

Subdivision curves and surfaces

**Brian Curless
CSE 557
Autumn 2017**

Reading

Required:

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

Note: there is an error in Stollnitz, et al., section A.5.
Equation A.3 should read:

$$\mathbf{MV} = \mathbf{V}\Lambda$$

This is already fixed in the handout.

Subdivision curves

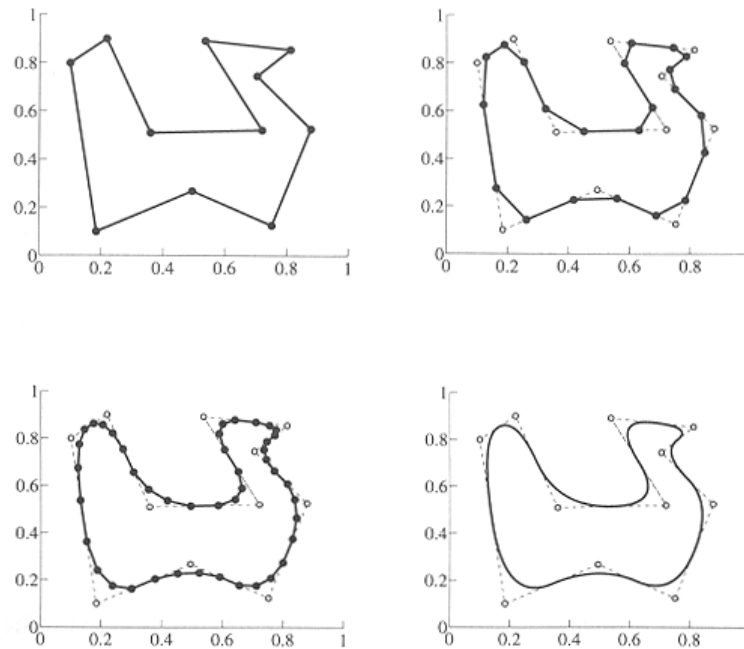
Idea:

- ◆ repeatedly refine the control polygon

$$P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow \dots$$

- ◆ curve is the limit of an infinite process

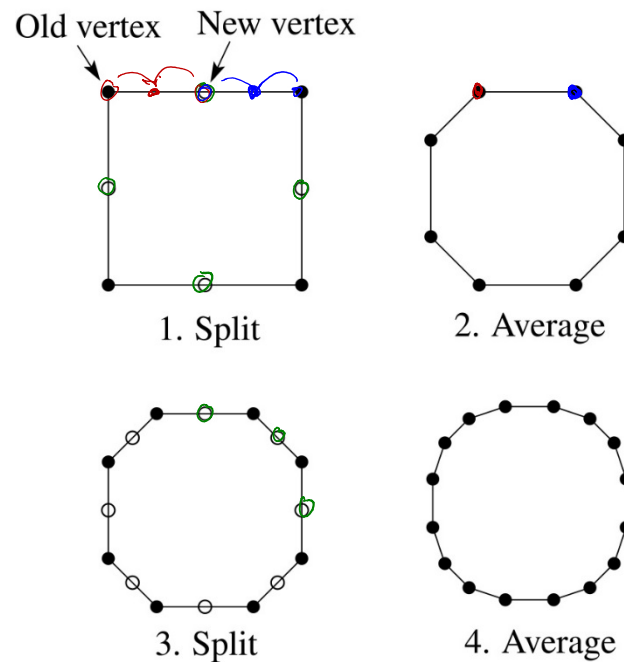
$$Q = \lim_{j \rightarrow \infty} P_j$$



Chaikin's algorithm

Chaikin 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" (clockwise) 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:

$$a = (\dots, a_{-1}, a_0, a_1, \dots)$$

In the case of Chaikin's algorithm:

$$a = \left[0 \quad \frac{1}{2} \quad \frac{1}{2} \right]$$

Different averaging masks lead to different curves.

For example,

$$a = \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{pmatrix}$$

Leads to **cubic** B-spline curves.

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 in one step?

