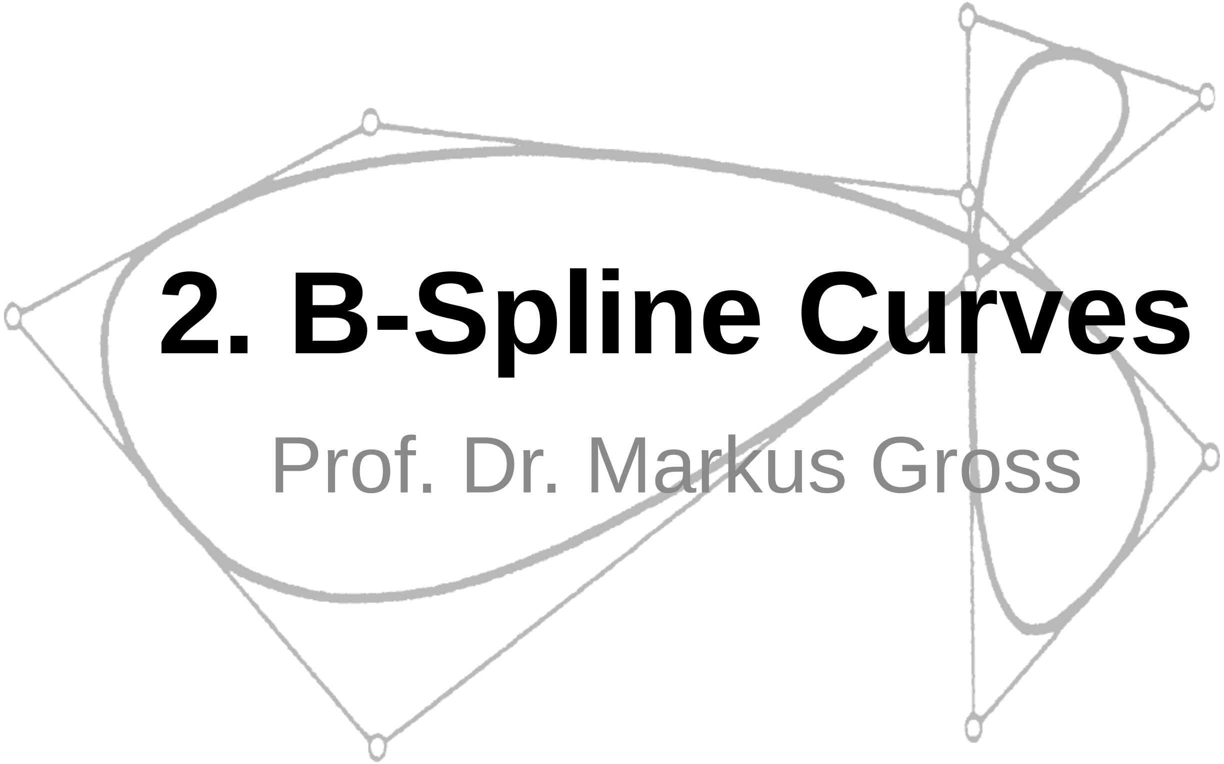
2. B-Spline Curves

Prof. Dr. Markus Gross



ETH zürich

Overview

- B-Spline Basis Functions
- B-Spline Curves
- deBoor Algorithm
- End Conditions
- Interpolation

B-Spline Curves

- Disadvantages of Bézier curves:
  - Global support of the basis functions
  - Insertion of new control points comes along with degree elevation
  - C^r-continuity between individual segments of a Bézier curve

