Algorithmic Geometry WS 2017/2018

Prof. Dr. Hans Hagen
Benjamin Karer M.Sc.

http://gfx.uni-kl.de/~alggeom

B-Spline Curves

B-Splines

Idea: spline functions with local support

recursion formula De Boor - Cox

$$\kappa = (x_i)_{i \in \mathbb{N}_0}, x_0 < x_1 < \ldots < x_k$$

(i) $$N_{i,0}(s) := \begin{cases} 1 & x_i \leq s < x_{i+1} \\ 0 & \text{elsewhere} \end{cases}$$

(ii) $$N_{i,n}(s) := \left(\frac{s - x_i}{x_{i+n} - x_i}\right) N_{i,n-1}(s) + \left(\frac{x_{i+n+1} - s}{x_{i+n+1} - x_{i+1}}\right) N_{i+1,n-1}(s)$$

(iii) $0 \leq i \leq k - n$

k : number of intervals

n : polynomial degree

B-Spline Curves

Remarks

1. The "supporting" interval of $N_{i,n}(s)$ is $[x_i, x_{i+1}]$.
2. $N_{i,n}(s) > 0, \quad x_i < s < x_{i+n}$
   $N_{i,n}(s) = 0, \quad s \notin [x_i, x_{i+n}]$
3. If $\tau = (0, 1, \ldots, n)$ then $N_{i,0}(s) = N_{i+1,0}(s+1)$
4. The Bernstein polynomials $B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}, 0 \leq i \leq m$ are special "degenerated" B-splines. Their supporting interval is $\tau = (\underbrace{0, \ldots, 0}_{m+1}; \underbrace{1, \ldots, 1}_{m+1})$

B-Spline Curves

5. For different knots we have $N_{i,n}(s) \in C^{n-1}$. The differentiability reduces to C^{n-l_i} for each knot with multiplicity l_i.
6. Input of the De Boor - algorithm: $(k+1)$ knots, order $\sigma(k-\sigma)$

B-Spline Curves

Basis functions

$$f(t) = \sum_{i=0}^{r} a_i \cdot N_{i,\sigma}(t) \qquad r := k + \sigma - 1$$

Basis function derivatives

$$f^{(j)}(t) = \sum_{i=0}^{r} a_i^{(j)} \cdot N_{i,\sigma-j}(t) \quad j \leq 0$$

with $$a_i^{(j)} = \begin{cases} a_i & \text{for } j = 0 \\ (\sigma - j) \cdot \frac{a_i^{(j-1)} - a_{i-1}^{(j-1)}}{t_{i+\sigma-j} - t_i} & \text{for } j > 0 \end{cases}$$

and $$a_0^{(j)} := 0 \quad \text{for } j \leq 0.$$

Idea: Use B-Splines instead of Bernstein polynomials as blending functions.

Definition

B (Basis)-Spline curves

An *open B-Spline curve* of degree $M-1$ associated with the control polygon P_n with vertices $d_0, \ldots, d_n,\ n \geq M-1$) is given by:

$$y(s) := \sum_{i=0}^{n} d_i \cdot N_{i,M}(s)$$

with $0 \leq s \leq \tilde{x}_{n-M+2}$ and the "inner knot vector" $\tilde{\kappa} = (\tilde{x}_0, \ldots, \tilde{x}_{n-M+2});\ \tilde{x}_{i-1} < \tilde{x}_i$ for $i = 1, \ldots, n-M+2$ being the "core" of the knot vector $\kappa = (x_0, \ldots, x_{n+M})$.

B-Spline Curves

Definition (continued)

$$\begin{aligned} x_i &= \tilde{x}_0 & i &= 0, 1, \ldots, M-2 \\ x_{i+M-1} &= \tilde{x}_i & i &= 0, 1, \ldots, n-M+2 \\ x_{i+n+2} &= \tilde{x}_{n-M+2} & i &= 0, 1, \ldots, M-2 \end{aligned}$$

Definition

For a *closed B-Spline curve*, using $X = (x_0, \ldots, x_{n+1})$ with $x_i = (i - M\text{div}2) \mod n+1$ for $i = 0, \ldots, n+1$

B-Spline Curves

Remarks

Each control point movement only has local effects.
Choosing the knot vector in this way guarantees the start and end tangent property.
The key idea in the case of closed B-Spline curves is to use a "middle" B-Spline as blending function.

B-Spline Curves

The supporting intervals of a B-Spline curve are:

$$
\begin{aligned}
y(s) \quad = \quad & d_0 N_{0,M} && + d_1 N_{1,M} + \cdots + && d_n N_{n,M} \\
& \uparrow && && \\
& \{x_0, \ldots, x_M\} && && \\
& && \uparrow && \\
& && \{x_1, \ldots, x_{M+1}\} && \\
& && && \uparrow \\
& && && \{x_n, \ldots, x_{M+n}\}
\end{aligned}
$$

B-Spline Curves

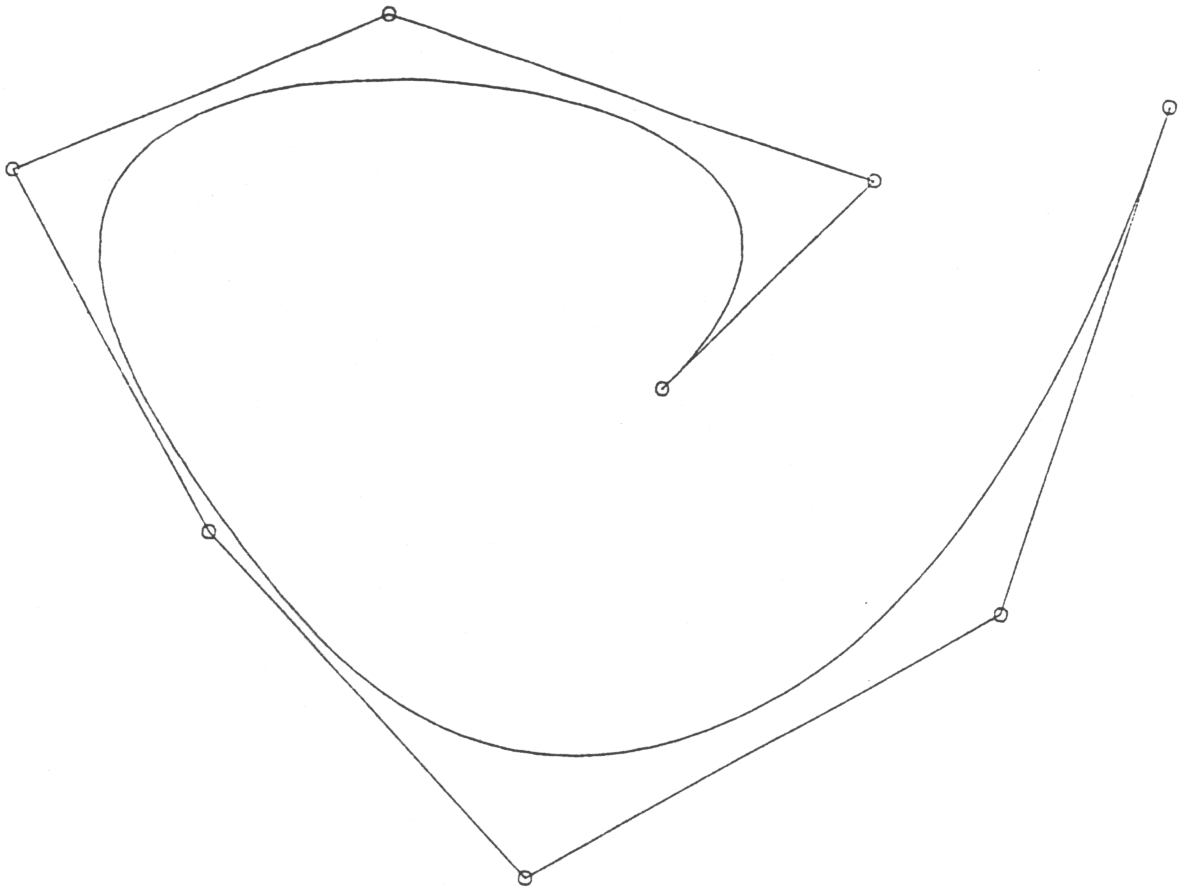


Figure: Example for an open B-Spline curve and associated control polygon

Definition

Convex Hull and Variation Diminishing Property

1. Each straight line does not intersect a B-Spline curve more often than it intersects the control polygon.
2. The B-Spline curve is contained completely inside the convex hull of the control polygon.

Remarks

In the case of $d_i = d_{i+1} = \cdots = d_{i+M-1}$ the B-Spline curve must go through the point d_i, since the convex hull consists of only the point d_i. The straight line containing M vertices of the control structure is part of the B-Spline curve.

Algorithm for B-Spline curves

Input:

- control polygon $d_0, \ldots, d_n$
- Spline order M with $(n \geq M - 1)$
- The "inner" knot vector $\tilde{\kappa} = (\tilde{x}_0, \ldots, \tilde{x}_{n-M+2})$ with $\tilde{x}_{i-1} < \tilde{x}_i$.

1.
   1. For a closed B-Spline curve choose the knot vector $\kappa = \{x_0, \ldots, x_{n+1}\}$ with $x_i = (i - M\text{div}2) \mod (n+1)$ for $i = 0, \ldots, n+1$.
   2. For an open B-Spline curve choose the knot vector $\kappa = \{x_0, \ldots, x_{n+M}\}$ with

$$\begin{aligned} x_i &= \tilde{x}_0 && \text{for } i = 0, \ldots, M-2 \\ x_{i+M-1} &= \tilde{x}_i && \text{for } i = 0, \ldots, n-M+2 \\ x_{i+n+2} &= \tilde{x}_{n-M+2} && \text{for } i = 0, \ldots, M-2 \end{aligned}$$

2. look for i with $x_i \leq s_0 \leq x_{i+1}$.

