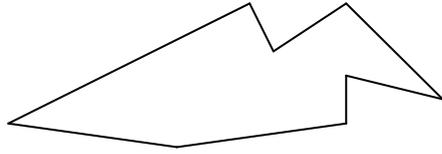


6. Modelle für Kurven

erster Ansatz: Kurve durch Polygon approximieren



Polygone sind stückweise *lineare* Approximationen für Kurven bzw. Flächen

Nachteile:

- hohe Zahl von Eckpunkten für genaue Repräsentation erforderlich
- interaktive Manipulation schwierig

Suche nach kompakteren und einfacher manipulierbaren Repräsentationen

⇒ "Freiformflächen", -kurven

insbesondere:

stückweise *polynomiale* Kurven bzw. Flächen

Geschichtlicher Hintergrund:

- ab ca. 1958 Anwendungen von Freiformflächen im CAD-Bereich, insbes. im Automobil-Karosseriebau
- **Pierre Bézier** (Renault): parametrische Repräsentation auf der Basis von Bernstein-Polynomen: System UNISURF
- **P. de Casteljaou** (Citroen)

Exkurs:

Welche Repräsentationen von Kurven und Flächen gibt es?

Formen der Darstellung von Kurven oder Flächen:

- **explizit**
- **implizit**
- **Parametrisch**

Kurze Diskussion am Beispiel der Kurven:

Explizite Darstellung

◆ $y = f(x)$

◆ Bsp.: $y = \sqrt{x^3} - \sqrt{x}$

◆ Probleme:

- Für ein x darf es nur einen y Wert geben (Probleme z.B. ist ein Kreis nicht geschlossen darstellbar)
- Beschreibung nicht invariant gegenüber Rotationen
- Keine Kurven mit (echt) vertikalen Tangenten möglich (impliziert unendliche Steigung)

(Krömker)

Implizite Darstellung

◆ $f(x,y) = 0$

◆ Bsp.: $x^3 - 2x^2 + x - y^2 = 0$

◆ Probleme:

- Gleichung kann mehr Lösungen als gewollt haben, was zur Notwendigkeit von Randbedingungen führt
- Richtung der Tangente ist schwer zu ermitteln

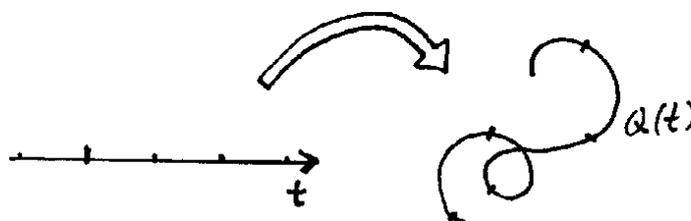
Parametrische Darstellung

◆ $Q(t) = (x(t), y(t))$

◆ Bsp.: $Q(t) = (t^2, t^3 - t)$

◆ Vorteile:

- Keine Mehrdeutigkeiten
 - Geometrische Steigungen (potentiell unendlich) werden durch Tangentenvektoren (niemals unendlich) ersetzt
 - Invarianz gegenüber Rotationen
- ◆ Wir verwenden daher im folgenden nur die **parametrische Darstellung**



Parametrische Kurven sind i.d.R.

- (ganz rationale) Polynome
- gebrochen rationale Polynome

n-ten Grades (n: höchster auftretender Exponent),
z.B.

$$Q(u) = p_0 + p_1u + p_2u^2 + \dots + p_ku^k$$

In der CG werden ganz überwiegend kubische
(also k=3) Repräsentationen genutzt:

Definition einer 3D-Kurve durch stückweise Spezifikation dreier
kubischer Polynome $x(t)$, $y(t)$, $z(t)$ mit dem Parameter t .

- Definition eines Kurvensegmentes $Q(t) = [x(t) \ y(t) \ z(t)]$ durch
($t \in [0, 1]$):

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$$

- andere Repräsentation: $Q(t) = [x(t) \ y(t) \ z(t)] = T \cdot C$ mit:

$$C = \begin{pmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{pmatrix} \quad \text{und} \quad T = [t^3 \ t^2 \ t \ 1]$$

Typen der Darstellung von Kurven (und Flächen)

- ◆ Exakte Darstellung
 - Jeder Punkt ist durch eine Formel definiert
 - Problem: Formel ist meist nicht bekannt oder zu komplex
- ◆ Interpolatorische Darstellung
 - Kurve ist durch Stützstellenbedingungen beschrieben
 - Kurve ist an den Stützstellen determiniert
- ◆ Approximative Darstellung
 - Kurve ist durch Stützstellenbedingungen beschrieben
 - Kurve ist an den Stützstellen nicht determiniert

Polynom-**Interpolation**:

- Gesucht sind die Koeffizienten eines Polynoms $P(t)$ derart, daß $P(t_i) = P_i$ für alle Stützpunkte P_i gilt
- Für $n+1$ paarweise verschiedene Stützpunkte gibt es genau ein Polynom vom Grad n , das die obige Bedingung erfüllt
- Nachteil: Berechnung der Koeffizienten ist aufwendig
- Nachteil: Änderung eines Stützpunktes bedingt Neuberechnung aller Koeffizienten

Durchführung der Polynom-Interpolation:

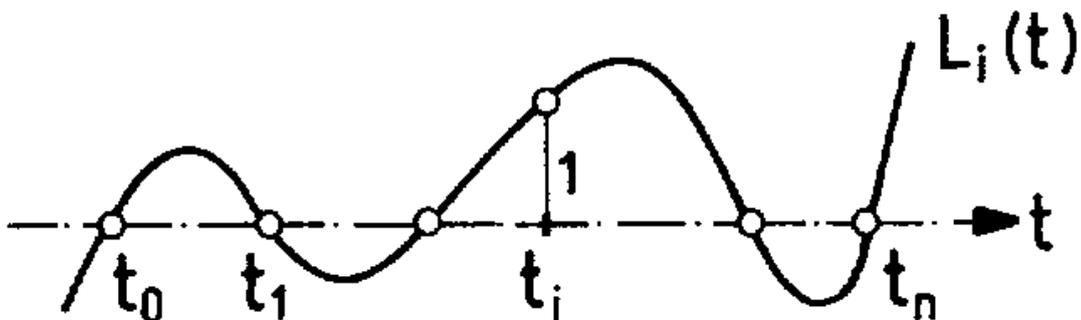
Eine mittels der Stützpunkte gewichtete Summe von sog. *Lagrange-Polynomen* liefert genau das Polynom, das durch alle P_i verläuft (**Lagrange-Interpolation**):

$$Q(t) = \sum_{i=0}^n P_i \cdot L_{n,i}(t)$$

(P_i : gegebene Punkte, t_i : entspr. Stützstellen, $i = 0, 1, \dots, n$)

