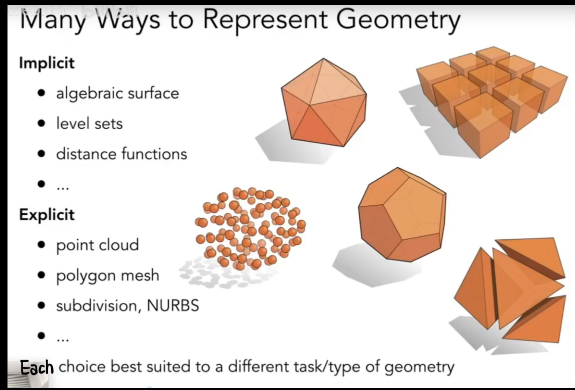
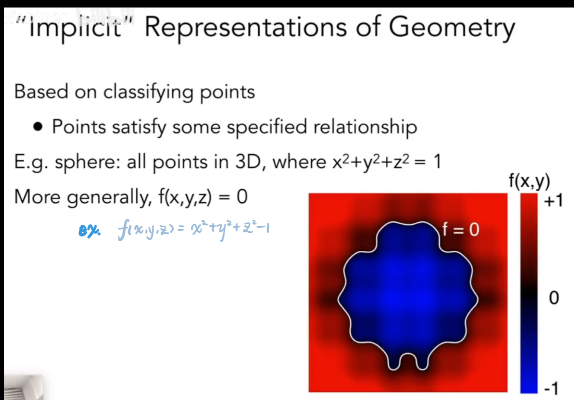
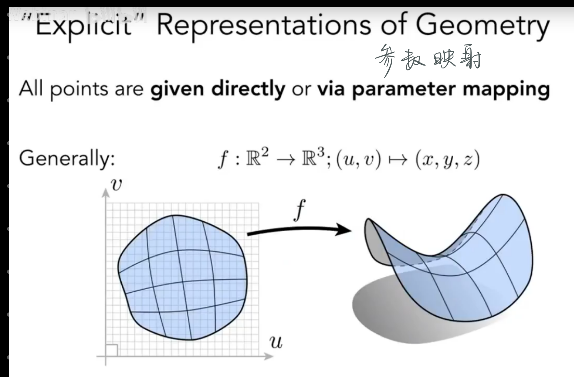


