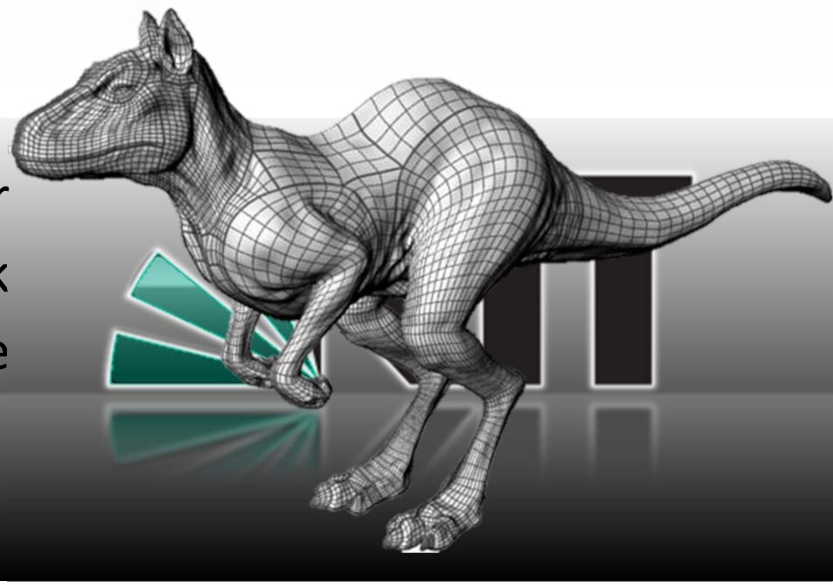
Computergrafik

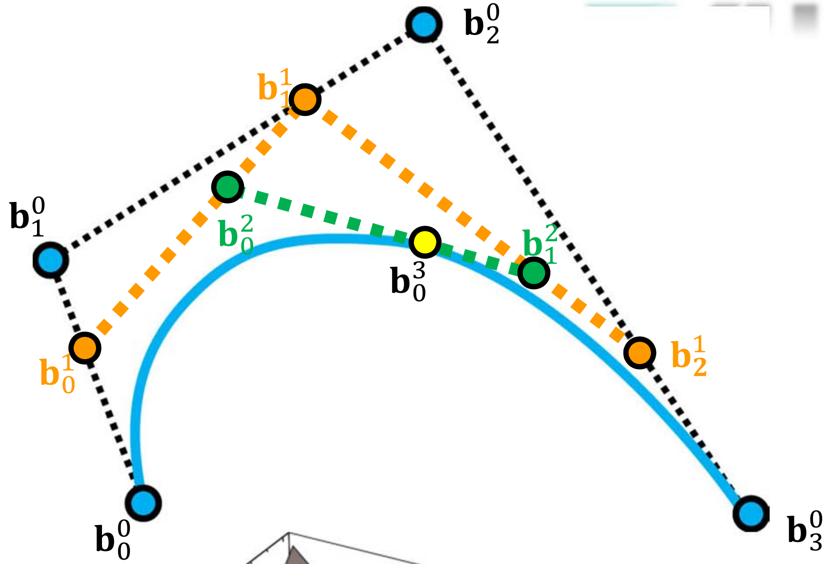
Vorlesung im Wintersemester 2014/15
Kapitel 9: Kurven und Flächen

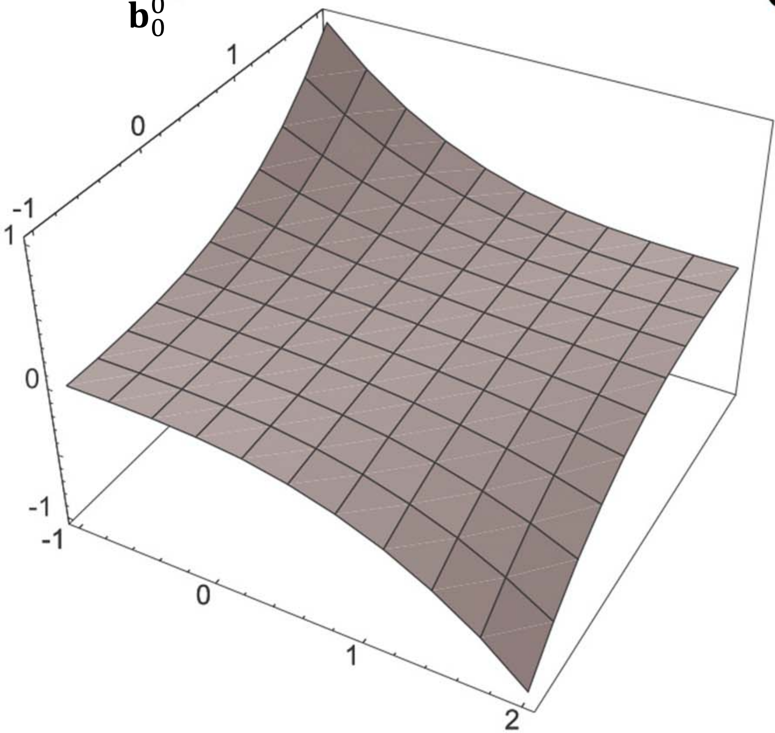


Prof. Dr.-Ing. Carsten Dachsbacher
Lehrstuhl für Computergrafik
Karlsruher Institut für Technologie

Kurven und Flächen – Inhalt

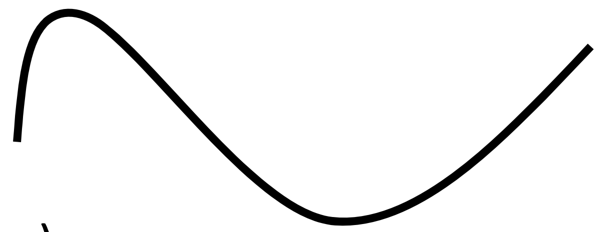
- Polynomkurven
  - Bernstein Polynome
- Bézierkurven
  - de Casteljau-Algorithmus
- Béziersplines, B-Splines
- Tensorproduktflächen

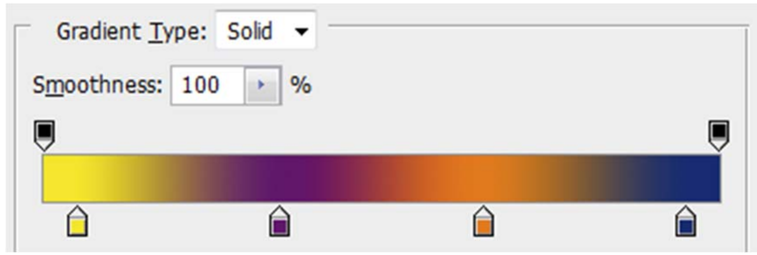


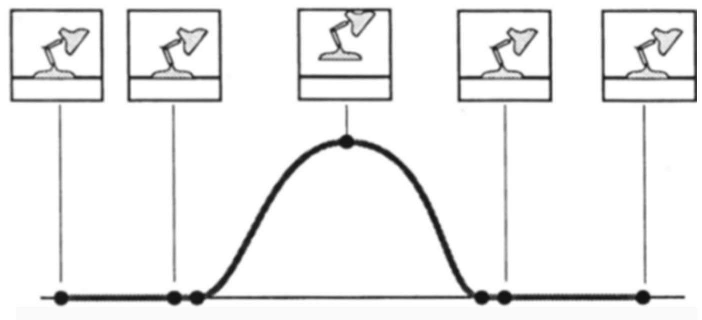


Kurven und Flächen

- Kurven in der Ebene oder in 3D
  - Spline = glatte, zusammengesetzte Kurven
- Anwendungen
  - 2D Illustration (z.B. Inkscape, Adobe Illustrator)
  - Schriften
  - 3D Modeling (Rotationskörper)
  - Farbverläufe
  - Animation, Trajektorien, Keyframe-Interpolation

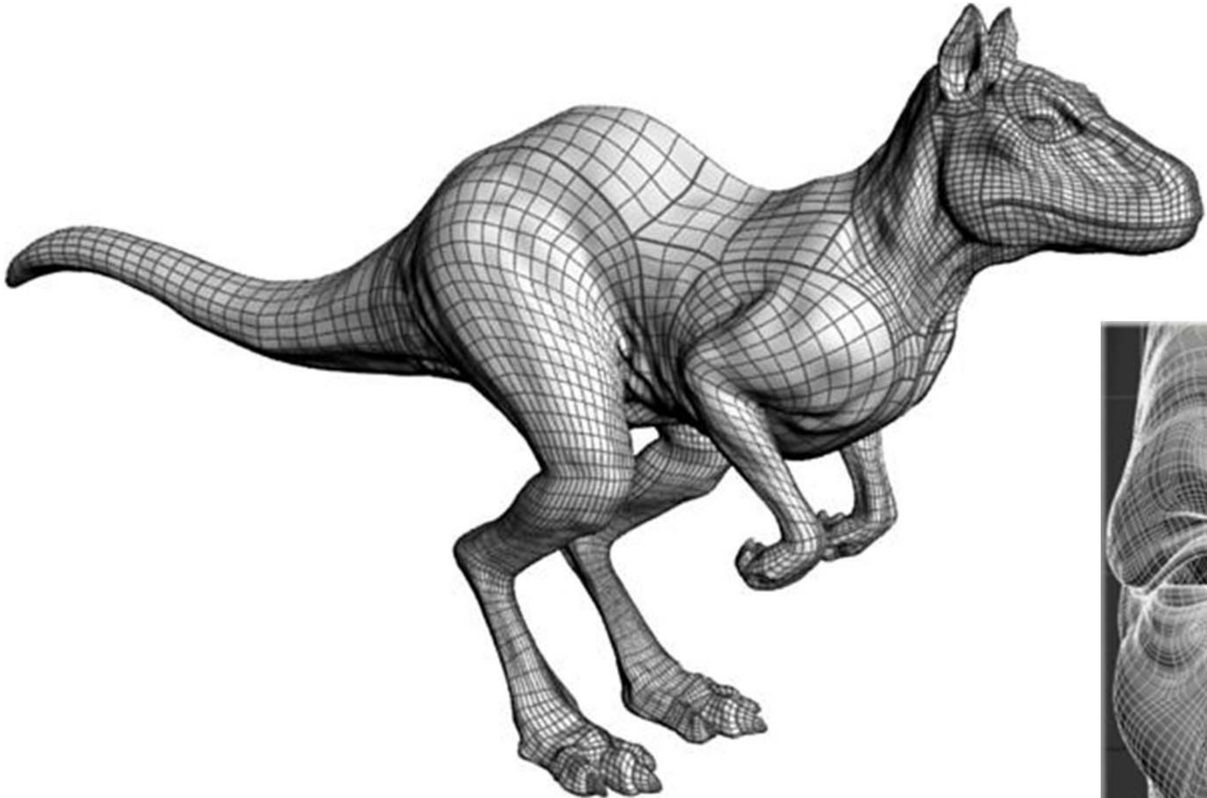


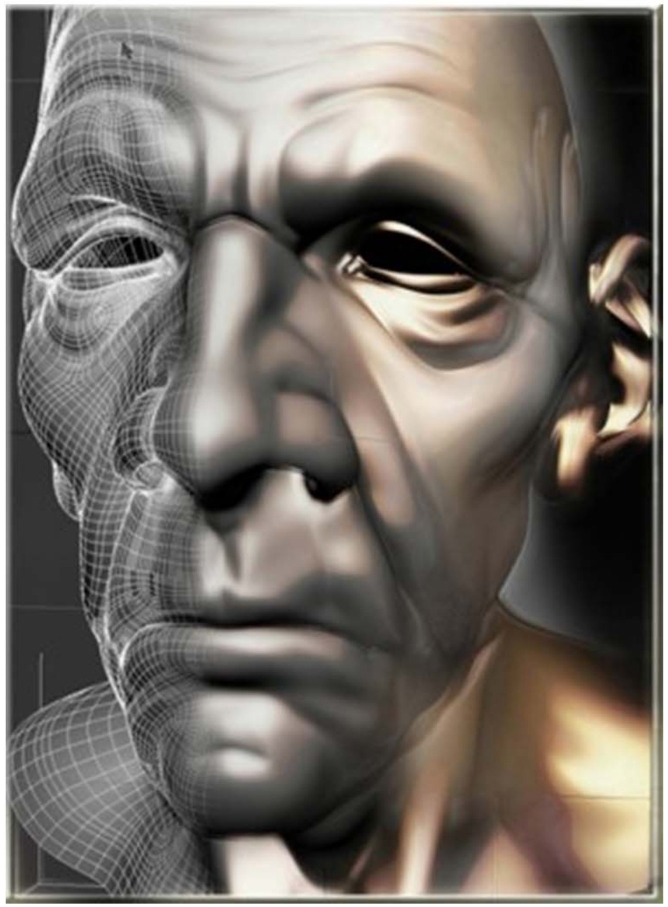




ACM © 1987 “Principles of traditional animation applied to 3D computer animation”

Kurven und Flächen



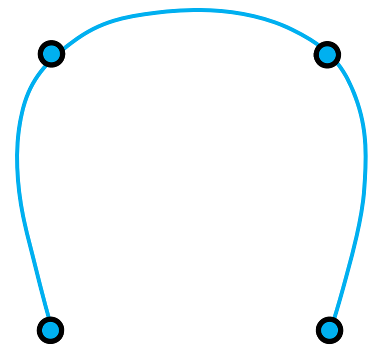


Kurven und Splines

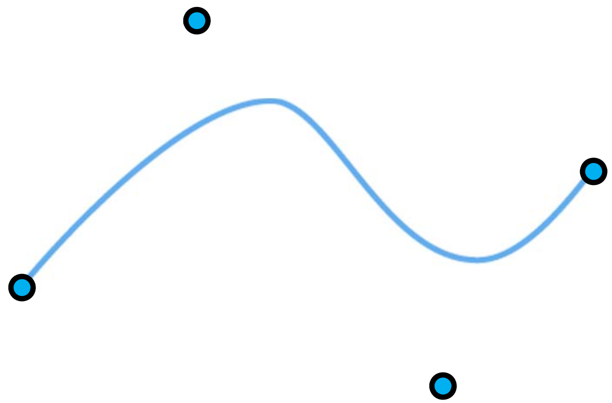
Sichtweisen und Anwendungen

- Definieren einer glatten parametrischen Kurve $P(u)$ durch Platzieren einiger „Kontrollpunkte“
  - wie können wir die Punkte mit einer Kurve approximieren oder interpolieren? (meist verbergen sich dahinter Polynome)

- Interpolation:
  die Kurve geht durch alle vorgegebenen Punkte
  - kann aber instabil sein und Überschwinger erzeugen



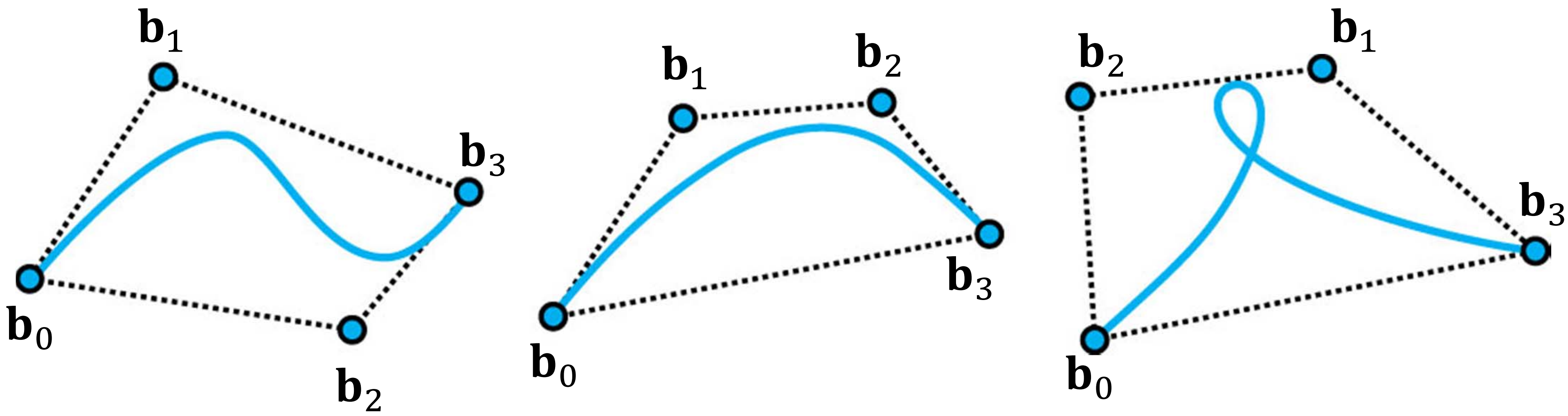
- Approximation:
  die Kurve geht nicht durch alle (oder auch gar keinen) Punkt
  - erlaubt Kurven mit „angenehmen“ Eigenschaften



