Announcements

• CAPE evaluation this Thursday at beginning of class

• This Friday 2-5pm: grading homework assignment #6
StarCAVE Tour

- Location: Atkinson Hall, 1st floor
- 18 Dell XPS PCs with Quad Core Intel CPUs
- CentOS 5.2 Linux
- Dual Nvidia Quadro 5600 graphics cards per node
- 34 JVC HD2k projectors (1920x1080 pixels): ~34 megapixels per eye
- Passive stereo with circular polarization filters
- 15 screens, ~8 x 4 feet each
- Floor projection
- Optical, wireless tracking system
- Software: COVISE
- Programming Language: C++

Tour Date:
- Thursday, Nov 19, 4:00-5:30pm
Lecture Overview

Curves
- Piecewise cubic Bézier curves

Surfaces
- Bi-linear patch
- Bi-cubic Bézier patch
- Advanced surface modeling
More control points

- Cubic Bézier curve limited to 4 control points
  - Cubic curve can only have one inflection (point where curve changes direction of bending)
  - Need more control points for more complex curves
- \( k-1 \) order Bézier curve with \( k \) control points
  - Hard to control and hard to work with
    - Intermediate points don’t have obvious effect on shape
    - Changing any control point changes the whole curve
    - Want *local support*: each control point only influences nearby portion of curve
Piecewise curves

- Sequence of simple (low-order) curves, end-to-end
  - Known as a *piecewise polynomial curve*
- Sequence of line segments
  - *Piecewise linear* curve

- Sequence of cubic curve segments
  - *Piecewise cubic* curve (here piecewise Bézier)
Continuity

- Want smooth curves
- $C^0$ continuity
  - No gaps
  - Segments meet at the endpoints
- $C^1$ continuity: first derivative is well defined
  - No corners
  - Tangents/normals are $C^0$ continuous (no jumps)
- $C^2$ continuity: second derivative is well defined
  - Tangents/normals are $C^1$ continuous
  - Important for high quality reflections
Global parameterization

- Given $N$ curve segments $x_0(t), x_1(t), \ldots, x_{N-1}(t)$
- Each is parameterized for $t$ from 0 to 1
- Define a piecewise curve
  - Global parameter $u$ from 0 to $N$
    
    $$
    x(u) = \begin{cases}
    x_0(u), & 0 \leq u \leq 1 \\
    x_1(u - 1), & 1 \leq u \leq 2 \\
    \vdots & \vdots \\
    x_{N-1}(u - (N - 1)), & N - 1 \leq u \leq N \\
    \end{cases}
    $$

    $$
    x(u) = x_i(u - i), \text{ where } i = \lceil u \rceil \quad (\text{and } x(N) = x_{N-1}(1))
    $$

- Alternate: $u$ also goes from 0 to 1

    $$
    x(u) = x_i(Nu - i), \text{ where } i = \lfloor Nu \rfloor
    $$
Given \( N+1 \) points \( p_0, p_1, \ldots, p_N \)

Define curve

\[
x(u) = \text{Lerp}(u - i, p_i, p_{i+1}), \quad i \leq u \leq i + 1
\]

\[
= (1 - u + i)p_i + (u - i)p_{i+1}, \quad i = \lfloor u \rfloor
\]

\( N+1 \) points define \( N \) linear segments

\( x(i) = p_i \)

\( C^0 \) continuous by construction

\( C^1 \) at \( p_i \) when \( p_i - p_{i-1} = p_{i+1} - p_i \)
Piecewise Bézier curve

- Given $3N + 1$ points $p_0, p_1, \ldots, p_{3N}$
- Define $N$ Bézier segments:

\[
x_0(t) = B_0(t)p_0 + B_1(t)p_1 + B_2(t)p_2 + B_3(t)p_3
\]
\[
x_1(t) = B_0(t)p_3 + B_1(t)p_4 + B_2(t)p_5 + B_3(t)p_6
\]
\[\vdots\]
\[
x_{N-1}(t) = B_0(t)p_{3N-3} + B_1(t)p_{3N-2} + B_2(t)p_{3N-1} + B_3(t)p_{3N}
\]
Piecewise Bézier curve

