Approximationsverfahren für die Kurvendarstellung

(a) Bézier-Kurven

- spezielle Form polynomialer Kurven
- spezifiziert durch n+1 Kontrollpunkte P_0, P_1, ..., P_n
- Kurve läuft nicht durch alle Kontrollpunkte, sondern wird von ihnen beeinflusst!

Der Kontrollpunkt P_i wird durch das sogenannte i-te *Bernstein-Polynom* gewichtet:

$$Q(t) = \sum_{i=0}^{n} P_i \cdot \underbrace{B_{i,n}(t)}_{\text{„Bernstein-Polynome“}}, \quad t \in [0;1]$$

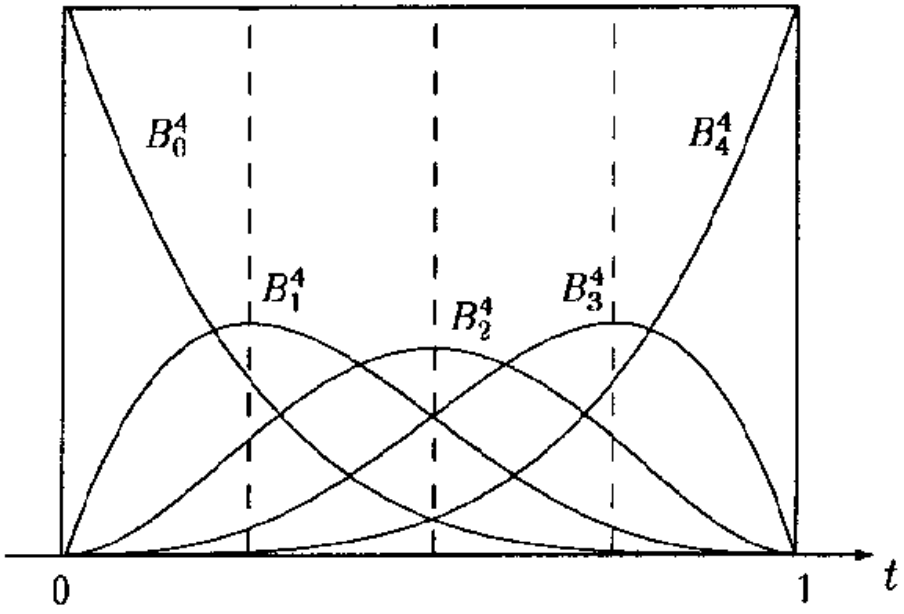
Definition:

$$B_{i,n}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i} \quad i = 0, \ldots, n$$

darin ist der Vorfaktor der bekannte Binomialkoeffizient:

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

Bernsteinpolynome vom Grad $n = 4$ (unterer Index = i):



Eigenschaften:

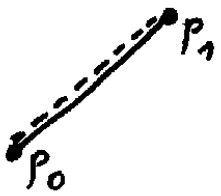
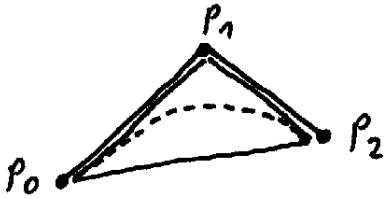
- Es gilt:

$$\sum_{i=0}^{n} B_{i,n}(t) = 1 \qquad \forall t \in [0;1]$$

Beweis:

$$\sum_{i=0}^{n} B_{i,n}(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} \underset{\text{binom. Satz}}{=} (t+(1-t))^n = 1^n = 1.$$

- Die approximierende Kurve $Q(t)$ liegt innerhalb der konvexen Hülle der Kontrollpunkte P_0, ..., P_n

$n = 2$:	$n = 3$:
	

- Für die Sortierung nach t-Potenzen def. man:

$$Q_{i,n} := \binom{n}{i} \sum_{j=0}^{i} \binom{i}{j} (-1)^j P_{i-j} = \binom{n}{i} \sum_{j=0}^{i} \binom{i}{j} (-1)^{i-j} P_j$$

dann gilt:

$$Q(t) = \sum_{i=0}^{n} Q_{i,n} \cdot t^i \qquad (t \in [0;1])$$

Die Berechnung der $Q_{i,n}$ aus den Kontrollpunkten lässt sich in Matrixschreibweise zusammenfassen:

$$\begin{pmatrix} Q_{0,n} \\ Q_{1,n} \\ \vdots \\ Q_{n,n} \end{pmatrix} = M_n \cdot \begin{pmatrix} P_0 \\ P_1 \\ \vdots \\ P_n \end{pmatrix}, \quad (M_n)_{i,j} = (-1)^{i-j} \binom{n}{i} \binom{i}{j}$$

$$\left[\binom{i}{j} = 0 \text{ für } j > i \right]$$

- bei n+1 vorgegebenen Kontrollpunkten ist $Q(t)$ Polynom n-ten Grades mit $Q(0) = P_0$ und $Q(1) = P_n$, also Übereinstimmung mit den Kontrollpunkten am Rand

