

Computergraphik II

Winter 2012/2013

3 Freiformkurven und -flächen

Versionsdatum: 7. Oktober 2012



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-1-

Computergraphik II

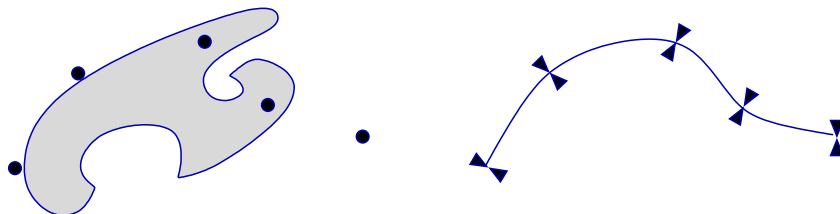
3 Freiformkurven und -flächen ...



Bemerkung: Historie

Ursprung der Geometrischen Modellierung: Industrielle Fertigung i.w. Automobilbau

vor 1960: Verschiedene Zeichenmethoden wie z.B. Kurven- oder Stahllineal.



- Ungenauigkeit bei Übertragung auf reale Formen für Guß, Stanzen etc.
- Flächen nur über Kurvenscharen handhabbar

nach 1960: Übertragung zeichentechnischer Ansätze auf Computer (NC-Steuerung)

- Modellierung gekrümmte Objekte (Kurven aber auch Flächen)
- intuitiven Kontrollparametern mit hoher Flexibilität



effizient auf Rechnern handhabbar (—→ **Polynome!**)

Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-2-

Computergraphik II

3.1 Bézier-Kurven



Ziel: Kontrolle von Polynomen

Beliebige Formen: **Funktionsgraphen** genügen nicht!

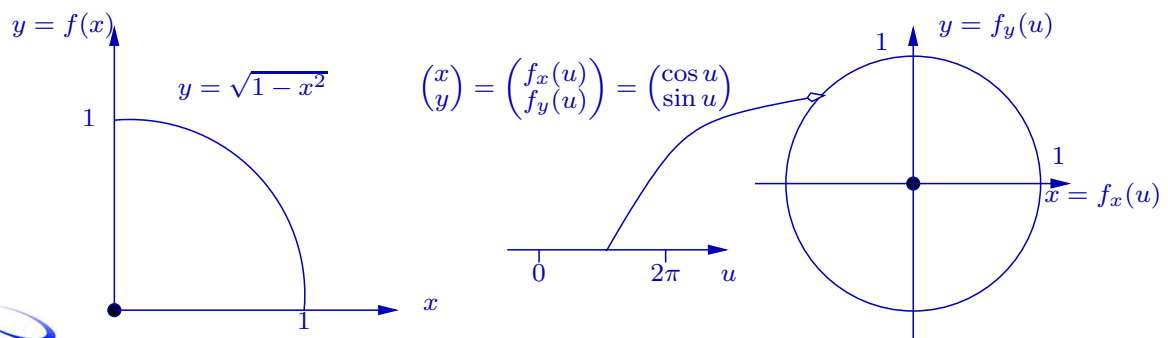
Intuitive Form-Kontrolle: **Monombasis** $1, u, u^2, \dots$ geht nicht!

Funktionsgraph vs. parametrische Kurve

Graph: Abtragen des abhängigen Parameters $y = f(x)$ über dem freien Parameter x

Param. Kurve: Darstellung mehrerer abhängiger Parameter

$x = f_x(u), y = f_y(u), \dots$ ohne Raumbezug zum freien Parameter u



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-3-

Computergraphik II

3.1 Bézier-Kurven ...

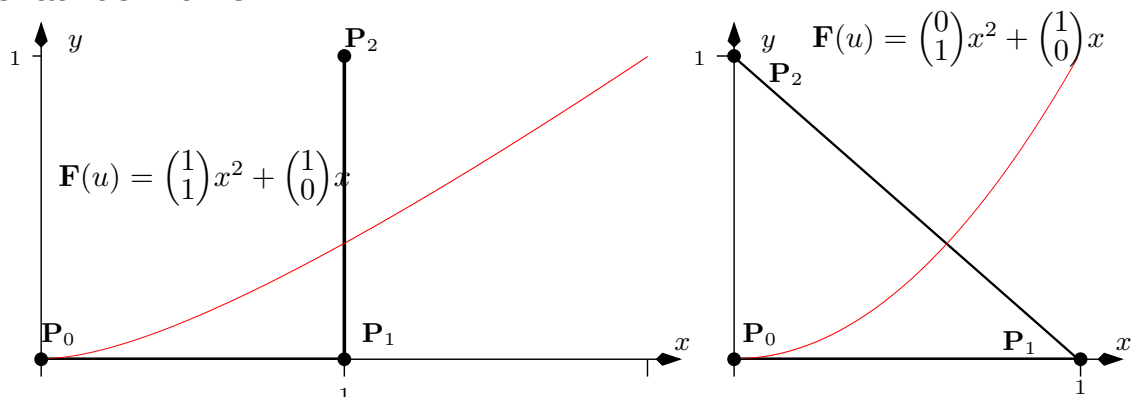


Kontrolle von Polynomen mit Monombasis

Direkte Nutzung der Monombasis für 2D parametrische Kurven vom Grad n

$$\mathbf{F}(u) = \begin{pmatrix} f_x(u) \\ f_y(u) \end{pmatrix} = \sum_{i=0}^n \mathbf{P}_i u^i, \quad \text{mit } \mathbf{P}_i \in \mathbb{R}^2, \quad i = 0, \dots, n, \quad u \in [0, 1]$$

Aber kein intuitiver Zusammenhang zwischen „Parameterpunkten“ $\mathbf{P}_i$ und Verlauf der Kurve:



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-4-

Computergraphik II

3.1 Bézier-Kurven ...

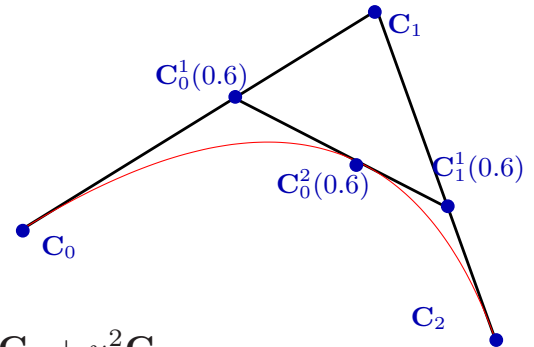


Beispiel: Erzeugung von parametrischen Parabel-Kurven

Gegeben: Drei Punkte: $C_0, C_1, C_2 \in \mathbb{R}^2$

Konstruktion: Sei $u \in [0, 1]$, dann bilde

$$\begin{aligned} C_0^1(u) &= (1-u)C_0 + uC_1 \\ C_1^1(u) &= (1-u)C_1 + uC_2 \\ C(u) = C_0^2(u) &= (1-u)C_0^1(u) + uC_1^1(u) \\ &= (1-u)^2C_0 + 2(1-u)uC_1 + u^2C_2 \end{aligned}$$



Eigenschaften dieser Konstruktion:

1. $C(u)$ ist **affine Kombination** der C_i , da $(1-u)^2 + 2(1-u)u + u^2 = ((1-u) + u)^2 = 1$
2. **Endpunktinterpolation:** $C_0^2(0) = C_0$, $C_0^2(1) = C_2$
3. $\{C(u) : u \in [0, 1]\} \subset \Delta(C_0, C_1, C_2)$, also Kurve verläuft in **konvexer Hülle** der Kontrollpunkte, da Gewichte ≥ 0



3.1 Bézier-Kurven ...



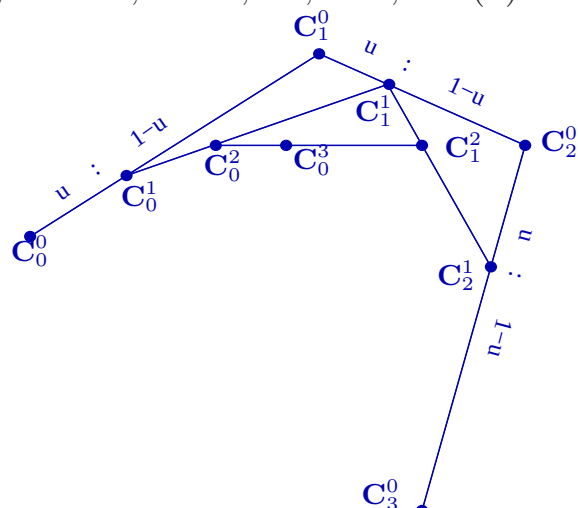
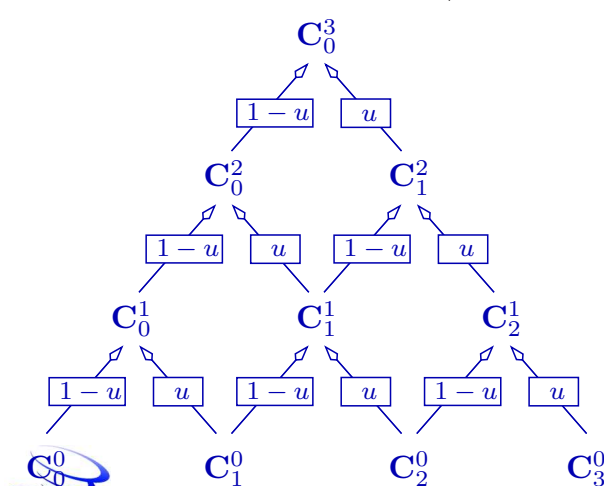
Algorithmus: de Casteljau

Gegeben: Kontrollpunkte C_i , $i = 0, \dots, n$ für ein $n \in \mathbb{N}$, sowie der Parameter $u \in [0, 1]$

Initialisierung: $C_i^0 = C_i$, $i = 0, \dots, n$

Rekursion: Affin-Kombinationen benachbarter Kontrollpunkte:

$$C_i^{k+1} = (1-u)C_i^k + uC_{i+1}^k, \quad i = 0, \dots, n-k-1, \quad k = 0, \dots, n-1, \quad C(u) = C_0^n$$



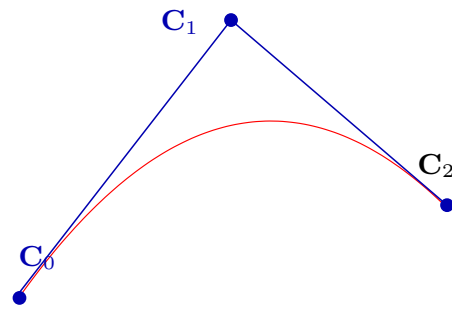
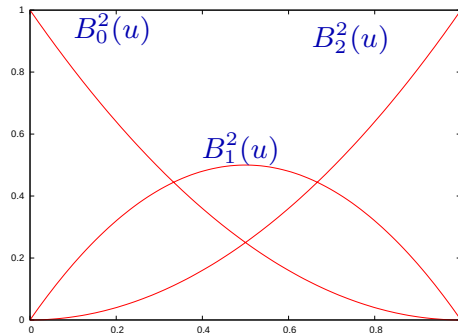
3.1 Bézier-Kurven ...



Bezeichnung: Bézier-Kurven ($n = 2$)

Parabel: $C(u) = \underbrace{(1-u)^2}_{=B_0^2(u)} C_0 + \underbrace{2(1-u)u}_{=B_1^2(u)} C_1 + \underbrace{u^2}_{=B_2^2(u)} C_2$

Bernstein-Bézier-Polynome: Gewichte $B_i^2(u)$ der Affin-Kombination der C_i



Beachte: $B_i^2(u)$ entsprechen variablen Gewichten einer Affin-Kombination



3.1 Bézier-Kurven ...



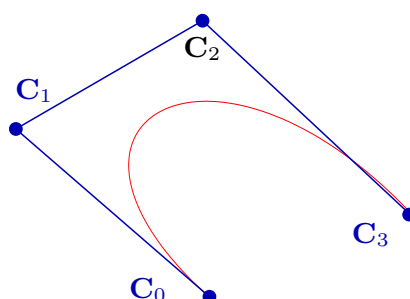
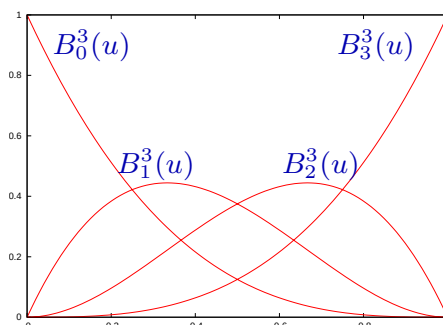
Bezeichnung: Bézier-Kurven (allgemeiner Fall)

Bernstein-Bézier-Polynome: Für beliebiges $n \in \mathbb{N}$ wird definiert:

$$C(u) = \sum_{i=0}^n C_i B_i^n(u), \text{ mit } B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i} \text{ und } \binom{n}{i} = \frac{n!}{i!(n-i)!}$$

Bézier-Kurve: Für Kontrollpunkte $C_0, \dots, C_n$ ist: $C(u) = \sum_{i=0}^n B_i^n(u) C_i$

Kubischer Fall $n = 3$: Häufig kommen Bézier-Kurven vom Grad 3 zum Einsatz:



3.1 Bézier-Kurven ...



Eigenschaften der Bernstein-Bézier-Polynome

Polynom-Basis: Polynome $P \in \mathcal{P}^n$ vom Grad n können als Bézier-Kurve dargestellt werden:

$$\forall P(u) = \sum_{i=0}^n A_i u^i : \exists_1 C_i, i = 0, \dots, n : P(u) = \sum_{i=0}^n C_i B_i^n(u)$$