Beispiel: Kubische Bézierkurven

Einführungsbeispiel kubische Bézierkurven

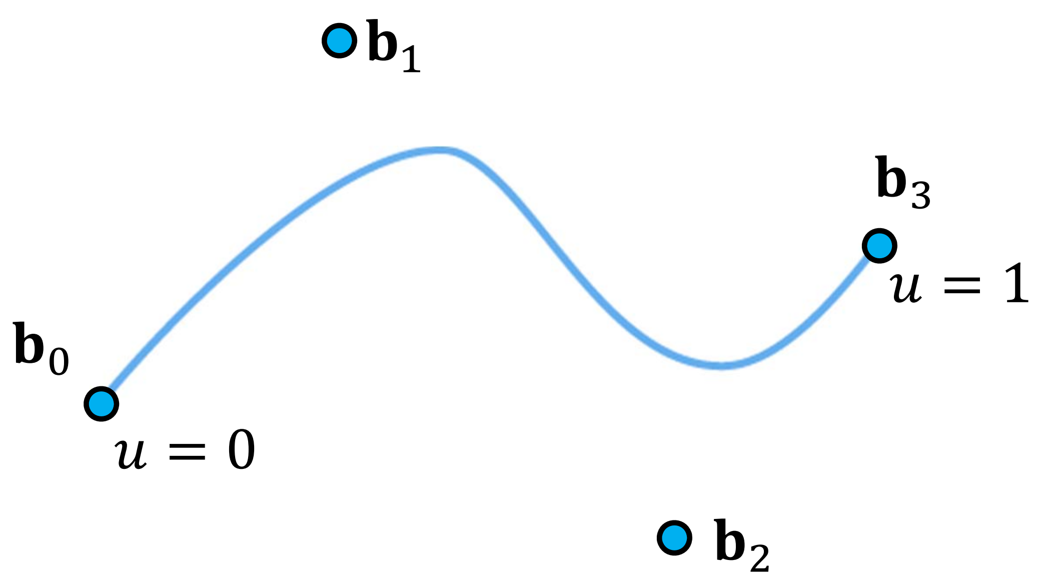
- definiert durch 4 Kontrollpunkte $\mathbf{b}_i$
- die Kurve geht durch den ersten und letzten Kontrollpunkt ($\mathbf{b}_0$ und $\mathbf{b}_3$)
- und approximiert die anderen beiden
- Kurve ist bei $\mathbf{b}_0$ tangential an $\overline{\mathbf{b}_0\mathbf{b}_1}$ und bei $\mathbf{b}_3$ tangential an $\overline{\mathbf{b}_3\mathbf{b}_2}$
- eine Bézierkurve liegt innerhalb der konvexen Hülle ihrer Kontrollpunkte
- mathematische Beschreibung mit einem kubischen Polynom



Beispiel: Kubische Bézierkurven

Einführungsbeispiel kubische Bézierkurven

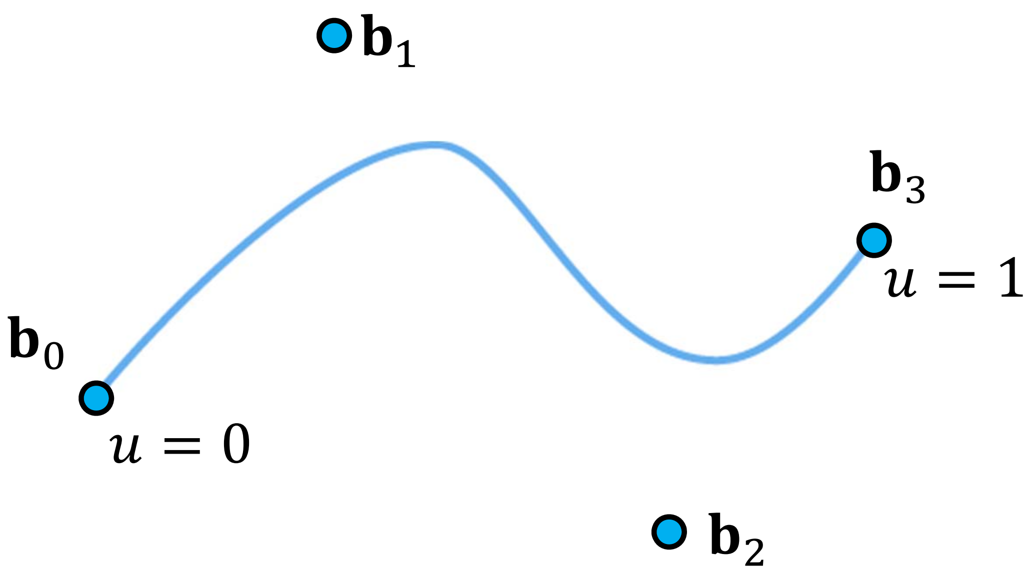
- $P(u) = (1-u)^3\mathbf{b}_0 + 3u(1-u)^2\mathbf{b}_1 + 3u^2(1-u)\mathbf{b}_2 + u^3\mathbf{b}_3$
  - mit $\mathbf{b}_0 = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$, $\mathbf{b}_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$, …
  - $\Rightarrow x(u) = (1-u)^3 x_0 + 3u(1-u)^2 x_1 + 3u^2(1-u)x_2 + u^3 x_3$
  - $\Rightarrow y(u) = \cdots$



Beispiel: Kubische Bézierkurven

Einführungsbeispiel kubische Bézierkurven

- $P(u) = (1-u)^3\mathbf{b}_0 + 3u(1-u)^2\mathbf{b}_1 + 3u^2(1-u)\mathbf{b}_2 + u^3\mathbf{b}_3$
  - mit $\mathbf{b}_0 = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$, $\mathbf{b}_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$, …
  - $\Rightarrow x(u) = (1-u)^3 x_0 + 3u(1-u)^2 x_1 + 3u^2(1-u) x_2 + u^3 x_3$
  - $\Rightarrow y(u) = \cdots$
- was passiert, wenn wir $u = 0$ oder $u = 1$ setzen?



Wo kommen die Formeln her?

- Möglichkeit 1
  - pure Magie, hab' ich aus dem Hut gezaubert, interpoliert bzw. approximiert erstaunlicherweise die Punkte

- Möglichkeit 2
  - es ist eine Linearkombination von geeigneten Basispolynomen

Polynomkurven

Polynomkurven

- eine d-dimensionale Polynomkurve ist eine Kurve $u \mapsto P(u)$:

$$P(u) = \sum_{i=0}^{n} \mathbf{a}_i u^i \;= \mathbf{a}_n u^n + \mathbf{a}_{n-1} u^{n-1} + \dots + \mathbf{a}_1 u + \mathbf{a}_0 \cdot 1, \text{ mit } \mathbf{a}_i \in \mathbb{R}^d$$

  - Menge der d-dimensionalen Polynomkurven von Grad n: $\mathbb{P}_n^d$

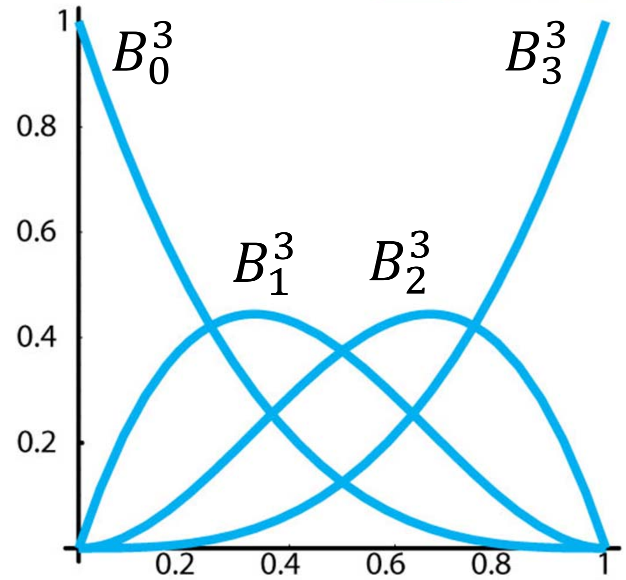
- Standardbasis für $\mathbb{P}_n$ ist die Monombasis $1, u, u^2, \dots, u^n$
  - Linearkombinationen dieser Basisvektoren spannen den Raum auf
  - kein Basisvektor durch Linearkombination der anderen darstellbar
- mögliche andere Basen aus denen mit Linearkombinationen alle kubischen Polynome dargestellt werden können:
  - $1,\ 1+u,\ 1+u+u^2, 1+u-u^2+u^3$
  - $u^3,\ u^3+u^2,\ u^3+u,\ u^3+1$
  - Lagrange-Polynome, …

- was ist eine geschickte Basis für (Bézier-)Kurven?

Bernstein-Polynome

Kubische Bernstein-Polynome (Basis des $\mathbb{P}_3$)

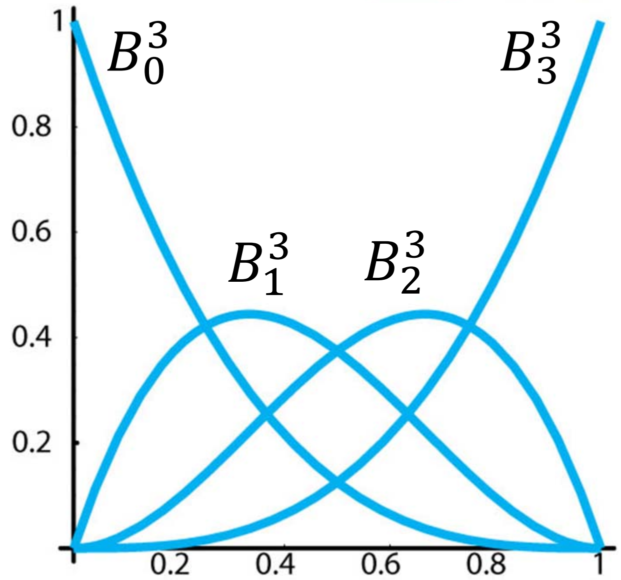
- $B_0^3\,(u) = (1-u)^3$
- $B_1^3\,(u) = 3u(1-u)^2$
- $B_2^3(u) = 3u^2(1-u)$
- $B_3^3(u) = u^3$



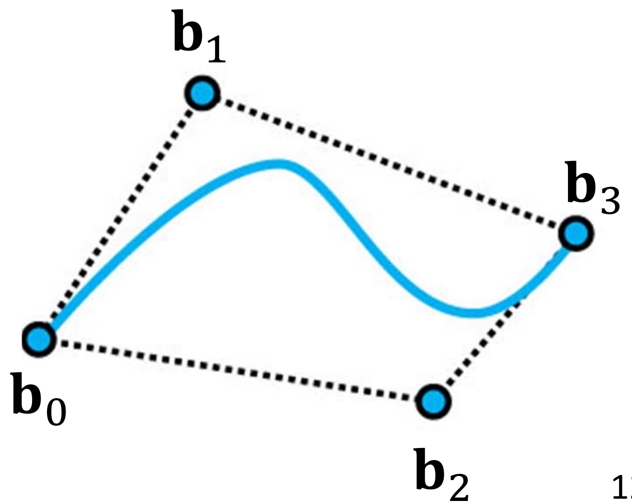
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

- $B_0^3(u) = (1-u)^3$
- $B_1^3(u) = 3u(1-u)^2$
- $B_2^3(u) = 3u^2(1-u)$
- $B_3^3(u) = u^3$



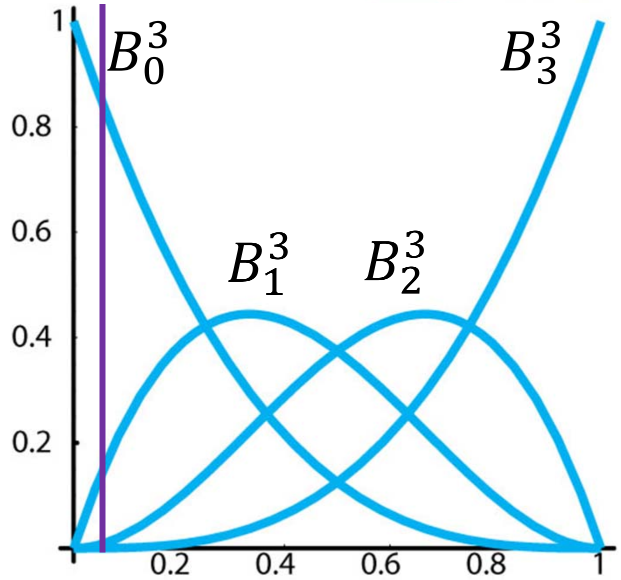
- Bézierkurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- anschaulich: jedes B_i^3 legt den Einfluss von $\mathbf{b}_i$ auf die Kurve an der Stelle u fest
  - erst hat $\mathbf{b}_0$ den größten Einfluss, dann $\mathbf{b}_1$, $\mathbf{b}_2$ und dann $\mathbf{b}_3$
  - $\mathbf{b}_1$ und $\mathbf{b}_2$ haben nie alleinigen Einfluss und werden daher nicht interpoliert



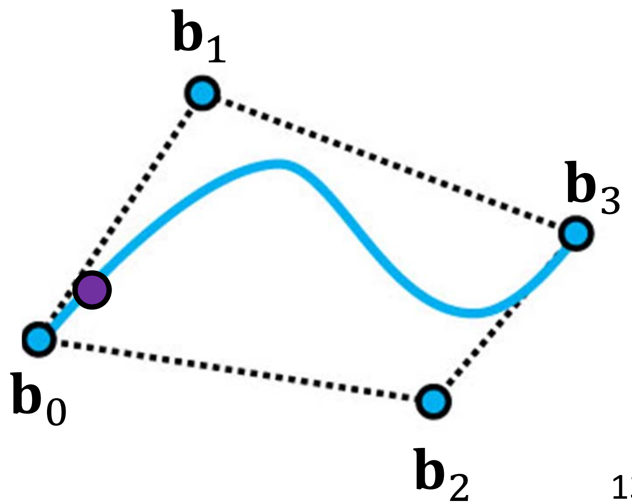
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

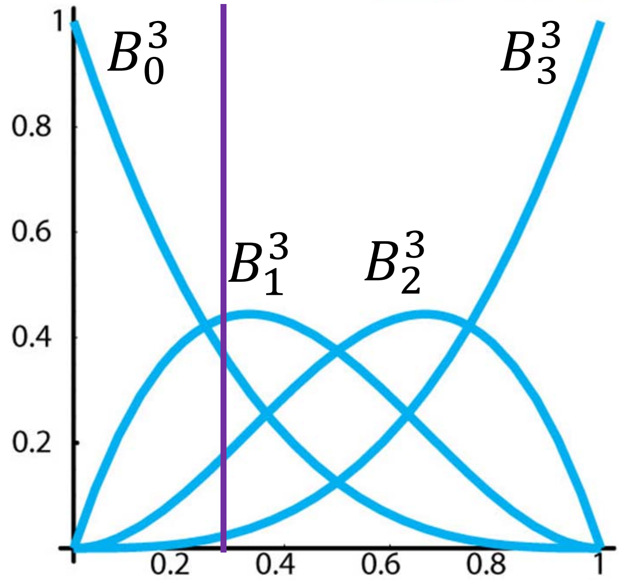
- $B_0^3(u) = (1-u)^3$
- $B_1^3(u) = 3u(1-u)^2$
- $B_2^3(u) = 3u^2(1-u)$
- $B_3^3(u) = u^3$



- Bézierkurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- anschaulich: jedes B_i^3 legt den Einfluss von $\mathbf{b}_i$ auf die Kurve an der Stelle u fest
  - erst hat $\mathbf{b}_0$ den größten Einfluss, dann $\mathbf{b}_1$, $\mathbf{b}_2$ und dann $\mathbf{b}_3$
  - $\mathbf{b}_1$ und $\mathbf{b}_2$ haben nie alleinigen Einfluss und werden daher nicht interpoliert