- die Kante P_0P_1 und die Kante $P_{n-1}P_n$ sind Tangenten an Q im Punkt $Q(0)$ bzw. $Q(1)$ und zeigen in Richtung der Differenz zum Nachbarpunkt:

  $$\frac{dQ}{dt} = \sum_{i=1}^{n} Q_{i,n} \cdot i \cdot t^{i-1}$$

  für $t = 0$:

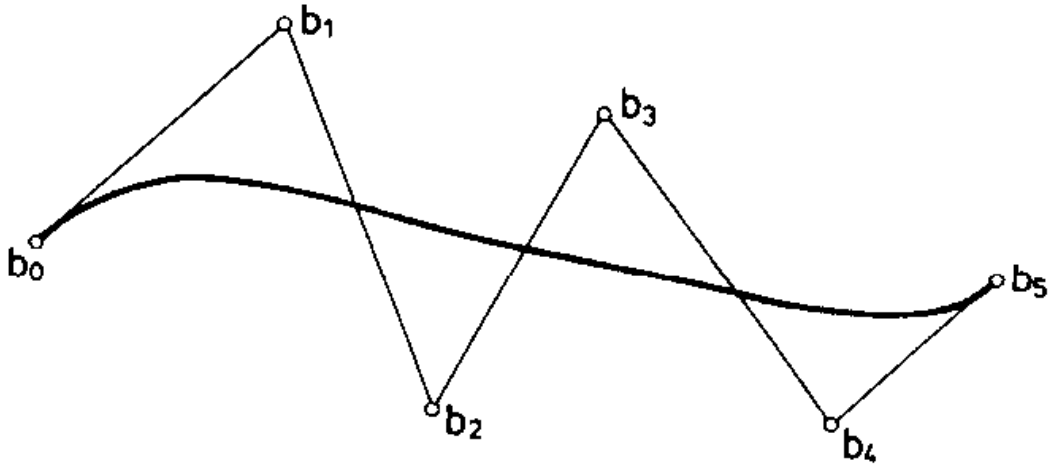
  $$\left.\frac{dQ}{dt}\right|_{t=0} = Q_{1,n} = n \cdot (-P_0 + P_1)$$

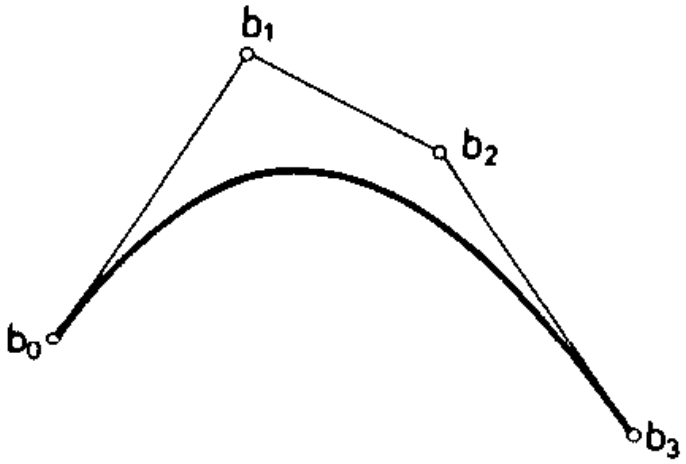
  für $t = 1$ entsprechend.

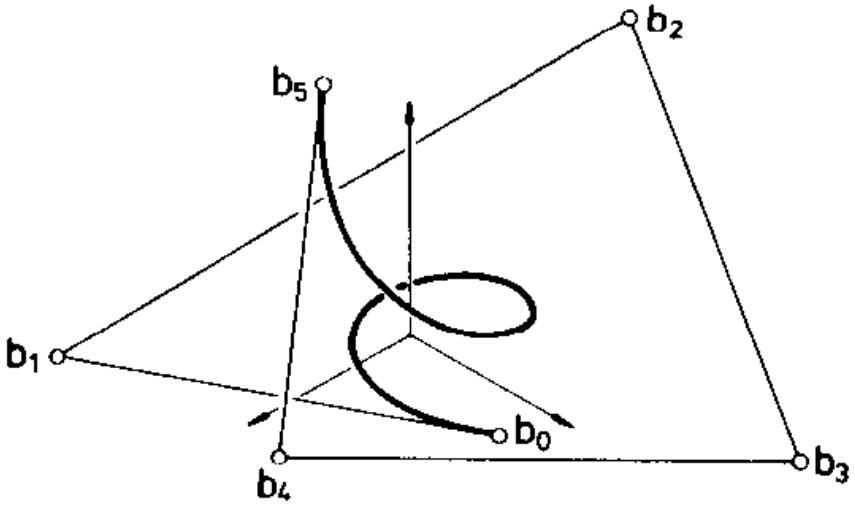
- das Verfahren hat globalen Charakter: Einfügen von zusätzlichen Kontrollpunkten bedingt höheren Grad des approximierenden Polynoms und engere Anlehnung an das Polygon der Kontrollpunkte ("charakteristisches Polygon").

