Curves and Surfaces

How to represent a shape?

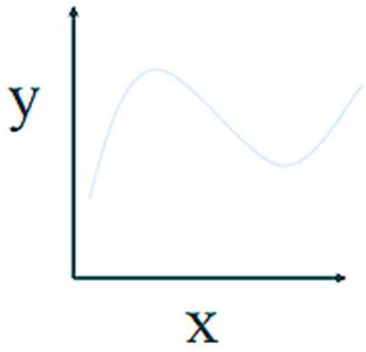
- There are many ways to represent 2D curves and 3D surfaces
- Want a good representation that is
  - Stable
  - Smooth
  - Easy to evaluate

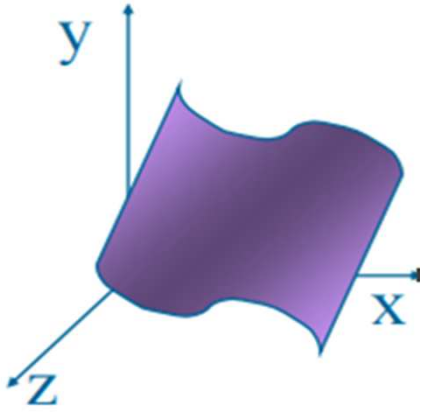
Explicit Representation

- Most familiar form of curve in 2D $y = f(x)$

where x is independent and y is dependent

- For a given curve, no guarantee of explicit representation
  - Line: $y = mx + h$ (not for vertical lines)
  - Circle: $y = \pm\sqrt{r^2 - x^2}$
- Extension to 3D $y = f(x), z = g(x)$
- The form $z = f(x, y)$ defines a surface





Implicit Representation

- Two dimensional curve(s) $f(x, y) = 0$

- Much more robust
  - All lines $ax + by + c = 0$
  - Circles $x^2 + y^2 + r^2 = 0$

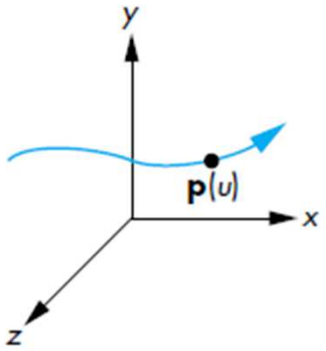
- Three dimensions $f(x, y, z) = 0$ defines a surface
  - Intersect two surface to get a curve

Parametric Representation

- Each spatial variable expressed in terms of an independent variable

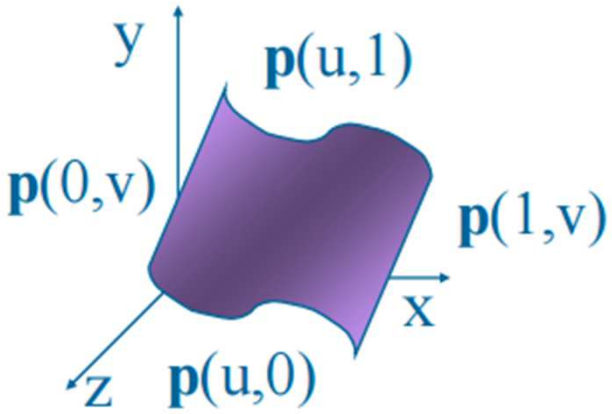
$x = x(u)$, $y = y(u)$, and $z = z(u)$. **u is the parameter**

$$p(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix}$$



- Parametric surfaces require two parameters

$x = x(u, v)$
$y = y(u, v)$
$z = z(u, v)$



Design Criteria for Curves

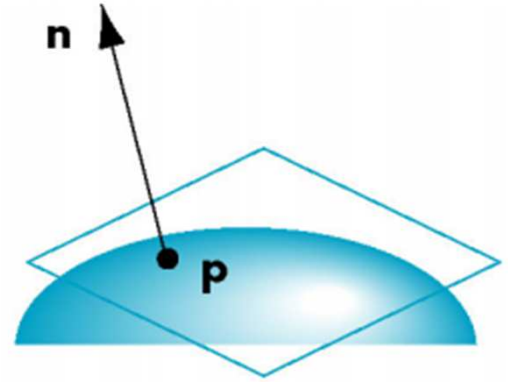
- Local control of shape: simple curve segments joined together provide better shape control
- Continuity and smoothness: touching segments and smooth transition between segments
- Derivative which can be evaluated: to evaluate smoothness
- Stability: smooth changing of parameters leads to intuitive changes in curve segment
- Ease of rendering

Normals

- We can differentiate with respect to u and v to obtain the normal at any point p

$$\frac{\partial p(u,v)}{\partial u} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial u} \\ \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial z(u,v)}{\partial u} \end{bmatrix}, \frac{\partial p(u,v)}{\partial v} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial v} \\ \frac{\partial y(u,v)}{\partial v} \\ \frac{\partial z(u,v)}{\partial v} \end{bmatrix}$$

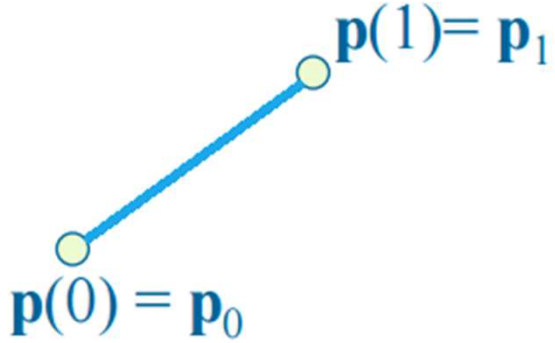
- $n = \frac{\partial p(u,v)}{\partial u} \times \frac{\partial p(u,v)}{\partial v}$



