МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Компьютерная графика»
Тема: Формирования различных кривых с использованием ортогонального проектирования на плоскость визуализации (экране дисплея)

Студент гр. 9308	___________________	Степовик В.С. Соболев М.С.
Преподаватель	___________________	Матвеева И.В.

Санкт-Петербург
2022

Оглавление

Цель работы

Исследовать формирования различных кривых с использованием ортогонального проектирования на плоскость визуализации.

Задание

Сформировать на плоскости кривую Безье на основе задающей ломаной, определяемой 3 и большим количеством точек. Обеспечить редактирование координат точек задающей ломаной с перерисовкой сплайна Безье.

Используемые ресурсы

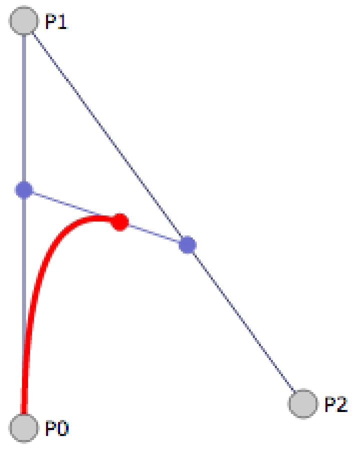
Для выполнения лабораторной работы использовался язык C++ и фреймворк Qt для визуализации.

Основные теоретические положения

Кривая Безье — это параметрическая кривая, определяемая набором точек, известных как контрольные точки. Она широко используется в компьютерной графике и смежных областях.

Выделим свойства кривой Безье:

1. непрерывность заполнения сегмента между начальной и конечной точками;
2. кривая всегда располагается внутри фигуры, образованной линиями, соединяющими контрольные точки;
3. при наличии только двух контрольных точек сегмент представляет собой прямую линию;
4. прямая линия образуется при коллинеарном (на одной прямой) размещении управляющих точек;
5. кривая Безье симметрична, то есть обмен местами между начальной и конечной точками (изменение направления траектории) не влияет на форму кривой;
6. масштабирование и изменение пропорций кривой Безье не нарушает её стабильности, поскольку с математической точки зрения она «аффинно-инвариантна»;
7. изменение координат хотя бы одной из точек ведет к изменению формы всей кривой Безье;
8. любой частичный отрезок кривой Безье также является кривой Безье;
9. степень (порядок) кривой всегда на одну ступень меньше числа контрольных точек. Например, при трёх контрольных точках форма кривой — парабола, так как парабола — кривая 2-го порядка;
10. окружность не может быть описана параметрическим уравнением кривой Безье.



Общее определение точки на кривой Безье выглядит следующим образом:

$$\mathbf{B}(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i \mathbf{P}_i$$

где n - степень кривой и $\binom{n}{i}$ биномиальные коэффициенты. Мы также можем представить это, как показано ниже.

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

Мы можем упростить основное уравнение

$$\mathbf{B}(t) = \sum_{i=0}^{n} b_{i,n}(t)\mathbf{P}_i, \quad 0 \le t \le 1$$

И получим базисные многочлены Бернштейна степени n.

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0, \ldots, n$$

Данное уравнение известно как полином Бернштейна, который представляет собой линейную комбинацию базисных полиномов.

Ход работы

Нам потребуется функция, которая возвращает вектор интерполированных точек Безье на всём диапазоне параметра t, где t может быть между 0 и 1 (оба включительно).

Входными данными для функции является список контрольных точек.

Разберем следующую функцию:

$$\mathbf{B}(t) = \sum_{i=0}^{n} b_{i,n}(t)\mathbf{P}_i, \quad 0 \le t \le 1$$

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0, \ldots, n$$

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

Реализация:

Из формул выше следует, что для каждой точки кривой Безье, зависящей от параметра t, справедлив представленный ниже алгоритм(код). В нём последовательно производится нахождение полинома Бернштейна для каждой контрольной точки, умножение координат данной точки на полином, с последующим и добавлением результата к результирующей координате.

```
void Bizye::countBizyePoints(QVector<QPoint> points){

    float t = 1.0f;
    double x,y;
    int N = points.size() - 1;
    while(!(t < -0.0f)){
        x = 0;
        y = 0;
        double bernPoly = 0;
        for(int k = N; k>=0; --k){

            double C = ((double)factorial(N)/(double)(factorial(k)*factorial(N-
k)));
            double p_t = powr(t,k);
            double p_mt = powr((1.0-t),N-k);
            bernPoly = C * p_t * p_mt;

            if(t==1){
                x = points[N].x();
                y = points[N].y();
            }
            else if(t==0){
                x = points[0].x();
                y = points[0].y();
            }
            else{
             x += points[k].x() * bernPoly;
             y += points[k].y() * bernPoly;
            };

    // qDebug()<< "t"<<t <<"koef" <<C << "(t^n)"<<p_t << "(1-t)"<<p_mt <<
"poly"<< bernPoly << "slogaemoe" << QPoint(points[k].x() *
bernPoly,points[k].y() * bernPoly);
        }
        qDebug() << QPointF(x,y) ;
        bizye_points.push_back(QPointF(x+5,y+5));
        t-=0.10f;
        if(t>0.098f && t<0.1f) t = 0.1f;//нюансы си
    }
}
```

