中山大学 SUN YAT-SEN UNIVERSITY

计算机图形学

作业五：多边形网格

数据科学与计算机学院　17大数据与人工智能

17341015　陈鸿峥

一、 实验原理

1. 文件读入

分别对OBJ、PLY和OFF文件解析进行读入，格式规范分别如下

- OBJ：以v开头的为顶点，之后三个数为(vx, vy, vz)坐标。以f开头的为面，之后三个数为三角形面上的三个顶点索引，**从1开始编号**。

```
v vx vy vz
f idx1 idx2 idx3
```

- PLY：从头部可以直接读出顶点数$|V|$和面数$|F|$，之后$|V|$行为顶点坐标，再之后$|F|$行为面的坐标。顶点**从0开始编号**。

```
ply
format ascii 1.0
comment VCGLIB generated
element vertex 5261
property float x
property float y
property float z
element face 10518
property list uchar int vertex_indices
end_header
```

- OFF：第一行为OFF标记，第二行为顶点数、面数，之后的读入方法同PLY文件。顶点坐标同样**从0开始编号**。

2. 模型显示

1. 先设好GL_MODELVIEW，然后将模型平移到合适位置
2. 同时需用gluPerspective设好观察点位置
3. 对于wireframe只需将顶点之间进行连线，flat对面进行着色，flat lines则连线和着色均要进行

二、 实验结果

当前文件夹下有三个执行文件，`cow.exe`、`cactus.exe`和`armadillo.exe`分别对应着三个输入模型。

每个执行文件双击即可运行，具体的功能按键如下

按键	功能
1	Wireframe模式
2	Flat模式
3	Flat lines模式
r	绕Y轴顺时针旋转
R	绕Y轴逆时针旋转
w	向上平移
a	向左平移
s	向下平移
d	向右平移