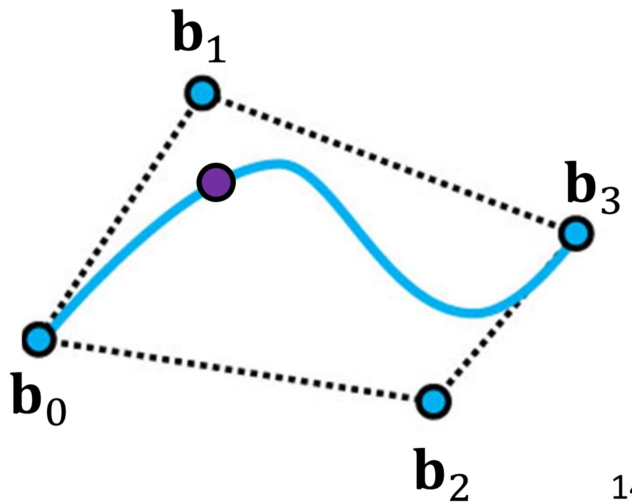
Bernstein-Polynome



Bernstein-Polynome als Gewichtungsfunktion

- $B_0^3(u) = (1-u)^3$
- $B_1^3(u) = 3u(1-u)^2$
- $B_2^3(u) = 3u^2(1-u)$
- $B_3^3(u) = u^3$

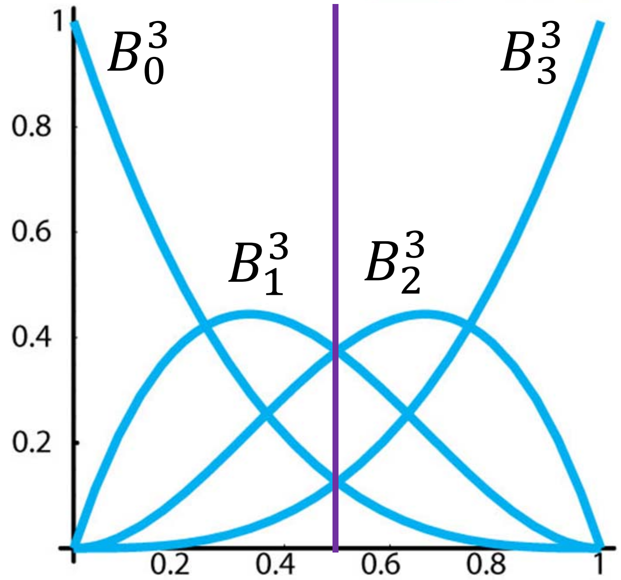
- Bézierkurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- anschaulich: jedes B_i^3 legt den Einfluss von $\mathbf{b}_i$ auf die Kurve an der Stelle u fest
  - erst hat $\mathbf{b}_0$ den größten Einfluss, dann $\mathbf{b}_1$, $\mathbf{b}_2$ und dann $\mathbf{b}_3$
  - $\mathbf{b}_1$ und $\mathbf{b}_2$ haben nie alleinigen Einfluss und werden daher nicht interpoliert



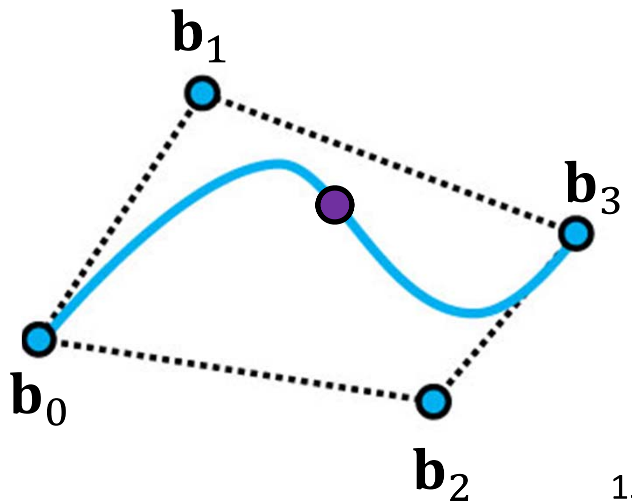
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

- $B_0^3(u) = (1-u)^3$
- $B_1^3(u) = 3u(1-u)^2$
- $B_2^3(u) = 3u^2(1-u)$
- $B_3^3(u) = u^3$



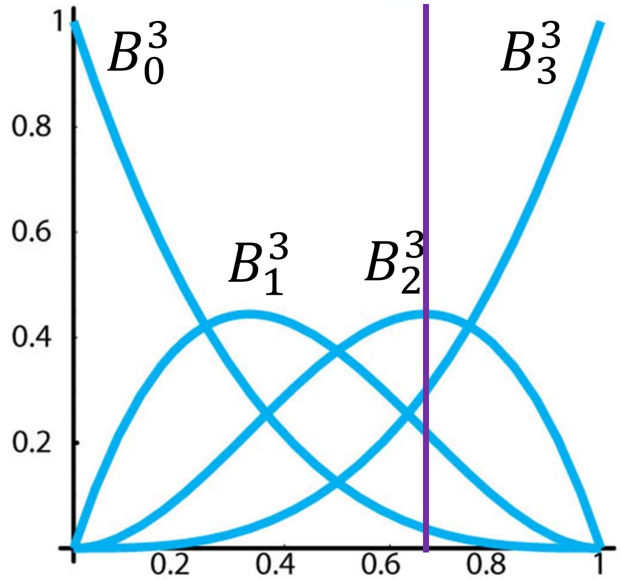
- Bézierkurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- anschaulich: jedes B_i^3 legt den Einfluss von $\mathbf{b}_i$ auf die Kurve an der Stelle u fest
  - erst hat $\mathbf{b}_0$ den größten Einfluss, dann $\mathbf{b}_1$, $\mathbf{b}_2$ und dann $\mathbf{b}_3$
  - $\mathbf{b}_1$ und $\mathbf{b}_2$ haben nie alleinigen Einfluss und werden daher nicht interpoliert



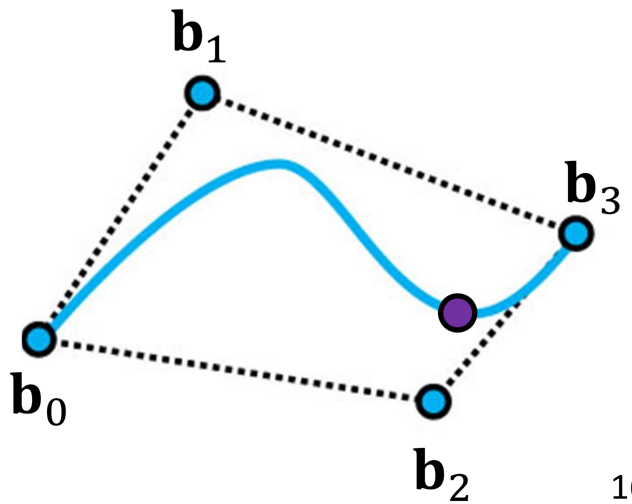
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

- $B_0^3(u) = (1-u)^3$
- $B_1^3(u) = 3u(1-u)^2$
- $B_2^3(u) = 3u^2(1-u)$
- $B_3^3(u) = u^3$



- Bézierkurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- anschaulich: jedes B_i^3 legt den Einfluss von $\mathbf{b}_i$ auf die Kurve an der Stelle u fest
  - erst hat $\mathbf{b}_0$ den größten Einfluss, dann $\mathbf{b}_1$, $\mathbf{b}_2$ und dann $\mathbf{b}_3$
  - $\mathbf{b}_1$ und $\mathbf{b}_2$ haben nie alleinigen Einfluss und werden daher nicht interpoliert



Bézierkurve

Allgemeine Definition

- eine Polynomkurve

$$F(u) = \sum_{i=0}^{n} B_i^n(u)\mathbf{b}_i \text{ , mit } \mathbf{b}_i \in \mathbb{R}^d$$

  heißt *Bézierkurve* vom Grad n, die Punkte $\mathbf{b}_i$ heißen *Kontroll*- oder *Bézierpunkte* und bilden das *Kontrollpolygon*

- auch bezeichnet als Bézier-Repräsentation einer Polynomkurve
- entwickelt von *Paul de Casteljau* 1959
- bekannt geworden durch *Pierre Étienne Bézier* 1962 für Karosseriedesign

Bernstein-Polynome

Definition (nach Sergeĭ N. Bernstein, 1912)

- i-tes Bernstein-Polynom vom Grad n

$$B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}, \text{ mit } i = 0, \dots, n$$

  - man vereinbart $B_i^n \equiv 0$ falls $i < 0$ oder $i > n$
  - Binomialkoeffizient $\binom{n}{k} := \frac{n!}{k!(n-k)!}$
    - rekursive Definition: $\binom{n}{0} := 1$, $\binom{0}{k} := 0$ und $\binom{n+1}{k+1} := \frac{n+1}{k+1}\binom{n}{k}$
    - Pascalsches Dreieck: $\binom{n+1}{k+1} := \binom{n}{k+1} + \binom{n}{k}$
    - Symmetrie $\binom{n}{k} = \binom{n}{n-k}$

Bernstein-Polynome

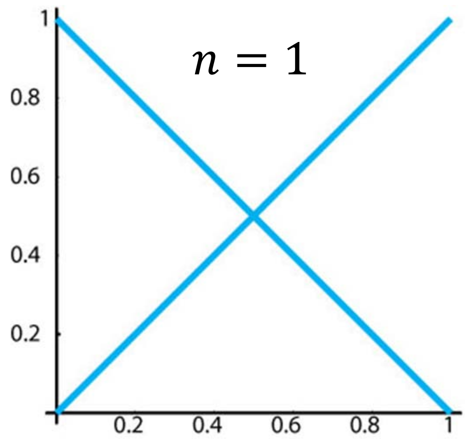
Eigenschaften

- Bernstein-Polynome vom Grad 1

$$B_0^1(u) = \binom{1}{0} u^0 (1-u)^{1-0} = 1-u$$
$$B_1^1(u) = \binom{1}{1} u^1 (1-u)^{1-1} = u$$

- resultiert in linearer Interpolation

$$F(u) = \sum_{i=0}^{1} B_i^1(u)\mathbf{b}_i = (1-u)\mathbf{b}_0 + u\mathbf{b}_1$$



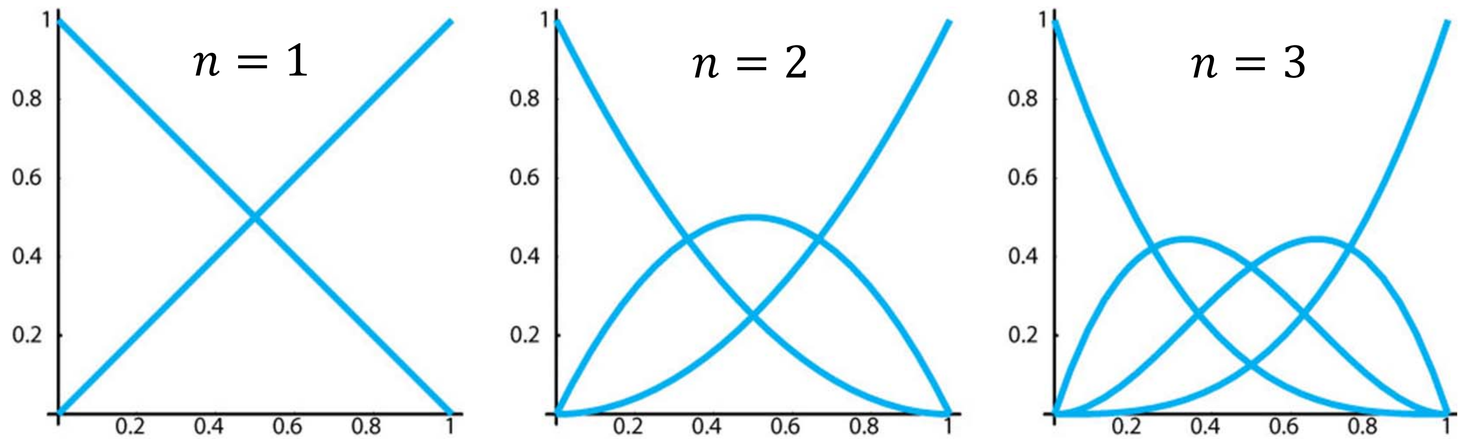
Bernstein-Polynome

Eigenschaften

- i-tes Bernstein-Polynom vom Grad n

$$B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}, \text{ mit } i = 0, \dots, n$$

- $\sum_{i=0}^{n} B_i^n(u) = 1$ (Partition der 1, Beweis über binomischen Lehrsatz)
- $B_i^n(u) \geq 0$ für $u \in [0;1]$ (Positivität)
- $B_i^n(u) = B_{n-i}^n(1-u)$
- $B_i^n(u)$ hat Maximum in $[0;1]$ bei $u = {}^{i}/_{n}$



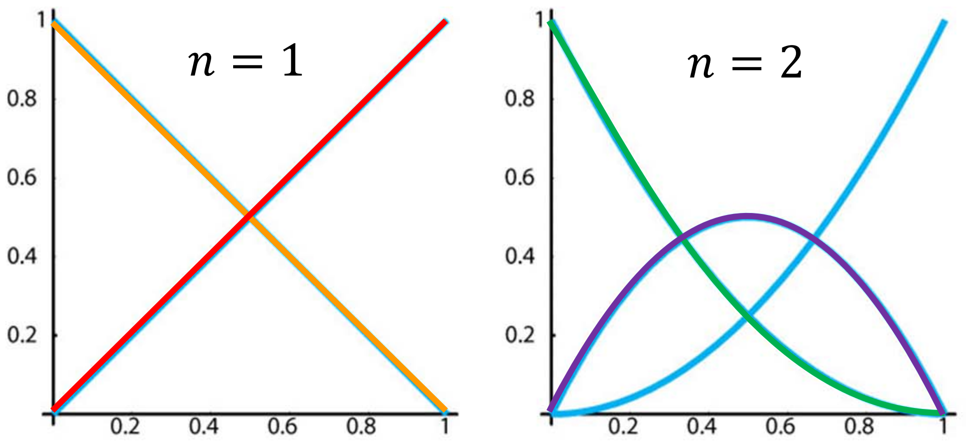
Bernstein-Polynome

Eigenschaften

- i-tes Bernstein-Polynom vom Grad n

$$B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}, \text{ mit } i = 0, \ldots, n$$

- $B_i^n(u) = u \cdot B_{i-1}^{n-1}(u) + (1-u) \cdot B_i^{n-1}(u)$ (Rekursion, wichtig!)
- $B_0^1(u) = 1 - u$ und $B_1^1(u) = u$
- $B_0^2(u) = u \cdot B_{-1}^1(u) + (1-u) \cdot B_0^1(u) = (1-u)^2 = 1 - 2u + u^2$
- $B_1^2(u) = u \cdot B_0^1(u) + (1-u) \cdot B_1^1(u) = u(1-u) + (1-u)u = 2u - 2u^2$



Basiswechsel

- die Bernstein-Polynome B_i^n bilden eine Basis des $\mathbb{P}_n$
  - wie sind die Koeffizienten einer Bézierkurve bzgl. der Monombasis? Also: gesucht $\{\mathbf{a}_i\}$ für $F(u) = \sum_i \mathbf{a}_i u^i = \sum_i \mathbf{b}_i B_i^n(u)$

- Basiswechsel kann durch eine Matrix ausgedrückt werden
  - Satz aus der linearen Algebra:
    geg. zwei Linearkombinationen, die einen Vektor $\mathbf{v}$ darstellen mit $\mathbf{v} = \sum_i \gamma_i \mathbf{c}_i = \sum_i \delta_i \mathbf{d}_i$, dann gilt:

$$\begin{pmatrix} \vdots \\ \mathbf{c}_i \\ \vdots \end{pmatrix} = \mathrm{M} \begin{pmatrix} \vdots \\ \mathbf{d}_i \\ \vdots \end{pmatrix} \Rightarrow \begin{pmatrix} \vdots \\ \delta_i \\ \vdots \end{pmatrix} = \mathrm{M}^{\mathrm{T}} \begin{pmatrix} \vdots \\ \gamma_i \\ \vdots \end{pmatrix}$$

  - Beweis:

$$\mathbf{c}_i = \sum_j m_{ij} \mathbf{d}_j$$

$$\Rightarrow \mathbf{v} = \sum_i \gamma_i \sum_j m_{ij} \mathbf{d}_j = \sum_j \left(\sum_i \gamma_i m_{ij} \right) \mathbf{d}_j \overset{!}{=} \sum_j \delta_j \mathbf{d}_j$$

$$\Rightarrow \delta_j = \sum_i m_{ij} \gamma_i$$

Basiswechsel

- die Bernstein-Polynome B_i^n bilden eine Basis des $\mathbb{P}_n$
  - wie sind die Koeffizienten einer Bézierkurve bzgl. der Monombasis? Also: gesucht $\{\mathbf{a}_i\}$ für $F(u) = \sum_i \mathbf{a}_i u^i = \sum_i \mathbf{b}_i B_i^n(u)$

