Parametric surfaces

Reading

Required:

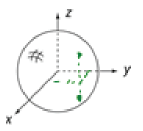
- Angel readings for "Parametric Curves" lecture, with emphasis on 12.1.2, 12.1.3, 12.1.5, 12.6.2, 12.7.3, 12.9.4.

Optional

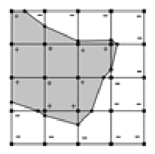
- Bartels, Beatty, and Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*, 1987.

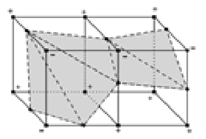
Mathematical surface representations

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



- Implicit $g(x,y,z) = 0$ $g(x,y,z)=x^2+y^2+z^2-r^2=0$


Isocontour from "marching squares"

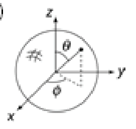

Isocontour from "marching cubes"

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

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

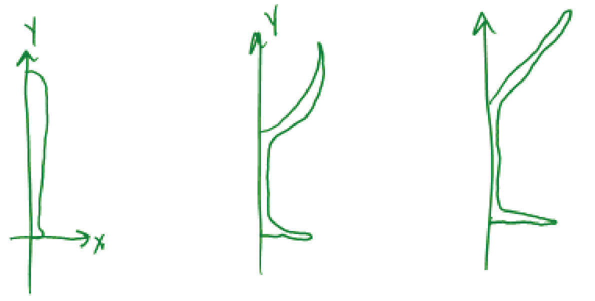
$y(u,v) = r \sin 2\pi v \sin \pi u$

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



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

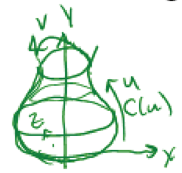
Surfaces of revolution



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

What kinds of shapes can you model this way?

Constructing surfaces of revolution



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

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

Let $R_y(\theta)$ be a rotation about the y-axis.

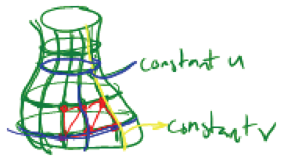
Find: A surface $S(u,v)$ which is $C(u)$ rotated about the y-axis.

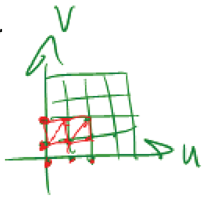
Solution:

$$S(u,v) = R_y(2\pi v)\, C(u)$$

Isoparameter curves and tangents

We can follow curves where v is constant, and u varies or vice versa. These are called **isoparameter curves** (where one parameter is held constant):





If we sample at equal spacing in u and v, we can create a quadrilateral mesh (or a triangle mesh).

We can compute tangents to the surface at any point by looking at (infintesimally) nearby points.

Holding one parameter constant, we can find nearby points by varying the other parameter. Thus, we can get two tangents:

$$\mathbf{t}_u = \frac{\partial S(u,v)}{\partial u} \qquad \mathbf{t}_v = \frac{\partial S(u,v)}{\partial v}$$

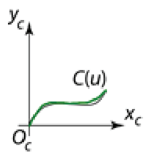
How would we compute the normal?

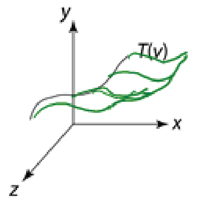
$$\pm\, t_u \times t_v$$

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 $T(v)$.

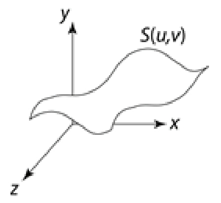
Orientation

The big issue:

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

Here are two options:

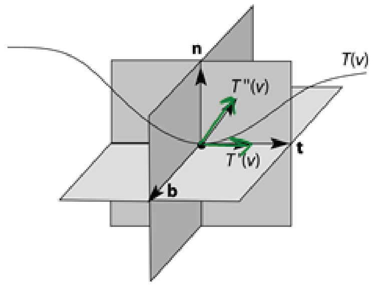
1. **Fixed** (or **static**): Just translate O_c along $T(v)$.



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.