- Parameter in $0 \leq u \leq 3N$

$$x(u) = \begin{cases} x_0(\frac{1}{3}u), & 0 \leq u \leq 3 \\ x_1(\frac{1}{3}u - 1), & 3 \leq u \leq 6 \\ \vdots & \vdots \\ x_{N-1}(\frac{1}{3}u - (N - 1)), & 3N - 3 \leq u \leq 3N \end{cases}$$

$$x(u) = x_i\left(\frac{1}{3}u - i\right), \text{ where } i = \left\lfloor \frac{1}{3}u \right\rfloor$$
Piecewise Bézier curve

- $3N+1$ points define $N$ Bézier segments
- $x(3i) = p_{3i}$
- $C^0$ continuous by construction
- $C^1$ continuous at $p_{3i}$ when $p_{3i} - p_{3i-1} = p_{3i+1} - p_{3i}$
- $C^2$ is harder to get

![Diagram showing Bézier segments and continuity conditions.](image)
Piecewise Bézier curves

- Used often in 2D drawing programs

- Inconveniences
  - Must have 4 or 7 or 10 or 13 or ... (1 plus a multiple of 3) control points
  - Some points interpolate, others approximate
  - Need to impose constraints on control points to obtain $C^1$ continuity
  - $C^2$ continuity more difficult

- Solutions
  - User interface using “Bézier handles”
  - Generalization to B-splines
Bézier handles

- Segment end points (interpolating) presented as curve control points
- Midpoints (approximating points) presented as “handles”
- Can have option to enforce $C^1$ continuity
Rational curves

• Weight causes point to “pull” more (or less)

• With proper points & weights, can do circles
NURBS

- **Non uniform rational B-splines**
- Generalization of Bézier curves
  - Easier to guarantee smoothness of curve
  - Can represent conic sections (circles, ellipses)
Lecture Overview

Curves
- Piecewise cubic Bézier curves

Surfaces
- Bi-linear patch
- Bi-cubic Bézier patch
- Advanced surface modeling
Curved surfaces

Curves

- Described by a 1D series of control points
- A function $x(t)$
- Segments joined together to form a longer curve

Surfaces

- Described by a 2D mesh of control points
- Parameters have two dimensions (two dimensional parameter domain)
- A function $x(u,v)$
- Patches joined together to form a bigger surface
Parametric surface patch

- $\mathbf{x}(u,v)$ describes a point in space for any given $(u,v)$ pair
  - $u,v$ each range from 0 to 1
• $\mathbf{x}(u,v)$ describes a point in space for any given $(u,v)$ pair
  - $u,v$ each range from 0 to 1

• Parametric curves
  - For fixed $u_0$, have a $v$ curve $\mathbf{x}(u_0,v)$
  - For fixed $v_0$, have a $u$ curve $\mathbf{x}(u,v_0)$
  - For any point on the surface, there are a pair of parametric curves that go through point
Tangents

• The tangent to a parametric curve is also tangent to the surface
• For any point on the surface, there are a pair of (parametric) tangent vectors
• Note: not necessarily perpendicular to each other
Tangents

• Notation:
  • The tangent along a $u$ curve, AKA the tangent in the $u$ direction, is written as:
    \[ \frac{\partial x}{\partial u}(u,v) \text{ or } \frac{\partial}{\partial u} x(u,v) \text{ or } x_u(u,v) \]
  • The tangent along a $v$ curve, AKA the tangent in the $v$ direction, is written as:
    \[ \frac{\partial x}{\partial v}(u,v) \text{ or } \frac{\partial}{\partial v} x(u,v) \text{ or } x_v(u,v) \]
  • Note that each of these is a vector-valued function:
    • At each point $x(u,v)$ on the surface, we have tangent vectors $\frac{\partial}{\partial u} x(u,v)$ and $\frac{\partial}{\partial v} x(u,v)$
Surface Normal

- Cross product of the two tangent vectors
- Order matters!

\[
\vec{n}(u, v) = \frac{\partial x}{\partial u}(u, v) \times \frac{\partial x}{\partial v}(u, v)
\]

Typically we are interested in the unit normal, so we need to normalize

\[
\vec{n}^*(u, v) = \frac{\vec{n}(u, v)}{|\vec{n}(u, v)|}
\]
Bilinear patch

- Control mesh with four points \( p_0, p_1, p_2, p_3 \)
- Compute \( x(u, v) \) using a two-step construction scheme
For a given value of $u$, evaluate the linear curves on the two $u$-direction edges.