- zum Aneinanderfügen zweier Bézier-Kurven mit stetig differenzierbarem Übergang müssen die entsprechenden Endstücke der charakteristischen Polygone kollinear sein.

Beispiele verschiedener Bézier-Kurven mit ihren charakteristischen Polygonen (Kontrollpunkten):





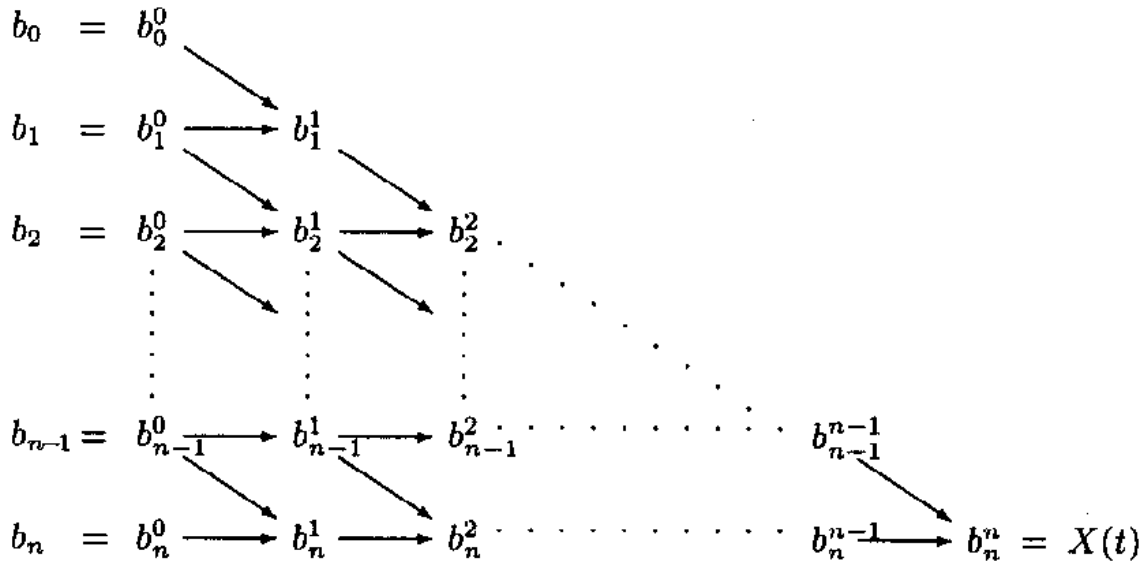


Berechnung von Punkten auf der Bézier-Kurve:
iterativ in endlich vielen Schritten möglich

→ *Algorithmus von de Casteljau*

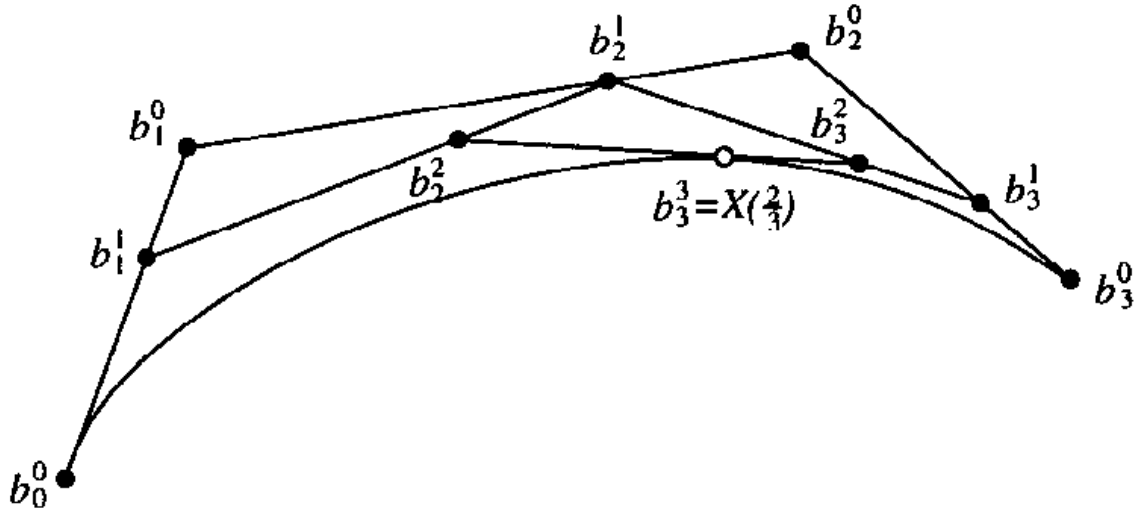
Initialisierung:
$b_i^0 = P_i$ für $i = 0, ..., n$
Iteration:
$k = 1, ..., n$:
  $i = k, ..., n$:
    $b_i^k = (1 - t)\, b_{i-1}^{k-1} + t\, b_i^{k-1}$
Ergebnis:
$Q(t) = b_n^n$.
(hochgestellte Zahl hier kein Exponent, sondern Index.)

