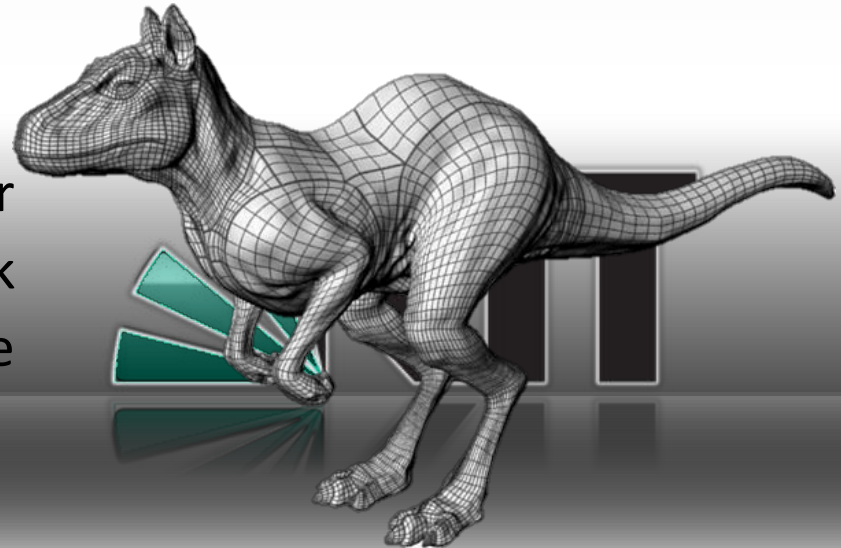


Computergrafik

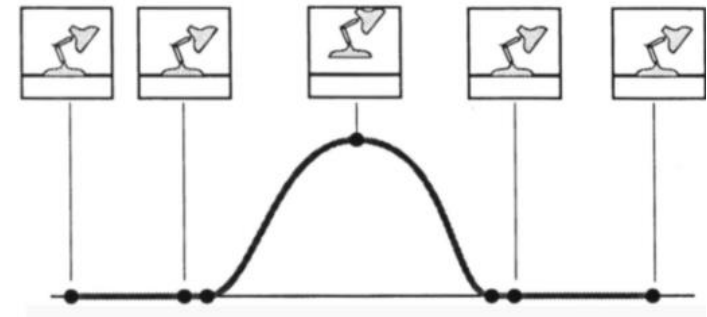
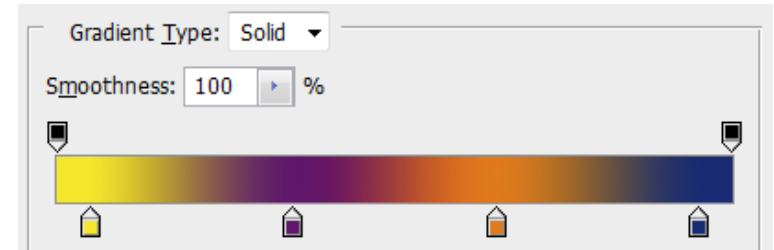
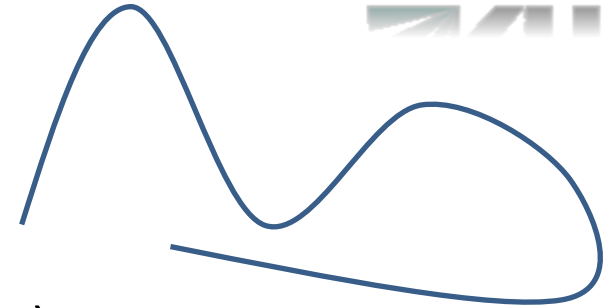
Vorlesung im Wintersemester 2024/25 Kapitel 9: Kurven und Flächen

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



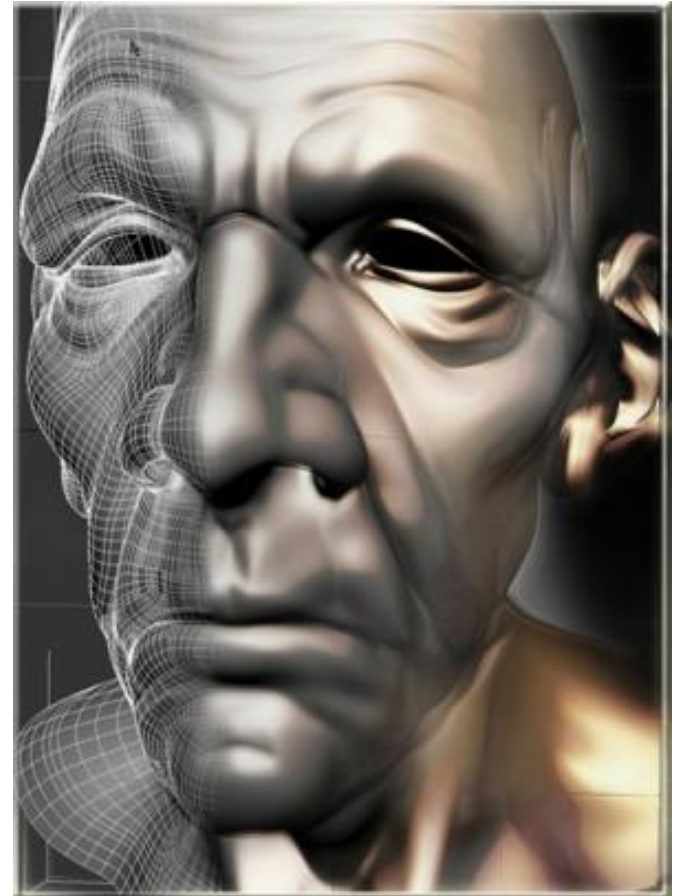
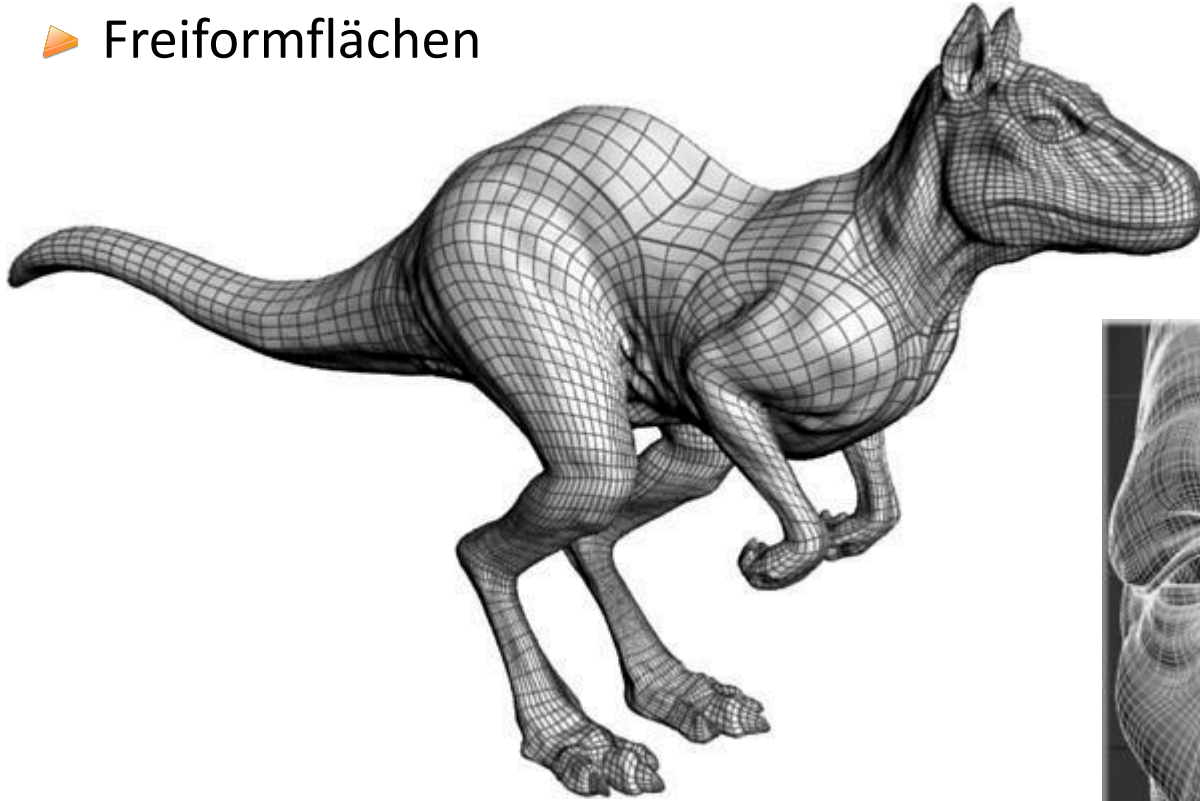
Kurven und Flächen

- ▶ 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



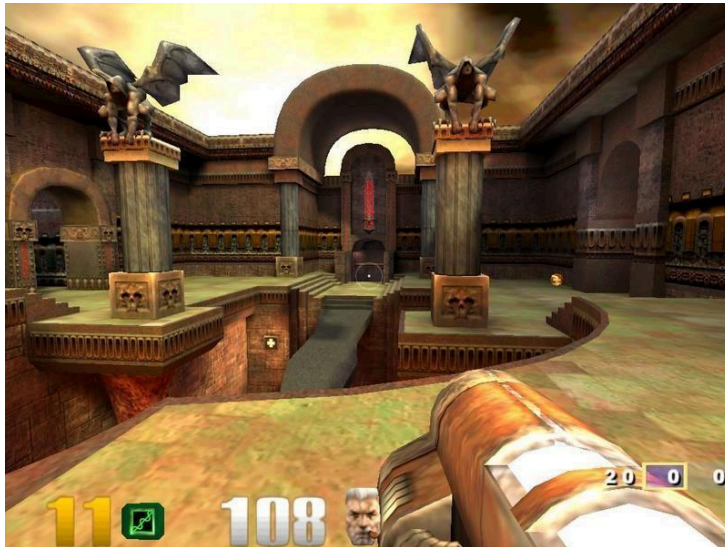
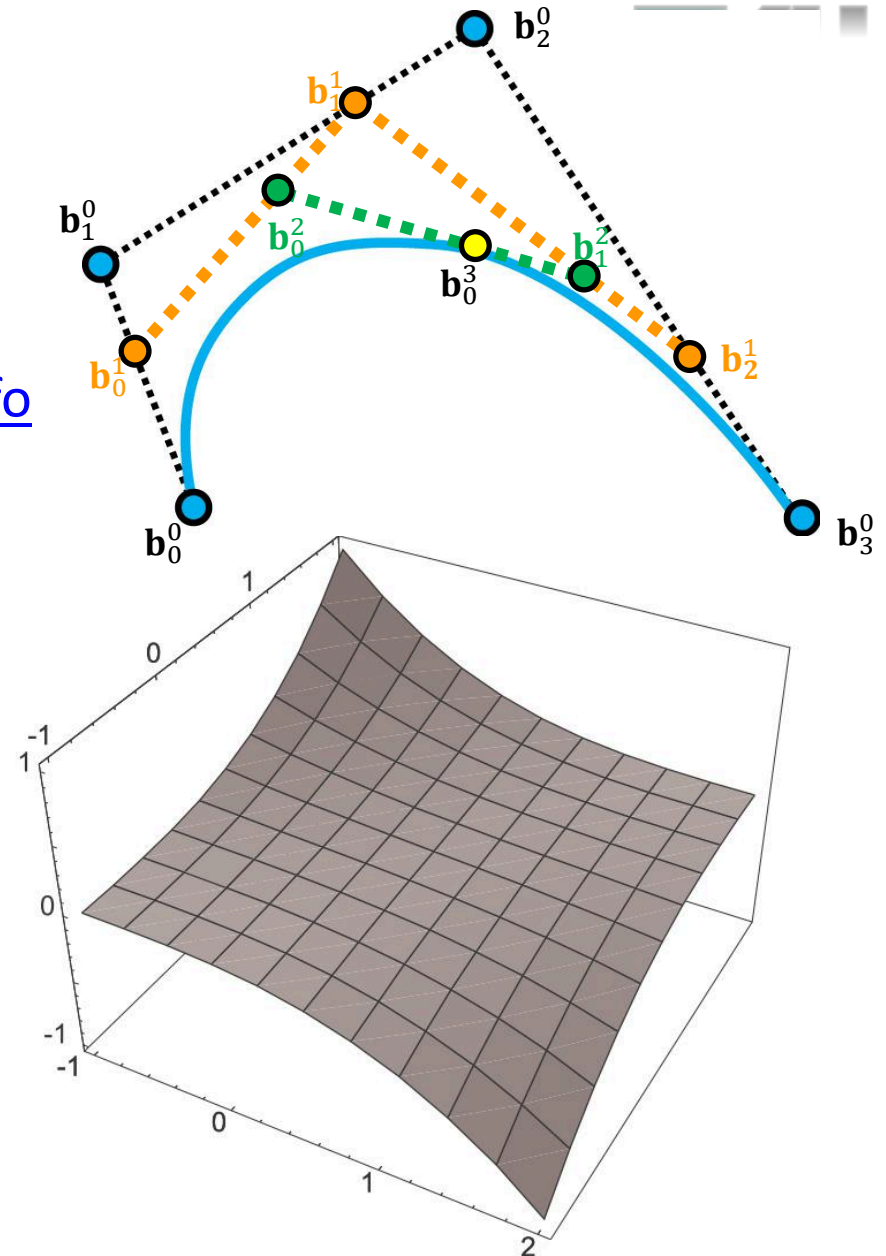
ACM © 1987 "Principles of traditional animation applied to 3D computer animation"

▶ Freiformflächen



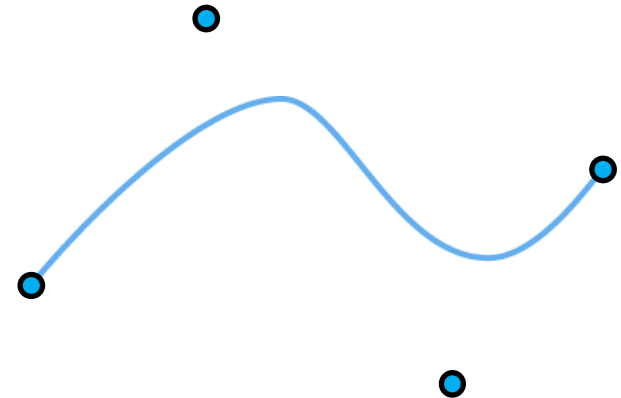
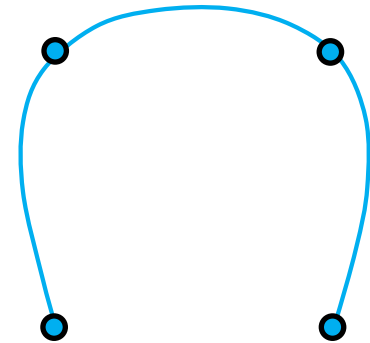
Kurven und Flächen – Inhalt

- ▶ Polynom- und Bézier-Kurven
 - ▶ Bernstein Polynome
 - ▶ de Causteljau-Algorithmus
 - ▶ freies Buch: <https://pomax.github.io/bezierinfo>
 - ▶ Video: <https://www.youtube.com/watch?v=aVwxzDHniEw>
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



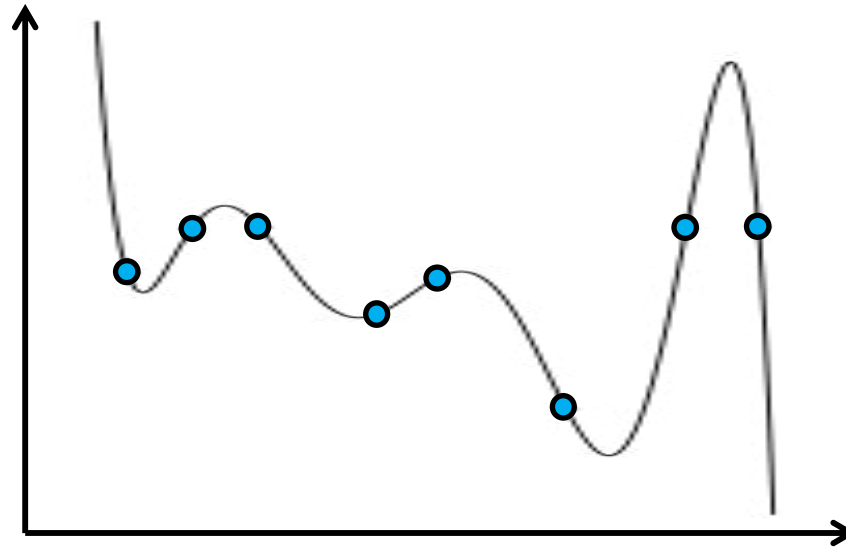
Sichtweisen und Anwendungen

- ▶ Definieren einer glatten parametrischen Kurve $P(u)$ durch Platzieren einiger sog. **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



