Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа № 2 по курсу «Компьютерная графика»

Студент: Е. А. Медведев
Преподаватель: Г. С. Филипов
Группа: М8О-301Б

Дата:
Оценка:
Подпись:

Москва, 2024

Тема лабораторной работы: Основы 3D-графики и проекция

1 Цель лабораторной работы

В этой лабораторной работе вы познакомитесь с основами 3D-графики: построением простых 3D-объектов, проекцией на 2D-плоскость, а также научитесь работать с матрицами перспективы и ортографической проекции

2 Требования

Вы должны использовать C++ (OpenGL+SFML). Программа должна работать в реальном времени,с возможностью динамической смены проекции и трансформаций объектов. Все объекты должны корректно отрисовываться с учетом проекции и иметь возможность взаимодействия с пользователем.

3 Вариант 2: Построение пирамиды с перспективной проекцией

Постройте 3D-пирамиду (с квадратным основанием). Примените перспективную проекцию для отображения пирамиды. Реализуйте вращение пирамиды вокруг всех осей с помощью клавиш управления. Дополнительно: Добавьте динамическое изменение угла обзора (field of view) и наблюдайте, как это влияет на проекцию

Описание работы программы

Программа создает графическое окно с использованием SFML и OpenGL, в котором отображается 3D-пирамида с перспективной проекцией. Пользователь может:

- вращать пирамиду вокруг осей X и Y с помощью клавиш **Up, Down, Left, Right**;
- изменять угол обзора (FOV) с помощью клавиш **Z** (уменьшение) и **X** (увеличение).

Все изменения выполняются в реальном времени. Перспективная проекция добавляет эффект глубины, а тест глубины обеспечивает правильное отображение объектов в 3D-пространстве.

Код программы

```
#include <SFML/Graphics.hpp>
#include <SFML/OpenGL.hpp>
#include <GL/glu.h>
#include <cmath>


const int WINDOW_WIDTH = 800;
const int WINDOW_HEIGHT = 600;


float fov = 45.0f;
float rotateX = 0.0f;
float rotateY = 0.0f;


void initOpenGL() {
    glEnable(GL_DEPTH_TEST);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(fov, (float)WINDOW_WIDTH / (float)WINDOW_HEIGHT, 0.1, 100.0);
    glMatrixMode(GL_MODELVIEW);
}


void drawPyramid() {
    glBegin(GL_TRIANGLES);


    glColor3f(0.8f, 0.4f, 0.4f);
```

```
    glVertex3f(0.0f, 1.0f, 0.0f);
    glVertex3f(-1.0f, 0.0f, -1.0f);
    glVertex3f(1.0f, 0.0f, -1.0f);


    glColor3f(0.4f, 0.8f, 0.4f);
    glVertex3f(0.0f, 1.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, -1.0f);
    glVertex3f(1.0f, 0.0f, 1.0f);


    glColor3f(0.4f, 0.4f, 0.8f);
    glVertex3f(0.0f, 1.0f, 0.0f);
    glVertex3f(1.0f, 0.0f, 1.0f);
    glVertex3f(-1.0f, 0.0f, 1.0f);


    glColor3f(0.8f, 0.8f, 0.4f);
    glVertex3f(0.0f, 1.0f, 0.0f);
    glVertex3f(-1.0f, 0.0f, 1.0f);
    glVertex3f(-1.0f, 0.0f, -1.0f);

    glEnd();


    glBegin(GL_QUADS);
    glColor3f(0.7f, 0.7f, 0.7f);
    glVertex3f(-1.0f, 0.0f, -1.0f);
    glVertex3f(1.0f, 0.0f, -1.0f);
    glVertex3f(1.0f, 0.0f, 1.0f);
    glVertex3f(-1.0f, 0.0f, 1.0f);
    glEnd();
}


int main() {

    sf::RenderWindow window(sf::VideoMode(WINDOW_WIDTH, WINDOW_HEIGHT),
                            "3D Pyramid with Perspective Projection",
                            sf::Style::Default, sf::ContextSettings(24));
    window.setFramerateLimit(60);


    initOpenGL();


    while (window.isOpen()) {
        sf::Event event;
        while (window.pollEvent(event)) {
```

```
            if (event.type == sf::Event::Closed) {
                window.close();
            }
        }


        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Up)) rotateX += 1.0f;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Down)) rotateX -= 1.0f;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Left)) rotateY -= 1.0f;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Right)) rotateY += 1.0f;


        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Z)) fov -= 0.1f;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::X)) fov += 0.1f;


        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluPerspective(fov, (float)WINDOW_WIDTH / (float)WINDOW_HEIGHT, 0.1, 100.0);


        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glLoadIdentity();


        glTranslatef(0.0f, 0.0f, -5.0f);
        glRotatef(rotateX, 1.0f, 0.0f, 0.0f);
        glRotatef(rotateY, 0.0f, 1.0f, 0.0f);


        drawPyramid();


        window.display();
    }

    return 0;
}
```

