[2008][02-1][02-2]

Computer aided ship design

Part 1. Curve & Surface

September 2008

Prof. Kyu-Yeul Lee

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

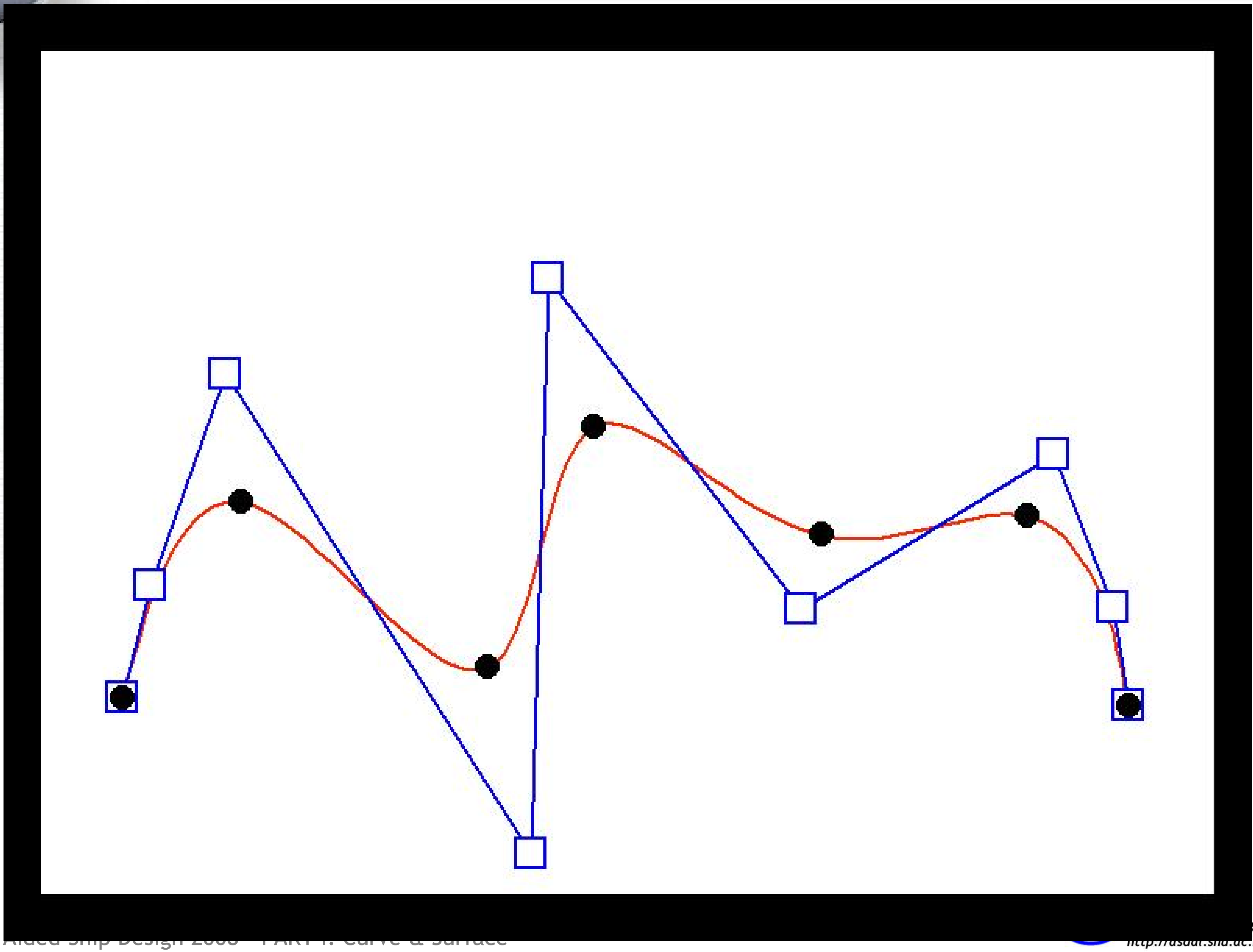
Advanced
Ship
Design
Automation
Laboratory

2.3.5 B-spline curve Interpolation

2.3.5.1 Determine # of curve segments & Knots values
2.3.5.2 Problem definition of B-spline curve interpolation
2.3.5.3 Determine Bezier end control points by end tangent vectors
2.3.5.4 Determine Bezier control points by C^1 continuity condition
2.3.5.5 Determine B-spline control points by C^2 continuity condition
2.3.5.6 Tridiagonal matrix 해법을 이용한 B-spline 곡선 조정점 결정
2.3.5.7 Bessel end condition
2.3.5.8 Sample code of cubic B-spline curve interpolation

Advanced
Ship
Design
Automation
Laboratory

Example of B-spline Interpolation



2.3.5.1 Determine # of Bezier curve segment & Knot value (1)

- Given: fitting points $\mathbf{P}_i$ and corresponding parameter t_i

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

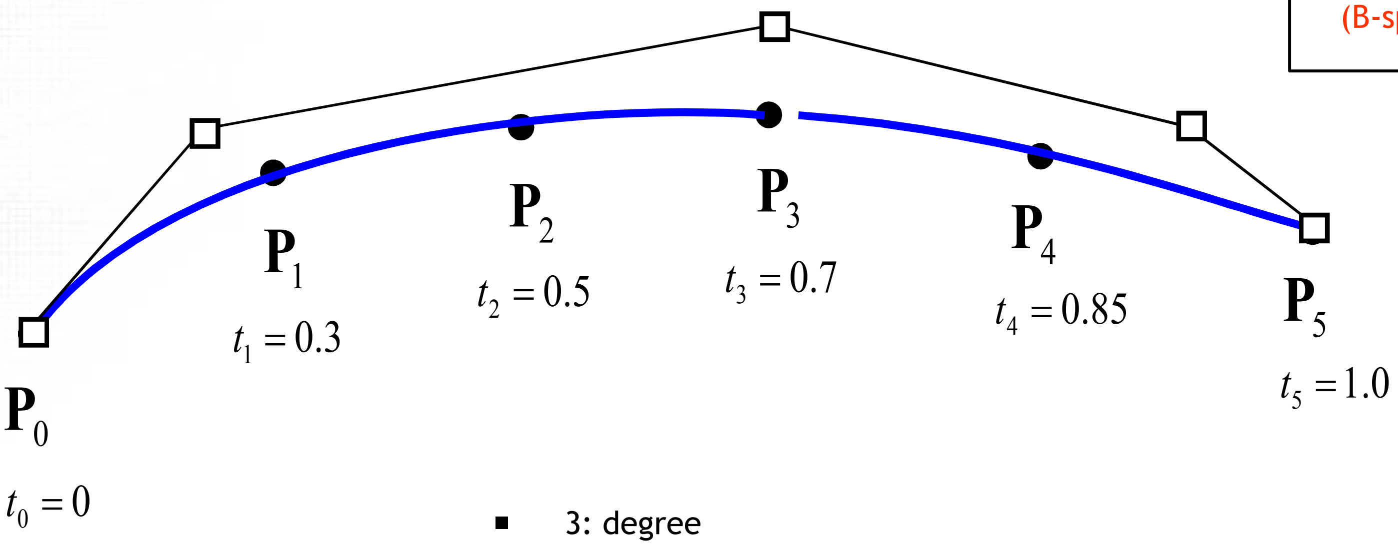
- First, determine # of Bezier curve segment and its knots

Given:

곡선 상의 점 $\mathbf{p}_i, t_i$
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0, \mathbf{t}_1$

Find:

곡선 상의 점 $\mathbf{p}_i$을 지나고
C^2 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)



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

2.3.5.1 Determine # of Bezier curve segment & Knot value (2)

- Given: fitting points $\mathbf{P}_i$ and corresponding parameter t_i

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

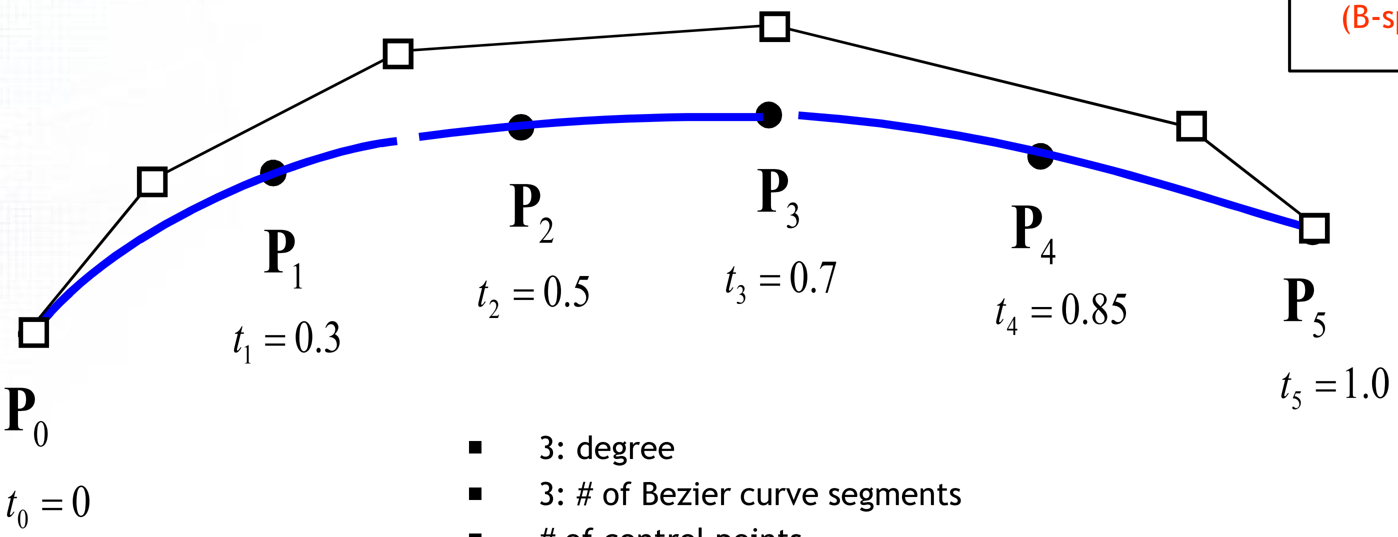
- First, determine # of Bezier curve segment and its knots

