
Curves

Curves I

Curves

Representations of Curves and Surfaces

Hermite Curves

Bezier Curves

Notice: A program called curves is available

- On class web site
- You can run this and try different curve ideas
- You can also examine the source to see how it works.

Discussion

Why Curves?

- Lines and surfaces...

Why not just tessellate?

Curve advantages

Scale independence

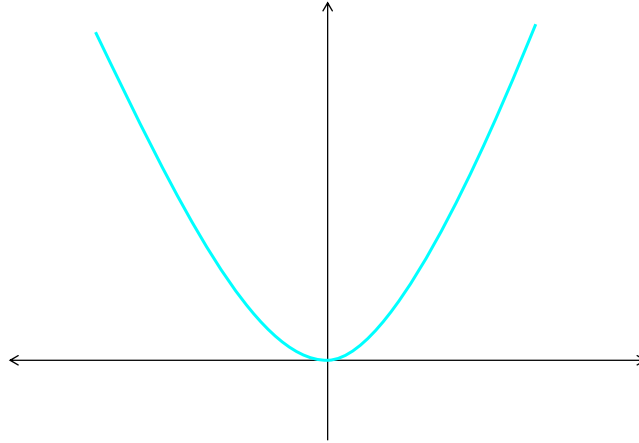
- You can tessellate when you know the actual display scale

Natural description

Great for simulating continuous motion!

- Curves between fixed control points...

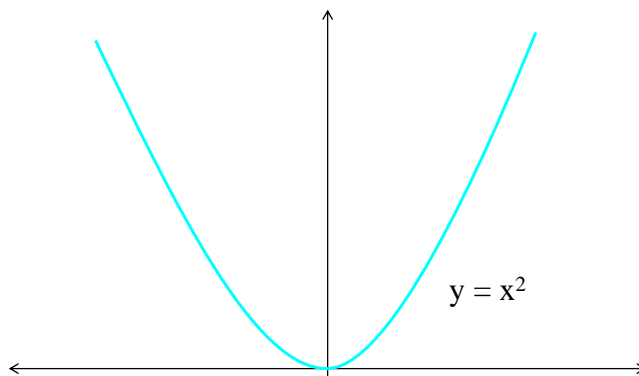
How would you describe this?



CSE 472 Spring 2009

5

Explicit Representation



Explicit representation –
represents a curve with one
variable as a function of one or
more other variables. $y=f(x)$

CSE 472 Spring 2009

6

Explicit Representation

2D

- $y = f(x)$ (or $x = f(y)$)

3D

- $z = f(x, y)$

2D Line

- $y = mx + b$

What's the representation for a 3D line?

Explicit falls apart

3D Line requires two equations

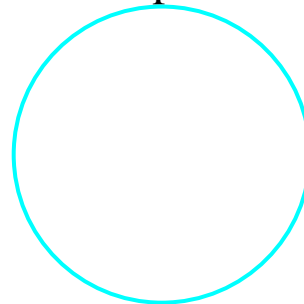
- $y = f(x)$

- $z = f(x)$

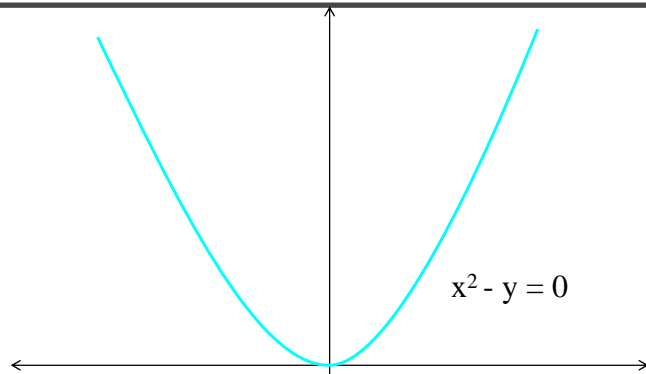
- But, what about a line in a plane parallel to yz ?

What about a circle?