Algorithm for B-Spline curves (continued)

3

$$
\begin{aligned}
d^{(1)}_{i-M+2} &:= \lambda^{(1)}_{i-M+2} \cdot d_{i-M+2} + \left(1 - \lambda^{(1)}_{i-M+2}\right) \cdot d_{i-M+1} \\
d^{(1)}_{i-M+3} &:= \lambda^{(1)}_{i-M+3} \cdot d_{i-M+3} + \left(1 - \lambda^{(1)}_{i-M+3}\right) \cdot d_{i-M+2} \\
&\vdots \qquad\qquad \vdots \\
d^{(1)}_{i} &:= \lambda^{(1)}_{i} \cdot d_{i} + \left(1 - \lambda^{(1)}_{i}\right) \cdot d_{i-1}
\end{aligned}
$$

Algorithm for B-Spline curves (continued)

4.

$$
\begin{aligned}
d^{(2)}_{i-M+3} &:= \lambda^{(2)}_{i-M+3} \cdot d^{(1)}_{i-M+3} + \left(1 - \lambda^{(2)}_{i-M+3}\right) \cdot d^{(1)}_{i-M+2} \\
d^{(2)}_{i-M+4} &:= \lambda^{(2)}_{i-M+4} \cdot d^{(1)}_{i-M+4} + \left(1 - \lambda^{(2)}_{i-M+4}\right) \cdot d^{(1)}_{i-M+3} \\
&\vdots \qquad\qquad \vdots \\
d^{(2)}_{i} &:= \lambda^{(2)}_{i} \cdot d^{(1)}_{i} + \left(1 - \lambda^{(2)}_{i}\right) \cdot d^{(1)}_{i-1}
\end{aligned}
$$

Algorithm for B-Spline curves (continued)

Step $M+1$

$$d_i^{(M-1)} := \lambda_i^{(M-1)} \cdot d_i^{(M-2)} + \left(1 - \lambda_i^{(M-1)}\right) \cdot d_{i-1}^{(M-2)}$$

Step $M+2$

$$y(s_0) := d_i^{(M-1)}$$

$$\text{with } \lambda_j^{(k)} = \frac{s_0 - x_j}{x_{j+M-k} - x_j}$$

$$\text{and } j = i - M + k + 1, \ldots, i$$

$$\text{and } k = 1, \ldots, M-1$$

B-Spline Curves

Remarks

For a closed B-Spline curve we have:

$$\lambda_j^{(k)} = \frac{s_0 - x_j}{M - k}$$

In the special case of a knot vector $\kappa = \{\overbrace{0, \ldots, 0}^{M}, \overbrace{1, \ldots, 1}^{M}\}$, the B-Splines degenerate to Bernstein polynomials.

B-Spline Curves

Bézier curves are degenerated B-Spline curves!

example

Input: closed polygon $d_0, d_1, \ldots, d_{12}$; order $M = 4$; $s_0 = 7.6$
knot vector $\kappa = (0, 1, \ldots, 13)$

1.

$$\begin{aligned} x_0 &= (0-2) \mod 13 = 11 \\ x_1 &= (1-2) \mod 13 = 12 \\ x_2 &= 0 \equiv 13 (\mod 13) \end{aligned}$$

$$\begin{array}{llll} x_3 = 1 & x_6 = 4 & x_9 = 7 & x_{12} = 10 \\ x_4 = 2 & x_7 = 5 & x_{10} = 8 & x_{13} = 11 = x_0 \\ x + 5 = 3 & x_8 = 6 & x_{11} = 9 & \end{array}$$

example (cont.)

2. $x_9 < 7.6 < x_1 0$
3.

$$
\begin{aligned}
d_7^{(1)} &:= \lambda_7^{(1)} \cdot d_7 + \left(1 - \lambda_7^{(1)}\right) \cdot d_7 \\
&= \frac{7.6 - 5}{4 - 1} \cdot d_7 + \left(1 - \frac{2.6}{3}\right) \cdot d_6 \\
&= 0.87 \cdot d_7 + (1 - 2.6)\, d_6 \\
d_8^{(1)} &= 0.53 \cdot d_8 + 0.47 \cdot d_7 \\
d_9^{(1)} &= 0.2 \cdot d_9 + 0.8 \cdot d_8
\end{aligned}
$$

example (cont.)

4.
$$\begin{aligned} d_8^{(2)} &= 0.8 d_8^{(1)} + 0.2 d_7^{(1)} \\ d_9^{(2)} &= 0.3 d_9^{(1)} + 0.7 d_8^{(1)} \end{aligned}$$

5. $d_9^{(3)} = 0.6 d_9^{(2)} + 0.4 d_8^{(2)}$

6. $d_9^{(3)} = y(7.6)$

Scheme of the Algorithm for B-Spline curves

$$
\begin{array}{llll}
d_{i-M+1} & & & \\
d_{i-M+2} & d_{i-M+2}^{(1)} & & \\
\vdots & \vdots & \ddots & \\
d_i & d_i^{(1)} & \cdots & d_i^{(M-1)}
\end{array}
$$

Example

$$s_0 = 1.5, d_0 = \begin{pmatrix} -2 \\ 0 \end{pmatrix}, d_1 = \begin{pmatrix} 0 \\ 2 \end{pmatrix}, d_2 = \begin{pmatrix} 2 \\ 0 \end{pmatrix}, d_3 = \begin{pmatrix} 0 \\ -2 \end{pmatrix}, M = 3$$

Inner knot vector:

$\tilde{\kappa} = \{\tilde{x}_0, \tilde{x}_1, \tilde{x}_2\} = \{0, 1, 2\} \Rightarrow$ knot vector: $\{0, 0, 0, 1, 2, 2, 2\}$

$i = 3$ since $x_3 = 1 < 1.5 < 2 = x_4$

$$d_2^{(1)} = \lambda_2^{(1)} d_2 + (1 - \lambda_2^{(1)}) d_1$$
$$d_3^{(1)} = \lambda_3^{(1)} d_3 + (1 - \lambda_3^{(1)}) d_2$$

$$\lambda_2^{(1)} = \frac{1.5 - x_2}{x4 - x2} = 0.75; \; \lambda_3(1) = \frac{1.5 - x_3}{x_5 - x_3} = 0.5$$

Example (cont.)

$$d_2^{(1)} = 0.75 d_2 + 0.25 d_1 = \begin{pmatrix} 1.5 \\ 0.5 \end{pmatrix}$$

$$d_3^{(1)} = 0.5 d_3 + 0.5 d_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$\begin{aligned} d_3^{(2)} &= \lambda_3^{(2)} * d_3^{(1)} + (1 - \lambda_3^{(2)}) d_2^{(1)} \\ &= 0.5 \begin{pmatrix} 1 \\ -1 \end{pmatrix} + 0.5 \begin{pmatrix} 1.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 1.25 \\ -0.25 \end{pmatrix} \end{aligned}$$

$$y(1.5) = \begin{pmatrix} 1.25 \\ -0.25 \end{pmatrix}$$

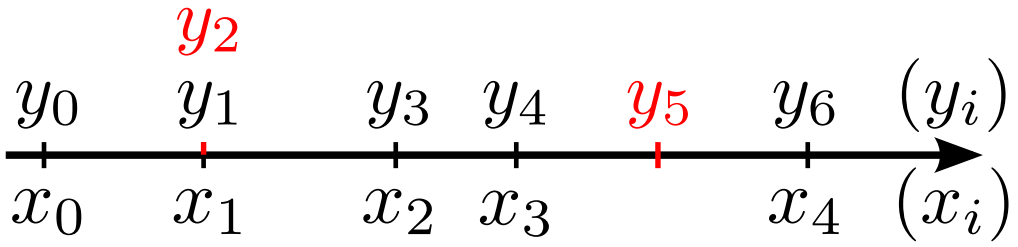
Knot insertion

Every spline with knot vector $\varkappa = (x_i)$ and corresponding B-Splines $N_{i,n}$ (degree $n = M - 1$)

$$f(t) = \sum d_i N_{i,n}(t)$$

can also be represented as a spline over a refined knot vector $\varkappa' = (y_i)$, which contains (x_i), with corresponding B-Splines $N'_{i,n}$:

$$f(t) = \sum c_j N'_{j,n}(t).$$



B-Spline Curves

The c_i can be computed by successively inserting single knots. In this case, compared to the knot vector (x_i), the knot vector (y_i) contains only one additional knot $y = y_{r+1}$ with $x_r \leq y < x_{r+1}$.

