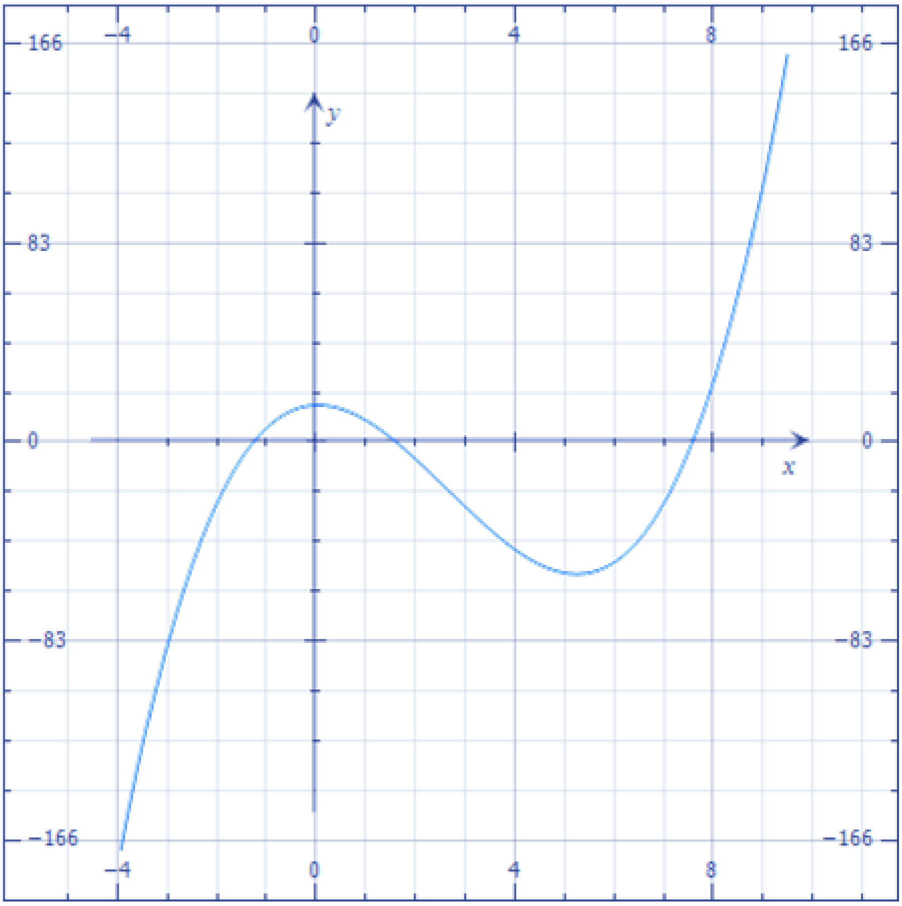
曲线和表面

Cubic Curves

由于其使用的简单和灵活性，三次多项式曲线再计算机图形中应用很广泛。一个基础的三次曲线可以由以下式子表示：

$$Q(t) = a + bt + ct^2 + dt^3$$



其中a,b,c,d均为常向量，$Q(t)$是曲线上关于t的一个点，将公式拆分为坐标轴分量形式，可以表示为:

$$Q_x(t) = a_x + b_x t + c_x t^2 + d_x t^3$$
$$Q_y(t) = a_y + b_y t + c_y t^2 + d_y t^3$$
$$Q_z(t) = a_z + b_z t + c_z t^2 + d_z t^3$$

通过矩阵的方式来表示上述式子将更加方便，如下所示：

$$Q(t) = \begin{bmatrix} a_x & b_x & c_x & d_x \\ a_y & b_y & c_y & d_y \\ a_z & b_z & c_z & d_z \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

用更加紧（抽）凑（像）的符号表示，可以写成

$$Q(t) = CT(t)$$

这里的C代表矩阵的参数，$T(t) \equiv < 1, t, t^2, t^3 >$。$Q(t)$的导数可以代表曲线的t的位置时的切线方向，可以表示为如下：

$$Q'(t) = C\frac{d}{dt}T(t) = C \begin{bmatrix} 0 \\ 1 \\ 2t \\ 3t^3 \end{bmatrix}$$

