REPORT - ASSIGNMENT 1

MONSOON SEMESTER 2022

Aayush Kumar - 2020008 | CSE 333/533 | 14-09-2022

Introduction

In Computer Graphics, many concepts are widely used in various fields. Two of these concepts are “Bezier Curves” and “Shape Modeling,” on which the assignment was given. A summary of the work completed as part of the given assignment in CSE333/533 in the area of Computer Graphics is presented in this report.

QUESTION 1

In this question , our objective is to create a cylinder using its parametric equation.

Parametric equation of a cylinder is

X = a cos(t)

Y = a sin(t)

Z = h

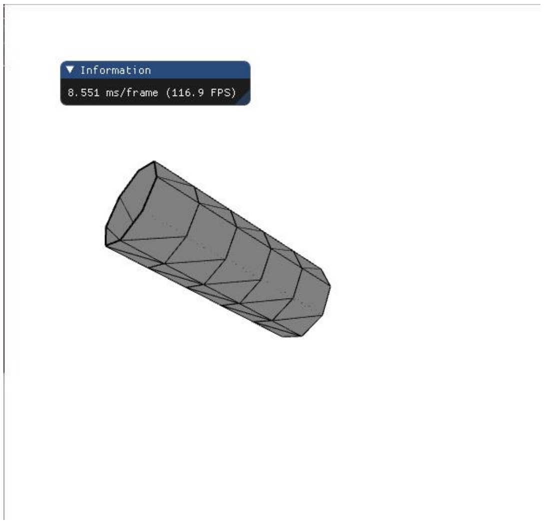
```
//Shape data
size_t nVertices = 5*2*4*2*3; // No. of vertices of the shape
GLfloat *shape_vertices = new GLfloat[nVertices*3];
float pi = M_PI;
float arr[9][3];
for(int u = 0 ; u <= 8 ; u++){
    float x = 5*cos(u*pi/(float)4.0f);
    float y = 5*sin(u*pi/(float)4.0f);
    float z = 0.0f;
    arr[u][0] = x;
    arr[u][1] = y;
    arr[u][2] = z;
}
```

Here, we create an array to store the set of points of x,y,z. Radius is constant at 5.

```
for(int j = 1; j<6 ;j++){
    for (size_t i = 0; i < 8; i++) {
        int k = ((j-1)*8*2*3*3)+(i*18);
        shape_vertices[k+9] = shape_vertices[k+0] = arr[i][0];
        shape_vertices[k+10] = shape_vertices[k+1] = arr[i][1];
        shape_vertices[k+11] = shape_vertices[k+2] = j*5;

        shape_vertices[k+3] = arr[i][0];
        shape_vertices[k+4] = arr[i][1];
        shape_vertices[k+5] = (j-1)*5;

        shape_vertices[k+12] = arr[i+1][0];
        shape_vertices[k+13] = arr[i+1][1];
        shape_vertices[k+14] = j*5;

        shape_vertices[k+15] = shape_vertices[k+6] = arr[i+1][0];
        shape_vertices[k+16] = shape_vertices[k+7] = arr[i+1][1];
        shape_vertices[k+17] = shape_vertices[k+8] = (j-1)*5;
    }
}
```

Here, we are storing all the coordinates of the vertices of the shape.

Output of the code is shown below



The code is also attached in the zip file submitted on the classroom.

QUESTION 2

To design an interpolating piecewise cubic Bezier curve, we must understand the concept of linear interpolation.

We know that, by linear interpolation, we can express any point x on a straight line such that t divides the line in the ratio t : 1-t, where t is the parameter.

For the cubic Bezier curve, the approach is pretty exact. Moreover, it will be based on the principle of repeated linear interpolation.

Here, we have degree 3 of the polynomial, and we know that degree = no. of pts -1 or no. of pts = degree + 1.

Therefore, we will have 4 control points to draw the Bezier curve.

We also know that to find the number of linear interpolations needed to draw a curve of n degree and n+1 control points, we can find by using the formula.

$$N = n(n+1)/2$$

Here, n = 3

Therefore, no. of linear interpolations required = 6.

Pseudo Code

```
Display_Bezier(P0,P1,P2,P3) {

        if(P0,P1,P2,P3 are flat or collinear){

                draw straight line P0P3;
```

else{

subdivide P0,P1,P2,P3 into L0,L1,L2,L3 and R0,R1,R2,R3

Display_Bezier(L0,L1,L2,L3);

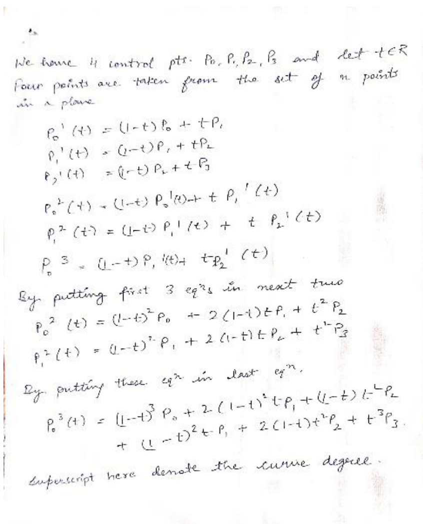
Display_Bezier(R0,R1,R2,R3);

End

End

This algorithm is also known as Recursive Subdivision display algorithm.(Also, given in the Lec 02 resources).

