

컴퓨터 그래픽스

OpenGL 곡선 곡면

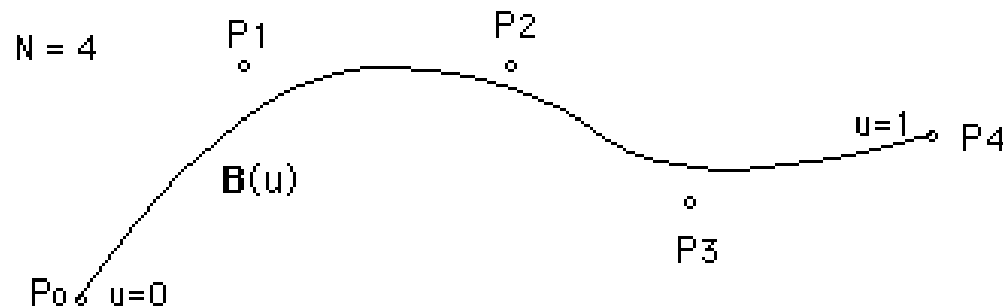
2016년 2학기

3. OpenGL 곡선 곡면 내용

- 곡선 곡면
 - 베지에 곡선
 - 베지에 곡면

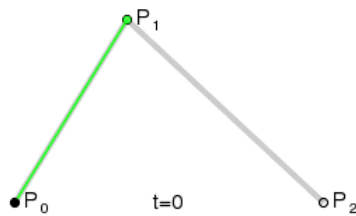
베지에 곡선

- 다항식으로 표현되는 근사곡선
 - CAD에서 많이 사용되는 커브
 - 주어진 제어점의 위치에 의해 곡선의 형태가 결정되는 근사곡선
 - 어떤 숫자의 제어점에도 베지에 커브는 적용될 수 있다.
 - 제어점의 수는 베지에 다항식의 차수를 결정
 - 2개의 제어점: 두 점 사이의 선분
 - 3개의 제어점: 2차원 곡선
 - 4개의 제어점: 3차원 곡선

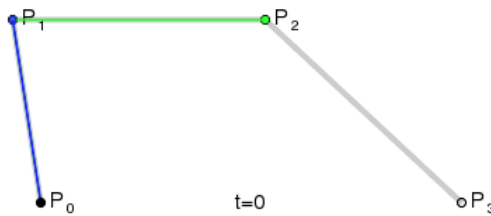


베지에 곡선

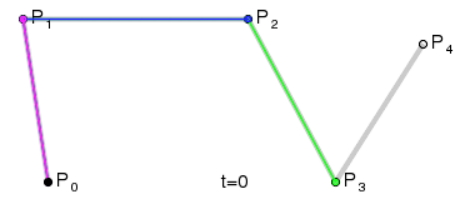
- 제어점 중 처음과 끝 점을 빼고는 보통 어느 조절점과도 만나지 않는다.



2차원 커브



3차원 커브



4차원 커브

- 곡선은, 항상 제어점으로 이루어진 다각형 안에 들어가고, 곡선이 조절점들 사이에서 크게 벗어나지 않는다.
- 배합함수(Blending Function)
 - 어느 한 점에서 각 제어점이 미치는 영향
 - 제어점의 차수보다 하나 작은 다항식

베지에 곡선

- 베지에 곡선

- 매개변수 u , v 를 사용하여 곡선이나 곡면을 그린다.

- 제어점을 지정
 - 매개변수인 u 와 v 의 범위를 결정
 - 좌표값 계산 함수를 실행

- 제어점 설정

- `void glMap1f (GLenum target, GLfloat u1, GLfloat u2, GLint stride, GLint order, const GLfloat *points);`
 - Target: GL_MAP1_VERTEX_3
 - u1, u2: 각각 매개변수 u 의 하한값과 상한값
 - stride: *포인트 데이터 구조내에 있는 제어점 배열 내에서 제어점 간의 데이터 개수
 - order: 전체 제어점의 수
 - Points: 제어점 배열을 가리키는 포인터

