2023학년도 1학기

컴퓨터애니메이션실습

컴퓨터 애니메이션

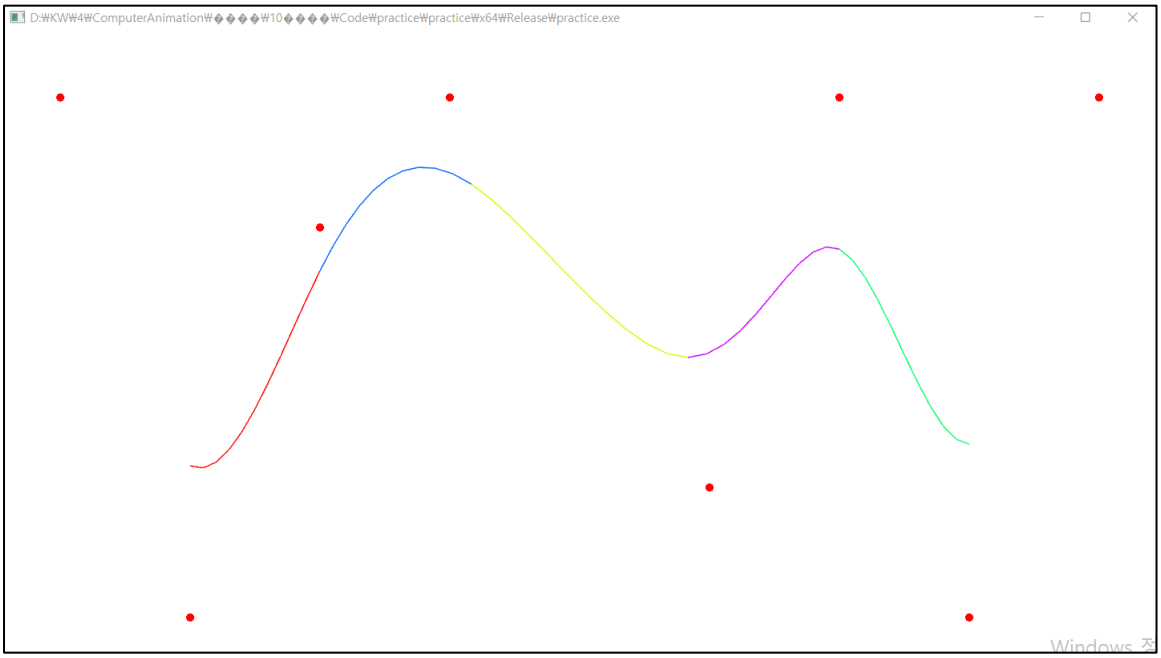
실습 보고서

Self-Scoring Table

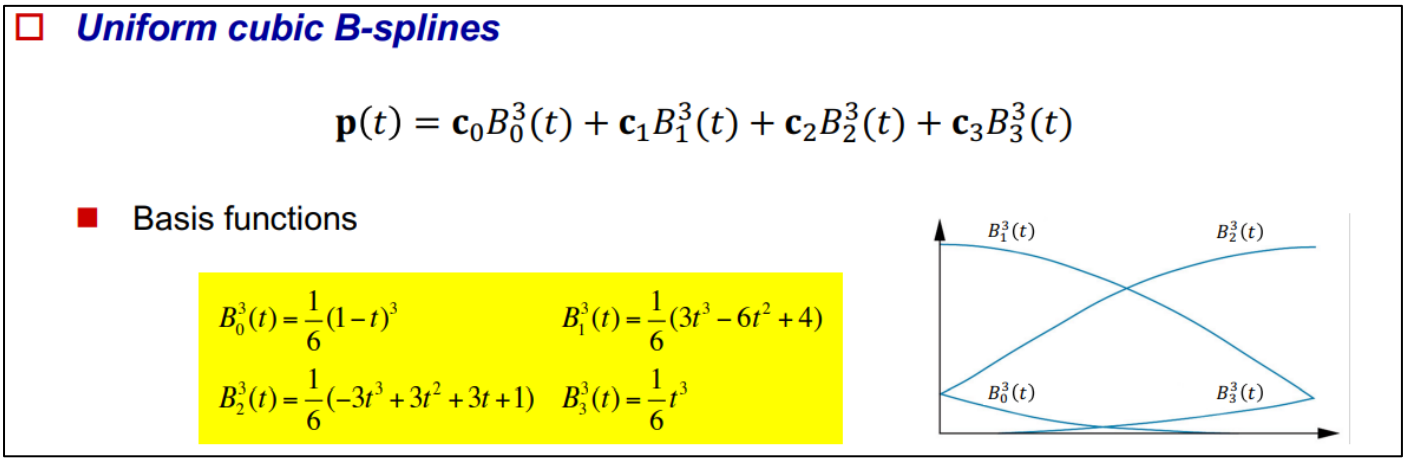
	P1	P2	P3	E1	E2
Score	1	1		1	1

정보융합학부 2018204058 김민교

P1 - Computing/drawing a uniform cubic B-spline curve



B-spline curve를 그렸다.