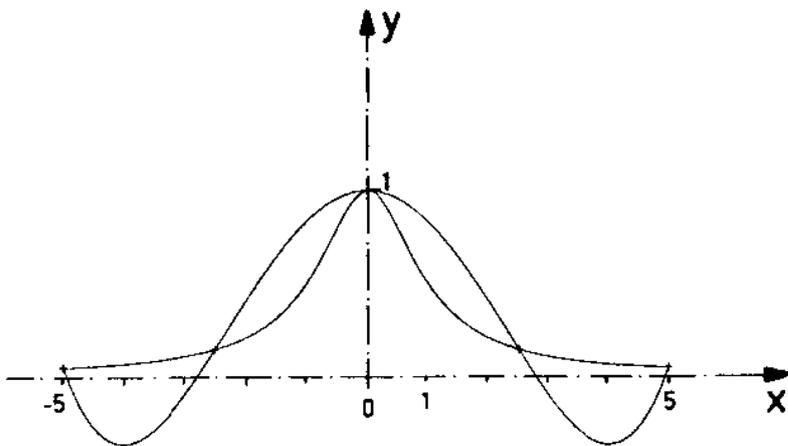
$$\text{mit } L_{n,i}(t) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{t - t_j}{t_i - t_j}.$$

Die Lagrange-Polynome $L_{n,i}(t)$ haben die Eigenschaft, an der Stützstelle t_i den Wert 1 zu liefern und an allen anderen Stützstellen 0:

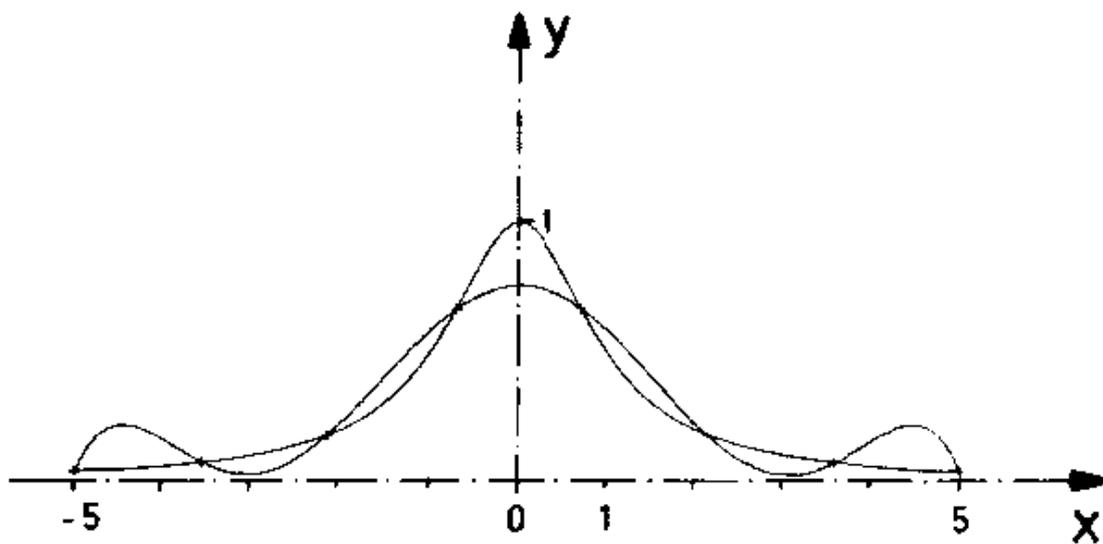


- bei Hinzufügen eines weiteren Stützpunktes müssen alle Koeffizienten neu berechnet werden
- "globales Verfahren": bei einer lokalen Änderung, z.B. eines einzelnen Stützpunktes, kann sich die Gestalt der ganzen Kurve ändern
- genereller Nachteil der Interpolation mit Polynomen höheren Grades: **Oszillationen**

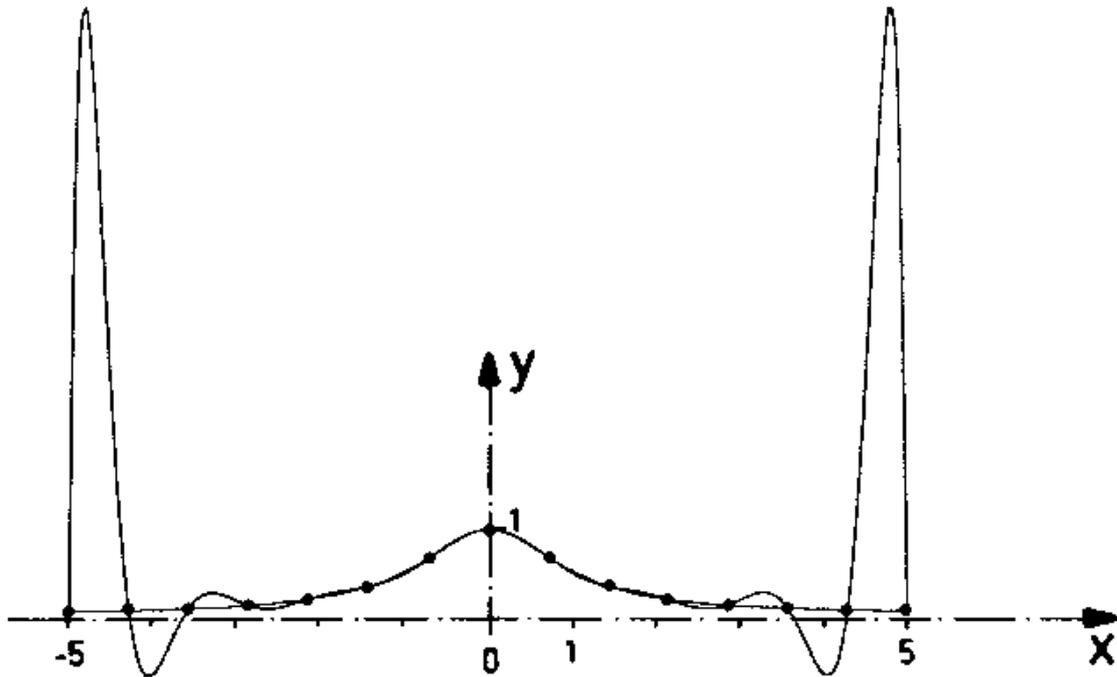
Beispiel: Interpolation des Funktionsgraphen von $y = 1/(1+x^2)$ zwischen -5 und 5