Use the same value $u$ for both:

$$q_0 = \text{Lerp}(u, p_0, p_1)$$  
$$q_1 = \text{Lerp}(u, p_2, p_3)$$
Bilinear patch (step 2)

- Consider that $q_0$, $q_1$ define a line segment
- Evaluate it using $v$ to get $x$

$$x = \text{Lerp}(v, q_0, q_1)$$
Bilinear patch

- Combining the steps, we get the full formula

\[ x(u, v) = \text{Lerp}(v, \text{Lerp}(u, p_0, p_1), \text{Lerp}(u, p_2, p_3)) \]
Bilinear patch

- Try the other order
- Evaluate first in the $v$ direction

$$r_0 = Lerp(v, p_0, p_2) \quad r_1 = Lerp(v, p_1, p_3)$$
Bilinear patch

- Consider that $r_0, r_1$ define a line segment
- Evaluate it using $u$ to get $x$

$$x = \text{Lerp}(u, r_0, r_1)$$
Bilinear patch

- The full formula for the $v$ direction first:

$$x(u, v) = \text{Lerp}(u, \text{Lerp}(v, p_0, p_2), \text{Lerp}(v, p_1, p_3))$$
Bilinear patch

- It works out the same either way!

\[
x(u, v) = \text{Lerp}(v, \text{Lerp}(u, p_0, p_1), \text{Lerp}(u, p_2, p_3))
\]

\[
x(u, v) = \text{Lerp}(u, \text{Lerp}(v, p_0, p_2), \text{Lerp}(v, p_1, p_3))
\]
Bilinear patch

- Visualization
Bilinear patches

- Weighted sum of control points

\[ x(u, v) = (1-u)(1-v)p_0 + u(1-v)p_1 + (1-u)v p_2 + uv p_3 \]

- Bilinear polynomial

\[ x(u, v) = (p_0 - p_1 - p_2 + p_3)uv + (p_1 - p_0)u + (p_2 - p_0)v + p_0 \]

- Matrix form exists, too
Properties

- Interpolates the control points
- The boundaries are straight line segments
- If all 4 points of the control mesh are co-planar, the patch is flat
- If the points are not co-planar, we get a curved surface
  - saddle shape (hyperbolic paraboloid)
- *The parametric curves are all straight line segments!*
  - a (doubly) *ruled surface*: has (two) straight lines through every point

- Not terribly useful as a modeling primitive
Lecture Overview

Curves
- Piecewise cubic Bézier curves

Surfaces
- Bi-linear patch
- Bi-cubic Bézier patch
- Advanced surface modeling
Bicubic Bézier patch

- Grid of 4x4 control points, \( p_0 \) through \( p_{15} \)
- Four rows of control points define Bézier curves along \( u \)
  \( p_0, p_1, p_2, p_3; p_4, p_5, p_6, p_7; p_8, p_9, p_{10}, p_{11}; p_{12}, p_{13}, p_{14}, p_{15} \)
- Four columns define Bézier curves along \( v \)
  \( p_0, p_4, p_8, p_{12}; p_1, p_5, p_9, p_{13}; p_2, p_6, p_{10}, p_{14}; p_3, p_7, p_{11}, p_{15} \)
Bézier patch (step 1)

- Evaluate four $u$-direction Bézier curves at $u$
- Get points $q_0$ ... $q_3$

\[
q_0 = \text{Bez}(u, p_0, p_1, p_2, p_3) \\
q_1 = \text{Bez}(u, p_4, p_5, p_6, p_7) \\
q_2 = \text{Bez}(u, p_8, p_9, p_{10}, p_{11}) \\
q_3 = \text{Bez}(u, p_{12}, p_{13}, p_{14}, p_{15})
\]
Bézier patch (step 2)

- Points $q_0 \ldots q_3$ define a Bézier curve
- Evaluate it at $v$

$$x(u,v) = \text{Bez}(v, q_0, q_1, q_2, q_3)$$
Bézier patch

- Same result in either order (evaluate $u$ before $v$ or vice versa)

\[
q_0 = Bez(u, p_0, p_1, p_2, p_3) \\
q_1 = Bez(u, p_4, p_5, p_6, p_7) \\
q_2 = Bez(u, p_8, p_9, p_{10}, p_{11}) \iff r_2 = Bez(v, p_2, p_6, p_{10}, p_{14}) \\
q_3 = Bez(u, p_{12}, p_{13}, p_{14}, p_{15}) \iff r_3 = Bez(v, p_3, p_7, p_{11}, p_{15}) \\
x(u, v) = Bez(v, q_0, q_1, q_2, q_3) = Bez(u, r_0, r_1, r_2, r_3)
\]
Properties

