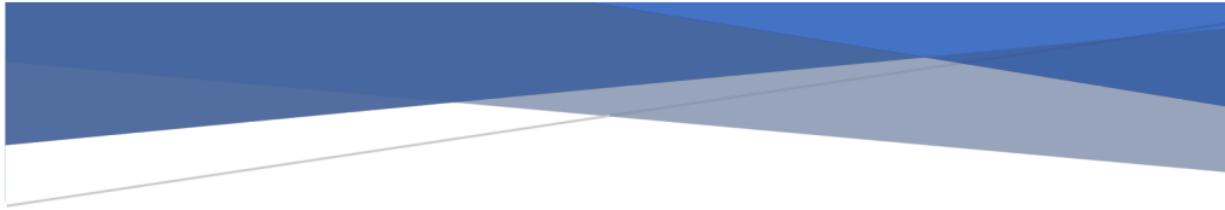


GGP Report 1802143

by Kyle Lovell

FILE	12002_KYLE_LOVELL_GGP_REPORT_1802143_311302_2062350722.DOCX (500.65K)		
TIME SUBMITTED	11-DEC-2020 11:34PM (UTC+0000)	WORD COUNT	541
SUBMISSION ID	139337477	CHARACTER COUNT	2416



GRAPHICS AND GAMES PROGRAMMING

ACCA6021

Kyle Lovell

1802143@student.uwtsd.ac.uk

Contents

Algorithms.....	2
MouseClicked.....	2
KeyboardButton.....	2
findPoint.....	3
Demonstration	4
Computer's Result.....	4
Author's Result.....	5
X	5
Y.....	5
Conclusion.....	5
Screenshots.....	6
Place Control Point.....	6
Draw Curve.....	6
Change Resolution	7
Change Colour	8

Algorithms

MouseClicked

```
void MouseClick(int button, int state, int x, int y)
{
    if (button == 0)
    {
        if (state == GLUT_DOWN)
        {
            controlPoints[count][0] = x;
            controlPoints[count][1] = 500 - y;
            count++;

            if (count > 3)
                count = 0;
        }
    }
    if (button == 2)
    {
        if (state == GLUT_DOWN)
        {
            mousePosY = y;
            rightClickDown = true;
        }
        if (rightClickDown == true && state == GLUT_UP)
        {
            deltaRes = ((mousePosY - y) / resStep);
            currentRes += deltaRes;
            if (currentRes > maxRes)
                currentRes = maxRes;
            else if (currentRes < minRes)
                currentRes = minRes;

            rightClickDown = false;
        }
    }
}
```

The MouseClick function was used to register inputs from the user's mouse. If the left mouse button were pressed, the program would take the mouse's x and y coordinates and create a control point at that location. If the right mouse button were pressed (and held), the program would take the difference in the y coordinate from when the button was pressed and released and would change the resolution of the curve depending on the direction and magnitude of the change.

KeyboardButton

```
void KeyboardButton(unsigned char key, int x, int y)
{
    if (key == 'c')
    {
        colour++;
        if (colour == 7) colour = 0;
        graphics.setColour(colourArray[colour].r, colourArray[colour].g, colourArray[colour].b);
    }
}
```

The KeyboardButton function was used to register inputs from the user's keyboard. If the 'c' key was pressed, the program would cycle through colourArray and set the RGB values of the curve and control points to the next colour in the array.

findPoint

```
void findPoint(int i)
{
    float t = float(i) / float(currentRes);

    float basisFuncs[4] = { pow((1 - t), 3), (3 * pow(t,3)) - (6 * pow(t,2)) + 4, (-3 * pow(t,3)) + (3 * pow(t,2)) + (3 * t) + 1, pow(t,3) };

    float tempX = ((basisFuncs[0] * float(controlPoints[0][0])) + (basisFuncs[1] * float(controlPoints[1][0])) + (basisFuncs[2] * float(controlPoints[2][0])) + (basisFuncs[3] * float(controlPoints[3][0]))) / 6;
    float tempY = ((basisFuncs[0] * float(controlPoints[0][1])) + (basisFuncs[1] * float(controlPoints[1][1])) + (basisFuncs[2] * float(controlPoints[2][1])) + (basisFuncs[3] * float(controlPoints[3][1]))) / 6;

    graphics.drawPixel(tempX, tempY);
}
```

