Computergrafik

Prof. Dr.-Ing. Kerstin Müller

Kapitel 5

Parametrische Kurven und Flächen

Kapitelübersicht (1)

- 5.1 Parametrische Kurven und Flächen
- 5.2 Monom-Darstellung
- 5.3 Die Bernstein-Polynome
- 5.4 Bézier-Kurven
- 5.5 Eigenschaften von Bézier-Kurven
- 5.6 Der deCasteljau Algorithmus
- 5.7 Bézier-Kurven in globalen Parametern
- 5.8 Unterteilung von Bézier-Kurven
- 5.9 (Zeichnen und Schneiden von Bézier-Kurven)
- 5.10 (Anschluß-Konstruktionen)
- 5.11 Splines

Kapitelübersicht (2)

- 5.12 Die B-Spline Basis
- 5.13 deBoor Algorithmus
- 5.14 Splines in B-Spline Darstellung
- 5.15 Tensorprodukt Flächen-Darstellungen
- 5.16 (Rationale Darstellungen (NURBS))

Polynome und Graphen

- Definition: Ein Polynom ist eine Summe von Vielfachen von Potenzen mit natürlich-zahligen Exponenten einer Variablen.

$$x(t) = \sum_{i=0}^{m} a_i \cdot t^i =: \sum_{i=0}^{m} a_i M_i(t), \quad t \in [a,b],\ a_i \in \mathbb{R},\ i \in \mathbb{N}$$

Die $M_i(t) = t^i$ heißen Monome vom Grad i.

- Definition: Der Graph $\mathbf{x}(t)$ einer (polynomialen) Funktion ist definiert als

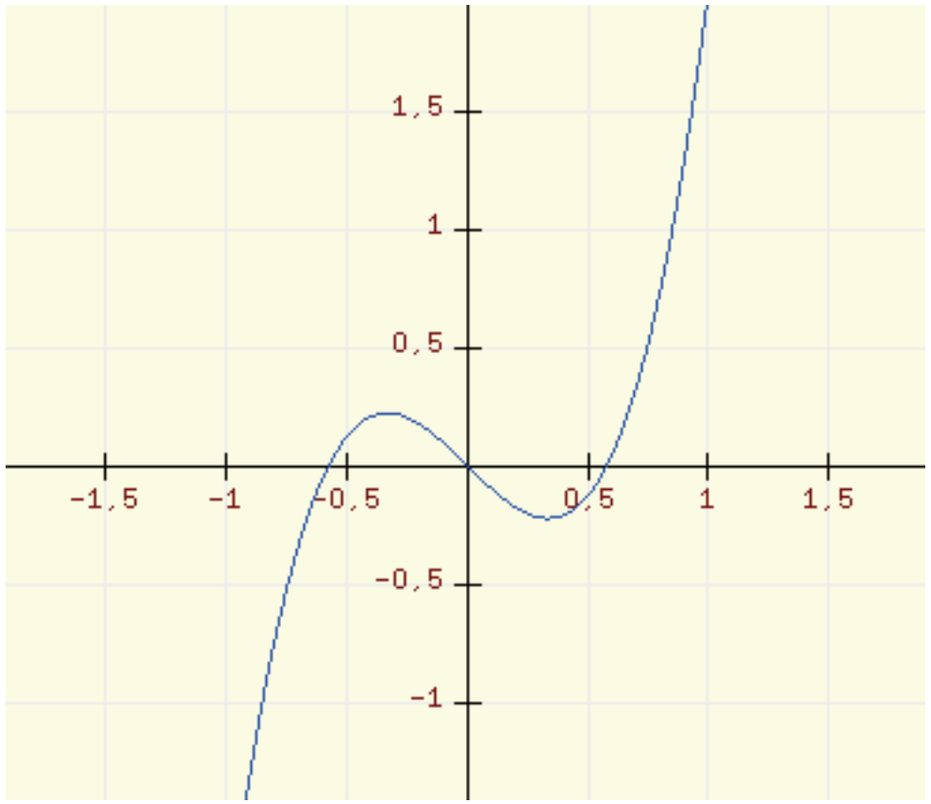
$$\mathbf{x}(t) = \begin{bmatrix} t \\ x(t) \end{bmatrix}$$

$\mathbf{x}(t)$ wird auch als funktionale Kurve bezeichnet.

Graph einer Funktion

- Beispiel:

$$\mathbf{x}(t) = \begin{bmatrix} t \\ 3 \cdot t^3 - t \end{bmatrix}$$



- Bemerkung: Bei der Beschreibung von Bewegungsabläufen oder Freiformgeometrie ist es oft schwer oder gar unmöglich, die Form der Kurve durch den Graph einer Funktion zu beschreiben.

Parametrische Kurven

- Idee: Beschreibe die augenblickliche Position auf einer Kurve als kartesische Koordinaten (x,y), oder (x,y,z) bei einer Raumkurve, die sich mit der Zeit verändern.
- Definition: Ein Element $\mathbf{x} \in \mathbb{R}^3$, dessen Koordinaten von einem Parameter t abhängen, beschreibt eine parametrische Kurve:

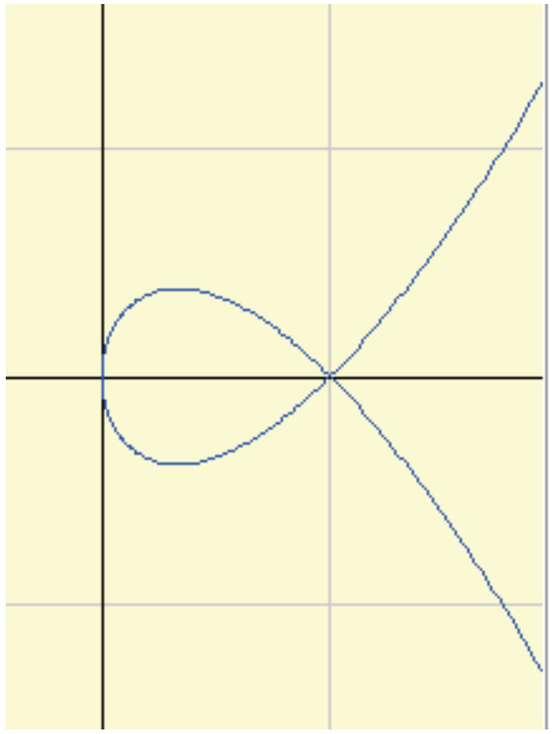
$$\mathbf{x}(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix}$$

Sind $x(t),\ y(t),\ z(t)$ Polynome vom Grad $\leq n$, so heißt $\mathbf{x}(t)$ auch polynomiale Kurve.

Polynomiale Kurven

- Beispiel:

$$\mathbf{x}(t) = \begin{bmatrix} t^2 \\ t^3 - t \end{bmatrix}$$



Schnittpunkt: t =1, t = -1.

- Bemerkung: Der Graph einer Funktion $\mathbf{x}(t)$ ist eine spezielle parametrische Kurve im $\mathbb{R}^2$.

Bivariate Polynome

- Definition: Ein bivariates Polynom $x(s,t)$, d.h. ein Polynom in zwei Variablen, ist gegeben durch:

$$x(s,t) = a_{00} + a_{01} \cdot s + a_{10} \cdot t + a_{11} \cdot s \cdot t + a_{12} \cdot s \cdot t^2 + \ldots + a_{02} \cdot t^2 + \ldots$$
$$= \sum_i \sum_j a_{ij} \cdot s^i \cdot t^j$$

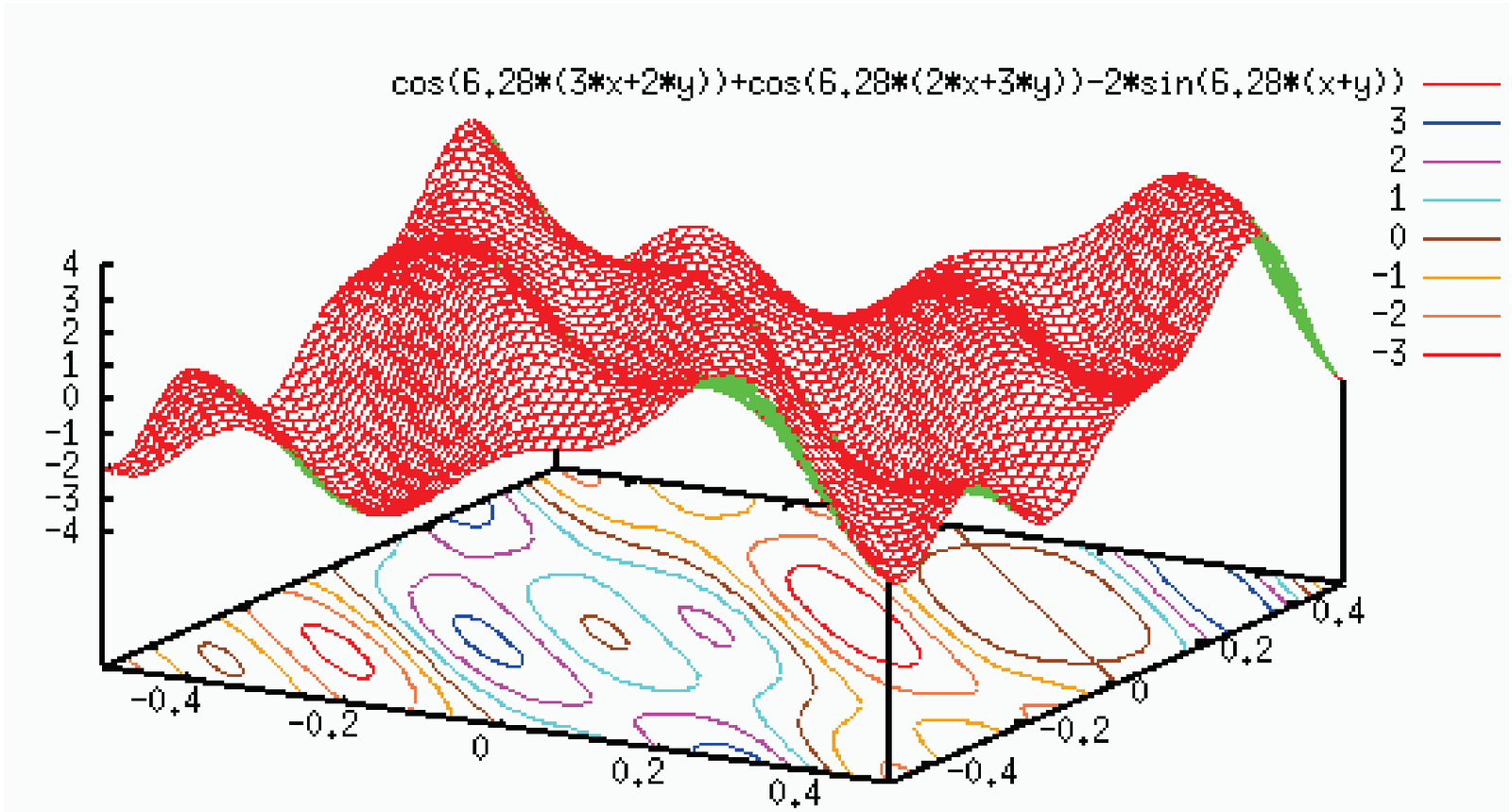
- Definition: Der Graph $\mathbf{x}(s,t)$ einer (polynomialen) Funktion $x(s,t)$ ist definiert als

$$\mathbf{x}(s,t) = \begin{bmatrix} s \\ t \\ x(s,t) \end{bmatrix}$$

$\mathbf{x}(s,t)$ wird auch als funktionale Fläche bezeichnet.

Funktionale Flächen

- Beispiel:



- Anwendung: Zur Modellierung von Landschaften, Höhenprofilen etc. gut geeignet.

Parametrische Flächen

- Bemerkung: Ebenso wie bei funktionalen Kurven ist eine Beschreibung beliebiger Flächenformen durch den Graphen einer Funktion nicht immer möglich.
- Idee: Analog zu den Kurven die augenblickliche Position auf der Fläche als kartesische Koordinaten in Abhängigkeit von zwei Parametern s und t beschreiben.
- Definition: Ein Element $\mathbf{x} \in \mathbb{R}^3$, dessen Koordinaten von zwei Parametern abhängen, beschreibt eine parametrische Fläche:

$$\mathbf{x}(s,t) = \begin{bmatrix} x(s,t) \\ y(s,t) \\ z(s,t) \end{bmatrix}$$

Parametrische Flächen

- Bemerkung: Eine parametrische Fläche heißt polynomial vom Grad n , wenn alle x, y, z Polynome vom totalen Grad $\leq n$ sind.
- Definition: Der totale Grad eines Polynoms $x(s,t)$ ist definiert als:

$$deg(x(s,t)) = max\{i+j | a_{ij} \neq 0\}$$

wobei $x(s,t)$ wie bisher: $x(s,t) = \sum_i \sum_j a_{ij} \cdot s^i \cdot t^j$

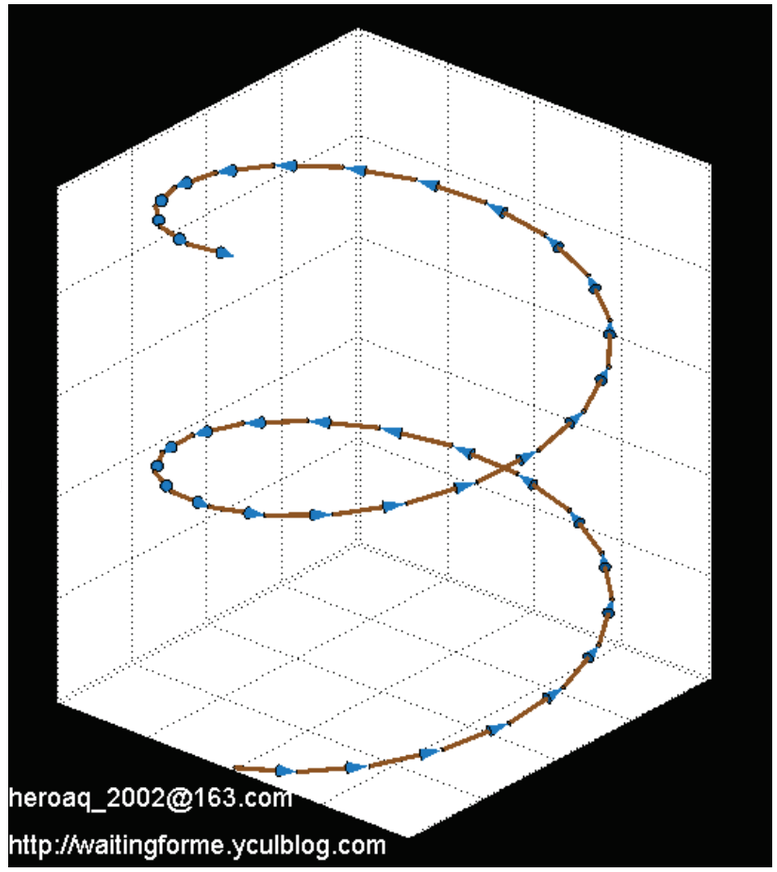
Beispiele für parametrische Kurven

- Weitere Beispiele:
  - Ein Beispiel einer Raumkurve ist gegeben durch

$$\mathbf{x}(t) = \begin{bmatrix} R \cdot \cos t \\ R \cdot \sin t \\ H \cdot t \end{bmatrix}$$

Diese Parameterkurve stellt eine Helix oder Schraubenlinie mit Radius R dar. Je nach Parameterintervall erreicht die Schraube eine bestimmte Höhe. Für das Intervall [0,2π] wird genau eine Umdrehung realisiert, die Höhe durch H*2π gegeben.

