Parametric Curves & Surfaces

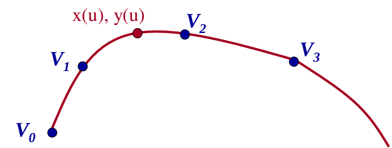
Thomas Funkhouser
Princeton University
COS 526, Fall 2006

Parametric curves

Parametric functions B(u) “blend” control points

$$x(u) = \sum_{i=0}^{n} B_i(u) * Vi_x$$

$$y(u) = \sum_{i=0}^{n} B_i(u) * Vi_y$$

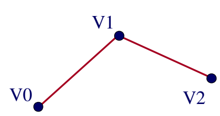


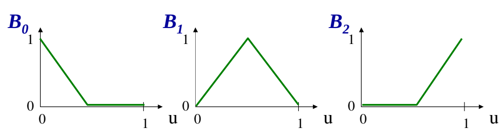
Parametric curves

What B(u) functions should we use?

$$x(u) = \sum_{i=0}^{n} B_i(u) * Vi_x$$

$$y(u) = \sum_{i=0}^{n} B_i(u) * Vi_y$$



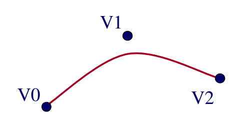


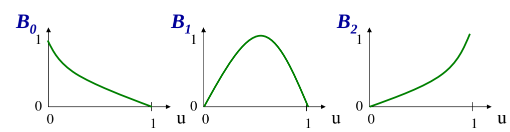
Parametric curves

What B(u) functions should we use?

$$x(u) = \sum_{i=0}^{n} B_i(u) * Vi_x$$

$$y(u) = \sum_{i=0}^{n} B_i(u) * Vi_y$$





Parametric Polynomial Curves

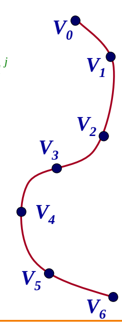
- Blending functions are polynomials:

$$x(u) = \sum_{i=0}^{n} B_i(u) * Vi_x$$

$$y(u) = \sum_{i=0}^{n} B_i(u) * Vi_y$$

$$B_i(u) = \sum_{j=0}^{m} a_j u^j$$

- Advantages of polynomials
  - Easy to compute
  - Infinitely continuous
  - Easy to derive curve properties

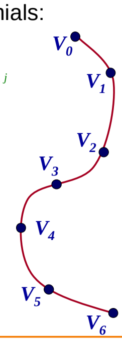


Parametric Polynomial Curves

- Blending functions are cubic polynomials:

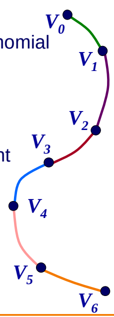
$$Q(u) = \sum_{i=0}^{n} B_i(u) * V_i \qquad B_i(u) = \sum_{j=0}^{3} a_j u^j$$

- Advantages of cubics
  - Efficient to compute
  - Enough control for most apps
  - Not too much wiggle room



Piecewise Parametric Cubic Curves

- Splines:
  - Split curve into segments
  - Each segment defined by low-order polynomial blending subset of control vertices
- Motivation:
  - Provides control & efficiency
  - Same blending function for every segment
  - Prove properties from blending functions
- Challenges
  - How guarantee continuity at joints?



Cubic Splines

- Types of splines
  - Natural Hermite – interpolating, C^2 continuous
  - B-Spline – approximating, C^2 continuous, local control
  - Catmull-Rom - interpolating, C^1 continuous, local control
  - Many others

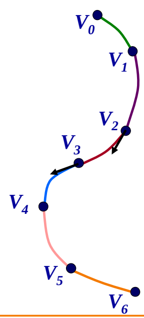
Each has different blending functions resulting in different properties

