Experiment No. 8
Implement Curve: Bezier for n control points.
Name: Aryan Gaikwad
Roll Number: 09
Date of Performance:
Date of Submission:

Experiment No. 8

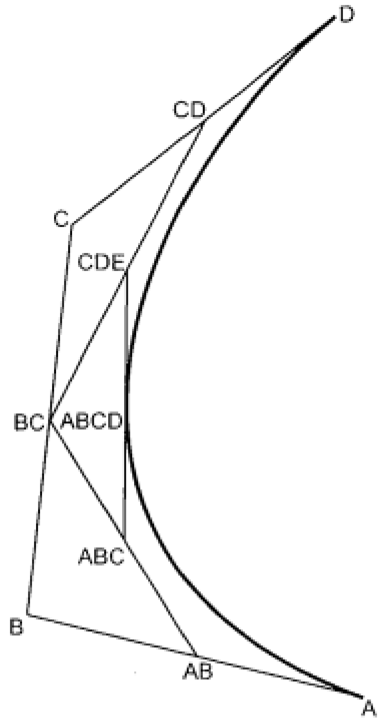
Aim: To implement Bezier curve for n control points. (Midpoint approach)

Objective:

Draw a Bezier curves and surfaces written in Bernstein basis form. The goal of interpolation is to create a smooth curve that passes through an ordered group of points. When used in this fashion, these points are called the control points.

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.

Algorithm:

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.

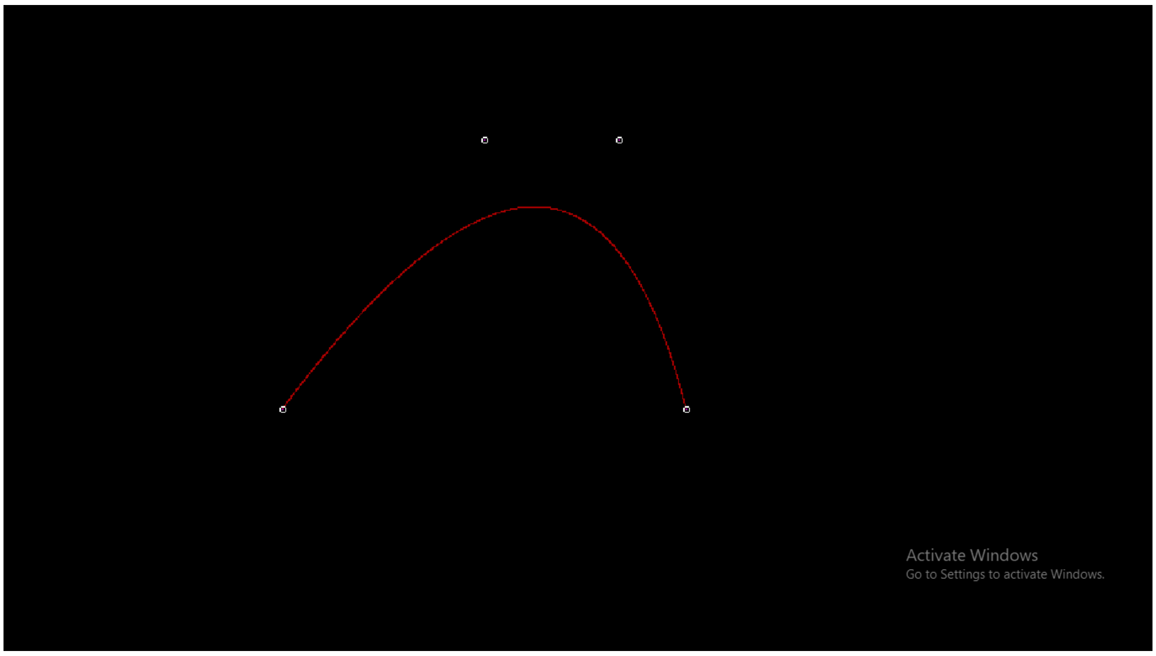
6) Stop.

Program:

```
#include<graphics.h>
#include<math.h>
int x[4],y[4];
void bezier(int x[4],int y[4])
{
int gd=DETECT,gm,i;

double t,xt,yt;
initgraph(&gd,&gm," ");
for(t=0.0;t<1.0;t+=0.0005)
{
xt=pow((1.0-t),3)*x[0]+3*t*pow((1.0-t),2)*x[1]+3*pow(t,2)*(1.0-t)*x[2]+pow(t,3)*x[3];
yt=pow((1.0)-t,3)*y[0]+3*t*pow((1.0)-t,2)*y[1]+3*pow(t,2)*(1.0-t)*y[2]+pow(t,3)*y[3];
putpixel(xt,yt,4);
delay(5);
}
for(i=0;i<4;i++)
{
putpixel(x[i],y[i],5);
circle(x[i],y[i],2);
delay(2);
}
getch();
closegraph();
}
int main()
{
int i,x[4],y[4];
printf("Enter the four control points : ");
for(i=0;i<4;i++)
{
scanf("%d %d",&x[i],&y[i]);
}
bezier(x,y);
}
```

Output:



Conclusion

– Comment on

1. Difference from arc and line- Bezier curves are parametric curves defined by control points, allowing for versatile shape definition. They offer greater flexibility than arcs, which are limited to circular or elliptical segments. Bezier curves can create diverse curve shapes, making them suitable for designing complex, custom paths in computer graphics and vector drawing. Their control points enable precise and fine-tuned curve manipulation, making them a popular choice for creating smooth and intricate curves in various applications

2. Importance of control point- Control points in a Bezier curve are pivotal in shaping and controlling the curve's trajectory. These points grant a substantial degree of control, enabling precise and customized manipulation of the curve's path. By adjusting the number and positioning of control points, you can significantly influence the curve's overall shape, direction, and curvature. They serve as the building blocks for crafting unique and tailored curve designs, making them a fundamental aspect of curve creation in computer graphics and design.

3. Applications-

1. Vector Graphics Software: Bezier curves with n control points are fundamental in vector graphic design tools like Adobe Illustrator and Inkscape for creating custom shapes, logos, and illustrations.

2. Animation: In animation software like Blender or Adobe After Effects, Bezier curves with n control points are used to define motion paths for objects, cameras, and characters, enabling smooth and customized animations.

3. Font Design: Typography designers use Bezier curves with multiple control points to create and fine-tune the shapes of characters and fonts.

4. 3D Modeling: Bezier curves can be employed in 3D modeling software (e.g., Autodesk Maya) for creating complex curves used in surfaces, splines, and character rigging.

5. Engineering and CAD: Engineers and architects use Bezier curves with n control points in computer-aided design (CAD) software to define intricate curves for objects, structures, and mechanical components.

6. Game Development: In video game development, Bezier curves are utilized for defining paths, motion trajectories, and camera movements, allowing for realistic and dynamic gameplay experiences.

7. Special Effects: Bezier curves are essential in software like Adobe After Effects and Blender for creating visual effects, such as motion blur, particle animations, and fluid simulations.

8. Robotics: In robotics, Bezier curves can be used for path planning and controlling robot movements, providing smoother and more precise trajectories.

9. Aircraft Design: Aerospace engineers use Bezier curves to define the curves of airfoil shapes, ensuring optimal aerodynamic performance.

10. Medical Imaging: In medical imaging software, Bezier curves can be employed to define complex organ shapes and contours for accurate diagnostic and treatment planning.