Cubic Bezier splines

- \( P(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4 \)
Bernstein Polynomials

For cubic:

- $B_1(t) = (1-t)^3$
- $B_2(t) = 3t(1-t)^2$
- $B_3(t) = 3t^2(1-t)$
- $B_4(t) = t^3$

- (careful with indices, many authors start at 0)

- But defined for any degree
General Spline Formulation

\[ Q(t) = GBT(t) = \text{Geometry } G \cdot \text{Spline Basis } B \cdot \text{Power Basis } T(t) \]

- Geometry: control points coordinates assembled into a matrix \((P_1, P_2, \ldots, P_{n+1})\)
- Power basis: the monomials \(1, t, t^2, \ldots\)
- Cubic Bézier:

\[
P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}
\]
Questions?
• What if we want to transform each point on the curve with a linear transformation $M$?

$$P'(t) = M \begin{pmatrix} P_1,x & P_2,x & P_3,x & P_4,x \\ P_1,y & P_2,y & P_3,y & P_4,y \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$
Linear Transformations & Cubics

• What if we want to transform each point on the curve with a linear transformation $M$?
  – Because everything is linear, it’s the same as transforming the only the control points

$$P'(t) = M \begin{pmatrix} P_{1,x} & P_{2,x} & P_{3,x} & P_{4,x} \\ P_{1,y} & P_{2,y} & P_{3,y} & P_{4,y} \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

$$= M \begin{pmatrix} P_{1,x} & P_{2,x} & P_{3,x} & P_{4,x} \\ P_{1,y} & P_{2,y} & P_{3,y} & P_{4,y} \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$
Linear Transformations & Cubics

• Homogeneous coordinates also work
  – Means you can translate, rotate, shear, etc.
  – Also, changing $w$ gives a “tension” parameter

• Note though that you need to normalize $P'$ by $1/w$

$$P'(t) = M \begin{pmatrix} P_{1,x} & P_{2,x} & P_{3,x} & P_{4,x} \\ P_{1,y} & P_{2,y} & P_{3,y} & P_{4,y} \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$
Differential properties of curves

• Motivation
  – Compute normal for surfaces
  – Compute velocity for animation
  – Analyze smoothness
Velocity

- First derivative w.r.t. $t$
- Can you compute this for Bezier curves?

$$P(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4$$

- You know how to differentiate polynomials...
Velocity

- First derivative w.r.t. $t$
- Can you compute this for Bezier curves?

\[ P(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4 \]

\[ P'(t) = -3(1-t)^2 P_1 + [3(1-t)^2 - 6t(1-t)] P_2 + [6t(1-t) - 3t^2] P_3 + 3t^2 P_4 \]

Sanity check: $t=0; t=1$

Can also write this using a matrix $B'$

– try it out!
Tangent

• The tangent to the curve $P(t)$ can be defined as $T(t) = \frac{P'(t)}{||P'(t)||}$
  – normalized velocity, $||T(t)|| = 1$

• This provides us with one orientation for swept surfaces in a while
Questions?
Curvature

• Derivative of unit tangent
  – \( K(t) = T'(t) \)
  – Magnitude \( ||K(t)|| \) is constant for a circle
  – Zero for a straight line

• Always orthogonal to tangent, ie. \( K \cdot T = 0 \)
  – Can you prove this? (Hints: \( ||T(t)||=1 \), \( (x(t)y(t))'=? \))
Geometric Interpretation

• $K$ is zero for a line, constant for circle
  – What constant? $1/r$

• $1/||K(t)||$ is the radius of the circle that touches $P(t)$ at $t$ and has the same curvature as the curve
Curve Normal

- Normalized curvature: $T'(t)/\|T'(t)\|$
Questions?
Orders of Continuity

- $C^0 = \text{continuous}$
  - The seam can be a sharp kink
- $G^1 = \text{geometric continuity}$
  - Tangents point to the same direction at the seam
- $C^1 = \text{parametric continuity}$
  - Tangents are the same at the seam, implies $G^1$
- $C^2 = \text{curvature continuity}$
  - Tangents and their derivatives are the same
Connecting Cubic Bézier Curves

• How can we guarantee $C^0$ continuity?
• How can we guarantee $G^1$ continuity?
• How can we guarantee $C^1$ continuity?
• $C^2$ and above gets difficult
Connecting Cubic Bézier Curves

- Where is this curve
  - $C^0$ continuous?
  - $G^1$ continuous?
  - $C^1$ continuous?