- Convex hull: any point on the surface will fall within the convex hull of the control points
- Interpolates 4 corner points
- Approximates other 12 points, which act as “handles”
- The boundaries of the patch are the Bézier curves defined by the points on the mesh edges
- The parametric curves are all Bézier curves
Tangents of Bézier patch

- Remember parametric curves $\mathbf{x}(u, v_0), \mathbf{x}(u_0, v)$ where $v_0, u_0$ is fixed
- Tangents to surface = tangents to parametric curves
- Tangents are partial derivatives of $\mathbf{x}(u, v)$
- Normal is cross product of the tangents
Tangents of Bézier patch

\[ q_0 = \text{Bez}(u, p_0, p_1, p_2, p_3) \]
\[ q_1 = \text{Bez}(u, p_4, p_5, p_6, p_7) \]
\[ q_2 = \text{Bez}(u, p_8, p_9, p_{10}, p_{11}) \]
\[ q_3 = \text{Bez}(u, p_{12}, p_{13}, p_{14}, p_{15}) \]

\[ \frac{\partial x}{\partial v}(u, v) = \text{Bez}'(v, q_0, q_1, q_2, q_3) \]

\[ r_0 = \text{Bez}(v, p_0, p_4, p_8, p_{12}) \]
\[ r_1 = \text{Bez}(v, p_1, p_5, p_9, p_{13}) \]
\[ r_2 = \text{Bez}(v, p_2, p_6, p_{10}, p_{14}) \]
\[ r_3 = \text{Bez}(v, p_3, p_7, p_{11}, p_{15}) \]

\[ \frac{\partial x}{\partial u}(u, v) = \text{Bez}'(u, r_0, r_1, r_2, r_3) \]
Tessellating a Bézier patch

- Uniform tessellation is most straightforward
  - Evaluate points on a grid of \( u, v \) coordinates
  - Compute tangents at each point, take cross product to get per-vertex normal
  - Draw triangle strips (several choices of direction)

- Adaptive tessellation/recursive subdivision
  - Potential for “cracks” if patches on opposite sides of an edge divide differently
  - Tricky to get right, but can be done
Piecewise Bézier surface

- Lay out grid of adjacent meshes of control points
- For $C^0$ continuity, must share points on the edge
  - Each edge of a Bézier patch is a Bézier curve based only on the edge mesh points
  - So if adjacent meshes share edge points, the patches will line up exactly
- But we have a crease…
C¹ continuity

- We want the parametric curves that cross each edge to have C¹ continuity
  - So the handles must be equal-and-opposite across the edge:

http://www.spiritone.com/~english/cyclopedia/patches.html
Modeling with Bézier patches

- Original Utah teapot specified as Bézier Patches
Lecture Overview

Curves
• Piecewise cubic Bézier curves

Surfaces
• Bi-linear patch
• Bi-cubic Bézier patch
• Advanced surface modeling
Advanced surface modeling

- B-spline/NURBS patches
- For the same reason as using B-spline/NURBS curves
  - More flexible (can model spheres)
  - Better mathematical properties, continuity
Advanced surface modeling

- Trim curves: cut away part of surface
  - Implement as part of tessellation/rendering
Modeling headaches

- Original Teapot is not “watertight”
  - spout & handle intersect with body
  - no bottom
  - hole in spout
  - gap between lid and body
Modeling headaches

NURBS surfaces are flexible

- Conic sections
- Can blend, merge, trim...

...but

- Any surface will be made of quadrilateral patches (quadrilateral topology)
Quadrilateral topology

Makes it hard to

• join or abut curved pieces
• build surfaces with awkward topology or structure
Subdivision surfaces

- Arbitrary mesh of control points, not quadrilateral topology
  - No global $u,v$ parameters
- Can make surfaces with arbitrary topology or connectivity
- Work by recursively subdividing mesh faces
  - Per-vertex annotation for weights, corners, creases
- Used in particular for character animation
  - One surface rather than collection of patches
  - Can deform geometry without creating cracks
Next Lecture

- CAPE
- Advanced shader programming