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 № 2

по курсу «Компьютерная графика»

«Модельно-видовые и проективные преобразования»

Студент группы ИУ9-42Б Волохов А. В.

Преподаватель Цалкович П. А.

Москва 2024

1 Задача

Необходимо:

1.Определить куб в качестве модели объекта сцены.

2.Определить преобразования, позволяющие получить заданный вид проекции (в соответствии с вариантом). Для демонстрации проекции добавить в сцену куб (в стандартной ориентации, не изменяемой при модельно-видовых преобразованиях основного объекта).

3.Реализовать изменение ориентации и размеров объекта (навигацию камеры) с помощью модельно-видовых преобразований (без gluLookAt). Управление производится интерактивно с помощью клавиатуры и/или мыши.

4.Предусмотреть возможность переключения между каркасным и твердотельным отображением модели (glFrontFace/ glPolygonMode).

2 Основная теория

В начале программы инициализируется библиотека GLFW для создания окна. Если инициализация прошла успешно, создается окно заданного размера. Затем устанавливается текущий контекст OpenGL для данного окна.

Функция display() отвечает за отрисовку сцены. Вначале происходит очистка буферов цвета и глубины. Затем устанавливается матрица проекции и вызывается функция Proj(), которая реализует фронтальную диметрию. После этого вызывается функция cube(), которая отрисовывает куб с помощью примитивов OpenGL.

Функция key callback() отвечает за обработку нажатий клавиш. При нажатии клавиш стрелок происходит изменение углов a и b, что влияет на положение куба в пространстве. Также при нажатии клавиши "F"происходит переключение между режимами отображения куба: сплошной заливкой или только границами.

3 Практическая реализация

Код представлен в Листинге 1.