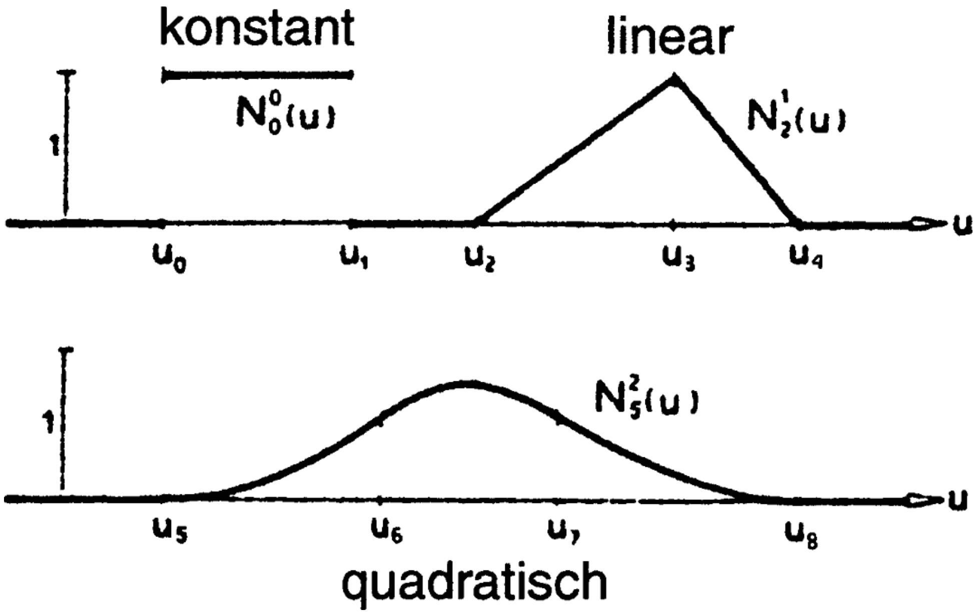
⇒ ***B-Spline bases help to overcome these problems (Local support, continuity control, arbitrary knot vector)***

JAVA-Applet

- Disadvantages of Bézier curves:
  - Global support of the basis functions
  - Insertion of new control points comes along with degree elevation
  - C^r-continuity between individual segments of a Bézier curve

$\Rightarrow$ ***B-Spline bases help to overcome these problems (Local support, continuity control, arbitrary knot vector)***

B-Spline Bases of Different Degree



B-Spline Functions

- Definition:
  - A B-Spline curve $\boldsymbol{s}(u)$ built from piecewise polynomial bases

$$\boldsymbol{s}(u) = \sum_{i=0}^{k} \boldsymbol{d}_i N_i^n(u)$$

  - Coefficients $\boldsymbol{d}_i$ of the B-Spline basis function are called ***de Boor*** points
  - Bases are piecewise, recursively defined polynomials over a sequence of knots $u_0 < u_1 < u_2 < \ldots$
  - Defined by a knot vector $\boldsymbol{T} = \boldsymbol{u} = [u_0, \ldots, u_{k+n+1}]$

JAVA-Applet

- B-Spline bases:
  - Different degrees
  - Piecewise polynomial
  - Local support
  - uniform / non-uniform
  - B-Splines-Bernstein polynomials

B-Spline Functions

- Properties:
  - Partition of Unity: $\sum_i N_i^n(u) \equiv 1$
  - Positivity: $N_i^n(u) \geq 0$
  - Compact support: $N_i^n(u) = 0, \quad \forall u \notin [u_i, u_{i+n+1}]$
  - Continuity: N_i^n is (n-1) times continuously differentiable

B-Spline Functions

- From the recurrence formula we obtain:

$$N_i^1(u) = \begin{cases} \dfrac{u - u_i}{u_{i+1} - u_i}, & u \in [u_i, u_{i+1}] \\ \dfrac{u_{i+2} - u}{u_{i+2} - u_{i+1}}, & u \in [u_{i+1}, u_{i+2}] \end{cases}$$

$$N_i^2(u) = \frac{u - u_i}{u_{i+2} - u_i} N_i^1(u) + \frac{u_{i+3} - u}{u_{i+3} - u_{i+1}} N_{i+1}^1(u)$$

$$= \begin{cases} \dfrac{u - u_i}{u_{i+2} - u_i} \cdot \dfrac{u - u_i}{u_{i+1} - u_i} & i \in [u_i, u_{i+1}] \\ \dfrac{u - u_i}{u_{i+2} - u_i} \cdot \dfrac{u_{i+2} - u}{u_{i+2} - u_{i+1}} + \dfrac{u_{i+3} - u}{u_{i+3} - u_{i+1}} \cdot \dfrac{u - u_{i+1}}{u_{i+2} - u_{i+1}}, & i \in [u_{i+1}, u_{i+2}] \\ \dfrac{u_{i+3} - u}{u_{i+3} - u_{i+1}} \cdot \dfrac{u_{i+3} - u}{u_{i+3} - u_{i+2}} & i \in [u_{i+2}, u_{i+3}] \end{cases}$$