- $y = \sqrt{r^2 - x^2}$



Implicit Representation



Implicit representation –
represents a curve as a function
of all variables equal zero:

$$f(x, y) = 0$$

Implicit Representation

2D

- $f(x, y) = 0$
- $ax + by + c = 0$ Line
- $x^2 + y^2 - r^2 = 0$ Circle

3D

- $f(x, y, z) = 0$
- $ax + by + cz + d = 0$ Plane
- $x^2 + y^2 + z^2 - r^2 = 0$ Sphere



Problems with Implicit Form

The function is really a membership test

- $f(x, y) = 0$ Does $f(x,y) = 0$ for a point?
- Impractical to test points

Parametric Representation

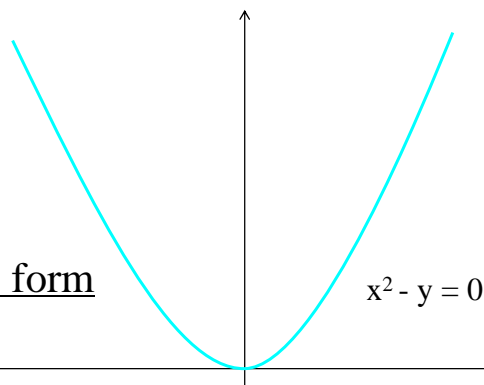
We introduce a new, artificial parameter

One equation for each dimension

- $x = x(u), y = y(u)$
- Example:
- $y = u^2$
- $x = u$

Also called:

- parametric form



Example: A Line

Let $ax + by + c = 0$ describe a line.

- $x = -cu/a$
- $y = c(u-1)/b$
- (Unless horizontal or vertical)
 - Vertical ($ax + c = 0$): $x = -c/a$, $y = u$
 - Horizontal (you figure it out...)



Don't believe me?

$ax + by + c = 0$ describes the line

I said:

- $x = -cu/a$
- $y = c(u-1)/b$

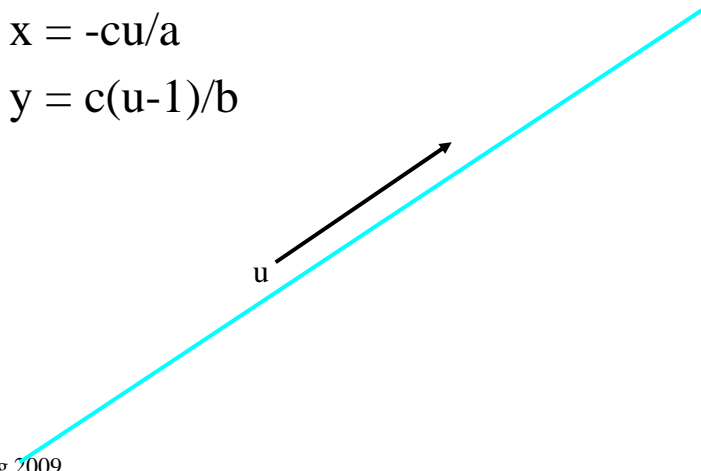
Plug them in:

- $a(-cu/a) + b(c(u-1)/b) + c = 0$
- $-cu + cu - c + c = 0$

What does u represent?

For a line:

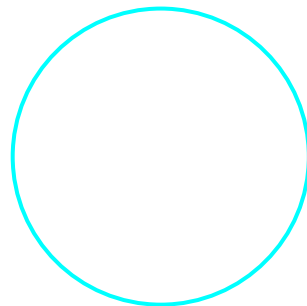
- $x = -cu/a$
- $y = c(u-1)/b$



Circle?

Assume $x^2 + y^2 - r^2 = 0$

Can you come up with a parametric representation?



How about this one...

$$x^2 + y^2 - r^2 = 0$$

- $x = r \sin(u)$

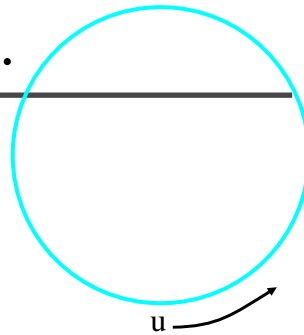
- $y = r \cos(u)$

Plug it in:

- $r^2 \sin(u)^2 + r^2 \cos(u)^2 - r^2 = 0$

- $r^2 (\sin(u)^2 + \cos(u)^2) - r^2 = 0$

- $r^2 - r^2 = 0$



Parametric Form in 3D

Single variable (lines)

- $x = f_x(u)$

- $y = f_y(u)$

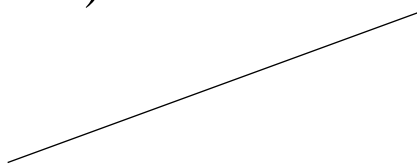
- $z = f_z(u)$

Two variables (surfaces)

- $x = f_x(u, v)$

- $y = f_y(u, v)$

- $z = f_z(u, v)$



Are parametric forms unique?

