**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Компьютерная графика»

**Лабораторная работа № 7**

Тема: Построение плоских полиномиальных кривых.

Студент: Алексеев Владислав Евгеньевич

Группа: М8О-306Б-19

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2021

1. **Постановка задачи**

Написать программу, строящую полиномиальную кривую по заданным точкам - Сегмент кубического сплайна по конечным точкам и касательным. Обеспечить возможность изменения позиции точек и, при необходимости, значений касательных векторов и натяжения.

1. **Описание программы**

Программа состоит из окна для взаимодействия с полиномиальной кривой по заданным точкам, из canvas для визуализации, и изменения некоторых ее параметров.

Программа написана на JavaScript с использованием библиотеки для отрисовки графиков. Уравнение для сплайна Эрмита: где – это крайние точки сплайна, – это производные в крайних точках.

При помощи библиотеки отрисовываются подвижные точки сплайна. После каждого изменения графика точки пересчитываются по уравнению выше.

1. **Набор тестов**

1. Изначальный вид.

2. Перемещение точек.

3. Увеличение количества точек кривой.

4. Перемещение точек.

5. Увеличение количества точек кривой.

6. Перемещение точек.

1. **Результаты выполнения тестов**

1. Изначальный вид.



Рис. 4.1 Результат открытия окна

2. Перемещение точек.



Рис. 4.2 Результат перемещение точек

3. Увеличение количества точек кривой.

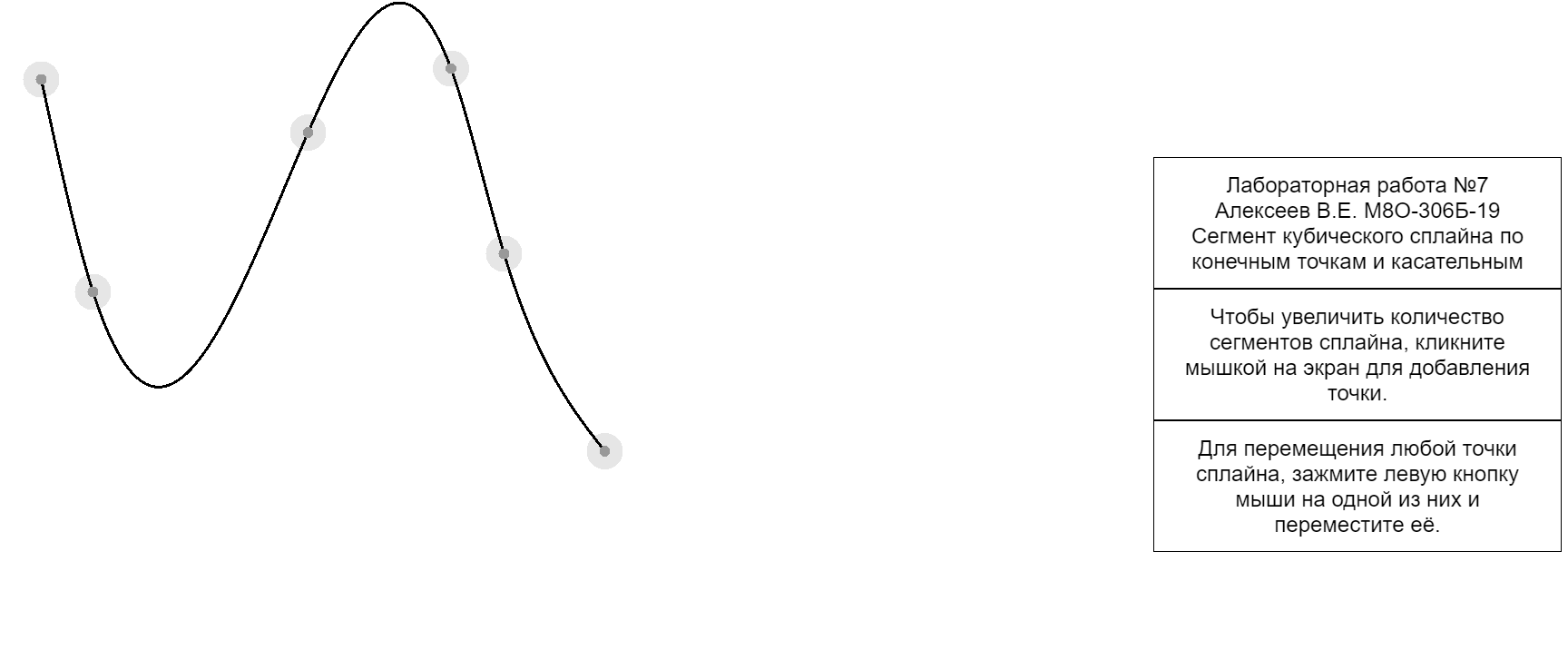


Рис. 4.3 Результат увеличения количества точек кривой

4. Перемещение точек.