一条很长的曲线可以看成是由多个分段的三次曲线连接端点组成的。在相邻的两个分段的连接点处，有参数连续性和几何连续性的概念。符号C^n用于表示n阶参数连续性，G^n用于表示n阶几何连续性。如果两个曲线在连接点处具有一样方向和大小的切线，则被成为具有C^1连续性。如果切线有同样的方向，但是不一样的大小，则被称为具有G^1连续性。通常来说，两个曲线如果n阶导相等，则C^n连续，如果n阶导不为0且方向相同，大小不同，则具有G^n连续。C^n连续就隐含了G^n连续，除非n阶导为0。而C^0和G^0是等价的，表示曲线具有一样的端点。

参数连续性确保了拼接的分段曲线，在区间内的是连续的。

几何连续性确保了在区间内，曲线是平滑的。

本章节中我们定义的三次曲线，是通过特定的几何约束来表示的。例如端点位置（比如：$Q(0)$, Q(1)）或者端点的切向方向（比如：$Q'(0), Q'(1)$）。由于定义一个任意的三次曲线，需要四个参数来约束（因为有四个系数a, b, c, d，要求出四个系数，至少需要四个点代入才能解出来），我们将约束点命名为g_1, g_2, g_3, g_4，可以将$Q(t)$表示为：

$$\begin{aligned} Q(t) = &(a_1 + b_1 t + c_1 t^2 + d_1 t^3) g_1 \\ &+ (a_2 + b_2 t + c_2 t^2 + d_2 t^3) g_2 \\ &+ (a_3 + b_3 t + c_3 t^2 + d_3 t^3) g_3 \\ &+ (a_4 + b_4 t + c_4 t^2 + d_4 t^3) g_4 \end{aligned}$$

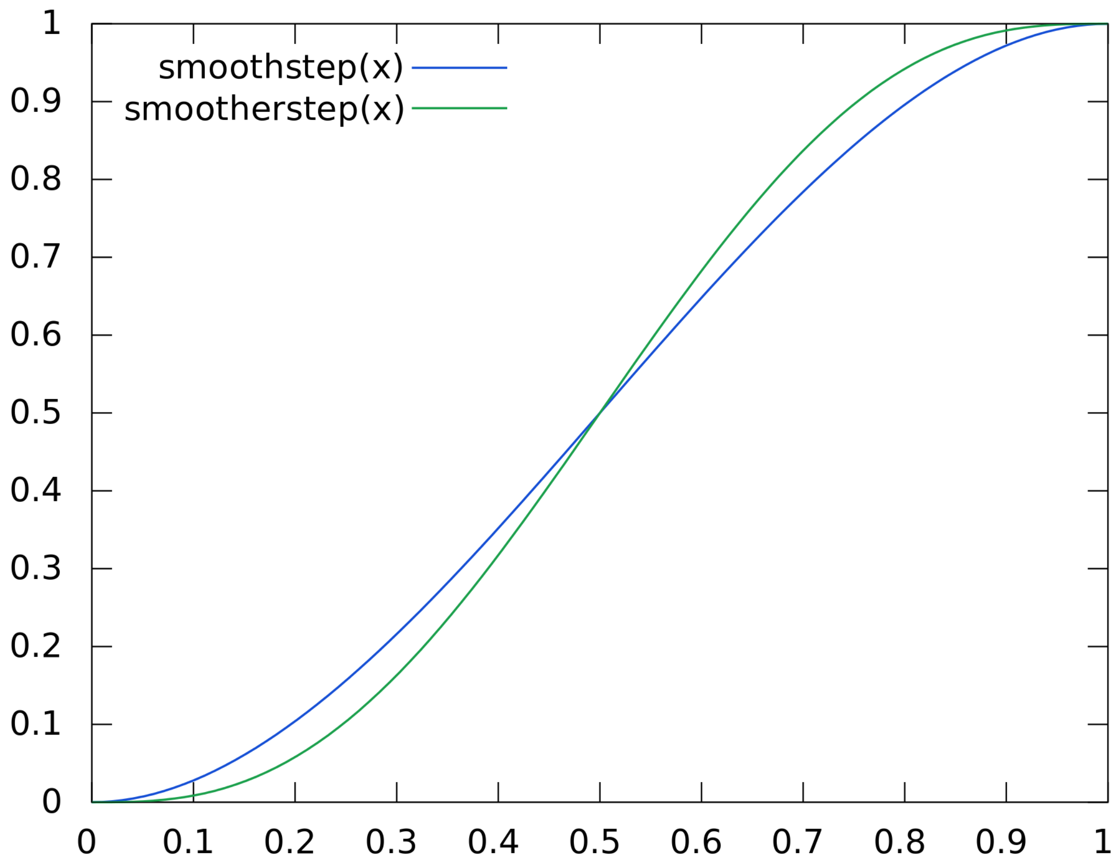
上式可以看成是一个加权和的形式，而式子$a_i + b_i t + c_i t + d_i t$又被称为blending funtions。写成矩阵形式如下：

