6.837 Introduction to Computer Graphics

Splines

Administrivia

1) C++ Tutorial at 7:30 in 32-D507

2) Sign up for both the students and discuss lists

3) Assignment 0 due Wednesday 8pm
  - You **must** have a README file with the proper stuff.
  - Your assignment **must** be in the correct turnin directory (called "zero")
  - Your assignment **must** compile and run (without any issues) on Athena Linux.

Any violation of these rules will result in massive points off.

Big message of the day

- Linearity makes life easy
- If something's linear, matrices are useful

Vectors spaces, matrices

- Basis, cartesian coordinates
  - Basis: set of vector that span the space
  - That is, every vector is a linear combination of the basis vector
- Special basis: orthonormal
- Dot product
  - Geometric: length of projection onto other vector
  - In orthonormal basis: sum of product of coordinates
  - In orthonormal basis: dot product with basis vectors provides coordinates
- Cross product
  - Provides orthogonal vector, length = product times sine, right-hand rule

---

Linearity

- What characterizes a linear function/operator?
  - Function over a vector space
  - For two vectors V & W: f(V+W)=f(V)+f(W)
  - For a scalar a f(aV)=af(V)
    - Consequence f(0)=0
- This is a fundamental property, makes life easy
- It enables us to focus on a basis
  - f(xI+yJ)=x f(I) + y f(J)

Matrices

- Convenient notation for linear transforms

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

- Means

x'=ax+by
y'=cx+dy

- Matrices can be multiplied, transposed, sometimes inverted

Example 2x2 matrices

- Scale

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

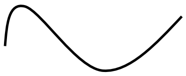
- Rotation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Affine operators: add translation

- Consider the transformation of R^2
  - $F(x,y) = (x+t_x, y+t_y)$
- It is not linear
  - E.g. f(0, 0) is not 0
  - $F(x+x', y+y')=F(x, y)+f(x', y') + (t_x, t_y)$
- Cannot (directly) be represented by a matrix
- It's kind of confusing, because from a polynomial perspective, we would say that F is a linear function.

Splines



- Smooth curves in the plane or in 3D
- Many usages
  - 2D illustration (e.g. Adobe Illustrators)
  - Fonts
  - 3D modeling
  - Color ramps
  - Animation: trajectories
  - In general, interpolate keyframes

ACM © 1987 "Principles of traditional animation applied to 3D computer animation"

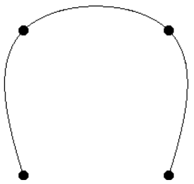
Two definitions of curves

- A continuous set of points on the plane/in space
- A mapping from an interval of R onto the plane
  - That is, P(t) is the point of the curve at parameter t

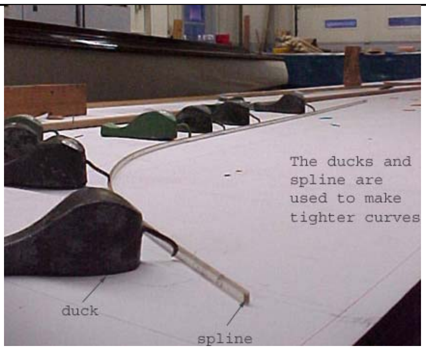
- Big difference: the second definition can describe trajectories, the speed at which we move on the curve

General principle

- User specifies control points
- Defines a smooth curve



Physical splines



www.abm.org

Two ways to look at the problem

- Approximation/interpolation
  - We have data points, how can we interpolate?
- User interface/modeling
  - What is an easy way to specify a smooth curve
  - The main perspective today

UI/modeling

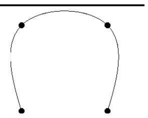
- How can we specify a curve?
  - Provide parametric function (x,y) = f(t)
    - Like a trajectory overt time
  - Analyical implicit description f(x,y)=0
  - Draw it with mouse!
  - Provide a few points
    - That’s what we will do today