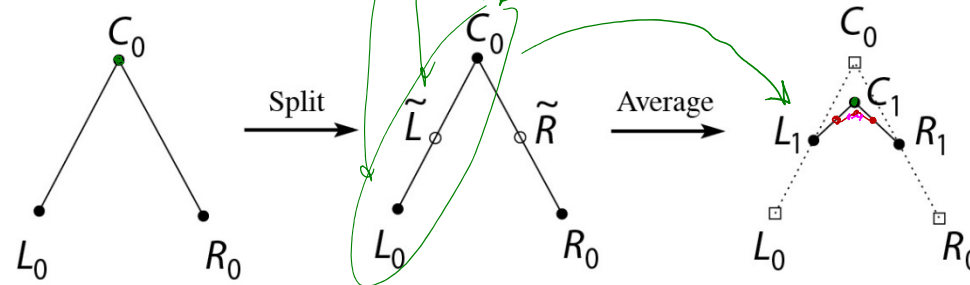


Local subdivision matrix

Consider the cubic B-spline subdivision mask:

$$\begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}$$

Now consider what happens during splitting and averaging in a small neighborhood:



We can write equations that relate points at one subdivision level to points at the previous:

$$\tilde{L} = \frac{1}{2}L_0 + \frac{1}{2}C_0 \quad \tilde{R} = \frac{1}{2}C_0 + \frac{1}{2}R_0$$

$$L_1 = \frac{1}{4}L_0 + \frac{1}{2}\tilde{L} + \frac{1}{4}C_0$$

$$= \frac{1}{4}L_0 + \frac{1}{2}\left(\frac{1}{2}L_0 + \frac{1}{2}C_0\right) + \frac{1}{4}C_0$$

$$= \frac{1}{2}L_0 + \frac{1}{2}C_0$$

$$R_1 = \frac{1}{2}C_0 + \frac{1}{2}R_0$$

$$C_1 = \frac{1}{4}\tilde{L} + \frac{1}{2}C_0 + \frac{1}{4}\tilde{R}$$

$$= \frac{1}{4}\left(\frac{1}{2}L_0 + \frac{1}{2}C_0\right) + \frac{1}{2}C_0 + \frac{1}{4}\left(\frac{1}{2}C_0 + \frac{1}{2}R_0\right)$$

$$= \frac{1}{8}L_0 + \frac{3}{4}C_0 + \frac{1}{8}R_0$$

Local subdivision matrix

We can write this as a recurrence relation in matrix form:

$$\begin{bmatrix} L_j \\ C_j \\ R_j \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/8 & 3/4 & 1/8 \\ 0 & 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} L_{j-1} \\ C_{j-1} \\ R_{j-1} \end{bmatrix}$$

where the L, R, C s are (for convenience) *row* vectors.

In 2D, we can write out all the elements as follows:

$$\begin{bmatrix} L_j^x & L_j^y & 1 \\ C_j^x & C_j^y & 1 \\ R_j^x & R_j^y & 1 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/8 & 3/4 & 1/8 \\ 0 & 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} L_{j-1}^x & L_{j-1}^y & 1 \\ C_{j-1}^x & C_{j-1}^y & 1 \\ R_{j-1}^x & R_{j-1}^y & 1 \end{bmatrix}$$

A_j M A_{j-1}

We can re-write this as:

$$\mathbf{A}_j = \mathbf{M} \mathbf{A}_{j-1}$$

and \mathbf{M} is the **local subdivision matrix**.

Local subdivision matrix, cont'd

Starting from the initial control polygon, we can track the original vertex and its original neighborhood through subdivision:

$$\underline{\mathbf{A}}_j = \mathbf{M}\mathbf{A}_{j-1} = \mathbf{M}(\mathbf{M}\mathbf{A}_{j-2}) = \mathbf{M}(\mathbf{M}(\mathbf{M}\mathbf{A}_{j-3})) = \mathbf{M}^j \mathbf{A}_0$$

The limit position of the neighborhood is then:

$$\mathbf{A}_\infty = \lim_{j \rightarrow \infty} \mathbf{M}^j \mathbf{A}_0$$

OK, so how do we apply a matrix an infinite number of times??

Eigenvectors and eigenvalues

We now need to look at the eigenvectors and eigenvalues of \mathbf{M} . Let \mathbf{v} be a vector such that:

$$\mathbf{M}\mathbf{v} = \lambda\mathbf{v}$$

We say that \mathbf{v} is an eigenvector of \mathbf{M} with eigenvalue λ .

