Curves

- Representations (Explicit, Implicit, Parametric)
- Cubic polynomial form
- Interpolating curves
- Hermite Curves
- Bezier Curves

Motivation

- Problems with series of points used to model a curve
  - large amounts of co-ordinate and connectivity information
  - difficult to exactly design smoothly varying shapes
  - piecewise linear - Does not accurately model a smooth line
  - it's tedious
  - expensive to manipulate curve because all points must be repositioned
- Need a compact and easily manipulated high-level representation of smoothly varying curves and surfaces
- Must be internally convertible to line segment or polygon approximation for polygon-scan conversion rendering

Goals

- How do we specify a surface?
  - Explicit, implicit, parametric
- How do we approximate a surface?
  - Interpolation (use only points)
  - Hermite (use points and tangents)
  - Bezier (use points, and more points for tangents)

Explicit Representation

- Curve in 2D: $y = f(x)$
- Curve in 3D: $y = f(x)$, $z = g(x)$
- Surface in 3D: $z = f(x,y)$
- Problems:
  - How about a vertical line x = c as y = f(x)?
  - Circle $y = \pm(r^2 - x^2)^{1/2}$ two or zero values for x
- Too dependent on coordinate system
- Rarely used in computer graphics

Implicit Representation

- Curve in 2D: $f(x,y) = 0$
  - Line: $ax + by + c = 0$
  - Circle: $x^2 + y^2 - r^2 = 0$
- Surface in 3d: $f(x,y,z) = 0$
  - Plane: $ax + by + cz + d = 0$
  - Sphere: $x^2 + y^2 + z^2 - r^2 = 0$
- f(x,y,z) can describe 3D object:
  - Inside: $f(x,y,z) < 0$
  - Surface: $f(x,y,z) = 0$
  - Outside: $f(x,y,z) > 0$

Algebraic Surfaces

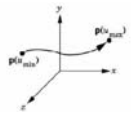
- Special case of implicit representation
- $f(x,y,z)$ is polynomial in x, y, z
- Quadrics: degree of polynomial <= 2
- Render more efficiently than arbitrary surfaces
- Implicit form often used in computer graphics
- How do we represent curves implicitly?

Implicit and Explicit Representations

- Weaknesses:
  - may have too many or too few solutions
  - affine transformations can be difficult
  - vertical tangents (infinite slope) present problems
  - difficult to join separate curves together smoothly
- Strengths:
  - good for point on curve tests

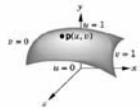
Parametric Form

- Curves: use single parameter u (e.g. time)
  - *P(u) = (x(u), y(u), z(u))*
  - *x = x(u), y = y(u), z = z(u)*
  - example: line:
    - $P(u) = (1-u).P_0 + u.P_1$
    - $x = (1-u).x_0 + u.x_1$
    - $y = (1-u).y_0 + u.y_1$
    - $z = (1-u).z_0 + u.z_1$
  - Describes surface as u varies



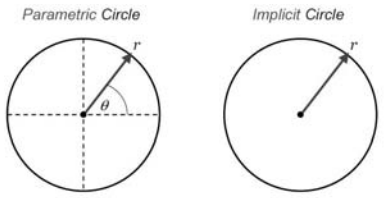
A curve segment

- Surfaces: Use two parameters u and v
  - *P(u, v) = (x(u, v), y(u, v), z (u, v))*
  - *x = x(u,v), y = y(u,v), z = z(u,v)*
  - Describes surface as both u and v vary



A surface patch

Parametric vs. Implicit



$$x(\theta) = r\cos\theta$$
$$y(\theta) = r\sin\theta$$

$$F(x,y) = x^2 + y^2 - r^2 = 0$$

inside: $F < 0$

outside: $F > 0$

Parametric Form

- Strengths:
  - parameters often have natural meaning
  - calculating slope
  - generating points on the curve
  - joining separate curves together
  - easier to apply affine transformations
- Weaknesses:
  - point on curve tests are difficult

