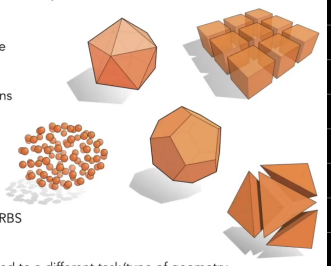
Many Ways to Represent Geometry

Implicit

- algebraic surface
- level sets
- distance functions
- ...

Explicit

- point cloud
- polygon mesh
- subdivision, NURBS
- ...



Each choice best suited to a different task/type of geometry

隐式 Implicit　　Explicit 显式

"Implicit" Representations of Geometry

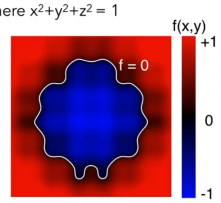
Based on classifying points

- Points satisfy some specified relationship

E.g. sphere: all points in 3D, where $x^2+y^2+z^2 = 1$

More generally, f(x,y,z) = 0

$f(x,y,z) = x^2+y^2+z^2-1$

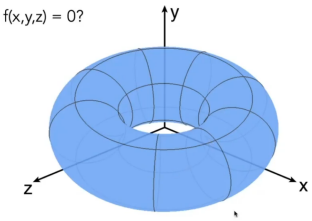


Hard to find all points / shape

Implicit Surface – Sampling Can Be Hard

$f(x,y,z) = (2-\sqrt{x^2+y^2})^2 + z^2 - 1$

What points lie on f(x,y,z) = 0?



Some tasks are hard with implicit representations

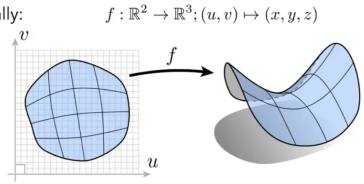
但是判断点是否在物体内外是

"Explicit" Representations of Geometry

参数映射

All points are **given directly** or **via parameter mapping**

Generally: $f : \mathbb{R}^2 \to \mathbb{R}^3; (u,v) \mapsto (x,y,z)$



遍历 u v 可得到物体

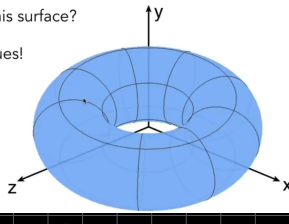
Explicit Surface – Sampling Is Easy

$f(u,v) = ((2+\cos u)\cos v, (2+\cos u)\sin v, \sin u)$

x　y　z

What points lie on this surface?

Just plug in (u,v) values!



很好验证内外

根据需要选择

Many Implicit Representations in Graphics

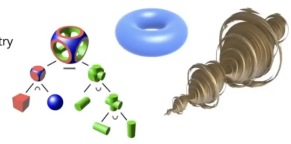
Algebraic surfaces

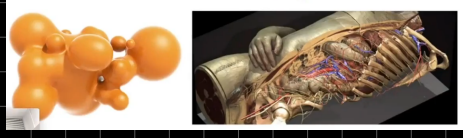
Constructive solid geometry

Level set methods

Fractals

...





point cloud 点云

通过密集点表示立体

一典型应用：3D 扫描

Point Cloud (Explicit)

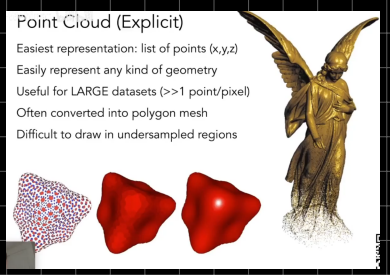
Easiest representation: list of points (x,y,z)

Easily represent any kind of geometry

Useful for LARGE datasets (>>1 point/pixel)