Veranschaulichung des Rechenschemas
(hier $b_i = P_i$, $X(t) = Q(t)$):



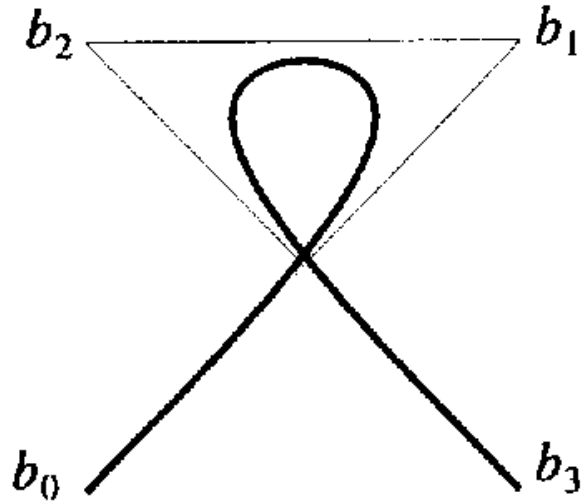
Entsprechende geometrische Konstruktion eines Punktes zum Parameterwert t auf der Kurve:
Teilung der Kanten des charakteristischen Polygons im Verhältnis $t : (1–t)$, Verbinden der Teilungspunkte, Iteration

Beispiel: Konstruktion nach de Casteljau für $n = 3$ und $t = 2/3$:



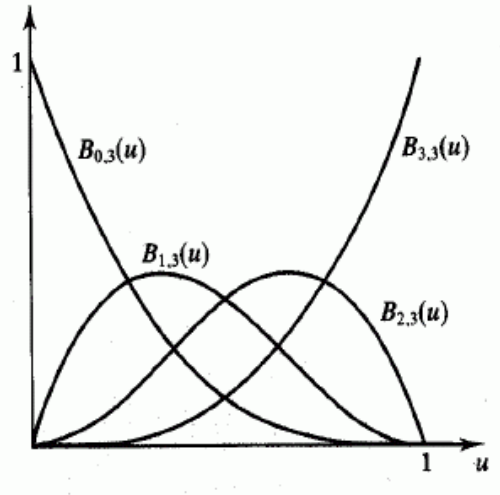
weitere Eigenschaften der Bézier-Kurven:

- monotoner Datensatz führt zu monotoner Kurve
- konvexer Datensatz führt zu konvexer Kurve
- Rotations- und Translationsinvarianz
- lokale Änderungen in den Kontrollpunkten wirken sich zwar global aus, **aber** Einfluss ist nur in der Umgebung des geänderten Punktes signifikant
- es können Doppelpunkte auftreten:



Bézierkurve vom Grad 4 mit einem Doppelpunkt

häufig gebraucht: Spezialfall *n* = 3, *kubische Bézier-Kurve*



$$\mathbf{Q}(\mathrm{u}) = \sum_{\mathrm{i}=0}^{3} \mathbf{p}_\mathrm{i} B_{i,3}(u) \quad mit$$

$$B_{0,3}(u) = (1-u)^3$$

$$B_{1,3}(u) = 3u\ (1-u)^2$$

$$B_{2,3}(u) = 3u^2(1-u)$$

$$B_{3,3}(u) = u^3$$

- Die Bezier Basisfunktionen
- Beobachtungen: p_0 hat dominierenden Einfluß für u < 0,1
- Mit wachsendem u nimmt der Einfluß der anderen Kontrollpunkte zu

Matrix-Notation der kubischen Bézier-Kurve:

$$\mathbf{Q}(\mathrm{u}) = [\mathrm{x(u)\ y(u)\ z(u)}] = \mathbf{U}\mathbf{M}_\mathbf{B}\mathbf{P}_\mathbf{c} = [\mathrm{u^3\ u^2\ u\ 1}] \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \\ \mathbf{p}_4 \end{bmatrix}$$