Parametric Curves

described in general by a vector-valued function

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

we can define unit tangent and normal vectors as follows

$$t(u) = \frac{p'(u)}{|p'(u)|} \qquad n(u) = \frac{p''(u)}{|p''(u)|}$$

$$p'(u) = \frac{d}{du}p(u) = \begin{bmatrix} x'(u) \\ y'(u) \\ z'(u) \end{bmatrix}$$

Parametric Surfaces

described in general by two varying parameters

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

Partial derivatives describe tangent plane at each point

$$\frac{d}{du}p(u,v) = \begin{bmatrix} \frac{dx(u,v)}{du} \\ \frac{dy(u,v)}{du} \\ \frac{dz(u,v)}{du} \end{bmatrix} \qquad \frac{d}{dv}p(u,v) = \begin{bmatrix} \frac{dx(u,v)}{dv} \\ \frac{dy(u,v)}{dv} \\ \frac{dz(u,v)}{dv} \end{bmatrix}$$

Parametric Polynomial Curves

- Restrict x(u), y(u), z(u) to be polynomial in u
- Fix degree n

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

- each c_k is a column vector

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

Approximating Curves

- Input: some control points
- output: a parametric form curve

- Important concepts:
  - Join points for segments and patches
  - Control points to interpolate
  - Tangents and smoothness
  - Blending functions to describe interpolation

Cubic Polynomial Form

- Degree 3 appears to be a useful compromise

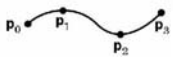
$$p(u) = \sum_{k=0}^{3} c_k u^k = c_0 + c_1 u + c_2 u^2 + c_3 u^3$$

- Each c_k is a column vector $[c_{xk}\ c_{yk}\ c_{zk}]^T$
- From control information (points, tangents) we can derive 12 values for c_{xk}, c_{yk}, c_{zk}
- These values determine cubic polynomial form

Interpolation

- Simplest case, although rarely used
- Curves:
  - Given 4 control points p_0, p_1, p_2, p_3
  - All should lie on curve (cubic interpolation segment)
  - 12 conditions, 12 unknowns

- Space values for parameter $0 \le u \le 1$ evenly: 0, 1/3, 2/3, 1
- $p_0 = p(0)$, $p_1 = p(1/3)$, $p_2 = p(2/3)$, $p_3 = p(1)$



a cubic interpolating curve

Interpolation (2)

To determine the curve, we need to determine the How to determine values for c_k

$$p(u) = c_0 + c_1 u + c_2 u^2 + c_3 u^3 \Rightarrow$$
$$p_0 = p(0) = c_0$$
$$p_1 = p(\tfrac{1}{3}) = c_0 + \tfrac{1}{3}c_1 + (\tfrac{1}{3})^2 c_2 + (\tfrac{1}{3})^3 c_3$$
$$p_2 = p(\tfrac{2}{3}) = c_0 + \tfrac{2}{3}c_1 + (\tfrac{2}{3})^2 c_2 + (\tfrac{2}{3})^3 c_3$$
$$p_3 = p(1) = c_0 + c_1 + c_2 + c_3$$
$$\Rightarrow$$
$$\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & (\frac{1}{3})^2 & (\frac{1}{3})^3 \\ 1 & \frac{2}{3} & (\frac{2}{3})^2 & (\frac{2}{3})^3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

note: p_k and c_k are vectors

Interpolation (3)

$$p = Ac \Rightarrow c = A^{-1} p$$

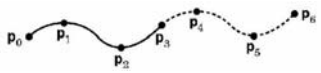
invert A to obtain interpolating geometry matrix :

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -5.5 & 9 & -4.5 & 1 \\ 9 & -22.5 & 18 & 4.5 \\ -4.5 & 13.5 & -13.5 & 4.5 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Again note that p_k and c_k are vectors!

