

CSE 167:
Introduction to Computer Graphics
Lecture #13: Surface Patches

Jürgen P. Schulze, Ph.D.
University of California, San Diego
Fall Quarter 2019

Announcements

Overview

- ▶ **Bi-linear patch**
- ▶ Bi-cubic Bézier patch

Curved Surfaces

Curves

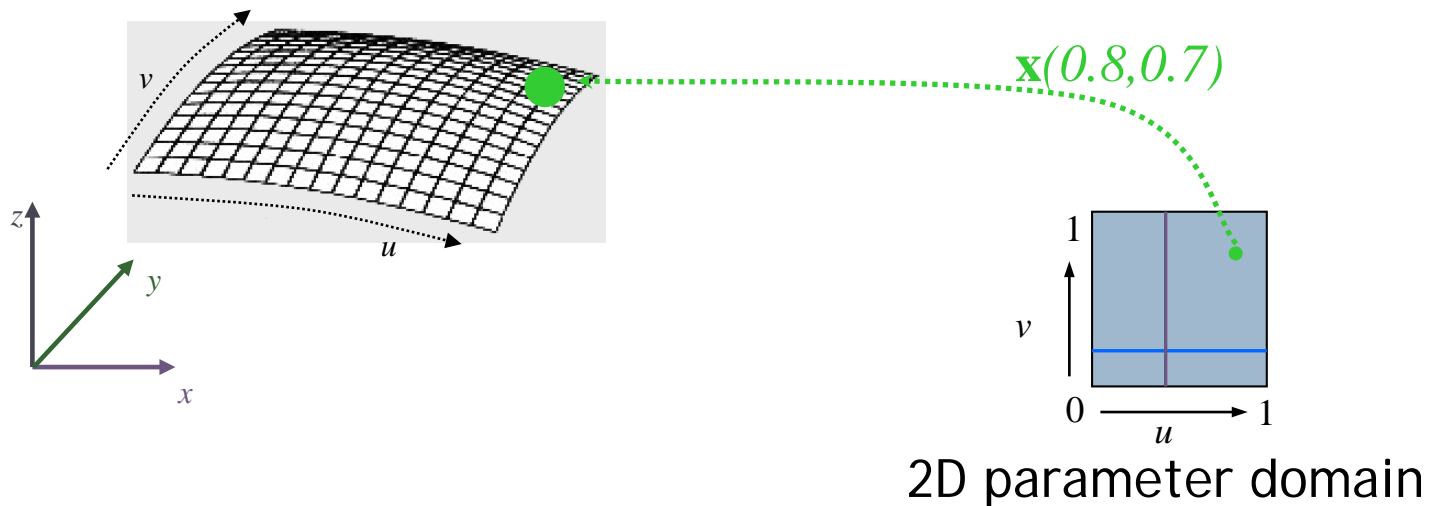
- ▶ Described by a 1D series of control points
- ▶ A function $\mathbf{x}(t)$
- ▶ Segments joined together to form a longer curve

Surfaces

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

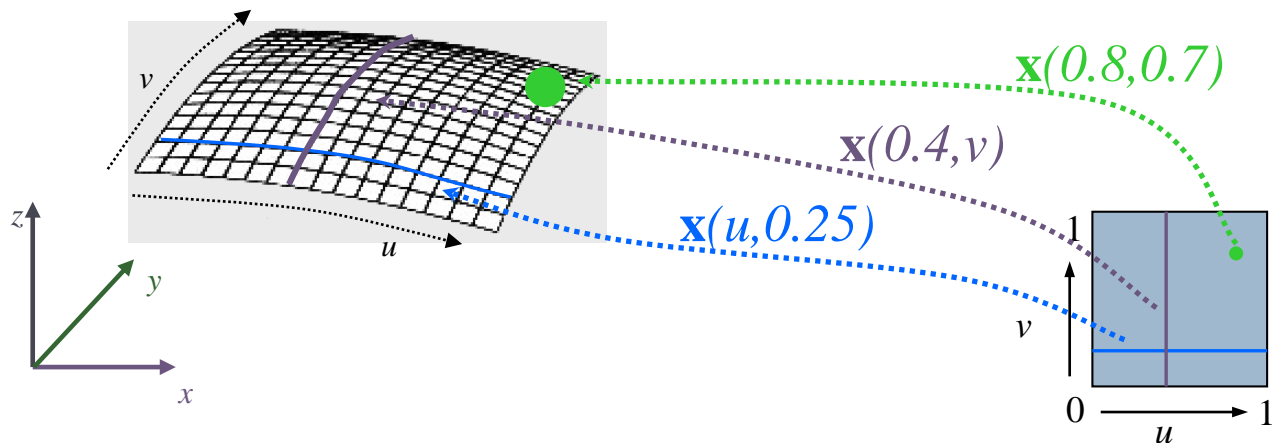
Parametric Surface Patch

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



Parametric Surface Patch

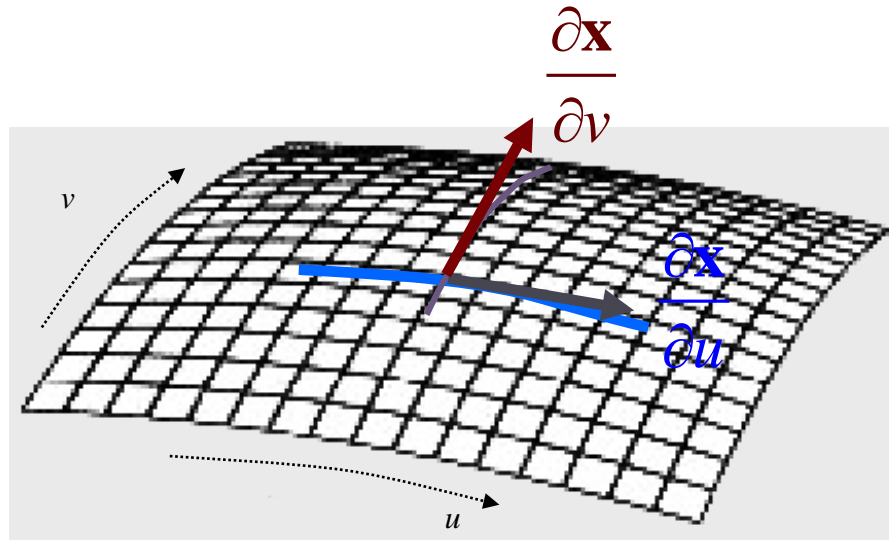
- ▶ $\mathbf{x}(u,v)$ describes a point in space for any given (u,v) pair
 - ▶ u,v each range from 0 to 1



- ▶ 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 through that point

Tangents

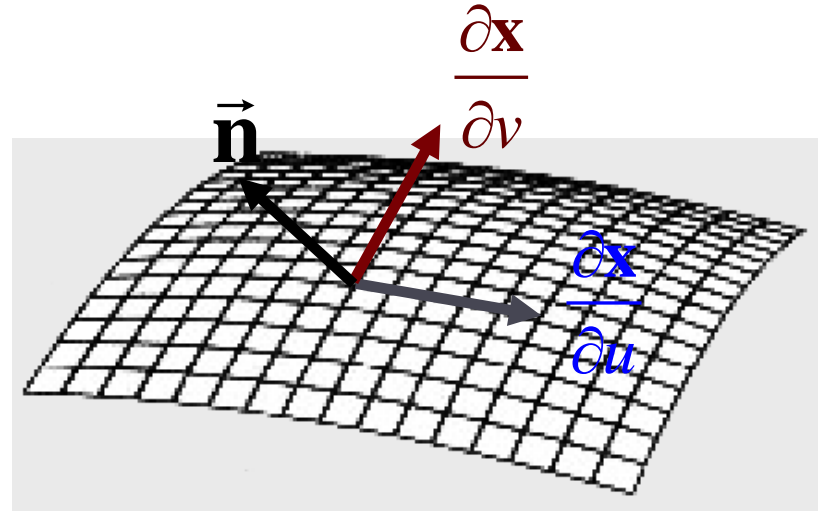
- ▶ The tangent to a parametric curve is also tangent to the surface
- ▶ For any point on the surface, there are a pair of (parametric) tangent vectors
- ▶ Note: these vectors are not necessarily perpendicular to each other



