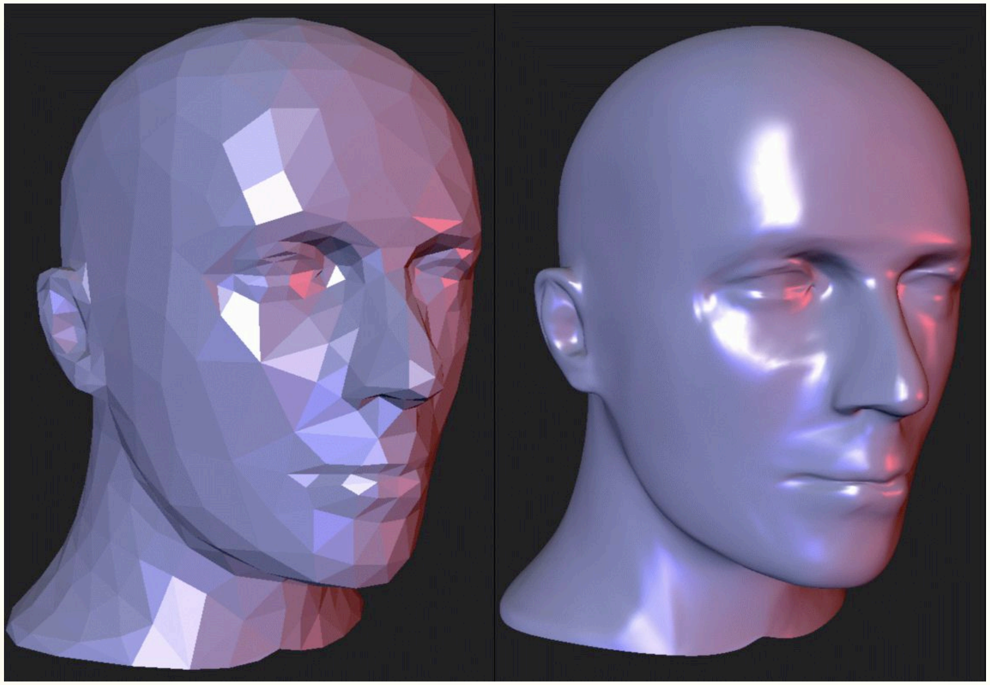
Subdivision surfaces

Reading

- Recommended:
  - Stollnitz, DeRose, and Salesin. *Wavelets for Computer Graphics: Theory and Applications*, 1996, section 10.2.

Building complex models

We can extend the idea of subdivision from curves to surfaces…

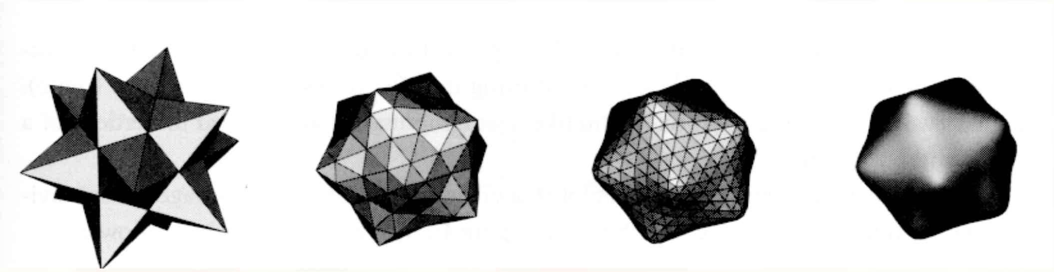


Subdivision surfaces

- Chaikin's use of subdivision for curves inspired similar techniques for subdivision surfaces.
- Iteratively refine a **control polyhedron** (or **control mesh**) to produce the limit surface

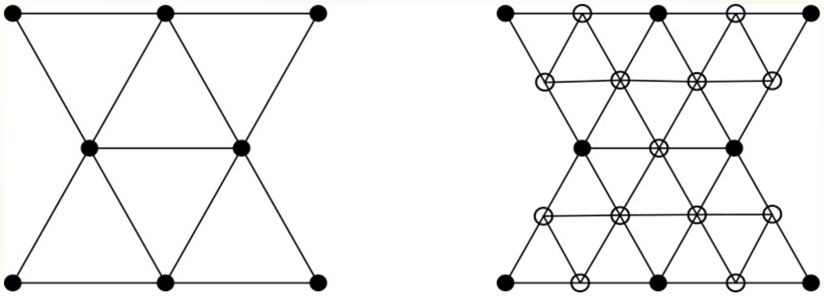
$$\sigma = \lim_{j \to \infty} M^j$$

using splitting and averaging steps.