隐式 Implicit

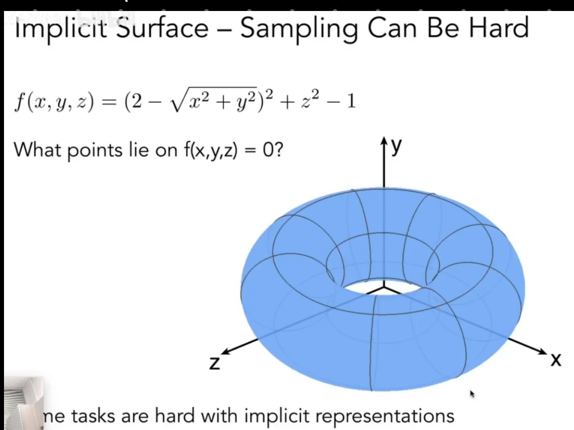
Explicit 显式

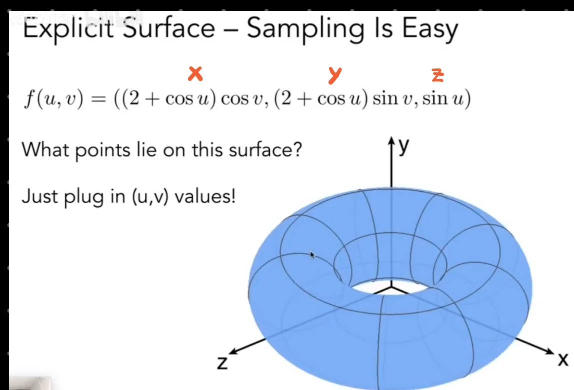




Hard to find all points / shape

遍历 u v 以得到形状

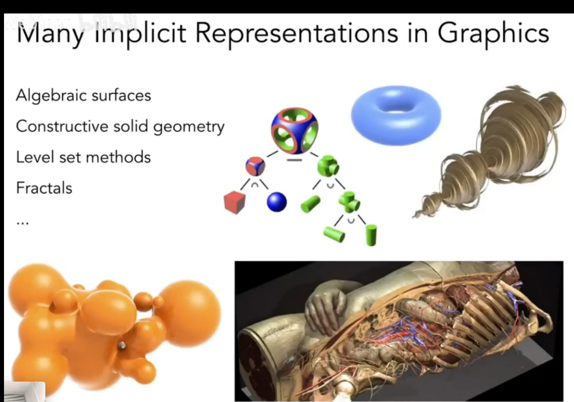




难以验证内外

但易于判断点是否在物体内外表

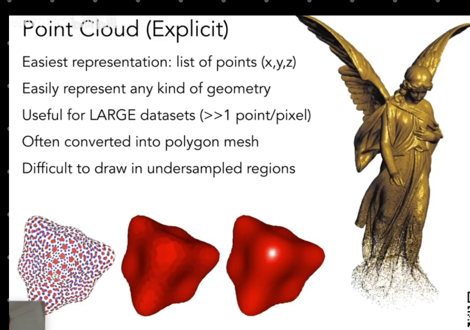
根据需要选择



point cloud 点云

通过密集点表示立体

- 典型运用：3D扫描



Algebraic Surfaces (Implicit)

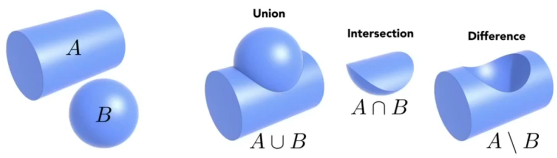
Surface is zero set of a polynomial in x, y, z

$x^2+y^2+z^2=1$ $\quad (R-\sqrt{x^2+y^2})^2+z^2=r^2$ $\quad (x^2+\frac{9y^2}{4}+z^2-1)^3 = x^2z^3+\frac{9y^2z^3}{80}$

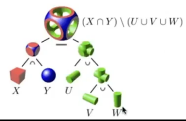
更复杂的图形如何咋整？

Constructive Solid Geometry

Combine implicit geometry via Boolean operations



Boolean expressions:

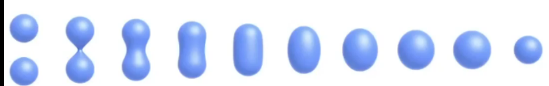


Distance Functions 距离函数

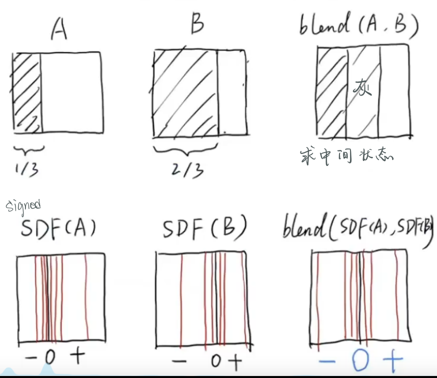
Instead of Booleans, gradually blend surfaces together using

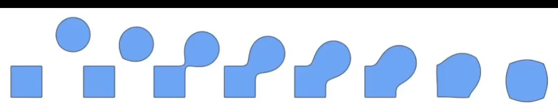
Distance functions:
giving minimum distance (could be **signed** distance)
from anywhere to object

内 负
外 正



An Example: Blending a moving boundary





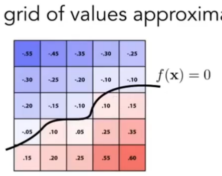
Level Set Methods (Also implicit)

(水平集)

当距离函数无法用式子表达时

Closed-form equations are hard to describe complex shapes

Alternative: store a grid of values approximating function



Surface is found where interpolated values equal zero

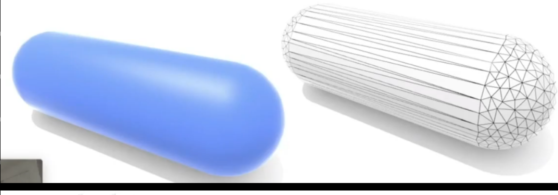
Provides much more explicit control over shape (like a texture)

polygon mesh

Polygon Mesh (Explicit)

Store vertices & polygons (often triangles or quads)

Easier to do processing / simulation, adaptive sampling



The Wavefront Object File (.obj) Format

Commonly used in Graphics research

Just a text file that specifies vertices, normals, texture coordinates **and their connectivities**

