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: Кривая Безье.

Студент гр. 9304 _____ Попов Д.С.

Преподаватель _____ Герасимова Т.В.

Санкт-Петербург

2022

Задание

Реализовать интерактивное приложение, отображающее кривую Безье. При этом для кривых, состоящих из нескольких сегментов, должно быть обеспечено свойство непрерывной кривизны. Программа должна позволять пользователю: интерактивно менять положение контрольных точек, касательных, натяжений.

Общие сведения

Сплайны - это гладкие (имеющие несколько непрерывных производных) кусочно-полиномиальные функции, которые могут быть использованы для представления функций, заданных большим количеством значений и для которых неприменима аппроксимация одним полиномом. Так как сплайны гладки, экономичны и легки в работе, они используются при построении произвольных функций для:

- моделирования кривых;
- аппроксимации данных с помощью кривых;
- выполнения функциональных аппроксимаций;
- решения функциональных уравнений.

Важным их свойством является простота вычислений. На практике часто используют сплайны вида полиномов третьей степени. С их помощью довольно удобно проводить кривые, которые интуитивно соответствуют человеческому субъективному понятию гладкости.

В параметрической форме кубическая кривая Безье (n = 3) описывается следующим уравнением:

$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3t(1-t)^2\mathbf{P}_1 + 3t^2(1-t)\mathbf{P}_2 + t^3\mathbf{P}_3, \quad t \in [0,1]$$

Четыре опорные точки $\mathbf{P_0}$, $\mathbf{P_1}$, $\mathbf{P_2}$ и $\mathbf{P_3}$, заданные в 2-мерном пространстве, определяют форму кривой.

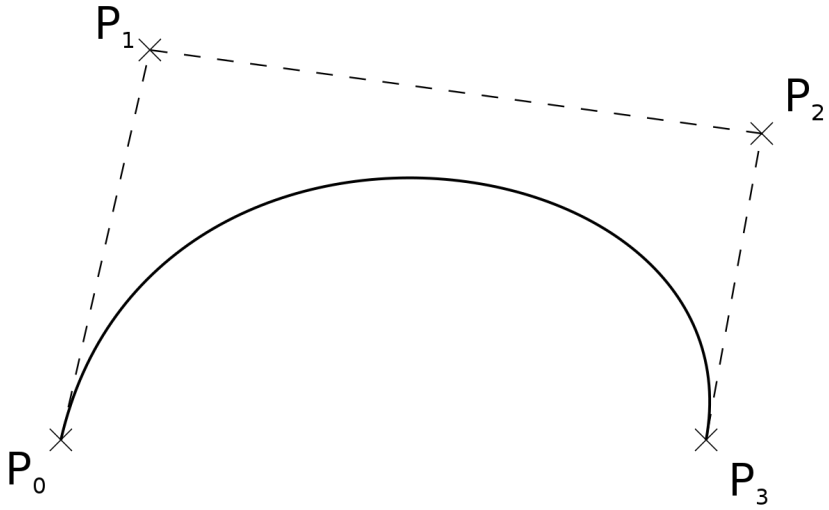


Рисунок 1 - Кубическая кривая Безье.

На рисунке 1 линия берёт начало из точки P_0, направляясь к P_1 и заканчивается в точке P_3, подходя к ней со стороны P_2. То есть, кривая не проходит через точки P_1 и P_2, они используются для указания её направления. Длина отрезка между P_0 и P_1 определяет, как скоро кривая повернёт к P_3.

В матричной форме кубическая кривая Безье записывается следующим образом:

$$\mathbf{B}(t) = [\,t^3 \quad t^2 \quad t \quad 1\,]\mathbf{M}_B \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}$$

где $\mathbf{M}_B$ называется базисной матрицей Безье:

$$\mathbf{M}_B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Выполнение работы

Работа выполнена в среде разработки Qt.

