14. Surfaces

Reading

Recommended:

- Hearn and Baker, Section 10.8, 10.9, 10.14.
- Bartels, Beatty, and Barsky, *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*, 1987.
- Stollnitz, DeRose, and Salesin, *Wavelets for Computer Graphics: Theory and Applications*, 1996. Section 10.2.

Outline

1. Surfaces of revolution
2. General sweep surfaces
3. Tensor product surfaces
4. Subdivision surfaces

Surfaces of revolution

Idea: Rotate a 2D "profile" curve around an axis.

Q: What kinds of shapes can you model this way?

Constructing a surface of revolution

Given: A curve $C(u) : u \in [0, 1]$ in the yz-plane:

$$C(u) = \begin{bmatrix} 0 \\ c_y(u) \\ c_z(u) \\ 1 \end{bmatrix}$$

Let: $R_z(\theta)$ be a rotation matrix about the z-axis.

Find: A surface $S(u, v) : u, v \in [0, 1]$, which is $C(u)$ rotated about the z-axis.

Solution:

General sweep surfaces

The "surface of revolution" is a special case of a "sweep surface"....

Idea: Trace out surface $S(u, v)$ by moving a "profile" curve $C(u)$ along a "trajectory" $T(v)$.

More specifically:

- Suppose that $C(u)$ lies in an $(\hat{x}_c, \hat{y}_c)$ coordinate system with origin $\mathcal{O}_c$.
- For every point along $T(v)$, lay in $C(u)$ so that $T(v)$ coincides with $\mathcal{O}_c$.

Orientation

The big issue:

- How to orient $C(u)$ as it moves along $T(v)$.

Lots of options. Here are two:

1. Fixed (or "static"): Just translate $\mathcal{O}_c$ along $T(v)$:

2. Moving: Use the "Frenet frame" of $T(v)$.
   - Allows smoothly varying orientation
   - Permits surfaces of revolution, for example

Frenet frames

Motivation: Given a curve $T(v)$, we want to attach a smoothly-varying coordinate system.

Idea: To get a 3D coordinate system, we need 3 independent direction vectors.

- Tangent $\hat{t}(v) = \mathit{normalize}(T'(v))$
- Bi-normal $\hat{b}(v) = \mathit{normalize}(T'(v) \times T''(v))$
- Normal $\widehat{n}(v) = \hat{b}(v) \times \hat{t}(v)$

As we move along $T(v)$, the Frenet frame $(\hat{t}, \hat{b}, \widehat{n})$ varies smoothly.

Frenet swept surfaces

Idea: Orient the profile curve $C(u)$ using the Frenet frame of the trajectory $T(v)$:

- Put $C(u)$ in the "normal" plane $\widehat{n}\widehat{b}$.
- Place $\mathcal{O}_c$ on $T(v)$.
- Align $\widehat{x}_c$ for $C(u)$ with $-\widehat{n}$.
- Align $\widehat{y}_c$ for $C(u)$ with $\widehat{b}$.

Cool: If $T(v)$ is a circle, you get a surface of revolution exactly!

Problem: What do you do when the curvature changes sign or goes to 0?

Variations

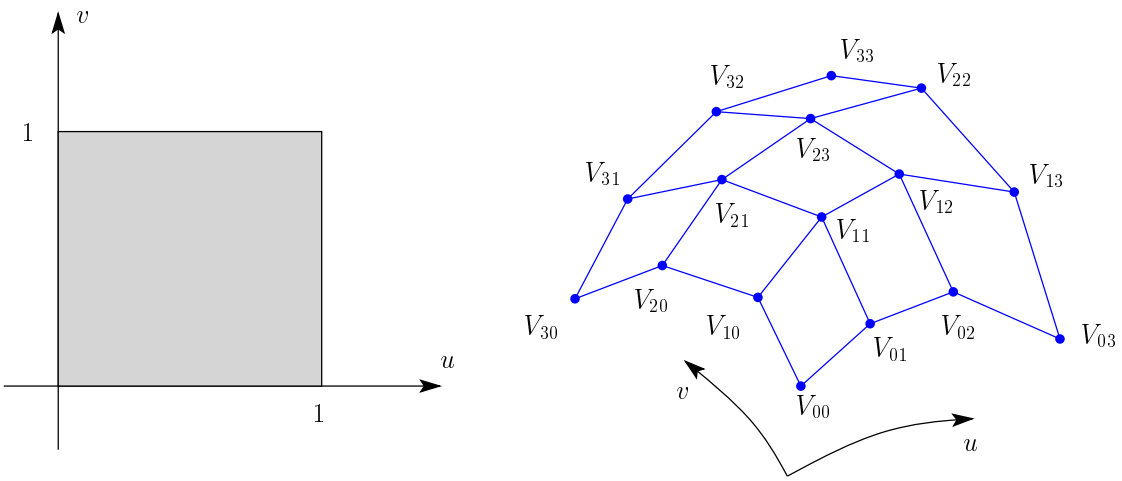
Several variations are possible:

- Scale $C(u)$ as it moves, possibly using length of $T(v)$ as a scale factor — great for seashells!

- Morph $C(u)$ into some other curve $C'(u)$ as it moves along $T(v)$.

- Use your imagination....

Tensor product Bézier surfaces



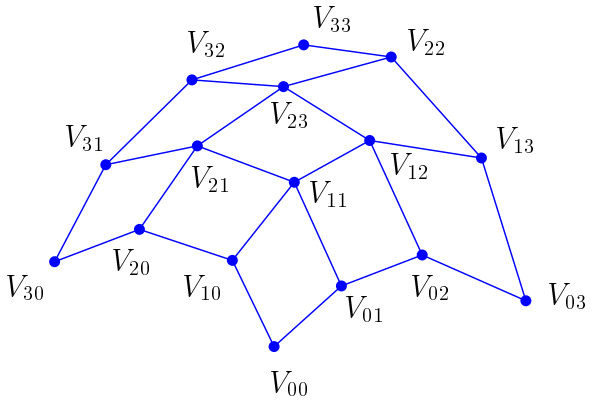
Given: A grid of control points $V_{ij} : i, j = 0, \ldots, n$. Connecting these together we get a "control net."

Construct: A surface $S(u, v)$ by

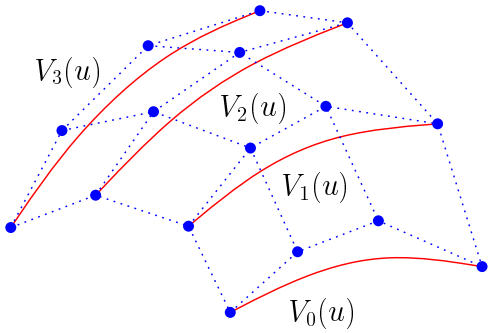
- Treating rows of V as control points for curves $V_0(u), \ldots, V_n(u)$.
- Treating $V_0(u), \ldots, V_n(u)$ as control points for a curve parameterized by v.

Tensor product Bézier surfaces, cont'd

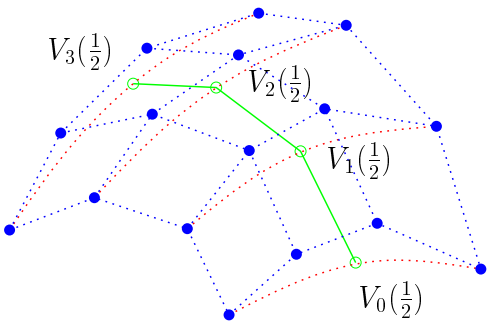
Let's walk through the steps:



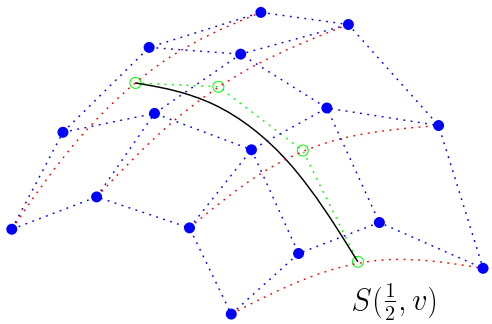
Control net



Control curves in u



Control polygon at $u = \frac{1}{2}$



Curve at $S(\frac{1}{2}, v)$

Q: Which control points are interpolated by the surface?