Für H=0, oder fehlende 3. Koordinate wird ein Kreis realisiert.



5.2 Monomdarstellung

- Sind die Koordinatenfunktionen einer parametrischen Kurve als Polynome vom Grad $\leq n$ gegeben, etwa

$$\mathbf{x}(t) = \begin{bmatrix} x(\quad t) \\ y(\quad t) \\ z(\quad t) \end{bmatrix} = \begin{bmatrix} a_0^x + a_1^x \cdot t + a_2^x \cdot t^2 + \ldots \\ a_0^y + a_1^y \cdot t + a_2^y \cdot t^2 + \ldots \\ a_0^z + a_1^z \cdot t + a_2^z \cdot t^2 + \ldots \end{bmatrix}$$

- so kann man dies schreiben als:

$$\mathbf{x}(t) = \begin{bmatrix} a_0^x \\ a_0^y \\ a_0^z \end{bmatrix} + \begin{bmatrix} a_1^x \\ a_1^y \\ a_1^z \end{bmatrix} \cdot t + \begin{bmatrix} a_2^x \\ a_2^y \\ a_2^z \end{bmatrix} \cdot t^2 = \sum_{i=0}^{n} \begin{bmatrix} a_i^x \\ a_i^y \\ a_i^z \end{bmatrix} \cdot t^i$$

$$= \mathbf{a}_0 + \mathbf{a}_1 \cdot t + \mathbf{a}_2 \cdot t^2 + \ldots = \sum_{i=0} \mathbf{a}_i \cdot t^i, \quad \mathbf{a}_i \in \mathbb{R}^3$$

Monomdarstellung

- Bemerkung: Die $t^i,\ i = 0, \ldots, n$ heißen Monome vom Grad i. Alle Polynome vom Grad $\leq n$ können als Summe von Vielfachen dieser „Elementarpolynome“ geschrieben werden. Mathematisch gesprochen bilden die Monome vom Grad $\leq n$ eine Basis des Vektorraums aller Polynome vom Grad $\leq n$.
- Betrachte nochmals

$$\mathbf{x}(t) = \mathbf{a}_0 + \mathbf{a}_1 \cdot t + \mathbf{a}_2 \cdot t^2 + \ldots = \sum_{i=0} \mathbf{a}_i \cdot M_i(t), \quad \mathbf{a}_i \in \mathbb{R}^3$$

Beobachtung: $\mathbf{a}_i \in \mathbb{R}^3, \quad M_i(t) \in \mathbb{R}$ für ein festes t.

Kurvendarstellung: neue Sichtweise

- Die Kurve entsteht durch das Zusammenmischen der beteiligten „Punkte“ $\mathbf{a}_i$ im 3D-Raum mit den Mischfunktionen M_i (auch: „blending functions“).
- Ein Kurvenpunkt zum Zeitpunkt t entsteht also durch das Zusammenmischen aller Punkte $\mathbf{a}_i$ mit ihrem jeweiligen Gewicht $M_i(t)$.
- Die M_i müssen nicht notwendig Polynome sein.
- Falls die M_i Polynome sind, müssen sie nicht unbedingt Monome sein, da es verschiedene Basen für den Vektorraum aller Polynome vom Grad $\leq n$ gibt.

Monomdarstellung geeignet?

- (1) Haben die aus den Koeffizienten gebildeten Punkte einen geometrischen Sinn?
- (2) Sind die $M_i(t) = t^i$ geeignete Mischfunktionen?

- Zu (1): Nur wenige Punkte haben einen Sinn, es gilt lediglich $\mathbf{x}(0) = \mathbf{a}_0,\ \mathbf{x}'(0) = \mathbf{a}_1$ („Ableitung komponenten-weise“). Alle anderen Punkte haben keine geometrische Bedeutung. Noch schwerer wiegt die Tatsache, dass die Verschiebung einzelner Punkte keine intuitive Veränderung der Kurve verursacht.

Monomdarstellung

- Zu (2): Beispiel

$$\mathbf{x}(t) = \mathbf{a}_0 + \mathbf{a}_1 \cdot t \quad : \quad \text{Strecke zwischen } \mathbf{a}_0 \text{ und } \mathbf{a}_0 + \mathbf{a}_1 \text{ falls } t \in [0,1] \text{ .}$$

- Verschiebe die Kontrollpunkte $\mathbf{a}_0$ und $\mathbf{a}_1$ um den Vektor $\mathbf{c}$. Wünschenswert wäre, dass sich die Kurve ebenfalls um $\mathbf{c}$ verschiebt, aber „in sich“ unverändert bleibt.
- Rechnung:

$$\begin{aligned} & (\mathbf{a}_0 + \mathbf{c}) + (\mathbf{a}_1 + \mathbf{c}) \cdot t \\ = \quad & \mathbf{a}_0 + \mathbf{a}_1 \cdot t + \mathbf{c} + \mathbf{c} \cdot t \\ = \quad & \mathbf{x}(t) + \underbrace{(1+t)}_{\neq 1} \mathbf{c} \end{aligned}$$

Monomdarstellung

- Vermutung: Für Translations-Invarianz muss gelten

$$\sum_{i=0}^{n} M_i(t) = 1$$ und zwar für jede Wahl von t.

- Symmetrische Darstellung:

$$\begin{aligned}
\mathbf{x}(t) &= \mathbf{a}_0 + \mathbf{a}_1 \cdot t \\
&= \underbrace{\mathbf{a}_0}_{=:\mathbf{p}_0}(1-t) + \underbrace{(\mathbf{a}_0 + \mathbf{a}_1)}_{=:\mathbf{p}_1} \cdot t \\
&=: \mathbf{p}_0(1-t) + \mathbf{p}_1 \cdot t
\end{aligned}$$

Symmetrische Darstellung

- Die symmetrische Darstellung ist translations-invariant, d.h. die Verschiebung der $\mathbf{p}_i,\ i = 1, 2$ entspricht der Verschiebung von $\mathbf{x}(t)$, denn:

$$
\begin{aligned}
& (\mathbf{p}_0 + \mathbf{c}) \cdot (1 - t) + (\mathbf{p}_1 + \mathbf{c}) \cdot t \\
= \quad & \mathbf{p}_0 \cdot (1 - t) + \mathbf{p}_1 \cdot t + \mathbf{c} \cdot ((1 - t) + t) \\
= \quad & \mathbf{x}(t) + \mathbf{c}
\end{aligned}
$$

- Zusätzlich sind $\mathbf{p}_0$ und $\mathbf{p}_1$ die Endpunkte der Strecke, also haben alle Kontrollpunkte eine geometrische Bedeutung.
- Ebenso gilt mit $M_0(t) = (1 - t)$ und $M_1(t) = t$
  $M_0(t) + M_1(t) = 1$

5.3 Die Bernstein-Polynome

- Allgemein: Gesucht ist eine Basis des Vektorraums der Polynome vom Grad $\leq n$ mit günstigen geometrischen Eigenschaften, insbesondere Translations-Invarianz.

$$\mathbf{x}(t) = \sum_{i=0}^{n} \mathbf{p}_i B_i^n(t) \qquad \mathbf{x}(t) + \mathbf{c} = \sum_{i=0}^{n} (\mathbf{p}_i + \mathbf{c}) \cdot B_i^n(t)$$

$$= \sum_{i=0}^{n} \mathbf{p}_i \cdot B_i^n(t) + \mathbf{c} \sum_{i=0}^{n} \cdot B_i^n(t)$$

- $\Rightarrow \sum_{i=0}^{n} B_i^n(t) = 1$ ist notwendig.

Herleitung Bernstein-Polynome

- Trick:

$$\begin{aligned} ((1-t)+t) &= 1 \\ ((1-t)+t)^n &= 1 \\ \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} \cdot t^i &= 1 \end{aligned}$$

- denn aus Formelsammlung: $(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^{n-i} \cdot b^i$

- Definition: Die Polynome $\binom{n}{i} (1-t)^{n-i} \cdot t^i =: B_i^n(t)$

  heißen Bernstein-Polynome vom Grad n.

Eigenschaften der Bernstein-Polynome

- Erinnerung: Der Binomialkoeffizient ist definiert als

$$\binom{n}{i} = \begin{cases} \frac{n!}{i! \cdot (n-i)!}, & 0 \leq i \leq n \\ 0, & \text{sonst.} \end{cases}$$

- Eigenschaften der Bernstein-Polynome:
  - Die Bernstein-Polynome vom Grad n sind linear unabhängig und bilden eine Basis des Vektorraums der Polynome vom Grad n.
  - Die Bernstein-Polynome bilden eine Teilung der Eins, d.h. für alle

$$t \in [0,1] \quad \text{gilt:} \quad \sum_{i=0}^{n} B_i^n(t) = 1$$

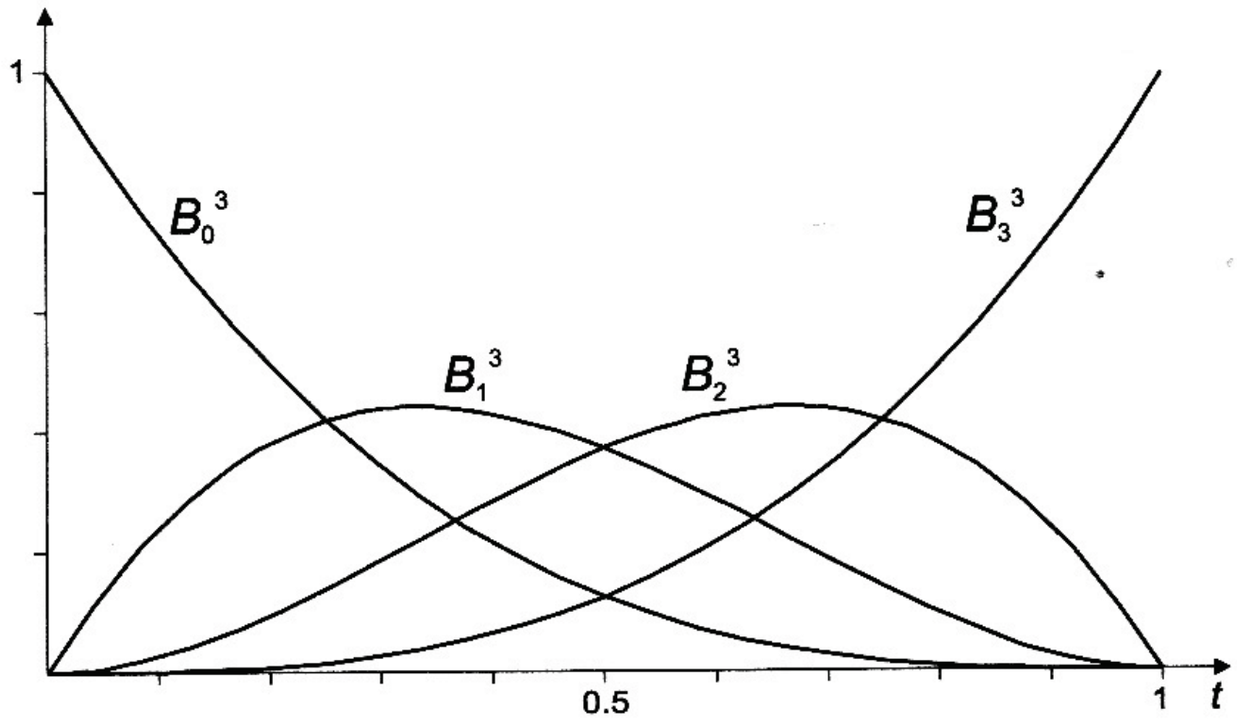
Eigenschaften der Bernstein-Polynome

- Die Bernstein-Polynome sind alle nicht-negativ in [0,1]. Im Inneren des Intervalls sind sie positiv.
- Die Bernstein-Polynome sind symmetrisch, d.h

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

- Beispiel:

  Die 4 Bernstein Polynome vom Grad 3.



Eigenschaften der Bernstein-Polynome

- Ein Bernstein-Polynom vom Grad n lässt sich als Konvex-Kombination (d.h. als gewichtete Summe mit positiven Gewichten die sich zu Eins summieren) von Bernstein-Polynomen vom Grad n-1 darstellen. Für das Intervall $t \in [0,1]$ bedeutet dies:

$$B_i^n(t) = t \cdot B_{i-1}^{n-1}(t) + (1-t) \cdot B_i^{n-1}(t) \quad \text{mit} \quad B_{-1}^n = B_{n+1}^n = 0,\ B_0^0 = 1$$

  Beweis: Übung.

- Für die erste Ableitung eines Bernstein-Polynoms gilt für $t \in [0,1]$:

$$\frac{d}{dt} B_i^n(t) = n \cdot \left(B_{i-1}^{n-1}(t) - B_i^{n-1}(t) \right)$$

  Dies benötigt man beispielsweise um Tangenten-Richtungen von Kurven, die aus Bernstein-Polynomen konstruiert wurden, zu bestimmen.