Given:

곡선 상의 점 $\mathbf{p}_i, t_i$
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0, \mathbf{t}_1$

Find:

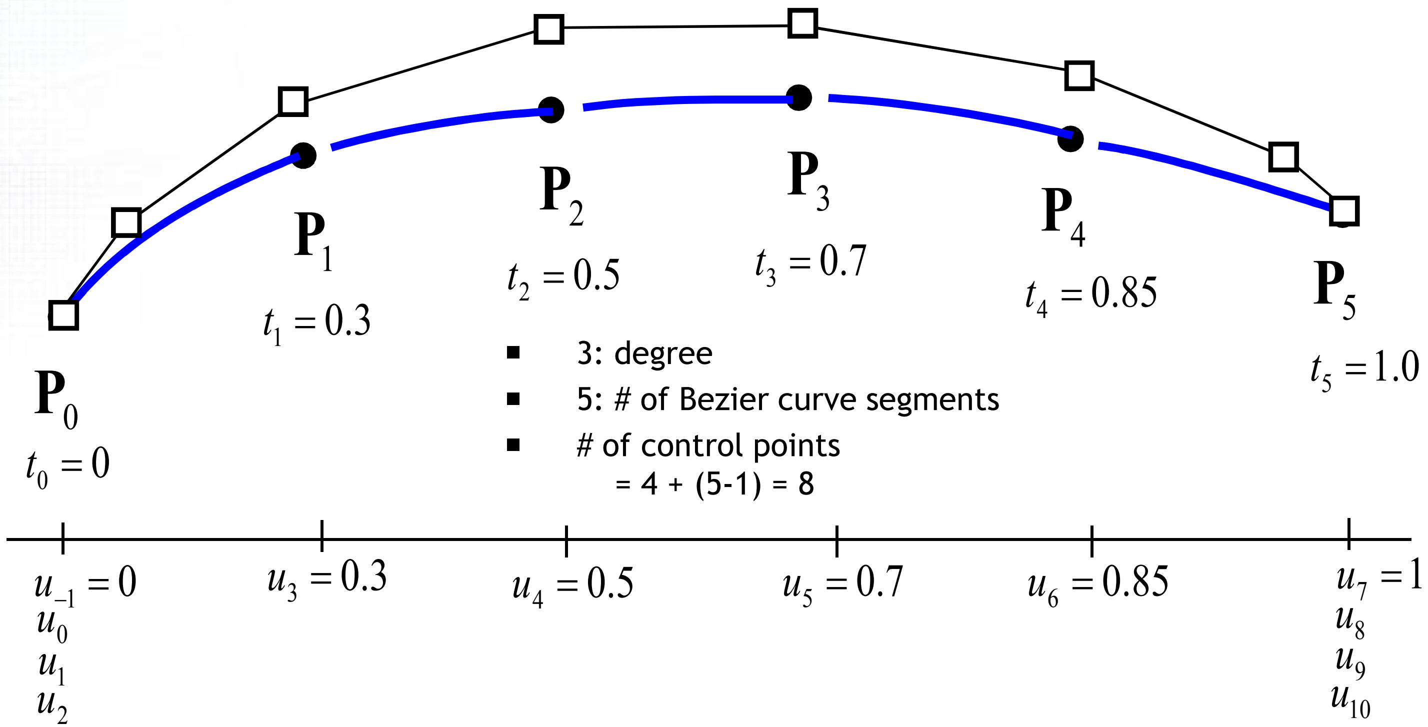
곡선 상의 점 $\mathbf{p}_i$을 지나고
C^2 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)



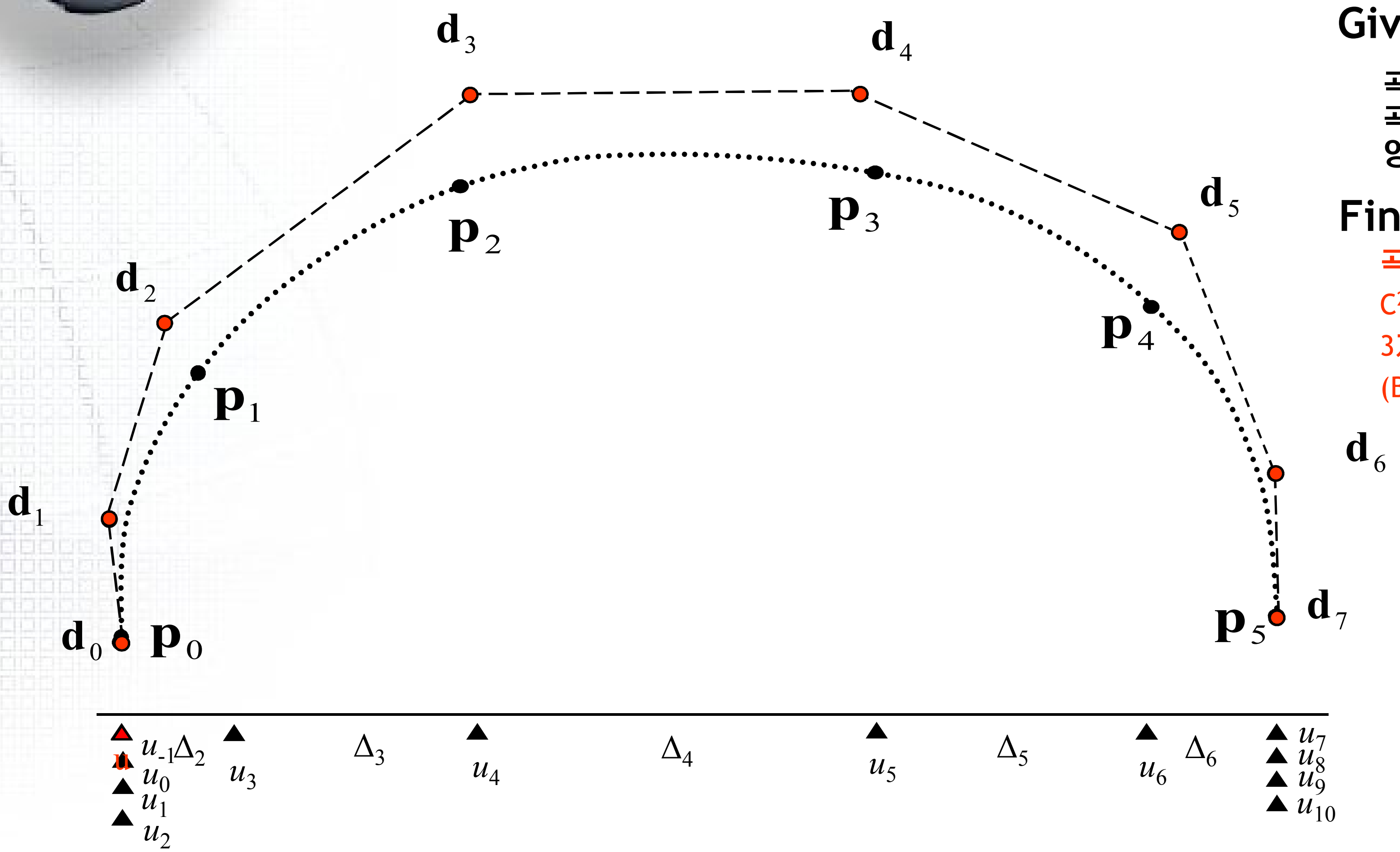
- 3: degree
- 3: # of Bezier curve segments
- # of control points
  = 4 + (3-1) = 6
- How we determine Knots ?
  (= start / end points of each
  cubic Bezier curve)

2.3.5.1 Determine # of Bezier curve segment & Knot value (3)

- Given: fitting points $\mathbf{P}_i$ and corresponding parameter t_i
  where, $i = 0,1,\ldots,m$ and $t_0 = 0,\ t_m = 1,$
- ① determine # of Bezier curve segment to be (# of fitting point -1)
- ② We can determine knots to be the same as the parameters t_i
- ③ How about the B-spline control points ?