Пример работы программы

Пример работы программы представлен на рисунках ниже:

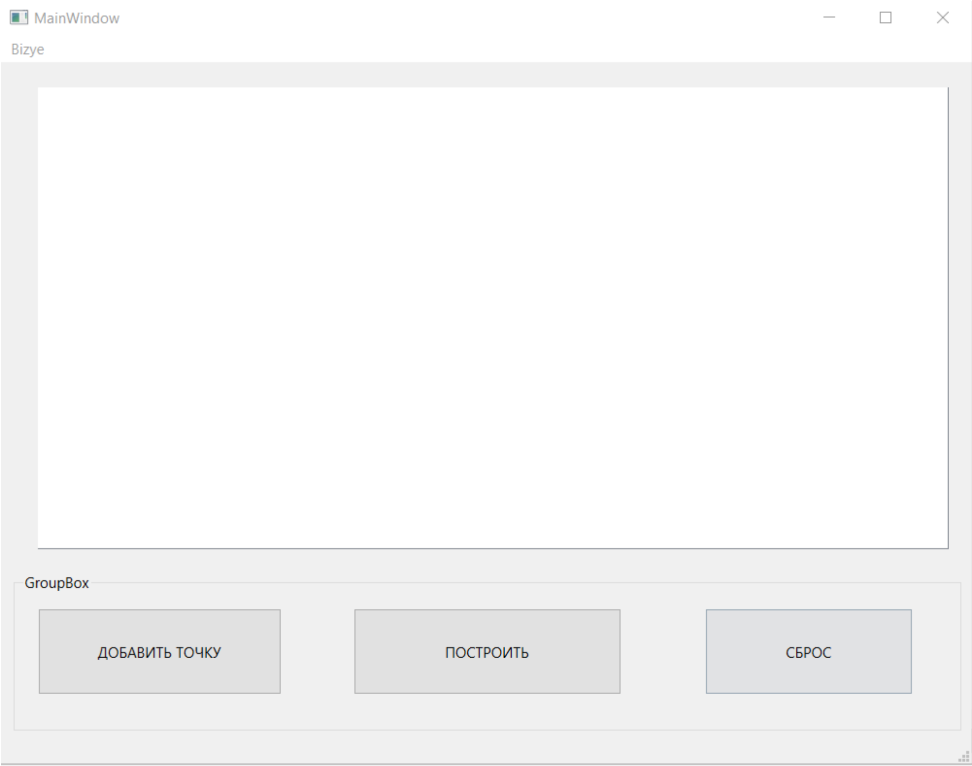


Рисунок 1. Начальное окно

Кнопка «ДОБАВИТЬ ТОЧКУ» добавляет точку на сцену.

Кнопка «ПОСТРОИТЬ» строит кривую Безье по указанным точкам и соединяет их линиями.

Кнопка «СБРОС» убирает со сцены все элементы графики.

