Computer Aided Ship design

-Part II. Hull Form Modeling-

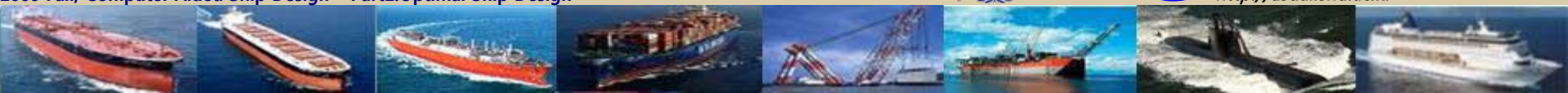
Nonember, 2009

Prof. Kyu-Yeul Lee

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

Seoul National Univ.

@SDAL Advanced Ship Design Automation Lab.
http://asdal.snu.ac.kr



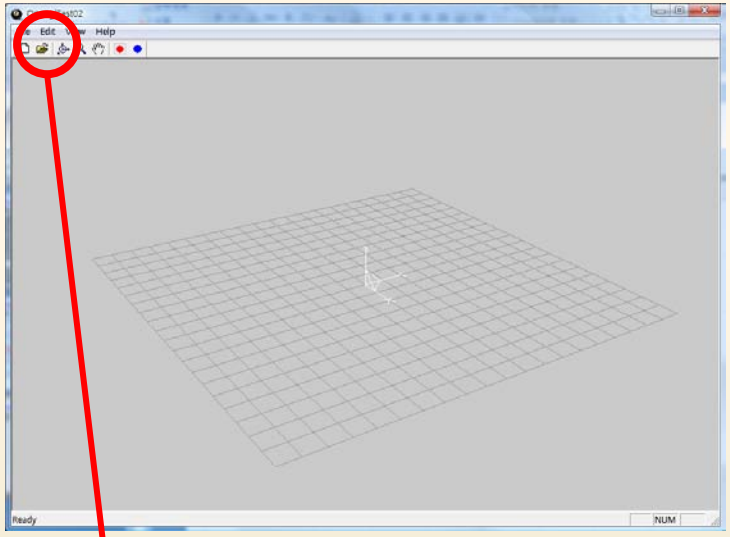
PA#6 3차원 공간상의 주어진 점을 지나는 B-Spline Curve 보간 프로그램

2009.11.11

서울대학교 조선해양공학과
선박설계자동화연구실

B-spline 곡선 프로그래밍 가이드 (1)

