華東理工大學

计算机图形学

2023年11月

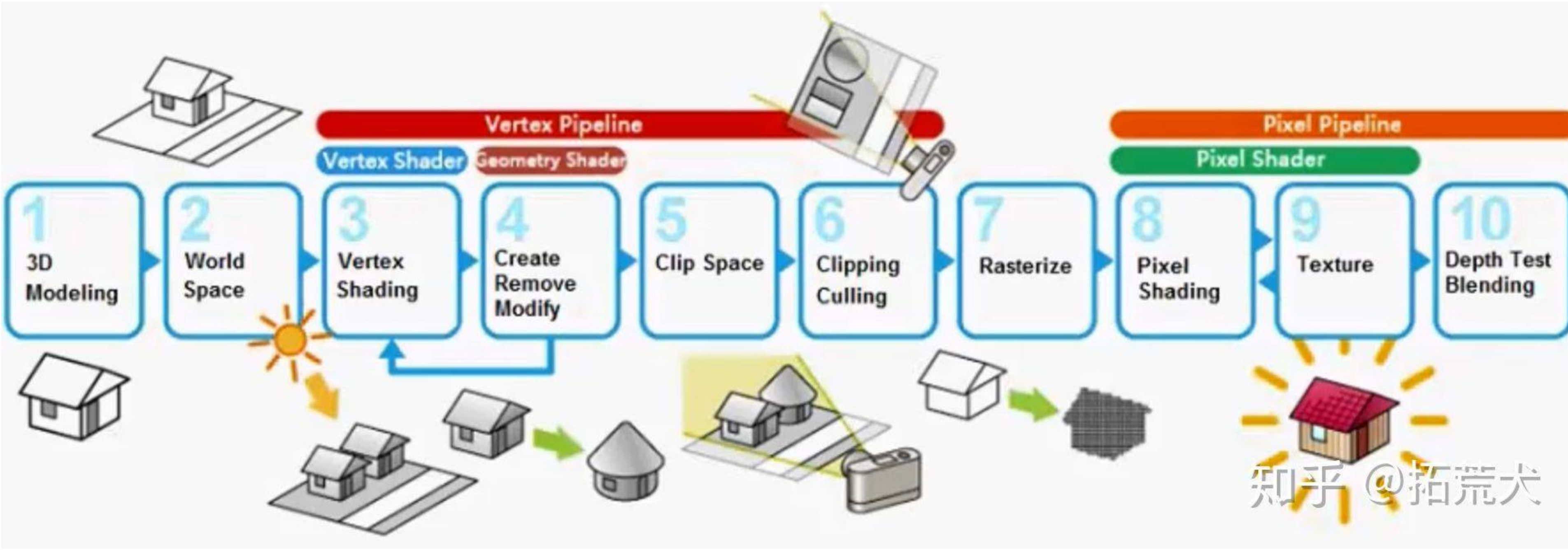
奉贤校区

勤 奋 求 实 / 励 志 明 德

08

曲线和曲面

Curve and Surface



Vertex Pipeline
Pixel Pipeline
Vertex Shader
Geometry Shader
Pixel Shader
1
3D Modeling
2
World Space
3
Vertex Shading
4
Create Remove Modify
5
Clip Space
6
Clipping Culling
7
Rasterize
8
Pixel Shading
9
Texture
10
Depth Test Blending
知乎 @拓荒犬

曲线和曲面

1 基本概念

2 三次样条

3 Bezier曲线曲面

4 B样条曲线曲面

5 有理样条曲线曲面

B样条曲线

- 定义

$$p(t)=\sum_{k=0}^{n} P_k B_{k,m}(t)$$

- de Boor点：P_k
- B样条控制多边形　n+1个顶点
- B样条基函数：是分段m 阶（m-1 次）多项式，它们由**节点向量**（$T=(t_0, t_1, \cdots t_{n+m})$）唯一决定；节点向量则是一串非减（non-decreasing）的实数序列。

m可取2到n+1之间的任意整数

- 曲线定义的范围：$[t_{m-1}, \cdots t_{n+1}]$

B样条曲线

$$B_{k,1}(t)=\begin{cases}1 & 若 t_k \le t < t_{k+1} \\ 0 & 其它\end{cases}$$

$$B_{k,m}(t)=\frac{t-t_k}{t_{k+m-1}-t_k}B_{k,m-1}(t)+\frac{t_{k+m}-t}{t_{k+m}-t_{k+1}}B_{k+1,m-1}(t)$$