- Beispiel $n = 2$: $F(u) = \mathbf{b}_0 B_0^2(u) + \mathbf{b}_1 B_1^2(u) + \mathbf{b}_2 B_2^2(u)$
  - $B_0^2(u) = \binom{2}{0} u^0 (1-u)^{2-0} = (1-u)^2 = 1 - 2u + u^2$
  - $B_1^2(u) = \binom{2}{1} u^1 (1-u)^{2-1} = 2u(1-u) = 2u - 2u^2$
  - $B_2^2(u) = \binom{2}{2} u^2 (1-u)^{2-2} = u^2$
  - $\begin{pmatrix} B_0^2 \\ B_1^2 \\ B_2^2 \end{pmatrix} = \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ u \\ u^2 \end{pmatrix} \Rightarrow \begin{pmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$

Basiswechsel

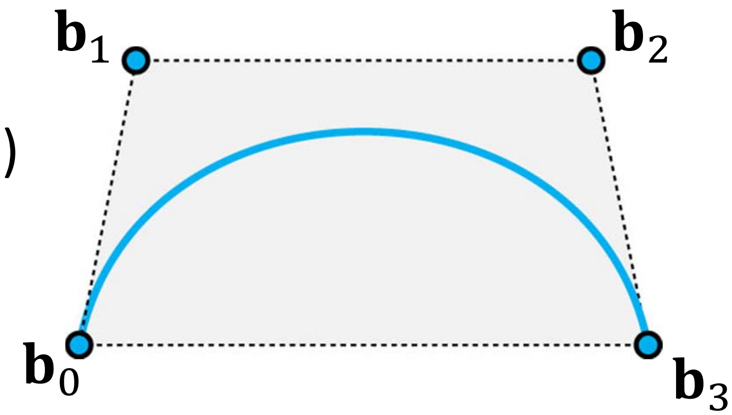
- die Bernstein-Polynome B_i^n bilden eine Basis des $\mathbb{P}_n$
  - wie sind die Koeffizienten einer Bézierkurve bzgl. der Monombasis? Also: gesucht $\{\mathbf{a}_i\}$ für $F(u) = \sum_i \mathbf{a}_i u^i = \sum_i \mathbf{b}_i B_i^n(u)$

- Beispiel $n = 3$: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
  - $B_0^3(u) = (1-u)^3$
  - $B_1^3(u) = 3u(1-u)^2$
  - $B_2^3(u) = 3u^2(1-u)$
  - $B_3^3(u) = u^3$
  - $F(u) = (\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3) \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ u \\ u^2 \\ u^3 \end{pmatrix}$

Bézierkurve

Eigenschaften (Lemma von Bézier)

- für $F(u) = \sum_{i=0}^{n} B_i^n(u)\mathbf{b}_i$ gilt
  - $F(u)$ **liegt in der abgeschlossenen konvexen Hülle des Kontroll-polygons** (für $u \in [0;1]$, $F(u)$ ist Konvexkombination der $\{\mathbf{b}_i\}$)
    - Grund:
      - $\sum_{i=0}^{n} B_i^n(u) = 1$ (Partition der 1)
      - $B_i^n(u) \geq 0$ für $u \in [0;1]$ (Positivität)



  - **Endpunktinterpolation** $F(0) = \mathbf{b}_0$ und $F(1) = \mathbf{b}_3$
  - **Tangentenbedingung** $F'(0) = n(\mathbf{b}_1 - \mathbf{b}_0) \wedge F'(1) = n(\mathbf{b}_n - \mathbf{b}_{n-1})$
    - allgemein $F'(u) = n\sum_{i=0}^{n-1} B_i^{n-1}(u)\,(\mathbf{b}_{i+1} - \mathbf{b}_i)$

$$F''(u) = n(n-1)\sum_{i=0}^{n-2} B_i^{n-2}(u)\,(\mathbf{b}_{i+2} - 2\mathbf{b}_{i+1} + \mathbf{b}_i)$$

Bézierkurve

Eigenschaften (Lemma von Bézier)

- **affine Invarianz**: sei $\varphi(\mathbf{x}) = \mathrm{A}\mathbf{x} + \mathbf{t}$ eine affine Abbildung, dann gilt

$$\varphi\big(F(u)\big) = \sum_{i=0}^{n} B_i^n(u)\varphi(\mathbf{b}_i)$$

  - d.h. um die Kurve zu transformieren genügt es, die $\{\mathbf{b}_i\}$ zu transf.

- „Beweis“: lineare Transformation A für $n = 2$

  - $F_\mathrm{A}(u) = \mathrm{A}\left((\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2) \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ u \\ u^2 \end{pmatrix} \right) =$

$$= (\mathrm{A}\mathbf{b}_0, \mathrm{A}\mathbf{b}_1, \mathrm{A}\mathbf{b}_2) \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ u \\ u^2 \end{pmatrix}$$

Bézierkurve

Eigenschaften (Lemma von Bézier)

- **affine Invarianz**: sei $\varphi(\mathbf{x}) = \mathrm{A}\mathbf{x} + \mathbf{t}$ eine affine Abbildung, dann gilt

$$\varphi\big(F(u)\big) = \sum_{i=0}^{n} B_i^n(u)\varphi(\mathbf{b}_i)$$

  - d.h. um die Kurve zu transformieren genügt es, die $\{\mathbf{b}_i\}$ zu transf.

- Beweis: Translation $\mathbf{t}$ für $n = 2$:
  - $F_{\mathbf{t}}(u) = (\mathbf{b}_0 + \mathbf{t})B_0^2(u) + (\mathbf{b}_1 + \mathbf{t})B_1^2(u) + (\mathbf{b}_2 + \mathbf{t})B_2^2(u) =$

    $= \mathbf{b}_0 B_0^2(u) + \mathbf{b}_1 B_1^2(u) + \mathbf{b}_2 B_2^2(u) + \mathbf{t}\underbrace{\Big(B_0^2(u) + B_1^2(u) + B_2^2(u)\Big)}_{=1} =$

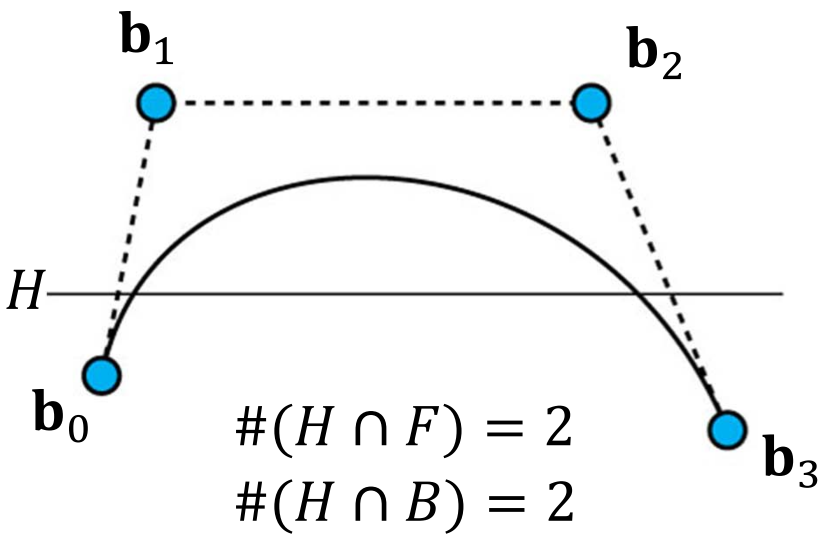
    $= F(u) + \mathbf{t}$

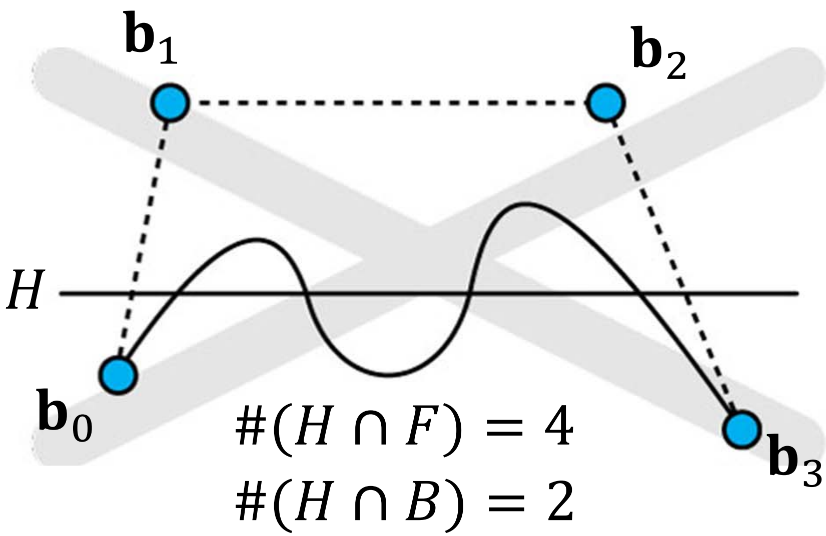
  - Beweis für $\varphi(\mathbf{x}) = \mathrm{A}\mathbf{x} + \mathbf{t}$ und andere n analog

Bézierkurve

Eigenschaften (Lemma von Bézier)

- für $F(u) = \sum_{i=0}^{n} B_i^n(u)\mathbf{b}_i$ gilt
  - **Variationsreduzierung** („variation diminishing property"):
    „Eine Bézier-Kurve F wackelt nicht stärker als ihr Kontrollpolygon B"
  - sei H eine beliebige Hyperebene in $\mathbb{R}^d$ (Gerade in $\mathbb{R}^2$, Ebene in $\mathbb{R}^3$)
    dann gilt $\#(H \cap F) \leq \#(H \cap B)$
  - Beweis: G. Farin. „Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide". Academic Press, 4. Auflage, 1997





de Casteljau-Algorithmus

Effiziente Auswertung und Unterteilung von Bézierkurven

- möglich durch die rekursive Darstellung der Bernstein-Polynome
  - zur Erinnerung: $B_i^n(u) = u \cdot B_{i-1}^{n-1}(u) + (1-u) \cdot B_i^{n-1}(u)$
  - wir haben die rekursive Darstellung symbolisch betrachtet, setzt man die tatsächlichen Kontrollpunkte ein, wertet man die Kurve direkt aus

- Algorithmus
  - $\mathbf{b}_i^0 \coloneqq \mathbf{b}_i$
  - $\mathbf{b}_i^j \coloneqq (1-u) \cdot \mathbf{b}_i^{j-1} + u \cdot \mathbf{b}_{i+1}^{j-1},\ i = 0, \dots, n-j$
  - $\Rightarrow \mathbf{b}_0^n = F(u)$ ist ein Punkt auf der Bézierkurve

- Berechnung von $F(u) = \sum_{i=0}^{n} B_i^n(u) \mathbf{b}_i$ durch fortgesetzte lineare Interpolation, **ohne die Bernstein-Polynome zu berechnen**

- wir betrachten den Algorithmus anhand von Beispielen…

de Casteljau-Algorithmus

Effiziente Auswertung und Unterteilung von Bézierkurven

- Beispiel: quadratische Bézierkurven
  - $B_0^2(u) = \binom{2}{0} u^0 (1-u)^{2-0} = (1-u)^2$
  - $B_1^2(u) = \binom{2}{1} u^1 (1-u)^{2-1} = 2u(1-u)$
  - $B_2^2(u) = \binom{2}{2} u^2 (1-u)^{2-2} = u^2$
- Algorithmus: rekursive Auswertung
  - $\mathbf{b}_i^0 := \mathbf{b}_i$
  - $\mathbf{b}_i^j := (1-u) \cdot \mathbf{b}_i^{j-1} + u \cdot \mathbf{b}_{i+1}^{j-1}, \; i = 0, \dots, n-j$
- Beispiel (Ausmultiplizieren):
  - $\mathbf{b}_0^2 = (1-u) \cdot \mathbf{b}_0^1 + u \cdot \mathbf{b}_1^1$

$$= (1-u) \cdot \left((1-u) \cdot \mathbf{b}_0^0 + u \cdot \mathbf{b}_1^0 \right) + u \cdot \left((1-u) \cdot \mathbf{b}_1^0 + u \cdot \mathbf{b}_2^0 \right)$$

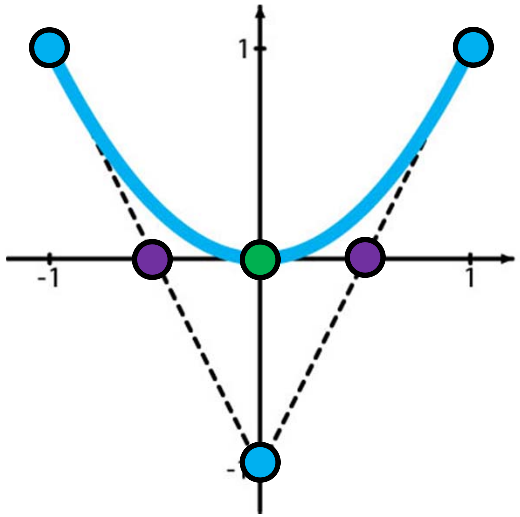
$$= (1-u)^2 \cdot \mathbf{b}_0^0 + 2u(1-u) \cdot \mathbf{b}_1^0 + u \cdot u \cdot \mathbf{b}_2^0$$

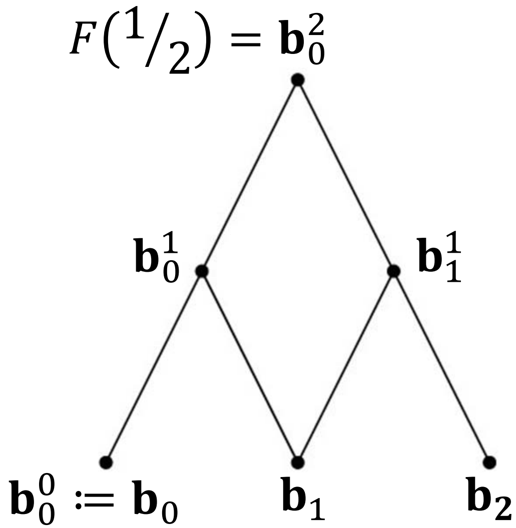
de Casteljau-Algorithmus

Effiziente Auswertung und Unterteilung von Bézierkurven

- möglich durch die rekursive Darstellung der Bernstein-Polynome
  - $\mathbf{b}_i^0 := \mathbf{b}_i$
  - $\mathbf{b}_i^j := (1-u) \cdot \mathbf{b}_i^{j-1} + u \cdot \mathbf{b}_{i+1}^{j-1},\ i = 0, \dots, n-j$

- Beispiel: $\mathbf{b}_0 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \mathbf{b}_1 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \mathbf{b}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, u = {}^1\!/_2$
  - $\mathbf{b}_0^1 = \frac{1}{2}\mathbf{b}_0 + \frac{1}{2}\mathbf{b}_1 = \begin{pmatrix} -{}^1\!/_2 \\ 0 \end{pmatrix}$
  - $\mathbf{b}_1^1 = \frac{1}{2}\mathbf{b}_1 + \frac{1}{2}\mathbf{b}_2 = \begin{pmatrix} {}^1\!/_2 \\ 0 \end{pmatrix}$
  - $\mathbf{b}_0^2 = \frac{1}{2}\mathbf{b}_0^1 + \frac{1}{2}\mathbf{b}_1^1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$





