МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра САПР

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Компьютерная графика»

Тема: «Исследование математических методов представления и преобразования графических объектов на плоскости и в пространстве»

Студент гр. 8374 _____ Адаменко Е.А.

Студент гр. 8374 _____ Зелинский М.В.

Студент гр. 8374 _____ Стрелков А.Н.

Преподаватель _____ Матвеева И.В.

Санкт-Петербург

2021

Цель работы

Исследовать математические методы представления и преобразования графических объектов на плоскости. Задание №2 - Отображение плоского объекта, относительно прямой, которая задается двумя точками, с возможностью редактирования положения этих точек.

Теоретические положения

В качестве языка программирования был выбран Python с использованием библиотеки OpenGL для реализации рабочего поля интерфейса. OpenGL — это мощный программный интерфейс, применяемый для получения высококачественных, программно генерируемых изображений и интерактивных приложений, использующих двух- и трехмерные объекты.

Использованы были следующие функции:

```
# Использовать двойную буферизацию и цвета в формате RGB (Красный, Зеленый, Синий)
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB)
# Указание начальных размеров окна (ширина, высота)
glutInitWindowSize(600, 600)
# Указание начального положение окна относительно левого верхнего угла экрана
glutInitWindowPosition(50, 50)
glutInit(sys.argv)
# Создание окна с заголовком
glutCreateWindow(b"lab_1")
# Вызов функции инициализации
init()
glutDisplayFunc(draw)
# Запуск основного цикла
glutMainLoop()
```

Задание: отобразить плоский объект, относительно прямой, которая задается двумя точками, с возможностью редактирования положения этих точек. Выбран треугольник с заданными вершинами по умолчанию
(6, 2), (7, 3), (8, 1).

Для реализации задачи была использована следующая формула в матричном виде:

$$[T] = [T'][R][R'][R]^{-1}[T']^{-1}$$

где T' - матрица перемещения, R - матрица поворота вокруг начала координат, R' - матрица отражения.

Описание интерфейса

Get first point (format: x,y): // вводим первую точку прямой
Get second point (format: x,y): // вводим вторую точку прямой
Полученный результат выводится в отдельном окне lab_1
Исходный треугольник зеленого цвета, черный результат работы программы