Positivität: Für $u \in [0, 1]$ gilt stets: $B_i^n(u) \geq 0$

Endpunktverhalten: $B_i^n(0) = \begin{cases} 1 & \text{falls } i = 0 \\ 0 & \text{sonst} \end{cases} \quad B_i^n(1) = \begin{cases} 1 & \text{falls } i = n \\ 0 & \text{sonst} \end{cases}$

Zerlegung der 1: $1 = [u + (1 - u)]^n = \sum_{i=0}^n \binom{n}{i} u^i (1 - u)^{n-i} = \sum_{i=0}^n B_i^n(u)$

Ableitung:

$$(B_i^n)'(u) = \begin{cases} -n B_0^{n-1}(u) & \text{falls } i = 0 \\ n B_{n-1}^{n-1}(u) & \text{falls } i = n \\ n (B_{i-1}^{n-1}(u) - B_i^{n-1}(u)) & \text{falls } 0 < i < n \end{cases}$$



3.1 Bézier-Kurven ...



Eigenschaften der Bézier-Kurven

Endpunktinterpolation: $C(0) = C_0$, $C(1) = C_n$, da $B_0^n(0) = B_n^n(1) = 1$, sonst $B_i^n(0) = B_i^n(1) = 0$

Globaler Einfluss der C_i : C_i beeinflusst $C(u)$, $u \in]0, 1[$, da hier $B_i^n(u) \neq 0$, $\forall i$

Affine Invarianz: Die Kurve C ist affin invariant bzgl. seiner Kontrollpunkte:

$$T \text{ affine Abbildung} \implies T(C(u)) = \sum_{i=0}^n T(C_i) B_i^n(u) \quad \text{da} \quad \sum B_i^n \equiv 1$$

Konvexe Hülle: Aufgrund der Positivität der B_i^n und der affinen Invarianz verläuft $C(u)$, $u \in [0, 1]$ innerhalb der konvexen Hülle der Kontrollpunkte

Variation Diminishing Property (ohne Beweis): Die Bézier-Kurve hat max. so viele Wendepunkte wie ihr Kontrollpolygon:

Fall $n = 2$: $\forall$ Gerade g gilt: $\#\{g \cap \text{Kontrollpolygon}\} \geq \#\{g \cap \text{Kurve}\}$

Fall $n = 3$: $\forall$ Ebene E gilt: $\#\{E \cap \text{Kontrollpolygon}\} \geq \#\{E \cap \text{Kurve}\}$



3.1 Bézier-Kurven ...



Eigenschaften der Bézier-Kurven (Forts.)

Ableitung: Bézier-Darstellung der ersten Ableitung:

$$\begin{aligned} \mathbf{C}'(u) &= n \left(-\mathbf{C}_0 B_0^{n-1}(u) + \mathbf{C}_n B_{n-1}^{n-1}(u) + \sum_{i=1}^{n-1} \mathbf{C}_i [B_{i-1}^{n-1}(u) - B_i^{n-1}(u)] \right) \\ &= n \sum_{i=0}^{n-1} (\mathbf{C}_{i+1} - \mathbf{C}_i) B_i^{n-1}(u) \end{aligned}$$

End-Tangenten (Spezialfall): Für $u \in \{0, 1\}$ gilt:

$$\mathbf{C}'(0) = n (\mathbf{C}_1 - \mathbf{C}_0) \quad \mathbf{C}'(1) = n (\mathbf{C}_n - \mathbf{C}_{n-1})$$

Bezug zu de Casteljau: Die letzte Kante des de Casteljau-Schemas entspricht der Tangentenrichtung der Kurve:

$$\mathbf{C}'(u) = n (\mathbf{C}_1^{n-1} - \mathbf{C}_0^{n-1})$$

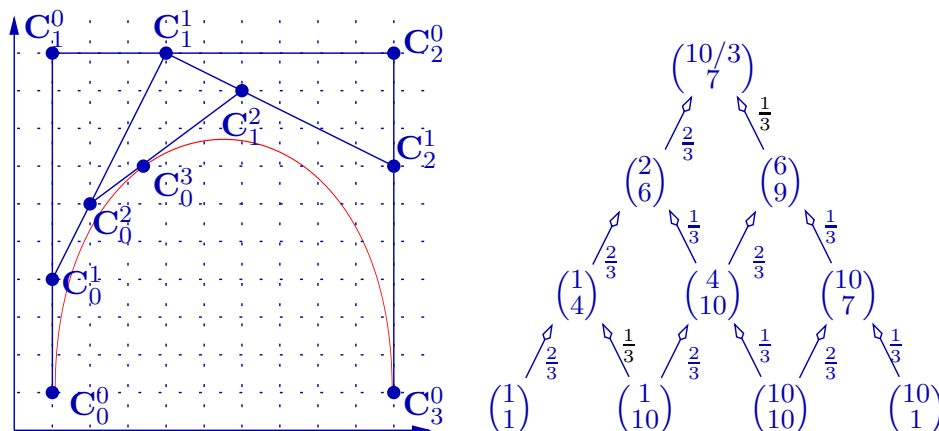


3.1 Bézier-Kurven ...



Beispiel:

Gegeben: Punkte $\mathbf{C}_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $\mathbf{C}_1 = \begin{pmatrix} 1 \\ 10 \end{pmatrix}$, $\mathbf{C}_2 = \begin{pmatrix} 10 \\ 10 \end{pmatrix}$, $\mathbf{C}_3 = \begin{pmatrix} 10 \\ 1 \end{pmatrix}$. Gesucht: $\mathbf{C}(\frac{1}{3})$



Endpunkte: $\mathbf{C}(0) = \mathbf{C}_0 = (1, 1)$, $\mathbf{C}(1) = \mathbf{C}_3 = (10, 1)$

Ableitung: $\mathbf{C}'(\frac{1}{3}) = 3 \left(\begin{pmatrix} 6 \\ 9 \end{pmatrix} - \begin{pmatrix} 2 \\ 6 \end{pmatrix} \right) = \begin{pmatrix} 12 \\ 9 \end{pmatrix}$ und



$$\mathbf{C}'(u) = 3 \sum_{i=0}^2 (\mathbf{C}_{i+1} - \mathbf{C}_i) B_i^2(u) = 3 \left(\begin{pmatrix} 0 \\ 9 \end{pmatrix} B_0^2(u) + \begin{pmatrix} 9 \\ 0 \end{pmatrix} B_1^2(u) + \begin{pmatrix} 0 \\ -9 \end{pmatrix} B_2^2(u) \right)$$



Auswertung von Polynomkurven

Bewertung de Casteljau: ○ sehr stabil (nur Interpolation)

○ Komplexität: $\mathcal{O}(n^2)$ pro Auswertung, genauer

Vektor-Additionen: $\frac{(n+1)n}{2}$ Skalare Multiplikationen: $(n+1)n$

Horner-Schema: Ausgehend von der Polynomkurve $\mathbf{F}(u) = \sum_{i=0}^n \mathbf{A}_i u^i \in \mathcal{P}^n$
($\mathcal{P}^n$: Raum der Polynome vom Grad $\leq n$)

$$\mathbf{F}(u) = \sum_{i=0}^n \mathbf{A}_i u^i = \mathbf{A}_0 + u(\mathbf{A}_1 + u(\mathbf{A}_2 + u(\mathbf{A}_3 \dots + u(\mathbf{A}_{n-1} + u\mathbf{A}_n)))) \dots)$$

○ weniger stabil

○ Komplexität: $\mathcal{O}(n)$, genauer: n skalare Multiplikationen, n Vektor-Additionen



3.1 Bézier-Kurven ...



Vorwärts-Differenzen

Ziel: Auswertung von $\mathbf{F}(u) = \sum_{i=0}^n \mathbf{A}_i u^i$ an äquidistanten Parametern

$$u_j = u_0 + j \cdot \Delta, \quad j = 0, \dots, k \text{ für ein } k \in \mathbb{N} \text{ und Schrittweite } \Delta > 0$$

Erste Vorwärtsdifferenz: Betrachte Polynom $\delta^1(u) := \mathbf{F}(u + \Delta) - \mathbf{F}(u)$; es gilt:

$$\begin{aligned} \delta^1(u) &= \mathbf{F}(u + \Delta) - \mathbf{F}(u) = \mathbf{A}_n ((u + \Delta)^n - u^n) + \mathbf{G}(u), \quad \mathbf{G} \in \mathcal{P}^{n-1} \\ &= \mathbf{A}_n \left(\left(\sum_{i=0}^n \binom{n}{i} u^i \Delta^{n-i} \right) - u^n \right) + \mathbf{G}(u) \quad \boxed{\implies \delta^1 \in \mathcal{P}^{n-1}} \end{aligned}$$

i -te Vorwärtsdifferenz: Rekursive Differenzbildung

$$\delta^i(u) := \delta^{i-1}(u + \Delta) - \delta^{i-1}(u) \in \mathcal{P}^{n-i} \quad \text{insbesondere: } \delta^n(u) \in \mathcal{P}^0 \text{ konstant}$$

$$\text{Damit ist } \delta^{i-1}(u + \Delta) = \delta^i(u) + \delta^{i-1}(u).$$



3.1 Bézier-Kurven ...



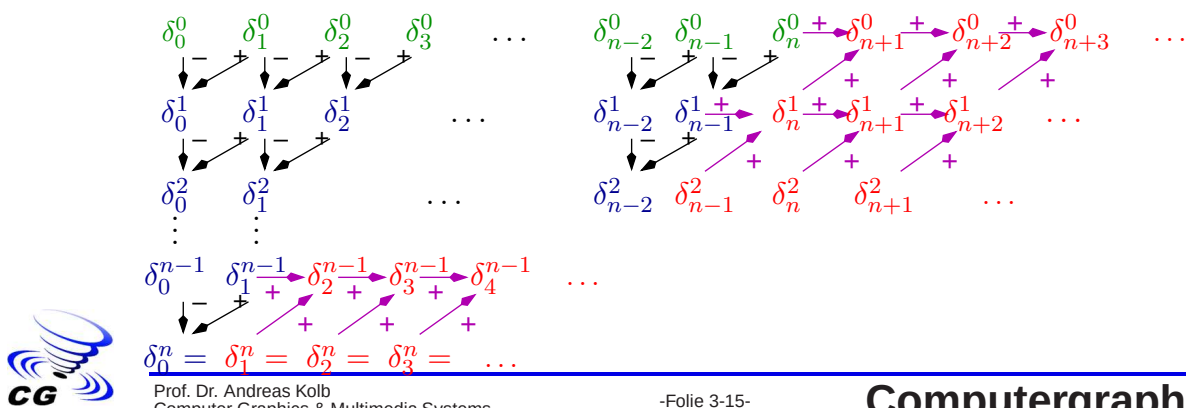
Algorithmus: Vorwärts-Differenzen

Initialisierung: Berechne ausgehend von u_0 und festem $\Delta > 0$:

1. Punkte auf der Kurve: $\delta_j^0 := \mathbf{F}(u_j) = \mathbf{F}(u_0 + j\Delta)$, $j = 0, \dots, n$
2. Differenzen
 $\delta_j^i := \delta^i(u_j) = \delta^{i-1}(u_{j+1}) - \delta^{i-1}(u_j)$, $i = 1, \dots, n$, $j = 0, \dots, n-i$

Iteration: Sukzessive Berechnung der $(m+1)$ -ten „Diagonale“ ($m \geq n$):

$$\delta_{(m-n)+1}^n = \delta_{m-n}^n = \text{const}, \quad \delta_{(m-r)+1}^r = \delta_{m-r}^r + \delta_{m-r}^{r+1}, \quad r = 0, \dots, n-1$$



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-15-

Computergraphik II

3.1 Bézier-Kurven ...



Algorithmus: Vorwärts-Differenzen (Forts.)

Komplexität $\mathcal{O}(n)$, n Vektor Add. pro Punkt nach der Initialisierung

Beispiel: Beispiel Vorwärts-Differenzen

Gegeben: $f(u) = u^3 - 4u^2 - 2u$, $u_0 = 0, \Delta = 1$

Lösung:

$u :$	0	1	2	3	4	5	6	7	8
$\delta_j^0 :$	0	-5	-12	-15	-8	15	60	133	240
$\delta_j^1 :$	-5	-7	-3	7	23	45	73	107	
$\delta_j^2 :$	-2	4	10	16	22	28	34		
$\delta_j^3 :$	6	6	6	6	6	6			



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-16-

Computergraphik II

3.1 Bézier-Kurven ...



Umwandlung Bézier nach Polynome

Erinnerung: Jedes Polynom $\in \mathcal{P}^n$ lässt sich in der Bernstein-Bézier-Basis darstellen:

$$\text{span}\{1, u, u^2, \dots, u^n\} = \text{span}\{B_0^n(u), \dots, B_n^n(u)\}$$

Basis-Transformation: Die Umwandlung der Darstellung erfolgt über eine $(n+1) \times (n+1)$ -Matrix.