Базовый класс представления кривой Безье выглядит следующим образом:

```
#ifndef BEZIER_H
#define BEZIER_H

#include <QList>

#include "ifigure.h"

class Bezier : public IFigure
{
private:
    QList<QPoint> points;           ///< набор точек.
    GLint r;                        ///< радиус точек.
    GLint currentPoint;             ///< захваченная курсором точка.
    /**
     * @brief Отрисовка кривой Безье на основе точек из point
     */
    void drawBezierLine();
public:
    Bezier(QWidget* parent = nullptr);
    ~Bezier() = default;
    void createFigure(GLfloat size) override;
protected:
    virtual void mousePressEvent(QMouseEvent* e) override;
    virtual void mouseMoveEvent(QMouseEvent* e) override;
    virtual void mouseReleaseEvent(QMouseEvent* e)  override;
    virtual void paintGL() override;
    virtual void resizeGL(int, int) override;
};

#endif // BEZIER_H
```

В нем представлены перегруженные методы события мышки, которые отвечают за перетаскивание точек, а так же набор приватных полей, которые хранят в себе информацию о координатах этих точек и их радиус.

Определение перегруженного метода createFigure:

```
15  void Bezier::createFigure(GLfloat /*size*/)
16  {
17      glPointSize(r);
18      glBegin(GL_POINTS);
19
20          glColor4f(1.0f, 0.0f, 0.0f, 0.0f);
21
22          for (const auto& i : qAsConst(points))
23          {
24              glVertex3d(i.x(), i.y(), 0);
25          }
26
27      glEnd();
28
29      glLineWidth(2);
30      glBegin(GL_LINE_STRIP);
31
32          glColor4f(0.0f, 1.0f, 0.0f, 0.0f);
33
34          for (const auto& i : qAsConst(points))
35          {
36              glVertex3d(i.x(), i.y(), 0);
37          }
38
39      glEnd();
40
41      if (points.count() >= 3)
42          drawBezierLine();
43
44  }
```

В нем происходит отрисовка точек, а так же соединяющих их линии. Если кол-во точек больше 3-х, то отрисовывается кривая Безье.

