Flächendarstellung

man unterscheidet 2 Typen:

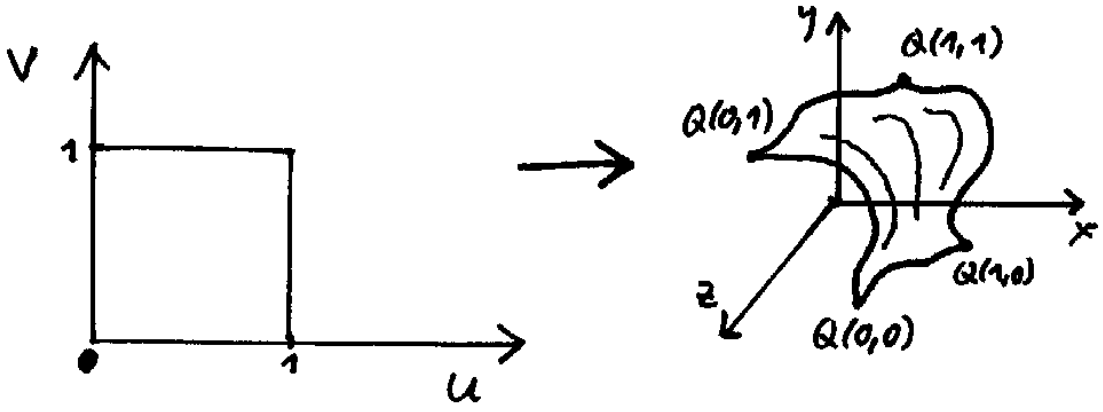
- *finite* Interpolationen / Approximationen: endliche Zahl von Stützstellen / Kontrollpunkten vorgegeben
- *transfinite*: unendliche Menge von Punkten ist vorgegeben, z.B. die Randkurven der zu modellierenden Fläche

  Rand der Fläche soll exakt eingehalten werden: häufige Anforderung, z.B. in der Konstruktion von Bauteilen

1. Transfinite Interpolationsverfahren

Ausgangspunkt:
Parameterdarstellung eines Flächenstücks
mit 2 reellwertigen Parametern *u*, *v*

$u \in [a, b]$, $v \in [c, d]$
Abbildung des Recktecks $[a, b] \times [c, d]$ in den dreidim. Raum:



Parameterdarstellung der Fläche:

$$Q(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix}$$

für festes $u_i \in [a, b]$: $Q(u_i, v)$
für festes $v_j \in [c, d]$: $Q(u, v_j)$ stellen Linien auf der Fläche dar

Randkurven: $Q(a, v)$, $Q(b, v)$, $Q(u, c)$, $Q(u, d)$

Vorgegebene Daten:
- Eckpunkte $Q(a, c)$, $Q(a, d)$, $Q(b, c)$, $Q(b, d)$
- Randkurven
- oft zusätzlich: Richtungsvektoren in den Randpunkten oder auf dem Rand

man definiert:

$$q^{(r,s)}(u_i, v_j) = \frac{\partial^{r+s}}{\partial u^r \partial v^s} Q(u, v)\Big|_{\substack{u=u_i \\ v=v_j}}$$

Oft normiert man auf $a = c = 0$, $b = d = 1$: also Einheitsquadrat als Parameterbereich.

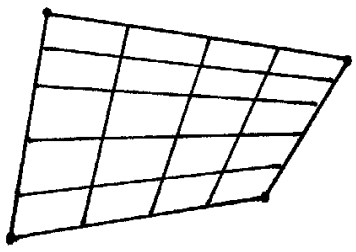
Wie kann man zwischen den 4 Randkurven sinnvoll interpolieren?

Idee: zweimalig linear, in beiden Parameterrichtungen (u und v)

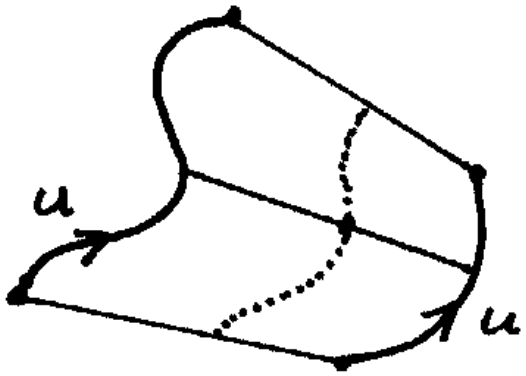
Ausgangspunkt:
bilineare Interpolation zwischen den 4 *Eckpunkten*

$$Q_b(u, v) = Q(0, 0)(1-u)(1-v) + Q(0, 1)(1-u)v + Q(1, 0)u(1-v) + Q(1, 1)uv$$

liefert *bilineares Flächenstück* (i.allg. keine Ebene, sondern parabolisches Hyperboloid = (Ausschnitt aus) Sattelfläche!)



2. Schritt:
Lineare Interpolation zwischen 2 gegenüberliegenden Randkurven: "**Lofting**"



$Q_1(u, v) = Q(u, 0)(1–v) + Q(u, 1)v \qquad (u, v \in [0, 1]).$

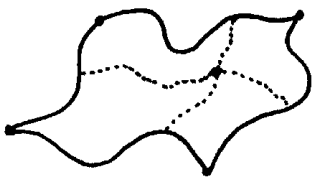
analog für das andere Randkurven-Paar:
$Q_2(u, v) = Q(0, v)(1–u) + Q(1, v)u$

Nachteil: die beiden Interpolations-Flächen enthalten nur je eines der vorgegebenen Randkurven-Paare (das andere Paar ist durch Geradenstücke ersetzt)

3. Schritt: Addition von Q_1 und Q_2; zur Korrektur muss das bilineare Flächenstück Q_b wieder abgezogen werden:

$$Q(u, v) = Q_1(u, v) + Q_2(u, v) – Q_b(u, v)$$

$\Rightarrow Q(u, v)$ hat die 4 vorgegebenen Randkurven



"**Coons-Patch**", "Coons-Pflaster", Coons-Flächenstück.

