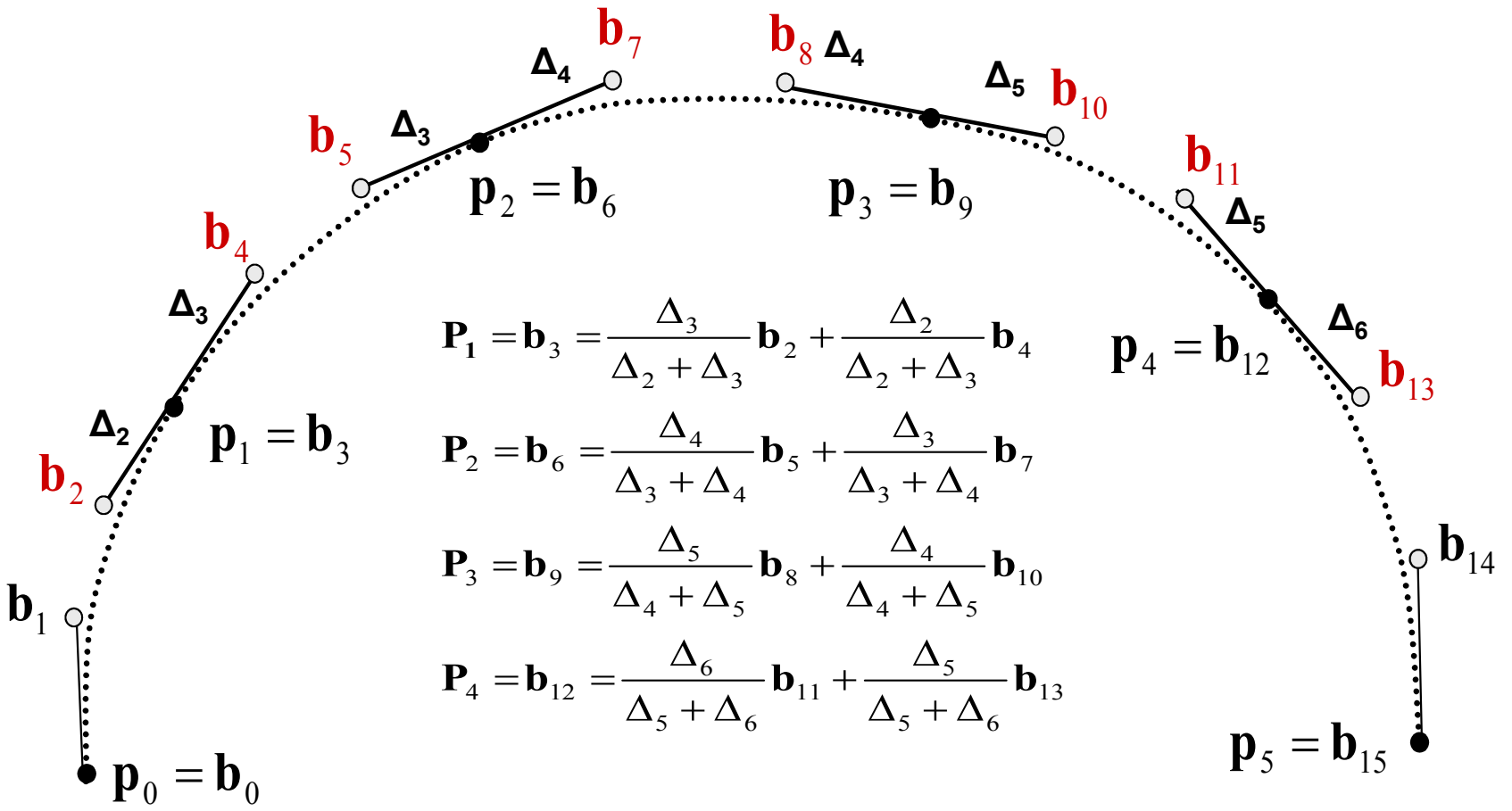


Assumption of basis function: Each curve segment is 3rd-degree Bezier curve.
Constraints: Satisfying C^0, C^1, C^2 continuity conditions at knots

Determination of Bezier End Control Points by Using End Tangent Vectors



Determination of Bezier Control Points Satisfying C^0 and C^1 Continuity Conditions



$$\mathbf{P}_1 = \mathbf{b}_3 = \frac{\Delta_3}{\Delta_2 + \Delta_3}\mathbf{b}_2 + \frac{\Delta_2}{\Delta_2 + \Delta_3}\mathbf{b}_4$$

$$\mathbf{P}_2 = \mathbf{b}_6 = \frac{\Delta_4}{\Delta_3 + \Delta_4}\mathbf{b}_5 + \frac{\Delta_3}{\Delta_3 + \Delta_4}\mathbf{b}_7$$

$$\mathbf{P}_3 = \mathbf{b}_9 = \frac{\Delta_5}{\Delta_4 + \Delta_5}\mathbf{b}_8 + \frac{\Delta_4}{\Delta_4 + \Delta_5}\mathbf{b}_{10}$$

$$\mathbf{P}_4 = \mathbf{b}_{12} = \frac{\Delta_6}{\Delta_5 + \Delta_6}\mathbf{b}_{11} + \frac{\Delta_5}{\Delta_5 + \Delta_6}\mathbf{b}_{13}$$

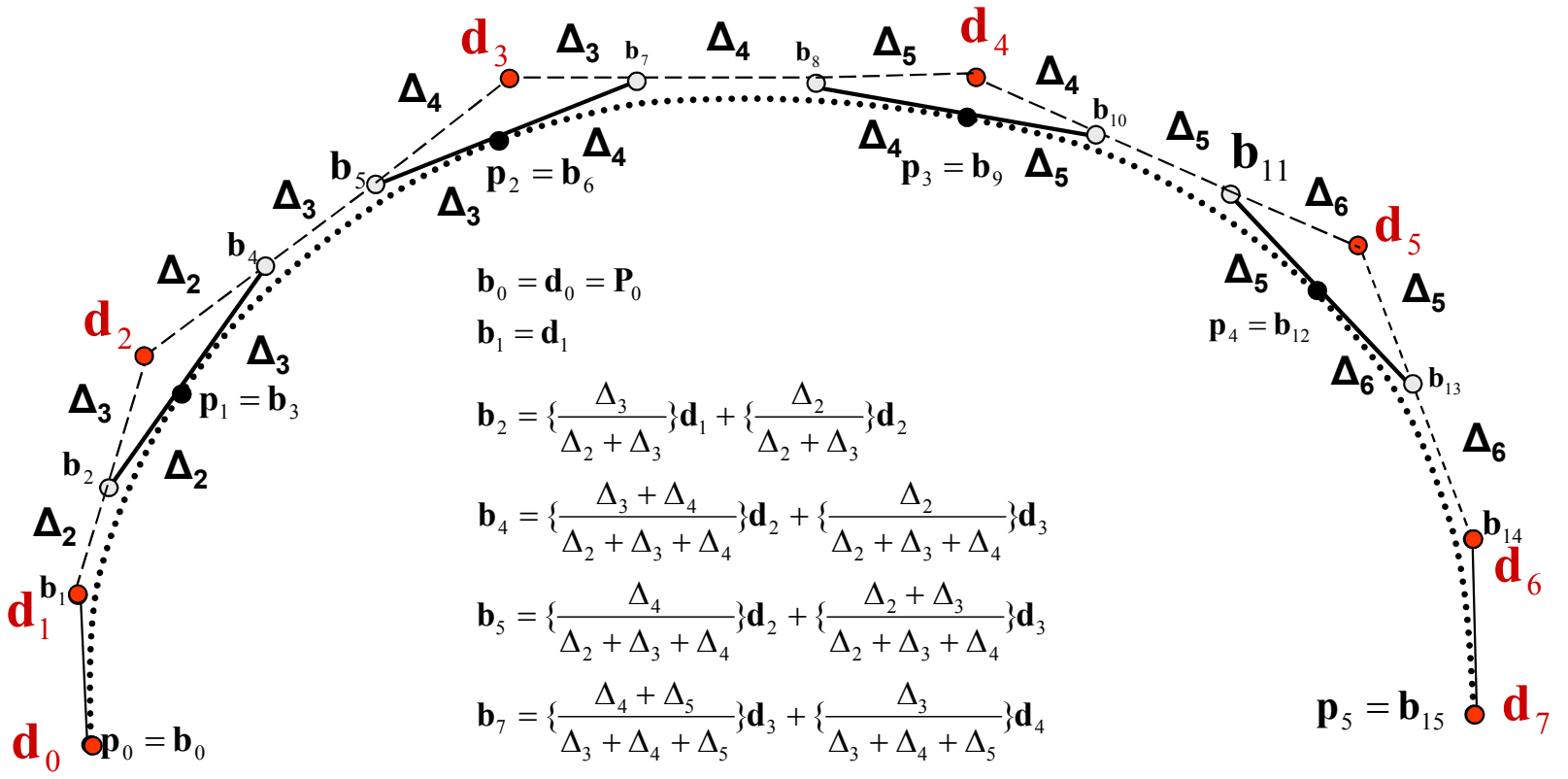
$$(\mathbf{b}_3 - \mathbf{b}_2) : (\mathbf{b}_4 - \mathbf{b}_3) = \Delta_2 : \Delta_3$$

$$(\mathbf{b}_6 - \mathbf{b}_5) : (\mathbf{b}_7 - \mathbf{b}_6) = \Delta_3 : \Delta_4$$

$$(\mathbf{b}_9 - \mathbf{b}_8) : (\mathbf{b}_{10} - \mathbf{b}_9) = \Delta_4 : \Delta_5$$

$$(\mathbf{b}_{12} - \mathbf{b}_{11}) : (\mathbf{b}_{13} - \mathbf{b}_{12}) = \Delta_5 : \Delta_6$$

Determination of B-Spline Control Points Satisfying C^2 Continuity Condition



$$\mathbf{b}_0 = \mathbf{d}_0 = \mathbf{P}_0$$

$$\mathbf{b}_1 = \mathbf{d}_1$$

$$\mathbf{b}_2 = \{\frac{\Delta_3}{\Delta_2 + \Delta_3}\}\mathbf{d}_1 + \{\frac{\Delta_2}{\Delta_2 + \Delta_3}\}\mathbf{d}_2$$

$$\mathbf{b}_4 = \{\frac{\Delta_3 + \Delta_4}{\Delta_2 + \Delta_3 + \Delta_4}\}\mathbf{d}_2 + \{\frac{\Delta_2}{\Delta_2 + \Delta_3 + \Delta_4}\}\mathbf{d}_3$$

$$\mathbf{b}_5 = \{\frac{\Delta_4}{\Delta_2 + \Delta_3 + \Delta_4}\}\mathbf{d}_2 + \{\frac{\Delta_2 + \Delta_3}{\Delta_2 + \Delta_3 + \Delta_4}\}\mathbf{d}_3$$

$$\mathbf{b}_7 = \{\frac{\Delta_4 + \Delta_5}{\Delta_3 + \Delta_4 + \Delta_5}\}\mathbf{d}_3 + \{\frac{\Delta_3}{\Delta_3 + \Delta_4 + \Delta_5}\}\mathbf{d}_4$$

$$\mathbf{b}_8 = \{\frac{\Delta_5}{\Delta_3 + \Delta_4 + \Delta_5}\}\mathbf{d}_3 + \{\frac{\Delta_3 + \Delta_4}{\Delta_3 + \Delta_4 + \Delta_5}\}\mathbf{d}_4$$

$$\mathbf{b}_{10} = \{\frac{\Delta_5 + \Delta_6}{\Delta_4 + \Delta_5 + \Delta_6}\}\mathbf{d}_4 + \{\frac{\Delta_4}{\Delta_4 + \Delta_5 + \Delta_6}\}\mathbf{d}_5$$

$$\mathbf{b}_{11} = \{\frac{\Delta_6}{\Delta_4 + \Delta_5 + \Delta_6}\}\mathbf{d}_4 + \{\frac{\Delta_4 + \Delta_5}{\Delta_4 + \Delta_5 + \Delta_6}\}\mathbf{d}_5$$

$$\mathbf{b}_{13} = \{\frac{\Delta_6}{\Delta_5 + \Delta_6}\}\mathbf{d}_5 + \{\frac{\Delta_5}{\Delta_5 + \Delta_6}\}\mathbf{d}_6$$

$$\mathbf{b}_{14} = \mathbf{d}_6$$

$$\mathbf{b}_{15} = \mathbf{d}_7 = \mathbf{p}_5$$

Formulation of Equations to Determine d_i Satisfying C^0, C^1, and C^2 Conditions

C0, C¹ Conditions

$$\mathbf{P}_1 = \mathbf{b}_3 = \frac{\Delta_3}{\Delta_2 + \Delta_3}\mathbf{b}_2 + \frac{\Delta_2}{\Delta_2 + \Delta_3}\mathbf{b}_4$$

$$\mathbf{P}_2 = \mathbf{b}_6 = \frac{\Delta_4}{\Delta_3 + \Delta_4}\mathbf{b}_5 + \frac{\Delta_3}{\Delta_3 + \Delta_4}\mathbf{b}_7$$

$$\mathbf{P}_3 = \mathbf{b}_9 = \frac{\Delta_5}{\Delta_4 + \Delta_5}\mathbf{b}_8 + \frac{\Delta_4}{\Delta_4 + \Delta_5}\mathbf{b}_{10}$$

$$\mathbf{P}_4 = \mathbf{b}_{12} = \frac{\Delta_6}{\Delta_5 + \Delta_6}\mathbf{b}_{11} + \frac{\Delta_5}{\Delta_5 + \Delta_6}\mathbf{b}_{13}$$

