Blending Functions

- We can rewrite $\mathbf{p}(u)$ as

$$\mathbf{p}(u) = \mathbf{u}^T\mathbf{c} = \mathbf{u}^T\mathbf{M}_I\mathbf{p}$$

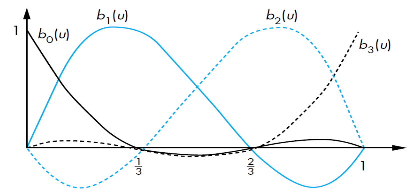
$$\mathbf{p}(u) = \mathbf{b}(u)^T\mathbf{p}, \quad \text{where} \quad \mathbf{b}(u) = \mathbf{M}_I^T\mathbf{u}$$

- $\mathbf{b}(u)$ is a column matrix of 4 **blending polynomials**, where each is a cubic

$$\mathbf{b}(u) = \begin{bmatrix} b_0(u) \\ b_1(u) \\ b_2(u) \\ b_3(u) \end{bmatrix} \quad \text{where} \quad \begin{aligned} b_0(u) &= -\frac{9}{2}\left(u-\frac{1}{3}\right)\left(u-\frac{2}{3}\right)(u-1), \\ b_1(u) &= \frac{27}{2}u\left(u-\frac{2}{3}\right)(u-1), \\ b_2(u) &= -\frac{27}{2}u\left(u-\frac{1}{3}\right)(u-1), \\ b_3(u) &= \frac{9}{2}u\left(u-\frac{1}{3}\right)\left(u-\frac{2}{3}\right). \end{aligned}$$

Blending Functions

- A plot of the blending functions for interpolation



- When we express $\mathbf{p}(u)$ in terms of these blending polynomials as

$$\mathbf{p}(u) = b_0(u)\mathbf{p}_0 + b_1(u)\mathbf{p}_1 + b_2(u)\mathbf{p}_2 + b_3(u)\mathbf{p}_3 = \sum_{i=0}^{3} b_i(u)\mathbf{p}_i,$$

we can see that the polynomials blend together the individual contributions of each control point

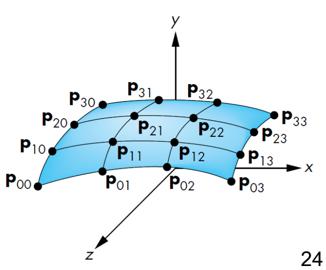
Cubic Interpolation Patch

- A bicubic surface patch can be written in the form

$$\mathbf{p}(u, v) = \sum_{i=0}^{3}\sum_{j=0}^{3} u^i v^j \mathbf{c}_{ij},$$

$$\mathbf{p}(u, v) = \mathbf{u}^T\mathbf{C}\mathbf{v}, \quad \text{where} \quad \mathbf{C} = [\,\mathbf{c}_{ij}\,], \quad \mathbf{v} = \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

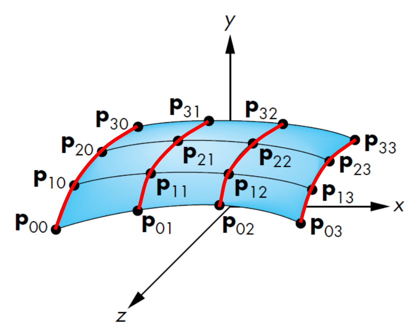
- A bicubic interpolating surface patch interpolates 16 control points, $\mathbf{p}_{ij}$, $i = 0, \ldots, 3$ and $j = 0, \ldots, 3$



Cubic Interpolation Patch

- We can obtain the coefficients $\{\mathbf{c}_{ij}\}$ by independently finding the coefficients for the 4 cubic interpolation curves that interpolate $\{\mathbf{p}_{00}, \mathbf{p}_{10}, \mathbf{p}_{20}, \mathbf{p}_{30}\}$, $\{\mathbf{p}_{01}, \mathbf{p}_{11}, \mathbf{p}_{21}, \mathbf{p}_{31}\}$, $\{\mathbf{p}_{02}, \mathbf{p}_{12}, \mathbf{p}_{22}, \mathbf{p}_{32}\}$, and $\{\mathbf{p}_{03}, \mathbf{p}_{13}, \mathbf{p}_{23}, \mathbf{p}_{33}\}$ respectively