- 곡선 그리기 상태 활성화

- `glEnable (GL_MAP1_VERTEX_3);`

베지에 곡선

- 1차원 혹은 2차원 상에서 해당되는 점의 좌표 값을 계산
 - void **glEvalCoord1f** (GLfloat u);
 - void **glEvalCoord1d** (GLdouble u);
 - void **glEvalCoord2f** (GLfloat u, GLfloat v);
 - void **glEvalCoord2d** (GLdouble u, GLdouble v);
 - 곡선 상에서 해당되는 점의 좌표값을 계산한다.
 - 내부적으로 glVertex3f 함수를 호출한다.
 - u: 2차원 맵의 점의 좌표값 계산
 - v: 3차원 맵의 점의 좌표값 계산

베지에 곡선

- 사용 예)

// 컨트롤 포인트 설정

```
GLfloat ctrlpoints[4][3] = {  
    {-40.0, -40.0, 0.0}, {-20.0, 40.0, 0.0}, {20.0, 20.0, 0.0} , {40.0, 0.0, 0.0}};
```

// 곡선 제어점 설정: 매개변수 u 의 최소값은 0, 최대값은 1,
// 제어점간의 데이터 개수는 3, 제어점은 4개를 사용

```
glMap1f (GL_MAP1_VERTEX_3, 0.0, 1.0, 3, 4, &ctrlpoints[0][0]);
```

```
glEnable (GL_MAP1_VERTEX_3);
```

// 제어점 사이의 곡선위의 점들을 계산한다. 제어점 사이를 30개로 나누어 그 점들을 연결한다. ➔ 곡선위의 점 계산

```
glBegin (GL_LINE_STRIP);  
    for (i = 0; i <= 30; i++)  
        glEvalCoord1f ((GLfloat)i / 30.0);
```

```
glEnd ();
```

```
glDisable (GL_MAP1_VERTEX_3);
```

// 제어점에 점을 그린다.

```
glPointSize (5.0);  
glColor3f (0.0, 0.0, 1.0);  
glBegin(GL_POINTS);  
    for ( i = 0; i < 4; i++)  
        glVertex3fv (&ctrlpoints[i][0]);
```

```
glEnd ();
```

베지에 곡선

- 곡선위의 점 계산을 아래의 방법으로도 할 수 있다.

- 맵핑 격자 정의

- **glMapGrid1f** (GLint un, GLfloat u1, GLfloat u2);

- un: u 방향의 격자 분할 수 지정

- u1, u2: u 방향의 격자 영역 최소치와 최고치 지정

- 선으로 구성된 격자 계산

- **glEvalMesh1** (GLenum mode, GLint i1, GLint i2);

- mode: 매쉬 모드 설정 (GL_POINT / GL_LINE / GL_FILL)

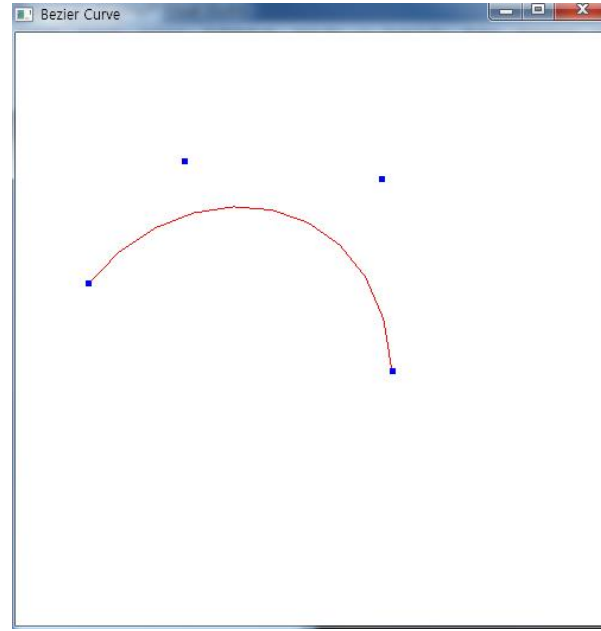
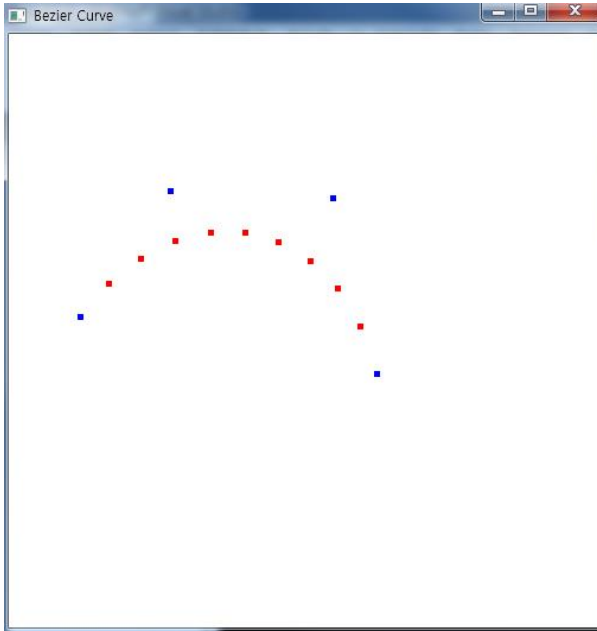
- i1, i2: u 값의 최소치와 최고치 지정

