KAIST CS492 - Homework 3

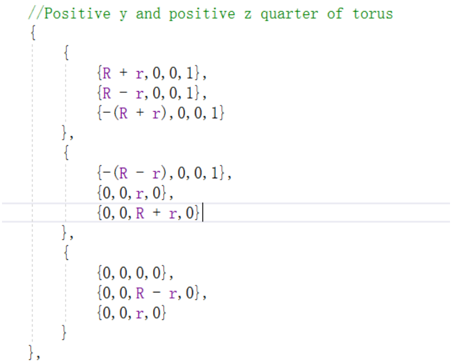
XU YIN - 20205445

November 8, 2021

Question 2

0.1 Evaluating a point on the surface

- Fill the control points (see Fig. 1).



```
//Positive y and positive z quarter of torus
{
    {
        {R + r, 0, 0, 1},
        {R - r, 0, 0, 1},
        {-(R + r), 0, 0, 1}
    },
    {
        {-(R - r), 0, 0, 1},
        {0, 0, r, 0},
        {0, 0, R + r, 0}
    },
    {
        {0, 0, 0, 0},
        {0, 0, R - r, 0},
        {0, 0, r, 0}
    }
},
```

Figure 1: fill the control points of one quarter.

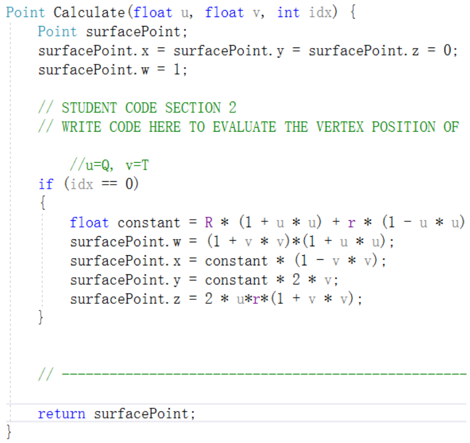
- Fill in the function Calculate(u,v,k)(show in Fig. 2).
- Demonstration (see Fig. 3).

0.2 Construct a mesh

- list the vertexes and insert the face (see Fig. 4).
- Demonstration (see Fig. 5).

0.3 A whole torus

- Insert all control points and use the symmetry calculation (see Fig. 6).



```
Point Calculate(float u, float v, int idx) {
    Point surfacePoint;
    surfacePoint.x = surfacePoint.y = surfacePoint.z = 0;
    surfacePoint.w = 1;

    // STUDENT CODE SECTION 2
    // WRITE CODE HERE TO EVALUATE THE VERTEX POSITION OF

        //u=Q, v=T
    if (idx == 0)
    {
        float constant = R * (1 + u * u) + r * (1 - u * u)
        surfacePoint.w = (1 + v * v)*(1 + u * u);
        surfacePoint.x = constant * (1 - v * v);
        surfacePoint.y = constant * 2 * v;
        surfacePoint.z = 2 * u*r*(1 + v * v);
    }


    // ---------------------------------------------------

    return surfacePoint;
}
```

Figure 2: fill the calculation function.

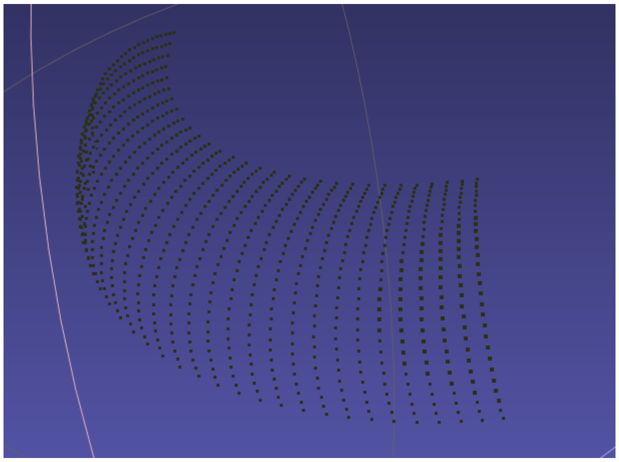
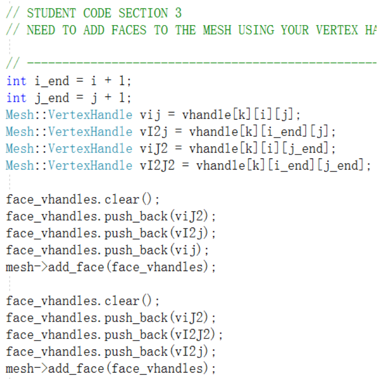


Figure 3: fill the control points of one quarter.

- Add faces by different indices.
- Demonstration of the whole torus.



```
// STUDENT CODE SECTION 3
// NEED TO ADD FACES TO THE MESH USING YOUR VERTEX HA

// ---------------------------------------------------
int i_end = i + 1;
int j_end = j + 1;
Mesh::VertexHandle vij = vhandle[k][i][j];
Mesh::VertexHandle vI2j = vhandle[k][i_end][j];
Mesh::VertexHandle viJ2 = vhandle[k][i][j_end];
Mesh::VertexHandle vI2J2 = vhandle[k][i_end][j_end];

face_vhandles.clear();
face_vhandles.push_back(viJ2);
face_vhandles.push_back(vI2j);
face_vhandles.push_back(vij);
mesh->add_face(face_vhandles);

face_vhandles.clear();
face_vhandles.push_back(viJ2);
face_vhandles.push_back(vI2J2);
face_vhandles.push_back(vI2j);
mesh->add_face(face_vhandles);
```

Figure 4: Demonstration of a quarter.