A 3x3 matrix can have 3 eigenvalues and eigenvectors:

$$\mathbf{M}\mathbf{v}_1 = \lambda_1\mathbf{v}_1$$

$$\mathbf{M}\mathbf{v}_2 = \lambda_2\mathbf{v}_2$$

$$\mathbf{M}\mathbf{v}_3 = \lambda_3\mathbf{v}_3$$

In matrix form:

$$\mathbf{M} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix} = \begin{bmatrix} \lambda_1\mathbf{v}_1 & \lambda_2\mathbf{v}_2 & \lambda_3\mathbf{v}_3 \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

$$\mathbf{M}\mathbf{V} = \mathbf{V}\Lambda$$

To infinity, but not beyond...

Now let's apply \mathbf{M} to original neighborhood \mathbf{A}_0 :

$$\mathbf{A}_1 = \mathbf{M}\mathbf{A}_0 = \underbrace{\mathbf{M}\mathbf{V}\mathbf{V}^{-1}} \mathbf{A}_0 = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}\mathbf{A}_0$$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

Now let's advance another subdivision:

$$\mathbf{A}_2 = \mathbf{M}^2\mathbf{A}_0 = \mathbf{M}\mathbf{A}_1 = \underbrace{\mathbf{M}\mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}} \mathbf{A}_0 = \mathbf{V}\mathbf{\Lambda}\mathbf{\Lambda}\mathbf{V}^{-1}\mathbf{A}_0 = \mathbf{V}\mathbf{\Lambda}^2\mathbf{V}^{-1}\mathbf{A}_0$$

Do it j times:

$$\mathbf{A}_j = \mathbf{M}^j\mathbf{A}_0 = \mathbf{V}\mathbf{\Lambda}^j\mathbf{V}^{-1}\mathbf{A}_0$$

$$\mathbf{\Lambda}^j = \begin{bmatrix} \lambda_1^j & 0 & 0 \\ 0 & \lambda_2^j & 0 \\ 0 & 0 & \lambda_3^j \end{bmatrix}$$

What if we do this an infinite number of times?

$$\mathbf{A}_\infty = \mathbf{M}^\infty\mathbf{A}_0 = \mathbf{V}\mathbf{\Lambda}^\infty\mathbf{V}^{-1}\mathbf{A}_0 = \mathbf{V} \begin{bmatrix} \lambda_1^\infty & 0 & 0 \\ 0 & \lambda_2^\infty & 0 \\ 0 & 0 & \lambda_3^\infty \end{bmatrix} \mathbf{V}^{-1}\mathbf{A}_0$$

Let's assume the eigenvalues are non-negative and sorted so that:

$$\lambda_1 > \lambda_2 > \lambda_3 \geq \dots \geq \lambda_n \geq 0$$

If $\lambda_1 > 1$, then: *blows up*

If $\lambda_1 < 1$, then: *collapse*

If $\lambda_1 = 1$, then: *stable*

Evaluation masks

For cubic B-splines, the local subdivision matrix \mathbf{M} is:

$$\mathbf{M} = \begin{pmatrix} 1/2 & 1/2 & 0 \\ 1/8 & 3/4 & 1/8 \\ 0 & 1/2 & 1/2 \end{pmatrix} \quad \leftarrow$$

It's eigenvalues and eigenvectors are:

$$\begin{array}{ccc} \lambda_1 = 1 & \lambda_2 = \frac{1}{2} & \lambda_3 = \frac{1}{4} \\ \mathbf{v}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} & \mathbf{v}_2 = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} & \mathbf{v}_3 = \begin{bmatrix} 2 \\ -1 \\ 2 \end{bmatrix} \end{array}$$

$\lambda_1 = 1 > \lambda_2 > \lambda_3$, so we're OK!

We can write out Λ and \mathbf{V} :

$$\Lambda = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \quad \mathbf{V} = \begin{bmatrix} 1 & -1 & 2 \\ 1 & 0 & -1 \\ 1 & 1 & 2 \end{bmatrix}$$

We will also need \mathbf{V}^{-1} , which turns out to be:

$$\mathbf{V}^{-1} = \begin{bmatrix} 1/6 & 2/3 & 1/6 \\ -1/2 & 0 & 1/2 \\ 1/6 & -1/3 & 1/6 \end{bmatrix}$$

Evaluation masks (cont'd)

$$\lambda_1 > \lambda_2 > \lambda_3 \dots$$

So, we have:

$$\Lambda = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \quad \mathbf{V} = \begin{bmatrix} 1 & -1 & 2 \\ 1 & 0 & -1 \\ 1 & 1 & 2 \end{bmatrix} \quad \mathbf{V}^{-1} = \begin{bmatrix} 1/6 & 2/3 & 1/6 \\ -1/2 & 0 & 1/2 \\ 1/6 & -1/3 & 1/6 \end{bmatrix}$$