Um mehr Freiheitsgrade zu erhalten (z.B. für differenzierbare Übergänge zwischen aneinanderhängenden Flächenstücken!):

Verallgemeinerung auf nichtlineare Interpolation

die Faktoren (1–*u*), *u*, (1–*v*), *v* in obiger Formel für *Q*(*u*, *v*) werden ersetzt durch vorgegebene Funktionen: "Überblend-Funktionen", blending functions, Mischfunktionen, Bindefunktionen

$f_{0,0}(u)$ ersetzt (1–*u*)
$f_{0,1}(u)$ ersetzt *u*
und analog für *v*
(man kann auch verschiedene Funktionspaare für *u* und *v* nehmen)
[– der erste Index ($_0$) steht für "nullte Ableitung", vgl. Erweiterung des Ansatzes unten]

Bedingung: $f_{0,0}$ und $f_{0,1}$ sind auf [0, 1] def. und erfüllen

$$\forall\, i,\, j \in \{0;1\} : f_{0,i}(j) = \delta_{i,j} = \begin{cases} 1 \text{ falls } & i = j \\ 0 & \text{sonst} \end{cases}$$

Damit ergibt sich die Def. des *verallgemeinerten Coons-Patches*:

$$Q(u, v) = \sum_{i=0}^{1} Q(i, v)\, f_{0,i}(u) + \sum_{j=0}^{1} Q(u, j)\, f_{0,j}(v)$$

$$- \sum_{i=0}^{1} \sum_{j=0}^{1} Q(i, j)\, f_{0,i}(u)\, f_{0,j}(v) \qquad (u, v \in [0; 1])$$

Beim Aneinanderfügen von Coons-Patches fordert man die *Stetigkeit der 1. Ableitung* (und eventuell auch höherer Ableitungen) [Vermeidung von "Knicken"]

– man braucht weitere Bindefunktionen: $f_{1,0}$, $f_{1,1}$

Bedingungen an die neuen Bindefunktionen:

$$\forall i, j \in \{0;1\} : \frac{d}{du} f_{1,i}(u)\Big|_{u=j} = \delta_{i,j}$$

Definition der verallgemeinerten Coons-Patches mit C^1-stetigem Übergang am Rand:

$$\begin{aligned} Q(u,v) = & \sum_{i=0}^{1} \sum_{r=0}^{1} q^{(r,0)}(i,v) f_{r,i}(u) \\ & + \sum_{j=0}^{1} \sum_{s=0}^{1} q^{(0,s)}(u,j) f_{s,j}(v) \\ & - \sum_{i,j,r,s \in \{0;1\}} q^{(r,s)}(i,j) f_{r,i}(u) f_{s,j}(v) \end{aligned}$$

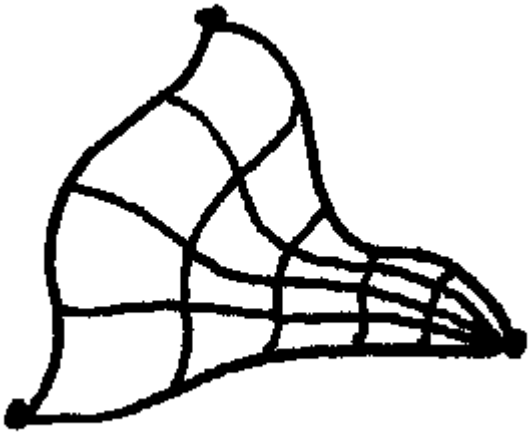
Schwierigkeit bei diesem Modell:

Man benötigt für $q^{(1,1)}$ die gemischten 2. Ableitungen an den Eckpunkten des Patches (die sog. "Twist-Vektoren") – schwierig zu ermitteln, müssen i.allg. geschätzt werden.

Wahl der Bindefunktionen *f*:
oft verwendet: kubische Hermite-Interpolationsfunktionen

$f_{0,0}(u) = H_{0,0}(u) = 2u^3 - 3u^2 + 1$
$f_{0,1}(u) = H_{1,0}(u) = -2u^3 + 3u^2$
$f_{1,0}(u) = H_{0,1}(u) = u^3 - 2u^2 + u$
$f_{1,1}(u) = H_{1,1}(u) = u^3 - u^2$

Coons-Fläche durch 3 Punkte:
man lasse eine der Randkurven zu einem Punkt schrumpfen



"degeneriertes Flächenstück"

– analog für 2 Punkte (2 gegebene Randkurven) oder sogar für nur 1 Punkt (1 gegebene, geschlossene Randkurve) möglich.

2. Finite Verfahren

Oft will man nur wenige Stütz- oder Kontrollpunkte haben
(z.B. beim schnellen interaktiven Editieren von Freiflächen)

Vorgehen analog zur Bézier- oder B-Spline-Modellierung von Kurven

Grundidee:
- ausgehen von Kurvendarstellungen, die wir schon kennen
- "Kurven von Kurven", um auf 2D zu kommen

→ "**Tensorprodukt-Flächen**"

- gegeben eine stückweise polynomiale Kurve $F(u)$ vom Grad n

$$F(u) = \sum_{i=0}^{n} C_i N_i(u) \quad 0 \leq u \leq 1$$

- gegeben eine zweite stückweise polynomiale Kurve $G(v)$ vom Grad m

$$G(v) = \sum_{j=0}^{m} C_j N_j(v) \quad 0 \leq v \leq 1$$

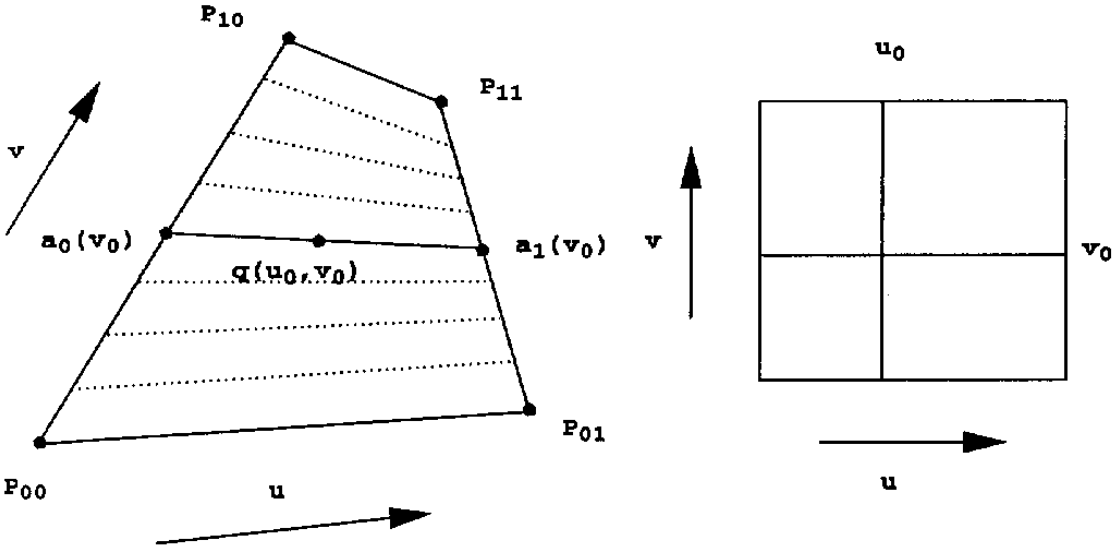
- Tensorproduktfläche $S(u,v)$ beschrieben durch:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} C_{ij} N_i(u) N_j(v) \qquad u,v \in [0,1]$$

- Interpretation als „Kurve von Kurven“
- andere Notation:

$$S(u,v) = \sum_{i=0}^{n} N_i(u) C_i(v) \quad \text{mit} \quad C_i(v) = \sum_{j=0}^{m} N_j(v) C_{ij}$$

Konstruktion analog zur bilinearen Interpolation von Geradenstücken:



– statt der Geraden z.B. kubische Splines oder Bézier-Kurven verwenden
– Kontroll- bzw. Stützpunkte auf den Rändern und innerhalb der Fläche

Beispiel: Tensorproduktfläche, basierend auf kubischen Polynomen (häufig verwendet)

Grundsätzliches zu bikubischen parametrischen Flächen

$$\mathbf{Q}(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} p_{ij} b_i(u) b_j(v)$$

mit $p_{i,j}$ *sind die* 16 *Kontrollpunkte*

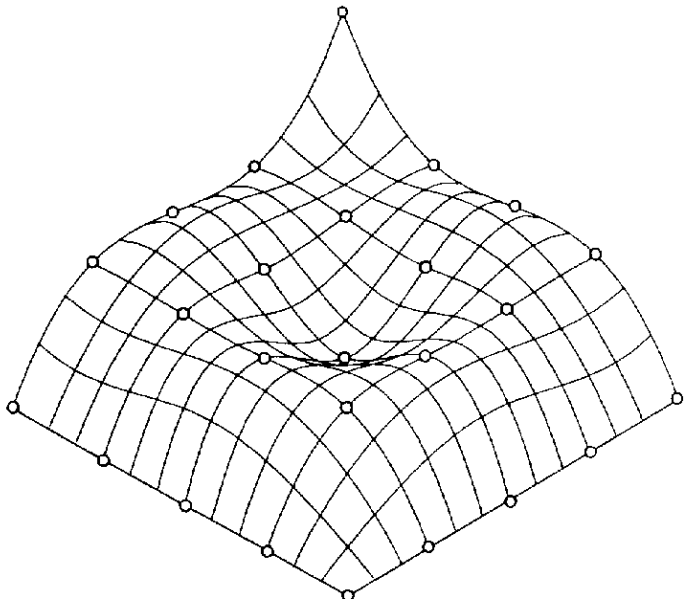
Ein Punkt Q = (x,y,z) der Fläche im kartesischen Raum ist durch (u,v) im Parameterraum bestimmt.

Komplizierte Flächen werden durch eine Menge von solchen Flächenelemente (Patches) zusammengesetzt.

2.1. Bikubische Splineflächen: Interpolation

- analog zu kubischen Spline-Kurven

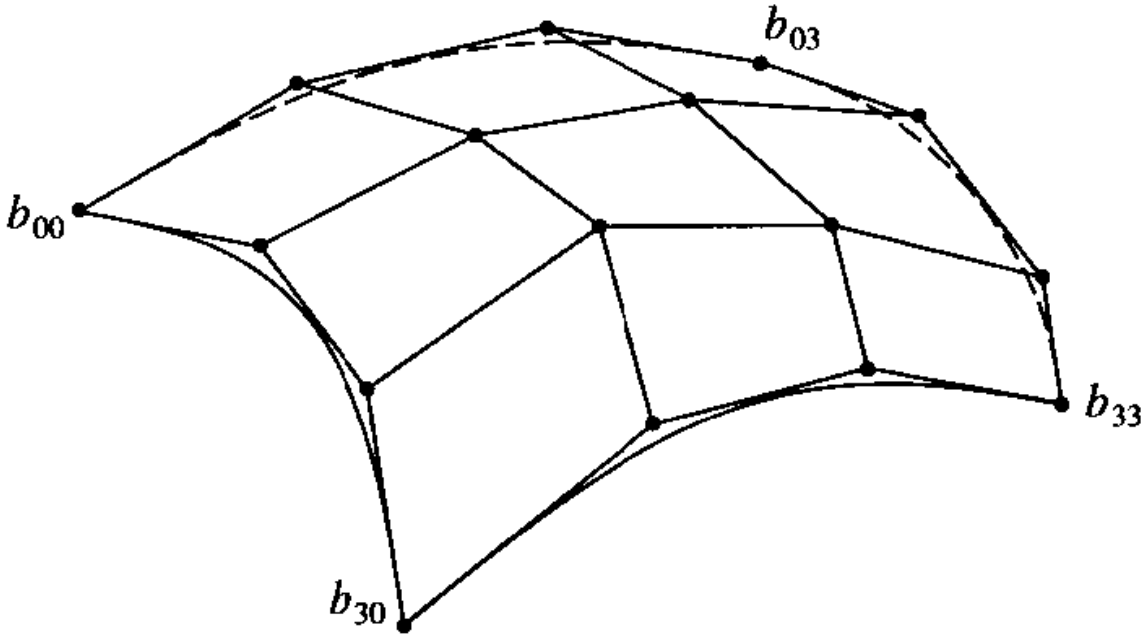
Beispiel (16 einzelne solche Patches, C^1-stetig verschmolzen):



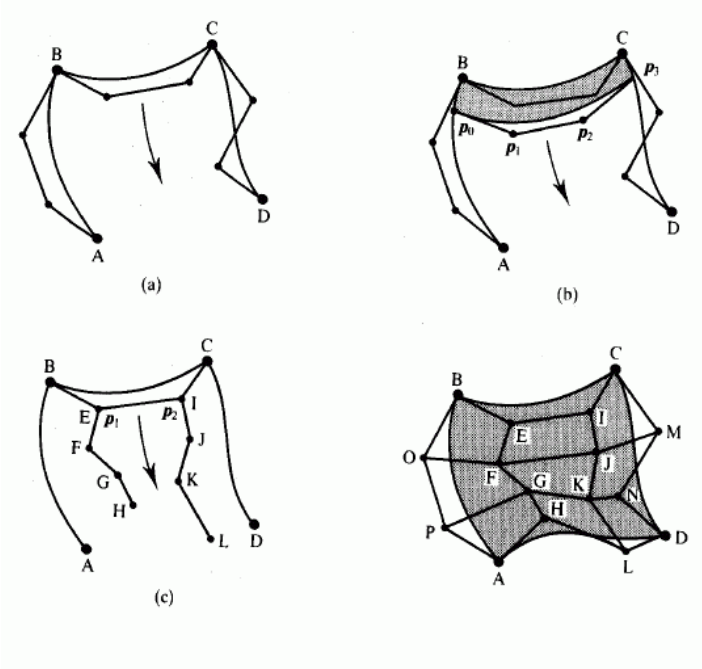
Beachte: die Fläche geht genau durch die vorgegebenen Punkte. Berechnung der Polynom-Koeffizienten i.wesentl. durch Lösen linearer Gleichungssysteme (wie bei Kurven). Siehe Hoschek & Lasser 1992.

