Geometry

Ways to Represent Geometry

Implicity

Definition

based on classifying points

example

- algebraic surface, e.g. $f(x, y, z) = 0$
- constructive solid geometry(CSG): combine implicit geometry via Boolean operations
- distance functions: giving minimum distance (could be signed distance) from anywhere to object
- level set method

Pros

- compact description
- certain queries easy
- good for ray-to-surface intersection
- for simple shapes, exact description
- easy to handle changes in topology

Cons

- difficult to model complex shapes

Explicit

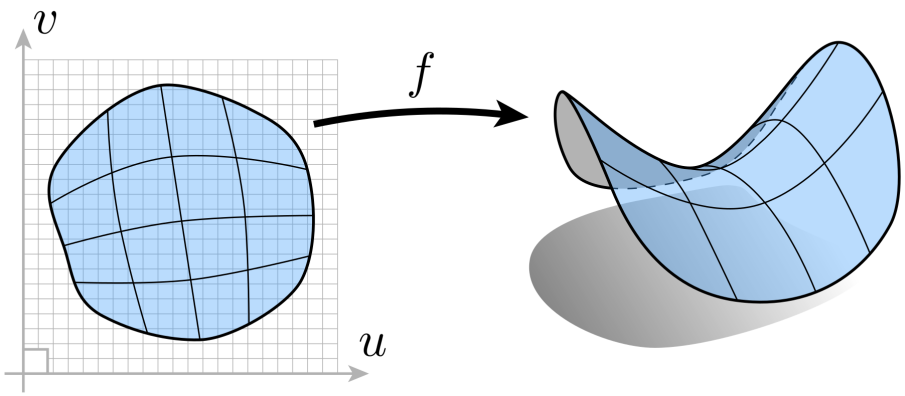
definition

all points are given directly or via parameter mapping

example

- point cloud: list of points (x, y, z)
- polygon mesh: easier to do processing/simulation, adaptive sampling, using wavefront object file(.object) to store info
- sampling is easy, e.g. $f(u, v) = ((2 + \cos u) \cos v, (2 + \cos u) \sin v, \sin u)$
- hard to tell if a point is inside/outside

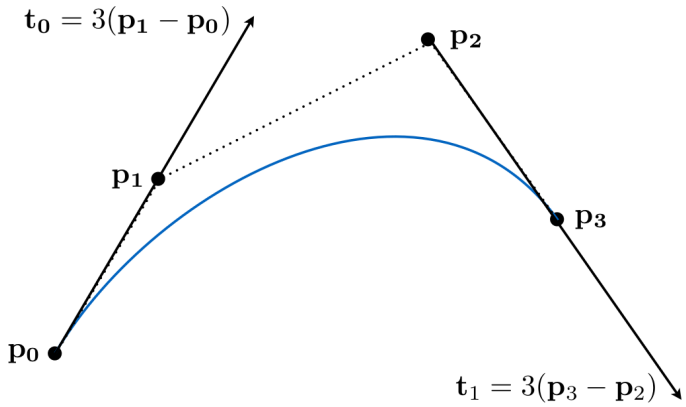
Generally: $f : \mathbb{R}^2 \to \mathbb{R}^3; (u, v) \mapsto (x, y, z)$



Bezier Curves

Idea

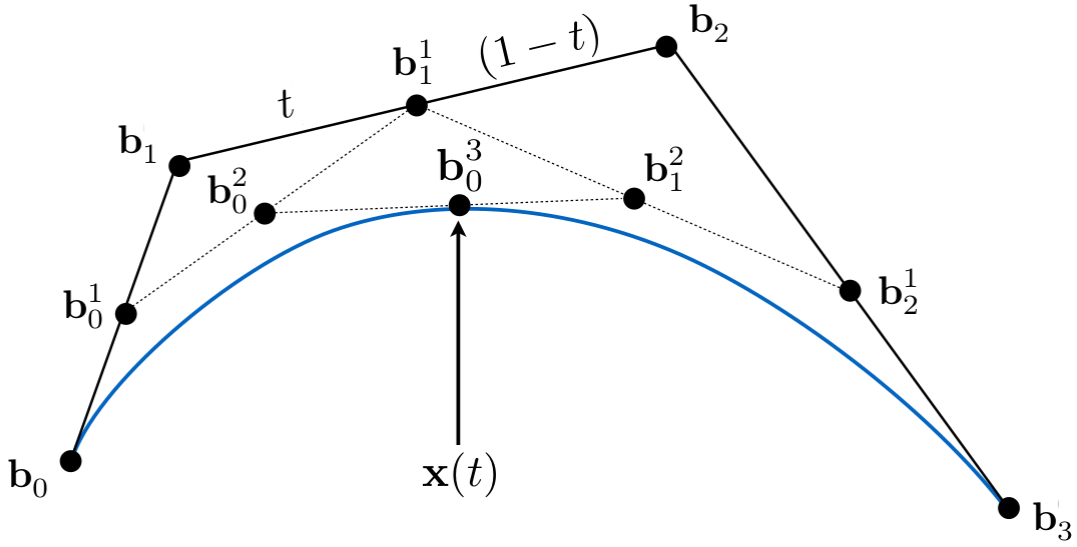
defining cubic Bezier curve with tangents