Kurven und Splines

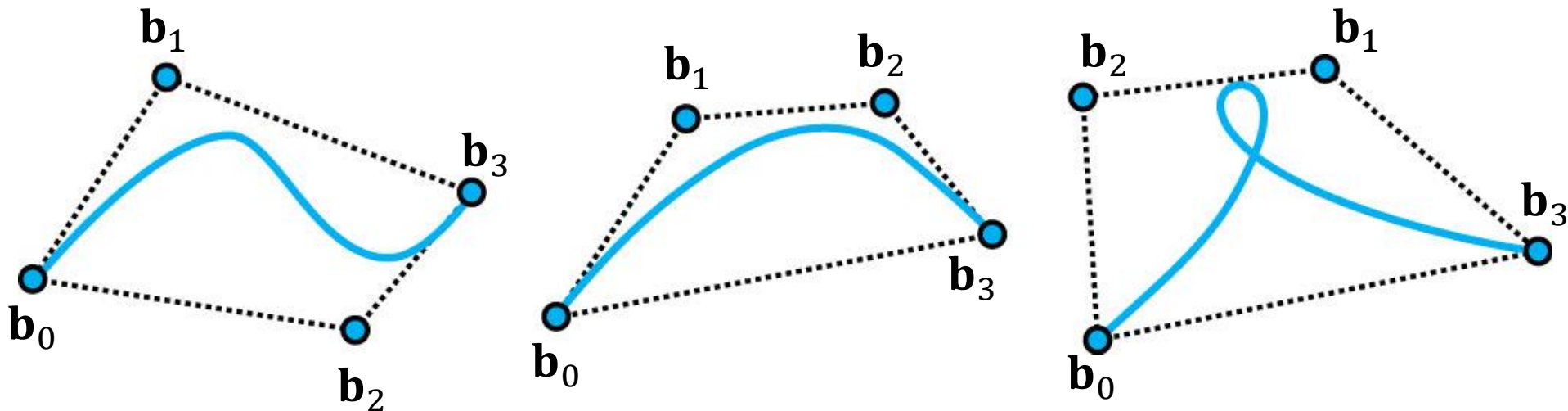
- ▶ Interpolation führt i.d.R. zu Überschwingern
- ▶ Beispiel: Lagrange-Interpolation (Polynom 7. Grades)



Beispiel: Kubische Bézier-Kurven

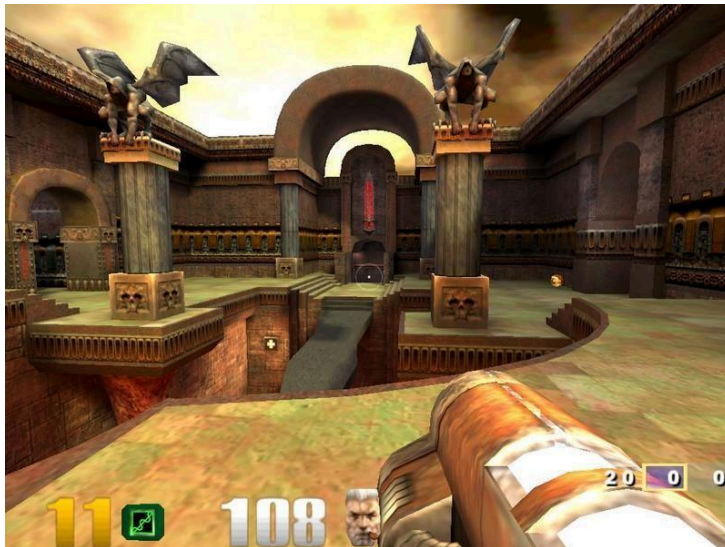
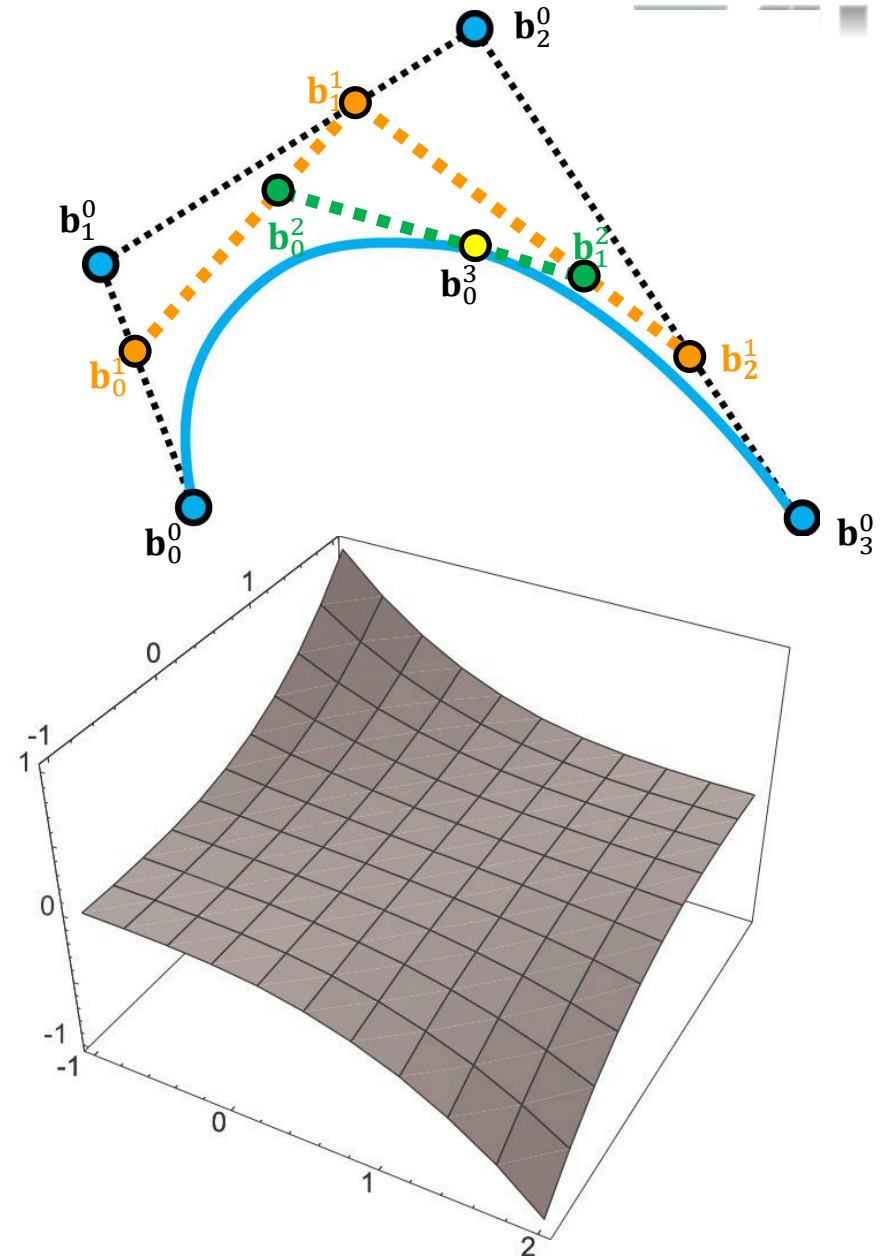
Einführungsbeispiel kubische Bézier-Kurven

- ▶ mathematische Beschreibung mit einem kubischen Polynom
- ▶ 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 Bézier-Kurve liegt innerhalb der konvexen Hülle ihrer Kontrollpunkte



Kurven und Flächen – Inhalt

- ▶ Polynomkurven
- ▶ Bernstein-Polynome
- ▶ Basiswechsel
- ▶ Eigenschaften von Bézier-Kurven
- ▶ de Casteljau-Algorithmus
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



Polynomkurven

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

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

- ▶ 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)

▶ $B_0^3(u) = \binom{3}{0} u^0 (1-u)^3$

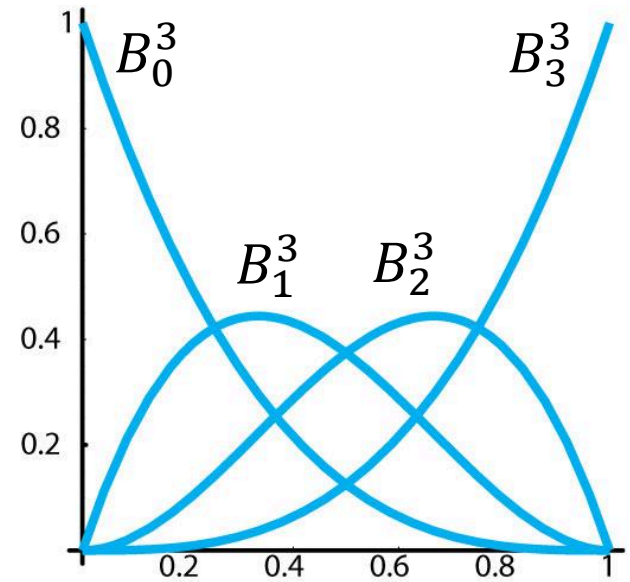
▶ $B_1^3(u) = \binom{3}{1} u^1 (1-u)^2$

▶ $B_2^3(u) = \binom{3}{2} u^2 (1-u)^1$

▶ $B_3^3(u) = \binom{3}{3} u^3 (1-u)^0$



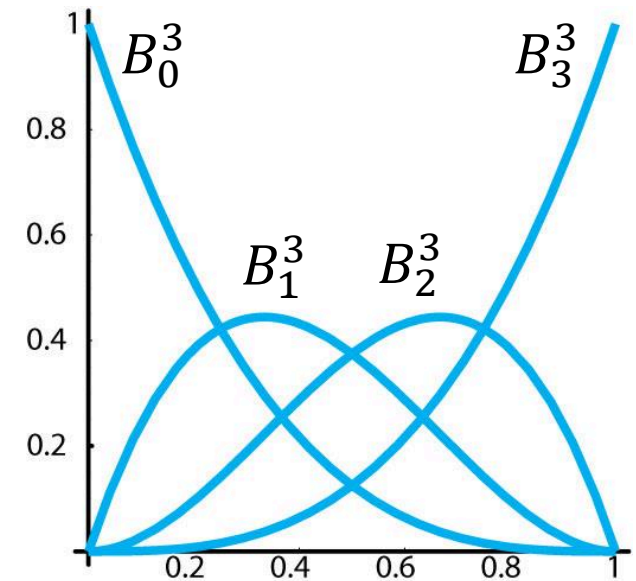
Binomialkoeffizient



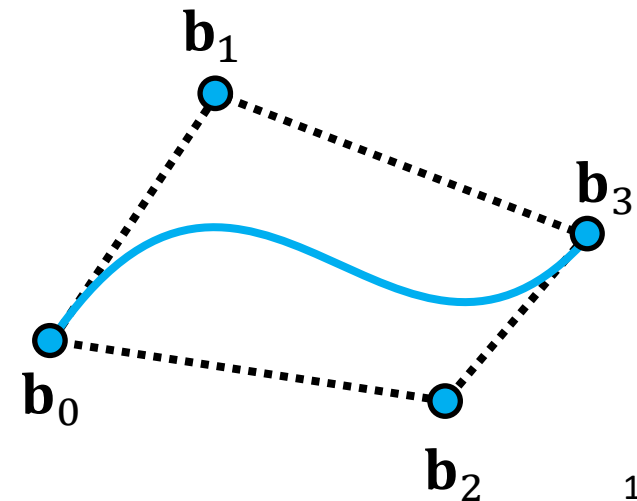
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

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



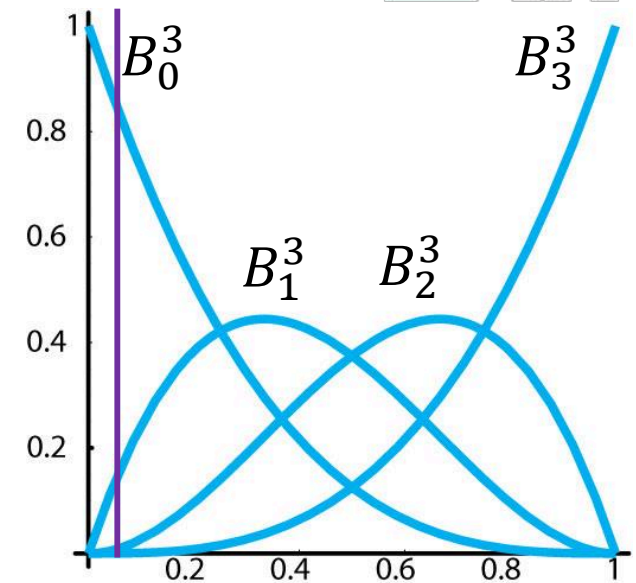
- ▶ Bézier-Kurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- ▶ anschaulich: jedes B_i^3 legt den Einfluss von \mathbf{b}_i auf die Kurve an der Stelle u fest
 - ▶ erst hat \mathbf{b}_0 den größten Einfluss, dann \mathbf{b}_1 , \mathbf{b}_2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



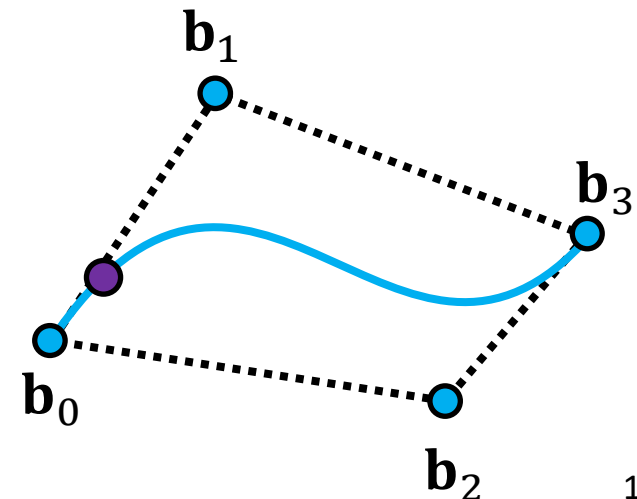
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

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



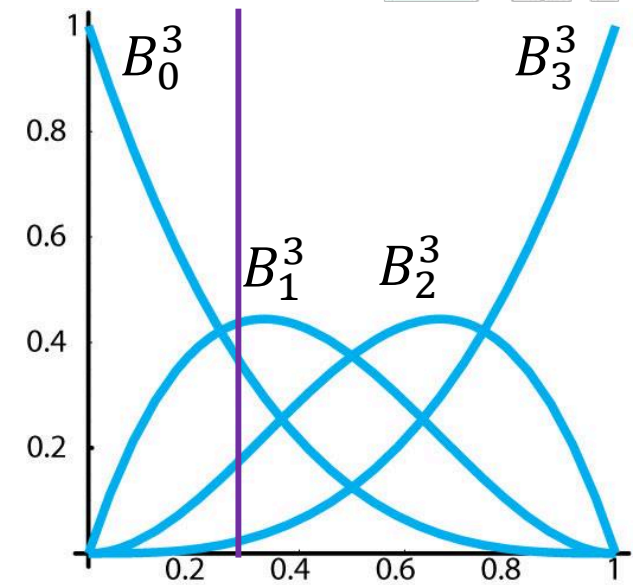
- ▶ Bézier-Kurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- ▶ anschaulich: jedes B_i^3 legt den Einfluss von \mathbf{b}_i auf die Kurve an der Stelle u fest
 - ▶ erst hat \mathbf{b}_0 den größten Einfluss, dann \mathbf{b}_1 , \mathbf{b}_2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



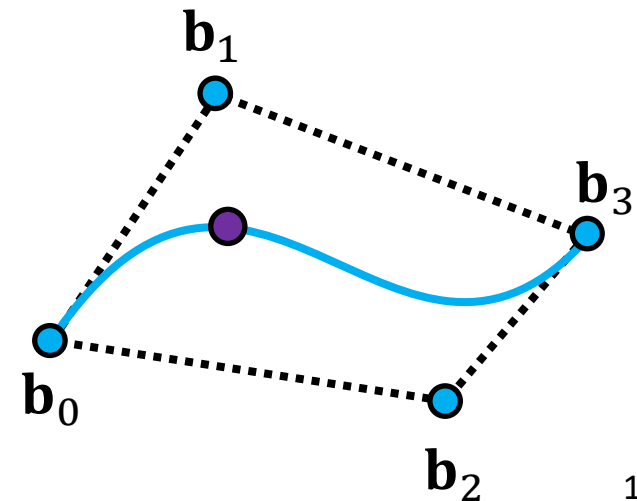
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