Surface Normal

- ▶ Normal is cross product of the two tangent vectors
- ▶ Order of vectors matters!

$$\vec{\mathbf{n}}(u, v) = \frac{\partial \mathbf{x}}{\partial u}(u, v) \times \frac{\partial \mathbf{x}}{\partial v}(u, v)$$



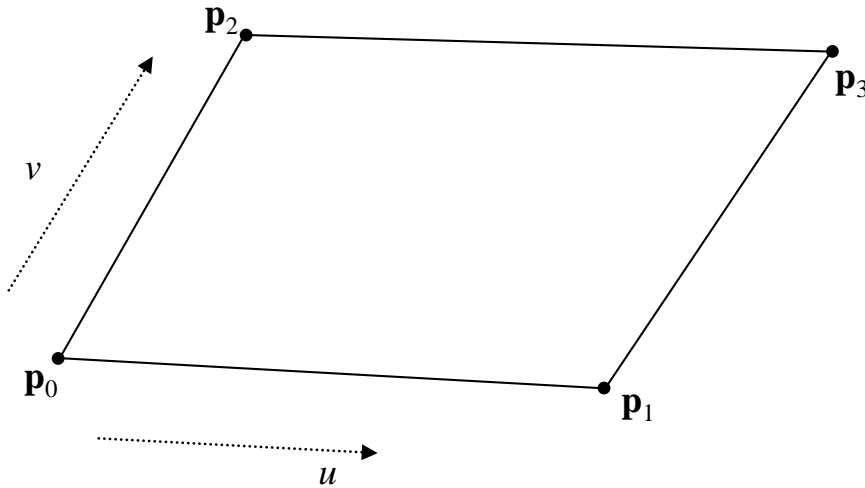
Typically we are interested in the unit normal, so we need to normalize

$$\vec{\mathbf{n}}^*(u, v) = \frac{\partial \mathbf{x}}{\partial u}(u, v) \times \frac{\partial \mathbf{x}}{\partial v}(u, v)$$

$$\vec{\mathbf{n}}(u, v) = \frac{\vec{\mathbf{n}}^*(u, v)}{|\vec{\mathbf{n}}^*(u, v)|}$$

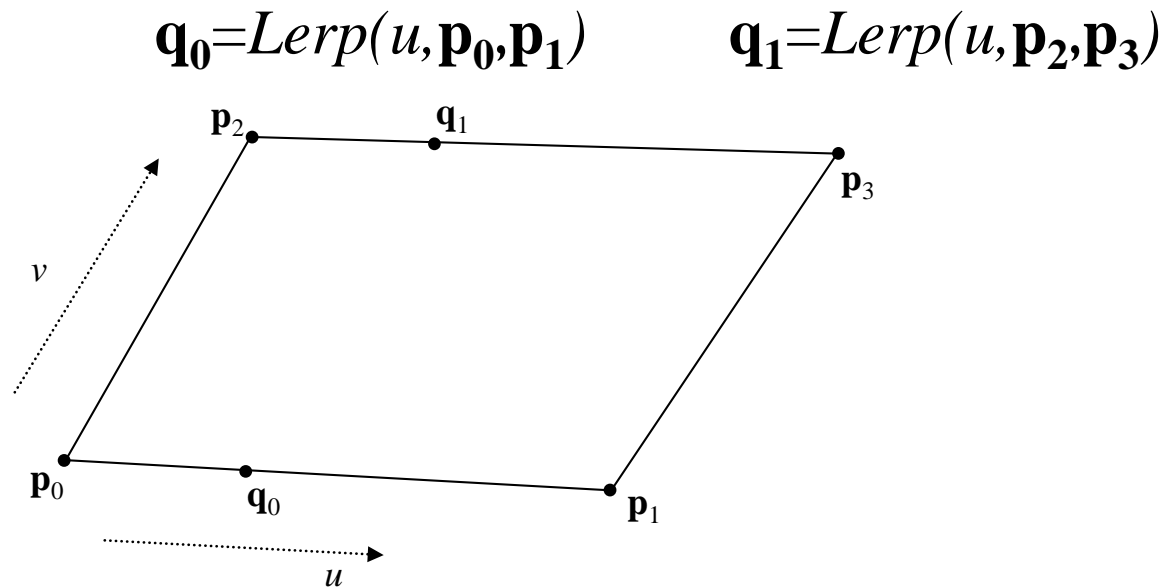
Bilinear Patch

- ▶ Control mesh with four points $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$
- ▶ Compute $\mathbf{x}(u, v)$ using a two-step construction scheme



Bilinear Patch (Step 1)

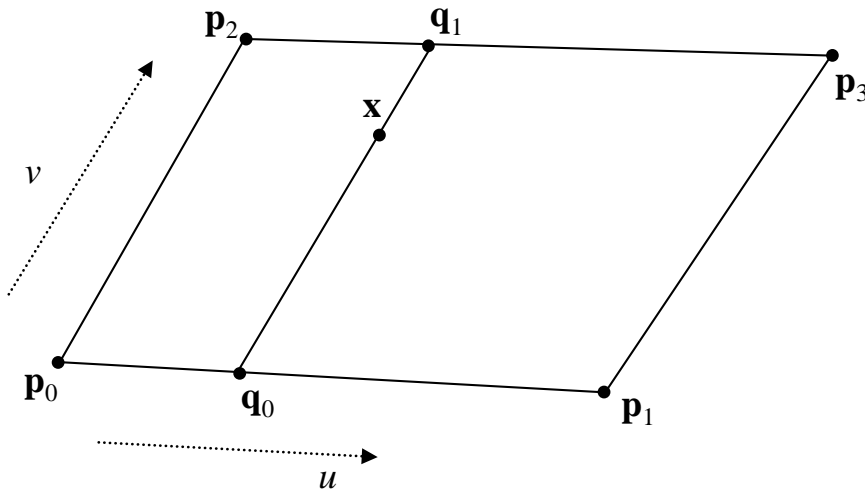
- ▶ For a given value of u , evaluate the linear curves on the two u -direction edges
- ▶ Use the same value u for both:



Bilinear Patch (Step 2)

- ▶ Consider that $\mathbf{q}_0, \mathbf{q}_1$ define a line segment
- ▶ Evaluate it using v to get \mathbf{x}

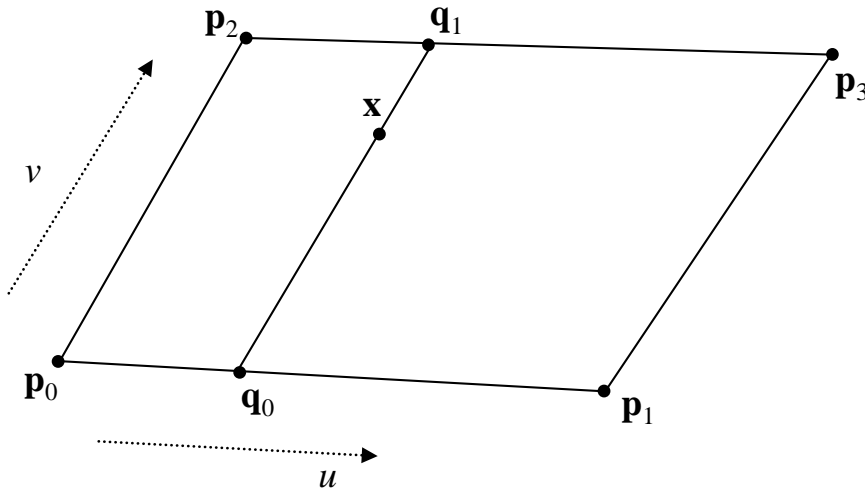
$$\mathbf{x} = \text{Lerp}(v, \mathbf{q}_0, \mathbf{q}_1)$$



