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

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Компьютерная графика»

Тема: «Построение фракталов»

Вариант 23

Студент гр. 7383 _____ Власов Р.А.

Преподаватель _____ Герасимова Т.В.

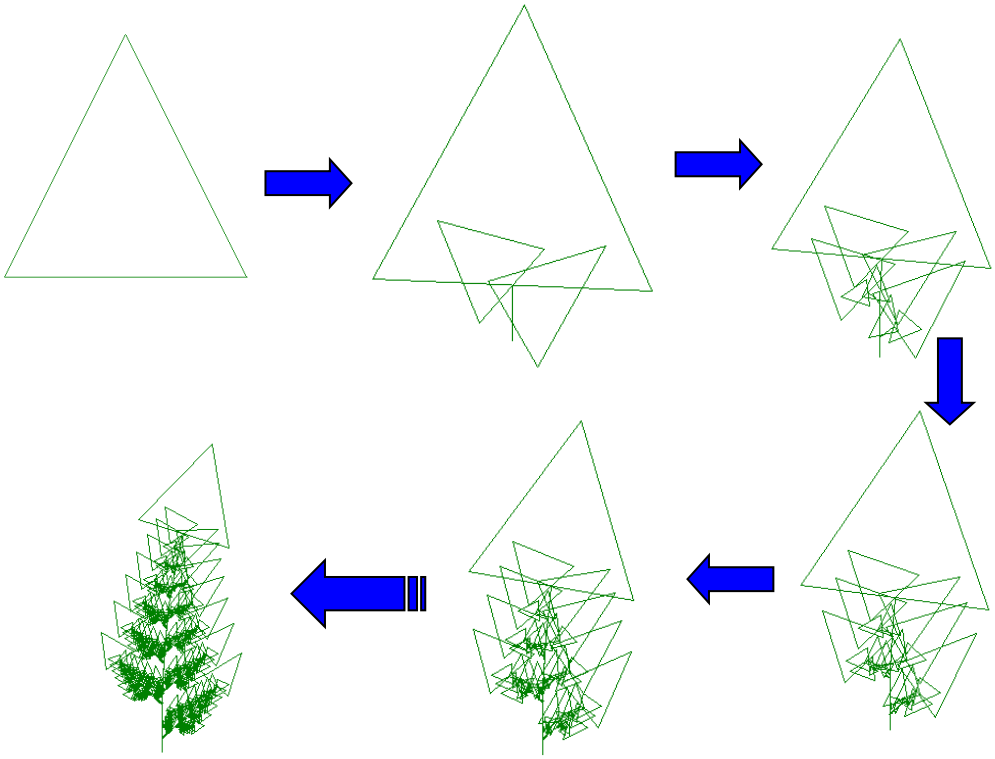
Санкт-Петербург

2020

Задание.

На базе предыдущей лабораторной работы разработать программу реализующую фрактал по индивидуальному заданию.

Задание 23: IFS-фракталы



Общие сведения.

Фрактал (лат. fractus — дробленый) — термин, означающий геометрическую фигуру, обладающую свойством самоподобия, то есть составленную из нескольких частей, каждая из которых подобна всей фигуре целиком.

Существует большое число математических объектов называемых фракталами (треугольник Серпинского, снежинка Коха, кривая Пеано, множество Мандельброта). Фракталы с большой точностью описывают многие физические явления и образования реального мира: горы, облака, турбулентные (вихревые) течения, корни, ветви и листья деревьев, кровеносные сосуды, что далеко не соответствует простым геометрическим фигурам.

Одним из способов построения фракталов является система итерирующих функций IFC. Применение таких преобразований, которые дают ту фигуру, которую необходимо. Система итерирующих функций — это совокупность сжимающих аффинных преобразований. Как известно, аффинные преобразования включают в себя масштабирование, поворот и параллельный перенос. Афинное преобразование считается сжимающим, если коэффициент масштабирования меньше единицы.

Ход работы.

Программа разработана на языке программирования Python. Графический интерфейс реализован с помощью библиотеки Qt.