- What’s the relationship between:
  - the # of control points, and the # of cubic Bézier subcurves?
Questions?
Cubic B-Splines

- $\geq 4$ control points
- Locally cubic
  - Cubics chained together, again.
Cubic B-Splines

- ≥ 4 control points
- Locally cubic
  - Cubics chained together, again.
Cubic B-Splines

- $\geq 4$ control points
- Locally cubic
  - Cubics chained together, again.
Cubic B-Splines

- $\geq 4$ control points
- Locally cubic
  - Cubics chained together, again.
Cubic B-Splines

- $\geq 4$ control points
- Locally cubic
  - Cubics chained together, again.
- Curve is not constrained to pass through any control points

A BSpline curve is also bounded by the convex hull of its control points.
Cubic B-Splines: Basis

\[ B_1(t) = \frac{1}{6}(1 - t)^3 \]

\[ B_3(t) = 16(-3t^3 + 3t^2 + 3t + 1) \]

\[ B_2(t) = \frac{1}{6}(3t^3 - 6t^2 + 4) \]

\[ B_4(t) = \frac{1}{6}t^3 \]

These sum to 1, too!
Cubic B-Splines: Basis

\[ Q(t) = \frac{(1 - t)^3}{6} P_{i-3} + \frac{3t^3 - 6t^2 + 4}{6} P_{i-2} + \frac{-3t^3 + 3t^2 + 3t + 1}{6} P_{i-1} + \frac{t^3}{6} P_i \]

\[ Q(t) = \text{GBT}(t) \]

\[ B_{B-\text{Spline}} = \frac{1}{6} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 4 & 0 & -6 & 3 \\ 1 & 3 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \]
Cubic B-Splines

- Local control (windowing)
- Automatically $C^2$, and no need to match tangents!
B-Spline Curve Control Points

- Default B-Spline
- BSpline with derivative discontinuity
- BSpline which passes through end points

Repeat interior control point
Repeat end points
Bézier ≠ B-Spline

But both are cubics, so one can be converted into the other!
Converting between Bézier & BSpline

- Simple with the basis matrices!
  - Note that this only works for a single segment of 4 control points
  \[
  \begin{pmatrix}
  1 & -3 & 3 & -1 \\
  0 & 3 & -6 & 3 \\
  0 & 0 & 3 & -3 \\
  0 & 0 & 0 & 1
  \end{pmatrix}
  \]

- \[ P(t) = G \cdot B_1 \cdot T(t) = \]
  \[
  G \cdot B_1 \cdot (B_2^{-1} \cdot B_2) \cdot T(t) =
  \]
  \[
  (G \cdot B_1 \cdot B_2^{-1}) \cdot B_2 \cdot T(t) =
  \]
  \[
  \frac{1}{6}
  \begin{pmatrix}
  1 & -3 & 3 & -1 \\
  4 & 0 & -6 & 3 \\
  1 & 3 & 3 & -3 \\
  0 & 0 & 0 & 1
  \end{pmatrix}
  \]

- \( G \cdot B_1 \cdot B_2^{-1} \) are the control points for the segment in new basis.

\[ Q(t) = GBT(t) = \text{Geometry } G \cdot \text{Spline Basis } B \cdot \text{Power Basis } T(t) \]
Why Bother with B-Splines?