We can now compute the limit position of the neighborhood \mathbf{A}_0 :

$$\mathbf{A}_\infty = \mathbf{M}^\infty \mathbf{A}_0 = \mathbf{V} \Lambda^\infty \mathbf{V}^{-1} \mathbf{A}_0$$

$$= \begin{bmatrix} 1 & -1 & 2 \\ 1 & 0 & -1 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/6 & 2/3 & 1/6 \\ -1/2 & 0 & 1/2 \\ 1/6 & -1/3 & 1/6 \end{bmatrix} \mathbf{A}_0$$

$$= \begin{bmatrix} 1 & -1 & 2 \\ 1 & 0 & -1 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1/6 & 2/3 & 1/6 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{A}_0$$

$$= \begin{bmatrix} 1/6 & 2/3 & 1/6 \\ 1/6 & 2/3 & 1/6 \\ 1/6 & 2/3 & 1/6 \end{bmatrix} \begin{bmatrix} L_0 \\ C_0 \\ R_0 \end{bmatrix}$$

$$C_{\infty} = \begin{bmatrix} 1/6 & 2/3 & 1/6 \end{bmatrix} \begin{bmatrix} L_0 \\ C_0 \\ R_0 \end{bmatrix}$$

Recipe for subdivision curves

The row vector \mathbf{u}_1^T that pushes the original vertex to the limit position is called the **evaluation mask**:

$$\mathbf{u}_1^T = \begin{bmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{bmatrix}$$

Note that we do **not** need start with the 0th level control points and push them to the limit.

If we subdivide and average the control polygon j times, we can push the vertices of the refined polygon to the limit as well:

$$\mathbf{A}_\infty = \mathbf{M}^\infty \mathbf{A}_j = \mathbf{u}_1^T \mathbf{A}_j$$

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.

$$U \equiv \underline{V^{-1}}$$

$$MV = V\Lambda$$

$$V^{-1}MV = V^{-1}V\Lambda$$

$$V^{-1}MV = \Lambda$$

$$V^{-1}MV V^{-1} = \Lambda V^{-1}$$

$$V^{-1}M = \Lambda V^{-1}$$

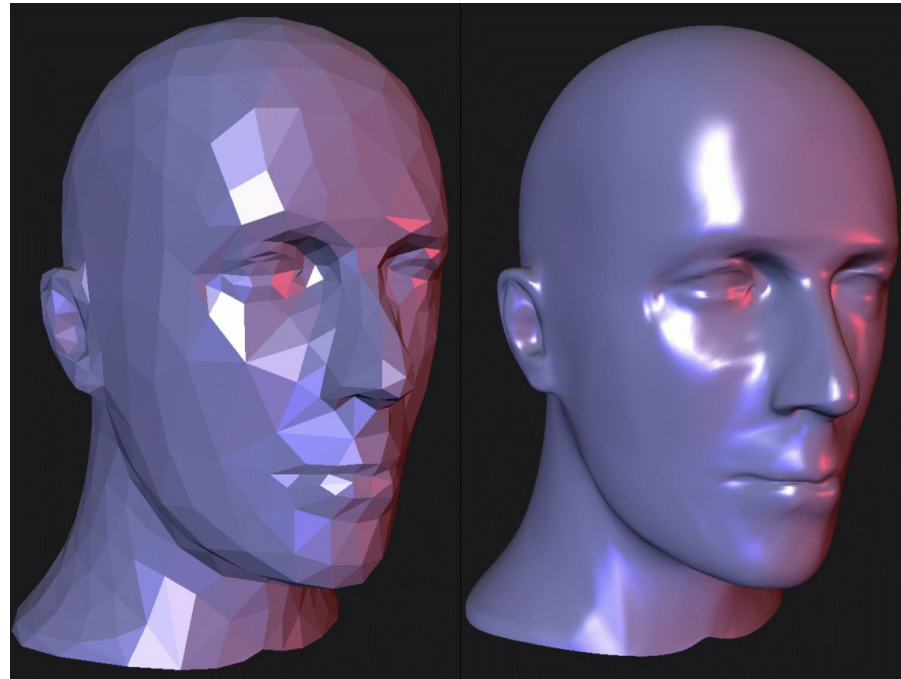
$$U M = \Lambda U$$

$$\mathbf{u}_1^T M = \Lambda \mathbf{u}_1^T$$

$$U = \begin{bmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \mathbf{u}_3^T \end{bmatrix}$$