Bilinear Patch

- ▶ Combining the steps, we get the full formula

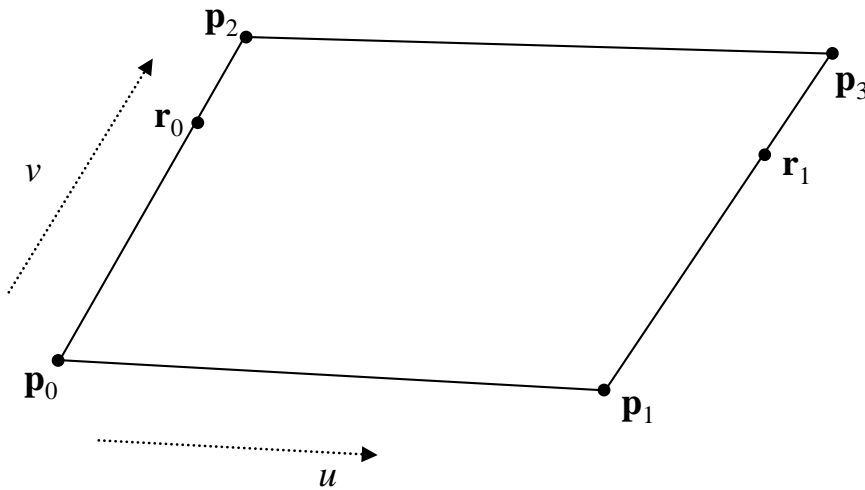
$$\mathbf{x}(u, v) = \text{Lerp}(v, \text{Lerp}(u, \mathbf{p}_0, \mathbf{p}_1), \text{Lerp}(u, \mathbf{p}_2, \mathbf{p}_3))$$



Bilinear Patch

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

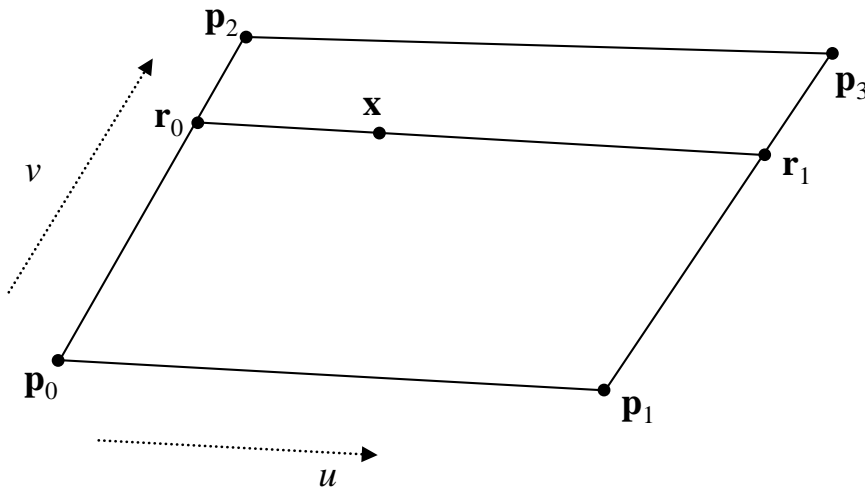
$$\mathbf{r}_0 = \text{Lerp}(v, \mathbf{p}_0, \mathbf{p}_2) \quad \mathbf{r}_1 = \text{Lerp}(v, \mathbf{p}_1, \mathbf{p}_3)$$



Bilinear Patch

- ▶ Consider that $\mathbf{r}_0, \mathbf{r}_1$ define a line segment
- ▶ Evaluate it using u to get \mathbf{x}

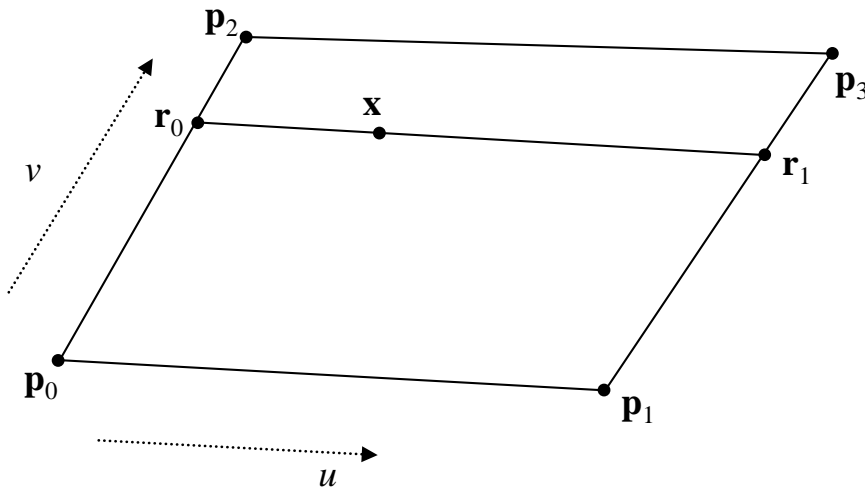
$$\mathbf{x} = \text{Lerp}(u, \mathbf{r}_0, \mathbf{r}_1)$$



Bilinear Patch

- ▶ The full formula for the v direction first:

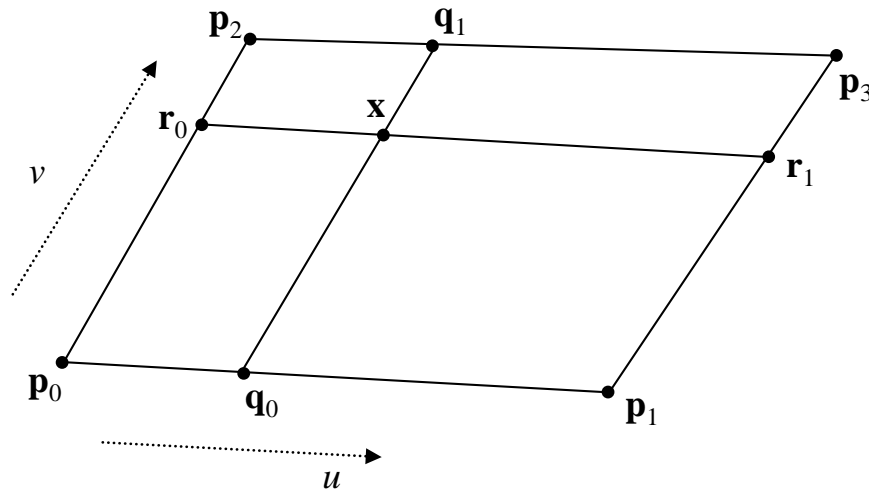
$$\mathbf{x}(u, v) = \text{Lerp}(u, \text{Lerp}(v, \mathbf{p}_0, \mathbf{p}_2), \text{Lerp}(v, \mathbf{p}_1, \mathbf{p}_3))$$