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



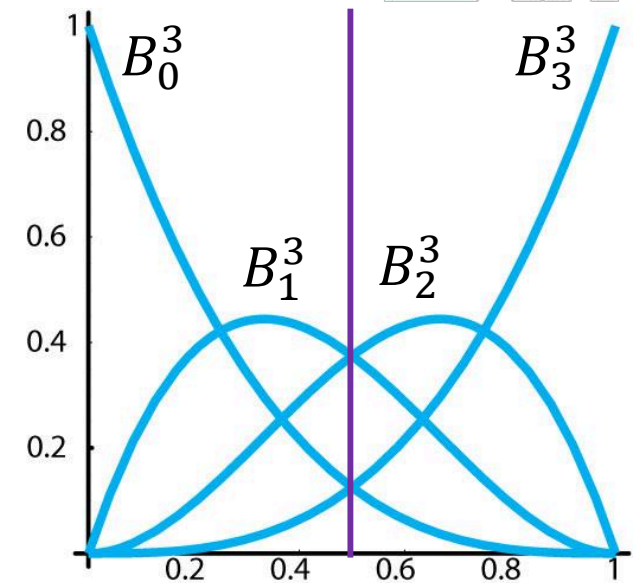
- ▶ Bézier-Kurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- ▶ anschaulich: jedes B_i^3 legt den Einfluss von \mathbf{b}_i auf die Kurve an der Stelle u fest
 - ▶ erst hat \mathbf{b}_0 den größten Einfluss, dann \mathbf{b}_1 , \mathbf{b}_2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



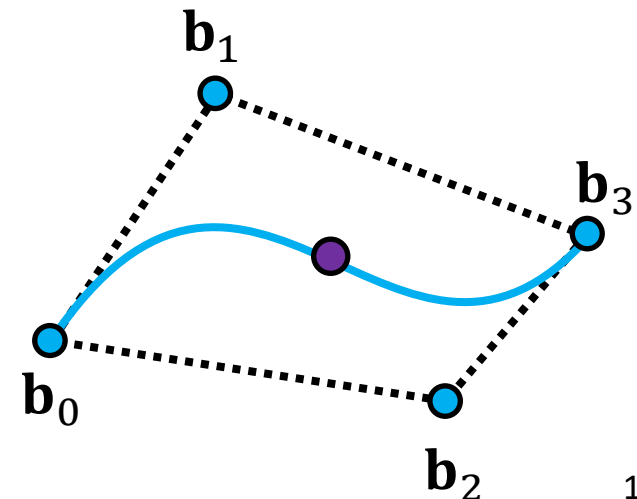
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

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



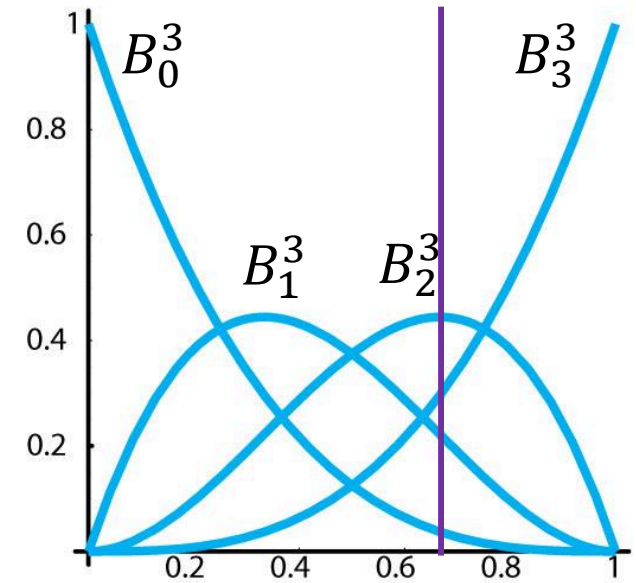
- ▶ Bézier-Kurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- ▶ anschaulich: jedes B_i^3 legt den Einfluss von \mathbf{b}_i auf die Kurve an der Stelle u fest
 - ▶ erst hat \mathbf{b}_0 den größten Einfluss, dann \mathbf{b}_1 , \mathbf{b}_2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



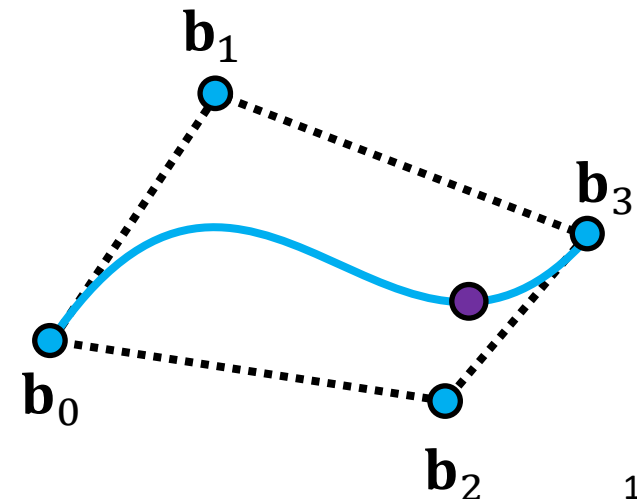
Bernstein-Polynome

Bernstein-Polynome als Gewichtungsfunktion

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



- ▶ Bézier-Kurve: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$
- ▶ 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



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 *Bézier-Kurve* vom Grad n , die Punkte \mathbf{b}_i heißen *Kontroll-* oder *Bézier-Punkte* und bilden das *Kontrollpolygon*

- ▶ auch bezeichnet als Bézier-Repräsentation einer Polynomkurve
- ▶ formalisiert und bekannt geworden durch *Pierre Étienne Bézier* 1962
- ▶ Algorithmen für effiziente Auswertung von Polynomkurven von *Paul de Casteljau* 1959
- ▶ entwickelt für Karosseriedesign bei Citroën

Definition (nach Sergej 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 $i > n$,
um $B_i^n(u)$ rekursiv zu formulieren \rightarrow gleich!

▶ Binomialkoeffizient $\binom{n}{k} := \frac{n!}{k!(n-k)!}$

▶ Symmetrie $\binom{n}{k} = \binom{n}{n-k}$

▶ 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: $\binom{n+1}{k+1} := \binom{n}{k+1} + \binom{n}{k}$

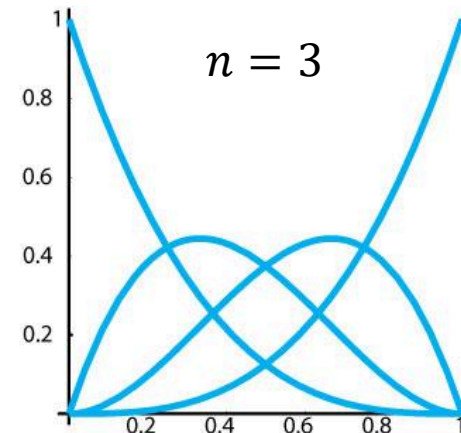
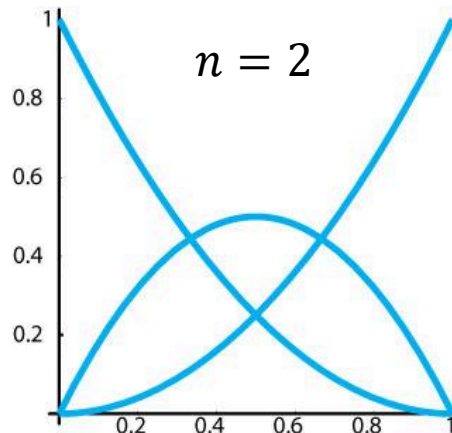
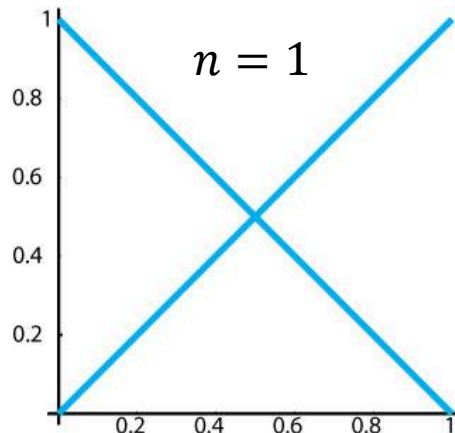
Eigenschaften

- ▶ i -tes Bernstein-Polynom vom Grad n

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

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

← entscheidend für die Eigenschaften der Kurve!



Eigenschaften

- ▶ i -tes Bernstein-Polynom vom Grad n

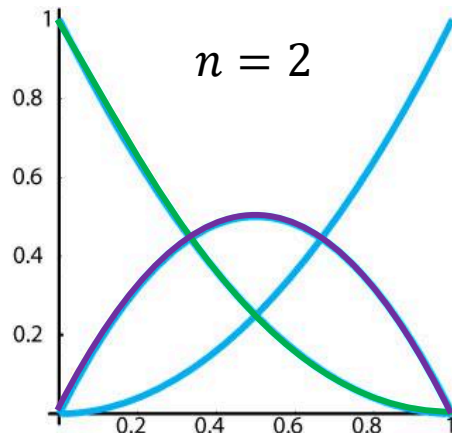
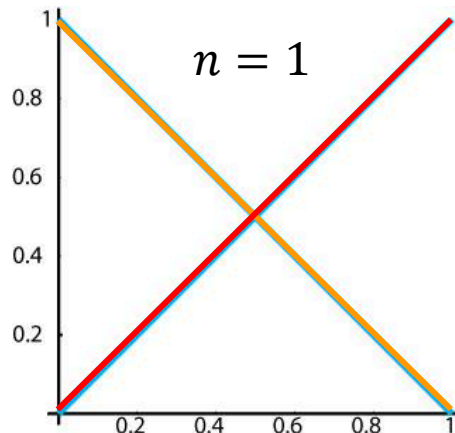
$$B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}, \text{ mit } i = 0, \dots, n$$

- ▶ $B_i^n(u) = u \cdot B_{i-1}^{n-1}(u) + (1-u) \cdot B_i^{n-1}(u)$
(Rekursion, Eigenschaft wichtig für effiziente Auswertungsalgorithmen!)

- ▶ $B_0^1(u) = 1 - u$ und $B_1^1(u) = u$

- ▶ $B_0^2(u) = u \cdot B_{-1}^1(u) + (1-u) \cdot B_0^1(u) = (1-u)^2 = 1 - 2u + u^2$

- ▶ $B_1^2(u) = \underbrace{u \cdot B_0^1(u)}_{=u} + (1-u) \cdot B_1^1(u) = u(1-u) + (1-u)u = 2u - 2u^2$

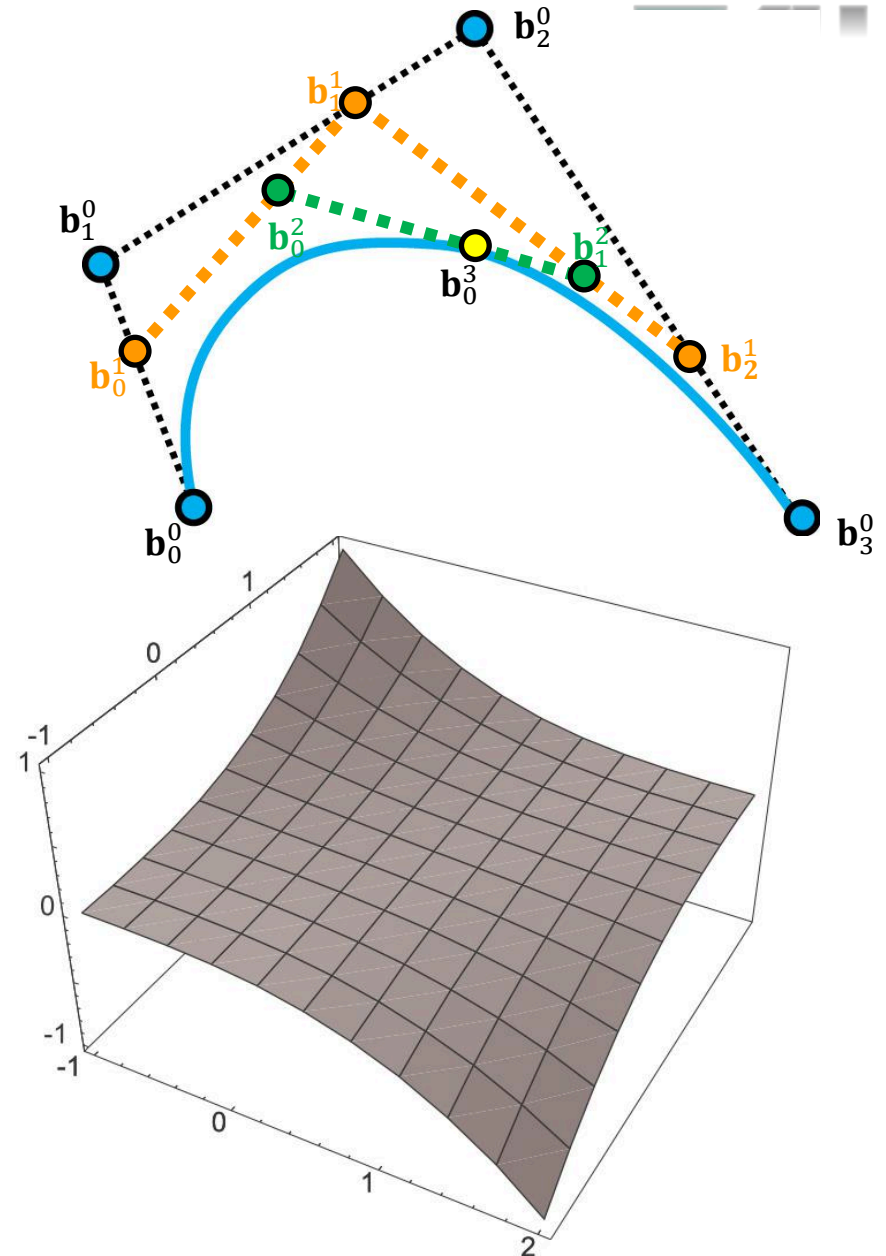
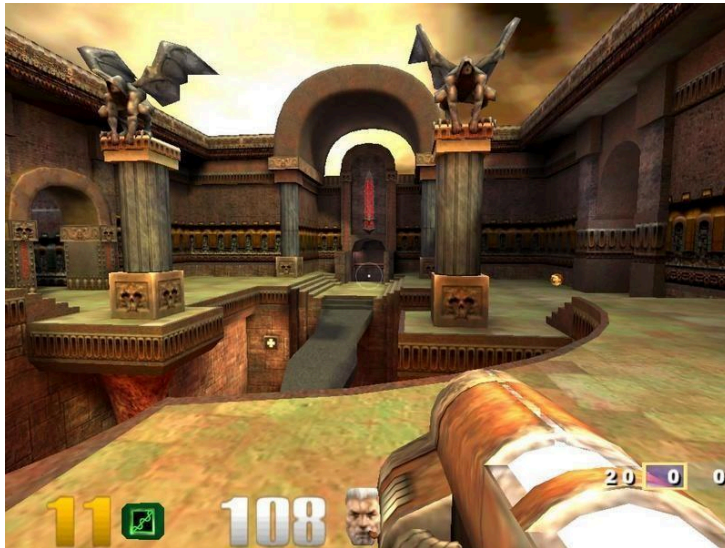


$$= 2u(1-u)$$

$$= \binom{2}{1} u^1 (1-u)^{2-1}$$

Kurven und Flächen – Inhalt

- ▶ Polynomkurven
- ▶ Bernstein-Polynome
- ▶ Basiswechsel
- ▶ Eigenschaften von Bézier-Kurven
- ▶ de Casteljau-Algorithmus
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



- ▶ die Bernstein-Polynome B_i^n bilden eine Basis des \mathbb{P}_n
 - ▶ wie sind die Koeffizienten einer Bézier-Kurve 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:

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

- ▶ Beweis:

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

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

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

- ▶ die Bernstein-Polynome B_i^n bilden eine Basis des \mathbb{P}_n
 - ▶ wie sind die Koeffizienten einer Bézier-Kurve 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 $n = 2$: $F(u) = \mathbf{b}_0 B_0^2(u) + \mathbf{b}_1 B_1^2(u) + \mathbf{b}_2 B_2^2(u)$