2.3.5.2 Problem definition of cubic B-spline curve interpolation



Given:

곡선 상의 점 $\mathbf{p}_i$, t_i
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0$, $\mathbf{t}_1$

Find:

곡선 상의 점 $\mathbf{p}_i$을 지나고
C^2 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)

가정 : 각 곡선 세그먼트는 3차 Bezier Curve 이다.
연결점에서는 C^1, C^2 연속조건을 만족한다.

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

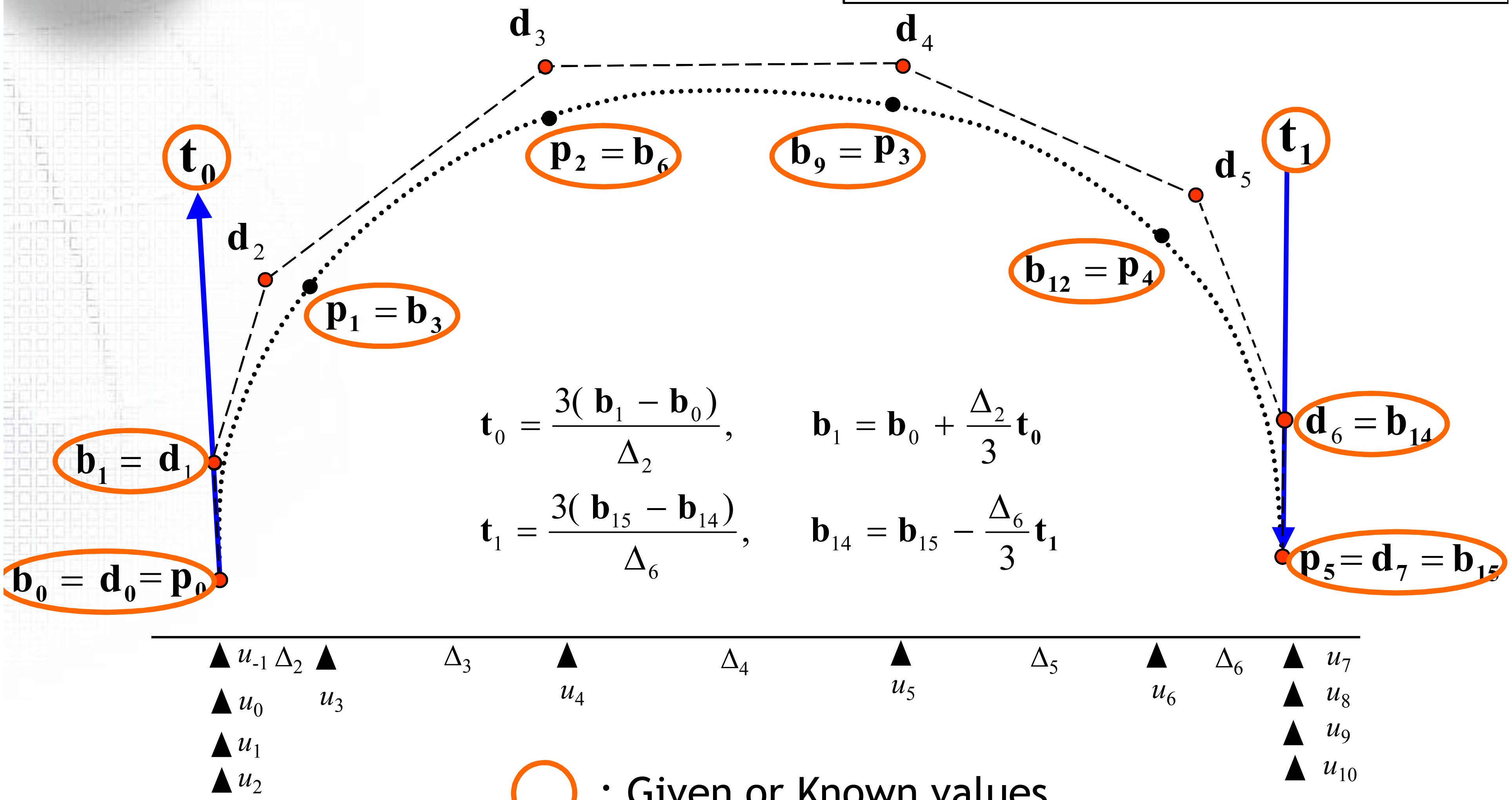
2.3.5.3 Determine **Bezier end control points** by end tangent vectors

Given:

곡선 상의 점 $\mathbf{p}_i, t_i$
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0, \mathbf{t}_1$

Find:

곡선 상의 점 $\mathbf{p}_i$을 지나고
C^2 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)



○ : Given or Known values

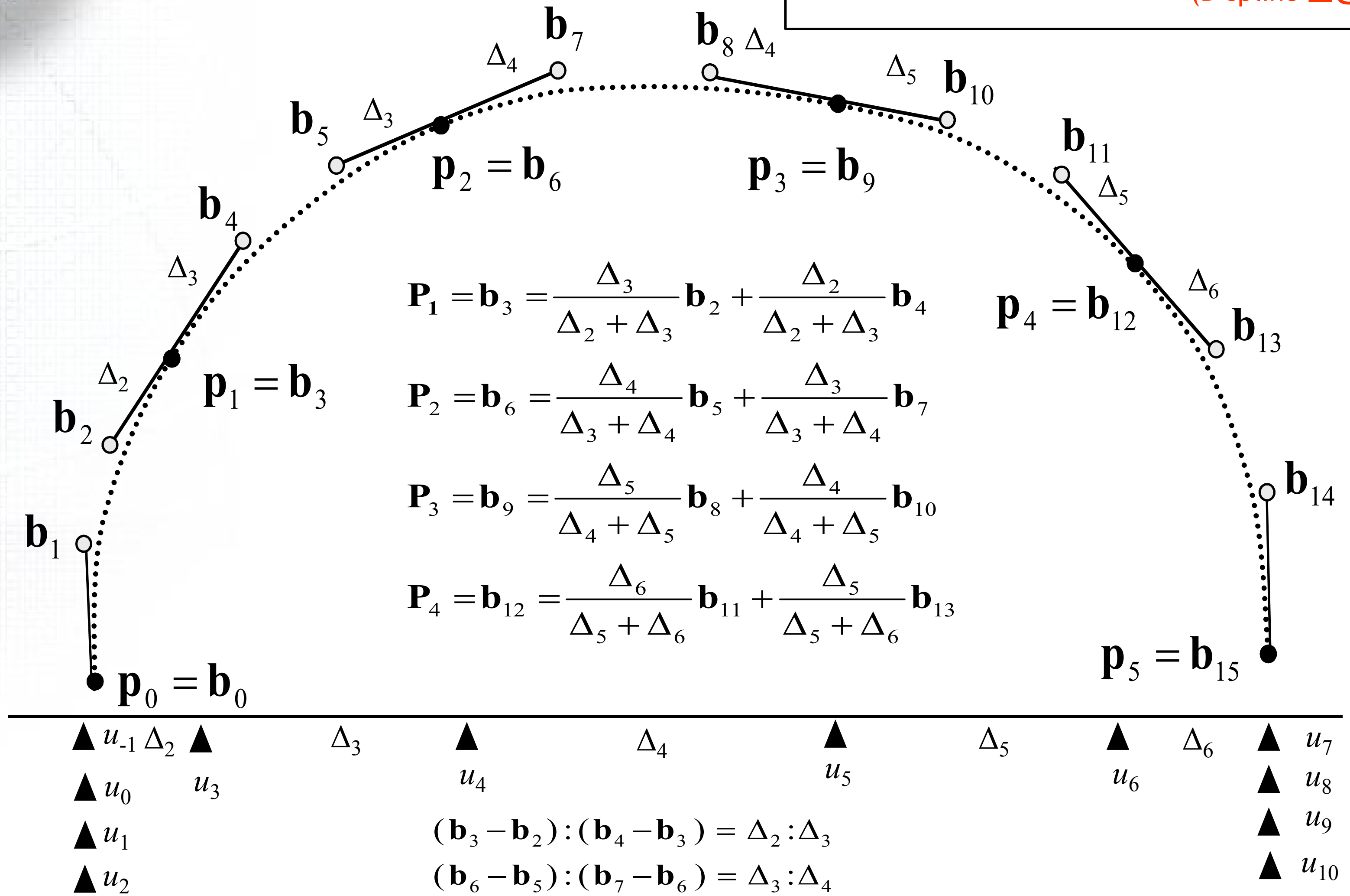
2.3.5.4 Determine **Bezier control points** by C^1 continuity condition

Given:

곡선 상의 점 $\mathbf{p}_i$, t_i
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0$, $\mathbf{t}_1$

Find:

곡선 상의 점 $\mathbf{p}_i$을 지나고
C^2 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)



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

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

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

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

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

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

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

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

2.3.5.5 Determine B-spline control points by C² continuity condition (1)

Given:

곡선 상의 점 $\mathbf{p}_i$, t_i

곡선의 놋트 u_j

양끝단의 접선 벡터 $\mathbf{t}_0$, $\mathbf{t}_1$

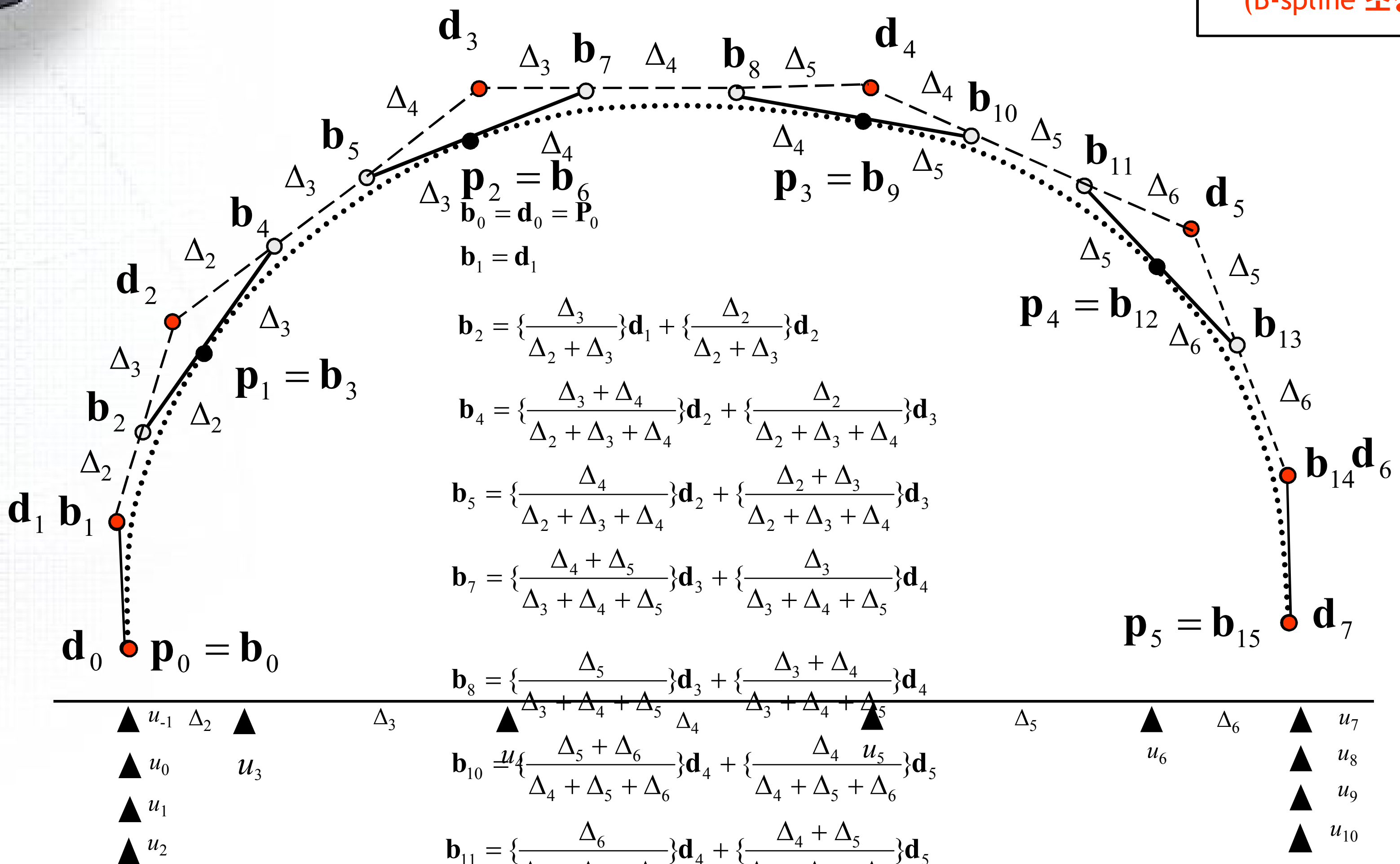
Find:

곡선 상의 점 $\mathbf{p}_i$을 지나고

C² 연속 조건을 만족하는

3차 B-spline 곡선 $\mathbf{r}(u)$

(B-spline 조정점: $\mathbf{d}_i$)



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

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

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

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

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

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

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

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

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

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

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

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

2.3.5.5 Determine B-spline control points by C^2 continuity condition (2)

Given:

곡선 상의 점 $\mathbf{p}_i, t_i$
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0, \mathbf{t}_1$

Find:

곡선 상의 점 $\mathbf{p}_i$을 지나고
C^2 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)

C^1, C^2 조건을 이용하여 P_i에 관한 식 유도

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

2.3.5.5 Determine B-spline control points by C² continuity condition (3)

Given:
곡선 상의 점 $\mathbf{p}_i, t_i$
곡선의 놋트 u_j
양끝단의 접선 벡터 $\mathbf{t}_0, \mathbf{t}_1$

Find:
곡선 상의 점 $\mathbf{p}_i$을 지나고
C² 연속 조건을 만족하는
3차 B-spline 곡선 $\mathbf{r}(u)$
(B-spline 조정점: $\mathbf{d}_i$)

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

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

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

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

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

$$\beta_i = \left\{\frac{\Delta_{i+2}(\Delta_i+\Delta_{i+1})}{(\Delta_i+\Delta_{i+1}+\Delta_{i+2})} + \frac{\Delta_{i+1}(\Delta_{i+2}+\Delta_{i+3})}{(\Delta_{i+1}+\Delta_{i+2}+\Delta_{i+3})}\right\}/(\Delta_{i+1}+\Delta_{i+2})$$

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

주어진 것 　　　　구해야 하는 것

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

2.3.5.6 Tridiagonal matrix 해법을 이용한 B-spline 곡선 조정점($\mathbf{d}_i$) 결정(1)

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

주어진 것 계산할 수 있는 것 구해야 하는 것

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

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

그런데 행렬 A가 Tri-diagonal matrix이므로 간단하게 A^{-1}를 계산할 수 있음

2.3.5.6 Tridiagonal matrix 해법을 이용한 B-spline 곡선 조정점($\mathbf{d}_i$) 결정(2)

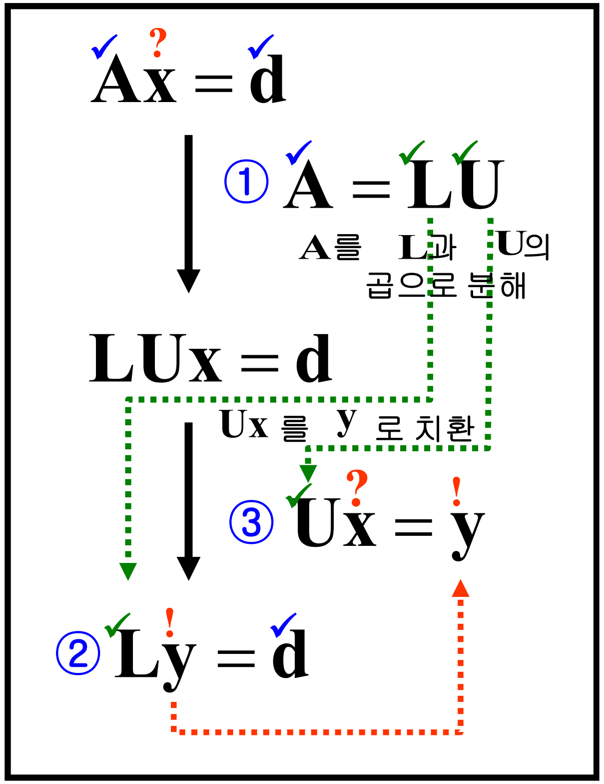
Tridiagonal matrix

대각 성분과 그 위/아래, 좌/우 성분만 0이 아닌 값이고, 나머지 성분은 0 값인 행렬
즉, 대각 성분을 중심으로 3개의 성분만 0이 아닌 값 ➡ Tri + Diagonal

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

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

$\mathbf{A}$와 $\mathbf{d}$를 알고 있을 때, $\mathbf{x}$ 구하기



① $\mathbf{A}$를 $\mathbf{L}$과 $\mathbf{U}$의 곱으로 분해

② $\mathbf{Ly} = \mathbf{d}$를 만족하는 $\mathbf{y}$구하기

③ $\mathbf{Ux} = \mathbf{y}$를 만족하는 $\mathbf{x}$를 구하면, 곧 $\mathbf{Ax} = \mathbf{d}$를 만족하는 $\mathbf{x}$를 구하는 것임

2.3.5.6 Tridiagonal matrix 해법을 이용한 B-spline 곡선 조정점($\mathbf{d}_i$) 결정(3)

① $\mathbf{A} = \mathbf{L}\mathbf{U}$

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

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

	$b_0 = \beta_0$	$c_0 = \beta_0\gamma_1$
$a_1 = \alpha_1$	$b_1 = \alpha_1\gamma_1 + \beta_1$	$c_1 = \beta_1\gamma_2$
$a_2 = \alpha_2$	$b_2 = \alpha_2\gamma_2 + \beta_2$	$c_2 = \beta_2\gamma_3$
$\vdots$	$\vdots$	$\vdots$
$a_{n-1} = \alpha_{n-1}$	$b_{n-1} = \alpha_{n-1}\gamma_{n-1} + \beta_{n-1}$	$c_{n-1} = \beta_{n-1}\gamma_n$
$a_n = \alpha_n$	$b_n = \alpha_n\gamma_n + \beta_n$	