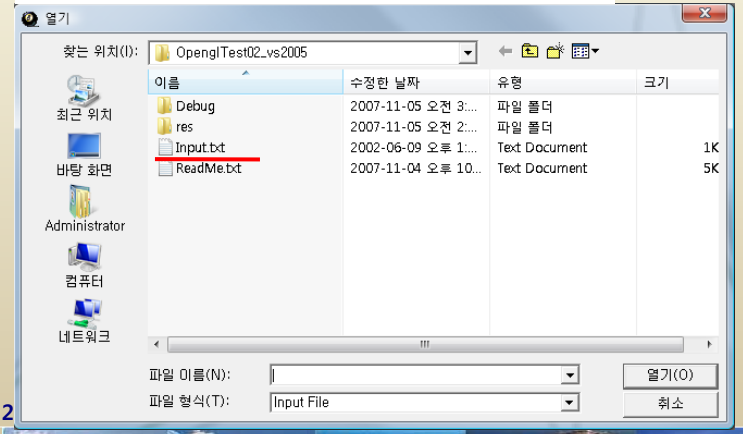
① 초기 화면



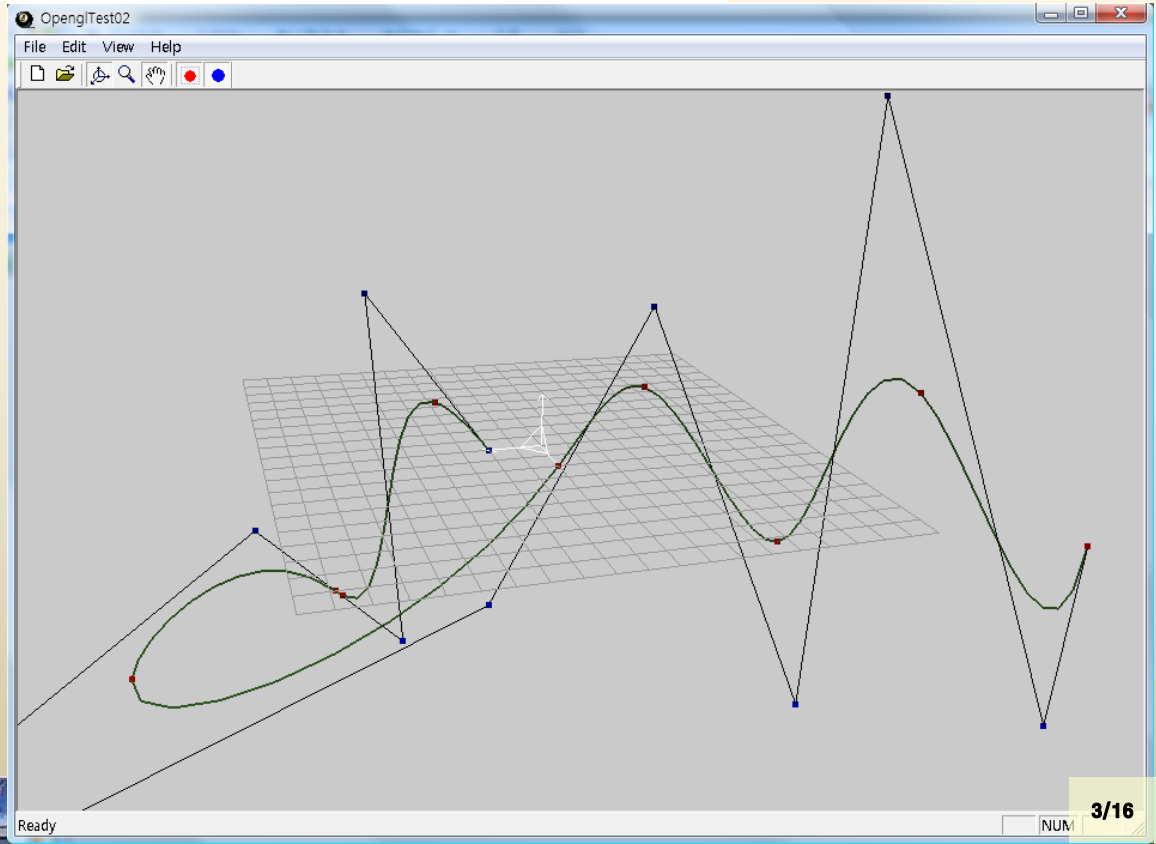
Given : 3차원 공간 상의 점과 접선 벡터

Find : 주어진 점을 지나는 **B-spline** 곡선

② **File open using Dialog box**



③ **B-spline** 점, 곡선, 조정점 가시화



B-spline 곡선 프로그래밍 가이드 (2)

(1) 파일로 부터 위치 벡터(점의 좌표)를 입력

(2) B-spline 곡선 생성
(Knot 및 조정점을 필요한 개수만큼 동적할당)

(3) Knot 간격 설정
(두 점 사이의 거리를 기준으로 함)

법선벡터 주어지 않음

법선벡터
주어짐

(4) Bessel End Condition으로
법선 벡터구함

(5) LU 분해법을 통해 조정점 계산

(6) 파라미터 값(u)를 변경시켜가며
곡선 위의 점을 구함

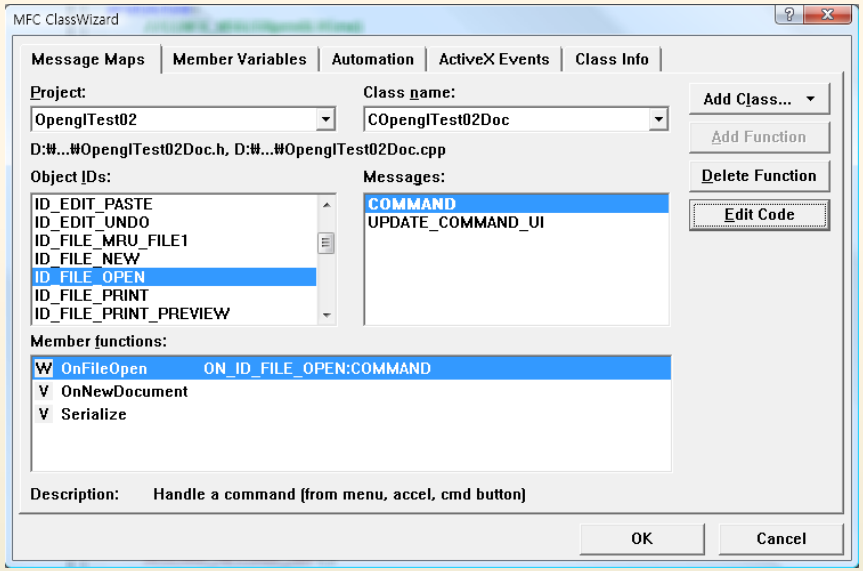
< Input Text file >

10			Number of Point
100	0	0	Position Vector
200	0	100	
400	200	-150	
410	190	-145	
700	600	0	
0	100	0	
-200	0	100	
-400	200	-150	
-500	400	200	
-700	600	0	
1	0	0	Tangent Vector (Optional)
0	0	1000	

Tangent Vector가 주어지지 않는 경우
Bessel End Condition으로 구함

B-spline 곡선 프로그래밍 가이드 (3)

① 파일로 부터 위치 벡터(점의 좌표)를 입력



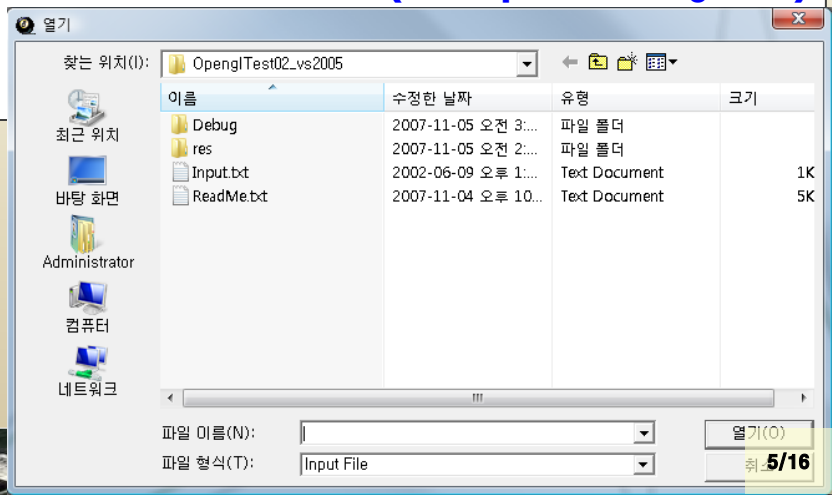
① class wizard 에서 ID_FILE_OPEN을 선택

② COpenglTest02Doc 선택

③ COMMAND를 선택 후 Add Function Click

```
void COpenglTest02Doc::OnFileOpen()
{
   char szfilter[]="Input File | *.txt; | All Files (*.*)|*.*||";
   CFileDialog dlg(TRUE,NULL,NULL,OFN_HIDEREADONLY,szfilter

   FILE* fp;
   CString filename;

   if (dlg.DoModal() == IDOK)
   {
      filename = dlg.GetPathName();
      fp = fopen(filename, "r");

      //점을 입력 받는 코드 추가

   }
   fclose(fp);
   ...
   ...
}
```