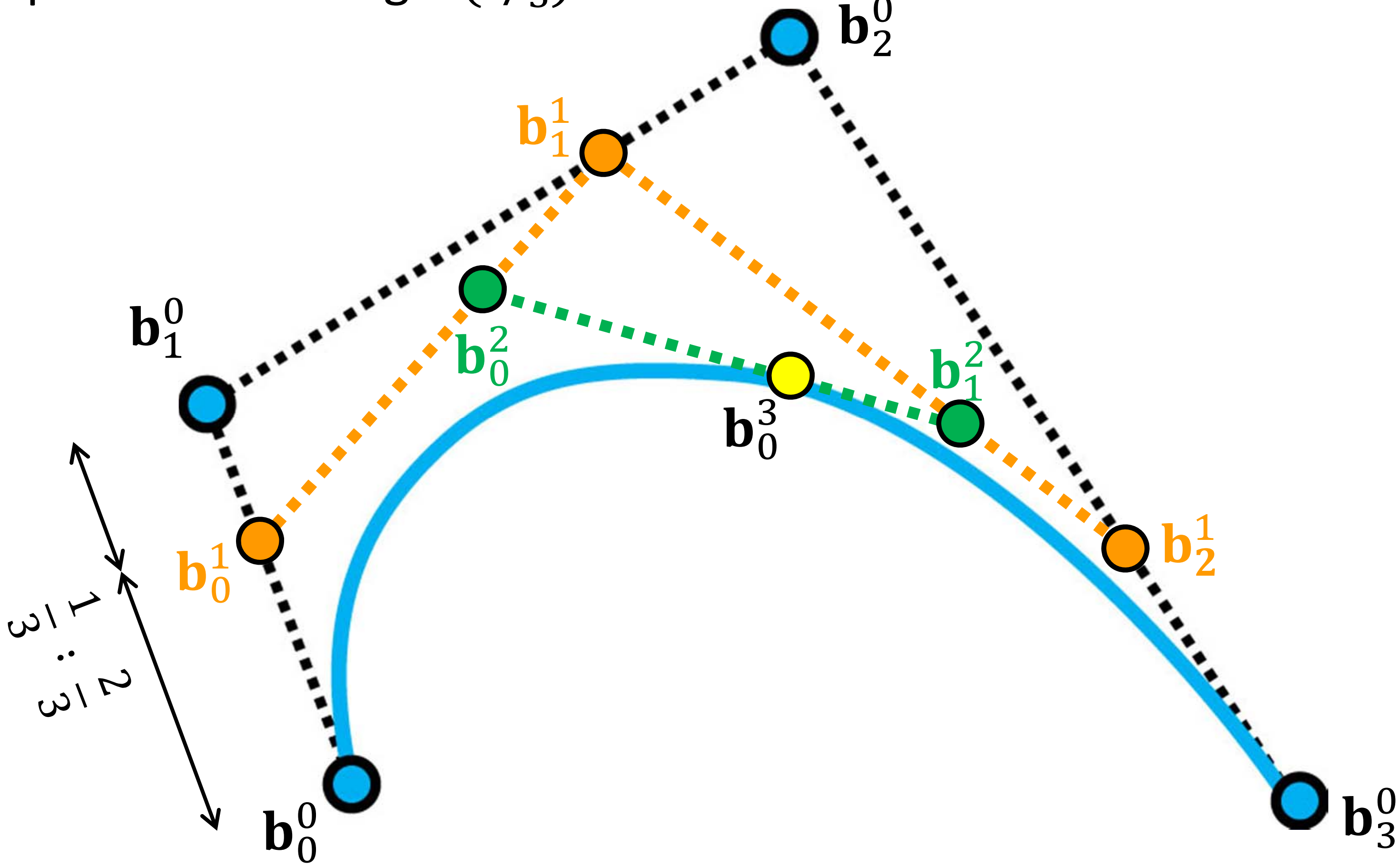
de Castelјau-Algorithmus

Effiziente Auswertung: grafische Interpretation/Konstruktion

- $\mathbf{b}_i^j$ ist lineare Interpolation gemäß u zwischen $\mathbf{b}_i^{j-1}$ und $\mathbf{b}_{i+1}^{j-1}$:

$$\mathbf{b}_i^j := (1-u)\cdot\mathbf{b}_i^{j-1} + u\cdot\mathbf{b}_{i+1}^{j-1}$$

- Beispiel: Auswertung $F(^2/_3)$

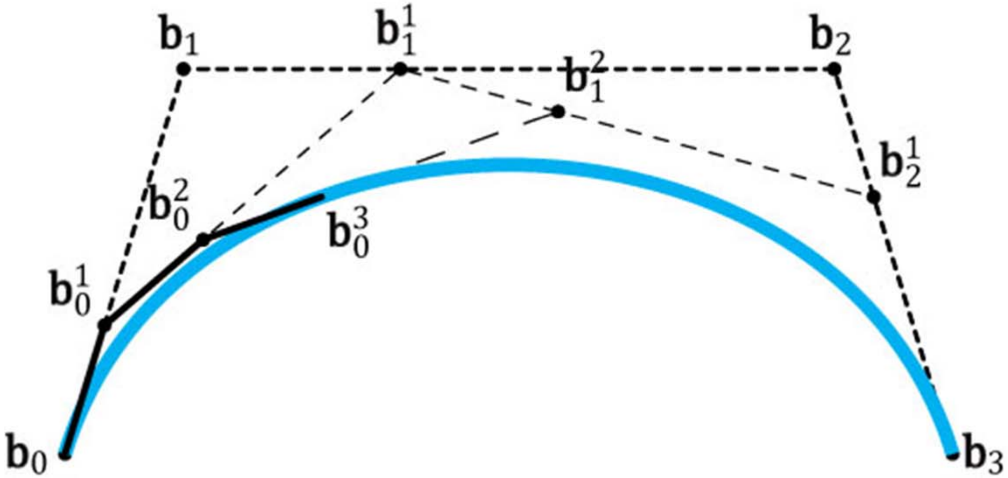


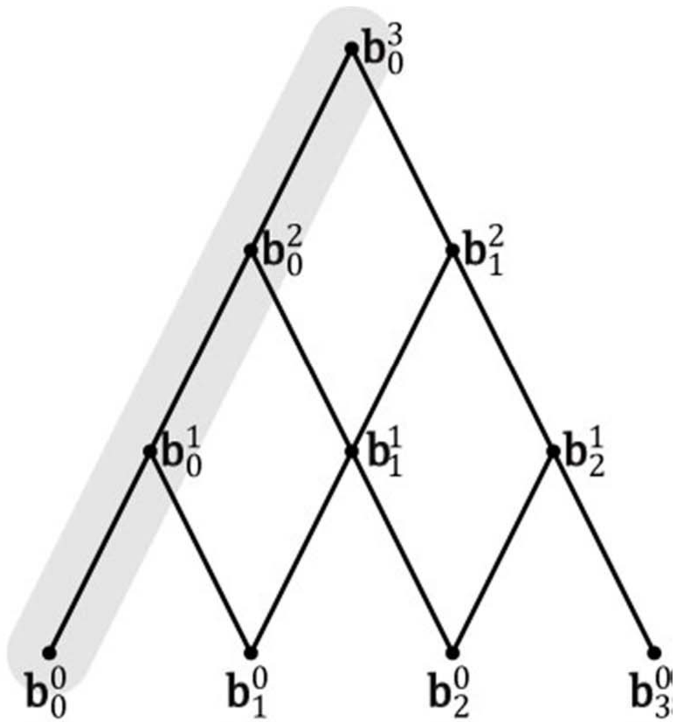
de Casteljau-Algorithmus

Unterteilung von Bézierkurven

- Bézierkurve $F(u) = \sum_{i=0}^{n} B_i^n(u)\mathbf{b}_i$
  - $\mathbf{b}_i^j(q)$ sind die Punkte des de Casteljau-Algorithmus bei q

- Beispiel: Unterteilung einer kubischen Bézierkurve in zwei Kurven $(\mathbf{b}_0, \mathbf{b}_0^1, \mathbf{b}_0^2, \mathbf{b}_0^3)$ und $(\mathbf{b}_0^3, \mathbf{b}_1^2, \mathbf{b}_2^1, \mathbf{b}_3)$



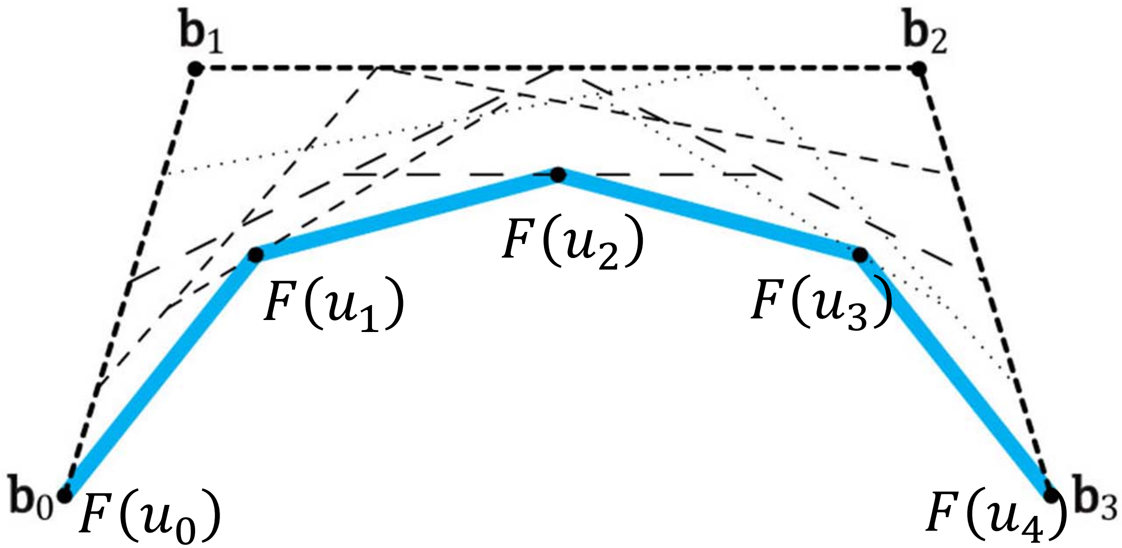


Zeichnen einer Bézierkurve

Algorithmus 1

- werte $F(u)$ an m Stellen u_j mit $u_j < u_{j+1}$ aus
  (mittels de Casteljau oder Auswerten der Bernstein-Polynome)
- zeichne den Polygonzug $F(u_0), \ldots, F(u_{m-1})$

- Beispiel: $n = 3, u_0 = 0, u_1 = \frac{1}{4}, u_2 = \frac{1}{2}, u_3 = \frac{3}{4}, u_4 = 1$
  - drei Auswertungen, da $F(u_0)$ und $F(u_4)$ bekannt sind

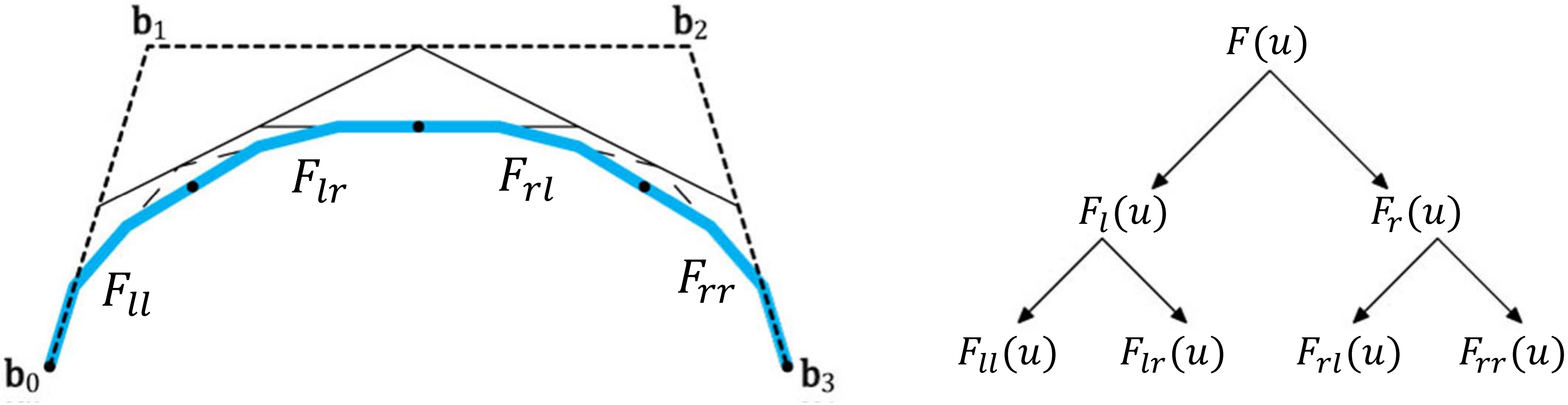


Zeichnen einer Bézierkurve

Algorithmus 2

- unterteile $F(u)$ in der parametrischen Mitte ($u = \frac{1}{2}$) in $F_l(u)$ und $F_r(u)$ mittels de Casteljau
- führe rekursiv bis zu einer Rekursionstiefe r fort
- zeichne die Kontrollpolygone der jeweils letzten Rekursion

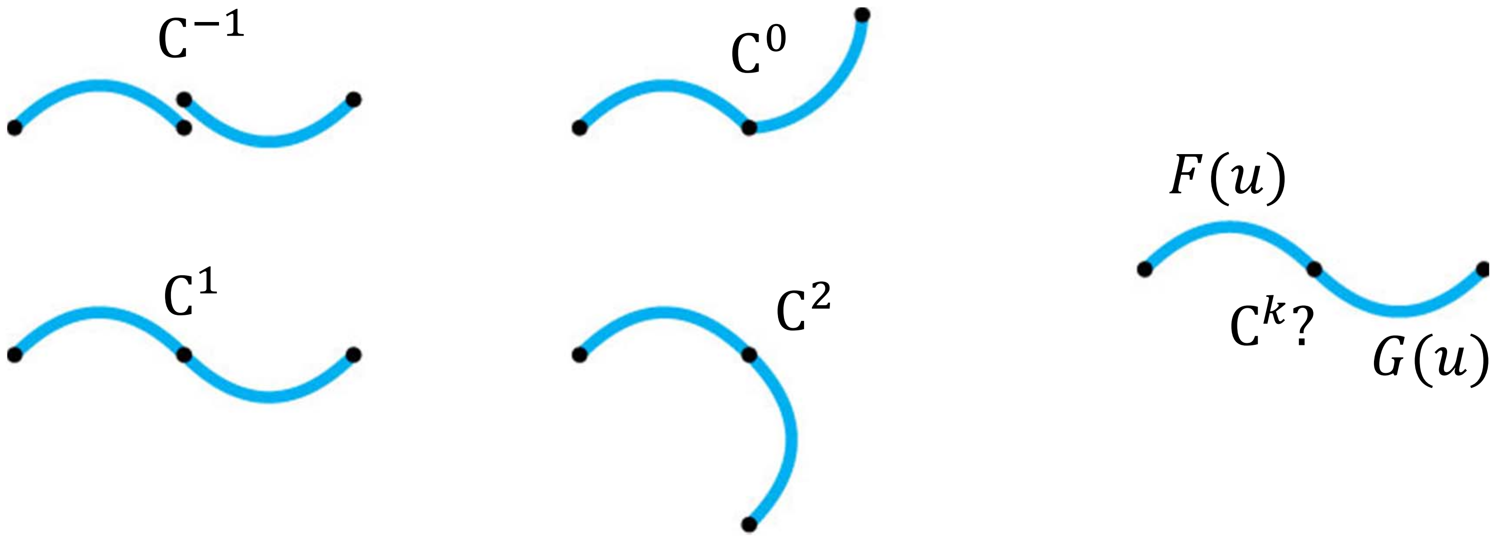
- Beispiel: $n = 3, r = 2$
  - gleicher Aufwand wie Algorithmus 1, aber vgl. Ergebnis!



Bézier-Splines

- Modellieren von komplexeren Formen mit Bézierkurven
  - 1) verwende eine Bézierkurve von hohem Grad
    - u.U. numerische Probleme, aber noch wichtiger: jeder Kontrollpunkt beeinflusst die ganze Kurve → schwierig bei der Modellierung
  - 2) füge mehrere Bézierkurven niedrigen Grades stückweise aneinander

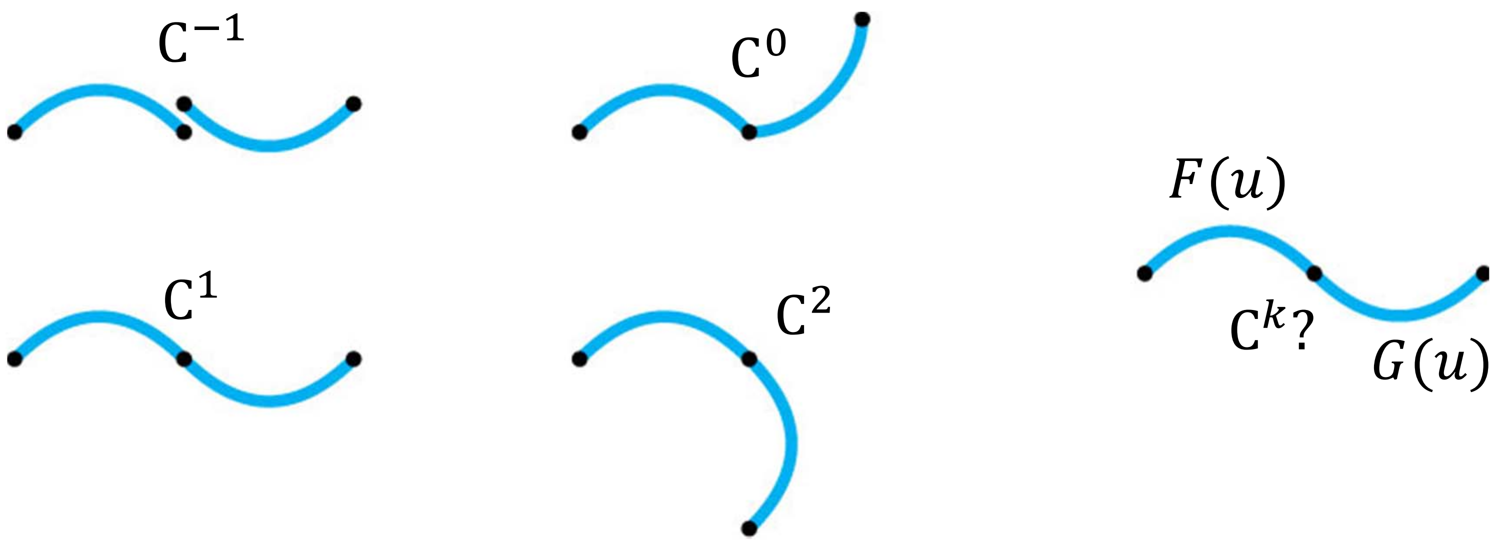
- **Bézier-Spline**: stückweise polynomielle Kurve, deren einzelne Abschnitte durch Bézierkurven beschrieben sind
  - (parametrische) Stetigkeit des Überganges? Glattheit der Kurve?