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

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

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

with $\beta_0 = b_0$

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

↓ ① $\mathbf{A} = \mathbf{L}\mathbf{U}$

$\mathbf{L}\mathbf{U}\mathbf{x} = \mathbf{d}$

↓ ③ $\mathbf{U}\mathbf{x} = \mathbf{y}$

② $\mathbf{L}\mathbf{y} = \mathbf{d}$

2.3.5.6 Tridiagonal matrix 해법을 이용한 B-spline 곡선 조정점($\mathbf{d}_i$) 결정(4)

② $\mathbf{L}\mathbf{y} = \mathbf{d}$

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

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

$$\beta_0 y_0 = d_0$$

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

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

$$\vdots$$

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

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

Forward substitution

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

with $y_0 = \dfrac{d_0}{\beta_0}$

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

① $\mathbf{A} = \mathbf{L}\mathbf{U}$

$\mathbf{L}\mathbf{U}\mathbf{x} = \mathbf{d}$

③ $\mathbf{U}\mathbf{x} = \mathbf{y}$

② $\mathbf{L}\mathbf{y} = \mathbf{d}$

2.3.5.6 Tridiagonal matrix 해법을 이용한 B-spline 곡선 조정점($\mathbf{d}_i$) 결정(5)

③ $\mathbf{U}\mathbf{x} = \mathbf{y}$

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

$$x_0 + \gamma_0 x_1 = y_0$$

$$x_1 + \gamma_1 x_2 = y_1$$

$$x_2 + \gamma_2 x_3 = y_2$$

$$\vdots$$

$$x_{n-1} + \gamma_{n-1} x_n = y_{n-1}$$

$$x_n = y_n$$

Backward substitution

$$x_i = y_i - \gamma_{i+1} x_{i+1}$$

$$i = n-1, \cdots, 0$$

with $x_n = y_n$

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

① $\mathbf{A} = \mathbf{L}\mathbf{U}$

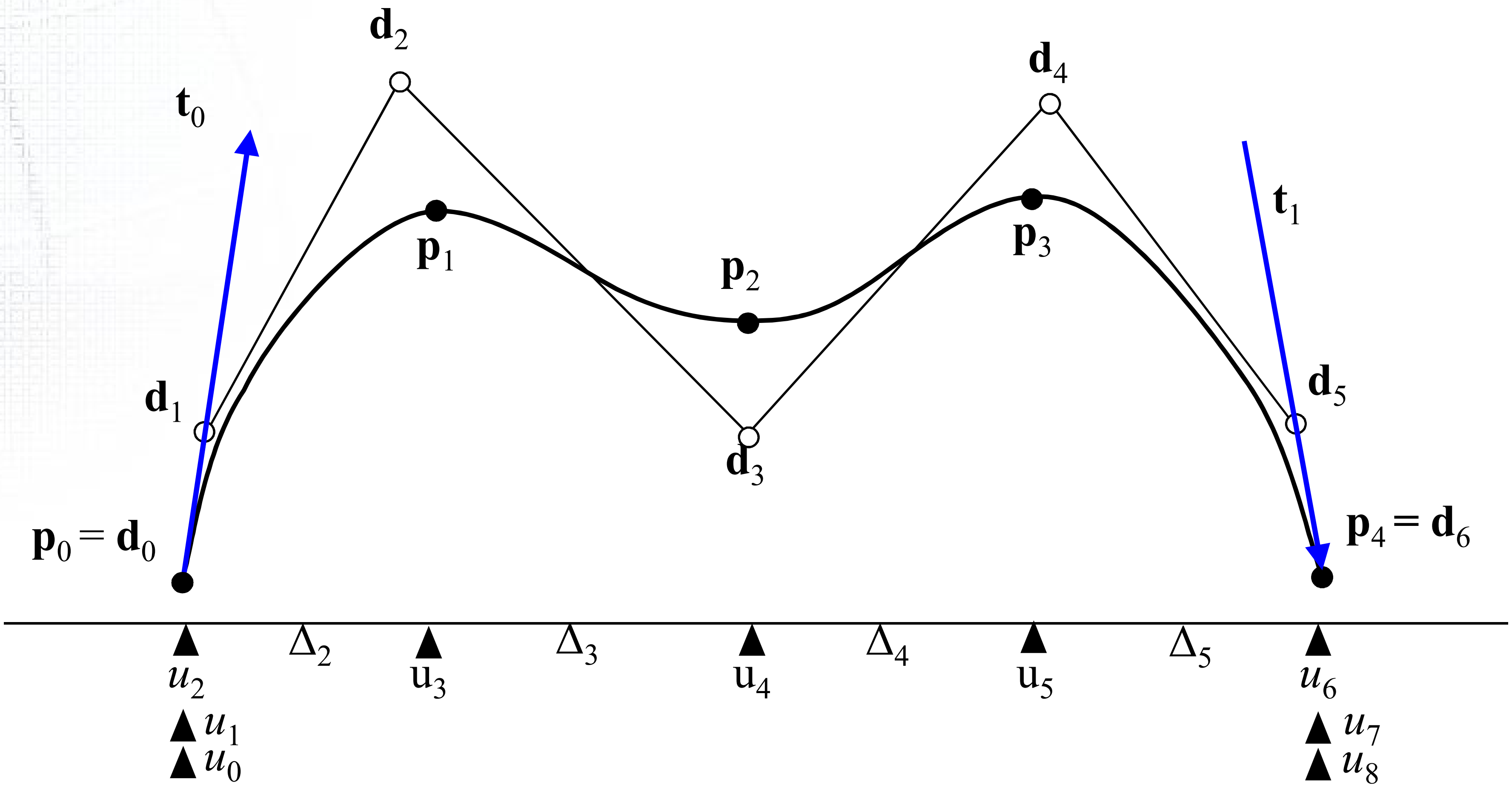
$\mathbf{L}\mathbf{U}\mathbf{x} = \mathbf{d}$

