Parametric Curves

Jason Lawrence

CS4810: Introduction to Graphics

Acknowledgment: slides by Misha Kazhdan, Allison Klein, Tom Funkhouser, Adam Finkelstein and David Dobkin
Parametric Curves and Surfaces

Part 1: Curves
Part 2: Surfaces

Courtesy of C.K. Shene
Curves

• Splines: mathematical way to express curves

• Motivated by “loftsman’s spline”
  o Long, narrow strip of wood/plastic
  o Used to fit curves through specified data points
  o Shaped by lead weights called “ducks”
  o Gives curves that are “smooth” or “fair”

• Have been used to design:
  o Automobiles
  o Ship hulls
  o Aircraft fuselage/wing
Goals

• Some attributes we might like to have:
  o Predictable/local control
  o Simple
  o Continuous

• We’ll satisfy these goals using:
  o Piecewise
  o Polynomials
Many applications in graphics

- Animation paths
- Shape modeling
- etc…

Animation
*(Angel, Plate 1)*

Shell
*(Douglas Turnbull)*
What is a Spline in CG?

A spline is a *piecewise polynomial function* whose derivatives satisfy some *continuity constraints* across curve boundaries.

So let’s look at what this means…
What is a Spline in CG?

**Piecewise**: the spline is actually a collection of individual segments joined together.

**Polynomial functions**: each of these segments is expressed by a polynomial function.
A parametric curve in $d$-dimensions is defined by a collection of 1D functions of one variable that give the coordinates of points on the curve at each value of $u$:

$$\Phi(u) = (x_1(u), \ldots, x_d(u))$$

Note:
A parametric curve is not the graph of a function, it is the path traced out as the value of $t$ is allowed to change.
Derivatives

If \( \Phi(u) = (x(u), y(u)) \) is the parametric equation of a curve, the parametric derivative of the curve at a point \( u_0 \) is the vector:

\[
\Phi'(u_0) = \left( x'(u_0), y'(u_0) \right)
\]

which points in a direction tangent to the curve.

Note:
The direction of the derivative is determined by the path that the curve traces out.
The magnitude of the parametric derivative is determined by the tracing speed.
Polynomials

A polynomial in the variable $u$ is:

- “An algebraic expression written as a sum of constants multiplied by different powers of a variable.”

$$P(u) = a_0 + a_1 u + a_2 u^2 + \ldots + a_n u^n = \sum_{k=0}^{n} a_k u^k$$

The constant $a_k$ is referred to as the $k$-th coefficient of the polynomial $P$. 
Polynomials (Degree)

\[ P(u) = a_0 + a_1 u + a_2 u^2 + ... + a_n u^n = \sum_{k=0}^{n} a_k u^k \]

A polynomial \( P \) has degree \( n \) if for all \( k > n \), the coefficients of the polynomial satisfy \( a_k = 0 \).
Polynomials (Degree)

A polynomial $P$ has degree $n$ if for all $k > n$, the coefficients of the polynomial satisfy $a_k = 0$.

A polynomial of degree $n$ has $n + 1$ degrees of freedom.

Knowing $n + 1$ pieces of information about a polynomial of degree $n$ gives enough information to reconstruct the coefficients.
Polynomials (Matrices)

\[ P(u) = a_0 + a_1 u + a_2 u^2 + \cdots + a_n u^n = \sum_{k=0}^{n} a_k u^k \]

The polynomial \( P \) can be expressed as the matrix multiplication of a column vector and a row vector:

\[
P(u) = \begin{pmatrix} u^n & \cdots & u^0 \end{pmatrix} \begin{pmatrix} a_n \\ \vdots \\ a_0 \end{pmatrix}
\]
Polynomials (Matrices)

Example:

If we know the values of the polynomial $P$ at $n+1$ different values:

$$P(u_0) = p_0, \ldots, P(u_n) = p_n$$

We can compute the coefficients of $P$ by inverting the appropriate matrix:

$$
\begin{pmatrix}
  p_0 \\
  \vdots \\
  p_n \\
\end{pmatrix}
= 
\begin{pmatrix}
  (u_0)^n & \cdots & (u_0)^0 \\
  \vdots & \ddots & \vdots \\
  (u_n)^n & \cdots & (u_n)^0 \\
\end{pmatrix}
\begin{pmatrix}
  a_n \\
  \vdots \\
  a_0 \\
\end{pmatrix}
= 
\begin{pmatrix}
  a_n \\
  \vdots \\
  a_0 \\
\end{pmatrix}
= 
\begin{pmatrix}
  (u_0)^n & \cdots & (u_0)^0 \\
  \vdots & \ddots & \vdots \\
  (u_n)^n & \cdots & (u_n)^0 \\
\end{pmatrix}^{-1}
\begin{pmatrix}
  p_0 \\
  \vdots \\
  p_n \\
\end{pmatrix}
$$
Polynomials (Matrices)

\[ P(u) = \sum_{k=0}^{n} a_k u^k \]

Example:

So, if we are given the values of the polynomial \( P \) at the \( n+1 \) positions \( u_0, \ldots, u_n \), we can compute the value of \( P \) at any position \( u \) by solving:

\[
P(u) = \begin{pmatrix} u^n & \cdots & u^0 \end{pmatrix} \begin{pmatrix} (u_0)^n & \cdots & (u_0)^0 \\ \vdots & \ddots & \vdots \\ (u_n)^n & \cdots & (u_n)^0 \end{pmatrix}^{-1} \begin{pmatrix} p_0 \\ \vdots \\ p_n \end{pmatrix}
\]
Parametric Polynomial Curves

• A parametric polynomial curve of degree $n$ in $d$ dimensions is a collection of $d$ polynomials, each of which is of degree no larger than $n$:

\[
\Phi(\mathbf{u}) = \left( x_1(\mathbf{u}) = \sum_{k=0}^{n} a_{1,k} u^k, ..., x_d(\mathbf{u}) = \sum_{k=0}^{n} a_{d,k} u^k \right)
\]
Parametric Polynomial Curves

Examples:

• When $x(u) = u$, the curve is just the graph of $y(u)$.

• Different parametric equations can trace out the same curve.

• As the degree gets larger, the complexity of the curve increases.
Parametric Curves

Goal:

Given a collection of $m$ points in $d$ dimensions:

$$\{ p_1 = (x_{1,1}, \ldots, x_{1,d}) \ldots, p_m = (x_{m,1}, \ldots, x_{m,d}) \}$$

define a parametric curve that passes through (or near) the points
Parametric Curves

Direct Approach:

Solve for the $m$ coefficients of a parametric polynomial curve of degree $m-1$, passing through the points.
Parametric Curves

Direct Approach:

Solve for the $m$ coefficients of a parametric polynomial curve of degree $m-1$, passing through the points.