Line

- $x = -cu/a$
- $y = c(u-1)/b$

Circle

- $x = r \sin(u)$
- $y = r \cos(u)$

Notation

It's common to express a parametric equation in terms of vectors:

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

Segments

We can scale u and v any way we want:

- $f(u)$ or $f(2u)$ are both parametric equations

Generally, we are describing *segments* not infinite curves or lines

- We are happy to describe one loop around the circle
- $x = \sin(2\pi u)$
- $y = \cos(2\pi u)$
- What about a line segment!

A line segment

Line from (x_1, y_1) to (x_2, y_2)

- $x = x_1 + u(x_2 - x_1)$
- $y = y_1 + u(y_2 - y_1)$

What we're interested in here is *curve segments*

We'll assume u, v in the range

- $0 \leq u \leq 1, 0 \leq v \leq 1$



Polynomials

Polynomials are curved naturally and easy to work with

- $x = c_{x0} + c_{x1}u + c_{x2}u^2 + \dots + c_{xn}u^n$

- In general:

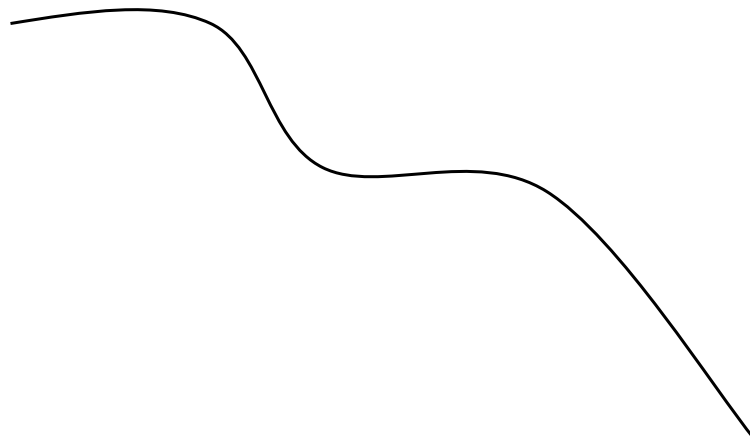
$$p(u) = \sum_{k=0}^n u^k c_k$$

- where:

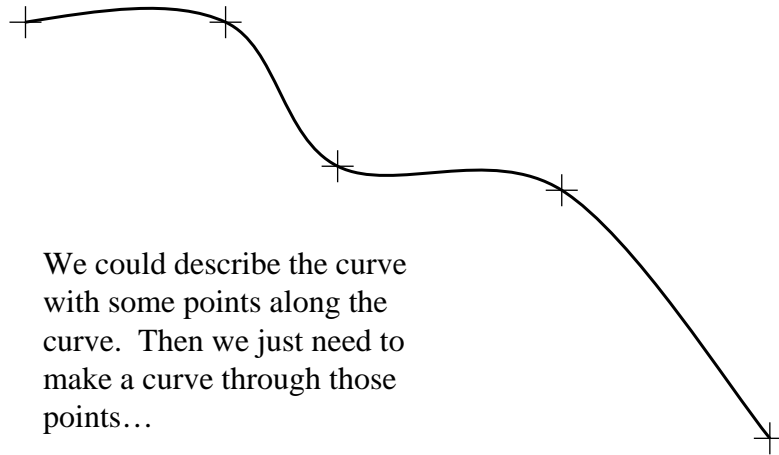
“n + 1 degrees of freedom” or “order n”