Beispiel $n = 3$: $C(u) = \sum_{i=0}^3 C_i B_i^3(u) = \sum_{i=0}^3 A_i u^i$

$$\begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} \quad \text{da} \quad \begin{array}{ll} B_0^3(u) = (1-u)^3 = -u^3 + 3u^2 - 3u + 1 \\ B_1^3(u) = 3(1-u)^2 u = 3u^3 - 6u^2 + 3u \\ B_2^3(u) = 3(1-u)u^2 = -3u^3 + 3u^2 \\ B_3^3(u) = u^3 = u^3 \end{array}$$



3.1 Bézier-Kurven ...



Rationale Bézier-Kurven

Bislang: Nur Polynomkurven darstellbar

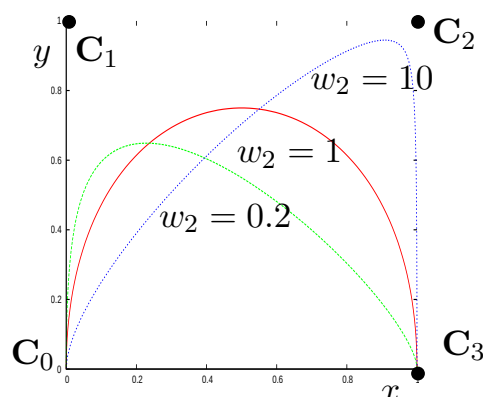
Ziel: Bessere Kurvenkontrolle durch Einführung von **Gewichten** w_i für $B_i^n(u)$

Wichtig: Affine Invarianz muss erhalten bleiben, also muss normiert werden

$$C(u) = \frac{\sum_{i=0}^n C_i \cdot w_i B_i^n(u)}{\sum_{i=0}^n w_i B_i^n(u)} = \sum_{i=0}^n C_i \tilde{B}_i^n(u) \quad \text{mit} \quad \tilde{B}_i^n(u) = \frac{w_i B_i^n(u)}{\sum_{i=0}^n w_i B_i^n(u)}$$

ist $\sum \tilde{B}_i^n(u) = 1$

Beispiel: Veränderung des Gewichtes bei C_2



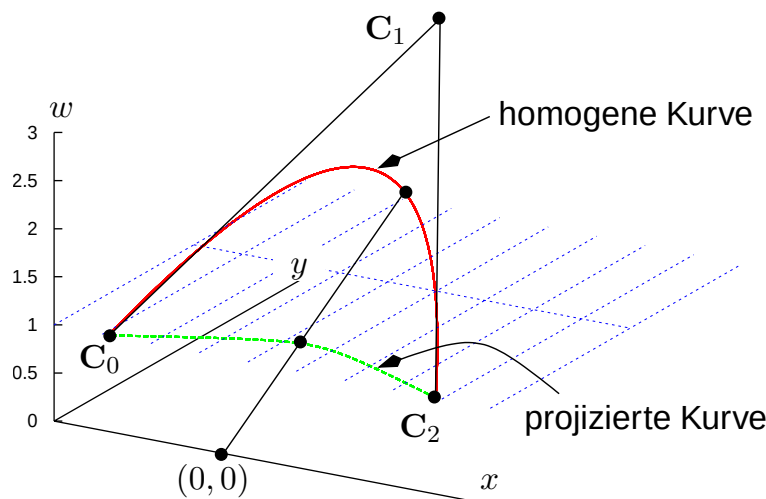
3.1 Bézier-Kurven ...



Rationale Bézier-Kurven als homogene Kurven

Alternativer Ansatz: Zeichne Kurve in *Homogenen Koordinaten* und projiziere Kurve auf $w = 1$:

$$C_i = \begin{bmatrix} w_i x_i \\ w_i y_i \\ w_i z_i \\ w_i \end{bmatrix} \quad C(u) = \frac{\sum_{i=0}^n w_i \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} B_i^n(u)}{\sum_{i=0}^n w_i B_i^n(u)}$$



3.1 Bézier-Kurven ...



Einfluss der Gewichte

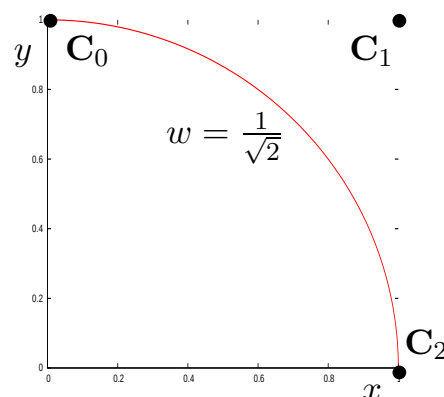
Allgemein: Je größer w_i (bei konstanten $w_j, j \neq i$), desto näher verläuft $C(u)$ bei C_i

Spezialfall Kegelschnitte für $n = 2$ und $w_0 = w_2 = 1$ ergibt sich durch Variation von w_1 :

$$C \text{ ist Segment einer } \begin{cases} \text{Ellipse} \\ \text{Parabel} \\ \text{Hyperbel} \end{cases} \Leftrightarrow \begin{cases} w_1 < 1 \\ w_1 = 1 \\ w_1 > 1 \end{cases}$$

Spezialfall Kreissegment für $n = 2$:

$$w_0 = w_2 = 1 \text{ und } w_1 = 1/\sqrt{2}$$

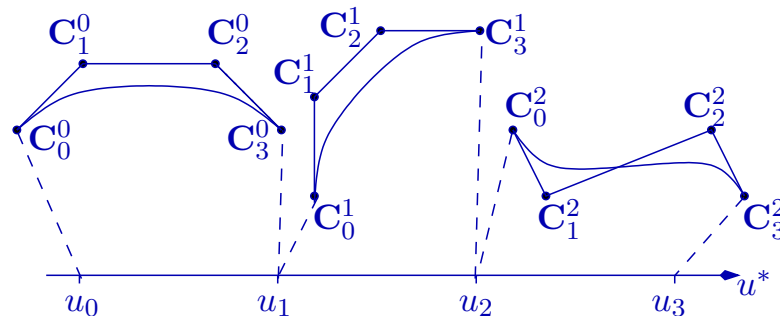


3.1 Bézier-Kurven ...



Allgemeines Parameter-Intervall

Beliebige Parameter-Intervalle $[a, b]$ statt $[0, 1]$; Bsp. stückweise Kurven



Umparametrisierung: Lineare Transformation $\phi : I = [a, b] \rightarrow [0, 1]$ ändert Polynomgrad nicht:

$$\phi(u) := \frac{u - a}{b - a} \in [0, 1], \forall u \in I \text{ und}$$

$$B_i^{I,n}(u) := B_i^n(\phi(u)) = \binom{n}{i} \left(\frac{u - a}{b - a} \right)^i \left(\frac{b - u}{b - a} \right)^{n-i}$$



Beachte: Die Geometrie der Kurve bleibt unverändert!

Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-21-

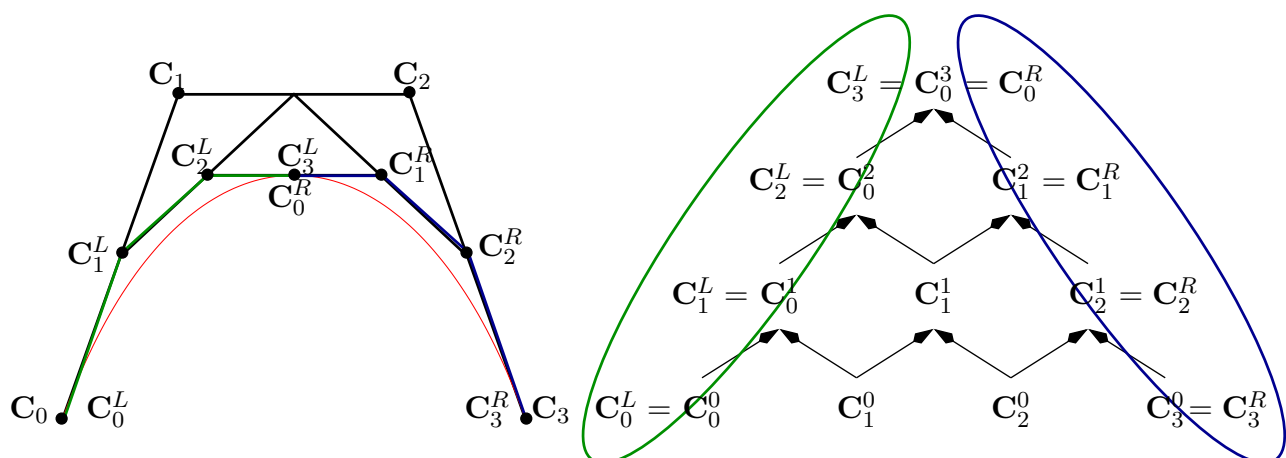
Computergraphik II

3.1 Bézier-Kurven ...



Unterteilung der Bézier-Kurve $C(u)$

Auswertung von C bei $a \in [0, 1]$ liefert Bézier-Darstellung der Teilkurven auf $[0, a]$ bzw. $[a, 1]$:



Beachte: Neue Bézier-Kurven beschreiben nur einen Teil der ursprünglichen Geometrie



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-22-

Computergraphik II



Schnitt zwischen Bézier-Kurven mit Bisektion

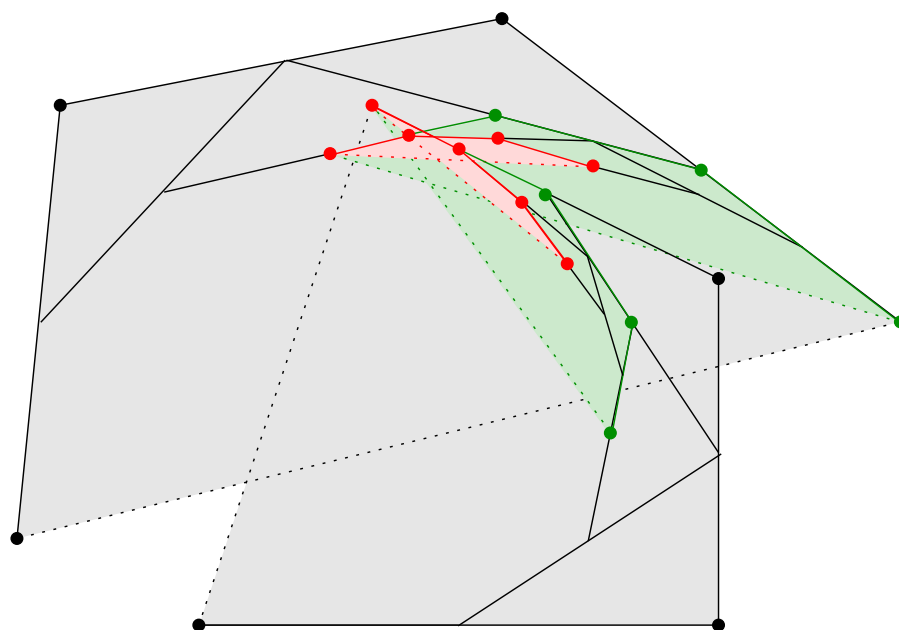
Konvexe-Hülle der Kontrollpunkte disjunkt $\Rightarrow$ kein Schnitt der Bézier-Kurven
 $\Rightarrow$ **Idee:** Rekursive Unterteilung der Bézier-Kurven liefert Schnittpunkte

Algorithmus: Zum Schnitt von Bézier-Kurven vom Grad n

```
void intersectCurve(Point cpA[], cpB[], vector intersectPts) {
    Point cpALeft[n+1], cpARight[n+1], cpBLeft[n+1], cpBRight[n+1];
    if ( ! intersectConvexHull(cpA, cpB) ) return;
    if ( sizeofBBBox(cpA) > epsilon ) {          // subdivide curve A
        subdivideCurve(cpA, cpALeft, cpARight);
        intersectCurve(cpALeft, cpB, intersectPts);
        intersectCurve(cpARight, cpB, intersectPts);
    }
    else if ( sizeofBBBox(cpB) > epsilon ) { // subdivide curve B
        subdivideCurve(cpB, cpBLeft, cpBRight);
        intersectCurve(cpA, cpBLeft, intersectPts);
        intersectCurve(cpA, cpBRight, intersectPts);
    }
    else { // both boxes smaller than epsilon
        intersectPts.append(computeCenter(cpA));
    }
    return;
}
```



Beispiel: Schnitt zwischen Bézier-Kurven



3.1 Bézier-Kurven ...



Bewertung Bézier-Kurven

Wesentliche Vorteile:

- Intuitive Kontrolle des Kurvenverlaufes
- stabile Berechnung (de Casteljau); alternativ schnelle, äquidistante Auswertung (Vorw.-Diff.)
- affine Invarianz und konvexe Hülle
- Darstellung aller Polynome und Kegelschnitte

Wesentliche Nachteile:

- Globaler Einfluss der Kontrollpunkte
- Nur Polynomkurven bzw. Kegelschnitte (rationale Bézier-Kurven)
-

Anzahl Freiheitsgrade = Anzahl Kontrollpkte = Polynomgrad + 1
⇒ mehr Freiheitsgrade nur durch höheren Polynomgrad und mehr Rechenaufwand!