- 앞 페이지의 예제에서 곡선위의 점 계산 부분을 아래의 코드로 전환 가능

- ```
glMapGrid1f (100.0, 0, 1.0); // 매개변수 0~1 사이를 100개로 나눔
```

- ```
glEvalMesh1 (GL_LINE, 0, 100); // 선분으로 나눈 부분 0~100까지 선으로 그림
```


베지에 곡선



베지에 곡면

- 베지에 곡선에 매개변수 하나 (v) 더 추가한다.
 - 제어점 설정 함수
 - **glMap2f** (GLenum target, GLfloat u1, GLfloat u2, GLint uStride, GLint uOrder, GLfloat v1, GLfloat v2, GLint vStride, GLint vOrder, GLfloat *points);
 - target: GL_MAP2_VERTEX_3
 - u1, u2: 매개변수 u 의 하한값과 상한값
 - uStride: *포인터 데이터 구조 내에 있는 제어점 배열 내에서 제어점 간의 데이터 개수
 - uOrder: u 방향의 전체 제어점의 수
 - v1, v2: 매개변수 v 의 하한값과 상한값
 - vStride: 제어점 배열 내에서 제어점 간의 데이터 개수
 - vOrder: v 방향의 전체 제어점의 수
 - Points: 제어점 배열을 가리키는 포인터

배지에 곡면

- 곡면 그리기 활성화
 - glEnable (GL_MAP2_VERTEX_3);
- 2D 맵핑 격자 정의
 - glMapGrid2f (GLint un, GLfloat u1, GLfloat u2, GLint vn, GLfloat v1, GLfloat v2);
 - un, vn: u나 v 방향의 격자 분할 수 지정
 - u1, u2: u 방향의 격자 영역 최소치와 최고치 지정
 - v1, v2: v 방향의 격자 영역 최소치와 최고치 지정
- 선으로 구성된 2D 격자 계산
 - glEvalMesh2 (GLenum mode, GLint i1, GLint i2, GLint j1, GLint j2);
 - mode: 매쉬 모드 설정 (GL_LINE)
 - i1, i2: u 값의 최소치와 최고치 지정
 - j1, j2: v 값의 최소치와 최고치 지정

베지에 곡면

- 사용 예)

// 3차원 상의 제어점 설정

```
GLfloat ctrlpoints[3][3][3] = {{{-4.0, 0.0, 4.0},{-2.0, 4.0, 4.0},{4.0, 0.0, 4.0}},  
                                {{-4.0, 0.0, 0.0}, {-2.0, 4.0, 0.0}, {4.0, 0.0, 0.0}},  
                                {{-4.0, 0.0, -4.0}, {-2.0, 4.0, -4.0}, {4.0, 0.0, -4.0}}};
```

