**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа №7**

по курсу «Компьютерная графика»

Тема: «Построение плоских полиномиальных кривых»

Студент: Мариничев И. А.

Группа: М8О-308Б-19

Преподаватель: Филиппов Г. С.

Оценка:

Москва

2021

**1. Постановка задачи.**

Написать программу, строящую полиномиальную кривую по заданным точкам. Обеспечить возможность изменения позиции точек и, при необходимости, значений касательных векторов и натяжения.

**Вариант №8**: Сегмент кривой Кэтмулла-Рома (Catmull-Rom)

**2. Описание программы.**

Сплайновая кривая Кэтмулла-Рома строится в соответствии со следующей формулой:

,

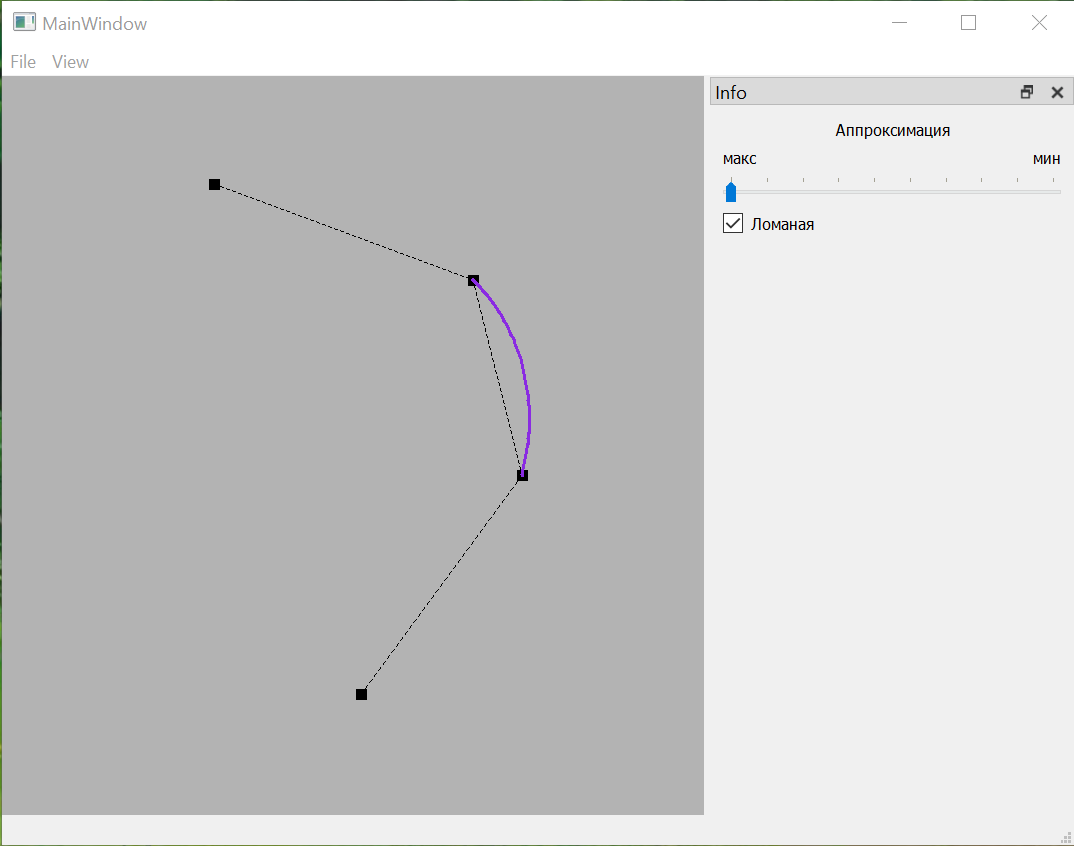
где — опорные точки, .

Для решения задачи я решил использовать C++ и фреймворк Qt, в котором использовал библиотеку QPainter.

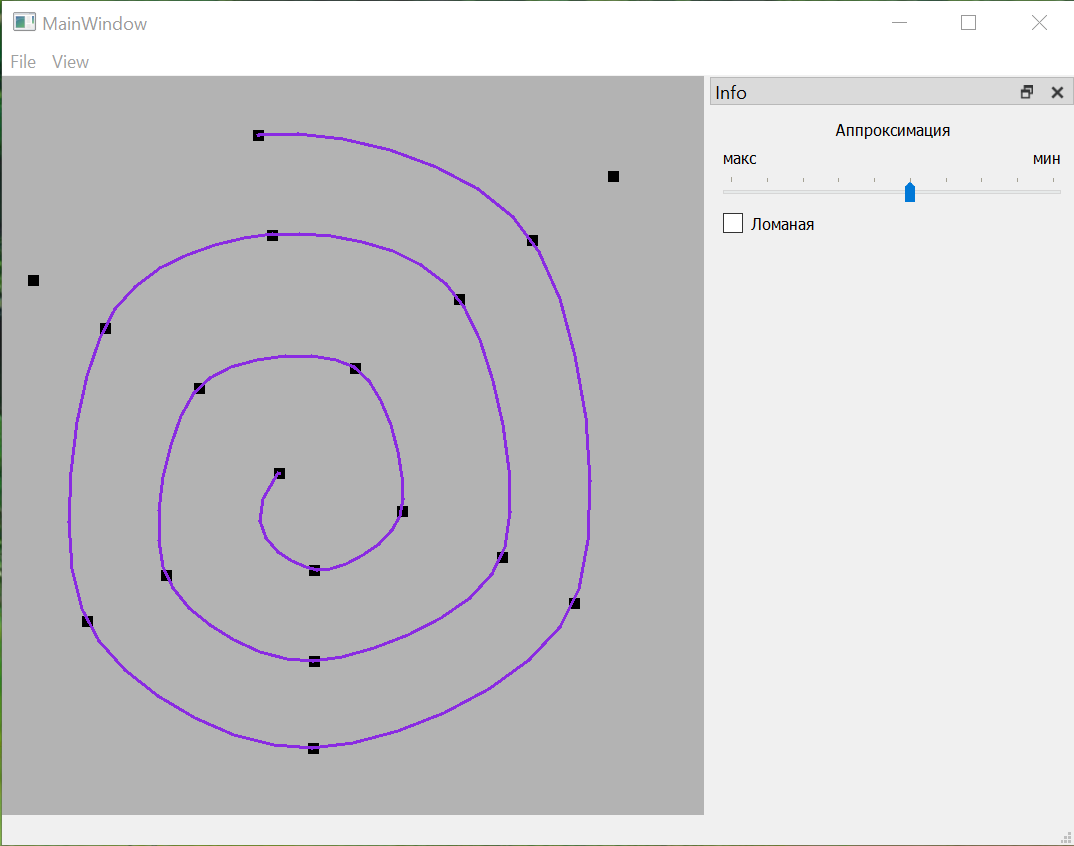
Для каждых двух точек между второй и предпоследней вычисляем координаты участка сплайна для t на отрезке от нуля до единицы, при этом задаём выпуклость линии относительно предыдущей точки и последующей.