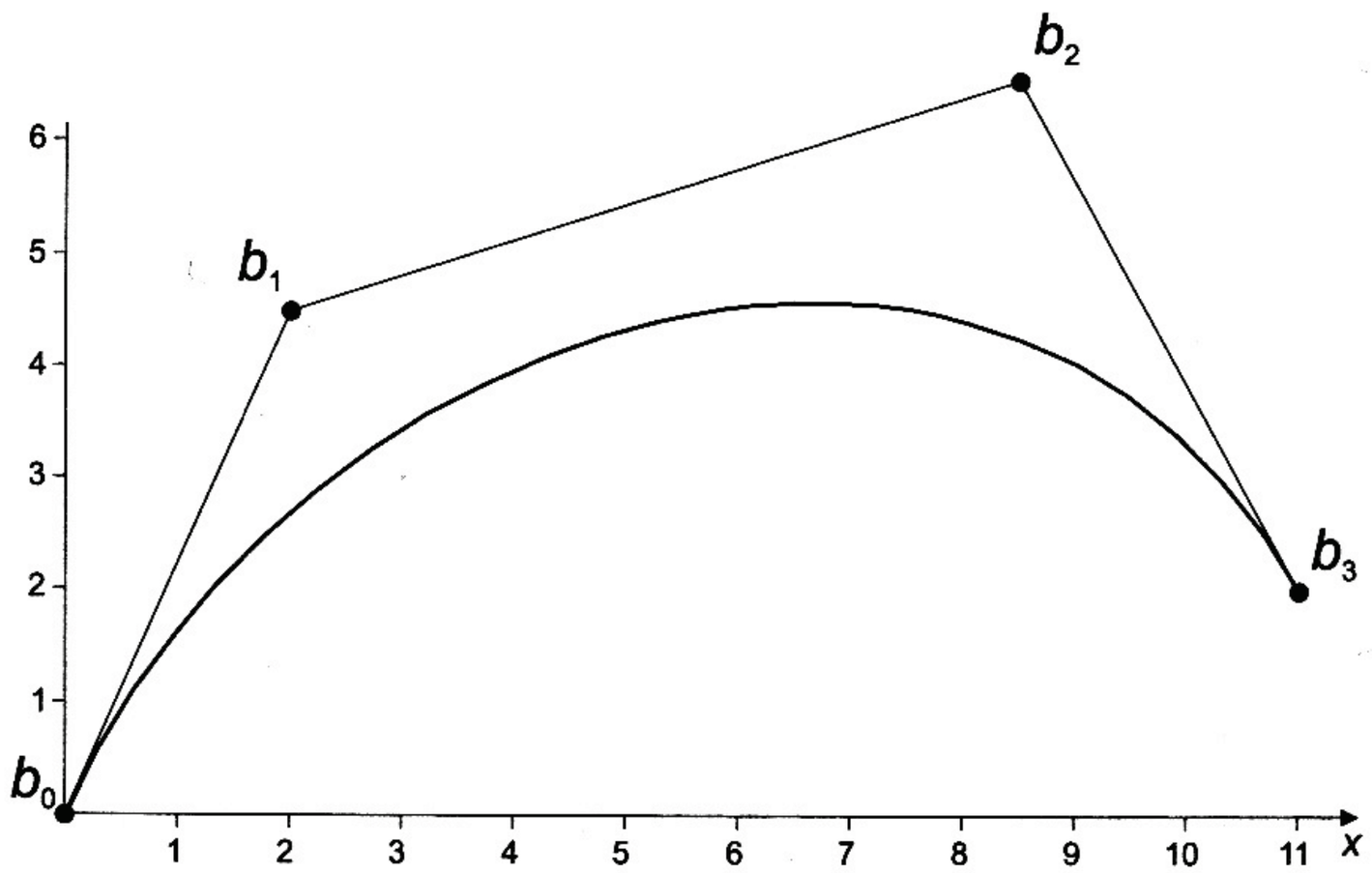
5.4 Bézier-Kurven

- Da die Bernstein-Polynome eine Basis für den Vektorraum der Polynome bilden, existiert für jede polynomiale Kurve $\mathbf{b}(t)$ eine mathematische Darstellung durch die Bernstein-Basis:

$$\mathbf{b}(t) = \sum_{i=0}^{n} \mathbf{b}_i \cdot B_i^n(t), \quad t \in [0,1]$$

- Die $\mathbf{b}_i$ heißen Bézier-Punkte; sie sind die Kontrollpunte für die Kurve.Im Gegensatz zu den $\mathbf{a}_i$ aus der Monom-Darstellung haben die $\mathbf{b}_i$ eine geometrische Bedeutung.
- Definition: Der durch die Kontrollpunkte definierte Polygonzug heißt Kontrollpolygon.

Bézier-Kurven



Vorteile von Bézier-Kurven

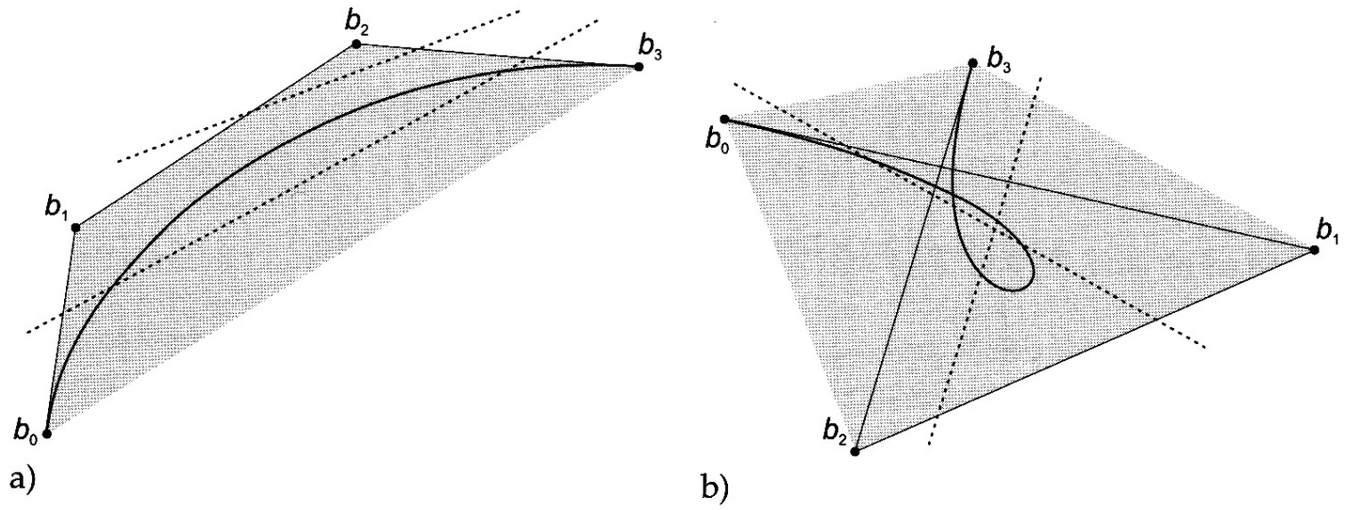
- Es existiert nun ein direkter Zusammenhang zwischen der mathematischen Beschreibung und der Geometrie, d.h. der Form der Kurve. Der Verlauf der Kurve folgt intuitiv dem Verlauf des Kontroll-Polygons; dies ist der Ansatz zum interaktiven Modellieren. Der Anwender manipuliert Kontrollpunkte und die Kurve folgt intuitiv der Verschiebung der Punkte.
- Es existiert ein eleganter Algorithmus zur Berechnung des Kurvenverlaufs, der sich die Rekursion der Bernstein-Polynome

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

zu Nutze macht, der deCasteljau Algorithmus. Dieser hat einen geometrischen Bezug zur Kurve und zum Kontrollpolygon.

Eigenschaften von Bézier-Kurven

- Erinnerung: Frage (2): Wie sieht die Kurve für eine Menge von gegebenen Kontrollpunkten aus.
- Konvexe Hülle Eigenschaft:
  - Erinnerung: Für jeden Parameterwert sind alle Mischfunktionen nicht negativ (Eigenschaft der Bernstein-Polynome) und teilen die Eins. Damit ist jeder Kurvenpunkt $\mathbf{b}(t)$ eine Konvex-Kombination aus allen Kontrollpunkten $\mathbf{b}_i$ mit Gewichten $B_i^n(t)$. Das bedeutet, sie kann die konvexe Hülle, die durch die Kontrollpunkte aufgespannt wird, nicht verlassen.

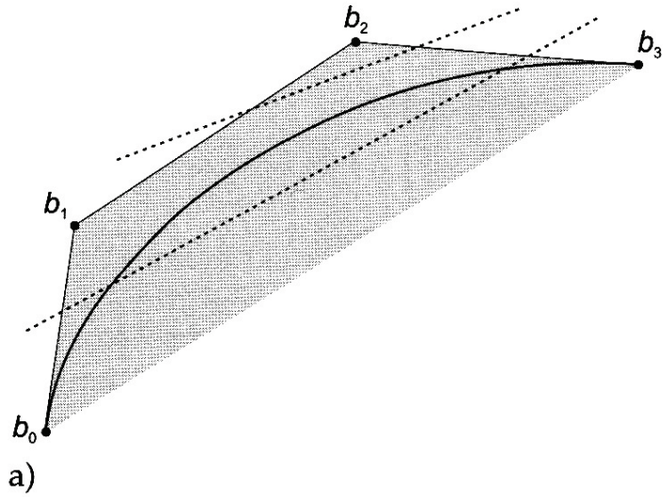


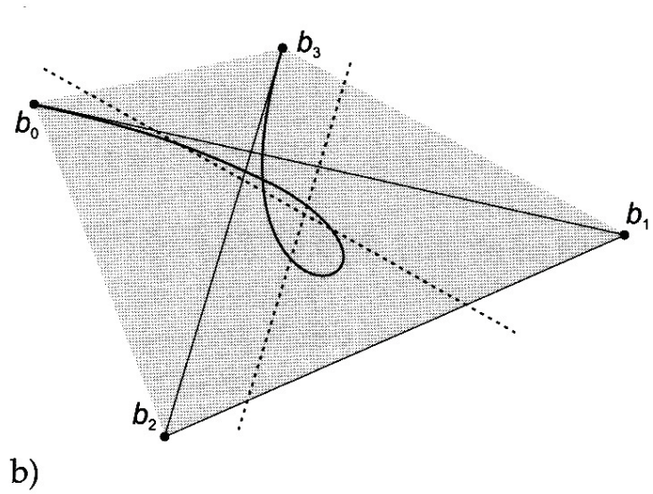
Konvexe Hülle Eigenschaft

- Konvexe Hülle:
  - Die Konvexe Hülle ist die Menge aller Konvex-Kombinationen der $\mathbf{b}_i$, d.h.

$$conv(\{\mathbf{b}_0, \ldots, \mathbf{b}_n\}) = \left\{ \sum_{i=0}^{n} \alpha_i \mathbf{b}_i \text{ mit } \sum_{i=0}^{n} \alpha_i = 1,\ \alpha_i \geq 0 \right\}$$

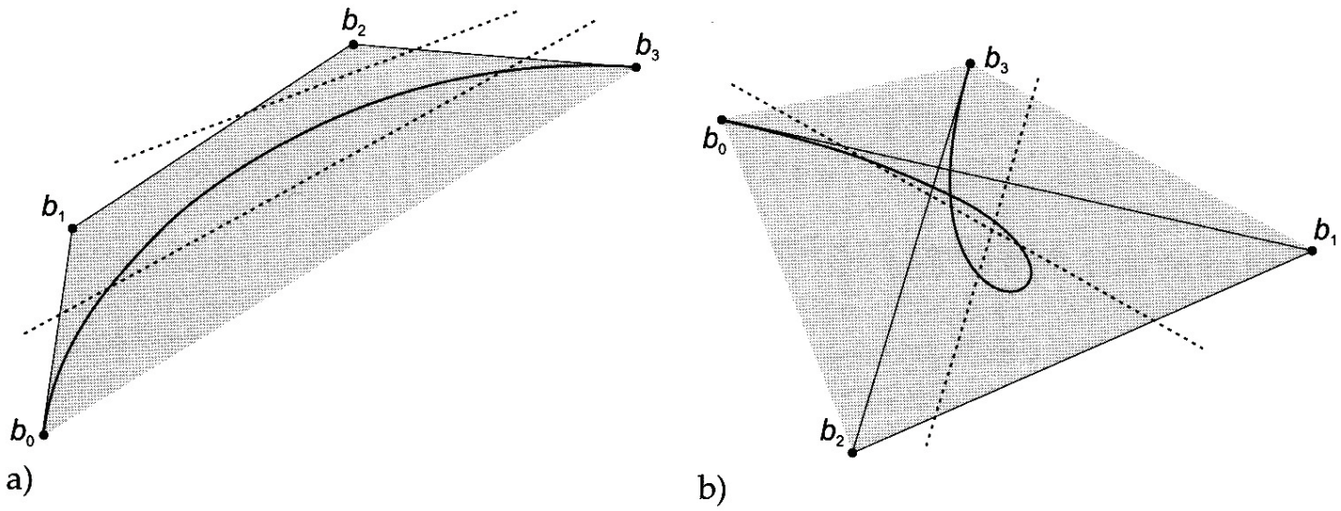
  - Anschaulich 2D: Lege einen Faden um die Kontrollpunkte, die umschlossene Fläche ist die konvexe Hülle.





Variations-Minderung

- Folgerung aus der Konvexe-Hülle Eigenschaft
  - Liegen alle Kontrollpunkte in einer Ebene, kann die Bézier-Kurve diese Ebene nicht verlassen. Liegen alle Kontrollpunkte auf einer Strecke, so ist die Bézier-Kurve Teil dieser Strecke.
- Variations-Minderungs-Eigenschaft
  - Eine Bézier-Kurve ist nicht “welliger” als ihr Kontroll-Polygon. D.h. eine Gerade kann eine Bézier-Kurve nicht häufiger schneiden als sie das zugehörige Kontroll-Polygon schneidet.
  - Nur für planare Kurven sinnvoll!



