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 No 3

по курсу «Компьютерная графика»

Студент группы ИУ9-41Б Пишикина М. В.

Преподаватель Цалкович П. А.

Москва 2024

1 Задача

1. Определить круговой тор в качестве модели объекта сцены.

2. Добавить в сцену круговой тор (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).

Вид проекции - горизонтальная изометрия

2 Основная теория

- glPushMatrix() #сохраняет текущую матрицу моделирования
- glPopMatrix() # восстанавливает текущую матрицу моделирования
- glGetIntegerv (GL_MAX_MODELVIEW_STACK_DEPTH) #определение глуб
- glGetIntegerv (GL_MODELVIEW_STACK_DEPTH) #определение текущей гл
- Преобразования объектов и камеры в OpenGL производятся с помощью умножения векторов координат на текущую матрицу в момент определения координат вершин

3 Практическая реализация

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

a, b, c = 0.06, 0.06, 0.06
alpha, beta = 0, 0
fill = True
automatic_rotation = False
rotation_speed = 1.0
```

```
def main():
    if not glfw.init():
        return
    window = glfw.create_window(600, 600, "лаба №3", 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)

    def rainbow_color(i, j, stacks, slices):
        r = cos(i / stacks)
        g = sin(j / slices)
        b = sin(i / stacks + j / slices)
        glColor3f(r, g, b)

    global a, b, c
    def hyperboloid():
```

```
    slices = 50
    stacks = 50

    for i in range(stacks):
        glBegin(GL_TRIANGLE_STRIP)
        for j in range(slices+1):
            u = 2.0 * pi * (i / stacks - 0.5)
            v = 2.0 * pi * j / slices

            x1 = a * cosh(u) * cos(v)
            y1 = b * cosh(u) * sin(v)
            z1 = c * sinh(u)

            x2 = a * cosh(u + 2*pi/stacks) * cos(v)
            y2 = b * cosh(u + 2*pi/stacks) * sin(v)
            z2 = c * sinh(u + 2*pi/stacks)

            rainbow_color(i, j, stacks, slices)
            glVertex3f(x1, y1, z1)

            rainbow_color(i+1, j, stacks, slices)
            glVertex3f(x2, y2, z2)
        glEnd()

global alpha, beta, automatic_rotation, rotation_speed

if automatic_rotation:
    alpha += rotation_speed

glRotate(alpha, 0, 0, 1)
glRotate(beta, 0, 1, 0)

hyperboloid()
```

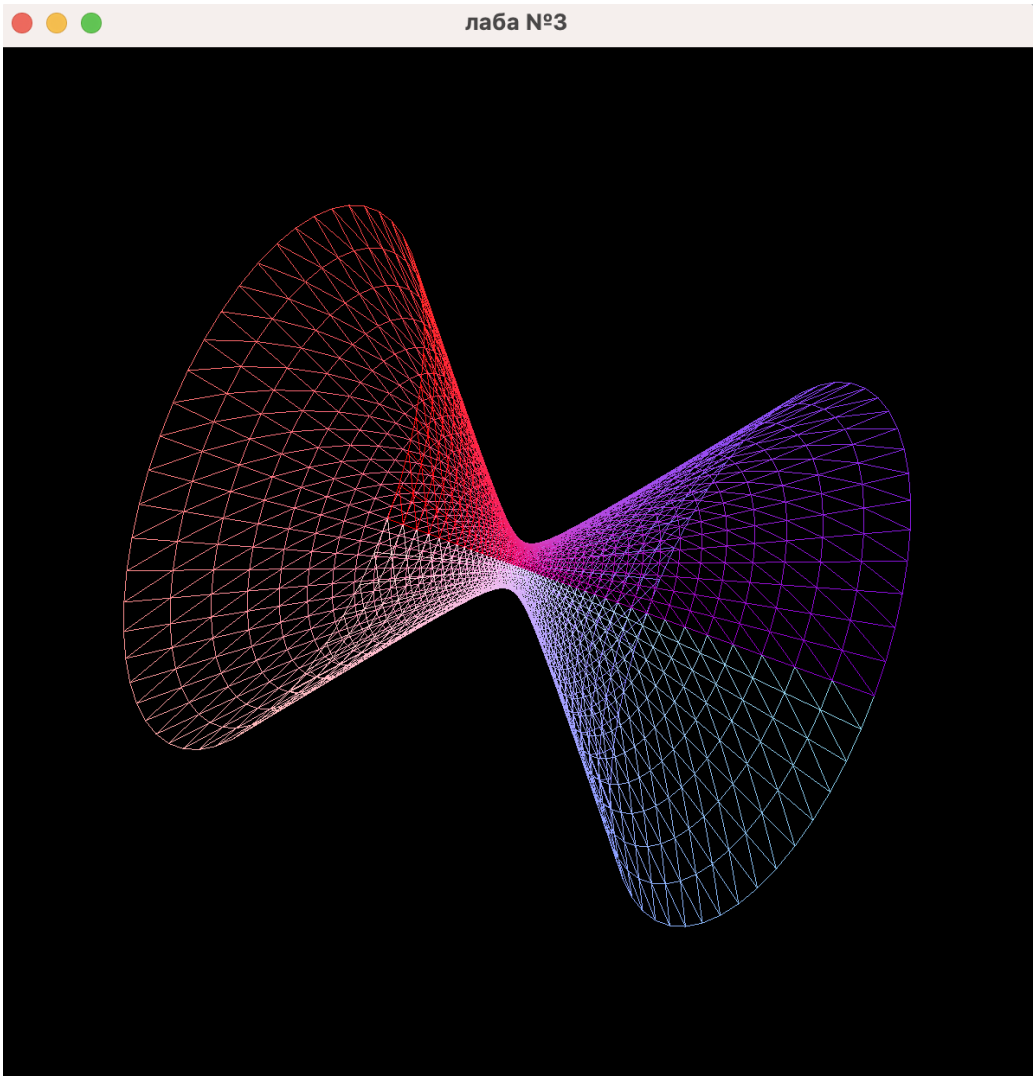
```
    glfw.swap_buffers(window)
    glfw.poll_events()


def key_callback(window, key, scancode, action, mods):
    global a, b, c, alpha, beta, automatic_rotation, rotation_speed
    if action == glfw.PRESS or action == glfw.REPEAT:
        if key == glfw.KEY_RIGHT:
            alpha -= 5
        elif key == glfw.KEY_DOWN:
            beta -= 5
        elif key == glfw.KEY_UP:
            beta += 5
        elif key == glfw.KEY_LEFT:
            alpha += 5
        elif key == glfw.KEY_EQUAL or key == glfw.KEY_KP_ADD:
            a *= 1.1
            b *= 1.1
            c *= 1.1
        elif key == glfw.KEY_MINUS or key == glfw.KEY_KP_SUBTRACT:
            a *= 0.9
            b *= 0.9
            c *= 0.9
        elif key == glfw.KEY_W:
            automatic_rotation = True
        elif key == glfw.KEY_S:
            automatic_rotation = False
        elif key == glfw.KEY_SPACE:
            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()
```

4 Пример работы программы (отправка на свою почту)



5 Заключение

В моей программе использовалась библиотека OpenGL, в которой реализованы основные функции для отрисовки фигур и их анимации. Мой вариант задания требовал реализации кругового сектора. Эта фигура должна была быть анимирована, а также должна была присутствовать неподвижная версия этой же фигуры. С помощью заданных матриц и соответствующих функций я успешно реализовала функционал анимации кругового сектора, его масштабирования, изменения цвета и заливки.