de Casteljau Algorithm

Four input points in total

Same recursive linear interpolations



Algebraic Formula

Bernstein form of a Bezier curve of order n:

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

where $B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$ is Bernstein polynomials

Properties of Bezier Curves

- $\mathbf{b}(0) = \mathbf{b}_0,\ \mathbf{b}(1) = \mathbf{b}_n$
- $\mathbf{b}'(0) = n(\mathbf{b}_1 - \mathbf{b}_0)$
- affine transformation property 对曲线作仿射变换等价于对控制点作仿射变换后再生成曲线
- convex hull property: curve is within convex hull of control points

Piecewise Bezier Curves

Definition

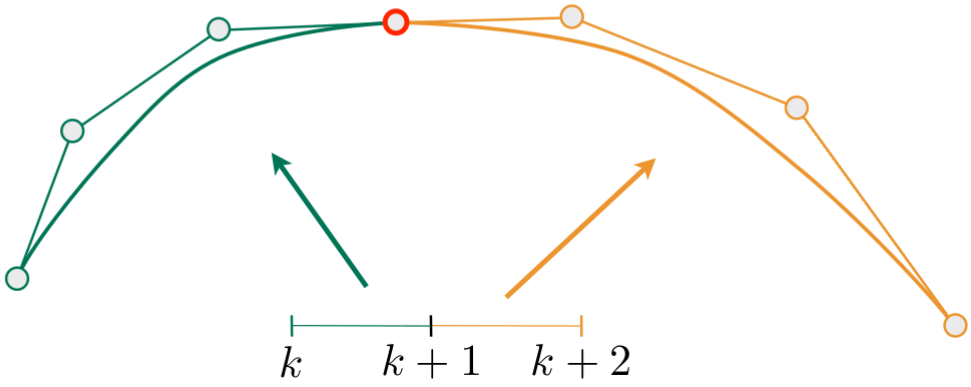
chain many low-order Bezier curve

Remark

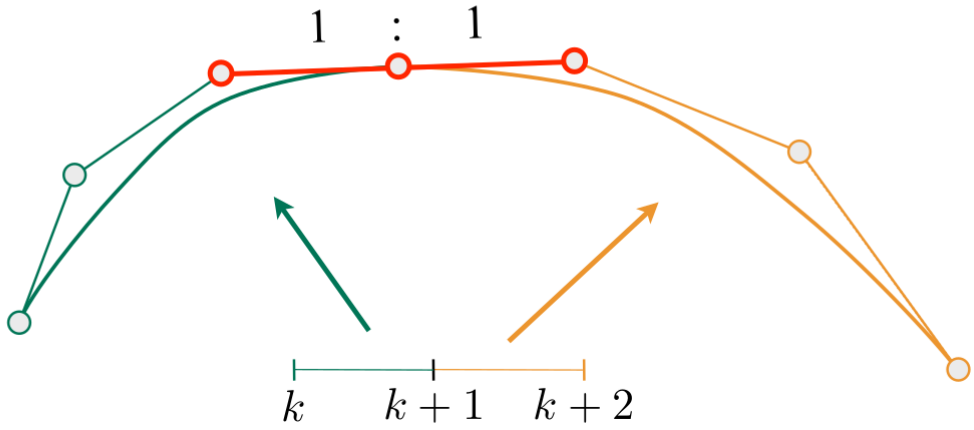
piecewise cubic Bezier the most common technique

Continuity

C^0 continuity: $\mathbf{a}_n = \mathbf{b}_0$



C¹ continuity: $\mathbf{a}_n = \mathbf{b}_0 = \frac{1}{2}(\mathbf{a}_{n-1} + \mathbf{b}_1)$