Triangular subdivision

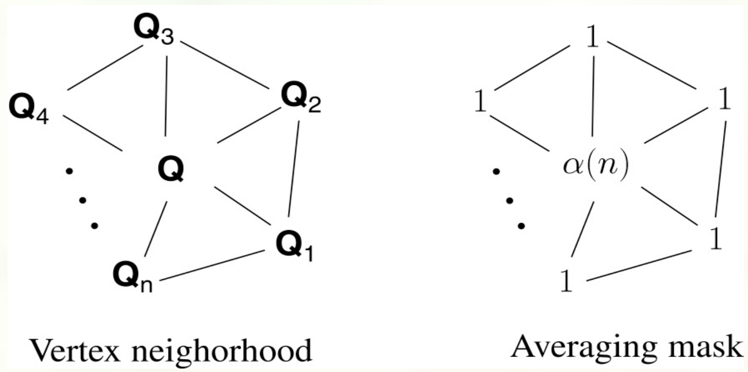
- There are a variety of ways to subdivide a poylgon mesh.
- A common choice for triangle meshes is 4:1 subdivision – each triangular face is split into four subfaces:



Original After splitting

Loop averaging step

- Once again we can use **masks** for the averaging step:



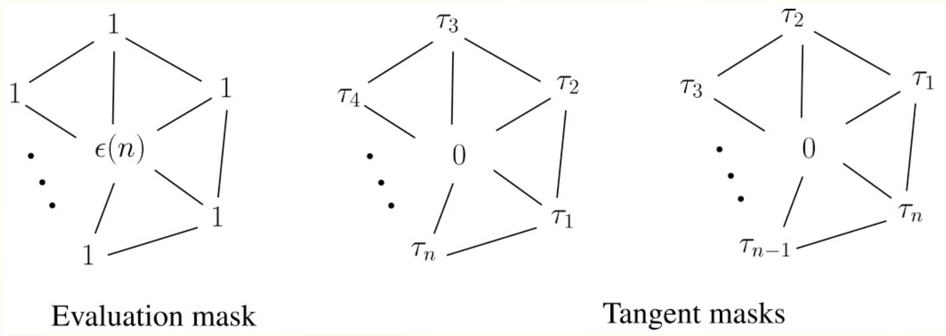
- where

$$\mathbf{Q} \leftarrow \frac{\alpha(n)\mathbf{Q} + \mathbf{Q}_1 + \cdots + \mathbf{Q}_n}{\alpha(n) + n} \qquad \alpha(n) = \frac{n(1-\beta(n))}{\beta(n)} \qquad \beta(n) = \frac{5}{4} - \frac{(3 + 2\cos(2\pi / n))^2}{32}$$

- These values, due to Charles Loop, are carefully chosen to ensure smoothness – namely, tangent plane or normal continuity.
- Note: tangent plane continuity is also known as G^1 continuity for surfaces.

Loop evaluation and tangent masks

- As with subdivision curves, we can split and average a number of times and then push the points to their limit positions.



$$\mathbf{Q}^{\infty} = \frac{\varepsilon(n)\mathbf{Q} + \mathbf{Q}_1 + \cdots + \mathbf{Q}_n}{\varepsilon(n) + n}$$

$$\mathbf{T}_1^{\infty} = \tau_1(n)\mathbf{Q}_1 + \tau_2(n)\mathbf{Q}_2 + \cdots + \tau_n(n)\mathbf{Q}_n$$

$$\mathbf{T}_2^{\infty} = \tau_n(n)\mathbf{Q}_1 + \tau_1(n)\mathbf{Q}_2 + \cdots + \tau_{n-1}(n)\mathbf{Q}_n$$

- where $\varepsilon(n) = \frac{3n}{\beta(n)} \quad \tau_i(n) = \cos(2\pi i/n)$

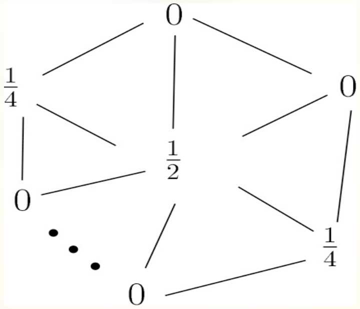
- How do we compute the normal?