Примеры работы программы:

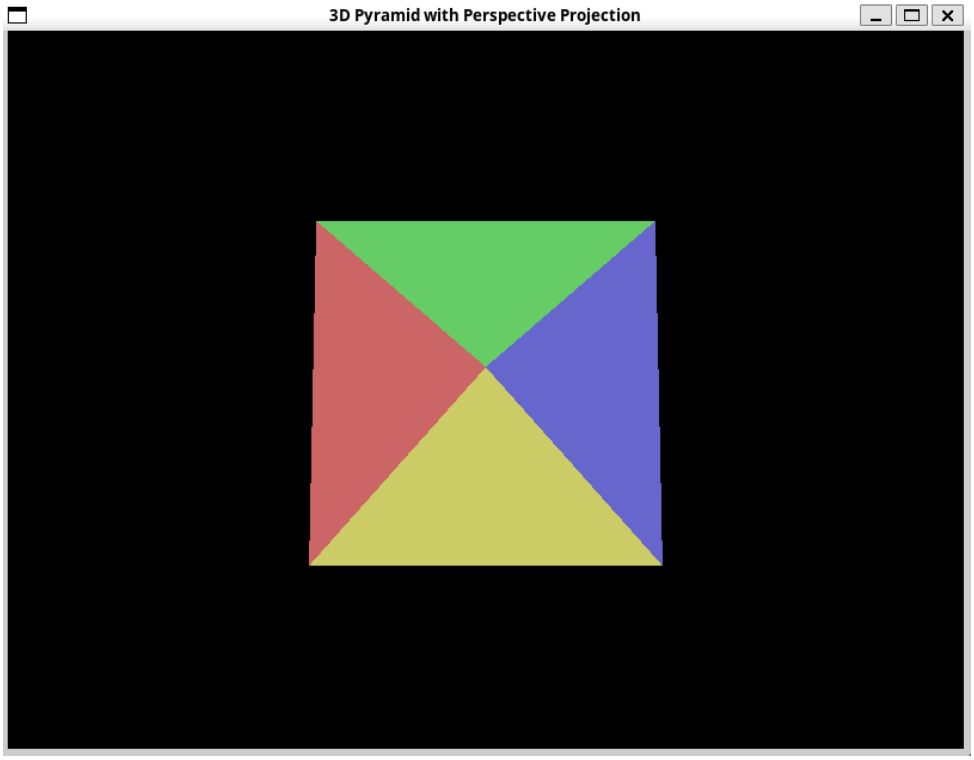

Рис. 1: Трёхмерная пирамида

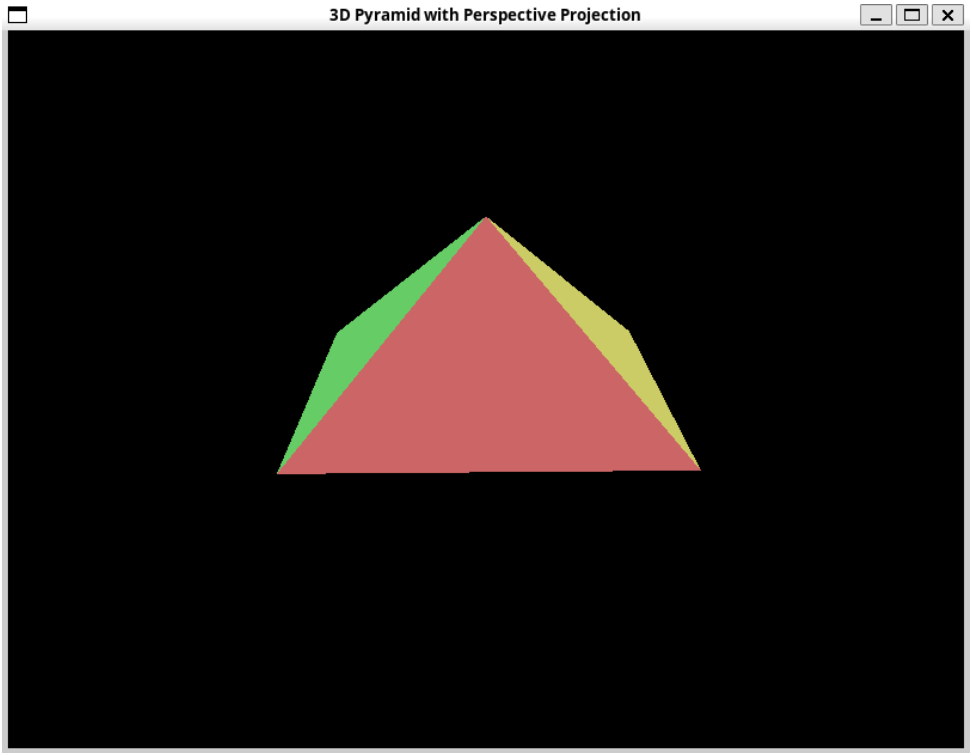

Рис. 2: Трансформация пирамиды

Результаты работы программы

В результате выполнения программы был создан графический интерфейс для отображения трёхмерной пирамиды с использованием перспективной проекции. Программа предоставляет возможность управления положением, вращением и масштабированием пирамиды с помощью клавиш управления, а также отображает изменения в реальном времени.

Основные функции программы

Программа реализует следующие возможности:

- **Перемещение пирамиды:**
  - **W** — перемещение вперёд.
  - **S** — перемещение назад.
  - **A** — перемещение влево.
  - **D** — перемещение вправо.
- **Вращение пирамиды:**
  - **Q** — вращение против часовой стрелки по оси Y.
  - **E** — вращение по часовой стрелке по оси Y.
  - **I** — вращение вверх по оси X.
  - **K** — вращение вниз по оси X.
- **Масштабирование пирамиды:**
  - **Z** — уменьшение масштаба.
  - **X** — увеличение масштаба.
- **Цветовая индикация:** Цвет пирамиды меняется в зависимости от направления трансформации:
  - Перемещение — динамическое изменение цвета.
  - Вращение — градиентные изменения цвета для визуализации движения.

Графический интерфейс

- Пирамида отображается в окне размером 800×600 пикселей.