Splines

- Specified by a few control points
  - Good for UI
  - Good for storage
- Results in a smooth parametric curve P(t)
  - Defined in Cartesian coordinates by x(t) and y(t)
  - Polynomial in practice
  - Convenient for animation where t is time
  - Convenient for *tesselation* because we can discretize t and approximate the curve with small linear segments

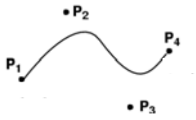
Interpolation vs. approximation

- Interpolation
  - Goes through all specified points
  - Sounds more logical



Interpolation

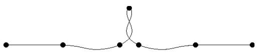
- Approximation
  - Does not go through all points



Approximation

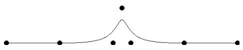
Interpolation vs. approximation

- Interpolation
  - Goes through all specified points
  - Sounds more logical
  - But can be more unstable, ringing



Interpolation

- Approximation
  - Does not go through all points
  - Turns out to be convenient
  - This is what we’ll focus on

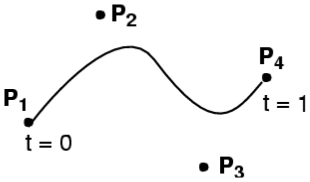


Approximation

Questions?

Cubic Bezier splines

- User specifies 4 control points *Pi*
- Curve goes through the two extremities
- Approximates the two other ones
- Cubic polynomial

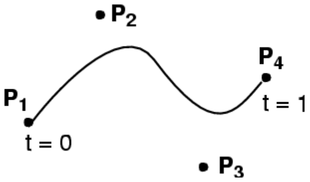


Cubic Bezier splines

- $$\begin{aligned} P(t) = \; & (1-t)^3 \; P_1 \\ + \; & 3t(1-t)^2 \; P_2 \\ + \; & 3t^2(1-t) \; P_3 \\ + \; & t^3 \; P_4 \end{aligned}$$

That is:

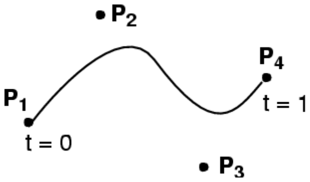
$x(t) = (1-t)^3 \, x_1$
$+3t\,(1-t)^2 x_2$
$+3t^2(1-t)x_3$
$+t^3 x_4$
$x(t) = (1-t)^3 \, y_1$
$+3t^2(1-t)y_2$
$+3t^2(1-t)y_3$
$+t^3 y_4$



Cubic Bezier splines

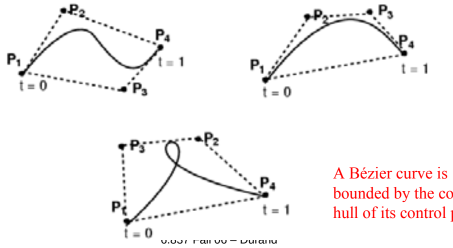
- $$\begin{aligned} P(t) = \; & (1-t)^3 \; P_1 \\ + \; & 3t(1-t)^2 \; P_2 \\ + \; & 3t^2(1-t) \; P_3 \\ + \; & t^3 \; P_4 \end{aligned}$$

Verify what happens for t=0 and t=1



Cubic Bézier Curve

- 4 control points
- Curve passes through first & last control point
- Curve is tangent at $\mathbf{P_0}$ to ($\mathbf{P_1}$-$\mathbf{P_2}$) and at $\mathbf{P_4}$ to ($\mathbf{P_4}$-$\mathbf{P_3}$)



A Bézier curve is bounded by the convex hull of its control points.

Questions?

Where is the formula coming from?

- Explanation 1:
  - It's magical, I pulled it off my hat, it happens to approximate the points
- Explanation 2:
  - It's a ***linear combination of basis polynomials***
  - Let's study this with a simpler case, 1D curves y=f(t)

Polynomials as a vector space

- Polynomials $y=a_0+a_1t+a_2t^2+\ldots+a_nt^n$
- Can be added: just add the coefficients
- Can be multiplied by a scalar: multiply the coefficients

- It's a vector space!

