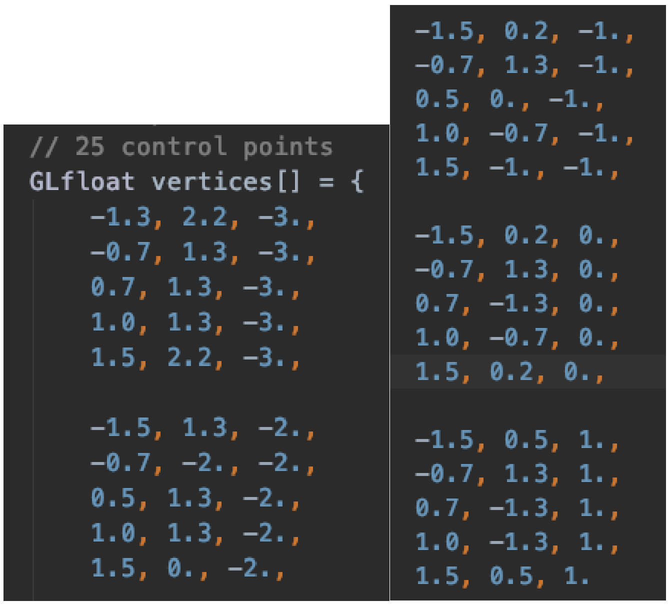
SAMI EMRE ERDOGAN
STUDENT ID: 2019280513

Assignment 5 Bezier surface

My code was referenced and built on top of our Bezier surface tutorial. The only difference from our homework was that instead of having 4 x 4 they asked for a 5 X5 from us. So, in order to achieve that I had to create 25 control points as shown below:

1) Use 25 (5 x 5) control points



```
// 25 control points
GLfloat vertices[] = {
    -1.3, 2.2, -3.,
    -0.7, 1.3, -3.,
    0.7, 1.3, -3.,
    1.0, 1.3, -3.,
    1.5, 2.2, -3.,

    -1.5, 1.3, -2.,
    -0.7, -2., -2.,
    0.5, 1.3, -2.,
    1.0, 1.3, -2.,
    1.5, 0., -2.,
```

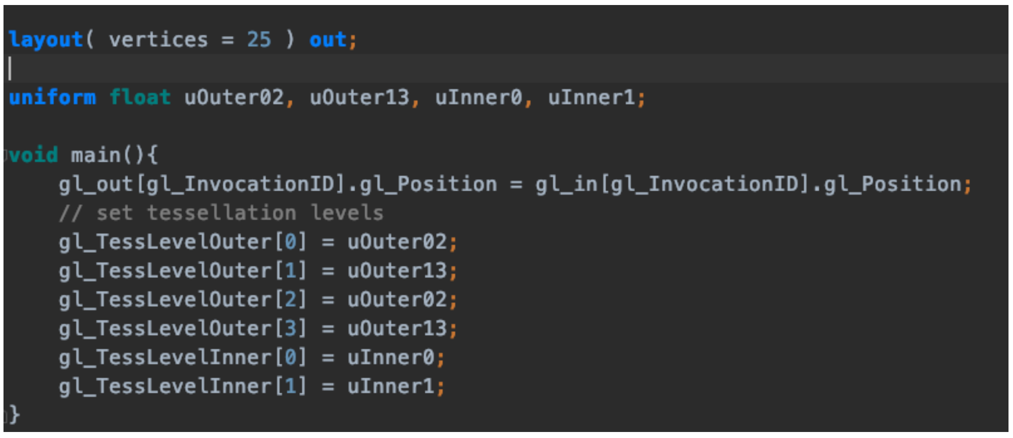
```
-1.5, 0.2, -1.,
-0.7, 1.3, -1.,
0.5, 0., -1.,
1.0, -0.7, -1.,
1.5, -1., -1.,

-1.5, 0.2, 0.,
-0.7, 1.3, 0.,
0.7, -1.3, 0.,
1.0, -0.7, 0.,
1.5, 0.2, 0.,

-1.5, 0.5, 1.,
-0.7, 1.3, 1.,
0.7, -1.3, 1.,
1.0, -1.3, 1.,
1.5, 0.5, 1.
```