Bilinear Patch

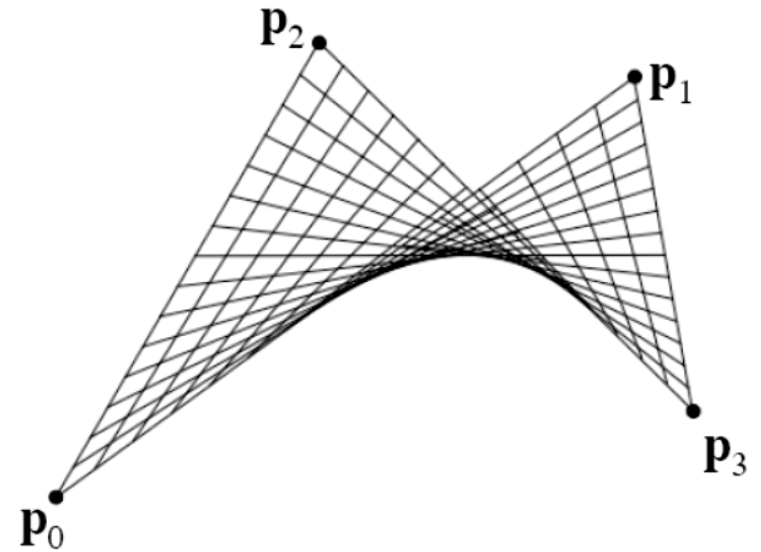
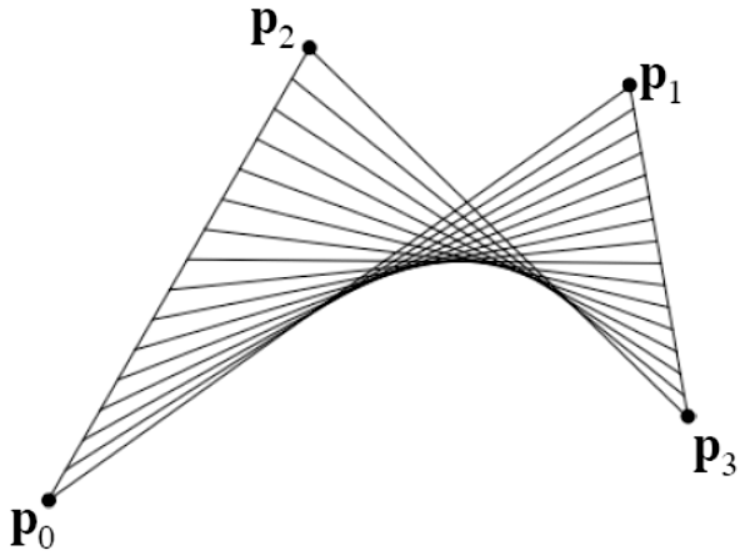
- ▶ Patch geometry is independent of the order of u and v

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



Bilinear Patch

► Visualization



Bilinear Patches

- ▶ **Weighted sum of control points**

$$\mathbf{x}(u, v) = (1-u)(1-v)\mathbf{p}_0 + u(1-v)\mathbf{p}_1 + (1-u)v\mathbf{p}_2 + uv\mathbf{p}_3$$

- ▶ **Bilinear polynomial**

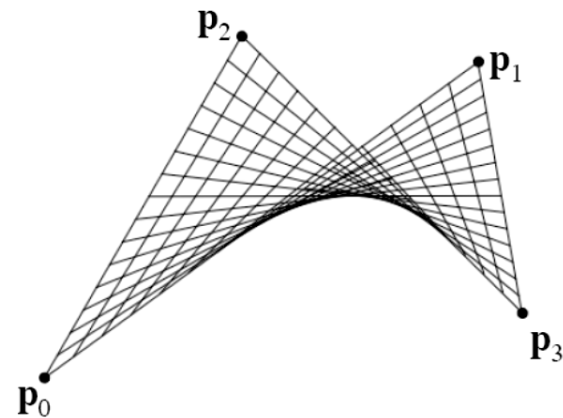
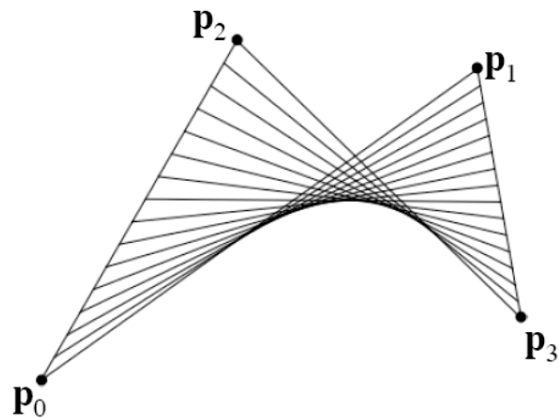
$$\mathbf{x}(u, v) = (\mathbf{p}_0 - \mathbf{p}_1 - \mathbf{p}_2 + \mathbf{p}_3)uv + (\mathbf{p}_1 - \mathbf{p}_0)u + (\mathbf{p}_2 - \mathbf{p}_0)v + \mathbf{p}_0$$

- ▶ **Matrix form**

$$\mathbf{x}(u, v) = \begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 & \mathbf{p}_2 \\ \mathbf{p}_1 & \mathbf{p}_3 \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix}$$

Properties

- ▶ Patch 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



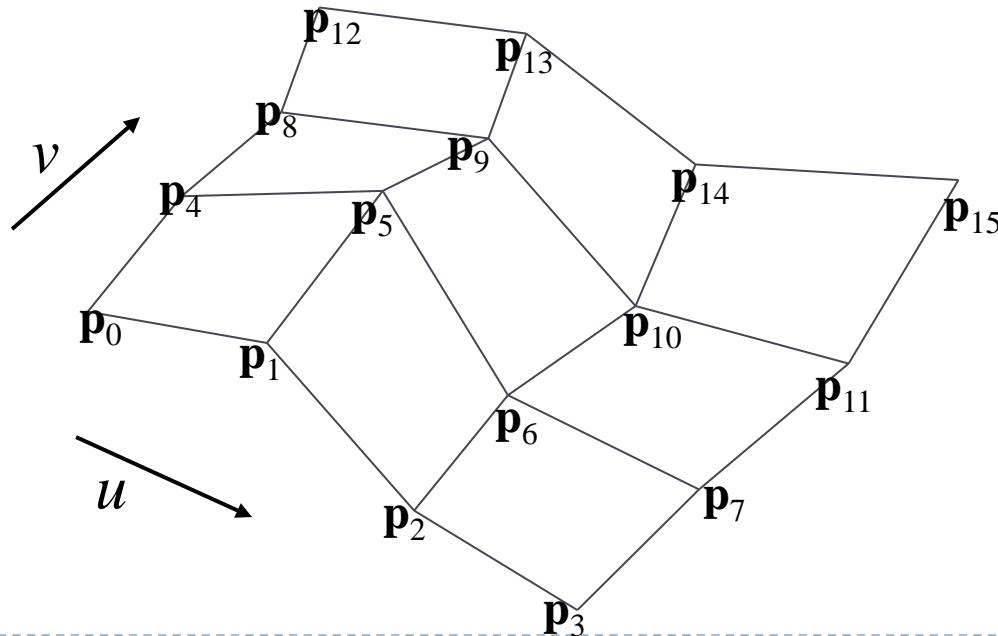
- ▶ Not terribly useful as a modeling primitive