- Automatic $C^2$ is nice!
- Also, B-Splines can be split into segments of non-uniform length without affecting the global parametrization.
  - “Non-uniform B-Splines”
  - We’ll not do this, but just so you know.
NURBS (Generalized B-Splines)

- Rational cubics
  - Use homogeneous coordinates, just add \( w \)!
    - Provides a “tension” parameter to control points

- NURBS: Non-Uniform Rational B-Spline
  - non-uniform = different spacing between the blending functions, a.k.a. “knots”
  - rational = ratio of cubic polynomials (instead of just cubic)
    - implemented by adding the homogeneous coordinate \( w \) into the control points.
Questions?
Representing Surfaces

• Triangle meshes
  – Surface analogue of polylines, this is what GPUs draw

• Tensor Product Splines
  – Surface analogue of spline curves

• Subdivision surfaces

• Implicit surfaces
  – $f(x,y,z)=0$

• Procedural
  – e.g. surfaces of revolution, generalized cylinder

• From volume data (medical images, etc.)
What you’ve used so far in Asst 0

Triangle represented by 3 vertices

**Pro:** simple, can be rendered directly

**Cons:** not smooth, needs many triangles to approximate smooth surfaces (tessellation)
Smooth Surfaces?

- \( P(u) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4 \)

What’s the dimensionality of a curve? 1D!

What about a surface?
How to Build Them? Here’s an Idea

- \( P(u) = (1-u)^3 \quad P_1 \) 
  + \( 3t(1-u)^2 \quad P_2 \) 
  + \( 3t^2(1-u) \quad P_3 \) 
  + \( u^3 \quad P_4 \) 

(Note! We relabeled \( t \) to \( u \))
How to Build Them? Here’s an Idea

- \( P(u) = (1-u)^3 \) \( P_1 \)
  + \( 3t(1-u)^2 \) \( P_2 \)
  + \( 3t^2(1-u) \) \( P_3 \)
  + \( u^3 \) \( P_4 \)

(Note! We relabeled \( t \) to \( u \))
Here’s an Idea

- \( P(u, v) = (1-u)^3 \) \( P_1(v) \)
  + \( 3t(1-u)^2 \) \( P_2(v) \)
  + \( 3t^2(1-u) \) \( P_3(v) \)
  + \( u^3 \) \( P_4(v) \)

- Let’s make the \( P_i \)s move along curves!
Here’s an Idea

- \( P(u, v) = (1-u)^3 \) \( P_1(v) \)
  + 3t(1-u)^2 \( P_2(v) \)
  + 3t^2(1-u) \( P_3(v) \)
  + u^3 \( P_4(v) \)

- Let’s make the \( P_i \)s move along curves!
Here’s an Idea

- \( P(u, v) = (1-u)^3 P_1(v) \)
  + \( 3t(1-u)^2 P_2(v) \)
  + \( 3t^2(1-u) P_3(v) \)
  + \( u^3 P_4(v) \)

- Let’s make the \( P_i \)s move along curves!
Here’s an Idea

- \( P(u, v) = (1-u)^3 \)  \( P_1(v) \)
  + 3t(1-u)^2  \( P_2(v) \)
  + 3t^2(1-u)  \( P_3(v) \)
  + u^3  \( P_4(v) \)

- Let’s make the \( P_i \)s move along curves!
Here’s an Idea

- \( P(u, v) = (1-u)^3 P_1(v) \)
  + \( 3t(1-u)^2 P_2(v) \)
  + \( 3t^2(1-u) P_3(v) \)
  + \( u^3 P_4(v) \)

- Let’s make the \( P_i \)s move along curves!
Here’s an Idea

- \( P(u, v) = (1-u)^3 \quad P_1(v) \)
  + \( 3t(1-u)^2 \quad P_2(v) \)
  + \( 3t^2(1-u) \quad P_3(v) \)
  + \( u^3 \quad P_4(v) \)

- Let’s make the \( P_i \)s move along curves!

A 2D surface patch!
In the previous, $P_i(v)$ were just some curves
What if we make them Bézier curves?
Tensor Product Bézier Patches

- In the previous, $P_i(v)$ were just some curves.
- What if we make them Bézier curves?
- Each $u=$const. and $v=$const. curve is a Bézier curve!
- Note that the boundary control points (except corners) are NOT interpolated!
Tensor Product Bézier Patches

A bicubic Bézier surface
Tensor Product Bézier Patches

- Note that only the 4 corners are interpolated!