▶ $B_0^2(u) = \binom{2}{0} u^0 (1-u)^{2-0} = (1-u)^2 = 1 - 2u + u^2$

▶ $B_1^2(u) = \binom{2}{1} u^1 (1-u)^{2-1} = 2u(1-u) = 2u - 2u^2$

▶ $B_2^2(u) = \binom{2}{2} u^2 (1-u)^{2-2} = u^2$

▶
$$\begin{pmatrix} B_0^2 \\ B_1^2 \\ B_2^2 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{pmatrix}}_M \begin{pmatrix} 1 \\ u \\ u^2 \end{pmatrix} \Rightarrow \begin{pmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{pmatrix}}_{M^T} \begin{pmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$$

- ▶ die Bernstein-Polynome B_i^n bilden eine Basis des \mathbb{P}_n
 - ▶ wie sind die Koeffizienten einer Bézier-Kurve 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 $n = 3$: $F(u) = \mathbf{b}_0 B_0^3(u) + \mathbf{b}_1 B_1^3(u) + \mathbf{b}_2 B_2^3(u) + \mathbf{b}_3 B_3^3(u)$

▶ $B_0^3(u) = (1 - u)^3$

▶ $B_1^3(u) = 3u(1 - u)^2$

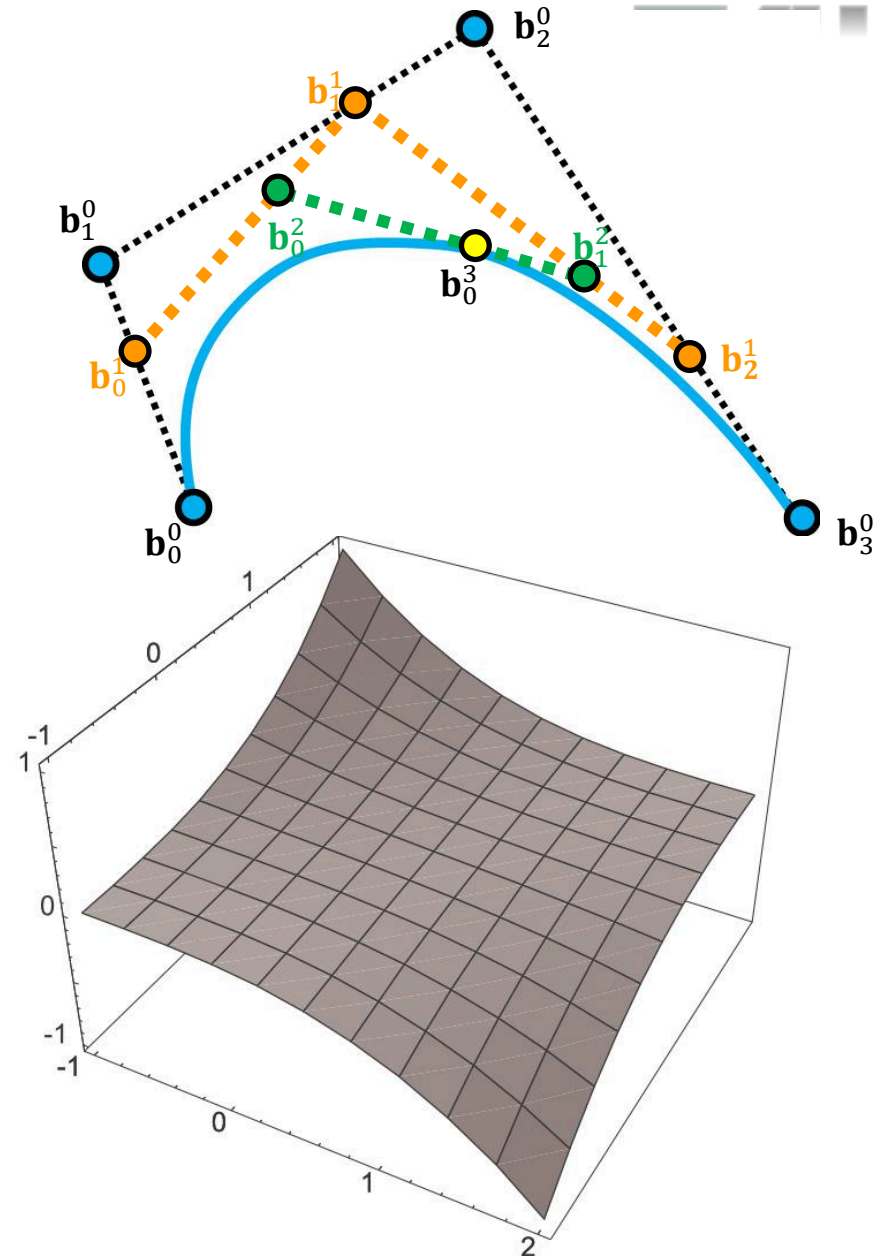
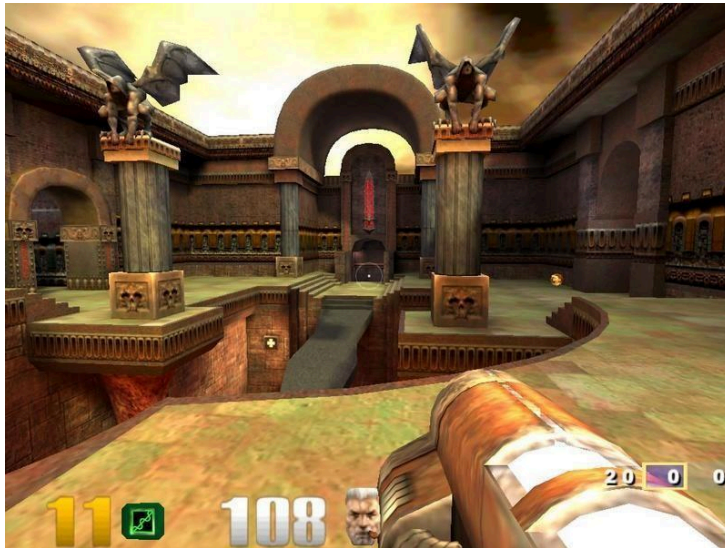
▶ $B_2^3(u) = 3u^2(1 - u)$

▶ $B_3^3(u) = u^3$

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

Kurven und Flächen – Inhalt

- ▶ Polynomkurven
- ▶ Bernstein-Polynome
- ▶ Basiswechsel
- ▶ Eigenschaften von Bézier-Kurven
- ▶ de Casteljau-Algorithmus
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



Eigenschaften (Lemma von Bézier)

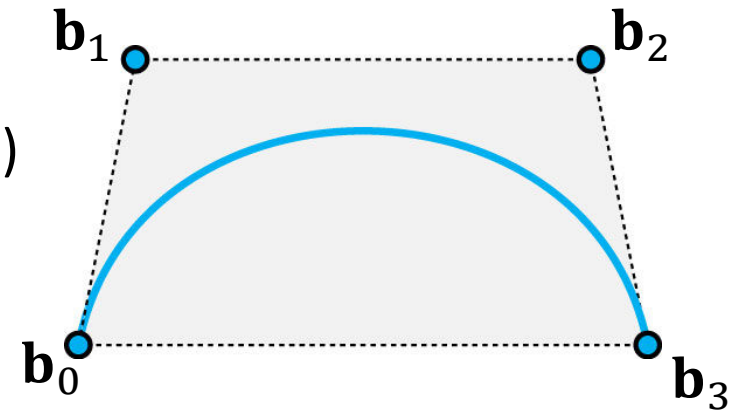
▶ für $F(u) = \sum_{i=0}^n B_i^n(u) \mathbf{b}_i$ gilt

▶ $F(u)$ liegt in der abgeschlossenen konvexen Hülle des Kontrollpolygons 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 $F(1) = \mathbf{b}_n$

▶ **Tangentenbedingung** $F'(0) = n(\mathbf{b}_1 - \mathbf{b}_0)$ und $F'(1) = n(\mathbf{b}_n - \mathbf{b}_{n-1})$

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

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

Eigenschaften (Lemma von Bézier)

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

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

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

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

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

Eigenschaften (Lemma von Bézier)

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

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

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

- ▶ Beweis: Translation \mathbf{t} für $n = 2$:

$$\begin{aligned} \text{▶ } F_{\mathbf{t}}(u) &= (\mathbf{b}_0 + \mathbf{t})B_0^2(u) + (\mathbf{b}_1 + \mathbf{t})B_1^2(u) + (\mathbf{b}_2 + \mathbf{t})B_2^2(u) = \\ &= \mathbf{b}_0B_0^2(u) + \mathbf{b}_1B_1^2(u) + \mathbf{b}_2B_2^2(u) + \mathbf{t} \underbrace{\left(B_0^2(u) + B_1^2(u) + B_2^2(u) \right)}_{=1} = \\ &= F(u) + \mathbf{t} \end{aligned}$$

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

Eigenschaften (Lemma von Bézier)

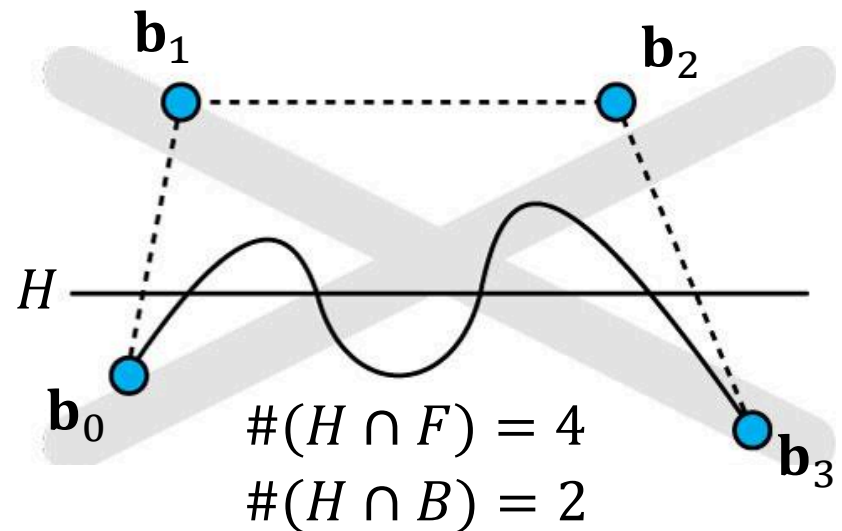
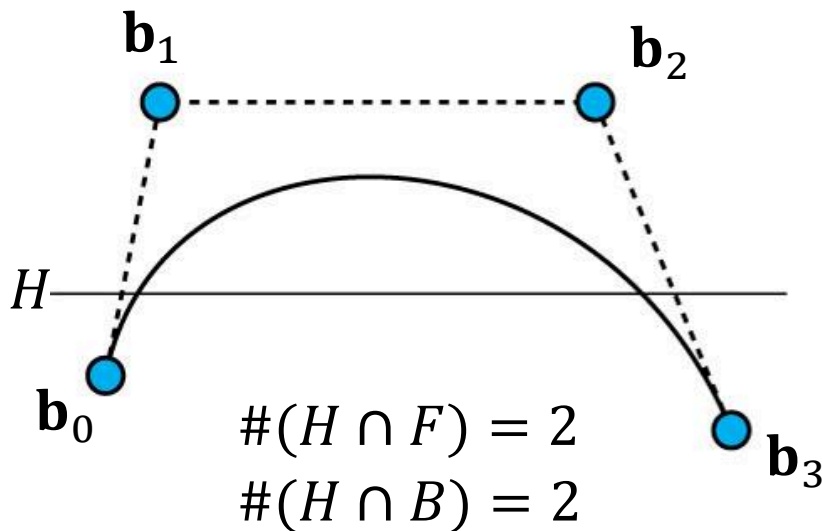
▶ für $F(u) = \sum_{i=0}^n B_i^n(u) \mathbf{b}_i$ gilt

▶ **Variationsreduzierung** („variation diminishing property“):

„Eine Bézier-Kurve F wackelt nicht stärker als ihr Kontrollpolygon B “

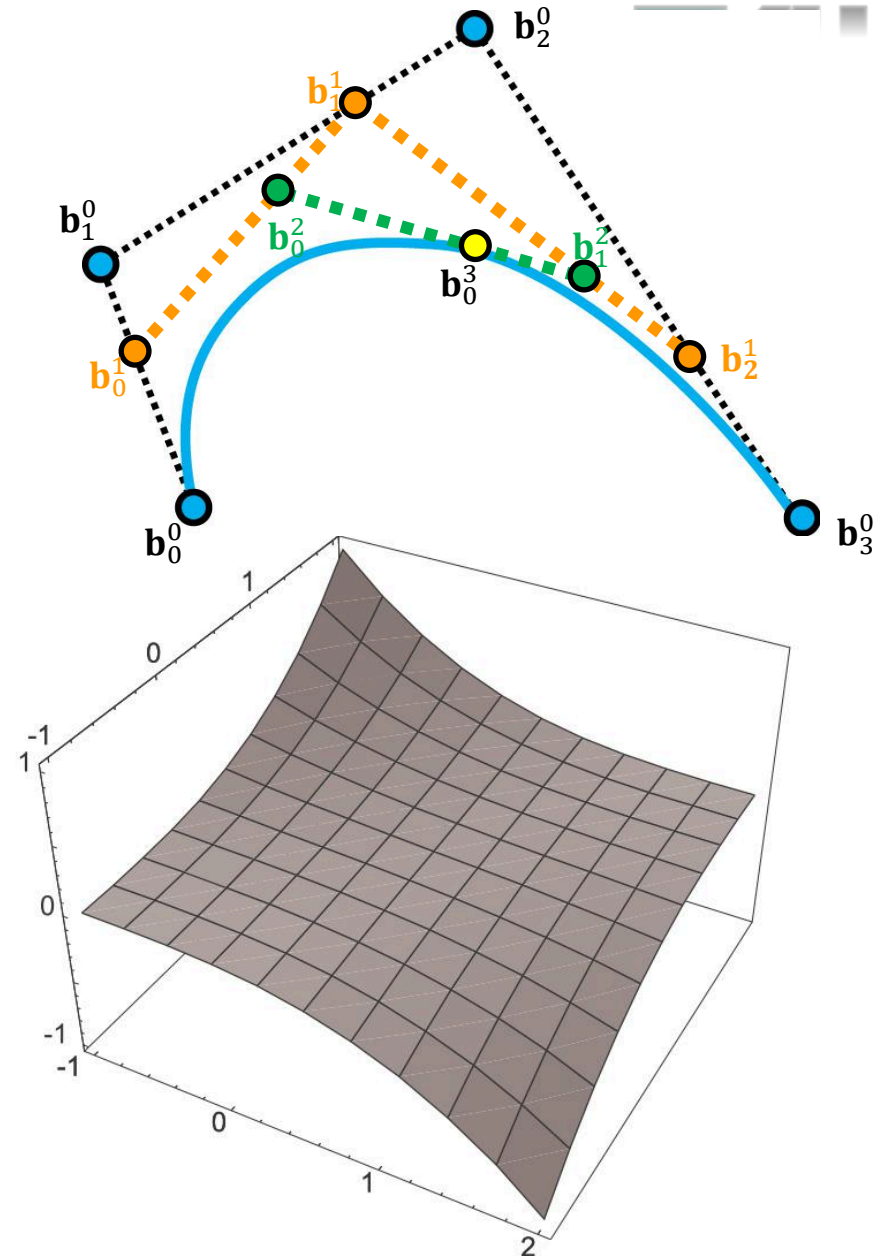
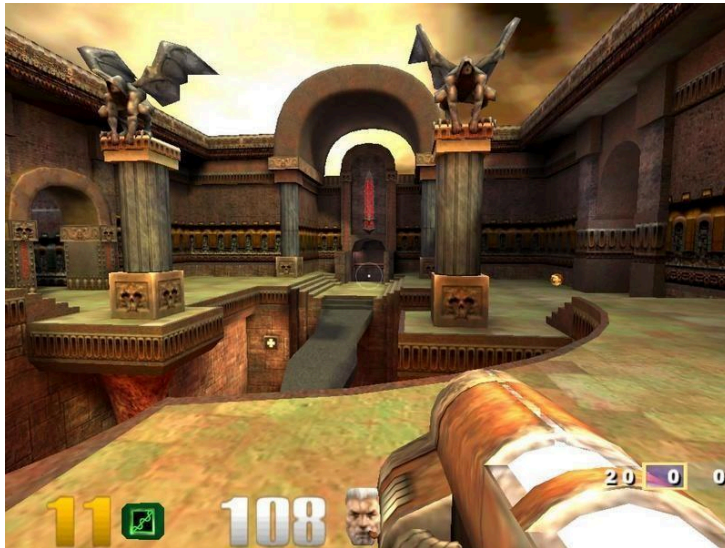
▶ sei H eine beliebige Hyperebene in \mathbb{R}^d (Gerade in \mathbb{R}^2 , Ebene in \mathbb{R}^3)
dann gilt $\#(H \cap F) \leq \#(H \cap B)$

▶ Beweis: G. Farin. „Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide“. Academic Press, 4. Auflage, 1997



Kurven und Flächen – Inhalt

- ▶ Polynomkurven
- ▶ Bernstein-Polynome
- ▶ Basiswechsel
- ▶ Eigenschaften von Bézier-Kurven
- ▶ de Casteljau-Algorithmus
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



Effiziente Auswertung und Unterteilung von Bézier-Kurven

- ▶ 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

- ▶ $\mathbf{b}_i^0 := \mathbf{b}_i$

- ▶ $\mathbf{b}_i^1 := (1-u) \cdot \mathbf{b}_i^0 + u \cdot \mathbf{b}_{i+1}^0$, usw.