Overview

- ▶ Bi-linear patch
- ▶ **Bi-cubic Bézier patch**

Bicubic Bézier patch

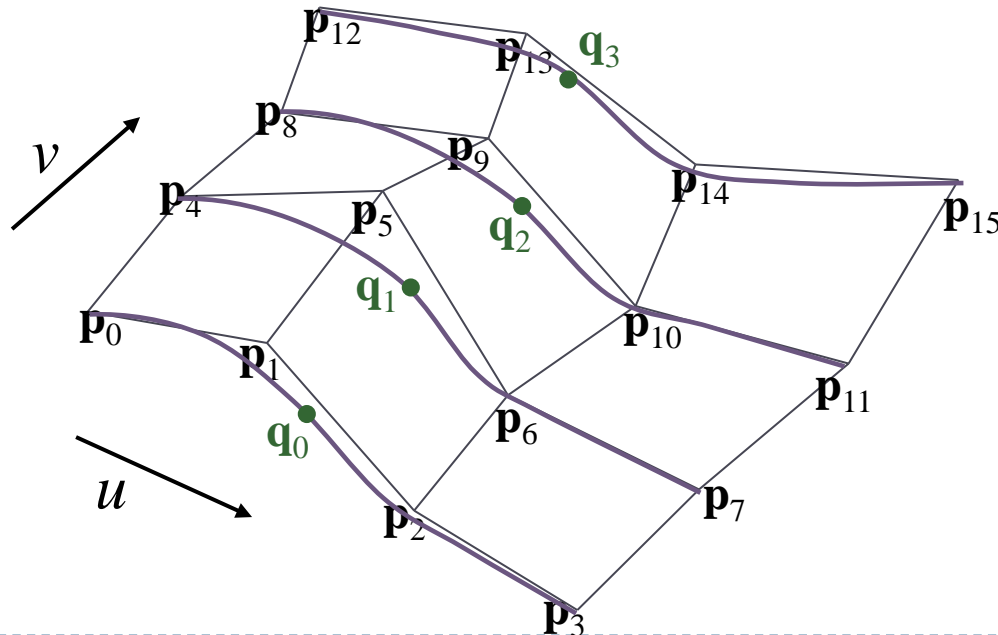
- ▶ Grid of 4x4 control points, \mathbf{p}_0 through \mathbf{p}_{15}
- ▶ Four rows of control points define Bézier curves along u
 $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$; $\mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7$; $\mathbf{p}_8, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_{11}$; $\mathbf{p}_{12}, \mathbf{p}_{13}, \mathbf{p}_{14}, \mathbf{p}_{15}$
- ▶ 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 scalar value u $[0..1]$

- ▶ Get points $\mathbf{q}_0 \dots \mathbf{q}_3$
 - $\mathbf{q}_0 = \text{Bez}(u, \mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$
 - $\mathbf{q}_1 = \text{Bez}(u, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7)$
 - $\mathbf{q}_2 = \text{Bez}(u, \mathbf{p}_8, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_{11})$
 - $\mathbf{q}_3 = \text{Bez}(u, \mathbf{p}_{12}, \mathbf{p}_{13}, \mathbf{p}_{14}, \mathbf{p}_{15})$

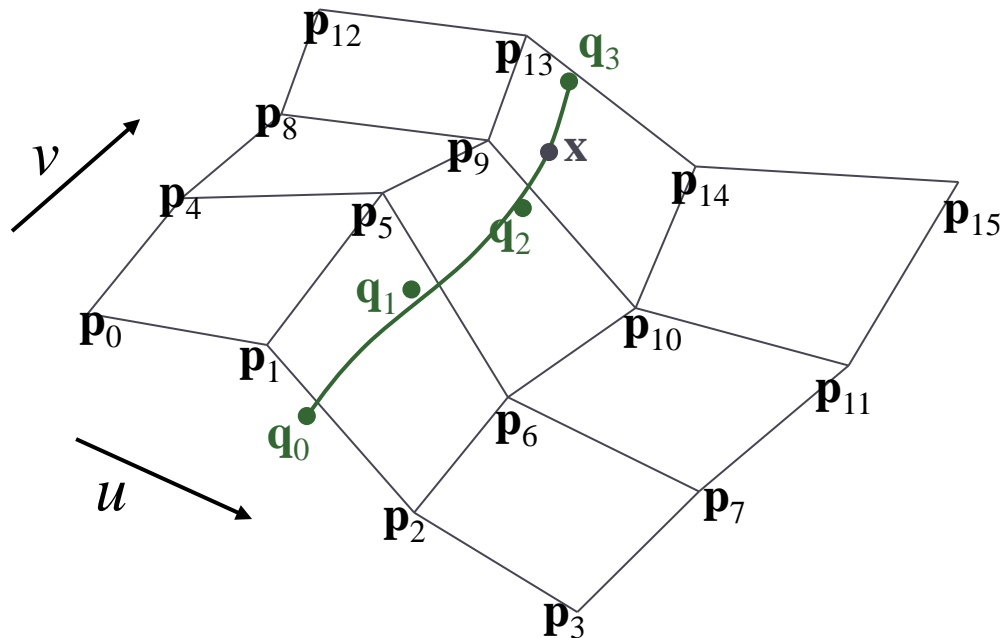


Bézier Patch (Step 2)

- ▶ Points $\mathbf{q}_0 \dots \mathbf{q}_3$ define a Bézier curve

- ▶ Evaluate it at $v [0..1]$

$$\mathbf{x}(u, v) = \text{Bez}(v, \mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$$



Bézier Patch

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

$$\mathbf{q}_0 = \text{Bez}(u, \mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$$

$$\mathbf{q}_1 = \text{Bez}(u, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7)$$

$$\mathbf{q}_2 = \text{Bez}(u, \mathbf{p}_8, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_{11}) \Leftrightarrow$$

$$\mathbf{q}_3 = \text{Bez}(u, \mathbf{p}_{12}, \mathbf{p}_{13}, \mathbf{p}_{14}, \mathbf{p}_{15})$$

$$\mathbf{r}_0 = \text{Bez}(v, \mathbf{p}_0, \mathbf{p}_4, \mathbf{p}_8, \mathbf{p}_{12})$$

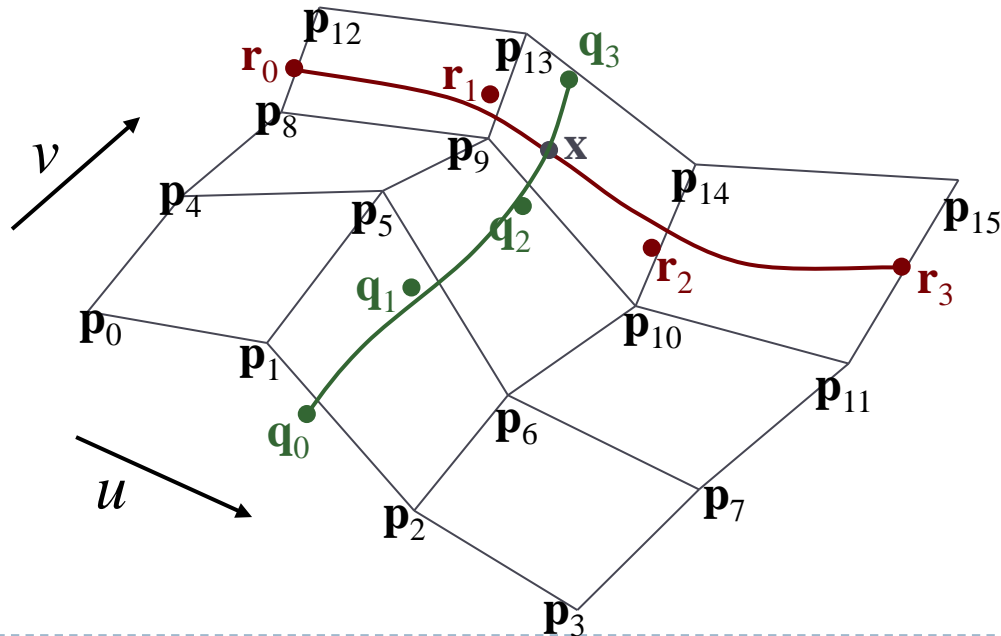
$$\mathbf{r}_1 = \text{Bez}(v, \mathbf{p}_1, \mathbf{p}_5, \mathbf{p}_9, \mathbf{p}_{13})$$

$$\mathbf{r}_2 = \text{Bez}(v, \mathbf{p}_2, \mathbf{p}_6, \mathbf{p}_{10}, \mathbf{p}_{14})$$

$$\mathbf{r}_3 = \text{Bez}(v, \mathbf{p}_3, \mathbf{p}_7, \mathbf{p}_{11}, \mathbf{p}_{15})$$

$$\mathbf{x}(u, v) = \text{Bez}(v, \mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$$

$$\mathbf{x}(u, v) = \text{Bez}(u, \mathbf{r}_0, \mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3)$$



