Today

- General curve and surface representations
- Splines and other polynomial bases
Geometry Representations

- Constructive Solid Geometry (CSG)
- Parametric
  - Polygons
  - Subdivision surfaces
- Implicit Surfaces
- Point-based Surface

Not always clear distinctions
  - i.e. CSG done with implicits
Geometry Representations

Object made by CSG
Converted to polygons
Converted to implicit surface

Geometry Representations

CSG on implicit surfaces
Geometry Representations

Point-based surface descriptions

Images from Subdivision.org

Subdivision surface (different levels of refinement)

Wednesday, October 15, 2008
Geometry Representations

- Various strengths and weaknesses
  - Ease of use for design
  - Ease/speed for rendering
  - Simplicity
  - Smoothness
  - Collision detection
  - Flexibility (in more than one sense)
  - Suitability for simulation
  - many others...

Parametric Representations

Curves: $x = x(u)$, $x \in \mathbb{R}^n$, $u \in \mathbb{R}$

Surfaces: $x = x(u, v)$, $x \in \mathbb{R}^n$, $u, v \in \mathbb{R}$

$\begin{align*}
x &= x(u) \\
u &\in \mathbb{R}^2
\end{align*}$

Volumes: $x = x(u, v, w)$, $x \in \mathbb{R}^n$, $u, v, w \in \mathbb{R}$

$\begin{align*}
x &= x(u) \\
u &\in \mathbb{R}^3
\end{align*}$

and so on...

Note: a vector function is really $n$ scalar functions
Parametric Rep. Non-unique

- Same curve/surface may have multiple formulae

\[ x(u) = [u, u] \]
\[ x(u) = [u^3, u^3] \]

Simple Differential Geometry

- Tangent to curve
  \[ t(u) = \frac{\partial x}{\partial u} \]

- Tangents to surface
  \[ t_u(u, v) = \frac{\partial x}{\partial u_{u,v}} \]
  \[ t_v(u, v) = \frac{\partial x}{\partial v_{u,v}} \]

- Normal of surface
  \[ \hat{n} = \frac{t_u \times t_v}{||t_u \times t_v||} \]

- Also: curvature, curve normals, curve bi-normal, others...
- Degeneracies: \( \partial x/\partial u = 0 \) or \( t_u \times t_v = 0 \)

Wednesday, October 15, 2008
Discretization

- Arbitrary curves have an uncountable number of parameters

\[ x(u) = \sum_{i=0}^{\infty} c_i \phi_i(u) \]

i.e. specify function value at all points on real number line

Discretization

- Arbitrary curves have an uncountable number of parameters
- Pick complete set of basis functions
  - Polynomials, Fourier series, etc.
- Truncate set at some reasonable point
  \[ x(u) = \sum_{i=0}^{3} c_i \phi_i(u) = \sum_{i=0}^{3} c_i u^i \]
- Function represented by the vector (list) of \( c_i \)
- The \( c_i \) may themselves be vectors
  \[ x(u) = \sum_{i=0}^{3} c_i \phi_i(u) \]
Polynomial Basis

- **Power Basis**
  \[ x(u) = \sum_{i=0}^{d} c_i u^i \]
  \[ x(u) = C \cdot \mathbf{P}^d \]
  \[ C = [c_0, c_1, c_2, \ldots, c_d] \]
  \[ \mathbf{P}^d = [1, u, u^2, \ldots, u^d] \]

  The elements of \( \mathbf{P}^d \) are linearly independent i.e. no good approximation
  \[ u^k \neq \sum_{i=k}^{d} c_i u^i \]

  Skipping something would lead to bad results... odd stiffness

---

Specifying a Curve

Given desired values (constraints) how do we determine the coefficients for cubic power basis?

For now, assume \( u_0 = 0 \), \( u_1 = 1 \)
Specifying a Curve

Given desired values (constraints) how do we determine the coefficients for cubic power basis?

\[ x(0) = c_0 = x_0 \]
\[ x(1) = c_1 = x_1 \]
\[ x'(0) = c_1 = x'_0 \]
\[ x'(1) = \sum_i i c_i = x'_1 \]

\[ \begin{bmatrix} x_0 \\ x_1 \\ x'_0 \\ x'_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \]

\[ p = B \cdot c \]
Specifying a Curve

Given desired values (constraints) how do we determine the coefficients for cubic power basis?

\[ c = \beta_u \cdot p \]

\[ \beta_u = B^{-1} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
-3 & 3 & -2 & 1 \\
2 & -2 & 1 & 1
\end{bmatrix} \]

\[ x(u) = P^3 \cdot c = P^3 \beta_u \cdot p \]
Specifying a Curve

Given desired values (constraints) how do we determine the coefficients for cubic power basis?

\[ c = \beta_n \cdot p \]

\[ x(u) = \begin{bmatrix} 1 + 0u - 3u^2 + 2u^3 \\ 0 + 0u + 3u^2 - 2u^3 \\ 0 + 1u - 2u^2 + 1u^3 \\ 0 + 0u - 1u^2 + 1u^3 \end{bmatrix} p \]

\[ x(u) = \sum_{i=0}^{3} p_i b_i(u) \]

Hermite basis functions

Wednesday, October 15, 2008
Hermite Basis

- Specify curve by
  - Endpoint values
  - Endpoint tangents (derivatives)
- Parameter interval is arbitrary (most times)
  - Don’t need to recompute basis functions
- These are cubic Hermite
  - Could do construction for any odd degree
  - $(d - 1)/2$ derivatives at end points