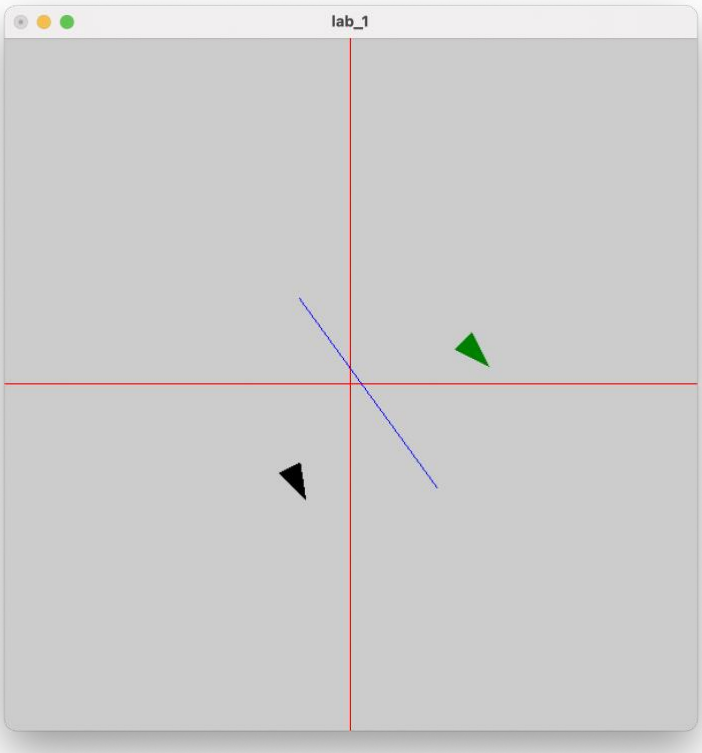

Рис. 1 пример вывода в окне

Ссылка на видео

Снимки экрана

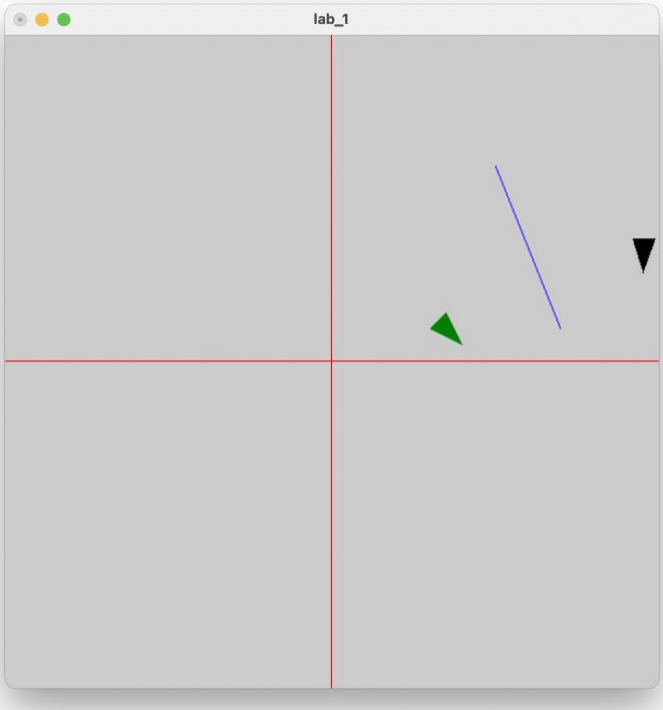

Рис. 2 итерация с точками прямой (10,12) (14,2)

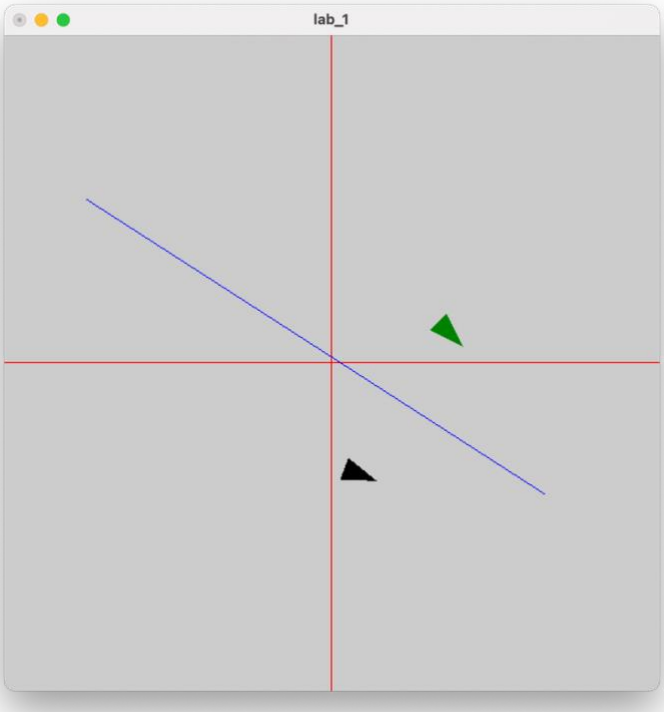

Рис. 3 итерация с точками прямой (-15,10) (13,-8)