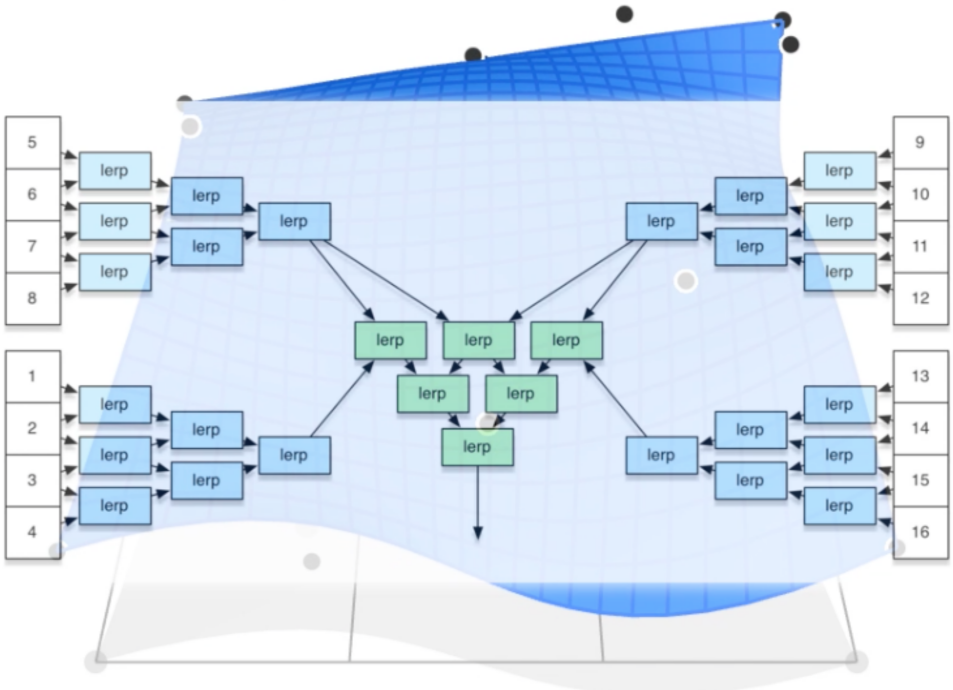
Remark

跟实变函数中对连续函数的连续性要求一样

Other types of splines

- spline
- B-splines

Bezier Surfaces



Mesh Operations: Geometry Processing

Mesh Subdivision

Idea

1. create more triangles(vertices)
2. tune their positions

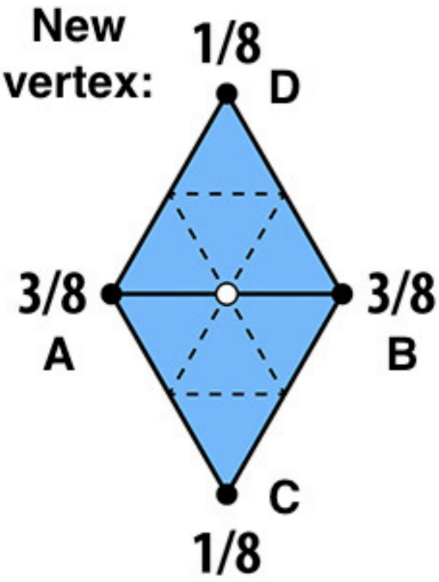
Loop Subdivision

1. split each triangle into four
2. assign new vertex position according to weights