Для успешного отображения изображения был реализован класс OpenGLView. В качестве родительского класса взят QGLWidget. Перепределены методы initializeGL, resizeGL и paintGL, отвечающие за инциализацию, изменение размера виджета и рисование изображения.

Также, для отображения фрактала понадобились следующие методы:

1. Метод, осуществляющий поворот точки вокруг начала координат.

```
def rotateLeftAroundCenter(self, point, angle):
    return [point[0] * cos(angle * pi / 180) - point[1] *
sin(angle * pi / 180),
            point[0] * sin(angle * pi / 180) + point[1] *
cos(angle * pi / 180)]
```

2. Метод, сдвигающий точку на вектор.

```
def movePoint(self, point, moveTo):
    return [point[0] + moveTo[0], point[1] + moveTo[1]]
```

3. Метод, рисующий треугольник.

```
def triangle(self, p, length, angle):
    h = length * sqrt(3) / 2
    a = self.movePoint(self.rotateLeftAroundCenter([0, h], angle),
```

```
p)
    b = self.movePoint(self.rotateLeftAroundCenter([length/2,0],
angle), p)
    c = self.movePoint(self.rotateLeftAroundCenter([-length/2,0],
angle), p)
    glBegin(GL_LINE_LOOP)
    glVertex2fv(a)
    glVertex2fv(b)
    glVertex2fv(c)
    glEnd()
```

4. Рекурсивный метод, рисующий фрактал.

```
def fractal(self, point, length, angle, rotateLeft, rotateRight,
iterations):
    self.triangle(point, length, angle)
    if iterations == 0:
        return
    tmpP = point
    tmpA = angle
    glBegin(GL_LINES)
    for i in range(0, iterations):
        glVertex2fv(self.movePoint(self.rotateLeftAroundCenter([0,
0], tmpA), tmpP))
        tmpP = self.movePoint(self.rotateLeftAroundCenter([0, -
length * 9 / 32], tmpA), tmpP)
        glVertex2fv(tmpP)
        tmpA = tmpA + 2
    glEnd()
    for i in range(0, iterations):
        dif = self.rotateLeftAroundCenter([0, -min(i, 1) *
length / 5], angle)
        point = [point[0] + dif[0], point[1] + dif[1]]
        pNew = self.movePoint(self.rotateLeftAroundCenter([0 -
(length + i) / 15, 0], angle), point)
```

```
        self.fractal(pNew, length * 3 / 8, angle + rotateLeft + 2
* i, rotateLeft, rotateRight, iterations - 1)
        pNew = self.movePoint(self.rotateLeftAroundCenter([0 +
(length + i) / 15, -length * 9 / 64], angle), point)
        self.fractal(pNew, length * 3 / 8, angle - rotateRight + 2
* i, rotateLeft, rotateRight, iterations - 1)
```

Тестирование.

Программа протестирована в операционной системе Ubuntu 19.04.

Результаты тестирования представлены на рис. 1-10.