// 곡면 제어점 설정

```
glMap2f (GL_MAP2_VERTEX_3, 0.0, 1.0, 3, 3, 0.0, 1.0, 9, 3, &ctrlpoints[0][0][0]);  
glEnable (GL_MAP2_VERTEX_3);
```

// 그리드를 이용한 곡면 드로잉

```
glMapGrid2f (10, 0.0, 1.0, 10, 0.0, 1.0);
```

// 선을 이용하여 그리드 연결

```
glEvalMesh2 (GL_LINE, 0, 10, 0, 10);
```

```
glPointSize (2.0); glColor3f (0.0, 0.0, 1.0);
```

```
glBegin(GL_POINTS);
```

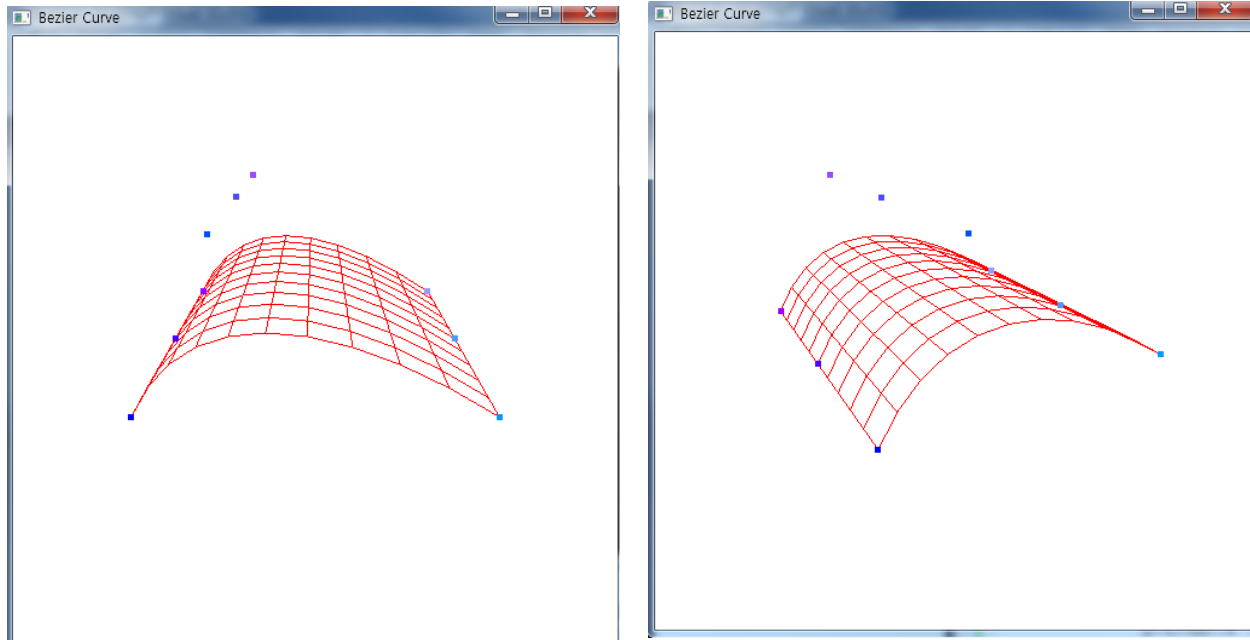
```
    for ( i = 0; i < 3; i++ )
```

```
        for ( j = 0; j < 3; j++ )
```

```
            glVertex3fv (ctrlpoints[i][j]);
```