Subset of polynomials: cubic

$y=a_0+a_1t+a_2t^2+a_3t^3$

- Closed under addition & multiplication by scalar
  - i.e. the result is still a cubic polynomial
- It is also a vector space, subspace of the full polynomial space
- The x and y coordinates of cubic Bezier curves belong to this space
- What is a smart basis?

Basis for cubic polynomials

What's a basis?

- Set of "vectors" in the space
  - Here, vector = cubic polynomial
- Linear combination of vectors spans the space
  - i.e. any cubic polynomial is a sum of those basis cubics
- Independent
  - No vector is combination of other vectors in basis

Canonical basis for cubics

1, t, t^2, t^3

- Any polynomial is a linear combination of these
  - $a_0+a_1t+a_2t^2+a_3t^3=a_0*1+a_1*t+a_2*t^2+a_3*t^3$
  - Duh!
- Are independent

- We often call them basis functions

Different basis

- For example:
  - 1, $1+t$, $1+t+t^2$, $1+t-t^2+t^3$
  - t^3, t^3+t^2, t^3+t, t^3+1

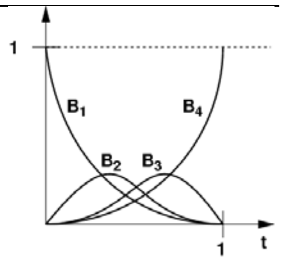
- Infinite number of possibilities, just like you have an infinite number of basis to span R^2
- For Bezier curve, there is a convenient basis: Bernstein polynomials

Questions?

Bernstein polynomials

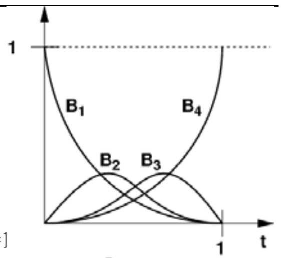
For cubic:

- $B_1(t)=(1-t)^3$
- $B_2(t)=3t(1-t)^2$
- $B_3(t)=3t^2(1-t)$
- $B_4(t)=t^3$
  - (careful with indices, many authors start at 0)
- But defined for any degree



Properties of Bernstein polynomials

- Sum to one for every t
  - Partition of unity
  - This is why Bezier curve is inside convex hull
- Only B_1 is non null at 0
  - Bezier interpolates P_1
  - Same for B_4 and P_4 for t=1



Bezier in Bernstein basis

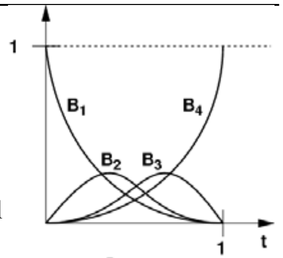
- $P(t) = P_1B_1(t)+P_2B_2(t)+P_3B_3(t)+P_4B_4(t)$
  - Where P_i are generalized coefficients (x, y)
- The control points (P_1, P_2, P_3, P_4) are the coordinates of the curve in the Bernstein basis of the abstract cubic space
  - Specifying a Bezier curve with control points is exactly like specifying a 2D points with its x and y coordinates

Bezier in Bernstein basis

- We're dealing here with two vector spaces:
  - The plane where the curve lies, a 2D vector space
  - The space of cubic polynomials, a 4D space
  - Don't be confused!
- The 2D data points can be trivially replaced by 3D points

Bernstein as influence function

- Each B_i specifies the influence of P_i
- First, P_1 is the most influential point then P_2, P_3, and P_4
- P_2 and P_3 never have full influence
  - Not interpolated



Questions?