2) Use TCS to set subdivision level

I had to specify I had 25 vertices in the TCS file



```
layout( vertices = 25 ) out;

uniform float uOuter02, uOuter13, uInner0, uInner1;

void main(){
    gl_out[gl_InvocationID].gl_Position = gl_in[gl_InvocationID].gl_Position;
    // set tessellation levels
    gl_TessLevelOuter[0] = uOuter02;
    gl_TessLevelOuter[1] = uOuter13;
    gl_TessLevelOuter[2] = uOuter02;
    gl_TessLevelOuter[3] = uOuter13;
    gl_TessLevelInner[0] = uInner0;
    gl_TessLevelInner[1] = uInner1;
}
```

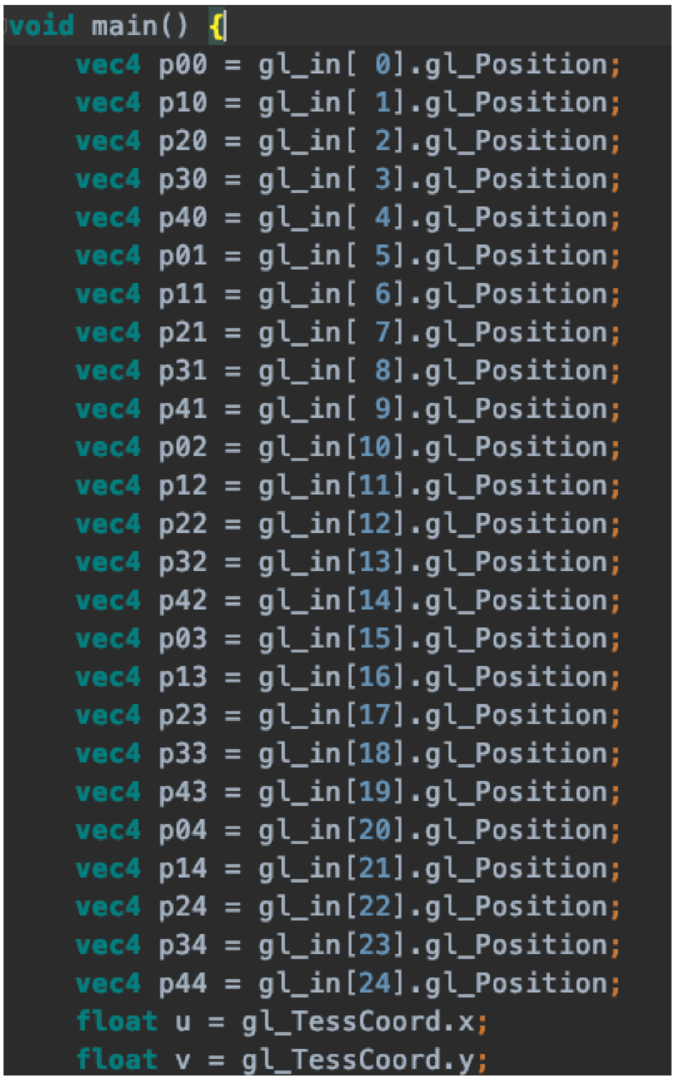
SAMI EMRE ERDOGAN
STUDENT ID: 2019280513

3) Use TES to calculate new vertex coordinates and texture coordinates according to the mathematical equation of Bezier surface