Cubic Bézier

- Similar to Hermite, but specify tangents indirectly

\[
\begin{align*}
x_0 &= p_0 \\
x_1 &= p_3 \\
x_0' &= 3(p_1 - p_0) \\
x_1' &= 3(p_3 - p_2)
\end{align*}
\]

Note: all the control points are points in space, no tangents.
Cubic Bézier

- Similar to Hermite, but specify tangents indirectly

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 \\
0 & 1 & 2 & 3
\end{bmatrix}
\begin{bmatrix}
1 \\
0 \\
0 \\
0
\end{bmatrix}
= \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
-3 & 3 & 0 & 0 \\
0 & 0 & -3 & 3
\end{bmatrix}
\begin{bmatrix}
x_0 \\
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\]

\[c = \begin{bmatrix}
1 & 0 & 0 & 0 \\
-3 & 3 & 0 & 0 \\
3 & -6 & 3 & 0 \\
-1 & 3 & -3 & 1
\end{bmatrix}
\begin{bmatrix}
x_0 \\
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\]

\[c = \beta_z p\]

Cubic Bézier

- Plot of Bézier basis functions
Changing Bases

- Power basis, Hermite, and Bézier all are still just cubic polynomials
- The three basis sets all span the same space
- Like different axes in $\mathbb{R}^3 \times \mathbb{R}^4$

- Changing basis

$$
c = \beta_z p_z$$
$$c = \beta_H p_H$$

Useful Properties of a Basis

- Convex Hull
  - All points on curve inside convex hull of control points
  - $\sum_i b_i(u) = 1 \quad b_i(u) \geq 0 \quad \forall u \in \Omega$
  - Bézier basis has convex hull property

Wednesday, October 15, 2008
Useful Properties of a Basis

- Invariance under class of transforms
  - Transforming curve is same as transforming control points
    - $x(u) = \sum p_i b_i(u) \Leftrightarrow T x(u) = \sum (T p_i) b_i(u)$
  - Bézier basis invariant for affine transforms
  - Bézier basis NOT invariant for perspective transforms
    - NURBS are though...

Useful Properties of a Basis

- Local support
  - Changing one control point has limited impact on entire curve
  - Nice subdivision rules
  - Orthogonality ($\int b_i(u) b_j(u) du = \delta_{ij}$)
  - Fast evaluation scheme
  - Interpolation -vs- approximation

Wednesday, October 15, 2008
DeCasteljau Evaluation

● A geometric evaluation scheme for Bézier

error...

$u = 0$

$u = 0.25$

$u = 0.5$

$u = 0.75$

$u = 1$

Adaptive Tessellation

● Midpoint test subdivision

● Possible problem
  ● Simple solution if curve basis has convex hull property

If curve inside convex hull and the convex hull is nearly flat: curve is nearly flat and can be drawn as straight line

Better: draw convex hull
Works for Bézier because the ends are interpolated
Bézier Subdivision

- Form control polygon for half of curve by evaluating at $u=0.5$

Wednesday, October 15, 2008
Bézier Subdivision

- Form control polygon for half of curve by evaluating at $u=0.5$

Repeated subdivision makes smaller/flatter segments

Also works for surfaces...

We'll extend this idea later on...

Joining

$c^0 \Leftrightarrow b = b$
$c^1 \Leftrightarrow b - a = c - b$
$g^1 \Leftrightarrow \frac{b - a}{||b - a||} = \frac{c - b}{||c - b||}$

If you change $a$, $b$, or $c$ you must change the others

But if you change $a$, $b$, or $c$ you do not have to change beyond those three. *LOCAL SUPPORT*
**“Hump” Functions**

- Constraints at joining can be built in to make new basis

![Diagram of hump functions](image)

---

**Tensor-Product Surfaces**

- Surface is a curve swept through space
- Replace control points of curve with other curves

\[
x(u, v) = \sum_i p_i b_i(u) \quad g_i(v) = \sum_j p_j b_j(v)
\]

\[
x(u, v) = \sum_{ij} p_{ij} b_i(u) b_j(v) = b_i(u) b_j(v)
\]

\[
x(u, v) = \sum_{ij} p_{ij} b_{ij}(u, v)
\]
Tensor-Product Surfaces

Hermite Surface Bases

Plus symmetries...

Wednesday, October 15, 2008
Hermite Surface Hump Functions

Plus symmetries...

Bézier Surface Patch
Adaptive Tessellation

- Given surface patch
  - If close to flat: draw it
  - Else subdivide 4 ways

Adaptive Tessellation

- Avoid cracking

Passes flatness test  Fails flatness test
Adaptive Tessellation

- Avoid cracking

Cracks may be okay in some contexts...
Adaptive Tessellation

- Avoid cracking

Test interior and boundary of patch
Split boundary based on boundary test
Table of polygon patterns
May wish to avoid “slivers”

Adaptive Tessellation

- Triangle Based Method (no cracks)
Adaptive Tessellation

- Triangle Based Method (no cracks)

\[ \frac{u_1 + u_2}{2} \]

\[ \frac{x_1 + x_2}{2} \]

\[ B\left(\frac{u_1 + u_2}{2}\right) \]

\[ B\left(\frac{x_1 + x_2}{2}\right) \]

Wednesday, October 15, 2008
Adaptive Tessellation

- Triangle Based Method (no cracks)

\[ ||B\left(\frac{u_1 + u_2}{2}\right) - \frac{x_1 + x_2}{2}|| < \tau ? \]

Center test tends to generate slivers. Often better to leave it out.
Adaptive Tessellation

Without center test

With center test

Second row shows typical error of swapping tests.

Wednesday, October 15, 2008
Adaptive Tessellation

Visible artifacts from cracks.

Apollo Ellis, CS184 S08