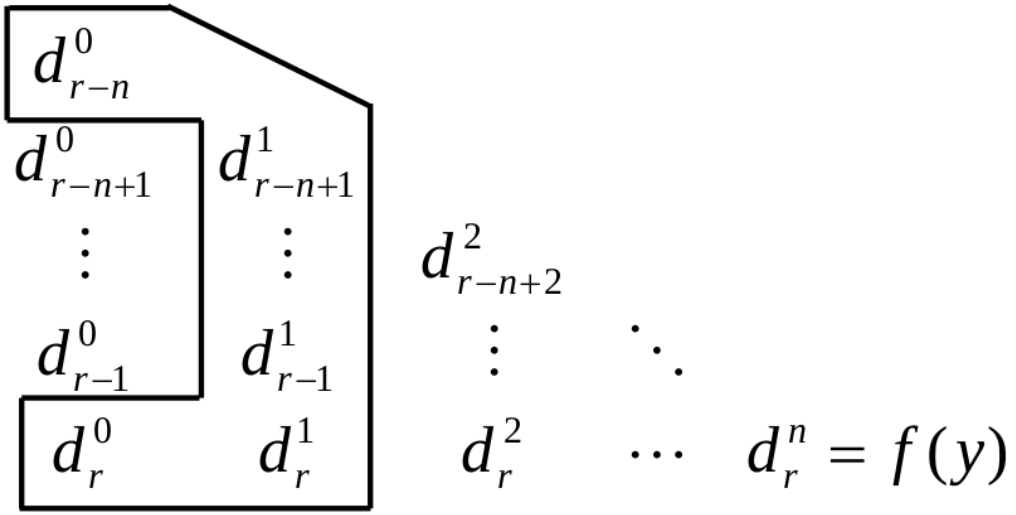
$$c_i = \begin{cases} d_i & 0 \leq i \leq r-n \\ \lambda_i \cdot d_i + (1-\lambda_i) \cdot d_{i-1} & r-n < i \leq r \\ d_{i-1} & r+1 \leq i \end{cases}$$

with

$$\lambda_i = \lambda_i^{(1)} = \frac{y - x_i}{x_{n+i} - x_i} \text{ and degree } n = M - 1$$

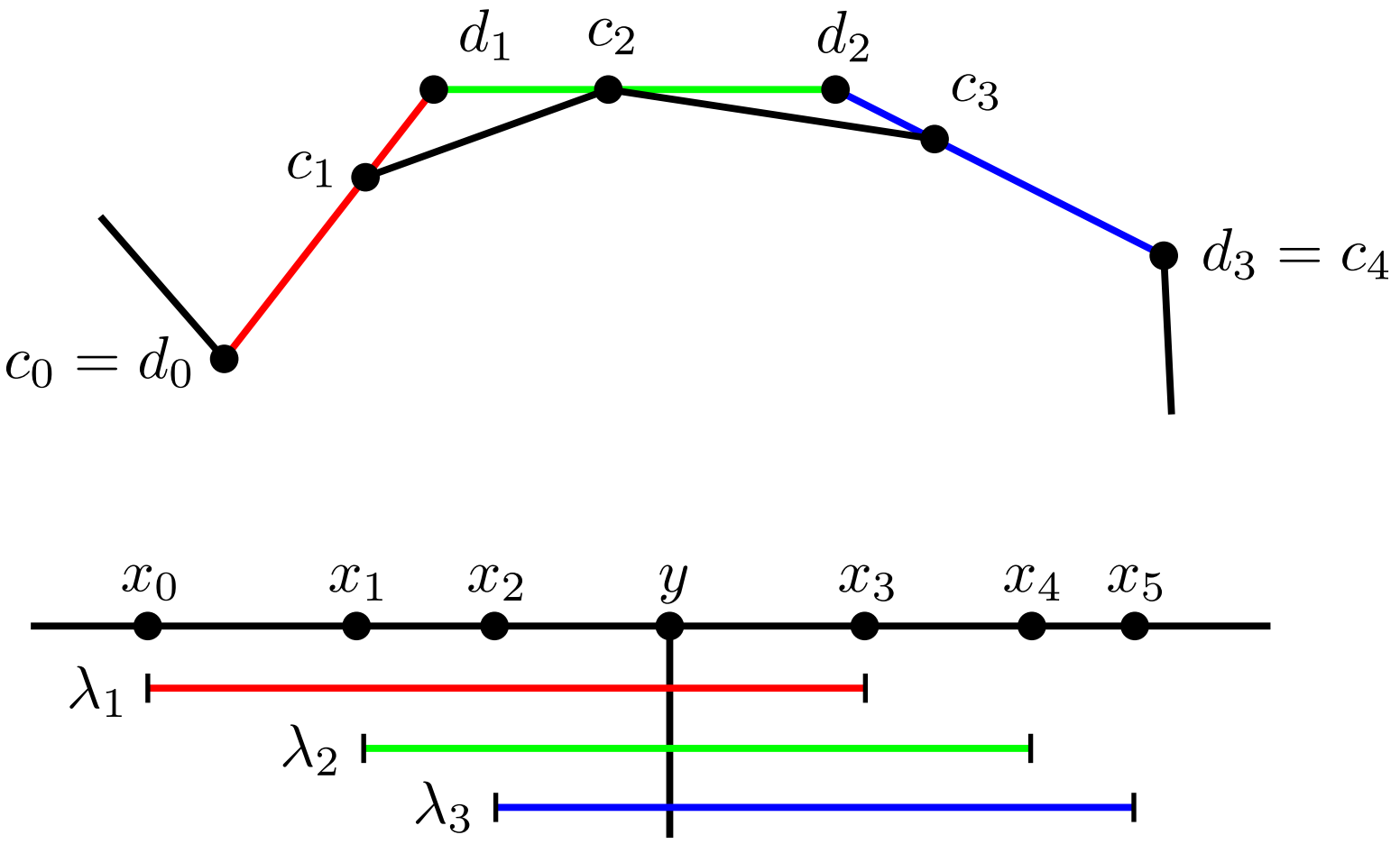
B-Spline Curves

This method is equal to the first step of the De-Boor-algorithm: the sub-polygon $d_{r-n+1}, \ldots, d_{r-1}$ is replaced by $d^{(1)}_{r-n+1}, d^{(1)}_{r-n+2}, \ldots, d^{(1)}_{r}$.



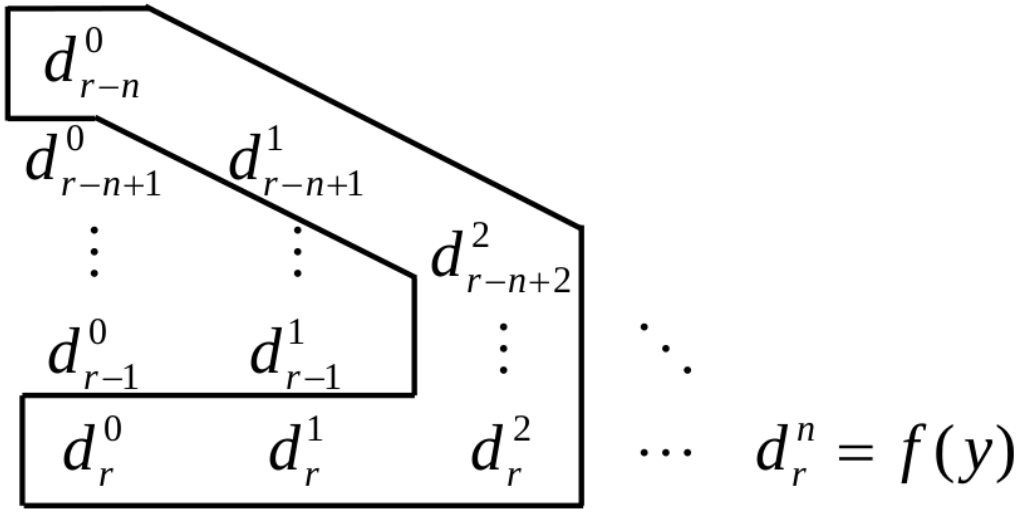
B-Spline Curves

Example for degree $n = 3$:



B-Spline Curves

Using the first $k+1$ steps of the De-Boor-algorithm, we can obtain the control points for inserting the knot y k times:



If all knots of a B-Spline curve have a multiplicity equal to the degree n, we have a Bézier representation of each segment.

Derivatives of B-Spline curves

$$y(s) = \sum_{i=1}^{n} d_i \cdot N_{i,M}(s) \quad \text{B-Spline curve of degree } M-1$$

$$\frac{\mathrm{d}}{\mathrm{d}s} y(s) = \sum_{i=1}^{n} d_i^{(1)} N_{i,M-1}(s)$$

$$\text{with } d_i^{(1)} := (M-1) \cdot \frac{d_i - d_{i-1}}{t_{i+M-1} - t_i}$$

Higher-order derivatives are easily found by recursion.

B-Spline Curves

Integration of B-Spline curves

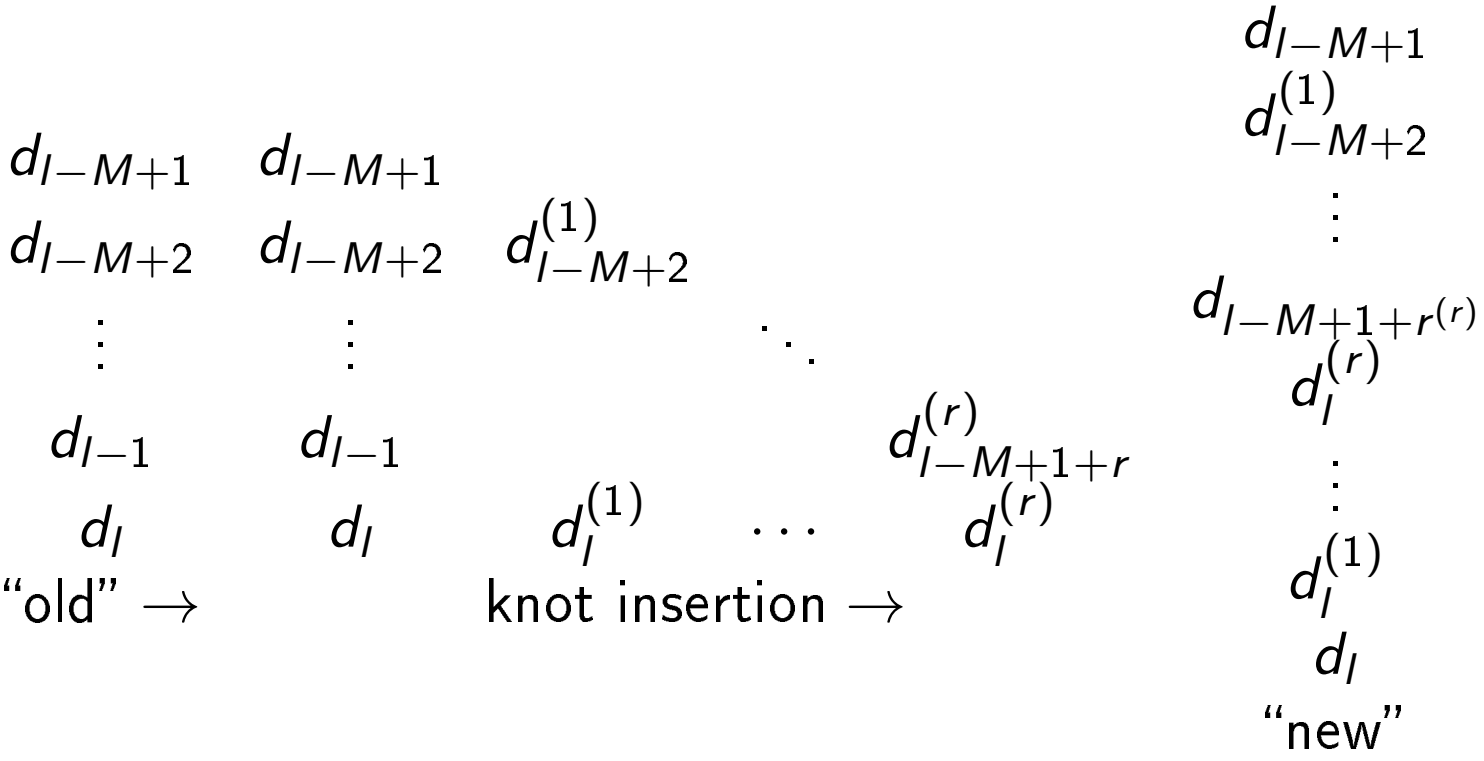
$$y(t) = \sum_{i=0}^{n} d_i \cdot N_{i,M}(t) \quad \text{B-Spline curve of degree } M-1$$