(Fileopen Dialog box)



B-spline 곡선 프로그래밍 가이드 (4)

※ B-spline Class

```
class Bspline
{
public:
//Constructor & Destroyer
   Bspline();
   Bspline(int degree, int num_of_data);
   ~Bspline();

//Member Variables
   int k;
   int NumOfKnot;
   int NumOfControl;
   double* knot;
   Vector* m_ControlPoint;

//Member Function
   void set(int degree,int num_of_data);
   void Interpolation(Vector* InputPoint);
   void calc_Knot(Vector* InputPoint);
   void set_up_system(int n, double* alpha, double* beta,double* gamma);
   void l_u_system(double* alpha, double* beta, double* gamma, int n, double* up, double* low);
   void solve_system(double* up, double* low, double* gamma, int n, Vector* InputPoint);

   void Evaluation(double u,Vector* point);
   double cox_deboor_recursion(int i, int k, double u);
};
```

LU 분해법 구현

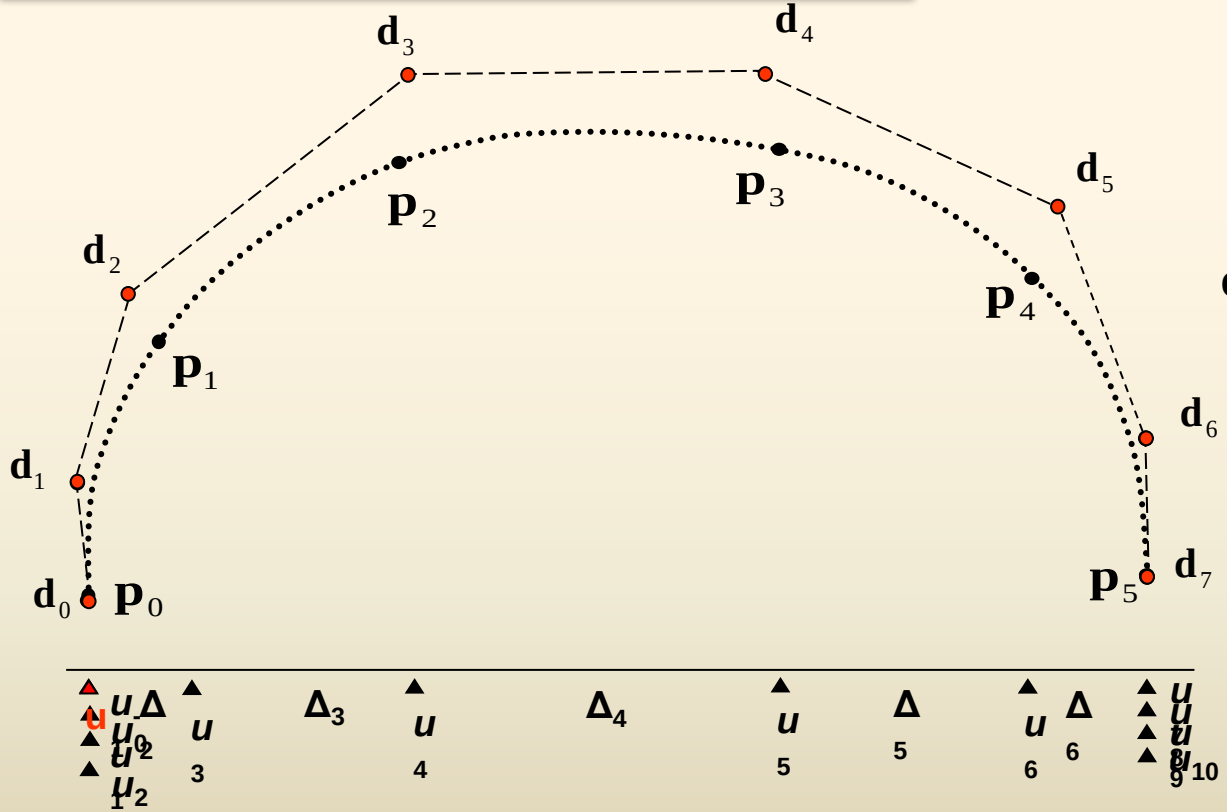
B-spline 곡선 프로그래밍 가이드 (5)

※ Interpolation 함수 내부

```
void Bspline::Interpolation(Vector* InputPoint)
{
        calc_Knot(InputPoint);

        double* alpha = new double[NumOfControl];
        double* beta = new double[NumOfControl];
        double* gamma = new double[NumOfControl];
        double* up = new double[NumOfControl];
        double* low = new double[NumOfControl];

        set_up_system(NumOfControl-1,alpha, beta, gamma);
        l_u_system(alpha, beta, gamma, NumOfControl-1, up, low);
        solve_system(up, low, gamma, NumOfControl-1,InputPoint);

        delete[] alpha;
        delete[] beta;
        delete[] gamma;
        delete[] up;
        delete[] low;
}
```