Parametric Lines

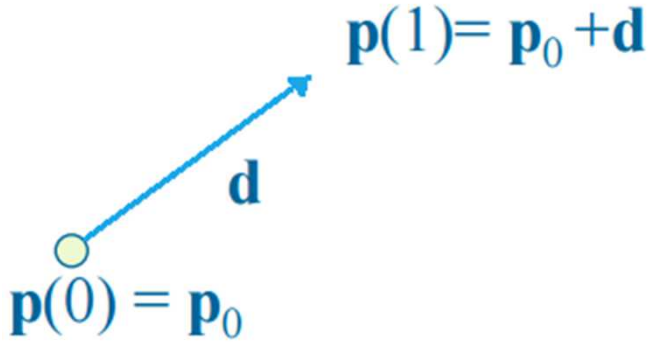
- We can normalize u to be over the interval (0,1)
- Line connecting two points p_0 and p_1

$$p(u) = (1-u)p_0 + up_1$$



- Ray from p_0 in the direction d

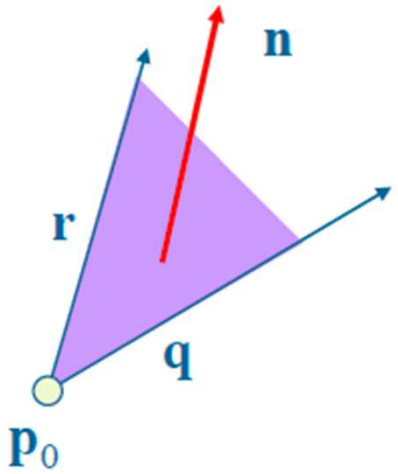
$$p(u) = p_0 + ud$$



Parametric Planes

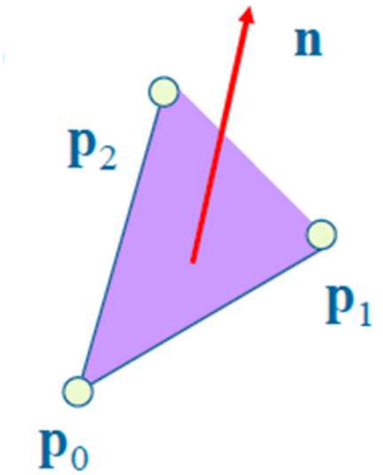
- Point-vector form

$$p(u,v) = p_0 + uq + vr$$
$$n = q \times r$$



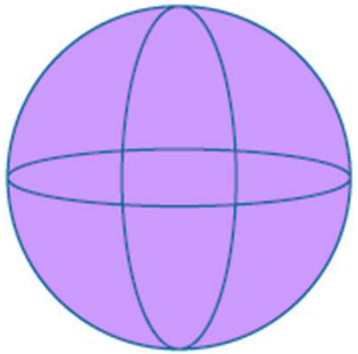
- Three-point form

$$q = p_1 - p_0$$
$$r = p_2 - p_0$$



Parametric Sphere

$$x(\theta, \phi) = r cos\theta sin\phi$$
$$y(\theta, \phi) = r sin\theta sin\phi$$
$$\mathrm{z}(\theta, \phi) = r cos\phi$$



- $360 \geq \theta \geq 0$
- $180 \geq \phi \geq 0$

- θ constant: circles of constant longitude
- ϕ constant: circles of constant latitude

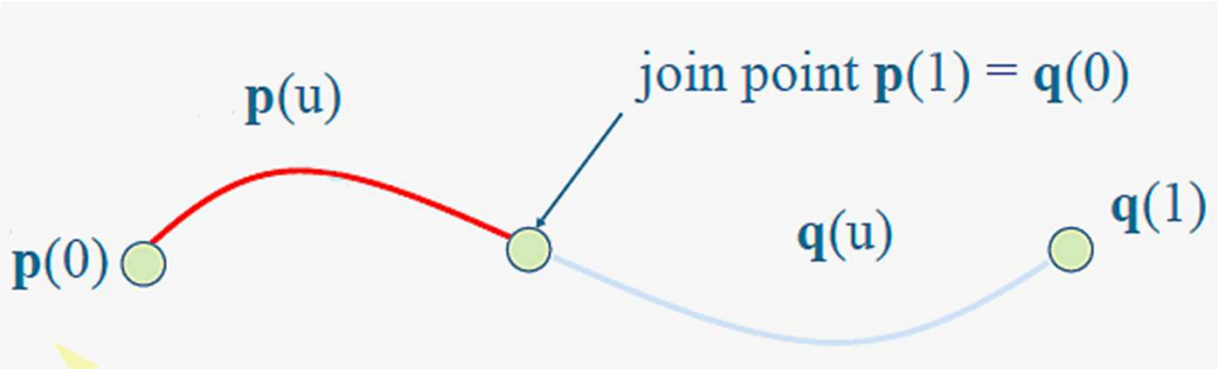
Curve Segments

- After normalizing u, each curve is written

$$p(u) = [x(u) \quad y(u) \quad z(u)]^T \quad 1 \geq u \geq 0$$

- In classical numerical methods, we design a single global curve

- In computer graphics and CAD, it is better to design small connected curve *segments*