$$\int_{-\infty}^{\infty} y(t)\mathrm{d}t = \frac{1}{M} \sum_{i=0}^{n} d_i \cdot (t_{i+M} - t_i)$$

We can use regular Basis transformations to transform B-Spline curves into Bézier curves (and back). The fact that a Bézier curve is a degenerated B-Spline offers an alternative: *knot insertion* (without changing the curve).

Knot insertion

The $M-2$ "old" control points $d_{l-M+2}, \ldots, d_{l-1}$ are replaced by the M-2+r "new" control points
$d_{l-M+2}, \ldots, d_{l-M-1+r}^{(1)}, \ldots, d_l^{(r)}, \ldots, d_l^{(1)}$.



Knot insertion (cont.)

Key idea: insert knots until the B-Spline curve degenerates to a Bézier curve.

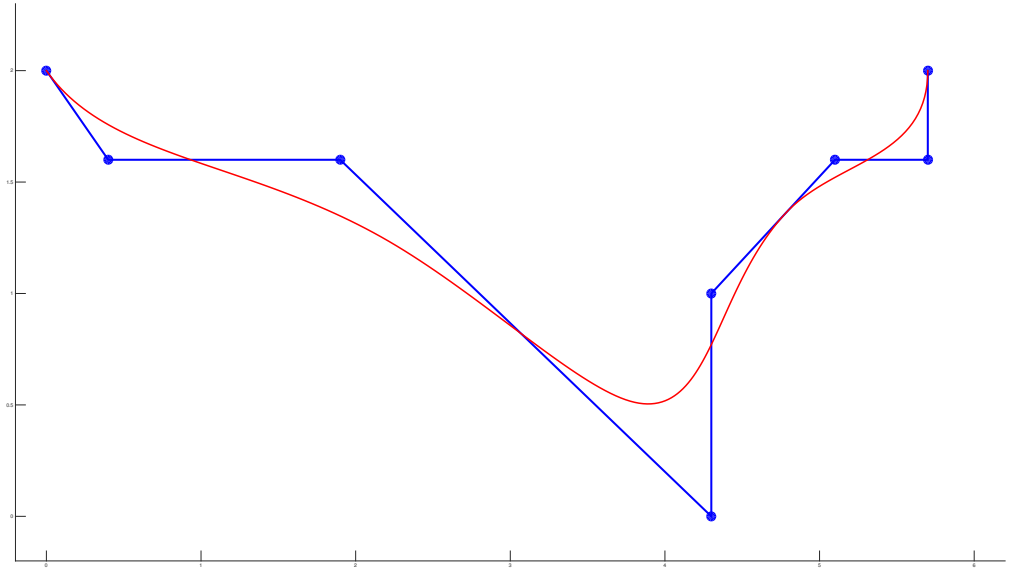


Figure: Example: keel line of a sailing ship

B-Spline Curves

Control structure:

$$d_0 = (0,2) \quad d_1 = (0.4,1.6) \quad d_2 = (1.9,1.6) \quad d_3 = (4.3,0)$$
$$d_4 = (4.3,1) \quad d_5 = (5.1,1.6) \quad d_6 = (5.7,1.6) \quad d_7 = (5.7,2)$$
$$M = 4$$

The B-Spline curve is of degree 3 with the knot vector

$$\{0,0,0,0,0.3,0.5,0.8,0.8,1,1,1,1\}$$

Define $b_0 = d_0$ and $b_1 = d_1$, then insert $t_4 = 0.3$ twice:

$$d_2^{(1)} := \lambda_2^{(1)} \cdot d_2 + \left(1 - \lambda_2^{(1)}\right) \cdot d_1 = 0.6d_2 + 0.4d_1 = (1.3,1.6)$$
$$d_3^{(1)} := \lambda_3^{(1)} \cdot d_3 + \left(1 - \lambda_3^{(1)}\right) \cdot d_2 = 0.375d_3 + 0.625d_2 = (2.8,1)$$
$$d_3^{(2)} := \lambda_3^{(2)} \cdot d_3^{(1)} + \left(1 - \lambda_3^{(2)}\right) \cdot d_2^{(1)} = 0.6d_3^{(1)} + 0.4d_2^{(1)} = (2.2,1.24)$$

B-Spline Curves

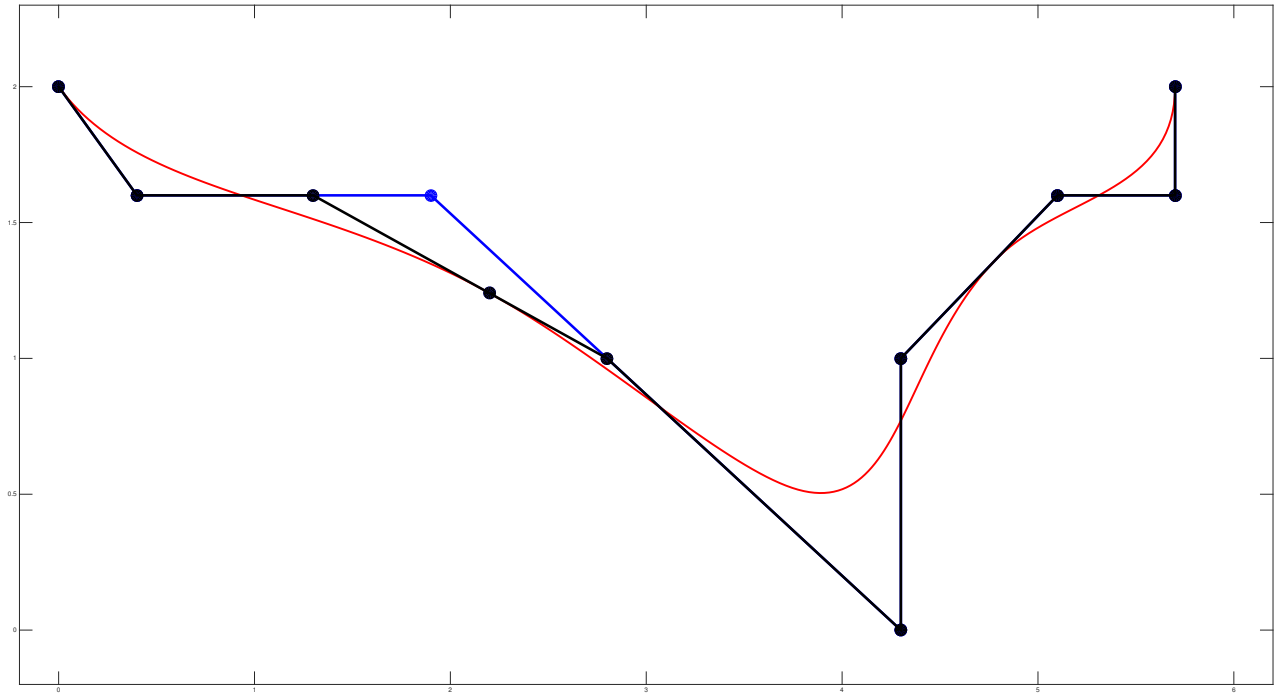


Figure: After inserting $t_4 = 0.3$ twice.