$$\mathbf{p}(u, v) = \sum_{i=0}^{3}\sum_{j=0}^{3} u^i v^j \mathbf{c}_{ij},$$

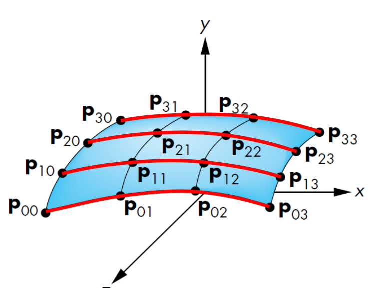


Cubic Interpolation Patch

- Alternatively, we can also obtain the coefficients $\{\mathbf{c}_{ij}\}$ by independently finding the coefficients for the 4 cubic interpolation curves that interpolate $\{\mathbf{p}_{00}, \mathbf{p}_{01}, \mathbf{p}_{02}, \mathbf{p}_{03}\}$, $\{\mathbf{p}_{10}, \mathbf{p}_{11}, \mathbf{p}_{12}, \mathbf{p}_{13}\}$, $\{\mathbf{p}_{20}, \mathbf{p}_{21}, \mathbf{p}_{22}, \mathbf{p}_{23}\}$, and $\{\mathbf{p}_{30}, \mathbf{p}_{31}, \mathbf{p}_{32}, \mathbf{p}_{33}\}$ respectively

- An example of **separable surfaces**, which can be written as

$$\mathbf{p}(u,v) = \mathbf{f}(u)\mathbf{g}(v)$$



Blending Patches

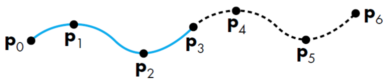
- The bicubic interpolating surface patch can be written as

$$\mathbf{p}(u, v) = \sum_{i=0}^{3}\sum_{j=0}^{3} b_i(u)b_j(v)\mathbf{p}_{ij}.$$

- Each term $b_i(u)b_j(v)$ describes a **blending patch**
  - The surface is formed by blending together 16 simple patches, each weighted by a control point

Geometric and Parametric Continuity

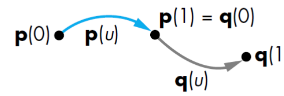
- A long curve is normally made of multiple curve segments joined together



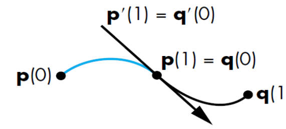
- We want to ensure continuity (or "smoothness") at the join points

Geometric and Parametric Continuity

- Consider two curve segments, $\mathbf{p}(u)$ and $\mathbf{q}(u)$
- If $\mathbf{p}(1) = \mathbf{q}(0)$, we say there is C^0 **parametric continuity** at the join point



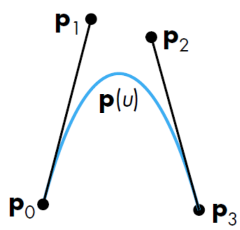
- If $\mathbf{p}'(1) = \mathbf{q}'(0)$, we say there is C^1 **parametric continuity** at the join point



- If $\mathbf{p}'(1) = \alpha\mathbf{q}'(0)$, for some positive number α, we say there is G^1 **geometric continuity** at the join point
- We can extend the idea to higher derivatives and talk about C^n and G^n continuity

Cubic Bézier Curves

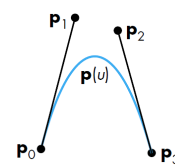
- Given control points $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$, we want
  - $\mathbf{p}(0) = \mathbf{p}_0$
  - $\mathbf{p}(1) = \mathbf{p}_3$
  - $\mathbf{p}'(0) = 3(\mathbf{p}_1 - \mathbf{p}_0)$
  - $\mathbf{p}'(1) = 3(\mathbf{p}_3 - \mathbf{p}_2)$