Natural Cubic Hermite Splines

- Definition:
  - Each segment defined by position and derivative at two adjacent control vertices
  - Blending functions are cubic polynomials

$$B(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$$

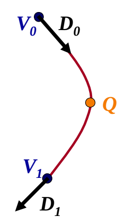
  - 4(n-1) degrees of freedom



Natural Cubic Hermite Splines

- Definition:
  - Each segment defined by position and derivative at two adjacent control vertices
  - Blending functions are cubic polynomials

$$B(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$$



$$Q(u) = B_0(u)*V_0 + B_1(u)*V_1 + B_2(u)*D_0 + B_3(u)*D_1$$

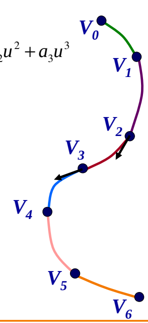
Natural Cubic Hermite Splines

- Definition:
  - 4(n-1) degrees of freedom
- Properties:

$$B(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$$

  - Interpolates control points
    - 2(n-2) constraints:

$$Q_i(0)=V_i \quad and \quad Q_i(1)=V_{i+1}$$



Natural Cubic Hermite Splines

- Definition:
  - 4(n-1) degrees of freedom
- Properties:

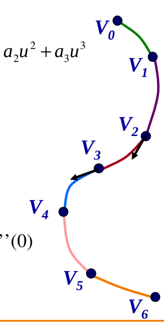
$$B(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$$

  - Interpolates control points
    - 2(n-2) constraints:

$$Q_i(0)=V_i \quad and \quad Q_i(1)=V_{i+1}$$

  - C^2 continuity
    - 2(n-1) constraints:

$$Q_i'(1)=Q_{i+1}'(0) \quad and \quad Q_i''(1)=Q_{i+1}''(0)$$



Natural Cubic Hermite Splines

- Definition:
  - 4(n-1) degrees of freedom

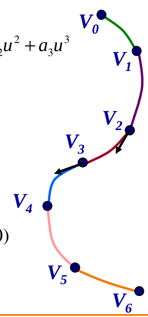
$$B(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$$

- Properties:
  - Interpolates control points
    - 2(n-2) constraints:

      $Q_i(0)=V_i$ *and* $Q_i(1)=V_{i+1}$
  - C^2 continuity
    - 2(n-1) constraints:

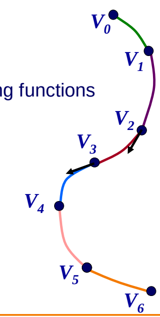
      $Q_i'(1)=Q_{i+1}'(0)$ *and* $Q_i''(1)=Q_{i+1}''(0)$

Solve system of equation for coefficients of blending functions



Natural Cubic Hermite Splines

- Problems:
  - No local control
    - Whole curve adjusts to any movement of control vertex
  - Every segment has different blending functions
    - Hard to prove properties



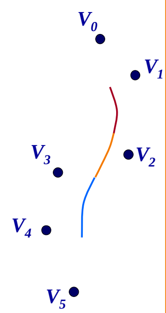
Cubic Splines

- Types of splines
  - Natural Hermite – interpolating, C^2 continuous
  - Ø B-Spline – approximating, C^2 continuous, local control
  - Catmull-Rom - interpolating, C^1 continuous, local control
  - Many others

Each has different blending functions resulting in different properties

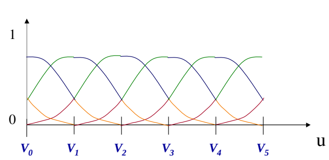
Uniform Cubic B-Splines

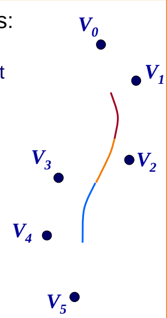
- Properties:
  - Local control
  - C^2 continuity
  - Approximating



B-Spline Blending Functions

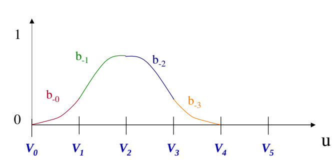
- Properties imply blending functions:
  - Cubic polynomials
  - Four control vertices affect each point
  - C^2 continuity

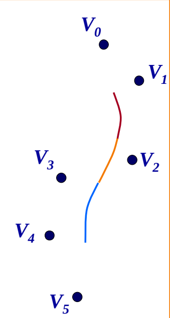




B-Spline Blending Functions

- How derive blending functions?
  - Cubic polynomials
  - Local control
  - C^2 continuity