Parametric Polynomial Curves

- A polynomial parametric curve of degree n is of the form

$$p(u) = \sum_{k=0}^{n} u^k c_k$$

- Where each c_k has independent x, y, and z components

$$c_k = \begin{bmatrix} c_{xk} \\ c_{yk} \\ c_{zk} \end{bmatrix}$$

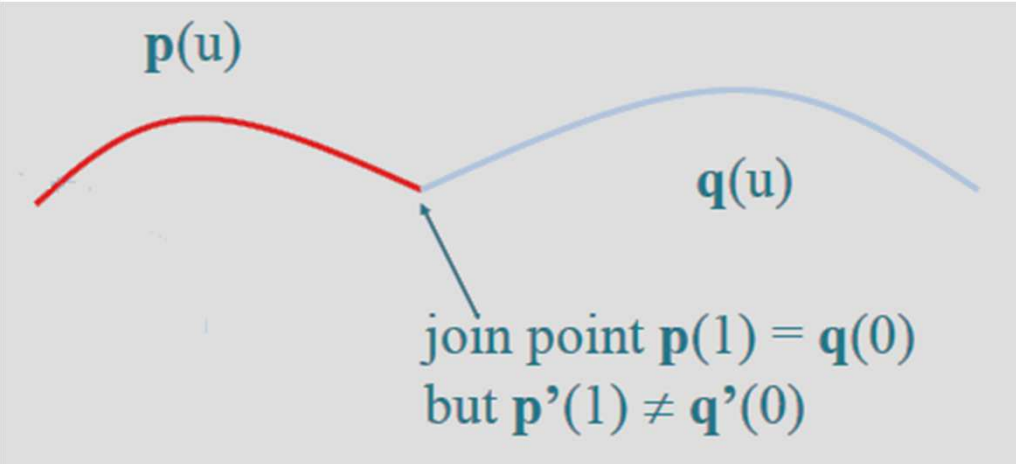
- $3(n+1)$ degrees of freedom on how to choose coefficient of a particular p

- Note that the curves for x, y, and z are independent, we can define each independently in an identical manner

Why Polynomials

- Easy to evaluate

- Continuous and differentiable everywhere
  - Must worry about continuity at join points including continuity of derivatives



Cubic Parametric Polynomials

- Give balance between ease of evaluation and flexibility in design

$$p(u) = \sum_{k=0}^{3} u^k c_k$$

- Four coefficients to determine for each of x, y, and z

- Seek four independent conditions for various values of u resulting in 4 equations in 4 unknowns for each of x, y, and z

- Conditions are a mixture of continuity requirements at the join points and conditions for fitting the data

Cubic Polynomial Surfaces

$$p(u,v) = [x(u,v) \quad y(u,v) \quad z(u,v)]^T$$

where

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

- Need 48 coefficients (3 independent sets of 16) to determine a surface patch

Designing Parametric Cubic Curves

- Introduce the types of curves
  - Interpolating
  - Hermite
  - Bezier
  - B-spline
- Analyze their performance

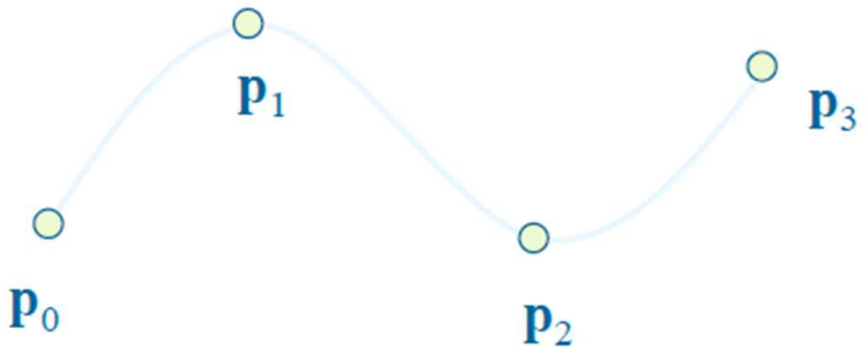
Matrix Vector Form

$$p(u) = c_0 + c_1 u + c_2 u^2 + c_3 u^3 = \sum_{k=0}^{3} u^k c_k$$

$$c = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \qquad u = \begin{bmatrix} 1 \\ u \\ u^2 \\ u^3 \end{bmatrix} \qquad c_k = \begin{bmatrix} c_{kx} \\ c_{ky} \\ c_{kz} \end{bmatrix}$$

$$p(u) = u^T c$$

Interpolating Curve



- Given four data (control) points p_0, p_1, p_2, p_3 determine cubic $\boldsymbol{p}(u)$ which passes through them

- Must find c_0, c_1, c_2, c_3

Interpolation Equations

- Decide at which values of the parameter u the interpolation takes place. we can take these values to be equally spaced $u = 0, \frac{1}{3}, \frac{2}{3}, 1$

$$p_0 = p(0) = c_0$$

$$p_1 = p\left(\frac{1}{3}\right) = c_0 + \frac{1}{3}c_1 + (\frac{1}{3})^2 c_2 + (\frac{1}{3})^3 c_3$$

$$p_2 = p\left(\frac{2}{3}\right) = c_0 + \frac{2}{3}c_1 + (\frac{2}{3})^2 c_2 + (\frac{2}{3})^3 c_3$$

$$p_3 = p(1) = c_0 + c_1 + c_2 + c_3$$

- Or in matrix form $p = Ac$, where $p = [p_0 \quad p_1 \quad p_2 \quad p_3]^T$

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & (\frac{1}{3})^2 & (\frac{1}{3})^3 \\ 1 & \frac{2}{3} & (\frac{2}{3})^2 & (\frac{2}{3})^3 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Interpolation Matrix