C² Condition

$$\mathbf{b}_0 = \mathbf{d}_0 = \mathbf{P}_0$$

$$\mathbf{b}_1 = \mathbf{d}_1$$

$$\mathbf{b}_2 = \left\{\frac{\Delta_3}{\Delta_2 + \Delta_3}\right\}\mathbf{d}_1 + \left\{\frac{\Delta_2}{\Delta_2 + \Delta_3}\right\}\mathbf{d}_2$$

$$\mathbf{b}_4 = \left\{\frac{\Delta_3 + \Delta_4}{\Delta_2 + \Delta_3 + \Delta_4}\right\}\mathbf{d}_2 + \left\{\frac{\Delta_2}{\Delta_2 + \Delta_3 + \Delta_4}\right\}\mathbf{d}_3$$

$$\mathbf{b}_5 = \left\{\frac{\Delta_4}{\Delta_2 + \Delta_3 + \Delta_4}\right\}\mathbf{d}_2 + \left\{\frac{\Delta_2 + \Delta_3}{\Delta_2 + \Delta_3 + \Delta_4}\right\}\mathbf{d}_3$$

$$\mathbf{b}_7 = \left\{\frac{\Delta_4 + \Delta_5}{\Delta_3 + \Delta_4 + \Delta_5}\right\}\mathbf{d}_3 + \left\{\frac{\Delta_3}{\Delta_3 + \Delta_4 + \Delta_5}\right\}\mathbf{d}_4$$

$$\mathbf{b}_8 = \left\{\frac{\Delta_5}{\Delta_3 + \Delta_4 + \Delta_5}\right\}\mathbf{d}_3 + \left\{\frac{\Delta_3 + \Delta_4}{\Delta_3 + \Delta_4 + \Delta_5}\right\}\mathbf{d}_4$$

$$\mathbf{b}_{10} = \left\{\frac{\Delta_5 + \Delta_6}{\Delta_4 + \Delta_5 + \Delta_6}\right\}\mathbf{d}_4 + \left\{\frac{\Delta_4}{\Delta_4 + \Delta_5 + \Delta_6}\right\}\mathbf{d}_5$$

$$\mathbf{b}_{11} = \left\{\frac{\Delta_6}{\Delta_4 + \Delta_5 + \Delta_6}\right\}\mathbf{d}_4 + \left\{\frac{\Delta_4 + \Delta_5}{\Delta_4 + \Delta_5 + \Delta_6}\right\}\mathbf{d}_5$$

$$\mathbf{b}_{13} = \left\{\frac{\Delta_6}{\Delta_5 + \Delta_6}\right\}\mathbf{d}_5 + \left\{\frac{\Delta_5}{\Delta_5 + \Delta_6}\right\}\mathbf{d}_6$$

$$\mathbf{b}_{14} = \mathbf{d}_6$$

$$\mathbf{b}_{15} = \mathbf{d}_7 = \mathbf{p}_5$$

Expression of the Points p_i on the Curve with B-Spline Control Points d_i and Representation of Its Matrix Form

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

$$\mathbf{P}_1 = \frac{1}{(\Delta_2+\Delta_3)(\Delta_2+\Delta_3+\Delta_4)}[(\Delta_3)^2(\Delta_2+\Delta_3+\Delta_4)/(\Delta_2+\Delta_3)\mathbf{d}_1$$
$$+\{\Delta_2\Delta_3(\Delta_2+\Delta_3+\Delta_4)+\Delta_2(\Delta_2+\Delta_3)(\Delta_3+\Delta_4)\}/(\Delta_2+\Delta_3)\mathbf{d}_2+(\Delta_2)^2\mathbf{d}_3]$$
$$=\alpha_1\mathbf{d}_1+\beta_1\mathbf{d}_2+\gamma_1\mathbf{d}_3$$

$$\mathbf{P}_2 = \frac{1}{(\Delta_3+\Delta_4)(\Delta_3+\Delta_4+\Delta_5)}[(\Delta_4)^2\mathbf{d}_2+\{\Delta_4(\Delta_2+\Delta_3)+$$
$$\Delta_3(\Delta_4+\Delta_5)\}\mathbf{d}_3+(\Delta_3)^2\mathbf{d}_4]=\alpha_2\mathbf{d}_2+\beta_2\mathbf{d}_3+\gamma_2\mathbf{d}_4$$

$$\mathbf{P}_3 = \frac{1}{(\Delta_4+\Delta_5)(\Delta_3+\Delta_4+\Delta_5)}[(\Delta_5)^2\mathbf{d}_3+\{\Delta_5(\Delta_3+\Delta_4)(\Delta_4+\Delta_5+\Delta_6)$$
$$+\Delta_4(\Delta_5+\Delta_6)(\Delta_3+\Delta_4+\Delta_5)\}/(\Delta_4+\Delta_5+\Delta_6)\mathbf{d}_4+(\Delta_4)^2(\Delta_3+\Delta_4+\Delta_5)$$
$$/(\Delta_4+\Delta_5+\Delta_6)\mathbf{d}_5]=\alpha_3\mathbf{d}_3+\beta_3\mathbf{d}_4+\gamma_3\mathbf{d}_5$$

$$\mathbf{P}_4 = \frac{1}{(\Delta_5+\Delta_6)(\Delta_4+\Delta_5+\Delta_6)}[(\Delta_6)^2\mathbf{d}_4+$$
$$\{\Delta_6(\Delta_4+\Delta_5)+\Delta_5\Delta_6(\Delta_4+\Delta_5+\Delta_6)\}\mathbf{d}_5$$
$$+(\Delta_5)^2(\Delta_4+\Delta_5+\Delta_6)\mathbf{d}_6]=\alpha_4\mathbf{d}_4+\beta_4\mathbf{d}_5+\gamma_4\mathbf{d}_6$$

$$\mathbf{P}_5 = \mathbf{d}_7$$

$$\alpha_i = \frac{(\Delta_{i+2})^2}{(\Delta_i+\Delta_{i+1}+\Delta_{i+2})(\Delta_{i+1}+\Delta_{i+2})}$$
$$\beta_i = \{\frac{\Delta_{i+2}(\Delta_i+\Delta_{i+1})}{(\Delta_i+\Delta_{i+1}+\Delta_{i+2})}+\frac{\Delta_{i+1}(\Delta_{i+2}+\Delta_{i+3})}{(\Delta_{i+1}+\Delta_{i+2}+\Delta_{i+3})}\}/(\Delta_{i+1}+\Delta_{i+2})$$
$$\gamma_i = \frac{(\Delta_{i+1})^2}{(\Delta_{i+1}+\Delta_{i+2}+\Delta_{i+3})(\Delta_{i+1}+\Delta_{i+2})}$$

Known Unknown

$$\begin{vmatrix}\mathbf{p}_0\\ \mathbf{t}_0\\ \mathbf{p}_1\\ \mathbf{p}_2\\ \mathbf{p}_3\\ \mathbf{p}_4\\ \mathbf{t}_1\\ \mathbf{p}_5\end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ \frac{-3}{\Delta_2} & \frac{3}{\Delta_2} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & \alpha_1 & \beta_1 & \gamma_1 & 0 & 0 & 0 & 0\\ 0 & 0 & \alpha_2 & \beta_2 & \gamma_2 & 0 & 0 & 0\\ 0 & 0 & 0 & \alpha_3 & \beta_3 & \gamma_3 & 0 & 0\\ 0 & 0 & 0 & 0 & \alpha_4 & \beta_4 & \gamma_4 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{-3}{\Delta_6} & \frac{3}{\Delta_6}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\end{vmatrix} \begin{vmatrix}\mathbf{d}_0\\ \mathbf{d}_1\\ \mathbf{d}_2\\ \mathbf{d}_3\\ \mathbf{d}_4\\ \mathbf{d}_5\\ \mathbf{d}_6\\ \mathbf{d}_7\end{vmatrix}$$

Calculation of B-Spline Control Points d_i by Using Tri-diagonal Matrix Solution

$$
\begin{vmatrix} \mathbf{p}_0 \\ \mathbf{t}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{t}_1 \\ \mathbf{p}_5 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-3}{\Delta_2} & \frac{3}{\Delta_2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_1 & \beta_1 & \gamma_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_2 & \beta_2 & \gamma_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_3 & \beta_3 & \gamma_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha_4 & \beta_4 & \gamma_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{-3}{\Delta_6} & \frac{3}{\Delta_6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} \mathbf{d}_0 \\ \mathbf{d}_1 \\ \mathbf{d}_2 \\ \mathbf{d}_3 \\ \mathbf{d}_4 \\ \mathbf{d}_5 \\ \mathbf{d}_6 \\ \mathbf{d}_7 \end{vmatrix}
$$

$= \mathbf{D}$ (Known) $= \mathbf{A}$ (Known) $= \mathbf{X}$ (Unknown)

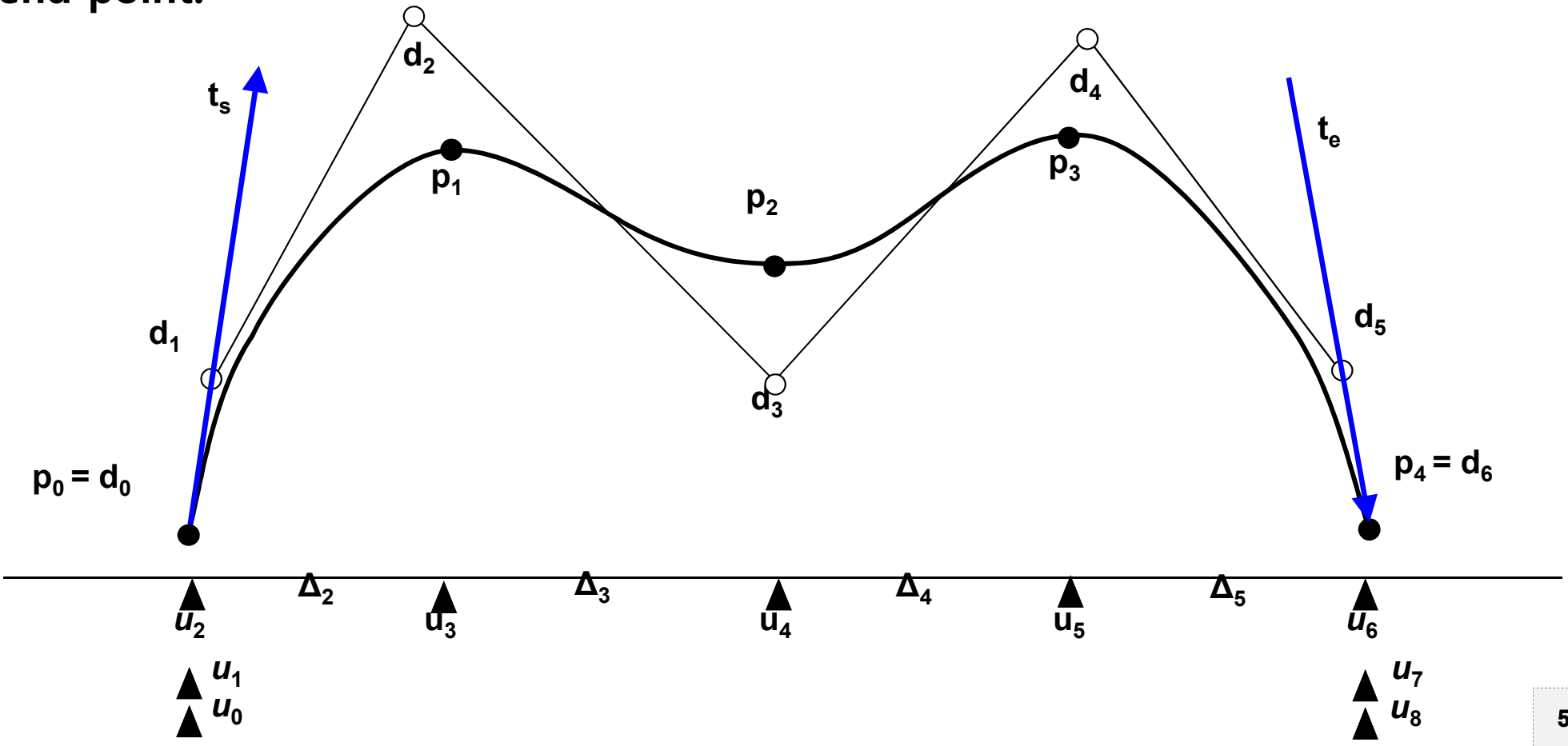
$$\mathbf{D} = \mathbf{A}\mathbf{X}$$

$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{D}$$

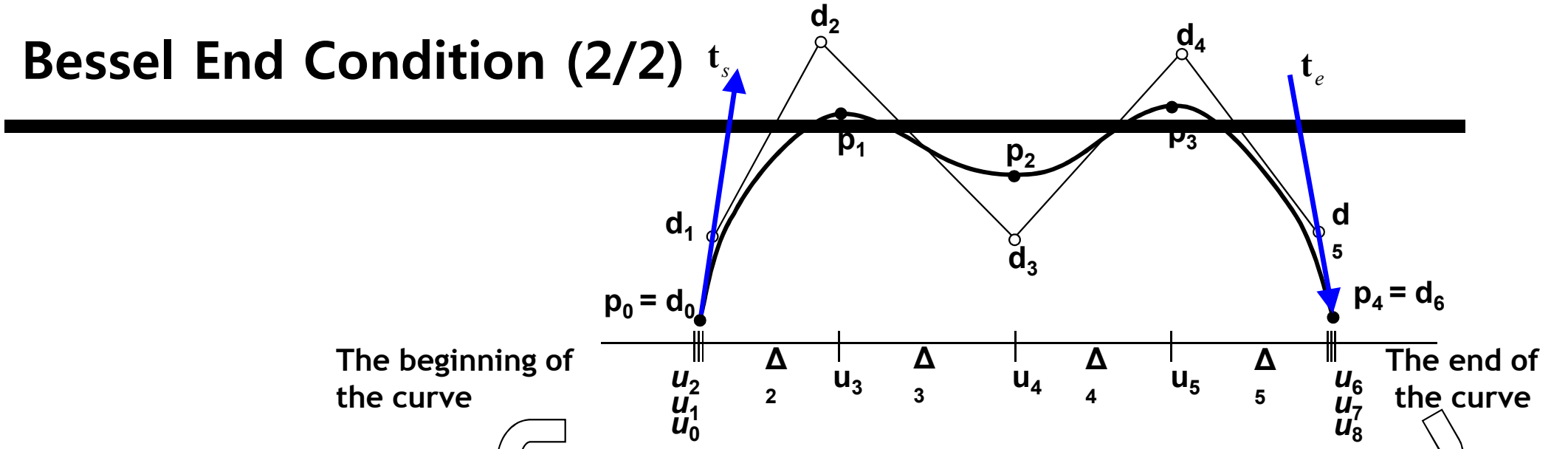
Because Matrix A is tri-diagonal matrix, the inverse matrix A^{-1} is easy to obtain.