立方体

```
# This is a comment

v 1.000000 -1.000000 -1.000000
v 1.000000 -1.000000 1.000000
v -1.000000 -1.000000 1.000000
v -1.000000 -1.000000 -1.000000
v 1.000000 1.000000 -1.000000
v 0.999999 1.000000 1.000001
v -1.000000 1.000000 1.000000
v -1.000000 1.000000 -1.000000
                                  #8个点
vt 0.748573 0.750412
vt 0.749279 0.501284
vt 0.999110 0.501077
vt 0.999455 0.750380
vt 0.250471 0.500702
vt 0.249682 0.749677
vt 0.001085 0.750380
vt 0.001517 0.499994
vt 0.499422 0.500239
vt 0.500149 0.750166              存在共用点
vt 0.748355 0.998230
vt 0.500193 0.998728                ~24
vt 0.498993 0.250415              #12个纹理
vt 0.748953 0.250920                  坐标

           #6个法线
vn 0.000000 0.000000 -1.000000
vn -1.000000 -0.000000 -0.000000
vn -0.000000 -0.000000 1.000000
vn -0.000001 0.000000 1.000000
vn 1.000000 -0.000000 0.000000
vn 1.000000 0.000000 0.000001
vn 0.000000 1.000000 -0.000000
vn -0.000000 -1.000000 0.000000
   v n t
f 5/1/1 1/2/1 4/3/1               连接关系
f 5/1/1 4/3/1 8/4/1
f 3/5/2 7/6/2 8/7/2
f 3/5/2 8/7/2 4/8/2
f 2/9/3 6/10/3 3/5/3
f 6/10/4 7/6/4 3/5/4
f 1/2/5 5/1/5 2/9/5
f 5/1/6 6/10/6 2/9/6
f 5/1/7 8/11/7 6/10/7
f 8/11/7 7/12/7 6/10/7
f 1/2/8 2/9/8 3/13/8
f 1/2/8 3/13/8 4/14/8
```

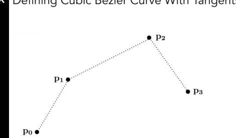
Curves

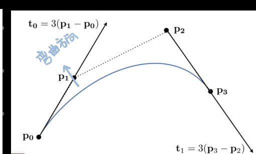
Bézier Curves

define curves with pivots

ex

Defining Cubic Bézier Curve With Tangents



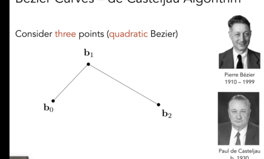


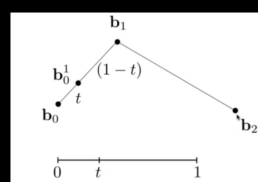
evaluating bezier curves

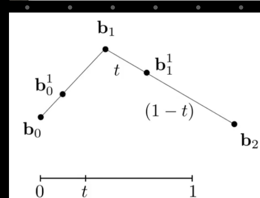
De Casteljau Algorithm

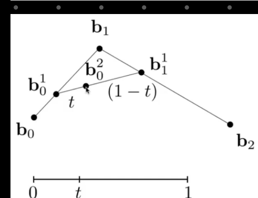
Bézier Curves – de Casteljau Algorithm

Consider three points (quadratic Bezier)



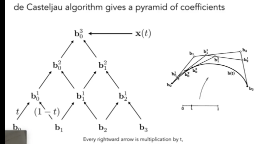






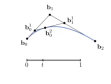
Bézier Curve – Algebraic Formula

de Casteljau algorithm gives a pyramid of coefficients



Every rightward arrow is multiplication by t,
Every leftward arrow by (1–t)

Example: quadratic Bézier curve from three points



$$\mathbf{b}_0^1(t) = (1-t)\mathbf{b}_0 + t\mathbf{b}_1$$
$$\mathbf{b}_1^1(t) = (1-t)\mathbf{b}_1 + t\mathbf{b}_2$$
$$\mathbf{b}_0^2(t) = (1-t)\mathbf{b}_0^1 + t\mathbf{b}_1^1$$
$$\mathbf{b}_0^2(t) = (1-t)^2\mathbf{b}_0 + 2t(1-t)\mathbf{b}_1 + t^2\mathbf{b}_2$$

对控制点进行插值

Fractals 分形 递归图形套娃

Exhibit self-similarity, detail at all scales

"Language" for describing natural phenomena

Hard to control shape!

Implicit Representations - Pros & Cons

Pros:

- compact description (e.g., a function)
- certain queries easy (inside object, distance to surface)
- good for ray-to-surface intersection (more later)
- for simple shapes, exact description / no sampling error
- easy to handle changes in topology (e.g., fluid)

Cons

Hard for complex objects

Bernstein form of a Bézier curve of order n:

$$\mathbf{b}^n(t) = \mathbf{b}_0^n(t) = \sum_{j=0}^{n} \mathbf{b}_j B_j^n(t)$$

n+1个控制点

Bézier curve order n (vector polynomial of degree n)

Bernstein polynomial (scalar polynomial of degree n)

Bézier control points (vector in $\mathbb{R}^N$)

Bernstein polynomials:

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Bernstein form of a Bézier curve of order n:

$$\mathbf{b}^n(t) = \sum_{j=0}^{n} \mathbf{b}_j B_j^n(t)$$

Example: assume n = 3 and we are in $\mathbb{R}^3$

i.e. we could have control points in 3D such as:

$$\mathbf{b}_0 = (0,2,3),\ \mathbf{b}_1 = (2,3,5),\ \mathbf{b}_2 = (6,7,9),\ \mathbf{b}_3 = (3,4,5)$$

These points define a Bezier curve in 3D that is a cubic polynomial in t:

$$\mathbf{b}^n(t) = \mathbf{b}_0\,(1-t)^3 + \mathbf{b}_1\,3t(1-t)^2 + \mathbf{b}_2\,3t^2(1-t) + \mathbf{b}_3\,t^3$$

Properties of Bézier Curves

Interpolates endpoints

- For cubic Bézier: $\mathbf{b}(0) = \mathbf{b}_0;\ \ \mathbf{b}(1) = \mathbf{b}_3$

Tangent to end segments

- Cubic case: $\mathbf{b}'(0) = 3(\mathbf{b}_1 - \mathbf{b}_0);\ \ \mathbf{b}'(1) = 3(\mathbf{b}_3 - \mathbf{b}_2)$

复制曲线

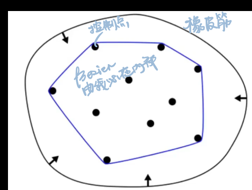
Affine transformation property 仿射 不可投影 后者和之前不是一个曲线了

仅变换控制点

- Transform curve by transforming control points

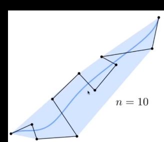
Convex hull property 凸包性质 曲线必在控制点凸包内

- Curve is within convex hull of control points



Piecewise 分段贝塞尔曲线

High order Bezier curves



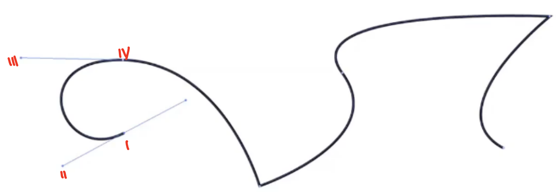
难以控制、难以绘制

<曲线上每一点都受全体控制点影响>

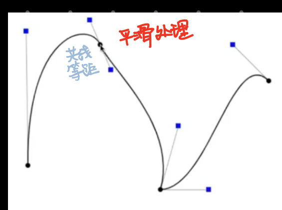
Piecewise Bézier Curves

Instead, chain many low-order Bézier curve

$\boxed{\textbf{Piecewise cubic Bézier}}$ the most common technique



Widely used (fonts, paths, Illustrator, Keynote, …)



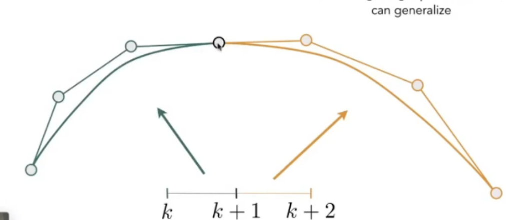
Piecewise Bézier Curve – Continuity

Two Bézier curves

$$\mathbf{a} : [k, k+1] \to \mathbb{R}^N$$

$$\mathbf{b} : [k+1, k+2] \to \mathbb{R}^N$$

Assuming integer partitions here, can generalize



C_0 continuity $a_n = b_0$ 相接即可

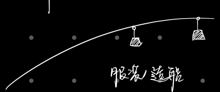
C_1 continuity $a_n = b_0 = \frac{1}{2}(a_{n-1} + b_1)$ 切线共线 <一阶导>