- Solving for c we find the interpolation matrix

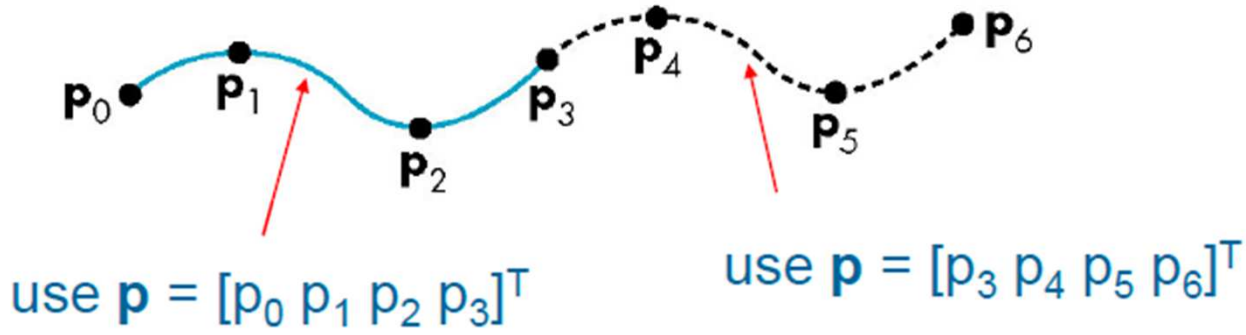
$$M_I = A^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -5.5 & 9 & -4.5 & 1 \\ 9 & -22.5 & 18 & -4.5 \\ -4.5 & 13.5 & -13.5 & 4.5 \end{bmatrix}$$

$$c = M_I p$$

- Note that M_I does not depend on input data and can be used for each segment in x, y, and z

Interpolating Multiple Segments

- Get continuity at join points but not continuity of derivatives



Blending Functions

- Obtain insights into smoothness of interpolating polynomial by rewriting the equation for $p(u)$

$$p(u) = u^T c = u^T M_I p = b(u)^T p$$

where

$$b(u) = [b_0(u) \quad b_1(u) \quad b_2(u) \quad b_3(u)]^T$$

$$b_0(u) = -4.5(u - \frac{1}{3})(u - \frac{2}{3})(u - 1)$$

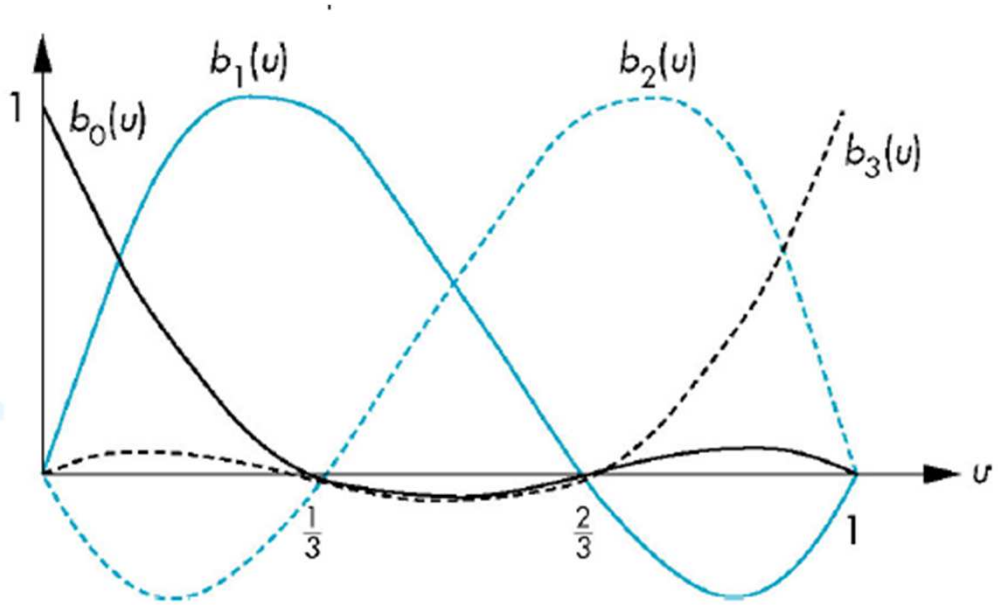
$$b_1(u) = 13.5u(u - \frac{2}{3})(u - 1)$$

$$b_2(u) = -13.5u(u - \frac{1}{3})(u - 1)$$

$$b_3(u) = 4.5u(u - \frac{1}{3})(u - \frac{2}{3})$$

Blending Functions

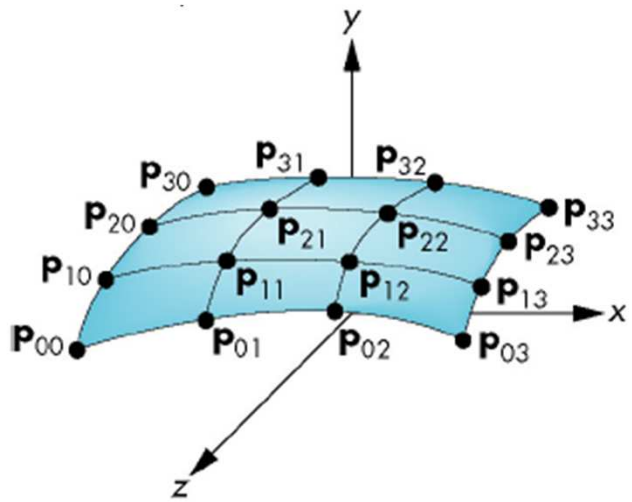
- These functions are not smooth
  - Hence the interpolation polynomial is not smooth
  - The lack of derivatives continuity at the join points account for limited use of interpolating polynomials



Interpolating Patch

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

- Need 16 conditions to determine the 16 coefficients c_{ij}, choose at $u, v = 0, \frac{1}{3}, \frac{2}{3}, 1$



Matrix Form

- Define $v = [1 \quad v \quad v^2 \quad v^3]^T$

$$C = [c_{ij}],\ P = [p_{ij}]$$

$$p(u, v) = u^T C v$$

- For example, $P_{00} = [1 \quad 0 \quad 0 \quad 0] C \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = c_{00}$