2.2. Bikubische Bézierfläche
(Approximation)

16 Kontrollpunkte je Patch, "Kontrollnetz"



Herleitung:



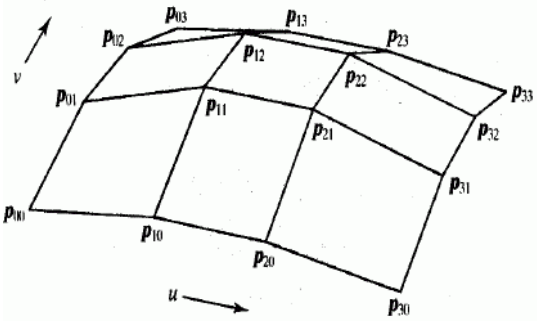
Gegeben: drei Randkurven AB, BC, CD (hier als kubische Bezier-Kurven mit ihren Kontrollpolygonen)

Die Kurve BC wird entlang BA und CD verschoben. Die Form kann sich dabei ändern!

Während des Verschiebens verändern sich die Punkte p_1 und p_2 und erzeugen neue Linien EFGH und IJKL

Es entstehen die 16 Kontrollpunkte (A, B, ..., P) und damit 9 Vierecke, die ein bikubisches parametrisches Bezier-Patch definieren.

rechnerisch:



$$\mathbf{Q}(u,v) = \mathbf{U}\mathbf{M}_B\mathbf{P}_C\mathbf{M}_B^T\mathbf{V}$$

mit

$\mathbf{M}_B$: Bézier Basismatrix

$\mathbf{U},\mathbf{V}$: Reihen – und Zeilenpotenzvektoren

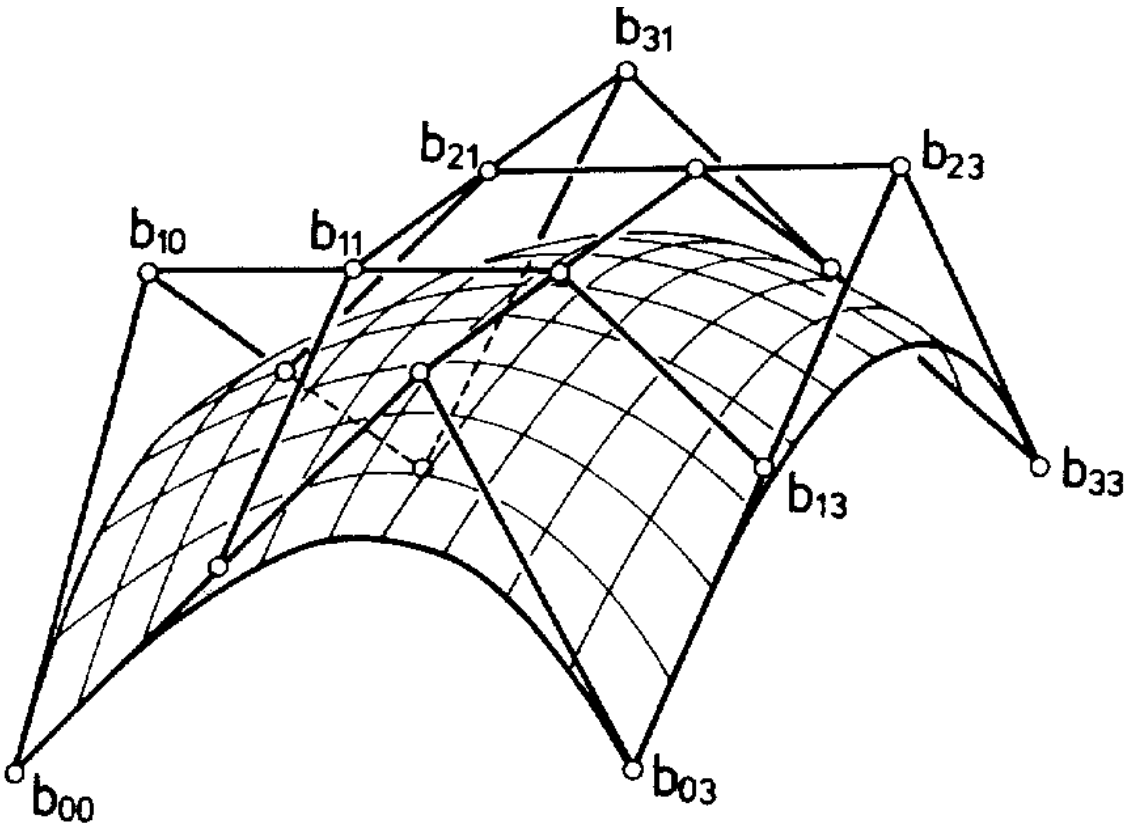
$\mathbf{P}_C$: Matrix der 16 Kontrollpunkte

$$\mathbf{Q}(u,v) = \sum_{i=0}^{k}\sum_{j=0}^{k}\mathbf{p}_{ij}B_{i,n}(u)B_{j,n}(v)$$

$B_{i,n}(u), B_{j,n}(v)$ sind Bézier Basisfunktionen oder *Blending Functions*

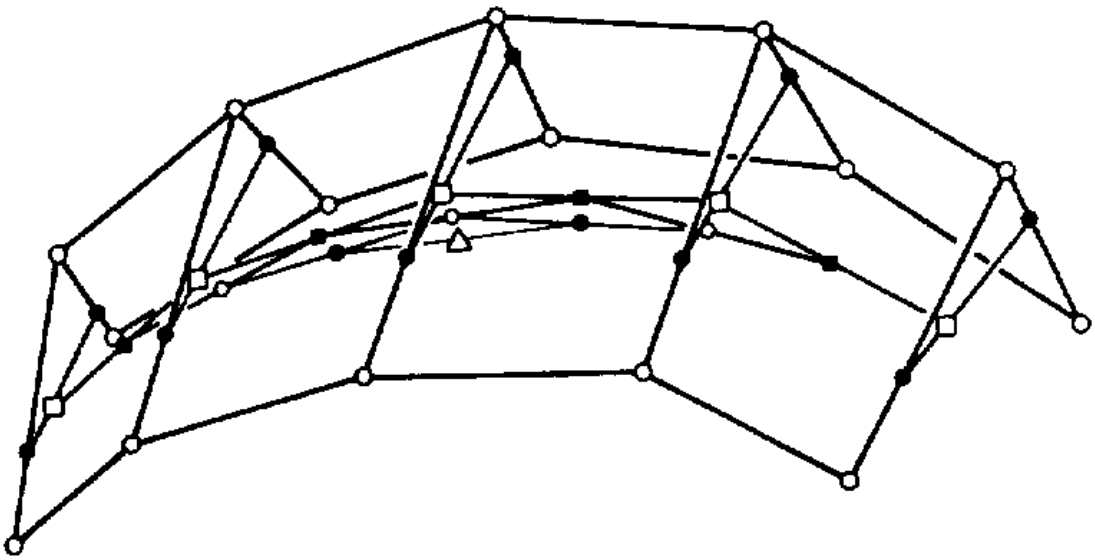
Beachte: Die Randkurven eines Bézier Patches sind Bézier Kurven

Bézier-Fläche mit Kontrollnetz (gut für die interaktive Bearbeitung geeignet):



Konstruktion der Bézier-Fläche:

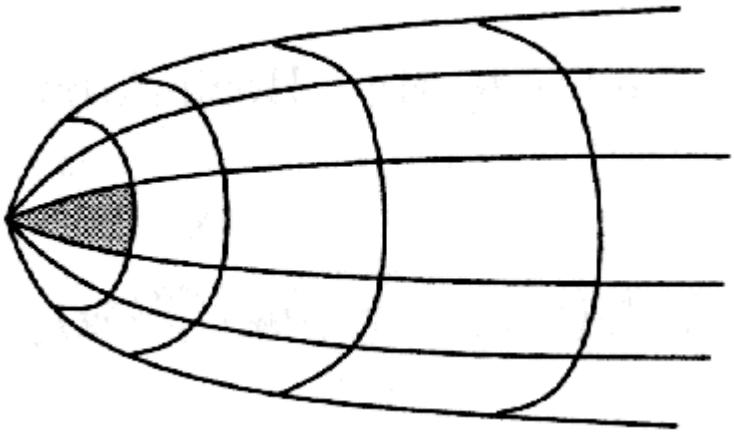
durch iterierte Anwendung des de Casteljau-Algorithmus



(gesuchter Flächenpunkt = kleines Dreieck)

Oft auch gebraucht:
Bézier-Flächen über Dreiecken

z.B. zur Modellierung bestimmter Bereiche in Werkstücken:

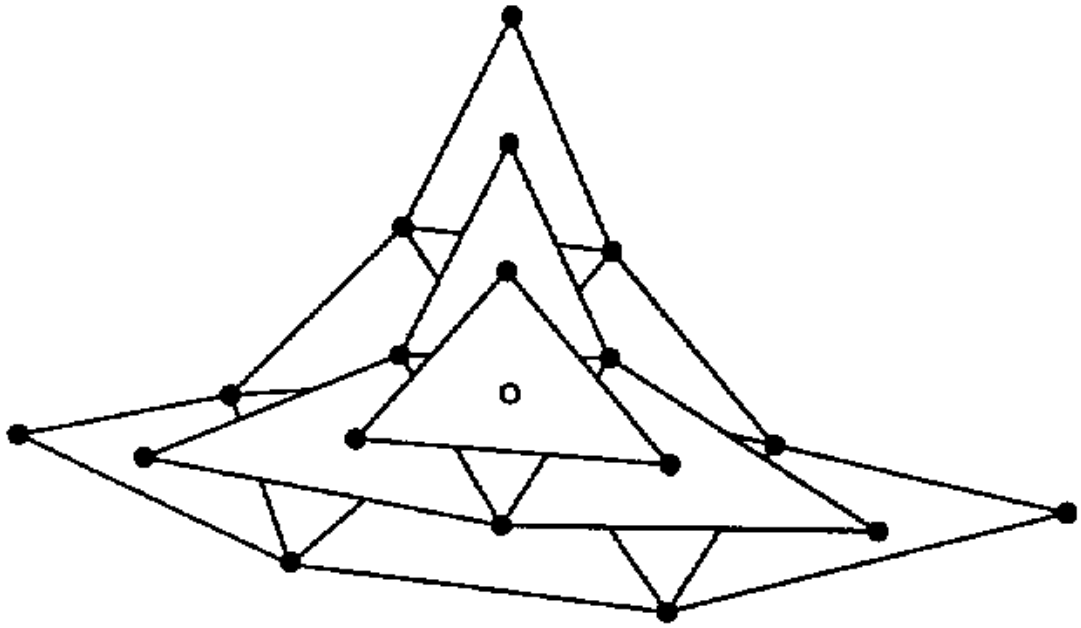


Bézier-Dreiecke können def. werden über verallgemeinerte Bernstein-Polynome:

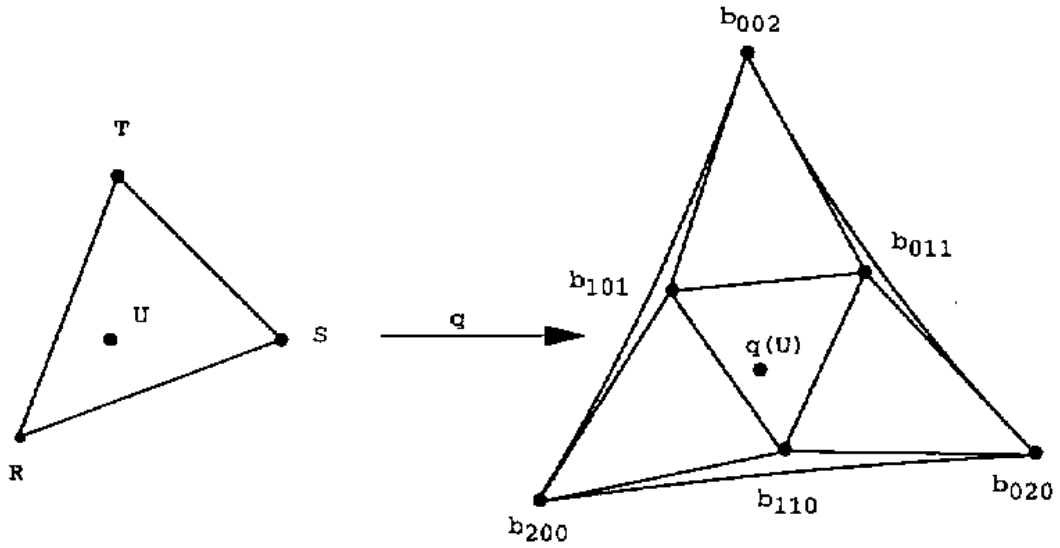
Seien 3 affin unabh. Punkte R, S, T im $\mathbb{R}^2$ gegeben und seien r, s und t die entspr. baryzentrischen Koordinaten eines gegebenen Punktes $U \in \mathbb{R}^2$, d.h.
$U = rR + sS + tT$ mit $r+s+t = 1$.

Dann def. man
$B_{i,j,k}^{n} = n!/(i!\, j!\, k!)\, r^i\, s^j\, t^k$ (für ein Tripel i,j,k mit $i+j+k = n$)
als das verallg. Bernstein-Polynom bzgl. des Referenzdreiecks RST.

