16. Subdivision curves and surfaces

Reading

Recommended:

- Stollnitz, DeRose, and Salesin. *Wavelets for Computer Graphics: Theory and Applications,* 1996, section 6.1-6.3, 10.2, A.5.

Subdivision curves

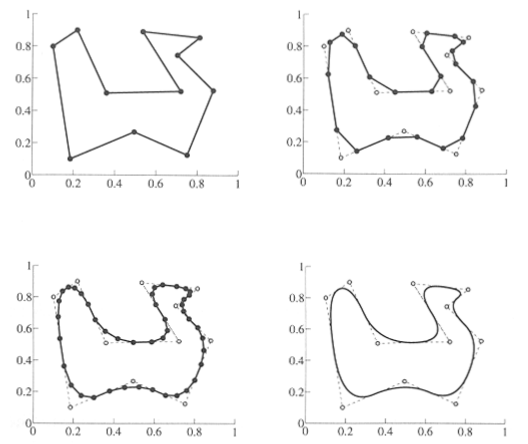
Idea:

- repeatedly refine the control polygon

$$P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow \cdots$$

- curve is the limit of an infinite process

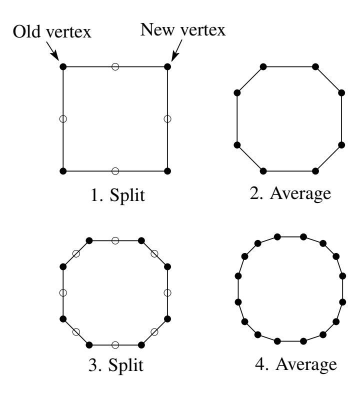
$$C = \lim_{i \to \infty} P_i$$



Chaikin's algorithm (1974)

Chakin introduced the following "corner-cutting" scheme in 1974:

- Start with a piecewise linear curve
- Insert new vertices at the midpoints (the **splitting step**)
- Average each vertex with the "next" neighbor (the **averaging step**)
- Go to the splitting step



Averaging masks

The limit curve is a quadratic B-spline!

Instead of averaging with the nearest neighbor, we can generalize by applying an **averaging mask** during the averaging step:

$$r = (\ldots, r_{-1}, r_0, r_1, \ldots)$$

In the case of Chaikin's algorithm:

$$r =$$

Lane-Riesenfeld algorithm (1980)

Use averaging masks from Pascal's triangle:

$$r = \frac{1}{2^n}\left(\binom{n}{0}, \binom{n}{1}, \ldots, \binom{n}{n}\right)$$

Gives B-splines of degree $n+1$.

n=0:

n=1:

n=2:

Subdivide ad nauseum?

After each split-average step, we are closer to the **limit curve**.

How many steps until we reach the final (limit) position?

Can we push a vertex to its limit position without infinite subdivision? Yes!

Recipe for subdivision curves

After subdividing and averaging a few times, we can push each vertex to its limit position by applying an **evaluation mask**.

Each subdivision scheme has its own evaluation mask, mathematically determined by analyzing the subdivision and averaging rules.

For Lane-Riesenfeld cubic B-spline subdivision, we get:

$$\frac{1}{6}\begin{pmatrix}1 & 4 & 1\end{pmatrix}$$

Now we can cook up a simple procedure for creating subdivision curves:

- Subdivide (split+average) the control polygon a few times. Use the averaging mask.
- Push the resulting points to the limit positions. Use the evaluation mask.

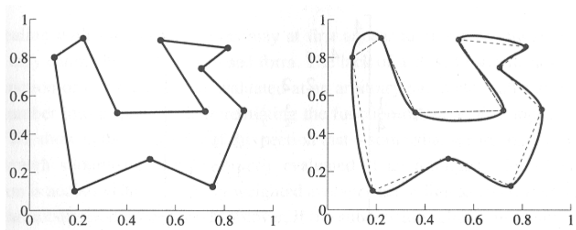
DLG interpolating scheme (1987)

Slight modification to algorithm:

- splitting step introduces midpoints
- averaging step *only changes midpoints*

For DLG (Dyn-Levin-Gregory), use:

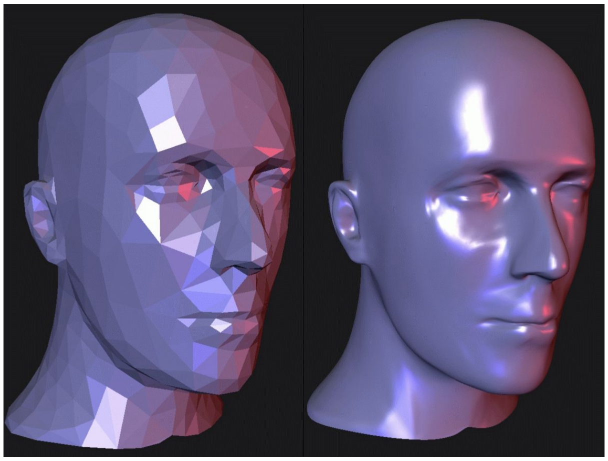
$$r = \frac{1}{16}(-2, 5, 10, 5, -2)$$



Since we are only changing the midpoints, the points after the averaging step do not move.