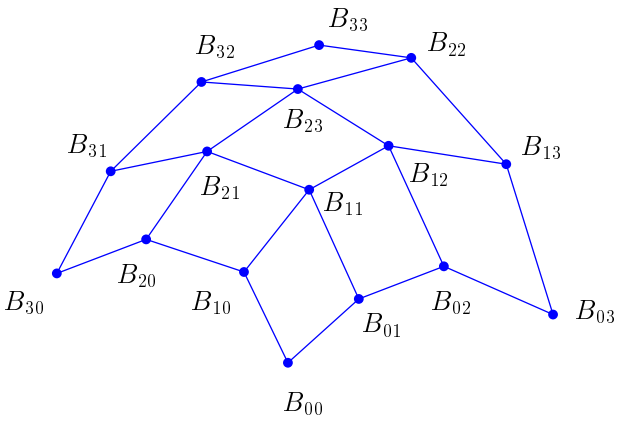
Matrix form

Tensor product surfaces can be written out explicitly:

$$
\begin{aligned}
S(u,v) &= \sum_{i,j=0,\ldots,n} V_{ij} B_i^n(u) B_j^n(v) \\
&= \begin{bmatrix} v^3 & v^2 & v & 1 \end{bmatrix} M_{\text{Bézier}}\, V\, M_{\text{Bézier}}^{\text{T}} \begin{bmatrix} u^3 \\ u^2 \\ u \\ 1 \end{bmatrix}
\end{aligned}
$$

Tensor product B-spline surfaces

As with spline curves, we can piece together a sequence of Bézier surfaces to make a spline surface. If we enforce C^2 continuity and local control, we get B-spline curves:



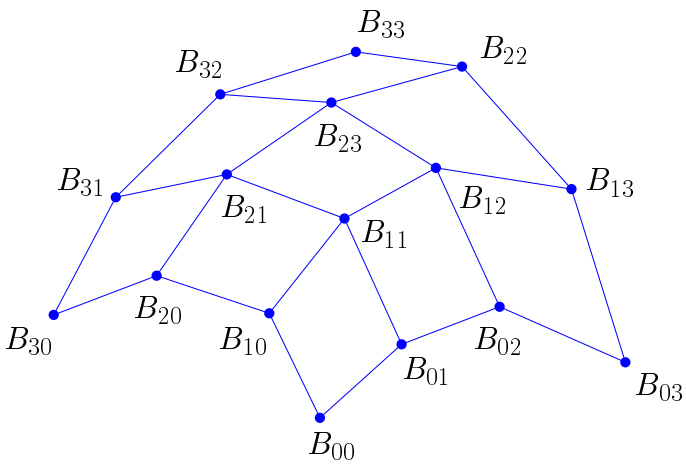
Given: A grid of control points $B_{ij} : i, j = 0, \ldots, n$.

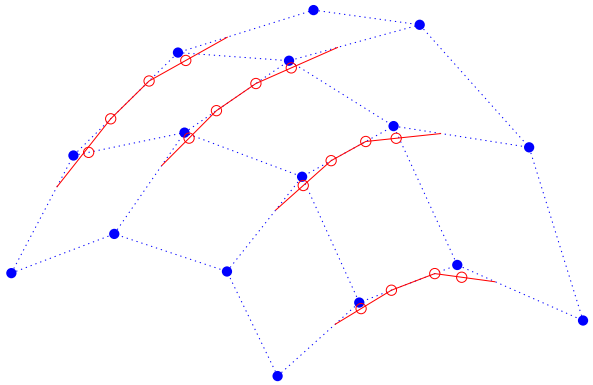
Construct: A Bézier control net $V_{ij} : i, j = 0, \ldots, n$.

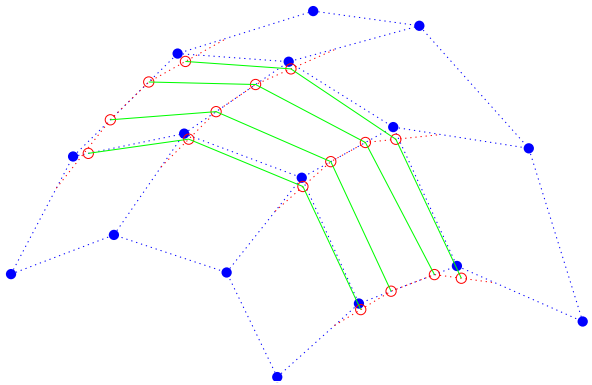
- Treating rows of B as control points to generate Bézier control points in u.
- Treating Bézier control points in u as B-spline control points in v.
- Treating B-spline control points in v to generate Bézier control points in v.

