1

Curves

Adam Finkelstein
Princeton University
C0S 426, Fall 2001

2

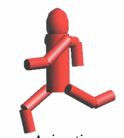
Introduction

- Splines: mathematical way to express curves
- Motivated by “loftsman’s spline”
  - Long, narrow strip of wood/plastic
  - Used to fit curves through specified data points
  - Shaped by lead weights called “ducks”
  - Gives curves that are “smooth” or “fair”
- Have been used to design:
  - Automobiles
  - Ship hulls
  - Aircraft fuselages and wings

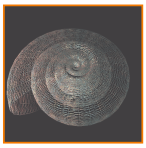
3

Many applications in graphics

- Fonts ABC
- Animation paths
- Shape modeling
- etc…



Animation
(Angel, Plate 1)

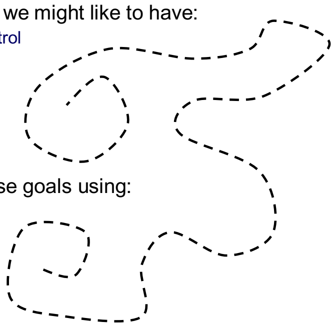


Shell
(Douglas Turnbull, CS 426, Fall99)

4

Goals

- Some attributes we might like to have:
  - Predictable control
  - Multiple values
  - Local control
  - Versatility
  - Continuity
- We’ll satisfy these goals using:
  - Piecewise
  - Parametric
  - Polynomials



5

Parametric curves

A parametric curve in the plane is expressed as:

$x = x(u)$
$y = y(u)$

Example: a circle with radius r centered at origin:

$x = r \cos u$
$y = r \sin u$

In contrast, an implicit representation is:

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

6

Parametric polynomial curves

- A parametric polynomial curve is described:

$$x(u) = \sum_{i=0}^{n} a_i u^i$$

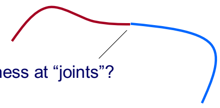
$$y(u) = \sum_{i=0}^{n} b_i u^i$$

- Advantages of polynomial curves
  - Easy to compute
  - Infinitely differentiable

Piecewise parametric polynomials

7

- Use different polynomial functions on different parts of the curve
  - Provides flexibility
  - How do you guarantee smoothness at "joints"? (*continuity*)

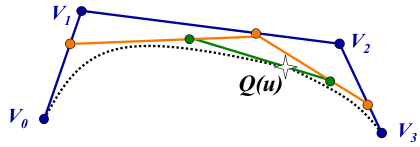


- In the rest of this lecture, we'll look at:
  - Bézier curves: general class of polynomial curves
  - Splines: ways of putting these curves together

Bézier curves

8

- Developed simultaneously in 1960 by
  - Bézier (at Renault)
  - deCasteljau (at Citroen)
- Curve $Q(u)$ is defined by nested interpolation:



V_i's are *control points*
$\{V_0, V_1, \ldots, V_n\}$ is *control polygon*

Basic properties of Bézier curves

9

- Endpoint interpolation:

$$Q(0) = V_0$$
$$Q(1) = V_n$$

- Convex hull:
  - Curve is contained within convex hull of control polygon
- Symmetry

$$Q(u) \text{ defined by } \{V_0,\ldots,V_n\} \equiv Q(1-u) \text{ defined by } \{V_n,\ldots,V_0\}$$

Explicit formulation

10

- Let's indicate level of nesting with superscript j:
- An explicit formulation of $Q(u)$ is given by:

$$V_i^j = (1-u)V_i^{j-1} + uV_{i+1}^{j-1}$$

- Case n=2:

$$\begin{aligned} Q(u) &= V_0^2 \\ &= (1-u)V_0^1 + uV_1^1 \\ &= (1-u)[(1-u)V_0^0 + uV_1^0] + u[(1-u)V_1^0 + uV_2^0] \\ &= (1-u)^2 V_0^0 + 2u(1-u)V_1^0 + u^2 V_2^0 \end{aligned}$$

More properties

11

- General case: Bernstein polynomials

$$Q(u) = \sum_{i=0}^{n} V_i \binom{n}{i} u^i (1-u)^{n-i}$$

- Degree: is a polynomial of degree n
- Tangents:

$$Q'(0) = n(V_1 - V_0)$$
$$Q'(1) = n(V_n - V_{n-1})$$

Cubic curves

12

- From now on, let's talk about cubic curves (n=3)
- In CAGD, higher-order curves are often used
- In graphics, piecewise cubic curves will do it
  - Specify points and tangents
  - Will describe curve in space
- All these ideas generalize to higher-order curves

13

Matrix form

- Bézier curves can also be described in matrix form:

$$Q(u) = \sum_{i=0}^{n} V_i \binom{n}{i} u^i (1-u)^{n-i}$$

$$= (1-u)^3 V_0 + 3u(1-u)^2 V_1 + 3u^2(1-u) V_2 + u^3 V_3$$

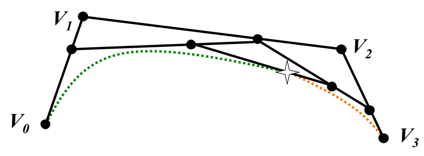
$$= \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \underbrace{\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{M}_{\mathbf{Bezier}}} \begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{pmatrix}$$

14

Display

Q: How would you draw it using line segments?