The “Control Mesh”
16 control points
Bicubics, Tensor Product

- \( P(u,v) = B_1(u) \times P_1(v) \)
  + \( B_2(u) \times P_2(v) \)
  + \( B_3(u) \times P_3(v) \)
  + \( B_4(u) \times P_4(v) \)

- \( P_i(v) = B_1(v) \times P_{i,1} \)
  + \( B_2(v) \times P_{i,2} \)
  + \( B_3(v) \times P_{i,3} \)
  + \( B_4(v) \times P_{i,4} \)
Bicubics, Tensor Product

- \( P(u,v) = B_1(u) \times P_1(v) \)
  + \( B_2(u) \times P_2(v) \)
  + \( B_3(u) \times P_3(v) \)
  + \( B_4(u) \times P_4(v) \)

- \( P_i(v) = B_1(v) \times P_{i,1} \)
  + \( B_2(v) \times P_{i,2} \)
  + \( B_3(v) \times P_{i,3} \)
  + \( B_4(v) \times P_{i,4} \)

\[
P(u, v) = \sum_{i=1}^{4} B_i(u) \left[ \sum_{j=1}^{4} P_{i,j} B_j(v) \right]
\
= \sum_{i=1}^{4} \sum_{j=1}^{4} P_{i,j} B_{i,j}(u, v)
\
B_{i,j}(u, v) = B_i(u) B_j(v)
\]
Bicubics, Tensor Product

- \( P(u, v) = B_1(u) \times P_1(v) + B_2(u) \times P_2(v) + B_3(u) \times P_3(v) + B_4(u) \times P_4(v) \)

- \( P_i(v) = B_1(v) \times P_{i,1} + B_2(v) \times P_{i,2} + B_3(v) \times P_{i,3} + B_4(v) \times P_{i,4} \)

\[
P(u, v) = \sum_{i=1}^{4} \sum_{j=1}^{4} P_{i,j} B_{i,j}(u, v)
\]

16 control points \( P_{i,j} \)

16 2D basis functions \( B_{i,j} \)

\[
B_{i,j}(u, v) = B_i(u) B_j(v)
\]
Recap: Tensor Bézier Patches

• Parametric surface $P(u,v)$ is a cubic polynomial of two variables $u$ & $v$
• Defined by $4 \times 4 = 16$ control points $P_{1,1}$, $P_{1,2}$, ..., $P_{4,4}$
• Interpolates 4 corners, approximates others
Tensor Product Bézier Patches

• Defined by 4x4=16 control points P_{1,1}, P_{1,2},..., P_{4,4}

• Basis are product of two Bernstein polynomials: B_1(u)B_1(v); B_1(u)B_2(v);... B_4(u)B_4(v)
Questions?
Tangents and Normals for Patches

- P(u,v) is a **3D point** specified by u, v
- The **partial derivatives** $\partial P / \partial u$ and $\partial P / \partial v$ are 3D vectors
  - Both are tangent to surface at P
Tangents and Normals for Patches

• $P(u,v)$ is a 3D point specified by $u$, $v$
• The partial derivatives $\partial P/\partial u$ and $\partial P/\partial v$ are 3D vectors
  – Both are tangent to surface at $P$
  – Normal is perpendicular to both, i.e.,
    \[ n = (\partial P/\partial u) \times (\partial P/\partial v) \]

$n$ is usually not unit, so must normalize!
Questions?
Recap: Matrix Notation for Curves

- Cubic Bézier in matrix notation

\[
P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}
\]

- Point on curve (2x1 vector)
- "Geometry matrix" of control points \(P_1..P_4\) (2 x 4)
- "Spline matrix" (Bernstein)
- Canonical "power basis"
# Hardcore: Matrix Notation for Patches

- Not required, but convenient!

\[ P^x(u, v) = \begin{pmatrix} P_{1,1}^x & \cdots & P_{1,4}^x \\ \vdots & \ddots & \vdots \\ P_{4,1}^x & \cdots & P_{4,4}^x \end{pmatrix} \begin{pmatrix} B_1(v) \\ \vdots \\ B_4(v) \end{pmatrix} \]