Limitations:

• No local control

• As the number of points increases, the dimension gets larger, and the curve oscillates more.
Splines

Approach:
Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:
- Local Control
- Simplicity
- Continuity/Smoothness
Splines

Approach:

Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:

- Local Control
- Simplicity
- Continuity/Smoothness
Splines

Approach:

Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:

- Local Control
- Simplicity
- Continuity/Smoothness
Splines

Approach:

Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:

- Local Control
- Simplicity
- Continuity/Smoothness
Splines

Approach:

Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:

- Local Control
- Simplicity
- Continuity/Smoothness
Piecewise parametric polynomials

Approach:

Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:

- Local Control:
  - Individual curve segments are defined using only local information

- Simplicity
  - Curve segments are low-order polynomials
Piecewise parametric polynomials

Approach:

Fit low-order polynomials to groups of points so that the combined curve passes through (or near) the points while providing:

- Local Control:
  - Individual curve segments are defined using only local information
- Simplicity
  - Curve segments are low-order polynomials
- Continuity/Smoothness
  - How do we guarantee smoothness at the joints?
Continuity/Smoothness

**Continuity:**

Within the parameterized domain, the polynomial functions are continuous and smooth.

The derivatives of our polynomial functions must satisfy continuity constraints across the curve boundaries.
Continuity/Smoothness

Parametric continuity: derivatives of the two curves are \textit{equal} where they meet.

- $C^0$ means two curves just meet
- $C^1$ means 1\textsuperscript{st} derivatives equal
- $C^2$ means both 1\textsuperscript{st} and 2\textsuperscript{nd} derivatives equal
Continuity/Smoothness

Geometric continuity: derivatives of the two curves are proportional (i.e. point in the same direction) where they meet.

- $G^0$ means two curves just meet
- $G^1$ means $G^0$ and 1st derivatives proportional
- $G^2$ means $G^1$ and 2nd derivatives proportional
- Parametric continuity used more frequently than geometric.
What is a Spline in CG?

A spline is a *piecewise polynomial function* whose derivatives satisfy some *continuity constraints* across curve boundaries.

\[ P_1(x) \quad x \in [0,1), \]
\[ P_2(x) \quad x \in [0,1), \]
\[ P_3(x) \quad x \in [0,1). \]

\[ P_i(x) = \sum_{j=0}^{n} a_{ij} x^j \]
What is a Spline in CG?

A spline is a *piecewise polynomial function* whose derivatives satisfy some *continuity constraints* across curve boundaries.

\[ P_i(x) = \sum_{j=0}^{n} a_{ij} x^j \]

\[ P_1(1) = P_2(0) \]
\[ P_1'(1) = P_2'(0) \]
\[ \ldots \]

\[ P_2(1) = P_3(0) \]
\[ P_2'(1) = P_3'(0) \]
\[ \ldots \]
Overview

• What is a Spline?

• Specific Examples:
  o Hermite Splines
  o Cardinal Splines
  o Uniform Cubic B-Splines

• Comparing Cardinal Splines to Uniform Cubic B-Splines
Specific Example: Hermite Splines

- Interpolating piecewise *cubic* polynomial

- Specified with:
  - A pair of control points
  - Tangent at each control point

- Iteratively construct the curve between adjacent end points
Specific Example: Hermite Splines

- Interpolating piecewise cubic polynomial
- Specified with:
  - A pair of control points
  - Tangent at each control point
- Iteratively construct the curve between adjacent end points
Specific Example: Hermite Splines

- Interpolating piecewise cubic polynomial
- Specified with:
  - A pair of control points
  - Tangent at each control point
- Iteratively construct the curve between adjacent end points
Specific Example: Hermite Splines

- Interpolating piecewise cubic polynomial

- Specified with:
  - A pair of control points
  - Tangent at each control point

- Iteratively construct the curve between adjacent end points
Specific Example: Hermite Splines

- Interpolating piecewise *cubic* polynomial
- Specified with:
  - A pair of control points
  - Tangent at each control point
- Iteratively construct the curve between adjacent end points
Specific Example: Hermite Splines

- Interpolating piecewise cubic polynomial

- Specified with:
  - A pair of control points
  - Tangent at each control point

- Iteratively construct the curve between adjacent end points
Specific Example: Hermite Splines

- Interpolating piecewise cubic polynomial

- Specified with:
  - A pair of control points
  - Tangent at each control point

- Iteratively construct the curve between adjacent end points

Because the end-points of adjacent curves share the same position and derivatives, the Hermite spline has \( C^1 \) continuity.
Let $P_k(u) = (P_{k,x}(u), P_{k,y}(u))$ with $0 \leq u \leq 1$ be a parametric cubic point function for the curve section between control points $p_k$ and $p_{k+1}$.

- Boundary conditions are:
  - $P_k(0) = p_k$
  - $P_k(1) = p_{k+1}$
  - $P_k'(0) = Dp_k$
  - $P_k'(1) = Dp_{k+1}$
Specific Example: Hermite Splines

- Let $P_k(u) = (P_{k,X}(u), P_{k,Y}(u))$ with $0 \leq u \leq 1$ be a parametric cubic point function for the curve section between control points $p_k$ and $p_{k+1}$.

- Boundary conditions are:
  - $P_k(0) = p_k$
  - $P_k(1) = p_{k+1}$
  - $P_k'(0) = Dp_k$
  - $P_k'(1) = Dp_{k+1}$

- Solve for the coefficients of the polynomials $P_{k,X}(u)$ and $P_{k,Y}(u)$ that satisfy the boundary condition.
Specific Example: Hermite Splines

We can express the polynomials:
• \( P(u) = au^3 + bu^2 + cu + d \)
• \( P'(u) = 3au^2 + 2bu + c \)
using the matrix representations:

\[
\begin{align*}
P(u) &= \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \\
P'(u) &= \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}
\end{align*}
\]
Specific Example: Hermite Splines

We can express the polynomials:

- \( P(u) = au^3 + bu^2 + cu + d \)
- \( P'(u) = 3au^2 + 2bu + c \)

using the matrix representations:

\[
P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}
P'(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}
\]

By abuse of notation, we will think of the coefficients \( a, b, c, \) and \( d \) as 2-vectors rather than scalars so that \( P \) is a function taking values in 2D.
Specific Example: Hermite Splines

Given the matrix representations:

\[ P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \\ a \\ b \\ c \\ d \end{bmatrix} \quad P'(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \\ a \\ b \\ c \\ d \end{bmatrix} \]
Specific Example: Hermite Splines