3.2 Bézier-Splines



Bezeichnung: Bézier-Splinekurve

Bézier-Splinekurve: Mehr Freiheitsgrade durch *Aneinanderfügen von Bézier-Kurven*

Konkret: $k + 1$ Bézier-Segmente $C^j(u)$, $j = 0, \dots, k$ vom Grad n

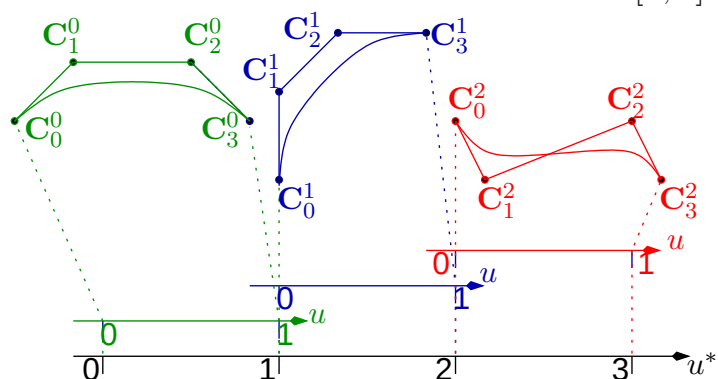
Lokaler Parameter: Segment C^j hat als Bézier-Kurve den Param. $u \in [0, 1]$

Globaler Parameter:

Parameter $u^* \in [0, k + 1]$
der gesamten Splinekurve
 C mit

$$u^* \in [j, j + 1[\\ \Rightarrow C(u^*) = C^j(u)$$

mit $u = u^* - j$



Frage: Welche Bedingungen müssen gelten, damit C eine stetige und/oder glatte Kurve ist?



3.2 Bézier-Splines ...



Exkursion: Stetigkeitsbedingungen

Formal: Eine Funktion $f(u)$ ist in u_0 C^k -stetig, wenn sie

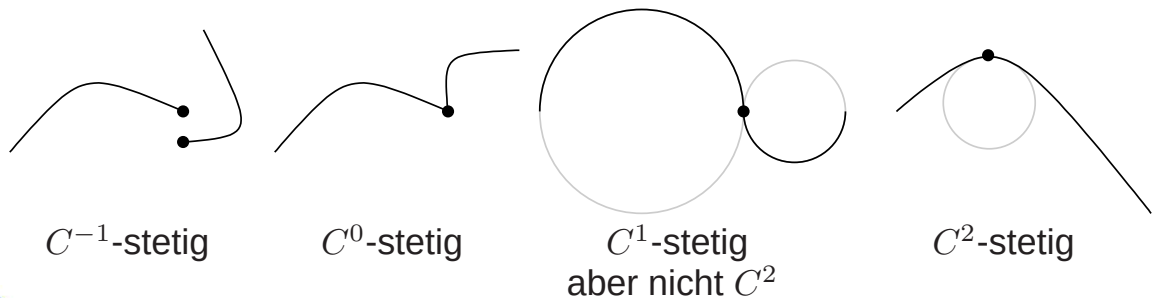
1. in u_0 k -mal differenzierbar ist und
2. die k -te Ableitung in u_0 stetig ist

Konkret: ○ C^{-1} : Die Kurve ist unstetig (Sprung)

○ C^0 : Die Kurve stetig aber nicht diff.-bar („Knick“)

○ C^1 : Die Kurve einmal diff.-bar mit stetiger Tangente

○ C^2 : Die Kurve zweimal diff.-bar mit 2. Ableitung (krümmungsstetig)



3.2 Bézier-Splines ...

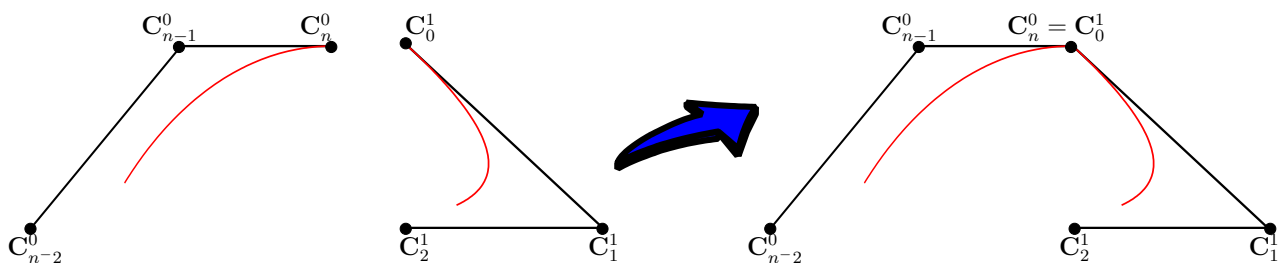


Anschlussbedingungen (C^0)

Gegeben: Kurve C^0 mit Kontrollpunkten $C_0^0, \dots, C_n^0$

Gesucht: Kontrollpunkte C_i^1 von C^1 , so dass C^1 stetig/glatt an C^0 in $C^0(1)$ anschließt.

C^0 -stetig: Es muss $C^0(1) = C^1(0)$ gelten, also $C_n^0 = C_0^1$
(Endpunktinterpolation)



3.2 Bézier-Splines ...



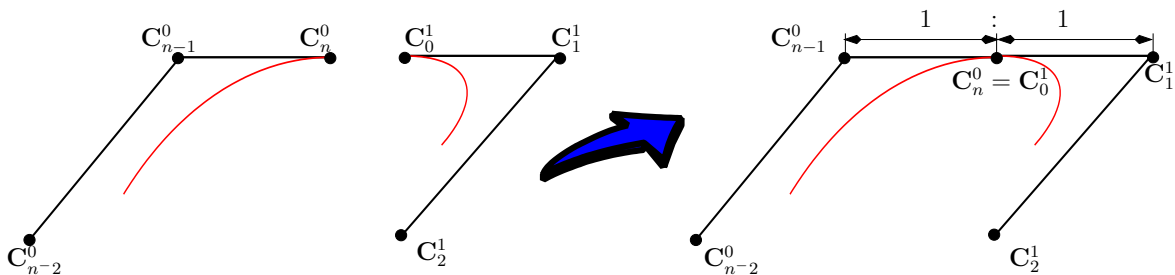
Anschlussbedingungen (C^1)

Voraussetzung: C^0 -Stetigkeit, also $C_n^0 = C_0^1$

Zudem muss gelten: $(C^0)'(1) = (C^1)'(0)$ (Interpolation der Endtangenten)

$$(C^0)'(1) = n(C_n^0 - C_{n-1}^0) \stackrel{!}{=} n(C_1^1 - C_0^1) = (C^1)'(0)$$

$$\overset{C^0\text{-stetig}}{\Leftrightarrow} C_1^1 = C_0^1 + (\underbrace{C_n^0 - C_{n-1}^0}_{=C_0^1})$$



3.2 Bézier-Splines ...



Anschlussbedingungen (C^2)

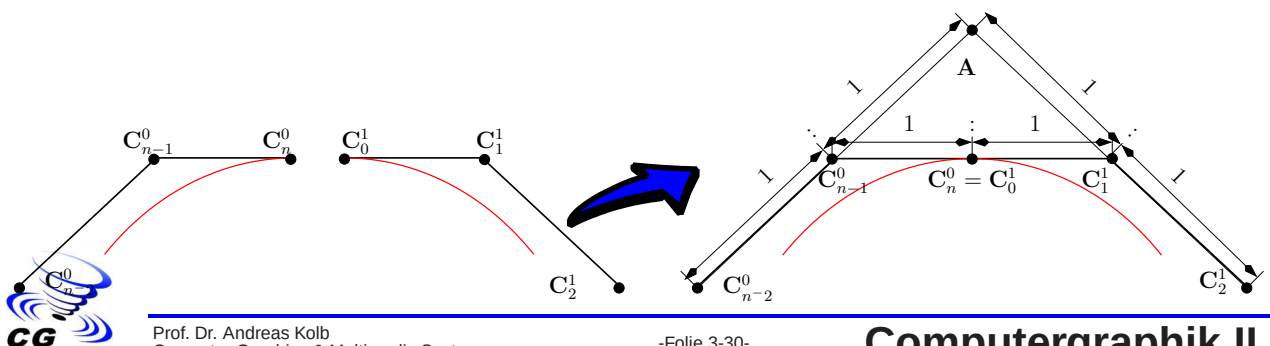
Voraussetzung: C^1 -Stetigkeit, also $C_n^0 = C_0^1$ und $C_n^0 = C_0^1 = \frac{1}{2}(C_{n-1}^0 + C_1^1)$

Zudem muss gelten: $(C^0)''(1) = (C^1)''(0)$, konkret

$$\text{Allgemein: } C''(u) = n(n-1) \sum_{i=0}^{n-2} (C_{i+2} - 2C_{i+1} + C_i) B_i^{n-2}(u)$$

$$\text{Damit: } (C^0)''(1) = (C^1)''(0) \Leftrightarrow C_n^0 - 2C_{n-1}^0 + C_{n-2}^0 = C_2^1 - 2C_1^1 + C_0^1$$

$$\overset{C^1\text{-stetig}}{\Leftrightarrow} C_2^1 = C_1^1 + \underbrace{(C_1^1 - (C_{n-1}^0 + (C_{n-1}^0 - C_{n-2}^0)))}_{=A}, \text{ da } C_n^0 = C_0^1$$



3.2.1 Catmull-Rom-Ansatz



Ansatz: Verwendung von *kubische* C^1 -stetigen Bézier-Splines (vgl. Hermite!)

Kontrollgrößen: Interpoliert werden

1. Kontrollpunkten $C_0^j, j = 0, \dots, k$ und C_3^k
2. Tangenten $\vec{t}_j = (C^j)'(0), j = 0, \dots, k$ und $\vec{t}_{k+1} = (C^k)'(1)$

Angabe von zwei Kontrolltypen (Punkte, Tangenten) ist unhandlich.

Catmull-Rom-Splines

Ansatz: Vorgabe der Kontrollpunkte & Schätzung der Tangenten

Gegeben: Interpolationspunkte $P_i, i = 0, \dots, k + 1$

Berechnung der Bézier-Punkte: Für das j -te Bézier-Segment ergibt sich

$$\text{Endpunkte: } C_0^j = P_j, C_3^j = P_{j+1}, \quad j = 0, \dots, k$$

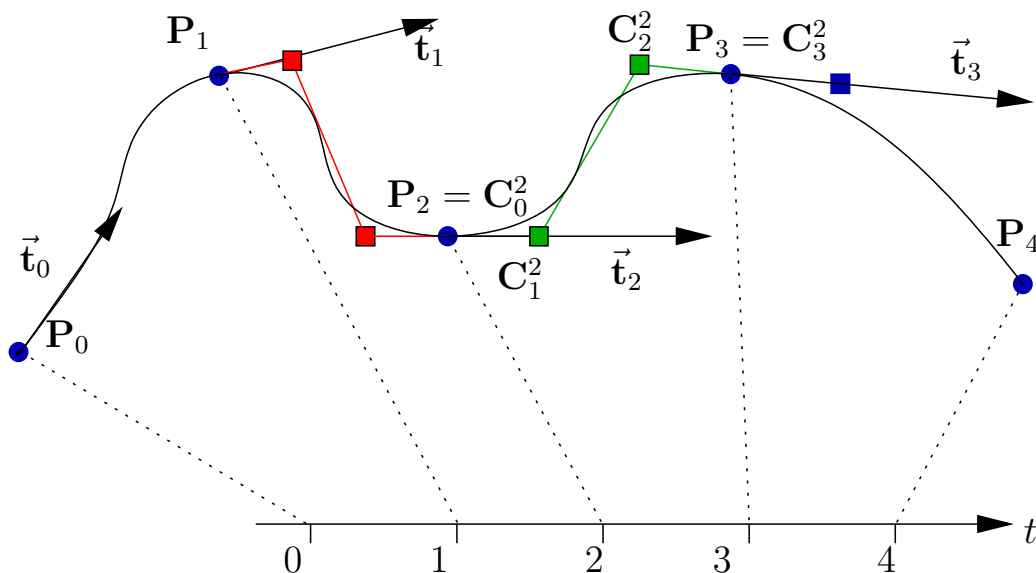
$$\text{Tangente bei } P_j : \vec{t}_j = \frac{1}{2} (P_{j+1} - P_{j-1}), \quad j = 1, \dots, k - 1$$

$$\text{Tangenten am Rand: Z.B. } \vec{t}_0 = \frac{1}{2} (P_1 - P_0) \text{ und } \vec{t}_{k+2} = \frac{1}{2} (P_{k+2} - P_{k+1})$$

$$\text{Innere Kontrollpunkte: } C_1^j - C_0^j = \frac{1}{3} \vec{t}_j \Rightarrow C_1^j = C_0^j + \frac{1}{3} \vec{t}_j, \text{ analog } C_2^j = C_3^j - \frac{1}{3} \vec{t}_{j+1}$$



3.2.1 Catmull-Rom-Ansatz ...



$$\begin{aligned} \text{Tangente bei } P_j : & \quad \vec{t}_j = \frac{1}{2} (P_{j+1} - P_{j-1}) \\ \text{Interpolation der Endpunkte:} & \quad C_0^j = P_j, C_3^j = P_{j+1} \\ \text{Tangentialstetigkeit:} & \quad (C_1^j - C_0^j) = (C_3^{j-1} - C_2^{j-1}) = \frac{1}{3} \vec{t}_j \end{aligned}$$





Bewertung Bézier-Splines

Wesentliche Vorteile:

- Alle Vorteile der Bézier-Kurven
- Zusätzliche Freiheitsgrade bei konstantem Grad

Wesentliche Nachteile:

- Erfüllung expliziter Übergangsbedingungen **oder**
- verschiedenartige Kontrollparameter (Catmull-Rom)

Wünschenswert:

- „Eingebaute“ Übergangsbedingung
- maximale Stetigkeit bei minimalem Einfluss der Kontrollpunkte



3.2.2 Hermite-Kurven



Bézier-Kurven interpolieren C_0, C_n und approximieren $C_i, i \notin \{0, n\}$

Hermite-Kurven interpolieren Position und Ableitungen in den Endpunkten

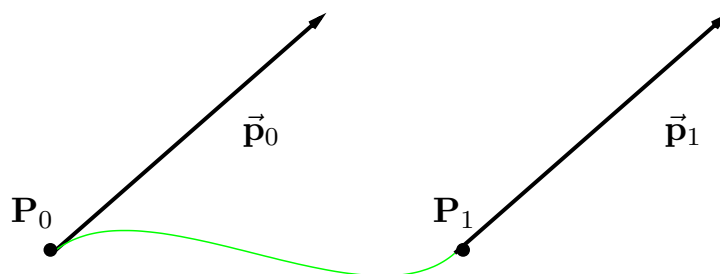
Zahl der Kontrollparameter legt den Grad der Kurve fest:

Position und 1. Ableitung $\rightarrow$ 4 Parameter $\rightarrow$ kubische Polynome

Position, 1. und 2. Ableitung $\rightarrow$ 6 Parameter $\rightarrow$ quintische Polynome etc.