B-spline 곡선 프로그래밍 가이드 (6)

(2) B-spline 곡선 생성
(Knot 및 조정점을 필요한 개수만큼 동적할당)



Given:

- 곡선상의 점의 개수 **(NumOfData)**
- **B-spline** 곡선의 차수 **(Degree)**

ex) NumOfData = 6

Degree = 3 (3차 B-spline 곡선)

# of Knot = 6 + 3X2 = 12 개

# of Control Point = 6 + 2 = 8 개

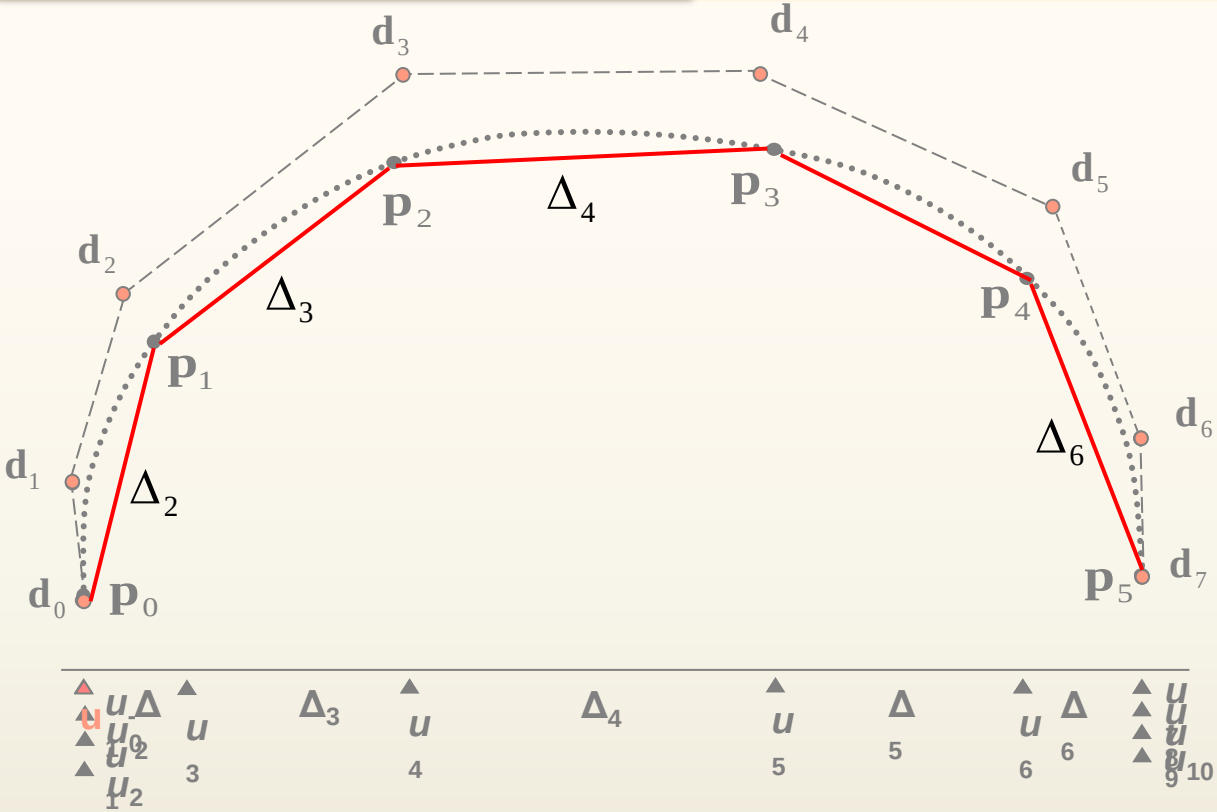
일반적으로 **Knot**와 **Control point**의 개수

number of Knot = NumOfData + 2 X Degree

number of Control Point = NumOfData + 2

B-spline 곡선 프로그래밍 가이드 (7)

(3) Knot 간격 설정
(두 점 사이의 거리를 기준으로 함)