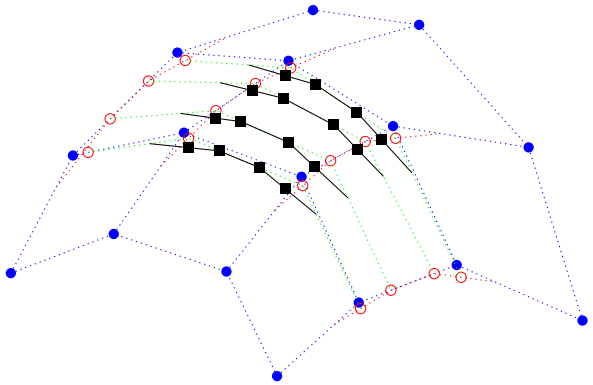
Tensor product B-spline surfaces, cont'd

Let's walk through the steps:





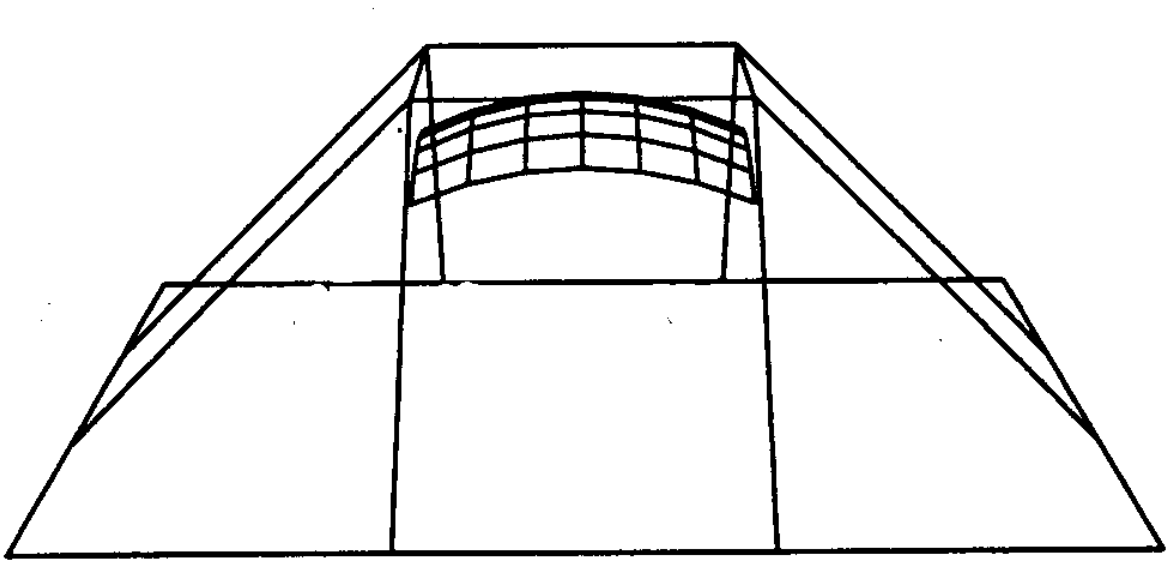




Q: Which B-spline control points are interpolated by the surface?

Tensor product B-spline surfaces, cont'd

Another example:



B-spline surface and control net (Bartels, et al)

Matrix form

Tensor product surfaces can be written out explicitly:

$$
\begin{aligned}
S(u,v) &= \sum_{i,j=0,\ldots,n} B_{ij} N_i^n(u) N_j^n(v) \\
&= \begin{bmatrix} v^3 & v^2 & v & 1 \end{bmatrix} M_{\text{Bézier}}\, M_{\text{B-spline}}\, B\, M_{\text{B-spline}}^{\text{T}}\, M_{\text{Bézier}}^{\text{T}} \begin{bmatrix} u^3 \\ u^2 \\ u \\ 1 \end{bmatrix}
\end{aligned}
$$

Trimmed NURBS surfaces

Uniform, B-spline surfaces are a special case of NURBS surfaces.

Sometimes, we want to have control over which parts of a NURBS surface get drawn.

For example:

We can do this by "trimming" the $u - v$ domain:

1. Define a closed curve in the $u - v$ domain (a "trim curve").
2. Do not draw the surface points inside of this curve.

Problem: It's really hard to maintain continuity in these regions, especially while animating.

