RHEINISCHE
FRIEDRICH-WILHELMS-
UNIVERSITÄT BONN

INSTITUT FÜR
INFORMATIK DER
UNIVERSITÄT BONN

Einführung in die Computergrafik

Kapitel 6: Parametrische Kurven

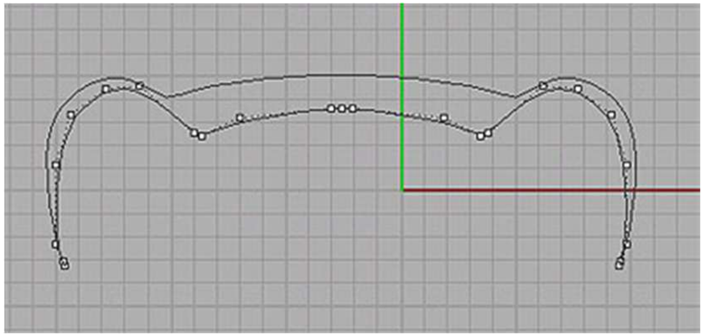
Prof. Dr. Matthias Hullin

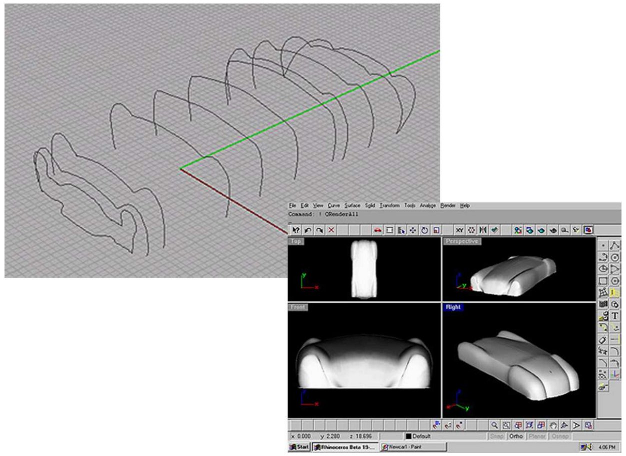
Institut für Informatik
Abteilung 2: Visual Computing Universität
Bonn

21. Mai 2021

Problem: Wie generiert man eine Kurve bzw. Fläche mit vorgegebenen Eigenschaften wie z.B.:

- Approximation / Interpolation gegebener Punkte
- Glattheit
- etc.





- Explizite Darstellungen

$$p_1 : [0, 2\pi] \to \mathbb{R}^3, \quad t \mapsto r \begin{pmatrix} \cos(t) \\ \sin(t) \\ 0 \end{pmatrix}$$

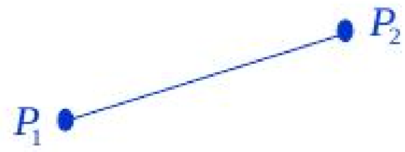
$$p_2 : [0, 2\pi] \times [-\pi/2, \pi/2] \to \mathbb{R}^3, \quad (u, v) \mapsto r \begin{pmatrix} \cos(u)\cos(v) \\ \sin(u)\cos(v) \\ \sin(v) \end{pmatrix}$$

- Implizite Darstellungen

$$f : \mathbb{R}^2 \to \mathbb{R}$$
$$K = \{(x, y) \in \mathbb{R}^2 : f(x, y) = 0\}$$
$$f(x, y) = x^2 + y^2 - r^2$$

$$g : \mathbb{R}^3 \to \mathbb{R}$$
$$K = \{(x, y, z) \in \mathbb{R}^3 : g(x, y, z) = 0\}$$
$$f(x, y) = x^2 + y^2 + z^2 - r^2$$

Wir interessieren uns im Folgenden für explizite Darstellungen. Hierbei kann eine Kurve mit unterschiedlichen Parametrisierungen dargestellt werden:



$$p_1 : \ [0,1] \to \mathbb{R}^d \qquad p(t) = t \cdot P_2 + (1-t) \cdot P_1$$
$$p_2 : \ [0,1] \to \mathbb{R}^d \qquad p(t) = t^2 \cdot P_2 + (1-t^2) \cdot P_1$$

Definition (Parametrisierung einer Kurve)

Eine *Parametrisierung einer Kurve* ist eine Abbildung $p\colon [a,b] \to \mathbb{R}^n$ von einem Intervall $[a,b]$ auf die Kurve.

Die Glattheit einer parametrisierten Kurve ist ein wichtiges Kriterium für deren Nützlichkeit:

Definition (Differenzierbarkeit von parametrisierten Kurven)

Eine Kurve heißt *n-mal stetig differenzierbar*, falls es eine Parametrisierung $p\colon [a, b] \to \mathbb{R}^n$ gibt, die *n-mal stetig differenzierbar* ist.

Bemerkung: Die Differenzierbarkeit bzw. Stetigkeit einer Kurve ist somit unabhängig von der Parametrisierung gegeben.

Die normalisierte Ableitung $p'/\|p'\|$ einer parametrisierten Kurve $p\colon [a, b] \to \mathbb{R}^n$ wird auch *Tangente* genannt.

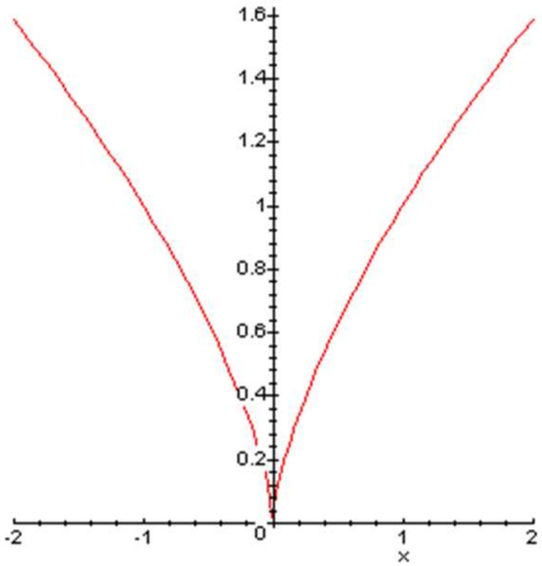
Beispiel:

$$p\colon [-2,2] \to \mathbb{R}^3 \qquad p(t) = (t^3, t^2, 0)^T$$

Nun bildet man die Ableitung von p:

$$p'(t) = (3t^2, 2t, 0)^T$$

Offensichtlich ist $p'(0) = 0$, d.h. p hat an der Stelle 0 eine Spitze.



Um solche Spitzen zu vermeiden, wird der Begriff der Regularität eingeführt:

Definition (Regularität von Kurven)

Eine parametrisierte Kurve $p\colon [a,b] \to \mathbb{R}^n$ heißt *regulär*, falls die Abbildung p einmal stetig differenzierbar ist und für die Ableitung gilt:

$$p'(x) \neq 0 \qquad \forall x \in [a,b]$$

Definition (Bogenlängenfunktion einer Kurve)

Sei $p\colon [a,b] \to \mathbb{R}^n$ eine Kurve, dann heißt die Funktion $\hat{s}\colon [a,b] \to [0,\infty)$ mit

$$\hat{s}(t) := \int_a^t \left\| p'(\tau) \right\|_2 d\tau$$

Bogenlängenfunktion von p.

Beispiel:

$$\begin{aligned} p(t) &= (\cos(t^2), \sin(t^2))^T \\ \left\| p'(t) \right\|_2 &= \left\| (-2t \cdot \sin(t^2), 2t \cdot \cos(t^2))^T \right\|_2 = 2t \\ \hat{s}(t) &= \int_0^t 2\tau \, d\tau = t^2 \end{aligned}$$

Definition (Parametrisierung nach Bogenlänge)

Für jede reguläre, parametrisierte Kurve $p(t)$ ist die Umparametrisierung

$$p_s(s) := p(\hat{s}^{-1}(s))$$

eine Parametrisierung nach Bogenlänge, d.h.

$$\left\| \frac{dp_s}{ds} \right\|_2 = 1$$

mit der Tangente

$$t := \frac{dp_s}{ds}$$

Beispiel:

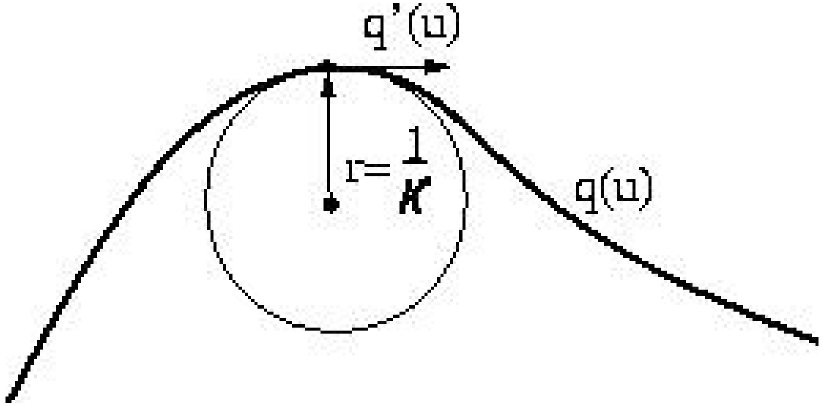
$$p(t) = (\cos(t^2), \sin(t^2))^T \qquad s(t) = \int_0^t 2\tau d\tau = t^2$$

$$p_s(s) = p(\sqrt{s}) = (\cos(s), \sin(s))^T$$

Bemerkung: Jede reguläre Kurve kann auf zwei Arten nach Bogenlänge parametrisiert werden.

Für nach der Bogenlänge parametrisierte Kurven gilt:

$T(s) := p'(s)$	Tangentenvektor
$K(s) := p''(s)$	Krümmungsvektor
$\kappa(s) := \left\|p''(s)\right\|$	Krümmung



Definition (Polynom n-ten Grades)

Die Abbildung $p : [a, b] \to \mathbb{R}^d$ mit

$$p(t) = c_0 + c_1 t + c_2 t^2 + \cdots + c_n t^n \qquad c_i \in \mathbb{R}^d$$

heißt *Polynom* vom Grad n im $\mathbb{R}^d$.

Eigenschaften von Polynomen:

- Die Menge aller Polynome vom Grad n bildet einen Vektorraum der Dimension $n + 1$:

  $$(\alpha p + \beta q)(t) = \alpha p(t) + \beta q(t)$$

  für reelle Zahlen α, β und Polynome p, q vom Grad n
- Die Monome $1, t, t^2 \ldots, t^n$ bilden eine Basis dieses Vektorraums
- effiziente Auswertung mittels Hornerschema:

  $$p(t) = c_n t^n + \cdots + c_1 t + c_0 = (\ldots((c_n t + c_{n-1})t + c_{n-2})t + \ldots c_1)t + c_0$$

  mit n Additionen und Multiplikationen.

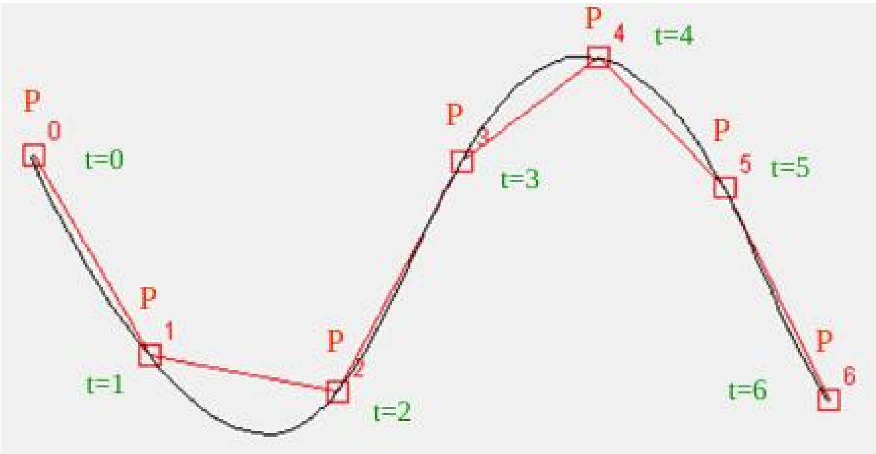
  $$p(t) = c_3 t^3 + c_2 t^2 + c_1 t + c_0 = ((c_3 t + c_2)t + c_1)t + c_0$$

Die Koeffizienten beschreiben die Ableitungen des Polynoms an der Stelle 0:

$$p(t) = c_0 + c_1 t + c_2 t^2 + \cdots + c_n t^n \qquad c_i \in \mathbb{R}^d$$

$$c_0 = p(0) \quad c_1 = p'(0) \quad c_2 = \frac{1}{2} \cdot p''(0) \quad \ldots \quad c_k = \frac{1}{k!} \cdot p^{(k)}(0)$$

Problem: Modellieren von Kurven ist mit Hilfe dieser Koeffizienten praktisch unmöglich.