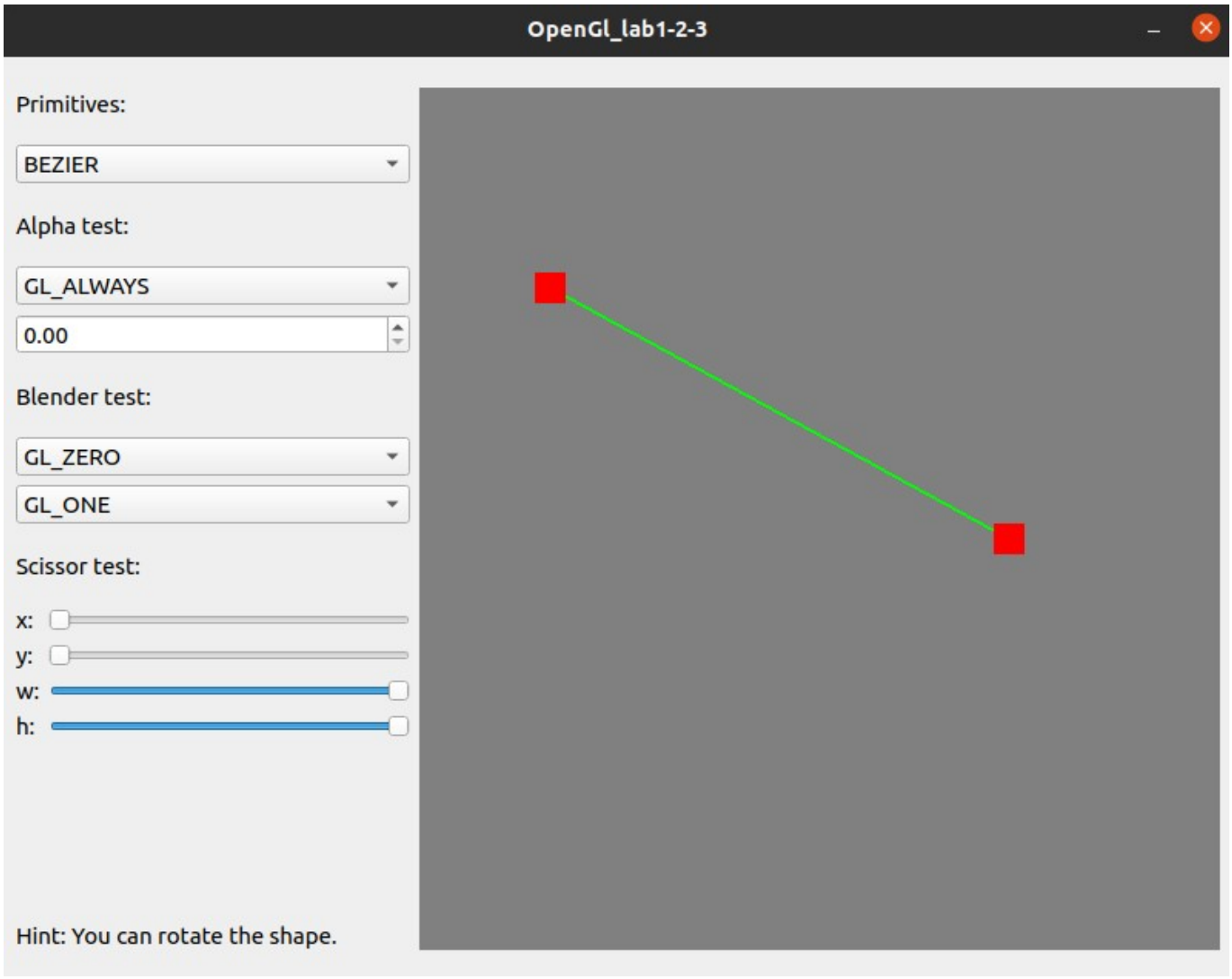
```
46  void Bezier::drawBezierLine()
47  {
48      glColor4f(0.0f, 0.0f, 1.0f, 0.0f);
49
50      glBegin(GL_LINE_STRIP);
51          float step = 1.0f / MAX_STEPS;
52
53          if (points.count() == 4)
54          {
55              for (float t = 0; t <= (1 + step); t += step)    ⚠ Variable 't' with floating point type 'float' should not be used as a loop counter [clang-analyzer-securit
56              {
57                  float x = pow((1 - t), 3) * points[0].x() + 3 * pow((1 - t), 2) * t * points[1].x() + 3 * (1 - t)* pow(t, 2) * points[2].x() + pow(t, 3) * points[3].x();
58                  float y = pow((1 - t), 3) * points[0].y() + 3 * pow((1 - t), 2) * t * points[1].y() + 3 * (1 - t)* pow(t, 2) * points[2].y() + pow(t, 3) * points[3].y();
59
60                  glVertex3d(x, y, 0);
61              }
62          }
63          else
64          {
65              for (float t = 0; t <= (1 + step); t += step)    ⚠ Variable 't' with floating point type 'float' should not be used as a loop counter [clang-analyzer-securit
66              {
67                  float x = pow((1 - t), 2) * points[0].x() + 2 * (1 - t) * t * points[1].x() + pow(t, 2) * points[2].x();
68                  float y = pow((1 - t), 2) * points[0].y() + 2 * (1 - t) * t * points[1].y() + pow(t, 2) * points[2].y();
69                  glVertex3d(x, y, 0);
70              }
71          }
72      glEnd();
73  }
```

