**Experiment No. 07**

**Bezier Curve**

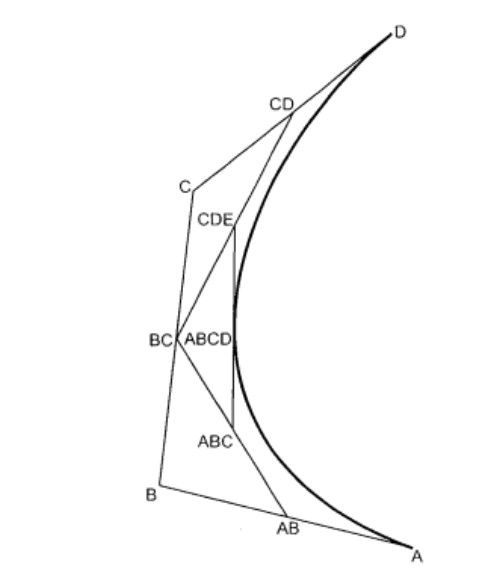
**Aim:** To implement Bezier curve for n control points. (Midpoint approach)

**Theory:** In midpoint approach Bezier curve can be constructed simply by taking the midpoints. In this approach midpoints of the line connecting four control points (A, B, C, D) are determined

(AB, BC, CD, DA). These midpoints are connected by line segment and their midpoints are

ABC and BCD are determined. Finally, these midpoints are connected by line segments and

its midpoint ABCD is determined as shown in the figure

****

The point ABCD on the Bezier curve divides the original curve in two sections. The original

curve gets divided in four different curves. This process can be repeated to split the curve into

smaller sections until we have sections so short that they can be replaced by straight lines.

**Procedure:**

1) Get four control points say A(xa, ya), B(xb, yb), C(xc, yc), D(xd, yd).

2) Divide the curve represented by points A, B, C, and D in two sections.

xab = (xa + xb) / 2

yab = (ya + yb) / 2

xbc = (xb + xc) / 2

ybc = (yb + yc) / 2

xcd = (xc + xd) / 2

ycd = (yc + yd) / 2

xabc = (xab + xbc) / 2

yabc = (yab + ybc) / 2

xbcd = ( xbc + xcd) / 2

ybcd = (ybc + ycd) / 2

xabcd = (xabc + xbcd) / 2

yabcd = (yabc + ybcd) / 2

3) Repeat the step 2 for section A, AB, ABC, ABCD and section ABCD, BCD, CD, D.

4) Repeat step 3 until we have sections so that they can be replaced by straight lines.

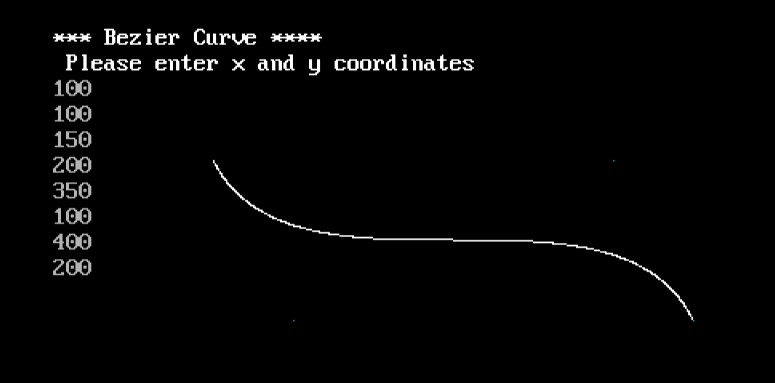
5) Repeat small sections by straight lines.

6) Stop.

**Program:**

| #include<graphics.h>  #include<math.h>  #include<conio.h>  #include<stdio.h>  void main()  {  int x[4],y[4],i;  double put\_x,put\_y,t;  int gd=DETECT,gm;  initgraph(&gd,&gm,"..\\BGI");  printf("\n\*\*\* Bezier Curve \*\*\*\*");  printf("\n Please enter x and y coordinates ");  for(i=0;i<4;i++)  {  scanf("%d%d",&x[i],&y[i]);  putpixel(x[i],y[i],3); // Control Points  }  for(t=0.0;t<=1.0;t=t+0.001) // t always lies between 0 and 1  {  put\_x = ((pow(1-t,3)\*x[0])+(3\*t\*pow(1-t,2)\*x[1])+(3\*t\*t\*(1-t)\*x[2])+(pow(t,3)\*x[3])); // Formula to draw curve  put\_y = ((pow(1-t,3)\*y[0])+(3\*t\*pow(1-t,2)\*y[1])+(3\*t\*t\*(1-t)\*y[2])+(pow(t,3)\*y[3]));  putpixel(put\_x,put\_y, WHITE); // putting pixel  }  getch();  closegraph();  } |  |
| --- | --- |

**Outcome:**



**Conclusion:**

1. Difference from arc and line - A cubic Bezier curve is expressed by polynomials of degree 3. An arc cannot be expressed by any polynomial function, the best you can do is a Taylor series, which is a limit of infinite-dimensional polynomials.

2. Importance of control point - A Bezier curve is significant with its control points. When control points are given, the Bezier curve can be written using De Casteljau's algorithm. An important property of Bezier curve is that every coordinate function is a polynomial.

3. Applications - Bezier curves have a lot of applications in the areas of science, engineering, and technology such as railway route or highway modeling, networks, animation, computer-aided design system, robotics, environment design, communications, and many other fields due to their computational simplicity and stability.