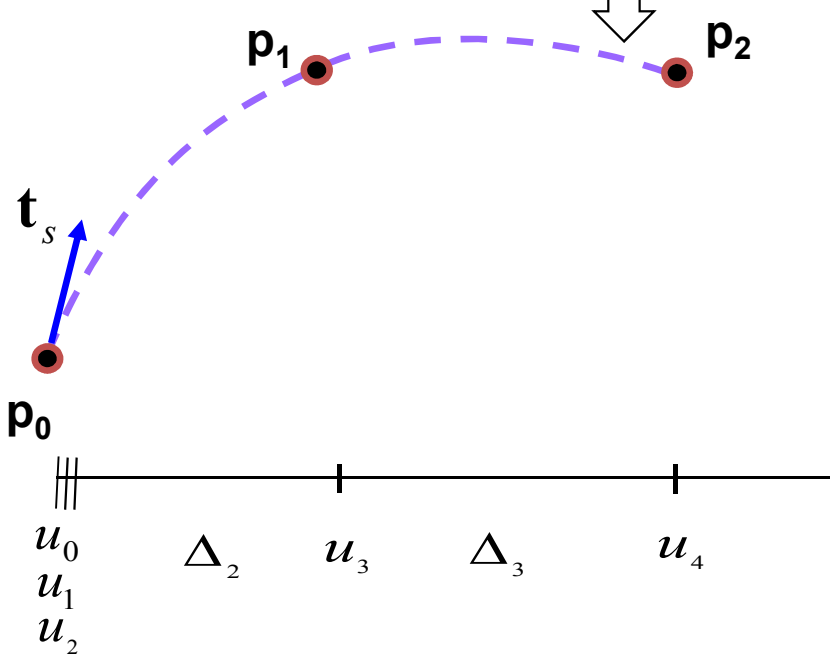
Bessel End Condition (1/2)

☑ If the tangent vectors t_s, t_e at both end points are not given,
(1) Construct 2nd-degree curve(quadratic curve) from the three consecutive points at both ends of the curve.
(2) And assume that the tangent vectors at each end point are the same with the value of the first derivatives of the constructed quadratic curves at each end point.

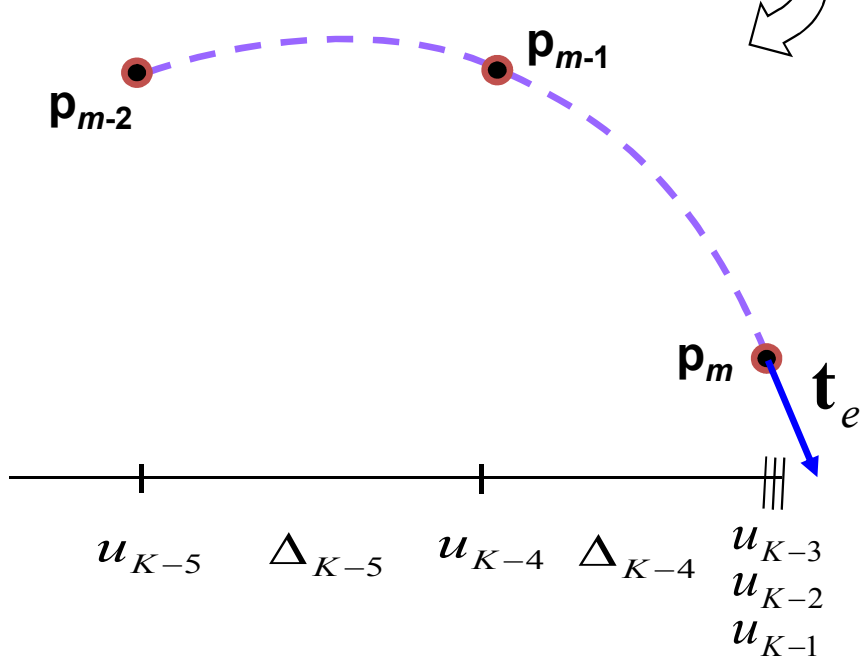


Bessel End Condition (2/2)



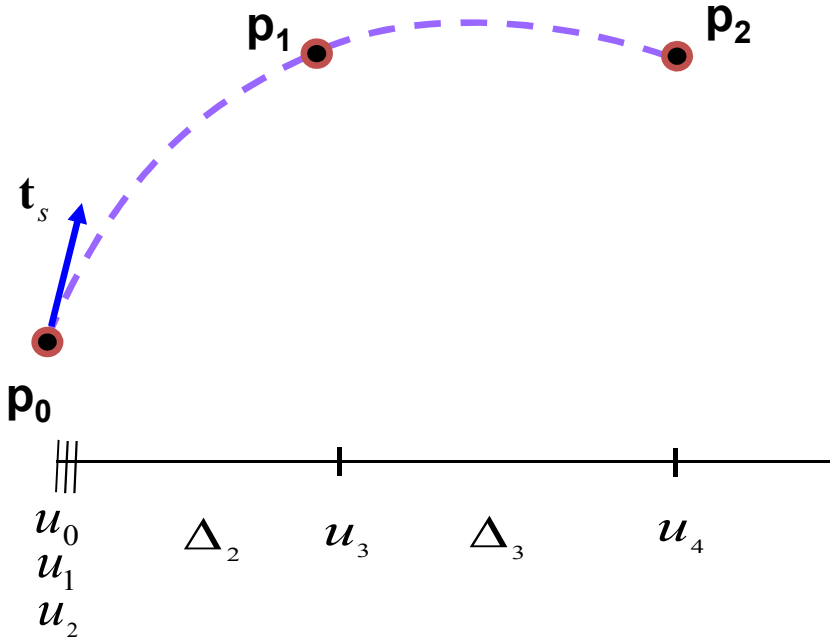


$$\mathbf{t}_s = \left(-\frac{2\Delta_2+\Delta_3}{\Delta_2(\Delta_2+\Delta_3)}\mathbf{p}_0 + \frac{(\Delta_2+\Delta_3)}{\Delta_2\Delta_3}\mathbf{p}_1 - \frac{\Delta_2}{\Delta_3(\Delta_2+\Delta_3)}\mathbf{p}_2\right)$$



$$\mathbf{t}_e = \left(\frac{\Delta_{K-4}}{\Delta_{K-5}(\Delta_{K-5}+\Delta_{K-4})}\mathbf{p}_{m-2} - \frac{(\Delta_{K-5}+\Delta_{K-4})}{\Delta_{K-5}\Delta_{K-4}}\mathbf{p}_{m-1} + \frac{(2\Delta_{K-4}+\Delta_{K-5})}{(\Delta_{K-5}+\Delta_{K-4})\Delta_{K-4}}\mathbf{p}_m\right)$$

Determination of t_s and t_e by Using Lagrange Polynomial (1/2)



$$\mathbf{x}(u) = L_0^2(u)\mathbf{p}_0 + L_1^2(u)\mathbf{p}_1 + L_2^2(u)\mathbf{p}_2$$

$$L_0^2(u) = \frac{(u-u_3)(u-u_4)}{(u_2-u_3)(u_2-u_4)} \qquad \dot{L}_0^2(u) = \frac{(u-u_4)+(u-u_3)}{(u_2-u_3)(u_2-u_4)}$$

$$L_1^2(u) = \frac{(u-u_2)(u-u_4)}{(u_3-u_2)(u_3-u_4)} \qquad \dot{L}_1^2(u) = \frac{(u-u_4)+(u-u_2)}{(u_3-u_2)(u_3-u_4)}$$

$$L_2^2(u) = \frac{(u-u_2)(u-u_3)}{(u_4-u_2)(u_4-u_3)} \qquad \dot{L}_2^2(u) = \frac{(u-u_3)+(u-u_2)}{(u_4-u_2)(u_4-u_3)}$$

$$\dot{\mathbf{x}}(u) = \dot{L}_0^2(u)\mathbf{p}_0 + \dot{L}_1^2(u)\mathbf{p}_1 + \dot{L}_2^2(u)\mathbf{p}_2$$

$$\dot{\mathbf{x}}(u_2) = \dot{L}_0^2(u_2)\mathbf{p}_0 + \dot{L}_1^2(u_2)\mathbf{p}_1 + \dot{L}_2^2(u_2)\mathbf{p}_2$$

$$= \frac{(u_2-u_4)+(u_2-u_3)}{(u_2-u_3)(u_2-u_4)}\mathbf{p}_0 + \frac{(u_2-u_4)+(u_2-u_2)}{(u_3-u_2)(u_3-u_4)}\mathbf{p}_1 + \frac{(u_2-u_3)+(u_2-u_2)}{(u_4-u_2)(u_4-u_3)}\mathbf{p}_2$$

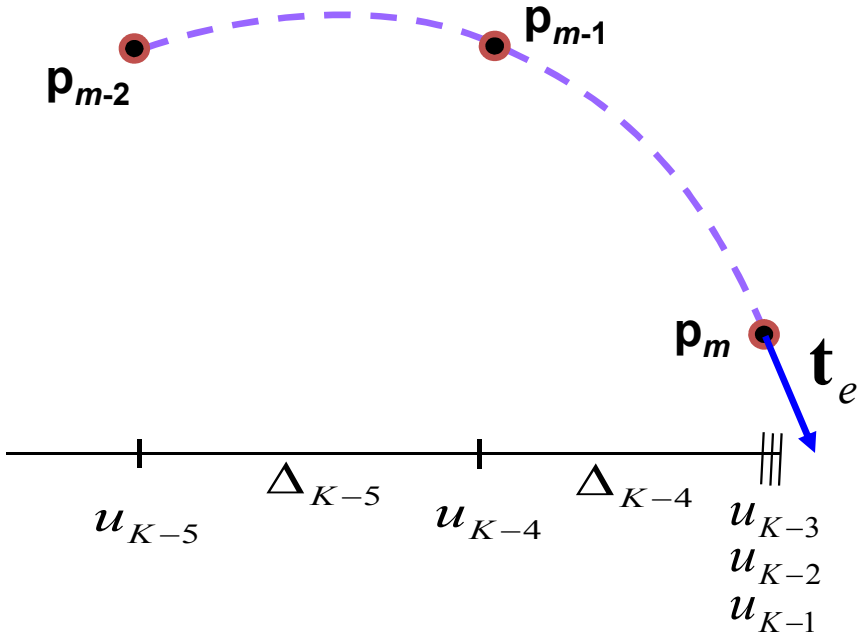
$$= \frac{-(\Delta_2+\Delta_3)-\Delta_2}{(-\Delta_2)(-(\Delta_2+\Delta_3))}\mathbf{p}_0 + \frac{-(\Delta_2+\Delta_3)}{\Delta_2(-\Delta_3)}\mathbf{p}_1 + \frac{-\Delta_2}{(\Delta_2+\Delta_3)(\Delta_2)}\mathbf{p}_2$$

$$= \left(-\frac{2\Delta_2+\Delta_3}{\Delta_2(\Delta_2+\Delta_3)}\mathbf{p}_0 + \frac{(\Delta_2+\Delta_3)}{\Delta_2\Delta_3}\mathbf{p}_1 - \frac{\Delta_2}{\Delta_3(\Delta_2+\Delta_3)}\mathbf{p}_2\right)$$

Therefore

$$\mathbf{t}_s = \left(-\frac{2\Delta_2+\Delta_3}{\Delta_2(\Delta_2+\Delta_3)}\mathbf{p}_0 + \frac{(\Delta_2+\Delta_3)}{\Delta_2\Delta_3}\mathbf{p}_1 - \frac{\Delta_2}{\Delta_3(\Delta_2+\Delta_3)}\mathbf{p}_2\right)$$

Determination of t_s and t_e by Using Lagrange Polynomial (2/2)



$$\mathbf{x}(u) = L_0^2(u)\mathbf{p}_{m-2} + L_1^2(u)\mathbf{p}_{m-1} + L_2^2(u)\mathbf{p}_m$$

$$L_0^2(u) = \frac{(u-u_{K-4})(u-u_{K-3})}{(u_{K-5}-u_{K-4})(u_{K-5}-u_{K-3})} \qquad \dot{L}_0^2(u) = \frac{(u-u_{K-3})+(u-u_{K-4})}{(u_{K-5}-u_{K-4})(u_{K-5}-u_{K-3})}$$

$$L_1^2(u) = \frac{(u-u_{K-5})(u-u_{K-3})}{(u_{K-4}-u_{K-5})(u_{K-4}-u_{K-3})} \qquad \dot{L}_1^2(u) = \frac{(u-u_{K-3})+(u-u_{K-5})}{(u_{K-4}-u_{K-5})(u_{K-4}-u_{K-3})}$$

$$L_2^2(u) = \frac{(u-u_{K-5})(u-u_{K-4})}{(u_{K-3}-u_{K-5})(u_{K-3}-u_{K-4})} \qquad \dot{L}_2^2(u) = \frac{(u-u_{K-4})+(u-u_{K-5})}{(u_{K-3}-u_{K-5})(u_{K-3}-u_{K-4})}$$

$$\dot{\mathbf{x}}(u) = \dot{L}_0^2(u)\mathbf{p}_{m-2} + \dot{L}_1^2(u)\mathbf{p}_{m-1} + \dot{L}_2^2(u)\mathbf{p}_m$$

$$\dot{\mathbf{x}}(u_{K-3}) = \dot{L}_0^2(u_{K-3})\mathbf{p}_{m-2} + \dot{L}_1^2(u_{K-3})\mathbf{p}_{m-1} + \dot{L}_2^2(u_{K-3})\mathbf{p}_m$$

$$= \frac{(u_{K-3}-u_{K-3})+(u_{K-3}-u_{K-4})}{(u_{K-5}-u_{K-4})(u_{K-5}-u_{K-3})}\mathbf{p}_{m-2} + \frac{(u_{K-3}-u_{K-3})+(u_{K-3}-u_{K-5})}{(u_{K-4}-u_{K-5})(u_{K-4}-u_{K-3})}\mathbf{p}_{m-1} + \frac{(u_{K-3}-u_{K-4})+(u_{K-3}-u_{K-5})}{(u_{K-3}-u_{K-5})(u_{K-3}-u_{K-4})}\mathbf{p}_m$$

$$= \frac{\Delta_{K-4}}{(-\Delta_{K-5})(-(\Delta_{K-4}+\Delta_{K-5}))}\mathbf{p}_{m-2} + \frac{-(\Delta_{K-4}+\Delta_{K-5})}{\Delta_{K-5}(-\Delta_{K-4})}\mathbf{p}_{m-1} + \frac{\Delta_{K-4}+(\Delta_{K-4}+\Delta_{K-5})}{(\Delta_{K-4}+\Delta_{K-5})(\Delta_{K-4})}\mathbf{p}_m$$

$$= \left(\frac{\Delta_{K-4}}{\Delta_{K-5}(\Delta_{K-5}+\Delta_{K-4})}\mathbf{p}_{m-2} - \frac{(\Delta_{K-5}+\Delta_{K-4})}{\Delta_{K-5}\Delta_{K-4}}\mathbf{p}_{m-1} + \frac{(2\Delta_{K-4}+\Delta_{K-5})}{(\Delta_{K-5}+\Delta_{K-4})\Delta_{K-4}}\mathbf{p}_m\right)$$

Therefore

$$\mathbf{t}_e = \left(\frac{\Delta_{K-4}}{\Delta_{K-5}(\Delta_{K-5}+\Delta_{K-4})}\mathbf{p}_{m-2} - \frac{(\Delta_{K-5}+\Delta_{K-4})}{\Delta_{K-5}\Delta_{K-4}}\mathbf{p}_{m-1} + \frac{(2\Delta_{K-4}+\Delta_{K-5})}{(\Delta_{K-5}+\Delta_{K-4})\Delta_{K-4}}\mathbf{p}_m\right)$$