③ $\mathbf{U}\mathbf{x} = \mathbf{y}$

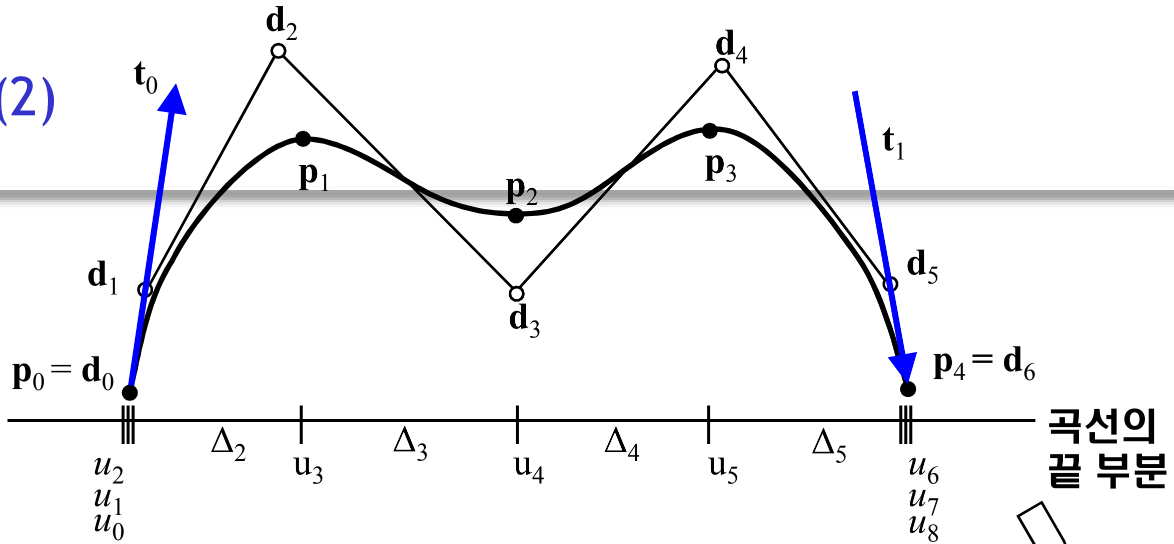
② $\mathbf{L}\mathbf{y} = \mathbf{d}$

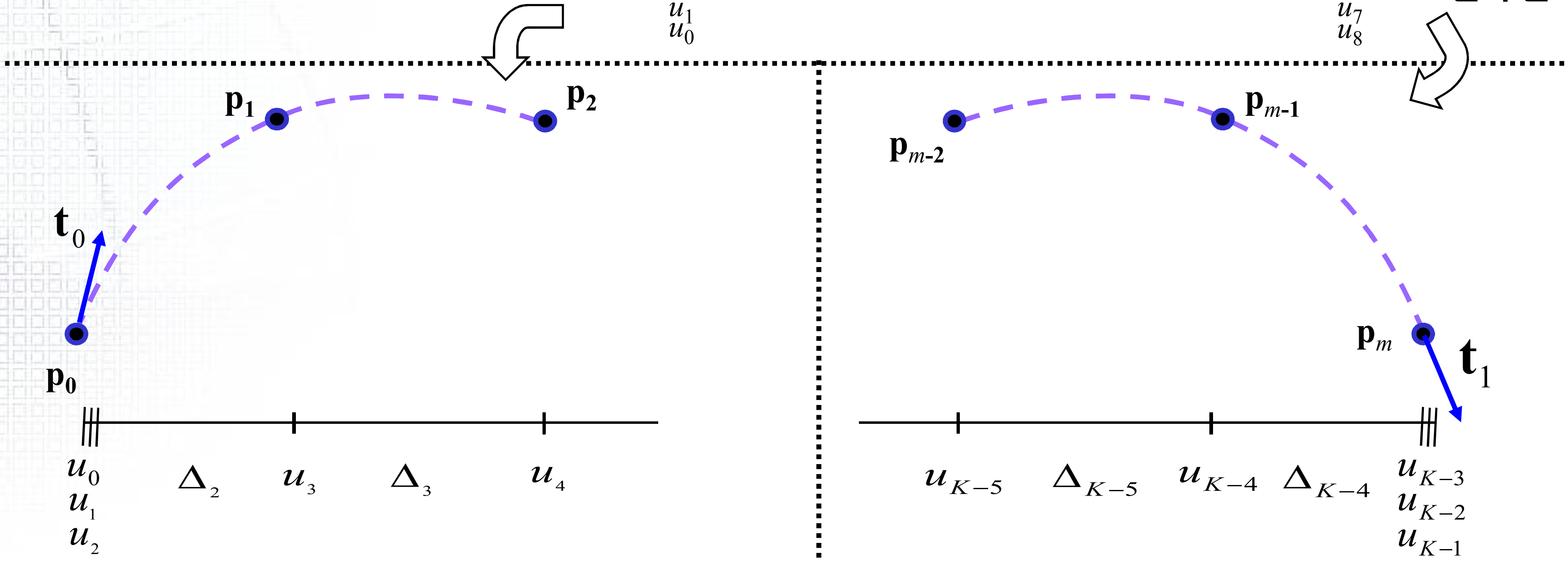
2.3.5.7 Bessel End Condition (1)

- B-spline curve interpolation에서 양끝점에서의 접선벡터 t_0, t_1이 주어지지 않았을 때,
  (1) 곡선의 양끝의 연속된 세 점으로부터 2차 곡선(quadratic curve)을 생성하고,
  (2) 생성된 2차 곡선의 양 끝점에서의 1차 미분값을 우리가 생성하고자 하는 B-spline curve의 양 끝점에서의 접선 벡터로 가정하는 방법



2.3.5.7 Bessel End Condition(2)





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

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

2.3.5.8 Sample code of Cubic B-spline Curve (1)

```
#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
```

2.3.5.8 Sample code of Cubic B-spline Curve (2)

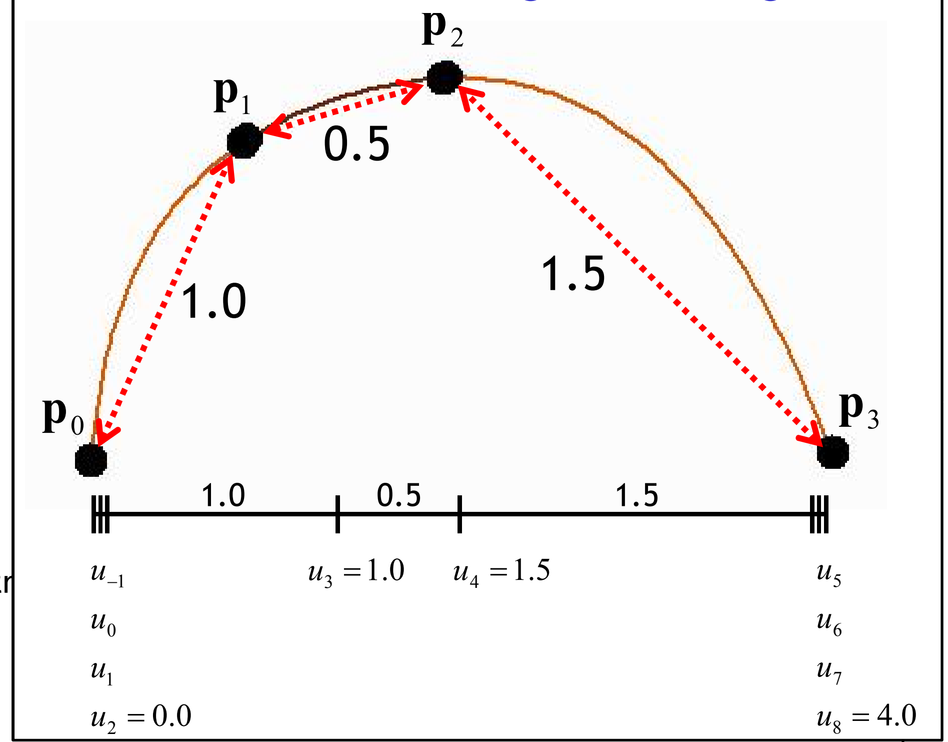
```
void CubicBsplineCurve::Interpolate(Vector *pFittingPoint, int nFittingPoint)
{
  // Generate Knot
    if(m_Knot) delete[] m_Knot;
    m_nKnot = (m_nFittingPoint - 2) + 2*(3+1);
    m_Knot = new double [m_nKnot];
    // Use Chord length or Centripetal method
    ......

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

  // 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



2.3.5.8 Sample code of Cubic B-spline Curve (2)

```
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] = EndTangentByBesselEndCondition;
```

Bessel End Condition

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

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

2.3.5.8 Sample code of Cubic B-spline Curve (3)

```
    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;
}
```

LU 분해법을 이용하여 역행렬을 계산

Term Project #1
B-Spline Curve Interpolation

Advanced
Ship
Design
Automation
Laboratory

Term Project 1. B-Spline Curve Interpolation
- 과제 개요

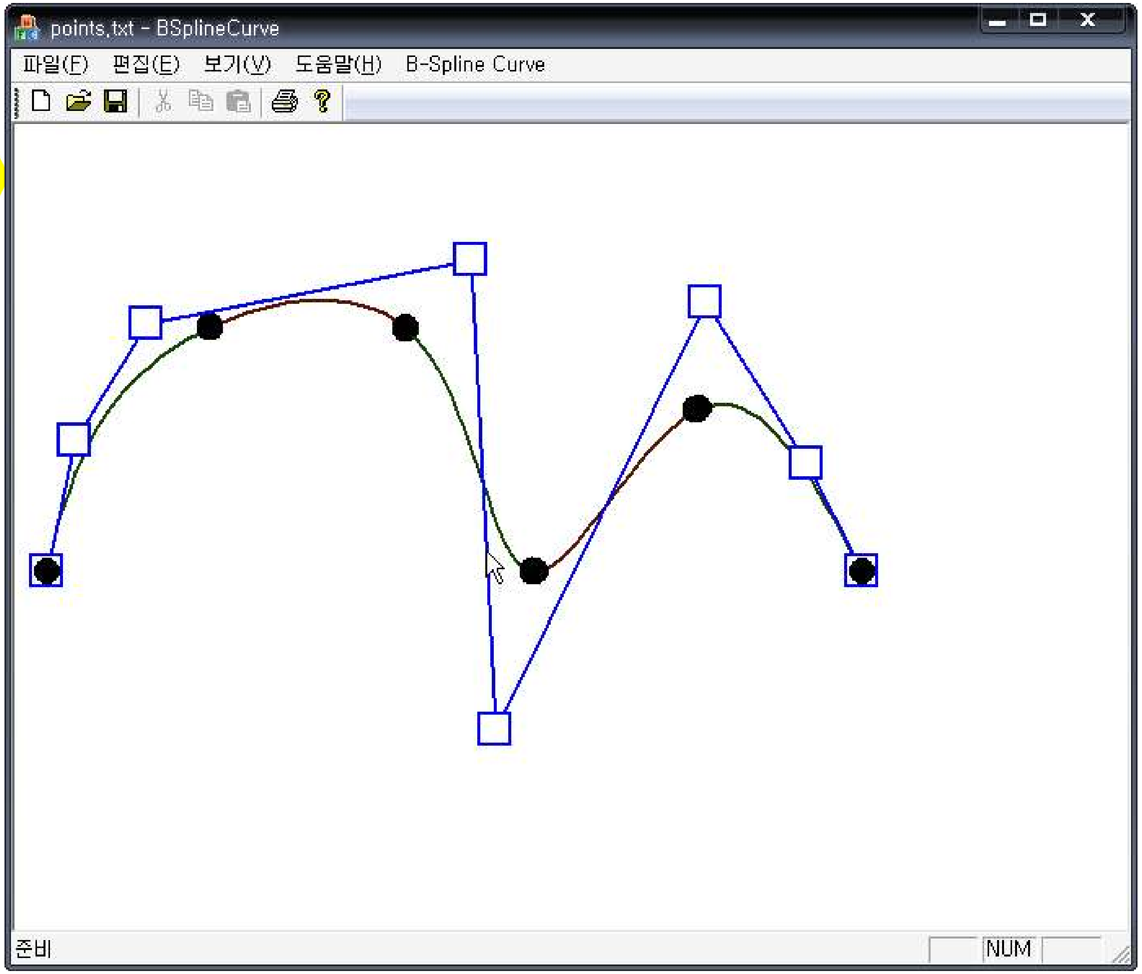
과제 목표: 입력 받은 점을 지나는 B-Spline Curve를 Interpolation 한다.

Input Data

3

6

0.0	200.0	0.0
100.0	350.0	0.0
220.0	350.0	0.0
300.0	200.0	0.0
400.0	300.0	0.0
500.0	200.0	0.0



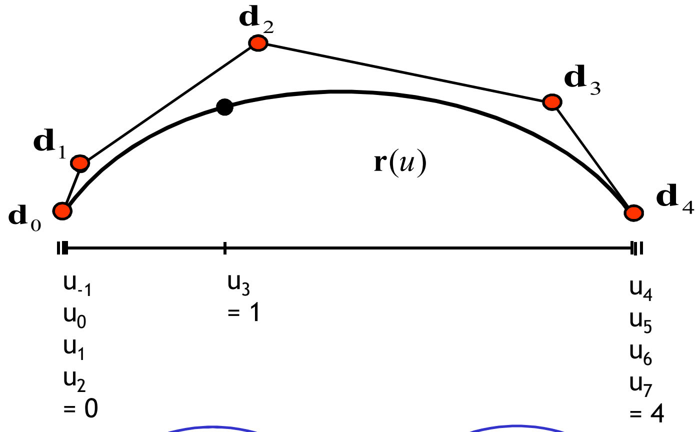
Term Project 1. B-Spline Curve Interpolation
- Input Data Format

Input Data

3			→ B-Spline 곡선의 차수
6			→ 입력 받을 점의 개수
0.0	200.0	0.0	→ 입력 받는 점의 3차원 좌표 (x, y, z)
100.0	350.0	0.0	
220.0	350.0	0.0	
300.0	200.0	0.0	
400.0	300.0	0.0	
500.0	200.0	0.0	

Cubic B-Spline 곡선식과
Cox-de Boor Recurrence Formula

- 예: Cubic B-Spline 곡선



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

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

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

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

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

Term Project 1. B-Spline Curve Interpolation

- 과제 수행 시 프로그램 작성 부분