**3. Демонстрация работы программы.**

1) Вид кривой для четырёх точек при максимальной аппроксимации и включённым соединением точек пунктирной ломаной линией.



2) Вид кривой для восемнадцати точек при средней аппроксимации и отключённым соединением точек пунктирной ломаной линией.

****

**4. Основной код программы.**

1) Метод для вычисления опорных точек и коэффициентов влияния соседних точек друг на друга для участка сплайна.

QVector2D display::**get\_spline\_point**(double t) {

int p0, p1, p2, p3;

p1 = (int)t + 1;

p2 = p1 + 1;

p3 = p2 + 1;

p0 = p1 - 1;

t -= (int)t;

double tt = t \* t;

double ttt = tt \* t;

*//* *influential* *field* *values*

float q1 = -ttt + 2.0f\*tt - t;

float q2 = 3.0f\*ttt - 5.0f\*tt + 2.0f;

float q3 = -3.0f\*ttt + 4.0f\*tt + t;

float q4 = ttt - tt;

float t\_x = 0.5f \* (points[p0].x() \* q1 + points[p1].x() \* q2 + points[p2].x() \* q3 + points[p3].x() \* q4);

float t\_y = 0.5f \* (points[p0].y() \* q1 + points[p1].y() \* q2 + points[p2].y() \* q3 + points[p3].y() \* q4);

*return* {t\_x, t\_y};

}

2) Метод для отрисовки сплайна Кэтмулла-Рома, ломаной линии и точек.

void display::***paintEvent***(QPaintEvent \*) {

QPainter ptr{*this*};

ptr.setPen(QColor(0, 0, 0));

*//* *draws* *dashed* *polyline*

*if* (cntPoints != 0 && displayAdditionalLines) {

ptr.setPen(Qt::*DashLine*);

*for* (unsigned int i = 0; i < cntPoints - 1; i++) {

ptr.drawLine(*static\_cast*<int>(points[i].x()) + SQUARE\_SIZE / 2,

*static\_cast*<int>(points[i].y()) + SQUARE\_SIZE / 2,

*static\_cast*<int>(points[i + 1].x()) + SQUARE\_SIZE / 2,

*static\_cast*<int>(points[i + 1].y()) + SQUARE\_SIZE / 2);

}

ptr.setPen(Qt::*SolidLine*);

}

ptr.setBrush(QColor(0, 0, 0));

*//* *draws* *square* *points* *of* *spline*

*for* (unsigned int i = 0; i < cntPoints; i++) {

QPolygon pol(QRect(*static\_cast*<int>(points[i].x()),

*static\_cast*<int>(points[i].y()),

SQUARE\_SIZE, SQUARE\_SIZE));

ptr.drawPolygon(pol);

}

*//draws* *Catmull-Rom* *spline*

*if* (cntPoints > 3) {

QPen newPen(QColor(138,43,226), 3);

ptr.setPen(newPen);

double prevX = points[1].x() + SQUARE\_SIZE / 2;

double prevY = points[1].y() + SQUARE\_SIZE / 2;

*for* (float t = 0; t < (float)points.size() - 4.0f; t += step) {

QVector2D pos = get\_spline\_point(t);

double x = pos.x() + SQUARE\_SIZE / 2;

double y = pos.y() + SQUARE\_SIZE / 2;

ptr.drawLine(*static\_cast*<int>(prevX),

*static\_cast*<int>(prevY),

*static\_cast*<int>(x),

*static\_cast*<int>(y));

prevX = x;

prevY = y;

*//* *setting* *connection* *to* *the* *last* *point* *according* *to* *the* *approximation*

*if* (t + step >= (float)points.size() - 4.0f) {

x = points[points.size() - 3].x() + SQUARE\_SIZE / 2;

y = points[points.size() - 3].y() + SQUARE\_SIZE / 2;

ptr.drawLine(*static\_cast*<int>(prevX),

*static\_cast*<int>(prevY),

*static\_cast*<int>(x),

*static\_cast*<int>(y));

}

}

}

}

**5. Выводы.**

В ходе данной лабораторной работы я изучил основы построения сплайнов, реализовал на языке C++ средствами Qt один из них, а именно сплайновую кривую Кэтмулла-Рома.