C_2 … 曲率连续 <二阶导> 多用于工业制造

扩展：B样条 B splines

nurbs

样条曲线 spline



B 样条 B-spline

贝塞尔曲线的扩展 → 局部性

控制点影响范围有局部性

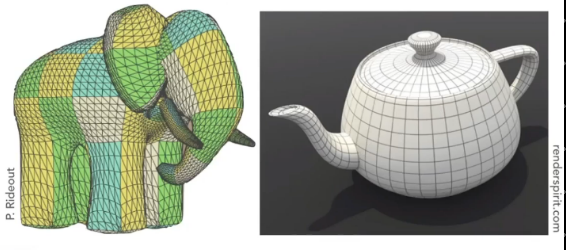
极其复杂 MAX

非均匀有理B样条 NURBS

空间中的贝塞尔曲线 — 贝塞尔曲面

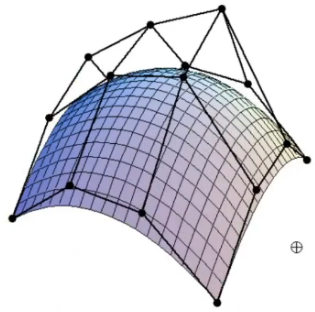
Bézier Surfaces

Extend Bézier curves to surfaces



Ed Catmull's "Gumbo" model   Utah Teapot

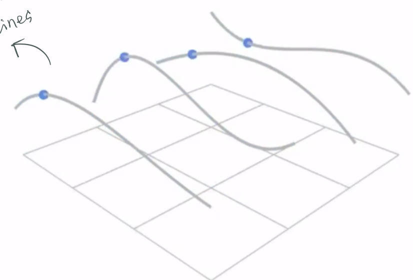
Bicubic Bézier Surface Patch

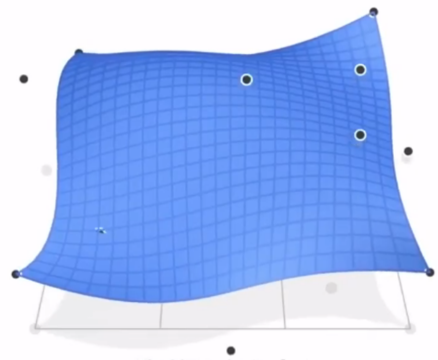


Bezier surface and 4 x 4 array of control points

Visualizing Bicubic Bézier Surface Patch

constrain into lines

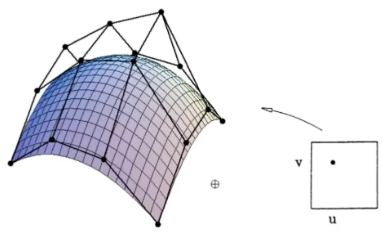




Bicubic Bezier Surface

Evaluating Surface Position For Parameters (u,v)

For bi-cubic Bezier surface patch,
  Input: 4x4 control points
  Output is 2D surface parameterized by (u,v) in $[0,1]^2$

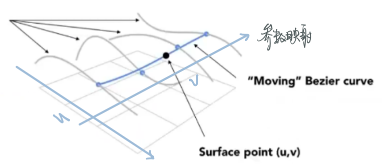


Method: Separable 1D de Casteljau Algorithm

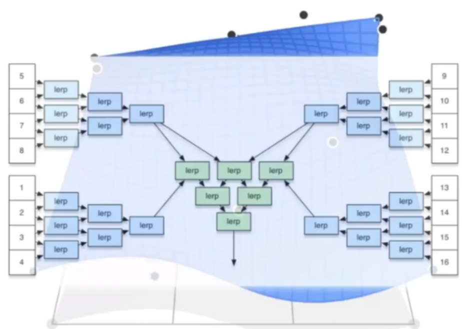
Goal: Evaluate surface position corresponding to (u,v)

(u,v)-separable application of de Casteljau algorithm

- Use de Casteljau to evaluate point u on each of the 4 Bezier curves in u. This gives 4 control points for the "moving" Bezier curve
- Use 1D de Casteljau to evaluate point v on the "moving" curve

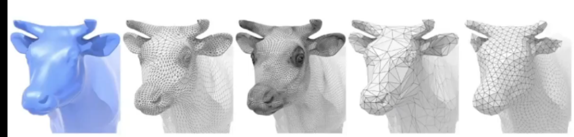


Method: Separable 1D de Casteljau Algorithm



Mesh Operation – Geometry Processing 几何处理

- Mesh subdivision 网格细分
- Mesh simplification 简化
- Mesh regularization 正规化 – 有用之性质



upsampling   downsampling

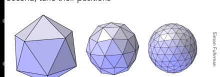
发明算法的人叫 Loop

Loop Subdivision

Common subdivision rule for triangle meshes

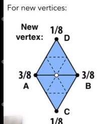
First, create more triangles (vertices)

Second, tune their positions



Loop Subdivision — Update

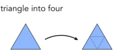
For new vertices:



Update to:
3/8 * (A + B) + 1/8 * (C + D)

Loop Subdivision

- Split each triangle into four

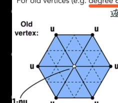


- Assign new vertex positions according to weights
  - New / old vertices updated differently

Loop Subdivision — Update

For old vertices (e.g. degree 6 vertices here):

连n条边的n度



Update to:
(1 - n*u) * original_position +
u * neighbor_position_sum

n: vertex degree

u: 3/16 if n=3, 3/(8n) otherwise