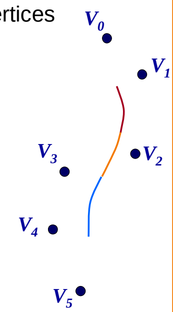




B-Spline Blending Functions

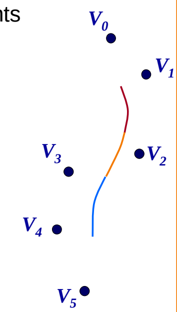
- Four cubic polynomials for four vertices
  - 16 variables (degrees of freedom)
  - Variables are a_i, b_i, c_i, d_i for four blending functions

$$
\begin{aligned}
b_{-0}(u) &= a_0u^3 + b_0u^2 + c_0u^1 + d_0 \\
b_{-1}(u) &= a_1u^3 + b_1u^2 + c_1u^1 + d_1 \\
b_{-2}(u) &= a_2u^3 + b_2u^2 + c_2u^1 + d_2 \\
b_{-3}(u) &= a_3u^3 + b_3u^2 + c_3u^1 + d_3
\end{aligned}
$$



B-Spline Blending Functions

- C2 continuity implies 15 constraints
  - Position of two curves same
  - Derivative of two curves same
  - Second derivatives same



B-Spline Blending Functions

Fifteen continuity constraints:

$$
\begin{array}{lll}
0 = b_{-0}(0) & 0 = b_{-0}'(0) & 0 = b_{-0}''(0) \\
b_{-0}(1) = b_{-1}(0) & b_{-0}'(1) = b_{-1}'(0) & b_{-0}''(1) = b_{-1}''(0) \\
b_{-1}(1) = b_{-2}(0) & b_{-1}'(1) = b_{-2}'(0) & b_{-1}''(1) = b_{-2}''(0) \\
b_{-2}(1) = b_{-3}(0) & b_{-2}'(1) = b_{-3}'(0) & b_{-2}''(1) = b_{-3}''(0) \\
b_{-3}(1) = 0 & b_{-3}'(1) = 0 & b_{-3}''(1) = 0
\end{array}
$$

One more convenient constraint:

$$b_{-0}(0) + b_{-1}(0) + b_{-2}(0) + b_{-3}(0) = 1$$

B-Spline Blending Functions

- Solving the system of equations yields:

$$
\begin{aligned}
b_{-3}(u) &= -\tfrac{1}{6}u^3 + \tfrac{1}{2}u^2 - \tfrac{1}{2}u + \tfrac{1}{6} \\
b_{-2}(u) &= \tfrac{1}{2}u^3 - u^2 + \tfrac{2}{3} \\
b_{-1}(u) &= -\tfrac{1}{2}u^3 + \tfrac{1}{2}u^2 + \tfrac{1}{2}u + \tfrac{1}{6} \\
b_{-0}(u) &= \tfrac{1}{6}u^3
\end{aligned}
$$

B-Spline Blending Functions

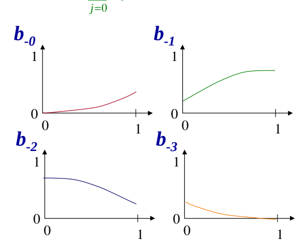
- In matrix form:

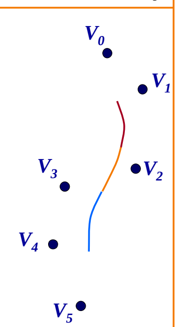
$$
Q(u) = \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \frac{1}{6} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix} \begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{pmatrix}
$$

B-Spline Blending Functionss

In plot form:

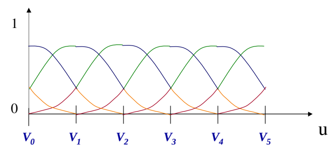
$$B_i(u) = \sum_{j=0}^{m} a_j u^j$$

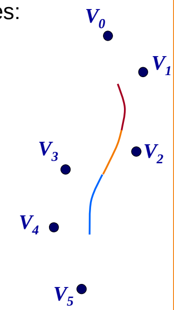




B-Spline Blending Functions

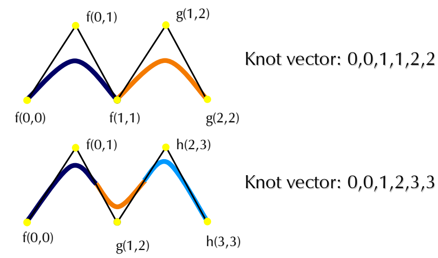
- Blending functions imply properties:
  - Local control
  - Approximating
  - C^2 continuity
  - Convex hull