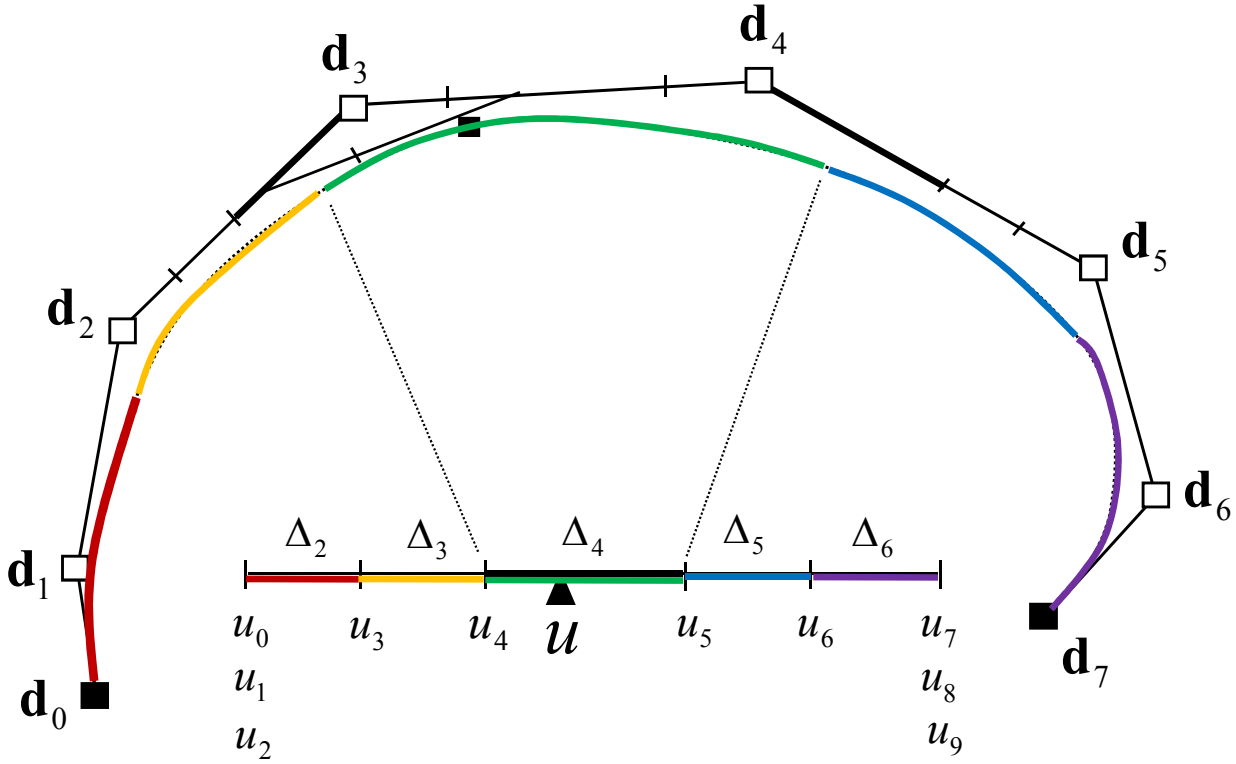
3.5 de Boor Algorithm (Generalization of the de Casteljau Algorithm)

Problem Statement of de Boor Algorithm

Input: $\mathbf{d}_i$ (de Boor Points)

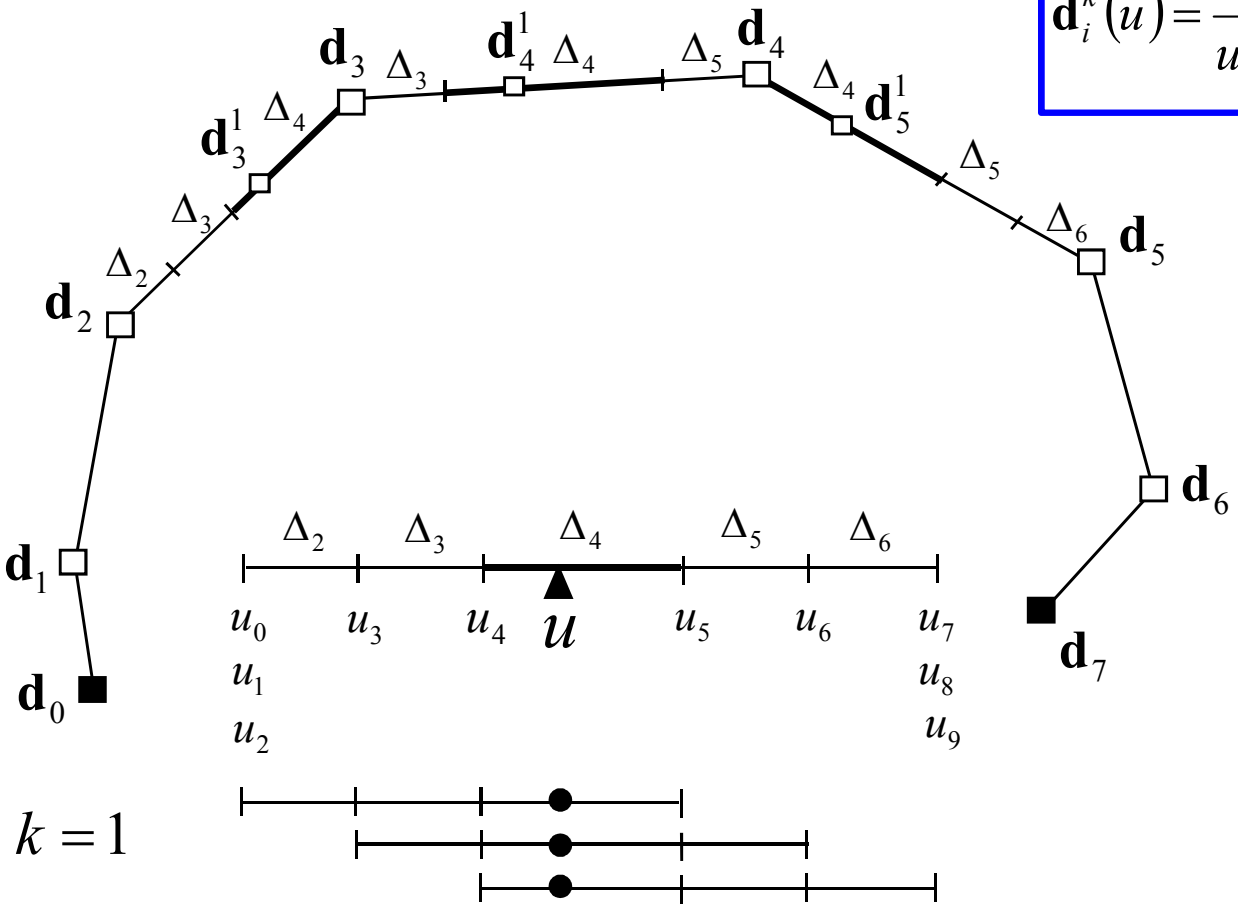
Processor: n-times sequential "linear interpolation" at u by marching (advancing) the segment of $\mathbf{d}_i\ \mathbf{d}_{i+1}$.

Output: Point on n^{th}-degree curve



de Boor Algorithm (1/4)

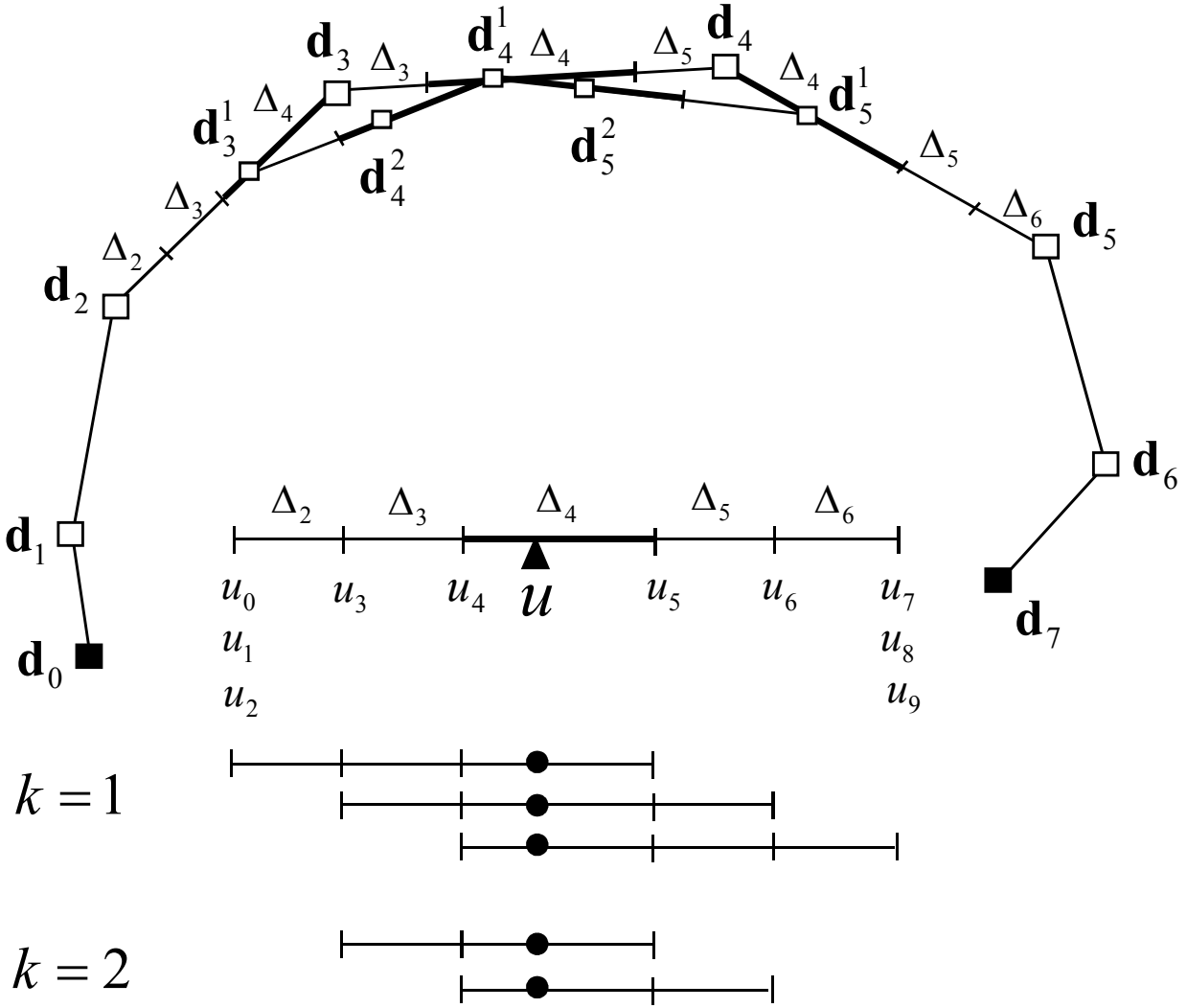
$$\mathbf{d}_i^k(u) = \frac{u_{i+n-k} - u}{u_{i+n-k} - u_{i-1}} \mathbf{d}_{i-1}^{k-1}(u) + \frac{u - u_{i-1}}{u_{i+n-k} - u_{i-1}} \mathbf{d}_i^{k-1}(u)$$



- The ratio of the linear interpolation used in the de Casteljau algorithm is constant.
- In contrast, the ratio of the linear interpolation used in the de Boor algorithm is not constant, because the intervals of the parameters of the Bezier curve segments, of which B-spline curve is composed, are different from each other.

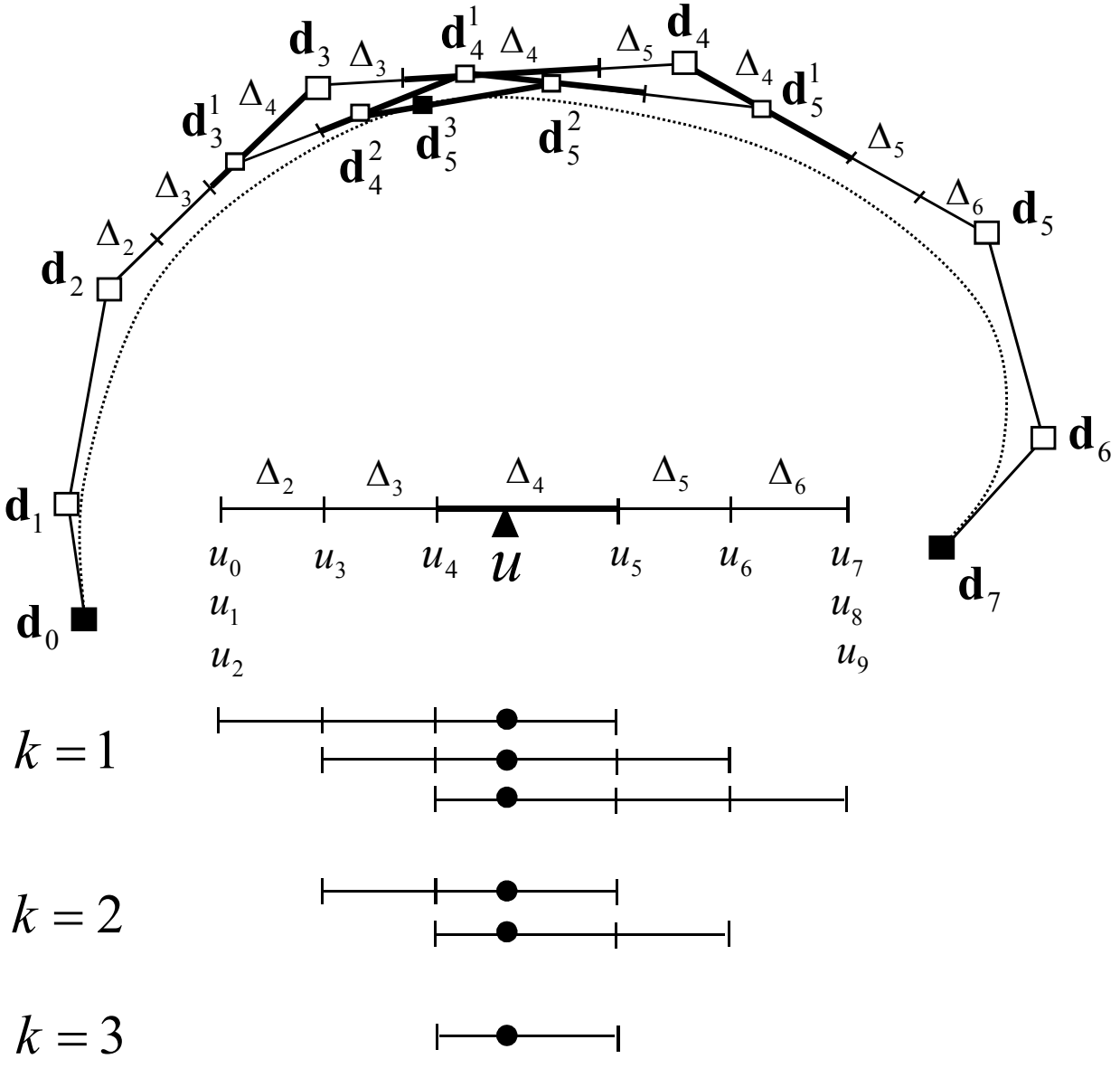
de Boor Algorithm (2/4)

$$\mathbf{d}_i^k(u) = \frac{u_{i+n-k} - u}{u_{i+n-k} - u_{i-1}} \mathbf{d}_{i-1}^{k-1}(u) + \frac{u - u_{i-1}}{u_{i+n-k} - u_{i-1}} \mathbf{d}_i^{k-1}(u)$$