B-Spline Curves

$$
\begin{array}{lllll}
b_0 := d_0 & & & & b_0 = (0,2) \\
b_1 := d_1 & & & & b_1 = (0.4,1.6) \\
\quad d_2 & d_2^{(1)} & & & b_2 := d_2^{(1)} = (1.3,1.6) \\
\quad d_3 & d_3^{(1)} & d_3^{(2)} & \rightarrow & b_3 := d_3^{(2)} = (2.2,1.24) \\
\quad d_4 & & & & b_4 := d_3^{(1)} = (2.8,1) \\
\quad d_5 & & & & d_3 \\
\quad \vdots & & & & \vdots
\end{array}
$$

Now insert $t_5 = 0.5$ two times:

$$
\begin{aligned}
d_3^{(1)} &:= \lambda_3^{(1)} \cdot d_3 + \left(1 - \lambda_3^{(1)}\right) \cdot d_2 = 0.625 d_3 + 0.375 d_2 = (3.4, 0.6) \\
d_4^{(1)} &:= \lambda_4^{(1)} \cdot d_4 + \left(1 - \lambda_4^{(2)}\right) \cdot d_3 = 0.4 d_4 + 0.6 d_3 = (4.3, 0.4) \\
d_4^{(2)} &:= \lambda_4^{(2)} \cdot d_4^{(1)} + \left(1 - \lambda_4^{(2)}\right) \cdot d_3^{(1)} = 0.4 d_4^{(1)} + 0.6 d_3^{(1)} = (3.76, 0.52)
\end{aligned}
$$

B-Spline Curves

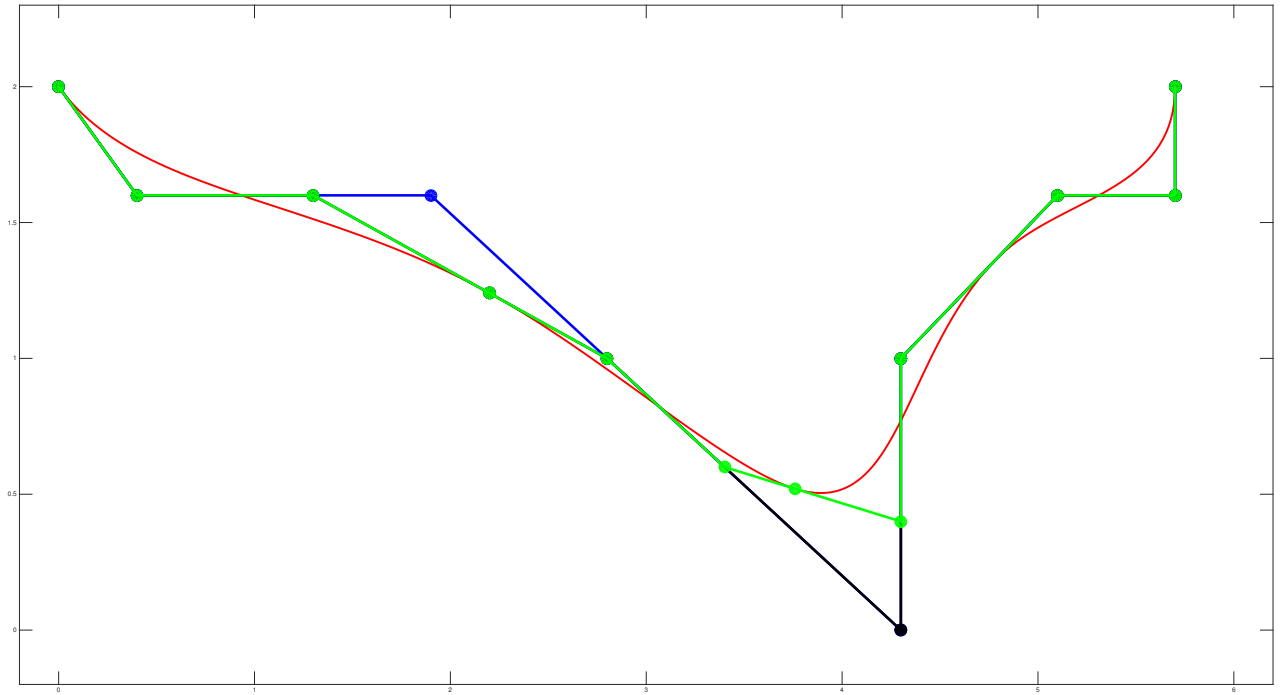


Figure: After inserting $t_5 = 0.5$ twice.

B-Spline Curves

Rename the points:

$$\begin{aligned} b_5 &:= d_3^{(1)} = (3.4, 0.6) \\ b_6 &:= d_4^{(2)} = (3.76, 0.52) \\ b_7 &:= d_4^{(1)} = (4.3, 0.4) \end{aligned}$$

$t_6 = t_7 = 0.8$ only has to be inserted once:

$$d_5^{(1)} := \lambda_5^{(1)} \cdot d_5 \left(1 - \lambda_5^{(1)}\right) \cdot d_4 = 0.6 d_5 + 0.4 d_4 = (4.62, 1.36)$$

B-Spline Curves

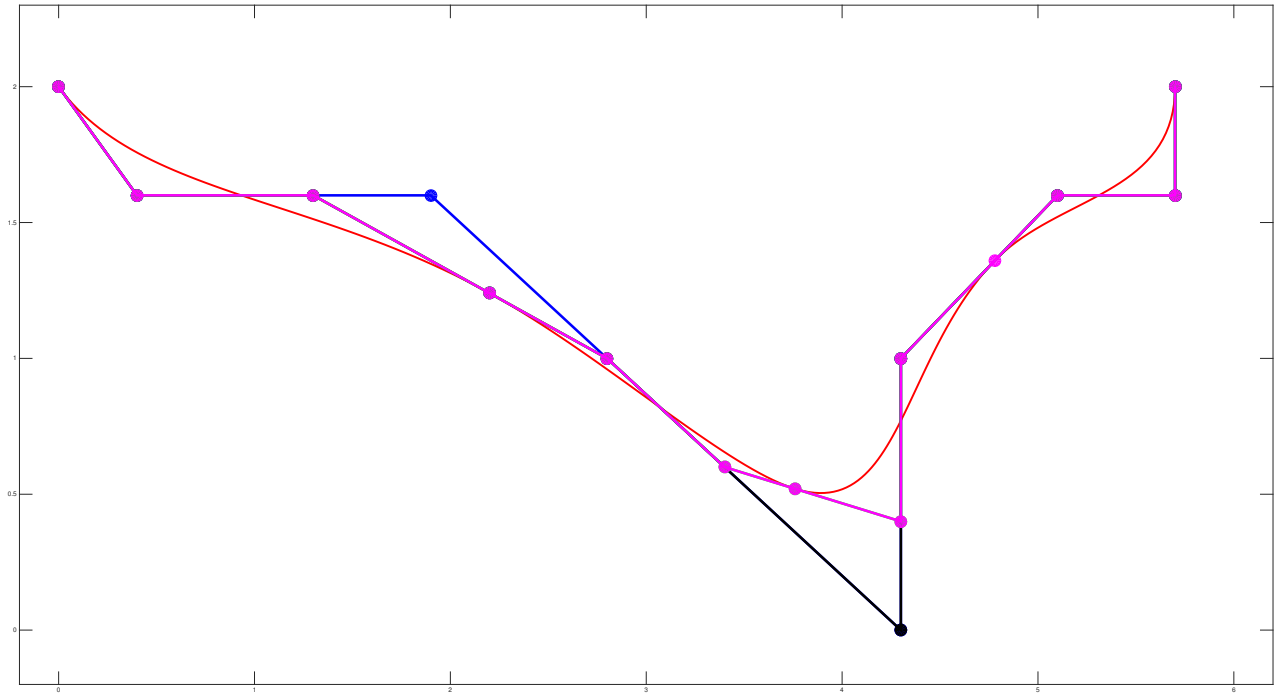


Figure: After inserting $t_6 = t_7 = 0.8$ once.

B-Spline Curves

After renaming the last batch of points,

$$\begin{aligned} b_8 &:= d_4 = (4.3, 1) \\ b_9 &:= d_5^{(1)} = (4.62, 1.36) \\ b_{10} &:= d_5 = (5.1, 1.6), \end{aligned}$$

and finally setting $b_{11} = d_6$, as well as $b_{12} = d_7$,
the Bézier spline then consists of the four segments:

$$\{b_0, b_1, b_2, b_3/b_3, b_4, b_5, b_6/b_6, b_7, b_8, b_9/b_9, b_{10}, b_{11}, b_{12}\}$$

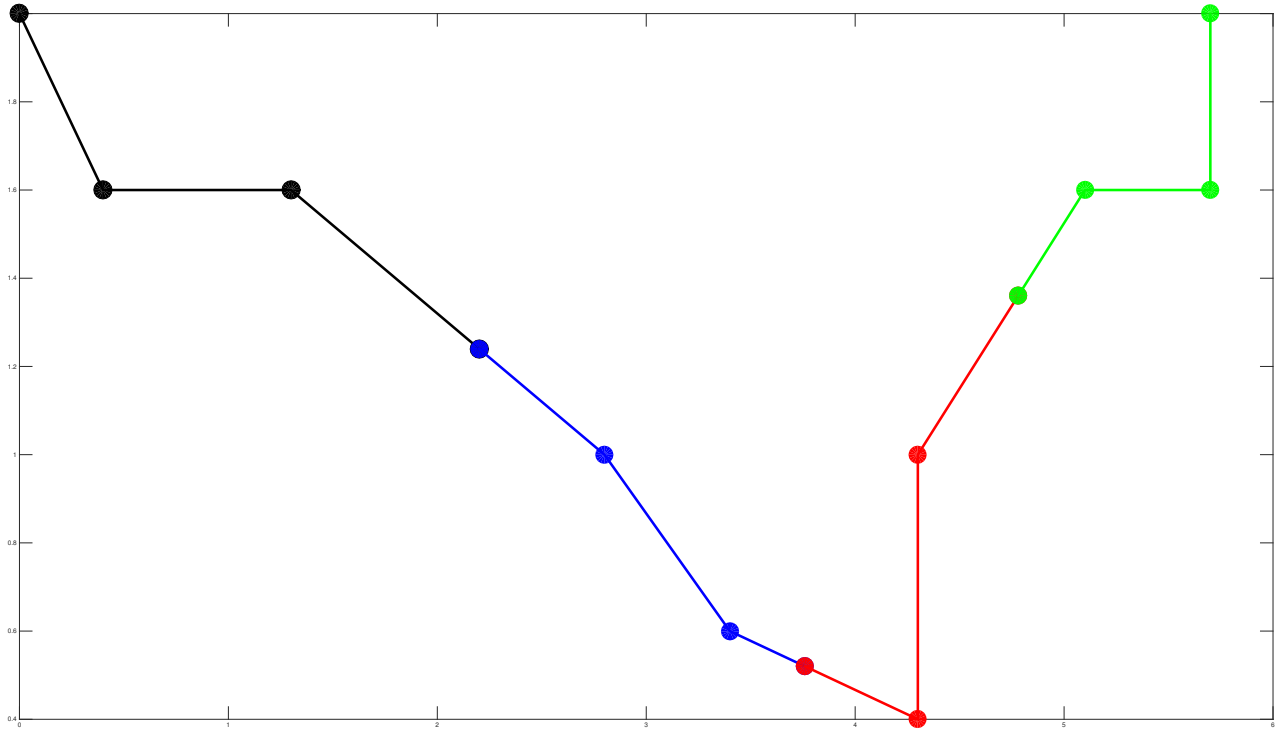


Figure: All Bézier polygons.