Uniform cubic B-splines의 식이다.

```
133   // Compute the point on the B-spline
134   Vector3f pointOnBspline(const Vector3f b[4],float t1)  // t의 1승
135   {
136       float t2 = t1 * t1; // t의 2승
137       float t3 = t2 * t1; // t의 3승
138
139       float B0 = 1 - 3 * t1 + 3 * t2 - t3;
140       float B1 = 4 - 6 * t2 + 3 * t3;
141       float B2 = 1 + 3 * t1 + 3 * t2 - 3 * t3;
142       float B3 = t3;
143
144       return (b[0] * B0 + b[1] * B1 + b[2] * B2 + b[3] * B3 ) / 6;
145   }
```

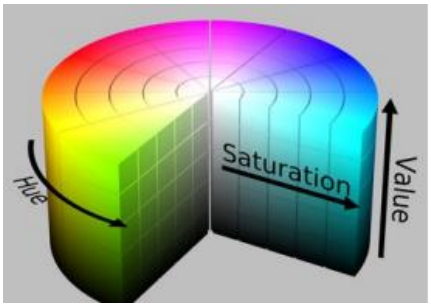
식을 코드로 구현한 것이다.

```
167         for (int j = 0; j < 4; j++)
168             b[j] = controlPoints[i + j];
169 
170         glBegin(GL_LINE_STRIP);
171         for (int j = 0; j <= N_SUB_SEGMENTS; j++)
172         {
173             float t = (float)j / N_SUB_SEGMENTS;
174             Vector3f pt = pointOnBspline(b, t);
175 
176             glVertex3fv(pt.data());
177         }
178         glEnd();
179     }
```

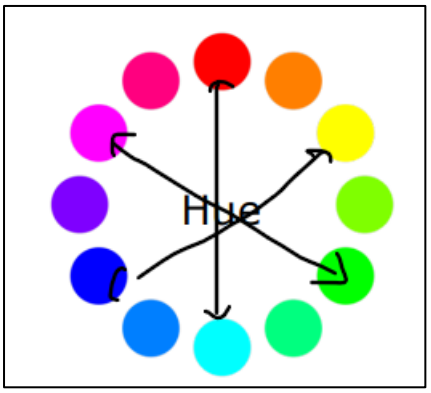
b에 총 4개의 control Point가 들어간다. i번째 control point부터 그 다음 뒤 3개의 point가 들어간다. 4개의 control point로 Uniform cubic B-splines를 통해 곡선의 방정식을 얻는다. 그래서 t점일 때의 위치를 알아내서 곡선을 그린다.

P2 - Employing a unique color for each curve segment using the HSV color space

HSV color model은 Hue, saturation, and value로 이루어져 있다. Hue는 색상이다. saturation은 채도, value는 명도이다.



hue 값은 각도가 돌아가면서 바뀐다. 0에서 360도까지 돌아간다. 세로 축은 value로 밝고 어두운 정도를 나타내고, saturation은 채도인데, 하얀색부터 얼마나 원색으로 가는 정도인지를 나타낸다.