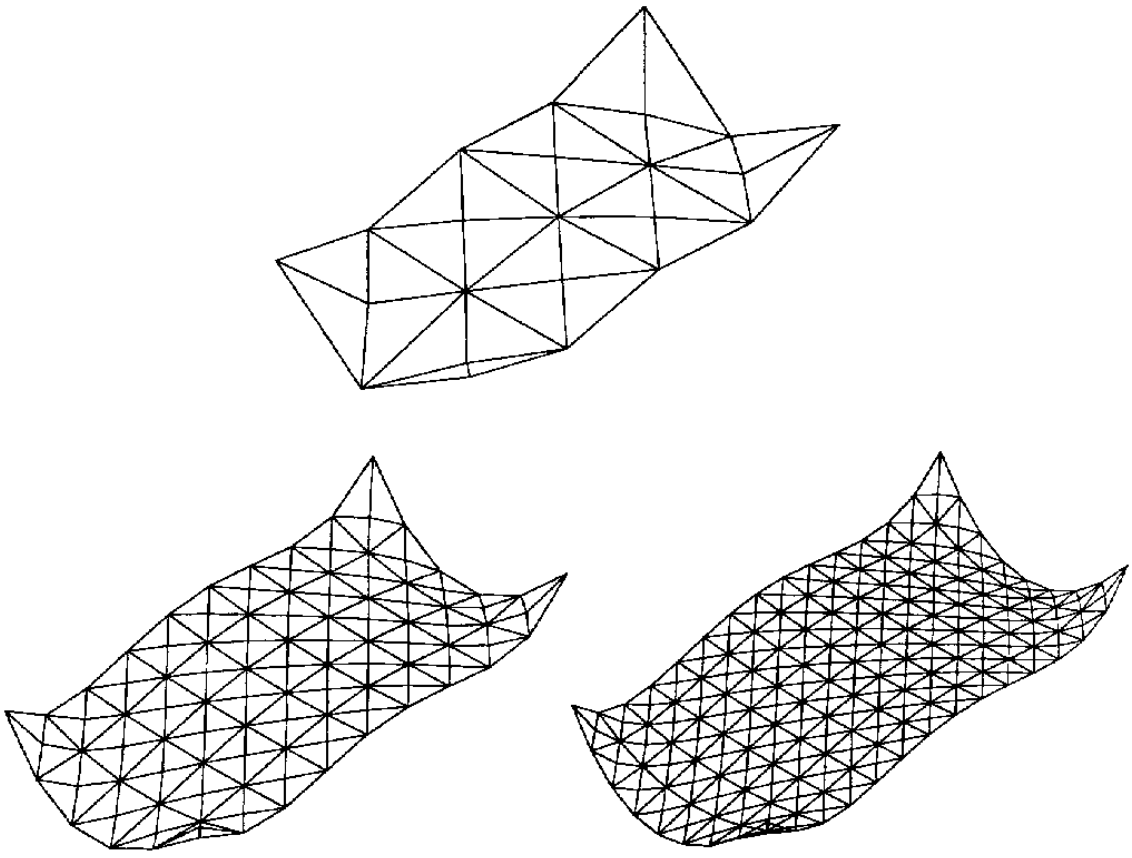
Diese Polynome können als Basis für Bézier-Dreiecke, analog zur Def. der Bézier-Kurven, verwendet werden.
Entspr. Verallgemeinerung der Konstruktionsvorschrift von de Casteljau von Strecken auf Dreiecke:



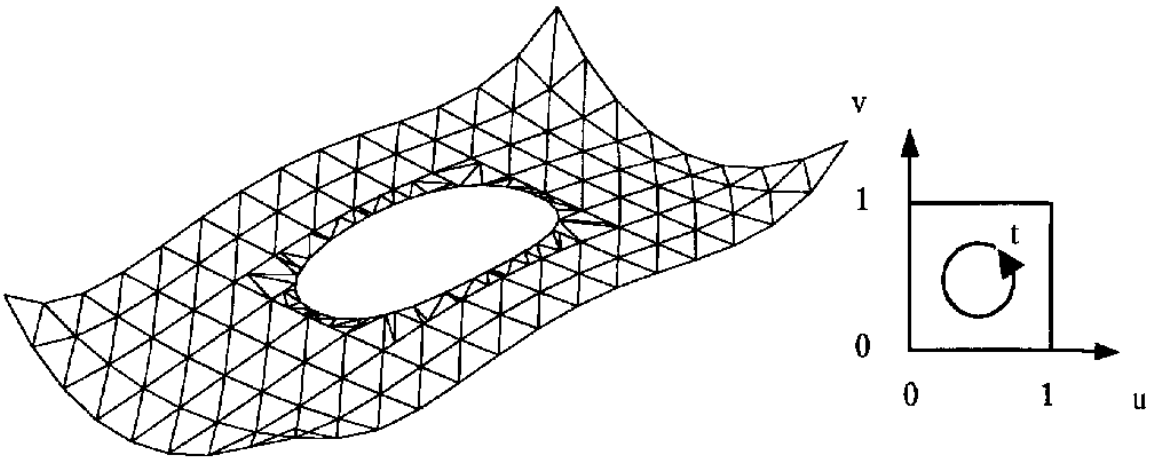
Referenzdreieck und Dreiecks-Bézier-Fläche vom Grad 2 mit Bézier-Netz:



Beispiele für Bézier-Flächen:



zusätzliche Techniken benötigt man, um vorgegebene Ränder und Löcher in Bézier-Flächen einzubeziehen:



Getrimmte Bézier-Fläche (aus Brüderlin & Meier 2001)

Vorgehensweise:

- zunächst Fläche ohne Loch modellieren
- Berandung des Loches durch kleine Geradensegmente approximieren
- Patches der großen Fläche, die ganz im Loch liegen, werden gelöscht
- Patches, die die Trimmkurve schneiden, werden im Inneren der Fläche unter Verwendung der Kurvensegmente neu unterteilt