Kubische Hermite-Kurve $H(u)$ interpolieren $P_0, P_1, \vec{p}_0, \vec{p}_1$:

$$H(0) = P_0, \quad H(1) = P_1, \quad H'(0) = \vec{p}_0, \quad H'(1) = \vec{p}_1$$



3.2.2 Hermite-Kurven ...



Kubische Hermite-Basisfunktionen

Kubische Hermite-Kurven definiert sich zu

$$\mathbf{H}(u) = \mathbf{P}_0 h_0(u) + \vec{\mathbf{p}}_0 h_1(u) + \vec{\mathbf{p}}_1 h_2(u) + \mathbf{P}_1 h_3(u)$$

Kubische Hermite-Basisfunktionen:

$$h_0(u) = 2u^3 - 3u^2 + 1$$

$$\text{mit } h_0(0) = 1, h_0(1) = 0, h'_0(0) = 0, h'_0(1) = 0$$

$$h_1(u) = u^3 - 2u^2 + u$$

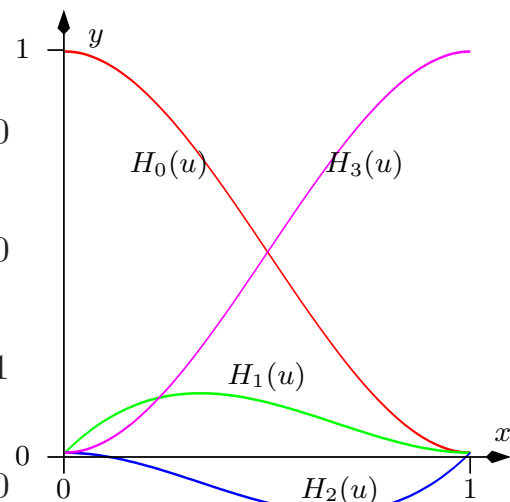
$$\text{mit } h_1(0) = 0, h_1(1) = 0, h'_1(0) = 1, h'_1(1) = 0$$

$$h_2(u) = u^3 - u^2$$

$$\text{mit } h_2(0) = 0, h_2(1) = 0, h'_2(0) = 0, h'_2(1) = 1$$

$$h_3(u) = -2u^3 + 3u^2$$

$$\text{mit } h_3(0) = 0, h_3(1) = 1, h'_3(0) = 0, h'_3(1) = 0$$



3.3 B-Spline-Kurven



Ansatz: Quadratische B-Spline Kurven

Idee: Nutze innere Kontrollpunkte C^1 -stetiger quadratischer Bézier-Splines als Kontrollpunkte

Gegeben: „Innere Kontrollpunkte“ $D_0, \dots, D_k$

Konstruktion: Sequenz

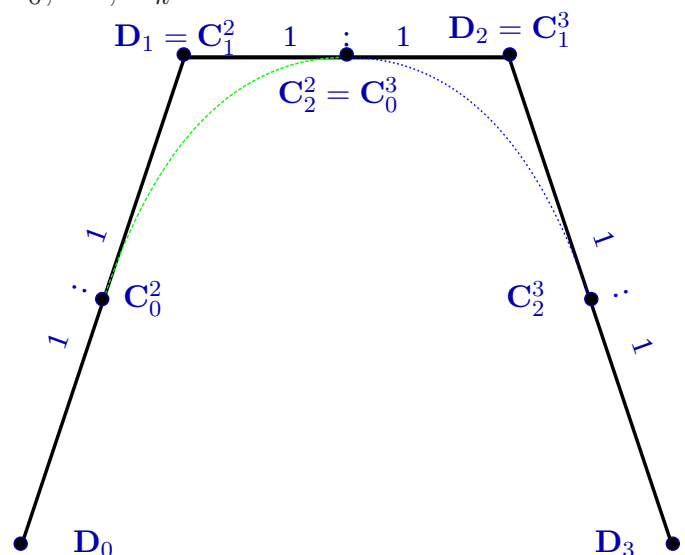
$D_{j-2}, \dots, D_j$ definiert die quadratische Bézier-Kurve C^j :

$$C_0^j = \frac{1}{2}D_{j-2} + \frac{1}{2}D_{j-1},$$

$$C_1^j = D_{j-1}$$

$$C_2^j = \frac{1}{2}D_{j-1} + \frac{1}{2}D_j$$

Beachte: Automatisch C^1 -stetig



3.3 B-Spline-Kurven ...



Ansatz: Kubische B-Spline Kurven

Idee: Nutze A-Frame-Punkte C^2 -stetiger kubischer Bézier-Splines als Kontrollpunkte

Gegeben: A-Frame Punkte $D_1, \dots, D_k$

Konstruktion: Sequenz

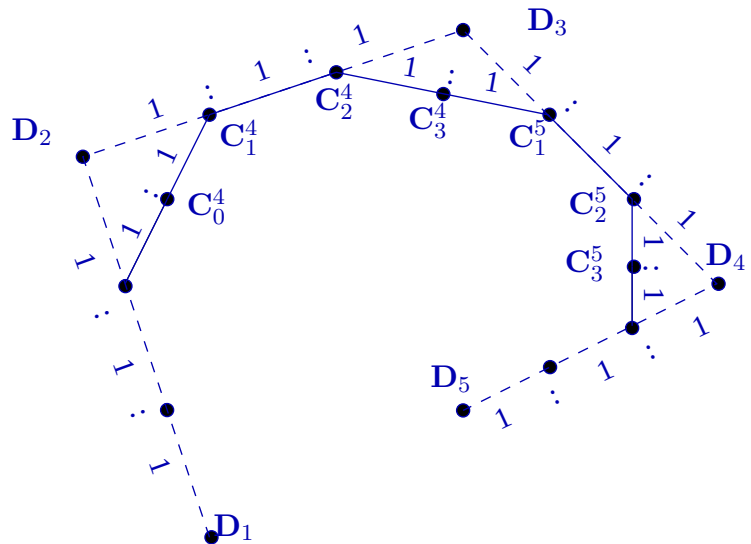
$D_{j-3}, \dots, D_j$ definiert
kubische Bézier-Kurve
 C^j :

$$C_1^j = \frac{2}{3}D_{j-2} + \frac{1}{3}D_{j-1}$$

$$C_2^j = \frac{1}{3}D_{j-2} + \frac{2}{3}D_{j-1}$$

$$C_0^j = \frac{1}{2}C_2^{j-1} + \frac{1}{2}C_1^j$$

$$C_3^j = \frac{1}{2}C_2^j + \frac{1}{2}C_1^{j+1}$$



3.3 B-Spline-Kurven ...



Bemerkung: Quadratische und kubische B-Splines

Obige Einführung ist sehr speziell, denn wir brauchen

- B-Splines für beliebigen Grad n & beliebige Parameterintervalle
- Auswertung ohne Umweg über die Bézier-Darstellung

Basisfunktionen: Angenommen, obige B-Spline-Kurven haben die Darstellung

$$D(u) = \sum_i D_i N_i^n(u) \text{ für unbekannte Basisfunktionen } N_i^n, \quad n = 2, 3$$

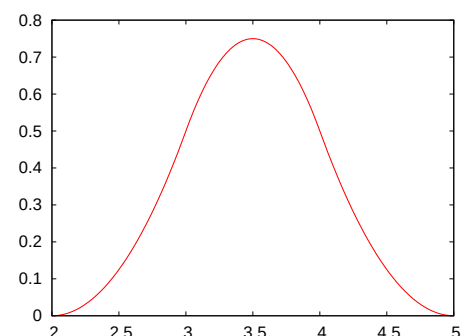
Für ein i_0 gewinnt man $N_{i_0}^n$ durch: $D_i = 0, i \neq i_0$ und $D_{i_0} = 1$.

Beispiel: Bestimmung von $N_2^2(u)$ als stückw.
quadratisches Polynom (lok. Param. für C^j :
 $u = u^* - j$):

$$u^* \in [2, 3[: \quad N_2^2(u^*) = C^2(u) = \frac{1}{2}B_2^2(u)$$

$$u^* \in [3, 4[: \quad N_2^2(u^*) = C^3(u) \\ = \frac{1}{2}(B_0^2(u) + B_2^2(u)) + B_1^2(u)$$

$$u^* \in [4, 5[: \quad N_2^2(u^*) = C^4(u) = \frac{1}{2}B_0^2(u)$$



3.3 B-Spline-Kurven ...



Ziel: Basisfunktion als stückweises Polynom $N_i^n(u)$ mit gegebenem Grad $n \in \mathbb{N}$, so dass

1. max. Stetigkeit zwischen den Teilkurven, also $(n-1)$ -**stetig**
2. möglichst geringer Einflussbereich der Kontrollpunkte (**Lokalität**)
3. Erlegung der Eins: $\sum N_i^n(u) \equiv 1$, $N_i^n(u) \geq 0$ (**affine Invarianz, konvexe Hülle**)

Knotenvektor: $T = \{\dots, t_{i-1}, t_i, t_{i+1}, \dots\}$, $t_i \leq t_{i+1}$, $t_i \in \mathbb{R}$ mit:

$$N_i^n|_{[t_i, t_{i+1}[} \in \mathcal{P}^n$$

Fall n=0: Offensichtlich sinnvolle Festlegung $N_i^0 = \chi([t_i, t_{i+1}[)$, wobei

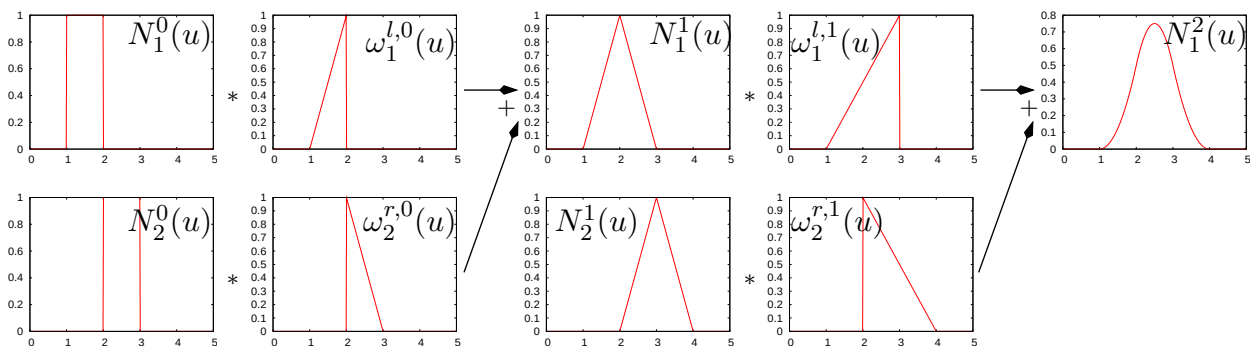
$$\chi(M)(u) = \begin{cases} 1 & u \in M \\ 0 & \text{sonst} \end{cases} \quad \text{charakteristische Funktion}$$



3.3 B-Spline-Kurven ...

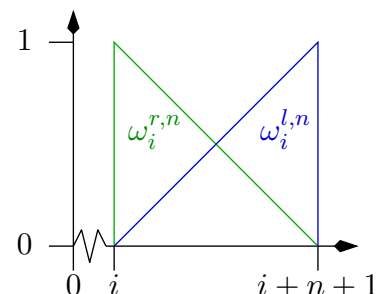


Rekursion: $N_i^{n+1}(u)$ ergibt sich aus gewichteter Summe von $N_i^n(u)$ und $N_{i+1}^n(u)$ mit linearen $\omega_i^{l,n}, \omega_i^{r,n}$



Bemerkung:

- $\omega_i^{l,n}, \omega_i^{r,n}$ steigt/fällt auf $\text{supp}(N_i^n)$ von 0 nach 1 an bzw. von 1 nach 0 ab
- Es gilt $\sum N_i^n \equiv 1$ (Zerlegung der Eins, s.u.) und $N_i^n(u) \geq 0$



3.3 B-Spline-Kurven ...



Definition: Normalisierte B-Spline-Basisfunktionen

Gegeben: Knotenvektor $T = \{\dots, t_{i-1}, t_i, t_{i+1}, \dots\}$, $t_i \leq t_{i+1}$, $t_i \in \mathbb{R}$
 Der Knotenvektor T heißt **uniform**, falls $t_i = i$