Код программы

```
# encoding:utf-8
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
import math
import numpy as np


triangle = [[6, 2], [7, 3], [8, 1]] # координаты исходного треугольника



def init():

    global x1
    global x2
    global y1
    global y2
    point = raw_input('Get first point (format: x,y): ').split(',') # считываем точку в формате x,y
    x1 = float(point[0])
    y1 = float(point[1])

    point = raw_input('Get second point (format: x,y): ').split(',') # считываем точку в формате x,y
    x2 = float(point[0])
    y2 = float(point[1])
    glClearColor(0.8, 0.8, 0.8, 1.0) # цвет фона
    gluOrtho2D(-20.0, 20.0, -20.0, 20.0) # масштаб рисования

def init_axes(): # прорисовкая осей координат
    glBegin(GL_LINES)
    glColor(1, 0, 0)
    glVertex2f(-20, 0)
    glVertex2f(20, 0)
    glVertex2f(0, -20)
    glVertex2f(0, 20)
    glEnd()
    glBegin(GL_LINES)
    glColor(0, 0, 1)
    glVertex2f(x1, y1)
    glVertex2f(x2, y2)
    glEnd()
```

```
def get_axes_crossing_point_y(x1, x2, y1, y2):
    return (-x1*(y2-y1)/(x2-x1)) + y1

def get_arctan(x1,x2,y1,y2):
    return math.atan((y2-y1)/(x2-x1))

def get_rotate_triangle():
    initial_triangle = np.matrix([[8,1,1], [7,3,1],[6,2,1]])
    one_zero_zero = [1,0,0]
    zero_one_zero = [0,1,0]
    zero_minus_one_zero = [0,-1,0]
    zero_zero_one = [0,0,1]
    middle_matrix =np.matrix([one_zero_zero,zero_minus_one_zero,zero_zero_one])
    shift_y = get_axes_crossing_point_y(x1,x2,y1,y2)
    shift_y_1=np.matrix([one_zero_zero, zero_one_zero,[0,-shift_y,1]])
    shift_y_2=np.matrix([one_zero_zero, zero_one_zero,[0,shift_y,1]])
    alpha = get_arctan(x1,x2,y1,y2)
    line_1_r1 = [math.cos(alpha),-math.sin(alpha),0]
    line_2_r1 = [math.sin(alpha),math.cos(alpha),0]
    rotate_1 = np.array([line_1_r1,line_2_r1,zero_zero_one])
    line_1_r2 = [math.cos(alpha),math.sin(alpha),0]
    line_2_r2 = [-math.sin(alpha),math.cos(alpha),0]
    rotate_2 = np.matrix([line_1_r2,line_2_r2,zero_zero_one])
    result = initial_triangle*shift_y_1*rotate_1*middle_matrix*rotate_2*shift_y_2
    return result[:, :2].tolist()

def draw():
    mirrored_triangle = get_rotate_triangle()
    glClear(GL_COLOR_BUFFER_BIT)
    glBegin(GL_POLYGON)
    glColor(0, 0.5, 0)
    print(triangle)
    for p in triangle:
        glVertex2f(p[0], p[1])
    glEnd()
    glBegin(GL_POLYGON)
    glColor(0, 0, 0)
    for p in mirrored_triangle:
        glVertex2f(p[0], p[1])
    glEnd()
```

```
    init_axes()
    glutSwapBuffers() # выводим все изменения на экран


# Здесь начинается выполнение программы

# Использовать двойную буферизацию и цвета в формате RGB (Красный, Зеленый, Синий)
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB)
# Указываем начальный размер окна (ширина, высота)
glutInitWindowSize(600, 600)
# Указываем начальное положение окна относительно левого верхнего угла экрана
glutInitWindowPosition(50, 50)
glutInit(sys.argv)
# Создаем окно с заголовком
glutCreateWindow(b"lab_1")

# Вызываем нашу функцию инициализации
init()
glutDisplayFunc(draw)

# Запускаем основной цикл
glutMainLoop()
```

Вывод

В ходе выполнения лабораторной работы мы освоили основы OpenGl с использованием языка программирования Python. Изучены методы отражения плоского объекта (в нашем случае треугольник). В результате получен интерфейс, реализующий отображение плоского объекта(треугольника), относительно прямой, которая задается двумя точками.