Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 3

по курсу «Компьютерная Графика»

Студент группы ИУ9-42Б Павлов И. П.

Преподаватель Цалкович П.А.

Москва 2024

1 Задача

- Определить объемную фигуру в качестве модели сцены
- Реализовать поддержку поворота фигуры по нажатию на кнопки

В соответствии с вариантом было необходимо реализовать отображение эллипсоида.

2 Основная теория

Для выполнения задания было необходимо добавить эллипсоид, поддерживающий модельно-видовые преобразования. Для построения эллипсоида была использована его формула в параметрическом виде:

$x = a * \sin(\theta) * \cos(\phi)$
$y = b * \sin(\theta) * \sin(\phi)$
$z = c * \cos(\theta)$

Также, для поворота эллипсоида были использованы следующие матрицы:

- Матрица поворота относительно Ox: $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
- Матрица поворота относительно Oy: $\begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

Так как использовалось отображение без шейдеров, используется отображение полигонов с mode=GL_LINE. Сама фигура состоит из GL_TRIANGLE_STRIP. Для более легкого восприятия объема заданной фигуры на нее нанесен градиент.

3 Практическая реализация

3.1 Реализация main.cpp

```
#include <cmath>
#include "main.hpp"

float phi = 0.0f;
float theta = 0.0f;

void DrawEllipsoid(int sector_count, int stack_count) {
  float a = 0.7f;
  float b = 0.5f;
  float c = 0.5f;

  GLfloat x, y, z;

  float sector_step = 2.0f * M_PI / sector_count;
  float stack_step = M_PI / stack_count;
  float beta, alpha;

  glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
  glLineWidth(3.0f);
  for(int i = 0; i <= stack_count; ++i)
  {
    alpha = M_PI / 2 - i * stack_step;
    z = c * sinf(alpha);              // r * sin(u)

    glBegin(GL_TRIANGLE_STRIP);
    for(int j = 0; j <= sector_count; ++j)
    {
      glColor3f(
          1.0f,
          static_cast<float>(abs(sector_count / 2 - j)) / static_cast<float>(sector_count),
          static_cast<float>(i) / static_cast<float>(stack_count));
      beta = j * sector_step;
      // vertex position (x, y, z)
      x = a * cosf(alpha) * cosf(beta);      // r * cos(u) * cos(v)
      y = b * cosf(alpha) * sinf(beta);      // r * cos(u) * sin(v)
      glVertex3d(x, y, z);
    }
    glEnd();
  }
}

void DisplayWindow(GLFWwindow* window) {
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glLoadIdentity();
  glEnable(GL_DEPTH_TEST);
  glPushMatrix();

  GLfloat rotate_x[] = {
      1, 0, 0, 0,
      0, cosf(phi), -sinf(phi), 0,
      0, sinf(phi), cosf(phi), 0,
      0, 0, 0, 1
  };
```

```
  GLfloat rotate_y[] = {
      cosf(theta), 0, sinf(theta), 0,
      0, 1, 0, 0,
      -sinf(theta), 0, cosf(theta), 0,
      0, 0, 0, 1
  };

  glMatrixMode(GL_MODELVIEW);
  glMultMatrixf(rotate_x);
  glMultMatrixf(rotate_y);

  DrawEllipsoid(100, 200);

  glPopMatrix();
  glFlush();
  glfwSwapBuffers(window);
  glfwPollEvents();
}

void KeyCallback(GLFWwindow* window, int key, int scancode, int action, int mods) {
  if (action == GLFW_PRESS || action == GLFW_REPEAT) {
    if (key == GLFW_KEY_RIGHT) {
      theta += 0.1;
    } else if (key == GLFW_KEY_LEFT) {
      theta -= 0.1;
    } else if (key == GLFW_KEY_UP) {
      phi += 0.1;
    } else if (key == GLFW_KEY_DOWN) {
      phi -= 0.1;
    }
  }
}

int main() {
  if (!glfwInit()) {
    return 1;
  }

  GLFWwindow* window = glfwCreateWindow(750, 750, "Lab2",
                                        nullptr, nullptr);

  if (window == nullptr) {
    glfwTerminate();
    return 1;
  }

  glfwMakeContextCurrent(window);
  glfwSetKeyCallback(window, KeyCallback);

  glEnable(GL_DEPTH_TEST);
  glDepthFunc(GL_LESS);

  while(!glfwWindowShouldClose(window)) {
    DisplayWindow(window);
  }

  glfwDestroyWindow(window);
  glfwTerminate();
```

```
  return 0;
}
```

3.2 Реализация main.hpp

```
#ifndef PROGRAM_MAIN_HPP
#define PROGRAM_MAIN_HPP

#include <cstdlib>
#include "GLFW/glfw3.h"

int main();
void KeyCallback(GLFWwindow*, int, int, int, int);
void DrawEllipsoid(int, int);
void DisplayWindow(GLFWwindow*);

#endif //PROGRAM_MAIN_HPP
```

4 Заключение

Во время выполнения лабораторной был получен навык работы с объемными фигурами, заданными уравнениями. Реализовано отображение фигуры поворота (эллипсоида). Освоен навык наложения градиента на фигуры при помощи связи цвета с координатами точек.