Fall $n = 0$: Setze $N_i^0 := \chi([t_i, t_{i+1}[)$

Rekursion $n \rightarrow n + 1$: $N_i^{n+1}(u) := \omega_i^{l,n}(u) \cdot N_i^n(u) + \omega_{i+1}^{r,n}(u) \cdot N_{i+1}^n(u)$ mit

$$\omega_i^{l,n}(u) = \frac{u - t_i}{t_{i+n+1} - t_i}, \quad \omega_{i+1}^{r,n}(u) = \frac{t_{i+n+1} - u}{t_{i+n+1} - t_i}$$

Bemerkung: Zerlegung der 1

N_i^n bilden Zerlegung der Eins **im Inneren des Knotenvektors**, da $N_i^n(u) \geq 0$ und per Induktion: $\sum N_i^0 \equiv 1$ und

$$\begin{aligned} \sum_i N_i^{n+1}(u) &= \sum_i \left(\omega_i^{l,n}(u) N_i^n(u) + \omega_{i+1}^{r,n}(u) N_{i+1}^n(u) \right) \\ &= \sum_i \underbrace{\left(\omega_i^{l,n}(u) + \omega_{i+1}^{r,n}(u) \right)}_{=1} N_i^n(u) = 1 \end{aligned}$$



3.3 B-Spline-Kurven ...



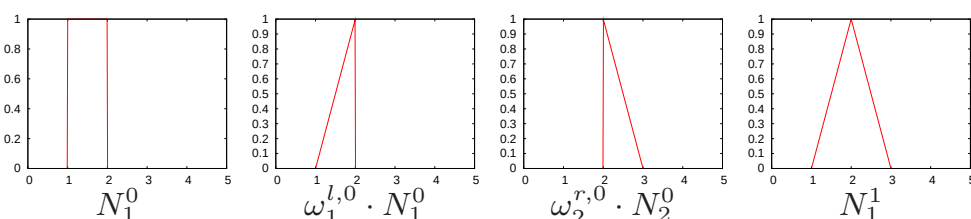
Beispiel: Normalisierte B-Spline-Basisfunktionen

Gegeben: Uniformer Knotenvektor

$$n = 0: N_i^0 = \chi[i, i + 1[= \begin{cases} 1 & \text{falls } u \in [i, i + 1[\\ 0 & \text{falls } u \notin [i, i + 1[\end{cases}$$

$$n = 1: \omega_i^{l,0}(u) = u - i, \quad \omega_i^{r,0}(u) = i + 1 - u \quad \text{und} \quad N_i^1 = \omega_i^{l,0} \cdot N_i^0 + \omega_{i+1}^{r,0} \cdot N_{i+1}^0$$

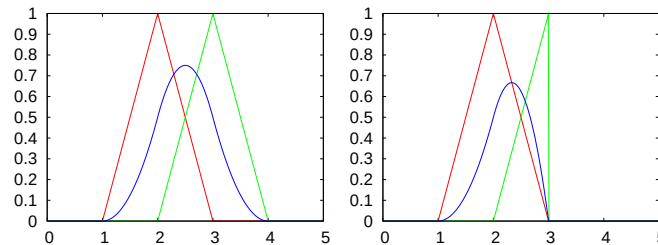
$$\begin{aligned} N_i^1(u) &= (u - i) \cdot \chi[i, i + 1[+ (i + 2 - u) \cdot \chi[i + 1, i + 2[\\ &= \begin{cases} u - i & \text{falls } u \in [i, i + 1[\\ i + 2 - u & \text{falls } u \in [i + 1, i + 2[\\ 0 & \text{sonst} \end{cases} \end{aligned}$$





Einfluss der Knoten t_i

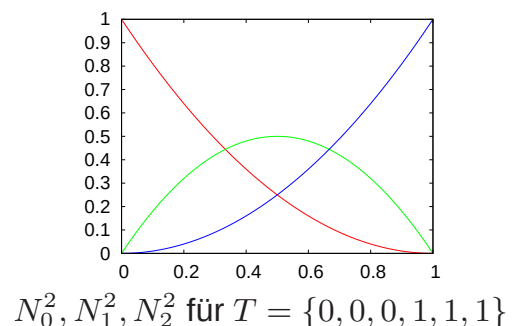
1. k -facher Knoten $t_{i-1} < t_i = t_{i+1} = \dots = t_{i+k-1} < t_{i+k}$ reduziert Stetigkeit um k ; Beispiel N_1^1, N_2^1, N_1^2 uniform (links) und $T = \{1, 2, 3, 3, 4\}$ (rechts)



2. $T = \{ \underbrace{0, \dots, 0}_{(n+1)\text{-fach}}, \underbrace{1, \dots, 1}_{(n+1)\text{-fach}} \}$ liefert:

$$N_i^n = B_i^n, \forall i = 0, \dots, n$$

3. $N_0^n, \dots, N_m^n, m > n$ bilden auf $[t_n, t_{m+1}]$ eine Basis der stückweisen Polynome vom Grad n mit der durch T gegebenen Stetigkeit



3.3 B-Spline-Kurven ...



B-Spline-Kurven

Eine *B-Spline-Kurve* ist eine Affin-Kombination von *de Boor-Punkten* D_i mit Gewichten N_i^n zu gegebenem Knotenvektor $T = \{t_0, \dots, t_{n+m+1}\}$

$$D(u) := \sum_{i=0}^m D_i N_i^n(u), \quad u \in [t_n, t_{m+1}]$$

Eigenschaften von B-Spline Kurven

Polynomiell: D ist eine stückweise Polynomkurve vom Grad n mit C^{n-1} -Übergängen bei disjunkten Knoten bzw. C^{n-k} bei k -fachem Knoten

Lokalität: D_i beeinflusst Kurve nur lokal, da

$$\text{supp}(N_i^n) = \{u : N_i^n(u) \neq 0\} =]t_i, t_{i+n+1}[\quad (\text{lokaler Träger})$$

Affine Invarianz und konvexe Hülle da $\sum_{i=0}^m N_i^n = 1$ und $N_i^n \geq 0$ auf $[t_n, t_{m+1}]$

Lokale konvexe Hülle: $D|_{[t_i, t_{i+1}[}$ verläuft in Hülle der $D_{i-n}, \dots, D_i$



3.3 B-Spline-Kurven ...



Algorithmus: de Boor

Gegeben: de Boor Punkte $D_0, \dots, D_m$ und der Parameter $u \in [t_n, t_{m+1}]$ (beliebig, aber fest)

Gesucht: Kurvenpunkt $D(u)$ direkt aus den D_i ohne Berechnung der N_i^n :

Relevante de Boor-Punkte: Für $u \in [t_r, t_{r+1}[$ sind alle D_i mit $N_i^n(u) \neq 0$ notwendig:

$$D_i^0 = D_i, \quad i = r - n, \dots, r$$

Rekursion: Affin-Kombinationen benachbarter Kontrollpunkte:

$$D_i^j = (1 - \alpha_i^j(u))D_{i-1}^{j-1} + \alpha_i^j(u)D_i^{j-1}, \quad i = r - n + j, \dots, r, \quad j = 1, \dots, n$$

mit Gewichten:

$$\alpha_i^j(u) = \frac{u - t_i}{t_{i+n+1-j} - t_i}, \quad i = r - n + j, \dots, r, \quad j = 1, \dots, n$$

Ergebnis: $D(u) = D_r^n$



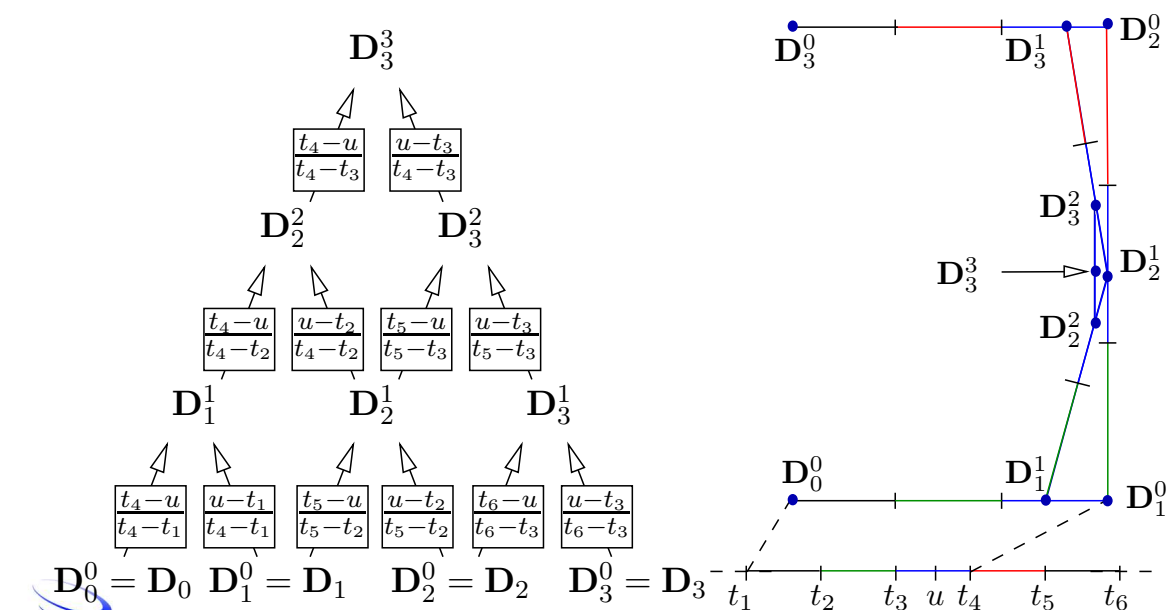
3.3 B-Spline-Kurven ...



de Boor: Geometrische Interpretation

Gegeben: de Boor-Punkte $D_0, \dots, D_m$ und $u \in [t_r, t_{r+1}[$

Geometrische Lösung: Hier mit $n = 3$ und $r = 3$



3.3 B-Spline-Kurven ...



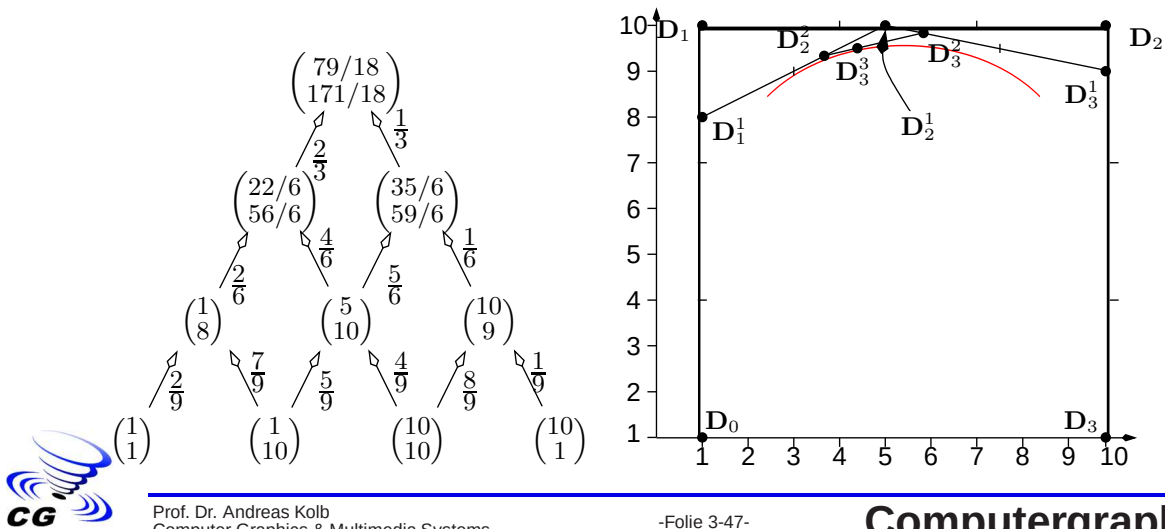
Beispiel: de Boor Algorithmus (uniformer, kubischer Fall)

Gegeben:

$$\mathbf{D}_0 = (1, 1), \mathbf{D}_1 = (1, 10), \mathbf{D}_2 = (10, 10), \mathbf{D}_3 = (10, 1), u = 10/3, r = 3$$

Gewichte: Ermittlung der Gewichte $\alpha_i^j = \alpha_i^j(10/3) = \frac{u-i}{4-j}$:

$$\alpha_1^1 = 7/9, \quad \alpha_2^1 = 4/9, \quad \alpha_3^1 = 1/9, \quad \alpha_2^2 = 4/6, \quad \alpha_3^2 = 1/6, \quad \alpha_3^3 = 1/3$$



Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-47-

Computergraphik II

3.3 B-Spline-Kurven ...



Ableitung von B-Spline-Kurven

Ableitung eines (stückw.) Polynoms reduziert den Grad von n auf $n - 1$

Ableitung des B-Splines mit Knotenvektor $T = \{t_0 \leq t_1 \leq \dots \leq t_{m+n+1}\}$:

$$\mathbf{D}(u) = \sum_{i=0}^m \mathbf{D}_i N_i^n(u) \implies \mathbf{D}'(u) = n \sum_{i=0}^m \frac{\mathbf{D}_i - \mathbf{D}_{i-1}}{t_{i+n} - t_i} N_i^{n-1}(u)$$

Knoteneinfügen (nach Böhm '80)