- 参数说明
  - m是曲线的阶数，（m−1）为B样条曲线的次数，曲线在连接点处具有(m−2)阶连续。

m可取2到n+1之间的任意整数

B样条曲线

- t_k是节点值，非减序列

  $T=（t_0，\ t_1，\cdots t_{n+m}）$

  构成了$m-1$次B样条函数的节点矢量。

- 节点矢量分为三种类型：均匀的，开放均匀的和非均匀的。

节点表是生成基本函数表的关键参数，非减（non-decreasing）的实数序列，大小等于（控制点数量-1）+阶数+1

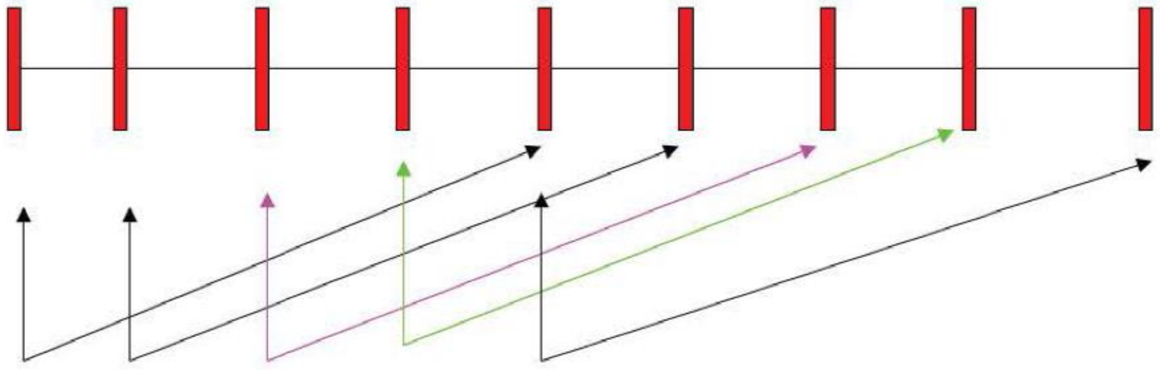
n + m +1

B样条基函数

- 以n=4, m=4为例：

$$p(t)=\sum_{k=0}^{n}P_k B_{k,m}(t)$$

$$t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8$$



NURBS曲线曲面的定义

- 定义

$$p(t)=\frac{\sum_{k=0}^{n} w_k P_k B_{k,m}(t)}{\sum_{k=0}^{n} w_k B_{k,m}(t)}$$

P_k为控制顶点；W_k是控制点的权因子，权因子越大，曲线越靠近对应的控制点；B(t)是B样条基函数。

Review

- 1. 给点控制点，B样条曲线的形状唯一？
- 否

- 2. 通常，为了减少B样条曲线某处的连续性，一般用什么方法？

- 插入重节点

Review

- 3. NURBS有哪些可以控制曲线形状？

- 节点表
- 权因子

de Casteljau算法

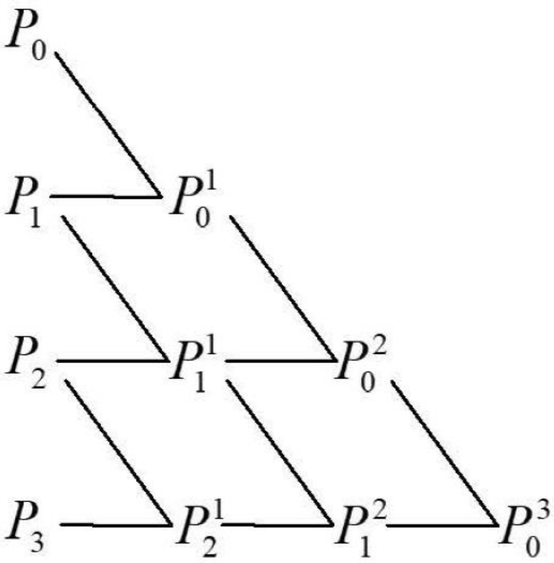
- 在工业应用中，经常需要计算曲线上参数t位置的点；
- 我们并不通过Bezier 曲线方程式来求解，而是通过一个递归的数值稳定的(numerical stable）算法。

- 我们可以得到Bezier 曲线的递推计算公式：

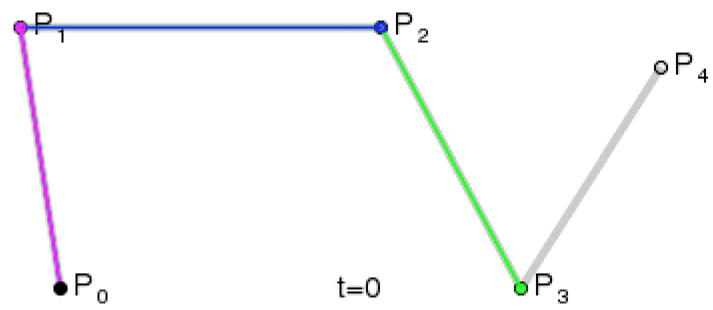
$$P(t) = P_0^n$$

$$P_i^k = \begin{cases} P_i & k = 0 \\ (1-t)P_i^{k-1} + tP_{i+1}^{k-1} & k = 1, 2, \ldots, n, \\ & i = 0, 1, \ldots, n-k \end{cases}$$