Given the matrix representations:

\[ P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \]
\[ P'(u) = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix} \]

we can express the values at the end-points as:

\[ p_k = P(0) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \]
\[ p_{k+1} = P(1) = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \]
\[ Dp_k = P'(0) = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \]
\[ Dp_{k+1} = P'(1) = \begin{bmatrix} 3 & 2 & 1 & 0 \end{bmatrix} \]
Specific Example: Hermite Splines

We can combine the equations

\[ p_k = P(0) = \begin{bmatrix} 0 & 0 & 0 & 1 \\ a \\ b \\ c \\ d \end{bmatrix} \quad \text{and} \quad Dp_k = P'(0) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ a \\ b \\ c \\ d \end{bmatrix} \]

\[ p_{k+1} = P(1) = \begin{bmatrix} 1 & 1 & 1 \\ a \\ b \\ c \\ d \end{bmatrix} \quad \text{and} \quad Dp_{k+1} = P'(1) = \begin{bmatrix} 3 & 2 & 1 & 0 \\ a \\ b \\ c \\ d \end{bmatrix} \]

into a single matrix expression:
Specific Example: Hermite Splines

We can combine the equations into a single matrix expression:

\[
\begin{align*}
    p_k &= P(0) = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \\
    Dp_k &= P'(0) = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \\
    p_{k+1} &= P(1) = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \\
    Dp_{k+1} &= P'(1) = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}
\end{align*}
\]

into a single matrix expression:

\[
\begin{bmatrix}
    p_k \\
    p_{k+1} \\
    Dp_k \\
    Dp_{k+1}
\end{bmatrix} =
\begin{bmatrix}
    0 & 0 & 0 & 1 \\
    1 & 1 & 1 & 1 \\
    0 & 0 & 1 & 0 \\
    3 & 2 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
    a \\
    b \\
    c \\
    d
\end{bmatrix}
\]
Specific Example: Hermite Splines

Inverting the matrix in the equation:

\[
\begin{bmatrix}
  p_k \\
  p_{k+1} \\
  Dp_k \\
  Dp_{k+1}
\end{bmatrix} =
\begin{bmatrix}
  0 & 0 & 0 & 1 \\
  1 & 1 & 1 & 1 \\
  0 & 0 & 1 & 0 \\
  3 & 2 & 1 & 0
\end{bmatrix}\begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix}
\]

we get:
Specific Example: Hermite Splines

Inverting the matrix in the equation:

\[
\begin{bmatrix}
p_k \\
p_{k+1} \\
Dp_k \\
Dp_{k+1}
\end{bmatrix} = \begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
3 & 2 & 1 & 0
\end{bmatrix} \begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix}
\]

we get:

\[
\begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix} = \begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
3 & 2 & 1 & 0
\end{bmatrix}^{-1} \begin{bmatrix}
p_k \\
p_{k+1} \\
Dp_k \\
Dp_{k+1}
\end{bmatrix}
\]
Specific Example: Hermite Splines

Inverting the matrix in the equation:

\[
\begin{bmatrix}
p_k \\
p_{k+1} \\
Dp_k \\
Dp_{k+1}
\end{bmatrix} =
\begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
3 & 2 & 1 & 0
\end{bmatrix}\begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix}
\]

we get:

\[
\begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix} = \begin{bmatrix}
0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
3 & 2 & 1 & 0
\end{bmatrix}^{-1}\begin{bmatrix}
p_k \\
p_{k+1} \\
Dp_k \\
Dp_{k+1}
\end{bmatrix} =
\begin{bmatrix}
2 & -2 & 1 & 1 \\
-3 & 3 & -2 & -1 \\
0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}\begin{bmatrix}
p_k \\
p_{k+1} \\
Dp_k \\
Dp_{k+1}
\end{bmatrix}
\]
Specific Example: Hermite Splines

Using the facts that:

\[
\begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix} \quad \text{and} \quad P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}
\]

we get:
Specific Example: Hermite Splines

Using the facts that:

\[
\begin{bmatrix}
  a \\
  b \\
  c \\
  d \\
\end{bmatrix} = \begin{bmatrix}
  2 & -2 & 1 & 1 \\
  -3 & 3 & -2 & -1 \\
  0 & 0 & 1 & 0 \\
  1 & 0 & 0 & 0 \\
\end{bmatrix}\begin{bmatrix}
  p_k \\
  p_{k+1} \\
  Dp_k \\
  Dp_{k+1} \\
\end{bmatrix}
\]

and

\[
P(u) = \begin{bmatrix}
  u^3 & u^2 & u & 1 \\
\end{bmatrix}
\]

we get:

\[
P(u) = \begin{bmatrix}
  u^3 & u^2 & u & 1 \\
\end{bmatrix}\begin{bmatrix}
  2 & -2 & 1 & 1 \\
  -3 & 3 & -2 & -1 \\
  0 & 0 & 1 & 0 \\
  1 & 0 & 0 & 0 \\
\end{bmatrix}\begin{bmatrix}
  a \\
  b \\
  c \\
  d \\
\end{bmatrix}
\]
Specific Example: Hermite Splines

and we can execute matrix multiplies below

\[
P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix}
\]

to get

\[
P(u) = p_k (2u^3 - 3u^2 + 1) + p_{k+1} (-2u^3 + 3u^2) + \\
Dp_k (u^3 - 2u^2 + u) + Dp_{k+1} (u^3 - u^2)
\]
Specific Example: Hermite Splines

Setting:
\[ oH_0(u) = 2u^3 - 3u^2 + 1 \]
\[ oH_1(u) = -2u^3 + 3u^2 \]
\[ oH_2(u) = u^3 - 2u^2 + u \]
\[ oH_3(u) = u^3 - u^2 \]

we can re-write the equation:

\[
P(u) = p_k (2u^3 - 3u^2 + 1) + p_{k+1} (-2u^3 + 3u^2) + \\
Dp_k (u^3 - 2u^2 + u) + Dp_{k+1} (u^3 - u^2)
\]

\[
P(u) = p_k H_0(u) + p_{k+1} H_1(u) + Dp_k H_2(u) + Dp_{k+1} H_3(u)
\]
Specific Example: Hermite Splines

Setting:

\[ \begin{align*}
\phi_0(u) &= 2u^3 - 3u^2 + 1 \\
\phi_1(u) &= -2u^3 + 3u^2 \\
\phi_2(u) &= u^3 - 2u^2 + u \\
\phi_3(u) &= u^3 - u^2
\end{align*} \]

\[ P(u) = p_k \phi_0(u) + p_{k+1} \phi_1(u) + Dp_k \phi_2(u) + Dp_{k+1} \phi_3(u) \]
Specific Example: Hermite Splines

