**计算机图形学**

**实验报告**

课程名称： 计算机图形学实验报告

专 业： 计算机科学与技术 班级： 2018-5

姓 名： 刘澳

学 号： 201801011312

**山 东 科 技 大 学**

**2021年 6月 27日**

# 实验五 动态绘制Bezier曲线

1. 实验目的
2. 掌握直线的参数表示法。
3. 掌握德卡斯特里奥算法的几何意义。
4. 掌握绘制二维Bezier曲线的方法。
5. 实验要求
6. 使用鼠标左键绘制个数为10以内的任意控制点，使用鼠标左键绘制个数为10以内的任意控制点，
7. 单击鼠标右键绘制Bezier曲线。
8. 在状态栏显示鼠标的位置坐标。
9. Bezier曲线使用德卡斯特里奥(de Casteljau)算法绘制。
10. 实验步骤

在画布上使用左键进行点的选取，选取点完成后按下“Enter”键，进行贝塞尔曲线的绘制。

1. 关键程序代码

<script *type*="text/javascript">

    var canvas = document.**getElementById**("myCanvas");

    var ctx = canvas.**getContext**('2d');

    ctx.fillStyle = "rgba(232,221,203,0.5)";

    ctx.**fillRect**(0, 0, 1000, 1000);

    position = [];

    canvas.**addEventListener**('click', **click**);

    canvas.**addEventListener**('onkeypress', onkeypress);

    document.**onkeypress** = function () {

        var keynum = window.event;

        if (keynum.keyCode == 13) {

**drawBezierCurve**(position);

            position = [];

        }

    }

*//修正函数*

**HTMLCanvasElement**.prototype.**getMousePosition** = function (event) {

    };

    function **click**(e) {

        console.**log**("000");

        position.**push**(canvas.**getMousePosition**(e));

        if (position.length >= 2) {

**drawLine**(position[position.length - 2].x, position[position.length - 1].x, position[position.length - 2].y, position[position.length - 1].y);

        }

    }

    function **drawLine**(x1, x2, y1, y2) {

        ctx.strokeStyle = "black";

        ctx.**moveTo**(x1, y1);

        ctx.**lineTo**(x2, y2);

        ctx.**stroke**();

    }

    function **drawBezierCurve**(position) {

        var delt = 1.0 / 1500;*//delt 步长*

        var x, y;

        px = [];

        py = [];

        for (var i = 0; i <= position.length - 1; i++) {

            px[i] = position[i].x;

            py[i] = position[i].y;

        }

        for (var t = 0; t < 1; t += delt) {

            x = **deCasteliau**(t, px);

            y = **deCasteliau**(t, py);

**putpixel**(x, y);

        }

    }

    function **deCasteliau**(t, p) {

        var length = p.length;

        var P = new **Array**(length);

        {

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

                P[i] = new **Array**(length);

            }

        }

        var n = length - 1;

        for (var k = 0; k <= n; k++) {

            P[0][k] = p[k];

        }

        for (var r = 1; r <= n; r++) {

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

                P[r][i] = (1 - t) \* P[r - 1][i] + t \* P[r - 1][i + 1];

                console.**log**(r,i);

                console.**log**(P[r][i]);

            }

        }

        var point = P[n][0];

        return point;

    }

    function **putpixel**(x, y) {

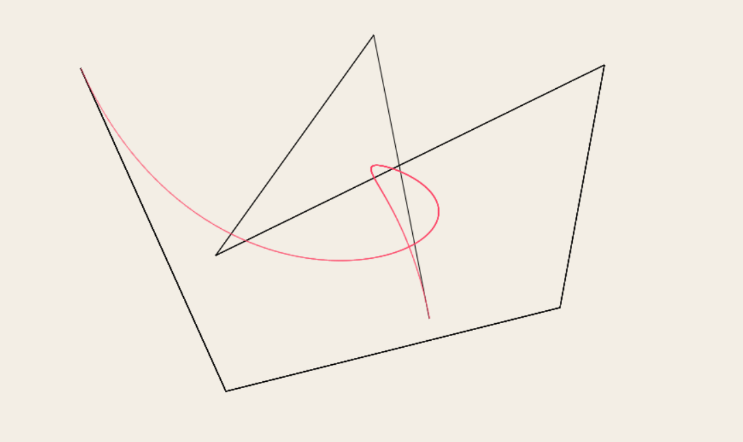
        ctx.fillStyle = "rgba(254, 67, 101, 1.2)";

        ctx.**fillRect**(x, y, 1, 1);

    }

</script>

1. 程序运行结果、界面等



1. 思考与总结等

贝塞尔曲线(Bézier curve)，又称贝兹曲线或贝济埃曲线，是应用于二维图形应用程序的数学曲线。一般的矢量图形软件通过它来精确画出曲线，贝兹曲线由[线段](https://baike.baidu.com/item/%E7%BA%BF%E6%AE%B5/8679802)与[节点](https://baike.baidu.com/item/%E8%8A%82%E7%82%B9/865052)组成，节点是可拖动的支点，线段像可伸缩的皮筋，我们在绘图工具上看到的钢笔工具就是来做这种矢量[曲线](https://baike.baidu.com/item/%E6%9B%B2%E7%BA%BF/12004395)的。

贝塞尔曲线的原理并不复杂，使用递归或者迭代就可以进行计算，但是我用js的时候发现：当步长很小时，计算过程会非常漫长。希望以后学习相关优化技术，提高计算速度。