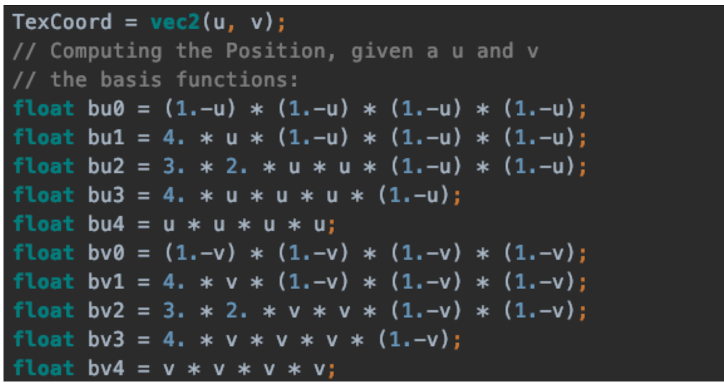
$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_{i,n}(u)\, B_{j,m}(v)\, \mathrm{p}_{i,j}$$

(From the tutorial)

For a 5x 5, I had to follow the formula and so I had to add p43, p04, p14, p24, p34, p44, bu4,bv4



```
void main() {
    vec4 p00 = gl_in[ 0].gl_Position;
    vec4 p10 = gl_in[ 1].gl_Position;
    vec4 p20 = gl_in[ 2].gl_Position;
    vec4 p30 = gl_in[ 3].gl_Position;
    vec4 p40 = gl_in[ 4].gl_Position;
    vec4 p01 = gl_in[ 5].gl_Position;
    vec4 p11 = gl_in[ 6].gl_Position;
    vec4 p21 = gl_in[ 7].gl_Position;
    vec4 p31 = gl_in[ 8].gl_Position;
    vec4 p41 = gl_in[ 9].gl_Position;
    vec4 p02 = gl_in[10].gl_Position;
    vec4 p12 = gl_in[11].gl_Position;
    vec4 p22 = gl_in[12].gl_Position;
    vec4 p32 = gl_in[13].gl_Position;
    vec4 p42 = gl_in[14].gl_Position;
    vec4 p03 = gl_in[15].gl_Position;
    vec4 p13 = gl_in[16].gl_Position;
    vec4 p23 = gl_in[17].gl_Position;
    vec4 p33 = gl_in[18].gl_Position;
    vec4 p43 = gl_in[19].gl_Position;
    vec4 p04 = gl_in[20].gl_Position;
    vec4 p14 = gl_in[21].gl_Position;
    vec4 p24 = gl_in[22].gl_Position;
    vec4 p34 = gl_in[23].gl_Position;
    vec4 p44 = gl_in[24].gl_Position;
    float u = gl_TessCoord.x;
    float v = gl_TessCoord.y;
```



```
TexCoord = vec2(u, v);
// Computing the Position, given a u and v
// the basis functions:
float bu0 = (1.-u) * (1.-u) * (1.-u) * (1.-u);
float bu1 = 4. * u * (1.-u) * (1.-u) * (1.-u);
float bu2 = 3. * 2. * u * u * (1.-u) * (1.-u);
float bu3 = 4. * u * u * u * (1.-u);
float bu4 = u * u * u * u;
float bv0 = (1.-v) * (1.-v) * (1.-v) * (1.-v);
float bv1 = 4. * v * (1.-v) * (1.-v) * (1.-v);
float bv2 = 3. * 2. * v * v * (1.-v) * (1.-v);
float bv3 = 4. * v * v * v * (1.-v);
float bv4 = v * v * v * v;
```

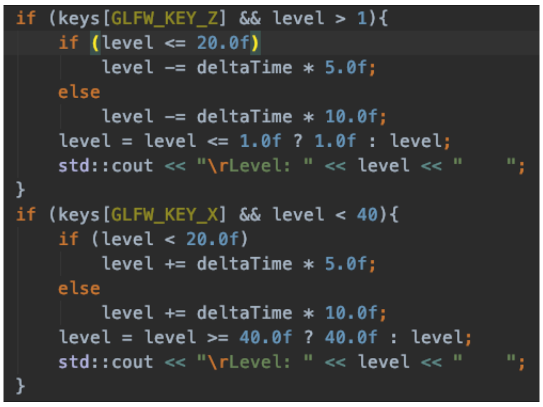
The final computation is as follows in the program:

gl_Position = bu0 * (bv0*p00 + bv1*p01 + bv2*p02 + bv3*p03 + bv4*p04) + bu1 * (bv0*p10 + bv1*p11 + bv2*p12 + bv3*p13 + bv4*p14) + bu2 * (bv0*p20 + bv1*p21 + bv2*p22 + bv3*p23 + bv4*p24) + bu3 * (bv0*p30 + bv1*p31 + bv2*p32 + bv3*p33 + bv4*p34) + bu4 * (bv0*p40 + bv1*p41 + bv2*p42 + bv3*p43 + bv4*p44);