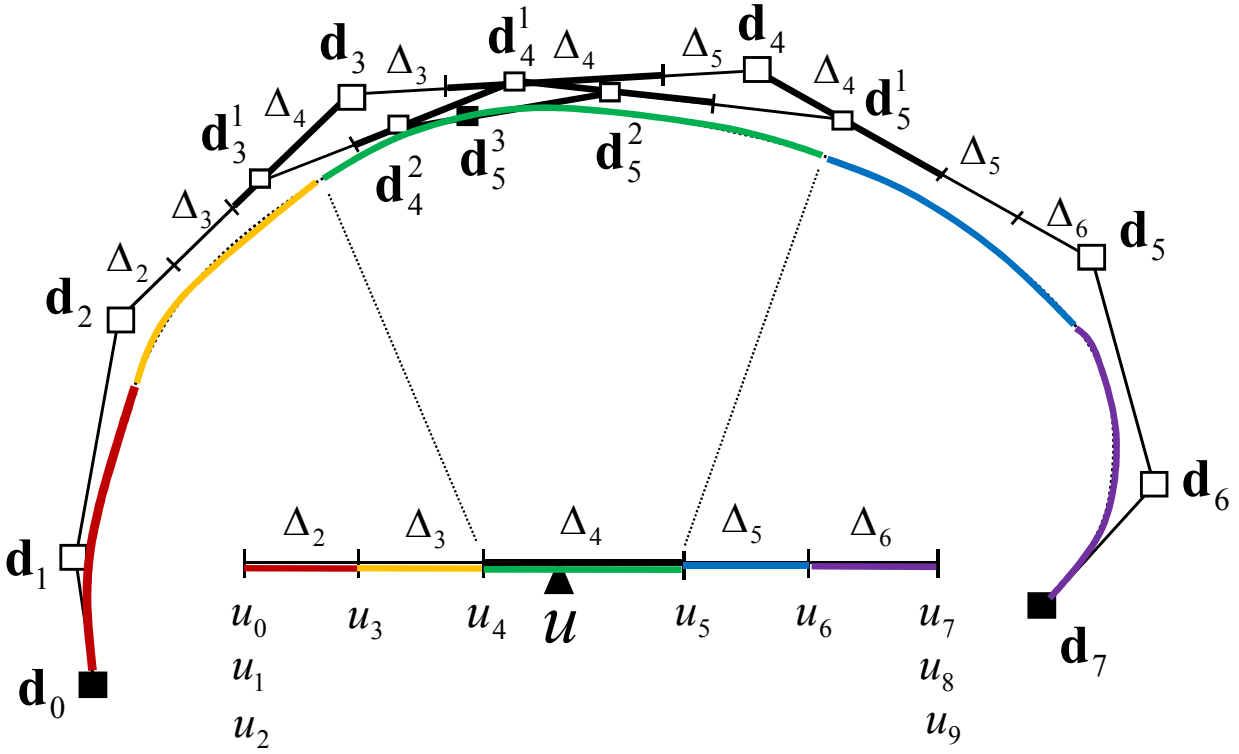
de Boor Algorithm (3/4)

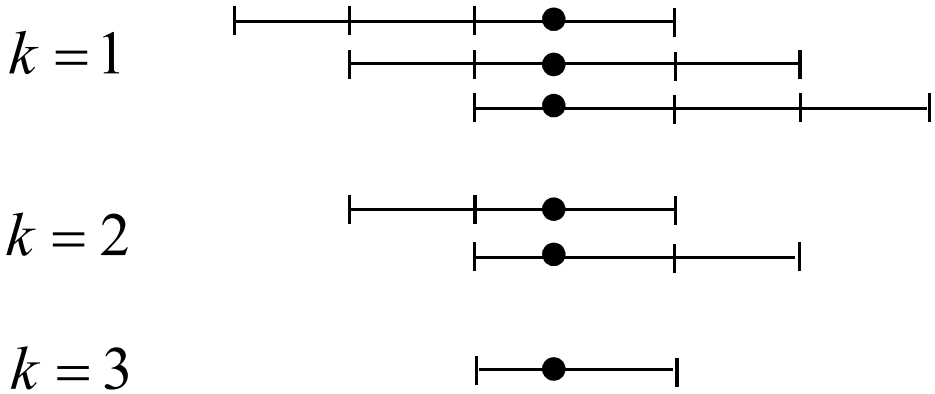
$$\mathbf{d}_i^k(u) = \frac{u_{i+n-k} - u}{u_{i+n-k} - u_{i-1}} \mathbf{d}_{i-1}^{k-1}(u) + \frac{u - u_{i-1}}{u_{i+n-k} - u_{i-1}} \mathbf{d}_i^{k-1}(u)$$



de Boor Algorithm (4/4)

$$\mathbf{d}_i^k(u) = \frac{u_{i+n-k} - u}{u_{i+n-k} - u_{i-1}} \mathbf{d}_{i-1}^{k-1}(u) + \frac{u - u_{i-1}}{u_{i+n-k} - u_{i-1}} \mathbf{d}_i^{k-1}(u)$$

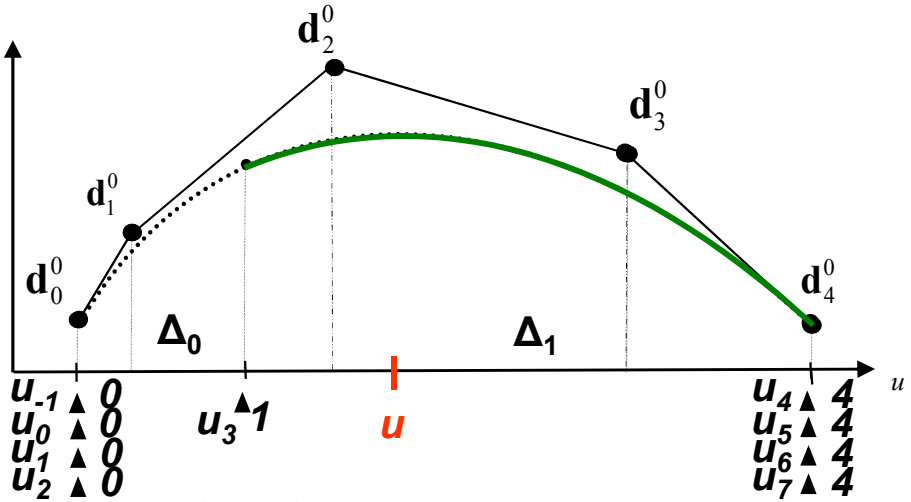


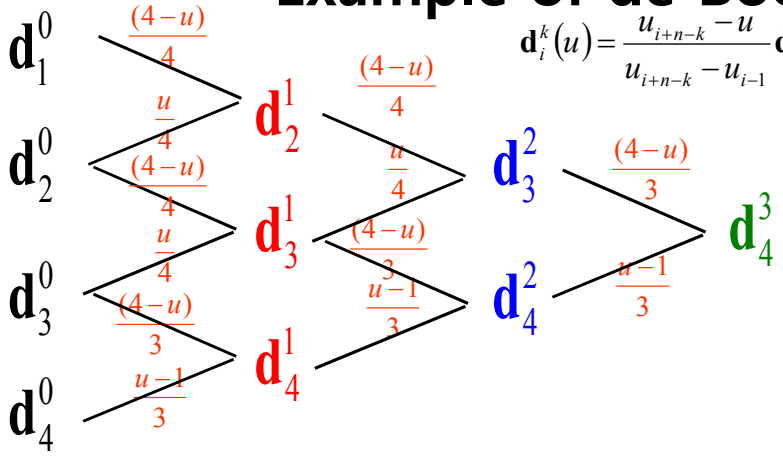


$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \cdots + \mathbf{d}_n N_n^3(u)$$

Example of de Boor Algorithm

$$\mathbf{d}_i^k(u)=\frac{u_{i+n-k}-u}{u_{i+n-k}-u_{i-1}}\mathbf{d}_{i-1}^{k-1}(u)+\frac{u-u_{i-1}}{u_{i+n-k}-u_{i-1}}\mathbf{d}_i^{k-1}(u)$$





$$\mathbf{d}_2^1=\frac{(4-u)}{4}\mathbf{d}_1^0+\frac{u}{4}\mathbf{d}_2^0$$

$$\mathbf{d}_3^1=\frac{(4-u)}{4}\mathbf{d}_2^0+\frac{u}{4}\mathbf{d}_3^0$$

$$\mathbf{d}_4^1=\frac{(4-u)}{3}\mathbf{d}_3^0+\frac{(u-1)}{3}\mathbf{d}_4^0$$

$$\mathbf{d}_3^2=\frac{(4-u)}{4}\mathbf{d}_2^1+\frac{u}{4}\mathbf{d}_3^1$$

$$\mathbf{d}_4^2=\frac{(4-u)}{3}\mathbf{d}_3^1+\frac{(u-1)}{4}\mathbf{d}_4^1$$

$$\mathbf{d}_4^3=\frac{(4-u)}{3}\mathbf{d}_3^2+\frac{(u-1)}{3}\mathbf{d}_4^2$$

$$\mathbf{d}_4^3=\frac{(4-u)}{3}\left(\frac{(4-u)}{4}\mathbf{d}_2^1+\frac{u}{4}\mathbf{d}_3^1\right)+\frac{(u-1)}{3}\left(\frac{(4-u)}{3}\mathbf{d}_3^1+\frac{(u-1)}{3}\mathbf{d}_4^1\right)$$

$$=\frac{(4-u)^2}{3\cdot4}\mathbf{d}_2^1+\frac{(4-u)u}{3\cdot4}\mathbf{d}_3^1+\frac{(u-1)(4-u)}{3\cdot3}\mathbf{d}_3^1+\frac{(u-1)^2}{3\cdot3}\mathbf{d}_4^1$$

$$=\frac{(4-u)^2}{3\cdot4}\mathbf{d}_2^1+\left(\frac{(4-u)u}{3\cdot4}+\frac{(u-1)(4-u)}{3^2}\right)\mathbf{d}_3^1+\frac{(u-1)^2}{3^2}\mathbf{d}_4^1$$

$$\mathbf{d}_4^3=\frac{(4-u)^2}{3\cdot4}\left(\frac{(4-u)}{4}\mathbf{d}_1^0+\frac{u}{4}\mathbf{d}_2^0\right)+\left(\frac{(4-u)u}{12}+\frac{(u-1)(4-u)}{9}\right)\left(\frac{(4-u)}{4}\mathbf{d}_2^0+\frac{u}{4}\mathbf{d}_3^0\right)+\frac{(u-1)^2}{9}\left(\frac{(4-u)}{3}\mathbf{d}_3^0+\frac{(u-1)}{3}\mathbf{d}_4^0\right)$$

$$\mathbf{d}_4^3=\frac{(4-u)^3}{48}\mathbf{d}_1^0+(4-u)^2\left(\frac{u}{24}+\frac{(u-1)}{36}\right)\mathbf{d}_2^0+(4-u)\left(\frac{u^2}{48}+\frac{u(u-1)}{36}+\frac{(u-1)^2}{27}\right)\mathbf{d}_3^0+\frac{(u-1)^3}{27}\mathbf{d}_4^0$$

de Boor Algorithm as a "Constructive Approach"

☑ de Boor Algorithm: "Constructive Approach"

Input: $\mathbf{d}_i$ (de Boor Points)

Processor: n-times sequential "linear interpolation" at u
by marching (advancing) the segment of $\mathbf{d}_i\ \mathbf{d}_{i+1}$

Output: Point on n^{th}-degree curve

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \cdots + \mathbf{d}_n N_n^3(u)$$

Relationship between the de Boor Algorithm and B-Spline Curves

☑ de Boor Algorithm: "Constructive Approach"

Input: $\mathbf{d}_i$ (de Boor Points)

Processor: n-times sequential "linear interpolation"

Output: Point on n^{th}-degree curve

☑ B-spline curve: "B-spline function evaluation Approach"

Input: $\mathbf{d}_i$ (de Boor Points)

Processor: "Blending" the de Boor points($\mathbf{d}_i$) in space

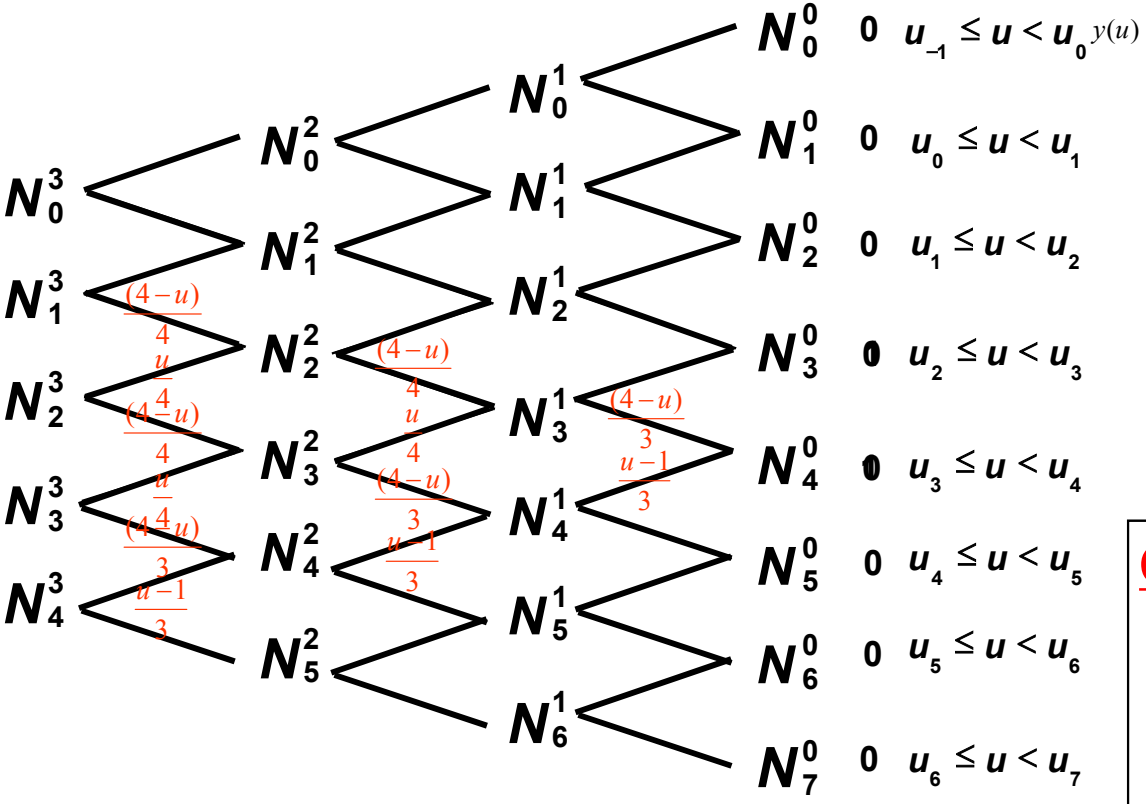
and B-spline basis function(Cox-de Boor recurrence formula)