Рис. 4.4 Результат перемещение точек

5. Увеличение количества точек кривой

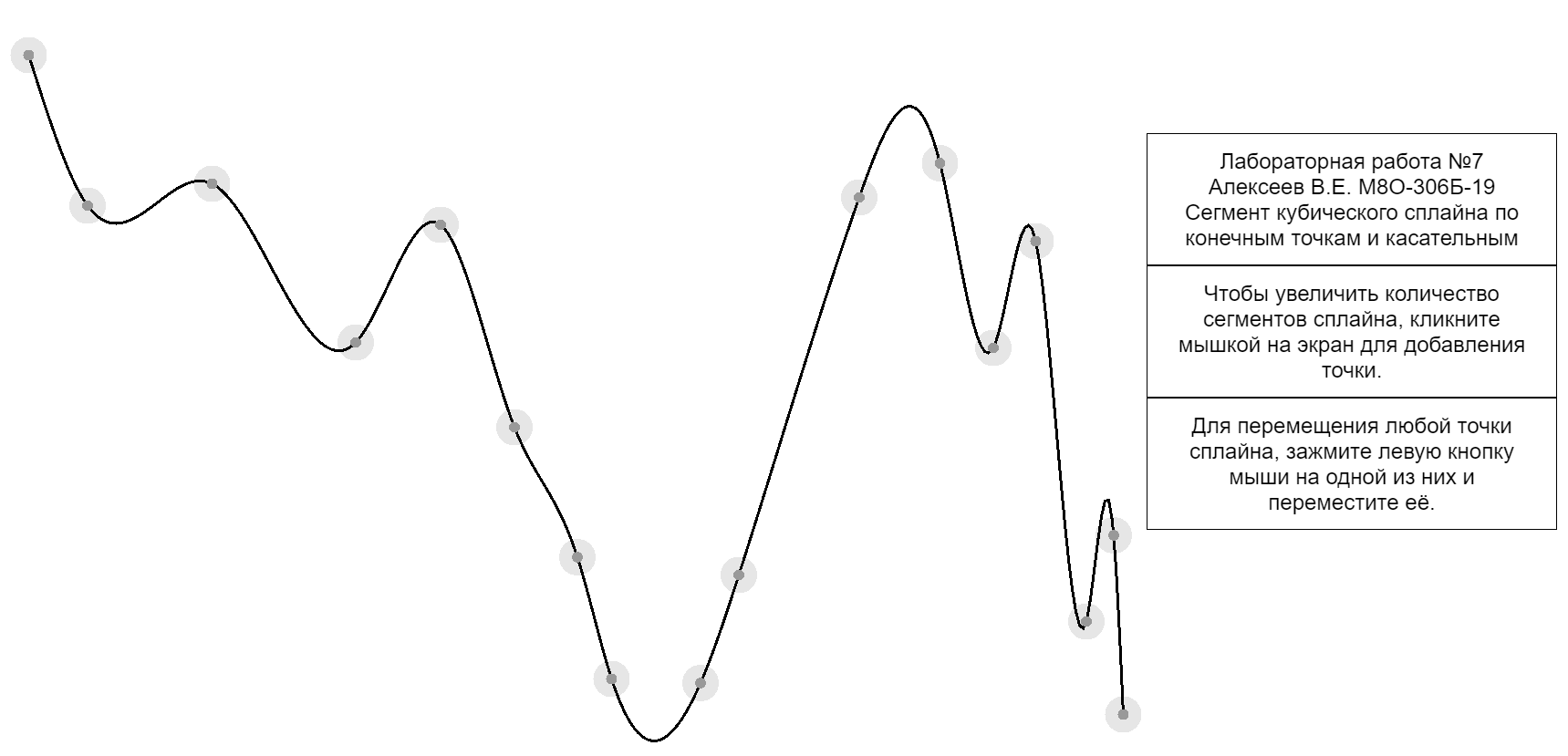


Рис. 4.5 Результат увеличения количества точек кривой

6. Перемещение точек.

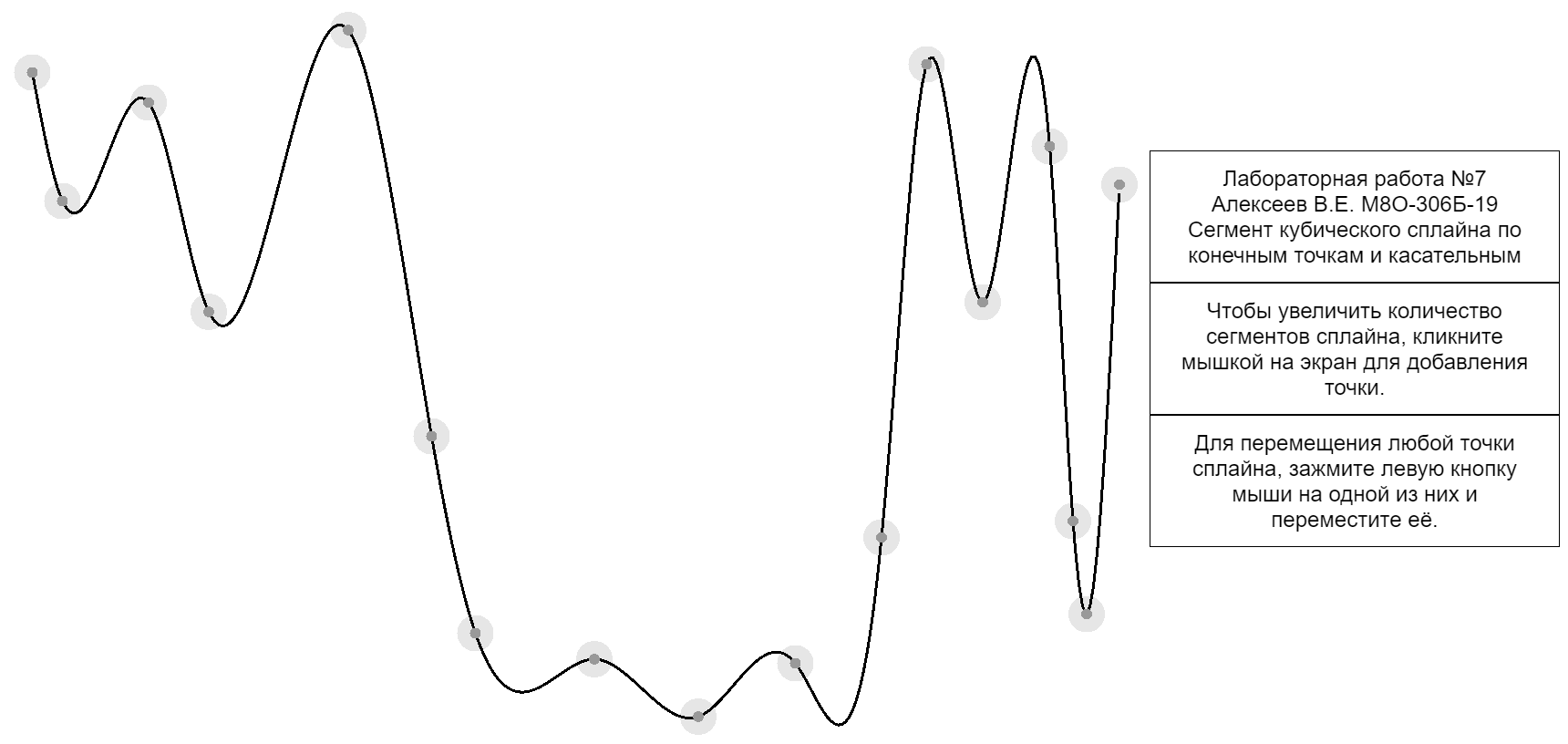


Рис. 4.6 Результат перемещение точек

1. **Листинг программы**

**main.html**

<!-- Алексеев В.Е. М8О-306Б-19

Построение плоских полиномиальных кривых.

Вариант: Сегмент кубического сплайна по конечным точкам и касательным. -->

<!doctype html>

<html>

<head>

<meta charset = 'utf-8'>

<title>Сегмент кубического сплайна по конечным точкам и касательным</title>

<link rel = 'stylesheet' href = './styles.css'>

<script src = "lib.js"></script>

<script src = "CSPL.js"></script>

<script src = './script.js'></script>

</head>

<body>

<canvas id = "default"></canvas>

<div class = 'help'>

<div>

Лабораторная работа №7 <br>

Алексеев В.Е. М8О-306Б-19 <br>

Сегмент кубического сплайна по конечным точкам и касательным

</div>

<div>

Чтобы увеличить количество сегментов сплайна, кликните мышкой на экран для добавления точки.

</div>

<div>

Для перемещения любой точки сплайна, зажмите левую кнопку мыши на одной из них и переместите ее.

</div>

</div>

</body>

</html>

**style.css**

.help {

position: fixed;

font-family: sans-serif;

top: 20%;

right: 10%;

width: 300px;

height: 600px;

text-align: center;

}

.help div {

padding: 10px;

border: 1px solid black;

}

**script.js**

var stage;

var s;

var dragged = null;

var dots = [];

window.innerWidth = 850;

window.innerHeight = 600;

function Start() {

stage = new Stage("default");

s = new Sprite();

stage.addChild(s);

for (var i = 0; i < 3; i++) {

var d = new Dot();

d.x = 50 + i \* 150;

d.y = 50 + i \* 100 + Math.sin(100 \* i);

d.addEventListener(MouseEvent.MOUSE\_DOWN, onMD);

stage.addChild(d);

dots.push(d);

}

stage.addEventListener(MouseEvent.MOUSE\_DOWN, addDot);

stage.addEventListener(MouseEvent.MOUSE\_MOVE, onMM);

stage.addEventListener(MouseEvent.MOUSE\_UP, onMU);

redraw();

}

function onMD(e) {

dragged = e.target;

}

function onMU(e) {

dragged = null;

}

function onMM(e) {

if (dragged == null) {

return;

}

dragged.x = stage.mouseX;

dragged.y = stage.mouseY;

redraw();

}

function addDot(e) {

if (e.target != stage) {

return;

}

var d = new Dot();

d.x = stage.mouseX;

d.y = stage.mouseY;

d.addEventListener(MouseEvent.MOUSE\_DOWN, onMD);

stage.addChild(d);

dots.push(d);

dragged = d;

redraw();

}

function redraw() {

dots.sort(function(a,b) {

return a.x - b.x;

});

var xs = []; ys = []; ks = [];

for (var i = 0; i < dots.length; i++) {

var d = dots[i];

xs[i] = d.x; ys[i] = d.y; ks[i] = 1;

}

CSPL.getNaturalKs(xs, ys, ks);

s.graphics.clear();

s.graphics.lineStyle(3, 0x000000);

var minx = dots[0].x;

var maxx = dots[dots.length-1].x;

s.graphics.moveTo(minx, CSPL.evalSpline(minx, xs, ys, ks));

for (var i = minx + 4; i <= maxx; i+= 4) {

s.graphics.lineTo(i, CSPL.evalSpline(i, xs, ys, ks));

}

}

function Dot() {

Sprite.apply(this); // наследуется от Sprite

this.graphics.beginFill(0x000000, 0.1);

this.graphics.drawCircle(0, 0, 20);

this.graphics.beginFill(0x999999, 1.0);

this.graphics.drawCircle(0,0, 6);

this.buttonMode = true;

}

Dot.prototype = new Sprite(); // Sprite Animation

window.onload = function () {

Start();

};

**CSPL.js**

function CSPL() {};

CSPL.\_gaussJ = {};

CSPL.\_gaussJ.solve = function(A, x) { // Наши решения в матрице

var m = A.length;

for (var k = 0; k < m; k++) { // Столбец

var i\_max = 0;

var vali = Number.NEGATIVE\_INFINITY;

for (var i = k; i < m; i++) {

if (Math.abs(A[i][k]) > vali) {

i\_max = i;

vali = Math.abs(A[i][k]);

}

}

CSPL.\_gaussJ.swapRows(A, k, i\_max);

// Для всех строк ниже вращения

for (var i = k + 1; i < m; i++) {

var cf = (A[i][k] / A[k][k]);

for (var j = k; j < m + 1; j++) {

A[i][j] -= A[k][j] \* cf;

}

}

}

for (var i = m - 1; i >= 0; i--) { // Строки равны столбцам

var v = A[i][m] / A[i][i];

x[i] = v;

for (var j = i - 1; j >= 0; j--) { // Строки

A[j][m] -= A[j][i] \* v;

A[j][i] = 0;

}

}

}

CSPL.\_gaussJ.zerosMat = function(r, c) {

var A = [];

for (var i = 0; i < r; i++) {

A.push([]);

for (var j = 0; j < c; j++) {

A[i].push(0);

}

}

return A;

}

CSPL.\_gaussJ.printMat = function(A) {

for (var i = 0; i < A.length; i++) {

console.log(A[i]);

}

}

CSPL.\_gaussJ.swapRows = function(m, k, l) {

var p = m[k];

m[k] = m[l];

m[l] = p;

}

CSPL.getNaturalKs = function(xs, ys, ks) { // in x values, in y values, out k values

var n = xs.length - 1;

var A = CSPL.\_gaussJ.zerosMat(n + 1, n + 2);

for (var i = 1; i < n; i++) { // Строки

A[i][i - 1] = 1 / (xs[i] - xs[i - 1]);

A[i][i] = 2 \* (1 / (xs[i] - xs[i - 1]) + 1 / (xs[i + 1] - xs[i]));

A[i][i + 1] = 1 / (xs[i + 1] - xs[i]);

A[i][n + 1] = 3 \* ((ys[i] - ys[i - 1]) / ((xs[i] - xs[i - 1]) \* (xs[i] - xs[i - 1]))

+ (ys[i + 1] - ys[i]) / ((xs[i + 1] - xs[i]) \* (xs[i + 1] - xs[i])));

}

A[0][0] = 2 / (xs[1] - xs[0]);

A[0][1] = 1 / (xs[1] - xs[0]);

A[0][n + 1] = 3 \* (ys[1] - ys[0]) / ((xs[1] - xs[0]) \* (xs[1] - xs[0]));

A[n][n - 1] = 1 / (xs[n] - xs[n - 1]);

A[n][n] = 2 / (xs[n] - xs[n - 1]);

A[n][n + 1] = 3 \* (ys[n] - ys[n - 1]) / ((xs[n] - xs[n - 1]) \* (xs[n] - xs[n - 1]));

CSPL.\_gaussJ.solve(A, ks);

}

CSPL.evalSpline = function(x, xs, ys, ks) {

var i = 1;

while (xs[i] < x) i++;

var t = (x - xs[i - 1]) / (xs[i] - xs[i - 1]);

var a = ks[i - 1] \* (xs[i] - xs[i - 1]) - (ys[i] - ys[i - 1]);

var b = -ks[i] \* (xs[i] - xs[i - 1]) + (ys[i] - ys[i - 1]);

var q = (1 - t) \* ys[i - 1] + t \* ys[i] + t \* (1 - t) \* (a \* (1 - t) + b \* t);

return q;

}

1. **Выводы**

Выполнив данную лабораторную работу, я закрепил знания по использованию HTML и запуску графический интерфейс, перерисовыванию его в зависимости от изменения окна и отрисовки полиномиальной кривой. Также улучшил свои навыки с отображением интерактивной 3D графики. Моделирование графики с использованием Three.js можно сравнить со съемочной площадкой, так как у нас есть возможность оперировать такими понятиями как сцена, свет, камера, объекты и их материалы. Также научился делать шейдерную анимацию эффектов и сложные построения полиномиальной кривой по заданным точкам.

**ЛИТЕРАТУРА**

1. Интерполяция данных / Хабр [Электронный ресурс]URL: <https://habr.com/ru/post/264191/> (Дата обращения: 25.11.2021).

2. Применение сегментированной кубической кривой сплайна [Электронный ресурс]URL: <https://russianblogs.com/article/2306554689/> (Дата обращения: 25.11.2021).

3. Cubic Hermite spline [Электронный ресурс]URL: <https://en.wikipedia.org/wiki/Cubic_Hermite_spline> (Дата обращения: 26.11.2021).