МИНОБРНАУКИ РОССИИ

Санкт-Петербургский государственный электротехнический университет « ЛЭТИ» им. В. И.Ульянова(Ленина)

Задание для лабораторной работы № 4

" КУБИЧЕСКИЕ СПЛАЙНЫ"

Выполнили:
Быков, Спас, 6383

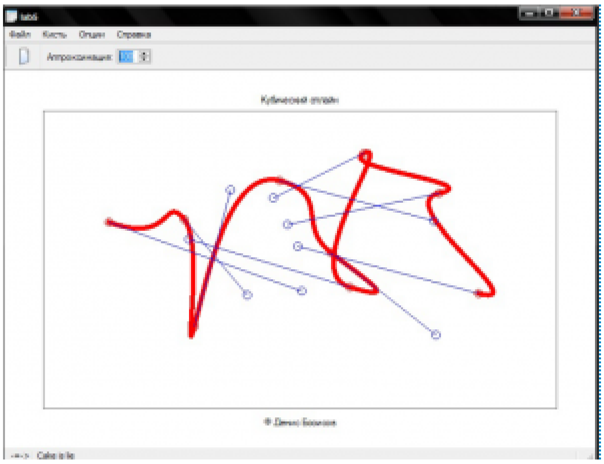
Факультет: КТИ

Преподаватель: Герасимова Т.В.

Санкт-Петербург
2019 г.

Задание.

Реализовать интерактивное приложение, отображающее заданные полиномиальные кривые



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.

Вариант 10. Сегмент кривой Кэтмулла-Рома.

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.

Здесь кратко излагаются некоторые основные положения и использования сплайнов в 3D графики.

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.

Определим искомую функцию $y = S(x)$, причем поставим два условия:

- Функция должна проходить через все точки: $S(x_i) = y_i$, $i = \overline{0, m}$;
- Функция должна быть дважды непрерывно дифференцируема, то есть иметь непрерывную вторую производную на всем отрезке $[x_0, x_m]$.

На каждом из отрезков $[x_i, x_{i+1}]$, $i = \overline{0, m-1}$, ищется функция в виде полинома третьей степени:

$$S_i(x) = \sum_{j=0}^{3} a_{ij} (x - x_i)^j .$$

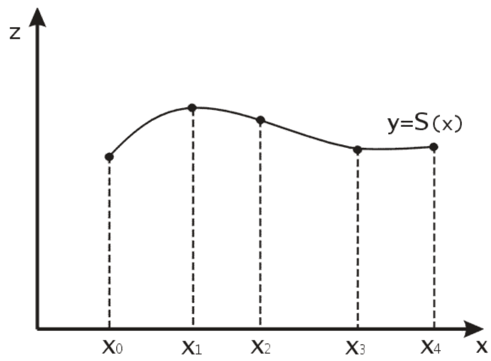


Рис. 1. Сплайновая функция

Поскольку для каждого из отрезков $[x_i, x_{i+1}]$ необходимо найти S Задача построения полинома сводится к нахождению коэффициентов a_{ij}, при этом общее количество искомых коэффициентов будет $4m$.

Перейдем к более сложному случаю – заданию кривых в трехмерном пространстве. Для функционального задания кривой $\begin{cases} y = f(x) \\ z = f(x) \end{cases}$ возможны многозначности в случае самопересечений и неудобства при значениях производных равных ∞.