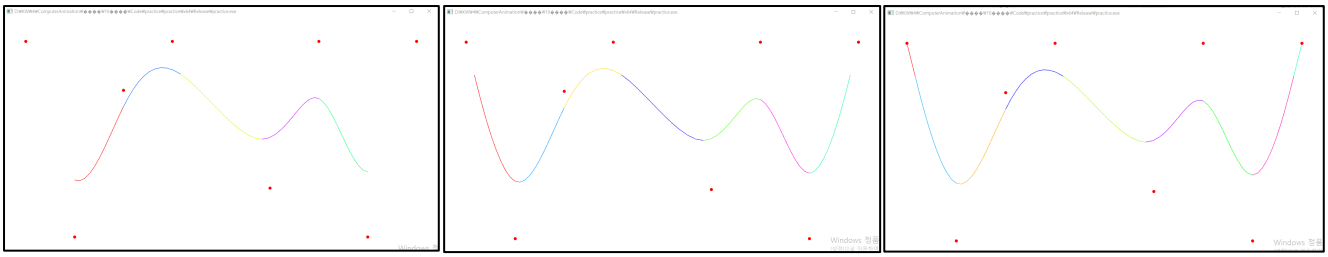


보색을 만들 때에 유용하다. 이런 식으로 정반대의 색깔이 보색이다. 각도를 통해 보색을 구할 수 있다.

```
150     // Colors
151     float hsv[3] = { 0,1,1 };// [0,360] (degree), [0,1], [0,1]
152     float rgb[3];
```

```
158         // To make an alternating complementary color
159         hsv[0] = 180.0f * i / (nControlPoints - 3) + ((i % 2) ? 180.0f : 0);
160         HSV2RGB(hsv, rgb);
161         glColor3f(rgb[0], rgb[1], rgb[2]);
```

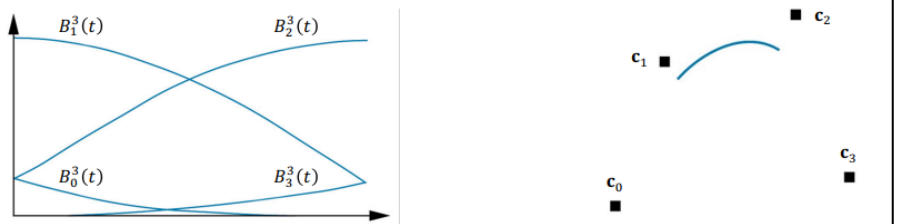
홀짝으로 홀수면 180도를 더해줘서 보색을 만들고 짝수면 그냥 냅둔다. HSV2RGB 함수로 hsv값을 rgb로 변경해서 색깔을 칠한다.



P3 - Demonstrating the endpoint interpolation

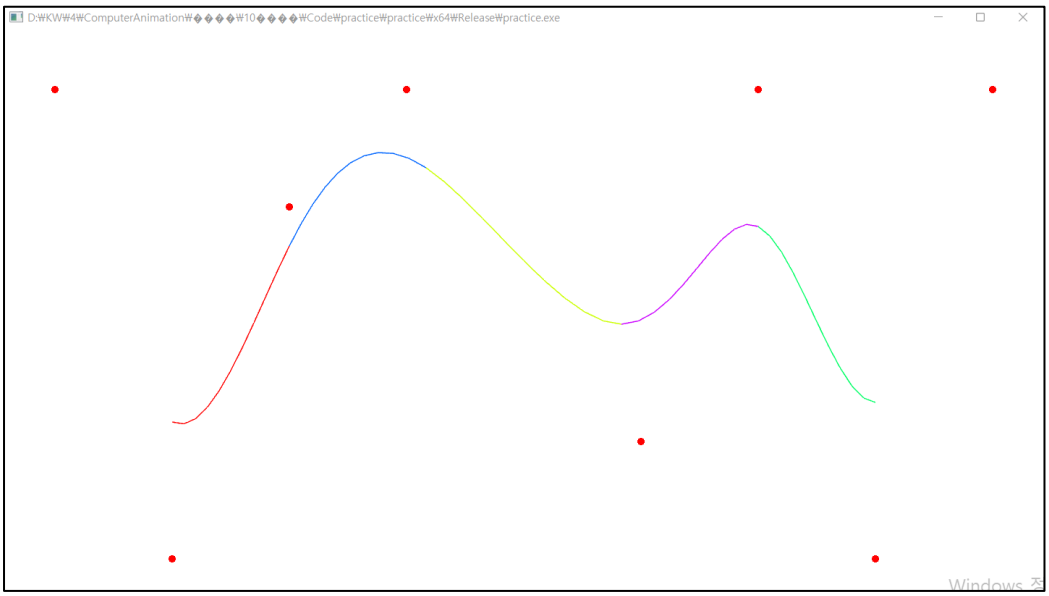
곡선이 endpoint 즉 양 끝점을 지나가게 어떻게 할까? endpoint에 점을 반복해서 위치시킨다.