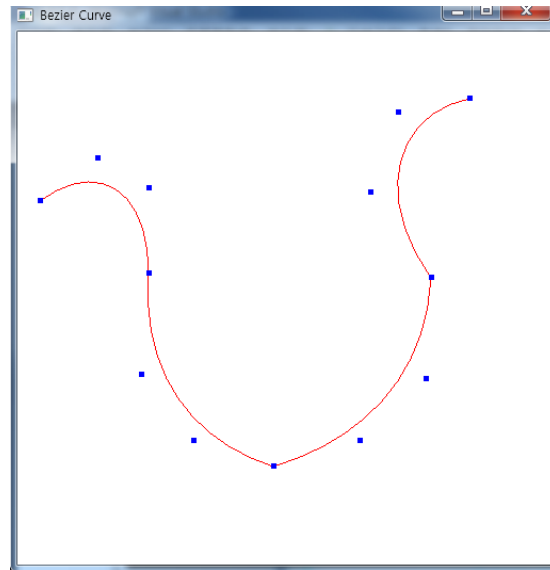
```
glEnd ();
```

베지에 곡면



실습 25

- 마우스를 찍어서 2차원 화면에 곡선을 그린다.
 - 3차 곡선을 그리도록 한다. (제어점 4개씩 사용)
 - 화면은 glOrtho 를 사용하여 2차원 평면으로 설정
 - r/R: 새로운 곡선을 그릴 수 있도록 한다.



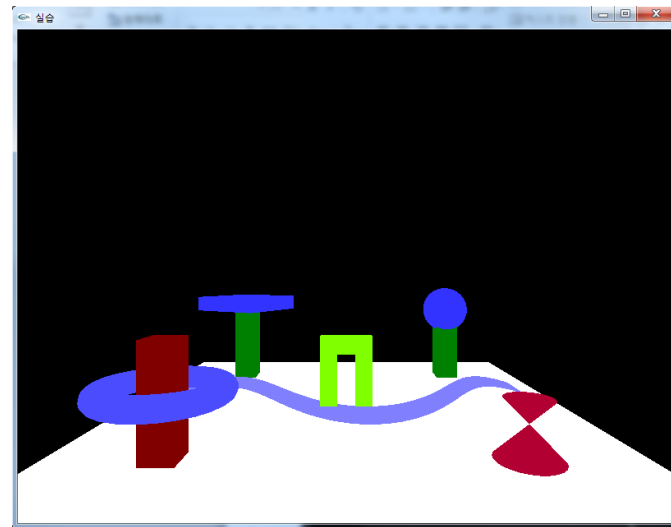
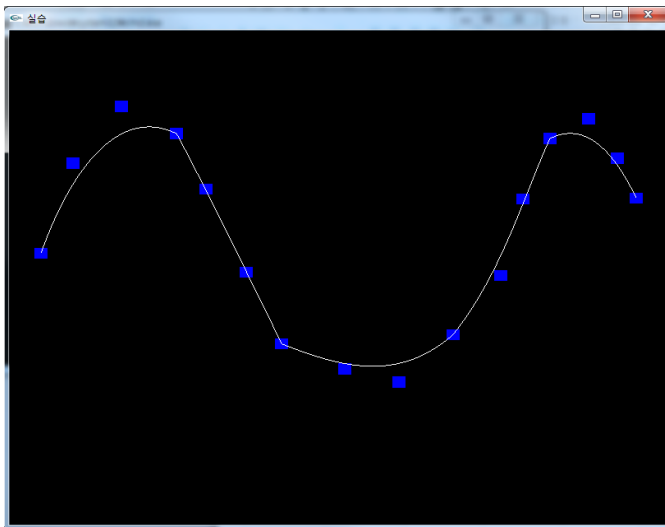
실습 26

- 실습 19 (또는 20) 에 바닥 곡면 넣기
 - 실습 19에서 바닥을 평면 대신 곡면을 사용 해 본다.
 - 꼭지점을 명시하여 곡면을 그린다.
 - 부분으로 나누어 그린다.
 - 키보드 명령어로 곡면의 제어점을 움직인다.
 - 곡면이 물결처럼 위, 아래로 움직이도록 제어점을 움직인다.

실습 27

- 실습 19에 곡면으로 길 넣기

- 실습 19에서 길을 곡선을 사용하여 그린다.
- 2차원 화면에서 25번 실습같이 곡선을 그리면 곡선이 반복되어 길 형태로 만든다.
- 곡선의 길을 19번 실습에서 바닥에 추가하여 그린다.
 - 2차원 화면과 3차원 화면을 전환할 수 있어야 한다.
- 카메라 이동/회전은 기본으로 추가한다.
 - Z축 이동
 - Y축 회전



실습 28