B-Spline Curves

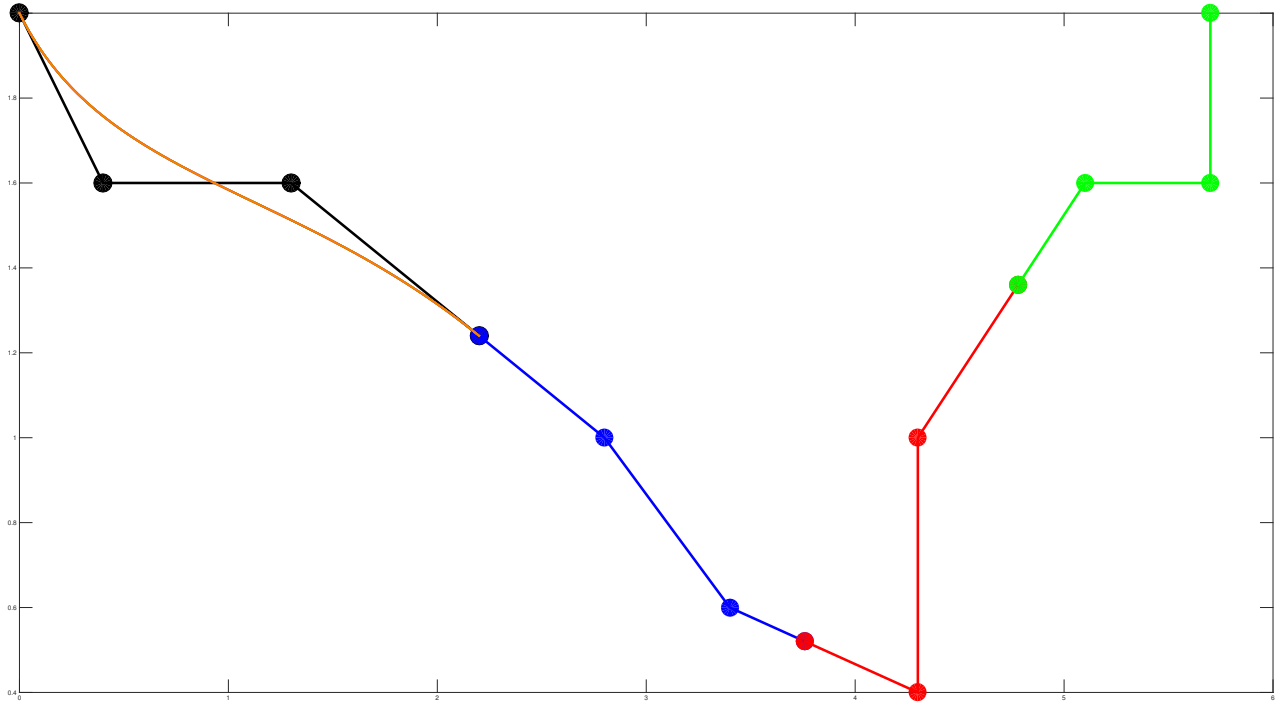


Figure: First Bézier segment.

B-Spline Curves

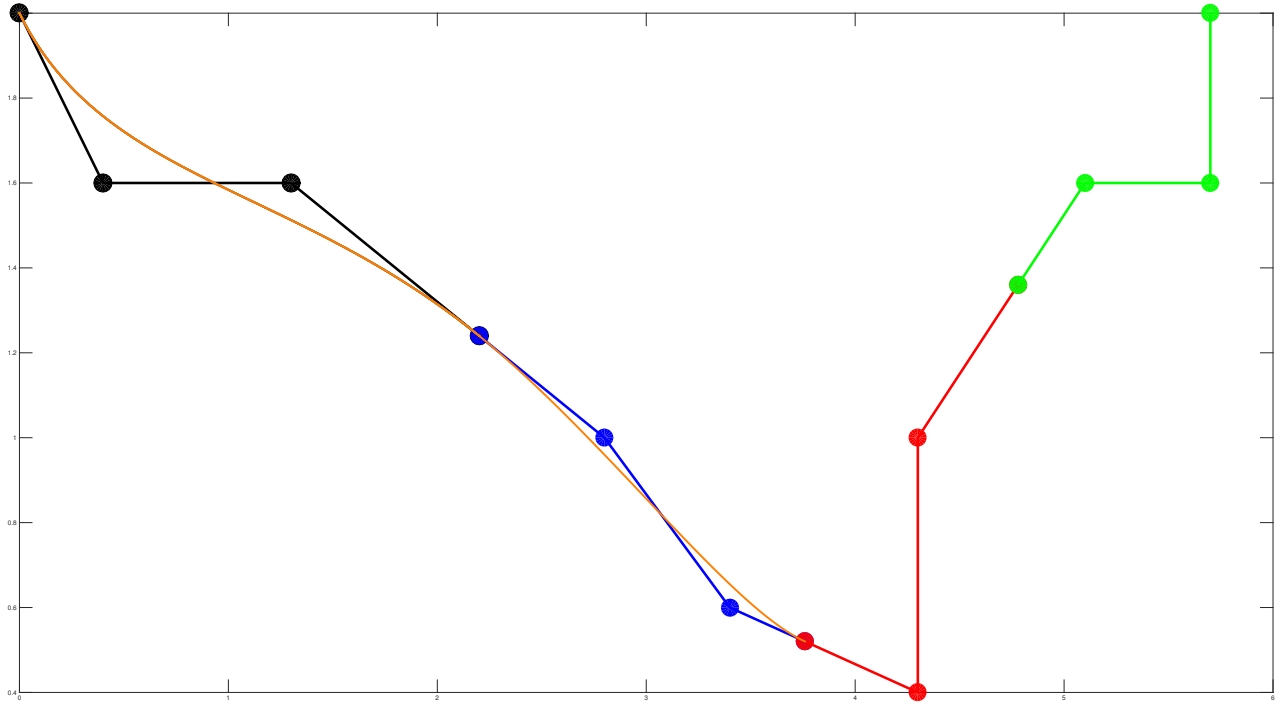


Figure: Second Bézier segment.

B-Spline Curves

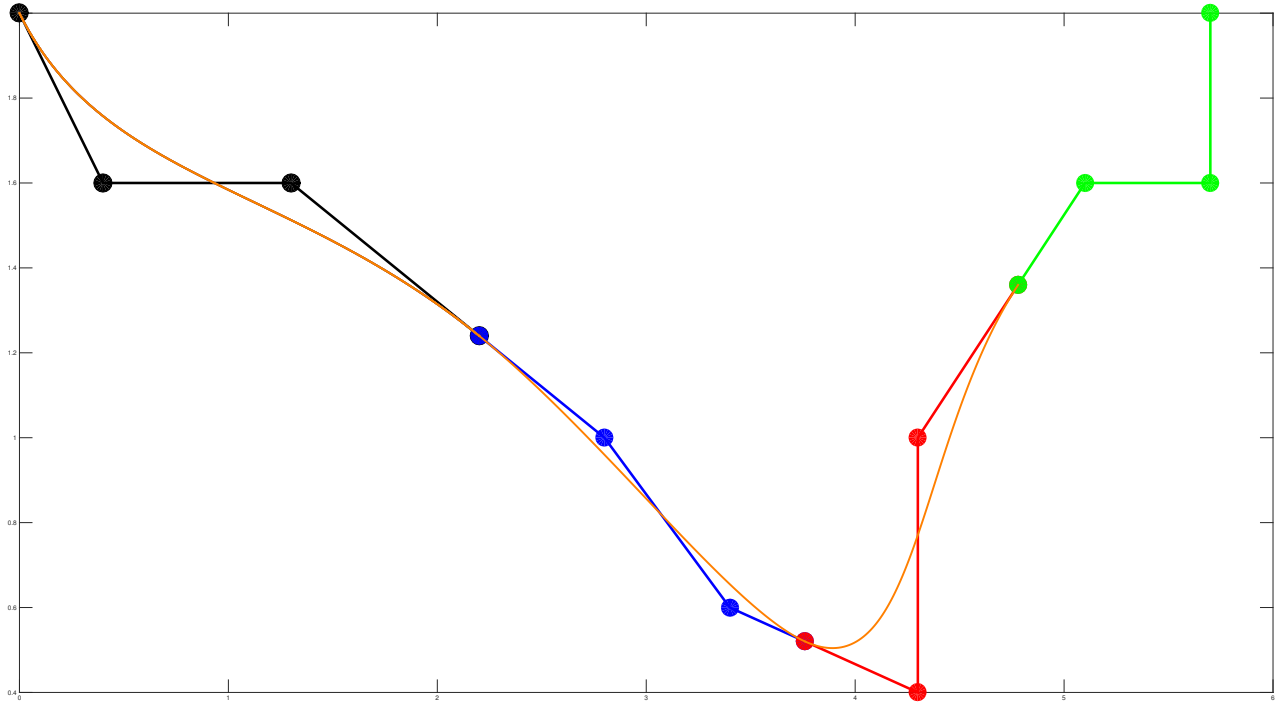


Figure: Third Bézier segment.