SAMI EMRE ERDOGAN
STUDENT ID: 2019280513

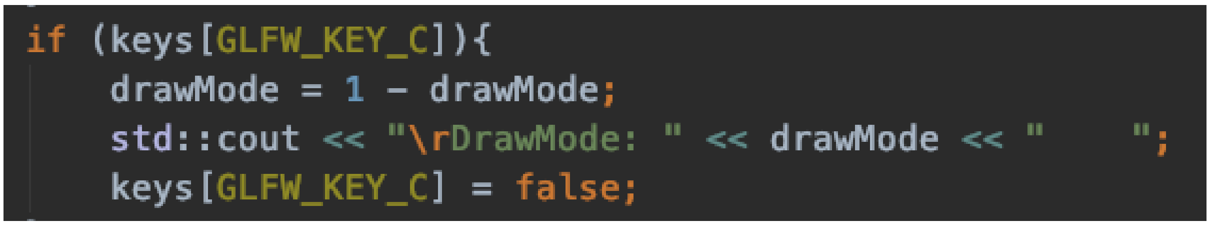
4) Change smoothness of the surface by keyboard

Use X AND Z key to change



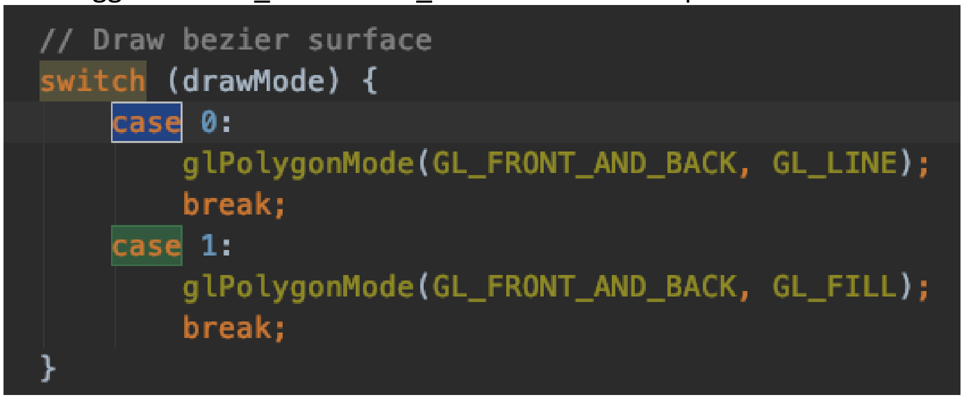
```
if (keys[GLFW_KEY_Z] && level > 1){
    if (level <= 20.0f)
        level -= deltaTime * 5.0f;
    else
        level -= deltaTime * 10.0f;
    level = level <= 1.0f ? 1.0f : level;
    std::cout << "\rLevel: " << level << "    ";
}
if (keys[GLFW_KEY_X] && level < 40){
    if (level < 20.0f)
        level += deltaTime * 5.0f;
    else
        level += deltaTime * 10.0f;
    level = level >= 40.0f ? 40.0f : level;
    std::cout << "\rLevel: " << level << "    ";
}
```

5) Support wireframe mode display.
Use the key C to change modes.



```
if (keys[GLFW_KEY_C]){
    drawMode = 1 - drawMode;
    std::cout << "\rDrawMode: " << drawMode << "    ";
    keys[GLFW_KEY_C] = false;
```