- We can compute

$$C = M_I P M_I^T$$

$$p(u, v) = u^T M_I P M_I^T v$$

Blending Patches

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

- Each $b_i(u)b_j(v)$is a blending patch

- Shows that we can build and analyze surfaces from our knowledge of curves

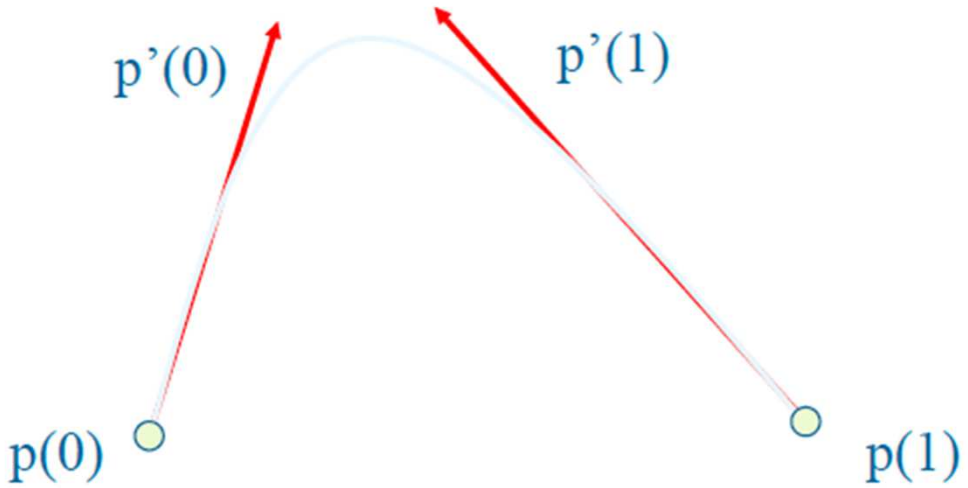
Other Curves and Surfaces

- How can we get around the limitations of the interpolating form
  - Lack of smoothness
  - Discontinuous derivatives at join points

- We have four conditions (for cubic) that we can apply to each segment
  - Use them other than for interpolation
  - Need only come close to the data

Hermit Curves and Surfaces

- Use two interpolating conditions and two derivative conditions per segment
- Ensure continuity and first derivative continuity between segments



Equations

- Interpolating conditions are the same at ends

$$p_0 = p(0) = c_0$$
$$p_3 = p(1) = c_0 + c_1 + c_2 + c_3$$

- Differentiating we find $p'(u) = c_1 + 2uc_2 + 3u^2c_3$

- Evaluating at end points

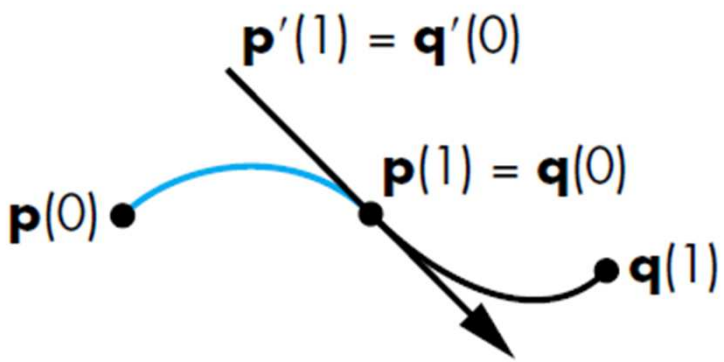
$$p'_0 = p'(0) = c_1$$
$$p'_3 = p'(1) = c_1 + 2c_2 + 3c_3$$

Matrix Form

$$q = \begin{bmatrix} p_0 \\ p_3 \\ p'_0 \\ p'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix} c$$

- Solving, we find $c = M_H q$ where M_H is the Hermite matrix

$$M_H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}$$



Blending Polynomials

$$p(u) = b(u)^T q$$

$$b(u) = \begin{bmatrix} 2u^3 - 3u^2 + 1 \\ -2u^3 - 3u^2 \\ u^3 - 2u^2 + u \\ u^3 - u^2 \end{bmatrix}$$

- Although these functions are smooth, the Hermite form is not used directly in Computer Graphics and CAD because we usually have control points but not derivatives

- However, the Hermite form is the basis of the Bezier form

Geometric and Parametric Continuity

- All three parametric components must be equal at joint points

$$p(1) = \begin{bmatrix} p_x(1) \\ p_y(1) \\ p_z(1) \end{bmatrix} = q(0) = \begin{bmatrix} q_x(0) \\ q_y(0) \\ q_z(0) \end{bmatrix}$$

- We call this property C^0 **parametric continuity**
- Consider derivatives

$$p'(1) = \begin{bmatrix} p'_x(1) \\ p'_y(1) \\ p'_z(1) \end{bmatrix} = q'(0) = \begin{bmatrix} q'_x(0) \\ q'_y(0) \\ q'_z(0) \end{bmatrix}$$

- We have C^1 **parametric continuity** if parametric and derivaitives match
- If derivatives are proportional $\mathrm{p}'(1) = \alpha q'(0)$. Same direction but different magnitude then we have G^1 **geometric continuity**

Higher Dimension

- The techniques for both interpolating and Hermite curves can be used with higher dimensional parametric polynomials
- For interpolating form, the resulting matrix becomes increasingly more ill-conditioned and the resulting curves less smooth and more prone to numerical errors
- In both cases, there is more work in rendering the resulting polynomial curves and surfaces