$$u_{-1} = u_0 = u_1 = u_2 = 0$$

$$u_3 = \Delta_2$$

$$u_4 = u_3 + \Delta_3$$

$$u_5 = u_4 + \Delta_4 = \Delta_2 + \Delta_3 + \Delta_4$$

$$u_6 = u_5 + \Delta_5 = \Delta_2 + \Delta_3 + \Delta_4 + \Delta_5$$

$$u_7 = u_6 + \Delta_6 = \Delta_2 + \Delta_3 + \Delta_4 + \Delta_5 + \Delta_6$$

$$u_7 = u_8 = u_9 = u_{10}$$

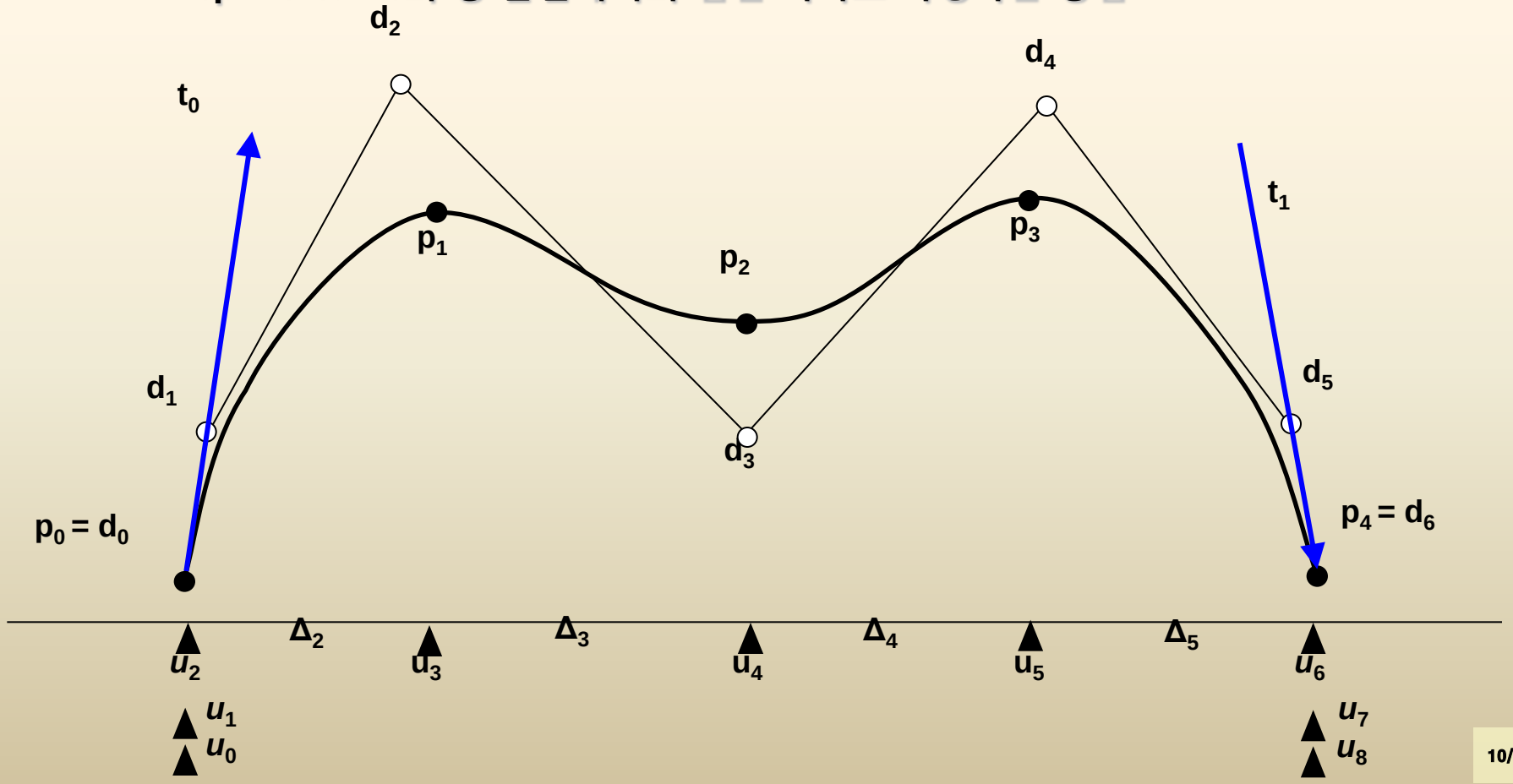
※ void calc_Knot(Vector* InputPoint) 함수

원래는 **Arc length**를 사용해야 하지만, 미리 알 수 없기 때문에,
곡선 상의 점간의 길이비를 이용하는 **Chord length parametrization** 방법을 사용

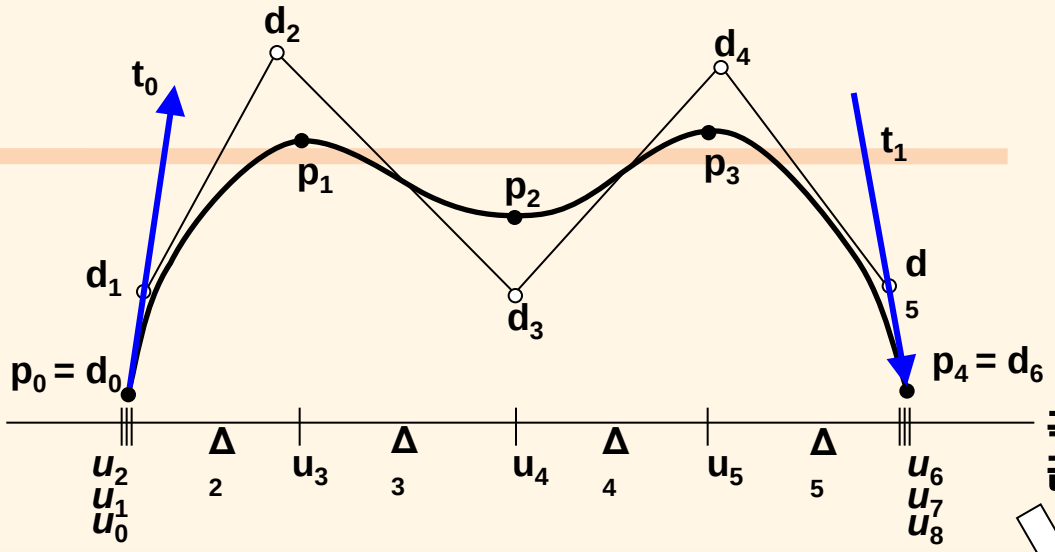
B-spline 곡선 프로그래밍 가이드 (8)