Polynom 4. Grades



Polynom 7. Grades



Polynom 14. Grades

- weiterer Nachteil: bei der Interpolation durch ein Polynom n -ten Grades weicht die Tangente stellenweise stark von der Tangente der darzustellenden Kurve ab

Abhilfe für den letzten Punkt:

man gibt neben den Stützpunkten auch die ersten m Ableitungen an den Stützpunkten vor und wählt das Polynom so, dass auch diese übereinstimmen.

für einzelnes Kurvensegment:

gegeben: $P_0^{(k)}, P_1^{(k)}$ ($k = 0; 1; \dots; m$) (Werte der Ableitungen an den beiden Stützstellen).

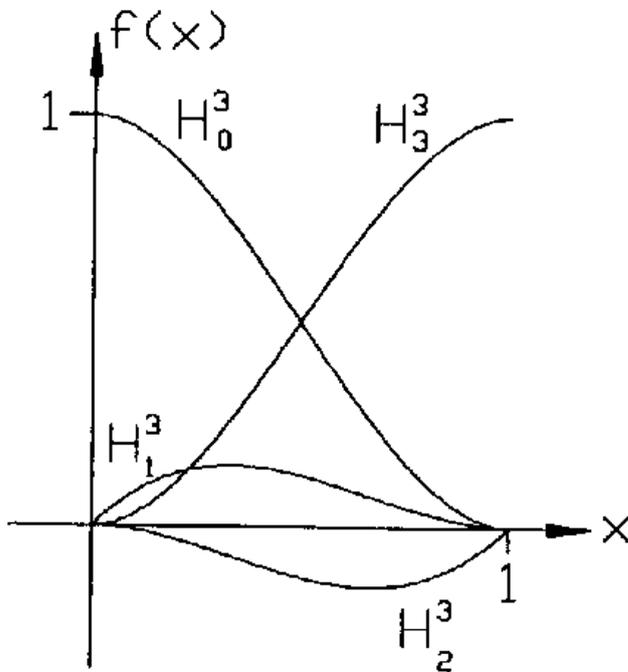
Der Parameterbereich sei $[t_0, t_1]$ ($t_0 < t_1$).

Ansatz für dieses Segment:

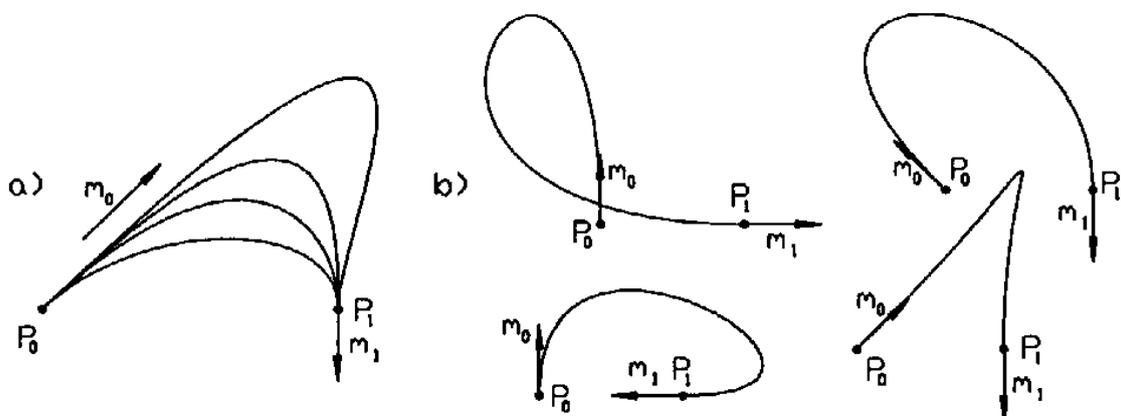
$$Q(t) = \sum_{i=0}^1 \sum_{k=0}^m P_i^{(k)} \cdot H_{i,k}(t), \quad t \in [t_0, t_1]$$

darin sind die $H_{i,k}(t)$ die sog. *Hermite-Interpolationspolynome* vom Grad $2m+1$.

Das folgende Diagramm zeigt die Hermite-Polynome vom Grad 3 über dem Intervall $[0; 1]$ (hier mit 0; 1; 2; 3 indiziert, entspr. $H_{0,0}$, $H_{0,1}$, $H_{1,1}$, $H_{1,0}$):



Durch die Vorgabe der Ableitungen, insbes. der ersten Ableitung (Tangentenvektor), kann der Kurvenverlauf besser gesteuert werden als allein durch die Positionen an den Stützstellen:



- a) Variation der Länge des Tangentenvektors im Anfangspunkt des Kurvensegments
- b) die Richtungen der beiden Tangenten in den Endpunkten werden variiert

zur Vermeidung von Oszillationen und globalen Auswirkungen bei lokalen Änderungen:
Zerlegung in Teilintervalle mit *verschiedenen*, nur begrenzt voneinander abhängigen Interpolationskurven

→ Idee der *Spline-Kurven*

Splines

spline (engl.): dünner Stab, von Schiffsbauern benutzt, um die richtige Form der "stringer" zu konstruieren (Planken, die in Schiffs-Längsrichtung quer zu den Spanten angebracht werden und die Außenwand des Rumpfes bilden)

Entspr. deutsches Wort: "Straklatte", daher in der älteren dt. Lit. für Splinefunktionen z.T. auch "Strakfunktionen".