Gegeben: Stützstellen $P_i \in \mathbb{R}^d$ und Parameterwerte $t_i \in \mathbb{R}$ für $i = 0, \ldots, n$
Gesucht: Polynomkurve mit $p(t_i) = P_i \ \forall i = 0, \ldots, n$

Beispiel:
Die Kurve soll bei p_0 beginnen, bei $u = 0.5$ durch p_1 verlaufen und bei p_2 enden.
Polynom: $p(t) = c_0 + c_1 t + c_2 t^2$ Monome: $(1, t, t^2)$

$$\begin{pmatrix} p_x(t) \\ p_y(t) \\ p_z(t) \end{pmatrix} = \begin{pmatrix} c_{0x} \\ c_{0y} \\ c_{0z} \end{pmatrix} + t \begin{pmatrix} c_{1x} \\ c_{1y} \\ c_{1z} \end{pmatrix} + t^2 \begin{pmatrix} c_{2x} \\ c_{2y} \\ c_{2z} \end{pmatrix}$$

$$f(0) = \begin{pmatrix} p_{0x} \\ p_{0y} \\ p_{0z} \end{pmatrix} = 1 \begin{pmatrix} c_{0x} \\ c_{0y} \\ c_{0z} \end{pmatrix}$$

$$f\left(\frac{1}{2}\right) = \begin{pmatrix} p_{1x} \\ p_{1y} \\ p_{1z} \end{pmatrix} = 1 \begin{pmatrix} c_{0x} \\ c_{0y} \\ c_{0z} \end{pmatrix} + \frac{1}{2} \begin{pmatrix} c_{1x} \\ c_{1y} \\ c_{1z} \end{pmatrix} + \frac{1}{4} \begin{pmatrix} c_{2x} \\ c_{2y} \\ c_{2z} \end{pmatrix}$$

$$f(1) = \begin{pmatrix} p_{2x} \\ p_{2y} \\ p_{2z} \end{pmatrix} = 1 \begin{pmatrix} c_{0x} \\ c_{0y} \\ c_{0z} \end{pmatrix} + 1 \begin{pmatrix} c_{1x} \\ c_{1y} \\ c_{1z} \end{pmatrix} + 1 \begin{pmatrix} c_{2x} \\ c_{2y} \\ c_{2z} \end{pmatrix}$$

Beispiel:
Die Kurve soll bei p_0 beginnen, bei $u = 0.5$ durch p_1 verlaufen und bei p_2 enden.

$$p_0 = f(0) \qquad p_1 = f\left(\frac{1}{2}\right) \qquad p_2 = f(1)$$

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} \\ 1 & 1 & 1 \end{pmatrix}}_{C} \begin{pmatrix} c_0^T \\ c_1^T \\ c_2^T \end{pmatrix} = \begin{pmatrix} p_0^T \\ p_1^T \\ p_2^T \end{pmatrix}$$

$$\begin{pmatrix} c_0^T \\ c_1^T \\ c_2^T \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{pmatrix}}_{C^{-1}} \begin{pmatrix} p_0^T \\ p_1^T \\ p_2^T \end{pmatrix}$$

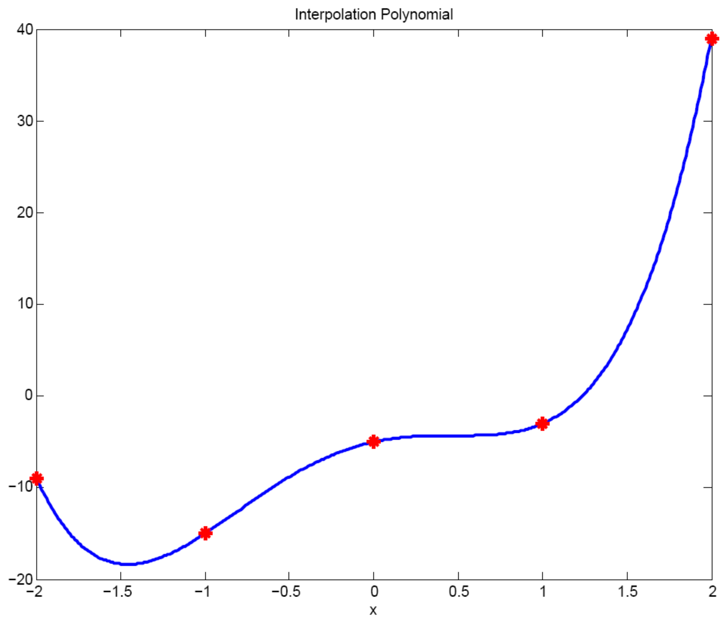
Beispiel: Mittelpunkt und Ableitungen
Die Kurve soll bei $u = 0.5$ durch p_0 verlaufen und und dort die Ableitungen p_1 und p_2 besitzen.

$$p_0 = f\left(\frac{1}{2}\right) \qquad p_1 = f'\left(\frac{1}{2}\right) \qquad p_2 = f''\left(\frac{1}{2}\right)$$

$$\underbrace{\begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{4} \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix}}_{C} \begin{pmatrix} c_0^T \\ c_1^T \\ c_2^T \end{pmatrix} = \begin{pmatrix} p_0^T \\ p_1^T \\ p_2^T \end{pmatrix}$$

$$\begin{pmatrix} c_0^T \\ c_1^T \\ c_2^T \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & -\frac{1}{2} & \frac{1}{8} \\ 0 & 1 & -\frac{1}{2} \\ 0 & 0 & 1 \end{pmatrix}}_{C^{-1}} \begin{pmatrix} p_0^T \\ p_1^T \\ p_2^T \end{pmatrix}$$

Wie kann ich sicherstellen, dass eine Polynomkurve an vorgegebenen Stützstellen (Parameterwerte bzw. Knoten $t_0 < t_1 < \cdots < t_n$) genau durch die Punkte $P_0, \ldots, P_n$ verläuft?



Dies lässt sich unter anderem mit den Lagrange-Polynomen erfüllen.

Gegeben: Stützstellen (Parameterwerte bzw. Knoten) $t_0 < t_1 < \cdots < t_n$

Lagrange-Polynome sind definiert durch

$$L_i^n(t) = \frac{(t-t_0)\ldots(t-t_{i-1})(t-t_{i+1})\ldots(t-t_n)}{(t_i-t_0)\ldots(t_i-t_{i-1})(t_i-t_{i+1})\ldots(t_i-t_n)} = \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{t-t_j}{t_i-t_j}$$

Setzt man nun t_k in dieses Polynom ein, dann gilt:

$$L_i^n(t_k) = \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{t_k-t_j}{t_i-t_j} = \delta_{ik} = \begin{cases} 1 & \text{, falls } i=k \\ 0 & \text{, sonst} \end{cases}$$

Somit kann das Interpolationsproblem wie folgt gelöst werden:

$$p(t) = \sum_{i=0}^{n} P_i L_i^n(t)$$

Am Knoten t_k hat Punkt P_k das Gewicht 1, alle anderen 0.

Eigenschaften der Lagrange-Polynome:

- Basis: Bilden eine Basis der Polynome von Grad n.
- Zerlegung der Eins :

$$\sum_{i=0}^{n} L_i^n(t) = 1$$

(Es gibt genau ein Polynom von Grad n mit $n+1$ Einsstellen!)

- Affine Invarianz : $(A(x) = U \cdot x + t)$

$$A\left(\sum_{i=0}^{n} P_i L_i^n(t)\right) = U \cdot \left(\sum_{i=0}^{n} P_i L_i^n(t)\right) + t = \left(\sum_{i=0}^{n} U \cdot P_i L_i^n(t)\right) + t$$

$$= \sum_{i=0}^{n} U \cdot P_i L_i^n(t) + \sum_{i=0}^{n} t \cdot L_i^n(t) = \sum_{i=0}^{n} (U \cdot P_i + t)\, L_i^n(t) = \sum_{i=0}^{n} A\left(P_i\right) L_i^n(t)$$

Dies folgt aus der Eigenschaft Zerlegung der Eins.

Bemerkung: Man erhält also dieselbe Interpolationskurve, unabhängig davon, ob man zuerst die Stützpunkte affin verschiebt und dann die Kurve berechnet oder ob man zuerst die Kurve berechnet und dann die Kurvenpunkte affin transformiert.

Zusammenfassung:

- Die Lagrange Polynome vom Grad n bilden eine Basis des $n+1$-dimensionalen Polynomraums. (Es gibt Matrixtransformation, die die Monombasis in die Lagrangebasis konvertiert)
- Die Koeffizienten P_i haben in der Lagrange-Darstellung des Polynoms eine geometrische Bedeutung
- Die Kurvenform hängt neben den Stützstellen (P_i) auch von der Wahl der t_i , d.h. der Parametrisierung ab
- Bei höheren Polynomgraden zeigt die Polynominterpolation eine unerwünschte Welligkeit

Hermite-Polynome

Die Hermite-Basis ist eine Polynom Basis mit der die Interpolation der Punkte und Ableitungen möglich ist. Sie ist definiert als die Lösung der Gleichungen:

$$H_i^3(0) = \delta_{i0} \qquad H_i^{3\prime}(0) = \delta_{i1} \qquad H_i^{3\prime}(1) = \delta_{i2} \qquad H_i^3(1) = \delta_{i3}$$

Die Lösungen lassen sich schreiben als

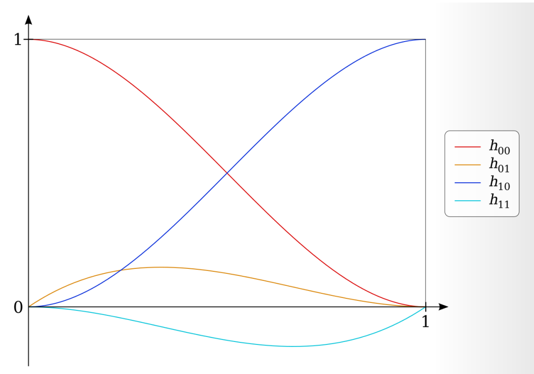
$$H_0^3(t) = (1-t)^2(1+2t) \qquad H_1^3(t) = t(1-t)^2$$

$$H_2^3(t) = -t^2(1-t) \qquad H_3^3(t) = (3-2t)t^2$$

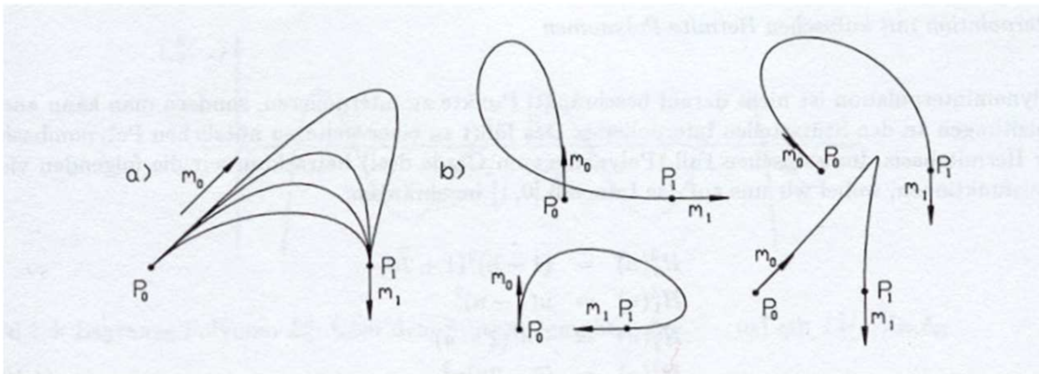
Die Kurve

$$p(t) = p_0 H_0^3(t) + m_0 H_1^3(t) + m_1 H_2^3(t) + p_1 H_3^3(t)$$

heißt Hermite-Kurve und geht an der Stelle 0 durch den Punkt p_0 mit der Ableitung m_0 und an der Stelle 1 durch den Punkt p_1 mit der Ableitung m_1.



Hermite-Basisfunktionen



Beispiele für Hermite-Kurven

Gegeben die Punkte und Ableitungen in $t = 0$ und $t = 1$:

$$p_0 = f(0) \qquad m_0 = f'(0) \qquad m_1 = f'(1) \qquad p_1 = f(1)$$

Gesucht ist das kubische Polynom, das diese Bedingungen erfüllt.

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 1 & 1 & 1 & 1 \end{pmatrix}}_{C} \begin{pmatrix} c_0^T \\ c_1^T \\ c_2^T \\ c_3^T \end{pmatrix} = \begin{pmatrix} p_0^T \\ m_0^T \\ m_1^T \\ p_1^T \end{pmatrix}$$

$$\begin{pmatrix} c_0^T \\ c_1^T \\ c_2^T \\ c_3^T \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & -1 & 3 \\ 2 & 1 & 1 & -2 \end{pmatrix}}_{C^{-1}} \begin{pmatrix} p_0^T \\ m_0^T \\ m_1^T \\ p_1^T \end{pmatrix}$$

Definition (Bernsteinpolynome)

Die Polynome

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i} \qquad t_i \in [0,1]$$

heißen *Bernsteinpolynome* über dem Intervall $[0,1]$ vom Grad n und bilden eine Basis des $(n+1)$-dimensionalen Polynomraum.