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

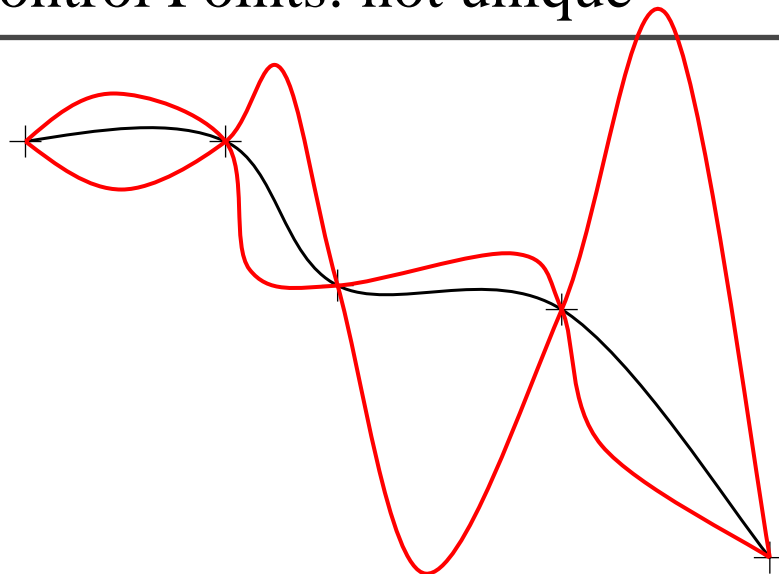
How to describe this curve?