实验结果如图1、图2和图3所示，从左到右依次是wireframe、flat、flat lines模式。

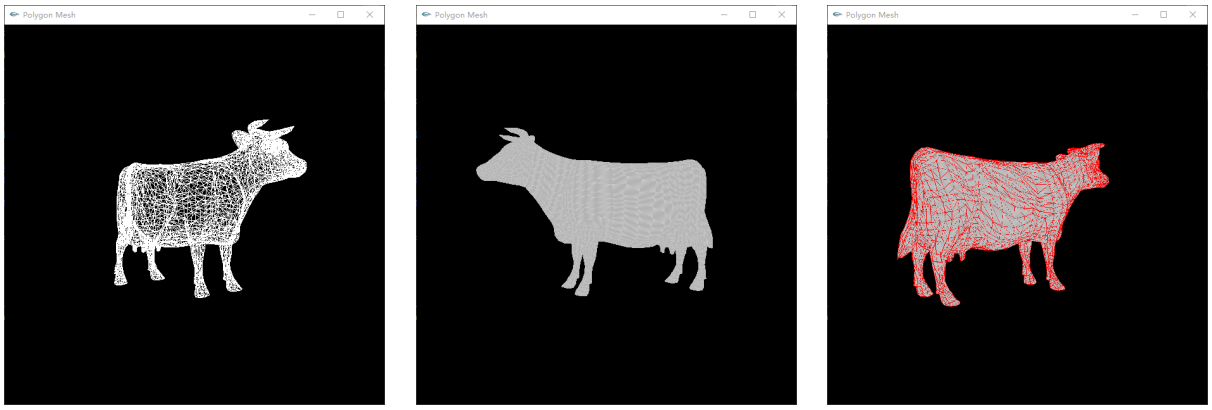

图 1: cow.obj模型

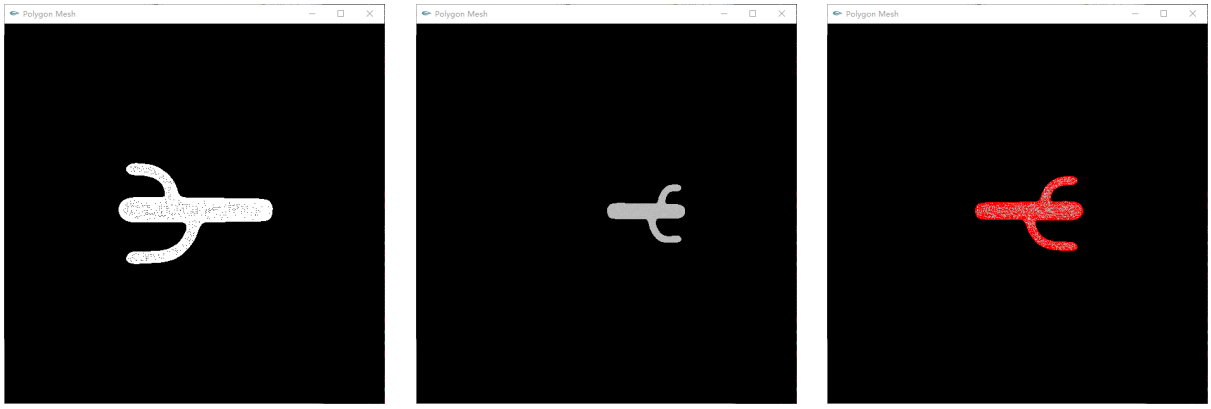

图 2: cactus.ply模型

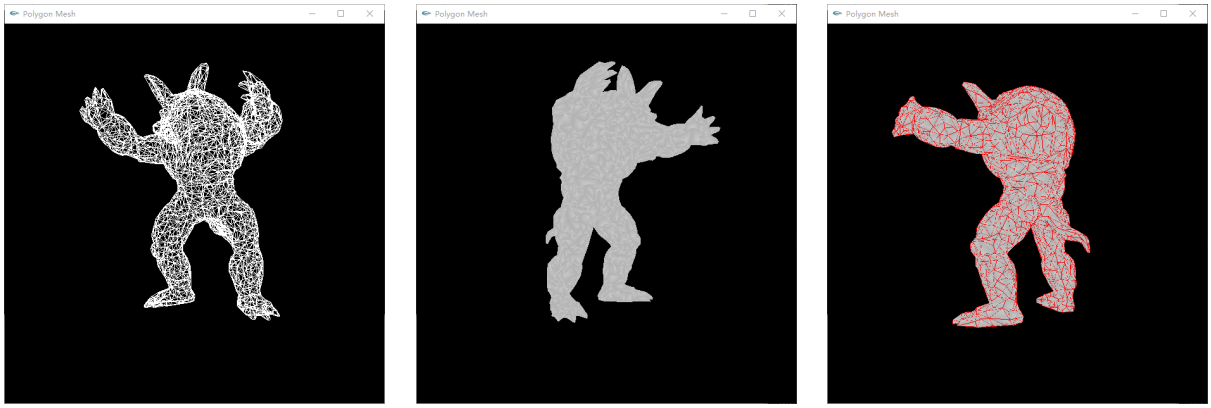

图 3: armadillo.off模型

附录 A. 源代码

mesh.h包含三种格式文件的读取器。

```
#ifndef MESH_H
#define MESH_H

#include <vector>
#include <cstdio>
#include <string>
#include <cstring>
#include <GL/glut.h>
using namespace std;

template <class T>
class vec3
{
public:
```

```
15      vec3() : x(0), y(0), z(0) {}
16      vec3(const T px, const T py, const T pz) {
17          set(px,py,pz);
18      }
19      vec3(const vec3<T>& v) {
20          set(v.x,v.y,v.z);
21      }
22      void set(const vec3<T>& v) {
23          set(v.x,v.y,v.z);
24      }
25      void set(const T px,const T py, const T pz) {
26          this->x = px;
27          this->y = py;
28          this->z = pz;
29      }
30      T x, y, z;
31  };
32
33  class Mesh
34  {
35  public:
36      Mesh() {}
37
38      // The following function is partly referenced from
39      // http://www.opengl-tutorial.org/beginners-tutorials/tutorial-7-model-loading
            ↪ /
40      bool loadOBJ(const char * path) {
41
42          fmode = 1;
43
44          printf("Loading OBJ file %s...\n", path);
45
46          FILE * file = fopen(path, "r");
47          if( file == NULL ){
48              printf("Error: No this file!\n");
49              getchar();
50              return false;
51          }
52
53          while( 1 ){
54
55              char lineHeader[128];
56              // read the first word of the line
57              int res = fscanf(file, "%s", lineHeader);
58              if (res == EOF)
```

```
59              break; // EOF = End Of File. Quit the loop.
60
61          // else : parse lineHeader
62          if ( strcmp( lineHeader, "v" ) == 0 ){
63              vec3<GLfloat> vertex;
64              fscanf(file, "%f %f %f\n", &vertex.x, &vertex.y, &vertex.z);
65              vertices.push_back(vertex);
66          } else if ( strcmp( lineHeader, "f" ) == 0 ){
67              vec3<GLint> face;
68              int matches = fscanf(file, "%d %d %d\n", &face.x, &face.y, &face.z);
69              if (matches != 3){
70                  printf("Error: File parser!\n");
71                  fclose(file);
72                  return false;
73              }
74              faces.push_back(face);
75          } else {
76              // Probably a comment, eat up the rest of the line
77              char stupidBuffer[1000];
78              fgets(stupidBuffer, 1000, file);
79          }
80
81      }
82      fclose(file);
83      printf("Finish loading obj file.\n");
84      return true;
85  }
86
87  bool loadPLY(const char * path) {
88
89      fmode = 2;
90
91      printf("Loading PLY file %s...\n", path);
92
93      FILE * file = fopen(path, "r");
94      if( file == NULL ){
95          printf("Error: No this file!\n");
96          getchar();
97          return false;
98      }
99
100     int vertex_num, face_num;