B-Spline Functions

- Recurrence relation:

$$N_i^n(u) = (u - u_i)\frac{N_i^{n-1}(u)}{u_{i+n} - u_i} + (u_{i+n+1} - u)\frac{N_{i+1}^{n-1}(u)}{u_{i+n+1} - u_{i+1}}$$

where:

$$N_i^0(u) = \begin{cases} 1, & u \in [u_i, u_{i+1}] \\ 0, & else \end{cases}$$

The student might verify that B-Spline bases of degree n have support over n+1 Intervals of the knot vector

B-Spline Functions

So-called B-Spline filters are widely use in signal processing. Cardinal B-Splines over uniform knot sequences can be computed using the convolution operator as:

$$N_i^n = N^{n-1} * N^0 = \int_0^x N^{n-1}(t)N^0(x-t)dt$$

$$N^0 : box - function$$

B-Spline Functions

- ***uniform B-Splines*** vs. ***non-uniform B-Splines***

Continuity: Curve is globally C^{n-1} continuous.

- Exception:
  multiple knots of order p with $u_j = \ldots = u_{j+p-1}$ lead to $\boldsymbol{C^{n-p}}$ continuous curves ($p < n+1$)
- Properties:

  $\Rightarrow$ ***variation diminishing property***: More restrictive, for $n+1$ adjacent deBoor points

  $\Rightarrow$ ***convex hull property***: More restrictive, for $n+1$ adjacent deBoor points

deBoor Algorithm

- Generalization of deCasteljau's method.
- Evaluation of a point on the curve at $u = t$.
- For a given t $\in [u_l, u_{l+1}]$ all $N_i^n(u)$ are vanishing in spite of $i \in \{l-n, \ldots, l\}$.

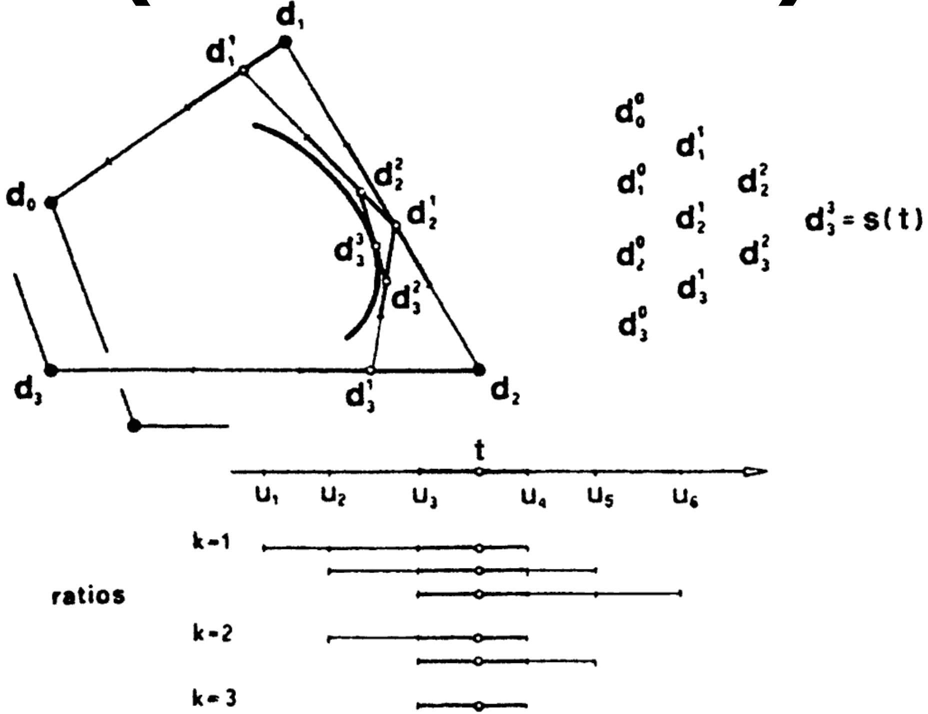
This is a direct consequence of the local support of the bases.

- Point $\mathbf{s}(t)$ computed by successive linear interpolation
- Control point in k-th step

$$\mathbf{d}_i^k = (1 - a_i^k)\mathbf{d}_{i-1}^{k-1} + a_i^k \mathbf{d}_i^{k-1} \qquad a_i^k = \frac{t - u_i}{u_{i+n+1-k} - u_i}$$

where $\quad \mathbf{d}_i^0 = \mathbf{d}_i \quad , \quad \mathbf{d}_n^n = \mathbf{s}(t)$

deBoor Algorithm (non-uniform)



JAVA-Applet

- deBoor algorithm:
  - Successive linear interpolation
  - Local support (Principles of locality)
  - Bernstein polynomials
  - Different end conditions

deBoor Algorithm

- Special case: First and last knot have multiplicity of $n+1$:

$0 = u_0 = u_1 = \ldots = u_n < u_{n+1} = u_{n+2} = \ldots = u_{2n+1}$

- with $u_{n+k} = 1$ for $k \in [1,\ldots,n+1]$ we obtain:

$$\mathbf{d}_i^k(u) = u\,\mathbf{d}_i^{k-1}(u) + (1-u)\,\mathbf{d}_{i+1}^{k-1}(u)$$

(de Casteljau-Algorithm)

End Conditions

- Open curves:
  - Design of endpoint interpolating B-Spline curves of degree n by knot vectors of type:

    $\boldsymbol{u} = \boldsymbol{T} = (u_0 = u_1 = \ldots = u_{n-1} = u_n \;, u_k = u_{k+1} = \ldots = u_{k+n})$

  - Sequencing of knots influences the sweep of the curve
  - Example: Cubic bases with $\boldsymbol{T}_1 = (0,0,0,0,1,2,3,4,5,5,5,5)$ and $\boldsymbol{T}_2 = (0,0,0,0,1,2.75,3.25,4,5,5,5,5)$:

    In both cases we get different bases at the boundaries $N_0^3(0) = 1 = N_0^3(5)$

End Conditions

- Closed curves:
  - Periodic repetition of the deBoor points and knots by $d_0 = d_{k+1}$

$$
\begin{aligned}
u_{k+1} &= u_0 \\
u_{k+2} &= u_{k+1} + (u_1 - u_0) \\
u_{k+3} &= u_{k+2} + (u_2 - u_1) \\
&\dots
\end{aligned}
$$

  - The knot vector:

$$\boldsymbol{T} = (u_0, u_1, \dots, u_k, u_{k+1} = u_0, u_{k+2} = u_2, \dots, u_{k+n} = u_{n-1})$$

End Conditions

- Parametric B-Spline curve:

$$\boldsymbol{s}(u)=\sum_{i=0}^{k}\boldsymbol{d}_{\boldsymbol{i}}N_i^n(u), \qquad u\in[u_0,u_{n-1}]$$

- Support of the bases:

$$N_0^n \Rightarrow [u_0,\dots,u_{n+1}]$$
$$N_1^n \Rightarrow [u_1,\dots,u_{n+2}]$$
$$N_2^n \Rightarrow [u_2,\dots,u_{n+3}]$$
$$\dots$$
$$N_{k-2}^n \Rightarrow [u_{k-2},u_{k-1},u_k,u_0,\dots,u_{n-2}]$$
$$N_{k-1}^n \Rightarrow [u_{k-1},u_k,u_0,\dots,u_{n-1}]$$
$$N_k^n \Rightarrow [u_k,u_0,\dots,u_n]$$

B-Spline Interpolation

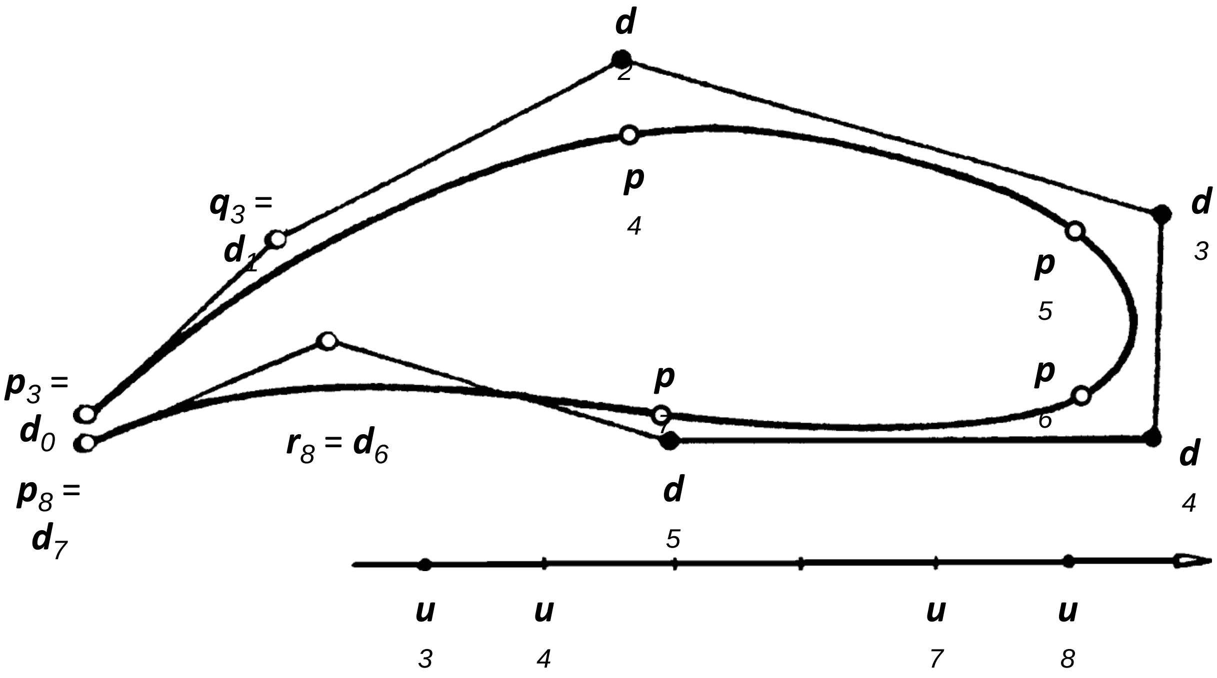
- Interpolate a given set of $k+1$ points $\boldsymbol{p}_j$
- Let $u_j \in [u_0,..,u_{k+n+1}]$ a straightforward insertion yields

$$\cancel{\boldsymbol{s}(u_j) = \sum_{i=0}^{k} \boldsymbol{d}_i N_i^n(u_j) = \boldsymbol{p}_j}$$

- However, the curve needs $n+1$ active bases in the interval of definition
- System is under-determined
- We need more control points $\boldsymbol{d}_0,\ldots,\boldsymbol{d}_{k+n-1}$

$$\boldsymbol{s}(u_j) = \sum_{i=0}^{k+n-1} \boldsymbol{d}_i N_i^n(u_j) = \boldsymbol{p}_j$$

Interpolating B-Spline



Endpoints $p_3 = d_0$ and $p_8 = d_7$ as well as tangents ($q_a = d_1$ and $r_b = d_6$) have to be preset

B-Spline Interpolation

- For endpoint interpolating splines we need $n+k$ equations, whereof $k-1$ define the interior intervals and $n+1$ the boundaries
- Interpolation costs two equations:

$$\boldsymbol{d}_0 = \boldsymbol{p}_0 \quad , \quad \boldsymbol{d}_{k+n-1} = \boldsymbol{p}_k$$

- Others can be used to specify tangency, curvature etc.

$$\boldsymbol{t}_0 = \boldsymbol{d}_1 - \boldsymbol{d}_0 \quad , \quad \boldsymbol{t}_k = \boldsymbol{d}_{k+n-2} - \boldsymbol{d}_{k+n-1}$$

JAVA-Applet

- Illustration of the interpolation problem

B-Spline Interpolation

- For a cubic:

$$
k+n\left\{\begin{bmatrix}
1 & 0 & & & & & & \\
-1 & 1 & & & & & & \\
 & * & * & * & & & & \\
 & & \cdot & \cdot & \cdot & & & \\
\cdot & & & \cdot & \cdot & \cdot & & \\
 & & & & \cdot & \cdot & \cdot & \\
 & & & & & & 1 & -1 \\
 & & & & & & 0 & 1
\end{bmatrix}\right.
\begin{bmatrix}
\mathbf{d}_0 \\ \mathbf{d}_1 \\ \vdots \\ \vdots \\ \mathbf{d}_{n+k-1}
\end{bmatrix}
=
\begin{bmatrix}
\mathbf{p}_0 \\ \mathbf{t}_0 \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_{k-1} \\ \mathbf{t}_k \\ \mathbf{p}_k
\end{bmatrix}
$$