Setting:

- \( H_0(u) = 2u^3 - 3u^2 + 1 \)
- \( H_1(u) = -2u^3 + 3u^2 \)
- \( H_2(u) = u^3 - 2u^2 + u \)
- \( H_3(u) = u^3 - u^2 \)

When \( u = 0 \):

- \( H_0(0) = 1 \)
- \( H_1(0) = 0 \)
- \( H_2(0) = 0 \)
- \( H_3(0) = 0 \)

So \( P(0) = p_k \)

\[
P(u) = p_k H_0(u) + p_{k+1} H_1(u) + Dp_k H_2(u) + Dp_{k+1} H_3(u)
\]
Specific Example: Hermite Splines

Setting:

- \( H_0(u) = 2u^3 - 3u^2 + 1 \)
- \( H_1(u) = -2u^3 + 3u^2 \)
- \( H_2(u) = u^3 - 2u^2 + u \)
- \( H_3(u) = u^3 - u^2 \)

When \( u = 1 \):
- \( H_0(u) = 0 \)
- \( H_1(u) = 1 \)
- \( H_2(u) = 0 \)
- \( H_3(u) = 0 \)

So \( P(1) = p_{k+1} \)

\[
P(u) = p_k H_0(u) + p_{k+1} H_1(u) + Dp_k H_2(u) + Dp_{k+1} H_3(u)
\]
Specific Example: Hermite Splines

Setting:

- \( H_0(u) = 2u^3 - 3u^2 + 1 \)
- \( H_1(u) = -2u^3 + 3u^2 \)
- \( H_2(u) = u^3 - 2u^2 + u \)
- \( H_3(u) = u^3 - u^2 \)

When \( u = 0 \):

- \( H_0'(u) = 0 \)
- \( H_1'(u) = 0 \)
- \( H_2'(u) = 1 \)
- \( H_3'(u) = 0 \)

So \( P'(0) = Dp_k \)

\[
P'(u) = p_k H_0'(u) + p_{k+1} H_1'(u) + Dp_k H_2'(u) + Dp_{k+1} H_3'(u)
\]
Specific Example: Hermite Splines

Setting:

- $H_0(u) = 2u^3 - 3u^2 + 1$
- $H_1(u) = -2u^3 + 3u^2$
- $H_2(u) = u^3 - 2u^2 + u$
- $H_3(u) = u^3 - u^2$

When $u=1$:
- $H_0'(u) = 0$
- $H_1'(u) = 0$
- $H_2'(u) = 0$
- $H_3'(u) = 1$

So $P'(1) = Dp_{k+1}$

\[ P'(u) = p_k H_0'(u) + p_{k+1} H_1'(u) + Dp_k H_2'(u) + Dp_{k+1} H_3'(u) \]
Specific Example: Hermite Splines

• Interpolating piecewise *cubic* polynomial

• Specified with:
  - Set of control points
  - Tangent at each control point

• Iteratively construct the curve between adjacent end points

Given the control points, how do we define the value of the tangents/derivatives?
Overview

• What is a Spline?

• Specific Examples:
  - Hermite Splines
  - Cardinal Splines
  - Uniform Cubic B-Splines

• Comparing Cardinal Splines to Uniform Cubic B-Splines
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

• Interpolating piecewise *cubic* polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise \textit{cubic} polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents

![Diagram of Cardinal Splines with control points labeled as $p_0$, $p_1$, $p_2$, $p_3$, $p_4$, $p_5$, $p_6$, $p_7$.]
Specific Example: Cardinal Splines

- Interpolating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents

![Cardinal Splines Diagram]
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents

Specific Example: Cardinal Splines

• Interpolating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents

Because the end-points of adjacent curves share the same position and derivatives, the Cardinal spline has $C^1$ continuity.
Specific Example: Cardinal Splines

- Let \( P_k(u)=(P_{k,X}(u),P_{k,Y}(u)) \) with \( 0 \leq u \leq 1 \) be a parametric cubic point function for the curve section between control points \( p_k \) and \( p_{k+1} \)

- Boundary conditions are:
  \[
  \begin{align*}
  &P(0) = p_k \\
  &P(1) = p_{k+1} \\
  &P'(0) = \frac{1}{2}(1-t)(p_{k+1}-p_{k-1}) \\
  &P'(1) = \frac{1}{2}(1-t)(p_{k+2}-p_k)
  \end{align*}
  \]

- Solve for the coefficients of the polynomials \( P_{k,X}(u) \) and \( P_{k,Y}(u) \) that satisfy the boundary condition
Specific Example: Cardinal Splines

Recall:

The Hermite matrix determines the coefficients of the polynomial from the positions and the derivatives of the end-points

\[
P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix}
\]

parameters \quad M_{\text{Hermite}} \quad \text{boundary info}
Specific Example: Cardinal Splines

Using same methods as with Hermite spline, from boundary conditions on previous slide we can get

\[ P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_{k+1} \\ s(p_{k+1} - p_{k-1}) \\ s(p_{k+2} - p_k) \end{bmatrix} \]

where \( s = (1 - t)/2 \)

The parameter \( t \) is called the \textit{tension parameter}.

- Controls looseness versus tightness of curve.
Specific Example: Cardinal Splines

We can express the boundary conditions as a matrix applied to the points \( p_{k-1}, p_k, p_{k+1}, \) and \( p_{k+2} \):

\[
\begin{bmatrix}
  p_k \\
p_{k+1} \\
s(p_{k+1} - p_{k-1}) \\
s(p_{k+2} - p_k)
\end{bmatrix}
= \begin{bmatrix}
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  -s & 0 & s & 0 \\
  0 & -s & 0 & s
\end{bmatrix}
\begin{bmatrix}
p_{k-1} \\
p_k \\
p_{k+1} \\
p_{k+2}
\end{bmatrix}
\]

to get
Specific Example: Cardinal Splines

We can express the boundary conditions as a matrix applied to the points $p_{k-1}$, $p_k$, $p_{k+1}$, and $p_{k+2}$:

\[
\begin{bmatrix}
  p_k \\
p_{k+1} \\
s(p_{k+1} - p_{k-1}) \\
s(p_{k+2} - p_k)
\end{bmatrix} =
\begin{bmatrix}
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  -s & 0 & s & 0 \\
  0 & -s & 0 & s
\end{bmatrix}
\begin{bmatrix}
p_{k-1} \\
p_k \\
p_{k+1} \\
p_{k+2}
\end{bmatrix}
\]

to get

\[
P(u) =
\begin{bmatrix}
u^3 & u^2 & u & 1
\end{bmatrix}
\begin{bmatrix}
  2 & -2 & 1 & 1 \\
  -3 & 3 & -2 & -1 \\
  0 & 0 & 1 & 0 \\
  1 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  -s & 0 & s & 0 \\
  0 & -s & 0 & s
\end{bmatrix}
\begin{bmatrix}
p_{k-1} \\
p_k \\
p_{k+1} \\
p_{k+2}
\end{bmatrix}
\]
Specific Example: Cardinal Splines

