Announcements

• Midterm
  - Exams returned today
  - Scores are on-line at GradeSource

• CAPE evaluation today
Additional Lab Hours

- Additional lab hours with Jurgen, first come first serve, no grading priority
- This week and the week after Thanksgiving in lab 250:
  - Wed 3:30-5pm
  - Thu 3:30-5pm
  - Fri 4-5pm
- Lab hours with Robert as before:
  - Mon 2-5pm: homework grading
  - Wed 2-4pm: homework introduction
  - Fri 1-4pm: late grading, homework support
Final Project

- Description is on-line
- Description due Monday Dec 1 (email)
- Project due Friday, Dec 5 (lab demo)
- Can be done in teams of two
StarCAVE Tour

- Location: Atkinson Hall, 1st floor
- Computers: 18 Dell XPS PCs with Quad Core Intel CPUs
- OS: CentOS Linux
- Graphics cards: 2 Nvidia Quadro 5600 per node
- Projectors: 34 JVC HD2k (1920x1080 pixels), ~34 megapixels per eye
- Stereo: passive with circular polarization filters
- 15 screens, ~8 x 4 feet each
- Floor projection
- Optical, wireless tracking system
- Visualization software: COVISE, Electro, CAVElib
- Programming Language: C++

Tour options:
- Mon 11/24, 1pm
- Mon 11/24, 5pm
- Tue 11/25, 11am
- Wed 12/3, 1pm
- Wed 12/10, 4pm
Piecewise Cubic Bézier curve

- Parameter in \( 0 \leq u \leq 3N \)

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

\[
x(u) = x_i \left( \frac{1}{3} u - i \right), \text{ where } i = \left\lfloor \frac{1}{3} u \right\rfloor
\]
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)
Today

Surfaces

- Bilinear patch
- Bicubic 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

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

2D parameter domain
**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

- **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:
    \[ \frac{\partial \mathbf{x}}{\partial u}(u,v) \text{ or } \frac{\partial}{\partial u} \mathbf{x}(u,v) \text{ or } \mathbf{x}_u(u,v) \]
  • The tangent along a $v$ curve, AKA the tangent in the $v$ direction, is written as:
    \[ \frac{\partial \mathbf{x}}{\partial v}(u,v) \text{ or } \frac{\partial}{\partial v} \mathbf{x}(u,v) \text{ or } \mathbf{x}_v(u,v) \]

• Note that each of these is a vector-valued function:
  • At each point $\mathbf{x}(u,v)$ on the surface, we have tangent vectors $\frac{\partial}{\partial u} \mathbf{x}(u,v)$ and $\frac{\partial}{\partial v} \mathbf{x}(u,v)$
**Surface Normal**

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

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

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

\[ \mathbf{n}^*(u,v) = \frac{\mathbf{n}(u,v)}{\| \mathbf{n}(u,v) \|} \]
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
Bilinear patch (step 1)

- For a given value of $u$, evaluate the linear curves on the two $u$-direction edges
- Use the same value $u$ for both:

$$q_0 = \text{Lerp}(u, p_0, p_1) \quad q_1 = \text{Lerp}(u, p_2, p_3)$$
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

$$r_0 = \text{Lerp}(v, p_0, p_2) \quad r_1 = \text{Lerp}(v, p_1, p_3)$$
Consider that $\mathbf{r}_0$, $\mathbf{r}_1$ define a line segment. Evaluate it using $u$ to get $\mathbf{x}$:

$$\mathbf{x} = \text{Lerp}(u, \mathbf{r}_0, \mathbf{r}_1)$$
Bilinear patch

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

$$x(u, v) = \text{Lerp}(u, \text{Lerp}(v, \mathbf{p}_0, \mathbf{p}_2), \text{Lerp}(v, \mathbf{p}_1, \mathbf{p}_3))$$
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
Today

Surfaces

- Bilinear patch
- Bicubic Bézier patch
- Advanced surface modeling
Bicubic Bézier patch

- Grid of 4x4 control points, \( \mathbf{p}_0 \) through \( \mathbf{p}_{15} \)
- Four rows of control points define Bézier curves along \( u \)
  \( \mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3; \, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7; \, \mathbf{p}_8, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_{11}; \, \mathbf{p}_{12}, \mathbf{p}_{13}, \mathbf{p}_{14}, \mathbf{p}_{15} \)
- Four columns define Bézier curves along \( v \)
  \( \mathbf{p}_0, \mathbf{p}_4, \mathbf{p}_8, \mathbf{p}_{12}; \, \mathbf{p}_1, \mathbf{p}_5, \mathbf{p}_9, \mathbf{p}_{13}; \, \mathbf{p}_2, \mathbf{p}_6, \mathbf{p}_{10}, \mathbf{p}_{14}; \, \mathbf{p}_3, \mathbf{p}_7, \mathbf{p}_{11}, \mathbf{p}_{15} \)
Bézier patch (step 1)

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

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

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

$$x(u, v) = Bez(v, q_0, q_1, q_2, q_3)$$
Bézier patch

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

\[
\begin{align*}
q_0 &= \text{Bez}(u, p_0, p_1, p_2, p_3) & r_0 &= \text{Bez}(v, p_0, p_4, p_8, p_{12}) \\
q_1 &= \text{Bez}(u, p_4, p_5, p_6, p_7) & r_1 &= \text{Bez}(v, p_1, p_5, p_9, p_{13}) \\
q_2 &= \text{Bez}(u, p_8, p_9, p_{10}, p_{11}) & r_2 &= \text{Bez}(v, p_2, p_6, p_{10}, p_{14}) \\
q_3 &= \text{Bez}(u, p_{12}, p_{13}, p_{14}, p_{15}) & r_3 &= \text{Bez}(v, p_3, p_{7}, p_{11}, p_{15})
\end{align*}
\]

\[
\begin{align*}
x(u, v) &= \text{Bez}(v, q_0, q_1, q_2, q_3) & x(u, v) &= \text{Bez}(u, r_0, r_1, r_2, r_3)
\end{align*}
\]
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 \( x(u, v_0), x(u_0, v) \) where \( v_0, u_0 \) is fixed
- Tangents to surface = tangents to parametric curves
- Tangents are partial derivatives of \( x(u, v) \)
- 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…

Grid of control points

Piecewise Bézier surface
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
Today

Surfaces

- Bilinear patch
- Bicubic 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 time

- Advanced shader programming