Surfaces

What is a surface?

Surfaces cs. Curves
Surfaces vs. Solids
Boundary Representations
Kinds of Surfaces

Polygon Meshes
  (Triangle Meshes)

Smooth Surface Representations
Tesselation

How do we break things into triangles?

Even bigger polygons
Good Triangles

Avoid sharp angles
Avoid “skinny” triangles
Good Connections

Avoid holes (unintended)
Avoid cracks

Avoid cracks that appear because of errors
Triangles share vertices

Good:
Points shared

Bad:
T-Intersection – might crack

T - Intersection
Polygon Soup

- Random Assortment
- Unstructured
  - At least get ordering right
- Tells little about how polygons connect
- Lots of redundancy
Cube Soup

```c
struct Triangle Cube[12] =
    {{{1,1,1},{1,0,0},{1,1,0}},
    {{1,1,1},{1,0,1},{1,0,0}},
    {{0,1,1},{1,1,1},{0,1,0}},
    {{1,1,1},{1,1,0},{0,1,0}},
    ...
};
```
Polygon Soup

- **Advantages**
  - Easy

- **Problems**
  - Redundancy
  - No global info
  - No open/closed info
  - Hard to edit
  - Hard to prevent degeneracies
  - No non-local information
    - Is it closed?
    - Is it connected?
    - Is this an edge or internal?
Mesh Data Structures

Share vertices
Share edges

Find triangles around a point
Find neighbor triangles
“Walk” around edges in order
Cut edges / faces (preserve connections)
Parameterized Surfaces (smooth)
Extrusions

2D shape + vector
Surface of Revolution

Curve + Axis
Generalize Cylinder

1 – Cylinder
2 – Axis not a line segment
3 - Radius not constant (cone)
4 – Cross section not a circle
5 – Cross section not constant
Many names

Generalized Sweeps
Sweep Surfaces
Lofted Surfaces
Free-Form Surfaces
Parametric forms

Curve (3D)
\[ x, y, z = f(u) \]

Surface (3D)
\[ x, y, z = f(u, v) \]
Parametric Surfaces

- Define points on the surface in terms of two parameters
- Simplest case: bilinear interpolation

\[ x(s,0) = (1-s)P_{0,0} + sP_{1,0} \]
\[ x(s,1) = (1-s)P_{0,1} + sP_{1,1} \]
\[ x(s,t) = (1-t)x(s,0) + tx(s,1) \]

\[ F_{0,s} = 1-s, \quad F_{1,s} = s \]
\[ F_{0,t} = 1-t, \quad F_{1,t} = t \]

\[ x(s,t) = \sum_{i=0}^{1} \sum_{j=0}^{1} P_{i,j} F_{i,s}(s) F_{j,t}(t) \]
Bilinear Patches

- Edges are lines (so its easy)
- Patches are not flat (actually are curved)
- For a specific u, line in v
- For a diagonal line in u,v, a curve (quadratic actually)
- How do I cut a circular hole in the patch?
- (and bilinear is the easiest!)
Tensor Product Surfaces

- Polynomial in $u$ and $v$

- Just like with curves, coefficients aren’t the easiest – so switch bases

- Just a lot more control points
  - $D^2$ (16 for cubics!)

- A nightmare to derive…

- Note for fixed $u$ or $v$, its just a polynomial in the other variable
  - Patch edges are polynomial curves

$$\sum \sum_{i<d, j<d} a_{ij} u^i v^j$$
Nightmare!

How to control the patches?
How to connect the patches?
How to cut holes in the patches?
What if you need to make complex shapes?
Subdivision to the rescue!

Start with a polygon mesh
   Easy to design!

Make a “smoothing rule”
Iterate until smooth
Limit surfaces (really smooth)
Regular Meshes

- Triangle “Grids” (regular hex patterns)
- Quad Grids (squares)
- Semi-Regular Meshes (few “extraordinary” points)
How to divide triangles

- Need to do all triangles the same way
  - Can’t break edges on one side, not the other
- Break edges at midpoint
  - Common way each triangle -> 4 triangles
  - All new points are ordinary (or edges)
- Don’t break edges (Uncommon triangle->3 triangle)

Midpoint (4) scheme
very common

Uncommon 3-scheme
Does exist
How to divide other polygons

- Middle of Edges + Center of Face
  - Face point connects to edge points
- After 1 subdivision, everything is a quad
- All new points are ordinary points (or edges)
- 2 kinds of new points (edges and faces)
Important Subdivision Schemes

Butterfly (Modified Butterfly)
  Interpolating Triangle Scheme

Loop
  Non-Interpolating Triangle Scheme

Catmull-Clark
  Non-Interpolating Quads
Butterfly Scheme

- Stationary (interpolating) scheme
- Only rule inserts new points between existing ones
- Regular mesh -> regular mesh
- $C(1)$ at ordinary points

\[ v = \frac{1}{2}a + \frac{1}{8}b - \frac{1}{16}c \]
In Practice..

Modified Butterfly

More complex stencil

Rules for extraordinary points

Smooth C(1) – but often wiggly (not fair)
Loop Subdivision

- Named for Charles Loop (not because of loops in the rules)

- Approximating Scheme
  - New Points from close neighborhood of edge
  - Old points are then moved based on their neighbors (including new ones)
Loop Subdivision

• Extra Ordinary Points
  – Center = 1 – k B
  – Each connected point = B
  – B = $1/n \left( \frac{5}{8} - \left( \frac{3+2\cos(2\pi / n))^2}{64} \right) \right)
  – Note: this gives the same answer as the ordinary case
  – B = $3 / (n (n+2))$ (simpler version, really close, but not exact)

• Use special edge rules
  – Edge points at midpoints
  – Old points = $1/8 \ 3/4 \ 1/8$
Catmull-Clark Subdivision

- Regular Case is quads
- Same rules apply to non-quads
- Only have non-quads at first iteration

- Generalization of cubic B-Splines
  - On uniform mesh, gives same things
  - But works on non-uniform meshes
Catmull-Clark Rules

- Face point = center of polygon
- Edge points = average of 4 neighbors
  - (2 old points, 2 adjacent face points)
- Move old points
  - (n-2)/n times itself
  - $1/n^2$ average of N adjacent edges
  - $1/n^2$ average of N adjacent faces
Making Creases

- Hard edge subdivision
  - Don’t displace points
  - Put edge points at midpoint

- Semi-hard edge subdivision
  - Use hard edge rules for 1st few iterations
  - Then use the regular rules
Exact Evaluation

• For regular points on Catmull-Clark – it's just a B-Spline!
• There are methods for extraordinary points (1998)

• For all types, “Masks” exist
  – Final answer depends on points in the neighborhood
  – Look them up in a book
The Joy of Subdivision
(at least Catmull-Clark / Loop)

Start with any mesh
  Any topology
Trim curves - easy
Stitching – easy
Creases – easy
Texturing – easy (actually any properties)
Normals / Tangents - easy