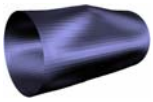
Subdivision I: The Basic Ideas

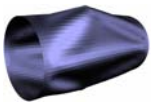
Peter Schröder

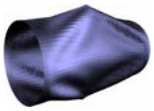
Modeling Geometry

Surface representations

- large class of surfaces
- interactive manipulation
- numerical modeling







Parameterized Surfaces

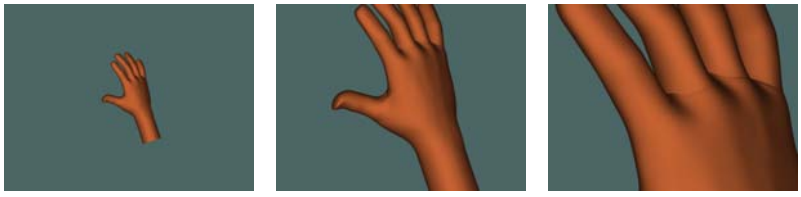
What about NURBS?

- spline patches are great, but…
  - difficult to stitch them together
    - trimming… (what a nightmare)
  - can be slow and cumbersome
  - we need scalable algorithms for the whole range from patches to fine meshes

Complex Shapes

Example: Building a hand

- Woody's hand from Pixar's Toy Story

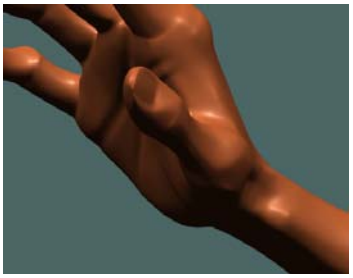


- very, very difficult to avoid seams

No More Seams

Subdivision solves the "stitching" problem

- a *single* smooth surface is defined
- example:
  - Geri's hand (Geri's Game; Pixar)



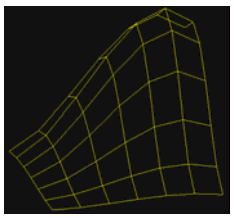
Control Polyhedron

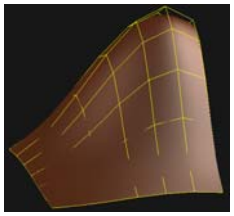
How are patch based surfaces controlled?

- control points (CVs)
- piecewise polynomial
  - write out coefficients

OR

  - procedural through subdivision!

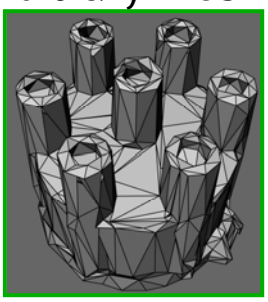




27 Years of Subdivision

The beginning

- create smooth surfaces out of arbitrary meshes



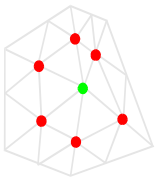
27 Years of Subdivision

The beginning

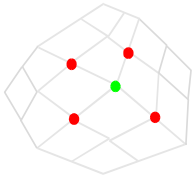
- the need to generalize spline patch modeling to arbitrary surfaces
  - Catmull-Clark
    - generalizes bi-cubic patches
  - Doo-Sabin
    - generalizes bi-quadratic patches

Extraordinary Vertices

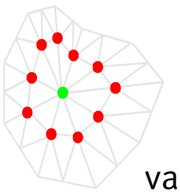
Triangle meshes Quad meshes



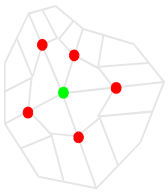
regular



valence 6 valence 4



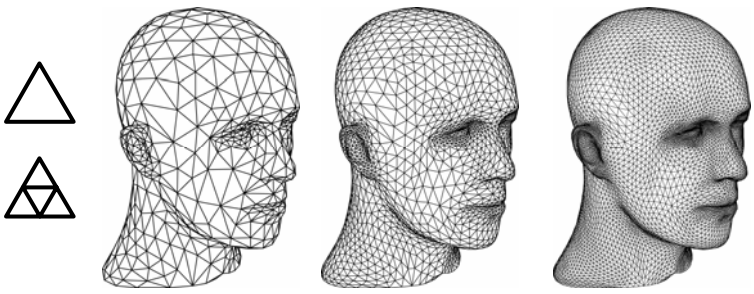
irregular



valence ≠ 6 valence ≠ 4

What is Subdivision?

