Parametric surfaces

Daniel Leventhal
Adapted from Brian Curless
CSE 457
Autumn 2011

Mathematical surface representations

- Explicit \( z = (x,y) \) (a.k.a., a “height field”)
  - what if the curve isn’t a function, like a sphere?

- Implicit \( (x, y) = 0 \)

- Parametric \( S(u,v) = (x(u,v), y(u,v), z(u,v)) \)
  - For the sphere:
    \[
    x(u,v) = r \cdot 2\pi\sin\pi u \\
    y(u,v) = 2\pi\sin\pi v \\
    z(u,v) = \cos\pi 
    \]
  - As with curves, we’ll focus on parametric surfaces.

Reading

Optional
- 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.

Surfaces of revolution

Idea: rotate a 2D profile curve

What kinds of shapes can you model this way?
Constructing surfaces of revolution

Given: -plane:

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

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

Find: -axis.

Solution:

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 (infinitesimally) nearby points.

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

\[ t_u = \frac{\partial S(u,v)}{\partial u} \quad t_v = \frac{\partial S(u,v)}{\partial v} \]

How would we compute the normal?

General sweep surfaces

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

\[ C(u) \] along a trajectory curve \( T(v) \)

More specifically:

- Suppose that \( C(u) \) lies in an \( x,y,z \) system with origin \( O_z \)
- For every point along \( C(u) \), lay \( C(u) \) so that it 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 (static): Just translate along \( C(u) \).

2. Moving. Use the of \( C(u) \):

- Allows smoothly varying orientation.
- Permits surfaces of revolution, for example.
**Frenet frames**

Motivation: Given a curve $\gamma(t)$, we want to attach a smoothly varying coordinate system.

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

- **Tangent**: $t(v) = \text{normalize}(\gamma'(v))$
- **Binormal**: $b(v) = \text{normalize}(\gamma'(v) \times \gamma''(v))$
- **Normal**: $n(v) = b(v) \times t(v)$

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

**Frenet swept surfaces**

Orient the profile curve $C(u)$ using the Frenet frame of the trajectory $\gamma(t)$:

- Put $\gamma(t)$ in the **normal plane**.
- Place $O_t$ on $T(v)$.
- Align $\gamma(t)$ with $b$.
- Align $\gamma(t)$ 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:

Where might these frames be ambiguous or undetermined?

**Variations**

Several variations are possible:

- Scale $C(u)$ as it moves, possibly using length of $T(v)$ as a scale factor.
- Morph $\gamma(t)$ into some other curve $\tilde{C}(u)$ as it moves along $T(v)$.
- ...
Given a grid of control points $V_i^j$, construct a surface $S(u,v)$ by:

- treating rows of $V$ (the matrix consisting of the $V_i^j$) as control points for curves $V_0^j, V_1^j, \ldots, V_n^j$ as control points for a curve
- treating $V_0^j(u), \ldots, V_n^j(u)$ as control points for a curve

Let's walk through the steps:

Which control points are interpolated by the surface?

Polynomial form of Bézier surfaces

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

$$Q(u) = \sum_{i=0}^{3} V_i b_i(u)$$

A tensor product Bézier surface can be written as:

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

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

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

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

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

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 $B$ 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 \textbf{trimming} -v

\begin{itemize}
  \item \textbf{trim curve}
  \item Do not draw the surface points inside of this curve.
\end{itemize}

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