Multiplying the interior matrices in:

\[
P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s & 0 & s & 0 \\ 0 & -s & 0 & s \end{bmatrix} \begin{bmatrix} p_{k-1} \\ p_k \\ p_{k+1} \\ p_{k+2} \end{bmatrix}
\]

we get the Cardinal matrix representation.
Specific Example: Cardinal Splines

Combining the matrices in:

\[
P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s & 0 & s & 0 \\ 0 & -s & 0 & s \end{bmatrix} \begin{bmatrix} p_{k-1} \\ p_k \\ p_{k+1} \\ p_{k+2} \end{bmatrix}
\]

we get the Cardinal matrix representation

\[
P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} -s & 2-s & s-2 & s \\ 2s & s-3 & 3-2s & -s \\ -s & 0 & s & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{k-1} \\ p_k \\ p_{k+1} \\ p_{k+2} \end{bmatrix}
\]
Specific Example: Cardinal Splines

Setting:

- \( C_0(u) = -su^3 + 2su^2 - su \)
- \( C_1(u) = (2-s)u^3 + (s-3)u^2 + 1 \)
- \( C_2(u) = (s-2)u^3 + (3-2s)u^2 + su \)
- \( C_3(u) = su^3 - su^2 \)

For \( s=0 \):

\[
P(u) = C_0(u)p_{k-1} + C_1(u)p_k + C_2(u)p_{k+1} + C_3(u)p_{k+2}
\]
Specific Example: Cardinal Splines

Setting:
- \( C_0(u) = -su^3 + 2su^2 - su \)
- \( C_1(u) = (2-s)u^3 + (s-3)u^2 + 1 \)
- \( C_2(u) = (s-2)u^3 + (3-2s)u^2 + su \)
- \( C_3(u) = su^3 - su^2 \)

For \( s=0 \):

\[
P(u) = C_0(u)p_{k-1} + C_1(u)p_k + C_2(u)p_{k+1} + C_3(u)p_{k+2}
\]

Properties:
- \( C_0(u) + C_1(u) + C_2(u) + C_3(u) = 1 \)
- \( C_j(u) = C_{3-j}(1-u) \)
- \( C_0(1) = C_3(0) = 0 \)
Specific Example: Cardinal Splines

- Interpolating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents
Specific Example: Cardinal Splines

- Interpolating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve between middle two points using adjacent points to define tangents

At the first and last end-points, you can:
- Not draw the final segments
- Double up end points
- Loop the spline around
Overview

• What is a Spline?

• Specific Examples:
  o Hermite Splines
  o Cardinal Splines
  o Uniform Cubic B-Splines

• Comparing Cardinal Splines to Uniform Cubic B-Splines
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise \textit{cubic} polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

• Approximating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

• Approximating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise cubic polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

• Let $P_k(u) = (P_{k,X}(u), P_{k,Y}(u))$ with $0 \leq u \leq 1$ be a parametric cubic point function for the curve section around the control points $p_k$ and $p_{k+1}$

• Boundary conditions are:
  - $P(0) = 1/6(p_{k-1} + 4p_k + p_{k+1})$
  - $P(1) = 1/6(p_k + 4p_{k+1} + p_{k+2})$
  - $P'(0) = 1/2(1 - t)(p_{k+1} - p_{k-1})$
  - $P'(1) = 1/2(1 - t)(p_{k+2} - p_k)$

• Solve for the coefficients of the polynomials $P_{k,X}(u)$ and $P_{k,Y}(u)$ that satisfy the boundary condition
Specific Example: Uniform Cubic B-Splines

Using same methods as with Hermite spline, from boundary conditions on previous slide we can get

\[ P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{k-1} + 4p_k + p_{k+1} \\ p_k + 4p_{k+1} + p_{k+2} \\ 3p_{k+1} - 3p_{k-1} \\ 3p_{k+2} - 3p_k \end{bmatrix} \]

\[ M_{Hermite} \]
Specific Example: Uniform Cubic B-Splines

We can express the boundary conditions as a matrix applied to the points $p_{k-1}$, $p_k$, $p_{k+1}$, and $p_{k+2}$:

$$
\begin{bmatrix}
  p_{k-1} + 4p_k + p_{k+1} \\
  p_k + 4p_{k+1} + p_{k+2} \\
  3p_{k+1} - 3p_{k-1} \\
  3p_{k+2} - 3p_k
\end{bmatrix} =
\begin{bmatrix}
  1 & 4 & 1 & 0 \\
  0 & 1 & 4 & 1 \\
  -3 & 0 & 3 & 0 \\
  0 & -3 & 0 & 3
\end{bmatrix}
\begin{bmatrix}
  p_{k-1} \\
  p_k \\
  p_{k+1} \\
  p_{k+2}
\end{bmatrix}
$$

to get

$$P(u) = \frac{1}{6} \begin{bmatrix}
  u^3 & u^2 & u & 1
\end{bmatrix}
\begin{bmatrix}
  2 & -2 & 1 & 1 \\
  -3 & 3 & -2 & -1 \\
  0 & 0 & 1 & 0 \\
  1 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
  1 & 4 & 1 & 0 \\
  0 & 1 & 4 & 1 \\
  -3 & 0 & 3 & 0 \\
  0 & -3 & 0 & 3
\end{bmatrix}
\begin{bmatrix}
  p_{k-1} \\
  p_k \\
  p_{k+1} \\
  p_{k+2}
\end{bmatrix}$$
**Specific Example: Uniform Cubic B-Splines**

Multiplying the interior matrices in:

\[
P(u) = \frac{1}{6} \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ -3 & 0 & 3 & 0 \\ 0 & -3 & 0 & 3 \end{bmatrix} \begin{bmatrix} p_{k-1} \\ p_k \\ p_{k+1} \\ p_{k+2} \end{bmatrix}
\]

we get the cubic B-spline matrix representation.
Specific Example: Uniform Cubic B-Splines