Ввиду этого ищется функция в параметрическом виде. Пусть t - независимый параметр, такой что $0 \le t \le 1$. Кубическим параметрическим сплайном назовем следующую систему уравнений:

$$\begin{cases} x(t) = a_x t^3 + b_x t^2 + c_x t + d_x ; \\ y(t) = a_y t^3 + b_y t^2 + c_y t + d_y ; \\ z(t) = a_z t^3 + b_z t^2 + c_z t + d_z . \end{cases}$$

Координаты точек на кривой описываются вектором $(x(t), y(t), z(t))$, а три производные задают координаты соответствующего касательного вектора в точке. Например, для координаты x:

$$\frac{dx}{dt} = 3a_x t^2 + 2b_x t + c_x.$$

Сплайн Катмулла-Рома

Сплайн Катмулла-Рома - это сплайн Эрмита, производные которого определяются по формуле:

$$S'(x_i) = \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}}$$

Как и сплайн Эрмита, сплайн Катмулла-Рома имеет непрерывную первую производную и разрывную вторую. Сплайн Катмулла-Рома локален - значения сплайна зависят только от значений функции в четырех соседних точках (двух слева, двух справа). Можно использовать два типа граничных условий:

- Сплайн, завершающийся параболой. В этом случае граничный отрезок сплайна представляется полиномом второй степени вместо третьей (для внутренних отрезков по-прежнему используются полиномы третьей степени). В ряде случаев это обеспечивает большую точность, чем естественные граничные условия.
- Периодические граничные условие (этот вид граничных условий используется при моделировании периодических функций).

Алгоритм построения и используемые формулы.

Пусть $P_i = [x_i\, y_i]^T$ - точка. Для сегмента кривой C определенного точками P_0, P_1, P_2, P_3 и последовательностью узлов t_0, t_1, t_2, t_3 сплайн Catmull-Rom может быть произведен как:

$$\mathbf{C} = \frac{t_2 - t}{t_2 - t_1}\mathbf{B}_1 + \frac{t - t_1}{t_2 - t_1}\mathbf{B}_2$$

где

$$\mathbf{B}_1 = \frac{t_2 - t}{t_2 - t_0}\mathbf{A}_1 + \frac{t - t_0}{t_2 - t_0}\mathbf{A}_2$$
$$\mathbf{B}_2 = \frac{t_3 - t}{t_3 - t_1}\mathbf{A}_2 + \frac{t - t_1}{t_3 - t_1}\mathbf{A}_3$$
$$\mathbf{A}_1 = \frac{t_1 - t}{t_1 - t_0}\mathbf{P}_0 + \frac{t - t_0}{t_1 - t_0}\mathbf{P}_1$$
$$\mathbf{A}_2 = \frac{t_2 - t}{t_2 - t_1}\mathbf{P}_1 + \frac{t - t_1}{t_2 - t_1}\mathbf{P}_2$$
$$\mathbf{A}_3 = \frac{t_3 - t}{t_3 - t_2}\mathbf{P}_2 + \frac{t - t_2}{t_3 - t_2}\mathbf{P}_3$$

а также

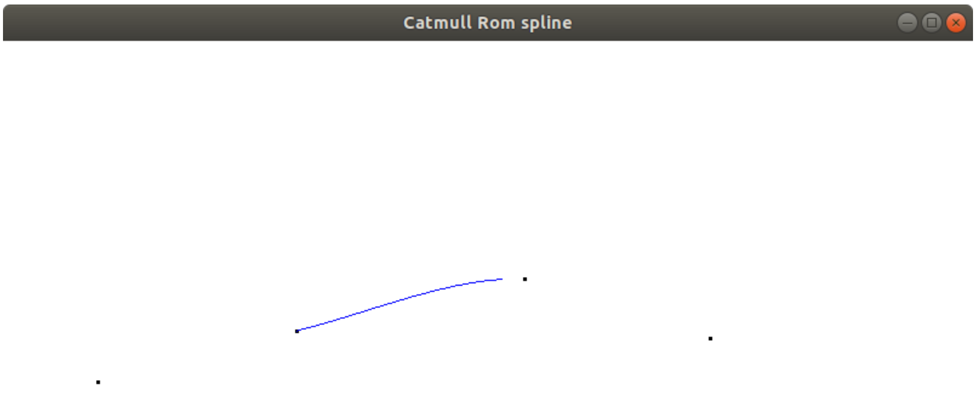
$$t_{i+1} = \left[\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}\right]^{\alpha} + t_i$$