Multipliziert man die Basismatrix $\mathbf{M}_\mathrm{B}$ mit dem Parametervektor **U** so errechnen sich die Basisfunktionen *(Blending Functions)*:

$$(B_{0,3}, B_{1,3}, B_{2,3}, B_{3,3}) = \mathbf{U}\mathbf{M}_\mathbf{B}$$

Zusammenfassung zu Bézier-Kurven:

- historisch bedeutendes Repräsentationsmodell
- schnelle + einfache Berechnung
- noch oft in interaktiver Anwendungssoftware genutzt: Kontrolle der Endpunkte und der Tangenten, ggf. interaktive Beeinflussung der Kontrollpunkte
- Verallgemeinerung auf Flächen: möglich, aber durch zu wenige Freiheitsgrade beim Modellieren nur beschränkt einsetzbar
- Verwendung von Bézier-Flächen für das Rendering

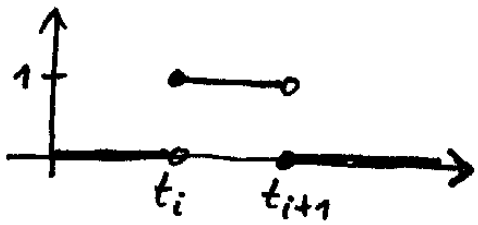
(b) B-Splines

Es sei (t_0, t_1, ..., t_n) ein "Knotenvektor" (Vektor von Kontroll-stellen) mit $t_i \leq t_j$ für $i < j$.
Die *(nicht-uniformen) normalisierten B-Spline-Basisfunktionen* $N_{i,k}$ der Ordnung k über dem Intervall $[t_0, t_n]$ sind def. durch

$$N_{i,1}(t) = \begin{cases} 1 & t_i \leq t < t_{i+1} \\ 0 & sonst \end{cases}$$

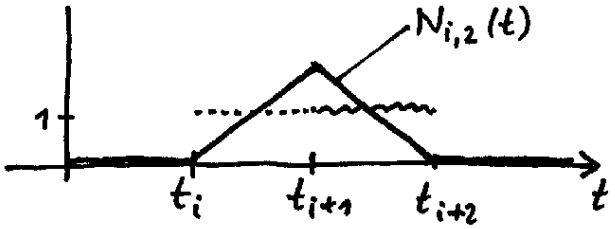
$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} \cdot N_{i,k-1}(t) + \frac{-t + t_{i+k}}{t_{i+k} - t_{i+1}} \cdot N_{i+1,k-1}(t)$$

Verlauf von $N_{i,1}$:

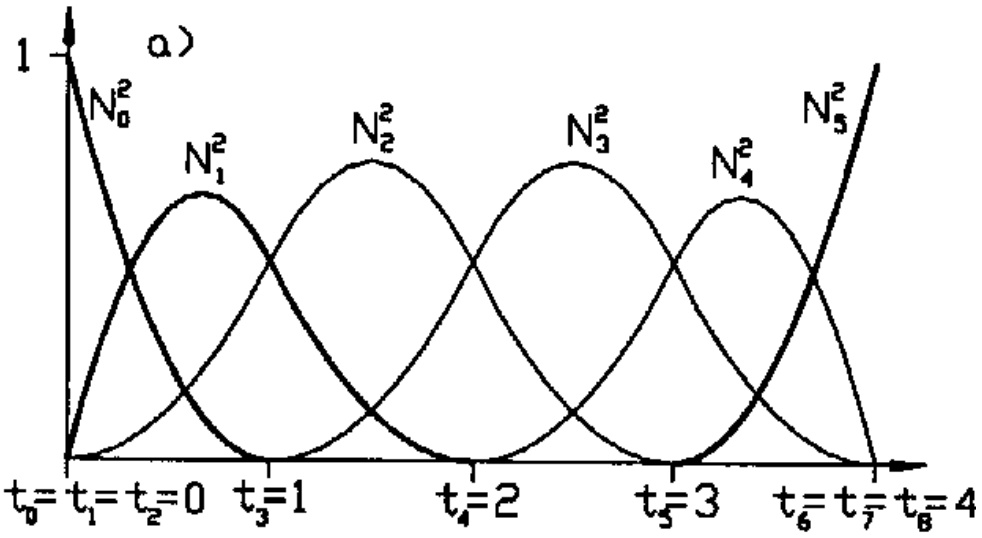


Bsp. $k=2$:

$$N_{i,2}(t) = \frac{t-t_i}{t_{i+1}-t_i} N_{i,1}(t) + \frac{t_{i+2}-t}{t_{i+2}-t_{i+1}} N_{i+1,1}(t)$$



Beispiel $k = 3$, Knotenvektor (0, 0, 0, 1, 2, 3, 4, 4, 4),
Verlauf der B-Spline-Basisfunktionen $N_{i,3}(t)$, $i = 0, ..., 5$:

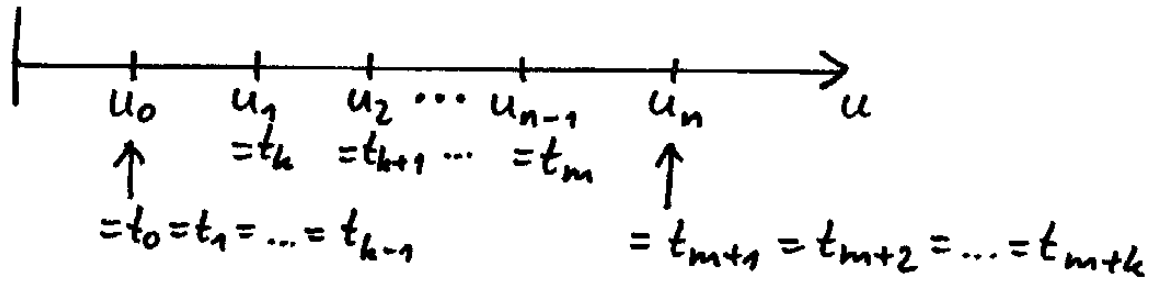


Allgemein ist $N_{i,k}(t) = 0$ für $t \notin [t_i, t_{i+k}]$.
Vereinbarung:

$$\frac{t - t_i}{t_{i+k-1} - t_i} := 0 \quad \text{für } t_{i+k-1} = t_i$$

$$\frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} := 0 \quad \text{für } t_{i+k} = t_{i+1}$$

Das Parameterintervall für die Kurve, $[u_0, u_n]$, wird zerlegt in $n = m–k+2$ Teilintervalle $[u_i, u_{i+1})$, i = 0; 1; ...; n–1.
Def. des Knotenvektors (t_0, t_1, ..., t_{k+m}):
$t_i = u_0$ für i = 0; 1; ...; k–2,
$t_{i+k-1} = u_i$ für i = 0; 1; ...; m–k+2,
$t_{i+m+2} = u_n$ für i = 0; 1; ...; k–2.



- Spezialfall der *uniformen* B-Splines: wähle uniforme (d.h. äquidistante) Teilung von $[u_0, u_n)$, also $u_0 = 0$, $u_1 = 1$, $u_2 = 2$, ..., $u_n = n$.

- Spezialfall $m = k–1$, $t_0 = 0$, $t_1 = 1$:

Knotenvektor = (0, 0, 0, ..., 0, 1, 1, 1, ..., 1) (k Nullen, k Einsen) liefert "degenerierte B-Spline-Basis":

$$N_{i,k}(t) = \binom{k-1}{i} t^i (1-t)^{k-1-i} = B_{i,k-1}(t)$$

(Bernstein-Polynom somit als Spezialfall).

- Für die B-Spline-Basisfunktionen gilt (im allgemeinen Fall):

$$\sum_{i=0}^{m} N_{i,k}(t) = 1 \quad \text{für } t \in [t_0; t_n).$$

(wichtig für die Konvexe-Hüllen-Eigenschaft, s.u.).
(Beweis durch Induktion über k.)

Approximation einer gegebenen Liste von Kontrollpunkten P_i, i= 0, ..., m, durch eine B-Spline-Kurve der Ordnung k:

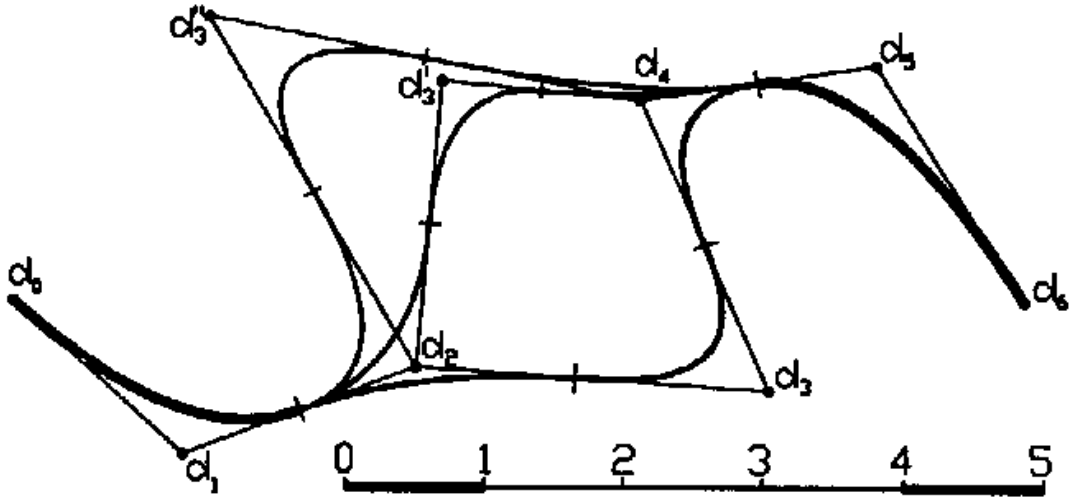
$$Q(t) = \sum_{i=0}^{m} P_i \cdot N_{i,k}(t), \quad t \in [t_0; t_n).$$

An den Rändern: $Q(t_0) = P_0$, $Q(t_n) = P_m$,
dazwischen werden die P_i i. allg. lediglich approximiert.