```cpp
void CBSplineCurveDoc::SetPointFromFile(CArchive& ar)
{
    int i = 0;

    // 파일 이름 가져오기
    CFile* pFile = ar.GetFile();
    CString strFile = pFile->GetFileName();

    char filename[256];
    strcpy(filename, strFile.GetBuffer(strFile.GetLength()));
    strFile.ReleaseBuffer();

    // 파일 열기
    FILE* fpIn = NULL;
    fpIn = fopen(filename, "r");

    ////////////////////////////////////////////////////////////////////
    // 파일로부터 데이터를 입력받는 코드를 작성하시오.

    // 곡선의 차수 입력


    // 입력 받을 점의 개수 입력


    // 입력 받은 점의 개수 만큼 벡터 포인터를 배열로 동적 할당


    // 점 입력


    // 파일 닫기
    fclose(fpIn);
    fpIn = NULL;
}
```

Class	CBSplineCurveDoc
Function	SetPointFromFile
내용	**입력 파일로부터 B-Spline의 곡선 차수, 입력 받을 점의 개수와 데이터를 읽어옴**

Term Project 1. B-Spline Curve Interpolation
- 과제 수행 시 프로그램 작성 부분

```cpp
double CBSpline::N(int n, int i, double u)
{
    // check invalid condition
    if (m_nNumOfKnot < 1) return 0.0;
    if (m_nDegree < 2) return 0.0;

    // initialize
    double val = 0.0;

    /*******************************************************
    /* B-Spline Basis Function을 작성하여 입력하시오.
    /*******************************************************
    // n != 0
    if (n != 0)
    {

    }
    // n = 0
    else
    {

    }

    return val;
}
```

Class	CBSpline
Function	N
내용	B-Spline Basis Function(Cox-de Boor Recurrence Formula) **를 계산**

Cox-de Boor Recurrence Formula
(B-Spline basis function)

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

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

Term Project 1. B-Spline Curve Interpolation
- 과제 수행 시 프로그램 작성 부분

```
Vector CBSpline::GetPoint(double u)
{
    // return value
    Vector vec(0.0, 0.0, 0.0);

    // check invalid condition
    if (m_nDegree < 2) return vec;
    if (m_nNumOfCP < 1) return vec;

    // initialize
    int i = 0;

    ///////////////////////////////////////////////////////////////////////
    // Parameter u에 대한 r(u)를 구하는 B-Spline 곡선식을 작성하시오.


    return vec;
}
```

Class	CBSpline
Function	GetPoint
내용	Parameter u**에 대한 곡선 상의 점** $\mathbf{r}(u)$**를 구하는 식 구현**

Cubic B-Spline Curve $\mathbf{r}(u)$

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

Term Project 1. B-Spline Curve Interpolation

- 과제 수행 시 프로그램 작성 부분

```
void CBSpline::SetKnotUsingChordLength(Vector* pPoint, int
{
    // check invalid condition
    if (nNumOfPoint < 1) return;
    if (m_nDegree < 2) return;

    // initialize
    int i = 0;

    // free memory
    if (m_pKnot != NULL) delete[] m_pKnot;

    ////////////////////////////////////////////////////////////////////////////
    // B-Spline Curve를 이용하여 Interpolation 시 Chord Length(Fitting Point 간의 거리를 이용하여
knot를 계산하시오.

}
```

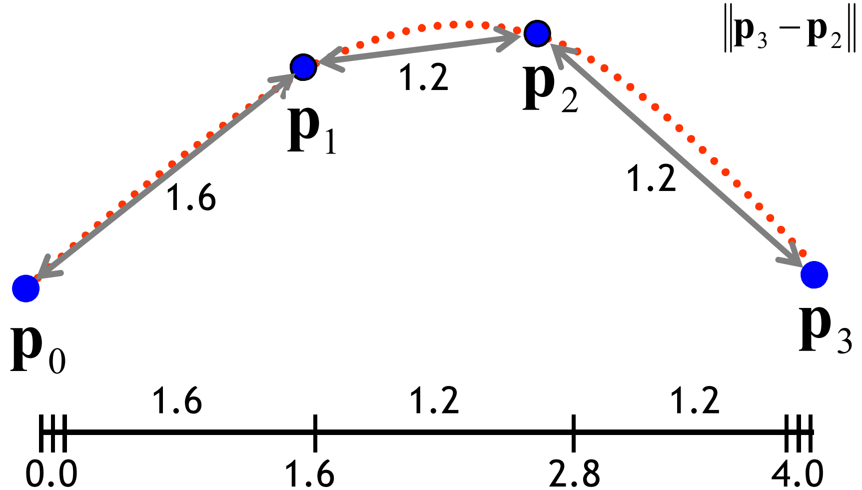
Class	CBSpline
Function	SetKnotUsingChordLength
내용	**주어진 점 간의 거리**(Chord Length)**를 이용하여** Knot**를 생성하는 함수**

점이 4개 주어졌을 때 Knot 설정 예시

$$\|\mathbf{p}_1 - \mathbf{p}_0\| = 1.6$$

$$\|\mathbf{p}_2 - \mathbf{p}_1\| = 1.2$$

$$\|\mathbf{p}_3 - \mathbf{p}_2\| = 1.2$$



Term Project 1. B-Spline Curve Interpolation

- 과제 수행 시 프로그램 작성 부분

```cpp
void CBSpline::Interpolation(Vector* pPoint, int nNumOfPoin
{
    ////////////////////////////////////////////////////////
    // 주어진 Fitting Point를 이용하여 3차 B-Spline Curve로