- Column vector of basis functions \( v \)

\[ P(u, v) = \sum_{i=1}^{4} B_i(u) \begin{pmatrix} \sum_{j=1}^{4} P_{i,j} B_j(v) \end{pmatrix} \]

- Row vector of basis functions \( u \)

- 4x4 matrix of x coordinates of the control points

\( x \) coordinate of surface at \((u,v)\)
Curves:

\[ P(t) = G B T(t) \]

Surfaces:

\[ P^x(u, v) = T(u)^T B^T G^x B T(v) \]

- \( T = \) power basis
- \( B = \) spline matrix
- \( G = \) geometry matrix

A separate 4x4 geometry matrix for x, y, z
Super Hardcore: Tensor Notation

- You can stack the $G^x, G^y, G^z$ matrices into a geometry tensor of control points
  - I.e., $G^k_{i,j} = \text{the } k:\text{th coordinate of control point } P_{i,j}$
  - A cube of numbers!

\[
P^k(u, v) = T^l(u) \ B^i_l \ G^k_{i,j} \ B^j_m \ T^m(v)
\]

- “Einstein summation convention”: Repeated indices are summed over (here $l, m, i, j$)
- Definitely not required, but nice!
  - See http://en.wikipedia.org/wiki/Multilinear_algebra
Tensor Product B-Spline Patches

• Bézier and B-Spline curves are both cubics
  – Can change between representations using matrices

• Consequently, you can build tensor product surface patches out of B-Splines just as well
  – Still 4x4 control points for each patch
  – 2D basis functions are pairwise products of B-Spline basis functions
  – Yes, simple!
Tensor Product Spline Patches

• Pros
  – Smooth
  – Defined by reasonably small set of points

• Cons
  – Harder to render (usually converted to triangles)
  – Tricky to ensure continuity at patch boundaries

• Extensions
  – Rational splines: Splines in homogeneous coordinates
  – NURBS: Non-Uniform Rational B-Splines
    • Like curves: ratio of polynomials, non-uniform location of control points, etc.
Utah Teapot: Tensor Bézier Splines

- Designed by Martin Newell
Cool: Displacement Mapping

• Not all surfaces are smooth...
Cool: Displacement Mapping

- Not all surfaces are smooth...
- “Paint” displacements on a smooth surface
  - For example, in the direction of normal
- Tessellate smooth patch into fine grid, then add displacement $D(u,v)$ to vertices
- Heavily used in movies, more and more in games
Displacement Mapping Example

Low-res mesh

ZBrush Render without Displacement-Map

Smooth base surface

ZBrush Render with Displacement-Map

Displaced Surface
• (Remedy Entertainment displacement example not included due to copyright reasons)
Questions?
Subdivision Surfaces

- Start with polygonal mesh
- Subdivide into larger number of polygons, smooth result after each subdivision
  - Lots of ways to do this.
- The limit surface is smooth!
Corner Cutting
Corner Cutting

Slide by Adi Levin
Corner Cutting
Corner Cutting
Corner Cutting
Corner Cutting
Corner Cutting
Corner Cutting
Corner Cutting

The control polygon

A control point

The “limit curve”

The control polygon

Slide by Adi Levin
Corner Cutting

It turns out corner cutting (Chaikin’s Algorithm) produces a quadratic B-Spline curve! (Magic!)

A control point

The “limit curve”

The control polygon
Corner Cutting

It turns out corner cutting (Chaikin’s Algorithm) produces a quadratic B-Spline curve! (Magic!)

(Well, not totally unexpected, remember de Casteljau)
Subdivision Curves and Surfaces

- Idea: cut corners to smooth
- Add points and compute weighted average of neighbors
- Same for surfaces
  - Special case for irregular vertices
    - vertex with more or less than 6 neighbors in a triangle mesh

Warren et al.
Subdivision Curves and Surfaces

• Advantages
  – Arbitrary topology
  – Smooth at boundaries
  – Level of detail, scalable
  – Simple representation
  – Numerical stability, well-behaved meshes
  – Code simplicity

• Little disadvantage:
  – Procedural definition
  – Not parametric, not implicit
  – Tricky at special vertices
Flavors of Subdivision Surfaces