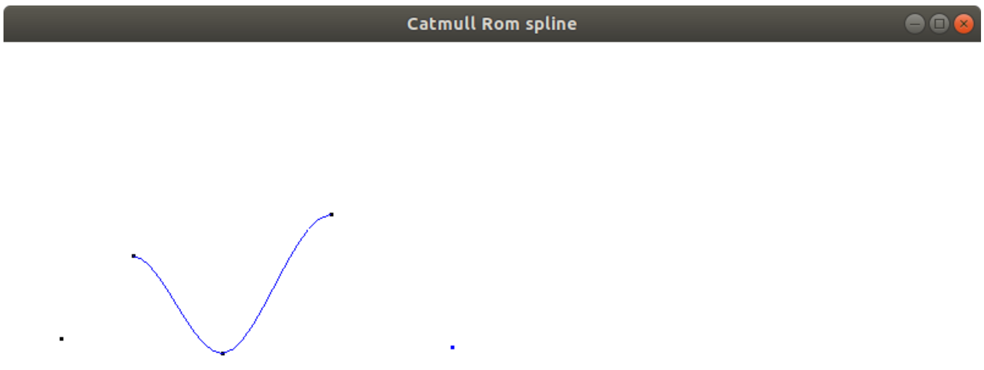
где, $\alpha=0{,}5$ для параметризации узла, и i = 0, 1, 2, 3 с $t_0 = 0$ для центростремительного сплайна Катмулла-Рома.

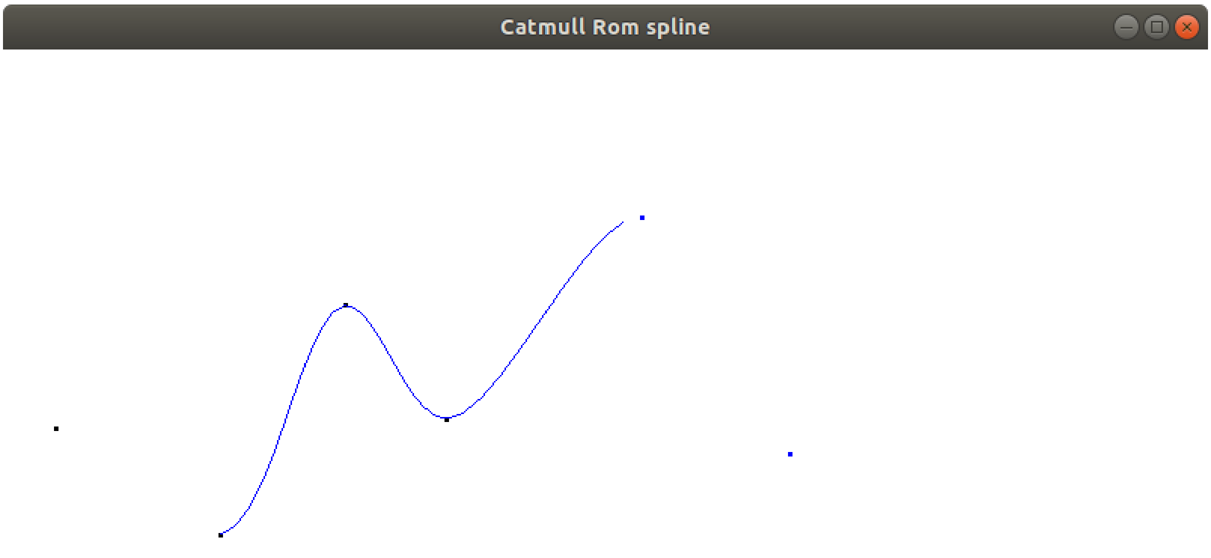
Центростремительный сплайн Катмулла-Рома имеет несколько желательных математических свойств по сравнению с оригинальным и другими типами формулировки Катмулла-Рома. Во-первых, он не будет образовывать петлю или самопересечение внутри сегмента кривой. Во-вторых, острие никогда не будет происходить внутри сегмента кривой. В-третьих, он более точно следует контрольным точкам.