Bézier Patch: Matrix Form

$$\mathbf{U} = \begin{bmatrix} u^3 \\ u^2 \\ u \\ 1 \end{bmatrix} \quad \mathbf{V} = \begin{bmatrix} v^3 \\ v^2 \\ v \\ 1 \end{bmatrix} \quad \mathbf{B}_{Bez} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \mathbf{B}_{Bez}^T$$

$$\mathbf{C}_x = \mathbf{B}_{Bez}^T \mathbf{G}_x \mathbf{B}_{Bez}$$

$$\mathbf{C}_y = \mathbf{B}_{Bez}^T \mathbf{G}_y \mathbf{B}_{Bez}$$

$$\mathbf{C}_z = \mathbf{B}_{Bez}^T \mathbf{G}_z \mathbf{B}_{Bez}$$

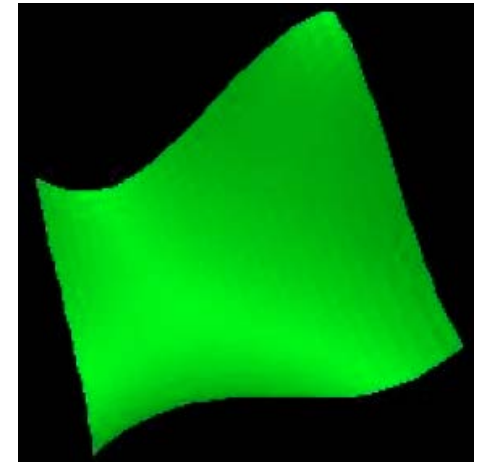
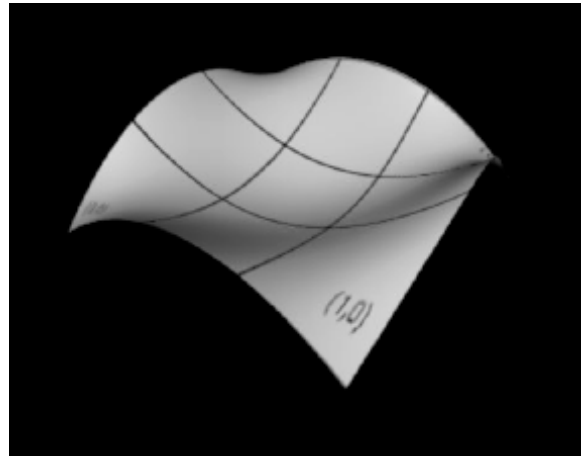
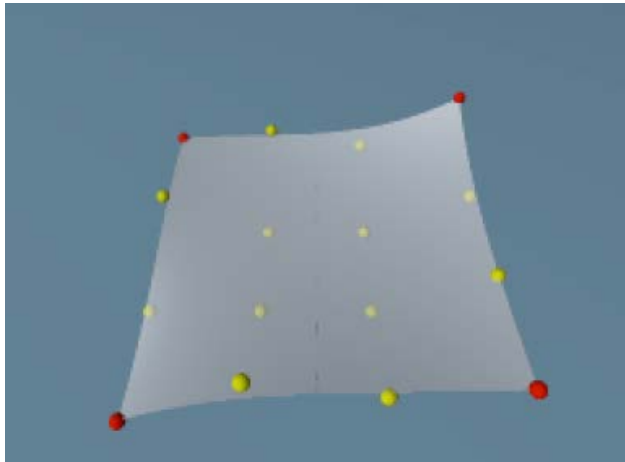
$$\mathbf{G}_x = \begin{bmatrix} p_{0x} & p_{1x} & p_{2x} & p_{3x} \\ p_{4x} & p_{5x} & p_{6x} & p_{7x} \\ p_{8x} & p_{9x} & p_{10x} & p_{11x} \\ p_{12x} & p_{13x} & p_{14x} & p_{15x} \end{bmatrix}, \quad \mathbf{G}_y = \dots, \quad \mathbf{G}_z = \dots$$

$$\mathbf{x}(u, v) = \begin{bmatrix} \mathbf{V}^T \mathbf{C}_x \mathbf{U} \\ \mathbf{V}^T \mathbf{C}_y \mathbf{U} \\ \mathbf{V}^T \mathbf{C}_z \mathbf{U} \end{bmatrix}$$

C stores the coefficients of the bicubic equation
G stores the control point geometry
B_{Bez} is the basis matrix (Bézier basis)
U and **V** are the vectors formed from the powers of u and v

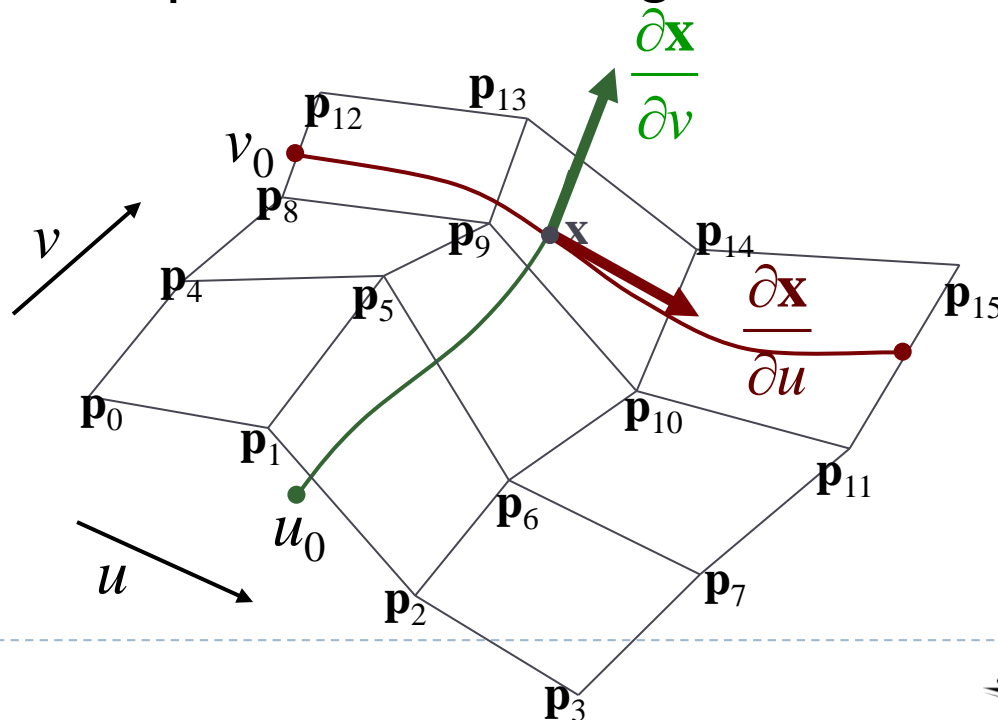
Properties

- ▶ 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 a 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 a Bézier patch

$$\mathbf{q}_0 = \text{Bez}(u, \mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$$

$$\mathbf{q}_1 = \text{Bez}(u, \mathbf{p}_4, \mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7)$$

$$\mathbf{q}_2 = \text{Bez}(u, \mathbf{p}_8, \mathbf{p}_9, \mathbf{p}_{10}, \mathbf{p}_{11})$$

$$\mathbf{q}_3 = \text{Bez}(u, \mathbf{p}_{12}, \mathbf{p}_{13}, \mathbf{p}_{14}, \mathbf{p}_{15})$$

$$\mathbf{r}_0 = \text{Bez}(v, \mathbf{p}_0, \mathbf{p}_4, \mathbf{p}_8, \mathbf{p}_{12})$$

