Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №

по дисциплине "Программирование графических приложений"

ТЕМА РАБОТЫ:

Плоские кривые линии в WebGL

Выполнил:

студент гр. ПРИм-124

Парахин К.В.

Принял:

Жигалов И.Е.

Владимир 2024 г.

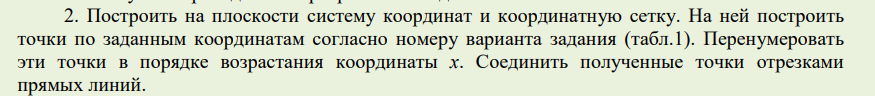
Цель работы:

Изучение способов построения плоских кривых линий при формировании моделей графических объектов с использованием WebGL.

Выполнение работы:

Индивидуальный вариант 2 (14 % 12 = 2)





    // сортировка по возрастанию x

    var new\_verticles = [

            4, 7, 0,

            5, 8, 0,

            7, 0, 0,

            8, 0, 0,

            9, 8, 0,

            9, 9, 0,

            10, 0, 0,

            10, 11, 0,

            11, 8, 0,

            11, 9, 0,

            12, 7, 0,

            14, 2, 0

        ];

    // установка буфера вершин

    vertexBuffer = gl.createBuffer();

    gl.bindBuffer(gl.ARRAY\_BUFFER, vertexBuffer);

    var verticles = new\_verticles.map((x) => x / 10 - 0.5);

    gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(verticles), gl.STATIC\_DRAW);

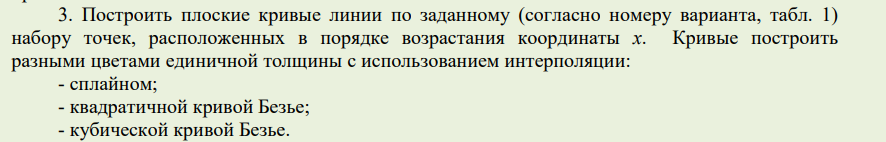
    vertexBuffer.itemSize = 3;

    vertexBuffer.numberOfItems = 12;

Отрисовка отрезков по точкам:

 gl.drawArrays(gl.LINE\_STRIP, 0, vertexBuffer.numberOfItems);





var spline = new THREE.SplineCurve([

new THREE.Vector2(4, 7),

new THREE.Vector2(5, 8),

new THREE.Vector2(7, 0),

new THREE.Vector2(8, 0),

new THREE.Vector2(9, 8),

new THREE.Vector2(9, 9),

new THREE.Vector2(10, 0),

new THREE.Vector2(10, 11),

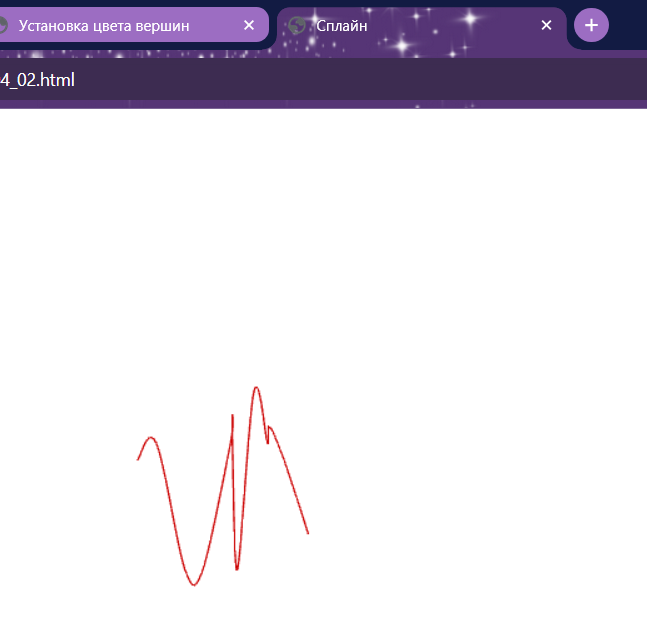
new THREE.Vector2(11, 8),

new THREE.Vector2(11, 9),

new THREE.Vector2(12, 7),

new THREE.Vector2(14, 2)