Bezier and B-Spline

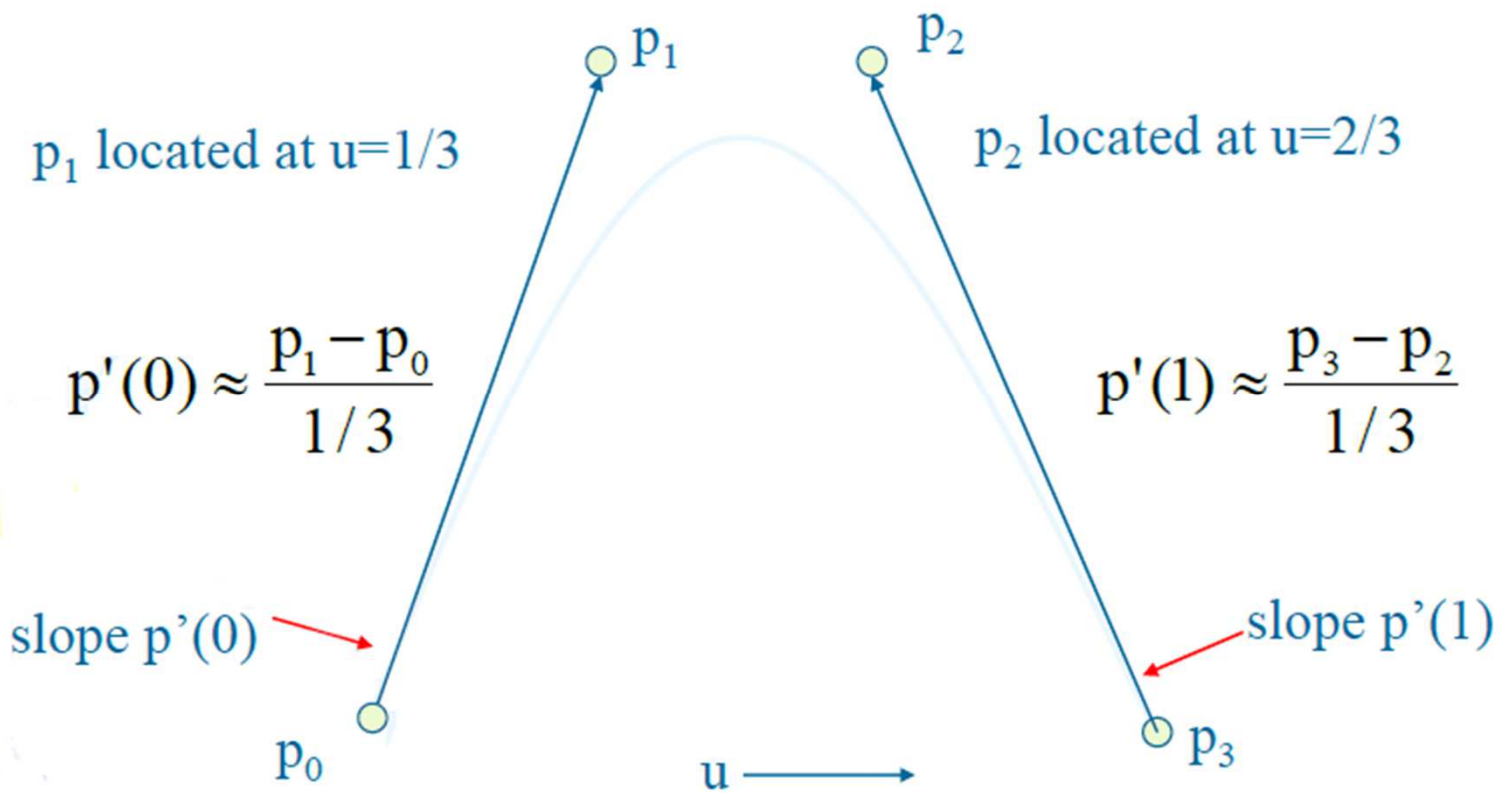
- Very popular representation of surfaces and curves
- More practical
- Available in CAD software

Bezier's Idea

- In graphics and CAD, we do not usually have derivative data

- Bezier suggested using the same 4 data points as with the cubic interpolating curve to approximate the derivatives in the Hermite form

Approximating Derivatives



Equations

- Interpolating conditions are the same

$$p_0 = p(0) = c_0$$
$$p_3 = p(1) = c_0 + c_1 + c_2 + c_3$$

- Approximating derivative conditions

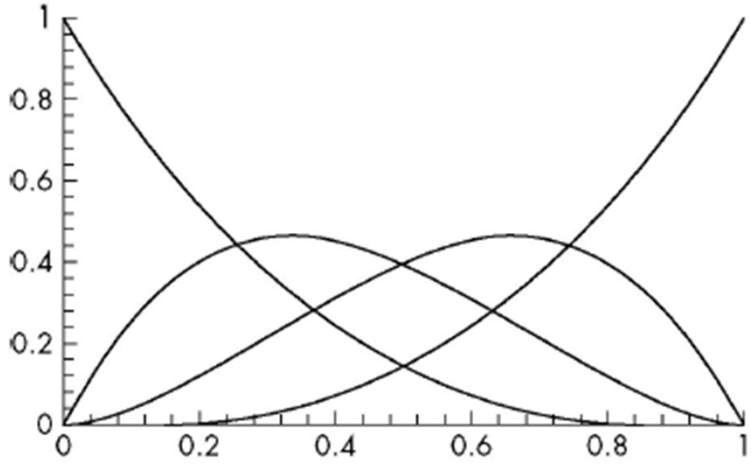
$$p'(0) \approx \frac{p_1 - p_0}{\frac{1}{3}} = 3(p_1 - p_0) = c_1$$
$$p'(1) \approx \frac{p_3 - p_2}{\frac{1}{3}} = 3(p_3 - p_2) = c_1 + 2c_2 + 3c_3$$