The findPoint function was used to generate the curve by calculating the position of each pixel on the curve. It does this by using the following equation, both provided by the author's lecturer:

$$(B_1(t) \quad B_2(t) \quad B_3(t) \quad B_4(t)) \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{pmatrix}$$

$$\frac{1}{6} \begin{pmatrix} (1-t)^3 & 3t^3 - 6t^2 + 4 & -3t^3 + 3t^2 + 3t + 1 & t^3 \end{pmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{pmatrix}$$

Demonstration

To demonstrate how the findPoint function works, the author will use the algorithm to calculate the x and y coordinate of the 50th point on the curve and compare their result to the coordinates that the program will output.

Computer's Result

```
void findPoint(int i)
{
    float t = float(i) / float(currentRes);

    float basisFuncs[4] = ( pow((1 - t), 3), (3 * pow(t,3)) - (6 * pow(t,2)) + 4, (-3 * pow(t,3)) + (3 * pow(t,2)) + (3 * t) + 1, pow(t,3) );

    float tempX = ((basisFuncs[0] * float(controlPoints[0][0])) + (basisFuncs[1] * float(controlPoints[1][0])) + (basisFuncs[2] * float(controlPoints[2][0])) + (basisFuncs[3] * float(controlPoints[3][0]))) / 6;
    float tempY = ((basisFuncs[0] * float(controlPoints[0][1])) + (basisFuncs[1] * float(controlPoints[1][1])) + (basisFuncs[2] * float(controlPoints[2][1])) + (basisFuncs[3] * float(controlPoints[3][1]))) / 6;

    graphics.drawPixel(tempX, tempY);

    //TESTING VALIDITY OF ALGORITHM
    if (i == 50 && controlPoints[0][0] > 0 && controlPoints[1][0] > 0 && controlPoints[2][0] > 0 && controlPoints[3][0] > 0 && check == true)
    {
        std::cout << std::endl << "\t Point 1: [" << controlPoints[0][0] << ", " << controlPoints[0][1] << "]" << std::endl;
        std::cout << "\t Point 2: [" << controlPoints[1][0] << ", " << controlPoints[1][1] << "]" << std::endl;
        std::cout << "\t Point 3: [" << controlPoints[2][0] << ", " << controlPoints[2][1] << "]" << std::endl;
        std::cout << "\t Point 4: [" << controlPoints[3][0] << ", " << controlPoints[3][1] << "]" << std::endl << std::endl;
        std::cout << "\t X: " << tempX << ", Y: " << tempY;
        check = false;
    }
}
```



```
Select File\Work\Year 3\Graphics & Games Programming\GraphicsTemplate\Debug\GraphicsTemplate.exe

Point 1: [167,416]
Point 2: [284,419]
Point 3: [164,294]
Point 4: [278,294]

X: 223.938, Y: 356.438
```

Author's Result

X

$$(((1-0.5)^3 \times 167) + (((3 \times 0.5^3) - (6 \times 0.5^2) + 4) \times 284) + (((-3 \times 0.5^3) + (3 \times 0.5^2) + (3 \times 0.5) + 1) \times 164) + (0.5^3 \times 278)) / 6$$

$$((0.125 \times 167) + ((0.375 - 1.5 + 4) \times 284) + ((-0.375 + 0.75 + 1.5 + 1) \times 164) + (0.125 \times 278)) / 6$$

$$(20.875 + 816.5 + 471.5 + 34.78) / 6$$

$$1343.625 / 6$$

223.9375

Y

$$(((1-0.5)^3 \times 416) + (((3 \times 0.5^3) - (6 \times 0.5^2) + 4) \times 419) + (((-3 \times 0.5^3) + (3 \times 0.5^2) + (3 \times 0.5) + 1) \times 294) + (0.5^3 \times 294)) / 6$$

$$((0.125 \times 416) + ((0.375 - 1.5 + 4) \times 419) + ((-0.375 + 0.75 + 1.5 + 1) \times 294) + (0.125 \times 294)) / 6$$