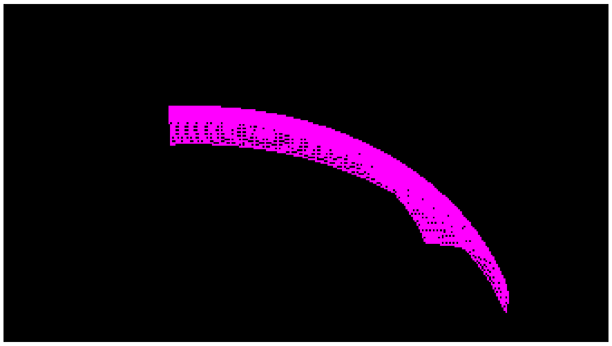
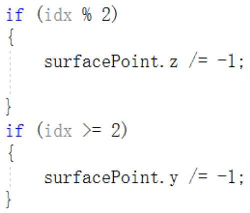
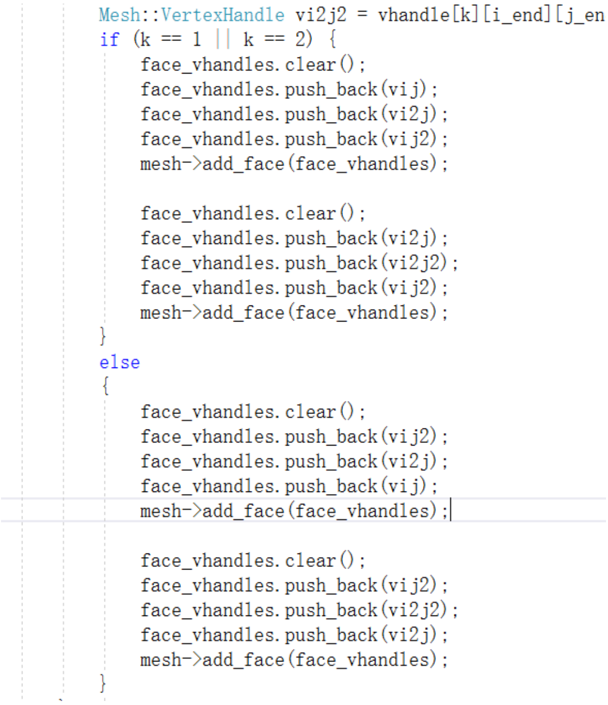


Figure 5: A quarter of torus.



```
if (idx % 2)
{
    surfacePoint.z /= -1;
}
if (idx >= 2)
{
    surfacePoint.y /= -1;
}
```

Figure 6: Using the symmetry calculation.



```
Mesh::VertexHandle vi2j2 = vhandle[k][i_end][j_en
if (k == 1 || k == 2) {
    face_vhandles.clear();
    face_vhandles.push_back(vij);
    face_vhandles.push_back(vi2j);
    face_vhandles.push_back(vij2);
    mesh->add_face(face_vhandles);

    face_vhandles.clear();
    face_vhandles.push_back(vi2j);
    face_vhandles.push_back(vi2j2);
    face_vhandles.push_back(vij2);
    mesh->add_face(face_vhandles);
}
else
{
    face_vhandles.clear();
    face_vhandles.push_back(vij2);
    face_vhandles.push_back(vi2j);
    face_vhandles.push_back(vij);
    mesh->add_face(face_vhandles);

    face_vhandles.clear();
    face_vhandles.push_back(vij2);
    face_vhandles.push_back(vi2j2);
    face_vhandles.push_back(vi2j);
    mesh->add_face(face_vhandles);
}
```

Figure 7: add respective faces.

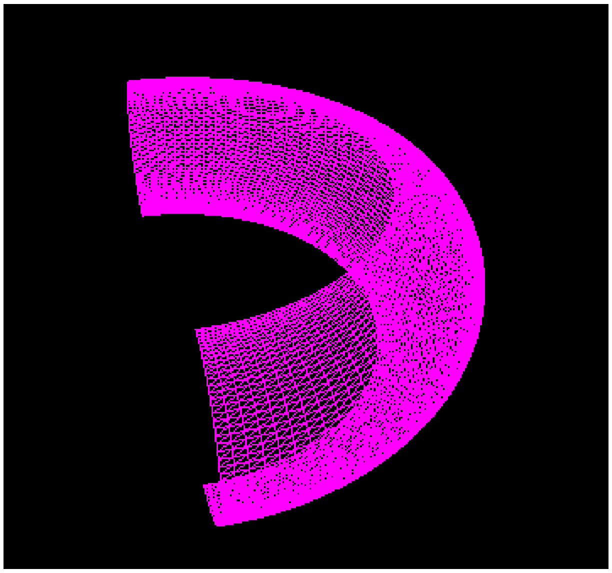


Figure 8: Show the whole torus.