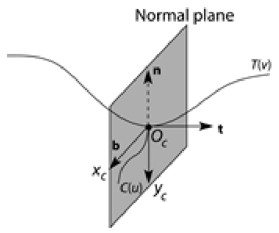
Tangent:	$\mathbf{t}(v) = \text{normalize}[T'(v)]$
Binormal:	$\mathbf{b}(v) = \text{normalize}[T'(v) \times T''(v)]$
Normal:	$\mathbf{n}(v) = \mathbf{b}(v) \times \mathbf{t}(v)$

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 $T(v)$:

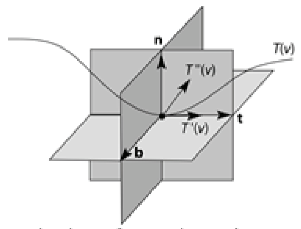
- Put $C(u)$ in the **normal plane**.
- Place O_c on $T(v)$.
- Align x_c for $C(u)$ with **b**.
- Align y_c for $C(u)$ with **-n**.



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

Degenerate frames

Let's look back at where we computed the coordinate frames from curve derivatives:

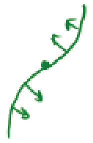


$t = n(T'(v))$
$b = n(T'(v) \times T''(v))$
$n = b \times t$

Where might these frames be ambiguous or undetermined?

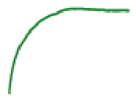
1. $T'(v) = 0 \Rightarrow$ go to arc length parameterization

2. $T''(v) = 0$



negate 2nd deriv. after passing inflection point

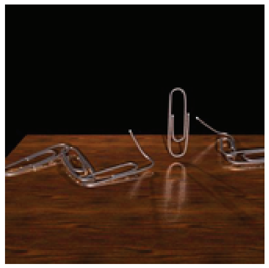
3.

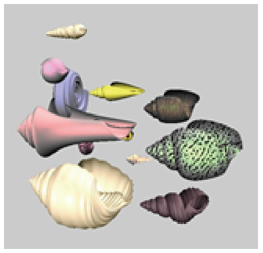


Variations

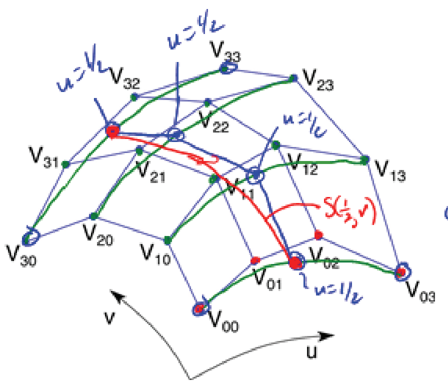
Several variations are possible:

- 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 $T(v)$.
- ...





Tensor product Bézier surfaces



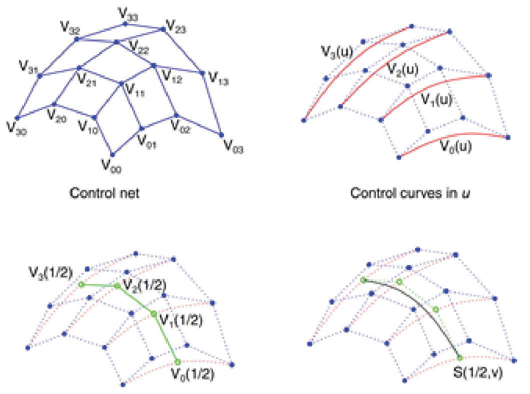
4 Corner control pts will be interpolated

Given a grid of control points V_{ij}, forming a **control net**, contruct a surface $S(u,v)$ by:

- treating rows of V (the matrix consisting of the V_{ij}) 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.

Let's walk through the steps:



Control net

Control curves in u

Control polygon at u=1/2

Curve at S(1/2,v)

Which control points are interpolated by the surface?