Stückw. Polynom über

$$T = \{t_0, t_1, \dots, t_{m+n+1}\}$$

ist auch stückw. Polynom
über

$$T^* = \{t_0, \dots, t_r, t^*, t_{r+1}, \dots, t_{m+n+1}\}$$

Kontrollpunkte berechnen sich
über de Boor-Algorithmus für

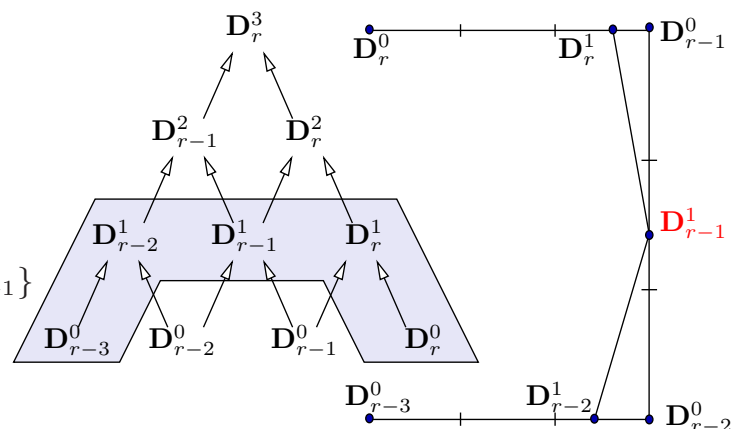


$$u = t^*$$

Prof. Dr. Andreas Kolb
Computer Graphics & Multimedia Systems

-Folie 3-48-

Computergraphik II



3.4 Tensor-Produkt Flächen



Idee: Tensor-Produkt Flächen

Ausgangslage: Kurvenschema $\mathbf{K}(u) = \sum_{i=0}^n \mathbf{P}_i F_i(u)$

Variation der Kontrollpunkte $\mathbf{P}_i$ entlang anderer Kurve

$$\mathbf{P}_i = \mathbf{P}_i(v) = \sum_{j=0}^m \mathbf{P}_{ij} F_j(v)$$

TP-Bézier-Flächen: I.a. werden Polynomkurven desselben Grades verwendet:

$$\mathbf{C}(u, v) = \sum_{i=0}^n \sum_{j=0}^n \mathbf{C}_{ij} B_i^n(u) B_j^n(v)$$

TP-B-Spline-Flächen haben i.a. unterschiedlich viele Segmente in u - und v -Richtung:

$$\mathbf{D}(u, v) = \sum_{i=0}^{m_u} \sum_{j=0}^{m_v} \mathbf{D}_{ij} N_i^n(u) N_j^n(v)$$



3.4 Tensor-Produkt Flächen ...

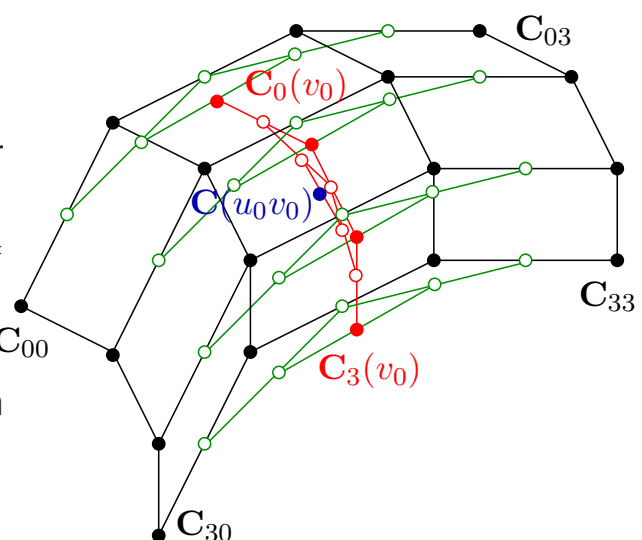


Auswertung von Bézier-Flächen

Zweistufiger de Casteljau: Separate Auswertung für den Parameter (u_0, v_0) :

1. $\forall i = 0, \dots, n : \mathbf{C}_i(v_0) = \sum_{j=0}^n \mathbf{C}_{ij} B_j^n(v_0)$
2. $\mathbf{C}(u_0, v_0) = \sum_{i=0}^n \mathbf{C}_i(v_0) B_i^n(u_0)$

Beachte: Die Rolle von u und v kann vertauscht werden



3.4 Tensor-Produkt Flächen ...



Bemerkung: Ableitungen von Funktionen mehrerer Veränderlicher:

Ableitung einer Funktion $f : \mathbb{R} \rightarrow \mathbb{R} : f'(u) = \frac{\partial f}{\partial u}(u) = \lim_{\delta \rightarrow 0} \frac{f(u+\delta) - f(u)}{\delta}$.

Ableitung einer Kurve $\mathbf{F} : \mathbb{R} \rightarrow \mathbb{R}^3 : \mathbf{F}'(u) = \frac{\partial \mathbf{F}}{\partial u}(u) = \lim_{\delta \rightarrow 0} \frac{\mathbf{F}(u+\delta) - \mathbf{F}(u)}{\delta}$.

Partielle Ableitungen einer Fläche $\mathbf{F} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$: Leite nach einer Variablen ab, halte die andere dabei konstant.

$$\frac{\partial \mathbf{F}}{\partial u}(u, v) := \lim_{\delta \rightarrow 0} \frac{\mathbf{F}(u + \delta, v) - \mathbf{F}(u, v)}{\delta}$$

$$\frac{\partial \mathbf{F}}{\partial v}(u, v) := \lim_{\delta \rightarrow 0} \frac{\mathbf{F}(u, v + \delta) - \mathbf{F}(u, v)}{\delta}$$

Also: Verwende Ableitungsregeln wie gewohnt, einmal nur für u , einmal nur für v .



3.4 Tensor-Produkt Flächen ...



Eigenschaften von TP-Bézier-Flächen

Interpolation der Ecken: $\mathbf{C}(0, 0) = \mathbf{C}_{00}, \mathbf{C}(1, 0) = \mathbf{C}_{n0}, \dots$

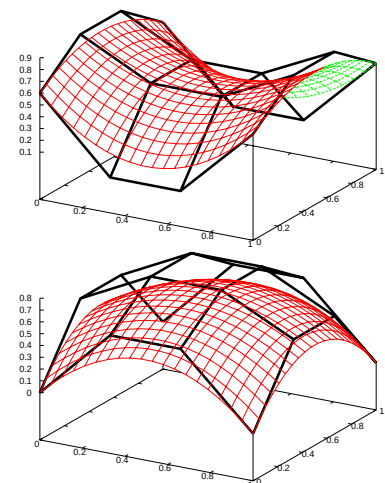
Ableitung:

$$\frac{\partial \mathbf{C}}{\partial u}(u, v) = \frac{\partial}{\partial u} \left(\sum_{i=0}^n \left(\sum_{j=0}^n \mathbf{C}_{i,j} B_j^n(v) \right) B_i^n(u) \right)$$

$$= n \sum_{(i,j)=(0,0)}^{(n-1,n)} (\mathbf{C}_{i+1,j} - \mathbf{C}_{i,j}) B_i^{n-1}(u) B_j^n(v)$$

$$\frac{\partial \mathbf{C}}{\partial u}(0, v) = n \sum_{j=0}^n (\mathbf{C}_{1,j} - \mathbf{C}_{0,j}) B_j^n(v)$$

$$\frac{\partial \mathbf{C}}{\partial u}(0, 0) = n (\mathbf{C}_{1,0} - \mathbf{C}_{0,0})$$



Globaler Einfluss der $\mathbf{C}_{ij}$, da $B_i^n(u) \cdot B_j^n(v) \neq 0$ auf $(u, v) \in]0, 1[^2$

Affine Invarianz & konvexe Hülle, da

$$\sum_{(i,j)} B_i^n(u) B_j^n(v) = (\sum_i B_i^n(u)) (\sum_j B_j^n(v)) = 1$$



3.4 Tensor-Produkt Flächen ...

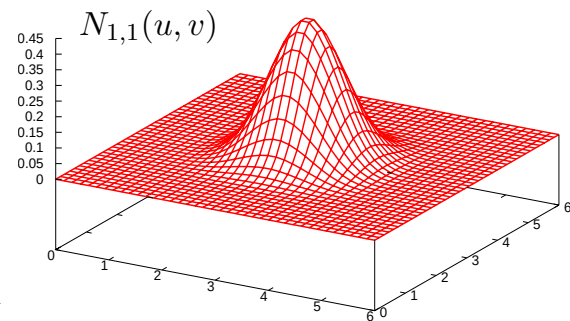
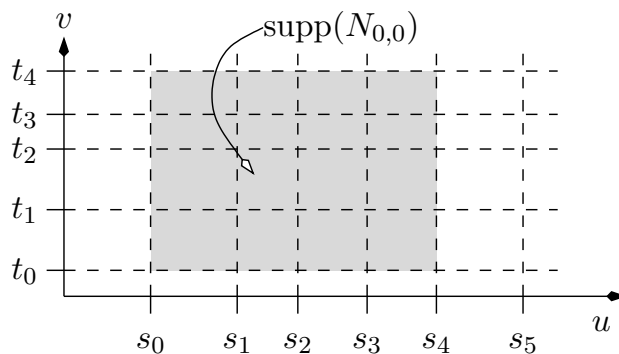


Eigenschaften von TP-B-Spline-Flächen

Knotenvektoren $S = \{s_0, s_1, \dots, s_{m_u+n+1}\}$, $T = \{t_0, t_1, \dots, t_{m_v+n+1}\}$ in u - bzw. v -Richtung

Lokaler Einfluss der D_{ij} : D_{ij} beeinflusst $D([s_i, s_{i+n+1}] \times [t_j, t_{j+n+1}])$, da

$$N_{ij}(u, v) := N_i^n(u) \cdot N_j^n(v) = 0 \quad \forall (u, v) \notin [s_i, s_{i+n+1}] \times [t_j, t_{j+n+1}]$$



Affine Invarianz und konvexe Hülle: gelten genauso, da

$$\sum_{(i,j)} N_i^n(u) N_j^n(v) = 1 \quad \text{auf } [s_n, s_{n+m_u+1}] \times [t_n, t_{n+m_v+1}]$$

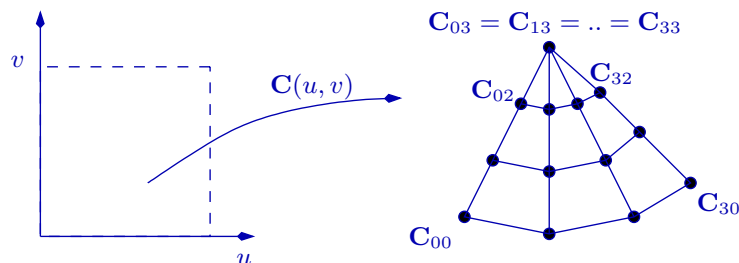


3.4 Tensor-Produkt Flächen ...



TP-Bézier-Splineflächen

- Probleme:**
1. Objekte i.a nicht mit einer Bézier- oder B-Spline-Fläche modellierbar.
 2. dreiecksförmige Flächen nur mit *Singularitäten* (zusammenfallen Kontrollpunkte)
 $\Rightarrow$ Verschwindende Ableitung, numerisch sehr problematisch!



Ansatz: Verwende mehrere Flächenstücke

Teilaufgaben beim Zusammenfügen von Splineflächen:

1. Kante-Kante Anschluß
2. Konfiguration um eine Ecke



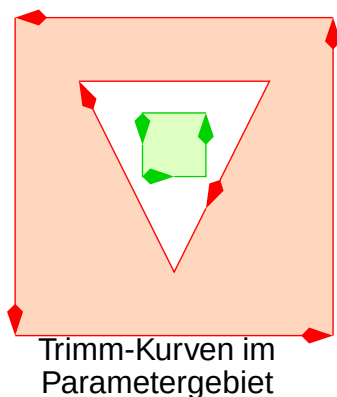
Getrimmte Splineflächen

Aufgabe: Beschreibung speziell geformte Flächen, z.B. mit Löchern

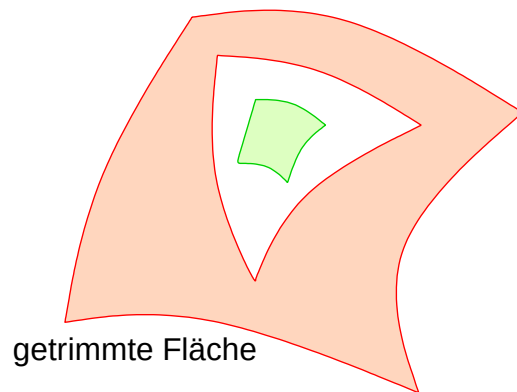
Ansatz: Verwende nur Teilbereiche des Parametergebiet, z.B. $[0, 1]^2$.