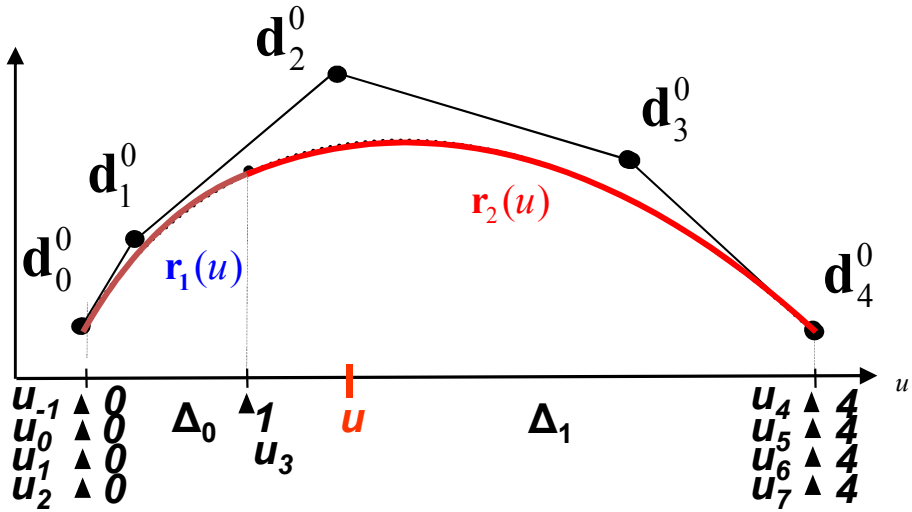
Output: Closed form of n^{th}-degree curve equation

3.6 Cox-de Boor Algorithm and de Boor Algorithm

Cox–de Boor Algorithm

- Evaluation of Cox–de Boor Basis Function





$$\mathbf{r}(u) = V_0^0 N_0^3(u) + V_1^0 N_1^3(u) + V_2^0 N_2^3(u) + V_3^0 N_3^3(u) + V_4^0 N_4^3(u)$$

Cox-de Boor Algorithm

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

$$N_1^3 = \frac{u-u_0}{u_3-u_2} N_1^2 + \frac{u_4-u}{u_4-u_1} N_2^2 = \frac{u-u_0}{u_3-u_2}\cdot\frac{u_3-u}{u_3-u_1}\cdot\frac{u_2-u}{u_3-u_2} + \frac{u_4-u}{u_4-u_1}\cdot\frac{u-u_1}{u_3-u_1}\cdot\frac{u_2-u}{u_3-u_2} + \frac{u_4-u}{u_4-u_1}\cdot\frac{u_4-u}{u_4-u_2}\cdot\frac{u-u_2}{u_3-u_2}$$

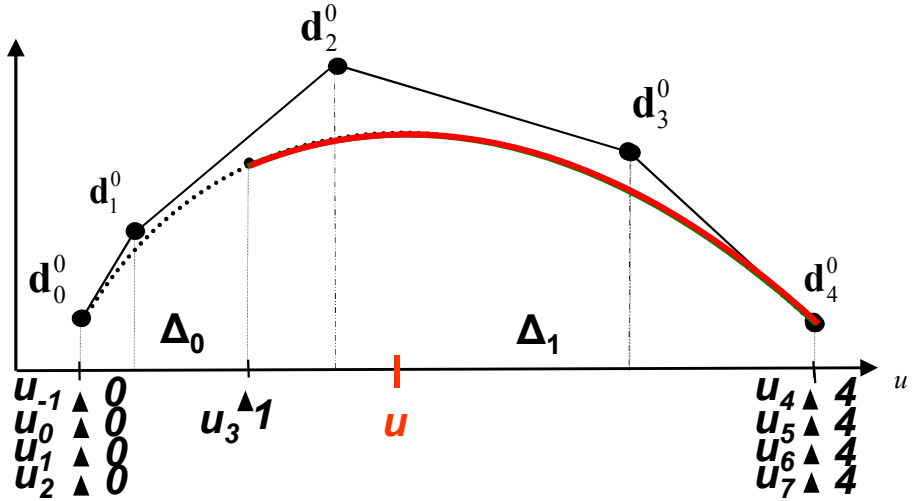
$$\mathbf{r}_2(u) = \frac{(4-u)^3}{48}\mathbf{d}_1^0 + (4-u)^2\left(\frac{u}{24} + \frac{u-1}{36}\right)\mathbf{d}_2^0 + (4-u)\left(\frac{u^2}{48} + \frac{u(u-1)}{36} + \frac{(u-1)^2}{27}\right)\mathbf{d}_3^0 + \frac{(u-1)^3}{27}\mathbf{d}_4^0$$

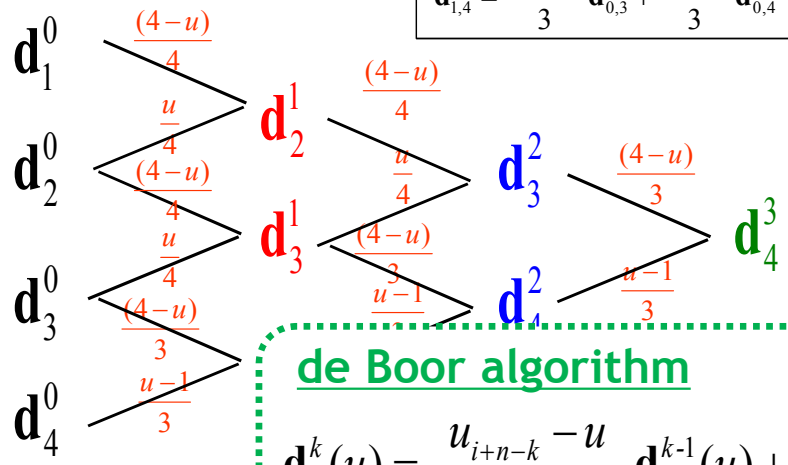
de Boor Algorithm

$$\mathbf{d}_{1,2}^{y}=\frac{(4-u)}{4}\mathbf{d}_{0,1}^{y}+\frac{u}{4}\mathbf{d}_{0,2}^{y} \qquad \mathbf{d}_{2,3}^{y}=\frac{(4-u)}{4}\mathbf{d}_{1,2}^{y}+\frac{u}{4}\mathbf{d}_{1,3}^{y}$$

$$\mathbf{d}_{1,3}^{y}=\frac{(4-u)}{4}\mathbf{d}_{0,2}^{y}+\frac{u}{4}\mathbf{d}_{0,3}^{y} \qquad \mathbf{d}_{2,4}^{y}=\frac{(4-u)}{3}\mathbf{d}_{1,3}^{y}+\frac{(u-1)}{3}\mathbf{d}_{1,4}^{y}$$

$$\mathbf{d}_{1,4}^{y}=\frac{(4-u)}{3}\mathbf{d}_{0,3}^{y}+\frac{(u-1)}{3}\mathbf{d}_{0,4}^{y} \qquad \mathbf{d}_{3,4}^{y}=\frac{(4-u)}{3}\mathbf{d}_{2,3}^{y}+\frac{(u-1)}{3}\mathbf{d}_{2,4}^{y}$$





$$\mathbf{d}_4^3=\frac{(4-u)^3}{48}\mathbf{d}_1^0+(4-u)^2\left(\frac{u}{24}+\frac{(u-1)}{36}\right)\mathbf{d}_2^0+$$
$$(4-u)\left(\frac{u^2}{48}+\frac{u(u-1)}{36}+\frac{(u-1)^2}{27}\right)\mathbf{d}_3^0+\frac{(u-1)^3}{27}\mathbf{d}_4^0$$

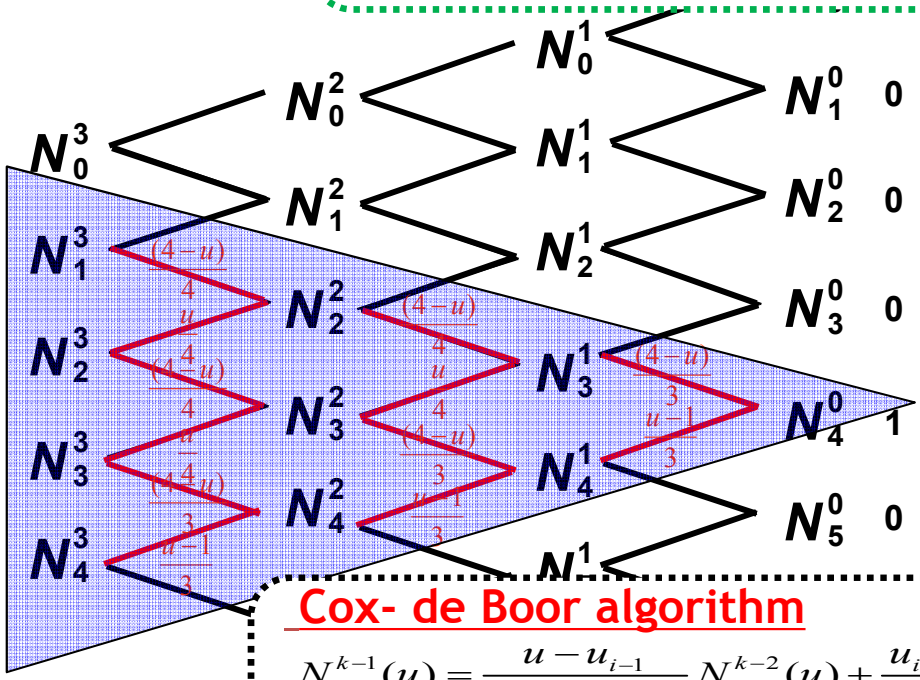
de Boor algorithm

$$\mathbf{d}_i^k(u)=\frac{u_{i+n-k}-u}{u_{i+n-k}-u_{i-1}}\mathbf{d}_{i-1}^{k-1}(u)+\frac{u-u_{i-1}}{u_{i+n-k}-u_{i-1}}\mathbf{d}_i^{k-1}(u)$$

The same cubic equation at the parameter u can be obtained.

➡ de Boor Algorithm is the same as Cox de-Boor Algorithm.

$$\mathbf{r}_2(u)=\frac{(4-u)^3}{48}\mathbf{d}_1^0+(4-u)^2\left(\frac{u}{24}+\frac{u-1}{36}\right)\mathbf{d}_2^0+$$
$$(4-u)\left(\frac{u^2}{48}+\frac{u(u-1)}{36}+\frac{(u-1)^2}{27}\right)\mathbf{d}_3^0+\frac{(u-1)^3}{27}\mathbf{d}_4^0$$



Cox- de Boor algorithm

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

$$N_i^0(u)=\begin{cases}1 & \text{if } u_{i-1}\le u<u_i\\ 0 & \text{else}\end{cases}$$

Bezier Curve and B-Spline Curve

Item		Bezier Curve	B-Spline Curve
Equation of Curves	Given	Bezier Control Point $\mathbf{b}_i$ Parameter t Bernstein Polynomial Func. $B_i^n(t)$	B-Spline Control Point $\mathbf{d}_i$ Parameter u B-Spline Basis Func. $N_i^n(u)$
	Find	Bezier Curve $\mathbf{r}(t)$ $\mathbf{r}(t)=\mathbf{b}_0B_0^n(t)+\mathbf{b}_1B_1^n(t)+\ldots\ldots+\mathbf{b}_nB_n^n(t).$	B-Spline Curve $\mathbf{r}(u)$ $\mathbf{r}(u)=\mathbf{d}_0N_0^3(u)+\mathbf{d}_1N_1^3(u)+\mathbf{d}_2N_2^3(u)+\cdots+\mathbf{d}_{D-1}N_{D-1}^3(u)$
Function Evaluation Approach		Bernstein Polynomial Function $B_i^n(t)=\binom{n}{i}t^i(1-t)^{n-i},$ $\binom{n}{i}={}_nC_i=\begin{cases}\dfrac{n!}{i!(n-i)!} & \textbf{if } 0\le i\le n\\ \mathbf{0} & \textbf{else}\end{cases}$	B-Spline Basis Function (Cox-de boor Recursive Formula) $N_i^n(u)=\dfrac{u-u_{i-1}}{u_{i+n-1}-u_{i-1}}N_i^{n-1}(u)+\dfrac{u_{i+n}-u}{u_{i+n}-u_i}N_{i+1}^{n-1}(u)$ $N_i^0(u)=\begin{cases}1 & \textbf{if } u_{i-1}\le u<u_i\\ 0 & \textbf{else}\end{cases},\ \sum_{i=0}^{D-1}N_i^n(u)=1$
Constructive Approach		de Casteljau Algorithm $\mathbf{b}_i^k(t)=(1-t)\mathbf{b}_i^{k-1}+t\mathbf{b}_{i+1}^{k-1}$	de Boor Algorithm $\mathbf{d}_i^k(u)=\dfrac{u_{i+n-k}-u}{u_{i+n-k}-u_{i-1}}\mathbf{d}_{i-1}^{k-1}(u)+\dfrac{u-u_{i-1}}{u_{i+n-k}-u_{i-1}}\mathbf{d}_i^{k-1}(u)$
Interpolation	Given	Points on Curve: $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n$	Points on Curve: $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n$
	Find	Bezier Control Point $\mathbf{b}_i$	B-Spline Control Point $\mathbf{d}_i$