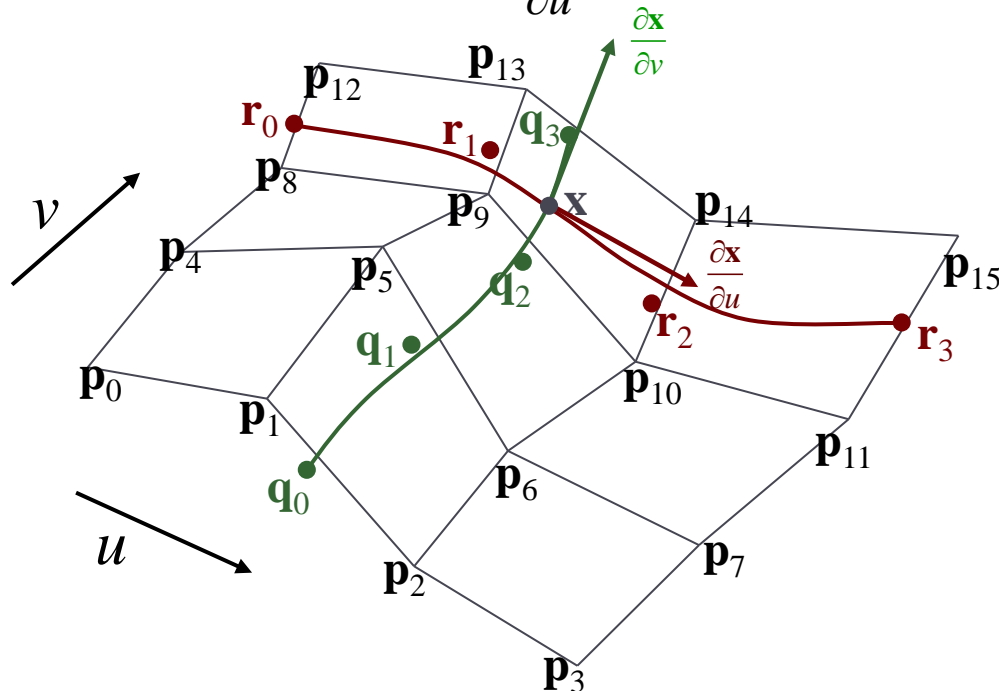
$$\mathbf{r}_1 = \text{Bez}(v, \mathbf{p}_1, \mathbf{p}_5, \mathbf{p}_9, \mathbf{p}_{13})$$

$$\mathbf{r}_2 = \text{Bez}(v, \mathbf{p}_2, \mathbf{p}_6, \mathbf{p}_{10}, \mathbf{p}_{14})$$

$$\mathbf{r}_3 = \text{Bez}(v, \mathbf{p}_3, \mathbf{p}_7, \mathbf{p}_{11}, \mathbf{p}_{15})$$

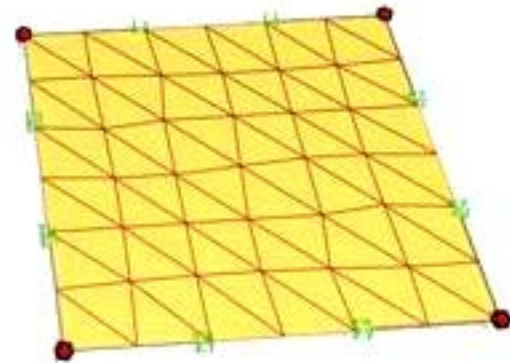
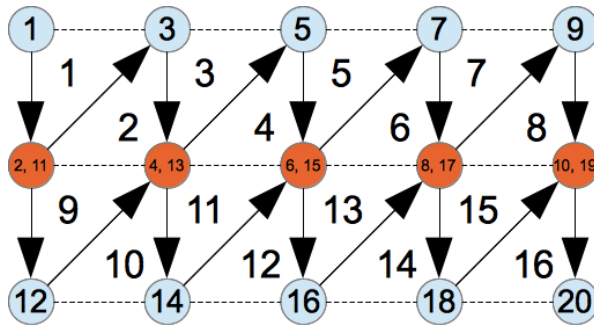
$$\frac{\partial \mathbf{x}}{\partial v}(u, v) = \text{Bez}'(v, \mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$$

$$\frac{\partial \mathbf{x}}{\partial u}(u, v) = \text{Bez}'(u, \mathbf{r}_0, \mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3)$$



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 with primitive type `GL_TRIANGLE_STRIP`



- ▶ Adaptive tessellation/recursive subdivision
 - ▶ Potential for “cracks” if patches on opposite sides of an edge divide differently
 - ▶ Tricky to get right, not usually worth the effort

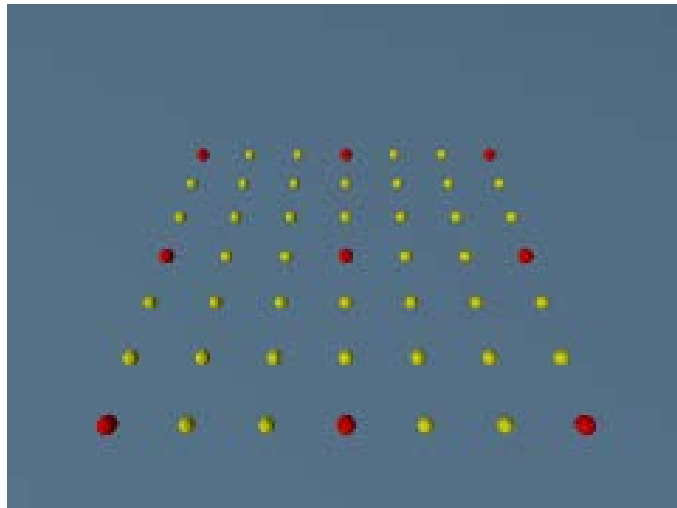
OpenGL Support

- ▶ OpenGL supports NURBS patches through GLU functions
- ▶ Structure:

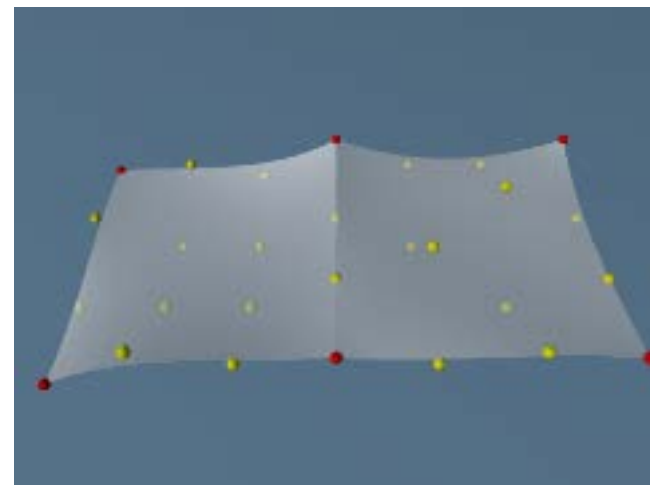
```
gluBeginSurface(nurbs);  
    gluNurbsSurface(GLUnurbs* nurbs,  
    GLint sKnotCount, GLfloat* sKnots,  
    GLint tKnotCount, GLfloat* tKnots,  
    GLint sStride, GLint tStride,  
    GLfloat* control,  
    GLint sOrder, GLint tOrder,  
    GLenum type);  
gluEndSurface(nurbs);
```

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...