101
102     while( 1 ){
103
```

```
104             char lineHeader[128];
105             // read the first word of the line
106             int res = fscanf(file, "%s", lineHeader);
107             if (res == EOF)
108                 break; // EOF = End Of File. Quit the loop.
109 
110             // else : parse lineHeader
111             if ( strcmp( lineHeader, "element" ) == 0 ){
112                 int num;
113                 char str[20];
114                 fscanf(file, "%s %d\n", &str, &num);
115                 if (strcmp(str,"vertex") == 0){
116                     vertex_num = num;
117                     printf("# Vertex: %d\n", vertex_num);
118                 } else if (strcmp(str,"face") == 0){
119                     face_num = num;
120                     printf("# Face: %d\n", face_num);
121                 }
122             } else if ( strcmp( lineHeader, "end_header" ) == 0 ){
123                 break;
124             }
125         }
126 
127         // read vertices
128         for (int i = 0; i < vertex_num; ++i){
129             vec3<GLfloat> vertex;
130             fscanf(file, "%f %f %f\n", &vertex.x, &vertex.y, &vertex.z);
131             vertices.push_back(vertex);
132         }
133         // read faces
134         for (int i = 0; i < face_num; ++i){
135             vec3<GLint> face;
136             int num;
137             int matches = fscanf(file, "%d %d %d %d\n", &num, &face.x, &face.y, &
                ↪ face.z);
138             faces.push_back(face);
139         }
140         fclose(file);
141         printf("Finish loading ply file.\n");
142         return true;
143     }
144 
145     bool loadOFF(const char * path) {
146 
147         fmode = 3;
```

```
148
149         printf("Loading OFF file %s...\n", path);
150
151         FILE * file = fopen(path, "r");
152         if( file == NULL ){
153             printf("Error: No this file!\n");
154             getchar();
155             return false;
156         }
157
158         int vertex_num, face_num, n;
159
160         char lineHeader[128];
161         fscanf(file, "%s", lineHeader);
162         fscanf(file, "%d %d %d", &vertex_num, &face_num, &n);
163
164         // read vertices
165         for (int i = 0; i < vertex_num; ++i){
166             vec3<GLfloat> vertex;
167             fscanf(file, "%f %f %f\n", &vertex.x, &vertex.y, &vertex.z);
168             vertices.push_back(vertex);
169         }
170         // read faces
171         for (int i = 0; i < face_num; ++i){
172             vec3<GLint> face;
173             int num;
174             int matches = fscanf(file, "%d %d %d %d\n", &num, &face.x, &face.y, &
                    ↪ face.z);
175             faces.push_back(face);
176         }
177         fclose(file);
178         printf("Finish loading off file.\n");
179         return true;
180     }
181
182     vector< vec3<GLfloat> > vertices;
183     vector< vec3<GLint> > faces;
184     int fmode;
185 };
186
187 #endif // MESH_H
```

