CG Practice 6

COLLEGE OF COMPUTING

HANYANG ERICA CAMPUS

Q YOUN HONG (홍규연)

이번 실습에서는…

- 메시 렌더링 실습
- 곡면 모델링

Drawing a Cylinder Mesh

Objectives

- Mesh object 그리기
  - 간단한 곡면을 mesh로 그려보기
  - Vertex의 vertex buffer 중복저장을 막는 방법
  - 여러 개의 VAO/VBO를 이용해서 Scene을 그리기
  - 일반적인 mesh를 그려보자

main_cylinder.cpp

- geometric object를 저장하기 위한 data structure 생성

```
class TriMesh
{
public:
    int NumVertices;
    int NumTris;

    GLuint vao;
    GLuint vbo;
    GLuint ebo; // element buffer object for indices

    std::vector<vec4> vertices;

    vec3 color;
    mat4 modelTransform;

    std::vector<unsigned int> indices;
…
```

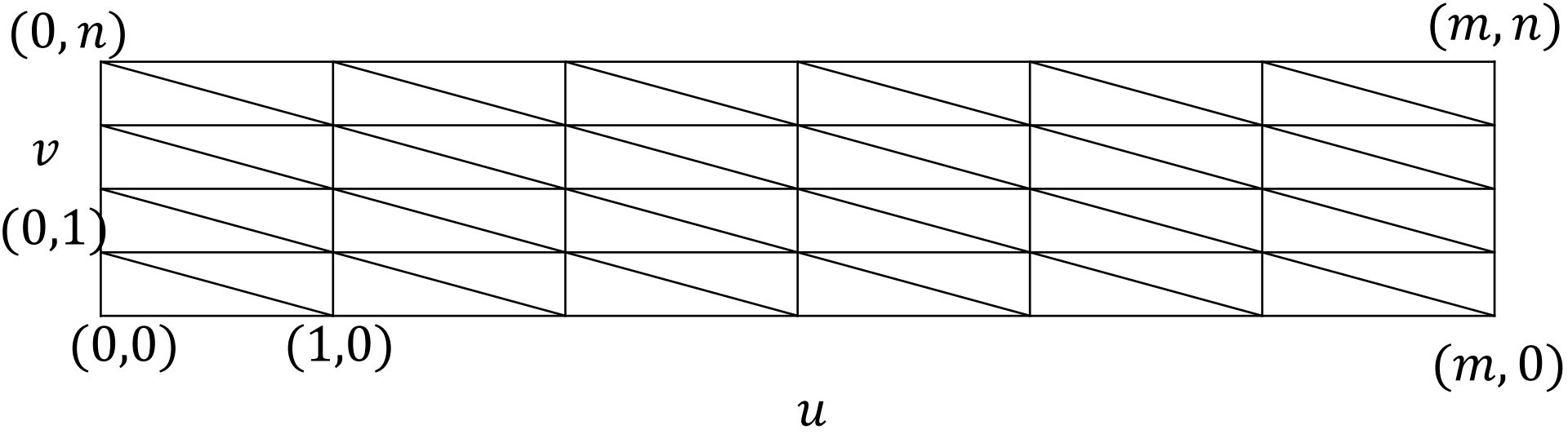
Mesh representaton

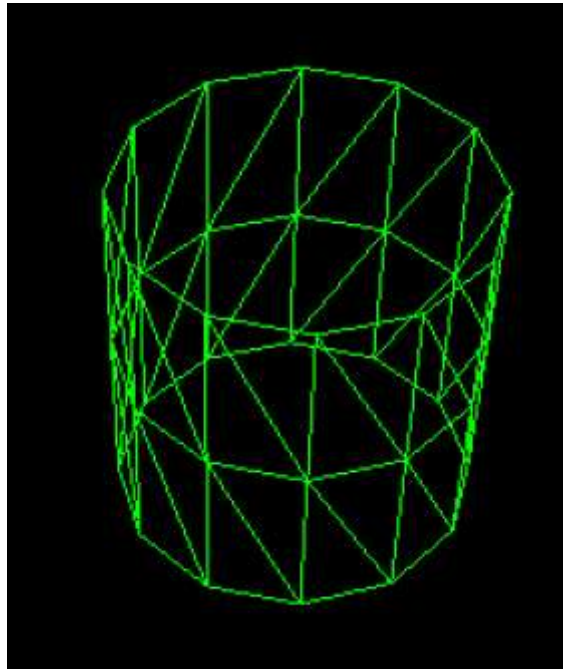
Vertex 1	x_1	y_1	z_1
Vertex 2	x_2	y_2	z_2
Vertex 3	x_3	y_3	z_3