Anfangs- und Endpunkt-Interpolation

- Anfangs- und Enpunkt-Interpolation
  - Der Anfangs- bzw. Endpunkt einer Bézier-Kurve ist mit dem Anfangs- bzw. Endpunkt des zugehörigen Kontroll-Polygons identisch.
  - Begründung: Gewichteverteilung: $\mathbf{b}_0$ hat Gewicht $B_0^n(t)$

$$\Rightarrow B_0^n(0) = 1,\ B_i^n(0) = 0 \text{ für } i \neq 0$$

analog, symmetrisch am Endpunkt

$$\Rightarrow B_0^n(0) = B_n^n(1) = 1,\ B_i^n(1) = 0 \text{ für } i \neq n$$

Modellier-Eigenschaft

Modellier-Eigenschaft

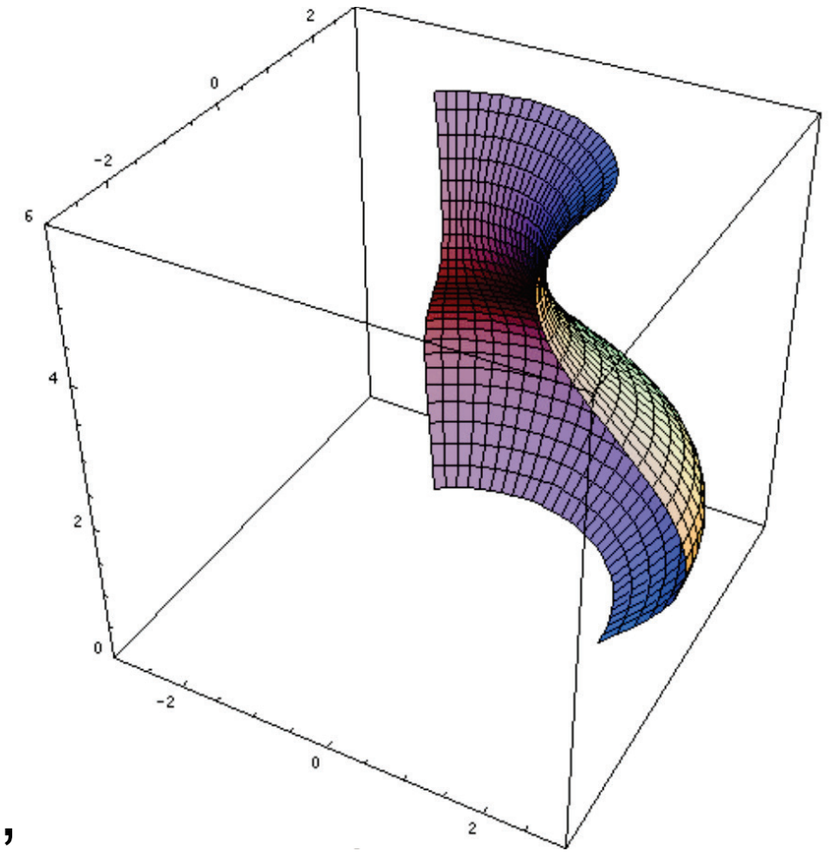
- Die Bézier-Kurve folgt intuitiv der Form seines Kontroll-Polygons. Dieses wiederum kann einfach durch den Anwender manipuliert werden. Jeder Kontrollpunkt hat in dem Kurventeil, dem er am nächsten liegt, den größten Einfluss. Und zwar maximal zu dem Zeitpunkt, an dem das zugehörige Bernstein-Polynom sein Maximum annimmt.
- Ein großer Nachteil der Bézier-Darstellung wird hier ebenfalls deutlich. Abgesehen vom Anfangs- und Endpunt über jeder Kontrollpunkt zu jedem Zeitpunkt einen gewissen Einfluss aus.
- Verschiebt man einen Kontrollpunkt, ist die Änderung dort am größten, wo man es intuitiv erwartet, aber insgesamt wird die komplette Kurve verändert.
- Dies nennt man pseudo-lokale Kontrolle.

Affine Invarianz

- Affine Invarianz
  - Die Bernstein-Basis war so konstruiert, dass Translationen die Kurve in sich unverändert lassen. Darüber hinaus gilt diese Eigenschaft der Invarianz unter allen affinen Abbildungen, insbesondere Rotationen.
  - Konkret: Das Bild von b(t) unter einer affinen Abbildung $\Phi(P)=AP+T$ ist eine Bézier-Kurve c(t) mit Bézier-Punkten $\Phi(b_i)$.
- Praktische Bedeutung:
  - Wird z.B. eine Rotation auf eine Bézier-Kurve angewendet, so reicht es aus, die Abbildung nur auf die Kontrollpunkte anzuwenden und dann das transformierte Kontrollpolygon zur Auswertung der Kurve zu nutzen, statt alle Kurvenpunkte einzeln zu transformieren.

Affine Invarianz

- Beispiel:
  - Wird die Kontour-Kurve der Schachfiguren als Bézier-Kurve modelliert, so reicht es die Kontrollpunkte zu rotieren (sukzessive in der gewünschten Auflösung) und dann die entstehende Kurven-Schaar auszuwerten.
  - Auch für die translierten und rotierten Figuren, z.B. in der Holzkiste analog möglich.



  - Mit der Monom-Darstellung wäre dies nicht möglich! Hier müsste jede einzelne Kurve, die später benötigt wird, gesondert modelliert und ausgewertet werden! (enormer Mehraufwand)
  - Die mathematische Darstellung ist unabhängig von der später benötigten Auflösung (Genauigkeit der Auswertung nach Bedarf).

5.6 Der deCasteljau-Algorithmus

- Wir betrachten nochmals die Rekursion der Bernstein-Polynome:

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

- Wir wenden dies jetzt iteriert auf die Bézier-Kurve an,

dabei ist $\mathbf{b}_i =: \mathbf{b}_i^0$

$$\mathbf{b}(t) = \sum_{i=0}^{n} \mathbf{b}_i^0 \cdot B_i^n(t) = \sum_{i=0}^{n} \mathbf{b}_i^0 \left(t \cdot B_{i-1}^{n-1} + (1-t) B_i^{n-1} \right)$$

da $B_i^{n-1} = 0$ für $i > n-1$ ist die Summe nur noch

$$\mathbf{b}(t) = \sum_{i=0}^{n-1} \mathbf{b}_i^0 \left(t \cdot B_{i-1}^{n-1} + (1-t) B_i^{n-1} \right)$$

Der deCasteljau-Algorithmus

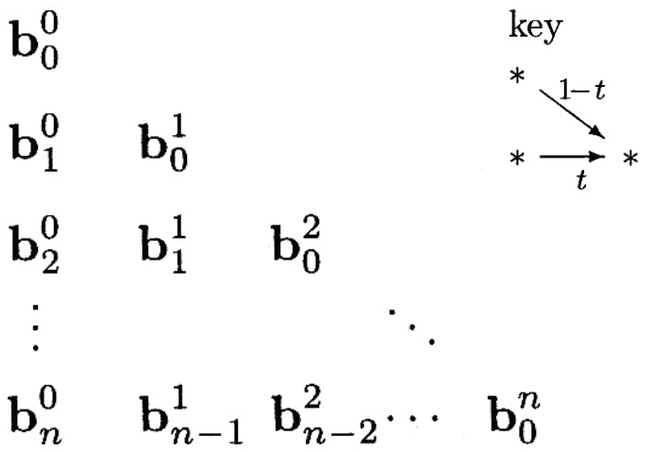
- sortieren jetzt die Indices neu nach den B_i, d.h für ein festes B_i sammle alle $\mathbf{b}'s$:

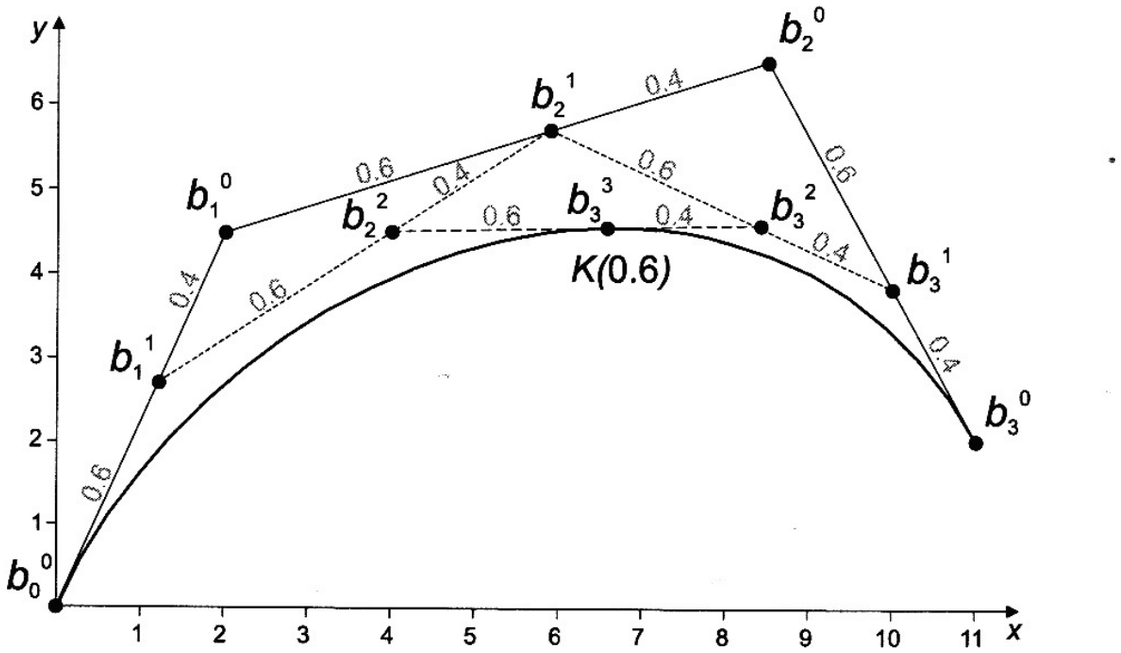
$$\begin{aligned}
\mathbf{b}(t) &= \sum_{i=0}^{n-1} \underbrace{\left((1-t)\cdot \mathbf{b}_i^0 + t\cdot \mathbf{b}_{i+1}^0\right)}_{=:\mathbf{b}_i^1} \cdot B_i^{n-1}(t) \\
&\vdots \\
&= \sum_{i=0}^{0} \mathbf{b}_i^n \cdot B_i^0(t) \qquad \text{mit } \mathbf{b}_i^{k+1} = (1-t)\cdot \mathbf{b}_i^k + t\cdot \mathbf{b}_{i+1}^k \\
&= \mathbf{b}_0^n
\end{aligned}$$

- Das bedeutet, dass man den Kurvenpunkt zu einem vorgegebenen Zeitpunkt t allein durch iterierte gewichtete Mittelpunktbildung der Kontrollpunkte $\mathbf{b}_i^0$ berechnen kann.

Der deCasteljau-Algorithmus

- schematisch:

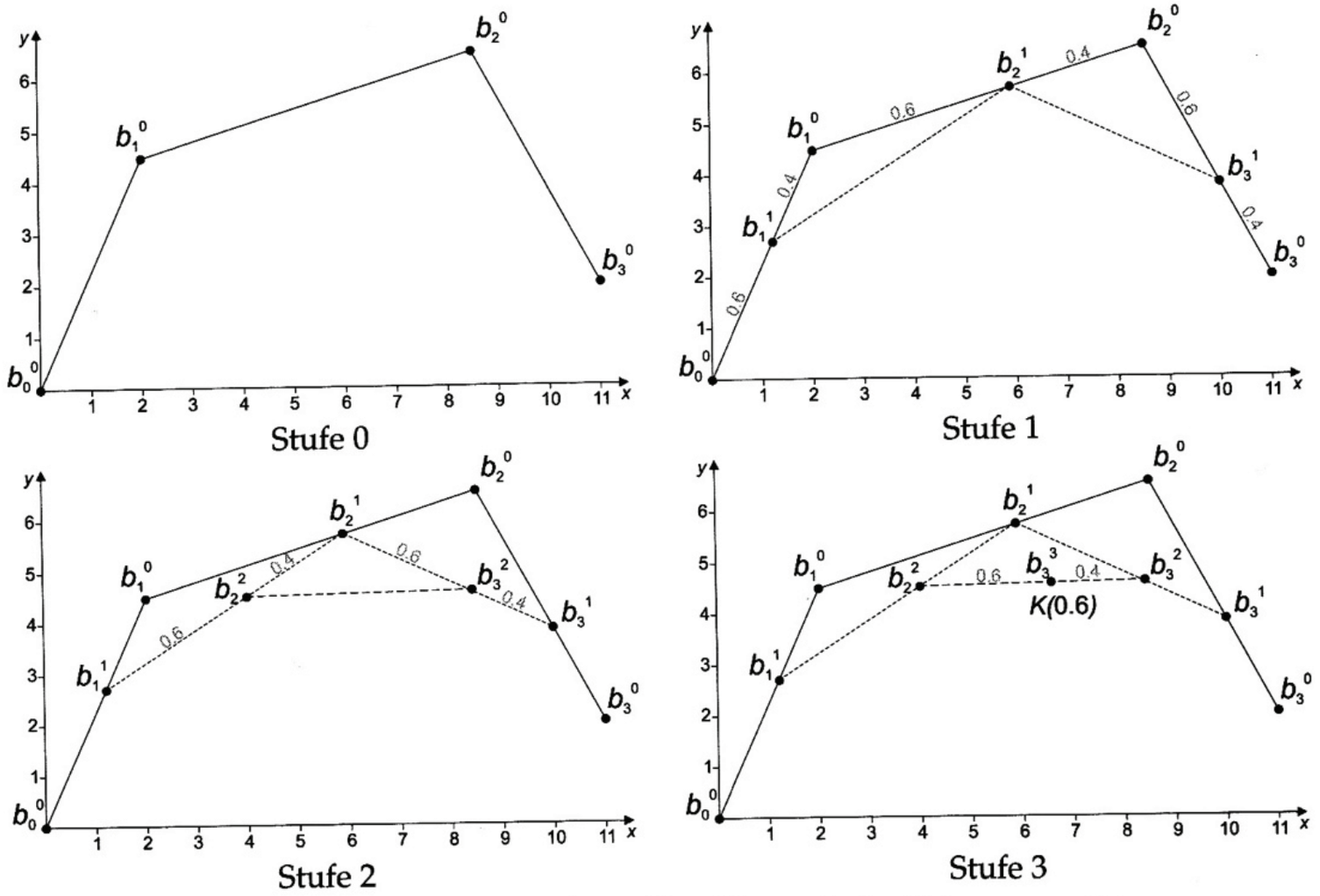




- Diesem Schema entsprechend wird der deCasteljau Algorithmus implementiert, indem die Punkte in die erste Spalte eines 2D Arrays gespeichert werden und dann sukzessive in den rechten Nachbarspalten die neuen Punkte erzeugt werden.

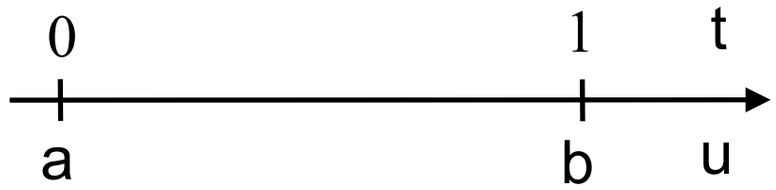
Der deCasteljau-Algorithmus

- Bemerkung: Für $t \in [0, 1]$ werden nur Konvex-Kombinationen verwendet. Das heißt, dass nicht nur die Auswertung von Potenzen und Fakultäten (bei der direkten Auswertung der Bernstein Polynome) vermieden werden, sondern auch mögliche Auslöschungs-Erscheinungen bei Subtraktionen auf Maschinenebene vermieden werden.