This triggers the GL_FILL and GL_LINE when KEY C is pressed

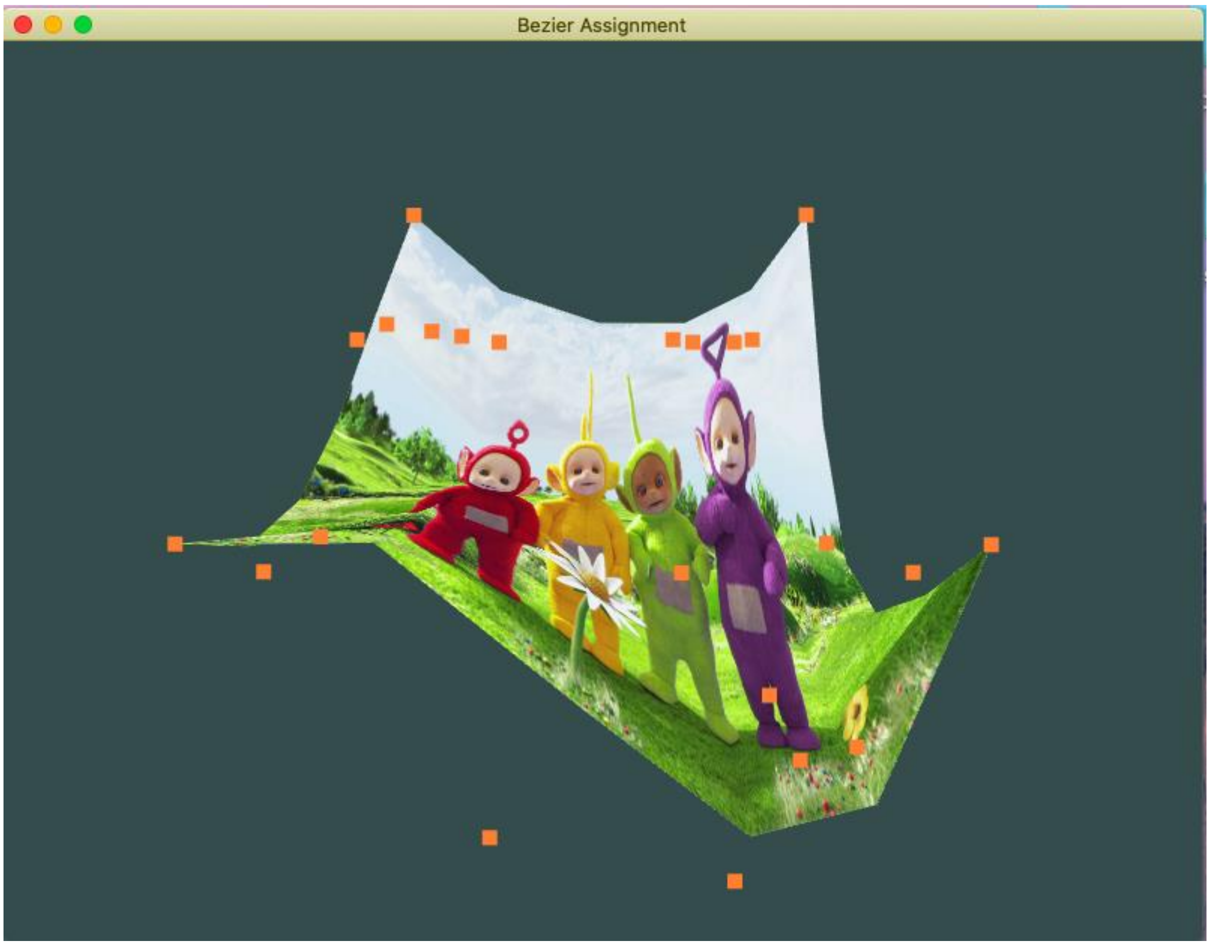


```
// Draw bezier surface
switch (drawMode) {
    case 0:
        glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
        break;
    case 1:
        glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
        break;
}
```

6) Add texture to Bezier surface. Choose the texture by yourself.
The textures are drawn by the shader files.

SAMI EMRE ERDOGAN
STUDENT ID: 2019280513

Final result:


Bezier Assignment