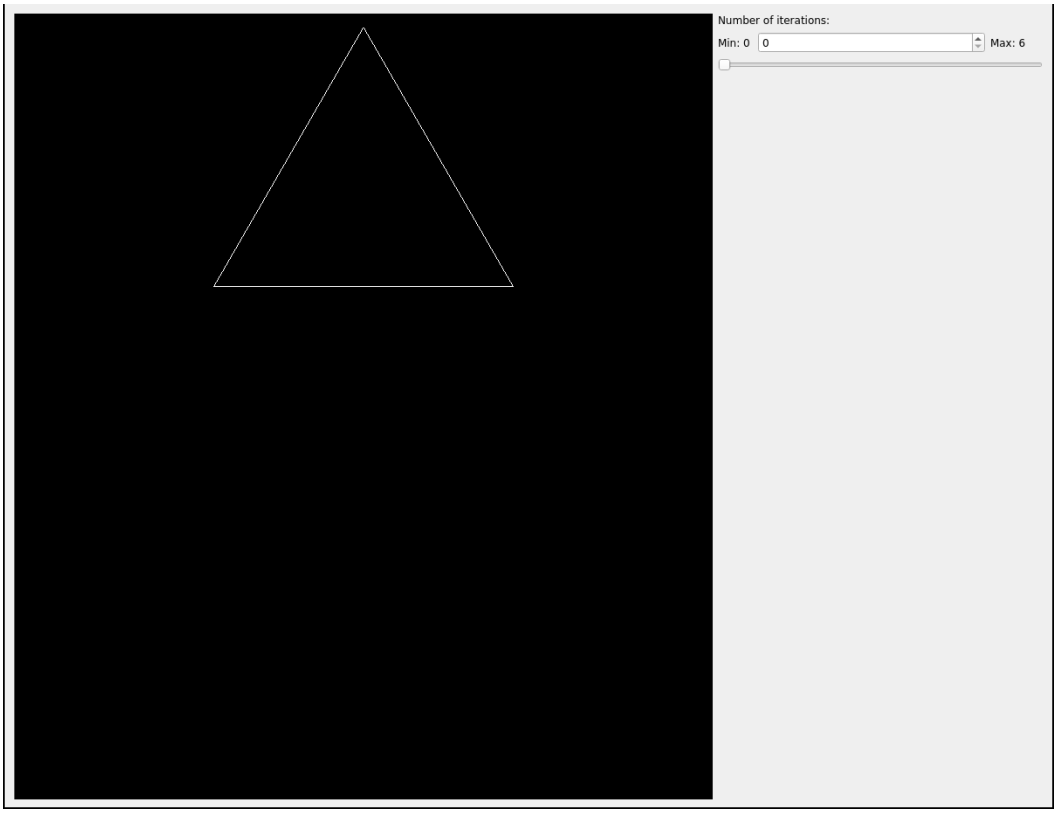


Рисунок 1 – Фрактал (0 итераций)