5.7 Bézier-Kurven in globalen Parametern

- Bisher war t immer aus dem Parameterintervall $t \in [0, 1]$, da hier die Bernstein-Polynome besonders schöne Eigenschaften haben.
- Möchte man einen Parameter u aus einem beliebigen Intervall $u \in [a, b]$ zulassen, kann man dies mit einem „Skalierungstrick“ tun, ohne die Eigenschaften der Bernstein-Polynome zu verlieren.
- Skaliere das Intervall [a,b] mit einer linearen Skalierung auf das Intervall [0,1].



Bézier-Kurven in globalen Parametern

- genauer:

$$
\begin{aligned}
u &= (1-t)\cdot a + t\cdot b \quad (*)\\
&= a - t\cdot a + t\cdot b\\
\Rightarrow\quad & u - a = t\cdot b - t\cdot a = t\cdot(b-a)\\
\Rightarrow\quad & t = \frac{u-a}{b-a} \quad (**)
\end{aligned}
$$

- Die Umskalierung ist gültig, wenn $b \neq a$, d.h. wenn das Parameterinterval nicht leer ist. Zusätzlich muss gelten $b > a$, damit die „Durchlaufrichtung“ des Parameters im Intervall nicht geändert wird.
- Notiz: Der mathematische Fachausdruck für eine solche Skalierung heißt affine Umparametrisierung.

Bézier-Kurven in globalen Parametern

- Bézier-Kurve in globalen Parametern:

$$\mathbf{b}(u) = \mathbf{b}(\underbrace{u(t)}_{*}) = \sum_{i=0}^{n} \mathbf{b}_i \cdot B_i^n(t), \quad t \in [0,1] \quad u \in [a,b]$$

- Berechnung eines Kurvenpunktes an einem Parameterwert $u \in [a,b]$:
  - Berechne $t^* = \dfrac{u-a}{b-a} \quad (**)$
  - Führe den deCasteljau Algorithmus mit t* statt t aus.

5.8 Unterteilung von Bézier-Kurven

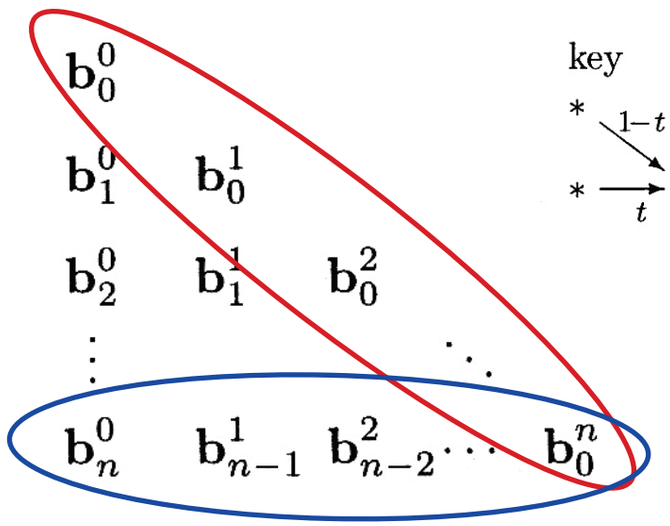
- Angenommen eine Bézier-Kurve, genauer ab jetzt ein Bézier-Kurven-Segment, b(t) vom Grad n mit Kontroll-punkten b_i (i=0,...,n) und dem Parameter-Intervall [0,1] sei vorgegeben.
- Das Kurvensegment stellt auf dem ganzen Intervall [0,1] eine polynomiale Kurve dar, insbesondere also auch auf den Teil-Intervallen [0,c] und [c,1], $c \in (0,1)$ bzw. $0 < c < 1$
- Frage: Kann für diese beiden Teil-Kurvensegmente je ein Kontroll-Polygon angegeben werden, so dass die Seg-mente unabhängig voneinander beschrieben werden können?

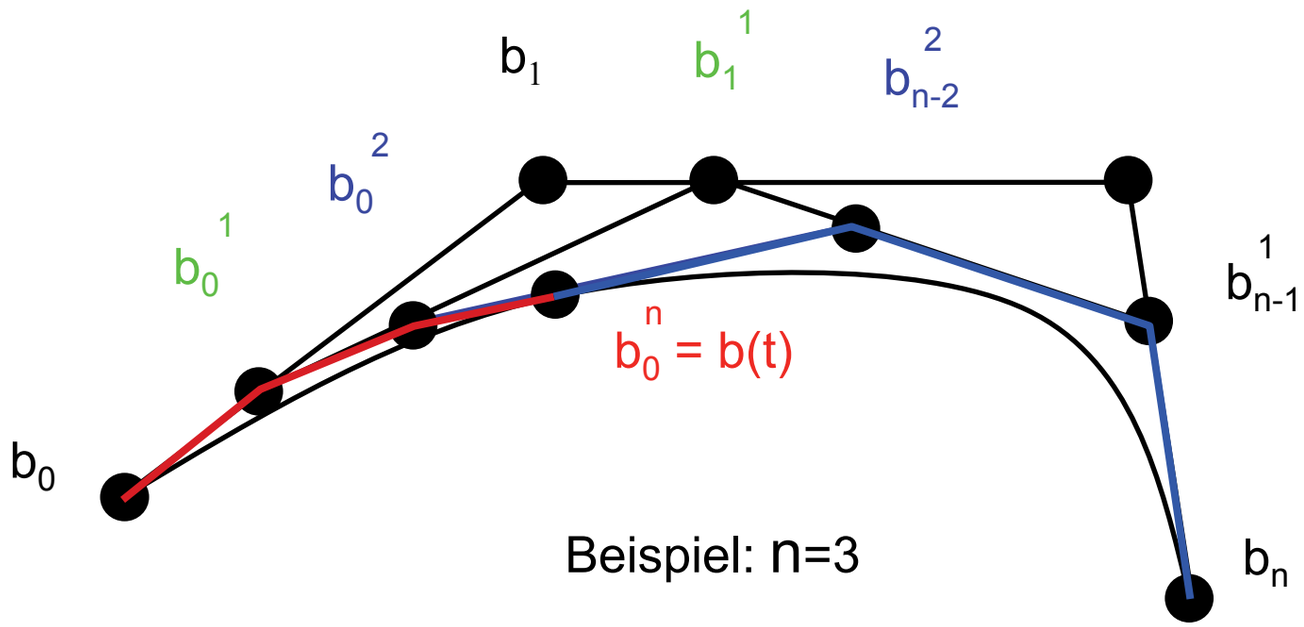
Unterteilung von Bézier-Kurven

- Die Antwort auf die Frage liegt im deCalsteljau Algorithmus.
- Dieser liefert bei der Auswertung der Kurve b(t) zum Zeitpunkt $c \in (0, 1)$ beide Kontrollstrukturen der Teilkurvensegmente gleich mit.
  - Die Kontrollstruktur des ersten Teil-Kurvensegments ergibt sich aus den Kontroll-Punkten auf der Diagonalen des deCalsteljau Dreicksschemas, also $\mathbf{b}_0^0, \mathbf{b}_1^1, \mathbf{b}_2^2, \mathbf{b}_3^3, ...$
  - Die Kontrollstruktur für das zweite Teilsegment ergibt sich aus der untersten Zeile des Dreiecksschemas rückwärts gelesen, d.h $\mathbf{b}_n^n, \mathbf{b}_n^{n-1}, \ldots, \mathbf{b}_n^1, \mathbf{b}_n^0$.

Unterteilung von Bézier-Kurven

- Anschaulich:





- Bemerkung: Die Berechnung der Bézier-Punkte über den Teil-Intervallen [0,c] und [c,1] mit Hilfe des deCasteljau-Algorithmus heißt Unterteilung.

Unterteilung von Bézier-Kurven

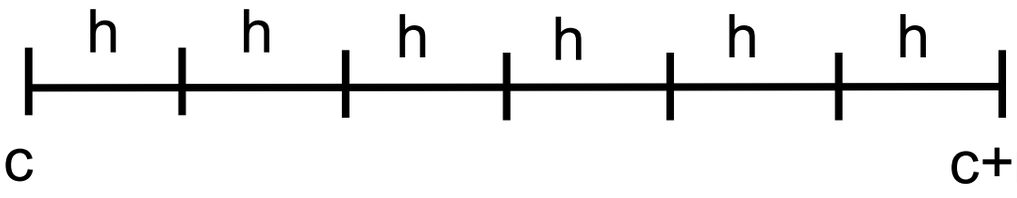
- Durch wiederholte Unterteilung erhält man die Bézier-Punkte von b(t) über einer beliebigen Anzahl aneinander grenzender Intervalle $[0,a_1],[a_1,a_2],[a_2,a_3],\ldots,[a_k,1]$. Zusammen bilden die Kontroll-Polygone über den Teilintervallen das zusammengesetzte Kontroll-Polygon von b(t) über $[0, a_1, a_2, a_3, \ldots, a_k, 1]$.
- Frage: Wozu dient die Unterteilung nun?
- Antwort:
  - um die Kurve (mit adaptiver Auflösung) zu zeichnen.
  - um effizient Schnittpunkte zwischen zwei oder mehreren Bézier-Kurven zu berechnen.

Konvergenz der Unterteilung

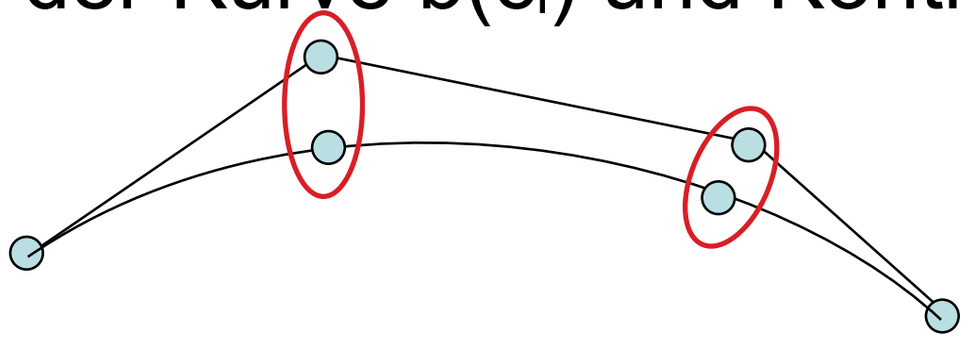
- Die zusammengesetzten Bézier-Polygone approximieren die Kurve mit wachsender Anzahl von Teilintervallen immer besser.
- Siehe Abbildung: Die Teil-Polygone liegen „näher“ an der Kurve als das ursprüngliche Kontrollpolygon.
- Genauer: Sei ein festes Intervall [a,b] gegeben. Die Frage ist, wie weit ist das Kontrollpolygon eines Teilintervalls [c,c+nh] Teilmenge von [a,b] von dem Kurventeil, der zu diesem Teilintervall gehört, entfernt?

Konvergenz der Unterteilung

- Seien $\mathbf{b}_0, \ldots, \mathbf{b}_n$ die Bézier-Punkte des Intervalls [c,c+nh] und sei c_i=c+ih i=0,...,n eine äquidistante Aufteilung des Parameter-Intervalls, also



- Das bedeutet man hat eine Korrespondenz zwischen Punkten auf der Kurve $b(c_i)$ und Kontrollpunkten b_i.



Konvergenz der Unterteilung

- Diese Korrespondenz ist sinnvoll, da der Kontrollpunkt b_i an der Stelle $b(c_i)$ auf der Kurve den größten Einfluss hat, weil genau an diesem Parameterwert das zugehörige Bernstein-Polynom sein Maximum hat.

- Satz: Für die Abstände $\| \underbrace{\mathbf{b}(c_i)}_{Kurve} - \underbrace{\mathbf{b}_i}_{KP} \|$ korrespondierender Punkte gilt:

ohne Beweis!

$$\max_i \|\mathbf{b}(c_i) - \mathbf{b}_i\| \leq M \cdot h^2$$

Grund für die Effizienz des Verfahrens

wobei M eine Konstante ist, die nicht von c abhängt.

Konvergenz der Unterteilung

- Erläuterung: Wenn man sukzessive unterteilt, d.h. das Parameter-Intervall immer kleiner macht, diese äquidistant in n Teile aufteilt, und dann die korrespondierenden Abstände betrachtet, so sind diese quadratisch kleiner als die Verkleinerung der Parameter-Intervalle (Verkleinerung Intervall heißt Verkleinerung von h).
- Praktischer Nutzen: Es gibt Abschätzungen für die Konstante M, die Abschätzung hängt von der Form des Kontroll-Polygons ab und vom Grad n der Kurve. Damit kann die notwendige Anzahl an Unterteilungen für eine gewisse Genauigkeit (z.B. sub-Pixel) im Vorhinein berechnet werden.