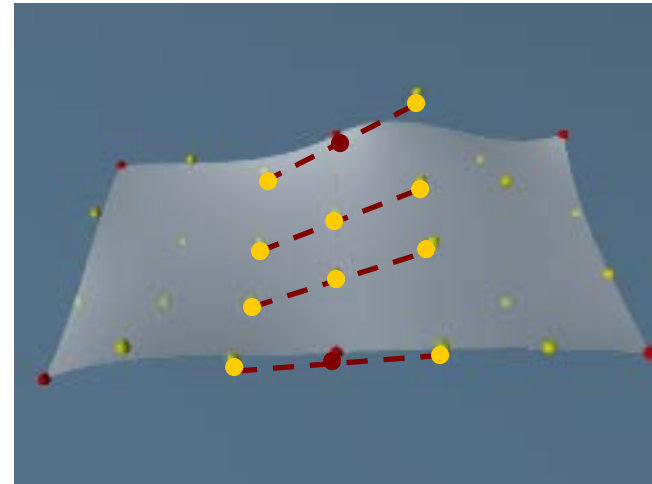
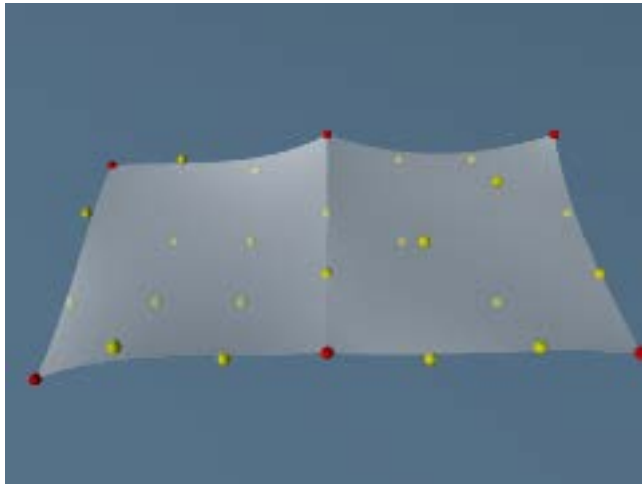
Grid of control points



Piecewise Bézier surface

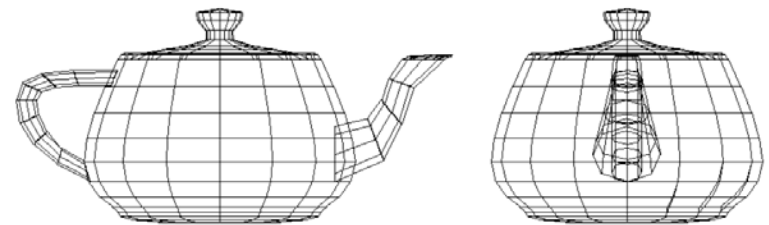
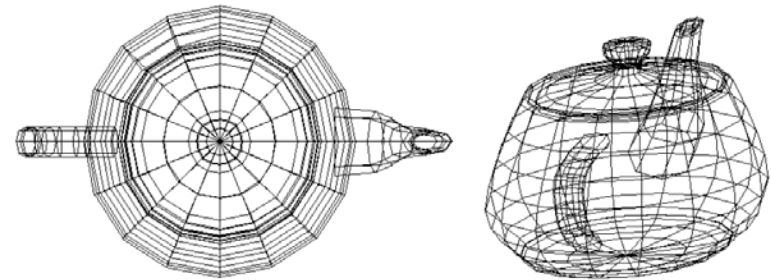
C^1 Continuity

- ▶ We want the parametric curves that cross each edge to have C^1 continuity
 - ▶ So the handles must be equal-and-opposite across the edge:



Modeling With Bézier Patches

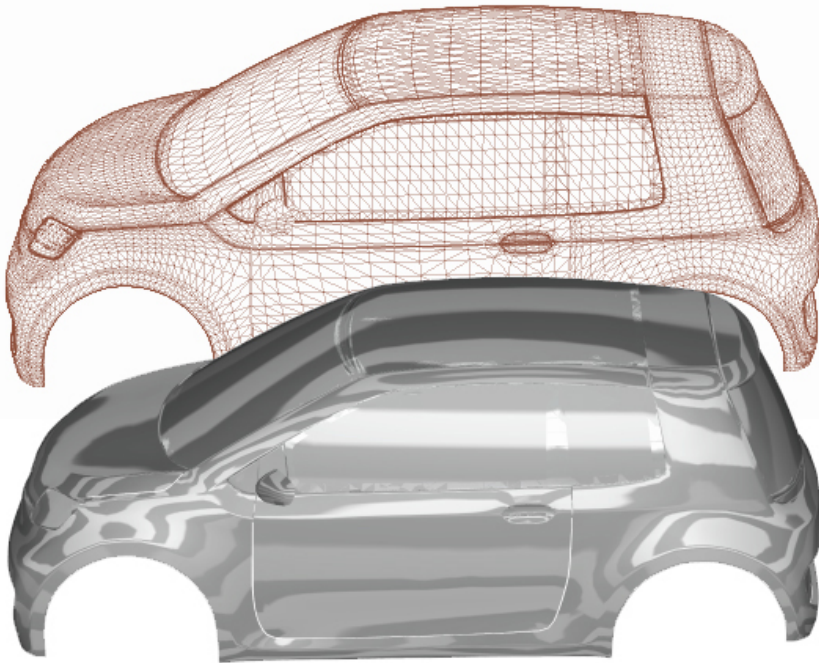
- ▶ Original Utah teapot, from Martin Newell's PhD thesis, consisted of 28 Bézier patches.
- ▶ The original had no rim for the lid and no bottom
- ▶ Later, four more patches were added to create a bottom, bringing the total to 32
- ▶ The data set was used by a number of people, including graphics guru Jim Blinn. In a demonstration of a system of his he scaled the teapot by .75, creating a stubbier teapot. He found it more pleasing to the eye, and it was this scaled version that became the highly popular dataset used today.



Pixar's walking teapot

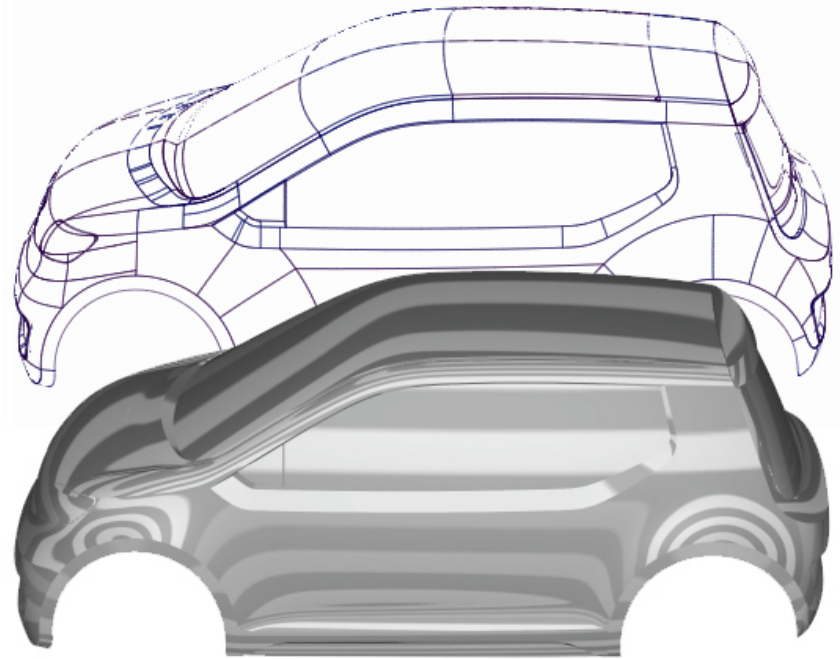
Comparing polygon to NURBS model

Polygon model



Poor surface quality

NURBS model



Pure, smooth highlights

Source: <https://www.aliasworkbench.com>