Matrix form of cubic Bézier curves and surfaces

Recall that cubic Bézier curves can be written in polynomial form (then expanded):

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

$$\rightarrow\ = (1-u)^3 V_0 + 3u(1-u)^2 V_1 + 3u^2(1-u) V_2 + u^3 V_3$$

$$\rightarrow\ = (-u^3+3u^2-3u+1)V_0 + (3u^2-6u+3)V_1 + (-3u^2+3u)V_2 + u^3 V_3$$

They can also be written in a matrix form:

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

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

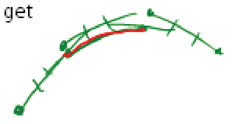
The tensor product surface can be written out similarly:

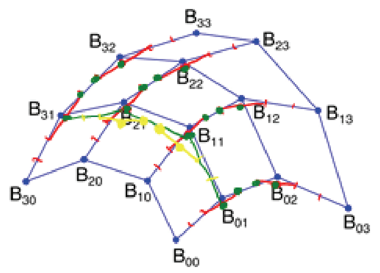
$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{n} V_{ij} b_i(u) b_j(v)$$

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

Tensor product B-spline surfaces

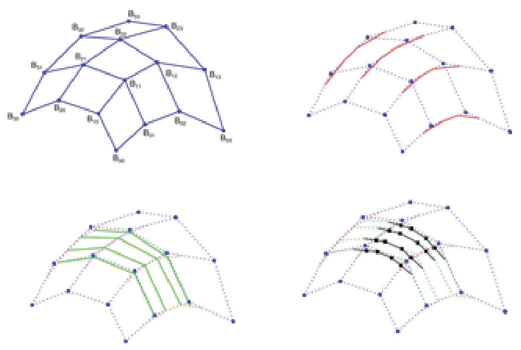
As with spline curves, we can piece together a sequence of Bézier surfaces to make a spline surface. If we enforce C^2 continuity and local control, we get B-spline curves:





- 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?

Matrix form of B-spline surfaces

For curves, we can write a matrix that generates Bezier control points from B-spline control points:

$$\begin{bmatrix} V_0^T \\ V_1^T \\ V_2^T \\ V_3^T \end{bmatrix} = \begin{bmatrix} 1/6 & 2/3 & 1/6 & 0 \\ 0 & 2/3 & 1/3 & 0 \\ 0 & 1/3 & 2/3 & 0 \\ 0 & 1/6 & 2/3 & 1/6 \end{bmatrix} \begin{bmatrix} B_0^T \\ B_1^T \\ B_2^T \\ B_3^T \end{bmatrix}$$

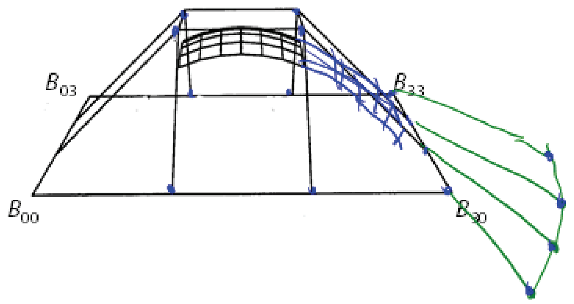
$$\mathbf{V}_{curve} = \mathbf{M}_{B\text{-}spline}\mathbf{B}_{curve}$$

We can arrive at a similar form for tensor product B-spline surfaces:

$$\mathbf{V}_{surface} = \mathbf{M}_{B\text{-}spline}\mathbf{B}_{surface}\mathbf{M}_{B\text{-}spline}^T$$

Tensor product B-splines, cont.

Another example:

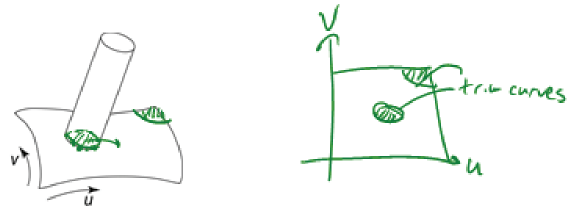


Trimmed NURBS surfaces

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.

Summary

What to take home:

- How to construct a surface of revolution
- How to construct swept surfaces from a profile and trajectory curve:
  - with a fixed frame
  - with a Frenet frame
- How to construct tensor product Bézier surfaces
- How to construct tensor product B-spline surfaces