A: Recursive subdivision!



15

Display

- Here is pseudocode for displaying Bézier curves:

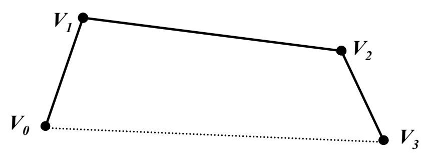
procedure Display($\{V_i\}$):
  if $\{V_i\}$ flat within ε
  then
    output line segment V_0V_n
  else
    subdivide to produce $\{L_i\}$ and $\{R_i\}$
    Display($\{L_i\}$)
    Display($\{R_i\}$)
  end if
end procedure

16

Flatness

Q: How do you test for flatness?

A: Compare the length of the control polygon to the length of the segment between endpoints

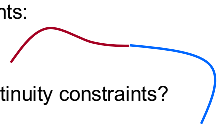


$$\frac{|V_1 - V_0| + |V_2 - V_1| + |V_3 - V_2|}{|V_3 - V_0|} < 1 + \varepsilon$$

17

Splines

- For more complex curves, piece together Béziers
- We want continuity across joints:
  - Positional (C^0) continuity
  - Derivative (C^1) continuity

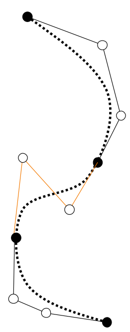


- Q: How would you satisfy continuity constraints?
- Q: Why not just use higher-order Bézier curves?
- A: Splines have several of advantages:
  - Numerically more stable
  - Easier to compute
  - Fewer bumps and wiggles

18

Catmull-Rom splines

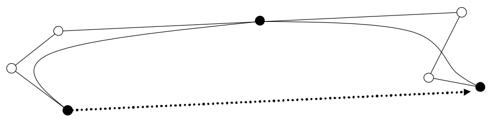
- Properties
  - Interpolate control points
  - Have C^0 and C^1 continuity
- Derivation
  - Start with joints to interpolate
  - Build cubic Bézier between each joint
  - Endpoints of Bézier curves are obvious
- What should we do for the other Bézier control points?



19

Catmull-Rom Splines

- Catmull & Rom use:
  - half the magnitude of the vector between adjacent CP's



- Many other formulations work, for example:
  - Use an arbitrary constant τ times this vector
  - Gives a "tension" control
  - Could be adjusted for each joint

20

Matrix formulation

- Express conversion from Catmull-Rom CP's to Bezier CP's with a matrix:

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \frac{1}{6}\begin{pmatrix} 0 & 6 & 0 & 0 \\ -1 & 6 & 1 & 0 \\ 0 & 1 & 6 & -1 \\ 0 & 0 & 6 & 0 \end{pmatrix}\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{pmatrix}$$

- *Exercise*: Derive this matrix.

21

Properties

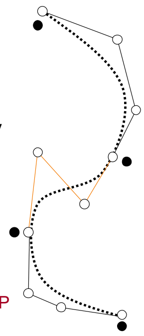
- Catmull-Rom splines have these attributes:
  - C1 continuity
  - Interpolation
  - Locality of control
  - No convex hull property

    (Proof left as an exercise.)

22

B-splines

- We still want local control
- Now we want C^2 continuity
- Give up interpolation
- It turns out we get convex hull property
- Constraints:
  - Three continuity conditions at each joint j
    - » Position of two curves same
    - » Derivative of two curves same
    - » Second derivatives same
  - Local control
    - » Each joint affected by small set of (4) CP



23

Matrix formulation for B-splines

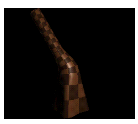
- Grind through some messy math to get:

$$Q(u) = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{pmatrix}$$

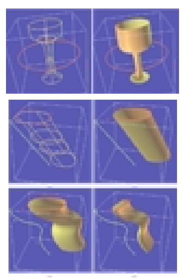
24

What's next?

- Use curves to create parameterized surfaces
- Surface of revolution
- Swept surfaces
- Surface patches



Przemyslaw Prusinkiewicz



Demetri Terzopoulos