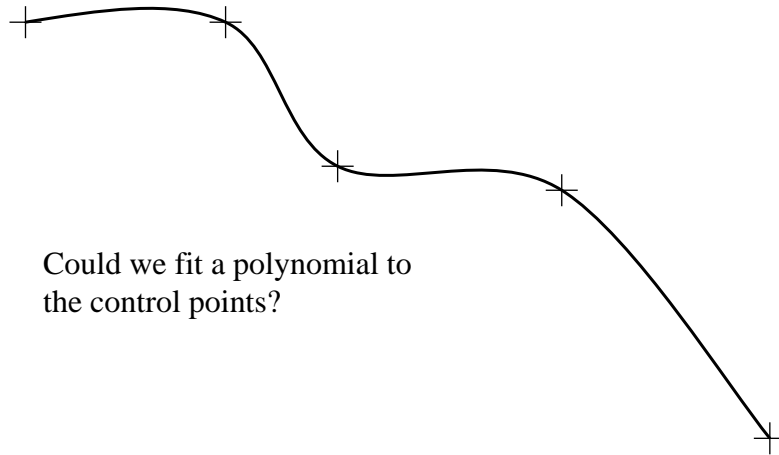
Control Points



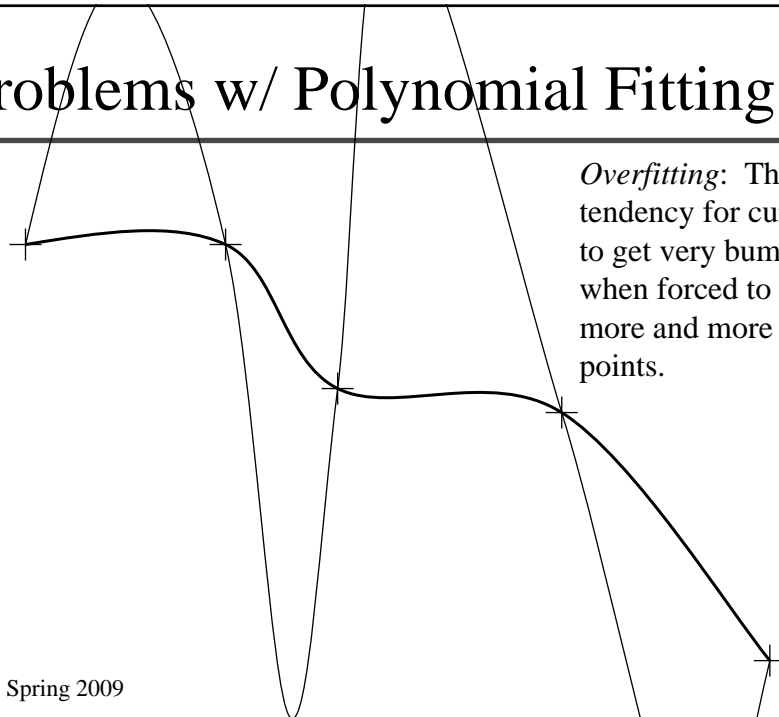
Control Points: not unique



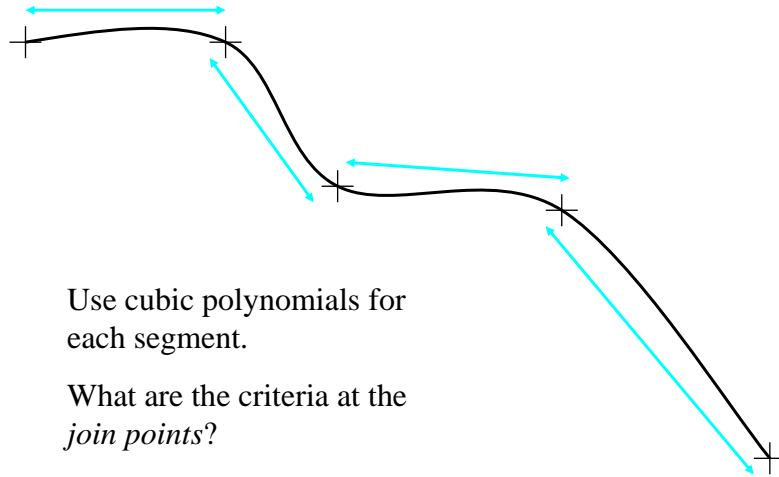
Polynomial Fitting



Problems w/ Polynomial Fitting



Answer: Do segments



Use cubic polynomials for each segment.

What are the criteria at the *join points*?

We love cubics

We'll use cubic polynomials for curve segments

- $x = c_{x0} + c_{x1}u + c_{x2}u^2 + c_{x3}u^3$
- Why cubics?
- Can be expressed as:

$$p(u) = \sum_{k=0}^3 u^k c_k \quad c_k = \begin{bmatrix} c_{kx} \\ c_{ky} \\ c_{kz} \end{bmatrix}$$



Criteria for curve design

Local control of shape

- Control points don't affect the entire curve
- Smoothness
- Continuity
- Stability
- Ease of rendering

Connections at join points...

At the join points:

- G^0 continuity – The ends touch (necessary)
- G^1 continuity – The *slope* at the joint is the same (necessary)
- C^1 continuity – The *first derivative* at the joint is the same (not necessary)



Derivatives

Derivatives are easy for cubics

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

Derivative:

- $p'(u) = c_1 + 2c_2u + 3c_3u^2$



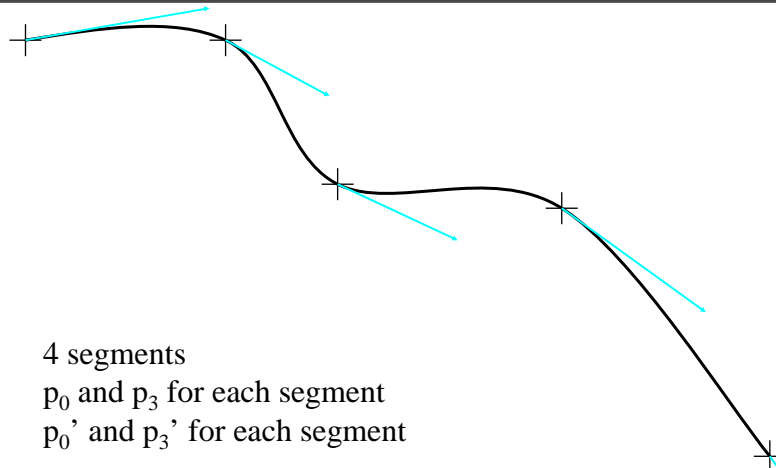
Hermite Curves

One way to design a curve segment:

- End points (p_0 and p_3)
- End tangent vectors (derivatives)
 - p_0' and p_3'
 - 4 parameters and is sufficient to define an $n=3$ polynomial.

This is the Hermite curve.

Specifying a Hermite Curve



Recall

We considered u to be in the range 0 to 1

- Do, $u=0$ is point p_0 and $u=1$ for point p_3 .
- Then $p(0)=p_0$ and $p(1)=p_3$
- And: $p'(0)=p_0'$ and $p'(1)=p_3'$

Cubic equation:

- $x = c_{x0} + c_{x1}u + c_{x2}u^2 + c_{x3}u^3$
- $x(0)=c_{x0}$, $x(1)=c_{x0} + c_{x1} + c_{x2} + c_{x3}$

Using previous notation:

- $p(0)=c_0$, $p(1)=c_0+c_1+c_2+c_3$

Building it up...

Using previous notation:

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

And...

- $p'(u) = c_1 + 2c_2u + 3c_3u^2$
- $p'(0) = c_1$
- $p'(1) = c_1 + 2c_2 + 3c_3$

Note: I have reordered from the conventional presentation...