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

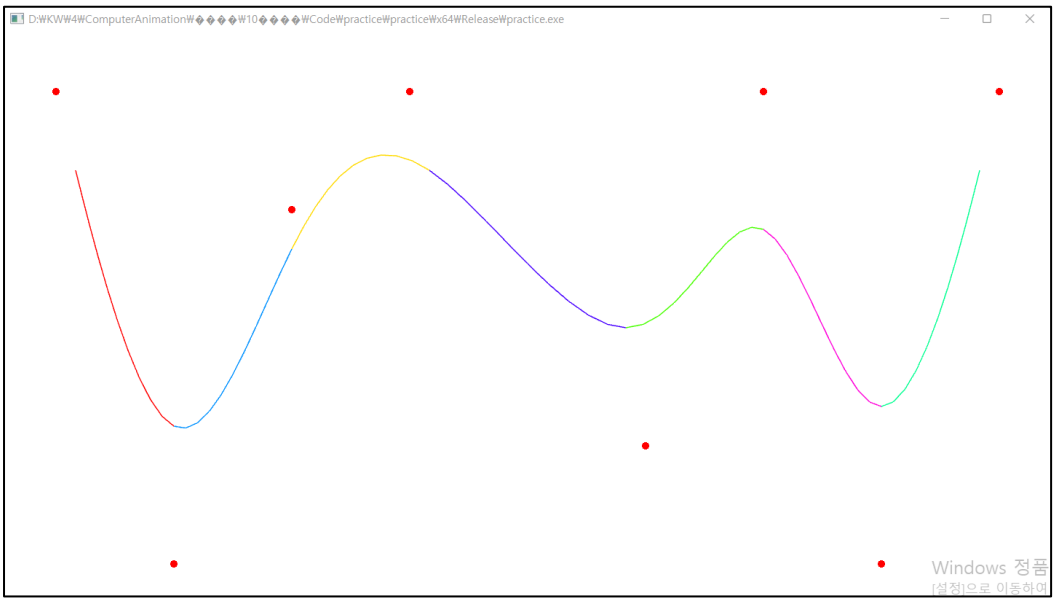


t의 범위가 [0,1]이라고 할 때, t=0일 때 c1의 영향력이 가장 크다. t=1일 때 c2의 영향력이 가장 크다. end point 점이 c0, c1, c2에 몰려있거나 c1,c2,c3에 몰려있다면 곡선은 반드시 c1이나 c2을 지나갈 것이다.