Bézier-Splines

- **Bézier-Spline**: stückweise polynomielle Kurve, deren einzelne Abschnitte durch Bézierkurven beschrieben sind

- zwei Polynomkurven $F\colon [r, s] \mapsto \mathbb{R}^d$ und $G\colon [s, t] \mapsto \mathbb{R}^d$ sind C^k-stetig bei s, wenn $F(s) = G(s)$, $F'(s) = G'(s)$, …, $F^k(s) = G^k(s)$
  - C^0-stetig = stetig, keine Sprungstellen
  - C^1-stetig = C^0-stetig + gleicher Tangentenvektor
  - C^2-stetig = C^1-stetig + gleicher Schmiegekreis

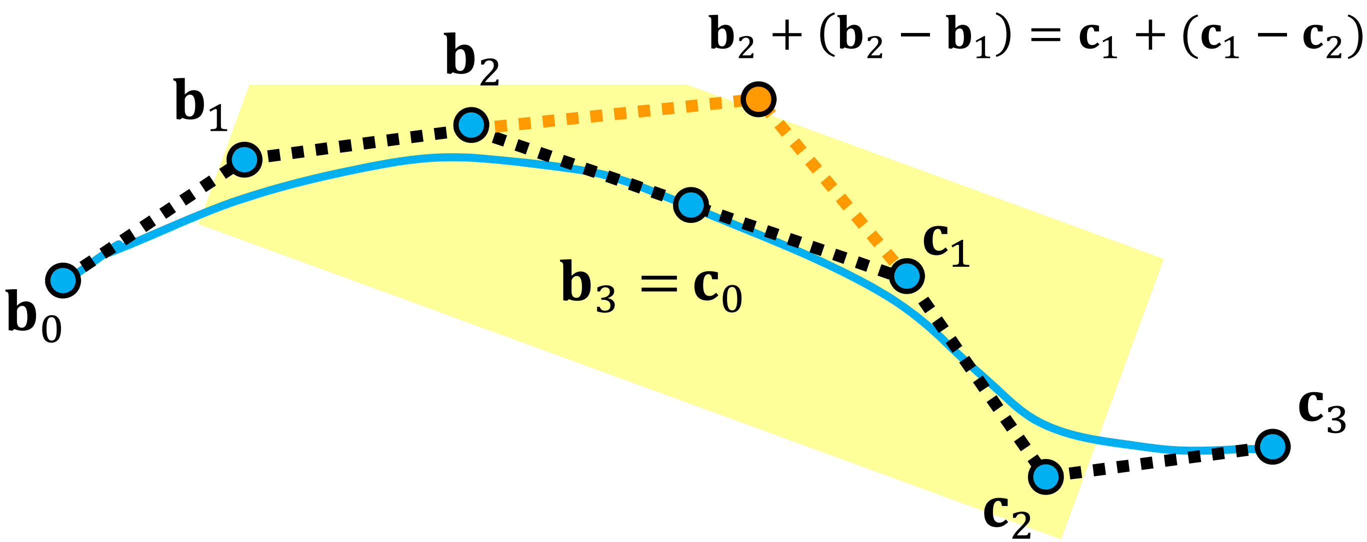


Bézier-Splines

C^k-Übergang zweier Bézierkurven (gleicher Grad)

- $F(u) = \sum_{i=0}^{n} B_i^n(u)\mathbf{b}_i$ und $G(u) = \sum_{i=0}^{n} B_i^n(u)\mathbf{c}_i$
  - C^0-stetig $\Leftrightarrow F(1) = G(0) \Leftrightarrow \mathbf{b}_n = \mathbf{c}_0$
  - C^1-stetig $\Leftrightarrow F'(1) = G'(0) \Leftrightarrow \mathbf{b}_n - \mathbf{b}_{n-1} = \mathbf{c}_1 - \mathbf{c}_0$ (und $\mathbf{b}_n = \mathbf{c}_0$)
  - C^2-stetig $\Leftrightarrow$ C^1-stetig und $F''(1) = G''(0)$
    $\Leftrightarrow \mathbf{b}_{n-1} + (\mathbf{b}_{n-1} - \mathbf{b}_{n-2}) = \mathbf{c}_1 + (\mathbf{c}_1 - \mathbf{c}_2)$

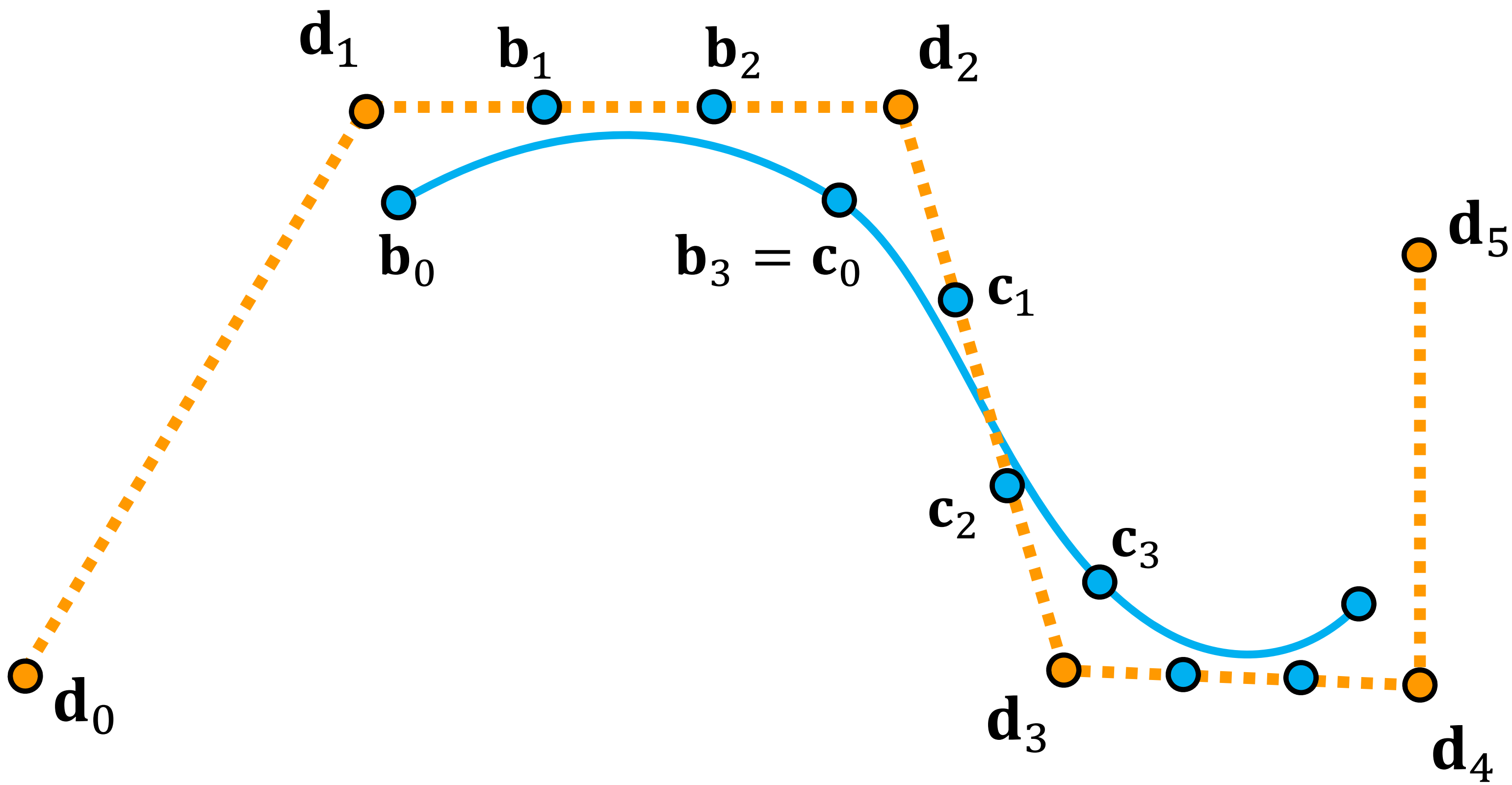
- Beispiel: kubische Bézierkurven und **„A-Frame“-Eigenschaft**



Bézier-Splines

B-Splines definiert durch A-Frames

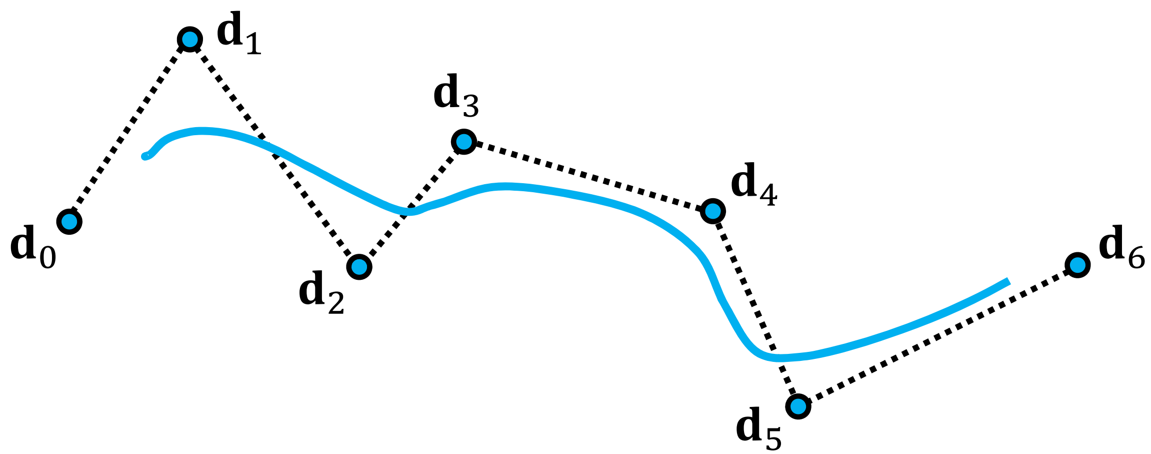
- statt Bézier-Punkte anzugeben kann man die Ecken der A-Frames festlegen, um einen C^2-stetigen sog. **B-Spline** zu konkstruieren
- diese neuen Kontrollpunkte $\mathbf{d}_i$ nennt man de Boor-Punkte
- je 4 de Boor-Punkte definieren eine kubische Bézier-Teilkurve



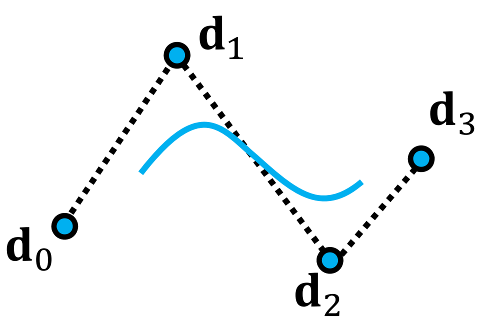
Kubische B-Splines

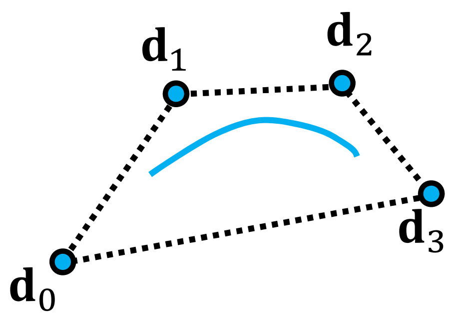
Kubische B-Splines (Splines mit einer anderen Polynombasis)

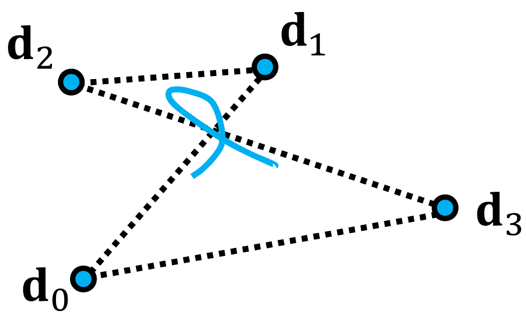
- 4 oder mehr Kontrollpunkte
- die Kurve ist aus kubischen Segmenten zusammengesetzt



- eine B-Spline Kurve liegt ebenfalls innerhalb der konvexen Hülle ihrer *(lokalen) Kontrollpunkte*

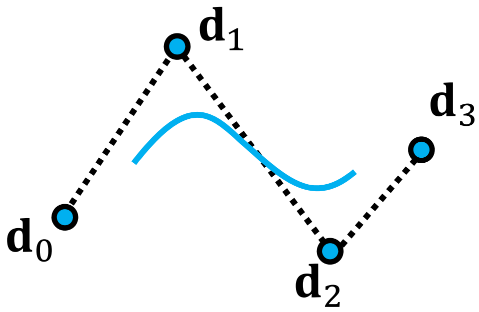


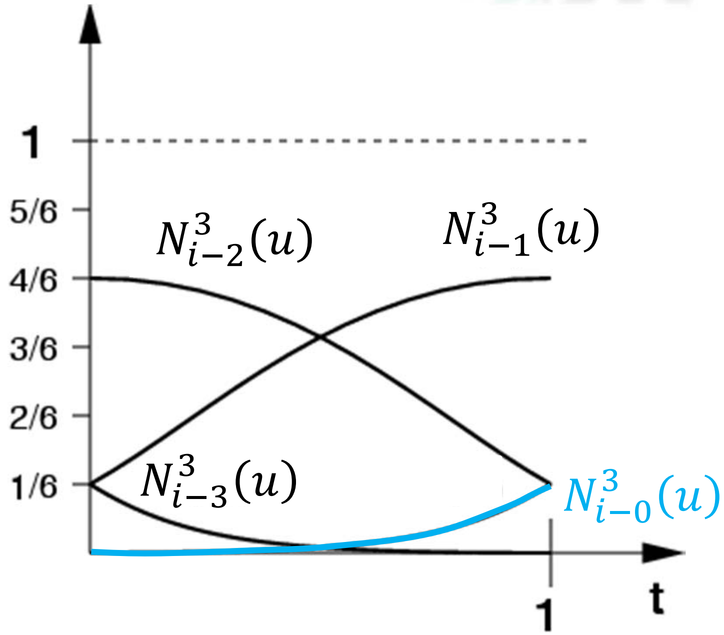




Exkurs: Kubische B-Splines

- Basisfunktionen $N_{i-3}^3, N_{i-2}^3, N_{i-1}^3, N_{i-0}^3$ für eine Teilkurve lassen sich auch direkt angeben





- kubisches Kurvensegment

$$F(u) = \frac{(1-u)^3}{6}\mathbf{d}_{i-3} + \frac{3u^3 - 6u^2 + 4}{6}\mathbf{d}_{i-2} + \frac{-3u^3 + 3u^2 + 3u + 1}{6}\mathbf{d}_{i-1} + \frac{u^3}{6}\mathbf{d}_i$$