Triangle 1	1	2	3
Triangle 2	3	2	4
Triangle 3	4	2	5
Triangle 4	7	5	6
Triangle 5	6	5	8
Triangle 6	8	5	1

main_cylinder.cpp

- Cylinder 곡면을 Mesh 형태로 저장하기
- Cylinder의 식: $S(u,v) = (R\cos u, R\sin u, v),\ u \in [0, 2\pi], v \in [0, H]$
- Cylinder의 원을 m등분, 세로축을 n등분하면 $(m+1)\cdot(n+1)$ 개의 점이 필요하고 이로부터 $2mn$개의 삼각형 생성





main_ cylinder.cpp

- Cylinder의 원을 m등분, 세로축을 n등분하면 $(m+1)\cdot(n+1)$개의 점이 필요하고 이로부터 $2mn$개의 삼각형 생성

```
int MakeCylinder(TriMesh *mesh, GLfloat radius, GLfloat height, int nR, int nV)
{
    …
    GLfloat deltaR = 2.0 * M_PI / (GLfloat)nR;
    GLfloat deltaV = height / (GLfloat)nV;

    mesh->NumVertices = (nR + 1) * (nV + 1);
    mesh->NumTris = nR * nV * 2;
    mesh->vertices.resize(mesh->NumVertices);
    mesh->indices.resize(mesh->NumTris * 3);

    for (int j = 0; j <= nV; j++) {
        for (int i = 0; i <= nR; i++) {
            vec3 pt;
            pt.x = radius * cos(deltaR * (GLfloat)i);
            pt.y = radius * sin(deltaR * (GLfloat)i);
            pt.z = deltaV * (GLfloat)j;
            mesh->vertices[(nR + 1) * j + i] = vec4(pt, 1.0);
        }
    }
```

main_ cylinder.cpp

- Cylinder의 원을 m등분, 세로축을 n등분하면 $(m+1)\cdot(n+1)$개의 점이 필요하고 이로부터 $2mn$개의 삼각형 생성

```
int MakeCylinder(TriMesh *mesh, GLfloat radius, GLfloat height, int nR, int nV)
{
    …
    for (int j = 0; j < nV; j++) {
        for (int i = 0; i < nR; i++) {
            int quadId = nR * j + i;
            mesh->indices[6 * quadId] = (nR + 1) * j + i;
            mesh->indices[6 * quadId + 1] = (nR + 1) * (j+1) + i;
            mesh->indices[6 * quadId + 2] = (nR + 1) * j + i+1;
            mesh->indices[6 * quadId + 3] = (nR + 1) * (j+1) + i;
            mesh->indices[6 * quadId + 4] = (nR + 1) * (j+1) + i+1;
            mesh->indices[6 * quadId + 5] = (nR + 1) * j + i+1;
        }
    }
    …
}
```

u

main_ cylinder.cpp

- Drawing: glDrawArrays() 대신 glDrawElements() 사용

```
void TriMesh::Render()
{
  //set a constant color
  glUniform3fv(color_loc, 1, color);
  glUniformMatrix4fv(model, 1, GL_TRUE, modelTransform);

  glBindVertexArray(vao);
  glDrawElements(GL_TRIANGLES, indices.size(), GL_UNSIGNED_INT, 0);
  //glBindVertexArray(0);
}
```

main_ cylinder.cpp

- TriMesh 자료 구조에는 각각의 VAO/VBO/EBO가 저장되어 있음
  - 원하는 메시를 그리기 전에 해당 메시 데이터를 저장하는 VAO를 활성화함

```
class TriMesh
{
public:
    …

    GLuint vao;
    GLuint vbo;
    GLuint ebo; // element buffer object for indices
void TriMesh::Render()
{
  //set a constant color
  glUniform3fv(color_loc, 1, color);
  glUniformMatrix4fv(model, 1, GL_TRUE, modelTransform);

  glBindVertexArray(vao);

  //glEnableVertexAttribArray(0);
  //glVertexAttribPointer(0, 4, GL_FLOAT, GL_FALSE, 0, BUFFER_OFFSET(0));

  glDrawElements(GL_TRIANGLES, indices.size(), GL_UNSIGNED_INT, 0);
}
```

Extra) Cylinder with Edges

- glPolygonMode(GLenum face, Glenum mode):