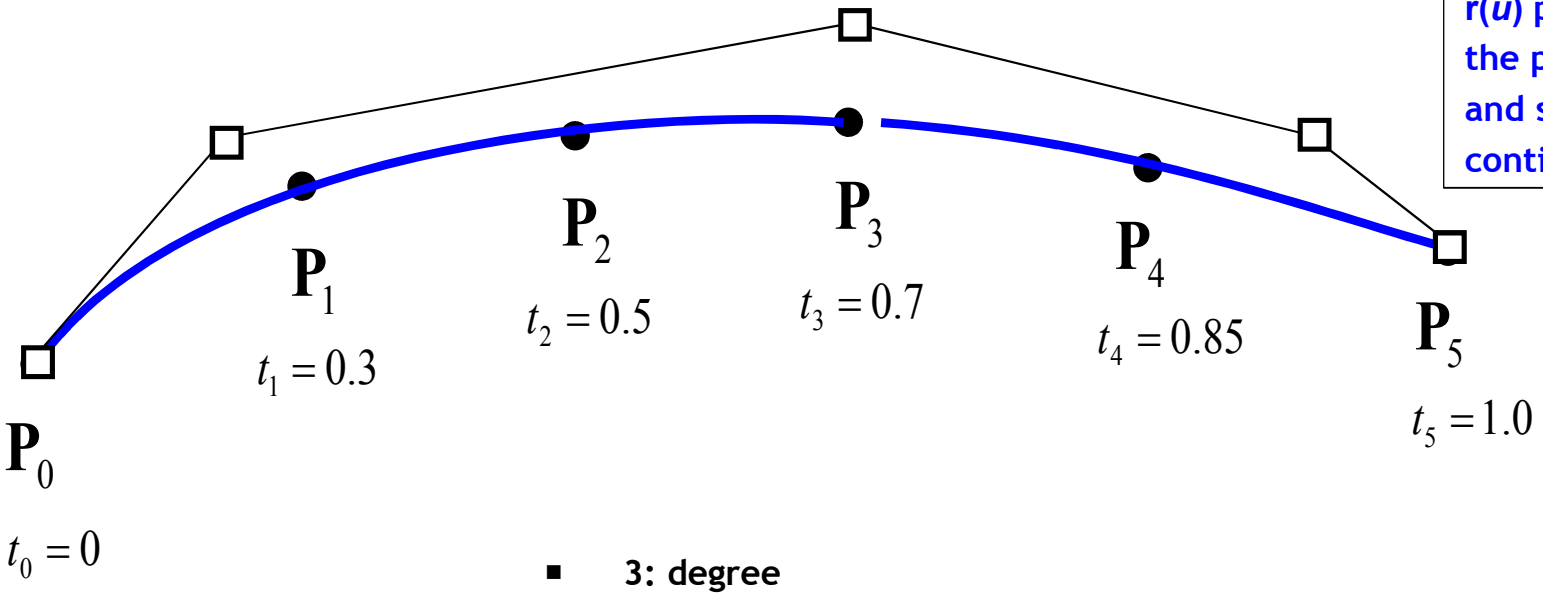
Reference Slides

[Appendix] Determination of the Number of Bezier Curve Segments and Knot Values (1/2)

☑ Given: Fitting points $\mathbf{P}_i$ and corresponding parameter t_i

where, $i = 0,1,\ldots,m$ and $t_0 = 0, \quad t_m = 1$

☑ First, determine number of Bezier curve segment and its knots



- 3: degree
- 2: number of Bezier curve segments
- number of control points
  = 4 + (2-1) = 5

Given:

- Points p_i on the curve
- Knots u_j of the given points on the curve
- Tangent vectors t_0, t_1 at both ends

Find:

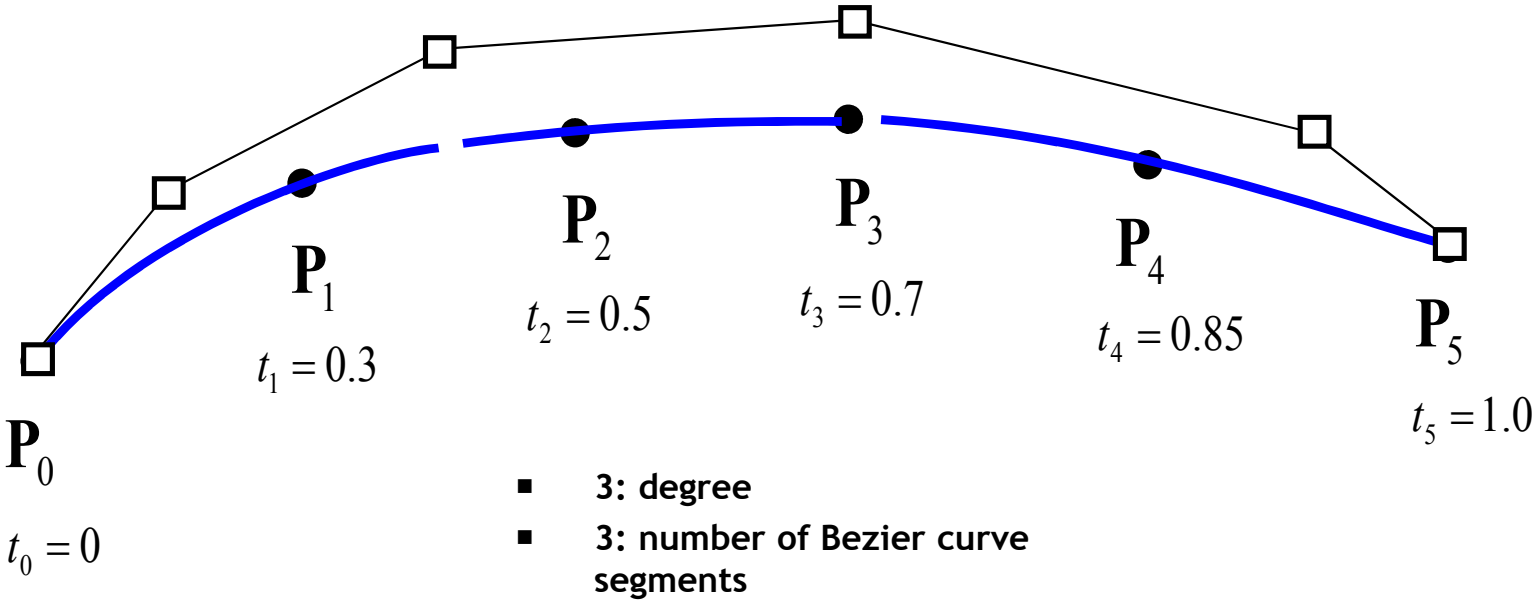
1. B-spline control point d_i
2. Cubic B-spline curve $r(u)$ passing through the points p_i on the curve and satisfying C^2 continuity condition.

[Appendix] Determination of the Number of Bezier Curve Segments and Knot Values (2/2)

☑ **Given: Fitting points $\mathbf{P}_i$ and corresponding parameter t_i**

where, $i = 0,1,\ldots,m$ **and** $t_0 = 0, \quad t_m = 1$

☑ **First, determine number of Bezier curve segment and its knots.**



- 3: degree
- 3: number of Bezier curve segments
- number of control points
  = 4 + (3-1) = 6
- How do we determine Knots? (= start / end points of each cubic Bezier curve)

[Appendix] LU Decomposition Method

[Appendix] Procedure to Determine X

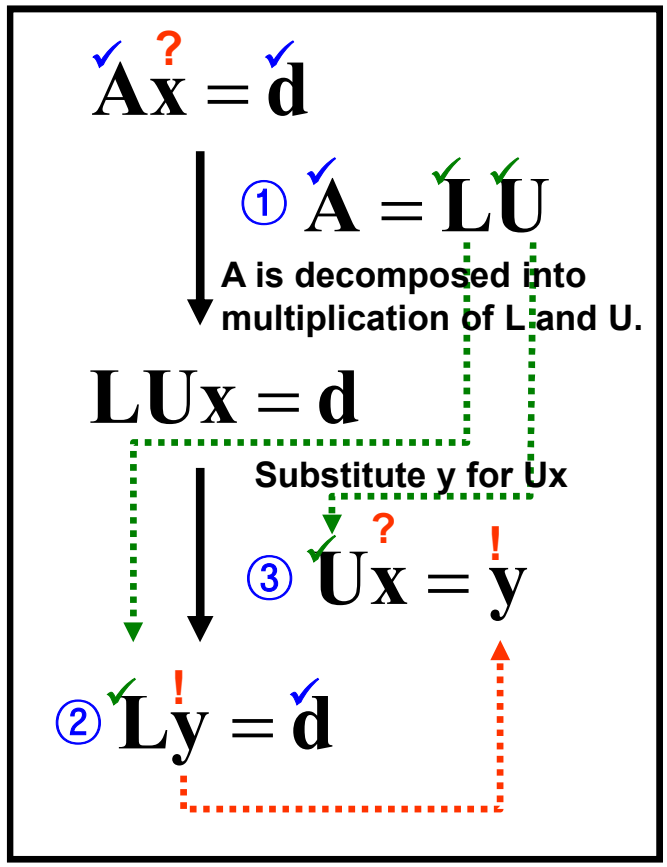
Tri-diagonal matrix

A tri-diagonal matrix has nonzero elements only in the main diagonal, the first diagonal below the main diagonal, and the first diagonal above the main diagonal. ➡ "Tri" + "Diagonal"

$$\begin{bmatrix} b_0 & c_0 & 0 & & & \\ a_1 & b_1 & c_1 & 0 & & \\ 0 & a_2 & b_2 & c_2 & 0 & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & 0 & a_n & b_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

Diagonal elements

$\mathbf{A}$ $\mathbf{x}$ $=$ $\mathbf{d}$



Procedure to determine $\mathbf{x}$

① A is decomposed into multiplication of L and U.

② Solve the equation "Ly = d" for y.

③ Solve the equation "Ux = y" for x.

[Appendix] Decomposition of A into Multiplication of L and U

① $\mathbf{A} = \mathbf{LU}$

$$\begin{bmatrix} b_0 & c_0 & 0 & & & \\ a_1 & b_1 & c_1 & 0 & & \\ 0 & a_2 & b_2 & c_2 & 0 & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & 0 & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & 0 & a_n & b_n \end{bmatrix} = \begin{bmatrix} \beta_0 & 0 & & & & \\ \alpha_1 & \beta_1 & 0 & & & \\ 0 & \alpha_2 & \beta_2 & 0 & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & 0 & \alpha_{n-1} & \beta_{n-1} & 0 \\ & & & 0 & \alpha_n & \beta_n \end{bmatrix} \begin{bmatrix} 1 & \gamma_1 & 0 & & & \\ 0 & 1 & \gamma_2 & 0 & & \\ & 0 & 1 & \gamma_3 & 0 & \\ & & \ddots & \ddots & \ddots & \\ & & & 0 & 1 & \gamma_n \\ & & & & 0 & 1 \end{bmatrix}$$

Diagonal elements

$\mathbf{A} = \mathbf{L} \quad \mathbf{U}$

$$b_0 = \beta_0 \qquad c_0 = \beta_0\gamma_1$$

$$a_1 = \alpha_1 \qquad b_1 = \alpha_1\gamma_1 + \beta_1 \qquad c_1 = \beta_1\gamma_2$$

$$a_2 = \alpha_2 \qquad b_2 = \alpha_2\gamma_2 + \beta_2 \qquad c_2 = \beta_2\gamma_3$$

$$\vdots \qquad \vdots \qquad \vdots$$

$$a_{n-1} = \alpha_{n-1} \qquad b_{n-1} = \alpha_{n-1}\gamma_{n-1} + \beta_{n-1} \qquad c_{n-1} = \beta_{n-1}\gamma_n$$

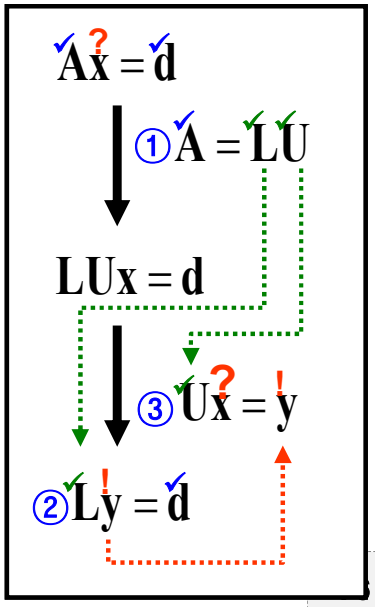
$$a_n = \alpha_n \qquad b_n = \alpha_n\gamma_n + \beta_n$$

$$\alpha_i = a_i \qquad i = 1, \cdots, n$$

$$\gamma_{i+1} = \frac{c_i}{\beta_i} \qquad i = 0, \cdots, n-1$$

$$\beta_{i+1} = b_{i+1} - \alpha_{i+1}\gamma_{i+1} \qquad i = 0, \cdots, n-1$$

with $\beta_0 = b_0$



[Appendix] Decomposition of A into Multiplication of L and U
- Forward Substitution

$$\textcircled{2}\ \mathbf{L}\mathbf{y} = \mathbf{d}$$

$$\begin{bmatrix} \beta_0 & 0 & & & & \\ \alpha_1 & \beta_1 & 0 & & & \\ 0 & \alpha_2 & \beta_2 & 0 & & \\ & \ddots & \ddots & \ddots & & \\ & & 0 & \alpha_{n-1} & \beta_{n-1} & 0 \\ & & & 0 & \alpha_n & \beta_n \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

Diagonal elements

$$\mathbf{L} \qquad \mathbf{y} = \mathbf{d}$$

$$\beta_0 y_0 = d_0$$

$$\alpha_1 y_0 + \beta_1 y_1 = d_1$$

$$\alpha_2 y_1 + \beta_2 y_2 = d_2$$

$$\vdots$$

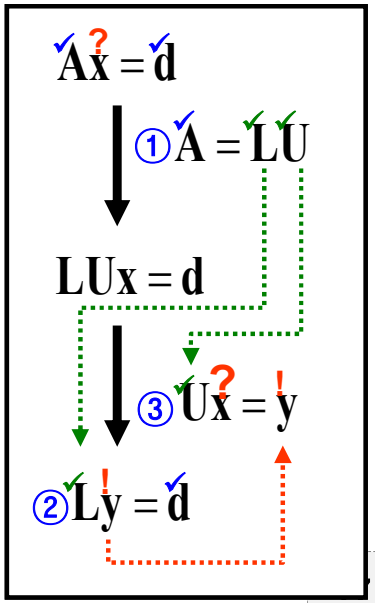
$$\alpha_{n-1} y_{n-2} + \beta_{n-1} y_{n-1} = d_{n-1}$$

$$\alpha_n y_{n-1} + \beta_n y_n = d_n$$

Forward substitution

$$y_i = \frac{d_i - \alpha_i y_{i-1}}{\beta_i} \qquad i = 1, \cdots, n$$

$$\text{with} \quad y_0 = \frac{d_0}{\beta_0}$$



[Appendix] Decomposition of A into Multiplication of L and U
- Backward Substitution

