15. Parametric surfaces

Reading

Required:

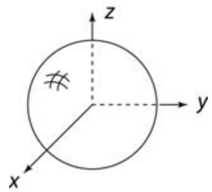
- Watt, 2.1.4, 3.4-3.5.

Optional

- Watt, 3.6.
- Bartels, Beatty, and Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling,* 1987.

Mathematical surface representations

- Explicit $z=f(x,y)$ (a.k.a., a "height field")
  - what if the curve isn't a function, like a sphere?



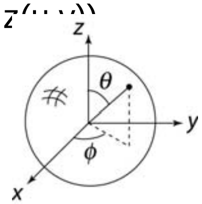
- Implicit $g(x,y,z) = 0$

- Parametric $S(u,v)=(x(u,v),y(u,v),z(u,v))$
  - For the sphere:

    $x(u,v) = r \cos 2\pi v \sin \pi u$

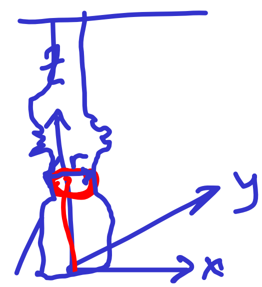
    $y(u,v) = r \sin 2\pi v \sin \pi u$

    $z(u,v) = r \cos \pi u$



As with curves, we'll focus on parametric surfaces.

Surfaces of revolution



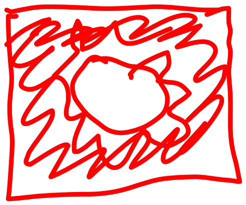
Idea: rotate a 2D **profile curve** around an axis.

What kinds of shapes can you model this way?

Find: A surface $S(u,v)$ which is *radius(z)* rotated about the Z axis.

$S(u,v) = (r(u)\cos\theta,\ r(u)\sin(\theta),\ u)$

Extruded surfaces



Given: A curve $C(u)$ in the xy-plane:

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

.

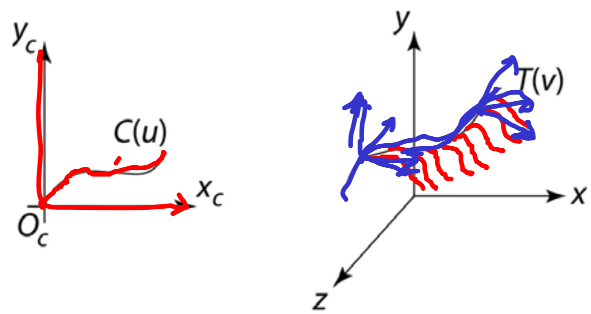
Find: A surface $S(u,v)$ which is $C(u)$ extruded along the Z axis.

Solution:

General sweep surfaces

The **surface of revolution** is a special case of a **swept surface**.

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



More specifically:

- Suppose that $C(u)$ lies in an (x_c,y_c) coordinate system with origin O_c.
- For every point along $T(v)$, lay $C(u)$ so that O_c coincides with $T(v)$.

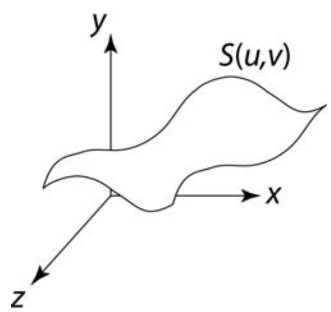
Orientation

The big issue:

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

Here are two options:

1. **Fixed** (or **static**): Just translate O_c along $T(v)$.

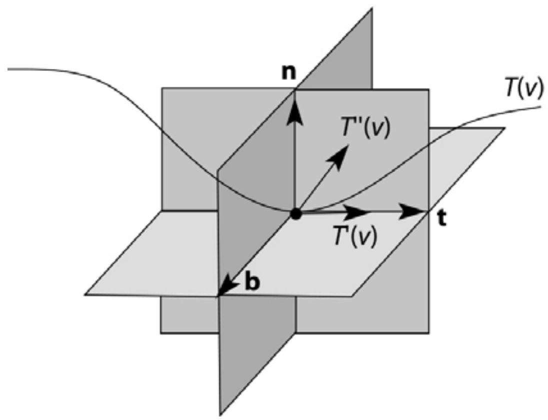


2. Moving. Use the **Frenet frame** of $T(v)$.

- 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.