Ableitungen einer Bézier-Kurve

- Für höhere Ableitungen ergibt sich die Rekursionsformel

$$\frac{d^{(p)}}{dt^p}\mathbf{b}(t) = \frac{n!}{(n-p)!}\sum_{i=0}^{n-p}\Delta^p\mathbf{b}_i B_i^{n-p}(t)$$

mit $\Delta^p b_i := \Delta^{p-1} b_{i+1} - \Delta^{p-1} b_i$ und $\Delta^0 b_i = b_i$

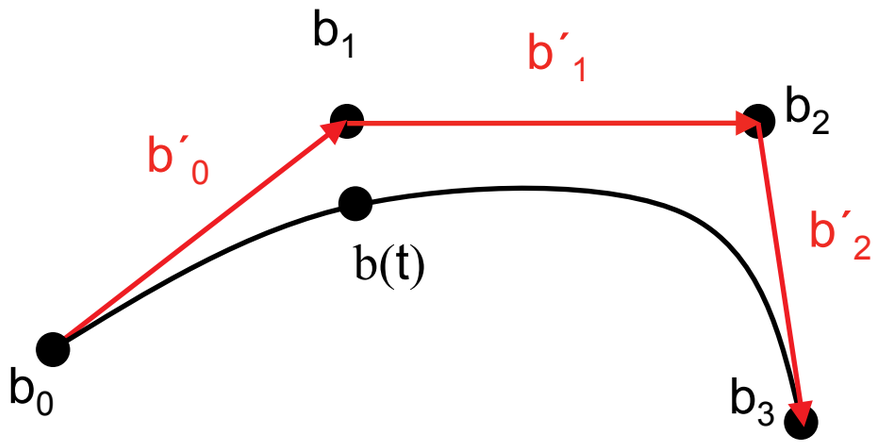
- In globalen Parametern kommt wegen der Kettenregel noch die Ableitung der Umparametrisierung hinzu:

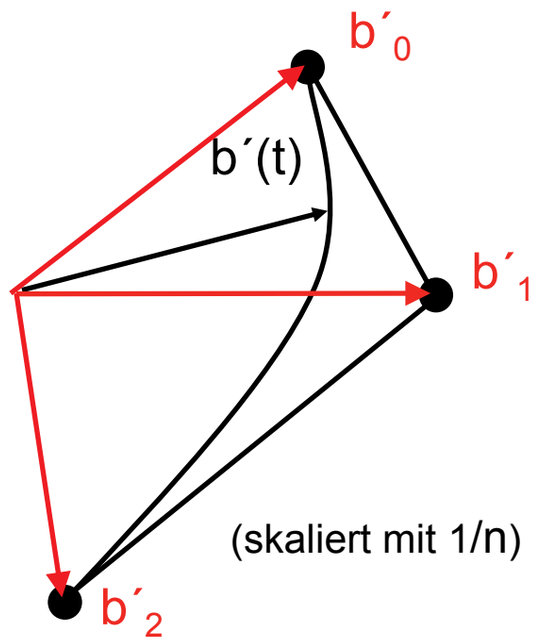
$$\frac{d^{(p)}}{du^p}\mathbf{b}(u) = \frac{n!}{(b-a)^p \cdot (n-p)!}\sum_{i=0}^{n-p}\Delta^p\mathbf{b}_i B_i^{n-p}\left(\frac{t-a}{b-a}\right)$$

Ableitungen einer Bézier-Kurve

- Ableitung: Die Ableitung einer Bézier-Kurve kann als Bézier-Kurve vom Grad n-1 dargestellt werden, mit den Bézier-Punkten

$$\mathbf{b}'_i = n(\mathbf{b}_{i+1} - \mathbf{b}_i), \quad i = 0, \ldots, n-1$$





d.h.

$$\frac{d}{dt}\mathbf{b}(t) = n \cdot \sum_{i=0}^{n-1} (\mathbf{b}_{i+1} - \mathbf{b}_i) B_i^{n-1}(t)$$

Ableitungen am Rand

- Ableitungsformel sehr kompliziert, aber
  - in den Randpunkten hängen die p-ten Ableitungen nur von p+1 Kontrollpunkten ab:

$$b'(0) = n(b_1 - b_0)$$

$$b''(0) = n(n-1)(b_2 - 2b_1 + b_0)$$

$$b'''(0) = n(n-1)(n-2)(b_3 - 3b_2 + 3b_1 - b_0)$$

$$b'(1) = n(b_n - b_{n-1})$$

$$b''(1) = n(n-1)(b_n - 2b_{n-1} + b_{n-2})$$

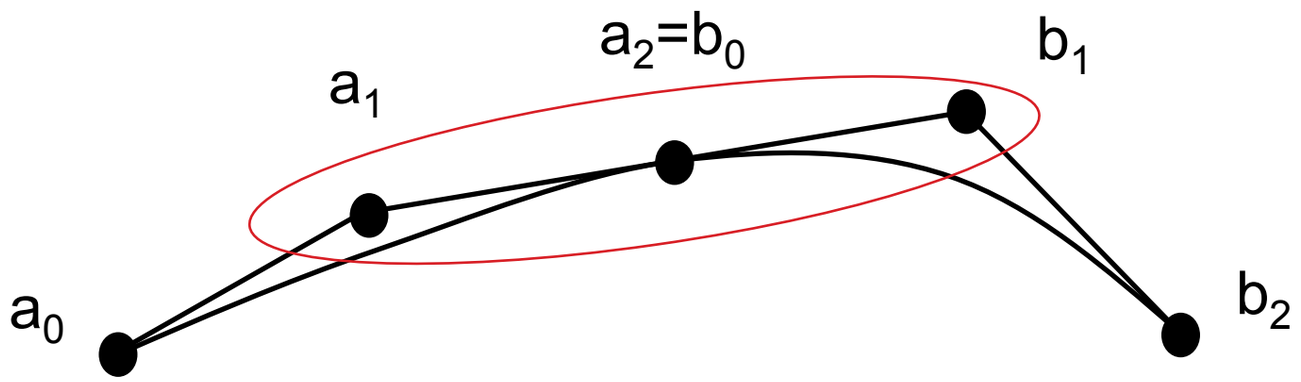
$$b'''(1) = n(n-1)(n-2)(b_n - 3b_{n-1} + 3b_{n-2} - b_{n-3})$$

5.11 Splines

- Definition: (C^k-Stetigkeit)
  - Eine Funktion f(t) ist C^k-stetig ($k \geq 0$), wenn die Funktion und ihre ersten k Ableitungen stetig sind.
  - C^k $[t_0,t_n]$ ist die Klasse der C^k-stetigen Funktionen auf dem Intervall $[t_0 ,t_n]$.
- Definition: (Spline) Sei $\tau = \{t_0,\ldots,t_n\}$ ein Knoten-Vektor mit reellen Knoten $t_i < t_{i+1}$. Eine Funktion S heißt Spline vom Grad k (von der Ordnung k+1), wenn gilt:
  - S ist ein Polynom vom Grad k in jedem Teilintervall $[t_i,t_i+1]$
  - S ist C^{k-1}-stetig auf $[t_0, t_n]$.

Bézier-Splines

- Im Fall zweier Bézier-Segmente mit uniformer Parametrisierung (d.h. beide Segmente haben das gleiche Parameterintervall) erhalten wir einen C^1-Übergang:

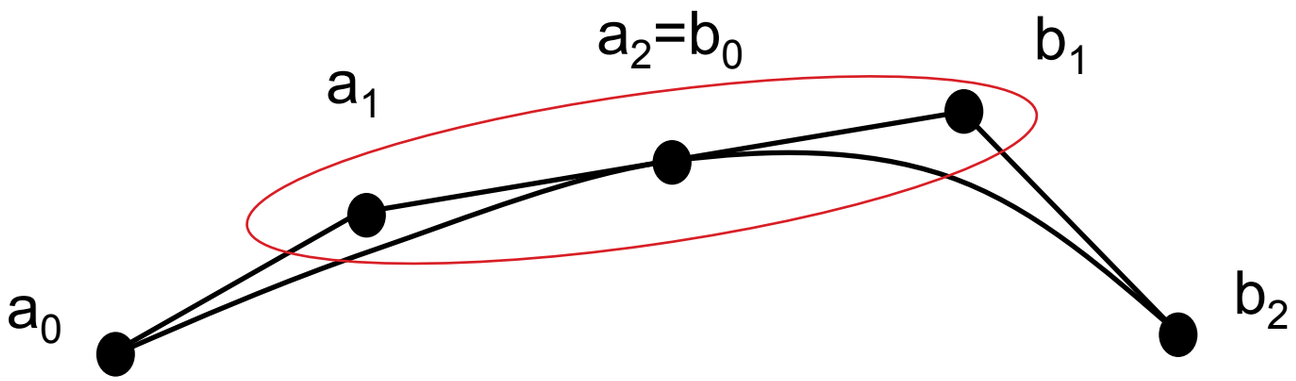


$$a_2 = b_0 = (a_1 + b_1)/2$$

- Die Segmenttrennpunkte a_2,b_0 sind also redundant.
- rot: Punkte fest , alle weiteren Punkte frei beweglich.

Bézier-Splines

- Analog erhält man eine Kurve mit stetiger Tagentialrich-tung, falls die rot umrandeten Punkte auf einer Geraden iegen, aber in beliebigem Teilverhältnis

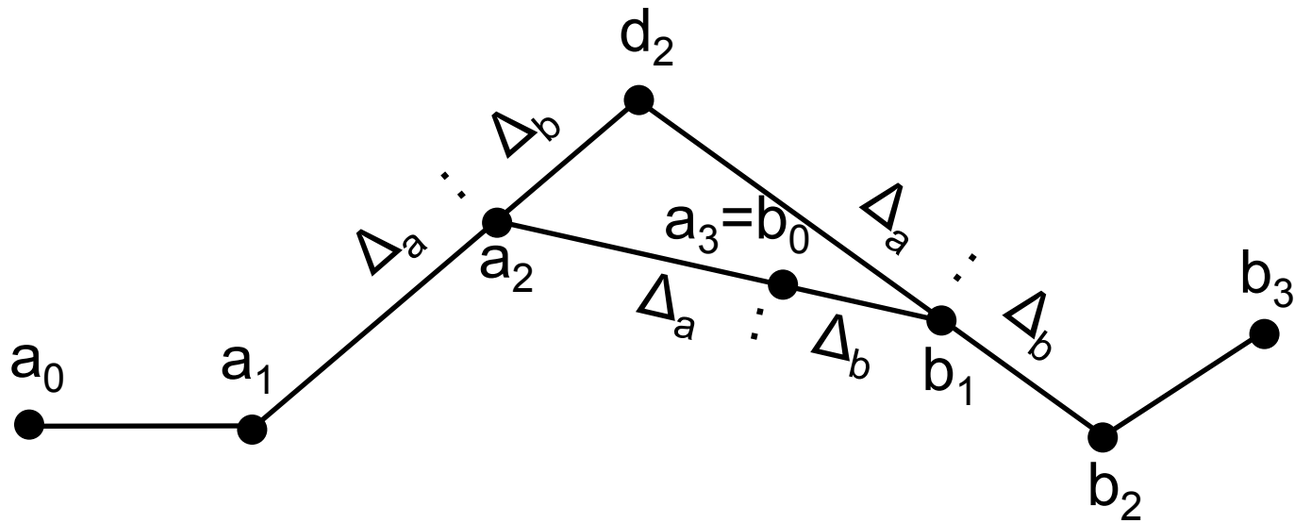


$$\mathbf{a}_2 = \mathbf{b}_0 = \beta \cdot \mathbf{a}_1 + (1-\beta)\mathbf{b}_1, \quad 0 < \beta < 1$$

- Hier ist jetzt also nur die Richtung der Ableitung aber nicht notwendig der Betrag gleich.
- rot: Punkte fest , alle weiteren Punkte frei beweglich.

Bézier-Splines

- Für den C^2-Übergang ergibt sich folgende Konstruktion:



Δ_a, Δ_b: Längen der Parameterintervalle der Kurven.

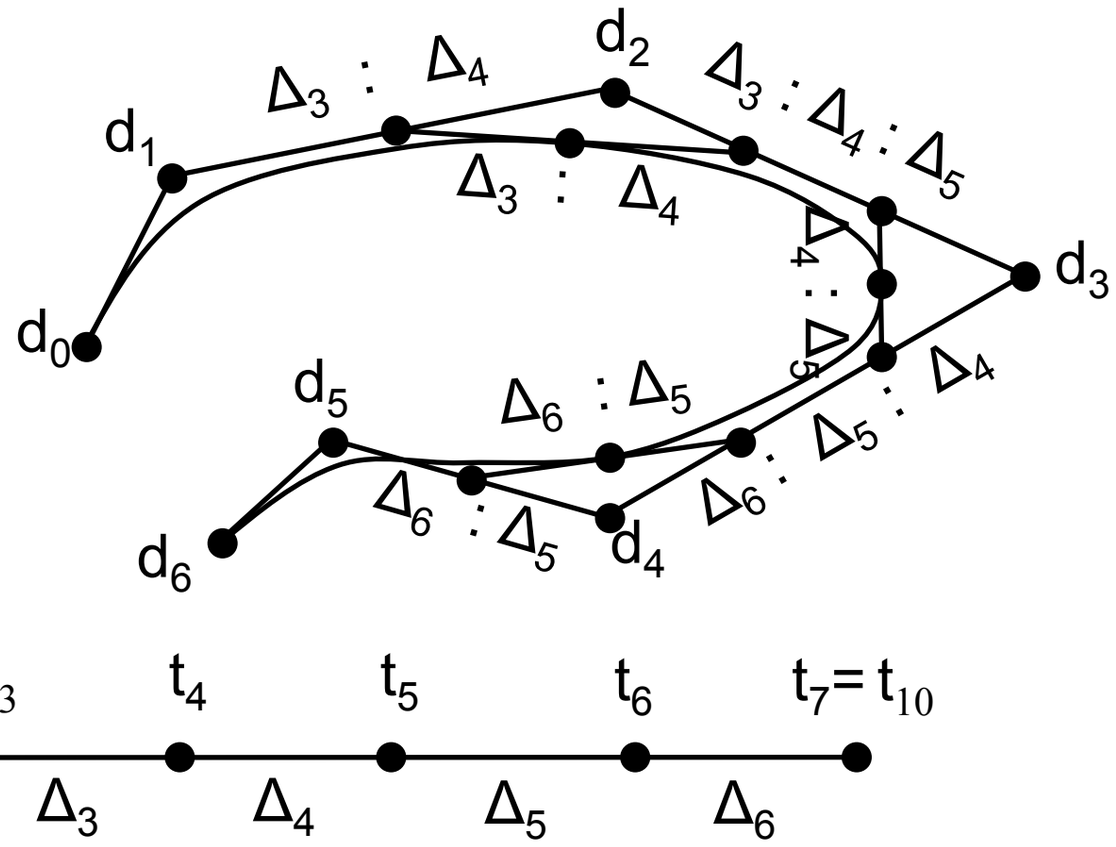
- Die Darstellung dieser zwei Segmente ist also durch die Punkte a_0, a_1, d_2, b_2, b_3 eindeutig festgelegt.
- Fügt man ein weiteres Bézier-Segment hinzu, so werden die Punkte b_2, b_3 durch einen weiteren Hilfspunkt d_3 ersetzt.

Bézier-Splines: Fazit

- Bézier-Segmente können durch eine geometrische Konstruktion zu Splines verbunden werden.
  - Vorteil: Einzelne Kurven-Segmente können (teilweise) unabhängig voneinander verändert werden. Damit ist tatsächlich lokale Kontrolle möglich.
- Obwohl die Konstruktion nur auf Teilverhältnissen beruht, ist sie doch relativ umständlich zu handhaben.
  - Grund: insbesondere deshalb, weil immer bekannt sein muss, welche Punkte festgelegt sind und welch frei beweglich sein dürfen.

Spline-Kurven

- Frage: Wenn sukzessive die Übergänge durch die d`s festgelegt werden, reicht es dann nicht die ganze Kurve durch die d`s und die Parameterintervalle zu definieren?
- Vorteil: Die Übergänge zwischen den Teilkurven wären automatisch vorhanden.
- Beispiel: Kubische Spline-Kurve mit Kontrollpunkten $d_0,\ldots,d_6$.