Combining the matrices in:

\[
P(u) = \frac{1}{6} \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ -3 & 0 & 3 & 0 \\ 0 & -3 & 0 & 3 \end{bmatrix} \begin{bmatrix} p_{k-1} \\ p_k \\ p_{k+1} \\ p_{k+2} \end{bmatrix}
\]

we get the cubic B-spline matrix representation

\[
P(u) = \frac{1}{6} \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_{k-1} \\ p_k \\ p_{k+1} \\ p_{k+2} \end{bmatrix}
\]
Specific Example: Uniform Cubic B-Splines

Setting:

- \( B_{0,3}(u) = \frac{1}{6}(1-u)^3 \)
- \( B_{1,3}(u) = \frac{1}{6}(3u^3-6u^2+4) \)
- \( B_{2,3}(u) = \frac{1}{6}(-3u^3+3u^2+3u+1) \)
- \( B_{3,3}(u) = \frac{1}{6}(u^3) \)

Blending Functions

\[
P(u) = B_{0,3}(u)p_{k-1} + B_{1,3}(u)p_k + B_{2,3}(u)p_{k+1} + B_{3,3}(u)p_{k+2}
\]
Specific Example: Uniform Cubic B-Splines

Setting:
- \( B_{0,3}(u) = \frac{1}{6}(1-u)^3 \)
- \( B_{1,3}(u) = \frac{1}{6}(3u^3 - 6u^2 + 4) \)
- \( B_{2,3}(u) = \frac{1}{6}(-3u^3 + 3u^2 + 3u + 1) \)
- \( B_{3,3}(u) = \frac{1}{6}(u^3) \)

Properties:
- \( B_{0,3}(u) + B_{1,3}(u) + B_{2,3}(u) + B_{3,3}(u) = 1 \)
- \( B_j(u) = B_{3-j}(1-u) \)
- \( B_{0,3}(1) = B_{3,3}(0) = 0 \)
- \( B_{j,3}(u) \geq 0 \)

\[
P(u) = B_{0,3}(u)p_{k-1} + B_{1,3}(u)p_k + B_{2,3}(u)p_{k+1} + B_{3,3}(u)p_{k+2}
\]
Specific Example: Uniform Cubic B-Splines

- Approximating piecewise *cubic* polynomial
- Specified with four control points
- Iteratively construct the curve around middle two points using adjacent points to define tangents
Specific Example: Uniform Cubic B-Splines

• Approximating piecewise cubic polynomial
• Specified with four control points
• Iteratively construct the curve around middle two points using adjacent points to define tangents

At the first and last end-points, you can:
• Not draw the final segments
• Double up end points
• Loop the spline around
Overview

• What is a Spline?

• Specific Examples:
  o Hermite Splines
  o Cardinal Splines
  o Uniform Cubic B-Splines

• Comparing Cardinal Splines to Uniform Cubic B-Splines
Blending Functions

Blending functions provide a way for expressing the functions \( P_k(u) \) as a weighted sum of the four control points \( p_{k-1}, p_k, p_{k+1}, \) and \( p_{k+2} \):

\[
P_k(u) = BF_0(u)p_{k-1} + BF_1(u)p_k + BF_2(u)p_{k+1} + BF_3(u)p_{k+2}
\]
Blending Functions

Properties:

• Translation Commutativity:
  \( BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1 \), for all \( 0 \leq u \leq 1 \).

• Continuity:
  \( BF_0(1) = BF_3(0) \), \( BF_1(1) = BF_0(0) \), \( BF_2(1) = BF_1(0) \), \( BF_3(1) = BF_2(0) \).

• Convex Hull Containment:
  \( BF_0(u), BF_1(u), BF_2(u), BF_3(u) \geq 1 \), for all \( 0 \leq u \leq 1 \).

• Interpolation:
  \( BF_0(0) = BF_2(0) = BF_3(0) = 0 \), \( BF_0(1) = BF_1(1) = BF_3(1) = 0 \).
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]
  If we translate all the control points by the same vector \( q \), the position of the new point at the value \( u \) will just be the position of the old value at \( u \), translated by \( q \):
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

  If we translate all the control points by the same vector \( q \), the position of the new point at the value \( u \) will just be the position of the old value at \( u \), translated by \( q \):

  \[
  Q_k(u) = BF_0(u)(q + p_{k-1}) + BF_1(u)(q + p_k) + BF_2(u)(q + p_{k+1}) + BF_3(u)(q + p_{k+2})
  \]
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]
  If we translate all the control points by the same vector \( q \), the position of the new point at the value \( u \) will just be the position of the old value at \( u \), translated by \( q \):

  \[
  Q_k(u) = BF_0(u)(q + p_{k-1}) + BF_1(u)(q + p_k) + BF_2(u)(q + p_{k+1}) + BF_3(u)(q + p_{k+2})
  \]
  \[
  = (BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u))q + P_k(u)
  \]

• Continuity:
  \[ BF_0(1) = BF_3(0) = 0 \]
  \[ BF_1(1) = BF_0(0) \]
  \[ BF_2(1) = BF_1(0) \]
  \[ BF_3(1) = BF_2(0) \]

• Convex Hull Containment:
  \[ BF_0(u), BF_1(u), BF_2(u), BF_3(u) \geq 1, \text{ for all } 0 \leq u \leq 1. \]

• Interpolation:
  \[ BF_0(0) = BF_2(0) = BF_3(0) = 0 \]
  \[ BF_0(1) = BF_1(1) = BF_3(1) = 0 \]
  \[ BF_1(0) = 1 \]
  \[ BF_2(1) = 1 \]
Blending Functions

Properties:

• Translation Commutativity:
  \( BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1 \), for all \( 0 \leq u \leq 1 \).
  
  If we translate all the control points by the same vector \( q \), the position of the new point at the value \( u \) will just be the position of the old value at \( u \), translated by \( q \):

  \[
  Q_k(u) = BF_0(u)(q + p_{k-1}) + BF_1(u)(q + p_k) + BF_2(u)(q + p_{k+1}) + BF_3(u)(q + p_{k+2}) \\
  = (BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u))q + P_k(u) \\
  = q + P_k(u)
  \]
Comparison: Cardinal vs. Cubic B

Cardinal Splines ($t=0$)

- $BF_0(u) = -\frac{1}{2}u^3 + \frac{1}{2}u$  
- $BF_1(u) = \frac{3}{2}u^3 - \frac{5}{2}u^2 + 1$  
- $BF_2(u) = -\frac{3}{2}u^3 + 2u^2 + \frac{1}{2}u$  
- $BF_3(u) = \frac{1}{2}u^3 - \frac{1}{2}u^2$

$BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1$

Cubic B-Splines

- $BF_0(u) = -\frac{1}{6}u^3 + \frac{1}{2}u^2 - \frac{1}{2}u + \frac{1}{6}$  
- $BF_1(u) = \frac{1}{2}u^3 - u^2 + \frac{2}{3}$  
- $BF_2(u) = -\frac{1}{2}u^3 + \frac{1}{2}u^2 + \frac{1}{2}u + \frac{1}{6}$  
- $BF_3(u) = \frac{1}{6}u^3$

$BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1$

$P_k(u) = BF_0(u)p_{k-1} + BF_1(u)p_k + BF_2(u)p_{k+1} + BF_3(u)p_{k+2}$
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

• Continuity:
  \[ BF_0(1) = BF_3(0) = 0 \]
  \[ BF_1(1) = BF_0(0) \]
  \[ BF_2(1) = BF_1(0) \]
  \[ BF_3(1) = BF_2(0) \]
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

• Continuity:
  \[ BF_0(1) = BF_3(0) = 0 \]
  \[ BF_1(1) = BF_0(0) \]
  \[ BF_2(1) = BF_1(0) \]
  \[ BF_3(1) = BF_2(0) \]

We need to have the curve \( P_{k+1}(u) \) begin where the curve \( P_k(u) \) ended:

\[ 0 = P_{k+1}(0) - P_k(1) \]
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

• Continuity:
  \[
  \begin{align*}
  BF_0(1) &= BF_3(0) = 0 \\
  BF_1(1) &= BF_0(0) \\
  BF_2(1) &= BF_1(0) \\
  BF_3(1) &= BF_2(0)
  \end{align*}
  \]

We need to have the curve \( P_{k+1}(u) \) begin where the curve \( P_k(u) \) ended:

\[
0 = P_{k+1}(0) - P_k(1)
\]

Since this equation has to hold true regardless of the values of \( p_k \), the conditions on the left have to be true.

\[
0 = \left( -BF_0(1) \right) p_{k-1} + \left( BF_0(0) - BF_1(1) \right) p_k + \left( BF_1(0) - BF_2(1) \right) p_{k+1} + \left( BF_2(0) - BF_3(1) \right) p_{k+2} + \left( BF_3(0) \right) p_{k+3}
\]
Comparison: Cardinal vs. Cubic B

Cardinal Splines (t=0)

\[ BF_0(u) = -\frac{1}{2} u^3 + u^2 - \frac{1}{2} u \]
\[ BF_1(u) = \frac{3}{2} u^3 - \frac{5}{2} u^2 + 1 \]
\[ BF_2(u) = -\frac{3}{2} u^3 + 2u^2 + \frac{1}{2} u \]
\[ BF_3(u) = \frac{1}{2} u^3 - \frac{1}{2} u^2 \]

\[ BF_0(0) = 0 \quad BF_0(1) = 0 \]
\[ BF_1(0) = 1 \quad BF_1(1) = 0 \]
\[ BF_2(0) = 0 \quad BF_2(1) = 1 \]
\[ BF_3(0) = 0 \quad BF_3(1) = 0 \]

P_k (u) = BF_0 (u) p_{k-1} + BF_1 (u) p_k + BF_2 (u) p_{k+1} + BF_3 (u) p_{k+2}

Cubic B-Splines

\[ BF_0(u) = -\frac{1}{6} u^3 + \frac{1}{2} u^2 - \frac{1}{2} u + \frac{1}{6} \]
\[ BF_1(u) = \frac{1}{2} u^3 - u^2 + \frac{2}{3} \]
\[ BF_2(u) = -\frac{1}{2} u^3 + \frac{1}{2} u^2 + \frac{1}{2} u + \frac{1}{6} \]
\[ BF_3(u) = \frac{1}{6} u^3 \]

\[ BF_0(0) = \frac{1}{6} \quad BF_0(1) = 0 \]
\[ BF_1(0) = \frac{2}{3} \quad BF_1(1) = \frac{1}{6} \]
\[ BF_2(0) = \frac{1}{6} \quad BF_2(1) = \frac{2}{3} \]
\[ BF_3(0) = 0 \quad BF_3(1) = \frac{1}{6} \]
Comparison: Cardinal vs. Cubic B

Cardinal Splines (t=0)

\[ BF_0(u) = -\frac{1}{2}u^3 + u^2 - \frac{1}{2}u \]
\[ BF_1(u) = \frac{3}{2}u^3 - \frac{5}{2}u^2 + 1 \]
\[ BF_2(u) = -\frac{3}{2}u^3 + 2u^2 + \frac{1}{2}u \]
\[ BF_3(u) = \frac{1}{2}u^3 - \frac{1}{2}u^2 \]

\[ BF_0'(0) = -\frac{1}{2} \quad BF_0'(1) = 0 \]
\[ BF_1'(0) = 0 \quad BF_1'(1) = -\frac{1}{2} \]
\[ BF_2'(0) = \frac{1}{2} \quad BF_2'(1) = 0 \]
\[ BF_3'(0) = 0 \quad BF_3'(1) = \frac{1}{2} \]

\[ P_k(u) = BF_0(u)p_{k-1} + BF_1(u)p_k + BF_2(u)p_{k+1} + BF_3(u)p_{k+2} \]

Cubic B-Splines

\[ BF_0(u) = -\frac{1}{6}u^3 + \frac{1}{2}u^2 - \frac{1}{2}u + \frac{1}{6} \]
\[ BF_1(u) = \frac{1}{2}u^3 - u^2 + \frac{2}{3} \]
\[ BF_2(u) = -\frac{1}{2}u^3 + \frac{1}{2}u^2 + \frac{1}{2}u + \frac{1}{6} \]
\[ BF_3(u) = \frac{1}{6}u^3 \]

\[ BF_0'(0) = -\frac{1}{2} \quad BF_0'(1) = 0 \]
\[ BF_1'(0) = 0 \quad BF_1'(1) = -\frac{1}{2} \]
\[ BF_2'(0) = \frac{1}{2} \quad BF_2'(1) = 0 \]
\[ BF_3'(0) = 0 \quad BF_3'(1) = \frac{1}{2} \]
Comparison: Cardinal vs. Cubic B

Cardinal Splines (t=0)

Cardinal B-Splines

\[ BF_0(u) = -\frac{1}{2} u^3 + u^2 - \frac{1}{2} u \]
\[ BF_1(u) = \frac{3}{2} u^3 - \frac{5}{2} u^2 + 1 \]
\[ BF_2(u) = -\frac{3}{2} u^3 + 2u^2 + \frac{1}{2} u \]
\[ BF_3(u) = \frac{1}{2} u^3 - \frac{1}{2} u^2 \]

\[ BF_0''(0) = 2 \quad \text{and} \quad BF_0'''(1) = 5 \]
\[ BF_1''(0) = -5 \quad \text{and} \quad BF_1'''(1) = 4 \]
\[ BF_2''(0) = 4 \quad \text{and} \quad BF_2'''(1) = -5 \]
\[ BF_3''(0) = -1 \quad \text{and} \quad BF_3'''(1) = 2 \]