Building complex models

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



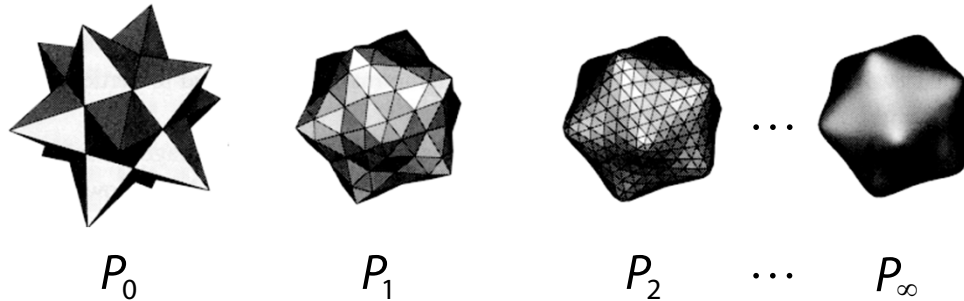
Subdivision surfaces

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

$$S = \lim_{j \rightarrow \infty} P_j$$

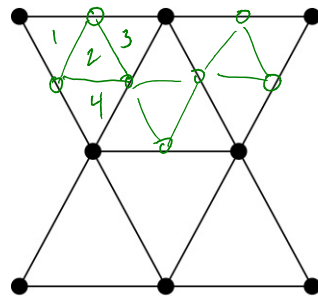
using splitting and averaging steps.



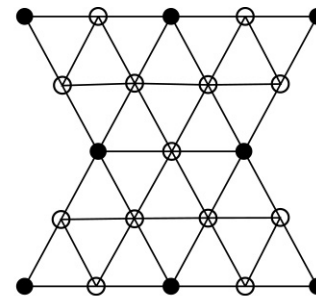
Triangular subdivision

There are a variety of ways to subdivide a polygon mesh.

A common choice for triangle meshes is 4:1 subdivision – each triangular face is split into four smaller triangles:



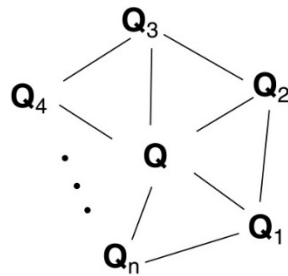
Original



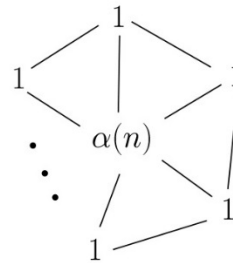
After splitting

Loop averaging step

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



Vertex neighborhood



Averaging mask
(before affine normalization)

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

$$\alpha(n) + \sum_{i=1}^n 1 = \alpha(n) + n$$

where

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

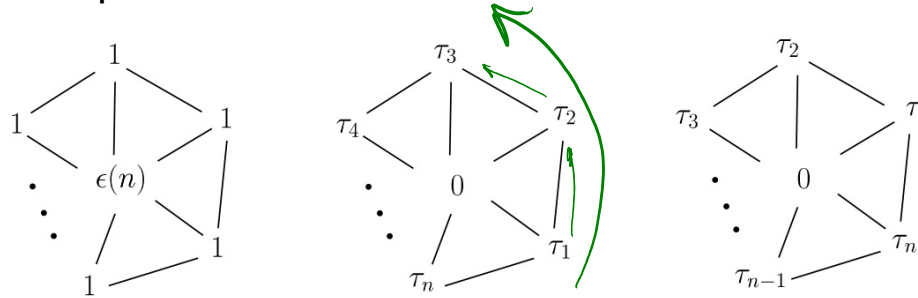
These values, due to Charles Loop, are carefully chosen to ensure smoothness – namely, tangent plane or normal continuity.

Note: tangent plane continuity is also known as G^1 continuity for surfaces.