Recipe for subdivision surfaces

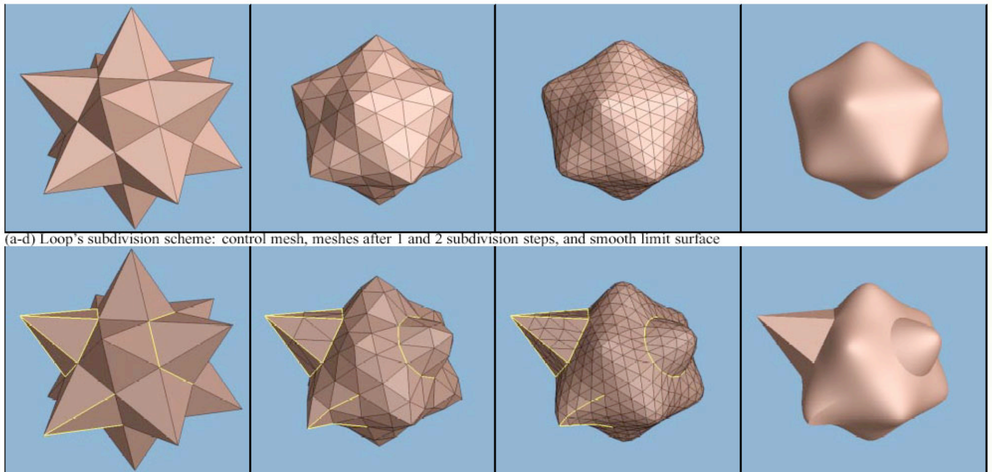
- As with subdivision curves, we can now describe a recipe for creating and rendering subdivision surfaces:
  - Subdivide (split+average) the control polyhedron a few times. Use the averaging mask.
  - Compute two tangent vectors using the tangent masks.
  - Compute the normal from the tangent vectors.
  - Push the resulting points to the limit positions. Use the evaluation mask.
  - Render!

Adding creases without trim curves

- In some cases, we want a particular feature such as a crease to be preserved. With NURBS surfaces, this required the use of trim curves.
- For subdivision surfaces, we can just modify the subdivision mask:



- This gives rise to G^0 continuous surfaces (i.e., having positional but not tangent plane continuity)



(a-d) Loop's subdivision scheme: control mesh, meshes after 1 and 2 subdivision steps, and smooth limit surface

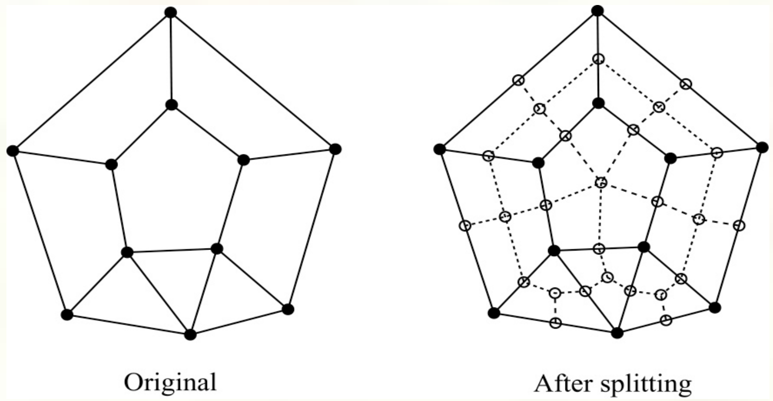
(e-h) Our piecewise smooth subdivision scheme: tagged control mesh, meshes after 1 and 2 subdivision steps, and piecewise smooth limit surface

Creases without trim curves, cont.

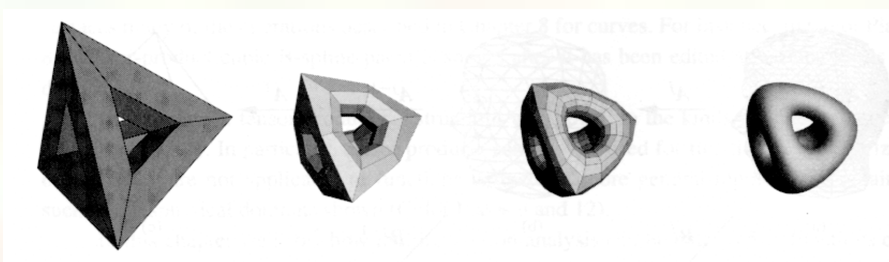
- Here's an example using Catmull-Clark surfaces (based on subdividing quadrilateral meshes):

Face schemes

- 4:1 subdivision of triangles is sometimes called a **face scheme** for subdivision, as each face begets more faces.
- An alternative face scheme starts with arbitrary polygon meshes and inserts vertices along edges and at face centroids:



- **Catmull-Clark subdivision**:



- Note: after the first subdivision, all polygons are quadilaterals in this scheme.

Subdivision = tensor-product patches!

