Lecture 30 – Parametric Surfaces and Subdivision

CS559 Lecture Notes
Not for Projection
November 2007 (borrow from previous years)

Surface Representations

- Very similar to curves
- Implicit f(x,y,z) = 0
- Parametric f(u,v) = 0
- Procedural f(?) -> points, polygons, …
- Subdivision
- Old days: parametric surfaces (NURBS)
- Now: Subdivision!

Surface Patches

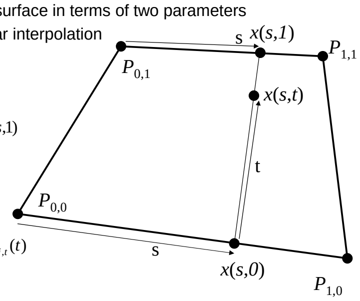
- A square (u,v) in (0->1, 0->1) that gets mapping into space
- Put squares together
  - Continuity Issues at edges
- Cut holes in patches
  - **Trim curves** defined in parameter space
- Stitch together at seams
  - Like sewing – cut pieces and sew them together
- Making things fit together requires dealing with the complicated math of the curve boundaries

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

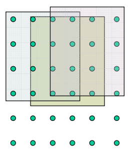
- Polynomial in u and v

$$\sum_{i<d}\sum_{j<d} a_{ij}u^i v^j$$

- 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

Tensor Product Cubics

- Each patch needs a 4x4 grid of control points
- Need to be very careful to make sure that there is continuity across edges
- B-Splines, Beziers, Cardinals, …



•Must be a regular grid

•Every point is in 16 patches

•Can't insert detail locally (need to add an entire row/column)

NURBS

- Each patch is a B-Spline (often cubic)
- Need Rational B-Splines to make spheres and conics
  - And projective invariance
- If you thought B-Spline curves were hard…
- Issues in trimming
- Issues in stitching (without cracking)
- Issues in adding detail
- Issues in tesselating it well
- We won't bother teaching you about these anymore

Last time

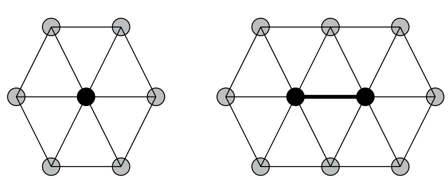
- Representing Smooth Surfaces
- Piecewise parametric surfaces (tensor products)
- B-Splines, NURBS
- General, mathematically elegant
- Problematic
- Subdivision
  - Basic ideas
  - Schemes for curves

Subdivision Concepts

- Start with initial, discrete representation
  - Control points, line segments (for curves), polygons (for surfaces)
- Subdivision rules to make finer resolution
  - Still get a discrete approximation to smooth thing
- **Limit Surface** (or Curve)
  - The "mathematical" result is what happens after infinite steps
- **Exact Evaluation** – tells about points on Limit Surface
- Stationary Points / Schemes – stay put (interpolate)
- Non-Stationary Points – move (approximate original)

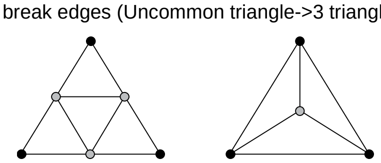
Regular Meshes

- 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)

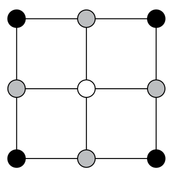


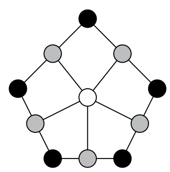
Midpoint (4) scheme
very common

Unconmmon 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)



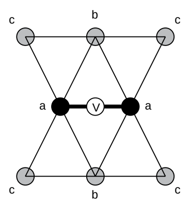


What does a Subdivision Scheme Need?

- Rules for ordinary points
- Rules for extra-ordinary points
- Rules for edges/corners
  - Treat them specially
  - Edges only depend on edges (so shared edges connect)
- Proof that the limit surface is continuous
- Exact evaluation methods
- Methods to introduce creases, provide texture coords, …

Butterfly Scheme

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



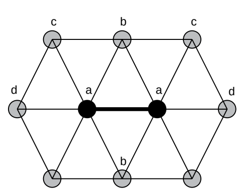
v = 1/2 a + 1/8 b - 1/16 c

What about extra-ordinary points?

- They do happen!
  - Edges, corners
  - Holes
  - Places where things are stitched together
- Tensor product surfaces can't handle them well either
- Easy Method: do "nothing" – leave midpoint at midpoint
- Problem gets smaller on each iteration
  - Only edges adjacent to extraordinary point
  - And these get cut in half each time
- In limit: "problem" is very localized
  - Surface is C(1) "almost everywhere" (except extra-ordinary points)

Modified Butterfly

- Introduce tension parameter, use 10 points
- New rules for extraordinary points



v = (1/2-w) **a** + (1/8+2w) **b** - (1/16-w) **c** + w **d**

tension parameter w
sum over all 10 neighbors

Modified Butterfly

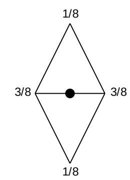
- Edge with 1 extra-ordinary point
  - Two extraordinary points? Do both as if 1, and average
  - Only happens on first pass
- For a K vertex – only use points around it (weight V=3/4)
  - S0 = point on edge we're dividing
  - K=3 S0=5/12, s1,s2=-1/12
  - K=4 S0=3/8, s1,s3 = 0, s2=-1/8
  - K>=5 (.25+cos(2pi j / K) + .5 * cos (4 pi j / K)) / K
    - J from 0->K-1
- Use 4 point curves around edges
  - -1/16, 9/16, 9/16, -1/16

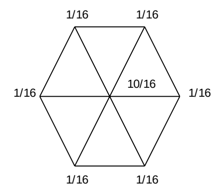
Why not Butterfly?

- Is C(1) and Interpolating
- Sensitive to noise in data (since it will interpolate)
- Not “Fair” (we get little wigglies)
- Not C(2)
- A lot like interpolating cubics

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 (5/8 – ((3+2cos(2pi / n))^2) / 64)
  - 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 ¾ 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 – its 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

Modeling with subdivision

- Any mesh can be subdivided
- Cut holes, create unusual topology, stitch pieces together
- No matter how complicated the mesh, it will lead to a smooth surface!

Why Subdivision

- B-Splines are Smooth
- B-Splines must be Tesselated
  - Sampling issues
  - How to decide triangle size
  - Need to worry about cracking
- B-Splines have uniform resolution
- Detail must be global

- Limit surfaces are smooth
- Subdivision gives meshes
  - Subdivide as needed
  - Always gives connected mesh
  - Get as many polys as you need
- Subdivision – put detail where you want it
- Detail is multi-resolution

Why Subdivision (2)

- B-Splines require regular grid
- Complex Topology is hard
  - No corners, holes, …
- Trimming is hard
- Stitching is hard
- Get a (u,v) parameterization
  - Not controllable
- Hard to make creases and sharp edges

- Subdivision of any mesh
- Any topology can be handled
  - Easy to make corners, holes, …
- Trimming is easy
- Stitching is easy
- (u,v) parameterization by subdivision of points
  - Controllable
- Easy to make creases and sharp edges