$$Q(t) = [g_1 \quad g_2 \quad g_3 \quad g_4] \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

同样的，抽象写法如下：

$$Q(t) = GMT(t)$$

总结一下就是，任何曲线，都可以通过基函数和参数的叠加多项式来拟合，类似泰勒展开式的感觉。g就是权重参数，基函数一般用三次多项式来表示，至于为什么选用三次多项式，主要是由于三次多项式可以比较方便的构造出类似smoothstep的曲线。



关于上式的矩阵G,定义如下：

$$G = [g_1 \quad g_2 \quad g_3 \quad g_4] = \begin{bmatrix} (g_1)_x & (g_2)_x & (g_3)_x & (g_4)_x \\ (g_1)_y & (g_2)_y & (g_3)_y & (g_4)_y \\ (g_1)_z & (g_2)_z & (g_3)_z & (g_4)_z \end{bmatrix}$$

矩阵M定义如下：

$$M = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{bmatrix}$$

M矩阵用来定义基函数，G矩阵用来控制形状。

Hermite Curves

一个三次Hermite曲线是由两个端点P_1, P_2，和对应的切线方向T_1, T_2来控制的。用这四个量来定义G矩阵，可以把Hermite曲线定义如下：

$$H(t) = [P_1 \quad P_2 \quad T_1 \quad T_2] M_H \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

M_H是基函数的参数矩阵，G矩阵为$[P_1 \quad P_2 \quad T_1 \quad T_2]$。这里我们分别取两点（t=0和t=1），可以得到下面四个等式：

$$\begin{aligned} H(0) &= [P_1 \quad P_2 \quad T_1 \quad T_2] M_H < 1,0,0,0 >= P_1 \\ H(1) &= [P_1 \quad P_2 \quad T_1 \quad T_2] M_H < 1,1,1,1 >= P_2 \\ H'(0) &= [P_1 \quad P_2 \quad T_1 \quad T_2] M_H < 0,1,0,0 >= T_1 \\ H'(1) &= [P_1 \quad P_2 \quad T_1 \quad T_2] M_H < 0,1,2,3 >= T_2 \end{aligned}$$

将上式用矩阵的形式合并运算可以写成：

$$[P_1 \quad P_2 \quad T_1 \quad T_2] M_H \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 0 & 3 \end{bmatrix} = [P_1 \quad P_2 \quad T_1 \quad T_2]$$

因此，可以推断出矩阵M_H其实就是右边常数矩阵的逆：

$$M_H = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 0 & 3 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & 0 & 3 & -2 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