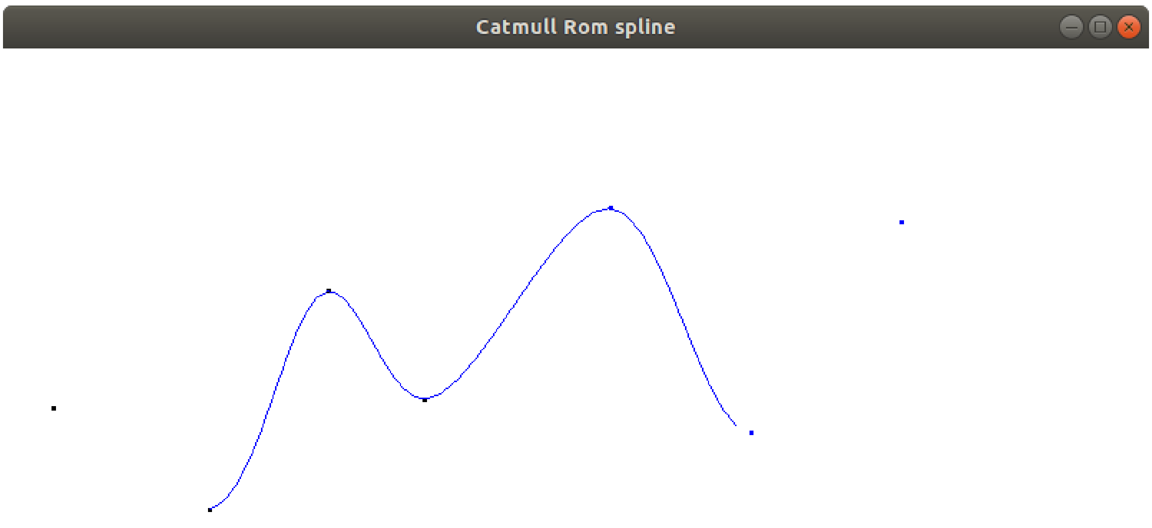
Реализация алгоритма приведена в приложении А.

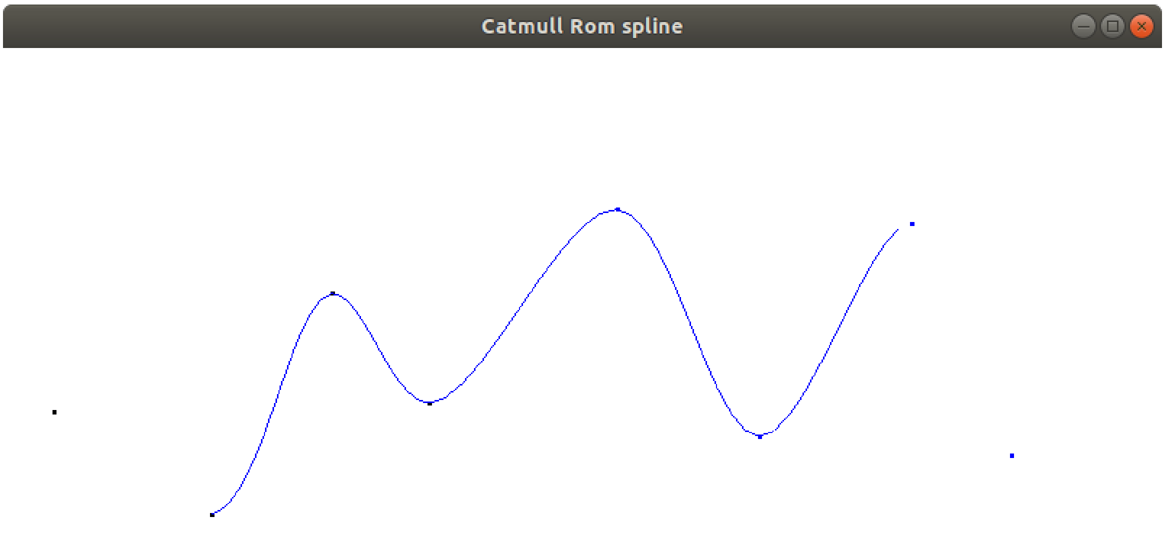
Пример работы программы.