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

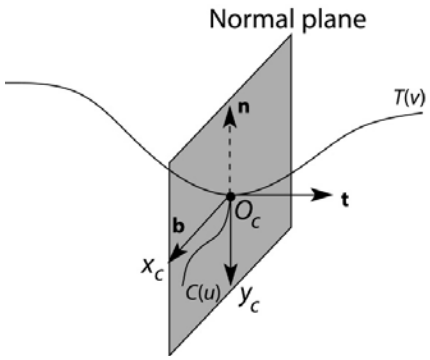
$$\mathbf{t}(v) = \text{normalize}[T'(v)]$$
$$\mathbf{b}(v) = \text{normalize}[T'(v) \times T''(v)]$$
$$\mathbf{n}(v) = \mathbf{b}(v) \times \mathbf{t}(v)$$

As we move along $T(v)$, the Frenet frame (t,b,n) varies smoothly.

Frenet swept surfaces

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

- Put *C*(*u*) in the **normal plane** .
- Place O_c on *T*(*v*).
- Align x_c for *C*(*u*) with **b**.
- Align y_c for *C*(*u*) with **-n**.



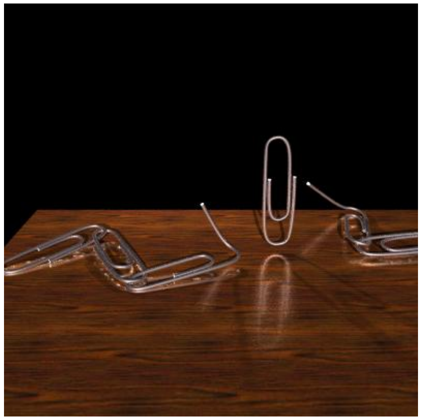
If *T*(*v*) is a circle, you get a surface of revolution exactly!

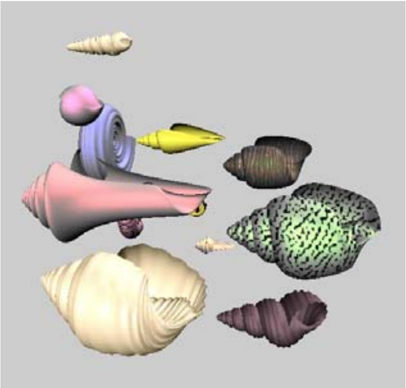
What happens at inflection points, i.e., where curvature goes to zero?

Variations

Several variations are possible:

- Scale *C*(*u*) as it moves, possibly using length of *T*(*v*) as a scale factor.
- Morph *C*(*u*) into some other curve $\tilde{C}(u)$ as it moves along *T*(*v*).
- …





Directly defining parametric surf.

Flashback to curves:

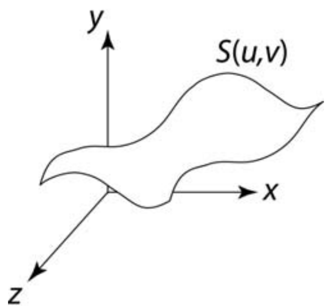
We directly defined parametric function f(u), as a cubic polynomial.

Why a cubic polynomial?

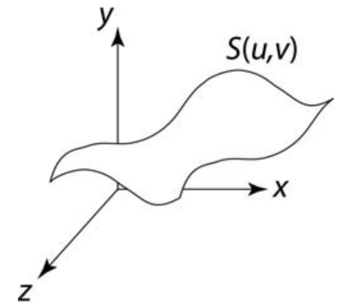
- minimum degree for C2 continuity
- "well behaved"

Can we do something similar for surfaces?

Initially, just think of a height field:
height = f(u,v).



Cubic patches



Cubics curves are good… Let's extend them in the obvious way to surfaces:

$$f(u) = 1 + u + u^2 + u^3$$
$$g(v) = 1 + v + v^2 + v^3$$
$$f(u)g(v) = 1 + u + v + uv + u^2 + v^2 + uv^2 + vu^2 + \ldots + u^3v^3$$

16 terms in this function.

Let's allow the user to pick the coefficient for each of them:

$$f(u)g(v) = c_0 + c_1u + c_2v + \ldots + c_{15}u^3v^3$$

Interesting properties

$$f(u,v) = c_0 + c_1 u + c_2 v + \ldots + c_{15} u^3 v^3$$

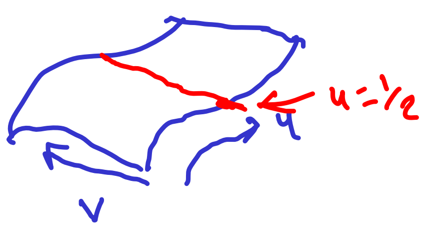
What happens if I pick a particular ‘u’?

$$f(\tfrac{1}{2}, v) = d_0 + d_1 v + d_2 v^2 + d_3 v^3$$

What happens if I pick a particular ‘v’?

$$f(u,v) =$$

What do these look like graphically on a patch?