③ $\mathbf{Ux} = \mathbf{y}$

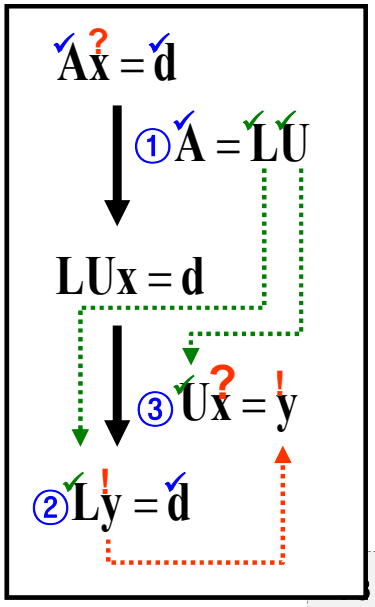
$$\begin{bmatrix} 1 & \gamma_1 & 0 & & & \\ 0 & 1 & \gamma_2 & 0 & & \\ & 0 & 1 & \gamma_3 & 0 & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & 0 & 1 & \gamma_n \\ & & & & & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}$$

$\mathbf{U}$ $\mathbf{x}$ $=$ $\mathbf{y}$

$$x_0 + \gamma_1 x_1 = y_0$$
$$x_1 + \gamma_2 x_2 = y_1$$
$$x_2 + \gamma_3 x_3 = y_2$$
$$\vdots$$
$$x_{n-1} + \gamma_n x_n = y_{n-1}$$
$$x_n = y_n$$

Backward substitution

$$x_i = y_i - \gamma_{i+1} x_{i+1}$$
$$i = n-1, \cdots, 0$$
with $x_n = y_n$



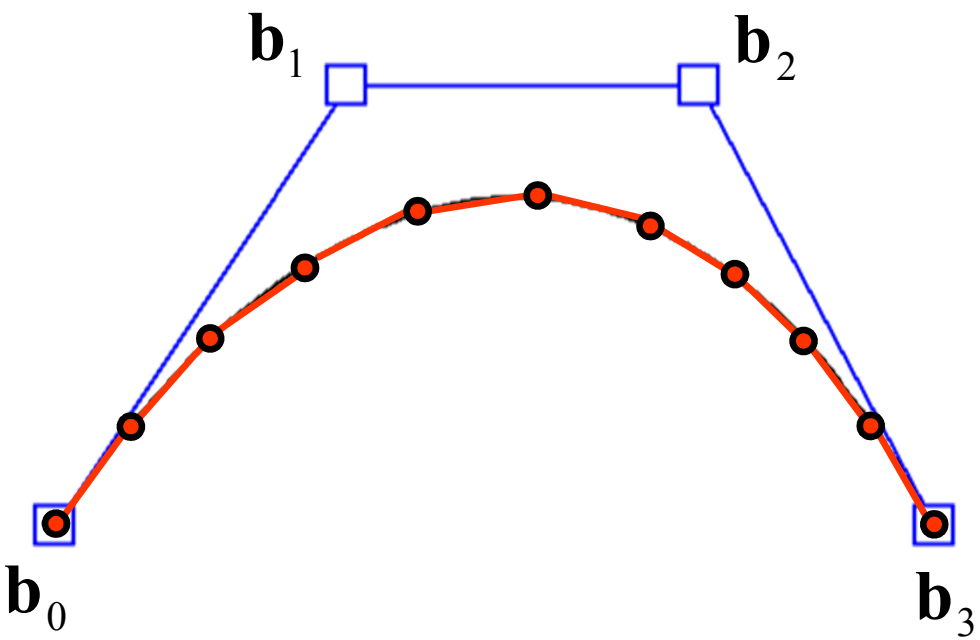
[Appendix] Programming for B-Spline Curve

(1) Sample Code for B-Spline Curve Class

(2) Sample Code of Cubic B-Spline Curve Interpolation

Programming for B-Spline Curve Class

Ex: Cubic B-spline curve



$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

1) Definition of B-spline Curve

- Degree
- Control Point

Member Variables of B-spline Curve Class
int n: degree of B-spline Curve
Vector* m_ControlPoint: Control Point
int m_nControlPoint: the number of Control Point

2) Calculation of B-spline Basis Function

(Cox-de Boor Recurrence Formula)

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

3) B-spline curve construction

- Divide the parameter u(u_{min}~u_{max}) into n equal parts
- Find the points on the curve at the each divided parameter
- Represent curve by connecting points with straight lines

Sample Code of Cubic B-Spline Curve (1/4)

Member Variables of B-spline Curve Class
int m_nDegree : degree of B-spline Curve
Vector* m_ControlPoint: Control Point
int m_nControlPoint: the number of Control Point

```
#include "vector.h“
class CubicBsplineCurve {
public:
   Vector* m_ControlPoint;   int m_nControlPoint;
   double* m_Knot; int m_nKnot;
   int m_nDegree;
   CubicBsplineCurve();
   ~CubicBsplineCurve();
   void SetControlPoint(Vector* pControlPoint, int nControlPoint);
   void SetKnot(double* pKnot, int nKnot);
   Vector CalcPoint(double u);
   double N(int d, int i, double u);                    // B-spline basis function
};
```

Sample Code of Cubic B-Spline Curve (2/4)

```
CubicBsplineCurve::CubicBsplineCurve () {
  m_ControlPoint = 0;       m_Knot = 0;
  m_nControlPoint = 0;      m_nKnot = 0;      int m_nDegree =3;
}
CubicBsplineCurve::~CubicBsplineCurve () {
  if(m_ControlPoint)  delete[] m_ControlPoint;
  if(m_Knot)  delete[] m_Knot;
}
void CubicBsplineCurve::SetControlPoint(Vector* pControlPoint, int nControlPoint) {
  m_ControlPoint = new Vector[nControlPoint];
  for(int i=0; i < nControlPoint; i++) {
    m_ControlPoint[i] = pControlPoint[i];
  }
}
void CubicBsplineCurve::SetKnot(double* pKnot, int nKnot){
  m_Knot = new double[nKnot];
  for(int i=0; i < nKnot; i++) {
    m_Knot[i] = pKnot[i];
  }
}
```

Sample Code of Cubic B-Spline Curve (3/4)

```
Vector CubicBsplineCurve::CalcPoint(double u)
{
    Vector PointOnCurve(0,0,0);
    if ( t < m_Knot[0] || t > m_Knot[m_nKnot-1] ) {
        return PointOnCurve;
    }
    for(int i = 0; i < m_nControlPoint; i++){
        PointOnCurve = PointOnCurve + m_ControlPoint[i] * N(m_nDegree, i, u);
    }
    return PointOnCurve;
}
```

Calculate points on the curve at parameter *u*

$$\mathbf{r}(u) = \mathbf{d}_0 N_0^3(u) + \mathbf{d}_1 N_1^3(u) + \mathbf{d}_2 N_2^3(u) + \mathbf{d}_3 N_3^3(u)$$

Sample Code of Cubic B-Spline Curve (4/4)

```
double CubicBsplineCurve:: N(int d, int i, double u) {
    // Find Span k
    // U i-1 <= U < U i  → k = i

    if( d == 0 ) {
        // return 0 or 1;
    } else {
        // return Cox de-Boor recurrence formula
    }
}
```

Calculation of B-spline Basis Function

(Cox-de Boor Recurrence Formula)

$$N_i^n(u) = \frac{u - u_{i-1}}{u_{i+n-1} - u_{i-1}} N_i^{n-1}(u) + \frac{u_{i+n} - u}{u_{i+n} - u_i} N_{i+1}^{n-1}(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \le u < u_i \\ 0 & \text{else} \end{cases}$$

Sample Code of Cubic B-Spline Curve Interpolation (1/4)

```
#ifndef __CubicBspline_h__
#define __CubicBspline_h__

#include "vector.h“

class CubicBsplineCurve {
public:
   Vector* m_ControlPoint;   int m_nControlPoint;
   double* m_Knot; int m_nKnot;  int m_nDegree;

   ……..
   void SetControlPoint(Vector* pControlPoint, int nControlPoint);
   void SetKnot(double* pKnot, int nKnot);
   Vector CalcPoint(double u);
   double N(int d, int i, double u);
   void Interpolate(Vector *pFittingPoint, int nFittingPoint);
   void Parameterization(int nType, Vector* FittingPoint, int nPoint, double* t);
};
#endif
```

Sample Code of Cubic B-Spline Curve Interpolation (2/4)

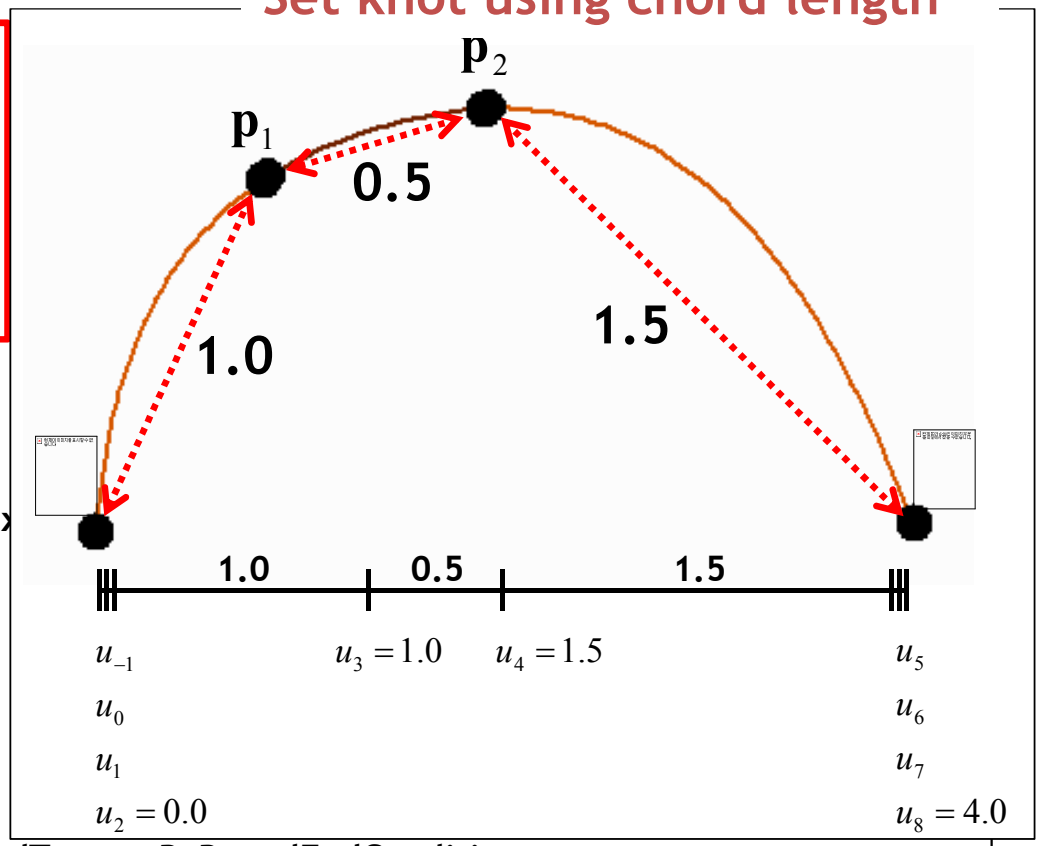
```
void CubicBsplineCurve::Interpolate(Vector *pFittingPoint, int
{
  // Generate Knot
    if(m_Knot) delete[] m_Knot;
    m_nKnot = (m_nFittingPoint - 2) + 2*(3+1);
    m_Knot = new double [m_nKnot];
    // Use Chord length or Centripetal method
    ......

//---------------------------------------
// Generate Matrix : (L+1) * (L+1)
   int L = m_nFittingPoint + 1;          // (L+1)*(L+1) size Matrix

  // Fill rhs
    Vector* rhs = new Vector[L+1];
    for(i = 1; i <= L-1 ; i++) rhs[i] = pFittingPoint[i-1];

  // Bessel End condition
    rhs[0] = rhs[1]; rhs[L] = rhs[L-1];
    rhs[1] = StartTangentByBesselEndCondition;   rhs[L-1] = EndTangentByBesselEndCondition;
```

Set knot using chord length



Sample Code of Cubic B-Spline Curve Interpolation (3/4)

```
void CubicBsplineCurve::Interpolate(Vector *pFittingPoint, int nFittingPoint)
{
  // Generate Knot
    if(m_Knot) delete[] m_Knot;
    m_nKnot = (m_nFittingPoint - 2) + 2*(3+1);
    m_Knot = new double [m_nKnot];
    // Use Chord length or Centripetal method
    ......

//---------------------------------------
// Generate Matrix : (L+1) * (L+1)
   int L = m_nFittingPoint + 1;          // (L+1)*(L+1) size Matrix

  // Fill rhs
    Vector* rhs = new Vector[L+1];
    for(i = 1; i <= L-1 ; i++) rhs[i] = pFittingPoint[i-1];

  // Bessel End condition
    rhs[0] = rhs[1]; rhs[L] = rhs[L-1];
    rhs[1] = StartTangentByBesselEndCondition;   rhs[L-1] = EndTangentByBess
```

Bessel End Condition

$$\mathbf{t}_s = -\frac{2\Delta_2 + \Delta_3}{\Delta_2(\Delta_2 + \Delta_3)}\mathbf{p}_0 + \frac{(\Delta_2 + \Delta_3)}{\Delta_2\Delta_3}\mathbf{p}_1 - \frac{\Delta_2}{\Delta_3(\Delta_2 + \Delta_3)}\mathbf{p}_2$$

$$\mathbf{t}_e = \frac{\Delta_{K-4}}{\Delta_{K-5}(\Delta_{K-5} + \Delta_{K-4})}\mathbf{p}_{m-2} - \frac{(\Delta_{K-5} + \Delta_{K-4})}{\Delta_{K-5}\Delta_{K-4}}\mathbf{p}_{m-1} + \frac{(2\Delta_{K-4} + \Delta_{K-5})}{(\Delta_{K-5} + \Delta_{K-4})\Delta_{K-4}}\mathbf{p}_m$$

Sample Code of Cubic B-Spline Curve Interpolation (4/4)

```
    double* alpha = new double[L+1];
    double* beta = new double[L+1];
    double* gamma = new double[L+1];
    double* up = new double[L+1];
    double* low = new double[L+1];
    if(m_ControlPoint) delete[] m_ControlPoint;
    m_nControlPoint = L+1;
    m_ControlPoint = new Vector[m_nControlPoint];
  // Fill alpha, beta, gamma
    ......
  // Solve LU system
    l_u_system(alpha, beta, gamma, L, up, low);
    solve_system(up, low, gamma, L, rhs, m_ControlPoint);

//------------------------------
// Release memory
  delete[] rhs;   delete[] alpha;   delete[] beta;   delete[] gamma;   delete[] up;   delete[] low;
}
```

Calculate inverse matrix by using LU decomposition