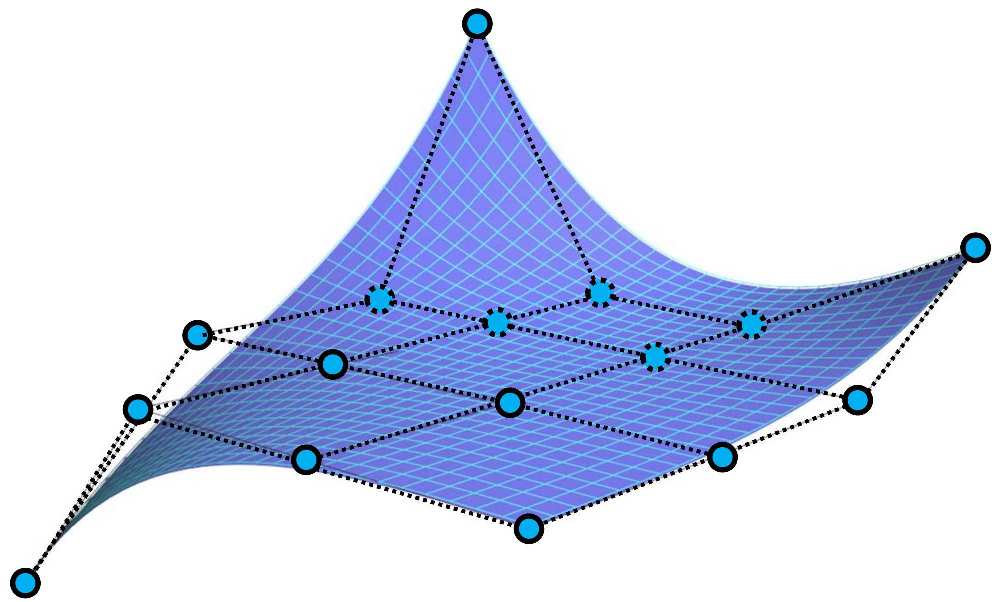
$$F(u) = (\mathbf{d}_0, \mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3)\,\frac{1}{6}\begin{pmatrix} 1 & -3 & 3 & -1 \\ 4 & 0 & -6 & 3 \\ 1 & 3 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ u \\ u^2 \\ u^3 \end{pmatrix}$$

Tensorprodukt-Bézier-Flächen

- gegeben zwei Basen zur Darstellung von Kurven in $\mathbb{R}^3$, z.B. Bézierkurven:
  $F(u) = \sum_{i=0}^{n} B_i(u)\mathbf{c}_i$ und $G(v) = \sum_{j=0}^{m} B_j(v)\mathbf{d}_j$
- das Tensorprodukt ist die TP-Bézier-Fläche:

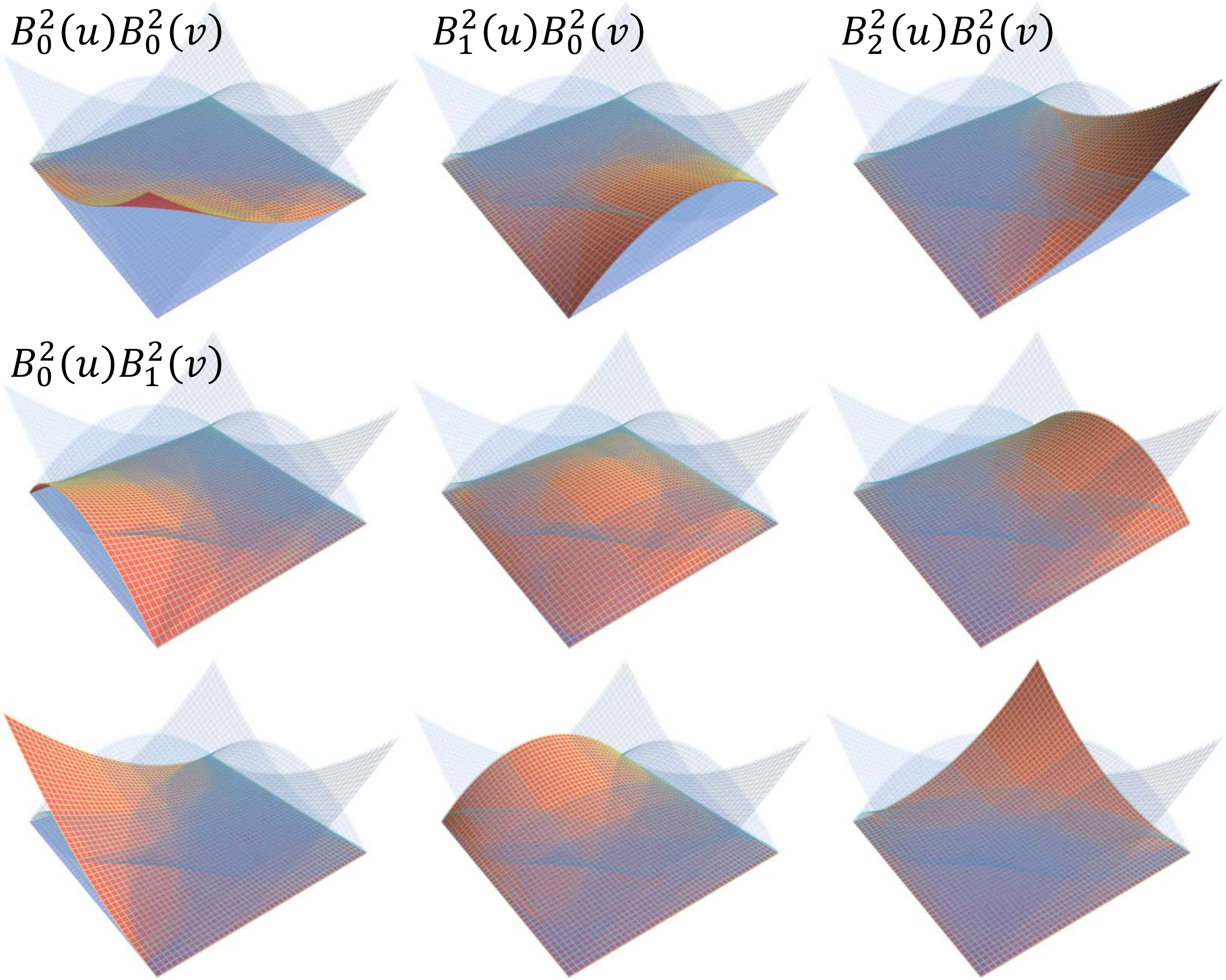
$$S(u,v) \coloneqq \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v)\mathbf{b}_{i,j}$$

- die Kontrollpunkte $\mathbf{b}_{i,j}$ bilden das **Kontrollnetz**
- Anm. es sind Kombinationen beliebiger Kurvenschemata und Grade möglich



Tensorprodukt-Bézier-Flächen

- Basisfunktionen $B_i^2(u)B_j^2(v)$



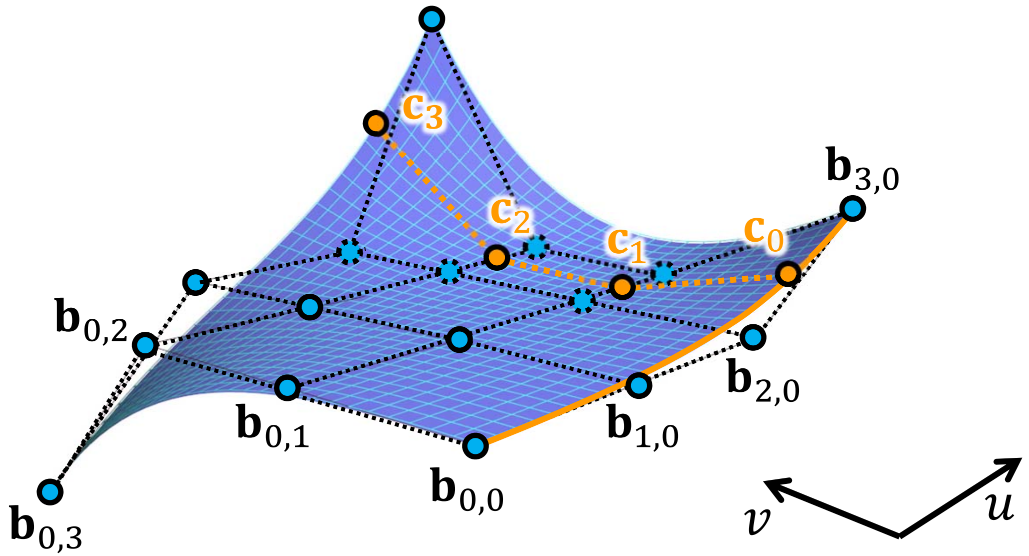
Tensorprodukt-Bézier-Flächen

Tensorprodukt-Fläche ist eine „Kurve von Kurven"

- Ausklammern

$$\sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v)\mathbf{b}_{i,j} = \sum_{i=0}^{n} B_i^n(u)\left(\sum_{j=0}^{m} B_j^m(v)\mathbf{b}_{i,j}\right) = \sum_{j=0}^{m} B_j^m(v)\underbrace{\left(\sum_{i=0}^{n} B_i^n(u)\mathbf{b}_{i,j}\right)}_{\mathbf{c}_j}$$

- u festhalten ergibt Kontrollpunkte der v-Kurve mit $\mathbf{c}_j = \sum_{i=0}^{n} B_i^n(u)\mathbf{b}_{i,j}$
- Beispiel: $u = {}^3/_4$ ergibt 4 Kontrollpunkte $\mathbf{c}_j$ einer Bézierkurve in v



Tensorprodukt-Bézier-Flächen

Formeigenschaften von TP-Bézier-Flächen

- konvexe Hülle-Eigenschaft

- Interpolation der 4 Ecken des Kontrollnetzes $\{\mathbf{b}_{i,j}\}$
  - $S(0,0) = \mathbf{b}_{0,0}$, $S(1,0) = \mathbf{b}_{n,0}$, $S(0,0) = \mathbf{b}_{0,m}$, $S(1,1) = \mathbf{b}_{n,m}$

- Fläche ist tangential in den Eckpunkten
  - z.B. bei $S(0,0)$ tangential zur Ebene aufgespannt von $\mathbf{b}_{1,0} - \mathbf{b}_{0,0}$ und $\mathbf{b}_{0,1} - \mathbf{b}_{0,0}$

- die Randkurven der Fläche sind Bézierkurven
  - z.B. $S(0, v) = F(v)$ ist eine Bézierkurve

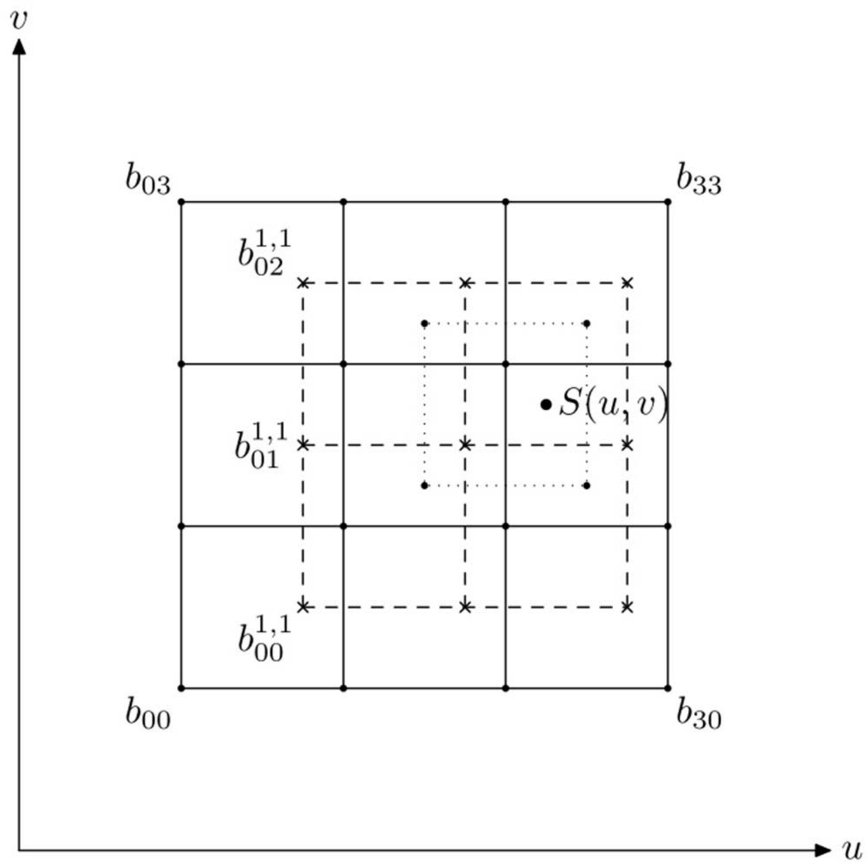
- affine Invarianz

- Variationsreduzierung **gilt nicht**!

Tensorprodukt-Bézier-Flächen

Auswertung für TP-Bézier-Flächen

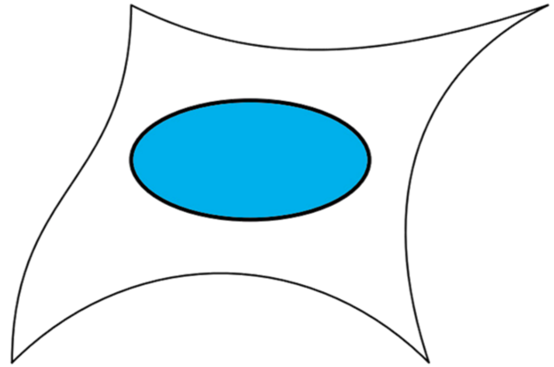
- de Casteljau für Bézierkurven basiert auf fortgesetzter linearer Interpolation
- durch bilineare Interpolation lässt sich ein de Casteljau-Algorithmus für Flächen formulieren



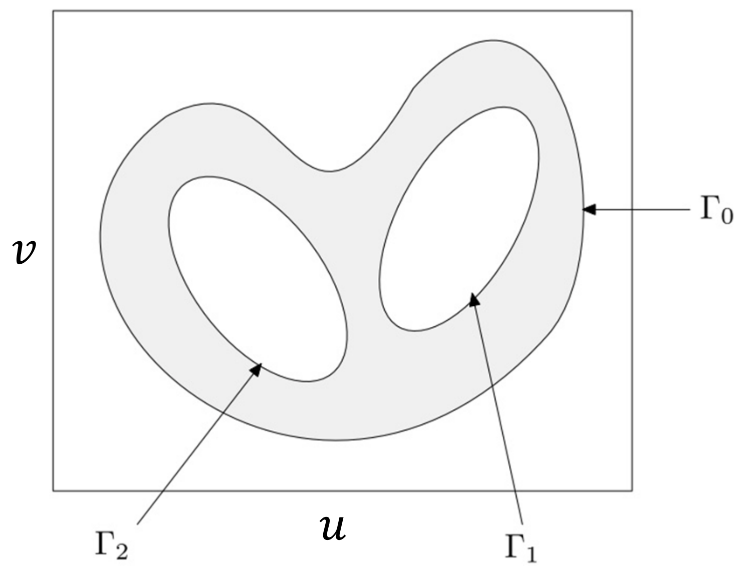
Quake 3 Arena verwendet bikubische Bézier-Flächen

Ausblicke

- Trimming („Zuschneiden") von Flächen:



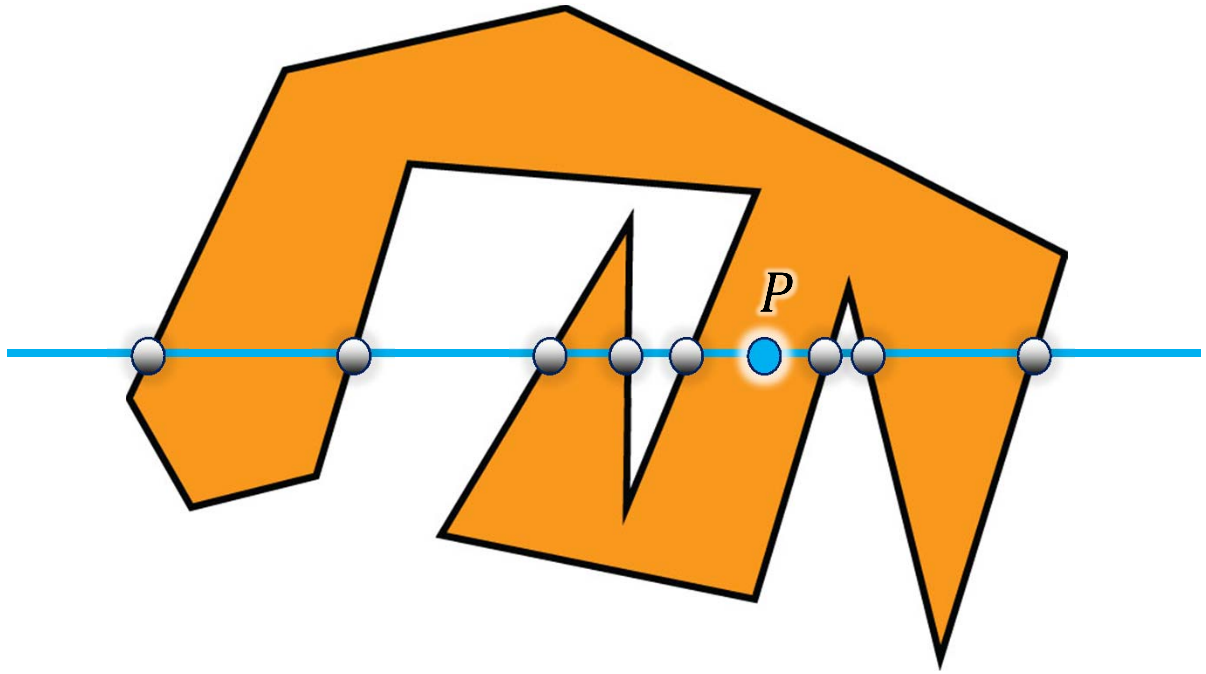
- bestimme eine Menge von Konturen Γ_i im Parametergebiet die festlegen, ob ein Punkt $S(u, v)$ zur Fläche gehört
  - Test ob ein Punkt (u, v) innerhalb liegt mit Odd-Even-Test