5.12 Die B-Spline Basis

- Bei der Konstruktion von Splines haben wir bisher polynomiale Basisfunktionen (Bernstein-Polynome) für die Darstellung der einzelnen Segmente betrachtet. Dies führt zur Konstruktion von Übergangsbedingungen an den Segment-Trennstellen.
- Ein eleganterer Ansatz zur Konstruktion von Splinekurven besteht darin, C^{k-2}-stetige Basisfunktionen aus mehreren Segmenten der Ordnung k zu konstruieren. Bei Verwendung dieser Basis Splines (B-Splines) entfallen die Übergangsbedingungen und es werden zudem weniger Kontrollpunkte (Koeffizienten) benötigt.

B-Splines

- Eine offene B-Splinekurve der Ordnung k (vom Grad k-1) ist gegeben durch
  - m+1 Kontrollpunkte $d_0,...,d_m$ (de Boor-Punkte) und
  - m+k+1 Knoten $x_0 \leq x_1 \leq ... \leq x_{m+k}$ (Knotenvektor).
- Das Definitionsgebiet der B-Spline Kurve ist das Intervall $[x_{k-1}, x_{m+1}]$, welches m-k+2 Segmente enthält. Die übrigen Knoten bestimmen das Verhalten der Kurve an den Rändern.
- Für das erste Splinesegment benötigt man k de Boor-Punkte, für jedes weitere Segment nur einen zusätzlichen de Boor-Punkt.

5.13 Der deBoor-Algorithmus

- Analog zum de Casteljau-Algorithmus für Bézier-Kurven ermöglicht der de Boor-Algorithmus die effiziente Auswertung von B-Splinekurven (ohne die einzelnen Bézier-Segmente herzuleiten).
- Zu gegebenem Parameter t bestimmt man zunächst den Index r, so dass $t \in [x_r , x_{r+1})$.
- Der zugehörige Punkt $f (t)$ der Splinekurve wird dann aus den de Boor-Punkten $d_{r-k+1},\ldots,d_r$ wie folgt ermittelt

Der deBoor-Algorithmus

- De Boor-Algorithmus

Für $t \in [x_r , x_{r+1})$ berechnet sich die B-Splinekurve aus

$$d_i^0 = d_i \quad (i = r-k+1,\ldots,r)$$

$$d_i^j = (1-\alpha_i^j)d_{i-1}^{j-1} + \alpha_i^j d_i^{j-1} \quad (i = r-k+j+1,\ldots,r;\ \ j = 1,..,k-1)$$

$$f(t) = d_r^{k-1}$$

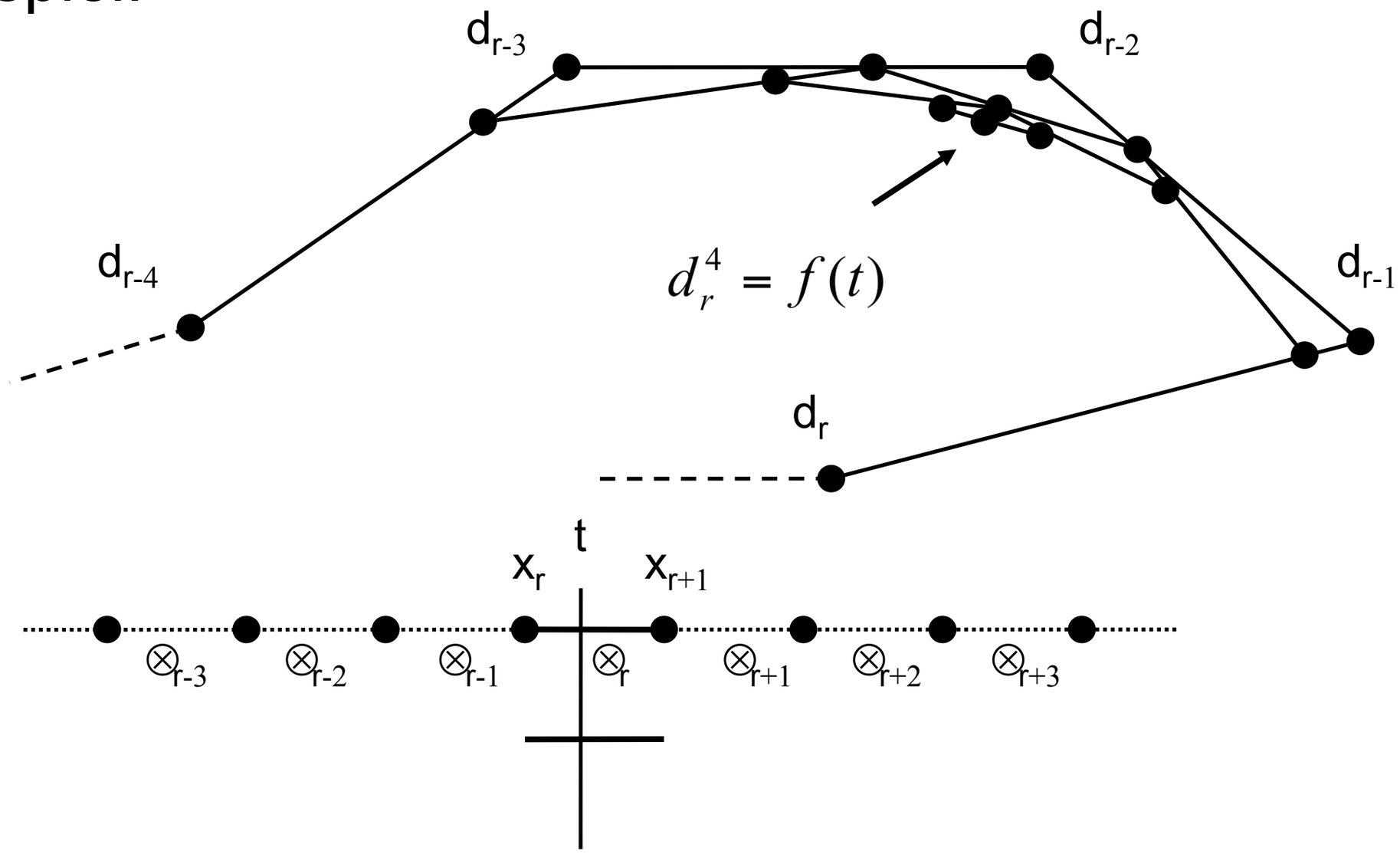
$$\alpha_i^j = \frac{t - x_i}{x_{i+k-j} - x_i}$$

$$\begin{array}{llll} d_{r-k+1}^0 & & & \\ d_{r-k+2}^0 & d_{r-k+2}^1 & & \\ \vdots & \vdots & \ddots & \\ d_r^0 & d_r^1 & \cdots & d_r^{k-1} = f(t) \end{array}$$

Schema des de Boor-Algorithmus

Der deBoor-Algorithmus

- Beispiel:



http://i33www.ibds.uni-karlsruhe.de/applets/mocca/html/noplugin/inhalt.html

Rekursion der B-Splines

- Aus dem de Boor-Algorithmus ergeben sich die Basis-funktionen $N_{i,k}(t)$ (B-Splines) zu den einzelnen de Boor-Punkten d_i (analog zu den Bernstein-Polynomen für Bézier-Kurven).

- Rekursionsformel von de Boor und Cox

$\tau = (x_i)_{i \in IN_0}$ sei eine nichtfallende Folge von Knoten.

$$N_{i,1}(t) := \begin{cases} 1 & x_i \le t < x_{i+1} \\ 0 & \text{sonst} \end{cases}$$

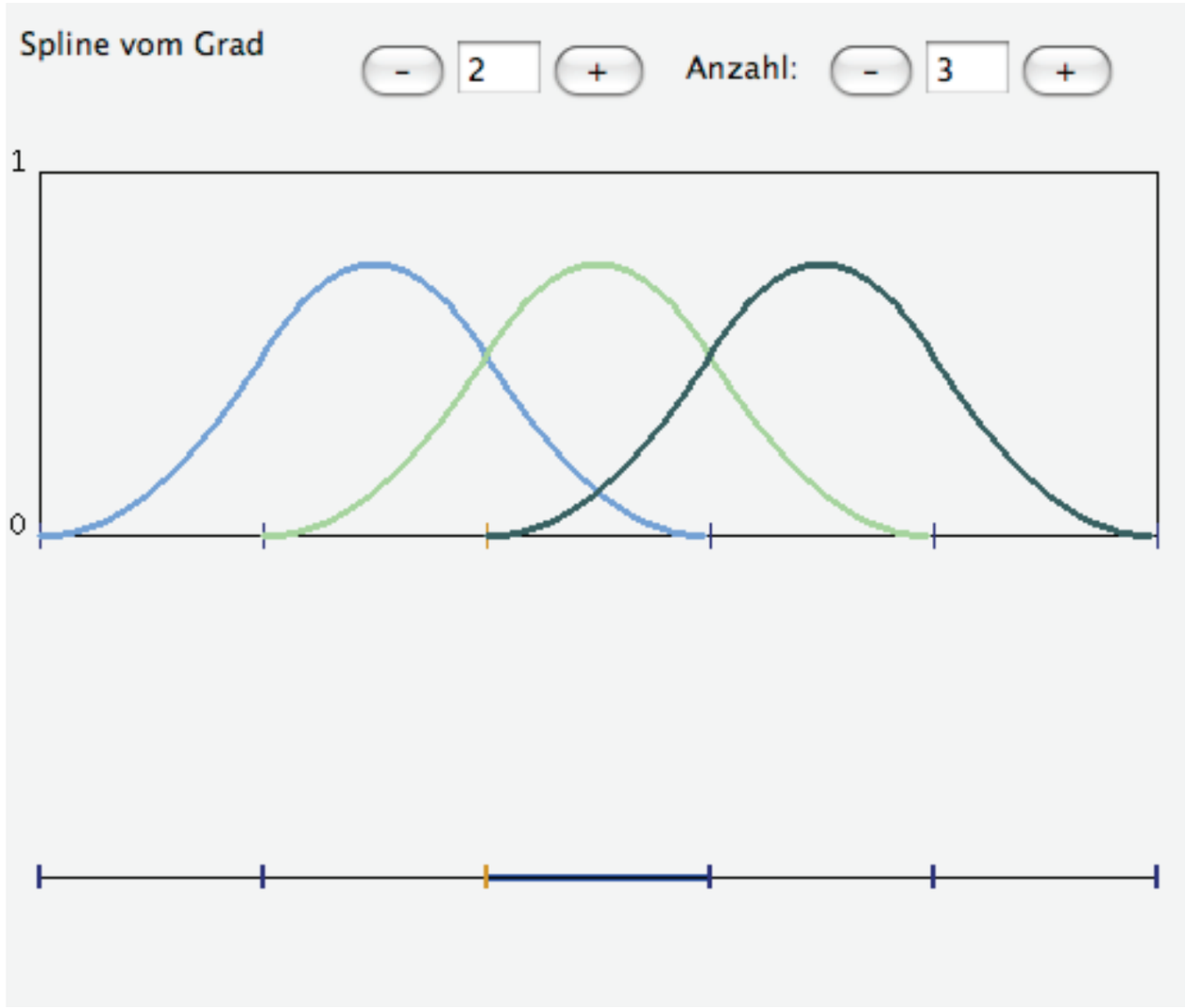
$$N_{i,k}(t) := \left(\frac{t - x_i}{x_{i+k-1} - x_i} \right) \cdot N_{i,k-1}(t) + \left(\frac{x_{i+k} - t}{x_{i+k} - x_{i+1}} \right) \cdot N_{i+1,k-1}(t)$$

B-Splines

- B-Splines vom Grad 2

$$N_{0,3} = \begin{cases} \frac{t^2}{2} & [0,1) \\ \frac{(2-t)\times t + (3-t)\times(t-1)}{2} & [1,2) \\ \frac{(3-t)^2}{2} & [2,3) \\ 0 & [3,4) \end{cases}$$

$$N_{1,3} = \begin{cases} 0 & [0,1) \\ \frac{(t-1)^2}{2} & [1,2) \\ \frac{(3-t)\times(t-1) + (4-t)\times(t-2)}{2} & [2,3) \\ \frac{(4-t)^2}{2} & [3,4) \end{cases}$$



B-Spline-Kurven

- Mit Hilfe dieser Basis Splines erhalten wir nun folgende Darstellung:
- Definition: B (Basis)-Splinekurven
  - Das Kontrollpolygon $d_0,...,d_m$, die Ordnung k (Grad k-1) und die m+k+1 Knoten $x_0 \leq x_1 \leq ... \leq x_{m+k}$ definieren eine offene B-Splinekurve, gegeben durch

$$f(t) := \sum_{i=0}^{m} \mathbf{d}_i \cdot N_{i,k}(t), \quad t \in [x_{k-1}, x_{m+1}]$$

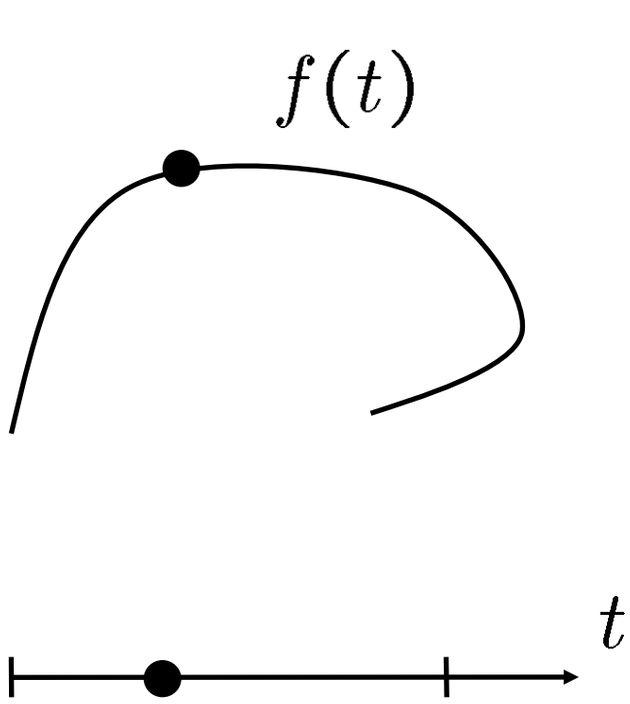
  - Es sind auch geschlossene Kurven möglich, wenn die Knoten modulo benutzt werden.