Derivation of the cubic curve



We have 4 control pts. P_0, P_1, P_2, P_3 and let $t \in R$
Four points are taken from the set of n points in a plane

$P_0^1(t) = (1-t)P_0 + tP_1$

$P_1^1(t) = (1-t)P_1 + tP_2$

$P_2^1(t) = (1-t)P_2 + tP_3$

$P_0^2(t) = (1-t)P_0^1(t) + t\,P_1^1(t)$

$P_1^2(t) = (1-t)P_1^1(t) + t\,P_2^1(t)$

$P_0^3 = (1-t)P_1^1(t) + tP_2^1(t)$

By putting first 3 eq^ns in next two

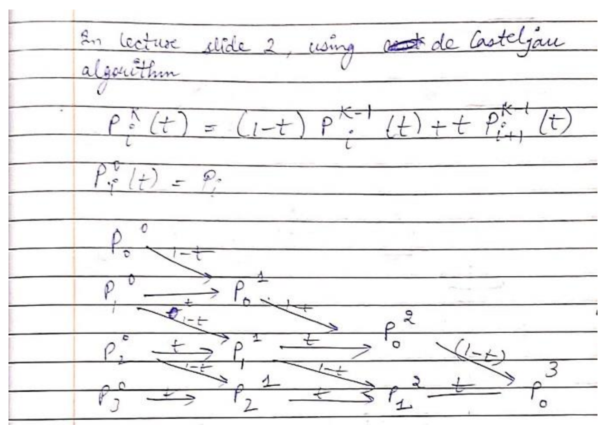
$P_0^2(t) = (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2$

$P_1^2(t) = (1-t)^2 P_1 + 2(1-t)tP_2 + t^2 P_3$

By putting these eq^n in last eq^n,

$P_0^3(t) = (1-t)^3 P_0 + 2(1-t)^2 tP_1 + (1-t)t^2 P_2 + (1-t)^2 tP_1 + 2(1-t)t^2 P_2 + t^3 P_3.$

Superscript here denote the curve degree.



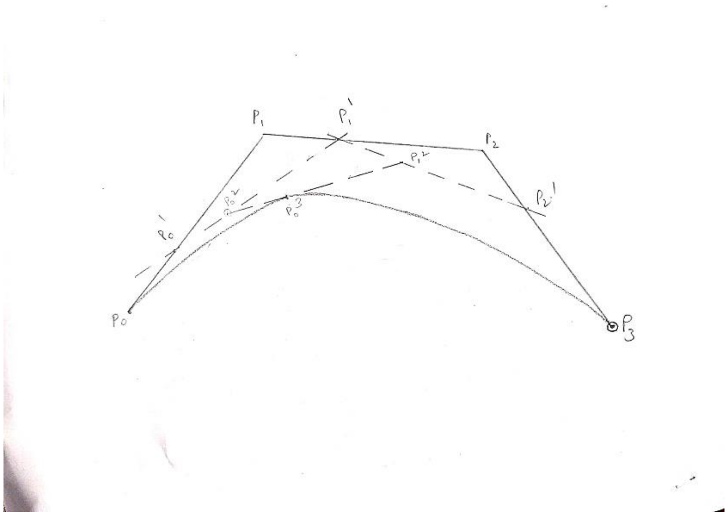
In lecture slide 2, using ~~cast~~ de Casteljau algorithm

$P_i^k(t) = (1-t)P_i^{k-1}(t) + t\,P_{i+1}^{k-1}(t)$

$P_i^0(t) = P_i$

From the equation derived, the equation is a cubic expression in t. Hence the curve obtained is a cubic Bezier curve.

Geometric Representation of the curve at t =3



Continuity

For continuity, the Bezier curves should obey the following two properties :-

- $d(P)/dt_{(At\ 0)} = d(Q)/dt_{(At\ 1)}$
- When the tangent drawn from the intersection point from both the curve, they should point in the same direction.

Now, consider 2 sets of four points such that there is a common point in both the curves.

The tangent from that common point always satisfies the direction for both the curves.

Hence, the curve will always be in C1 continuity.

SOURCES

https://ctan.math.illinois.edu/macros/latex/contrib/lapdf/bezinfo.pdf

https://www.kth.se/social/files/55492cacf276542be2fc547a/BezierCurvesAndSurfaces.pdf

https://www.cs.utexas.edu/users/fussell/courses/cs384g-fall2011/lectures/lecture16-Interpolating_curves.pdf