$$(52 + 1204.625 + 845.25 + 36.75) / 6$$

$$2138.625 / 6$$

356.4375

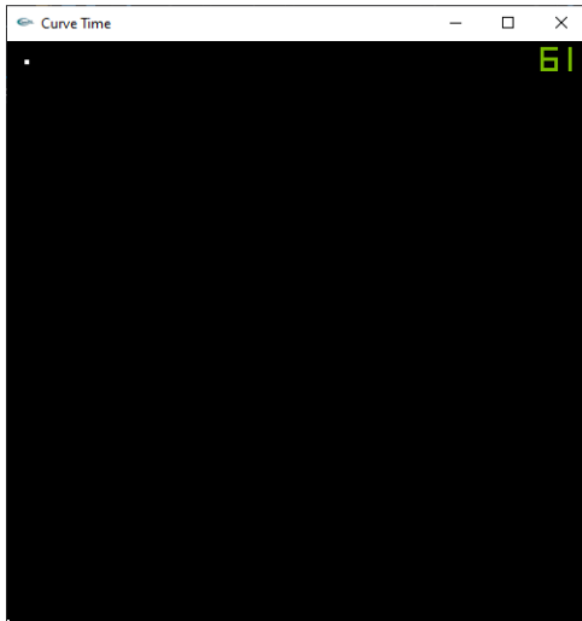
Conclusion

Comparing the computer's results to the author's results shows that, aside from general rounding issues (Which do not have an effect on the placement of the curve as the program would round the results to the nearest integer), the computer calculates the correct coordinates.

Screenshots

Place Control Point

This screenshot shows the program placing a control point (represented by a white pixel) where the user presses the left mouse button (mouse not shown due to capturing limitations).



Draw Curve

This screenshot shows the program drawing a curve between the 4 control points placed.



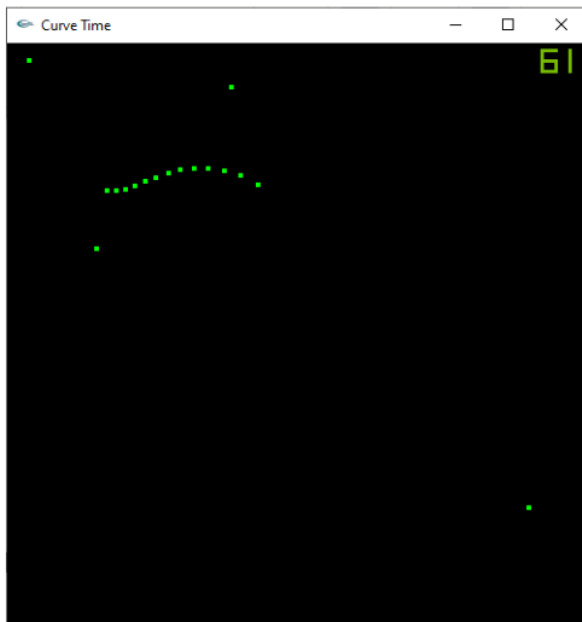
Change Resolution

This screenshot shows the program changing the resolution of the curve based on the direction and magnitude of the distance between the user pressing and releasing the right mouse button.



Change Colour

This screenshot shows the program changing the colour of the curve each time the user presses the 'c' key.



FINAL GRADE

GENERAL COMMENTS

Instructor

73/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

GRADING FORM: SMU GENERIC_GJ

KYLE LOVELL

73

RESEARCH-INFORMED LITERATURE



You need to include a references section at the end of your report, however I note you do cite the

5.5

lecturer's notes in your report's body. Essential topics covered, equations specified, etc. Good.

KNOWLEDGE AND UNDERSTANDING

 A very good understanding of the required concepts demonstrated here. Code generated also shows a good understanding of the necessary SE skills.

16

ANALYSIS

 Problem analysed and solution generated. Clear sign-posting of requirements undertaken/omitted. Good. Good use of SE skills and understanding of the concepts to synthesise a working solution. Additional control points would be nice to see considered.

16

PRACTICAL APPLICATION AND DEP.

 Some good work here. Resolution and rendering of points at different scales shown. Did you consider interpolation between points as the default approach?

30

SKILLS FOR PROFESSIONAL PRAC.

 Report structured but no reference section. Some good SE skills in evidence in code. Screen shots helpful.

5.5