(4) Bessel End Condition으로 법선 벡터구함

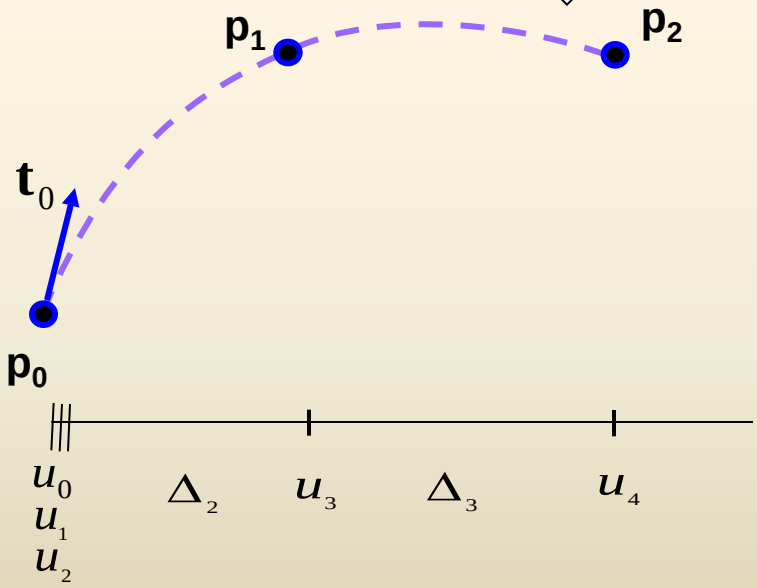
- B-spline curve interpolation에서 양끝점에서의 접선벡터 t_0, t_1이 주어지지 않았을 때,
  (1) 곡선의 양끝의 연속된 세 점으로부터 2차 곡선(quadratic curve)을 생성하고,
  (2) 생성된 2차 곡선의 양 끝점에서의 1차 미분값을 우리가 생성하고자 하는 B-spline curve의 양 끝점에서의 접선 벡터로 가정하는 방법



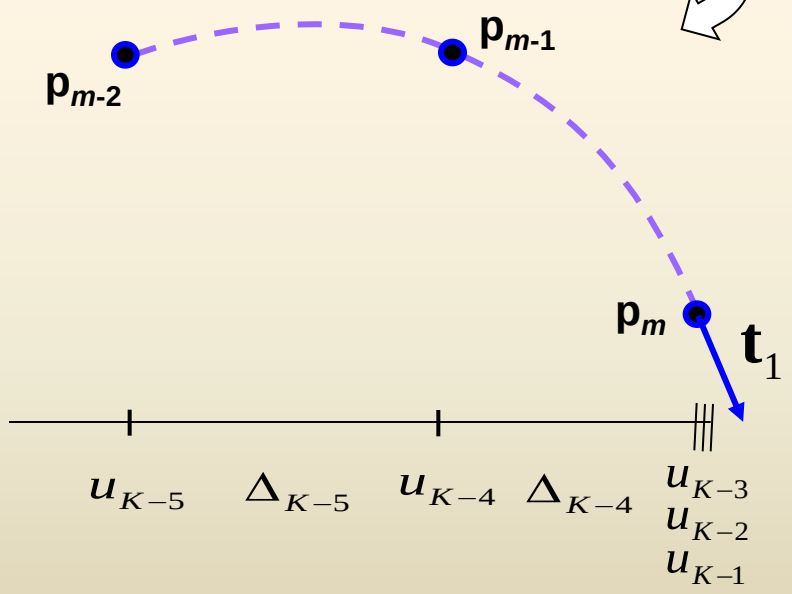
(4) Bessel End Condition으로 법선 벡터구함



곡선의 시작 부분



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



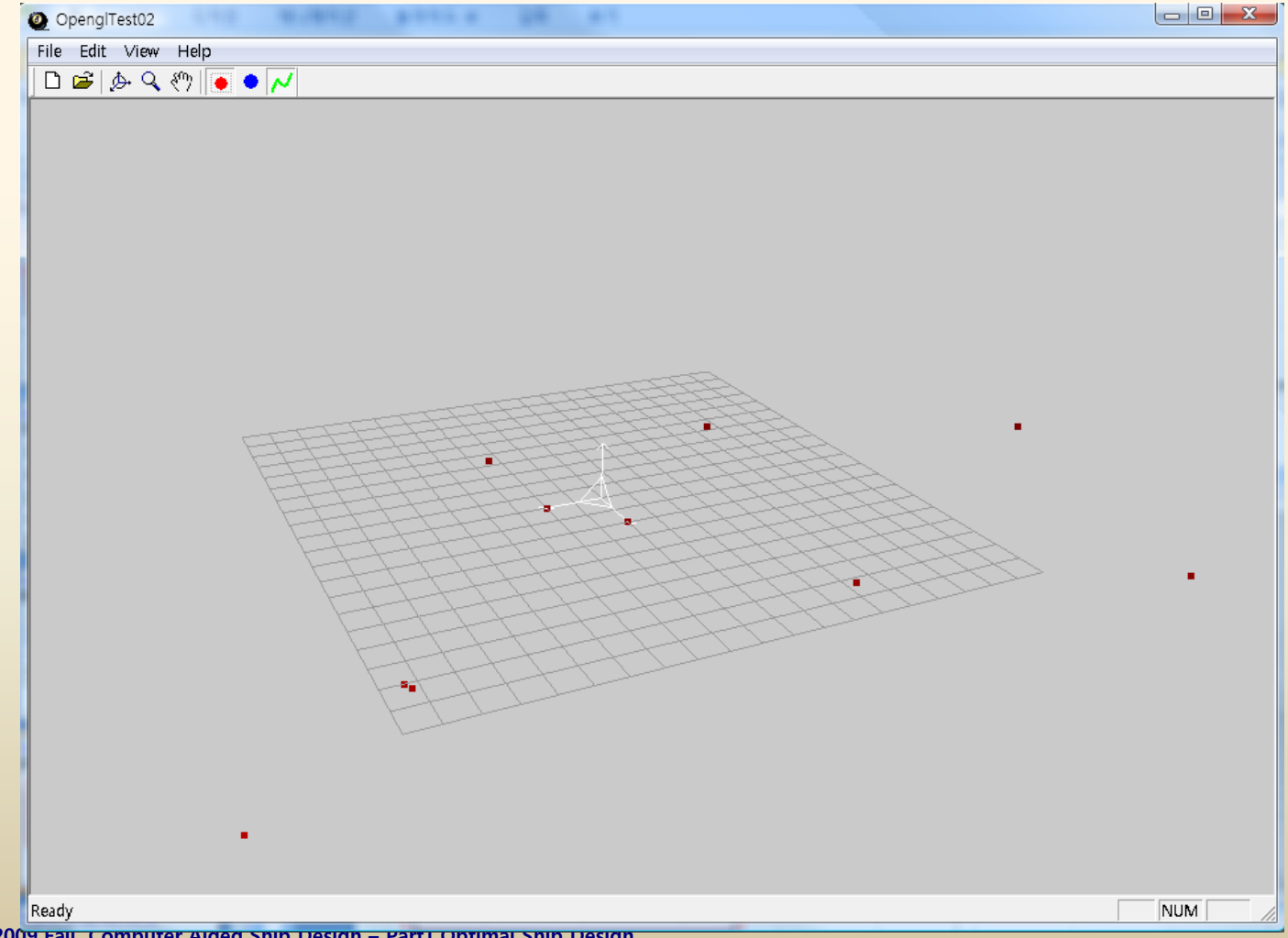
$$\mathbf{t}_e = \left(\frac{\Delta_{K-4}}{\Delta_{K-5}(\Delta_{K-5}+\Delta_{K-4})}\mathbf{p}_{m-2} - \frac{(\Delta_{K-5}+\Delta_{K-4})}{\Delta_{K-5}\Delta_{K-4}}\mathbf{p}_{m-1} + \frac{(2\Delta_{K-4}+\Delta_{K-5})}{(\Delta_{K-5}+\Delta_{K-4})\Delta_{K-4}}\mathbf{p}_m\right)$$