- 当n = 3 时，递推过程如下图所示：



- 实例



- 题目

一条3次Bézier曲线顶点为(10,0), (30,60), (130,60), (190, 0), 请问t=1/2处曲线的值为?

de Boor算法

- 计算B样条曲线的一点 P(t)，可以直接使用B样条的公式，但de Boor算法是一个更有效的算法。
- De Boor 算法：

$$P(t)=\sum_{i=0}^{n}P_iN_{i,k}(t)=\sum_{i=j-k+1}^{j}P_iN_{i,k}(t)$$

$$=\sum_{i=j-k+1}^{j}P_i\left[\frac{t-t_i}{t_{i+k-1}-t_i}N_{i,k-1}(t)+\frac{t_{i+k}-t}{t_{i+k}-t_{i+1}}N_{i+1,k-1}(t)\right]$$

$$=\sum_{i=j-k+1}^{j}\left[\frac{t-t_i}{t_{i+k-1}-t_i}P_i+\frac{t_{i+k-1}-t}{t_{i+k-1}-t_i}P_{i-1}\right]N_{i,k-1}(t)\qquad t\in[t_j,t_{j+1}]$$

- 令：

$$P_i^{[r]}(t)=\begin{cases}P_i, r=0, i=j-k+1, j-k+2,\cdots,j \\ \dfrac{t-t_i}{t_{i+k-r}-t_i}P_i^{[r-1]}(t)+\dfrac{t_{i+k-r}-t}{t_{i+k-r}-t_i}P_{i-1}^{[r-1]}(t), \\ \qquad r=1,2,\cdots,k-1; i=j-k+r+1, j-k+r+2,\cdots,j\end{cases}$$

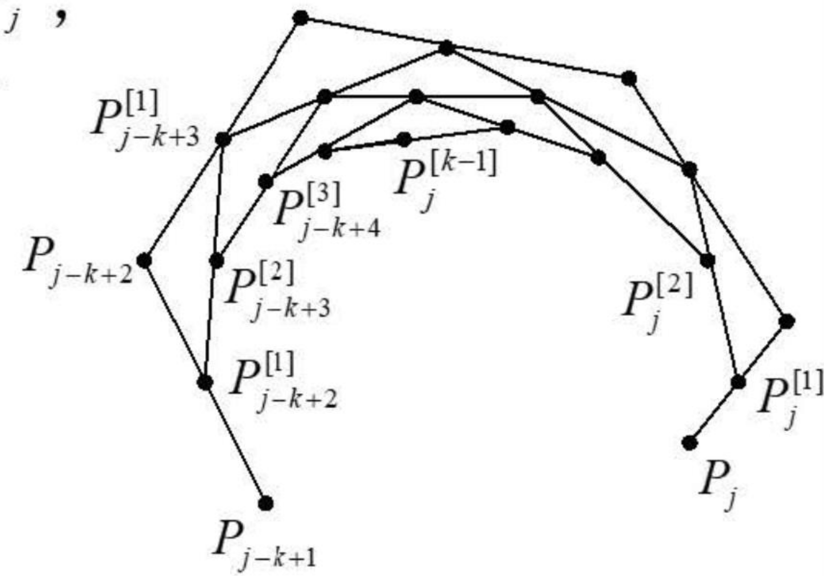
- 那么：

$$P(t)=\sum_{i=j-k+1}^{j}P_iN_{i,k}(t)=\sum_{i=j-k+2}^{j}P_i^{[1]}(t)N_{i,k-1}(t)$$

- 这就是 de Boor 算法。

- de Boor 算法的几何意义：
  - De Boor 算法有直观的几何解释：

    割角 (corner cutting)：
    - 每次使用线段 $P_i^{[r]}P_{i+1}^{[r]}$ 来切割角$P_i^{[r-1]}$
    - 初始多边形为$P_{j-k+1}P_{j-k+2}\cdots P_j$ ，经过 k–1 轮割角后，我们最终能够得到$P_j^{[r-1]}(t)$ 。



B样条的绘制

- 1. 采用定义或者de Boor方法

- 2. 用OpenGL提供的NURBS的函数。