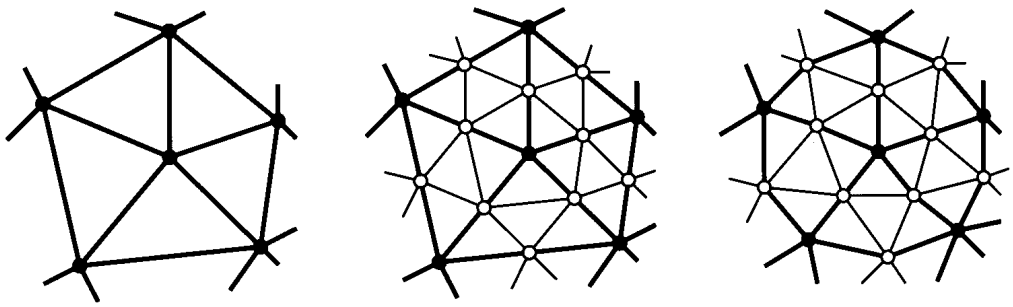
Subdivision surfaces

Chaikin's use of subdivision for curves inspired similar techniques for surfaces....

Idea: Iteratively refine a *control polyhedron* (or *control mesh*) $M^0 \to M^1 \to \cdots$ to a limit surface

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

using splitting and averaging steps:



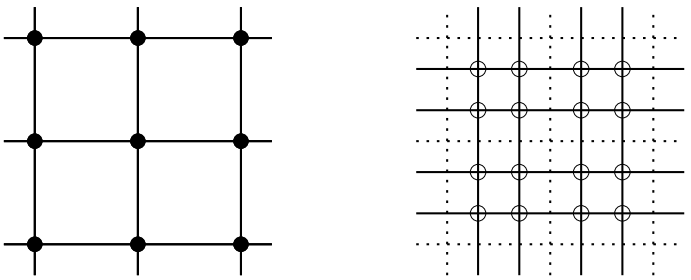
Splitting and averaging steps (Stollnitz et al., 10.4)

Two types of splitting steps:

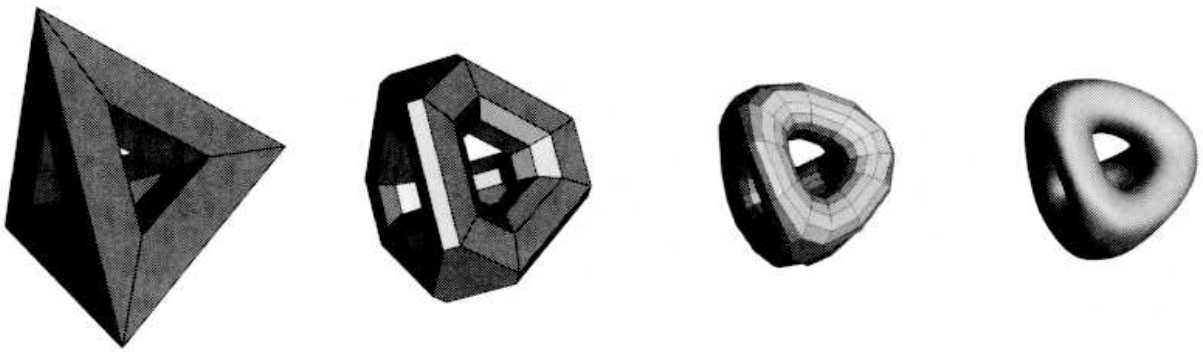
- *Vertex schemes*
- *Face schemes*

Vertex schemes

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



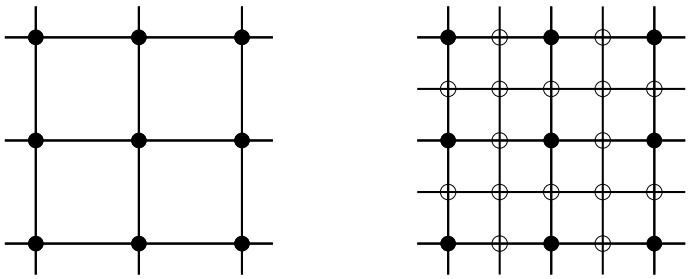
Splitting step for Doo-Sabin subdivision