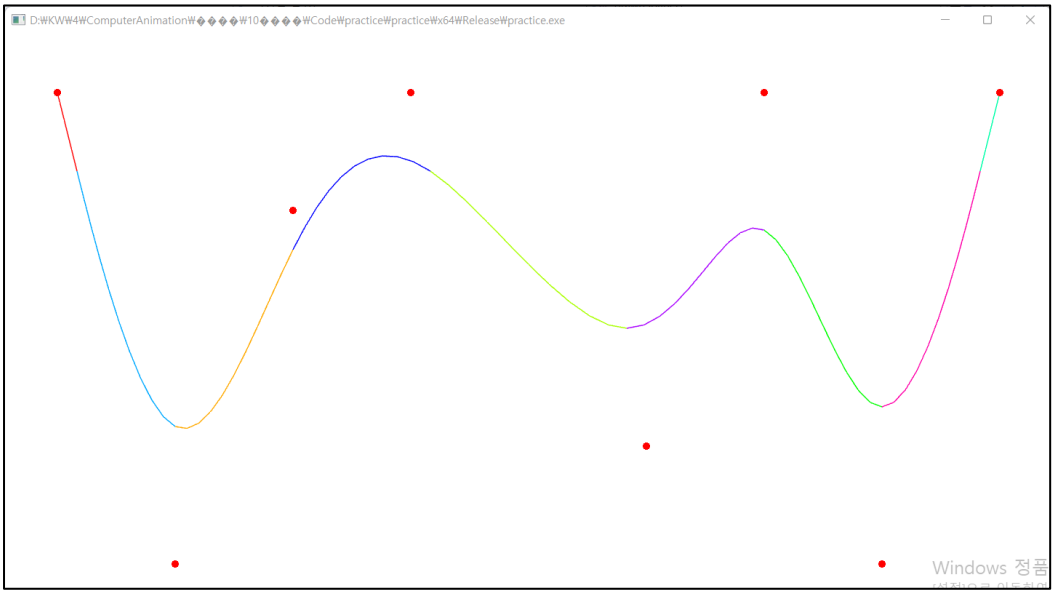
No repetition) endpoint점 반복을 안 했을 때



1 repetition) 양 끝에 똑 같은 위치의 점이 한 개 더 있게 했을 때

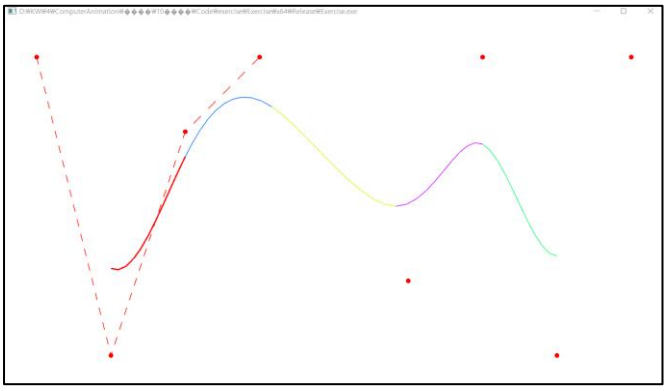


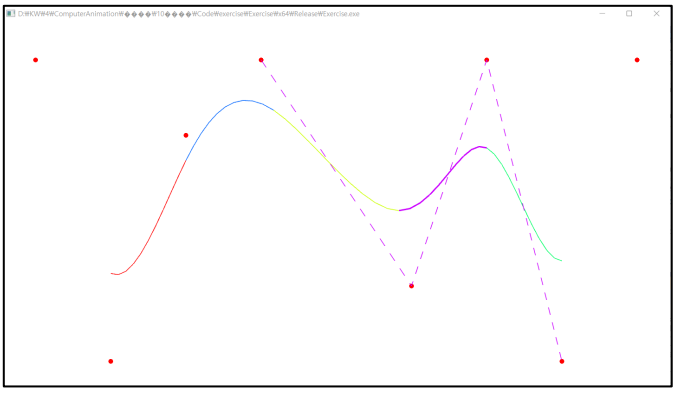
2 repetition) 양 끝에 똑 같은 위치의 점이 두 개 더 있게 했을 때

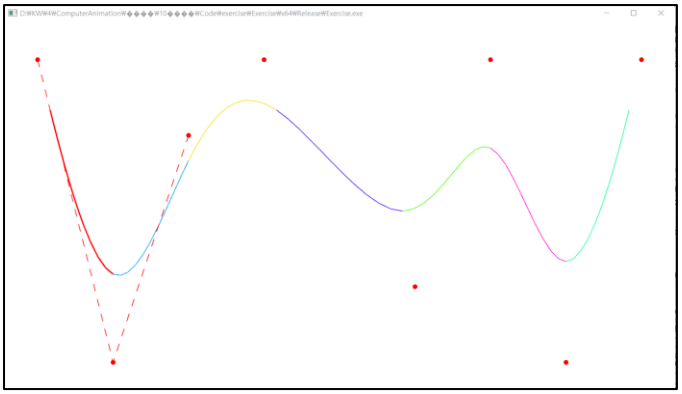


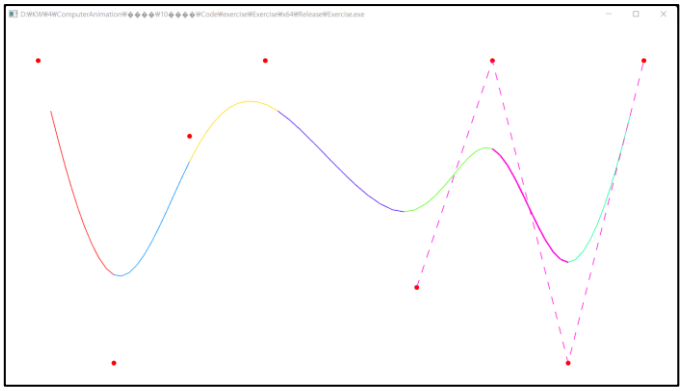
곡선이 endpoint를 지난다.