Seien $t_0, \dots, t_n \in \mathbb{R}$, $t_0 < t_1 < \dots < t_n$, Funktion S def. auf $[t_0, t_n]$.

Def.:

S heißt *Splinefunktion* der Ordnung k (oder vom Grad $k-1$), wenn gilt:

(1) S ist in jedem Teilintervall $[t_r, t_{r+1}]$ ein Polynom vom Grad $k-1$.

(2) S und die Ableitungen $S^{(m)}$, $m = 1; \dots; k-2$, sind stetig auf $[t_0, t_n]$.

S heißt *Sub-Splinefunktion*, wenn (1) gilt, S stetig ist, (2) aber nicht gilt.

Kubische Splines

$k = 4$, also Polynomsegmente vom Grad 3, zweimal stetig differenzierbar.

Eine kubische Splinefunktion beschreibt näherungsweise die Biegelinie eines dünnen Stabes durch eine vorgegebene Reihe von festen Stützhaken.

(Lösung d. Euler-Lagrange-Gleichung, Variationsrechnung, s. Hoschek & Lasser 1992)

Kubische Splines benutzen auf den Teilintervallen Hermite-Interpolationspolynome vom Grad $3 = 2m+1$ ($m = 1$).

Gegeben: $n+1$ Punkte $P_r^{(0)}$, $r = 0; \dots; n$ ($n > 0$);
Stützstellen $a = t_0 < t_1 < t_2 < \dots < t_n = b$.

Auf dem Teilintervall $[t_r, t_{r+1}]$: Hermite-Interpolation 3. Grades

$$Q_r(t) = \sum_{i=r}^{r+1} \sum_{k=0}^1 P_i^{(k)} \cdot H_{i,k}(t) = (t^3; t^2; t; 1) \cdot M_H \quad (t \in [t_r, t_{r+1}])$$

darin ist M_H die sog. Hermite-Matrix (4,3-Matrix)
(beachte: die $P_i^{(k)}$ werden hier als Zeilenvektoren interpretiert!)
 M_H ist unabhängig von t .

Die Berechnung von M_H erfolgt durch Einsetzen von t_r und t_{r+1} in die obige Gleichung und in die einmal differenzierte Version der obigen Gleichung.

Daraus lassen sich die Hermite-Polynome, und damit die Splinefunktion, berechnen.

Eine explizite Vorgabe der 1. Ableitungen auch in den inneren Stützstellen ist i. allg. nicht sinnvoll

- fordere stattdessen: Stetigkeit der 1. Ableitung (siehe Def. der Spline-Funktionen)

$2n$ Werte bislang unbestimmt (für jedes Segment die 1. Ableitungen in den beiden Endpunkten)

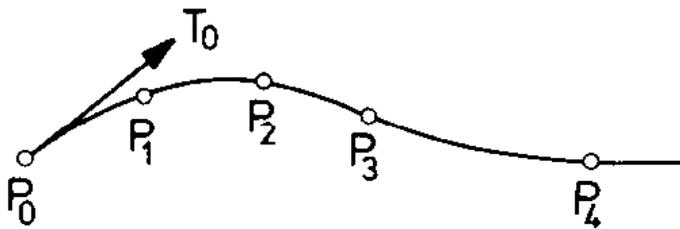
Stetigkeitsforderung liefert $n-1$ weitere Bedingungen

$\Rightarrow n+1$ Koeffizienten immer noch unbestimmt

fordere zusätzlich die Stetigkeit der 2. Ableitung
liefert $n-1$ weitere Bedingungen

⇒ 2 Koeffizienten immer noch unbestimmt

man gibt die 1. Ableitung in den Endpunkten vor



⇒ liefert Bestimmungsgleichungen für die restlichen 2 Koeffizienten

Berechnung der Koeffizienten für die kubischen Polynome in den Teilintervallen läuft auf Lösen eines lin. Gleichungssystems heraus!

Zusammenfassung zur stückweisen Polynom-Interpolation
(gilt entsprechend auch für Approximationsansätze):

- Grad $n = 1$:
Polygonzug, unstetige Steigungen an den Eckpunkten
- Grad $n = 2$:
In 3D können nur planare Kurven erhalten werden
(d.h. Kurve liegt immer in einer Ebene)
- **Grad $n = 3$ (übliche Wahl)**
Die vier Koeffizienten können z.B. durch Startpunkt, Endpunkt, Tangente am Startpunkt, Tangente am Endpunkt gegeben werden
- Grad $n > 3$:
Rechenaufwendig, nur in speziellen Anwendungen
⇒ benutzt

Allgemeine Darstellung für $k=3$

$$\mathbf{Q}(u) = \sum_{i=0}^k \mathbf{p}_i b_i(u)$$

mit $b_i(u)$: Basisfunktionen

\mathbf{p}_i : Kontrollpunkte

$$\mathbf{Q}(u) = [x(u) \ y(u) \ z(u)] = [u^3 \ u^2 \ u \ 1] \cdot \mathbf{M} \cdot \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

Parametervektor Basismatrix Geometrievektor
(Kontrollpunkte)

Blendingfunktionen

- ◆ Der erste Faktor ist der Parametervektor U
- ◆ Der zweite Faktor, die 4×4 Matrix M heißt **Basismatrix**
- ◆ Der dritte Faktor heißt **Geometrievektor G** , die G_i sind geometrische Nebenbedingungen (z.B. Punkte oder Tangentenvektoren)
- ◆ Das Produkt aus U und M ergibt die *Blendingfunktionen* (diese gewichten die geometrischen Nebenbedingungen G_i)