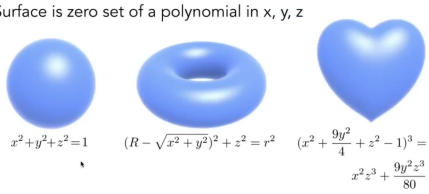
Often converted into polygon mesh

Difficult to draw in undersampled regions



Algebraic Surfaces (Implicit)

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

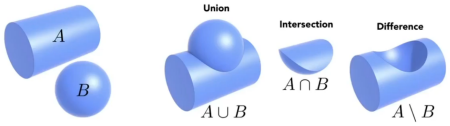


$x^2+y^2+z^2=1$ $(R-\sqrt{x^2+y^2})^2+z^2=r^2$ $(x^2+\frac{9y^2}{4}+z^2-1)^3=x^2z^3+\frac{9y^2z^3}{80}$

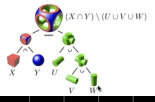
更复杂的图形如何建模？

Constructive Solid Geometry

Combine implicit geometry via Boolean operations



Boolean expressions:



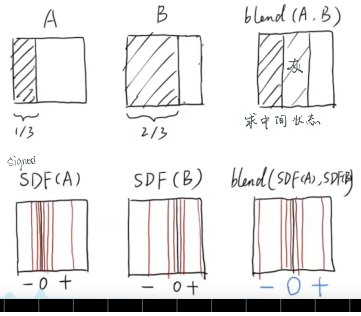
Distance Functions 距离函数

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



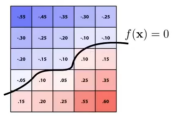
Level Set Methods (Also implicit)

(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

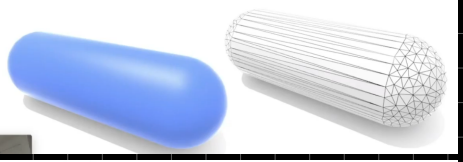
Provides much more explicit control over shape (like a texture)

polygon mesh

Polygon Mesh (Explicit)

Store vertices & polygons (often triangles or quads)

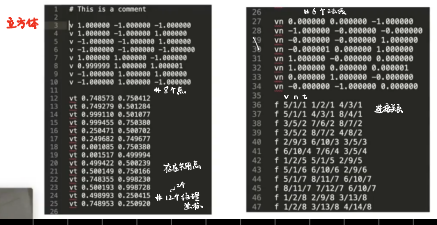
Easier to do processing / simulation, adaptive sampling



The Wavefront Object File (.obj) Format

Commonly used in Graphics research

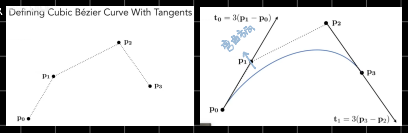
Just a text file that specifies vertices, normals, texture coordinates **and their connectivities**



Curves

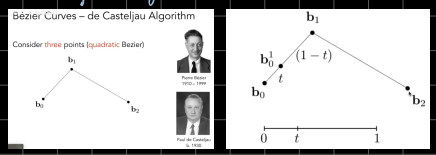
Bézier Curves

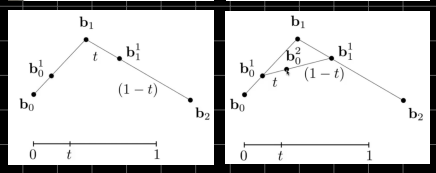
define curves with pivots

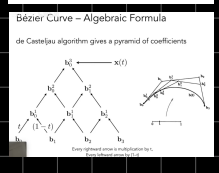


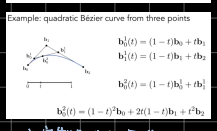
evaluating bezier curves

De Casteljau Algorithm









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!