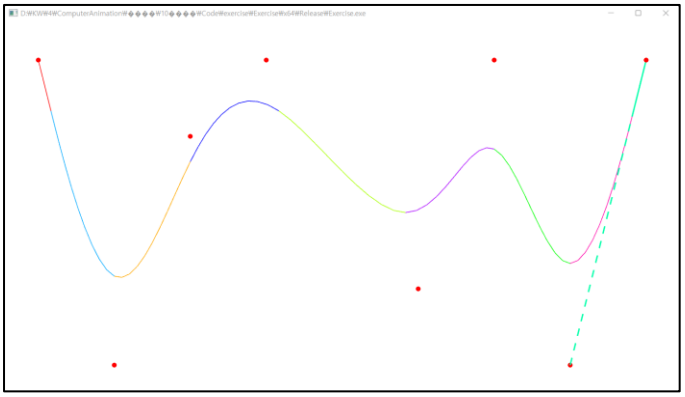
E1 - Control polygon

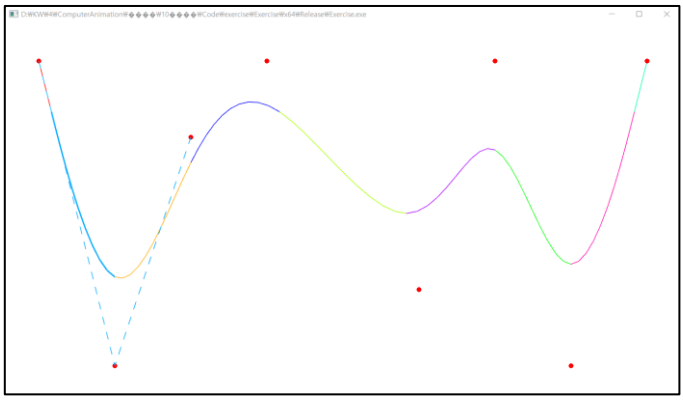










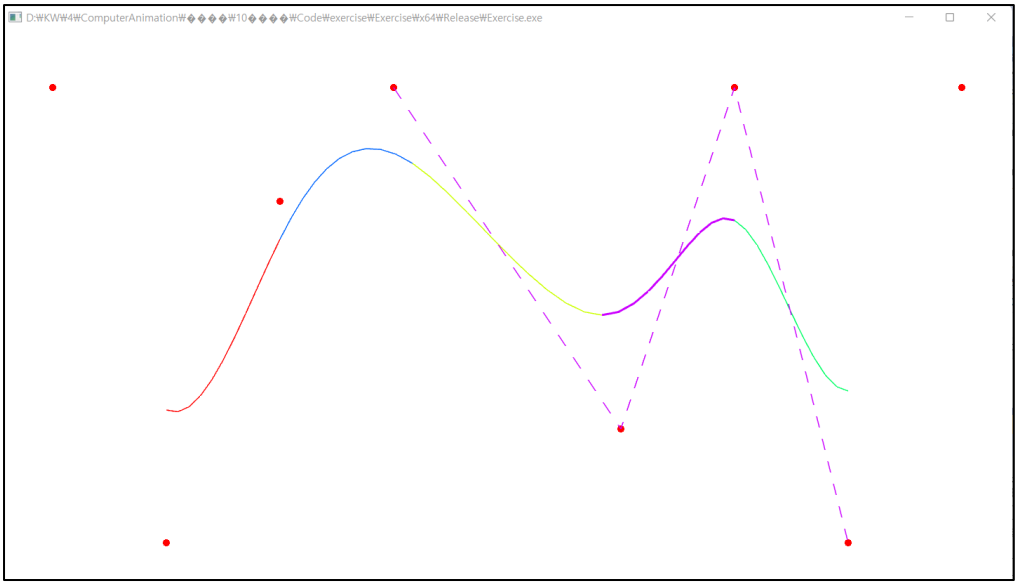


```
222  void drawControlPolygon()
223  {
224      glPointSize(5 * dpiScaling);
225      // Colors
226      float hsv[3] = { 0,1,1 };// [0,360] (degree), [0,1], [0,1]
227      float rgb[3];
228      glEnable(GL_LINE_STIPPLE);
229      hsv[0] = 180.0f * iSegment / (nControlPoints - 3) + ((iSegment % 2) ? 180.0f : 0);
230      HSV2RGB(hsv, rgb);
231      glColor3f(rgb[0], rgb[1], rgb[2]);
232
233      glLineStipple(2,0x00FF);
234      glBegin(GL_LINE_STRIP);
235      for (int j = 0; j < 4; j++)
236          glVertex3fv(controlPoints[iSegment + j].data());
237      glEnd();
238
239      glDisable(GL_LINE_STIPPLE);
240  }
241
```

line229~line231 : iSegment 값으로 현재 어떤 control point 가 c1인지 알 수 있다. 그래서 iSegment 값으로 선의 색을 결정해준다.

line 233~line 239 : iSegment부터 뒤의 Control point 3개까지 선을 그린다.

E2 - Thick curve segment




```
166         if (ctrlPolygonDrawingEnabled && i == iSegment)
167             glLineWidth(3 * dpiScaling); // ctrl polygon을 그릴 때에는 두껍게
168         else  glLineWidth(1.5f * dpiScaling);
```

지금 그리는 곡선이 iSegment와 같고, control Polygon을 그릴 것이라면 선의 굵기를 더 두껍게 해준다.