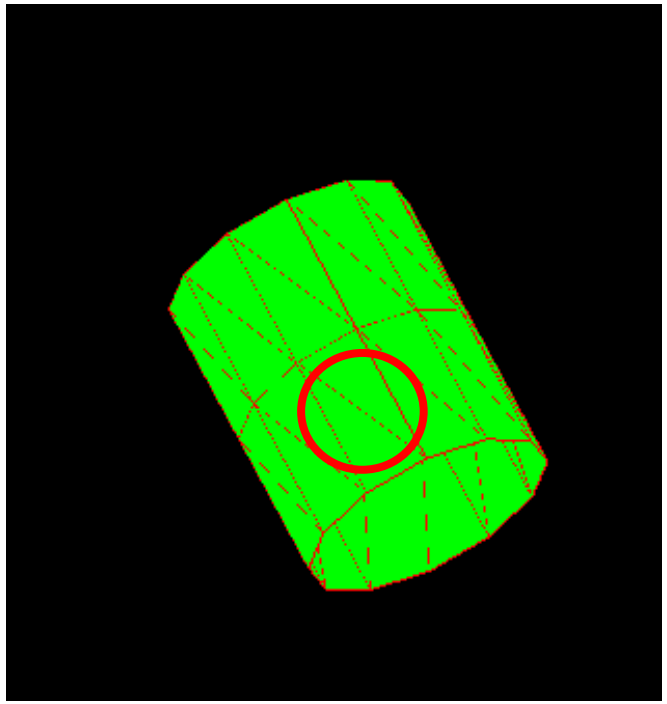
  https://registry.khronos.org/OpenGL-Refpages/gl4/html/glPolygonMode.xhtml

  - face: mode를 적용할 폴리곤의 면
  - `GL_FRONT_AND_BACK`: polygon의 앞면과 뒷면에 mode 적용
  - mode: 폴리곤이 어떻게 rasterize될 것인지 결정. 가능한 값은
  - `GL_POINT`, `GL_LINE`, `GL_FILL` 가능(default값 = `GL_FILL`)
- 메시를 wire-frame으로 그리기 위해서는

```
glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
glDrawArrays(…) or glDrawElements(…)
```

Extra) Cylinder With Edges

Cylinder의 면과 모서리를 같이 그리기 위해서 Cylinder를 두 번 그림



```
glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
Cylinder.color = vec3(0.0, 1.0, 0.0);
Cylinder.Render();

glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
Cylinder.color = vec3(1.0, 0.0, 0.0);
Cylinder.Render();
glutSwapBuffers();
```

- Z-fighting: 같은 평면에 있는 면과 직선이 동시에 그려질 때 면과 선 중 어느 물체가 더 눈에 가까운지 깊이 판별이 애매해짐
- Polygon들을 그릴 때, polygon의 fragment들을 viewer로부터 약간 멀어지게 그림

```
glEnable(GL_POLYGON_OFFSET_FILL);
glPolygonOffset(1.0, 1.0);
```

Practice) Surface Of Revolution 그리기

Q) make_cylinder.cpp를 수정하여 다른 형태의 곡면 그리기

- 실린더의 좌표 식

$$\boldsymbol{x}(u,v) = (R\cos u\,, R\sin u\,, v)$$

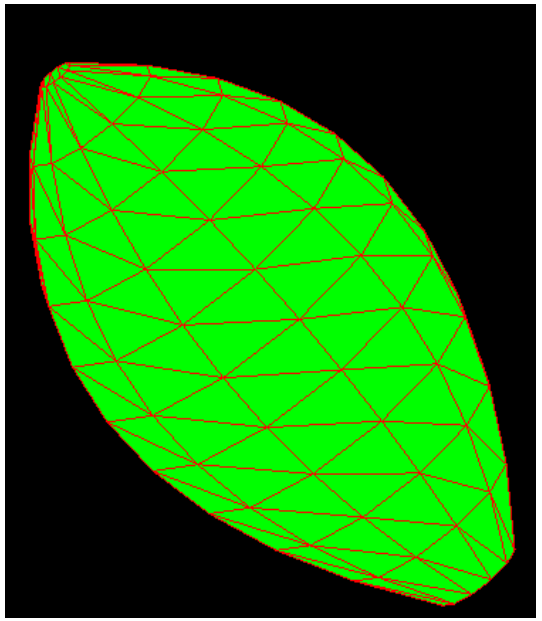
이 좌표 식에 의해서 각각의 좌표는 다음과 같이 계산되었음

```
pt.x = radius * cos(deltaR * (GLfloat)i);
pt.y = radius * sin(deltaR * (GLfloat)i);
pt.z = deltaV * (GLfloat)j;
```

- 이 좌표의 식을 Surface of Revolution의 식으로 바꾸기
  - Profile 곡선 좌표가 $\beta(t) = (\beta_x(t), 0, \beta_z(t))$이고 회전축이 z-축일때

$$x(u,v) = (\beta_x(u)\cos v\,, \beta_x(u)\cos v\,, \beta_z(u))$$

  - 이 곡면을 생성하기 위해 위의 코드를 어떻게 수정하겠는가?
  - E.g.) Profile curve가 $\beta(t) = \begin{bmatrix} 0.2 \\ 0 \\ -1.0 \end{bmatrix} B_0^2(t) + \begin{bmatrix} 3.6 \\ 0 \\ 0.2 \end{bmatrix} B_1^2(t) + \begin{bmatrix} 0.5 \\ 0 \\ 2.0 \end{bmatrix} B_2^2(t)$



