使用有理Bezier函数绘制圆形

绘制圆的控制点的要求

为了方便，我们使用二次有理Bezier曲线来绘制圆，因此需要三个控制点。

假如我们在圆弧上任取两点，并假定圆弧是由这两点为起点以及终点、以及由这两点确定的一个点为控制点，通过合理配置权重绘制的有理Bezier曲线。那么，由于Bezier曲线在起点与终点处的切线性质，我们可以很轻易地得知我们需要的第三个控制点实际上是分别过起点和终点的圆的切线的交点。

为了方便，我们取正方形的三个顶点，显而易见地，这三个顶点可以用于绘制一个半径等于正方形边长的四分圆。当我们在第一次绘制得到这些顶点数据之后，只需要使用旋转变换就能绘制一个整圆。

二次有理Bezier曲线的计算实现

有理Bezier曲线的参数方程如下：

由于我们使用的是二次有理Bezier曲线，因此$n = 2$， 我们可以提前算出需要参数：这个实现如下：

```
class Bezier{
    /**
    *
    * @param {number} t The variable
    * @param {Float32Array} wx The weight given by control points
    * @param {Float32Array} wy The weight given by control points
    * @param {Float32Array} r Ratio used to generate Rational Bezier point
    *
    * @returns {Float32Array} The Rational Bezire point
    */
    static RationalBezier(t, wx, wy, r) {
       let t2 = t * t;
       let mt = 1 - t;
       let mt2 = mt * mt;
       let f = [
           r[0] * mt2,
           2 * r[1] * mt * t,
           r[2] * t2
       ];

       let basis = f[0] + f[1] + f[2];

       let x = 0;
       for (let i = 0; i < 3; i++) {
           x += f[i] * wx[i];
       }
       x /= basis;

       let y = 0;
       for (let i = 0; i < 3; i++) {
           y += f[i] * wy[i];
       }
       y /= basis;

```

```
        return ([x, y]);
    }
}
```

我们将三个控制点的`x`坐标以及`y`坐标分别作为向量变量传入，该程序会输出一个坐标，这个坐标即是我们要求的目标点。

剩下的问题只剩下确定各个点的权重。考虑到二次有理Bezier曲线绘制的圆弧自然是对称的，我们将起点和终点的权重设为1即可，这样唯一的变量就是我们的控制点的权重。又因为对于一个四分圆而言，在$t = \frac{1}{2}$ 处，其横坐标与纵坐标相同，且都为$\frac{r}{\sqrt{2}}$。 因此我们可以通过这个性质计算权重。权重的计算结果是$\frac{\sqrt{2} - 1}{2 - \sqrt{2}}$

生成用于绘制的有理Bezier曲线上的点

我们在`\ScriptResources\geomery.js`中定义了生成顶点缓冲中的数据的方法：

```
class myArray{
    constructor() {
        //分辨率，控制t的步进长度
        let resolution = 1000;
        let delta_t = 1.0 / resolution;

        //定义三个控制点
        let p1 = [0, 1];
        let p2 = [1, 1];
        let p3 = [1, 0];

        //定义传入Bezier.RationalBezier()函数的变量
        let wx = [p1[0], p2[0], p3[0]];
        let wy = [p1[1], p2[1], p3[1]];
        let r  = [1, (Math.SQRT2 - 1) / (2 - Math.SQRT2), 1];

        //初始化存放顶点数据的数组
        this.position = [];

        //t按照分辨率步进，生成一系列的Bezier曲线上的点
        let t = 0;
        for (let i = 0; i < resolution; i++){
            let p = Bezier.RationalBezier(t, wx, wy, r);
            this.position.push(p[0], p[1]);
            t += delta_t;
        }

        //最后，为了直观，绘制两条控制线。
        this.position.push(1.0, 1.0, 0.0, 1.0);
    }
}
```

本工程中的其它脚本都只用于支持绘制，与主题无关，故不做赘述。注意我们使用`LINE_STRIP`参数来绘制连续的折线段以减少空间的浪费。

运行结果

请点击http://www.mckpersonal.cn/MyWebGL/CG_HW_4/Bezier.html以访问在线实例。

源代码

源代码托管在Github上：https://github.com/Sigmoni/MyWebGL/tree/main/CG_HW_4