Non-Uniform B-Splines

- Knot vector describes parameter values at every control vertex (provides further control)

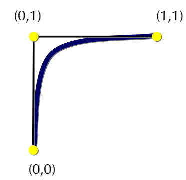


Rational B-Splines

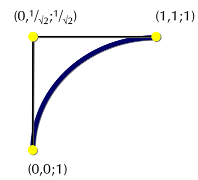
- Can't represent certain shapes (e.g. circles) with piecewise polynomials
- Wider class of functions: rational functions
  - Ratio of polynomials
  - Can represent any quadric (e.g. circles) exactly
- Mathematical trick: homogeneous coordinates
  - Ratio of 2 polynomials in 3d equivalent to single polynomial in 4d

Rational Splines

- Example: creating a circlular arc with 3 control points



Polynomial spline:
parabolic arc



Rational spline:
circular arc

NURBS

- Non-uniform (vary time interval per segment)
- Rational
- B-Splines
- Can model a wide class of curves and surfaces
- Same convenient properties of B-Splines
- Widely used in CAD systems (and OpenGL)

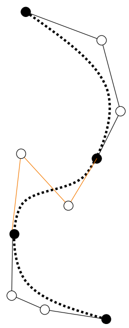
Cubic Splines

- Types of splines
  - Natural Hermite – interpolating, C^2 continuous
  - B-Spline – approximating, C^2 continuous, local control
  - Ø Catmull-Rom - interpolating, C^1 continuous, local control
  - Many others

Each has different blending functions resulting in different properties

Catmull-Rom splines

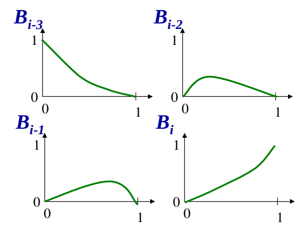
- Properties
  - Interpolate control points
  - Have C^1 continuity
- Derivation
  - Interpolate every third control point
  - Build cubic Bézier between each joint with C^1 continuity at joint

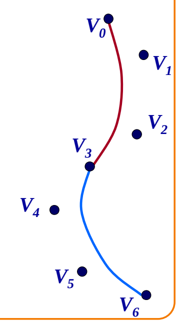


Cubic Bezier curves

Blending functions:

$$B_i(u) = \sum_{j=0}^{m} a_j u^j$$





Basic properties of Bézier curves

- Endpoint interpolation:

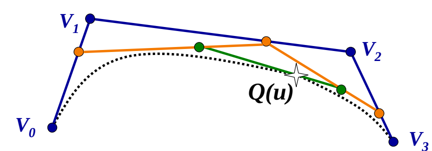
$$Q(0) = V_0$$
$$Q(1) = V_n$$

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

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

Bézier curves

- Curve $Q(u)$ can also be defined by nested interpolation (de Casteljau Algorithm):

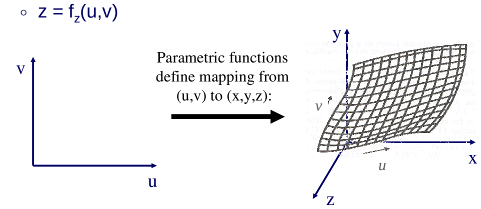


V_i's are control points
$\{V_0, V_1, \ldots, V_n\}$ is control polygon

Parameteric Surfaces

Parametric Surfaces

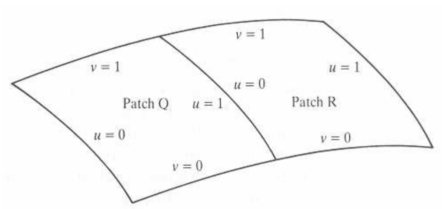
- Boundary defined by parametric functions:
  - $x = f_x(u,v)$
  - $y = f_y(u,v)$
  - $z = f_z(u,v)$



FvDFH Figure 11.42

Piecewise Parametric Surfaces

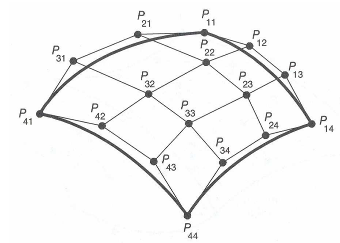
- Surface is partitioned into parametric patches:



Watt Figure 6.25

Parametric Patches

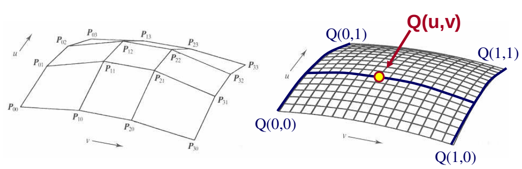
- Each patch is defined by blending control points



FvDFH Figure 11.44

Parametric Patches

- Point Q(u,v) on the patch is the tensor product of parametric curves defined by the control points



Watt Figure 6.21

Parametric Bicubic Patches

Point Q(u,v) on any patch is defined by combining control points with polynomial blending functions:

$$Q(u,v) = \mathbf{UM}\begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} \\ P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} \end{bmatrix}\mathbf{M}^{\mathbf{T}}\mathbf{V}^{\mathbf{T}}$$

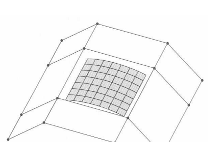
$$\mathbf{U} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \qquad \mathbf{V} = \begin{bmatrix} v^3 & v^2 & v & 1 \end{bmatrix}$$

Where M is a matrix describing the blending functions for a parametric cubic curve (e.g., Bezier, B-spline, etc.)

B-Spline Patches

$$Q(u,v) = \mathbf{UM}_{\text{B-Spline}}\begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} \\ P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} \end{bmatrix}\mathbf{M}^{\mathbf{T}}_{\text{B-Spline}}\mathbf{V}$$

$$\mathbf{M}_{\text{B-Spline}} = \begin{bmatrix} -1/6 & 1/2 & -1/2 & 1/6 \\ 1/2 & -1 & 1/2 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 1/6 & 2/3 & 1/6 & 0 \end{bmatrix}$$

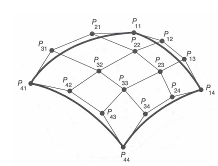


Watt Figure 6.28

Bezier Patches

$$Q(u,v) = \mathbf{UM}_{\text{Bezier}}\begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} \\ P_{4,1} & P_{4,2} & P_{4,3} & P_{4,4} \end{bmatrix}\mathbf{M}^{\mathbf{T}}_{\text{Bezier}}\mathbf{V}$$

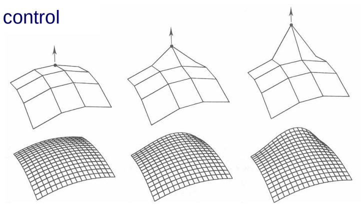
$$\mathbf{M}_{\text{Bezier}} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$



FvDFH Figure 11.42

Bezier Patches

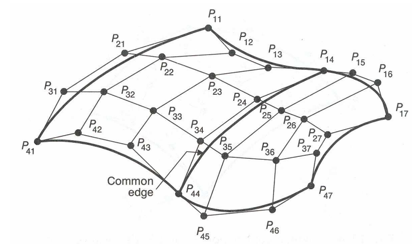
- Properties:
  - Interpolates four corner points
  - Convex hull
  - Local control



Watt Figure 6.22

Bezier Surfaces

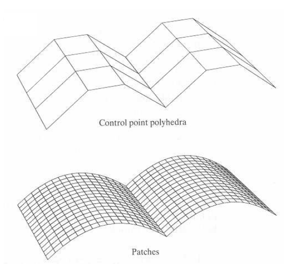
- Continuity constraints are similar to the ones for Bezier splines



FvDFH Figure 11.43

Bezier Surfaces

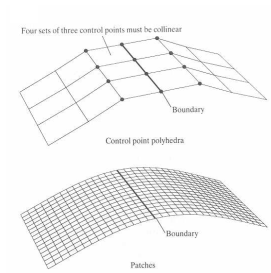
- C^0 continuity requires aligning boundary curves



Watt Figure 6.26a

Bezier Surfaces

- C^1 continuity requires aligning boundary curves and derivatives



Watt Figure 6.26b

Summary of Parametric Surfaces

- Advantages:
  - Easy to enumerate points on surface
- Disadvantages:
  - Control mesh must have specific topology (e.g., quads)