mesh.cpp核心显示代码。

```
1 #include <windows.h> // must be the first one to be included!
2 #include "mesh.h"
```

```
3  #include <GL/glut.h>
4  #include <cmath>
5  #include <vector>
6  using namespace std;
7
8  #define WIN_WIDTH 600
9  #define WIN_HEIGHT 600
10
11 static Mesh model;
12 static GLfloat angle = 0.0f;
13 static GLfloat pos_x = 0.0f;
14 static GLfloat pos_y = 0.0f;
15 static int mode = 1;
16
17 // init & reshape function are referenced from
18 // https://www.ntu.edu.sg/home/ehchua/programming/opengl/CG_Examples.html
19 void init(void)
20 {
21     glClearColor(0.0f, 0.0f, 0.0f, 1.0f); // Set background color to black and
       ↪ opaque
22     glColor3f(1.0,1.0,1.0); // white
23     glPointSize(2.0);
24     glClearDepth(1.0f);      // Set background depth to farthest
25     glEnable(GL_DEPTH_TEST); // Enable depth testing for z-culling
26     glDepthFunc(GL_LEQUAL); // Set the type of depth-test
27     glShadeModel(GL_SMOOTH); // Enable smooth shading
28     glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST); // Nice perspective
       ↪ corrections
29 }
30
31 void myDisplay()
32 {
33     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
34     glMatrixMode(GL_MODELVIEW);  // To operate on model-view matrix
35
36     // Render a color-cube consisting of 6 quads with different colors
37     glLoadIdentity();              // Reset the model-view matrix
38
39     if (model.fmode != 3)
40         glTranslatef(pos_x, pos_y, -3.0f);
41     else
42         glTranslatef(pos_x, -10.0f, -300.0f);
43     glRotatef(angle, 0.0f, 1.0f, 0.0f); // Rotate about (0,1,0)-axis
44
45     for (auto face : model.faces) {
```

```
46          vec3<GLfloat> v1;
47          vec3<GLfloat> v2;
48          vec3<GLfloat> v3;
49          if (model.fmode == 1){ // obj
50              v1.set(model.vertices[face.x - 1]);
51              v2.set(model.vertices[face.y - 1]);
52              v3.set(model.vertices[face.z - 1]);
53          } else { // ply, off
54              v1.set(model.vertices[face.x]);
55              v2.set(model.vertices[face.y]);
56              v3.set(model.vertices[face.z]);
57          }
58          if (mode == 1){
59              glColor3f(1,1,1);
60              glBegin(GL_LINES);
61              glVertex3f(v1.x,v1.y,v1.z);
62              glVertex3f(v2.x,v2.y,v2.z);
63              glVertex3f(v1.x,v1.y,v1.z);
64              glVertex3f(v3.x,v3.y,v3.z);
65              glVertex3f(v2.x,v2.y,v2.z);
66              glVertex3f(v3.x,v3.y,v3.z);
67              glEnd();
68          }
69          else if (mode == 2) {
70              glBegin(GL_TRIANGLES);
71              glColor3f(0.8f,0.8f,0.8f);
72              glVertex3f(v1.x,v1.y,v1.z);
73              glColor3f(0.7f,0.7f,0.7f);
74              glVertex3f(v2.x,v2.y,v2.z);
75              glVertex3f(v3.x,v3.y,v3.z);
76              glEnd();
77          } else if (mode == 3) {
78              glColor3f(1,0,0);
79              glBegin(GL_LINES);
80              glVertex3f(v1.x,v1.y,v1.z);
81              glVertex3f(v2.x,v2.y,v2.z);
82              glVertex3f(v1.x,v1.y,v1.z);
83              glVertex3f(v3.x,v3.y,v3.z);
84              glVertex3f(v2.x,v2.y,v2.z);
85              glVertex3f(v3.x,v3.y,v3.z);
86              glEnd();
87              glColor3f(1,1,1);