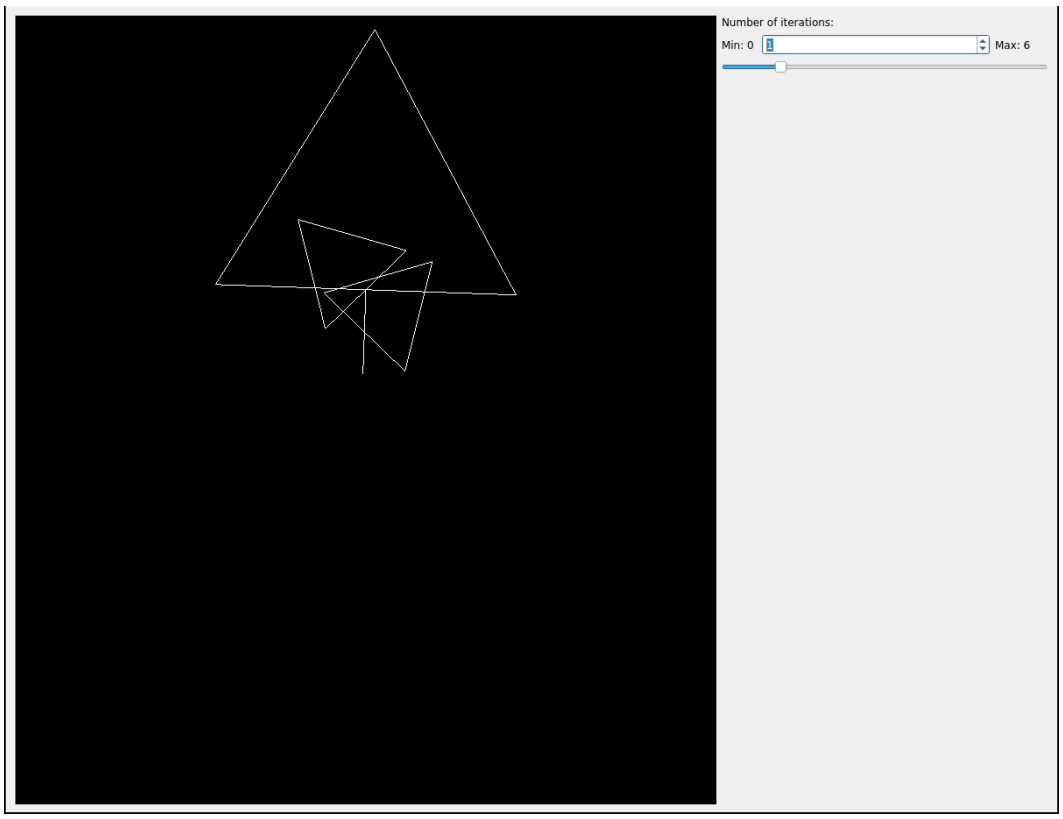


Рисунок 2 – Фрактал (1 итерация)

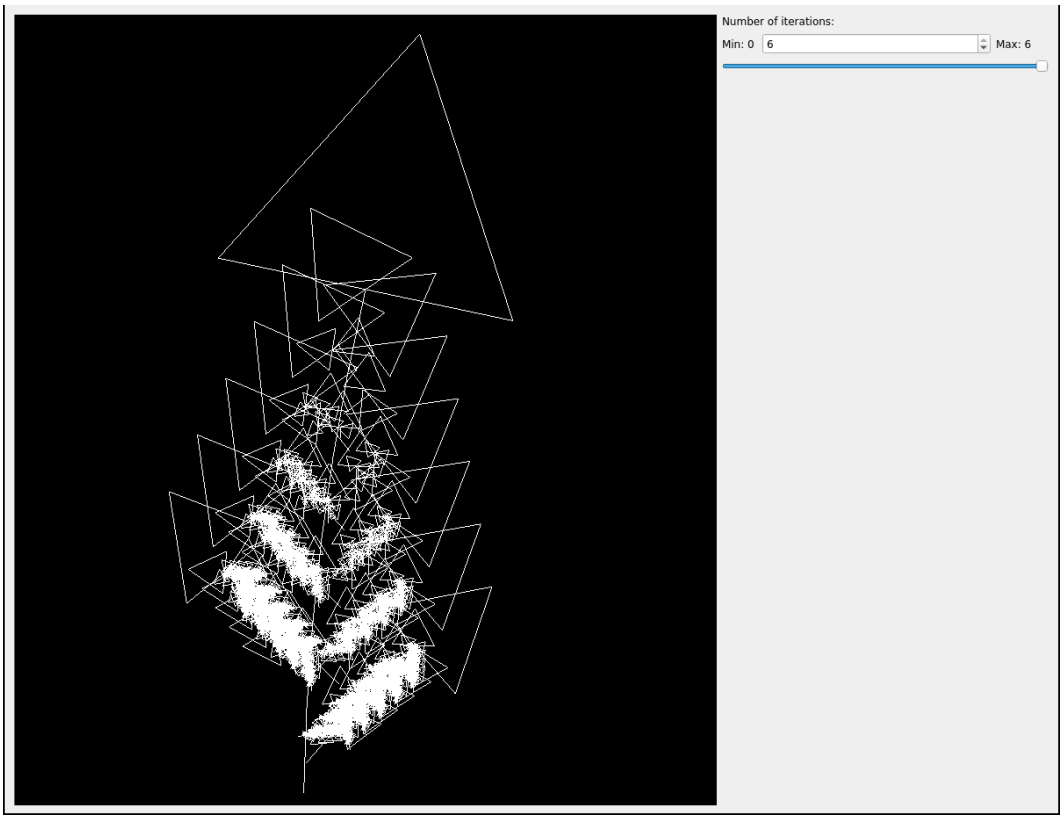


Рисунок 3 – Фрактал (6 итераций)

Вывод.

В результате выполнения лабораторной работы была разработана программа, реализующая отображение фрактала. Программа работает корректно. При выполнении работы были приобретены навыки работы с графической библиотекой OpenGL.