Allgemeiner Punkt-in-Polygon Test

Odd-Even Test (für Polygone, Trim-Kurven, Oberflächen in 3D)

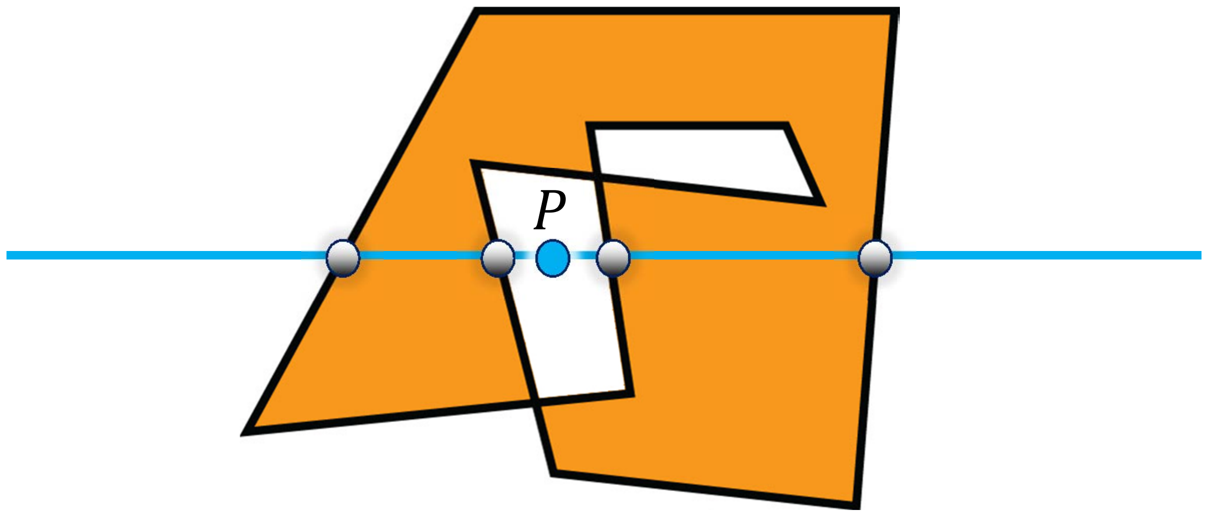
- Test für Punkt $P = (x_p, y_p)$
  - betrachte alle Schnitte der Gerade $y = y_p$ mit Kanten (Flächen in 3D)
  - P liegt im Polygon $\Leftrightarrow$ links und rechts eine **ungerade** Zahl von Schnitten existiert
  - P liegt außerhalb $\Leftrightarrow$ links und rechts eine **gerade** Zahl von Schnitten existiert



Allgemeiner Punkt-in-Polygon Test

Odd-Even Test (für Polygone, Trim-Kurven, Oberflächen in 3D)

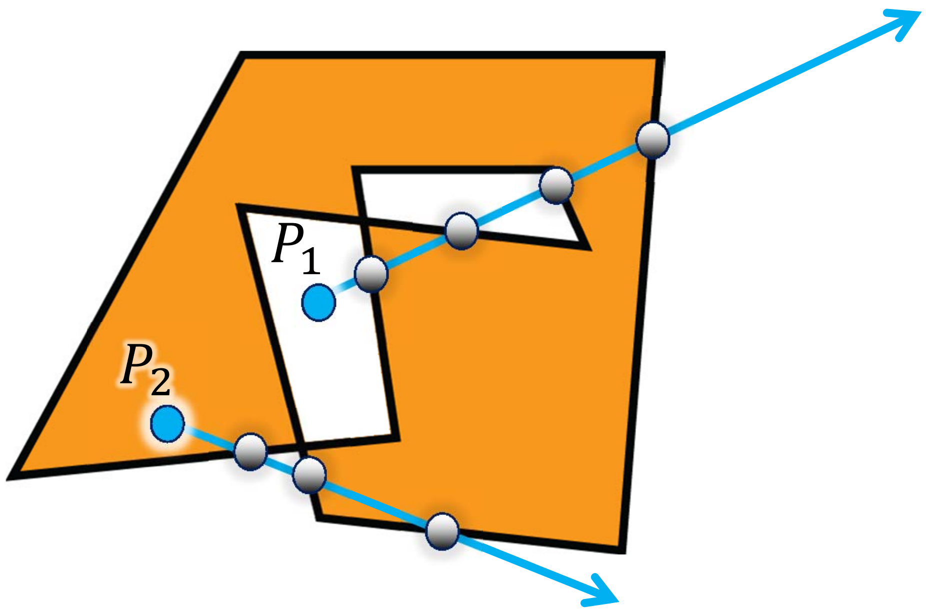
- Test für Punkt $P = (x_p, y_p)$
  - betrachte alle Schnitte der Gerade $y = y_p$ mit Kanten (Flächen in 3D)
  - P liegt im Polygon ⇔ links und rechts eine **ungerade** Zahl von Schnitten existiert
  - P liegt außerhalb ⇔ links und rechts eine **gerade** Zahl von Schnitten existiert



Allgemeiner Punkt-in-Polygon Test

Odd-Even Test (für Polygone, Trim-Kurven, Oberflächen in 3D)

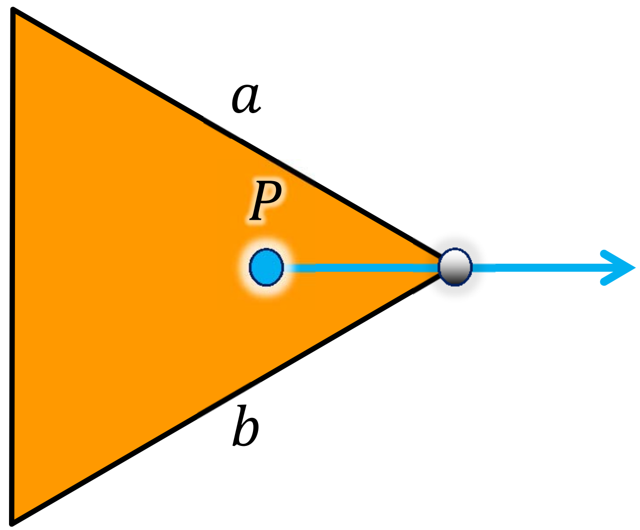
- Test für Punkt $P = (x_p, y_p)$
  - es genügt auch die Schnitte entlang einer Halbgerade von P in eine beliebige Richtung zu zählen
  - P liegt im Polygon $\Leftrightarrow$ **ungerade** Anzahl
  - P liegt außerhalb $\Leftrightarrow$ **gerade** Anzahl

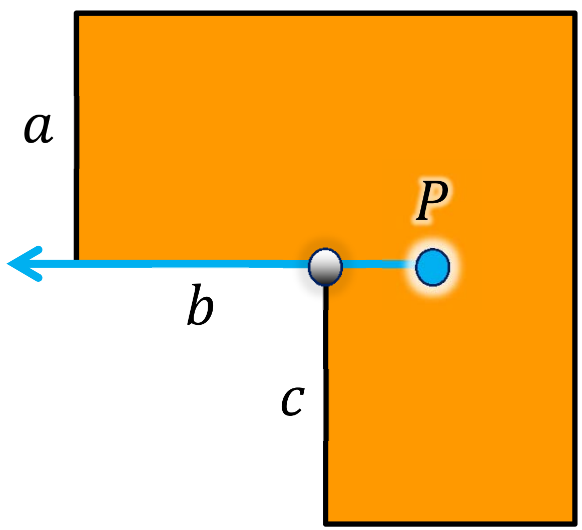


Allgemeiner Punkt-in-Polygon Test

Odd-Even Test (für Polygone, Trim-Kurven, Oberflächen in 3D)

- Spezialfälle (Polygone)
  - liegt ein Eckpunkt des Polygons auf der Gerade $y = y_p$, dann darf der Schnitt nur einmal gezählt werden (z.B. nur für Kante b im linken Bild)
  - zähle Schnitte nicht, für Kanten die auf oder über der Geraden $y = y_p$ liegen (rechtes Beispiel: der Schnitt zählt nur für Kante c)

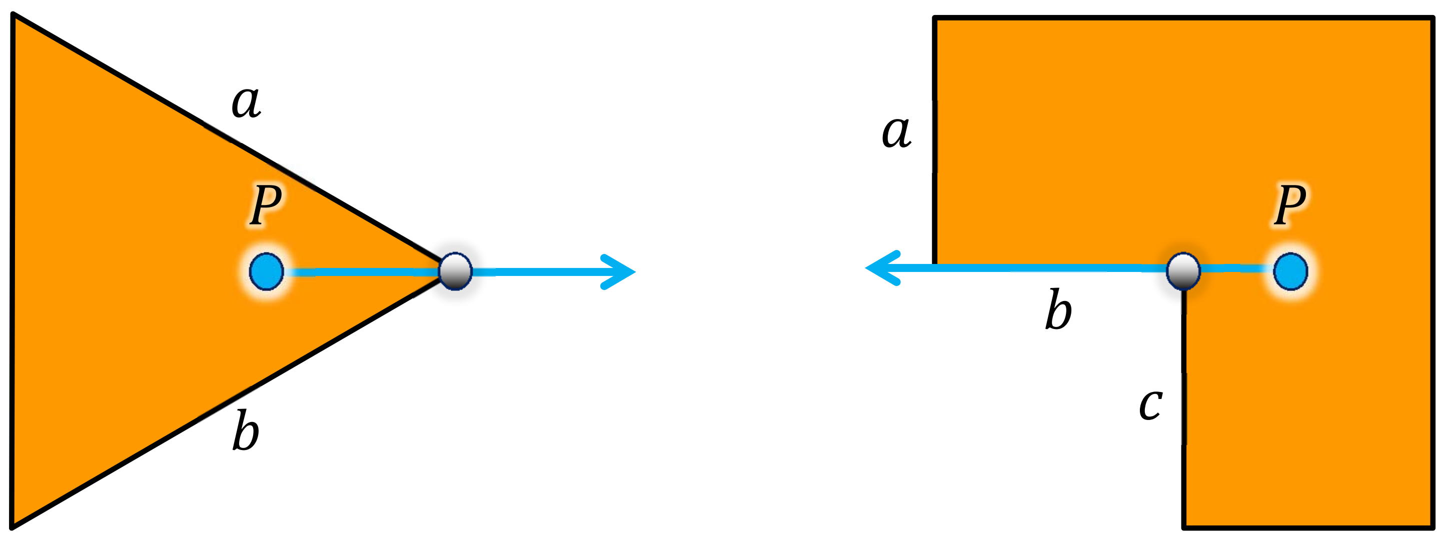




Allgemeiner Punkt-in-Polygon Test

Odd-Even Test (für Polygone, Trim-Kurven, Oberflächen in 3D)

- Computer Graphics: Principles and Practice
  (Foley, Van Dam, Feiner, Hughes)
  - Seite 34: "Next, choose a ray that starts at the test point and extends infinitely in any direction, and that does not pass through any vertices“
  - Seite 339: "intersections at vertices" are a "special case“
- Lösung: http://jedi.ks.uiuc.edu/~johns/raytracer/rtn/rtnv3n4.html#art22

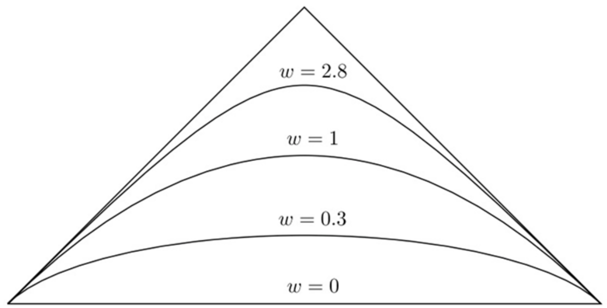


Ausblicke

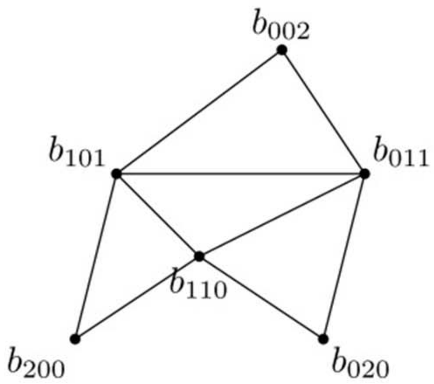
- mit polynomiellen Kurven ist es nicht möglich Kegelschnitte (z.B. einen Kreisbogen) exakt darzustellen
  - rationale Bézierkurven und rationale B-Spline-Kurven (NURBS)
  - rationale Bézierkurve mit Gewichten w_i:

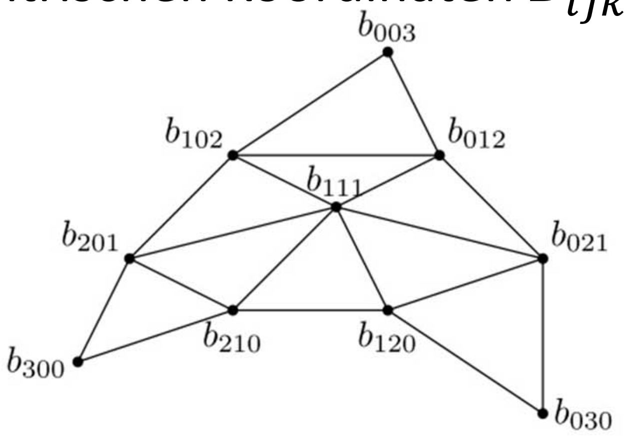
$$F(u) = \frac{\sum_{i=0}^{n} B_i^n(u) w_i \mathbf{b}_i}{\sum_{i=0}^{n} B_i^n(u) w_i}$$

  - Beispiel: quadratische rationale Bézierkurve



- Flächen lassen sich in OpenGL-Shadern tessellieren und auswerten
- Dreiecks-Bézierflächen: mehr topologische Flexibilität
  - Bernstein-Polynome in baryzentrischen Koordinaten $B_{ijk}^n(t_1, t_2, t_3)$





Lehrangebot Computergrafik

Vorlesung Computergraphik (6 ECTS)
(2D/3D Grafik, Raytracing, Texturen, Grafik-Hardware und OpenGL, …)

Praktikum: Grafik-Programmierung und Anwendungen (6 ECTS)

Computergrafik

- **Photorealistische Bildsynthese (5 ECTS)**
- **Interaktive Computergrafik (5 ECTS)**
- Visual Computing (geplant, 5 ECTS)

Visualisierung

- **Visualisierung (5 ECTS)**
- Advanced Visualization (geplant)

Seminare, Praktika

- Praktikum (3 ECTS) **GPGPU**
- Praktikum (6 ECTS) **GPU Computing**
- Proseminar (3 ECTS) **Algo. für Videospiele**
- Seminar (3 ECTS) **Rendering Techniken**

Angewandte Geometrie

- **Geometrische Optimierung**
- **Kurven und Flächen im CAD 1-3**
- **Netze und Punktwolken**
- **Unterteilungs-algorithmen**
- **Rationale Splines**

Bachelor- und Master-Arbeiten, HiWi Jobs

Klausur

- **Wann?** 11. März 2015 um 14.00 Uhr
- **Wo?** Hörsaal „Daimler“, „Benz“, „Grashof“
  - **wir kündigen auf der Webseite an, wer in welchem Hörsaal schreibt**
- **Wie lange?** 60 Minuten (plus 5 Minuten Lesezeit)
- **Anmeldung nicht vergessen!**
  - Anmeldung: 11. Februar 2013 bis 4. März 2013
    (Abmeldung bis 6. März möglich)
- Sonstiges…
  - keine Hilfsmittel
  - Studentenausweis nicht vergessen!
- Alte Klausuren: Webseite! … und vor dem Sekretariat
- Nachklausur:
  - 10. April 2013, 11 Uhr, Hörsaal „Daimler“ und „Benz“
  - → immer Ankündigungen auf der Webseite beachten!