88              glBegin(GL_TRIANGLES);
89              glColor3f(0.8f,0.8f,0.8f);
90              glVertex3f(v1.x,v1.y,v1.z);
```

```
91              glColor3f(0.7f,0.7f,0.7f);
92              glVertex3f(v2.x,v2.y,v2.z);
93              glVertex3f(v3.x,v3.y,v3.z);
94              glEnd();
95          }
96          // printf("Draw face %d %d %d\n",face.x,face.y,face.z);
97      }
98
99      glFlush();
100     printf("Done display!\n");
101 }
102
103 /* Handler for window re-size event. Called back when the window first appears and
104    whenever the window is re-sized with its new width and height */
105 void reshape(GLsizei width, GLsizei height) { // GLsizei for non-negative integer
106     // Compute aspect ratio of the new window
107     if (height == 0) height = 1;            // To prevent divide by 0
108     GLfloat aspect = (GLfloat)width / (GLfloat)height;
109
110     // Set the viewport to cover the new window
111     glViewport(0, 0, width, height);
112
113     // Set the aspect ratio of the clipping volume to match the viewport
114     glMatrixMode(GL_PROJECTION); // To operate on the Projection matrix
115     glLoadIdentity();           // Reset
116     // Enable perspective projection with fovy, aspect, zNear and zFar
117     if (model.fmode != 3)
118         gluPerspective(45.0f, aspect, 0.1f, 100.0f);
119     else
120         gluPerspective(45.0f, aspect, 100.0f, 500.0f);
121 }
122
123 void keyPressed(unsigned char key, int x, int y)
124 {
125     // int mod = glutGetModifiers(); // GLUT_ACTIVE_SHIFT
126     printf("Pressed %c! ", key);
127
128     switch (key){
129         case 'r':angle -= 5.0f;printf("Rotate clockwise");break;
130         case 'R':angle += 5.0f;printf("Rotate anticlockwise");break;
131         case 'w':pos_y += 0.1f;printf("Move up");break;
132         case 'a':pos_x -= 0.1f;printf("Move left");break;
133         case 's':pos_y -= 0.1f;printf("Move down");break;
134         case 'd':pos_x += 0.1f;printf("Move right");break;
135         case '1':mode = 1;printf("* Change to wireframe mode");break;
```

```
136         case '2':mode = 2;printf("* Change to flat mode");break;
137         case '3':mode = 3;printf("* Change to flat lines");break;
138     }
139 
140     printf("\n");
141     myDisplay();
142 }
143 
144 
145 int main(int argc, char *argv[])
146 {
147     glutInit(&argc, argv);
148 
149 #ifdef OBJ
150     model.loadOBJ("cow.obj");
151 #endif
152 #ifdef PLY
153     model.loadPLY("cactus.ply");
154 #endif
155 #ifdef OFF
156     model.loadOFF("Armadillo.off");
157 #endif
158 
159     glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
160 
161     glutInitWindowPosition(50, 50);
162     glutInitWindowSize(WIN_WIDTH, WIN_HEIGHT);
163 
164     glutCreateWindow("Polygon Mesh");
165 
166     glutDisplayFunc(myDisplay);
167     glutReshapeFunc(reshape);
168     glutKeyboardFunc(keyPressed);
169     init();
170 
171     // get into display
172     glutMainLoop();
173 
174     return 0;
175 }
```

编译指令如下：

```
g++ mesh.cpp -DOBJ=1 -I.\include -L.\lib -lglu32 -lglut32 -lopengl32 -o cow.exe
```

```
g++ mesh.cpp -DPLY=1 -I.\include -L.\lib -lglu32 -lglut32 -lopengl32 -o cactus.exe
g++ mesh.cpp -DOFF=1 -I.\include -L.\lib -lglu32 -lglut32 -lopengl32 -o armadillo.exe
```