Building complex models

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



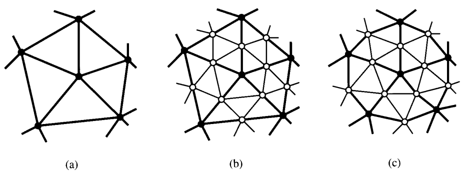
Subdivision surfaces

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.

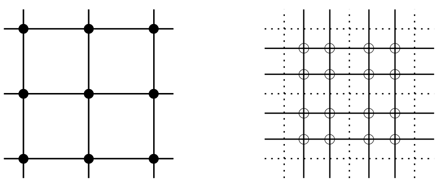


There are two types of splitting steps:

- **vertex schemes**
- **face schemes**

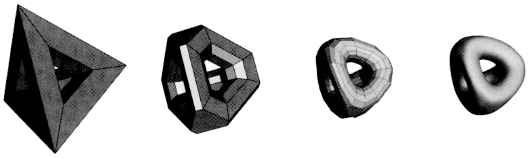
Vertex schemes

A vertex surrounded by *n* faces is split into *n* subvertices, one for each face:



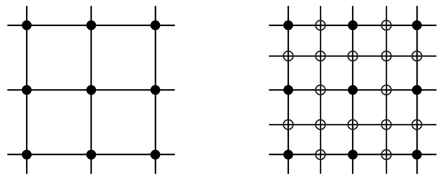
Original | After splitting

Doo-Sabin subdivision:



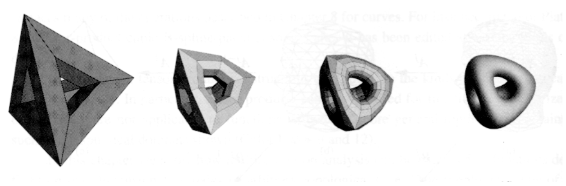
Face schemes

Each quadrilateral face is split into four subfaces:



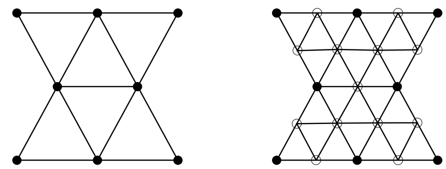
Original   After splitting

Catmull-Clark subdivision:



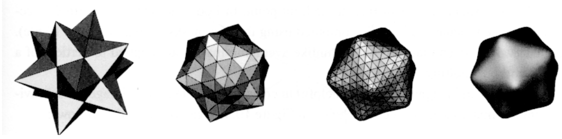
Face schemes, cont.

Each triangular face is split into four subfaces:



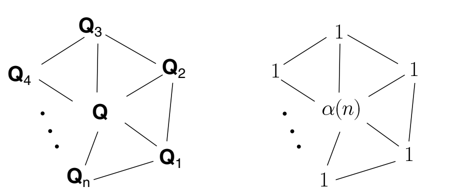
Original   After splitting

Loop subdivision:



Averaging step

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



Vertex neighorhood   Averaging mask

$$\mathbf{Q} \leftarrow \frac{\alpha(n)\mathbf{Q} + \mathbf{Q}_1 + \cdots + \mathbf{Q}_n}{\alpha(n) + n}$$

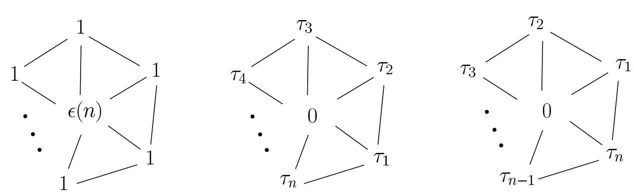
where

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

These values are carefully chosen to ensure smoothness – namely, G^1 (tangent) continuity.

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.



Evaluation mask   Tangent masks

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

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

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

where

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

How do we compute the normal?

Recipe for subdivision surfaces

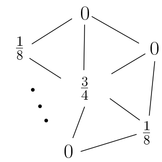
As with subdivision curves, we can now describe a recipe for creating and rendering subdivision surfaces:

- 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

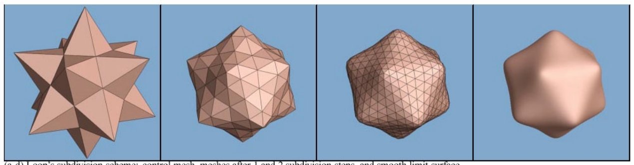
In some cases, we want a particular feature such as a crease to be preserved. With NURBS surfaces, this required the use of trim curves.

For subdivision surfaces, we can just modify the subdivision mask:

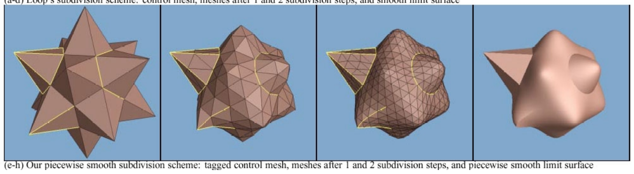


Loop crease/boundary edge

This gives rise to G^0 continuous surfaces.



(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 of the kind found in Geri's Game:

Interpolating subdivision surfaces

Interpolating schemes are defined by

- splitting
- averaging only new vertices

Summary

What to take home:

- The meanings of all the **boldfaced** terms.
- How to perform the splitting and averaging steps on subdivision curves.
- The various kinds of mesh splitting steps for subdivision surfaces, especially Loop.
- How to construct and render subdivision surfaces from their averaging masks, evaluation masks, and tangent masks.