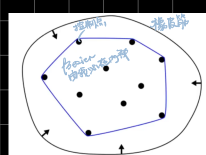
Implicit Representations - Pros & Cons

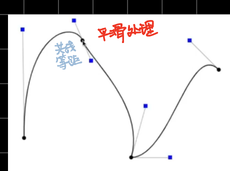
Pros:

- 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





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

拓展：B样条 B splines nurbs

空间中的贝塞尔曲线

Bernstein form of a Bézier curve of order n: n+1个控制点

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

Bézier curve order n (vector polynomial of degree n)

Bernstein polynomial (scalar polynomial of degree n)

Bézier control points (vector in 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 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: $\mathbf{b}'(0) = 3(\mathbf{b}_1 - \mathbf{b}_0);\ \ \mathbf{b}'(1) = 3(\mathbf{b}_3 - \mathbf{b}_2)$

Affine transformation property 仿射 不可投影

- Transform curve by transforming control points

Convex hull property 凸包性质

- Curve is within convex hull of control points

Piecewise 分段贝塞尔曲线

High order Bezier curves

Piecewise Bézier Curves

Instead, chain many low-order Bézier curve

Piecewise cubic Bézier the most common technique

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

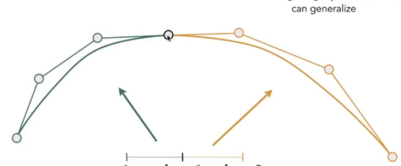
Piecewise Bézier Curve – Continuity

Two Bézier curves

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

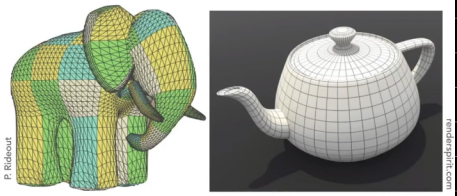
$$\mathbf{b} : [k+1, k+2] \to \mathbb{R}^N$$

Assuming integer partitions here, can generalize



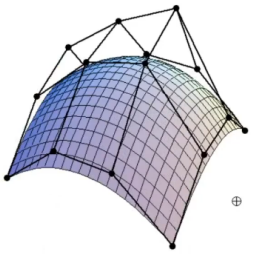
Bézier Surfaces

Extend Bézier curves to surfaces



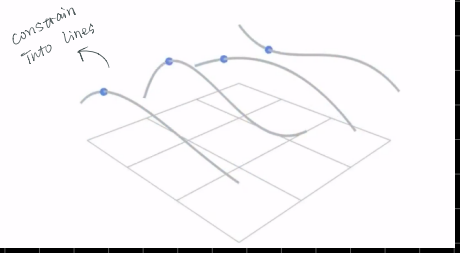
Ed Catmull's "Gumbo" model Utah Teapot

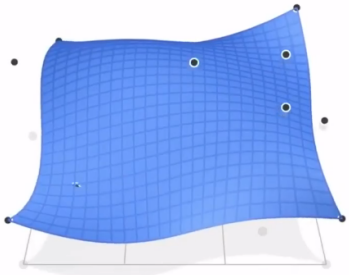
Bicubic Bézier Surface Patch



Bezier surface and 4 x 4 array of control points

Visualizing Bicubic Bézier Surface Patch

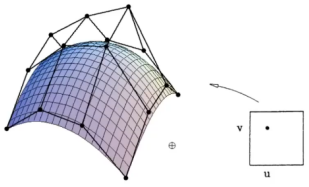




Bicubic Bezier Surface

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 $[0,1]^2$

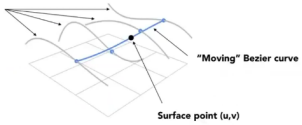


Method: Separable 1D de Casteljau Algorithm

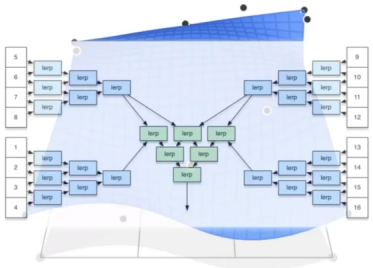
Goal: Evaluate surface position corresponding to (u,v)

(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 – 有用之世界