Eigenschaften der Bernsteinpolynome:

- Zerlegung der Eins :

$$\sum_{i=0}^{n} B_i^n(t) = ((1-t)+t)^n = 1$$

- Positivität :

$$B_i^n(t) \geq 0 \qquad t \in [0,1]$$

- Rekursion :

$$B_i^n(t) = t \cdot B_{i-1}^{n-1}(t) + (1-t) \cdot B_i^{n-1}(t)$$

- Symmetrie :

$$B_i^n(t) = B_{n-i}^n(1-t)$$

Definition (Bézierkurve)

Die Kurve

$$p(t) = \sum_{i=0}^{n} b_i B_i^n(t) \qquad t \in [0,1] \quad b_i \in \mathbb{R}^d$$

heißt *Bézierkurve* über dem Intervall $[0,1]$ vom Grad n. Die Punkte b_i heißen *Bézierpunkte* oder *Kontrollpunkte* und bilden das *Bézierpolygon* oder *Kontrollpolygon*.

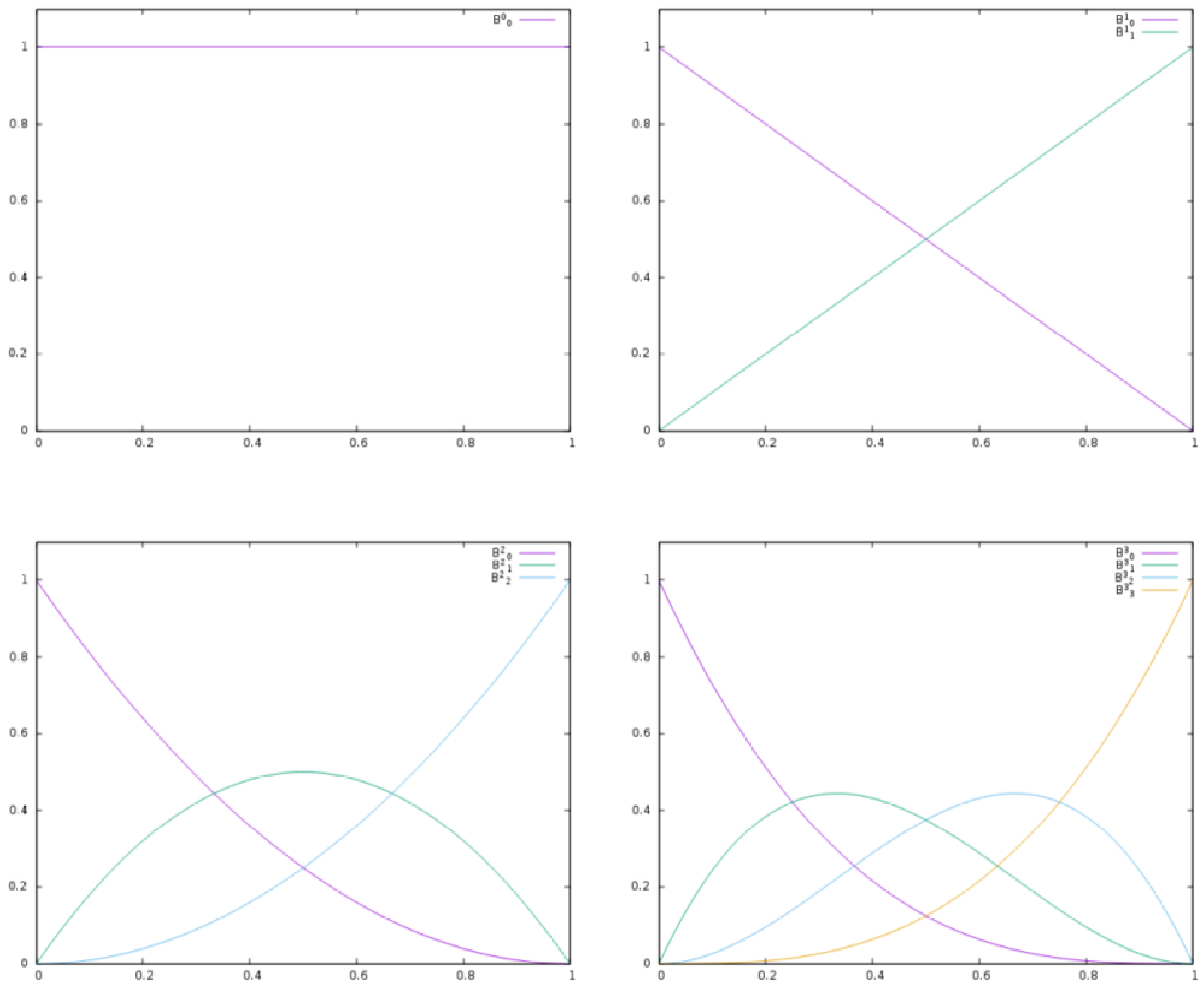
Eigenschaften von Bézierkurven:

- Approximation : Die Bezierkurve approximiert das Kontrollpolygon
- Affine Invarianz :

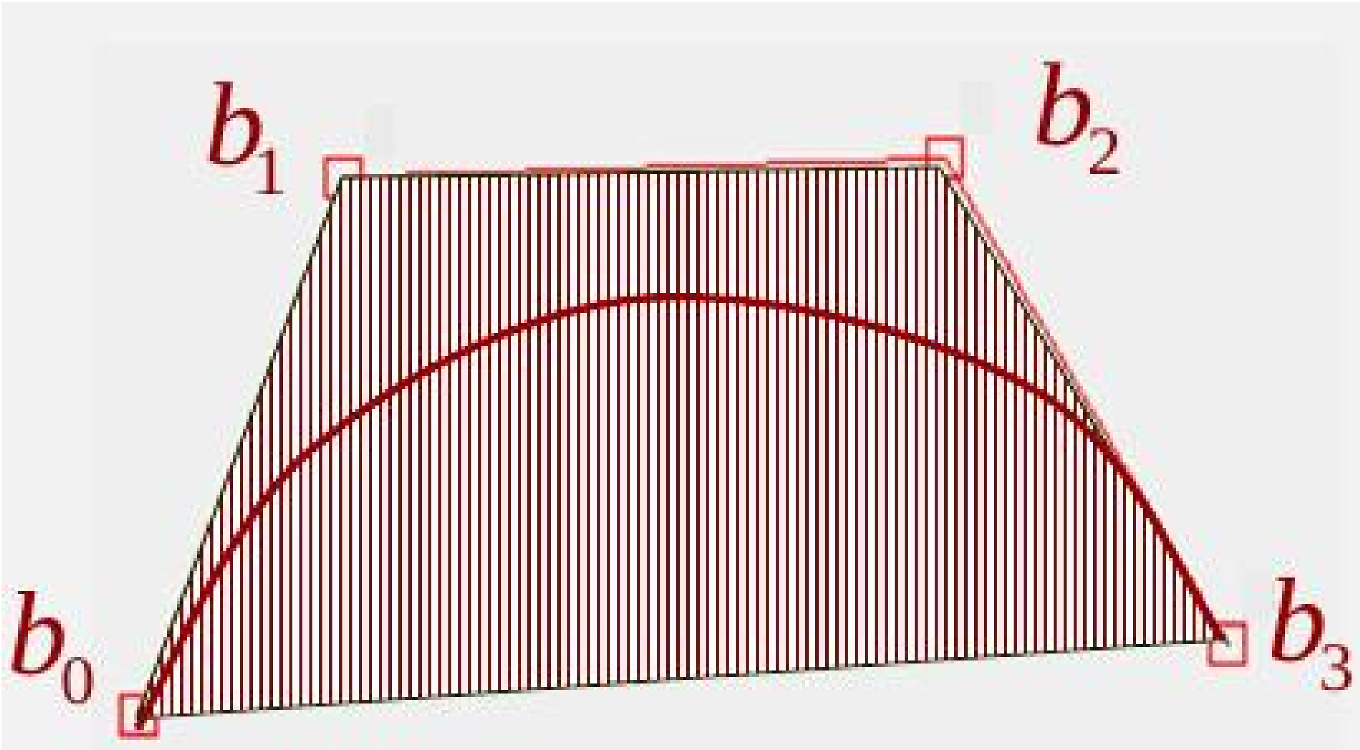
$$A\left(\sum_{i=0}^{n} b_i B_i^n(t)\right) = \sum_{i=0}^{n} A(b_i)\, B_i^n(t)$$

  Dies folgt aus der Eigenschaft der Bernsteinpolynome und der Definition von affinen Abbildungen
- Konvexität : Aus $B_i^n(t) \geq 0$, $t \in [0,1]$ folgt, dass für alle $t \in [0,1]$ die Kurvenpunkte $p(t)$ in der konvexen Hülle der Bézierpunkte liegen

Parametrisierungen von Polynomen



Beispiel einer Bézierkurve:



Es ist gut zu erkennen, dass die Kurve innerhalb der konvexen Hülle der Kontrollpunkte liegt.

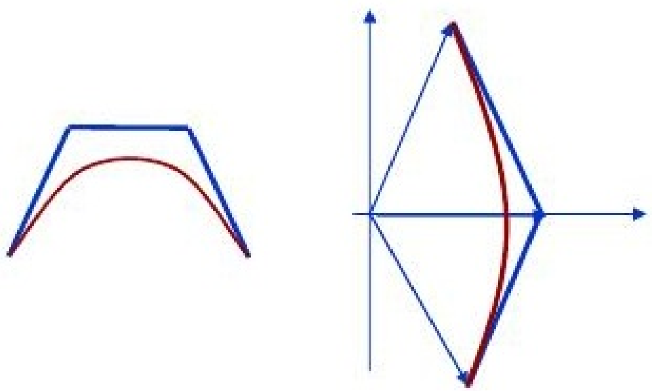
Die Ableitung einer Bézierkurve ist wieder ein Polynom. Definiert man $\Delta^k b_i$ rekursiv als

$$\Delta^0 b_i = b_i \qquad\qquad \Delta^k b_i = \Delta^{k-1} b_{i+1} - \Delta^{k-1} b_i$$

so lässt sich die Ableitung einer Bézierkurve selbst auch wieder in Bézierdarstellung schreiben:

$$p^{(k)}(t) = \frac{n!}{(n-k)!} \sum_{i=0}^{n-k} \Delta^k b_i \, B_i^{n-k}(t) \qquad\qquad t \in [0,1]$$

Zum Beispiel ist $p'(0) = n \cdot \Delta^1 b_0 = n(b_1 - b_0)$.



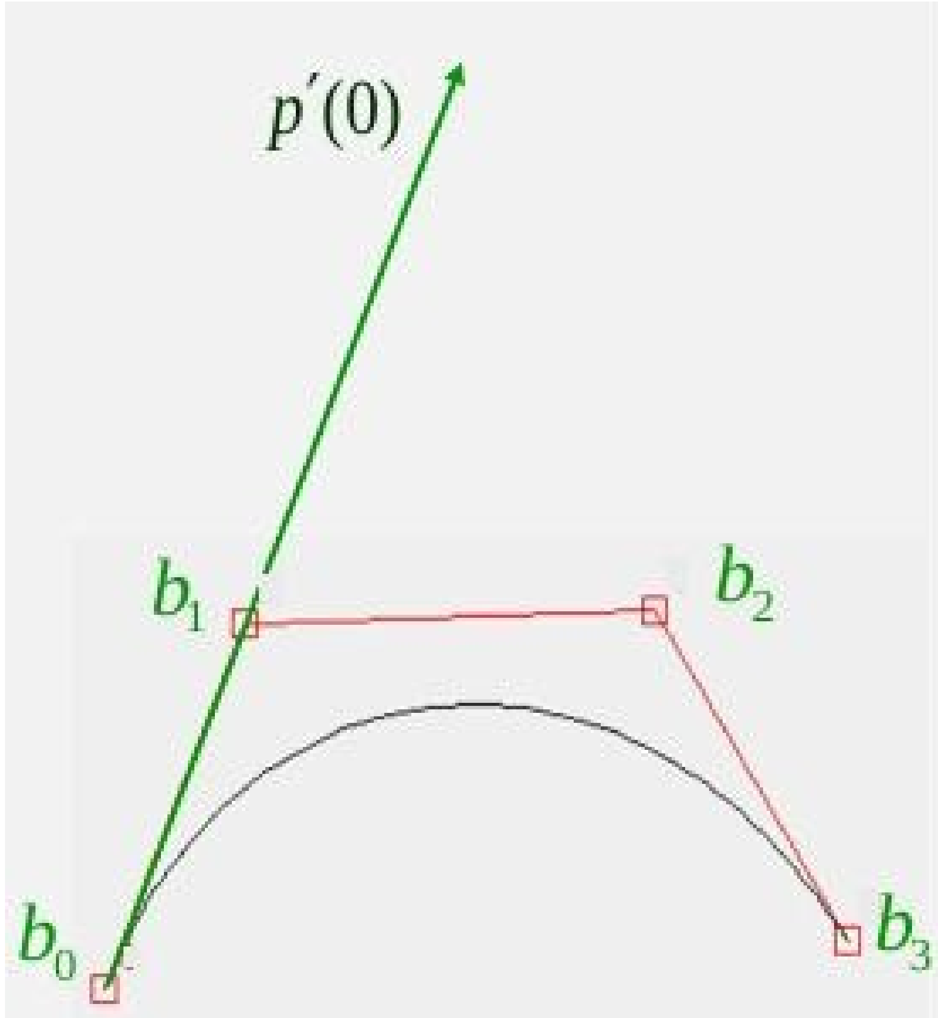
Bézierkurven verlaufen durch den Anfangs- und Endpunkt des Kontrollpolygons:

$$p(0) = b_0 \qquad p(1) = b_n$$

Für die Ableitungen in diesen Punkten gilt dabei:

$$p'(0) = n \cdot (b_1 - b_0)$$
$$p'(1) = n \cdot (b_n - b_{n-1})$$

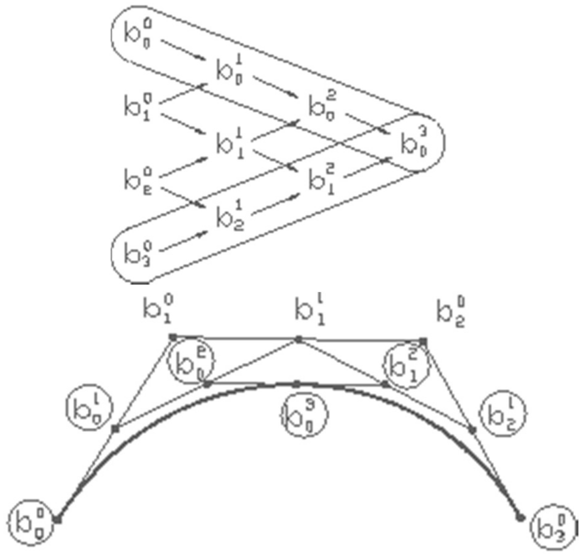
Die Tangenten von Anfangs- und Endpunkt verlaufen also entlang des Anfangs- und Endsegments des Kontrollpolygons und sind mit dem Faktor n skaliert worden.