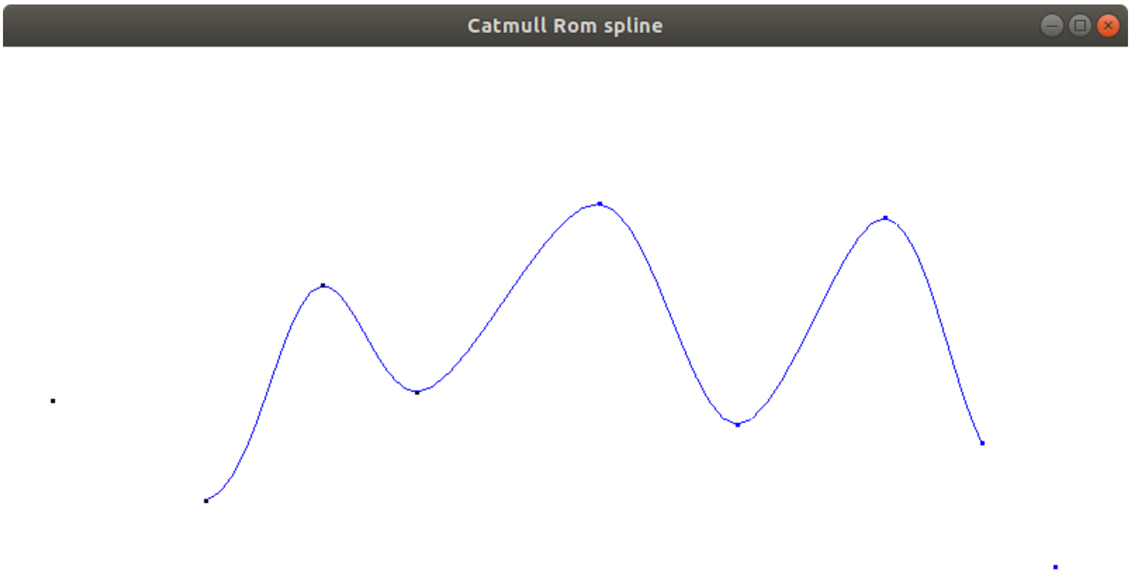


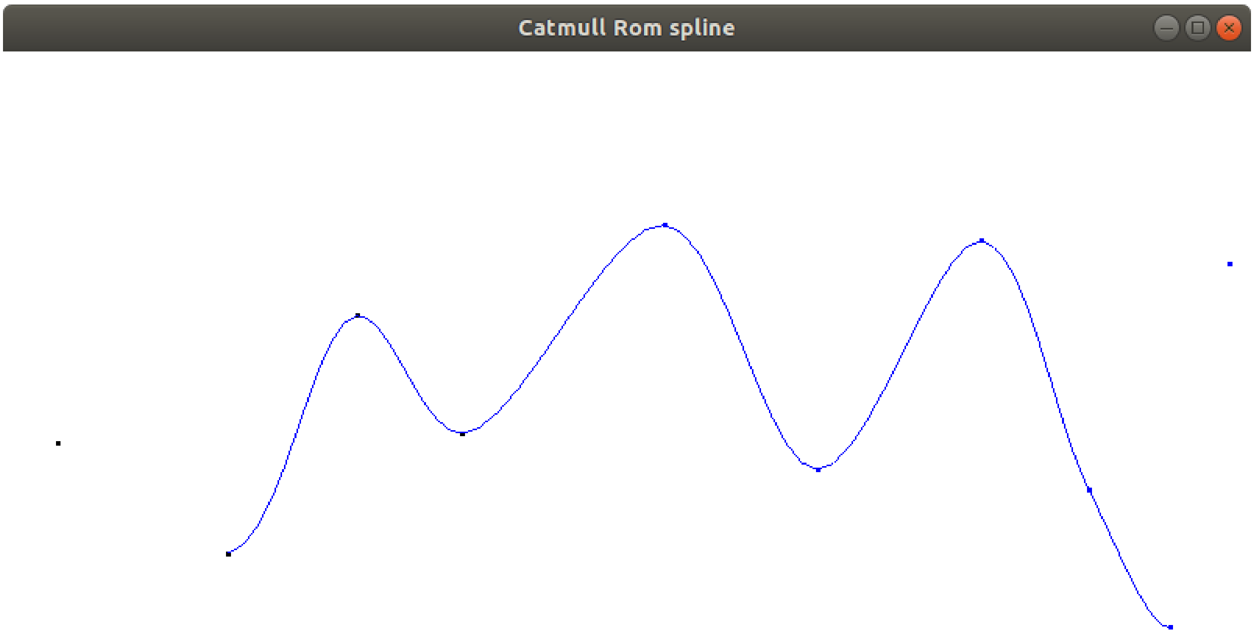



Catmull Rom spline


Catmull Rom spline


Catmull Rom spline


Catmull Rom spline


Catmull Rom spline

Вывод.

В результате выполнения лабораторной работы была разработана программа, отображающая сплайн Кэтмулла-Рома. Программа работает корректно. При выполнении работы были приобретены навыки работы с графической библиотекой OpenGL.

Приложение А

```
float tj(float ti, Point Pi, Point Pj) {
    float alpha = 0.5;
    float xi = Pi.getX();
    float yi = Pi.getY();
    float xj = Pj.getX();
    float yj = Pj.getY();
    return pow(pow((pow((xj - xi),2) + pow((yj - yi), 2)), 0.5), alpha) + ti;
}

std::vector < Point >  CatmullRomSpline(Point P0, Point P1, Point P2, Point
P3) {
     float amountOfPoints = 10;

     float t0 = 0;
     float t1 = tj(t0, P0, P1);
     float t2 = tj(t1, P1, P2);
     float t3 = tj(t2, P2, P3);
     std::vector < Point > res;