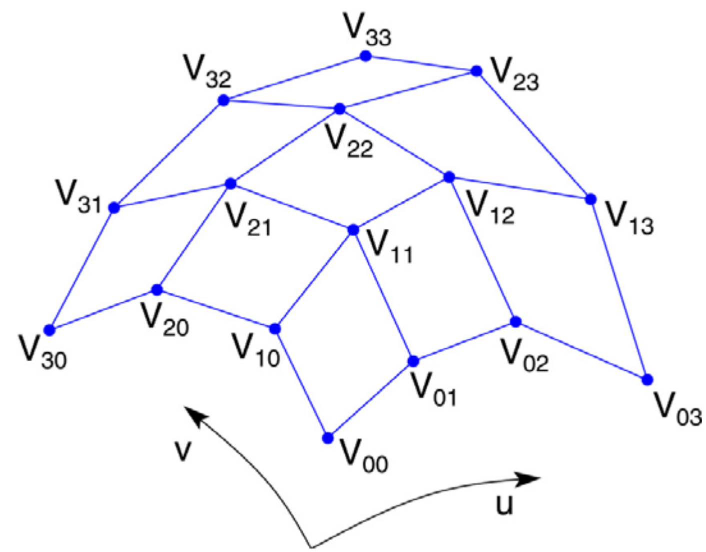


Use control points

As before, directly manipulating coefficients is not intuitive.

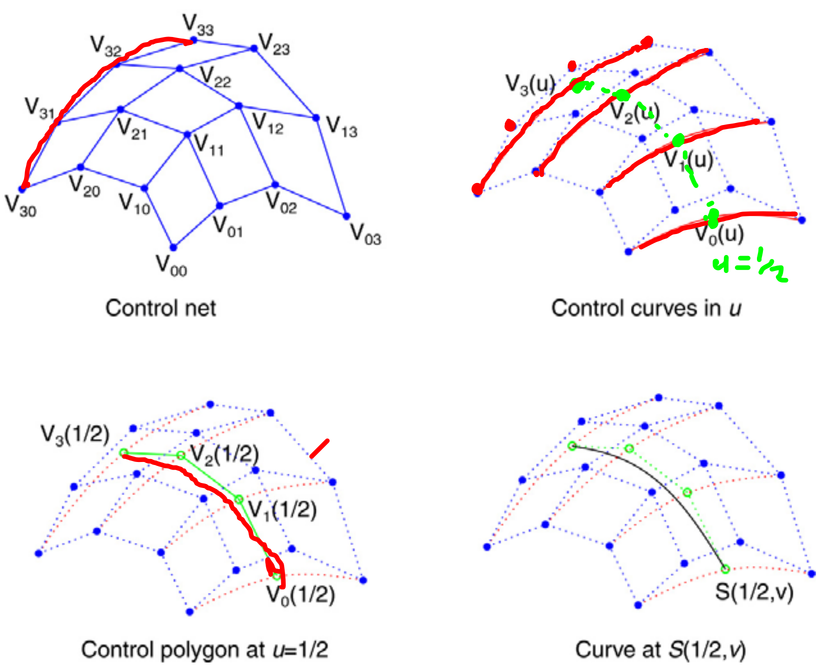
Instead, directly manipulate control points.

These control points indirectly set the coefficients, using approaches like those we used for curves.



Defining a tensor product Bézier surface from control points

Let's walk through the steps:



Which control points are interpolated by the surface?

Matrix form of Bézier curves and surfaces

Recall that Bézier curves can be written in terms of the Bernstein polynomials:

$$Q(u) = \sum_{i=0}^{n} V_i b_i(u)$$

They can also be written in a matrix form:

$$Q^T(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_0^T \\ V_1^T \\ V_2^T \\ V_3^T \end{bmatrix}$$

$$= \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \mathbf{M}_{\text{Bezier}} \mathbf{V}_{\text{curve}}$$

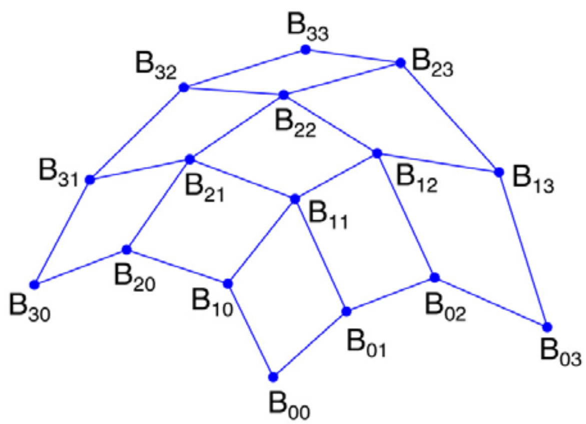
Tensor product surfaces can be written out similarly:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{n} V_{ij} b_i(u) b_j(v)$$

$$= \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \mathbf{M}_{\text{Bézier}} \mathbf{V}_{\text{surface}} \mathbf{M}_{\text{Bézier}}^T \begin{bmatrix} v^3 \\ v^2 \\ v \\ 1 \end{bmatrix}$$