    // Control Point의 개수 지정


    // set knots and delta using chord length


    // set fitting point

    // bessel end condition: start


    // bessel end condition: end


    // allocate pointer M


    // initialize M


    // set matrix M using alpha, beta, gamma


    // LU decomposition


    // set control point of b-spline curve


}
```

Bessel End Condition

Class	CBSpline
Function	Interpolation
내용	**입력 데이터를 이용하여** B-Spline Curve Interpolation**을 수행하는 함수**

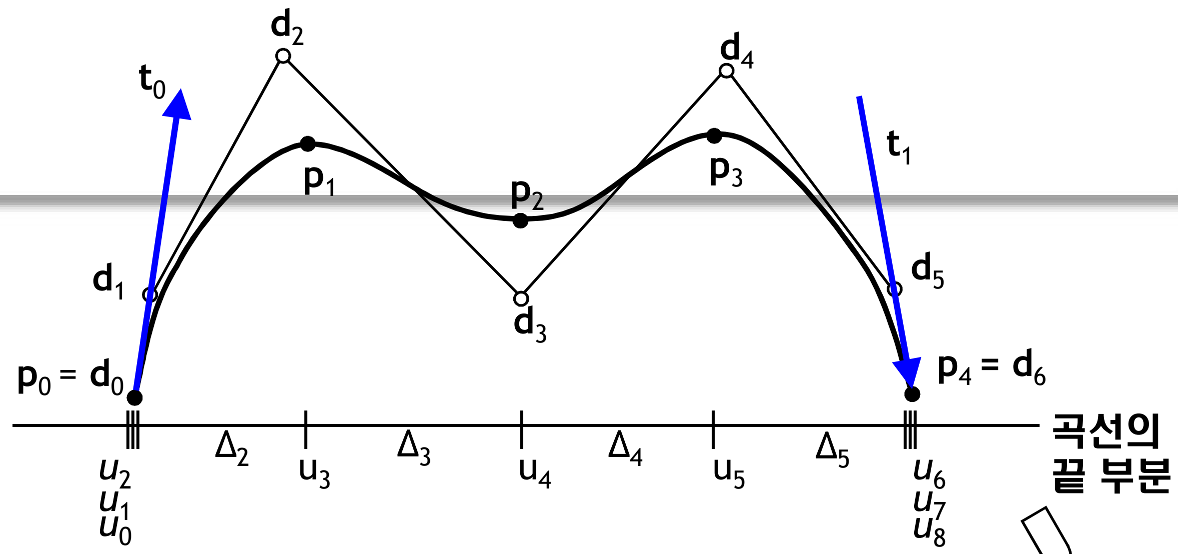
$$
\underbrace{\begin{bmatrix} \mathbf{p}_0 \\ \mathbf{t}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \\ \mathbf{p}_4 \\ \mathbf{t}_1 \\ \mathbf{p}_5 \end{bmatrix}}_{=\mathbf{D}}
=
\underbrace{\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{-3}{\Delta_2} & \frac{3}{\Delta_2} & 0 & 0 & 0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

주어진 것   **계산할 수 있는 것**   **구해야 하는 것**

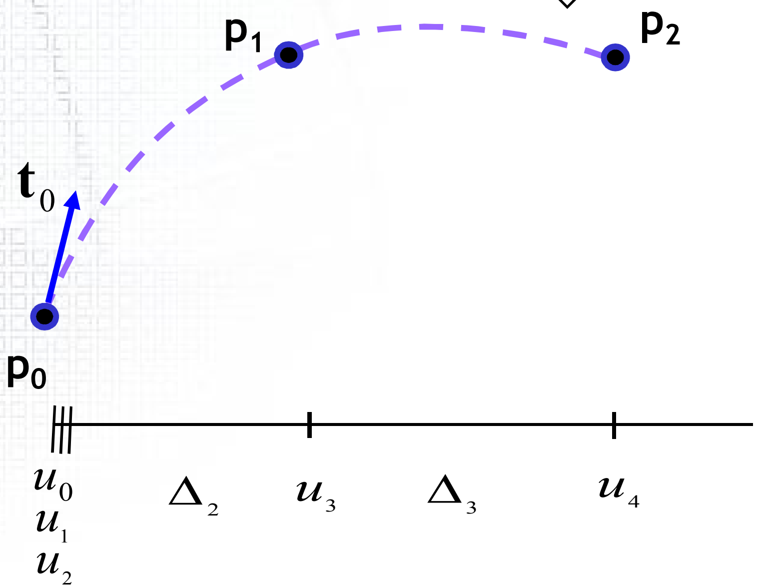
$$\mathbf{D} = \mathbf{A}\mathbf{X}$$

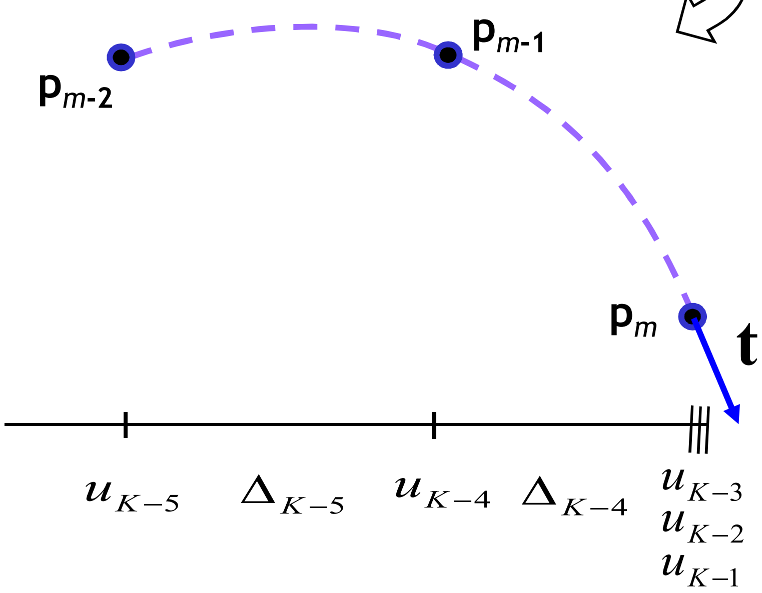
$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{D}$$

Bessel End Condition



곡선의 시작 부분





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

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

Term Project 1. B-Spline Curve Interpolation
- 과제 수행 시 프로그램 작성 부분

```
double CBSpline::GetAlpha(int n, double* delta)
{
    double val = 0.0;

    ////////////////////////////////////////////////////
    // Interpolation 시 사용하는 alpha에 관한 함수를 작성하

    return val;
}

double CBSpline::GetBeta(int n, double* delta)
{
    double val = 0.0;

    ////////////////////////////////////////////////////
    // Interpolation 시 사용하는 beta에 관한 함수를 작성하시

    return val;
}

double CBSpline::GetGamma(int n, double* delta)
{
    double val = 0.0;

    ////////////////////////////////////////////////////
    // Interpolation 시 사용하는 gamma에 관한 함수를 작성하

    return val;
}
```

Class	CBSpline
Function	GetAlpha, GetBeta, GetGamma
내용	Interpolation **시 사용하는** α_i, β_i, γ_i **를 구하는 함수**

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

$$\beta_i = \left\{\frac{\Delta_{i+2}(\Delta_i + \Delta_{i+1})}{(\Delta_i + \Delta_{i+1} + \Delta_{i+2})} + \frac{\Delta_{i+1}(\Delta_{i+2} + \Delta_{i+3})}{(\Delta_{i+1} + \Delta_{i+2} + \Delta_{i+3})}\right\} / (\Delta_{i+1} + \Delta_{i+2})$$

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

Term Project 1. B-Spline Curve Interpolation

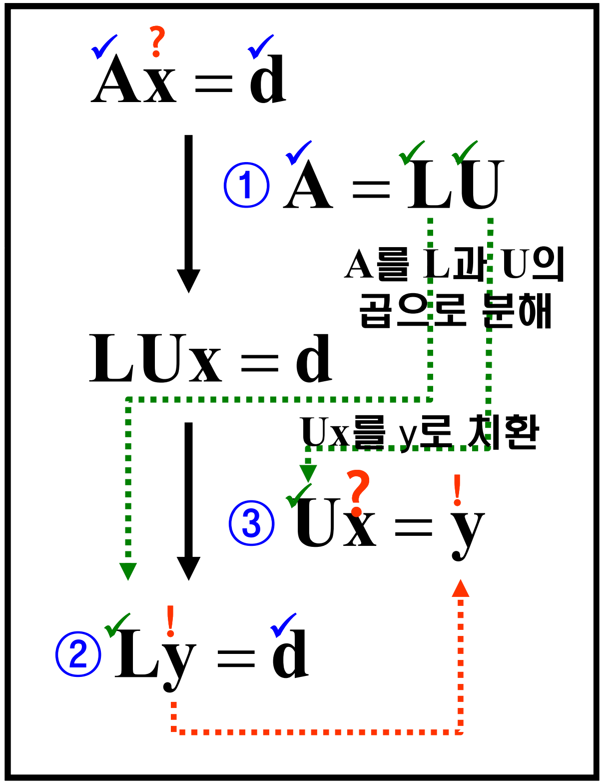
- 과제 수행 시 프로그램 작성 부분

```
void CBSpline::LU(int n, double** A, Vector* x, Vector* re
{
    ////////////////////////////////////////////////////////////////
    // LU 분해법에 대한 코드를 작성하시오.
}
```

Class	CBSpline
Function	LU
내용	LU **분해법을 구현**

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

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



A와 d를 알고 있을 때, x 구하기

① A를 L과 U의 곱으로 분해

② $\mathbf{Ly} = \mathbf{d}$ 를 만족하는 y 구하기

③ $\mathbf{Ux} = \mathbf{y}$ 를 만족하는 x 를 구하면 $\mathbf{Ax} = \mathbf{d}$ 를 만족하는 x를 구하는 것임