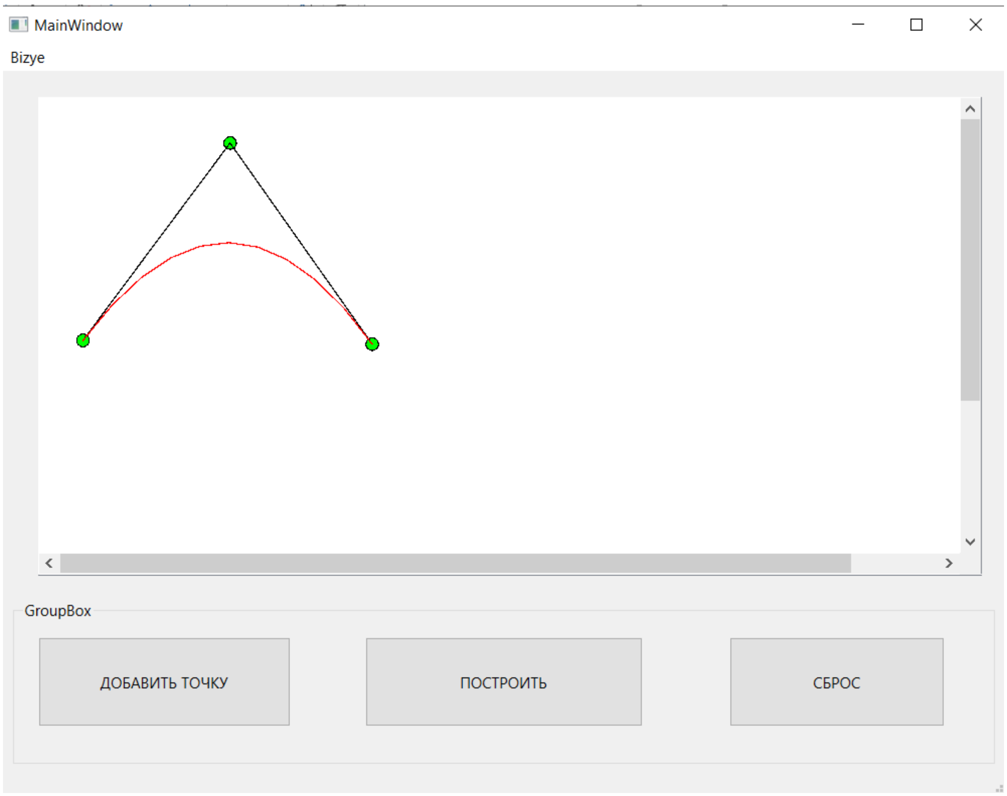


Рисунок 2. Кубическая кривая

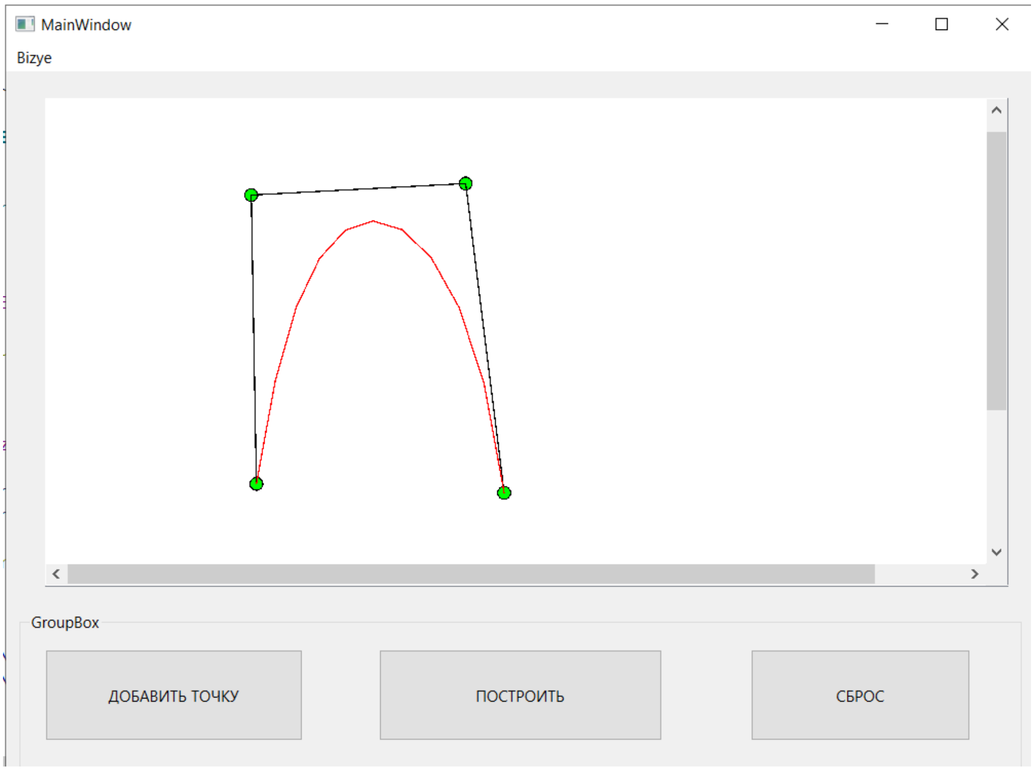


Рисунок 3. Кубическая кривая

Рисунок 4. Кривая 4го порядка.

Вывод

При выполнении лабораторной работы были изучены формирования различных кривых с использованием ортогонального проектирования на плоскость визуализации. В частности, исследована кривая Безье и ее построение на плоскости.