Parametric surfaces

CSE 457
Winter 2014

Reading

Required:

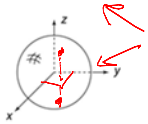
- Angel readings for "Parametric Curves" lecture, with emphasis on 10.1.2, 10.1.3, 10.1.5, 10.6.2, 10.7.3, 10.9.4.

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?



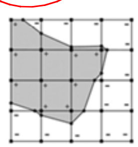
f(x, y) produces up to 2 answers
⇒ not a function

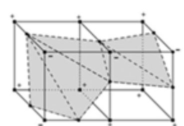
$x^2+y^2+z^2=r^2$

$\underbrace{x^2+y^2+z^2-r^2}=0$

$g(x,y,z) = \quad = 0$

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


Isocontour from "marching squares"

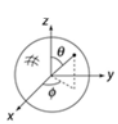

Isocontour from "marching cubes"

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

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

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

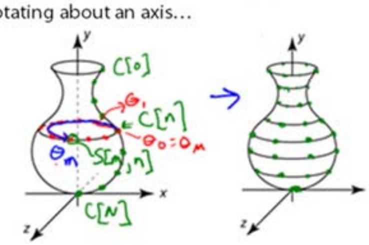
    $z(u,v) = r\cos \pi u$



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

Surfaces of revolution

Recall that surfaces of revolution are based on the idea of rotating about an axis…



Given: A set of points $C[n]$ on a curve in the xy-plane:

$$C[n] = \begin{bmatrix} C_x[n] \\ C_y[n] \\ 0 \\ 1 \end{bmatrix} \quad \text{where } n \in [0, N]$$

$\Delta\theta$

↓

$\theta_m = \frac{2\pi}{M} \cdot m$

Let $R_y(\theta_m)$ be a rotation about the y-axis by angle θ_m.

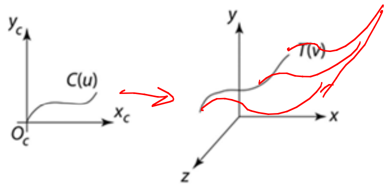
Find: A set of points $S[m,n]$ on the surface formed by rotating $C[n]$ rotated about the y-axis. Assume $m \in [0, M]$.

Solution: $S[m,n] = R_y(\frac{2\pi}{M}m)C[n]$

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 $T(v)$.

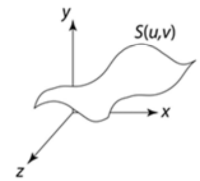
Orientation

The big issue:

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

Here are two options:

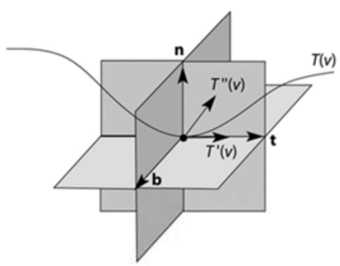
1. **Fixed** (or **static**): Just translate O_c along $T(v)$.



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

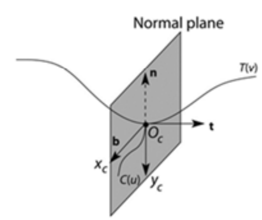
Normal: $\mathbf{n}(v) = \mathbf{b}(v) \times \mathbf{t}(v)$

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!

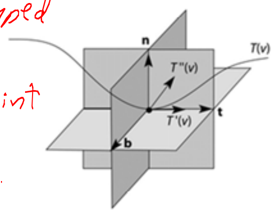
Degenerate frames

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

$T'(v) = 0 \Rightarrow$ Stopped

$\downarrow$

End point



Recall

$$t = \frac{T'(v)}{\|T'(v)\|}$$

$$b = \frac{T'(v) \times T''(v)}{\|T'(v) \times T''(v)\|}$$

$$n = \frac{b \times t}{\|b \times t\|}$$

Where might these frames be ambiguous or undetermined?

Sharp Turns $T'(v)$ undefined

$T'(v) \parallel T''(v)$

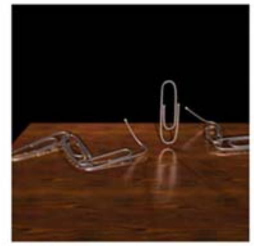
Handle with simpler model, e.g. Fixed/Static $C(u)$ orientation