Aber: wenn man $P_i = P_{i+1} = ... = P_{i+k-1}$ setzt, muss $Q(t)$ durch P_i verlaufen $\Rightarrow$ Durchlaufen eines Punktes lässt sich erzwingen.

Eigenschaften der B-Spline-Kurven:

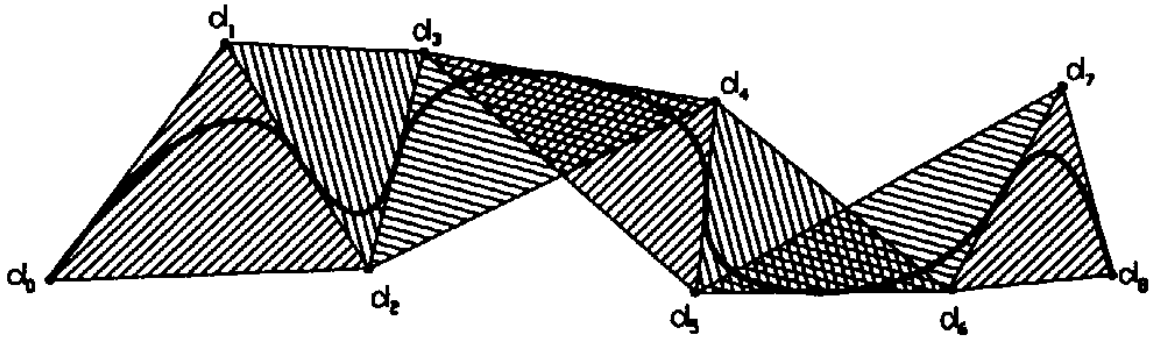
- wegen $N_{i,k}(t) = 0$ für $t \notin [t_i, t_{i+k})$ ist das Verfahren lokal.



lokale Wirkung der Kontrollpunkte. k = 3, Knotenvektor (0, 0, 0, 1, 2, 3, 4, 5, 5, 5); durch Verschieben von d_3 nach d_3' werden lediglich die Kurvensegmente mit Parameterwerten zwischen 1 und 4 beeinflusst.

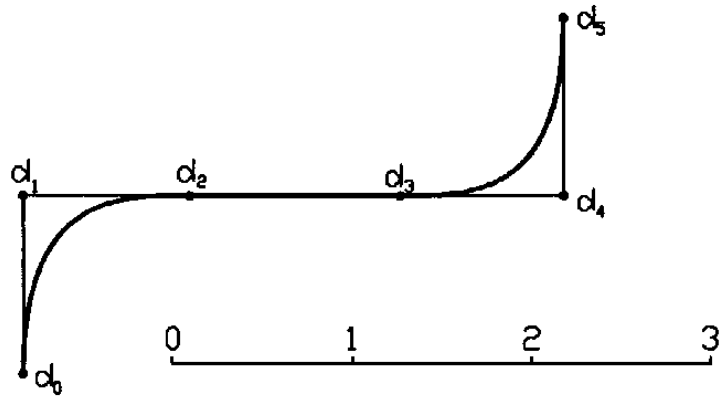
- Da sich die Basisfunktionen zu 1 summieren (s.o.), liegt die approximierende Kurve innerhalb der konvexen Hülle der Kontrollpunkte.

- Genauer: $Q(t)$ verläuft innerhalb der konvexen Hülle von je k aufeinanderfolgenden Kontrollpunkten:



$\Rightarrow$ für kleines k legt sich $Q(t)$ eng an das Polygon an,
für k = 2 ist $Q(t)$ *identisch* mit dem charakteristischen Polygon.