Write in matrix form:

$$\begin{bmatrix} p_0 \\ p_0' \\ p_3' \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

CSE 472 Spring 2009

Solve for the c's

$$\begin{bmatrix} p_0 \\ p_0' \\ p_3' \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} p_0 \\ p_0' \\ p_3' \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & -1 & 3 \\ 2 & 1 & 1 & -2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_0' \\ p_3' \\ p_3 \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

CSE 472 Spring 2009

```

c0.x = p0.x;
c0.y = p0.y;

c1.x = p0p.x;
c1.y = p0p.y;

c2.x = -3 * p0.x + -2 * p0p.x + -1 * p3p.x + 3 * p3.x;
c2.y = -3 * p0.y + -2 * p0p.y + -1 * p3p.y + 3 * p3.y;

c3.x = 2 * p0.x + 1 * p0p.x + 1 * p3p.x + -2 * p3.x;
c3.y = 2 * p0.y + 1 * p0p.y + 1 * p3p.y + -2 * p3.y;

```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & -1 & 3 \\ 2 & 1 & 1 & -2 \end{bmatrix} \begin{bmatrix} p_0 \\ p_0' \\ p_3 \\ p_3' \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

```

// Now we need to draw the curve
pDC->MoveTo(p0);
double stepsize = 1. / 100.;

for(double t = 0; t<=1.0; t += stepsize)
{
    double x = c0.x + c1.x * t + c2.x * t * t + c3.x * t * t * t;
    double y = c0.y + c1.y * t + c2.y * t * t + c3.y * t * t * t;

    pDC->LineTo(int(x + 0.5), int(y + 0.5));
}

```



CSE 472 Spring 2009

Note: 2D example.³⁹

Bézier Curves

Hermite curves are difficult to specify

- Derivatives are not natural
- Tangent vectors overlap
- Tangent vectors are a bit long for display

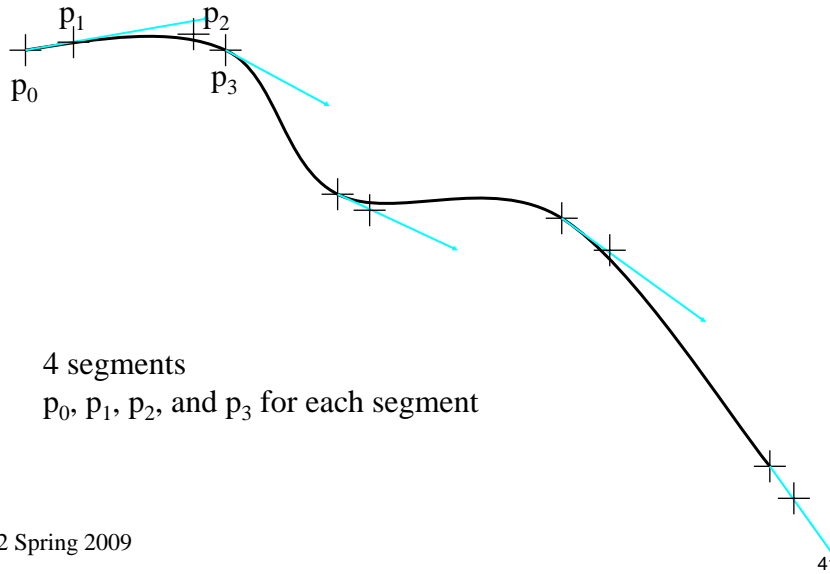
Bézier Curves

- We'll use end points of $1/3$ and $-1/3$ along the tangent vectors as control points.