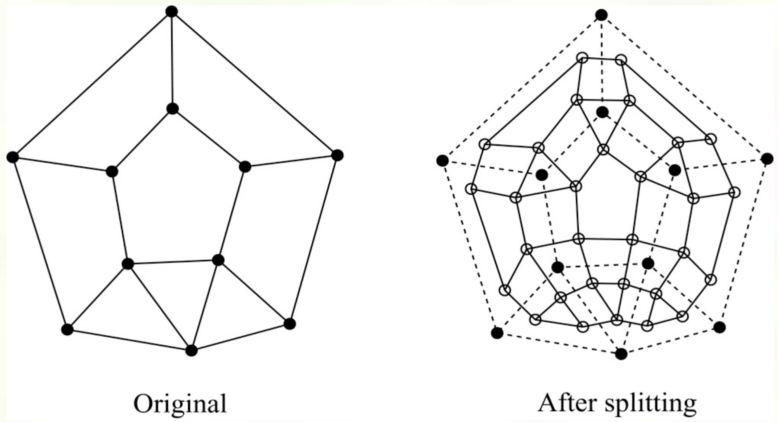
- For a regular quadrilateral mesh, Catmull-Clark subdivision produces the same surface as tensor-product cubic B-splines!
- But – it handles irregular meshes as well.

There are similar correspondences between other subdivision schemes and other tensor-product patch schemes.

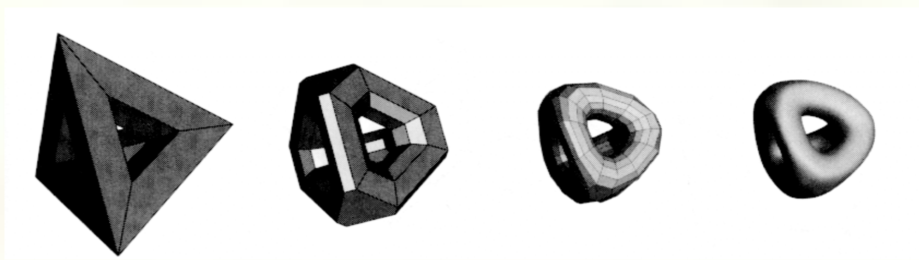
These correspondences can be proven (but we won't do it…)

Vertex schemes

- In a **vertex scheme**, each vertex begets more vertices. In particular, a vertex surrounded by n faces is split into n sub-vertices, one for each face:



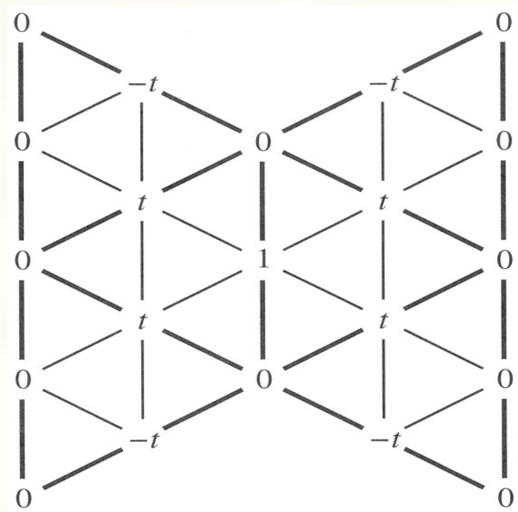
- **Doo-Sabin subdivision**:



- The number edges (faces) incident to a vertex is called its **valence**. Edges with only once incident face are on the **boundary**. After splitting in this subdivision scheme, all non-boundary vertices are of valence 4.

Interpolating subdivision surfaces

- Interpolating schemes are defined by
  - splitting
  - averaging only new vertices
- The following averaging mask is used in **butterfly subdivision**:



- Setting t=0 gives the original polyhedron, and increasing small values of t makes the surface smoother, until t=1/8 when the surface is provably G^1.

  There are several variants of Butterfly subdivision.

Next class: Projections & Z-Buffers

- Topics:
  - How do projections from 3D world to 2D image plane work?
  - How does the Z-buffer visibility algorithm (used in today's graphics hardware) work?

  Read:
  - Watt, Section 5.2.2 – 5.2.4, 6.3, 6.6 (esp. intro and subsections 1, 4, and 8–10)

  Optional:
  - Foley, et al, Chapter 5.6 and Chapter 6
  - David F. Rogers and J. Alan Adams, Mathematical Elements for Computer Graphics, 2nd Ed., McGraw-Hill, New York, 1990, Chapter 2.
  - I. E. Sutherland, R. F. Sproull, and R. A. Schumacker, A characterization of ten hidden surface algorithms, ACM Computing Surveys 6(1): 1-55, March 1974.