Тестирование

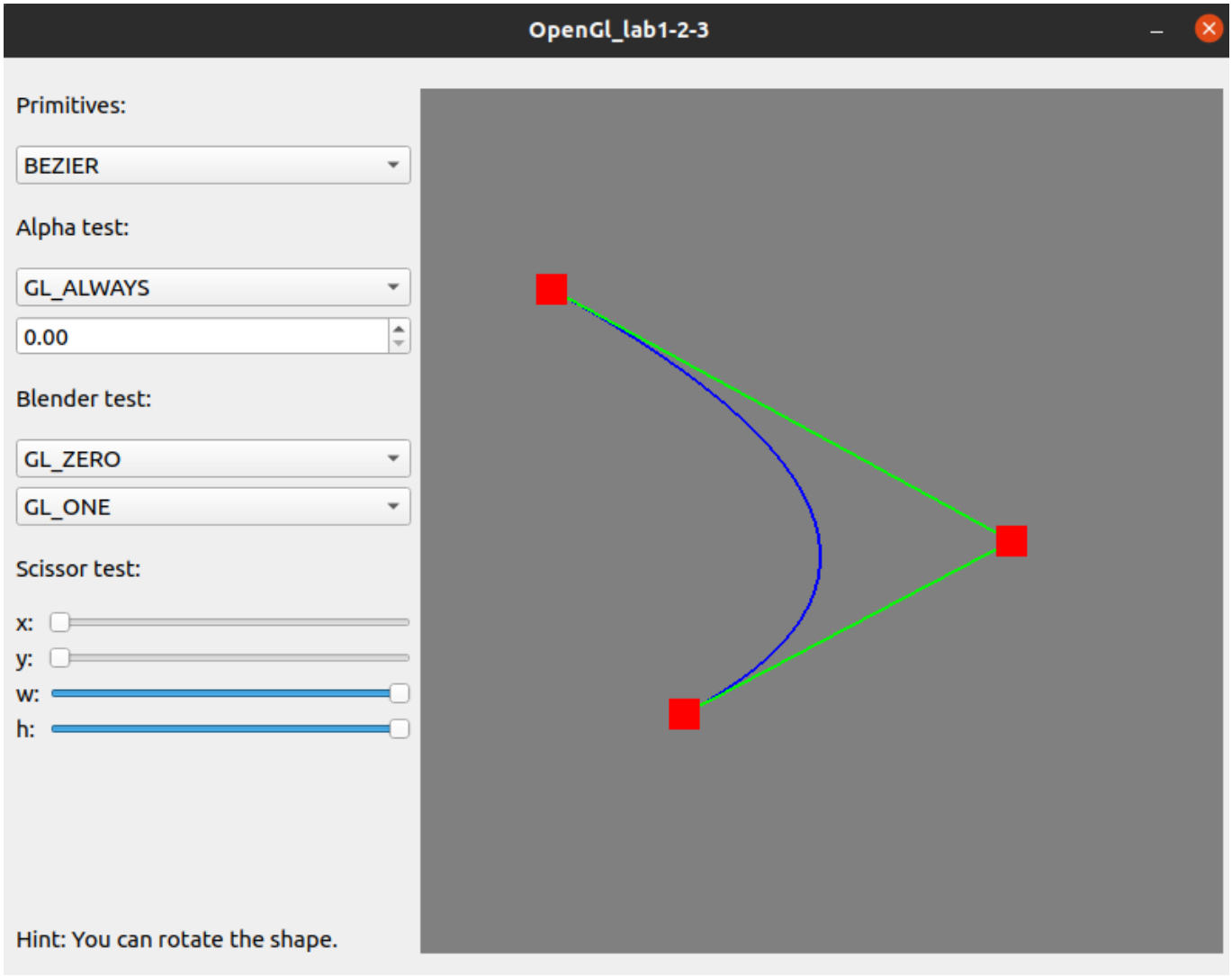
Результаты тестирования представлены на снимках экрана.