B-Spline Curves

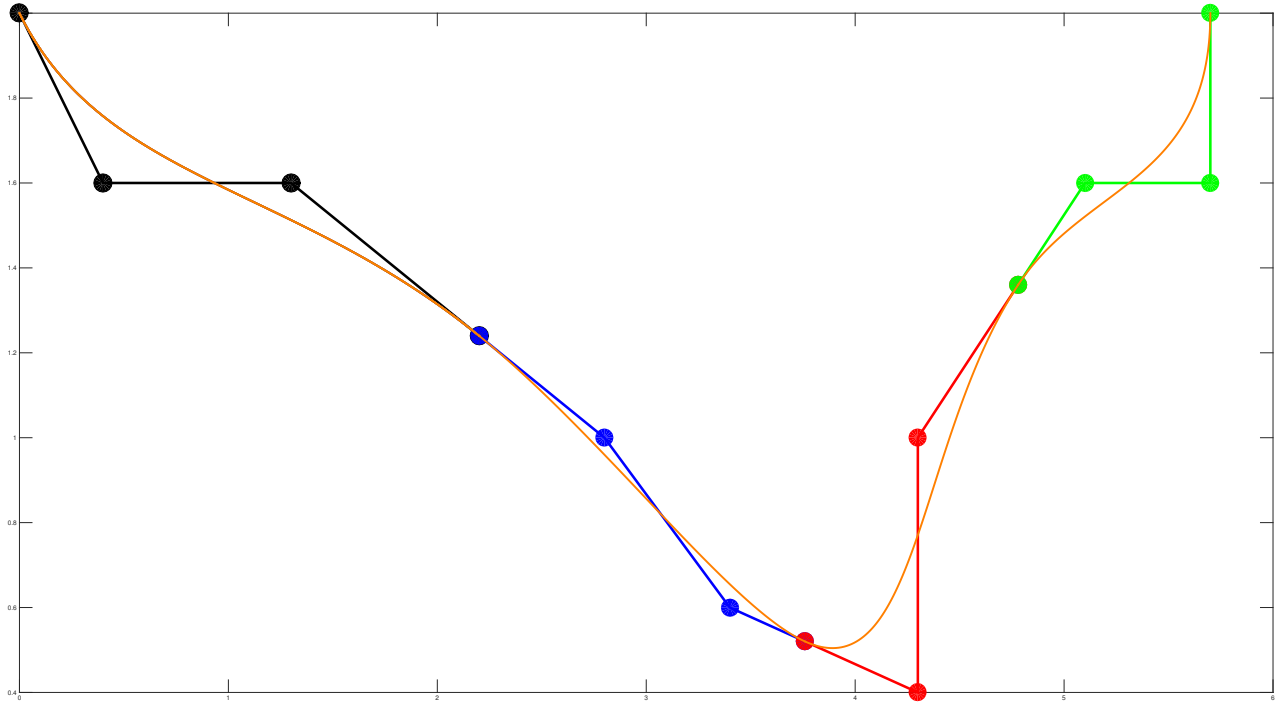


Figure: Fourth Bézier segment.

B-Spline Curves

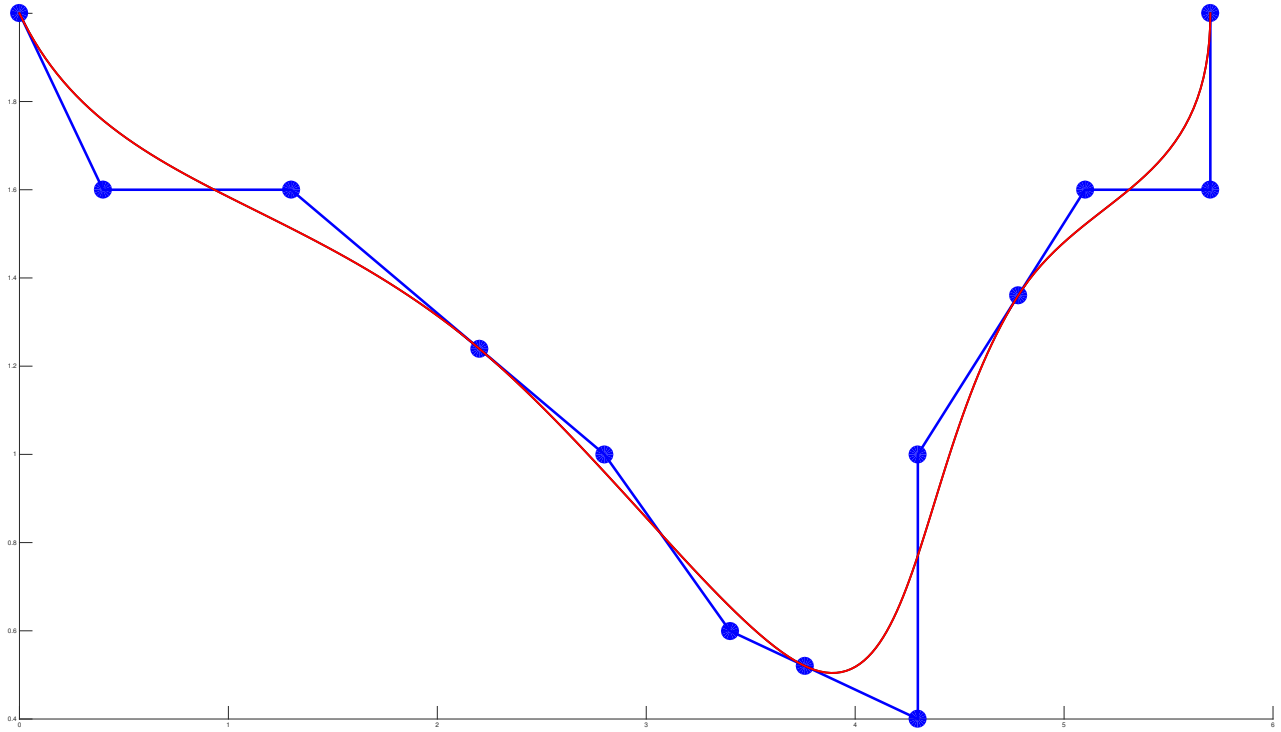


Figure: Drawing the B-Spline curve with refined control points and the knot vector $\{0, 0, 0, 0, 0.3, 0.3, 0.3, 0.5, 0.5, 0.5, 0.8, 0.8, 0.8, 1, 1, 1, 1\}$.

B-Spline Curves

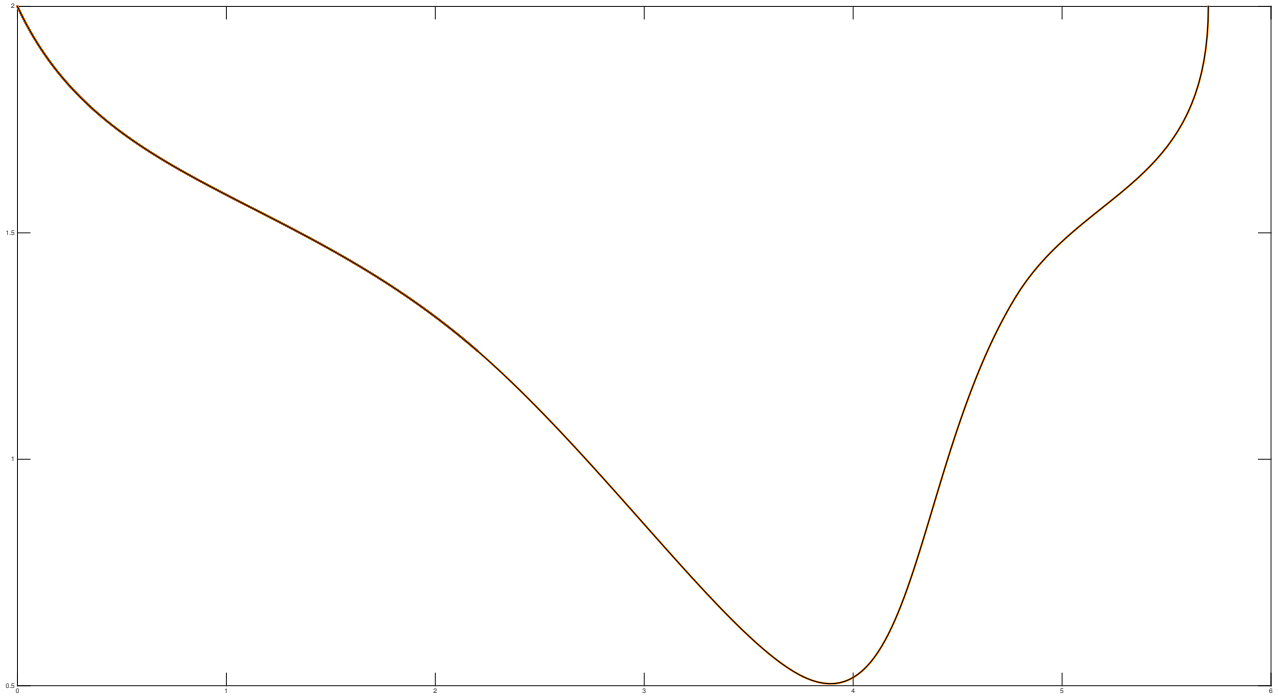


Figure: The B-Spline curve (orange) and the Bézier curve (black) are in fact identical.