Approximationsverfahren für die Kurvendarstellung

(a) Bézier-Kurven

- spezielle Form polynomialer Kurven
- spezifiziert durch $n+1$ Kontrollpunkte P_0, P_1, \dots, 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:

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

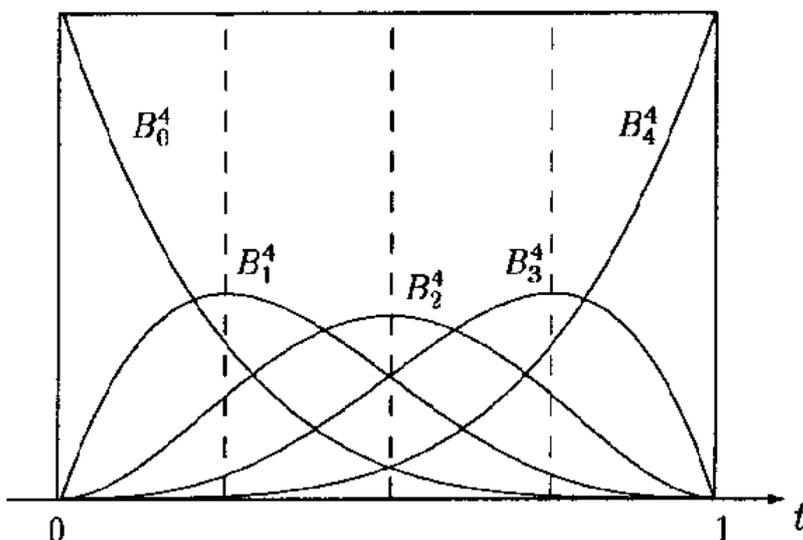
Definition:

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

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:

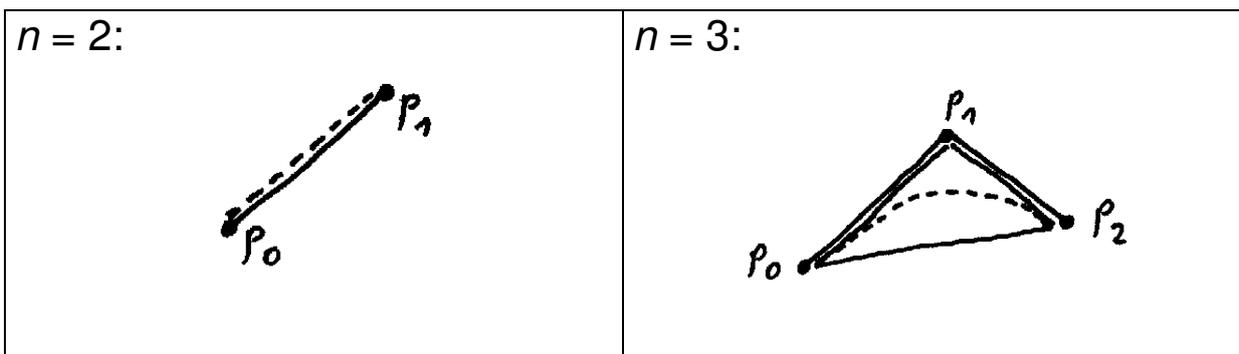
$$\sum_{i=0}^n B_{i,n}(t) = 1 \quad \forall t \in [0;1]$$

Beweis:

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

(↑ nach dem binomischen Satz)

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



- 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:

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

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

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

$$\binom{i}{j} = 0 \text{ für } j > i$$

- 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 $t = 0$:

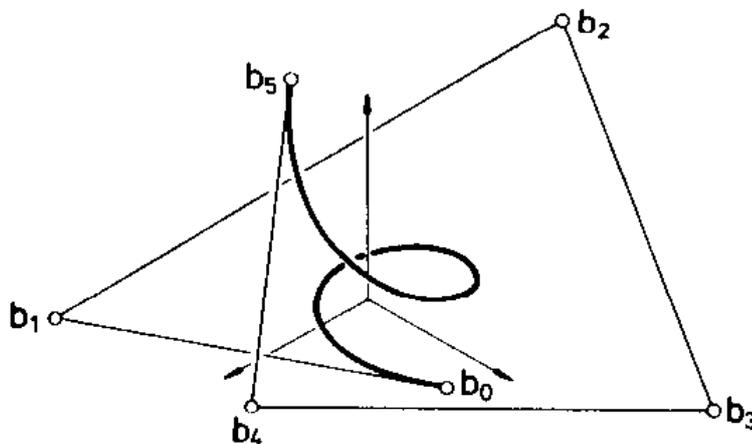
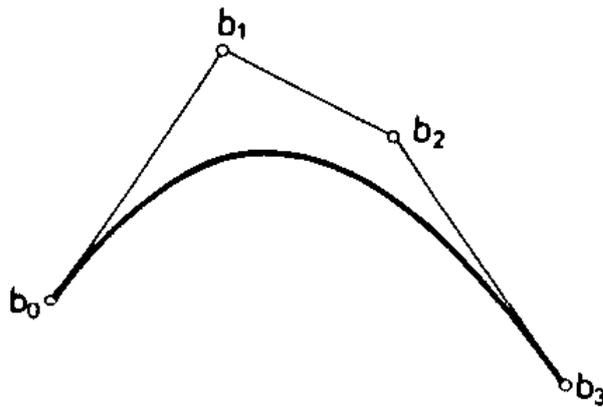
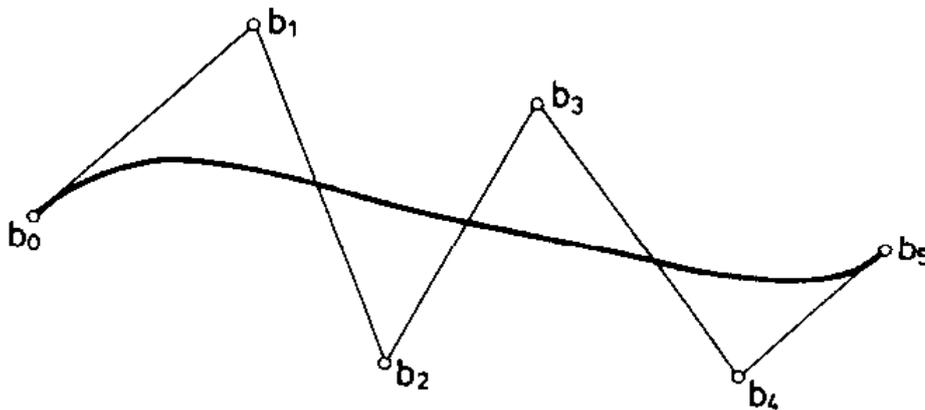
$$\left. \frac{dQ}{dt} \right|_{t=0} = Q_{1,n} = n \cdot (-P_0 + P_1)$$