Beschreibe Teilbereiche durch geschlossene Kurven im Parametergebiet

1. pos. orientierte Kurven definieren Innenbereich
2. neg. orientierte Kurven definieren Aussenbereich



Trimm-Kurven im
Parametergebiet



getrimmte Fläche



3.5 Differentialgeometrie für Kurven

Bislang: Zusammengesetzte Kurven über C^k -Stetigkeit erklärt

Mehrdeutigkeit: Geometrische Form (Kurve) durch verschiedene Funktionen darstellbar:

Für $C(u)$, $u \in [a, b]$ und bijektives $g : [c, d] \rightarrow [a, b]$ erzeugt

$$D(u) := C(g(u)), \quad u \in [c, d]$$

dieselbe Geometrie, aber Ableitungen stimmen nicht überein:

$$\text{Kettenregel: } D'(u) = C'(g(u)) \cdot g'(u) \neq C'(g(u)) \text{ falls } g'(u) \neq 1$$

Geometrische Stetigkeit: Kurven C und D sind in $D(u_0)$ G^k -stetig, falls:

$$\exists g \in C^k \text{ bijektiv, : } D^{(i)}(u_0) = \frac{d^i}{d u^i} C(g(u_0)), \quad i = 0, \dots, k$$

Konkret:

$$G^1\text{-Stetigkeit: } D'(u_0) = C'(g(u_0)) \cdot \gamma_1, \quad \gamma_1 > 0$$

$$G^2\text{-Stetigkeit: } D''(u_0) = C''(g(u_0)) \cdot (\gamma_1)^2 + C'(g(u_0)) \cdot \gamma_2, \quad \gamma_2 > 0$$



3.5 Differentialgeometrie für Kurven ...



Definition: Geometrische Größen einer Kurve

Ziel: Bestimme lokales Koordinatensystems zu einer Raumkurve (Kamerafahrt).

Gegeben: **Raumkurve** C , C^2 -stetig mit $C'(u)$, $C''(u) \neq \vec{0}$, $C'(u) \nparallel C''(u)$, $\forall u$

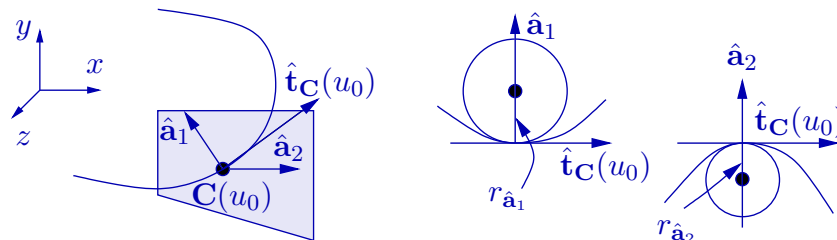
Geometrische Größen von C in u_0 :

Tangente $\hat{t}_C(u_0)$: Weist in Bewegungsrichtung: $\hat{t}_C(u_0) = C'(u_0) / \|C'(u_0)\|$

Krümmung $\kappa_C(u_0)$: Krümmung wird zunächst in einer Ebene

$\text{span}\{\hat{t}_C(u_0), \hat{a}\}$ mit beliebigem $\hat{a} \perp \hat{t}_C(u_0)$

definiert als: $\kappa_{C,\hat{a}}(u_0) = \frac{1}{r_{\hat{a}}}$, $r_{\hat{a}}$ Radius des bestangepassten Kreises



3.5 Differentialgeometrie für Kurven ...



Geometrische Größen einer Kurve (Fortsetzung)

Geometrische Größen von C in u_0 :

Normale $\hat{n}_C(u_0)$: Entspricht $\hat{a}$ mit max. Krümmung $\kappa_{C,\hat{a}}(u_0)$

Es gilt, dass $\hat{n}_C(u_0) \in \text{span}\{C'(u_0), C''(u_0)\}$.

Krümmung: $\kappa_C(u_0) = \kappa_{C,\hat{n}_C(u_0)}(u_0)$

Binormale $\hat{b}_C(u_0)$: Dritter Vektor senkrecht zur Tangente und Normale:

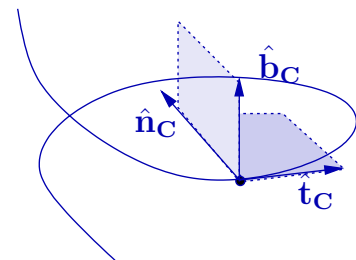
$$\hat{b}_C(u_0) := \hat{t}_C(u_0) \times \hat{n}_C(u_0)$$

Frenet'sche Dreibein: Das Koordinatensystem $\{\hat{t}_C(u_0), \hat{n}_C(u_0), \hat{b}_C(u_0)\}$ im Punkt $C(u_0)$.

Krümmungsvektor: $\kappa_C(u_0) \cdot \hat{n}_C(u_0)$

Schmiegeebene: $\text{span}\{\hat{t}_C(u_0), \hat{n}_C(u_0)\}$, lokal bestangepasste Ebene

Normalenebene: Ebene senkrecht zum Kurvenverlauf, also $\text{span}\{\hat{n}_C(u_0), \hat{b}_C(u_0)\}$



3.5 Differentialgeometrie für Kurven ...



Bemerkung: Berechnung des Frenet'schen Dreibeins

Das Koordinatensystem $\{\hat{\mathbf{t}}_{\mathbf{C}}(u_0), \hat{\mathbf{n}}_{\mathbf{C}}(u_0), \hat{\mathbf{b}}_{\mathbf{C}}(u_0)\}$ wird wie folgt ermittelt:

Tangente: $\hat{\mathbf{t}}_{\mathbf{C}}(u_0) = \frac{\mathbf{C}'(u_0)}{\|\mathbf{C}'(u_0)\|}$

Binormale: $\hat{\mathbf{b}}_{\mathbf{C}}(u_0) = \frac{\mathbf{C}'(u_0) \times \mathbf{C}''(u_0)}{\|\mathbf{C}'(u_0) \times \mathbf{C}''(u_0)\|}$

da $\mathbf{C}'(u_0)$ und $\mathbf{C}''(u_0)$ die Schmiegeebene aufspannen
und $\{\hat{\mathbf{t}}_{\mathbf{C}}(u_0), \hat{\mathbf{n}}_{\mathbf{C}}(u_0), \hat{\mathbf{b}}_{\mathbf{C}}(u_0)\}$ und damit auch
 $\{\mathbf{C}'(u_0), \mathbf{C}''(u_0), \hat{\mathbf{b}}_{\mathbf{C}}(u_0)\}$ rechtshändig ist

Normale: $\hat{\mathbf{n}}_{\mathbf{C}}(u_0) = \hat{\mathbf{b}}_{\mathbf{C}}(u_0) \times \hat{\mathbf{t}}_{\mathbf{C}}(u_0)$
(keine Normierung nötig, da beide schon orthonormal)

Krümmung: $\kappa_{\mathbf{C}}(u_0) = \frac{\|\mathbf{C}'(u_0) \times \mathbf{C}''(u_0)\|}{\|\mathbf{C}'(u_0)\|^3}$



3.5 Differentialgeometrie für Kurven ...



Beispiel: Frenet'sches Dreibein einer Spirale

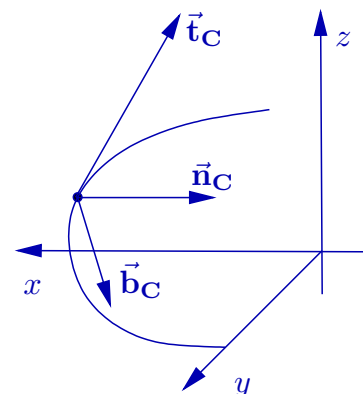
Spirale: $\mathbf{C}(u) = \begin{pmatrix} \sin u \\ \cos u \\ u \end{pmatrix}$; Ableitungen: $\mathbf{C}'(u) = \begin{pmatrix} \cos u \\ -\sin u \\ 1 \end{pmatrix}$; $\mathbf{C}''(u) = \begin{pmatrix} -\sin u \\ -\cos u \\ 0 \end{pmatrix}$

Mit $\|\mathbf{C}'(u)\| = \sqrt{2}$ und $\mathbf{C}'(u) \times \mathbf{C}''(u) = \begin{pmatrix} \cos u \\ -\sin u \\ -1 \end{pmatrix}$ folgt: $\hat{\mathbf{t}}_{\mathbf{C}}(u) = \frac{1}{\sqrt{2}} \begin{pmatrix} \cos u \\ -\sin u \\ 1 \end{pmatrix}$

$\hat{\mathbf{b}}_{\mathbf{C}}(u) = \frac{\mathbf{C}'(u) \times \mathbf{C}''(u)}{\|\mathbf{C}'(u) \times \mathbf{C}''(u)\|} = \frac{1}{\sqrt{2}} \begin{pmatrix} \cos u \\ -\sin u \\ -1 \end{pmatrix}$, $\hat{\mathbf{n}}_{\mathbf{C}}(u) = \hat{\mathbf{b}}_{\mathbf{C}}(u) \times \hat{\mathbf{t}}_{\mathbf{C}}(u) = \begin{pmatrix} -\sin u \\ -\cos u \\ 0 \end{pmatrix}$

Für $u = \frac{\pi}{2}$: $\hat{\mathbf{t}}_{\mathbf{C}}(\frac{\pi}{2}) = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix}$

$\hat{\mathbf{n}}_{\mathbf{C}}(\frac{\pi}{2}) = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$; $\hat{\mathbf{b}}_{\mathbf{C}}(\frac{\pi}{2}) = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ -1 \\ -1 \end{pmatrix}$



3.6 Interpolation mit B-Splines



Gegeben: k Parameter $\{u_1, \dots, u_k\}$ und zugehörige Punkte $\mathbf{P}_i \in \mathbb{R}^3, i = 1, \dots, k$.

Gesucht: **Interpolationskurve:** $C(u)$ mit $C(u_i) = \mathbf{P}_i, i = 1, \dots, k$.

Lösungsansätze: Folgende Kurven-Schemata sind denkbar

1. Catmull-Rom Splines (s.o.)
2. B-Splines

Ansatz: Interpolation an den Knoten mit kubischen uniformen B-Splines

Kurve: Kubische uniforme B-Spline-Kurve $\mathbf{D}$ mit (also $t_i = i$):

$$\mathbf{D}(u) = \sum_{i=0}^m \mathbf{D}_i N_i^3(u), \quad u \in [3, m+1]$$

Anzahl Segmente: In $[3, m+1]$ gibt es $m-1$ Knoten i , d.h.

$$u_1 = 3, u_2 = 4, \dots, u_k = k+2 \stackrel{!}{=} m+1 \text{ also } u_i = i+2 \\ \Rightarrow m = k+1, \text{ aber } m+1 = k+2 \text{ Kontrollpunkte (unterbestimmt!)}$$

Frage: Wie sind $\mathbf{D}_i$ zu wählen, so dass $\mathbf{D}(u_i) = \mathbf{D}(i+2) = \mathbf{P}_i, i = 1, \dots, k$?



3.6 Interpolation mit B-Splines ...



Ansatz: Interpolation mit uniformen B-Splines (Forts.)

Erinnerung: Uniforme, kubische B-Splines sind stückweise C^2 -stetige Bézier-Kurven

Umrechnung: $\mathbf{D}(u) \rightarrow \mathbf{C}^j(u)$ für $u \in [j, j+1]$:

$$\begin{aligned} \mathbf{C}_1^j &= \frac{2}{3}\mathbf{D}_{j-2} + \frac{1}{3}\mathbf{D}_{j-1} \\ \mathbf{C}_2^j &= \frac{1}{3}\mathbf{D}_{j-2} + \frac{2}{3}\mathbf{D}_{j-1} \\ \mathbf{C}_0^j &= \frac{1}{2}\mathbf{C}_2^{j-1} + \frac{1}{2}\mathbf{C}_1^j \\ &= \frac{1}{6}\mathbf{D}_{j-3} + \frac{2}{3}\mathbf{D}_{j-2} + \frac{1}{6}\mathbf{D}_{j-1} \\ \mathbf{C}_3^j &= \frac{1}{2}\mathbf{C}_2^j + \frac{1}{2}\mathbf{C}_1^{j+1} \\ &= \frac{1}{6}\mathbf{D}_{j-2} + \frac{2}{3}\mathbf{D}_{j-1} + \frac{1}{6}\mathbf{D}_j \end{aligned} \quad \underbrace{\begin{pmatrix} \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 & \dots & 0 \\ 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \dots & 0 \\ \vdots & \vdots & & \ddots & \ddots & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{pmatrix}}_{\in \mathbb{R}^{k, k+2}} \begin{pmatrix} \mathbf{D}_0 \\ \vdots \\ \mathbf{D}_{k+1} \end{pmatrix} = \begin{pmatrix} \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_k \end{pmatrix}$$

Bedingung: $\mathbf{P}_i = \mathbf{D}(u_i) = \mathbf{D}(i+2) = \mathbf{C}_0^{i+2} = \frac{1}{6}\mathbf{D}_{i-1} + \frac{2}{3}\mathbf{D}_i + \frac{1}{6}\mathbf{D}_{i+1}, i = 1, \dots, k$



3.6 Interpolation mit B-Splines ...



Gesucht: Kubische, uniforme B-Spline-Kurve D mit:

$$D(3) = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, D(4) = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, D(5) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, D(6) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Lösung: Gleichungssystem mit natürlichen Randbedingungen:

$$\begin{pmatrix} 1 & -2 & 1 & 0 & 0 & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 \\ 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 \\ 0 & 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & 0 & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{pmatrix} \begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \end{pmatrix} = \begin{pmatrix} (0, 0) \\ (-1, 0) \\ (-1, 1) \\ (1, 1) \\ (1, 0) \\ (0, 0) \end{pmatrix}$$

$$\Rightarrow D_0 = \begin{pmatrix} -1/3 \\ -6/5 \end{pmatrix}, D_1 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, D_2 = \begin{pmatrix} -5/3 \\ 6/5 \end{pmatrix},$$

$$D_3 = \begin{pmatrix} 5/3 \\ 6/5 \end{pmatrix}, D_4 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, D_5 = \begin{pmatrix} 1/3 \\ -6/5 \end{pmatrix}$$