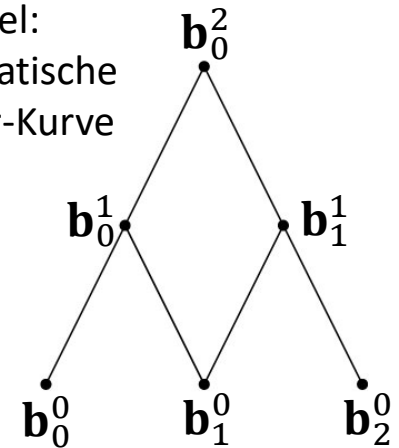
- ▶ allgemein:

$$\mathbf{b}_i^j := (1-u) \cdot \mathbf{b}_i^{j-1} + u \cdot \mathbf{b}_{i+1}^{j-1}, \quad i = 0, \dots, n-j$$

- ▶ $\Rightarrow \mathbf{b}_0^n = F(u)$ ist ein Punkt auf der Bézier-Kurve

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

Beispiel:
quadratische
Bézier-Kurve



Effiziente Auswertung und Unterteilung von Bézier-Kurven

▶ möglich durch die rekursive Darstellung der Bernstein-Polynome

▶ $\mathbf{b}_i^0 := \mathbf{b}_i$

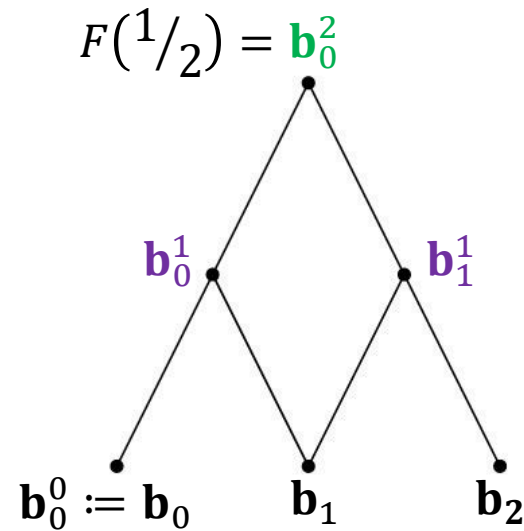
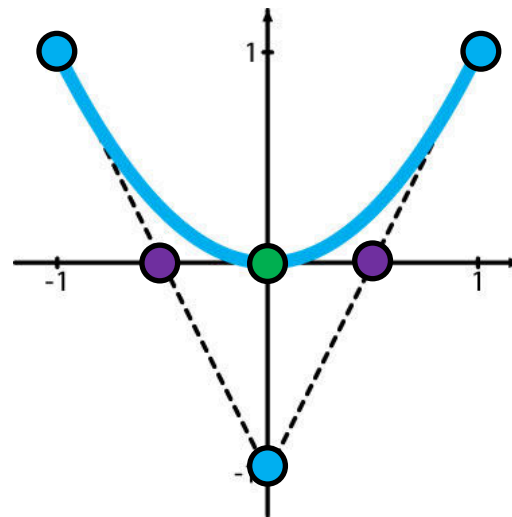
▶ $\mathbf{b}_i^j := (1 - u) \cdot \mathbf{b}_i^{j-1} + u \cdot \mathbf{b}_{i+1}^{j-1}, i = 0, \dots, n - j$

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

▶ $\mathbf{b}_0^1 = \frac{1}{2} \mathbf{b}_0 + \frac{1}{2} \mathbf{b}_1 = \begin{pmatrix} -1/2 \\ 0 \end{pmatrix}$

▶ $\mathbf{b}_1^1 = \frac{1}{2} \mathbf{b}_1 + \frac{1}{2} \mathbf{b}_2 = \begin{pmatrix} 1/2 \\ 0 \end{pmatrix}$

▶ $\mathbf{b}_0^2 = \frac{1}{2} \mathbf{b}_0^1 + \frac{1}{2} \mathbf{b}_1^1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$



Effiziente Auswertung und Unterteilung von Bézier-Kurven

▶ Beispiel: quadratische Bézier-Kurven

$$\text{▶ } B_0^2(u) = \binom{2}{0} u^0 (1-u)^{2-0} = (1-u)^2$$

$$\text{▶ } B_1^2(u) = \binom{2}{1} u^1 (1-u)^{2-1} = 2u(1-u)$$

$$\text{▶ } B_2^2(u) = \binom{2}{2} u^2 (1-u)^{2-2} = u^2$$

▶ Algorithmus zur Auswertung

$$\text{▶ } \mathbf{b}_i^0 := \mathbf{b}_i$$

$$\text{▶ } \mathbf{b}_i^j := (1-u) \cdot \mathbf{b}_i^{j-1} + u \cdot \mathbf{b}_{i+1}^{j-1}, \quad i = 0, \dots, n-j$$

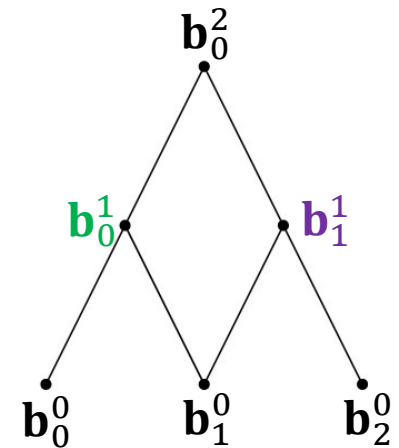
▶ Beispiel (Ausmultiplizieren):

$$\text{▶ } F(u) = \mathbf{b}_0^2 = (1-u) \cdot \mathbf{b}_0^1 + u \cdot \mathbf{b}_1^1$$

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

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

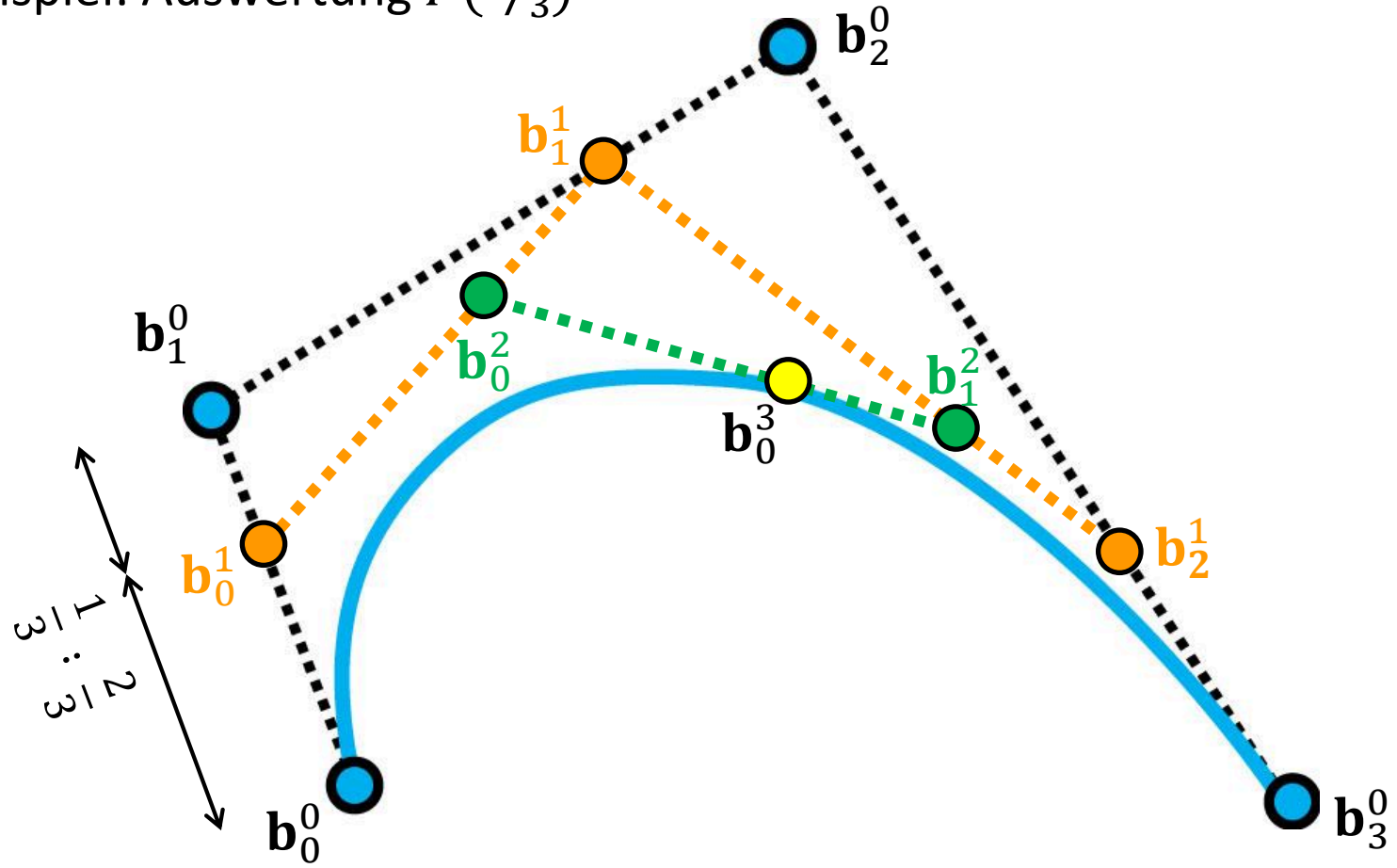
← „Kontrolle“



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 $F(2/3)$



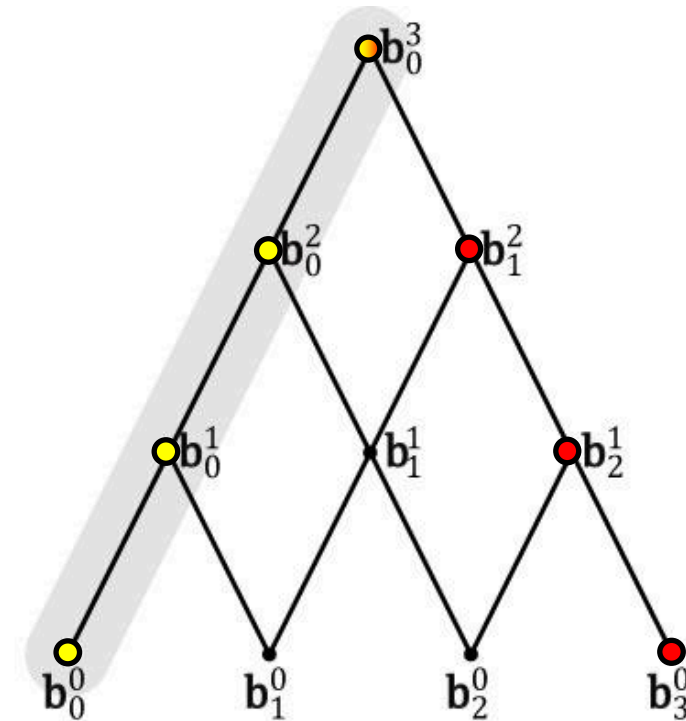
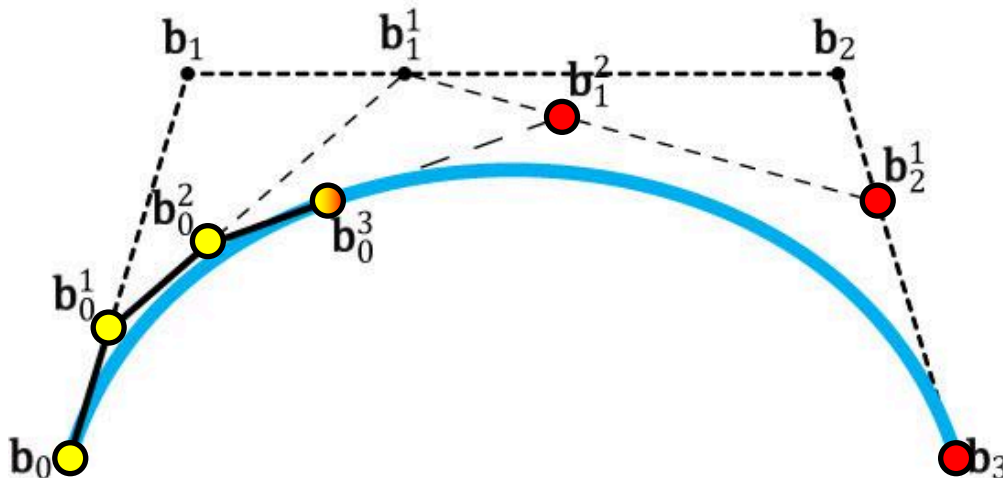
Unterteilung von Bézier-Kurven

▶ Bézier-Kurve $F(u) = \sum_{i=0}^n B_i^n(u) \mathbf{b}_i$

▶ $\mathbf{b}_i^j(q)$ sind die Punkte des de Casteljau-Algorithmus bei q

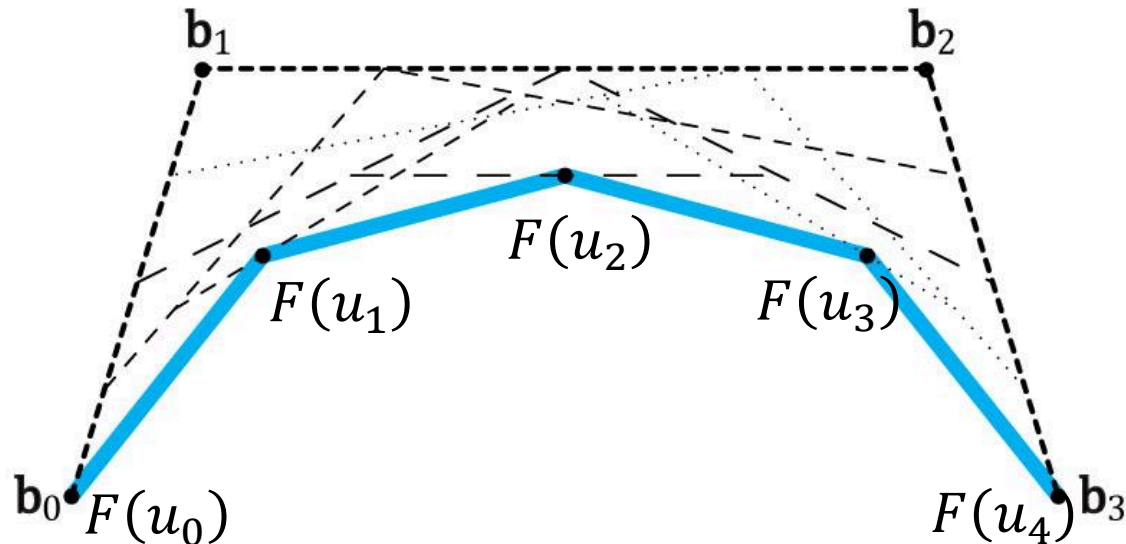
▶ Beispiel: Unterteilung einer kubischen Bézier-Kurve in zwei Kurven
($\mathbf{b}_0, \mathbf{b}_0^1, \mathbf{b}_0^2, \mathbf{b}_0^3$) und ($\mathbf{b}_0^3, \mathbf{b}_1^2, \mathbf{b}_2^1, \mathbf{b}_3$)