B-spline 곡선 프로그래밍 가이드 (9)

- 주의 사항
  - Knot 간격은 점사이의 거리로 가정하고 계산
  - Knot 값이 모두 구해진 뒤, 범위가 [0,1]이 되도록 나눠줌
  - Bessel End Condition을 사용할 경우, 시작점과 끝점에 서의 기울기를 Knot 간격으로 나눠줄 필요 없음

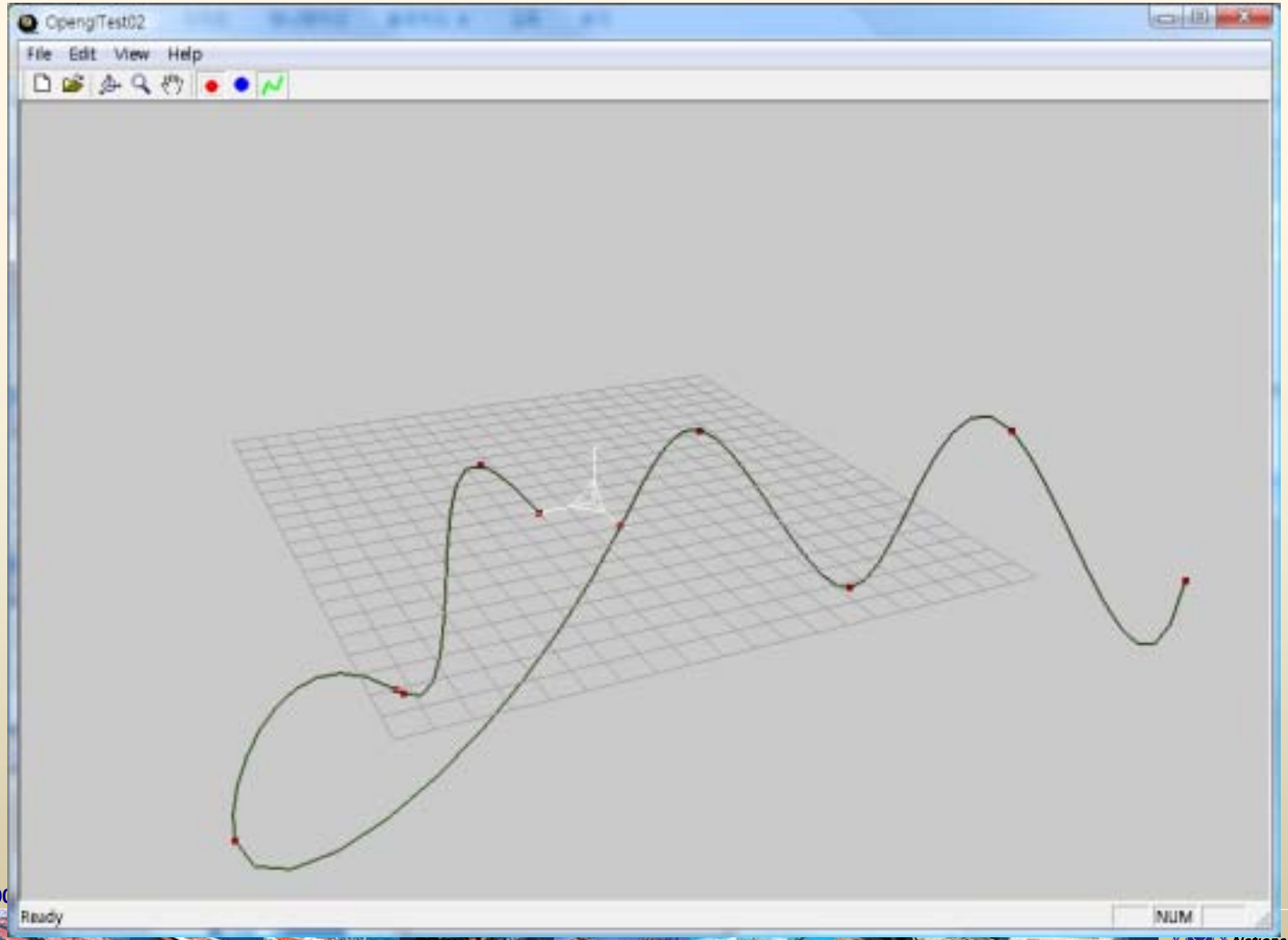
B-spline 곡선 프로그래밍 가이드 (10)