- Solve four linear equations for c $= M_B p$

Bezier Matrix

$$M_B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

- $p(u) = u^T M_B p = b(u)^T q$



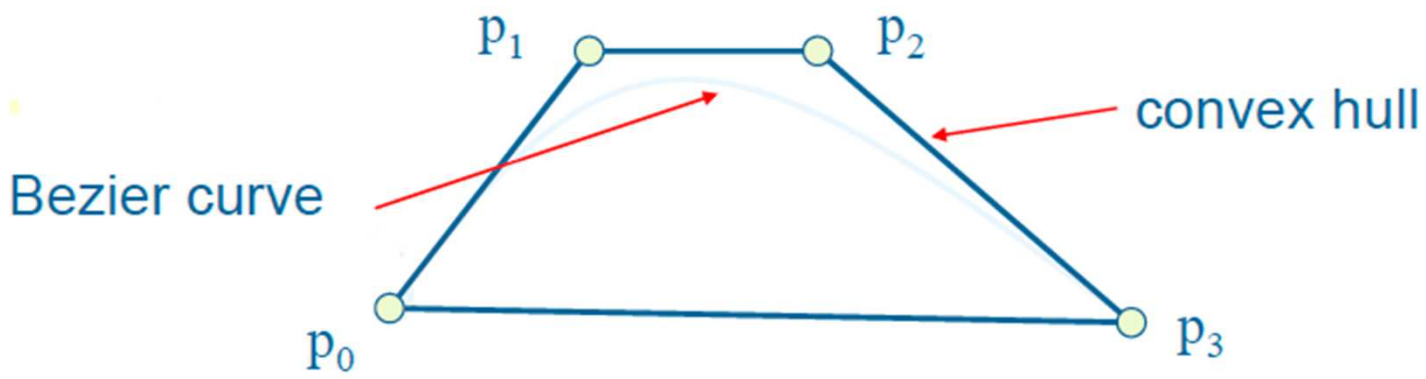
- $b(u) = \begin{bmatrix} (1-u)^3 \\ 3u(1-u)^2 \\ 3u^2(1-u) \\ u^3 \end{bmatrix}$

- Note that all zeros are either at 0 and 1 which forces the functions to be smooth over (0,1)

Convex Hull Property

- All Bezier curves lie in the convex hull of their control points

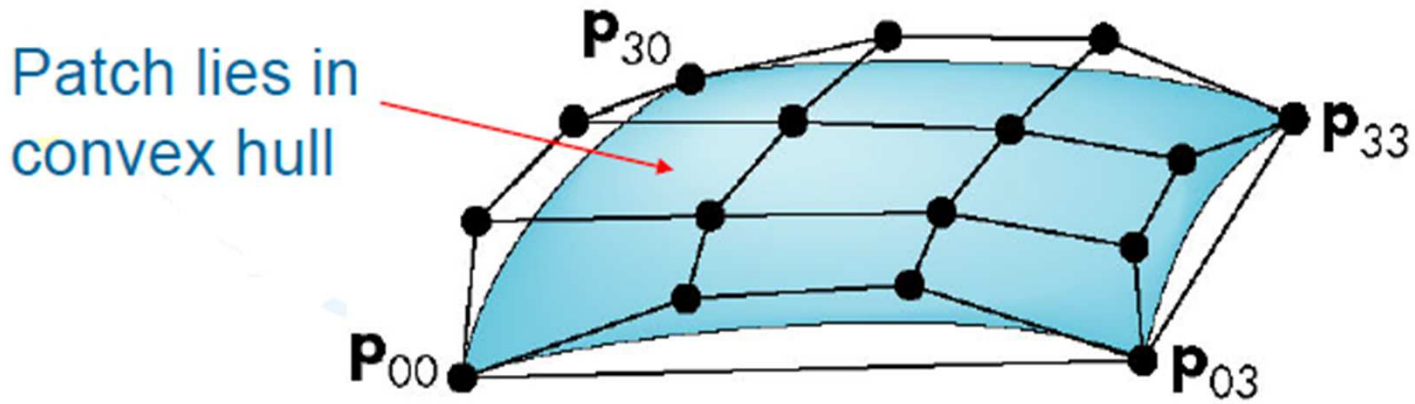
- Hence, even though we do not interpolate all the data, we cannot be too far away



Bezier Patches

- Using same data array $P = [p_{ij}]$ as with interpolating form

$$p(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} b_i(u)b_j(v)p_{ij} = u^T M_B P M_B^T v$$



Analysis

- Although the Bezier form is much better than the interpolating form, we have the derivatives are not continuous at join points

- Can we do better?
- Go to higher order Bezier
  - More work
  - Derivative continuity still only approximate
  - Supported by OpenGL

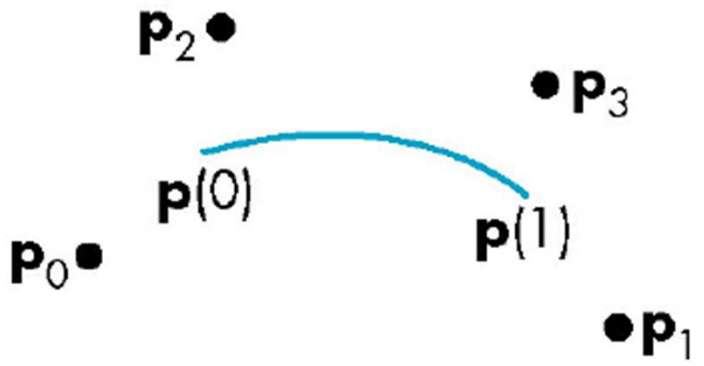
Cubic B-Splines

- Basis splines: use the data at $\mathrm{P} = [p_{i-2} \quad p_{i-1} \quad p_i \quad p_{i+1}]^T$ to define curve only between p_{i-1} and p_i