Splines in B-Spline Darstellung

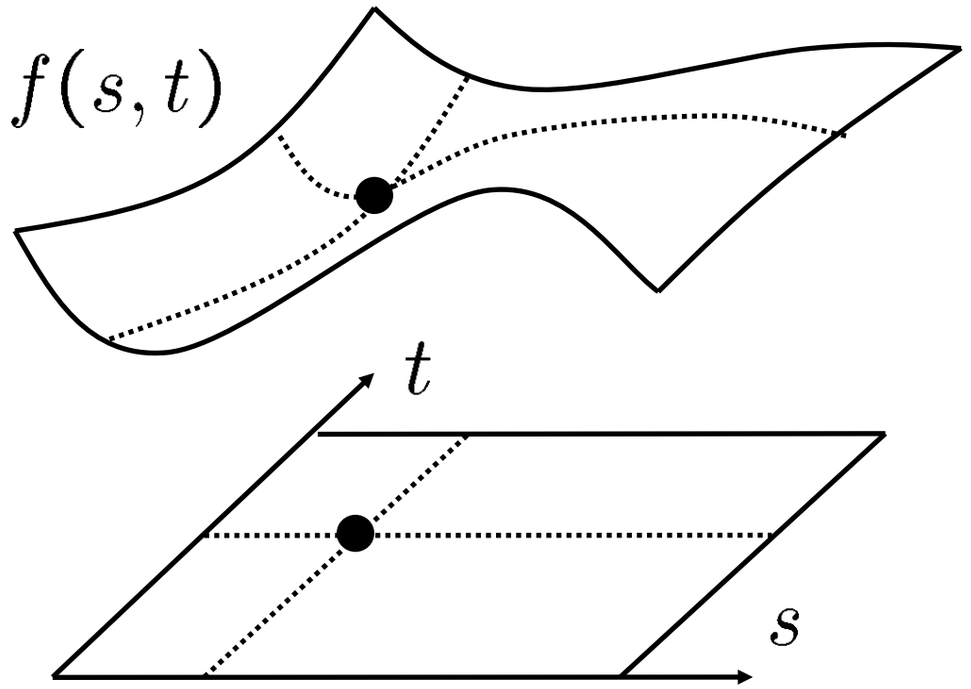
- Satz: Convex-Hull und Variation-Diminishing Property
  - Jede Gerade schneidet eine ebene B-Splinekurve nicht öfter als das zugehörige B-Spline-Polygon.
  - Die B-Splinekurve liegt in der konvexen Hülle des zugehörigen B-Spline-Polygons. Genau genommen liegt jeder Kurvenpunkt in der konvexen Hülle von k lokalen Kontrollpunkten.
- Beweis: ohne
- Bemerkung:
  - Liegen k aufeinander folgende Kontrollpunkte auf einer Geraden, so enthält auch die Kurve ein Geradensegment.
  - Eine Kurve in B-Spline Darstellung kann in eine stückweise Bezier-Darstellung umgerechnet werden.

5.15 Tensorprodukt Flächen-Darstellungen

- Tensorprodukt-Flächen sind parametrische Flächen, welche durch die eindimensionalen Basisfunktionen zweier Kurven-Darstellungen definiert sind.



Parameter-Kurve



Parameter-Fläche

5.15 Tensorprodukt Flächen-Darstellungen

- Aus zwei beliebigen Kurvendarstellungen

$$g(s) = \sum_{i=0}^{m} a_i \phi_i(s), \qquad h(t) = \sum_{j=0}^{n} b_j \psi_j(t),$$

- mit den Basen $\{\phi_i\}_{i=0}^{m}$ und $\{\psi_j\}_{j=0}^{n}$
- gewinnt man eine Tensorprodukt-Fläche:

$$f(s,t) = \sum_{i=0}^{m} \sum_{j=0}^{n} c_{ij} \phi_i(s) \psi_j(t)$$

$$= \sum_{j=0}^{n} c_j(s) \psi_j(t) \quad \text{mit} \quad c_j(s) = \sum_{i=0}^{m} c_{ij} \phi_i(s)$$

wobei c_{ij} die Koeffizientenmatrix der Fläche ist.

5.15 Tensorprodukt Flächen-Darstellungen

- Diese Konstruktion entspricht dem Auswerten einer Kurvenschar nach s:

$$c_j(s) = \sum_{i=0}^{m} c_{ij}\phi_i(s)$$

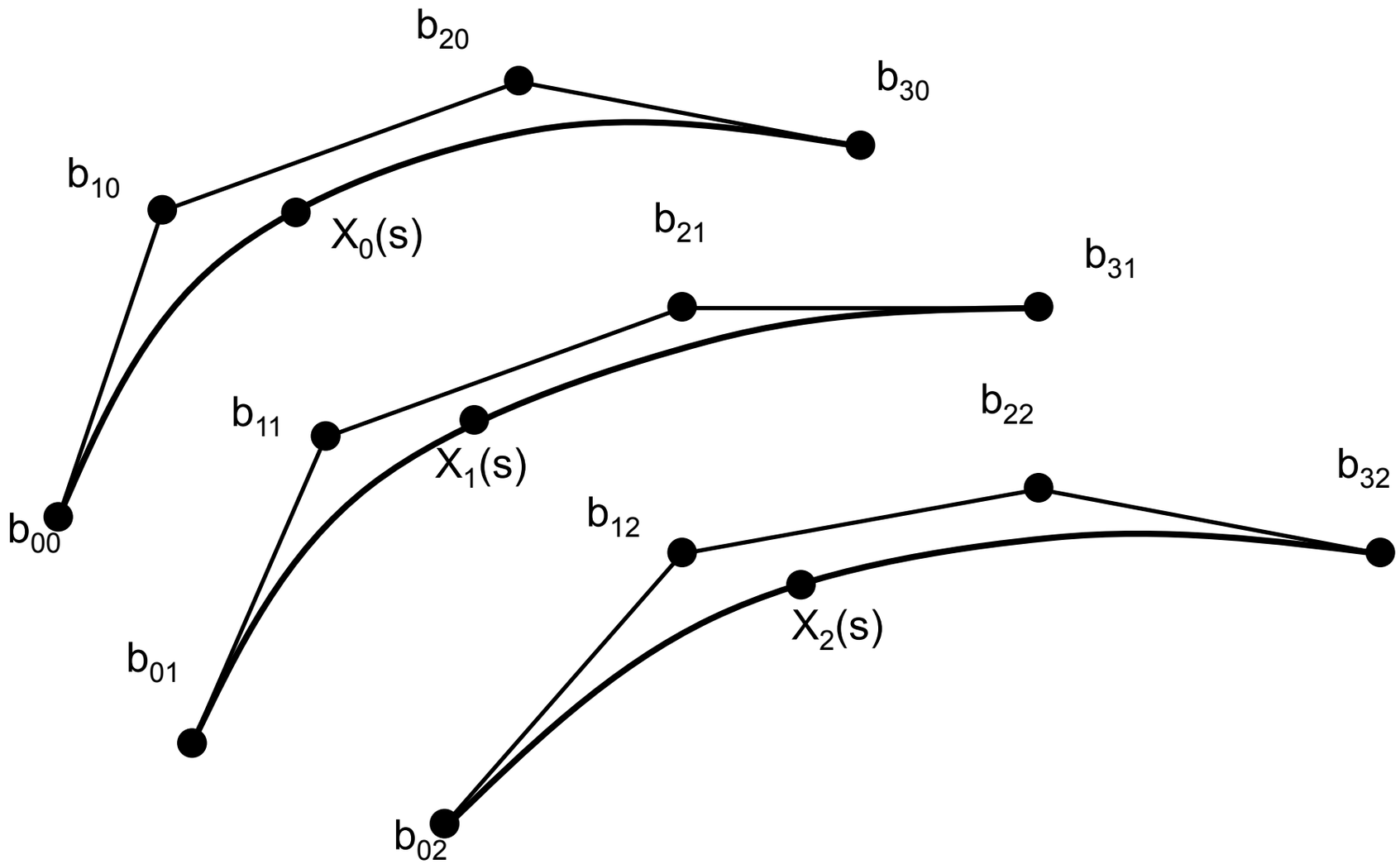
- Die daraus gewonnenen Koeffizienten $c_j(s)$ liefern eine Flächen-kurve (Isoparameterlinie) nach t :

$$f(s,t) = \sum_{j=0}^{n} c_j(s)\psi_j(t)$$

- Natürlich kann auch erst eine Kurvenschar nach t und dann eine Flächenkurve nach s ausgewertet werden.

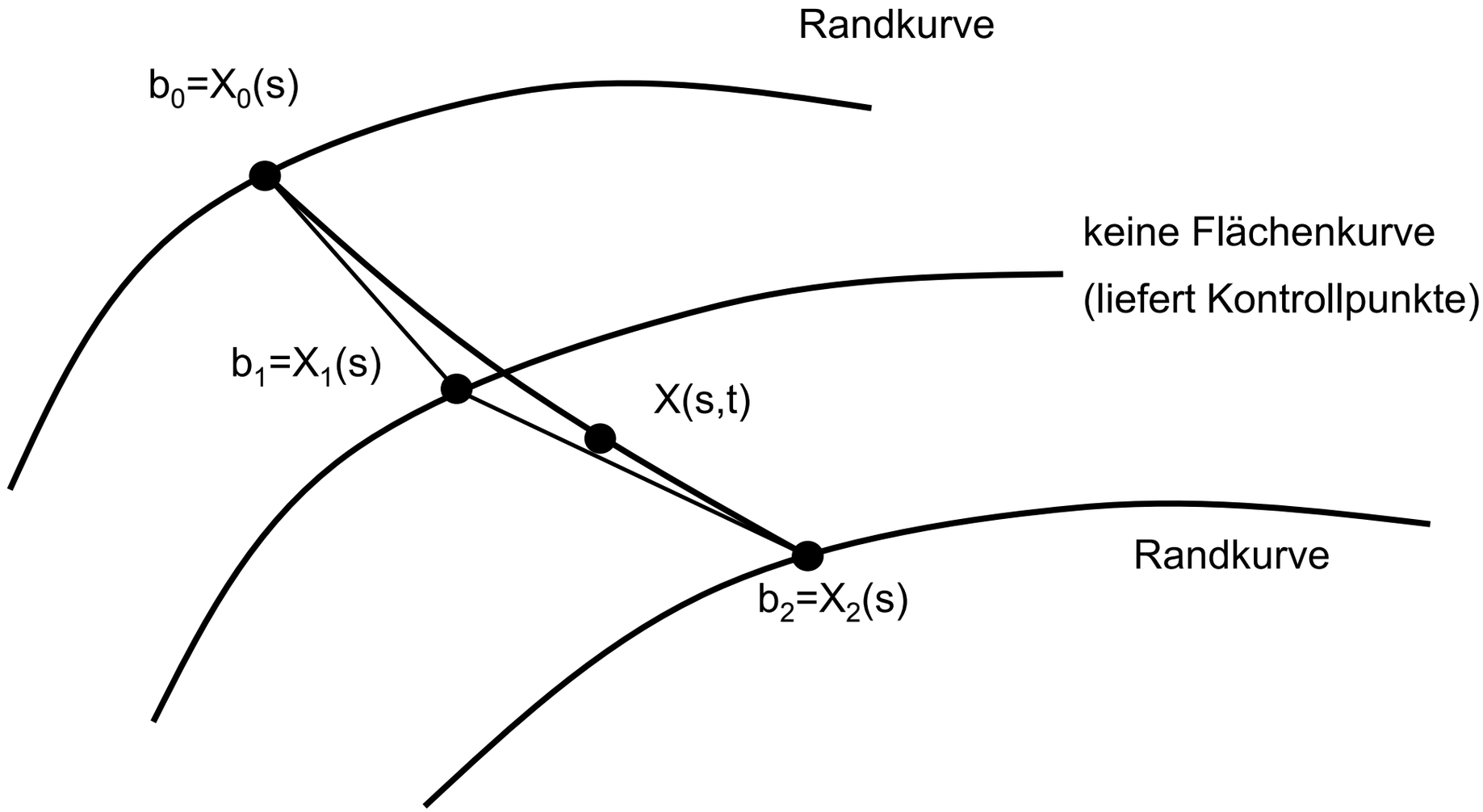
Tensorprodukt Bézier-Flächen

- Beispiel (1/2):

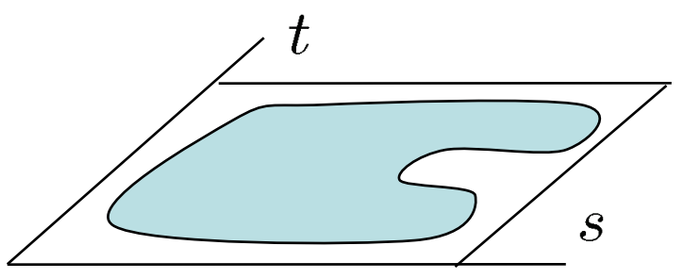


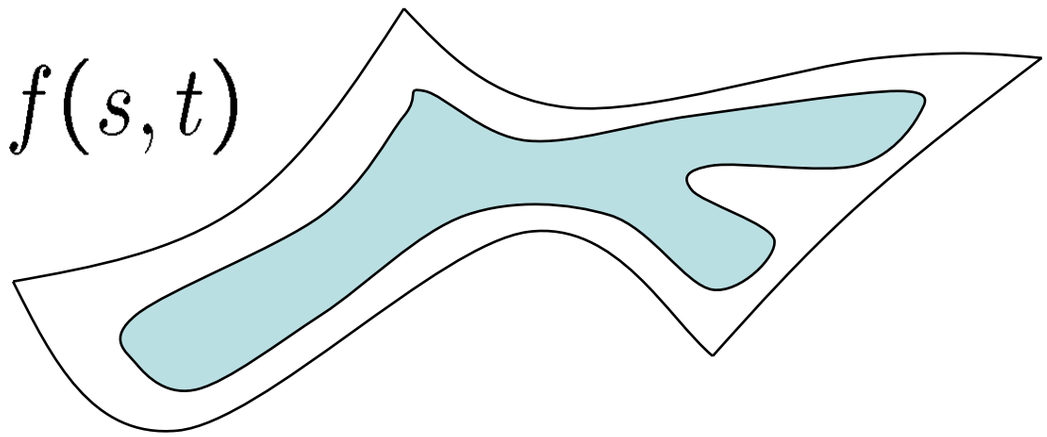
Tensorprodukt Bézier-Flächen

- Beispiel (2/2):



Trimming





Getrimmte Freiform-Fläche

Zusammenfassung

- Getrimmte NURBS-Flächen werden z.B. in CAD-Systemen eingesetzt, um die Oberflächensegmente von Solids zu repräsentieren (Boundary-Representation, B-Rep). Bei der Durchführung boolescher Operationen werden die Flächen der beteiligten Solids miteinander verschnitten. Die dabei entstehenden Teilflächen sind getrimmte Versionen der Ursprungsflächen, wobei die Trimmkurven (im Definitionsgebiet) den Schnittkurven (auf der Fläche) entsprechen.