Algorithmus von de Casteljau

Gegeben die Bézierkurve $p(t) = \sum_{i=0}^{n} b_i B_i^n(t), b_i \in \mathbb{R}^d$ betrachte die partiellen Bézierkurve:

$$b_i^r(t) = \sum_{j=0}^{r} b_{i+j} B_j^r(t)$$



So gilt:

$$b_0^n(t) = p(t)$$

$$b_i^0(t) := b_i$$

$$b_i^k(t) := t b_{i+1}^{k-1} + (1-t) b_i^{k-1} \tag{1}$$

Definition (Algorithmus von Casteljau)

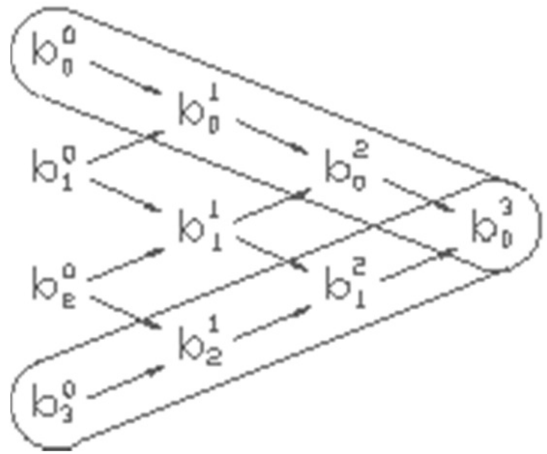
Berechne $p(t)$ rekursiv durch Konvexkombination (Gl. 1)

- Pro Iterationsschritt verringert sich die Anzahl der Punkte um eins.
- Stabile Auswertung durch Konvexkombinationen.
- Aufwand $O(n^2)$ mit Polynomgrad n

Algorithmus von de Casteljau

Ableitungen werden implizit mitberechnet:

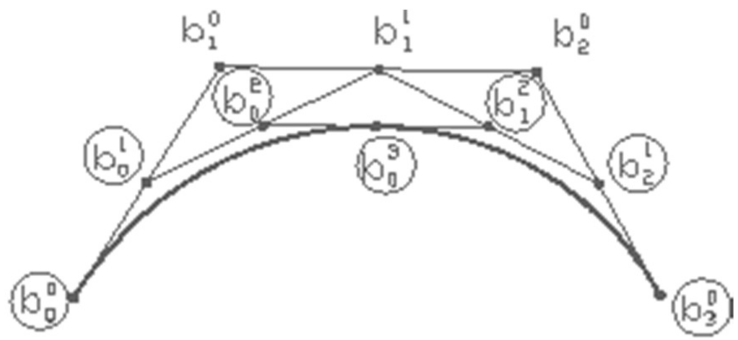
$$p^{(k)}(t) = \frac{n!}{(n-k)!} \sum_{i=0}^{k} \binom{k}{i} (-1)^{k-i} b_i^{n-k}(t)$$



Für die erste Ableitung gilt beispielsweise

$$p^{'}(t) = n \cdot (b_i^{n-1}(t) - b_0^{n-1}(t))$$

Unterteilung: Darüber hinaus lässt sich mit Hilfe des de Casteljau-Algorithmus eine Bézierkurve in zwei Teilsegmente zerlegen. Die neuen Kontrollpunkte ergeben sich aus dem oberen und unteren Rand des Schemas.
Nach wenigen Unterteilungsschritten (2-3) liefern die Kontrollpolygone der Teilkurven eine gute Approximation der Kurve (exponentielle Konvergenz)



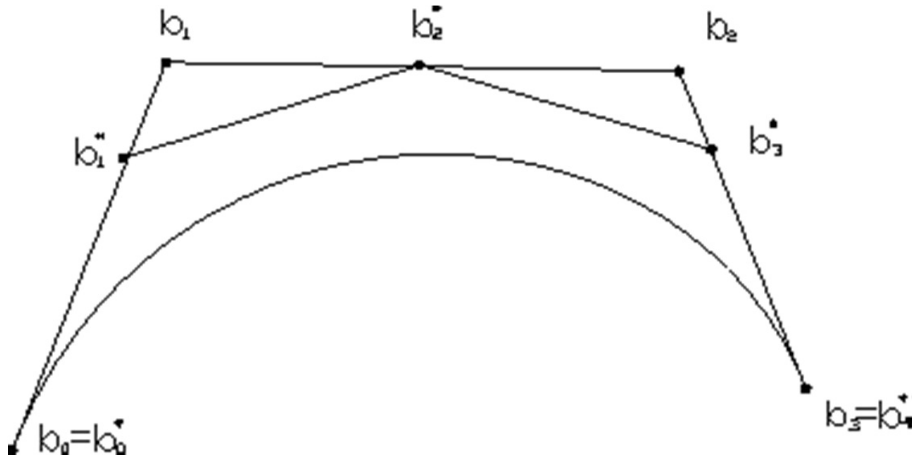
!

Wenn zur genaueren Approximation einer geometrischen Figur mehr Freiheitsgrade benötigt werden, so lässt sich der Grad der Kurve erhöhen:

$$b_0^* = b_0$$

$$b_k^* = \frac{k}{n+1} \cdot b_{k-1} + \left(1 - \frac{k}{n+1}\right) \cdot b_k \qquad k = 1, \ldots n$$

$$b_{n+1}^* = b_n$$



Man lässt also Anfangs- und Endpunkt stehen und bildet Konvexkombinationen ähnlich wie beim de Casteljau-Algorithmus und generiert sich somit die neuen Punkte. Insgesamt hat man so den Grad um Eins erhöht. Dies kann beliebig oft wiederholt werden.

Zusammenfassung:

+ Geometrisch anschauliche Bedeutung der Koeffizienten
+ Bézierpolygon vermittelt schnellen Übersicht über den möglichen Kurvenverlauf
+ Einfach zu implementieren
- Kurvengrad ist gekoppelt an die Zahl der Kontrollpunkte, was zu hohen Polynomgraden führt
- Die Änderung eines Bézierpunktes wirkt sich auf die **gesamte Kurve** aus und damit auch auf Bereiche, die eventuell nicht mehr geändert werden sollen

Insbesondere der letzte Punkte ist für das Modellieren sehr problematisch, da man meistens nur lokal Änderungen vornehmen möchte.

Problem: Wichtige Objekte wie z.B. Kreise oder Ellipsen lassen sich nicht durch Polynome parametrisieren (Fundamentalsatz der Algebra).

Lösung: Rationale Kurven

- Rationale Kurven im $\mathbb{R}^3$ werden als Polynome des $\mathbb{R}^4$ definiert. Danach wird eine perspektivische Projektion vom Ursprung des $\mathbb{R}^4$ auf die Hyperebene $w = 1$ durchgeführt, wobei w die vierte Komponente eines Vektors im $\mathbb{R}^4$ ist
- Auf diese Art können auch unendlich ferne Stützpunkte verwendet werden
- Kegelschnitte lassen sich exakt darstellen

Definition (Rationale Bézierkurve)

Eine *rationale Bézierkurve* ist definiert durch

$$R(u) = \frac{\sum_{i=0}^{n} w_i \cdot b_i \cdot B_i^n(u)}{\sum_{i=0}^{n} w_i \cdot B_i^n(u)}$$

mit Bézierpunkten $b_i \in \mathbb{R}^d$ und Gewichten $w_i \in \mathbb{R}, w_i \geq 0, w_0 = 1 = w_n$.

- Wegen $w_0 = 1 = w_n$ ist der Nenner ungleich Null und R somit wohldefiniert
- Wählt man $w_i = 1 \; \forall i = 0, \ldots, n$ ist R ist eine gewöhnliche Bézierkurve

Projektionseigenschaft: Für Bezier-Punkte $b_i = (x_i, y_i, z_i)^T$ und Gewichte w_i wie oben definiere homogene Bezierpunkte $b_i^h = (x_i \cdot w_i, y_i \cdot w_i, z_i \cdot w_i, w_i)^T \in \mathbb{R}^4$ und Bézierkurve im $\mathbb{R}^4$: $F(u) = \sum_{i=0}^{n} B_i^n(u) \cdot b_i^h$

Betrachte die Projektion (vgl. Folien dort; in H sind Fernpunkte mit $w = 0$)

$$\Pi\colon P(\mathbb{R}^4) \backslash H \to \mathbb{R}^3$$
$$(xw, yw, zw, w)^T \mapsto (x, y, z)^T$$

Dann gilt

$$\Pi(F(u)) = \left(\frac{F_x(u)}{F_w(u)}, \frac{F_y(u)}{F_w(u)}, \frac{F_z(u)}{F_w(u)} \right)^T = R(u)$$

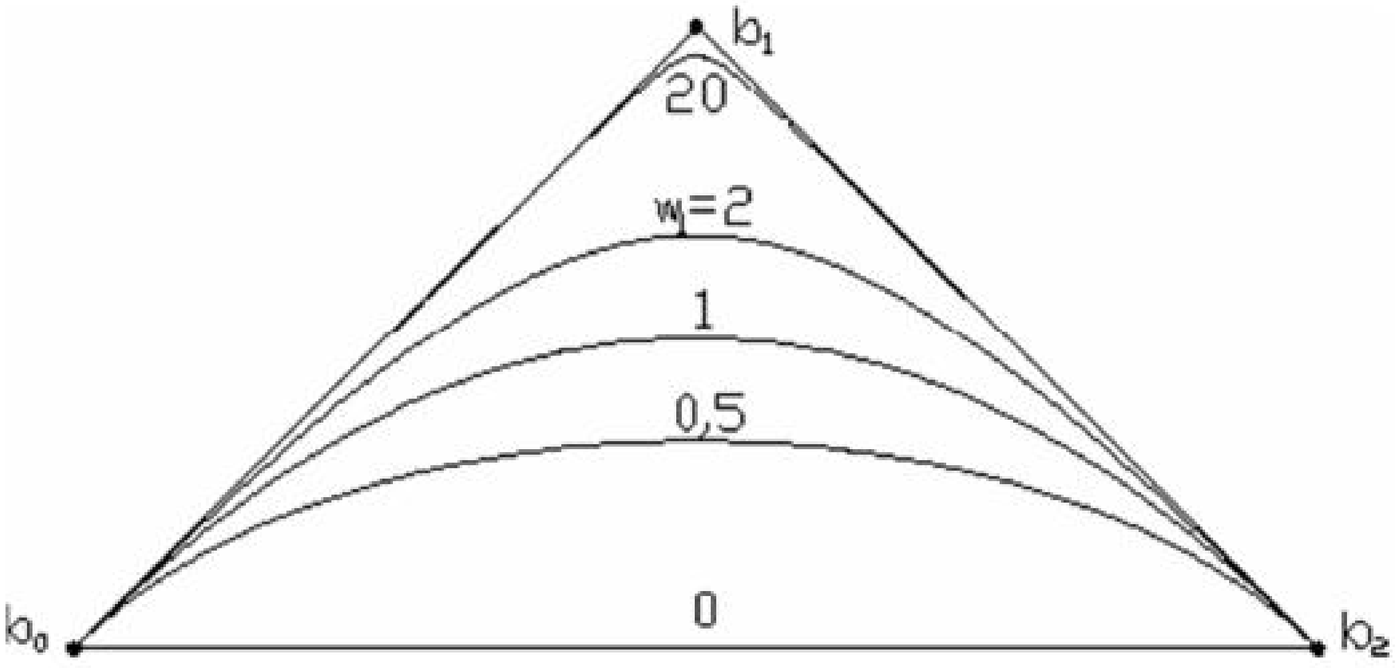
d.h. die rationalen Kurven im $\mathbb{R}^3$ erhält man aus Bézier-Kurven im $\mathbb{R}^4$, indem man die Bézierkurven des $\mathbb{R}^4$ vom Ursprung aus auf die Hyperebene $w = 1$ projiziert