Cubic B-Splines

Cubic B-Splines

\[ BF_0(u) = -\frac{1}{6} u^3 + \frac{1}{2} u^2 - \frac{1}{2} u + \frac{1}{6} \]
\[ BF_1(u) = \frac{1}{2} u^3 - u^2 + \frac{2}{3} \]
\[ BF_2(u) = -\frac{1}{2} u^3 + \frac{1}{2} u^2 + \frac{1}{2} u + \frac{1}{6} \]
\[ BF_3(u) = \frac{1}{6} u^3 \]

\[ BF_0''(0) = 1 \quad \text{and} \quad BF_0'''(1) = 0 \]
\[ BF_1''(0) = -2 \quad \text{and} \quad BF_1'''(1) = 1 \]
\[ BF_2''(0) = 1 \quad \text{and} \quad BF_2'''(1) = -2 \]
\[ BF_3''(0) = 0 \quad \text{and} \quad BF_3'''(1) = 1 \]

\[ P_k(u) = BF_0(u) p_{k-1} + BF_1(u) p_k + BF_2(u) p_{k+1} + BF_3(u) p_{k+2} \]
Blending Functions

Properties:

- Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

- Continuity:
  \[ BF_0(1) = BF_3(0) = 0 \]
  \[ BF_1(1) = BF_0(0) \]
  \[ BF_2(1) = BF_1(0) \]
  \[ BF_3(1) = BF_2(0) \]
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

• Continuity:
  \[ BF_0(1) = BF_3(0) = 0 \]
  \[ BF_1(1) = BF_0(0) \]
  \[ BF_2(1) = BF_1(0) \]
  \[ BF_3(1) = BF_2(0) \]

• Convex Hull Containment:
  \[ BF_0(u), BF_1(u), BF_2(u), BF_3(u) \geq 0, \text{ for all } 0 \leq u \leq 1. \]

This is because a point is inside the convex hull of a collection of points if and only if it can be expressed as the weighted average of the points, where all the weights are non-negative.
Comparison: Cardinal vs. Cubic B

Cardinal Splines (t=0)  Cubic B-Splines

\[ p_k(u) = BF_0(u)p_{k-1} + BF_1(u)p_k + BF_2(u)p_{k+1} + BF_3(u)p_{k+2} \]
**Comparison: Cardinal vs. Cubic B**

<table>
<thead>
<tr>
<th>Cardinal Splines (t=0)</th>
<th>Cubic B-Splines</th>
</tr>
</thead>
</table>

\[ P_k(u) = BF_0(u)p_{k-1} + BF_1(u)p_k + BF_2(u)p_{k+1} + BF_3(u)p_{k+2} \]
Blending Functions

Properties:

• Translation Commutativity:
  \( BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1 \), for all \( 0 \leq u \leq 1 \).

• Continuity:
  \( BF_0(1) = BF_3(0) = 0 \)
  \( BF_1(1) = BF_0(0) \)
  \( BF_2(1) = BF_1(0) \)
  \( BF_3(1) = BF_2(0) \)

• Convex Hull Containment:
  \( BF_0(u), BF_1(u), BF_2(u), BF_3(u) \geq 0 \), for all \( 0 \leq u \leq 1 \).

• Interpolation:
  \( BF_0(0) = BF_2(0) = BF_3(0) = 0 \)
  \( BF_0(1) = BF_1(1) = BF_3(1) = 0 \)
  \( BF_1(0) = 1 \)
  \( BF_2(1) = 1 \)
Blending Functions

Properties:

• Translation Commutativity:
  \[ BF_0(u) + BF_1(u) + BF_2(u) + BF_3(u) = 1, \text{ for all } 0 \leq u \leq 1. \]

• Continuity:
  \[ BF_0(1) = BF_3(0) = 0 \]
  \[ BF_1(1) = BF_0(0) \]
  \[ BF_2(1) = BF_1(0) \]
  \[ BF_3(1) = BF_2(0) \]

• Convex Hull Containment:
  \[ BF_0(u), BF_1(u), BF_2(u), BF_3(u) \geq 0, \text{ for all } 0 \leq u \leq 1. \]

• Interpolation:
  \[ BF_0(0) = BF_2(0) = BF_3(0) = 0 \]
  \[ BF_0(1) = BF_1(1) = BF_3(1) = 0 \]
  \[ BF_1(0) = 1 \]
  \[ BF_2(1) = 1 \]

Because we want the spline segments to satisfy:

• \( P_k(0) = p_{k+1} \)
• \( P_k(1) = p_{k+2} \)
Comparison: Cardinal vs. Cubic B

**Cardinal Splines (t=0)**

- \( BF_0(u) = -\frac{1}{2} u^3 + u^2 - \frac{1}{2} u \)
- \( BF_1(u) = \frac{2}{3} u^3 - \frac{5}{2} u^2 + u + 1 \)
- \( BF_2(u) = -\frac{3}{2} u^3 + 2u^2 + \frac{1}{2} u \)
- \( BF_3(u) = \frac{1}{2} u^3 - \frac{1}{2} u^2 \)

**Cubic B-Splines**

- \( BF_0(u) = -\frac{1}{6} u^3 + \frac{1}{2} u^2 - \frac{1}{2} u + \frac{1}{6} \)
- \( BF_1(u) = \frac{1}{3} u^3 - u^2 + \frac{2}{3} \)
- \( BF_2(u) = -\frac{1}{2} u^3 + \frac{1}{2} u^2 + \frac{1}{2} u + \frac{1}{6} \)
- \( BF_3(u) = \frac{1}{6} u^3 \)

**Support Completion**

\[ P_k(u) = BF_0(u)p_{k-1} + BF_1(u)p_k + BF_2(u)p_{k+1} + BF_3(u)p_{k+2} \]
Summary

• A spline is a *piecewise polynomial function* whose derivatives satisfy some *continuity constraints* across curve junctions.

• Looked at specification for 3 splines:
  - Hermite
  - Cardinal
  - Uniform Cubic B-Spline

  \begin{align*}
  \{ & \text{Interpolating, cubic, } C^1 \\
  & \text{Approximating, convex-hull containment, } \\
  & \text{cubic, } C^2 \}
  \end{align*}