得到参数矩阵后，可以把式子写成加权和的形式：

$$H(t) = (1 - 3t^2 + 2t^3)P_1 + t^2(3 - 2t)P_2 + t(t-1)^2 T_1 + t^2(t-1)T_2$$

通过上式可以看到，当t=0时，只有P_1的系数不为0，当t=1时，只有P_2的系数不为0。如下图所示：

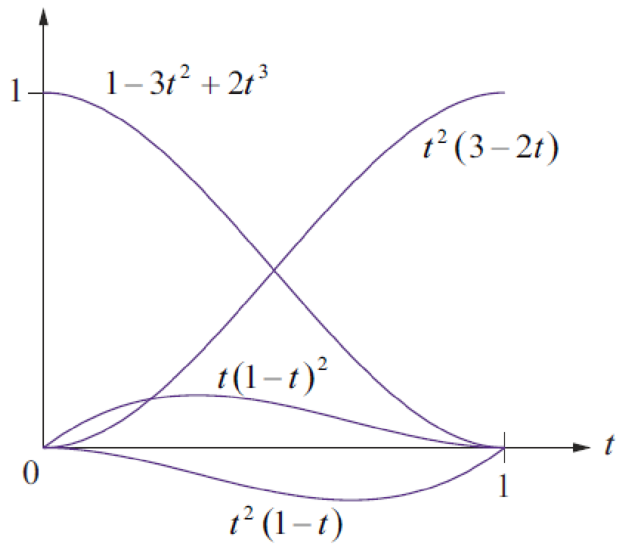


Figure 11.1. Blending functions for the Hermite class of cubic curves.

下图展示的是两个Hermite曲线的拼接，曲线分别为$H_1(t)$和$H_2(t)$，他们共享了一个端点P_2。如果曲线$H_1(t)$的几何矩阵是$[P_1 \quad P_2 \quad T_1 \quad T_2]$，并且曲线$H_2(t)$的几何矩阵是$[P_2 \quad P_3 \quad \mu T_2 \quad T_3]$ $(\mu > 0)$，则这两段曲线满足G^1连续，如果$\mu = 1$则满足C^1连续。

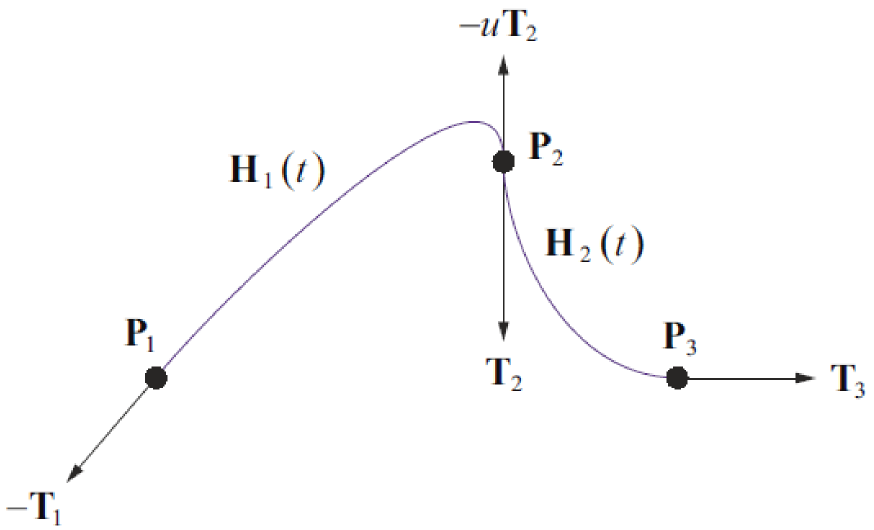


Figure 11.2. Two Hermite curves sharing the endpoint $\mathbf{P}_2$.

Catmull-Rom Splines

给 定 一 个 有 n+1 个 点 的 点 集 $\{P_0, P_1, \ldots, P_n\}(n \geq 3)$,Catmull-Rom spline 会 对 点 $\{P_1, P_2, \ldots, P_{n-1}\}$进行分段的三次曲线插值。点$P_i$切线向量$T$由下面的公式给出：

$$T_i = \frac{1}{2}(P_{i+1} - P_{i-1})$$

我们可以用下面的式子表示每一个分段$C_i(t)$，当$1 \leq i \leq n-2$,Hermite曲线有端点P_i和P_{i+1}，且切线T_i和T_{i+1}：

$$C_i(t) = [P_i \;\; P_{i+1} \;\; T_i \;\; T_{i+1}] M_H \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

现在我们需要找到一个基础的矩阵M_{CR}，让我们可以用四个点来表示几何矩阵G_{CR}：

$$[P_i \;\; P_{i+1} \;\; T_i \;\; T_{i+1}] = [P_{i-1} \;\; P_i \;\; P_{i+1} \;\; P_{i+2}] \begin{bmatrix} 0 & 0 & -\frac{1}{2} & 0 \\ 1 & 0 & 0 & -\frac{1}{2} \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