- Allows us to apply more continuity conditions to each segment

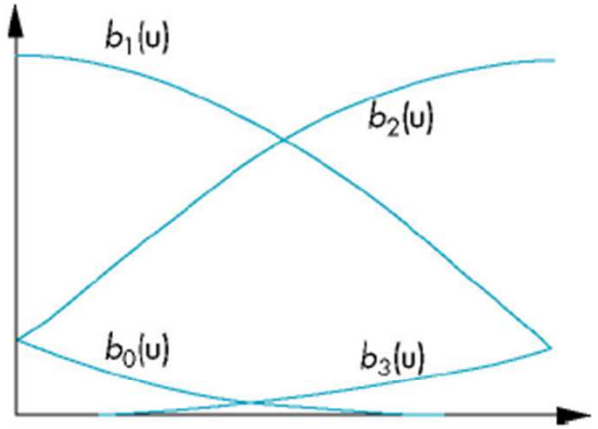
$$p(u)=u^T M_S p = b(u)^T p$$

- $M_S = \frac{1}{6}\begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$

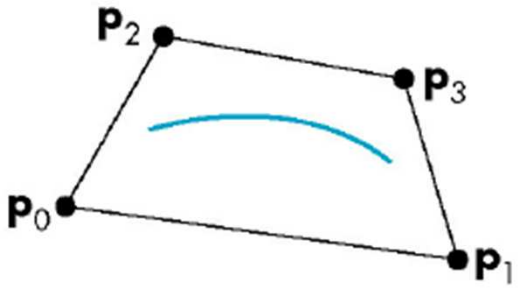


Blending Functions

- $b(u) = M_S^T u = \frac{1}{6}\begin{bmatrix} (1-u)^3 \\ 4 - 6u^2 + 3u^3 \\ 1 + 3u + 3u^2 - 3u^3 \\ u^3 \end{bmatrix}$

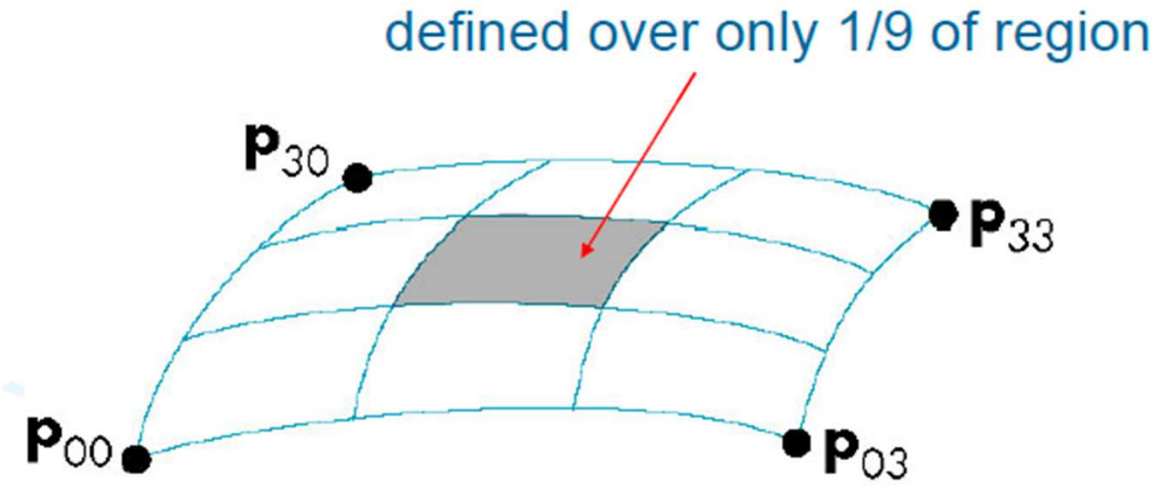


convex hull property



B-Spline Patches

$$p(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} b_i(u)b_j(v)p_{ij} = u^T M_S P M_S^T v$$



Splines and Basis

- If we examine the cubic B-spline from the perspective of each control (data) point, each interior point contributes (through the blending functions) to four segments

- We can rewrite p(u) in terms of the data points as

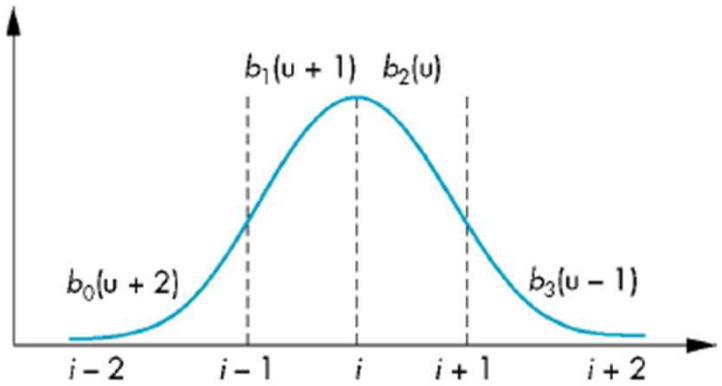
$$p(u) = \sum_{i=0}^{m-1} B_i(u)\, p_i$$

- defining the basis functions $\{B_i(u)\}$

Basis Functions

- In terms of the blending polynomials

$$B_i(u) = \begin{cases} 0 & u < i-2 \\ b_0(u+2) & i-2 \le u < i-1 \\ b_1(u+1) & i-1 \le u < i \\ b_2(u) & i \le u < i+1 \\ b_3(u-1) & i+1 \le u < i+2 \\ 0 & u \ge i+2 \end{cases}$$



NURBS

- Nonuniform Rational B-Spline curves and surfaces add a fourth variable w to x, y, and z
  - Can interpret as weight to give more importance to some control data

$$p(u) = \sum B_i(u)\, w_i p_i$$

- Requires a perspective division
  - NURBS act correctly for perspective viewing

- Quadrics are a special case of NURBS