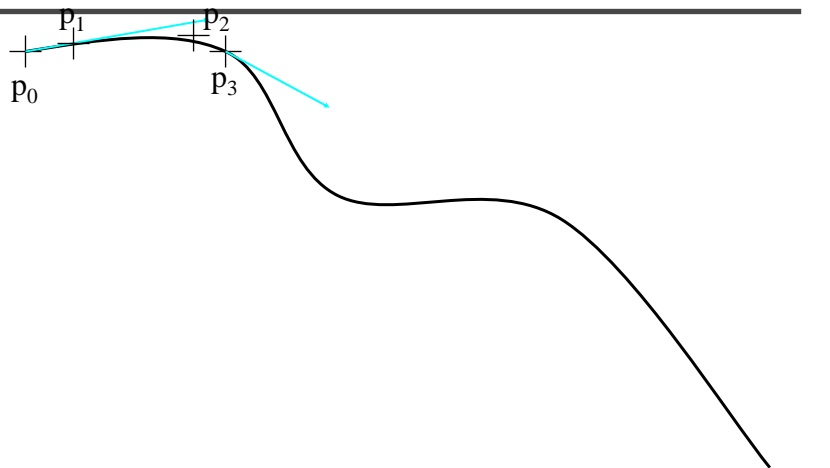
CSE 472 Spring 2009

40

A Bézier Curve



A Bézier Curve



The Control Points

The control points

- $p_0 = p(0)$, First end point
- $p_1 = p_0 + p'(0)/3$
- $p_2 = p_3 - p'(1)/3$
- $p_3 = p(1)$, Second end point

Remember:

- $p(u) = c_0 + c_1u + c_2u^2 + c_3u^3$

So:

- $p_0 = c_0$
- $p_3 = c_0 + c_1 + c_2 + c_3$ (like before)

The Control Points

$$p_0 = c_0$$

$$p_3 = c_0 + c_1 + c_2 + c_3$$

The control points

- $p_0 = p(0)$, First end point
- $p_1 = p_0 + p'(0)/3$
- $p_2 = p_3 - p'(1)/3$
- $p_3 = p(1)$, Second end point

Remember...

- $p'(u) = c_1 + 2c_2u + 3c_3u^2$

So:

- $p_1 = c_0 + 1/3 c_1$
- $p_2 = c_0 + c_1 + c_2 + c_3 - 1/3 (c_1 + 2c_2 + 3c_3)$
 $= c_0 + 2/3 c_1 + 1/3 c_2$

$$\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1/3 & 0 & 0 \\ 1 & 2/3 & 1/3 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Solution for Bézier Curves

$$\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1/3 & 0 & 0 \\ 1 & 2/3 & 1/3 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

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

```

c0.x = p0.x;
c0.y = p0.y;

c1.x = -3 * p0.x + 3 * p1.x;
c1.y = -3 * p0.y + 3 * p1.y;

c2.x = 3 * p0.x + -6 * p1.x + 3 * p2.x;
c2.y = 3 * p0.y + -6 * p1.y + 3 * p2.y;

c3.x = -1 * p0.x + 3 * p1.x + -3 * p2.x + 1 * p3.x;
c3.y = -1 * p0.y + 3 * p1.y + -3 * p2.y + 1 * p3.y;

// Now we need to draw the curve
pDC->MoveTo(p0);

double stepsize = 1. / 100.;

for(double t = 0; t<=1.0; t += stepsize)
{
    double x = c0.x + c1.x * t + c2.x * t * t + c3.x * t * t * t;
    double y = c0.y + c1.y * t + c2.y * t * t + c3.y * t * t * t;

    pDC->LineTo(int(x + 0.5), int(y + 0.5));
}

```

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

The Convex Hull

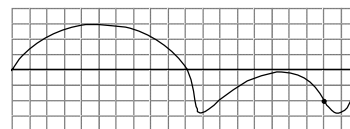
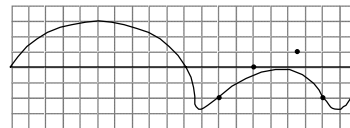
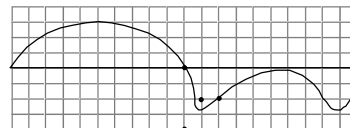
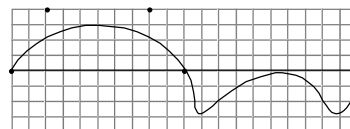
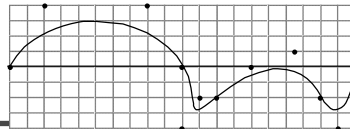
The convex hull of the control points contains the curve!

- This was a major initial design criteria

Some Bézier Examples

These were created using a drawing program called Canvas.

Most commercial drawing programs utilize Bézier curves



Stepping...

You may have noticed this:

```
double stepsize = 1. / 100.;;  
  
for(double t = 0; t<=1.0; t += stepsize)  
{  
    double x = c0.x + c1.x * t + c2.x * t * t + c3.x * t * t * t;  
    double y = c0.y + c1.y * t + c2.y * t * t + c3.y * t * t * t;  
  
    pDC->LineTo(int(x + 0.5), int(y + 0.5));  
}
```

How could we make this more efficient?