Loop evaluation and tangent masks

$$\lambda_1 > \lambda_2 = \lambda_3 > \lambda_4$$

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



Evaluation mask
(before affine normalization)

Tangent masks

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

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

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

where

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

How do we compute the normal?

$$\mathbf{N} \sim \mathbf{T}_1^\infty \times \mathbf{T}_2^\infty$$

Recipe for subdivision surfaces

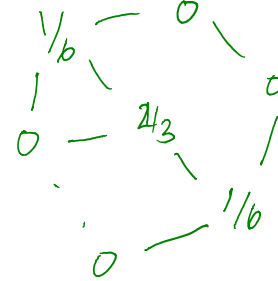
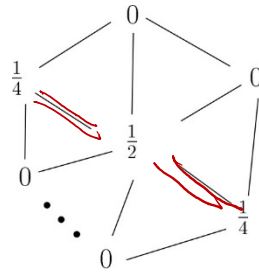
As with subdivision curves, we can now describe a recipe for creating and rendering subdivision surfaces:

- ◆ 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!

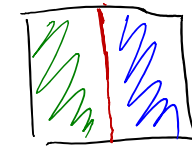
Adding creases without trim curves

For NURBS surfaces, adding sharp features like creases required the use of trim curves.

For subdivision surfaces, we can just modify the subdivision masks. E.g., we can mark some edges and vertices as "creases" and modify the subdivision mask for them (and their children):



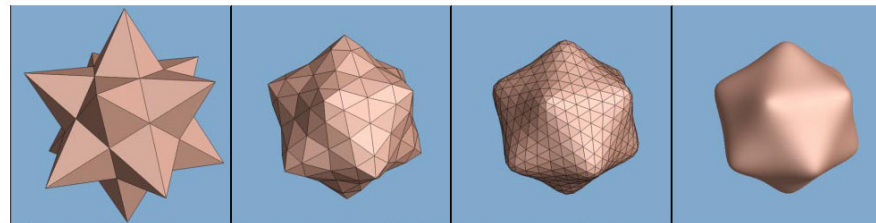
$$\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$



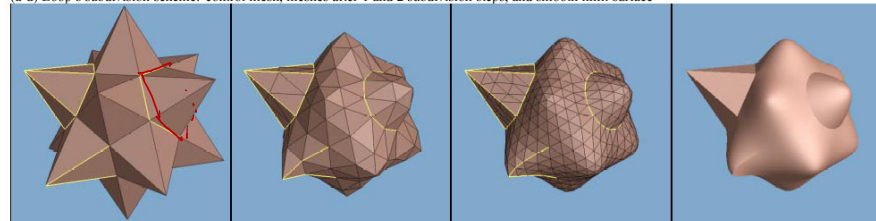
$$\begin{bmatrix} 1/a & 1/a & 1/a \\ 1/a & 1/a & 1/a \\ 1/a & 1/a & 1/a \end{bmatrix}$$

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

[Hoppe, SIGGRAPH 1994]



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

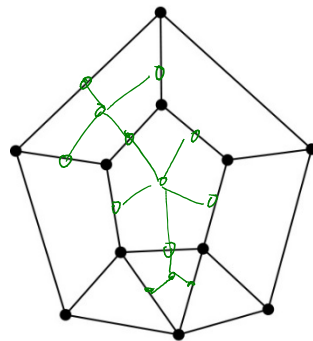


(e-h) Our piecewise smooth subdivision scheme: tagged control mesh, meshes after 1 and 2 subdivision steps, and piecewise smooth limit surface

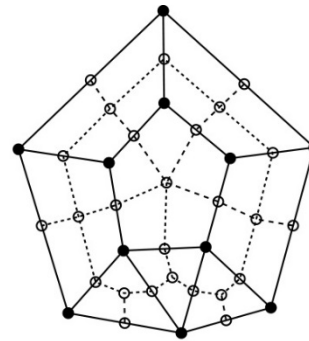
Catmull-Clark subdivision

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:

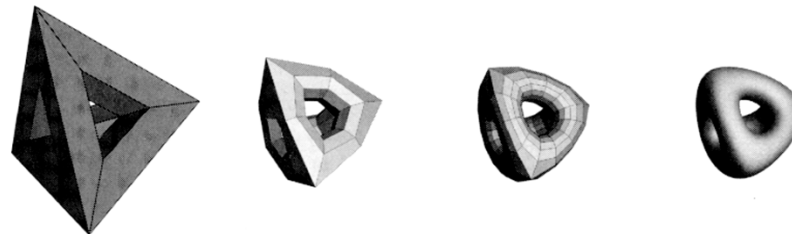


Original



After splitting

Catmull-Clark subdivision:



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

Creases without trim curves, cont.

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



This particular example uses the hybrid technique of DeRose, et al., which applies sharp subdivision rules at some creases for a finite number of steps, and then switches to smooth subdivision, giving more gentle creases. This technique was used in Geri's Game.