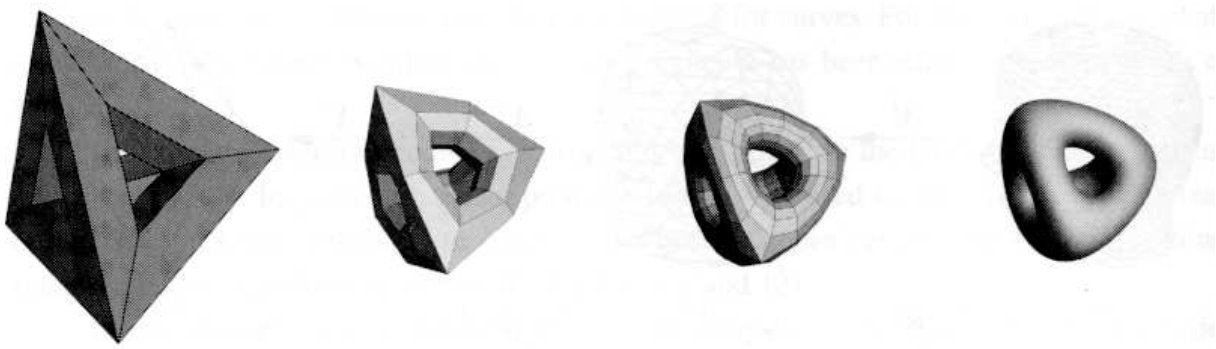
Doo-Sabin subdivision (Stollnitz et al., 10.3)

Face schemes

Each quadrilateral face is split into four subfaces:



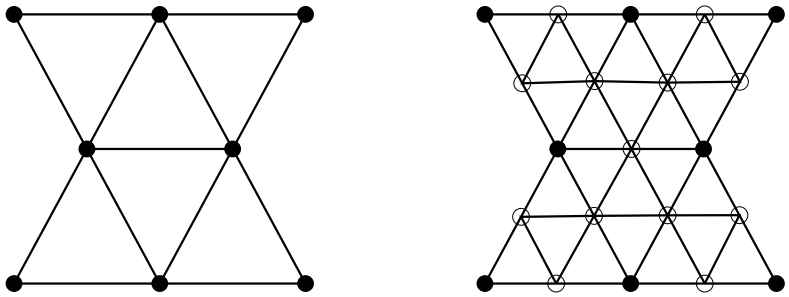
Splitting step for Catmull-Clark subdivision



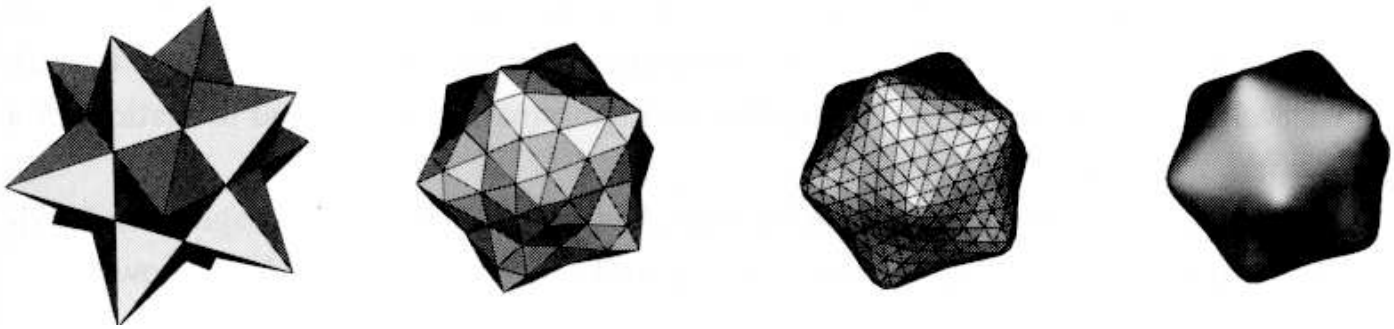
Catmull-Clark subdivision (Stollnitz et al., 10.2)

Face schemes, cont.