• Catmull-Clark
  – Quads and triangles
  – Generalizes bicubics to arbitrary topology!
• Loop, Butterfly
  – Triangles
• Doo-Sabin, sqrt(3), biquartic...
  – and a whole host of others
• Used everywhere in movie and game modeling!
• See http://www.cs.nyu.edu/~dzorin/sig00course/
Subdivision + Displacement
Subdivision + Displacement

Final Model

Control Mesh
Questions?
Specialized Procedural Definitions

• Surfaces of revolution
  – Rotate given 2D profile curve

• Generalized cylinders
  – Given 2D profile and 3D curve, sweep the profile along the 3D curve

• Assignment 1!
Surface of Revolution

- 2D curve $q(u)$ provides one dimension
  - Note: works also with 3D curve
- Rotation $R(v)$ provides 2nd dimension

$s(u,v) = R(v)q(u)$

where $R$ is a matrix, $q$ a vector, and $s$ is a point on the surface
General Swept Surfaces

- Trace out surface by moving a profile curve along a trajectory.
  - profile curve $q(u)$ provides one dimension
  - trajectory $c(u)$ provides the other
- Surface of revolution can be seen as a special case where trajectory is a circle

$$s(u,v) = M(c(v))q(u)$$

where $M$ is a matrix that depends on the trajectory $c$
General Swept Surfaces

• How do we get $M$?
  – Translation is easy, given by $c(v)$
  – What about orientation?

• Orientation options:
  – Align profile curve with an axis.
  – **Better**: Align profile curve with frame that “follows” the curve

$$s(u,v) = M(c(v))q(u)$$

where $M$ is a matrix that depends on the trajectory $c$
Normals for Swept Surfaces

- Need partial derivatives w.r.t. both $u$ and $v$
  
  \[ n = \left( \frac{\partial P}{\partial u} \right) \times \left( \frac{\partial P}{\partial v} \right) \]
  
  - Remember to normalize!

- One given by tangent of profile curve, the other by the trajectory $s(u,v) = M(c(v))q(u)$
  
  where $M$ is a matrix that depends on the trajectory $c$
Frames on Curves: Frenet Frame

- Frame defined by 1st (tangent), 2nd (curvature) and 3rd (torsion) derivatives of a 3D curve
- Looks like a good idea for swept surfaces...
Frenet: Problem at Inflection!

- Normal flips!
- Bad to define a smooth swept surface

An inflection is a point where curvature changes sign.
Smooth Frames on Curves

- Tangent is assumed reliable
- Build triplet of vectors
  - include tangent
  - orthonormal
  - coherent over the curve
- Idea:
  - use cross product to create orthogonal vectors
  - exploit discretization of curve
  - use previous frame to bootstrap orientation
  - See Assignment 1 instructions!
Questions?
Implicit Surfaces

• Implicit definition: \( f(x,y,z)=0 \)
  e.g. for a sphere: \( x^2+y^2+z^2=R^2 \)
• Often defined as “metaballs” with seed points
• \( f(x,y,z)=f_1(x,y,z)+f_2(x,y,z)+... \)
  – where \( f_i \) depends on distance to a seed point \( P_i \)

From Blinn 1982
Implicit Surfaces

• **Pros:**
  – Can handle weird topology for animation
  – Easy to do sketchy modeling
  – Some data comes this way (medical & scientific data)

• **Cons:**
  – Does not allow us to easily generate a point on the surface

---

6.837 Fall 09 – Lehtinen
Questions?
Point Set Surfaces

• Given only a noisy 3D point cloud (no connectivity), can you define a reasonable surface using only the points?
  – Laser range scans only give you points, so this is potentially useful
Point Set Surfaces

Alexa et al. 2001
Point Set Surfaces

- Modern take on implicit surfaces
- Cool math: Moving Least Squares (MLS), partitions of unity, etc.

- Not required in this class, but nice to know.

Ohtake et al. 2003
Questions?
That’s All for Today

• Further reading
  – Buss, Chapters 7 & 8

• Subvision curves and surfaces
  – http://www.cs.nyu.edu/~dzorin/sig00course/