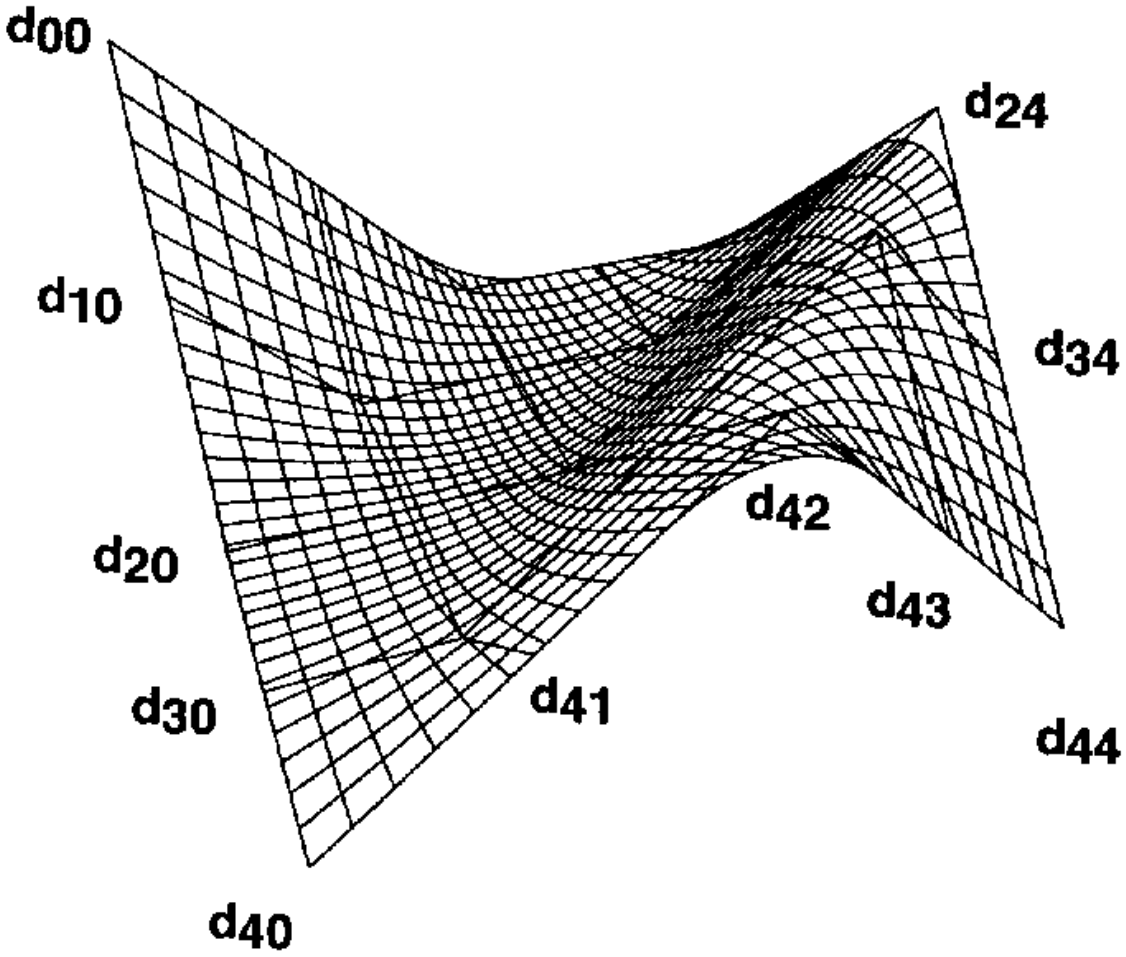
2.3. B-Spline-Flächen

Tensorprodukt-Flächen aus B-Spline-Kurven

$$Q(u,v) = \sum_{i=0}^{r} \sum_{j=0}^{s} P_{i,j} \; N_{i,m}(u) \; N_{j,n}(v)$$

Die Eigenschaften der B-Spline-Kurven übertragen sich, insbes. die konvexe-Hüllen-Eigenschaft.

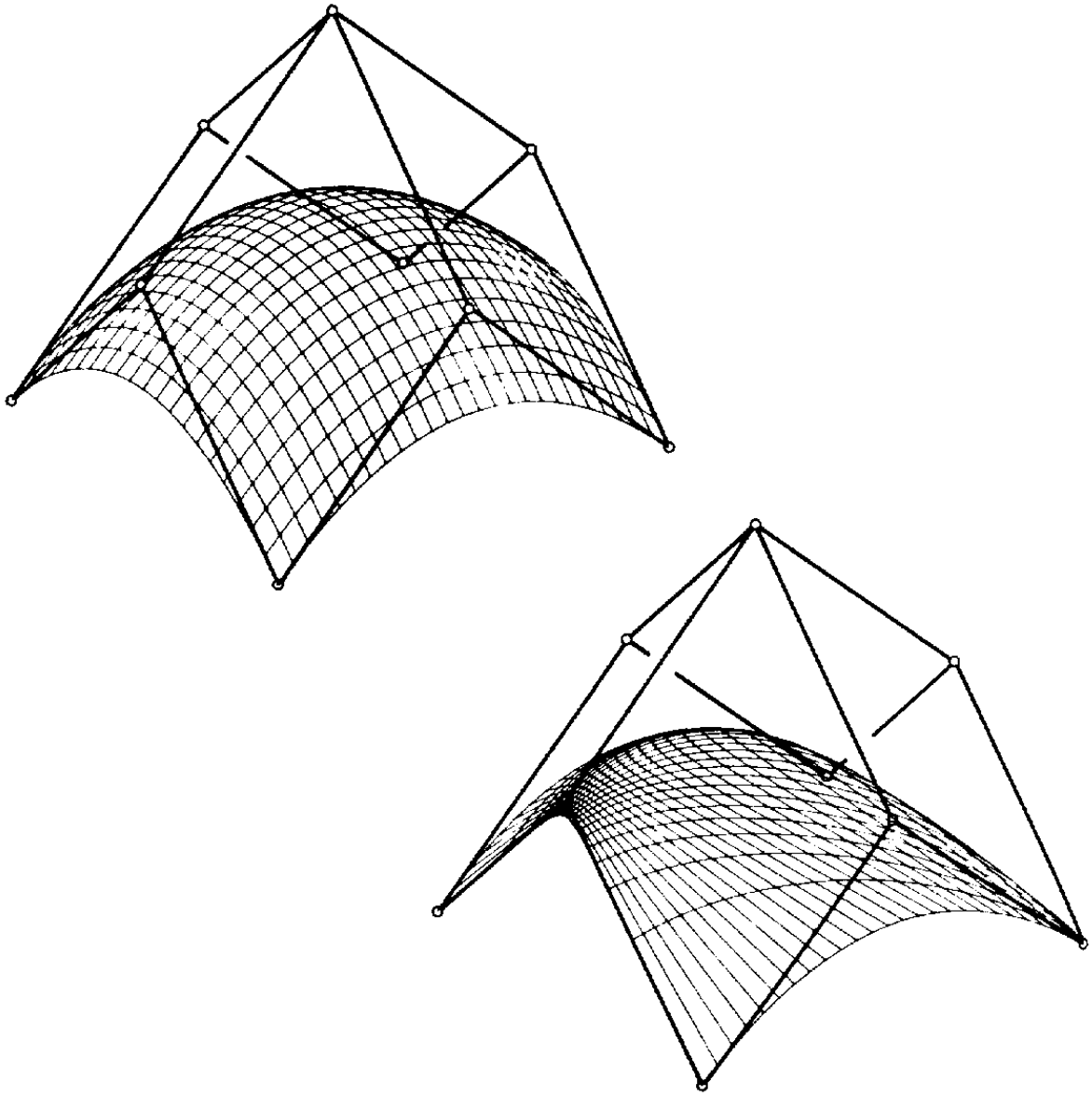
Beispiel einer kubischen B-Spline-Fläche mit Kontrollnetz (25 Kontrollpunkte):