将上面的式子代入，可以得到：

$$M_{CR} = \frac{1}{2} \begin{bmatrix} 0 & 0 & -1 & 0 \\ 2 & 0 & 0 & -1 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & 0 & 3 & -2 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -1 & 2 & -1 \\ 2 & 0 & -5 & 3 \\ 0 & 1 & 4 & -3 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

现在，可以把分段样条的表达式写为：

$$C_i(t) = [P_{i-1} \quad P_i \quad P_{i+1} \quad P_{i+2}] M_{CR} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

$[P_{i-1} \quad P_i \quad P_{i+1} \quad P_{i+2}]$是几何矩阵$G_{CR}$ 。下图展示了一个由四个样条组成的曲线。

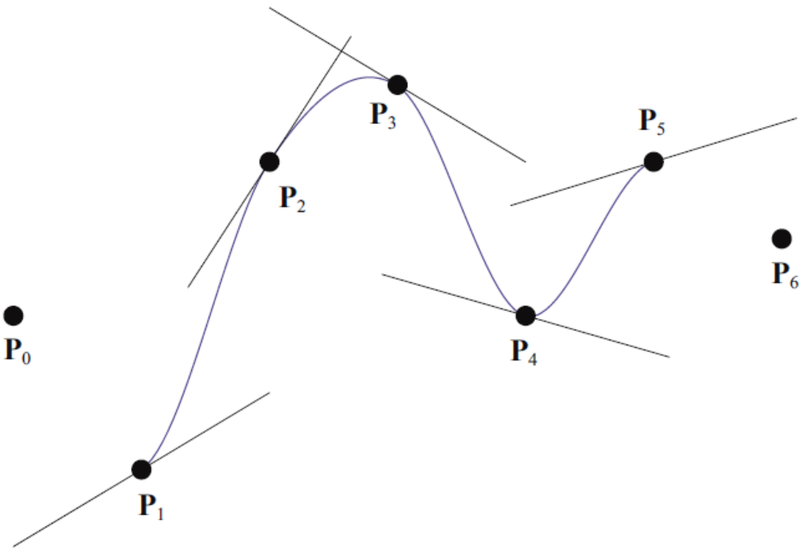


Figure 11.8. A Catmull-Rom spline interpolates a set of points in such a way that the tangent direction at each point is parallel to the line segment connecting the two neighboring points.

代码实现

参考上文所述，实际上曲线上的每个点都是由四个控制点加权和得到的，自变量t用于控制权重。核心代码如下：

```
public static readonly int k_CatmullRomPointCountLimit = 4;
private static readonly Matrix4x4 k_CatmullRomCoefficient = new (
    new Vector4( x: 0,  y: 1,  z: 0,  w: 0),
    new Vector4( x: -1/2f,  y: 0,  z: 1/2f,  w: 0),
    new Vector4( x: 1,  y: -5/2f,  z: 2,  w: -1/2f),
    new Vector4( x: -1/2f, y: 3/2f, z: -3/2f, w: 1/2f));

Frequently called  1 usage
public static Matrix4x4 BuildCatmullRomGeometry(Vector3 g1, Vector3 g2, Vector3 g3, Vector4 g4)
{
    return new Matrix4x4(
        new Vector4(g1.x, g1.y, g1.z,  w: 0f),
        new Vector4(g2.x, g2.y, g2.z,  w: 0f),
        new Vector4(g3.x, g3.y, g3.z,  w: 0f),
        new Vector4(g4.x, g4.y, g4.z,  w: 0f)
        );
}

Frequently called  1 usage
public static Vector3 EvalCatmullRomSplines(float t, Matrix4x4 geometry)
{
    var t2 :float = t * t;
    var tVector = new Vector4( x: 1,  y: t,  z: t2,  w: t2 * t);
    tVector = k_CatmullRomCoefficient * tVector;
    return geometry * tVector;
}
```

仓库地址：

https://github.com/eatdreamcat/CurveRendering