]);



v0 = new THREE.Vector2(4, 7);

v1 = new THREE.Vector2(5, 8);

v2 = new THREE.Vector2(7, 0);

v3 = new THREE.Vector2(8, 0);

v4 = new THREE.Vector2(9, 8);

v5 = new THREE.Vector2(9, 9);

v6 = new THREE.Vector2(10, 0);

v7 = new THREE.Vector2(10, 11);

v8 = new THREE.Vector2(11, 8);

v9 = new THREE.Vector2(11, 9);

v10 = new THREE.Vector2(12, 7);

v11 = new THREE.Vector2(14, 2);

var numberPoints = 12;

// ��������� ������ �����:

var curve = new THREE.QuadraticBezierCurve( v0, v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11);

var geometry = new THREE.Geometry;

var material = new THREE.LineBasicMaterial( { color: 0x95cc32 } );

for (var i = 0; i < numberPoints; i += 1)

{

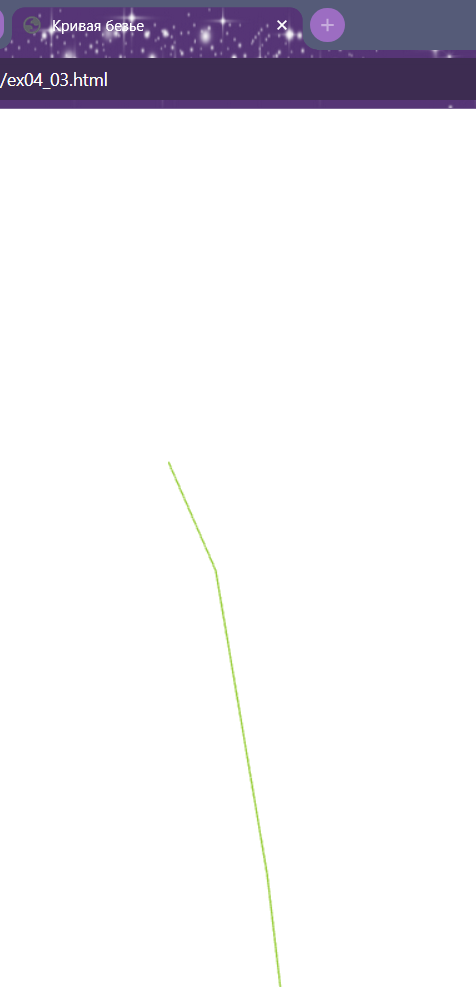
var x = curve.getPoint( i ).x;

var y = curve.getPoint( i ).y;

var vec = new THREE.Vector3( x, y, 0 );

geometry.vertices.push( vec );

}



v0 = new THREE.Vector2(4, 7);

v1 = new THREE.Vector2(5, 8);

v2 = new THREE.Vector2(7, 0);

v3 = new THREE.Vector2(8, 0);

v4 = new THREE.Vector2(9, 8);

v5 = new THREE.Vector2(9, 9);

v6 = new THREE.Vector2(10, 0);

v7 = new THREE.Vector2(10, 11);

v8 = new THREE.Vector2(11, 8);

v9 = new THREE.Vector2(11, 9);

v10 = new THREE.Vector2(12, 7);

v11 = new THREE.Vector2(14, 2);

var numberPoints = 12;

// ��������� ������ �����:

var curve = new THREE.CubicBezierCurve( v0, v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11);

var geometry = new THREE.Geometry;

var material = new THREE.LineBasicMaterial( { color: 0x00cc32 } );

for (var i = 0; i < numberPoints; i += 1)

{

var x = curve.getPoint( i ).x;

var y = curve.getPoint( i ).y;

var vec = new THREE.Vector3( x, y, 0 );

geometry.vertices.push( vec );