Each triangular face is split into four subfaces:



Splitting step for Loop subdivision



Loop subdivision (Stollnitz et al., 10.5)

Averaging step

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

where

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

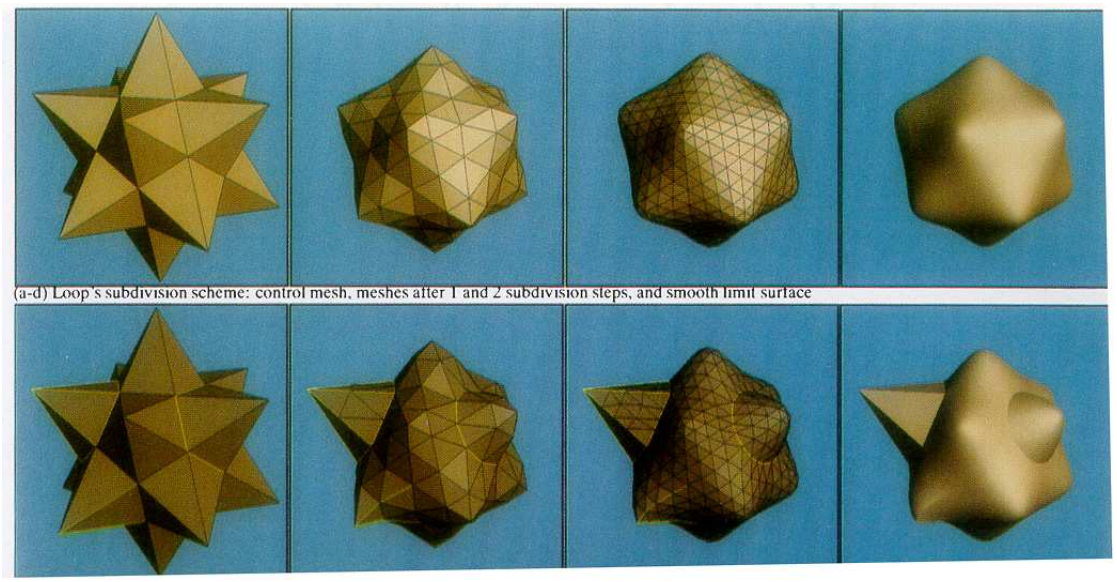
(α carefully chosen to ensure smoothness.)

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:

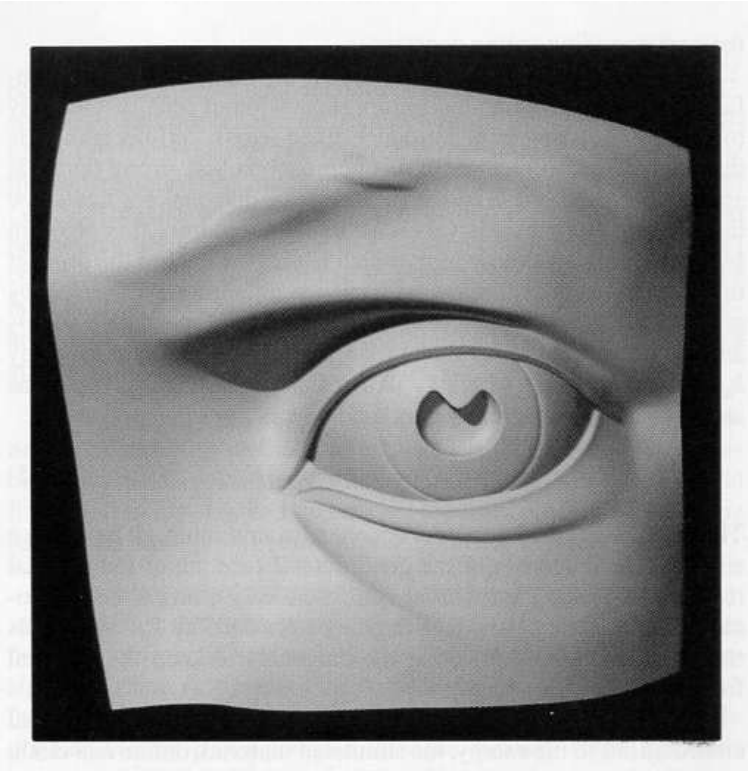
This gives rise to G^0 continuous surfaces:



From Hoppe, et al, SIGGRAPH '94.

Adding creases without trim curves, cont'd

Here's an example using Catmull-Clark surfaces of the kind found in Geri's Game:



From Derose and Kass, SIGGRAPH '98.

Interpolating subdivision surfaces

Interpolating schemes are defined by:

- Splitting
- Averaging only new vertices

Summary

What to take home from this set of lectures:

1. How to construct swept surfaces from a profile and trajectory curve:
    - With a fixed frame
    - With a Frenet frame
2. How to construct tensor product Bézier surfaces
3. How to construct tensor product B-spline surfaces
4. How to construct subdivision surfaces from their averaging masks