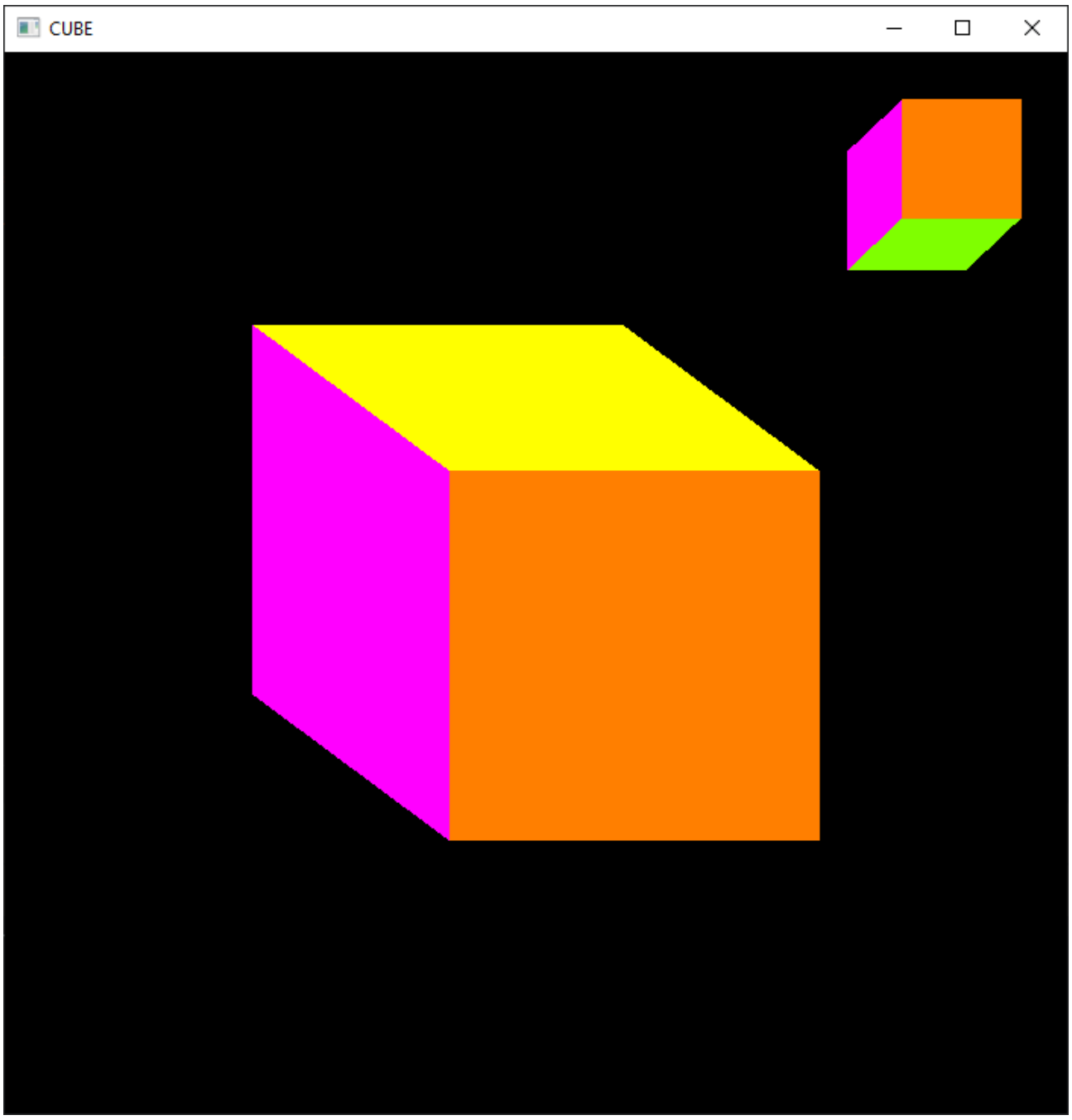
Листинг 1 - lab2.py

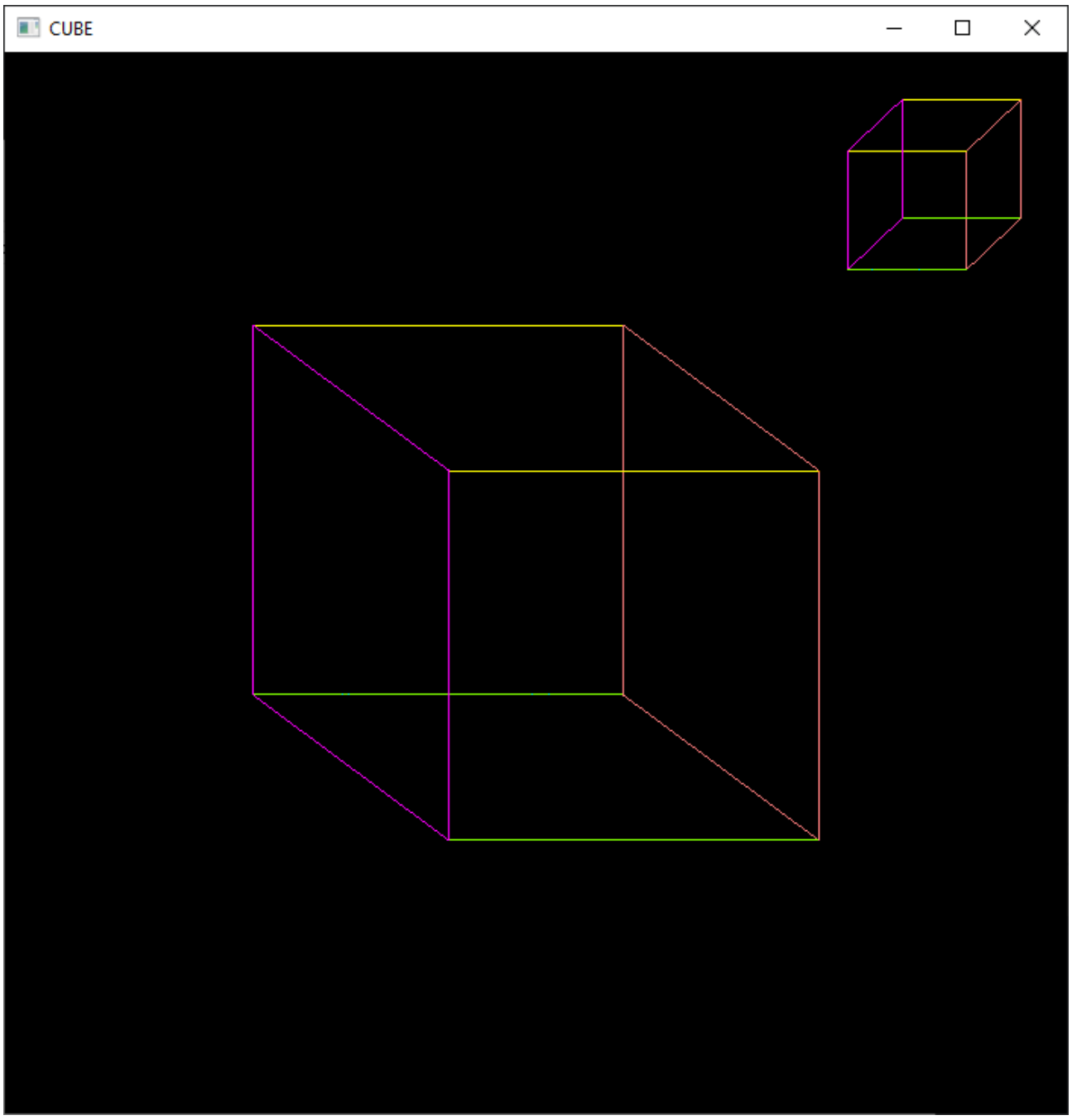
```
import glfw
from OpenGL.GL import *
from math import cos, sin

a = 0.0
b = 0.0
fill = True


def main():
    if not glfw.init():
        return
    window = glfw.create_window(700, 700, "CUBE", None, None)
    if not window:
        glfw.terminate()
        return
    glfw.make_context_current(window)
    glfw.set_key_callback(window, key_callback)

    glEnable(GL_DEPTH_TEST)
    glDepthFunc(GL_LESS)

    while not glfw.window_should_close(window):
        display(window)
    glfw.destroy_window(window)
    glfw.terminate()


def display(window):
    glLoadIdentity()
    glClear(GL_COLOR_BUFFER_BIT)
    glClear(GL_DEPTH_BUFFER_BIT)
    glMatrixMode(GL_PROJECTION)

    #
    glMultMatrixf([1, 0, 0, 0,
                   0, 1, 0, 0,
                   0, 0, 1, 0,
                   0.75, 0.75, 0, 1])
```

```
    t = -0.4
    p = 0.463648

    def Proj():  #
        glMultMatrixf([
            cos(p), 0, sin(p), 0,
            0, cos(p), -cos(p) * sin(t), 0,
                       sin(p) * cos(t), -sin(t), -cos(t) * cos(p), 0,
            0, 0, 0, 1,
        ])

    Proj()

    def cube(x):
        x = x / 2
        glBegin(GL_QUADS)
        glColor3f(1.0, 0.0, 1.0)
        glVertex3f(-x, -x, -x)
        glVertex3f(-x, x, -x)
        glVertex3f(-x, x, x)
        glVertex3f(-x, -x, x)
        glColor3f(1.0, 0.5, 0.5)
        glVertex3f(x, -x, -x)
        glVertex3f(x, -x, x)
        glVertex3f(x, x, x)
        glVertex3f(x, x, -x)
        glColor3f(0.5, 1.0, 0.0)
        glVertex3f(-x, -x, -x)
        glVertex3f(-x, -x, x)
        glVertex3f(x, -x, x)
        glVertex3f(x, -x, -x)
        glColor3f(1.0, 1.0, 0.0)