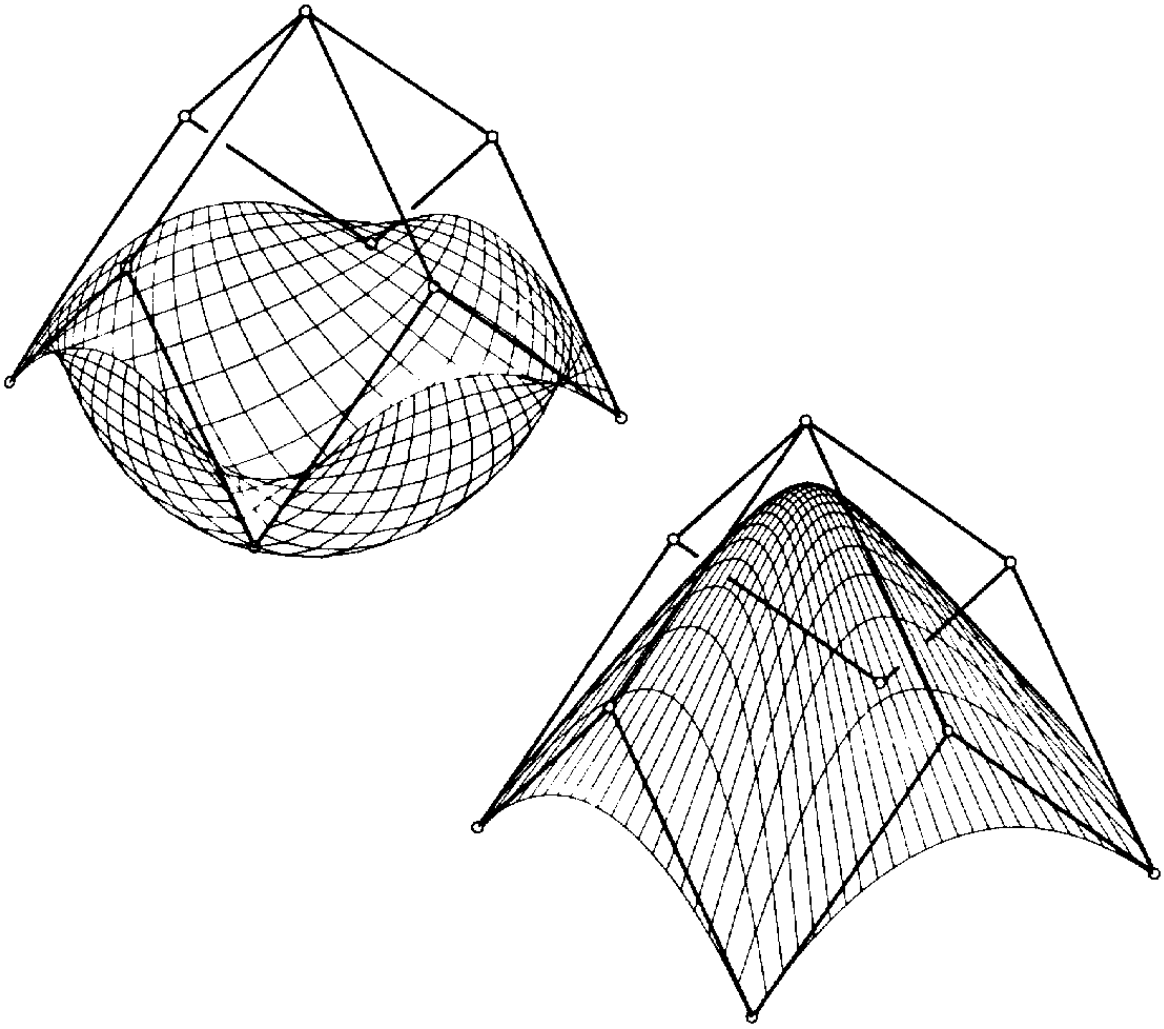
2.4. Rationale B-Spline-Flächen (NURBS-Patches) und rationale Bézier-Flächen

wie im Fall der Kurven: Hinzunahme einer weiteren Funktion in der homogenisierenden, vierten Koordinate, die als "Gewichtsfunktion" weitere Freiheitsgrade eröffnet

Beispiel: rationale Bézier-Flächen mit gleichem Kontrollpunkte-Netz, aber unterschiedlichen Gewichtsfunktionen (aus Hoschek & Lasser 1992):
(1. Bild: alle Gewichte = 1, 2. Bild: linker Rand mit Gewicht 10, rechter Rand mit Gewicht 0,1)



(Fortsetzung nächste Seite)



(oben links: mittlerer Punkt mit Gewicht –1,5,
unten rechts: mittl. Punkt mit Gewicht 10.)

3. Quadriken

sollten bekannt sein aus linearer Algebra...

Quadriken

implizite Oberflächen definiert durch eine Gleichung der Form:

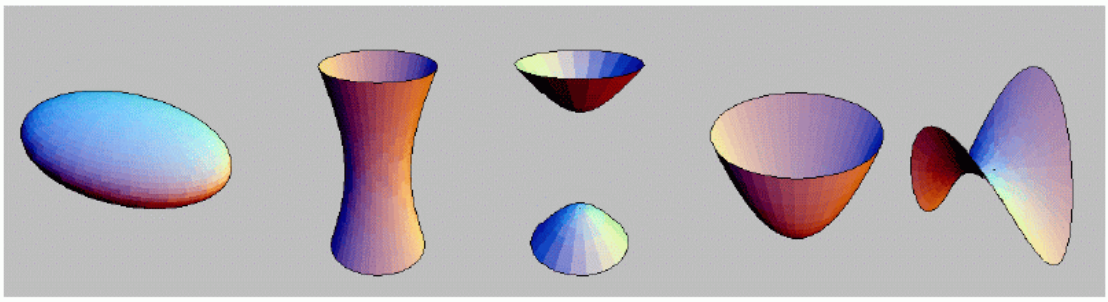
$$f(x,y,z) = ax^2+by^2+cz^2+2dxy+2eyz+2fxz+2gx+2hy+2jz+k = 0$$

alternative Repräsentation:

$$P^T \cdot Q \cdot P = 0 \text{ mit } Q = \begin{pmatrix} a & d & f & g \\ d & b & e & h \\ f & e & c & j \\ g & h & j & k \end{pmatrix} \text{ und } P = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Spezielle Oberflächen durch Auswahl der Parameter:

- Kugel: $x^2 + y^2 + z^2 = 1$
- Kreiskegel: $x^2 + y^2 - z^2 = 0$
- Paraboloid: $x^2 + y^2 - z = 0$
- Hyperbolisches Paraboloid: $x^2 - y^2 - z = 0$



Verwendung

- sehr spezielle Applikationen (Molekülmodellierung)
- aber auch allgemeine Volumenmodeller
- Quadriken sehr gut geeignet für:
  - Berechnung der Oberflächenormale
  - Test, ob ein Punkt auf der Oberfläche liegt (Einsetzen in Geleichung)
  - Wert für z aus x und y berechnen
  - Schnitte zweier Oberflächen berechnen

- Durch Quadriken repräsentierte Oberflächen lassen sich leicht transformieren:
  - gegeben eine Quadrik durch die Matrix Q und eine Transformationsmatrix M
  - Berechnung der transformierten Oberfläche durch

$$Q' = (M^{-1})^T \cdot Q \cdot M^{-1}$$

- Berechnung der Normale an eine durch eine Quadrik $f(x, y, z) = 0$ repräsentierte Oberfläche:

$$n = \left(\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz} \right)^T$$