Change of basis

- How do we go from Bernstein basis to the canonical monomial basis 1, t, t^2, t^3 ?
  - E.g. $f(t)=b_1B_1(t)+b_2B_2(t)+b_3B_3(t)+b_4B_4(t)$
  - That is, (b1, b2, b3, b4) in Bernstein basis
- With a matrix!
  - Slightly dirty, I'll left multiply

$$f(t)= (b_1, b_2, b_3, b_4) \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} t^3 \\ t^2 \\ t \\ 1 \end{pmatrix}$$

New basis vectors

Recap

- Cubic polynomials form a vector space
- Bernstein basis is canonical for Bezier
  - Can be seen as influence function of data points
  - Or data points are coordinates of the curve in the Bernstein basis
- We can change basis with matrices

Questions

General spline formulation

$$Q(t) = \mathbf{GBT(t)} = \text{Geometry } \mathbf{G} \cdot \text{Spline Basis } \mathbf{B} \cdot \text{Power Basis } \mathbf{T(t)}$$

- Geometry: control points coordinates assembled into a matrix (P_1, P_2, …, P_{n+1})
- Spline basis: defines the type of spline
  - Bernstein for Bezier
- Power basis: the monomials $^T(t^n, t^{n-1}, \ldots t^{2,} t, 1)$
- Advantage of general formulation
  - Compact expression
  - Easy to convert between types of splines

Cubic Bezier in general formulation

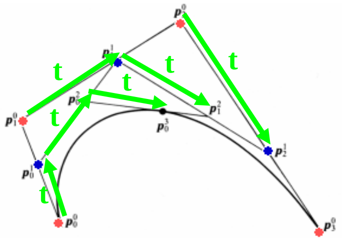
$$Q(t) = \mathbf{GBT(t)} = \text{Geometry } \mathbf{G} \cdot \text{Spline Basis } \mathbf{B} \cdot \text{Power Basis } \mathbf{T(t)}$$

$$P(t)= \begin{pmatrix} P_{1,x} & P_{2,x} & P_{3,x} & P_{4,x} \\ P_{1,y} & P_{2,y} & P_{3,y} & P_{4,y} \end{pmatrix} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} t^3 \\ t^2 \\ t \\ 1 \end{pmatrix}$$

Question?

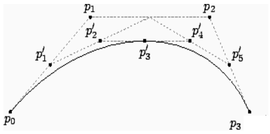
Cubic Bézier Curve

- de Casteljau's algorithm for constructing Bézier curves

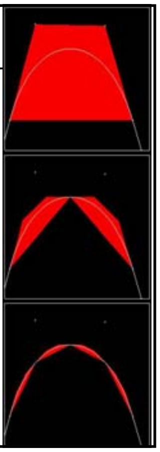


Neat Bezier Spline Trick

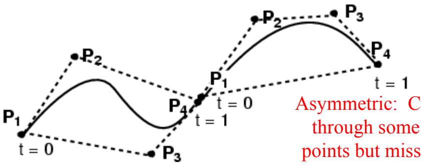
- A Bezier curve with 4 control points:
  - $P_0 \quad P_1 \quad P_2 \quad P_3$
- Can be split into 2 new Bezier curves:
  - $P_0 \quad P'_1 \quad P'_2 \quad P'_3$
  - $P'_3 \quad P'_4 \quad P'_5 \quad P_3$



A Bézier curve is bounded by the convex hull of its control points.



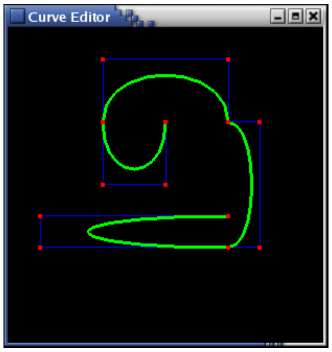
Connecting Cubic Bézier Curves



Asymmetric: Curve goes through some control points but misses others

- How can we guarantee C^0 continuity?
- How can we guarantee G^1 continuity?
- How can we guarantee C^1 continuity?
- Can't guarantee higher C^2 or higher continuity

Connecting Cubic Bézier Curves



- Where is this curve
  - C^0 continuous?
  - G^1 continuous?
  - C^1 continuous?
- What's the relationship between:
  - the # of control points, and
  - the # of cubic Bézier subcurves?