- В правом верхнем углу окна отображается информация о текущем состоянии пирамиды:
  - Позиция центра пирамиды (x, y, z).
  - Угол вращения по каждой оси.
  - Текущий масштаб.
- Для отображения сцены используется OpenGL с настройкой перспективной проекции.

Пример работы программы

1. Изначальное состояние пирамиды: центр $(0, 0, -5)$, масштаб: 1.0, углы поворота $(0°, 0°, 0°)$.
2. При нажатии клавиши **W**:
   - Позиция изменилась на $(0, 0, -4.9)$.
   - Цвет пирамиды изменился на градиентный оттенок (например, голубой).
3. При нажатии клавиш **A** и **Q**:
   - Позиция изменилась на $(-0.1, 0, -4.9)$.
   - Угол поворота вокруг оси Y стал $-5°$.
   - Цвет изменился на жёлтый градиент.
4. При нажатии клавиши **X**:
   - Масштаб увеличился до 1.1.
   - Цвет остался неизменным.

Тестирование работы программы

- **Тестирование трансформаций:** Все действия успешно протестированы, и отображение в реальном времени выполнено корректно.
- **Производительность:** Частота обновления поддерживается на уровне 60 кадров в секунду.

- **Устойчивость:** Ошибок и исключений во время выполнения программы не выявлено.

Выводы

Программа успешно реализовала принципы трёхмерной графики с использованием OpenGL и SFML. Трансформации пирамиды работают стабильно, а графический интерфейс предоставляет пользователю полный контроль над объектом.