B-Spline Curves

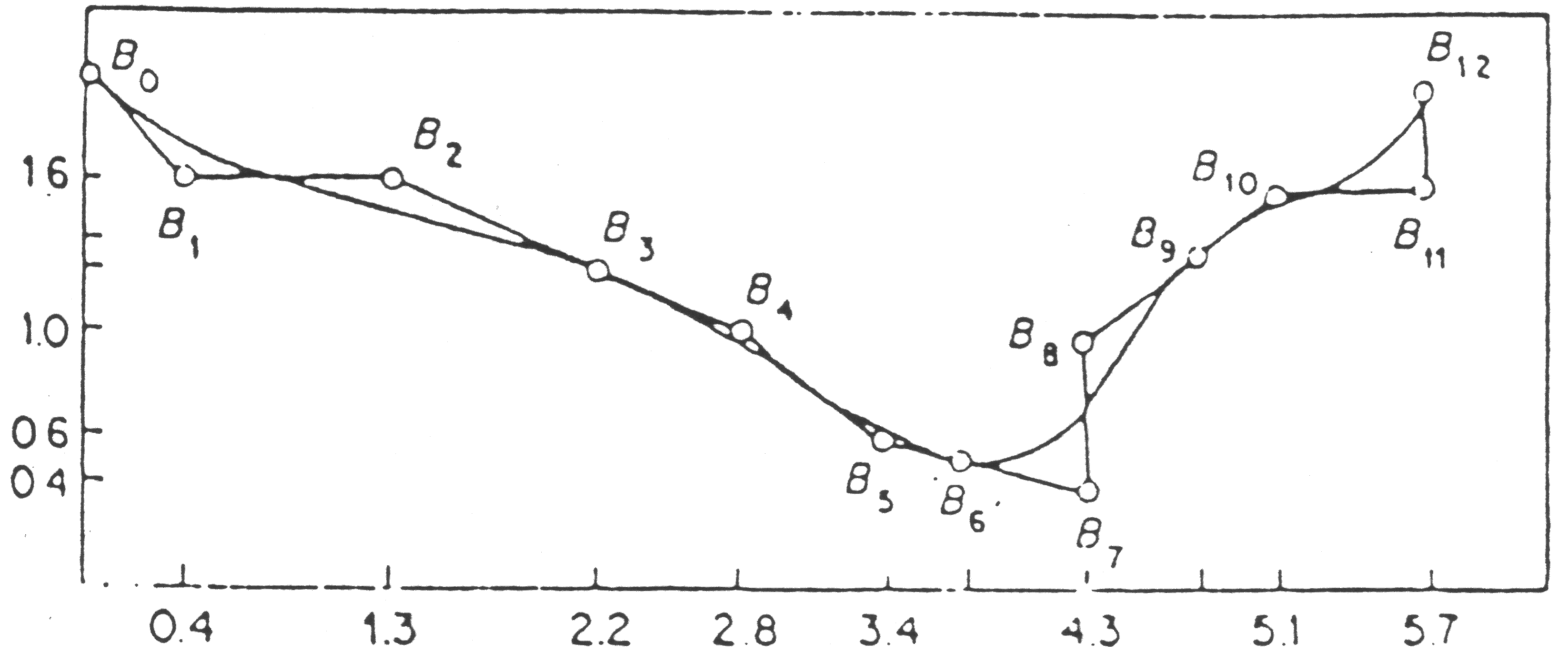


Figure: Result of converting the B-Spline to a Bézier curve

Remarks

It is fine to insert a knot $M-1$ times (with M the order of the curve), but not necessarily M times. The reason is:

$$d_r^{(k)} = d_{r-1}^{(k-1)} \quad \text{for } s = x_r \text{ in the algorithm.}$$

In our example, in the first step $s = 0.3 = t_4$

$$\begin{array}{llll} d_1 & & & \\ d_2 & d_2^{(1)} & & \\ d_3 & d_3^{(1)} & d_3^{(2)} & \\ d_4 & d_4^{(1)} & d_4^{(2)} & d_4^{(3)} \end{array} \quad \rightarrow \quad \begin{array}{l} b_1 := d_1,\ b_2 := d_2^{(1)},\ b_3 := d_3^{(2)} \\ d_4^{(3)} = d_3^{(2)} = b_3 \\ d_4^{(2)} = d_3^{(1)} = b_4 \\ d_4^{(1)} = d_3 \end{array}$$

B-Spline Curves

Remarks (cont.)

- It is certainly more efficient to insert knots only $M - 1$ times and to list endpoints twice.
- The weight points of cubic Bézier curves are the De Boor points of the associated B-Spline curve!

B-Spline Curves

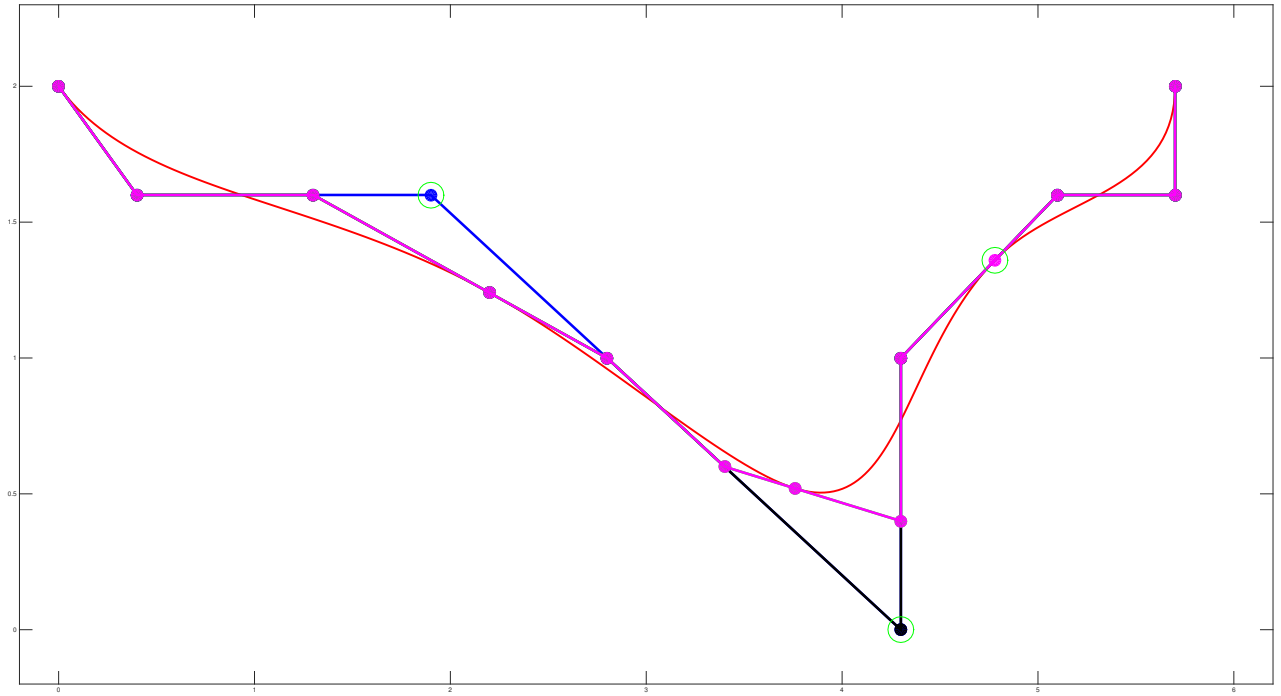


Figure: Weight points for C^2 transition.

The weight points of cubic Bézier curves are the De Boor points of the associated B-Spline curve!

Data exchange format

$$
\begin{aligned}
X(t) &= \begin{pmatrix} b_0 & b_1 & b_2 & b_3 \end{pmatrix} \cdot \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} t^3 \\ t^2 \\ t \\ 1 \end{pmatrix} \\
&= b_0 \cdot \left(-t^3 + 3t^2 - 3t + 1\right) \\
&\quad + b_1 \cdot \left(3t^3 - 6t^2 + 3t\right) \\
&\quad + b_2 \cdot \left(-3t^3 + 3t^2\right) \\
&\quad + b_3 \cdot t^3 \\
&= \begin{pmatrix} b_0 & b_1 & b_2 & b_3 \end{pmatrix} \cdot B_3 \cdot \begin{pmatrix} t^3 & t^2 & t & 1 \end{pmatrix}
\end{aligned}
$$

B_3 maps the basis $\begin{pmatrix} t^3 & t^2 & t & 1 \end{pmatrix}$ into the Bernstein basis.

B-Spline Curves

General case:

Data exchange format for Bézier curves

$$X(t) = \begin{pmatrix} b_0 & b_1 & \cdots & b_n \end{pmatrix} \cdot B_n \cdot \begin{pmatrix} t^n & \cdots & t & 1 \end{pmatrix}^T \quad (t \in [0,1])$$

with

$$B_n = (k_{ij})_{i,j=0}^{n} \quad \text{and} \quad k_{ij} = \begin{cases} \binom{n}{j}\binom{n-j}{n-i-j}(-1)^{n-i-j} & 0 \leq i+j \leq n \\ 0 & \text{otherwise} \end{cases}$$

The general case for B-Spline curves is not very “handy”, but there is a better way:

Data exchange format for B-Spline curves

1. Transform the B-Spline curve into a Bézier spline
2. Use the previously shown data exchange format for that spline

B-Spline Curves

Rational B-Splines

$$S(u) = \frac{\sum w_i d_i N_i^n(u)}{\sum w_i N_i^n(u)}$$

knot sequence $\{u_i\}$

control structure $\{d_0, \ldots, d_n\}$

weight sequence $\{w_0, \ldots, w_n\}$

rational De Boor algorithm $u \in [u_\rho, u_{\rho+1}]$

Rational B-Splines (cont.)

The scheme reads as follows:

$$
\begin{aligned}
d_i^k(u) &= \left(1-\alpha_i^k\right)\frac{w_{i-1}^{k-1}}{w_i^k}d_{i-1}^{k-1}(u) \\
&\quad +\alpha_i^k\frac{w_i^{k-1}}{w_i^k}d_i^{k-1}(u) \\
&\quad k=0,\ldots,n-r \quad \text{and} \quad i=\rho-n+k+1,\ldots,\rho+1 \\
\alpha_i^k &= \frac{u-u_{i-1}}{u_{i+n-k}-u_{i-1}} \\
w_i^k &= \left(1-\alpha_i^k\right)w_{i-1}^{k-1}+\alpha_i^k w_i^{k-1}
\end{aligned}
$$

There is no standard form for rational B-Splines, unlike rational Bézier curves, which can be reparametrized.

B-Spline Curves

Interpolation with rational curves

Input:

3D-points $x_0, \ldots, x_L$

associated parameter values $x_0, \ldots, u_L$

associated weights $w_0, \ldots, w_L$

Solution: solve a linear interpolation problem in $\mathbb{P}^3$!

$$\text{Data points:} \quad w_i \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$