Higher-Order Bézier Curves

- > 4 control points
- Bernstein Polynomials as the basis functions
  - For polynomial of order n, the ith basis function is

$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$$

- Every control point affects the entire curve
  - Not simply a local effect
  - More difficult to control for modeling

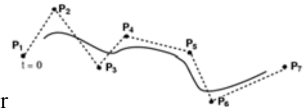
Recap

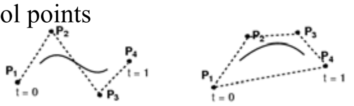
- Bezier curves: piecewise polynomials
- Linear combination of basis functions
  - Coefficient = data point
- Bernstein basis
- All linear, matrix algebra
- Subdivision de Casteljau algorithm

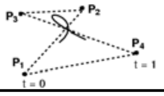
Questions?

Cubic BSplines

- ≥ 4 control points
- Locally cubic
  - Cubic chained together
- Curve is not constrained to pass through any control points

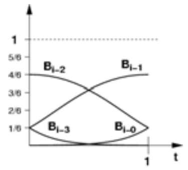






A BSpline curve is also bounded by the convex hull of its control points.

Cubic BSplines: basis

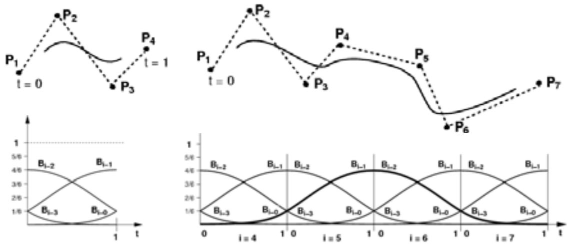


$$Q(t) = \frac{(1-t)^3}{6}P_{i-3} + \frac{3t^3 - 6t^2 + 4}{6}P_{i-2} + \frac{-3t^3 + 3t^2 + 3t + 1}{6}P_{i-1} + \frac{t^3}{6}P_i$$

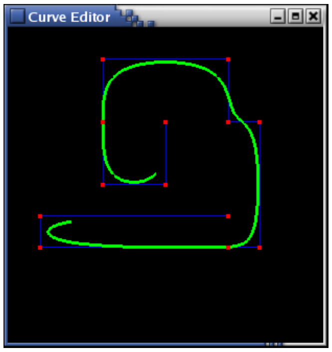
$$Q(t) = \mathbf{GBT(t)} \qquad B_{B-Spline} = \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 0 & 4 \\ -3 & 3 & 3 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Cubic BSplines

- Can be chained together
- Better control locally (windowing)

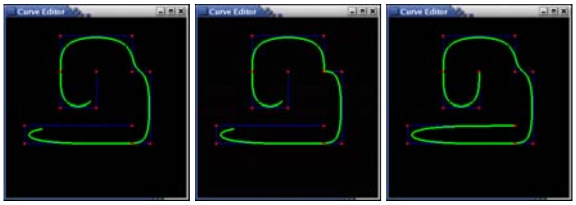


Connecting Cubic BSpline Curves



- What's the relationship between
  - the # of control points, and
  - the # of cubic BSpline subcurves?