Eigenschaften rationaler Bézier-Kurven

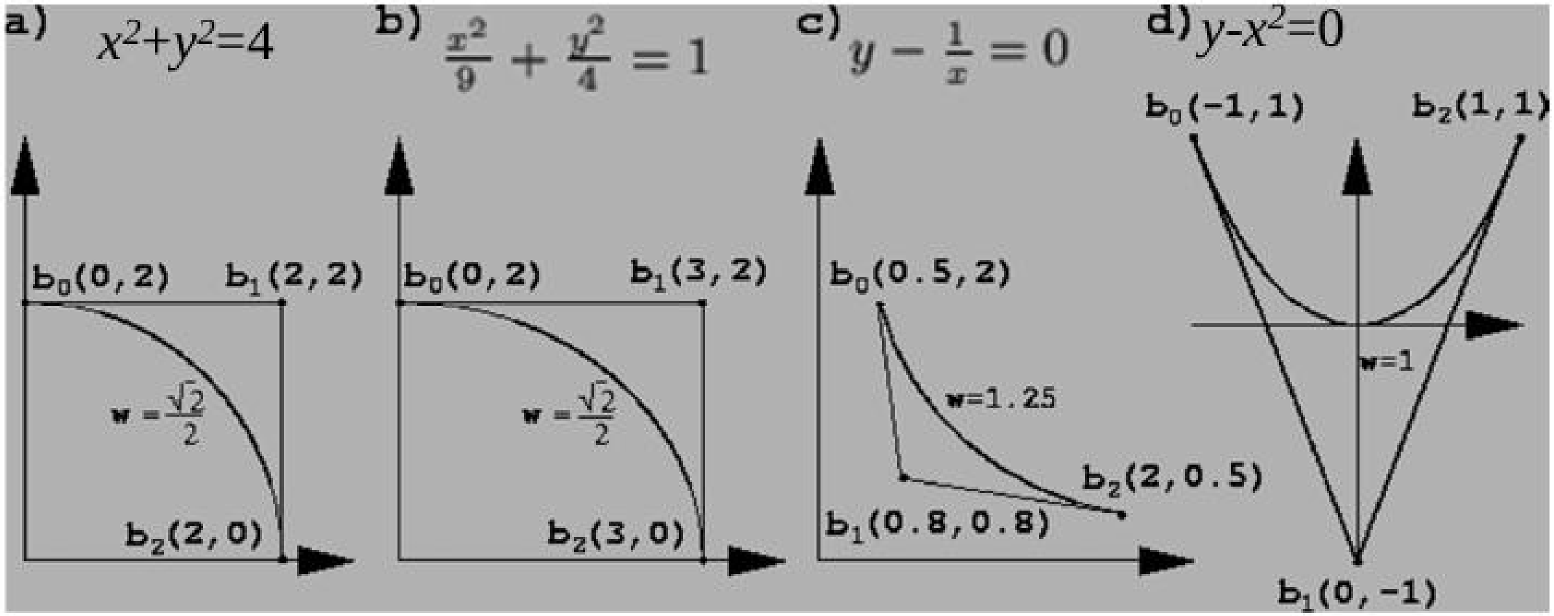
- Für $u \in [0,1]$ liegt die Kurve $R(u)$ in der konvexen Hülle der Bézier-Punkte
- Die Kurve verläuft durch Start-/Endpunkt des Kontrollpolygons. Kurve und Kontrollpolygon sind dort tangential
- Die Kurve ist invariant unter affinen Abbildungen
- Die Kurve ist invariant unter projektiven Abbildungen. Insbesondere ist das perspektivische Bild einer rationalen Kurve wieder eine rationale Kurve
- Die Kurve besitzt die "variation diminishing property" (d.h., sie ist glatter als das Kontrollpolygon und hat somit weniger Schnittpunkte mit bel. Ebene)

Wirkung der Gewichte:
Wird das Gewicht w_k vergrößert, so bewegt sich die Kurve zum k-ten Kontrollpunkt b_k:

$$\lim_{w_k \to \infty} R(t) = b_k \quad \forall t \in]0,1[$$



Kegelschnitte (z.B. Kreise, Ellipsen) können durch rationale Bézierkurven parametrisiert werden:



Splinekurven

Wie oben bereits angemerkt, möchte man nur lokal den Verlauf der Kurve ändern. Um dies zu erreichen, wurden die Splines eingeführt:

- Splines sind stückweise polynomielle Funktionen, wobei das zu jedem Segment gehörende Polynom nur einen beschränkten Grad hat
- Die Idee ist, dass an den Übergängen bestimmte differentiale Eigenschaften eingehalten (Stetigkeit) werden

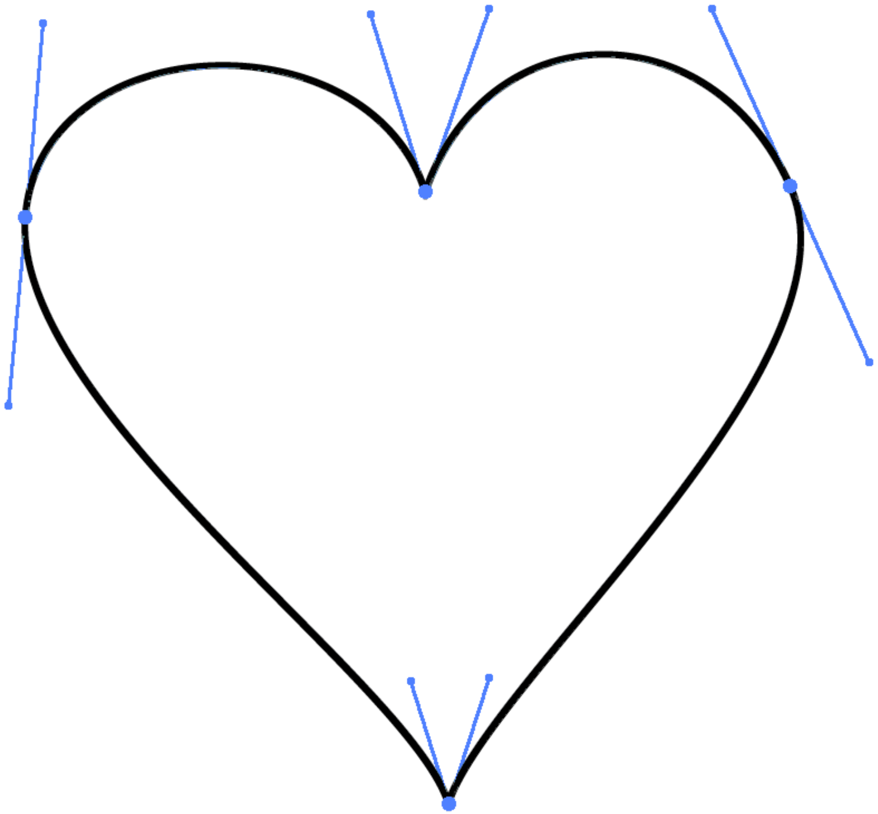
Definition (Spline)

Ein *Spline* ist eine stückweise stetig-differenzierbare Abbildung

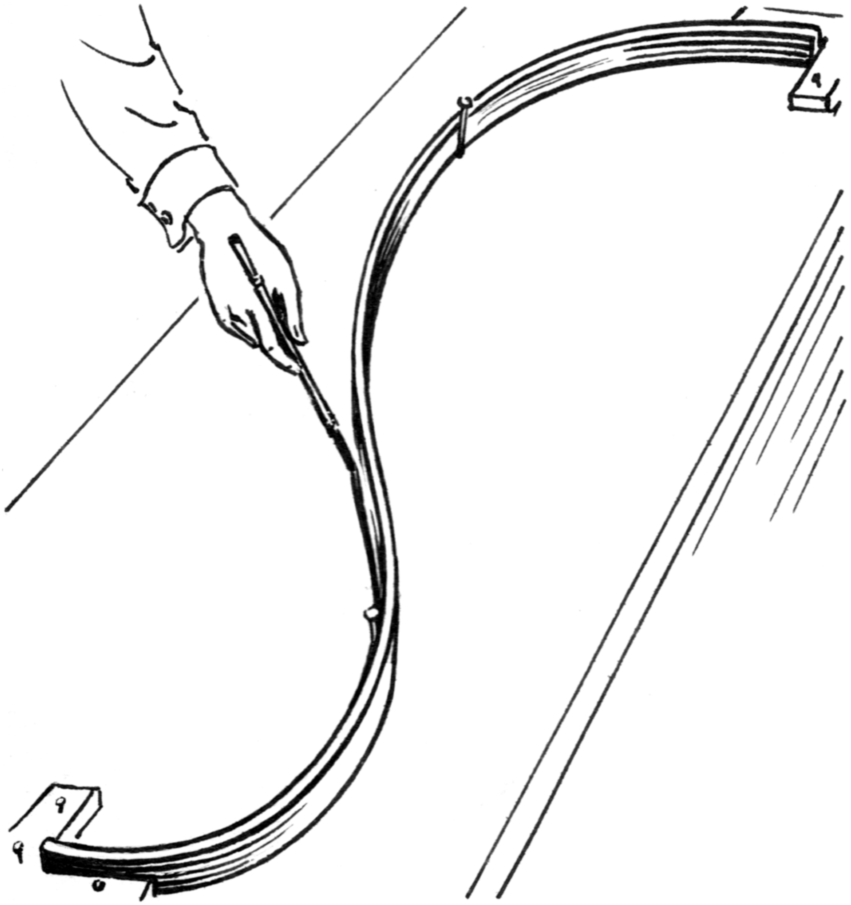
$$q : [t_0, t_n] \to \mathbb{R}^d \qquad q|_{[t_i,t_{i+1}]} \in C^\infty \ \forall i$$

von einer endlichen Menge von Intervallen in den $\mathbb{R}^d$. Die Intervalle $[t_i, t_{i+1}]$ werden dabei als *Stützstellenvektor* $T = (t_0, t_1, \dots, t_n)$ mit $t_0 \leq t_1 \leq \cdots \leq t_n$ geschrieben. Die Segmente $q|_{[t_i,t_{i+1}]}$ heißen *Spline-Segmente*.

Die Spline-Segmente stoßen an den Stützstellen zusammen. Die differenziellen Eigenschaften an diesen Stellen sind von großer Bedeutung.



Bézier-Spline in Illustrator



Mechanische Straklatte

UNIVERSITÄT BONN

Definition (Parametrisch stetiger Anschluss (C^n-stetiger Übergang))

Zwei n-mal stetig-differenzierbare Kurven

$$q : [a_1, b_1] \to \mathbb{R}^d$$
$$r : [a_2, b_2] \to \mathbb{R}^d$$

schließen an der Stelle b_1, a_2 C^n*-stetig aneinander*, falls

$$q^{(k)}(b_1) = r^{(k)}(a_2) \qquad \forall k \in \{0, \ldots, n\}$$

d.h. die Richtungen und die Länge der Ableitungen stimmen überein.

Definition (Geometrisch stetiger Anschluss (G^n-stetiger Übergang))

Zwei n-mal stetig-differenzierbare Kurven

$$q : [a_1, b_1] \to \mathbb{R}^d$$
$$r : [a_2, b_2] \to \mathbb{R}^d$$

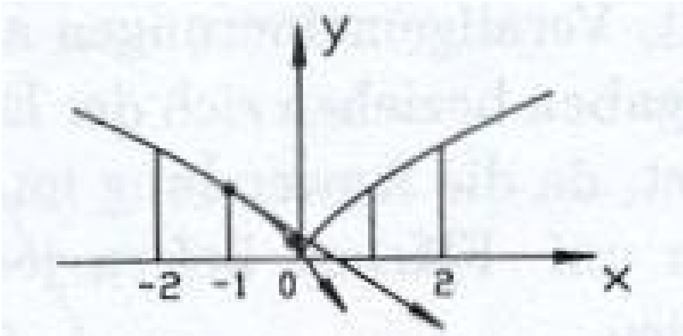
schließen an der Stelle b_1, a_2 G^n-stetig aneinander, falls es eine reguläre Umparametrisierung $r_2 = r(\alpha(t))$ gibt, dass q, r_2 C^n stetig aneinander anschliessen.

Im Allgemeinen gilt: p regulär und C^n- stetig $\Rightarrow$ G^n-stetig

Nicht reguläre Kurven: Betrachtet man die Kurve

$$q(u) = (u^3, u^2)^T \quad , \quad u \in [-4, 4]$$

als zwei im 'Punkt $u = 0$ aneinanderschließende Kurven, so sind diese dort C^n-stetig aber nicht G^n-stetig.