▶ sukzessive Unterteilung: Kontrollpolygone konvergieren gegen die Kurve selbst



Algorithmus 1

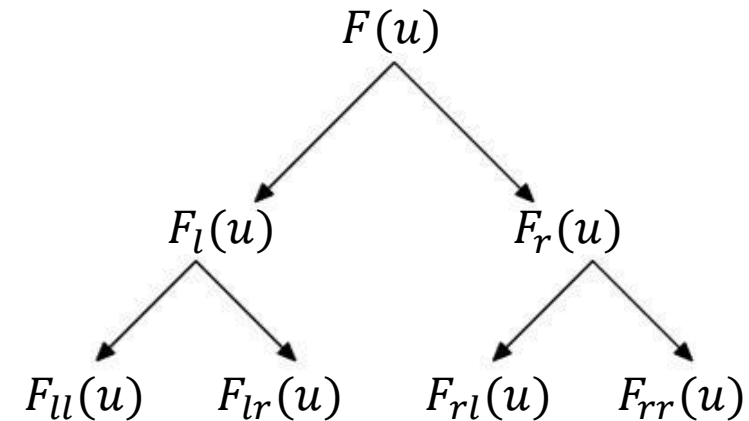
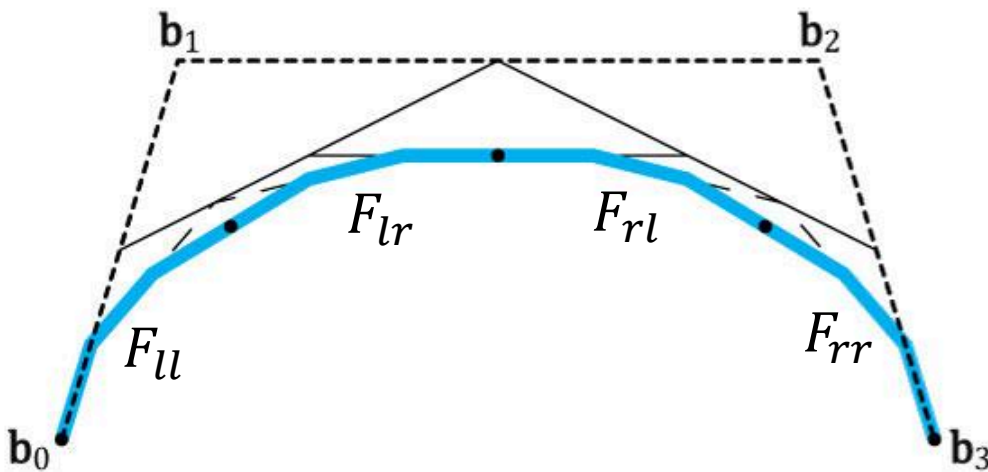
- ▶ werte $F(u)$ an m Stellen u_j mit $u_j < u_{j+1}$ aus
(mittels de Casteljau, besser nicht Auswerten der Bernstein-Polynome)
- ▶ zeichne den Polygonzug $F(u_0), \dots, F(u_{m-1})$
- ▶ Beispiel: $n = 3, u_0 = 0, u_1 = \frac{1}{4}, u_2 = \frac{1}{2}, u_3 = \frac{3}{4}, u_4 = 1$
 - ▶ drei Auswertungen, da $F(u_0)$ und $F(u_4)$ bekannt sind



Zeichnen einer Bézier-Kurve

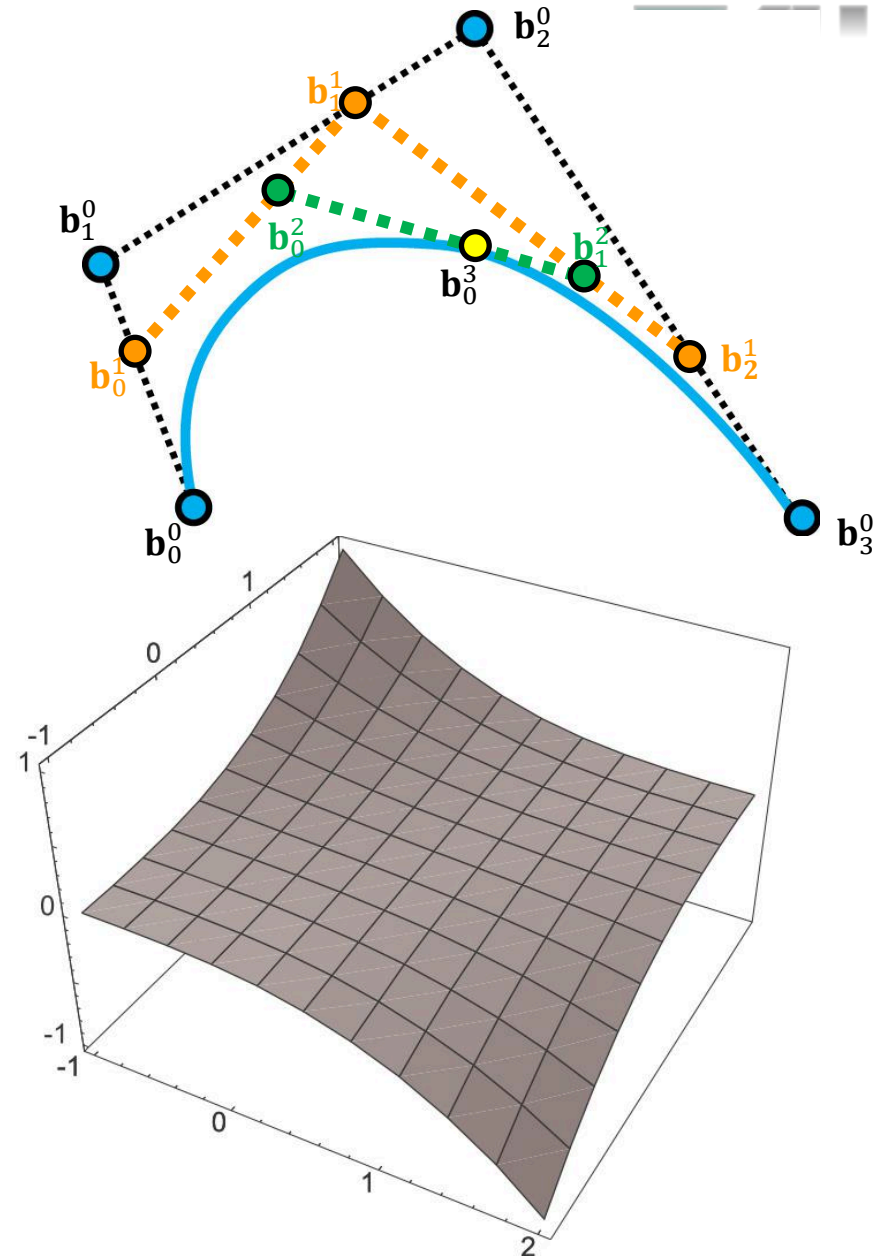
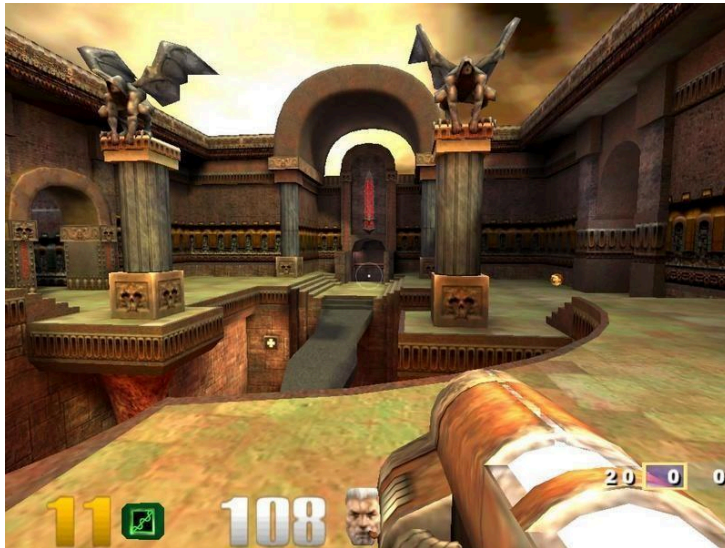
Algorithmus 2

- ▶ unterteile $F(u)$ in der parametrischen Mitte $u = \frac{1}{2}$ mittels des de Casteljau-Algorithmus in $F_l(u)$ and $F_r(u)$
- ▶ 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 deutlich besseres Ergebnis!

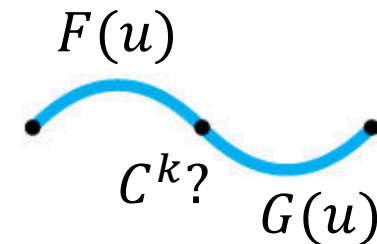
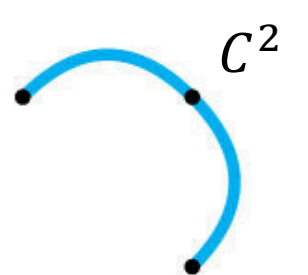
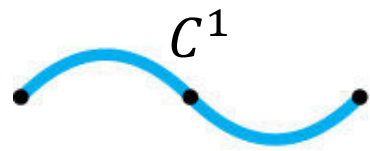
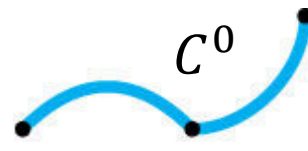
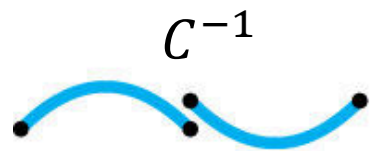


Kurven und Flächen – Inhalt

- ▶ Polynomkurven
- ▶ Bernstein-Polynome
- ▶ Basiswechsel
- ▶ Eigenschaften von Bézier-Kurven
- ▶ de Casteljau-Algorithmus
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



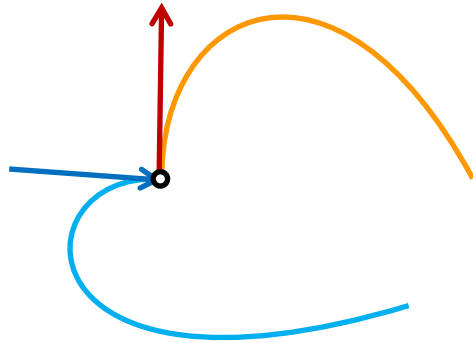
- ▶ Modellieren von komplexeren Formen mit Bézier-Kurven
 - ▶ verwende eine Bézier-Kurve von hohem Grad (u.U. numerische Probleme, aber noch wichtiger: jeder Kontrollpunkt beeinflusst die ganze Kurve → schwierig bei der Modellierung), oder
 - ▶ füge mehrere Bézier-Kurven niedrigen Grades stückweise aneinander
- ▶ **Bézier-Spline**: stückweise polynomielle Kurve, deren einzelne Abschnitte durch Bézier-Kurven beschrieben sind
 - ▶ (parametrische) Stetigkeit des Überganges? Glattheit der Kurve?



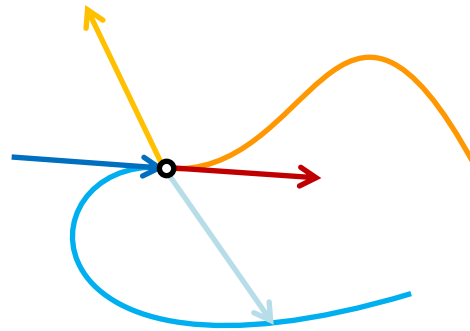
- ▶ Modellieren von komplexeren Formen mit Bézier-Kurven
 - ▶ verwende eine Bézier-Kurve von hohem Grad (u.U. numerische Probleme, aber noch wichtiger: jeder Kontrollpunkt beeinflusst die ganze Kurve → schwierig bei der Modellierung), oder
 - ▶ füge mehrere Bézier-Kurven niedrigen Grades stückweise aneinander
- ▶ **Bézier-Spline**: stückweise polynomielle Kurve, deren einzelne Abschnitte durch Bézier-Kurven beschrieben sind
 - ▶ (parametrische) Stetigkeit des Überganges? Glattheit der Kurve?
 - ▶ die n -ten Ableitungen stimmen überein in ...
 - ▶ ... Richtung bei G^n -Stetigkeit
 - ▶ ... Richtung und Länge bei C^n -Stetigkeit
 - ▶ geometrische Stetigkeit G^n kann je nach Anwendungsfall ausreichen

Bézier-Splines

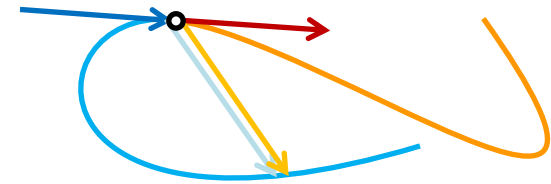
Parametrische und geometrische Stetigkeit



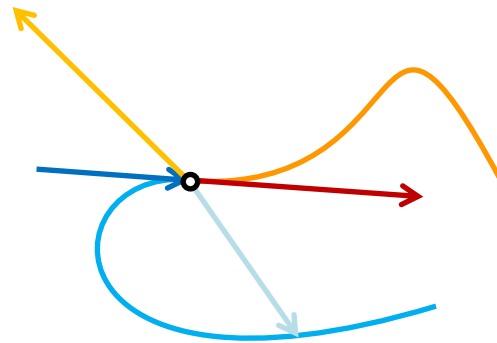
C^0 -stetig



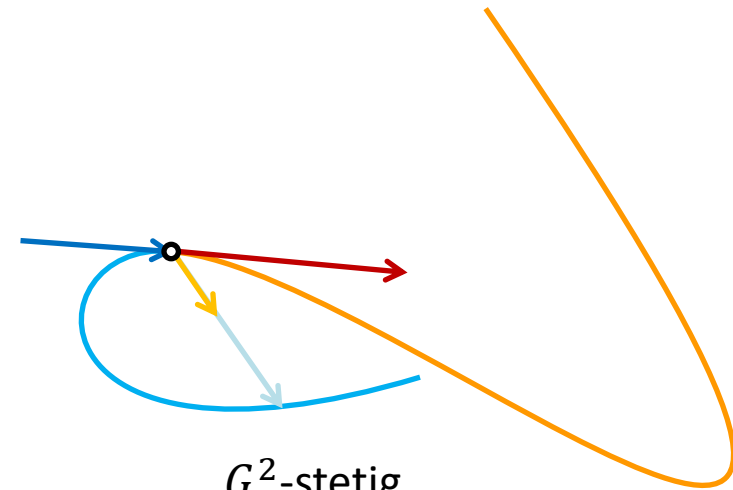
C^1 -stetig



C^2 -stetig



G^1 -stetig

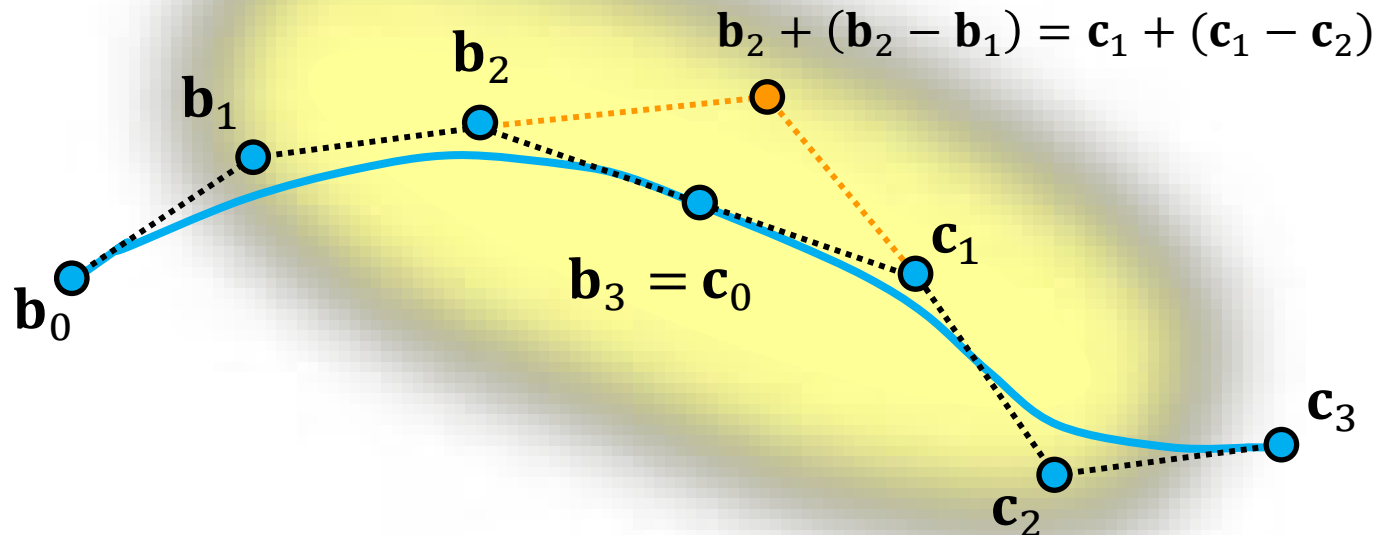


G^2 -stetig

C^k -Übergang zweier Bézier-Kurven (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 Bézier-Kurven und „A-Frame“-Eigenschaft