BSpline Curve Control Points



Default BSpline

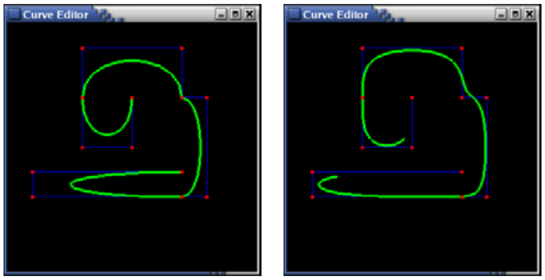
BSpline with Discontinuity

Repeat interior control point

BSpline which passes through end points

Repeat end points

Bézier is not the same as BSpline

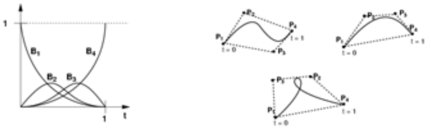


Bézier BSpline

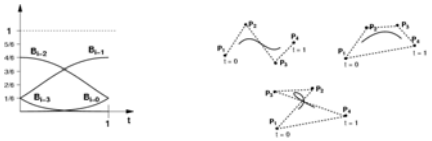
Bézier is not the same as BSpline

- Relationship to the control points is different

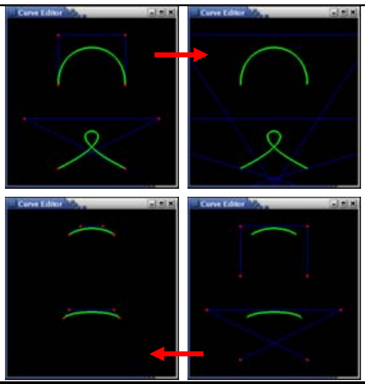
Bézier



BSpline



Converting between Bézier & BSpline



original control points as Bézier

new BSpline control points to match Bézier

new Bézier control points to match BSpline

original control points as BSpline

Converting between Bézier & BSpline

- Using the basis functions:

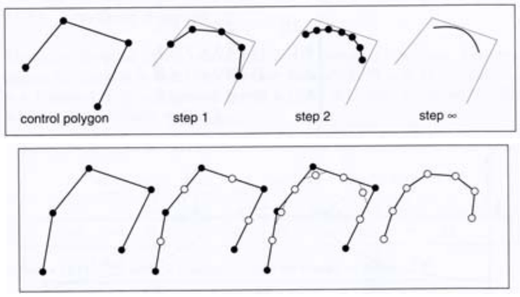
$$B_{Bezier} = \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$B_{B-Spline} = \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 0 & 4 \\ -3 & 3 & 3 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$Q(t) = \mathbf{GBT(t)} = \text{Geometry } \mathbf{G} \cdot \text{Spline Basis } \mathbf{B} \cdot \text{Power Basis } \mathbf{T(t)}$$

Cubic BSplines

- Iterative method for constructing BSplines



Shirley, Fundamentals of Computer Graphics

NURBS (generalized BSplines)

- BSpline: uniform cubic BSpline
- Rational Bezier/cubic
  - Use homogeneous coordinates (see later)
- NURBS: Non-Uniform Rational BSpline
  - non-uniform = different spacing between the blending functions, a.k.a. knots
  - rational = ratio of polynomials (instead of cubic)

Questions?

Linear Transformations & splines

- What happens when we transform the control points with a linear transform?
  - – Is it the same as transforming each point on the curve?
  - – Yes! Because everything is linear

$$P'(t) = M \left[\begin{pmatrix} P_{1,x} & P_{2,x} & P_{3,x} & P_{4,x} \\ P_{1,y} & P_{2,y} & P_{3,y} & P_{4,y} \end{pmatrix} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix} \right]$$

$$= \left[M \begin{pmatrix} P_{1,x} & P_{2,x} & P_{3,x} & P_{4,x} \\ P_{1,y} & P_{2,y} & P_{3,y} & P_{4,y} \end{pmatrix} \right] \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Affine Transformations & splines

- E.g. translations? F(P) =P+V
  - – Remember, affine transforms are not linear!
- We're lucky, it still works.
  - – This is because the sum of the basis/influence is always one

---

Affine Transformations & splines

- e.g. for Bezier where
  $P(t) = P_1B_1(t)+P_2B_2(t)+P_3B_3(t)+P_4B_4(t)$
- Let's transform the control points:

$F(P_1)B_1(t)+F(P_2)B_2(t)+F(P_3)B_3(t)+F(P_4)B_4(t)$

$= (P_1+V)B_1(t)+(P_2+V)B_2(t)+(P_3+V)B_3(t)+(P_4+V)B_4(t)$

$$=P_1B_1(t)+P_2B_2(t)+P_3B_3(t)+P_4B_4(t) + V \underbrace{(B_1(t)+B_2(t)+B_3(t)+B_4(t))}_{=1}$$

$=P_1B_1(t)+P_2B_2(t)+P_3B_3(t)+P_4B_4(t) + V$