**Practical No 5**

**Implementation of Bezier Curve.**

**Aim: Write a program to implement a Bezier Curve Drawing Algorithm.**

**Theory:**

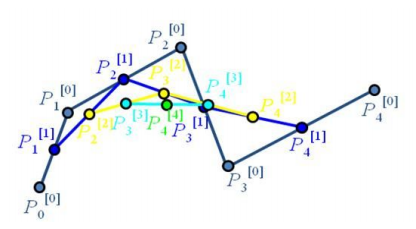
Bezier curves are parametric curves which are pretty much customizable and smooth. They are well suited for many applications. They were named after Pierre Bézier, a French mathematician and engineer who developed this method of computer drawing in the late 1960s while working for the car manufacturer Renault. People say that at the same time the same development took place during the research of Ford. There is still a confusion about who found it first. Because of my imaging background, my article will mainly focus on interpolation and curve fitting. In interpolation, what one would simply like to do is to find unknown points using known values. This way, a discrete case can be represented with a more continuous structure, and we can have a well-defined curve for missing points. The curve is initialized with certain data points, and it tries to generate new ones that are approximating (or interpolating) the old values.

**Properties of Bezier Curves:**

* They generally follow the shape of the control polygon, which consists of the segments joining the control points.
* They always pass through the first and last control points.
* They are contained in the convex hull of their defining control points.
* The degree of the polynomial defining the curve segment is one less that the number of defining polygon point. Therefore, for 4 control points, the degree of the polynomial is 3, i.e., cubic polynomial.
* A Bezier curve generally follows the shape of the defining polygon.
* The direction of the tangent vector at the end points is same as that of the vector determined by first and last segments.
* The convex hull property for a Bezier curve ensures that the polynomial smoothly follows the control points.
* No straight line intersects a Bezier curve more times than it intersects its control polygon.
* They are invariant under an affine transformation.
* Bezier curves exhibit global control means moving a control point alters the shape of the whole curve.
* A given Bezier curve can be subdivided at a point t=t0 into two Bezier segments which join together at the point corresponding to the parameter value t=t0.

**Algorithm:**

Given points Pi, i = 0,...,n, our goal is to determine a curve g (t), for all values t Î [0,1]. The idea is demonstrated below:



1. Select a value t Î [0,1]. This value remains constant for the rest of the steps.
2. Set Pi[0] (t) = Pi, for i = 0,...,n.
3. For j= 0,...,n, set Pi[j](t)=(1-t)Pi-1[j-1](t)+tPi[j-1](t) for i=j,...,n.
4. g(t)=Pn[n](t)

**Conclusion: We have implemented Bezier Curve Drawing Algorithm.**

**Code:**

#include<iostream.h>

#include<conio.h>

#include<dos.h>

#include<string.h>

#include<graphics.h>

void main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"C:/TURBOC3/BGI");

int x[4],y[4],px,py,i;

cout<<"Enter four control points of bezier curve: ";

for(i=0;i<4;i++)

{

cin>>x[i]>>y[i];

}

double t;

for(t=0.0;t<=1.0;t+=0.001) {

px=(1-t)\*(1-t)\*(1-t)\*x[0]+3\*t\*(1-t)\*(1-t)\*x[1]+3\*t\*t\*(1-t)\*x[2]+t\*t\*t\*x[3];

py=(1-t)\*(1-t)\*(1-t)\*y[0]+3\*t\*(1-t)\*(1-t)\*y[1]+3\*t\*t\*(1-t)\*y[2]+t\*t\*t\*y[3];

putpixel(px,py,WHITE);

delay(2);

}

getch();

closegraph();

}

**Output:**