für $t = 1$ entsprechend.

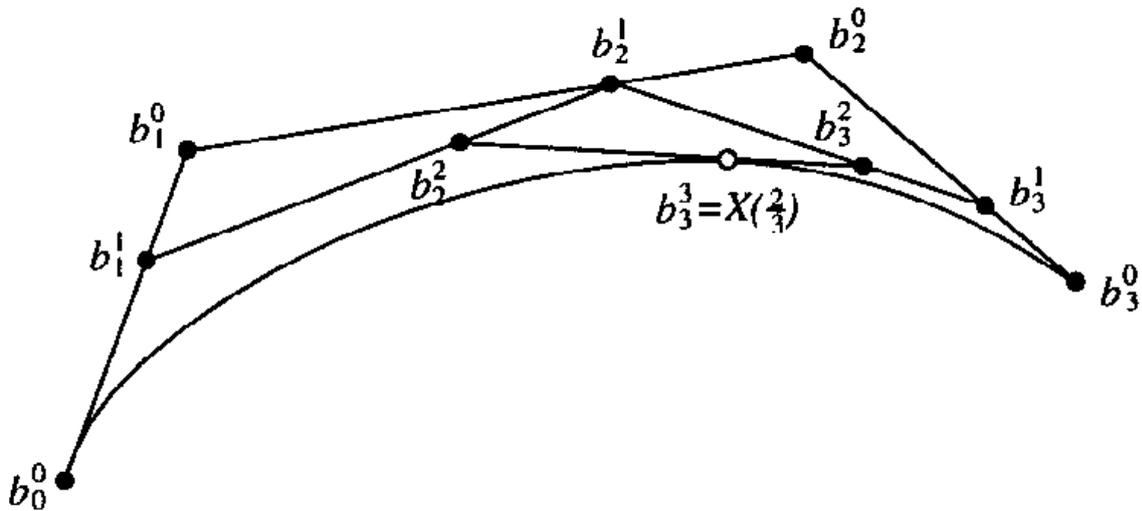
- das Verfahren hat globalen Charakter: Einfügen von zusätzlichen Kontrollpunkten bedingt höheren Grad des approximierenden Polynoms und engere Anlehnung an das Polygon der Kontrollpunkte ("charakteristisches Polygon").

- 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):

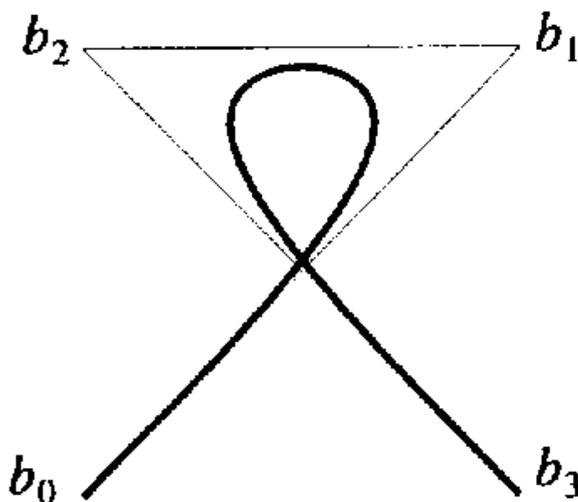


Beispiel: Konstruktion nach de Casteljau für $n = 3$ und $t = 2/3$:



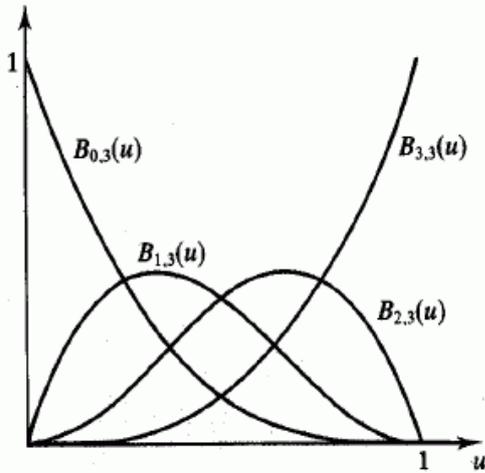
weitere Eigenschaften der Bézier-Kurven:

- 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*



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

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

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

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

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

- Die Bézier Basisfunktionen
- Beobachtungen: \mathbf{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:

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

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

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

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

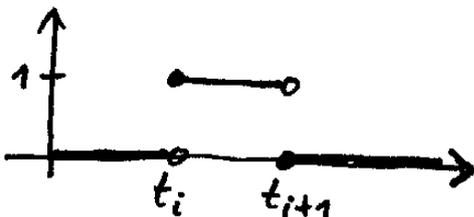
Es sei (t_0, t_1, \dots, t_n) ein "Knotenvektor" (Vektor von Kontrollstellen) mit $t_i \leq t_j$ für $i < j$.

