Parametric surfaces

Adriana Schulz
CSE 457
Spring 2020
Reading

Optional reading:

- Angel and Shreiner 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.
- Marschner and Shirley, 2.5.

Further reading

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.
Constructing surfaces of revolution

Given: A curve \( C(v) \) in the \( xy \)-plane:

\[
C(v) = \begin{bmatrix}
C_x(v) \\
C_y(v) \\
0 \\
1
\end{bmatrix}
\]

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

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

Solution: \( S(u,v) = R_y(2\pi u)C(v) \)
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 $T(v)$.

   ![Diagram](image-url)

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

2. Moving. Use the **Frenet frame** of $T(v)$.
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:** \( \mathbf{n}(v) = \mathbf{b}(v) \cdot \mathbf{t}(v) \)

As we move along \( T(v) \), the Frenet frame \( (\mathbf{t}, \mathbf{b}, \mathbf{n}) \) 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:

\[
\langle T', T' \rangle = 1 \\
\langle 2T', T' \rangle = 0 \\
0 = \langle T'', T' \rangle + \langle T', T' \rangle
\]

Where might these frames be ambiguous or undetermined?

\[
\begin{align*}
\langle T', n \rangle &= 0 \\
\langle T'', n \rangle &= 0
\end{align*}
\]
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

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:

Which control points are always interpolated by the surface?
Polynomial form of Bézier surfaces

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

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

A tensor product Bézier surface can be written as:

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

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 always 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