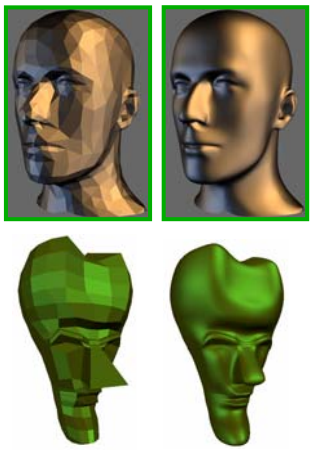
Define a smooth surface as the limit of a sequence of successive refinements



Why Subdivision?

Many advantages

- arbitrary topology
- scalable algorithms
- level of detail
- easy to implement
- efficient

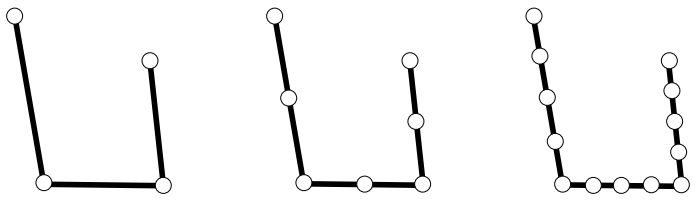


Subdivision in 1D

The simplest example

- piecewise linear subdivision

$$x_n = 1/2(x_l + x_r)$$
$$y_n = 1/2(y_l + y_r)$$



Subdivision in 1D

A more interesting example

- the 4pt scheme

$$p_{2i+1}^{j+1} = 1/16(-p_{i-1}^{j} + 9p_i^j + 9p_{i+1}^j - p_{i+2}^j)$$

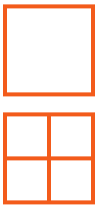
- Chaikin's algorithm

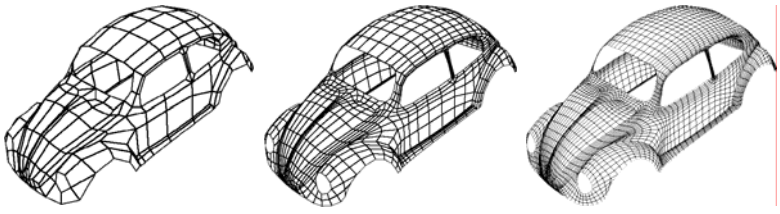
$$p_{2i}^{j+1} = 1/4(3p_i^j + p_{i+1}^j)$$
$$p_{2i+1}^{j+1} = 1/4(p_i^j + 3p_{i+1}^j)$$

Subdivision in 2D

Quadrilateral

- interpolating
  - Kobbelt scheme

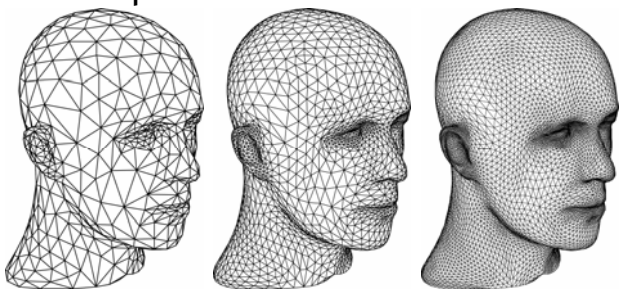


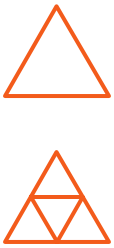


Subdivision in 2D

Triangular

- approximating
  - Loop scheme

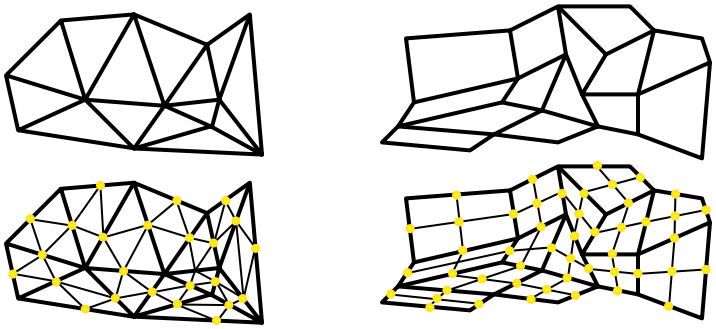




The Basic Setup

Topological rule

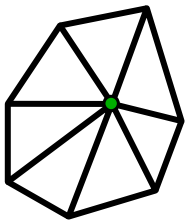
- modify connectivity



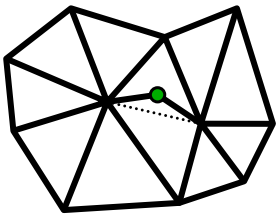
The Basic Setup

Geometric rule

- compute geometric positions
  - local linear combination of points



even at level i

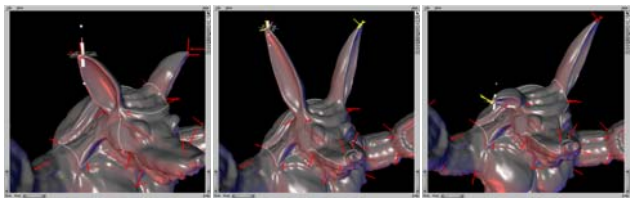


odd at level i

Some Conditions I

Subdivision rules should

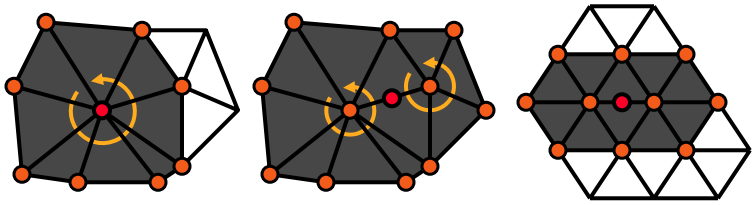
- be floating point efficient
- have compact support
  - influence of control point is finite



Some Conditions II

Subdivision rules should

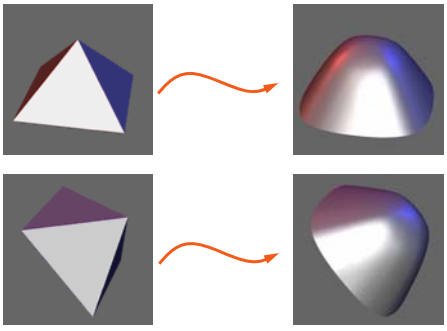
- have local definition
  - stencil weights only depend on the structure of a small neighborhood



Some Conditions III

Subdivision rules should

- be affinely invariant



Some Conditions IV

Subdivision rules should

- be simple
  - only a small set of different stencils
  - preferably precomputed offline
- achieve some order of smoothness
  - C^1 easy, C^2 much harder
  - perhaps we really want fairness...

Some Questions

Is Subdivision modeling inherently polygonal?

- NO. The surface consists of patches, we just build them procedurally
  - all rendering is polygonal...
- AND: no more worries about continuity conditions!

Some Questions

Subdivision patches

- do they have a parameterization?
- are they easy to evaluate?
- are they easy to implement?
- are they efficient?

All of the above and lots more!

- multiresolution, compression, etc.

Details

Begin with 1D setting

- review of splines
- splines through refinement
- general subdivision
  - not just splines
- some hints of analysis
  - continuity