     for(float t=t1; t<t2; t+=((t2 - t1) / amountOfPoints))
             {
                 Point A1 = ((t1-t)/(t1-t0)) * P0 + ((t-t0)/(t1-t0)) * P1;
                 Point A2 = ((t2-t)/(t2-t1)) * P1 + ((t-t1)/(t2-t1)) * P2;
                 Point A3 = ((t3-t)/(t3-t2)) * P2 + ((t-t2)/(t3-t2)) * P3;

                 Point B1 = ((t2-t)/(t2-t0)) * A1 + ((t-t0)/(t2-t0)) * A2;
                 Point B2 = ((t3-t)/(t3-t1)) * A2 + ((t-t1)/(t3-t1)) * A3;

                 Point C = ((t2-t)/(t2-t1)) * B1 + ((t-t1)/(t2-t1)) * B2;
                 res.push_back(C);
             }
     return res;
}

std::vector < Point > CatmullRomChain(std::vector < Point > points) {
    std::vector < Point > res;
    for (size_t i = 0; i < points.size() - 3; i++) {
        std::vector < Point > buf = CatmullRomSpline(points[i], points[i +
1], points[i + 2], points[i + 3]);
        res.insert(std::end(res), std::begin(buf), std::end(buf));
    }
    return res;
}
```