Ableitungen (zur Erinnerung)

▶ 1. Ableitung

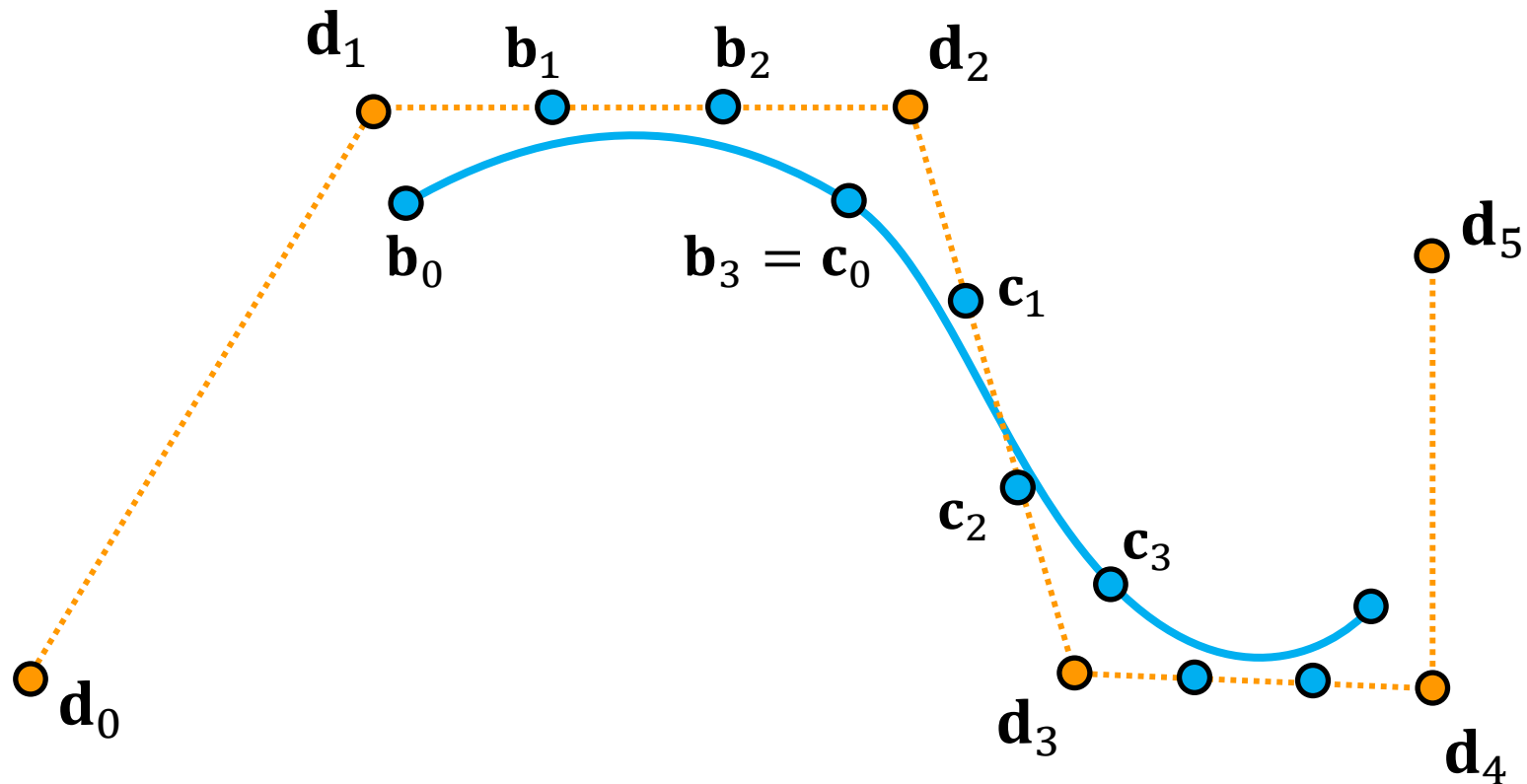
$$F'(u) = n \sum_{i=0}^{n-1} B_i^{n-1}(u) (\mathbf{b}_{i+1} - \mathbf{b}_i)$$

▶ 2. Ableitung

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

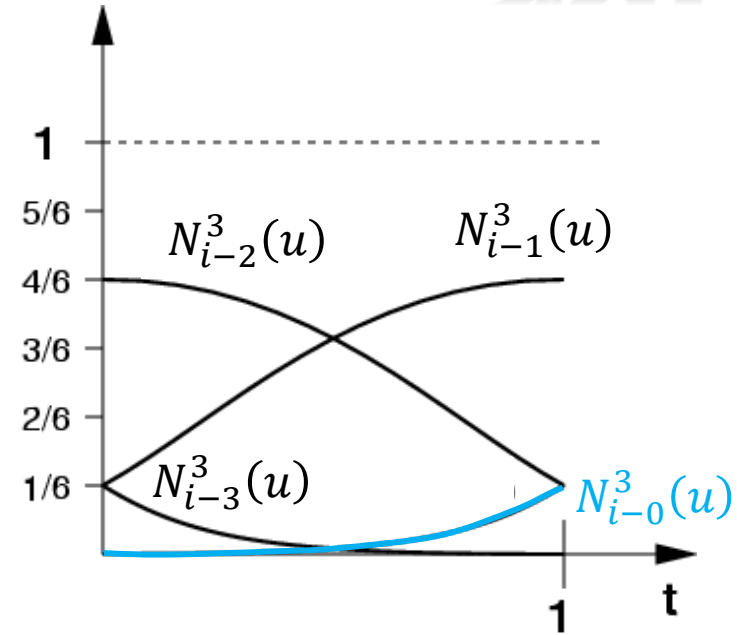
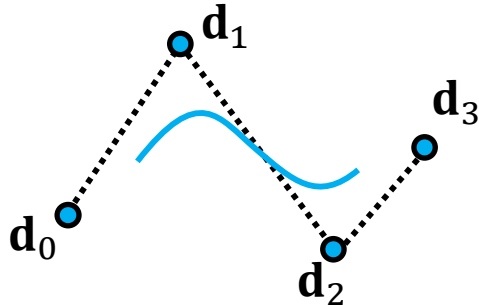
B-Splines definiert durch A-Frames

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



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



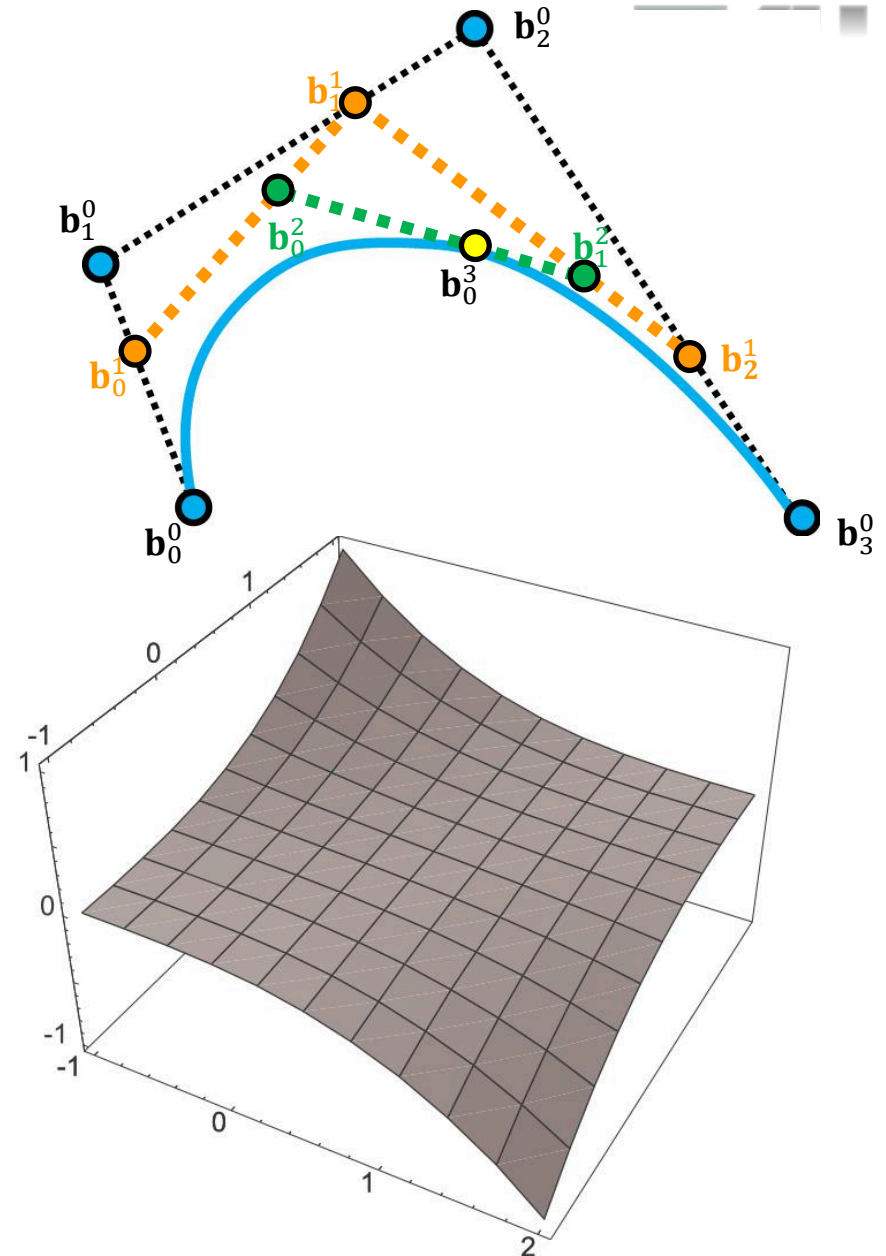
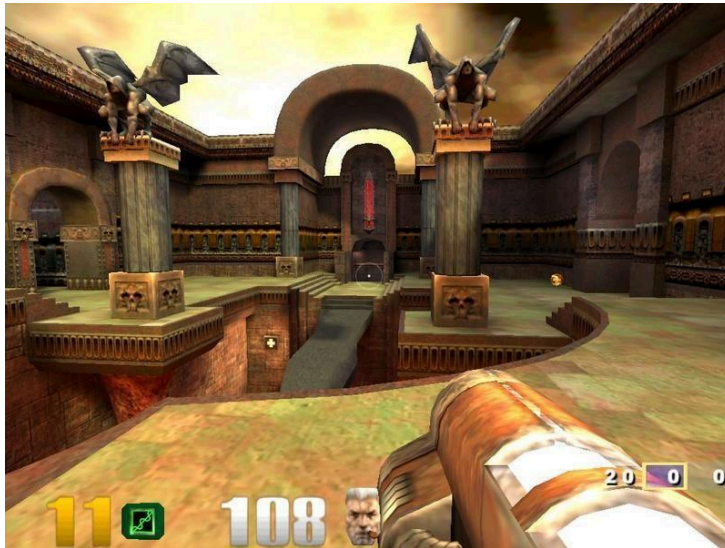
- ▶ kubisches Kurvensegment

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

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

Kurven und Flächen – Inhalt

- ▶ Polynomkurven
- ▶ Bernstein-Polynome
- ▶ Basiswechsel
- ▶ Eigenschaften von Bézier-Kurven
- ▶ de Casteljau-Algorithmus
- ▶ Bézier-Splines, B-Splines
- ▶ Tensorproduktflächen



Tensorprodukt-Bézier-Flächen

- ▶ gegeben zwei Basen zur Darstellung von Kurven in \mathbb{R}^d , z.B. Bézier-Kurven

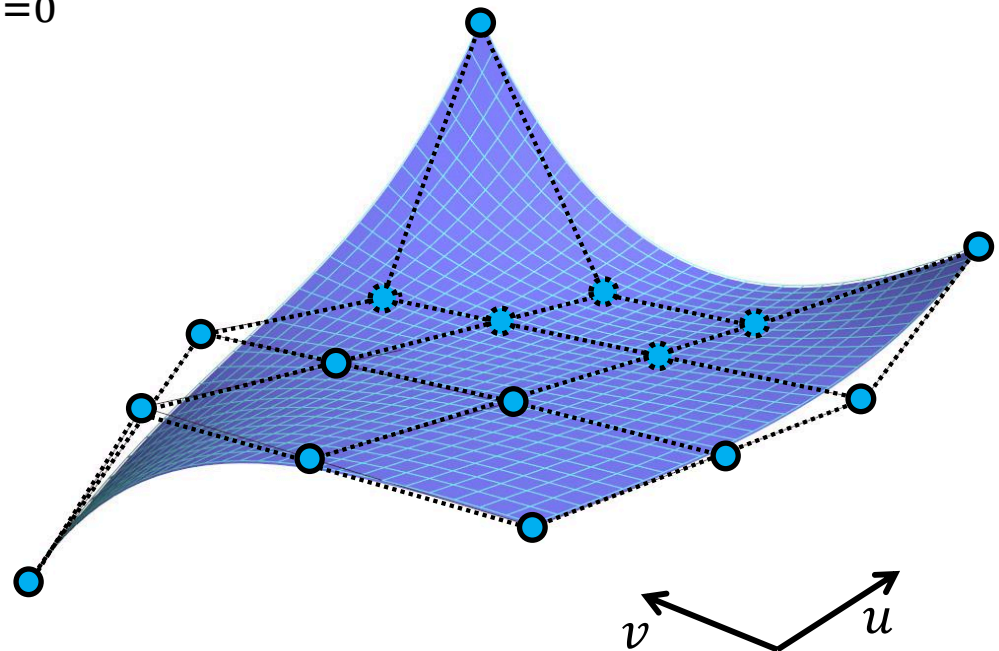
$$F(u) = \sum_{i=0}^n B_i^n(u) \mathbf{c}_i \quad \text{und} \quad G(v) = \sum_{j=0}^m B_j^m(v) \mathbf{d}_j$$

- ▶ das Tensorprodukt (= alle geordneten Paare der Basisfunktionen) ist die Basis für die TP-Bézier-Fläche:

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

- ▶ die Kontrollpunkte $\mathbf{b}_{i,j}$ bilden das **Kontrollnetz**

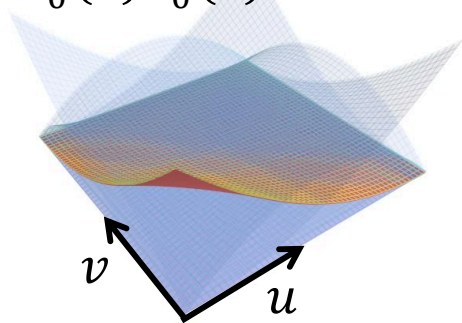
- ▶ Anm. es sind Kombinationen beliebiger Kurvenschemata und Grade möglich