(6) 파라미터 값**(u)**를 변경시켜가며 곡선 위의 점을 구함



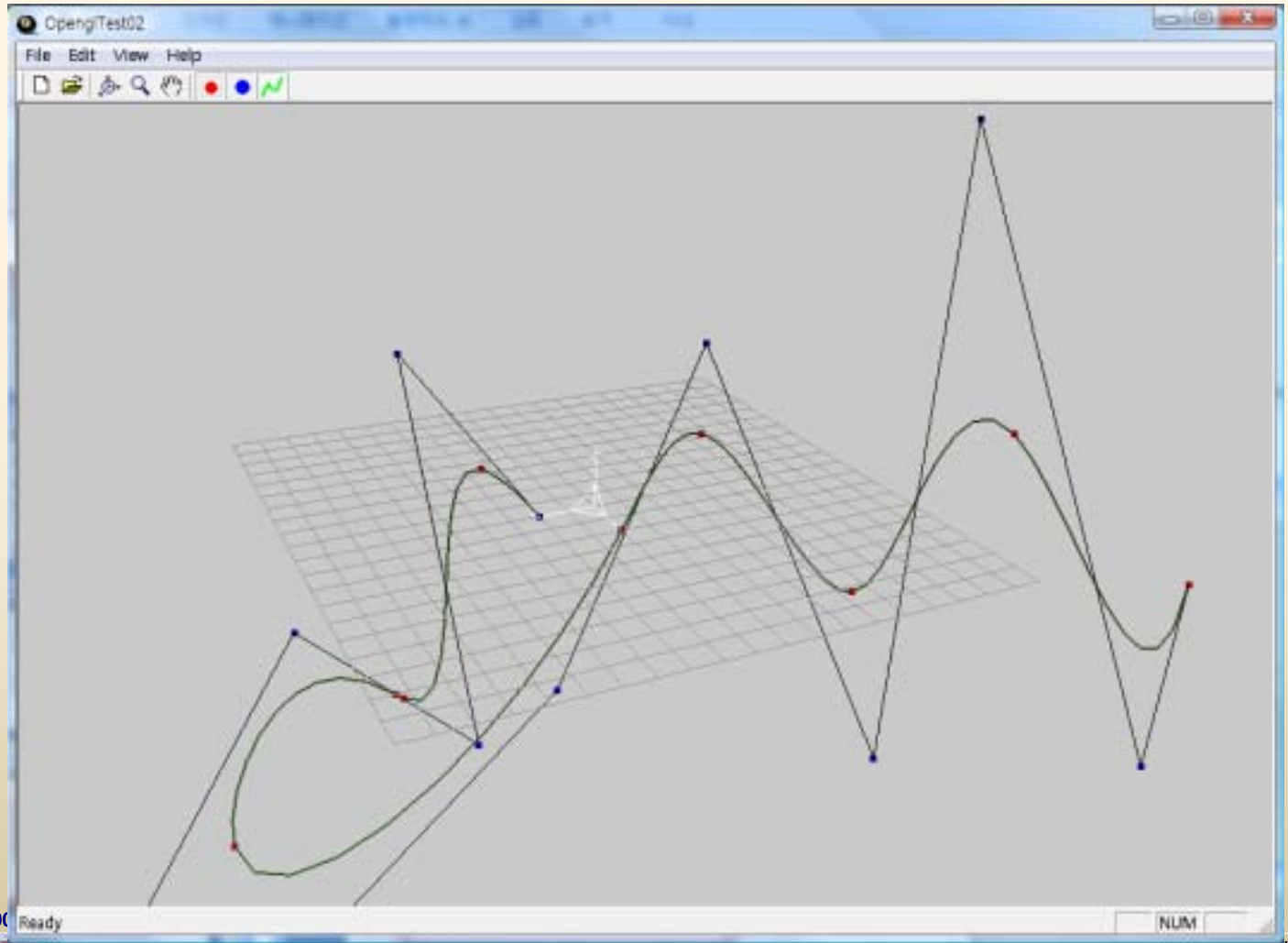
B-spline 곡선 프로그래밍 가이드 (11)

(6) 파라미터 값**(u)**를 변경시켜가며 곡선 위의 점을 구함



B-spline 곡선 프로그래밍 가이드 (12)

(6) 파라미터 값**(u)**를 변경시켜가며 곡선 위의 점을 구함



B-spline 곡선 프로그래밍 가이드 (13)

- Toolbar 및 메뉴를 사용하여 조정점 또는 주어진 점의 좌표를 변경하는 Dialog Box 작성

- 기타 구현 (자율적으로 반영 가능한 내용을 구현)시 가산점은 크지 않음