Joining Interpolating Segments

- Do not solve degree n for n points
- Divide into overlap sequences of 4 points
- p_0, p_1, p_2, p_3 then p_3, p_4, p_5, p_6, etc.



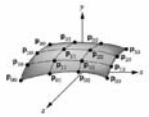
- At join points
  - Will be continuous (C^0 continuity)
  - Derivatives will usually not match (no C^1 continuity)

The Cubic Interpolation Patch

- There is a natural extension of the interpolating curve to an interpolating patch
- A Bicubic surface patch can be written in the form:

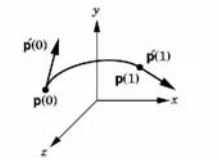
$$p(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} u^i v^j c_{ij} \quad \text{where}: c_{ij} = \begin{bmatrix} c_{ijx} \\ c_{ijy} \\ c_{ijz} \end{bmatrix}$$

- We have 16 three-dimensional control points, so we get three sets (x, y, z) of 16 equations in 16 unknowns!



Hermite Curves

- Another cubic polynomial curve
- Specify
  - two endpoints
  - and their tangents



Deriving the Hermite Form

- We have two points p(0) and p(1), we use p_0 and p_3 for numbering to be consistent with the interpolation notion

As before:

$$p(u) = c_0 + c_1 u + c_2 u^2 + c_3 u^3 \Rightarrow \begin{aligned} p_0 &= p(0) = c_0 \\ p_3 &= p(1) = c_0 + c_1 + c_2 + c_3 \end{aligned}$$

Calculate derivative:

$$p'(u) = c_1 + 2c_2 u + 3c_3 u^2 \quad \Rightarrow \begin{aligned} p'_0 &= p'(0) = c_1 \\ p'_3 &= p'(1) = c_1 + 2c_2 + 3c_3 \end{aligned}$$

we got the four equations we needed

Deriving the Hermite Form (2)

- Write in matrix form
- Remember p_k and p'_k and c_k are vectors!!!

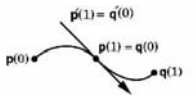
$$\begin{bmatrix} p_0 \\ p_3 \\ p'_0 \\ p'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \quad \text{or} \quad q = Ac \Rightarrow c = A^{-1}q$$

- we call $A^{-1} M_H$, the Hermite geometry matrix

$$c = M_H q \quad \text{where}: M_H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}$$

Joining Hermite Curves

- Match points and tangents (derivates)



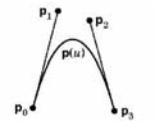
- Much smoother than point interpolation

Bezier Curves

- Widely used in computer graphics
- Approximate tangents by using control points

$$p'(0) = \frac{p_1 - p_0}{\frac{1}{3}} = 3(p_1 - p_0)$$

$$p'(1) = \frac{p_3 - p_2}{\frac{1}{3}} = 3(p_3 - p_2)$$



Equations for Bezier Curves

- Set up equations for cubic parametric curve

As before:

$$p(u) = c_0 + c_1 u + c_2 u^2 + c_3 u^3 \Rightarrow \begin{array}{l} p(0) = c_0 \\ p(1) = c_0 + c_1 + c_2 + c_3 \end{array}$$

Calculate derivative:

$$p'(u) = c_1 + 2c_2 u + 3c_3 u^2 \quad \Rightarrow \begin{array}{l} p'(0) = 3p(1) - 3p(0) = c_1 \\ p'(1) = 3p(3) - 3p(2) = c_1 + 2c_2 + 3c_3 \end{array}$$

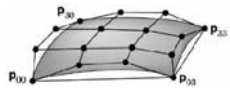
Equations for Bezier Curves (2)

- Calculate Bezier geometry matrix (M_B)

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = M_B \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \text{where}: M_B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 1 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

Bezier Surface Patches

- We can generate the Bezier surface patches having a 4x4 array of control points:



Summary

- Parametric Representations
- Cubic Polynomial Forms
- Interpolation
- Hermite Curves
- Bezier Curves and Surfaces