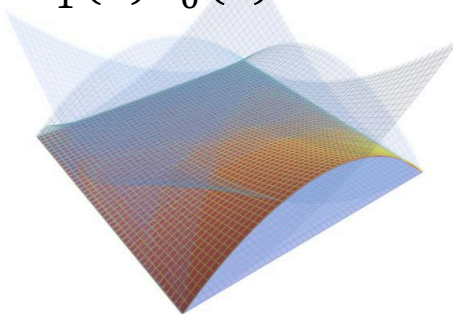
Tensorprodukt-Bézier-Flächen

► Basisfunktionen $B_i^2(u)B_j^2(v)$

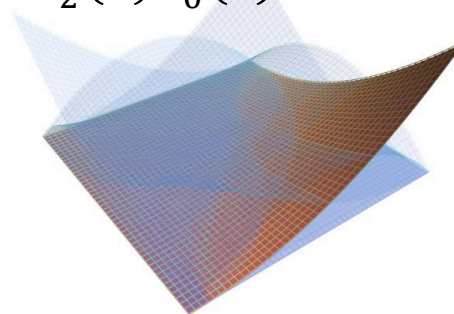
$$B_0^2(u)B_0^2(v)$$



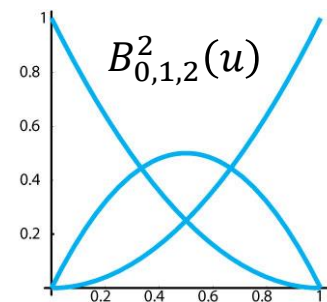
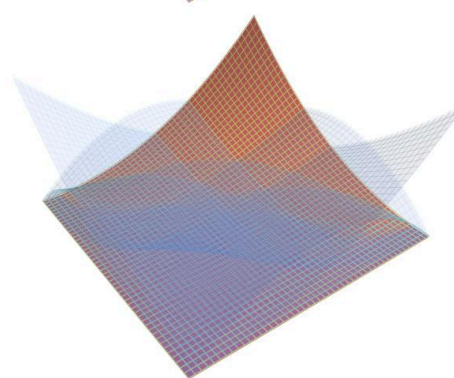
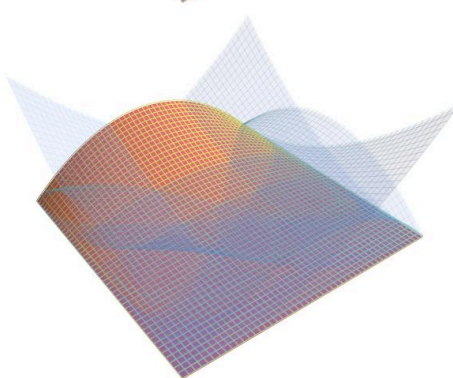
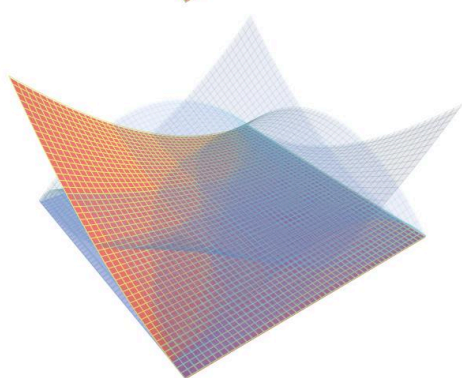
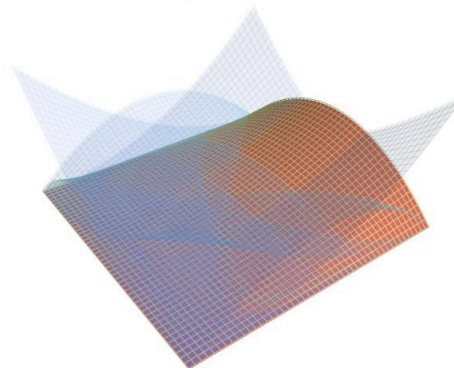
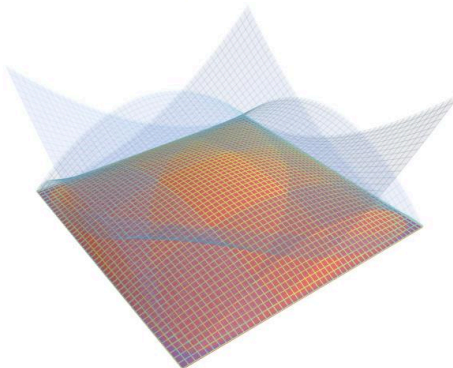
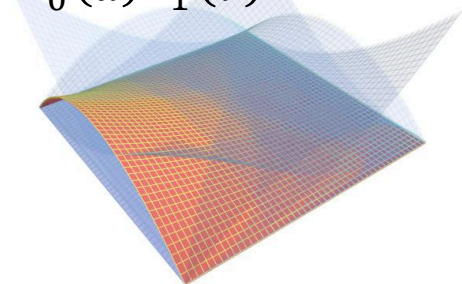
$$B_1^2(u)B_0^2(v)$$



$$B_2^2(u)B_0^2(v)$$



$$B_0^2(u)B_1^2(v)$$



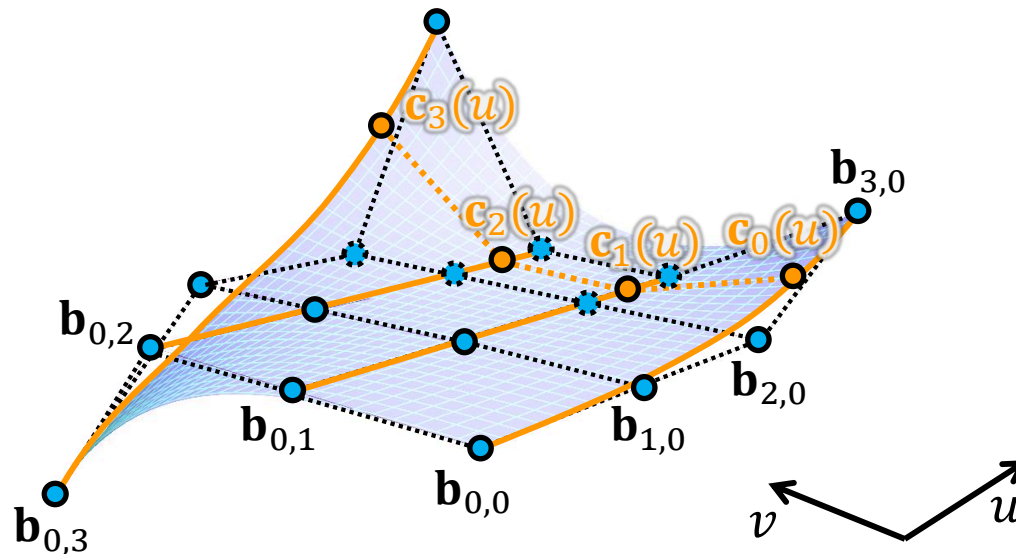
Tensorprodukt-Fläche ist eine „Kurve von Kurven“

► Ausklammern

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

► u festhalten ergibt Kontrollpunkte der v -Kurve mit $\mathbf{c}_j(u) = \sum_{i=0}^n B_i^n(u) \mathbf{b}_{i,j}$

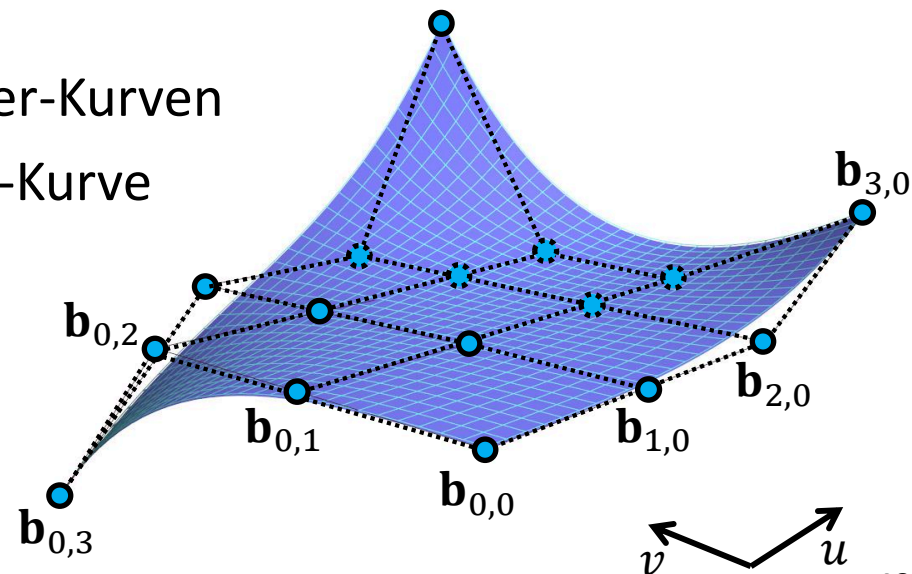
► Beispiel: $u = 3/4$ ergibt 4 Kontrollpunkte \mathbf{c}_j einer Bézier-Kurve in v



Tensorprodukt-Bézier-Flächen

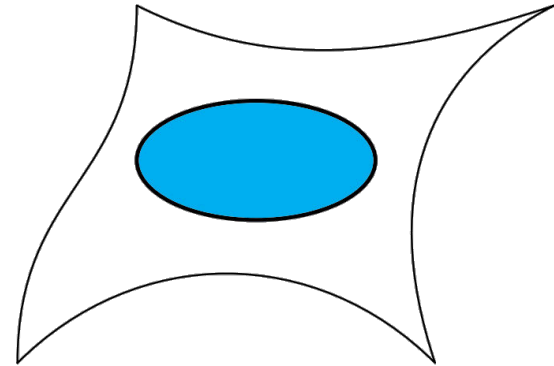
Formeigenschaften von TP-Bézier-Flächen

- ▶ konvexe Hülle-Eigenschaft
- ▶ Interpolation der 4 Ecken des Kontrollnetzes $\{\mathbf{b}_{i,j}\}$
 - ▶ $S(0,0) = \mathbf{b}_{0,0}, S(1,0) = \mathbf{b}_{n,0}, S(0,1) = \mathbf{b}_{0,m}, S(1,1) = \mathbf{b}_{n,m}$
- ▶ Fläche tangential in den Eckpunkten zu aufgespannten Ebenen der $\{\mathbf{b}_{i,j}\}$
 - ▶ 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 Bézier-Kurven
 - ▶ z.B. $S(0,v) = F(v)$ ist eine Bézier-Kurve
- ▶ affine Invarianz
- ▶ Variationsreduzierung **gilt nicht!**

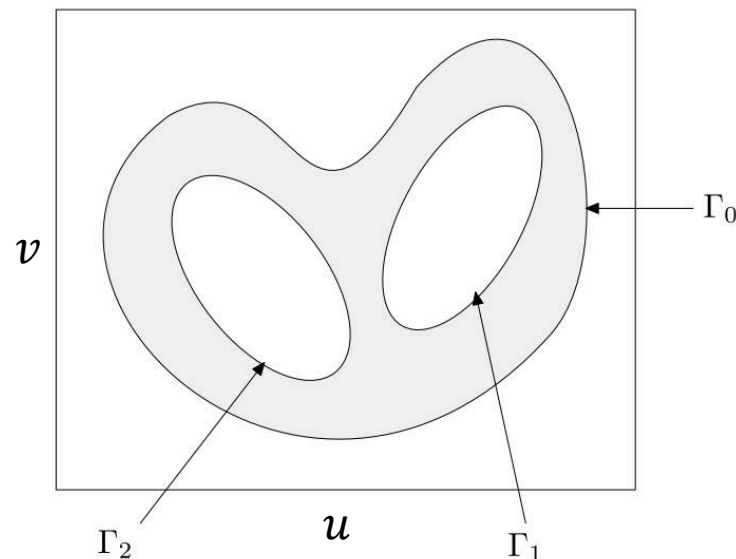


Trimming von Flächen

- ▶ Trimming („Zuschneiden“) von Flächen:
 - ▶ eine Möglichkeit: verwende Textur mit Alpha-Kanal



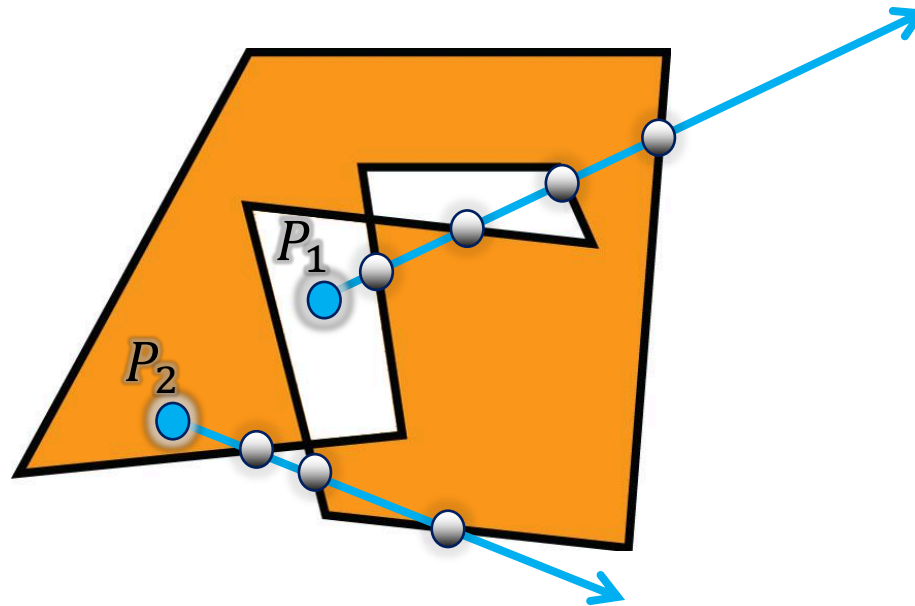
- ▶ bestimme eine Menge von Konturen Γ_i im Parametergebiet die festlegen, ob ein Punkt $S(u, v)$ zur Fläche gehört
 - ▶ Test, ob ein Punkt (u, v) innerhalb liegt mit Odd-Even-Test



Einschub: Punkt-in-Polygon/Polyeder Test

Odd-Even Test (für Polygone/Trim-Kurven in 2D, Oberflächen in 3D)

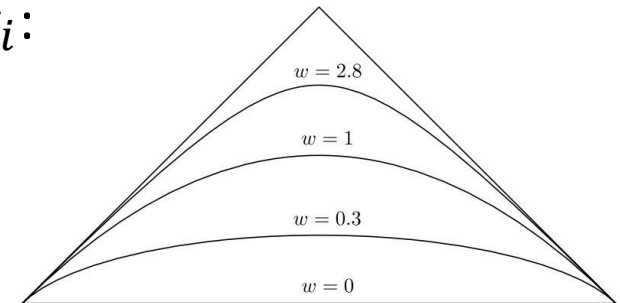
- ▶ Test für Punkt $P = (x_p, y_p)$
 - ▶ es genügt auch die Schnitte entlang einer Halbgerade von P in eine beliebige Richtung zu zählen
 - ▶ P liegt im Polygon \Leftrightarrow **ungerade** Anzahl
 - ▶ P liegt außerhalb \Leftrightarrow **gerade** Anzahl



- ▶ mit Polynomkurven ist es nicht möglich Kegelschnitte (z.B. einen Kreisbogen) exakt darzustellen, hierzu verwendet man:
 - ▶ rationale Bézier-Kurven und rationale B-Spline-Kurven (NURBS)

- ▶ Bsp. rationale Bézier-Kurve mit Gewichten w_i :

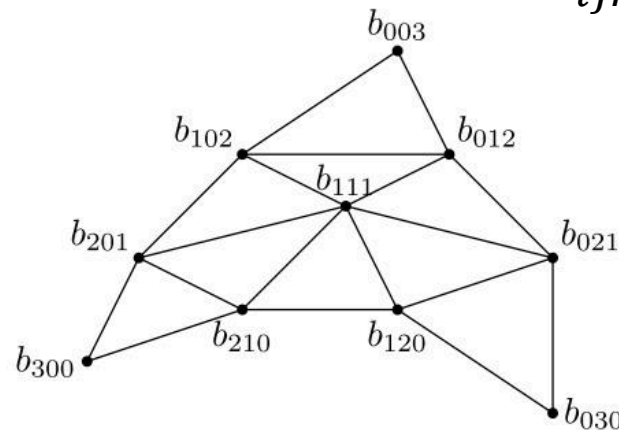
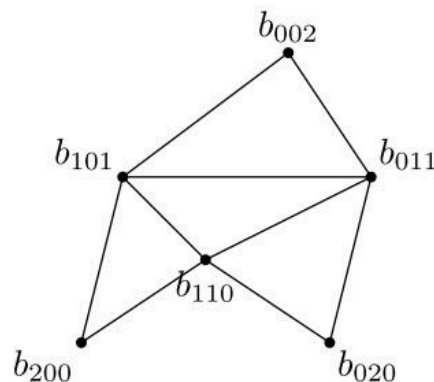
$$F(u) = \frac{\sum_{i=0}^n B_i^n(u) w_i \mathbf{b}_i}{\sum_{i=0}^n B_i^n(u) w_i}$$



- ▶ Abb. rechts: rationale Bézier-Kurven, $n = 2$

- ▶ Dreiecks-Bézierflächen: mehr topologische Flexibilität

- ▶ Bernstein-Polynome in baryzentrischen Koordinaten $B_{ijk}^n(u, v, w)$



- ▶ Flächen lassen sich in OpenGL-Shadern tessellieren und auswerten

Unterteilungsflächen (Subdivision Surfaces)