Schließen zwei Kurven $q_1\colon [a_1, b_1] \to \mathbb{R}^3, q_2\colon [a_2, b_2] \to \mathbb{R}^3$ an der Stelle (b_1, a_2) G^n-stetig aneinander, so gibt es eine zu q_1 äquivalente Kurve $r\colon [a, b] \to \mathbb{R}$ und eine bijektive differenzierbare Abbildung $\phi\colon [a_0, b_0] \to [a_1, b_1], \phi'(u) > 0$ für $u \in [a_0, b_0]$ mit $r_1 = q_1 \circ \phi$, sodass r_1 und q_2 C^n-stetig sind. Differenziert man r 1 nach der Kettenregel, so folgt:

$$
\begin{aligned}
q_2(a_2) &= r_1(b_0) = q_1(\varphi(b_0)) \\
q_2'(a_2) &= r_1'(b_0) = (q_1 \circ \varphi)'(b_0) = q_1'(\varphi(b_0))\varphi'(b_0) \\
q_2''(a_2) &= r_1''(b_0) = (q_1 \circ \varphi)''(b_0) = q_1''(\varphi(b_0))\varphi'(b_0)^2 + q_1'(\varphi(b_0))\varphi''(b_0)
\end{aligned}
$$

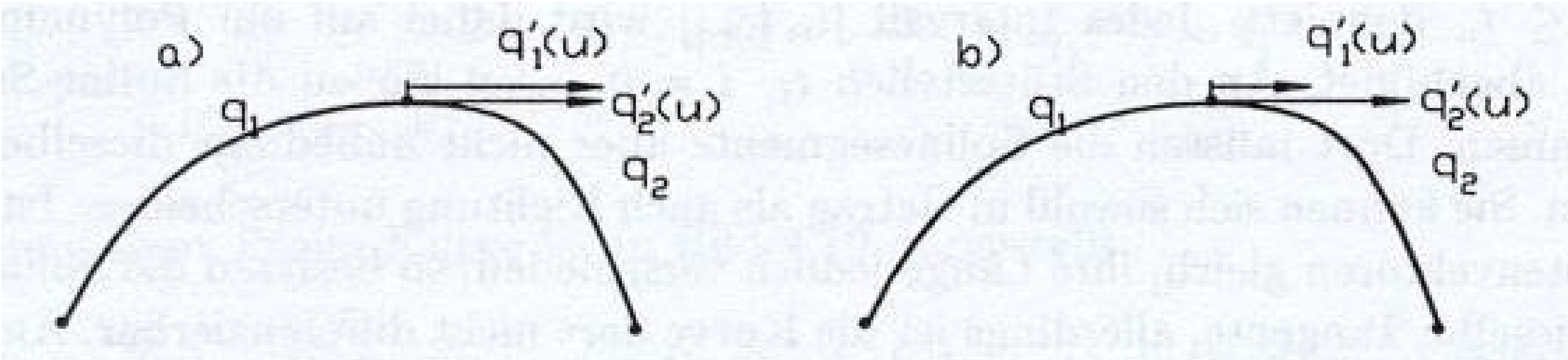
Die Koeffizienten $\beta_i := \varphi^{(i)}(b_0)$ werden als β-Constraints bezeichnet. Gilt zusätzlich $\varphi(b_0) = b_1$, so können die Gleichungen mit Hilfe der β-Constraints wie folgt geschrieben werden:

$$
\begin{aligned}
q_2(a_2) &= q_1(b_1) \\
q_2'(a_2) &= \beta_1 q_1'(b_1) \\
q_2''(a_2) &= \beta_1^2 q_1''(b_1) + \beta_2 q_1'(b_1) \\
&\dots
\end{aligned}
$$

Existieren umgekehrt β_1, β_2 mit $\beta_1 > 0$, sodass obige Gleichungen erfüllt werden, so definieren q_1 und q_2 eine G^2-Kurve.

Beziehungen zwischenden Stetigkeiten:

- G^0-Stetigkeit entspricht C^0-Stetigkeit
- G^1-Stetigkeit am Übergang von q_1 und q_2 ist äquivalent dazu, dass der Übergang stetig ist und beide Tangentenvektoren dort gleiche Richtung besitzen
- G^2-Stetigkeit ist äquivalent dazu, daß q_1 und q_2 stetig sind, die Tangenten am Übergang dieselbe Richtung besitzen und der Krümmungsvektor dort übereinstimmt



Die Kurvensegmente in der Monom- und Lagrange-Basis können auf einfache Weise in der Regel nur mit C^0-Stetigkeit aneinandergefügt werden.
Mit Hermitepolynomen lassen sich zumindest C^1-stetige Splines relativ einfach erzeugen:

Gegeben: Parameterwerte $t_0, \ldots, t_n$, die dazu gehörigen Punkte $p_0, \ldots, p_n$ und Ableitungen $m_0, \ldots, m_n$

Die Hermite-Segmente $q_i \ , \ i = 0, \ldots, n-1$ erfüllen die Interpolationsaufgabe:

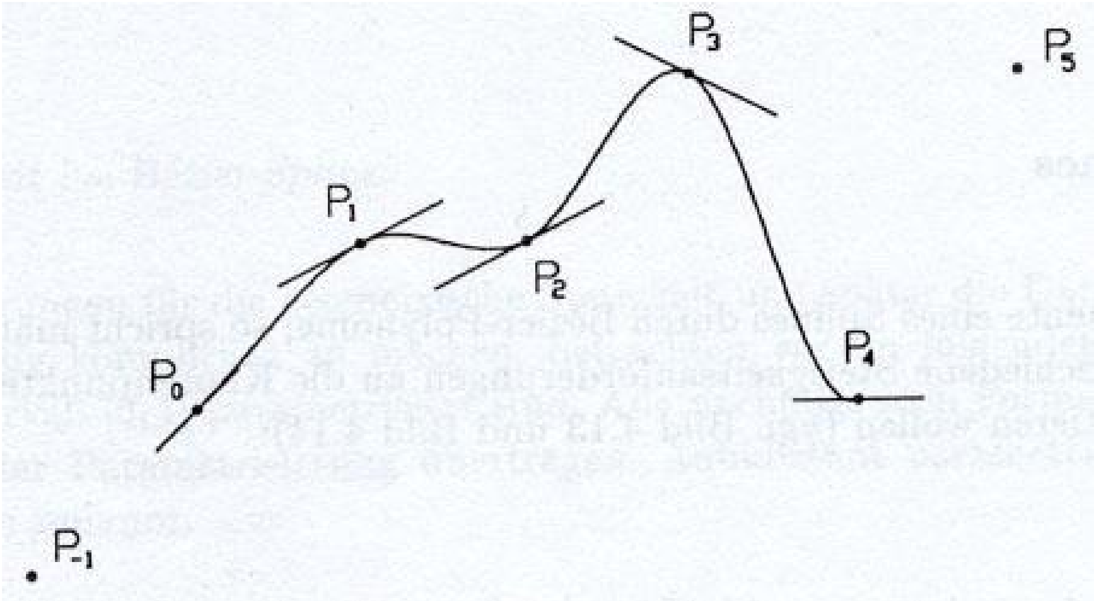
$$q_i(t) = p_i H_0^3(s_i) + \Delta_i m_i H_1^3(s_i) + \Delta_i m_{i+1} H_2^3(s_i) + p_{i+1} H_3^3(s_i)$$

mit $s_i = \frac{t-t_i}{t_{i+1}-t_i}$ und $\Delta_i = t_{i+1} - t_i$. Dies ist ein kubischer Hermite-Spline mit Segmenten q_i, die jeweils im Intervall $[t_i, t_{i+1}]$ die Kurve beeinflussen.

Problem: Die Tangenten m_i müssen bestimmt bzw. geschätzt werden. Die Länge dieser Vektoren bestimmt u. a. den Verlauf der Kurve stark. Dies ist in der Praxis meist sehr schwierig.

In der Praxis ist es schwierig, die Tangentenvektoren direkt anzugeben. Die Form der Kurve hängt stark von der Länge der Tangentenvektoren ab.

Die Tangentenvektoren müssen also geschätzt werden. Bei der FMILL Methode wird z. B. die Tangentenrichtung m_i im Punkt P_i parallel zur Sehne durch P_{i-1} und P_{i+1} gewählt. Der entstehende Interpolant wird als Catmull-Rom-Spline bezeichnet.



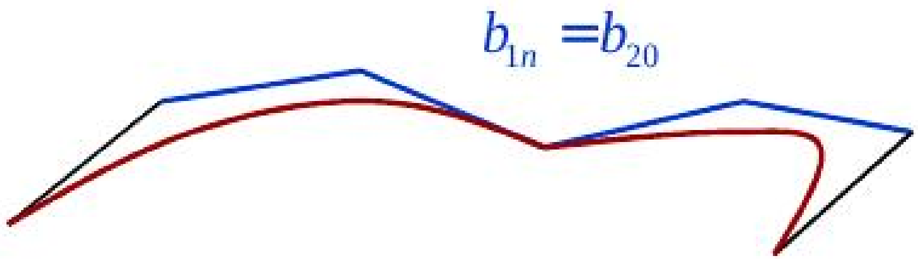
Gegeben: Bézierkurven $p_1(t) = \sum_{i=0}^{n} b_{1i} B_i^n(t)$ und $p_2(t) = \sum_{i=0}^{n} b_{2i} B_i^n(t)$
Diese können nun unterschiedlich aneinander angeschlossen werden:

C^0**-stetiger Anschluss:**

- Gemäß Definition muss gelten:

$$p_1(1) = p_2(0)$$

d.h. die Kurven schließen aneinander an:



Für Bézierkurven bedeutet dies also:

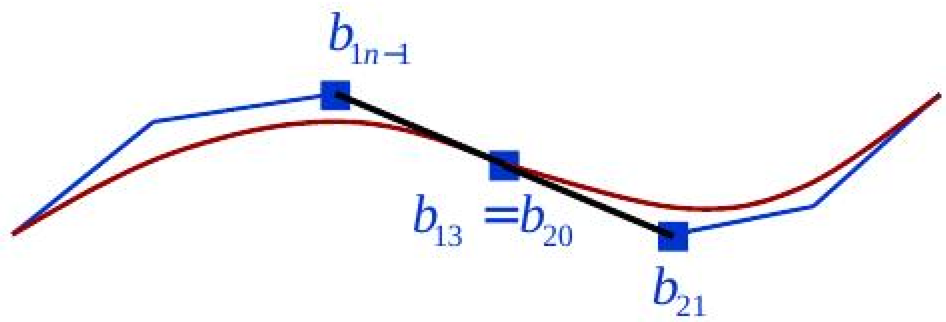
$$b_{1n} = b_{20}$$

C^1-stetiger Anschluss:

- Gemäß Definition muss gelten:

$$p_1(1) = p_2(0) \quad \text{und} \quad p_1^{'}(1) = p_2^{'}(0)$$

d.h. die Kurven schließen aneinander an und die Tangenten sind identisch:



Für Bézierkurven bedeutet dies also:

$$b_{1n} = b_{20}$$

und

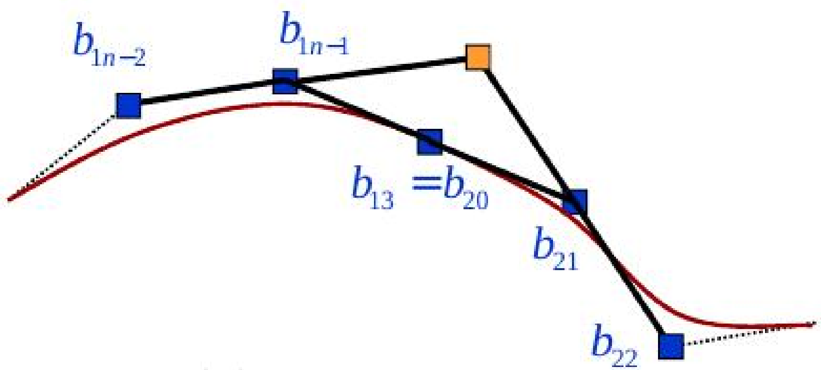
$$\begin{aligned} n \cdot (b_{1n} - b_{1n-1}) = n \cdot (b_{21} - b_{20}) &\Leftrightarrow b_{1n} - b_{1n-1} = b_{21} - b_{20} \\ &\Leftrightarrow b_{21} = b_{20} + (b_{1n} - b_{1n-1}) \end{aligned}$$

C^2-stetiger Anschluss:

- Gemäß Definition muss gelten:

$$p_1(1) = p_2(0) \quad \text{und} \quad p_1^{'}(1) = p_2^{'}(0) \quad \text{und} \quad p_1^{''}(1) = p_2^{''}(0)$$

d.h. die Kurven schließen aneinander an und die Tangenten und Krümmungen sind identisch:



Für Bézierkurven bedeutet dies also:

$$b_{1n} = b_{20}$$

und

$$b_{21} = b_{20} + (b_{1n} - b_{1n-1})$$

und

$$b_{1n-1} + (b_{1n-1} - b_{1n-2}) = b_{21} + (b_{21} - b_{22})$$

Analog zu den Bernsteinpolynomen bei Bézierkurven suchen wir nun Basisfunktionen für Splines:

Definition (Normalisierte B-Splines)

Sei $n \leq m$ und $T = (t_0 = \dots = t_n, t_{n+1}, \dots, t_m, t_{m+1} = \dots = t_{m+n+1})$ eine schwach monoton wachsende Folge von Knoten mit $t_i < t_{i+n+1}, 0 \leq i \leq m$.
Die rekursiv definierten Funktionen

$$N_i^0(t) := \begin{cases} 1 & \text{, falls } t_i \leq t < t_{i+1} \\ 0 & \text{, sonst} \end{cases}$$

$$N_i^k(t) := \frac{t - t_i}{t_{i+k} - t_i} \cdot N_i^{k-1}(t) + \frac{t_{i+1+k} - t}{t_{i+1+k} - t_{i+1}} \cdot N_{i+1}^{k-1}(t) \qquad 1 \leq k \leq n$$

heißen *normalisierte B-Splines* vom Grad n über T.