- wenn k+1 aufeinanderfolgende Kontrollpunkte kollinear sind, enthält die Kurve eine der Polygonkanten:



k = 3, 4 Punkte d_1, ..., d_4 kollinear: Erzwingung eines Geradenstücks

- es gibt ein iteratives Berechnungsverfahren als Verallgemeinerung des de Casteljau-Algorithmus:

  Algorithmus von de Boor
  (siehe Encarnação et al. 1996)

Für einige Anwendungen (insbes. bei Verwendung für Flächen) bei B-Splines immer noch nicht genug Freiheitsgrade

Idee: Def. der Kurve in homogenen Koordinaten
$\Rightarrow$ dadurch zusätzliche Freiheitsgrade

($x(t)$, $y(t)$, $z(t)$, $w(t)$)

Bei Normalisierung (Division durch $w(t)$) entstehen gebrochenrationale Funktionen.
Somit:

<u>(c) NURBS</u>
= Nicht-uniforme rationale B-Splines

NURBS

- gebräuchlichste Form: $x = \frac{x(t)}{w(t)}, y = \frac{y(t)}{w(t)}, z = \frac{z(t)}{w(t)}$
  (in homogenen Koordinaten: $(x(t), y(t), z(t), w(t))$)
- invariant bei Rotation, Skalierung, Translation und Projektion
- Nur Kontrollpunkte müssen projiziert werden!
  (letzteres gilt nicht für nichtrationale Kurven)

wie bei den einfachen B-Splines wird die Kurve stückweise zusammengesetzt:

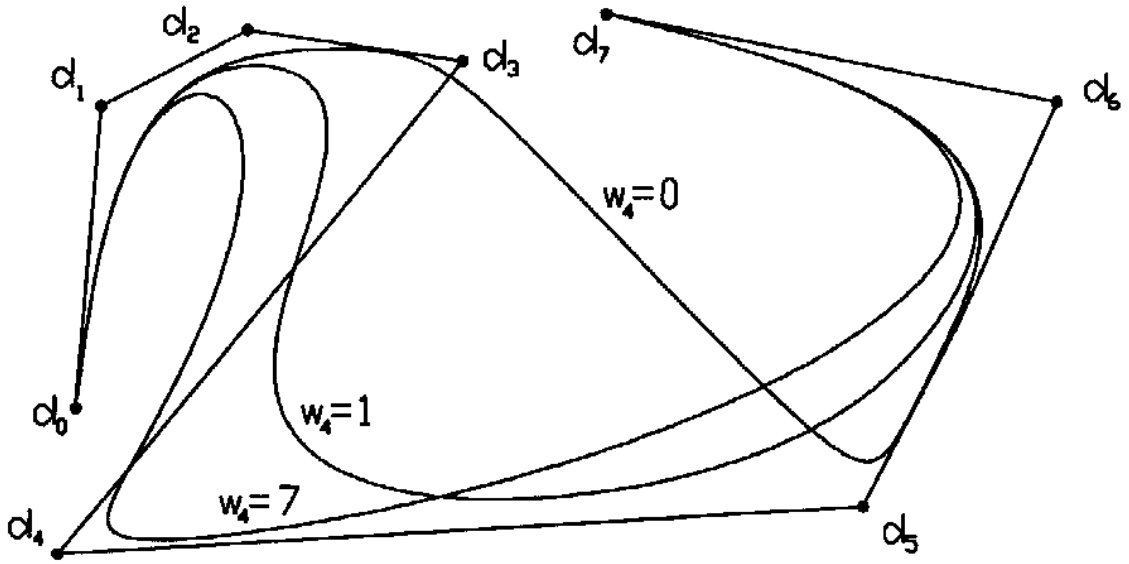
$$\mathbf{p}(u) = \frac{\sum_{i=0}^{n} \mathbf{p}_i w_i N_{i,k}(u)}{\sum_{i=0}^{n} w_i N_{i,k}(u)}$$

$$= \sum_{i=0}^{n} \mathbf{p}_i w_i R_{i,k}(u) \text{ mit } R_{i,k}(u) = \frac{N_{i,k}(u) w_i}{\sum_{j=0}^{n} N_{j,k}(u) w_j}$$

Eigenschaften von Rationalen B-Splines

- Haben die gleichen analytischen und geometrischen Eigenschaften wie Splines
- w_i =1 für alle i dann gilt
  $R_{i,k}(u) = N_{i,k}(u)$
- die mit jedem Kontrollpunkt assoziierten Gewichte w_i wirken als zusätzliche Formkontrollparameter. Wirken auch lokal.
- $w_i > 1$ ➔ Kurve nähert sich dem Kontrollpkt.
- $w_i < 1$ ➔ Kurve entfernt sich von dem Kontrollpkt

Wirkung der Gewichte auf NURBS: eine NURBS-Kurve mit $k = 5$, Knotenvektor (0, 0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4, 4), bei der das Gewicht w_4 variiert wurde:



Vorteile von NURBS

- Invarianz gegenüber Rotation, Skalierung, Translation und perspektivischer Transformation (d.h. nur die Kontroll-punkte müssen transformiert werden und nicht jeder Punkt der Kurve)
- NURBS können Kegelschnitte exakt beschreiben
- Gegenüber B-Splines durch die Gewichte ergänzende Editiermöglichkeiten