Deriving Cubic Bézier Curves

- We seek the coefficients $\mathbf{c}$ of the polynomial $\mathbf{p}(u) = \mathbf{u}^T\mathbf{c}$
- We have the following 4 conditions

$$
\begin{aligned}
\mathbf{p}_0 &= \mathbf{c}_0,\\
\mathbf{p}_3 &= \mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2 + \mathbf{c}_3.\\
3\mathbf{p}_1 - 3\mathbf{p}_0 &= \mathbf{c}_1,\\
3\mathbf{p}_3 - 3\mathbf{p}_2 &= \mathbf{c}_1 + 2\mathbf{c}_2 + 3\mathbf{c}_3
\end{aligned}
$$



Deriving Cubic Bézier Curves

- The desired coefficients are

$$\mathbf{c} = \mathbf{M}_B\mathbf{p} \quad \text{where} \quad \mathbf{p} = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

- $\mathbf{M}_B$ is the **Bézier geometry matrix**

$$\mathbf{M}_B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

Note that $\mathbf{M}_B$ is the same for *any* 4 control points

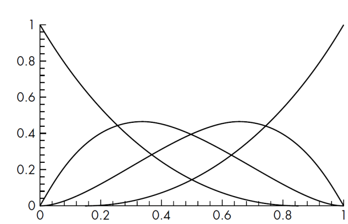
Blending Functions of Cubic Bézier Curves

- We can rewrite $\mathbf{p}(u)$ as

$$\mathbf{p}(u) = \mathbf{u}^T\mathbf{M}_B\mathbf{p}.$$

$$\mathbf{p}(u) = \mathbf{b}(u)^T\mathbf{p}, \quad \text{where} \quad \mathbf{b}(u) = \mathbf{M}_B^T\mathbf{u} = \begin{bmatrix} (1-u)^3 \\ 3u(1-u)^2 \\ 3u^2(1-u) \\ u^3 \end{bmatrix}$$

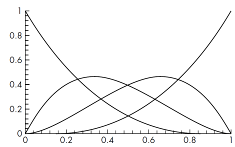
- The four blending polynomials are **Bernstein polynomials** of degree 3



Blending Functions of Cubic Bézier Curves

- The 4 Bernstein polynomials $b_i(u)$ have the properties

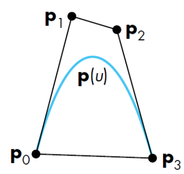
$$0 < b_i(u) < 1 \quad \text{for } 0 < u < 1$$
$$\sum_{i=0}^{3} b_i(u) = 1$$



- Therefore the following is a **convex sum**

$$\mathbf{p}(u) = b_0(u)\mathbf{p}_0 + b_1(u)\mathbf{p}_1 + b_2(u)\mathbf{p}_2 + b_3(u)\mathbf{p}_3 = \sum_{i=0}^{3} b_i(u)\mathbf{p}_i,$$

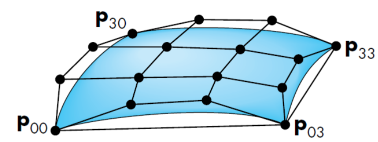
- Consequently, $\mathbf{p}(u)$ must lie in the **convex hull** of the four control points
  - Make it easy to interactively specify a Bézier curve



Bicubic Bézier Surface Patches

- Given 4 x 4 array of control points $\{\mathbf{p}_{ij}\}$, the corresponding Bézier patch is

$$\mathbf{p}(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} b_i(u)b_j(v)\mathbf{p}_{ij} = \mathbf{u}^T\mathbf{M}_B\mathbf{P}\mathbf{M}_B^T\mathbf{v}$$



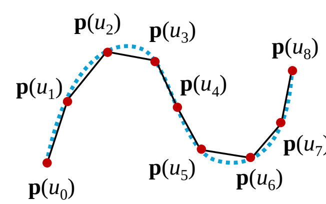
Rendering Curves and Surfaces

Rendering Polynomial Curves

- Given the representation of the curve segment

$$\mathbf{p}(u) = \sum_{i=0}^{n} \mathbf{c}_i u^i, \quad 0 \le u \le 1.$$

- Evaluate $\mathbf{p}(u)$ at a sequence of values $\{u_k\}$, and join the points using straight line segments (a polyline)