}





var spline = new THREE.SplineCurve3([

    new THREE.Vector3(4, 7, 0),

    new THREE.Vector3(5, 8, 0),

    new THREE.Vector3(7, 0, 0),

    new THREE.Vector3(8, 0, 0),

    new THREE.Vector3(9, 8, 0),

    new THREE.Vector3(9, 9, 0),

    new THREE.Vector3(10, 0, 0),

    new THREE.Vector3(10, 11, 0),

    new THREE.Vector3(11, 8, 0),

    new THREE.Vector3(11, 9, 0),

    new THREE.Vector3(12, 7, 0),

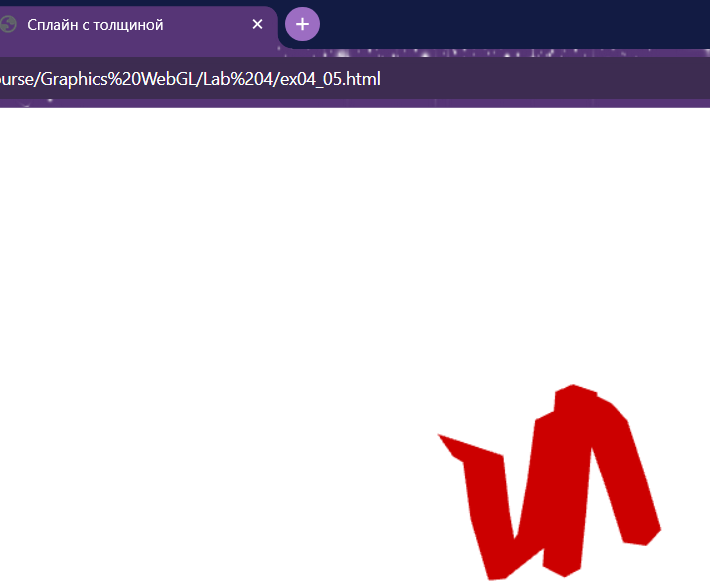
    new THREE.Vector3(14, 2, 0)

]);

var material = new THREE.LineBasicMaterial( { color: 0xcc0000 } );

var tubegeo = new THREE.TubeGeometry( spline, 16, 1, 4 );

var tube = new THREE.Mesh( tubegeo, material );



var v0 = new THREE.Vector3(4, 7, 0);

var v1 = new THREE.Vector3(5, 8, 0);

var v2 = new THREE.Vector3(7, 0, 0);

var v3 = new THREE.Vector3(8, 0, 0);

var v4 = new THREE.Vector3(9, 8, 0);

var v5 = new THREE.Vector3(9, 9, 0);

var v6 = new THREE.Vector3(10, 0, 0);

var v7 = new THREE.Vector3(10, 11, 0);

var v8 = new THREE.Vector3(11, 8, 0);

var v9 = new THREE.Vector3(11, 9, 0);

var v10 = new THREE.Vector3(12, 7, 0);

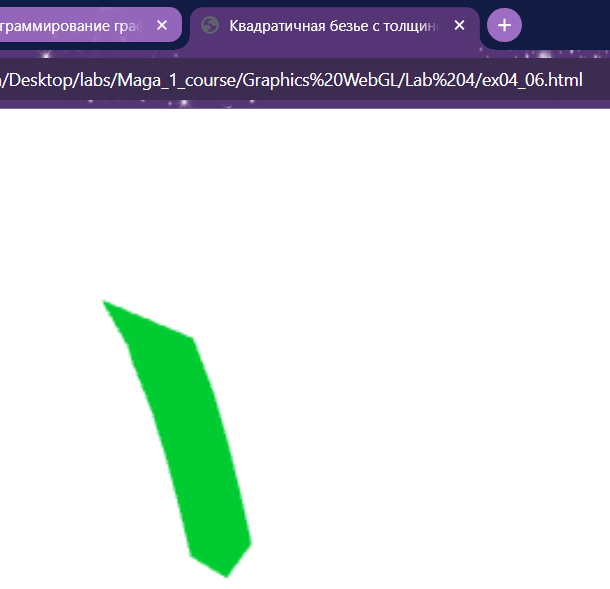
var v11 = new THREE.Vector3(14, 2, 0);

