Catmull-Clark Subdivision

1. Initial mesh
2. For each Face, compute its centroid

Centroid = average of all vertices

Consider storing these centroids in a way that you can access a centroid given its corresponding face (e.g. a map)
3. Compute the smoothed **midpoint** of each edge in the mesh

\[ e = \frac{(v_1 + v_2 + f_1 + f_2)}{4} \]

If only one face is incident to the edge (as is the case for the borders of this image), then

\[ e = \frac{(v_1 + v_2 + f)}{3} \]

Note that for this particular mesh, the smoothed midpoints happen to just be the midpoints of the edges.
4. Smooth the original vertices

\[ v' = (n-2)v/n + \frac{\text{sum}(e)}{n^2} + \frac{\text{sum}(f)}{n^2} \]

- \( v \) is the vertex’s original position
- \( \text{sum}(e) \) is the sum of all adjacent midpoints
- \( \text{sum}(f) \) is the sum of the centroids of all faces incident to \( v \)

This is the barycenter of the input components

\( n \) is the number of adjacent midpoints
Catmull-Clark Subdivision

5. For each original face, split that face into N quadrangle faces

- N is the number of vertices that the face originally had
- This is not a trivial task
  - Consider writing a Quadrangulate function that takes a Face, a centroid, and the set of half-edges that points to the midpoints on that face.
Common mistakes

- Forgetting to update/instantiate the following pointers:
  - HalfEdge::next
  - Face::start_edge
  - Vertex::incoming_edge
  - HalfEdge::face
  - HalfEdge::sym

- Smoothing the mesh one face at a time rather than performing each step globally
  - This leads to lopsided smoothed meshes

- Splitting each half-edge of the mesh into midpoints rather than splitting each full edge
  - Also leads to lopsided meshes
Subdividing with sharp edges

- Before a mesh is subdivided, some of its components may be tagged as “sharp”
- A sharp vertex never moves when subdivided
- A vertex bounded by $N$ sharp edges follows these rules:
  - $N < 2$: Treated as a regular vertex; follows normal subdivision rules
  - $N = 2$: $v' = (0.75)v + (0.125)v_1 + (0.125)v_2$
    - $v_1$ and $v_2$ are the vertices at the ends of the incident sharp edges
  - $N > 2$: Treated as a sharp vertex
- A sharp face treats all of its vertices as sharp
Subdividing with sharp edges

- Fully smooth
- Sharp edges
- Sharp vertex

Image source: http://xrt.wikidot.com/blog:31
Subdividing with sharp edges

- We can also introduce semi-sharpness, which is a floating point number between 0 and 1.
- If a component has a sharpness of 1, it is treated as being fully sharp as in the previous slide.
- A sharpness of 0 means the component follows normal Catmull-Clark subdivision rules.
- A sharpness between 0 and 1 means the component’s final position is a linear interpolation of its fully sharp position and its fully smoothed position, where the interpolation’s $t$ value is the sharpness value.
Subdividing with sharp edges

Semi-sharp  Sharper  Even sharper

Image source: http://xrt.wikidot.com/blog:31
Surfaces of revolution

- Technique for modeling surfaces that are symmetrical across a particular axis
- Given some curve $C$ and an axis $V$, rotate $C$ about $V$ by some interval of degrees a certain number of times
  - The rotation amount and the number of rotations are defined by the user
  - Example: To create an open pentagonal prism, rotate a vertical line about the Y axis 5 times, each time by 72 degrees (360/5).
Modeling: Surfaces of revolution

- For a linear surface, create a quadrangle between each pair of control points. Combined, they form our surface.
- For an implicit surface, use something like a NURBS or Bézier curve as our base curve, then create rotated copies of its control points, connecting each copied control point to its source.
Modeling: Extrusions

- Given a “shell” curve $S$ and a “path” curve $P$, create copies of $S$ along $P$ so that the $S$ copies can be connected via quadrangles.
- Need to rotate $S$ as we traverse $P$ so that the extrusion loses as little volume as possible.
How do we determine the amount by which we should rotate $S$?

Given a segment $AB$ of $P$ and another segment $BC$, find the axis and angle of rotation that takes us from an orientation along $AB$ to an orientation along $BC$

- Axis = $\text{normalize}(AB \times BC)$
- Angle = $\cos^{-1}(\text{dot}(AB, BC) / (||AB|| \times ||BC||))$

We can also apply this function to mesh faces.
Half-Edge Face Extrusion

- **Input:** Half-edge face $F$ and an optional path curve $P$
- **Result:** $D*N + 1$ faces where $D$ is the number of edges on $F$ and $N$ is the number of control points of $P$
- In essence, extrude each edge of $F$ along $P$, forming a quadrangle at each step.
Extruding an edge

- Input: HE1
- Goal: Create a new face between HE1’s face and HE2’s face
Extruding an edge

- Step 1: Create two new vertices \(V_3\) and \(V_4\)
Extruding an edge

- Step 2: Adjust HE1 so that it points to V3, and adjust HE1’s prev so that it points to V4.
Extruding an edge

- Step 3: Create two new half-edges HE1B and HE2B
  - HE1.sym = HE1B
  - HE2.sym = HE2B
  - HE1B.sym = HE1
  - HE2B.sym = HE2
  - HE1B.vert = V4
  - HE2B.vert = V1
Extruding an edge

- Step 4: Create a new face $F$ and another two half-edges $HE3$ and $HE4$
  - Set the face pointers of $HE1B$, $HE2B$, $HE3$, and $HE4$ to $F$
    - $HE3$.vert = $V3$
    - $HE4$.vert = $V2$
    - $HE1B$.next = $HE4$
    - $HE4$.next = $HE2B$
    - $HE2B$.next = $HE3$
    - $HE3$.next = $HE1B$
Extruding an edge

- Step 5: Set syms of HE3 and HE4
  - If we are extruding multiple edges as part of a face extrusion, then we set the sym pointers of HE3 and HE4 to be half-edges created by other edge extrusions
  - If this is a single edge extrusion, then we create sym half-edges that point to a NULL face
Edge Loops

- An edge loop is a set of edges where the loop follows a middle edge in every four way junction. The loop ends as soon as it encounters another type of junction.
Selecting an Edge Loop

- Can only select an edge loop if vertices on the loop has exactly 4 incoming half edges (or else the loop is ambiguous - for example a cube).
- Edge loops only defined for quads (or sections of a mesh that are quad).
- How do you select an edge loop?
  - Given a starting half edge, HE0, the next half edge in the loop is HE0->next->sym->next.
- Inserting an edge loop
  - Find two edge loops, where the one you want to insert is in between. Split each face between those loops into two.
Beveling

Bevel Face:
Same as face extrusion, but with some additional parameters to determine its positioning.

Bevel Edge:
- Same as half-edge extrusion, with additional parameters to determine its positioning.
Beveling

- Given an edge and a $t$ between 0 and 1, find the faces on either side of the edge.
  - Find both edges on each face that are incident to the selected edge and compute a point on each edge based on $t$
  - Form a quad that spans the four created vertices, deleting (or shifting) our original edge
  - If the number of edges incident to our original vertices was more than 2, then make edges from the new vertices to the remaining extra edges.
- Similar process for beveling a face