Die (nicht-uniformen) normalisierten B-Spline-Basisfunktionen $N_{i,k}$ der Ordnung k über dem Intervall $[t_0, t_n]$ sind def. durch

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

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

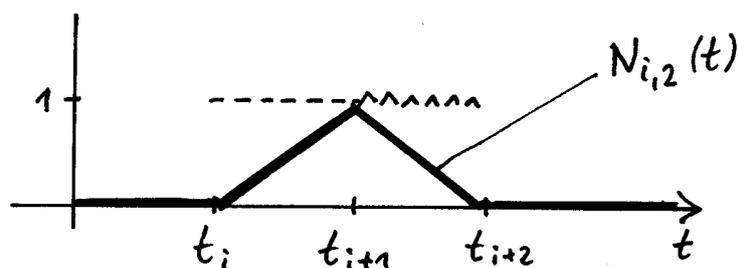
Verlauf von $N_{i,1}$:



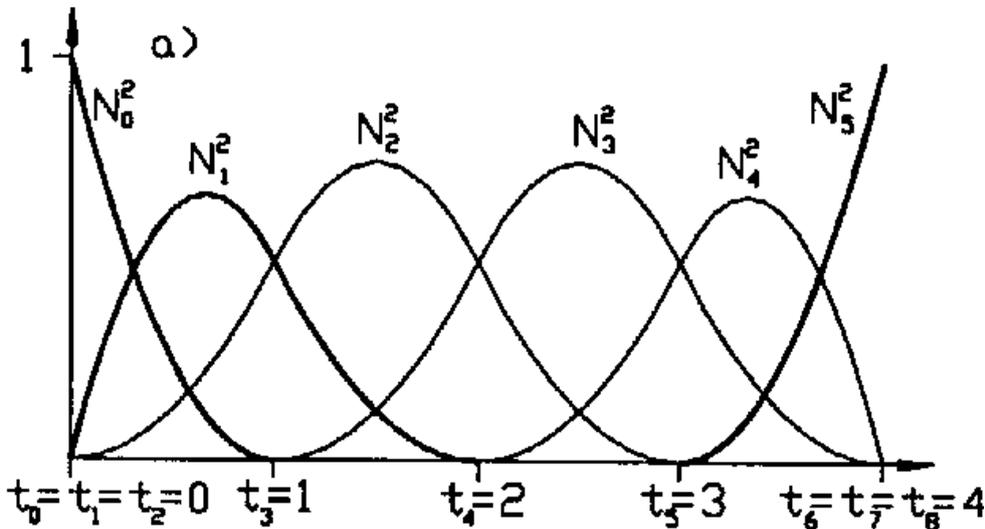
Beispiel $k = 2$:

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

..... ^ ^ ^ ^ ^ ^ ^ ^ ^ ^



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



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

Vereinbarung:

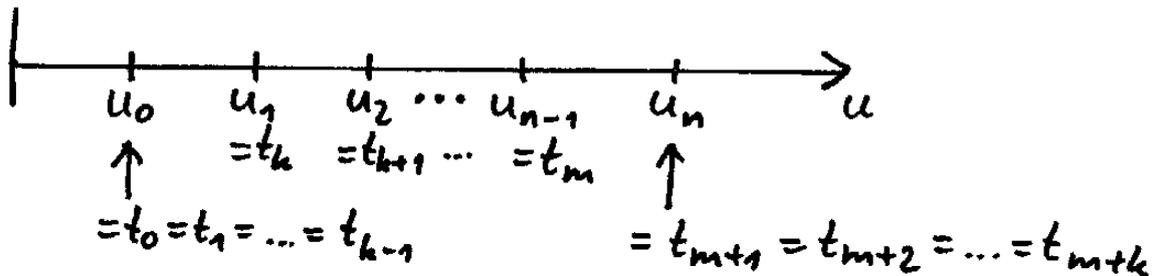
$$\frac{t - t_i}{t_{i+k-1} - t_i} := 0 \quad \text{für} \quad t_{i+k-1} = t_i$$

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

Das Parameterintervall für die Kurve, $[u_0, u_n]$, wird zerlegt in $n = m - k + 2$ Teilintervalle $[u_i, u_{i+1})$, $i = 0; 1; \dots; n-1$.

Def. des Knotenvektors $(t_0, t_1, \dots, t_{k+m})$:

$$\begin{aligned} t_i &= u_0 && \text{für } i = 0; 1; \dots; k-2, \\ t_{i+k-1} &= u_i && \text{für } i = 0; 1; \dots; m-k+2, \\ t_{i+m+2} &= u_n && \text{für } i = 0; 1; \dots; k-2. \end{aligned}$$



- Spezialfall der *uniformen* B-Splines: wähle uniforme (d.h. äquidistante) Teilung von $[u_0, u_n)$, also $u_0 = 0, u_1 = 1, u_2 = 2, \dots, u_n = n$.
- Spezialfall $m = k-1, t_0 = 0, t_1 = 1$: Knotenvektor = $(0, 0, 0, \dots, 0, 1, 1, 1, \dots, 1)$ (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):

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

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

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

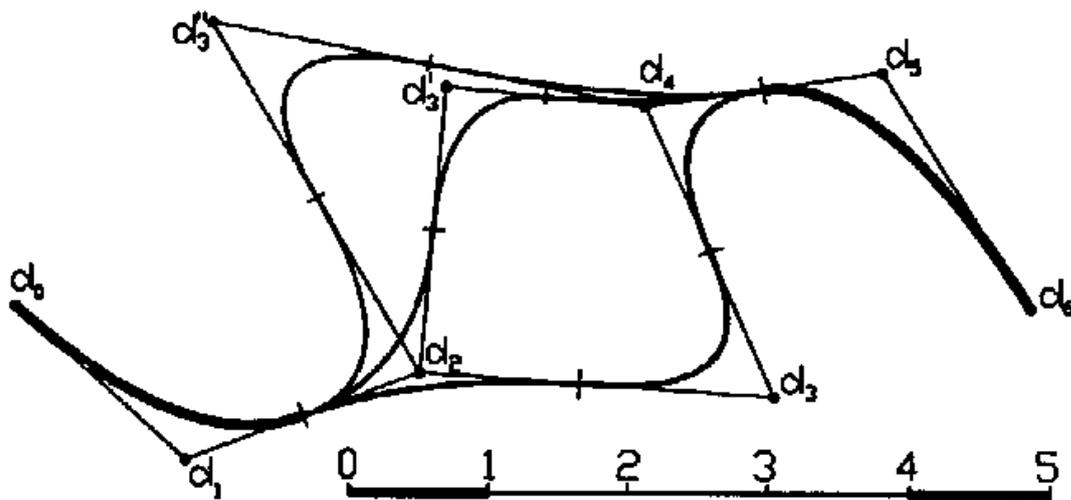
$$Q(t) = \sum_{i=0}^m P_i \cdot N_{i,k}(t), \quad t \in [t_0; t_n)$$