var quadBezie = new THREE.QuadraticBezierCurve3(v0, v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11);

var material = new THREE.LineBasicMaterial( { color: 0x00cc32 } );

var tubegeo = new THREE.TubeGeometry( quadBezie, 5, 1, 3 );

var tube = new THREE.Mesh( tubegeo, material );



var v0 = new THREE.Vector3(4, 7, 0);

var v1 = new THREE.Vector3(5, 8, 0);

var v2 = new THREE.Vector3(7, 0, 0);

var v3 = new THREE.Vector3(8, 0, 0);

var v4 = new THREE.Vector3(9, 8, 0);

var v5 = new THREE.Vector3(9, 9, 0);

var v6 = new THREE.Vector3(10, 0, 0);

var v7 = new THREE.Vector3(10, 11, 0);

var v8 = new THREE.Vector3(11, 8, 0);

var v9 = new THREE.Vector3(11, 9, 0);

var v10 = new THREE.Vector3(12, 7, 0);

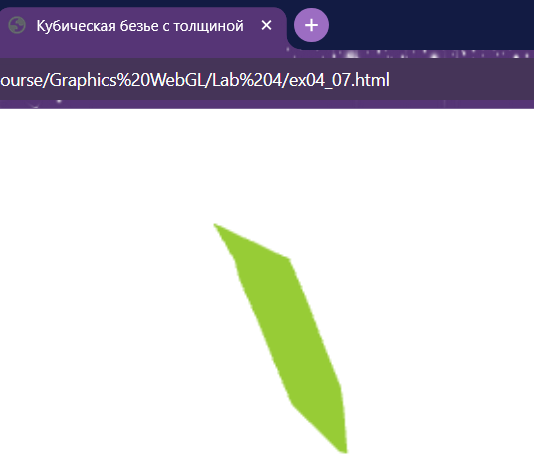
var v11 = new THREE.Vector3(14, 2, 0);

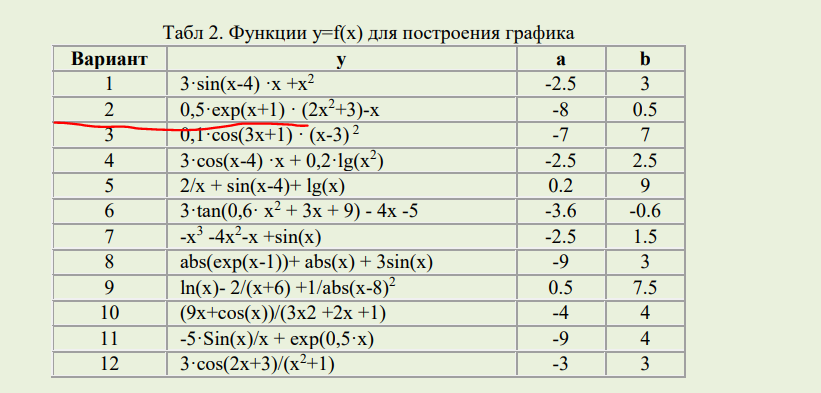
var quadBezie = new THREE.CubicBezierCurve3(v0, v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11);

var material = new THREE.LineBasicMaterial( { color: 0x97cc36 } );

var tubegeo = new THREE.TubeGeometry( quadBezie, 5, 1, 3 );

var tube = new THREE.Mesh( tubegeo, material );



    three.renderer.setClearColor(new THREE.Color(0xFFFFFF), 1.0);

      var camera = mathbox.camera( { proxy: true, position: [0, 0, 3] } );

      var view = mathbox.cartesian( {range: [[-8, 0.5], [-2, 2]], scale: [2,1]} );

      var xAxis = view.axis( {axis: 1, width: 8, detail: 40, color:"bllue"} );

      var yAxis = view.axis( {axis: 2, width: 8, detail: 40, color:"green"} );

      var grid = view.grid( {width: 2, divideX: 20, divideY: 10, opacity:0.25} );