Drawing a General Mesh

임의의 triangle mesh 그리기

- OBJ file format: triangle mesh를 저장하는데 많이 사용되는 file format
  - vertex elements
    - vertex positions: v x,y,z [w]-cooridnates i.e. v 0.123 0.234 0.345 1.0
    - texture cooridnates: vt u,v [w]-coordinate i.e. vt 0.5 0.5
    - vertex normal: vn x,y,z-coordinates (might not be a unit vector)

  …

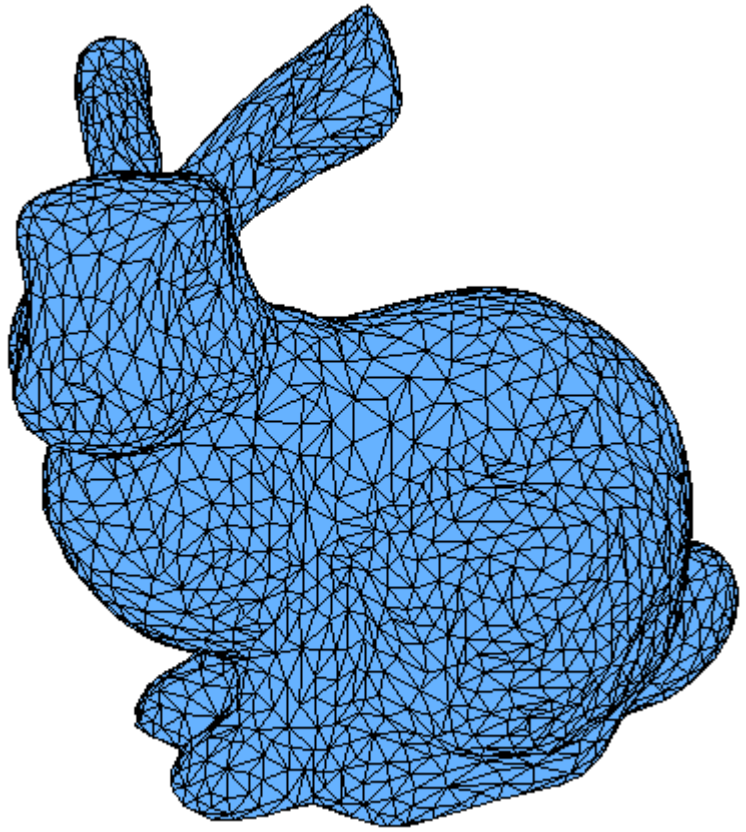
  - polygonal face element
    - face indices: f 1 2 3 or f 3/1 4/2 5/3 or f 6/4/1 3/5/3 7/6/5 or f 7//1 8//2 9//3 (index 1부터 시작)

  - Reference: https://en.wikipedia.org/wiki/Wavefront_.obj_file
  - OBJLoader.h 제공: 필요에 따라 수정

OBJ Mesh 읽기 코드 예시


```
CObjLoader* objLoader = new CObjLoader;
objLoader->Load("bunny.obj");

mesh.NumVertices = objLoader->getSize();
mesh.NumTris = objLoader->parts[0].faces.size();

mesh.vertices.resize(mesh.NumVertices);
for (int i = 0; i < mesh.NumVertices; i++) {
    tVertex vtx = objLoader->getVertex(i);
    mesh.vertices[i] = vec4(vtx.x, vtx.y, vtx.z, 1.0);
}

mesh.indices.resize(mesh.NumTris * 3);
for (int i = 0; i < mesh.NumTris; i++) {
    tFace* face = &(objLoader->parts[0].faces[i]);

    for (int j = 0; j < 3; j++)
        mesh.indices[3 * i + j] = (unsigned int)(face->v[j] - 1);
}

delete objLoader;
```

OBJ Mesh 읽기



Review Tasks 20231103 – Drawing Meshes

- Task: Bunny가 아닌 다른 obj에 저장된 메시 1개와 Surface of Revolution으로 만든 곡면 1개를 한 화면에 그리기
  - 다른 무료 OBJ data를 찾을 수 있는 곳
    - https://github.com/alecjacobson/common-3d-test-models/tree/master : 그래픽스 연구에서 많이 사용되는 mesh 데이터의 리스트
    - TurboSquid.com, Free3D.com, Thingiverse.com 등에서도 무료 모델 일부 제공
    - 숙제 제출 시 지나치게 용량이 크기 않은 OBJ 파일을 선택할 것
  - GLUT Window의 제목을 자신의 학번으로 할 것
  - main.cpp, vshader, fshader, obj 파일 + (실행파일 및 스크린샷)을 zip 파일에 압축하여 제출