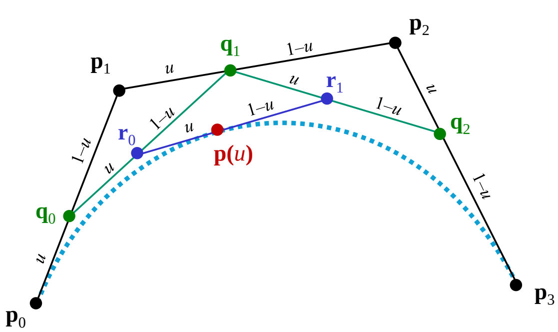


- Use **Horner's method** for more efficient evaluation of $\mathbf{p}(u)$

$$\mathbf{p}(u) = \mathbf{c}_0 + u(\mathbf{c}_1 + u(\mathbf{c}_2 + u(\ldots + \mathbf{c}_n u))).$$

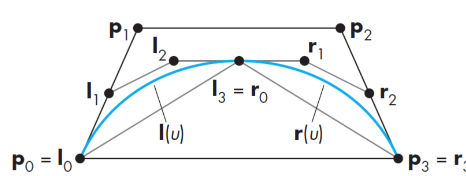
Rendering Bézier Curves

- **De Casteljau** algorithm
  - $\mathbf{p}(u)$ is computed by a sequence of recursive linear interpolations between successive control points



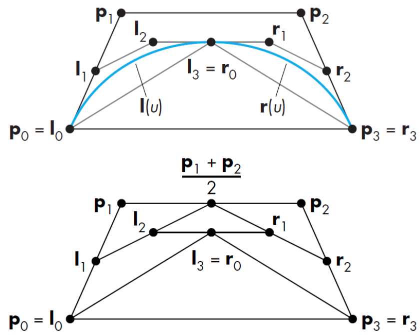
Recursive Subdivision of Bézier Curves

- We can recursively subdivide a Bezier curve segment into two shorter Bézier curve segments
  - A shorter curve segment will be drawn as a straight line segment if it is "straight" or "flat" enough
    - We can use the **convex hull** defined by the control points of the shorter curve segment to determine "straightness" or "flatness"



Recursive Subdivision of Bézier Curves

- Construction of the subdivision curve segments



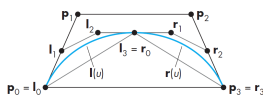
Note that no evaluation of $\mathbf{p}(u)$ is needed

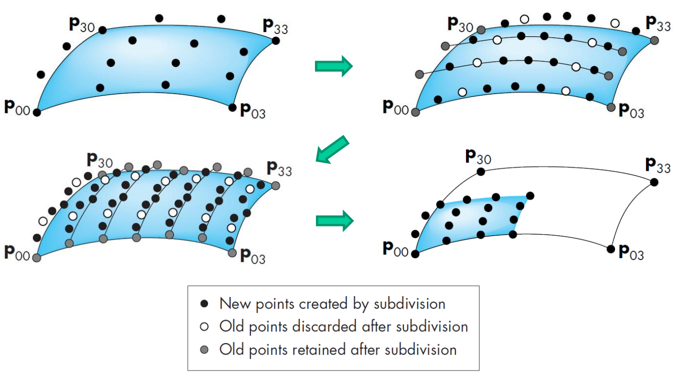
Rendering Other Polynomial Curves

- Other polynomial curve segments can be converted to (re-expressed as) Bezier curve segments, and rendered using the recursive subdivision approach

Recursive Subdivision of Bézier Patches

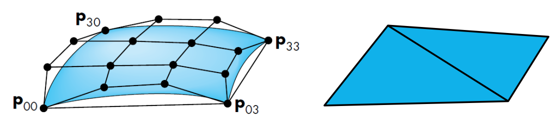
- The Bézier curve subdivision can be extended to Bézier patches

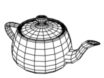




Recursive Subdivision of Bézier Patches

- We can use the **convex hull** defined by the control points of the subdivision patch to determine its "flatness"
  - If it is flat enough, we draw the patch as a **quadrilateral** or **two triangles**



The Utah Teapot 

- Created by **Mike Newell** at **University of Utah** for testing of various rendering algorithms
  - https://en.wikipedia.org/wiki/Utah_teapot

- Made of 32 bicubic Bézier patches
  - Data can be downloaded from http://www.holmes3d.net/graphics/teapot/

End of Lecture 10