Выставление 1-ой точки:

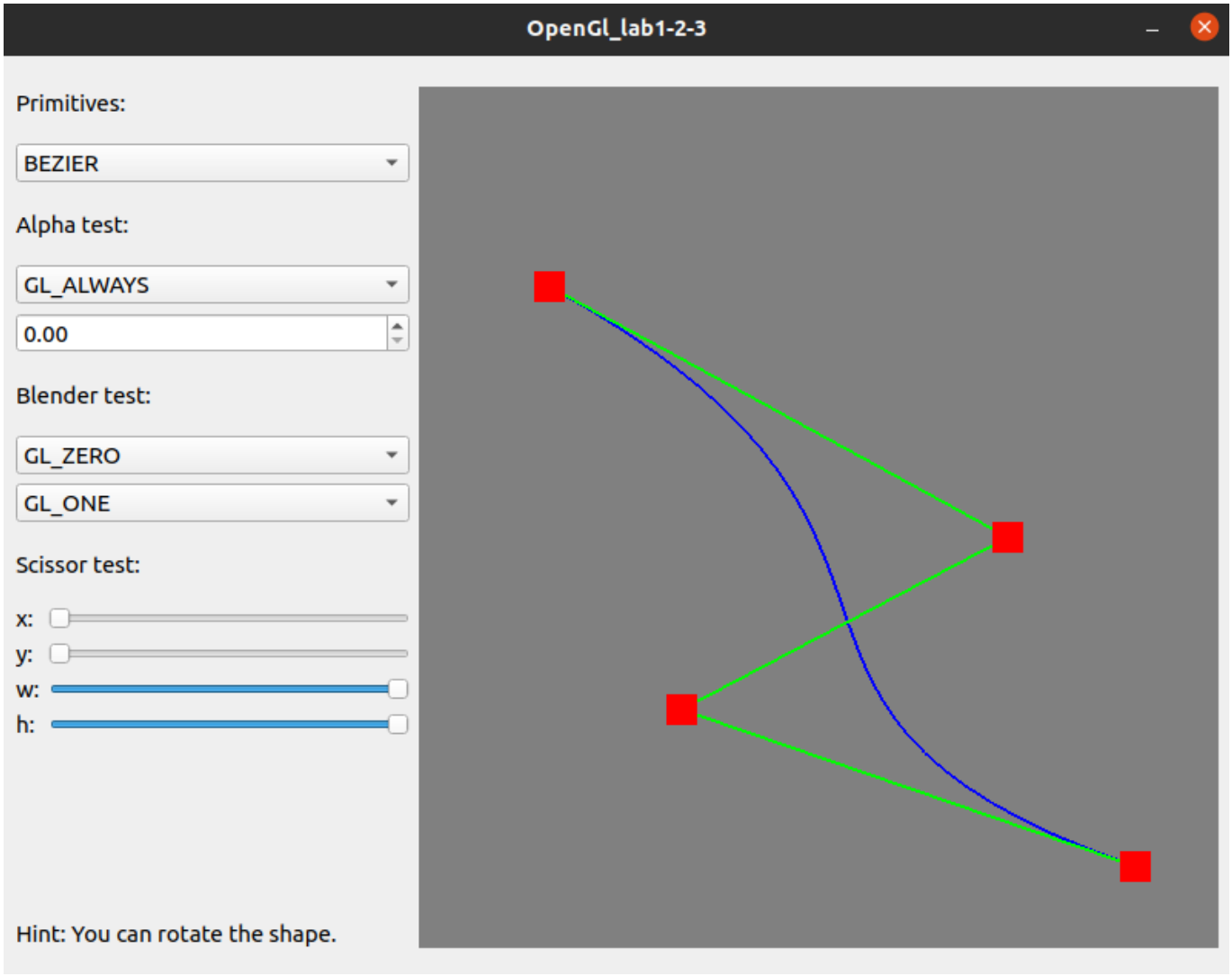
Выставление 2-х точек:



Выставление 3-х точек с отрисовкой кривой Безье:



Выставление 4-х точек с отрисовкой кривой Безье:



Вывод

В результате выполнения лабораторной работы была разработана программа, реализующая кривую Безье.