        glVertex3f(-x, x, -x)
        glVertex3f(-x, x, x)
        glVertex3f(x, x, x)
        glVertex3f(x, x, -x)
        glColor3f(0.0, 1.0, 1.0)
        glVertex3f(-x, -x, -x)
        glVertex3f(x, -x, -x)
        glVertex3f(x, x, -x)
        glVertex3f(-x, x, -x)
        glColor3f(1.0, 0.5, 0.0)
        glVertex3f(-x, -x, x)
        glVertex3f(x, -x, x)
        glVertex3f(x, x, x)
        glVertex3f(-x, x, x)
```

```
            glEnd()

        cube(0.25)

        glLoadIdentity()

        global a
        global b
        t = -0.4 + a
        p = 0.463648 + b

        Proj()

        cube(0.8)

        glfw.swap_buffers(window)
        glfw.poll_events()


def key_callback(window, key, scancode, action, mods):
    global a
    global b
    if action == glfw.PRESS or action == glfw.REPEAT:
        if key == glfw.KEY_RIGHT:
            b += 0.05
        elif key == glfw.KEY_LEFT:
            b -= 0.05
        elif key == glfw.KEY_UP:
            a += 0.05
        elif key == glfw.KEY_DOWN:
            a -= 0.05
        elif key == glfw.KEY_F:
            global fill
            fill = not fill
            if fill:
                glPolygonMode(GL_FRONT_AND_BACK, GL_FILL)
            else:
                glPolygonMode(GL_FRONT_AND_BACK, GL_LINE)


if __name__ == "__main__":
    main()
```

В результате работы программы получился следующий вывод:


CUBE


CUBE

4 Заключение

В ходе выполнения лабораторной работы был разработан простой пример трехмерной графики с использованием библиотеки OpenGL. Были реализованы основные элементы, такие как инициализация окна, отрисовка объектов и обработка клавиш для управления сценой. Полученные навыки могут быть полезны при дальнейшем изучении трехмерной графики и разработке графических приложений