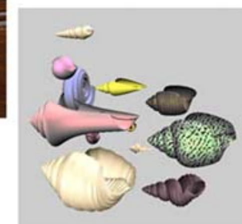
Inflection Points

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

Curves

Bezier ctrl polygon

$\downarrow$

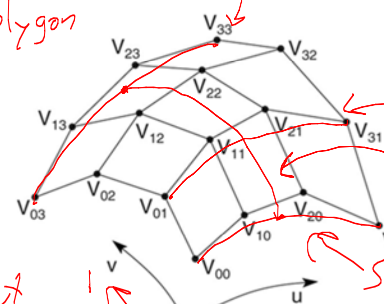
Bezier curve segment

Surfaces

Bezier ctrl net

$\downarrow$

Bezier surface patch



$S(u, v=1)$

$S(u, v=1/3)$

$S(u=1/2, v)$

$S(u, v=0)$

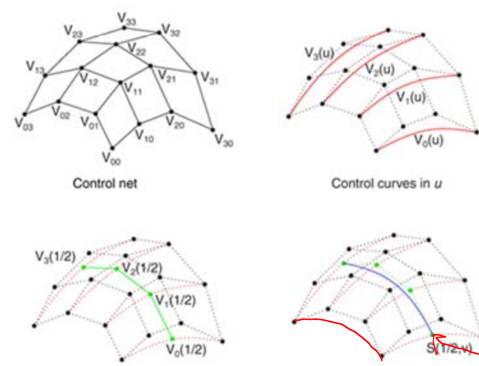
1 0 0 1

Given a grid of control points V_{ij}, forming a **control net**, construct 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 $S(1/2,v)$

Connect

Which control points are interpolated by the surface?

Corners

Polynomial form of Bézier surfaces

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

Curves

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

A tensor product Bézier surface can be written as:

Surfaces

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

In the previous slide, we constructed curves along u, and then along v. This corresponds to re-grouping the terms like so:

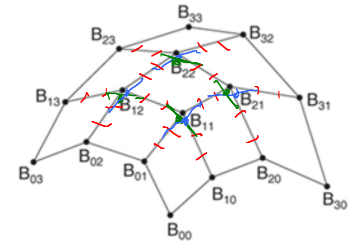
$$S(u,v) = \sum_{j=0}^{n} \left(\sum_{i=0}^{n} V_{ij} b_i(u) \right) b_j(v)$$

But, we could have constructed them along v, then u:

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

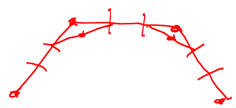
Tensor product B-spline surfaces

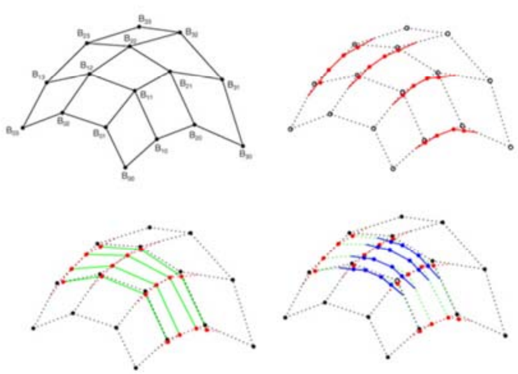
As with spline curves, we can piece together a sequence of Bézier surfaces to make a spline surface. If we enforce C^2 continuity and local control, we get B-spline curves:



- 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.

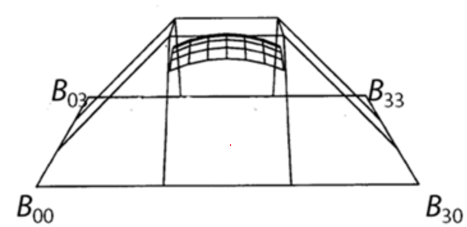




Which B-spline control points are interpolated by the surface?

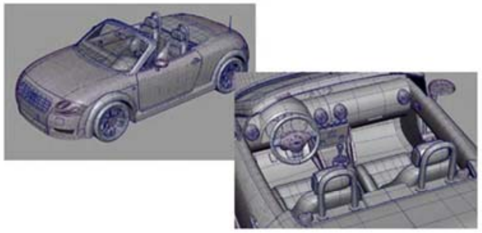
Tensor product B-splines, cont.

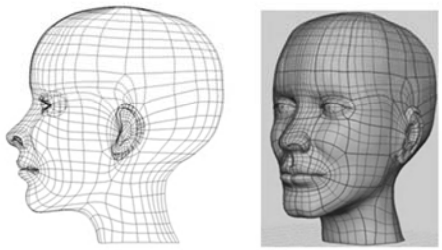
Another example:



NURBS surfaces

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

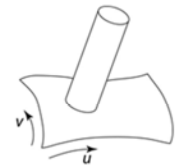




Trimmed NURBS surfaces

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 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