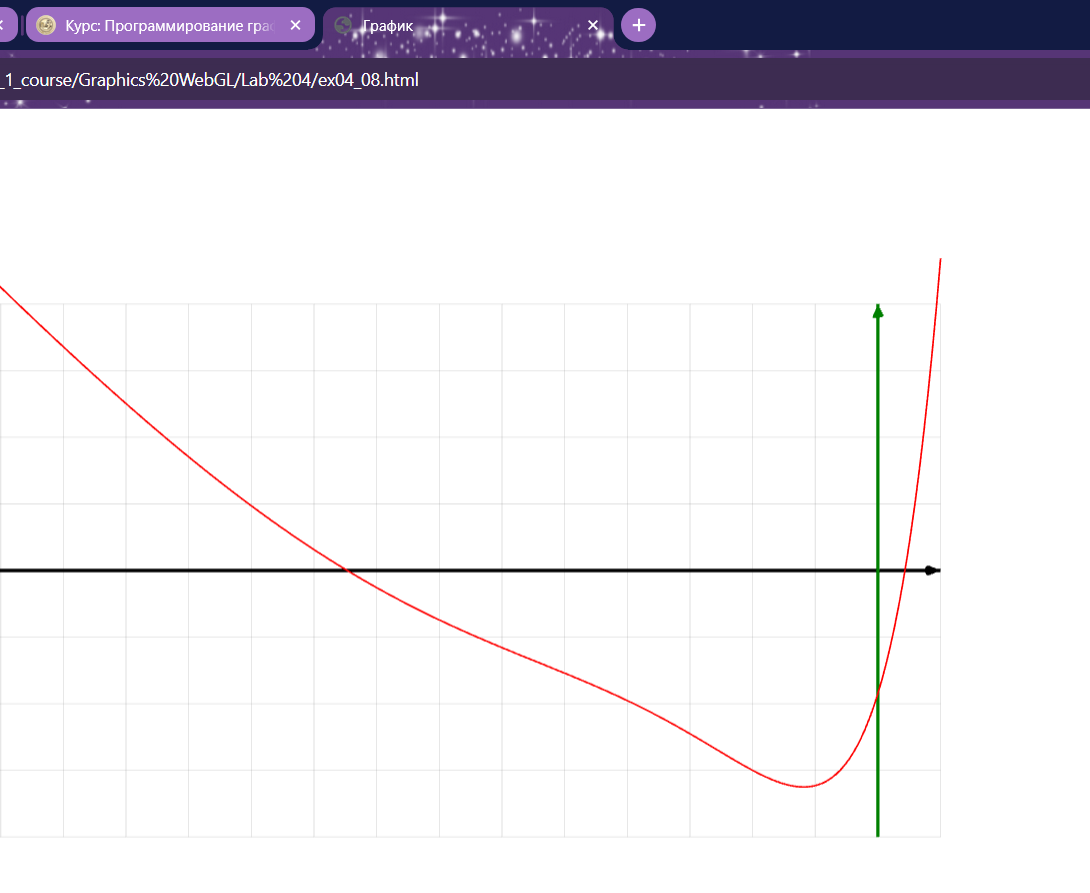
      var graphData = view.interval({

      expr: function (emit, x, i, t) { emit( x, 0.5 \* Math.exp(x + 1) \* (2 \* x \* x + 3) - x );  },

      width: 256,   channels: 2,

    });

    var graphView = view.line( {width: 4, color: "red"} );



Вывод

В результате выполнения работы я провел изучение способов построения плоских кривых линий при формировании моделей графических объектов с использованием WebGL.