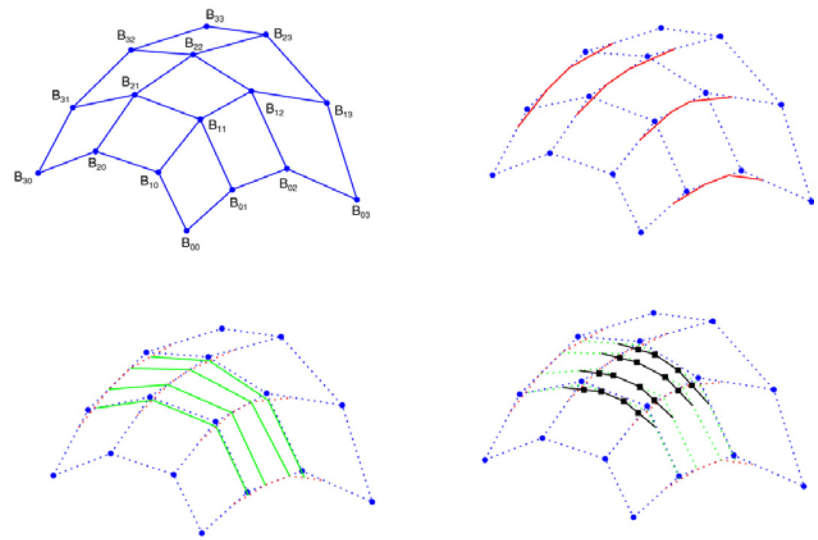
Tensor product B-spline surfaces

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:



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

Tensor product B-spline surfaces, cont.

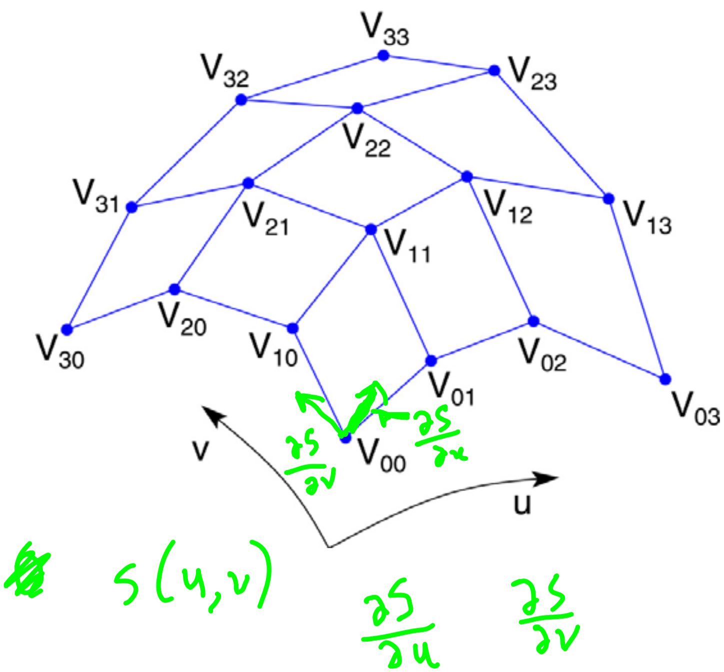


Which B-spline control points are interpolated by the surface?

Continuity for surfaces

Continuity is more complex for surfaces than curves. Must examine partial derivatives at patch boundaries.

G1 continuity refers to tangent plane.

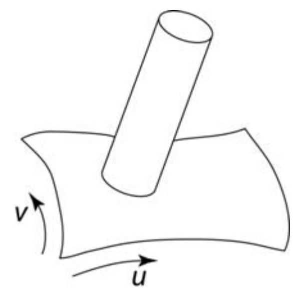


Trimmed NURBS surfaces

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.

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

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

Next class: Subdivision surfaces

Topic:

How do we extend ideas from subdivision curves to the problem of representing surfaces?

Recommended Reading:

- Stollnitz, DeRose, and Salesin. Wavelets for Computer Graphics: Theory and Applications, 1996, section 10.2.
  [Course reader pp. 262-268]