Bemerkung: Da der Abstand aufeinanderfolgender Knoten nicht konstant ist, werden sie auch als nicht uniforme normalisierte B-Splines bezeichnet.

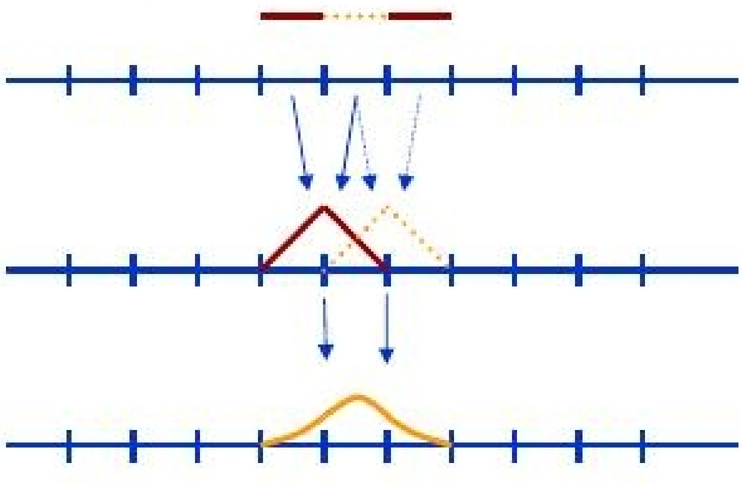
$N_i^n(t)$ besteht stückweise aus Polynomen vom Grad n über T:

$$N_i^0(t) := \begin{cases} 1 & \text{, falls } t_i \leq t < t_{i+1} \\ 0 & \text{, sonst} \end{cases}$$

$$N_i^k(t) := \frac{t - t_i}{t_{i+k} - t_i} \cdot N_i^{k-1}(t) + \frac{t_{i+1+k} - t}{t_{i+1+k} - t_{i+1}} \cdot N_{i+1}^{k-1}(t)$$

Die Funktionen $N_i^n(t)$ besitzen einen lokalen Träger, d.h.

$$N_i^n(t) = 0 \ \forall t \notin [t_i, t_{i+1+n}]$$



Eigenschaften von B-Splines:

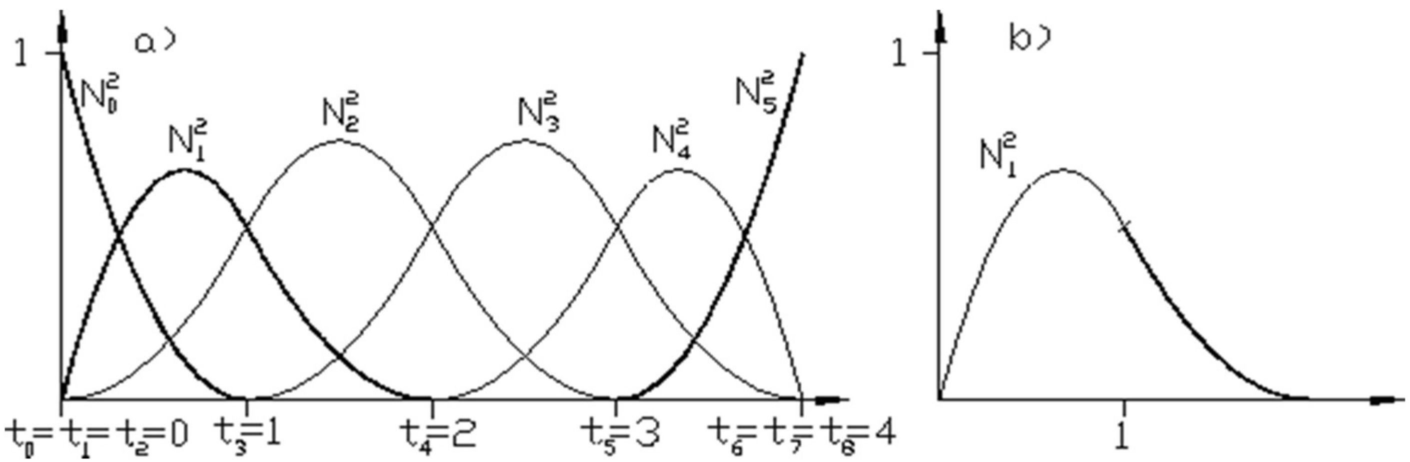
- Zerlegung der Eins :

$$\sum_{i=0}^{n} N_i^n(t) = 1$$

- Positivität :

$$N_i^n(t) \geq 0 \qquad t \in [t_0, t_{m+n+1}]$$

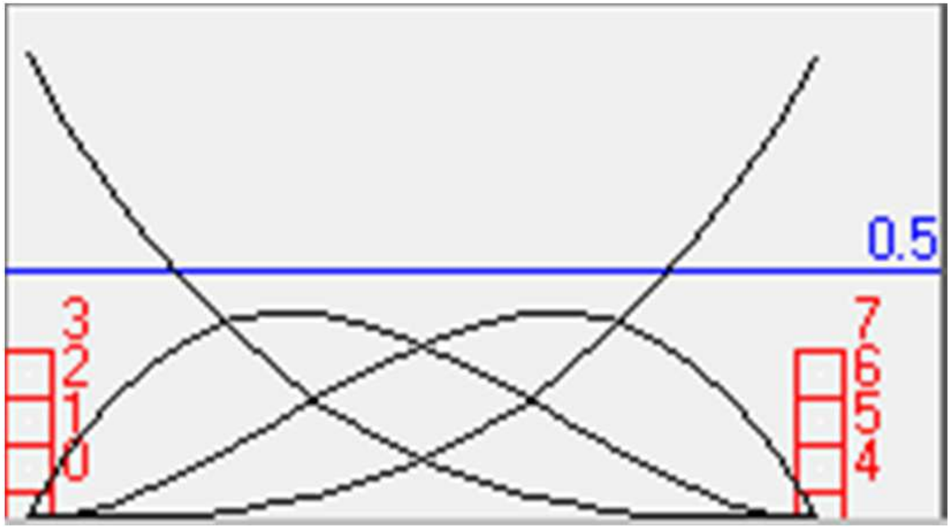
- Stetigkeit :
  - Ist t_j ein einfacher Knoten, d.h. $t_{j-1} < t_j < t_{j+1}$, so ist $N_i^n(t_j)$ mindestens C^{n-1}-stetig
  - Ist t_j ein Mehrfachknoten mit Multiplizität μ so ist $N_i^n(t_j)$ mindestens $C^{n-\mu}$-stetig



Die B-Splines $N_i^n(t)$ enthalten die Bernsteinpolynome $B_i^n(t)$ als Spezialfall:

$$T = \left(\underbrace{0, \ldots, 0}_{n+1}, \underbrace{1, \ldots, 1}_{n+1} \right)$$

Dann stimmen die B-Splines $N_i^n(t)$ auf T mit den Bernsteinpolynomen $B_i^n(t)$ überein.



Definition (B-Splinekurve)

Sei $n \leq m$ und $T = (t_0 = \cdots = t_n, t_{n+1}, \ldots, t_m, t_{m+1} = \cdots = t_{m+n+1})$ eine schwach monoton wachsende Folge von Knoten mit $t_i < t_{i+n+1}, 0 \leq i \leq m$ und $d_0, \ldots d_m \in \mathbb{R}^d$. Dann heißt die Kurve

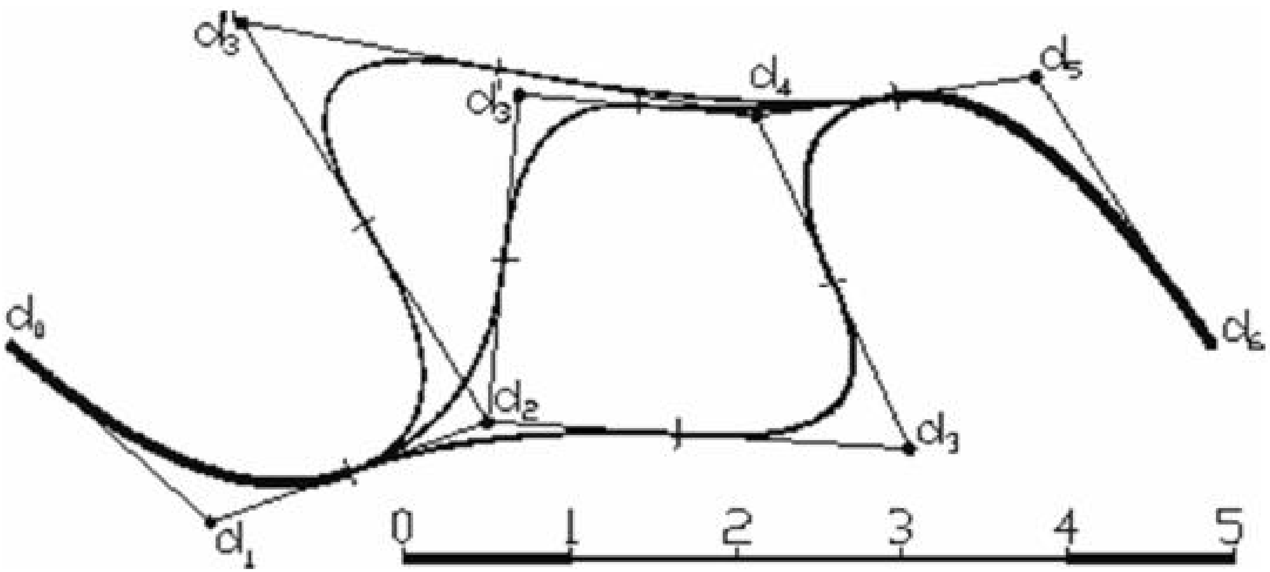
$$p(t) = \sum_{i=0}^{m} d_i N_i^n(t)$$

B-Splinekurve vom Grad n über T. Die Kontrollpunkte werden auch *de Boor-Punkte* genannt und bilden das Kontrollpolygon.

Eigenschaften von B-Splinekurve:

- Approximation : Die B-Splinekurve approximiert das Kontrollpolygon
- Affine Invarianz :
$$A\left(\sum_{i=0}^{m} d_i N_i^n(t)\right) = \sum_{i=0}^{m} A\left(d_i\right) N_i^n(t)$$
  Dies folgt aus der Eigenschaft der normalisieren B-Splines und der Definition von affinen Abbildungen
- Konvexität : Aus $N_i^n(t) \geq 0$, $t \in [0,1]$ folgt, dass für alle $t \in [t_0, t_{m+n+1}]$ die Kurvenpunkte $p(t)$ in der konvexen Hülle der B-Splinekurve liegen

- Da $N_i^n = 0$ falls $t \notin [t_i, t_{i+1}]$ beeinflußt der i-te de Boorpunkt d_i die Kurve nur über dem Parametergebiet $[t_i, t_{i+n+1}]$.
- Umgekehrt wird die Form der Kurve über dem Parameterintervall $[t_i, t_{i+1}]$ nur von den de Boorpunkten $d_{i-n}, \ldots, d_i$ beeinflußt.
  Beispiel:



Zu sehen sind drei B-Splinekurven. Der de Boor-Punkt d_3 wurde dabei verschoben und beeinflusst nur lokal die Kurve.