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

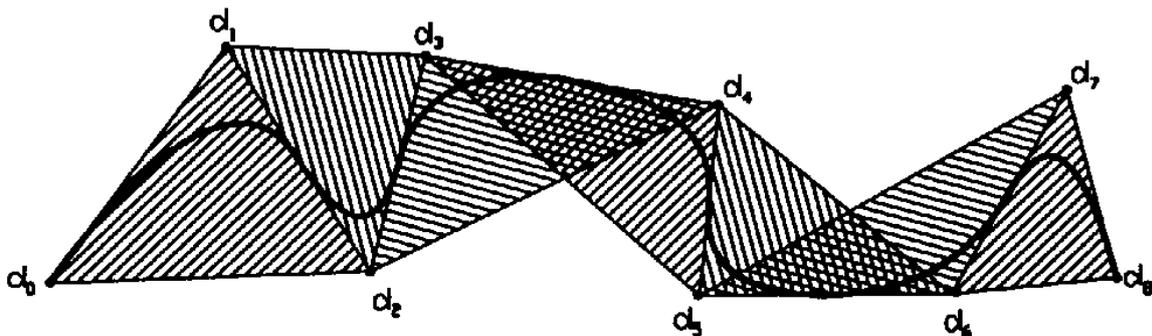
Eigenschaften der B-Spline-Kurven:

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



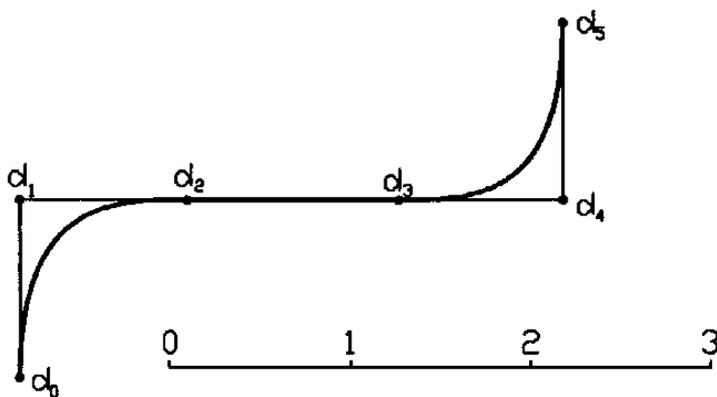
lokale Wirkung der Kontrollpunkte. $k = 3$, Knotenvektor $(0, 0, 0, 1, 2, 3, 4, 5, 5, 5)$; durch Verschieben von d_3 nach d_3' werden lediglich die Kurvensegmente mit Parameterwerten zwischen 1 und 4 beeinflusst.

- 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, \dots, 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
⇒ dadurch zusätzliche Freiheitsgrade

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

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

Somit:

(c) *NURBS*

= 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)

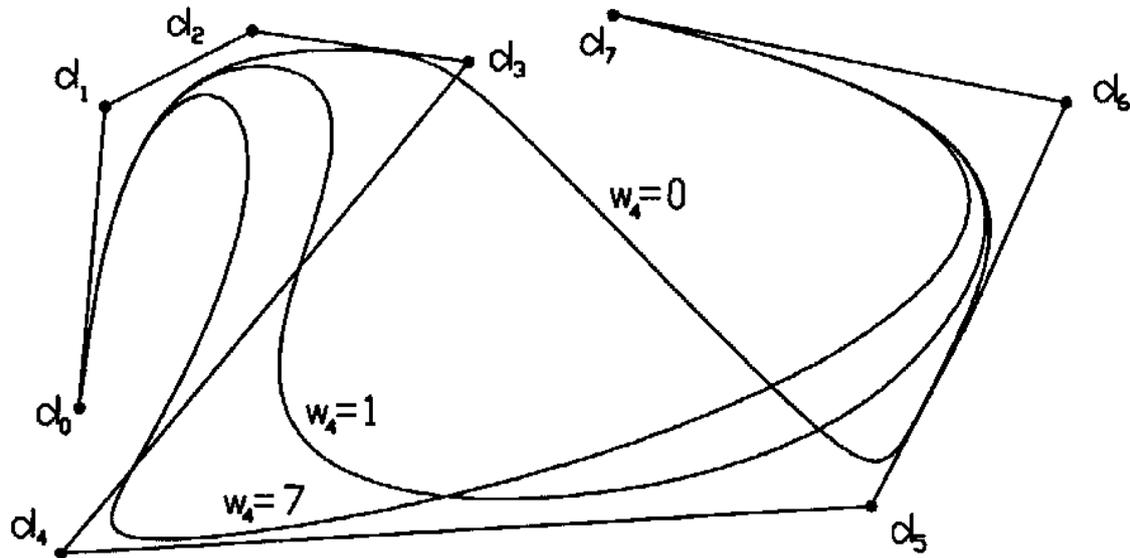
wie bei den einfachen B-Splines wird die Kurve stückweise zusammengesetzt:

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

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.
- ◆ $w_i > 1 \rightarrow$ Kurve nähert sich dem Kontrollpkt.
- ◆ $w_i < 1 \rightarrow$ 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 Kontrollpunkte müssen transformiert werden und nicht jeder Punkt der Kurve)
- ◆ NURBS können Kegelschnitte exakt beschreiben
- ◆ Gegenüber B-Splines durch die Gewichte ergänzende Editiermöglichkeiten