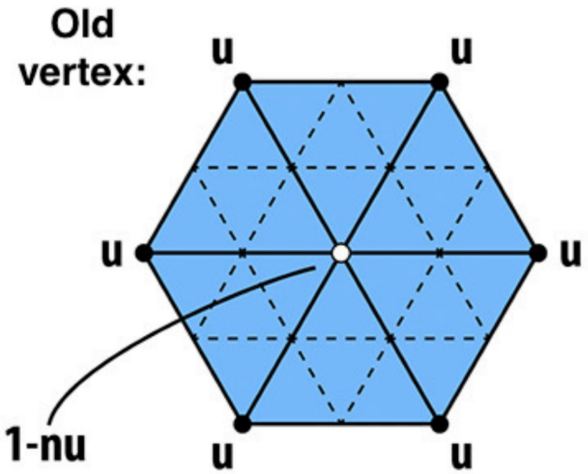
Remark

发明这个算法的人名字里有Loop

Update Vertex



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



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

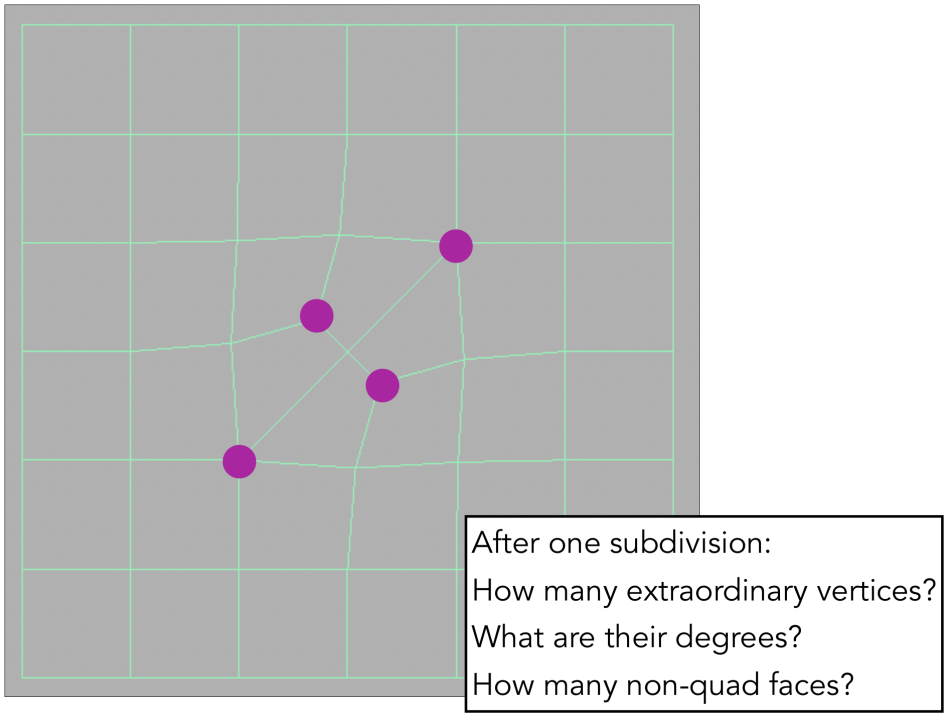
n: vertex degree

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

Catmull-Clark Subdivision(General Mesh)

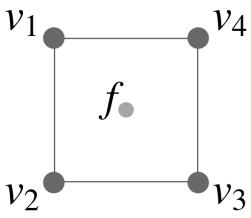
Definition

- eon-quad face
- extraordinary vertex(degree!=4)



Update Vertex

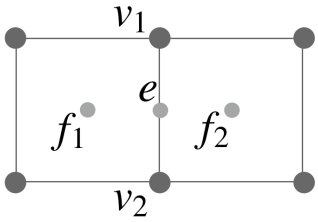
Face point

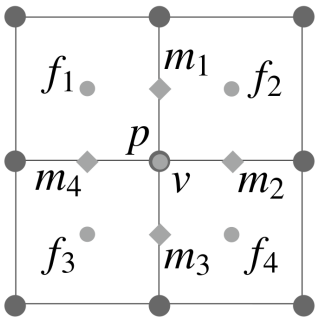


$$f = \frac{v_1 + v_2 + v_3 + v_4}{4}$$

$$e = \frac{v_1 + v_2 + f_1 + f_2}{4}$$

Edge point





Vertex point

$$v = \frac{f_1 + f_2 + f_3 + f_4 + 2(m_1 + m_2 + m_3 + m_4) + 4p}{16}$$

m midpoint of edge
p old "vertex point"

Mesh Simplification

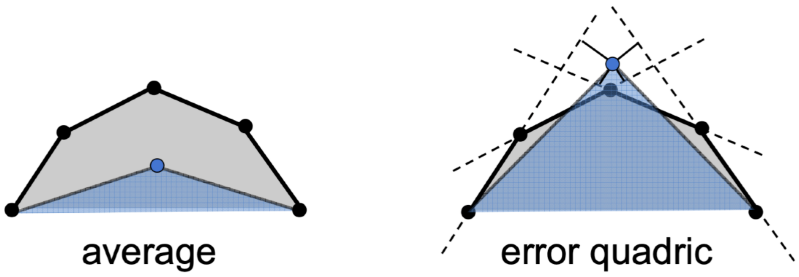
Goal

reduce number of mesh elements while maintaining the overall shape

Edge Collapsing

Quadric Error Metrics

minimized quadric error(L2 distance)



Algorithm

Iteratively collapse edges

Which edges? Assign score with quadric error metric*

- approximate distance to surface as sum of distances to planes containing triangles
- iteratively collapse edge with smallest score
- greedy algorithm... great results!

Mesh Regularization