Gegeben: B-Splinekurve $p(t) = \sum_{i=0}^{m} d_i N_i^n(t)$ über dem Knotenvektor $T = (t_0 = \cdots = t_n, t_{n+1}, \ldots, t_m, t_{m+1} = \cdots = t_{m+n+1})$

Analog zum Schema von de Casteljau, kann rekursiv einen Punkt auf der Kurve berechnen:

$$d_i^0(t) := d_i \qquad i = l-n, \ldots, l$$

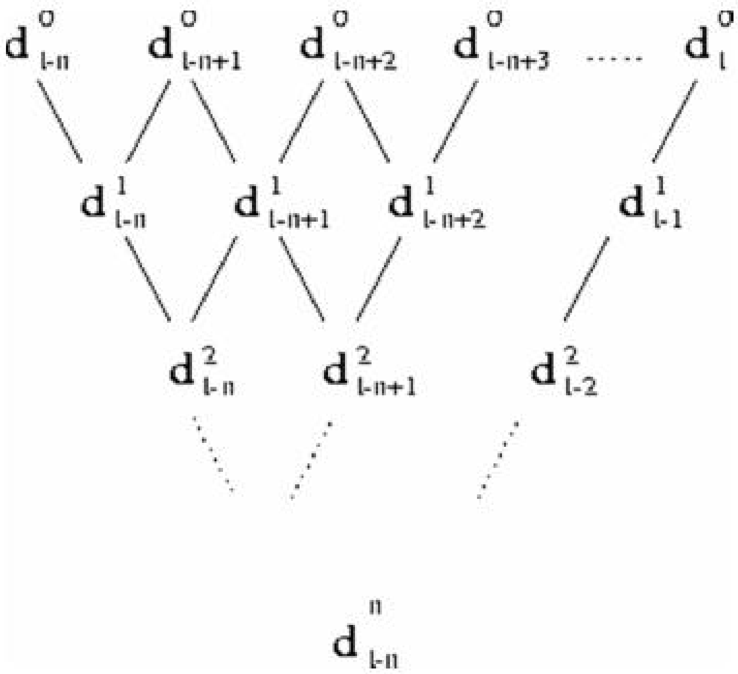
$$d_i^k(t) := \frac{t - t_{i+k}}{t_{i+n+1} - t_{i+k}} \cdot d_{i+1}^{k-1} + \left(1 - \frac{t - t_{i+k}}{t_{i+n+1} - t_{i+k}}\right) \cdot d_i^{k-1} \qquad i = l-n, \ldots, l-k$$

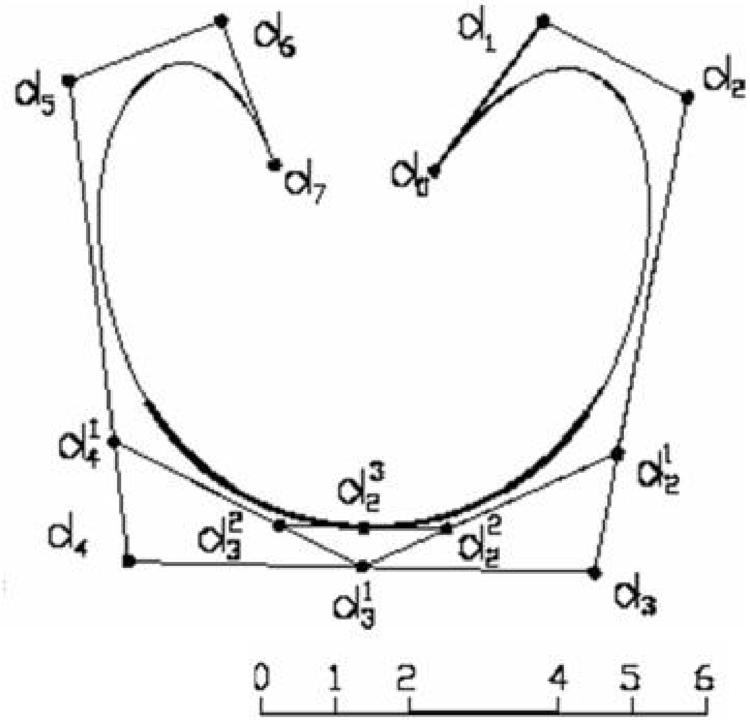
Man bildet somit Konvexkombinationen zweier Punkte und landet auf dem entsprechenden Segment des Kontrollpolygons. Dabei verkleinert sich pro Iterationsschritt die Anzahl der Punkte um Eins.

Zum Schluss gilt:

$$d_{l-n}^n(t) = p(t)$$

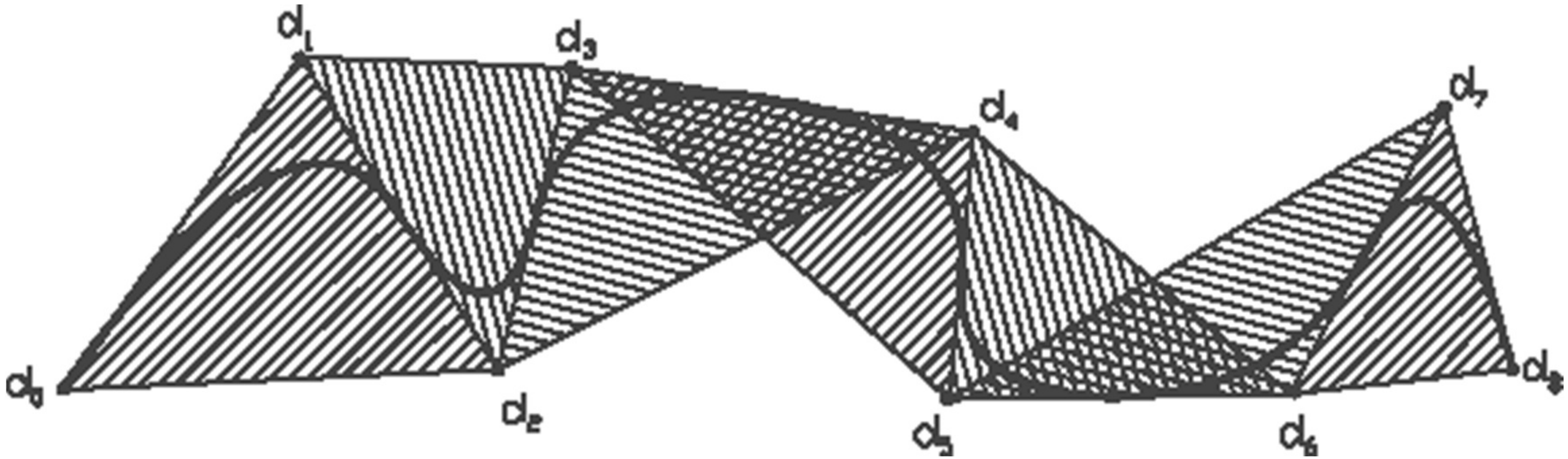
Der Algorithmus von de Boor ist eine Verallgemeinerung des Algorithmus von de Casteljau für B-Splines.


d o l-n
d o l-n+1
d o l-n+2
d o l-n+3
......
d o l
d 1 l-n
d 1 l-n+1
d 1 l-n+2
d 1 l-1
d 2 l-n
d 2 l-n+1
d 2 l-2
d n l-n

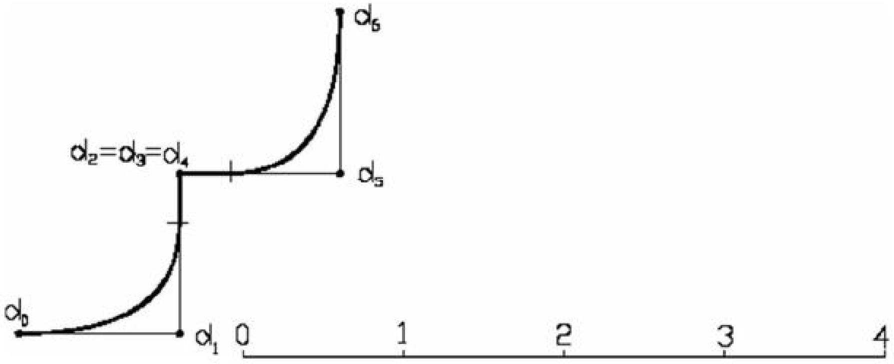

d5
d6
d1
d2
d7
d0
d4 1
d2 3
d2 1
d3 2
d2 2
d4
d3 1
d3
0
1
2
4
5
6

Eigenschaften:

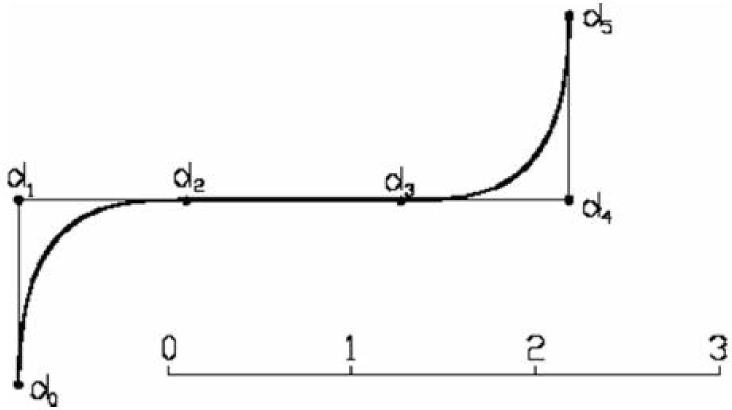
- Für $t_l \leq t \leq t_{l+1}$ liegt $p(t)$ in der konvexen Hülle der $n+1$ Kontrollpunkte $d_{l-n}, \ldots, d_l$



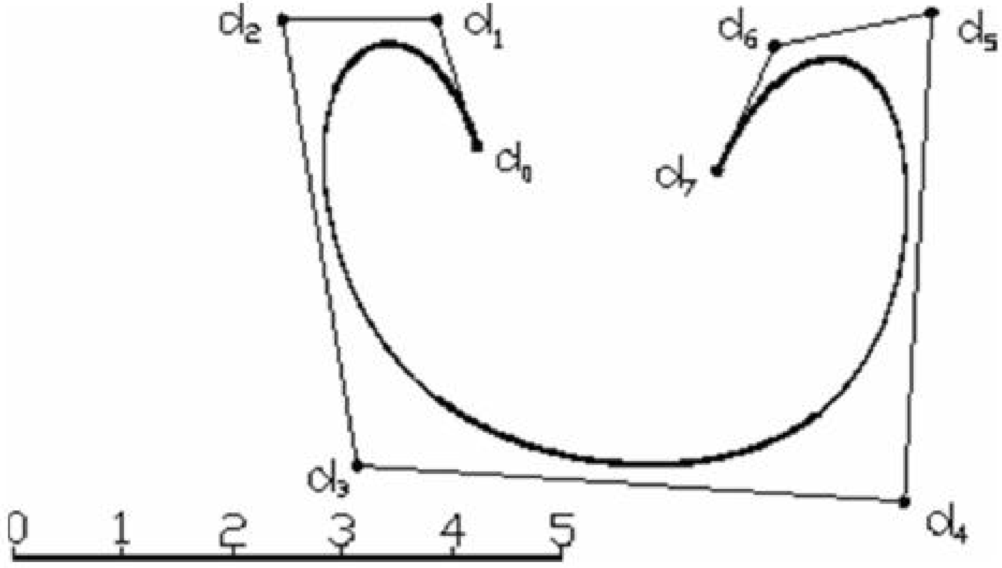
- Fallen n Kontrollpunkte $d_{l-n+1} = \cdots = d_l = d$ zusammen, so gilt $p(t_{l+1}) = d$, d.h. die Kurve verläuft durch den n-fachen Kontrollpunkt



- Liegen $n+1$ Kontrollpunkte $d_{l-n} = \cdots = d_l$ auf der Geraden L, so gilt $p(t) \in L$ für $t_l \leq t \leq t_{l+1}$, d.h. die Kurve hat mit der Geraden L ein gemeinsames Stück



- Fallen n Knoten $d_{l+1} = \cdots = d_{l+n} = t$ zusammen, so gilt $p(t) = d_l$ ein Kontrollpunkt und die Kurve ist dort tangential an das Kontrollpolygon. Insbesondere verläuft die Kurve bei $n+1$-fachen Anfangs- und Endknoten durch die Endpunkte des Polygons und ist dort tangential an das Kontrollpolygon



Analog zur Graderhöhung von Bézierkurven können bei B-Splines Knoten einfügt werden:

Gegeben: B-Splinekurve $p(t) = \sum_{i=0}^{m} d_i N_i^n(t)$ über dem Knotenvektor $T = (t_0, \ldots, t_{m+1} = \cdots = t_{m+n+1})$

Nach dem Einfügen eines Knotens t, z.B. $t_l \leq t < t_{l+1}$ besitzt p die Darstellung

$$p(t) = \sum_{i=0}^{m} d_i^* N_i^n(t)$$

mit dem verfeinerten Knotenvektor $T = (t_0, \ldots t_l, t, t_{l+1}, \ldots, t_{m+n+1})$. Dabei sind die neuen Kontrollpunkte wie folgt definiert:

$$d_i^* := a_i \cdot d_i + (1 - a_i) \cdot d_{i-1}$$

mit

$$a_i = \begin{cases} 1 & \text{, falls } i \leq l - n \\ \frac{t - t_i}{t_{i+n} - t_i} & \text{, falls } l - n + 1 \leq i \leq l \\ 0 & \text{, falls } l + 1 \leq i \end{cases}$$

Ein Vergleich dieses Algorithmus von Böhm mit dem de Boor Algorithmus zeigt

$$d_i^* = d_{i-1}^1(t)$$

d.h. das Einfügen eines Knotens t bis zur Multiplizität n liefert den de Boor-Algorithmus: