Technische Universität Dresden im WS 2004/05

Fakultät Informatik

Institut für Institut für Software- und Multimediatechnik

Proseminar:Computergrafik

Dozent:Dr. Mascolous

Referent: Patrick Brausewetter

Thema des Referats: Darstellung von Kurven und Flächen

Patrick Brausewetter

Tanneberger Weg 2

01169 Dresden

0351 / 2168997

pepe-croco@gmx.de

Diplomstudiengang

Informatik

Matr.-Nr.: 3172279

4. Semester; SS 2006

Informatik

Datum: 16.05.2006

Kurven und Flächen

Einleitung

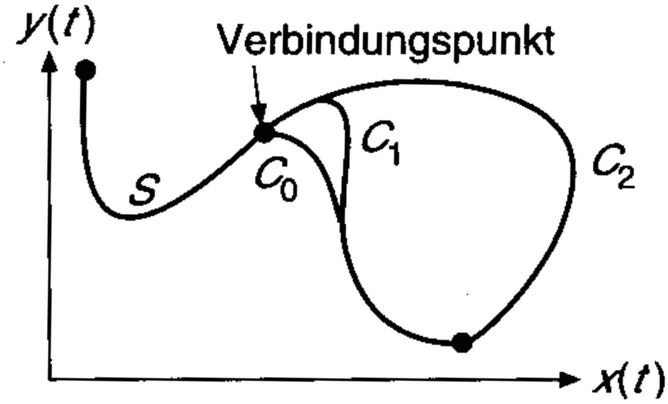
Wenn man eine Kurve im Computer darstellen will, kann man dies auf verschiedene Weisen tun: Man kann versuchen, die Kurve durch eine sehr große Menge von Kontrollpunkten darstellen, welche dann linear verbunden werden. Dies ist aber erstens recht ineffizient und zweitens entstehen unschöne Effekte bei der Vergrößerung einer Kurve, da man dann ganz deutlich die geraden Strecken sehen kann. Deswegen versucht man häufig, eine Kurve durch eine mathematische Beschreibung zu formulieren. Wenn die Berechnungsvorschrift klar definiert ist, muss man nur noch eine Menge an Punkten (die aber viel kleiner ist als die reale Punktmenge) speichern und kann daraus jederzeit (und in jeder Größe) beliebig genau diese Kurve darstellen. Diese Kurven werden z.B. bei der Darstellung von Fonts (Schriftarten) verwendet. (TrueType-Schriften: quadratische B-Splines, Postscript-Schriften: kubische Beziérkurven)

---

1. Kurven

1.1 Verbindung von zwei Kurven

Das Problem, welches auftritt, wenn man eine Kurve durch 2 Teilkurven niedrigeren Grades beschreiben will, ist die Frage nach der Art der Verbindung.
Die beiden Kurven müssen einen gemeinsamen Punkt besitzen (Ende der ersten und Anfang der zweiten Kurve). Dies nennt man G^0-Stetigkeit oder C^0-Stetigkeit. Dies allein kann sehr unschön aussehen (siehe Bild).



Haben die Tangentenvektoren der Kurven in diesem Schnittpunkt die gleiche Richtung, nennt man dies G^1-Stetigkeit.
Haben die Tangentenvektoren auch noch den gleichen Betrag (sind sie identisch), nennt man dies C^1-stetig, weil sie in der 1. Ableitung übereinstimmen. Damit sind sie automatisch G^1-stetig.
C^n-stetig bedeutet, dass sie bis zur n. Ableitung übereinstimmen.
Die erste Ableitung gibt quasi die Geschwindigkeit eines Punktes an, die zweite die Beschleunigung.
G^1-Stetigkeit ist eine Voraussetzung für eine glatt erscheinende Kurve, der Rest ist quasi Bonus.

1.2 Ausgewählte Kurven

Kurven sind allgemein Polynome, wobei die drei Kurven für die Achsen x, y, z jeweils durch ein Polynom n.ten Grades in Abhängigkeit von t beschrieben werden.
Kubische Kurven sind von der allgemeinen Form $x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$
Da man 4 Koeffizienten hat, braucht man 4 Bedingungen, um die Kurve zu charakterisieren.

1.2.1 Beziér-Kurven

Die Bézierkurve wurde Anfang der 1960er Jahre unabhängig voneinander von Pierre Bézier bei Renault und Paul de Casteljau (1910-1999) bei Citroën für den Automobilentwurf entwickelt und ist ein wichtiges Werkzeug im CAD.

Aussehen:
Ein Band liegt zwischen den beiden Endpunkten, und die Kontrollpunkte ziehen das Band zu sich hin mit einer Stärke, die sich proportional zum Abstand des Kontrollpunktes zur Kurve verhält. (siehe Programm –> Bézierkurve n.ten Grades)

Allgemeine Form:

Eine *n*-dimensionale Bézierkurve ist eine Kurve der Form $$C(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \quad ,$$

mit den Kontrollpunkten P_i und den Bernsteinpolynomen $$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

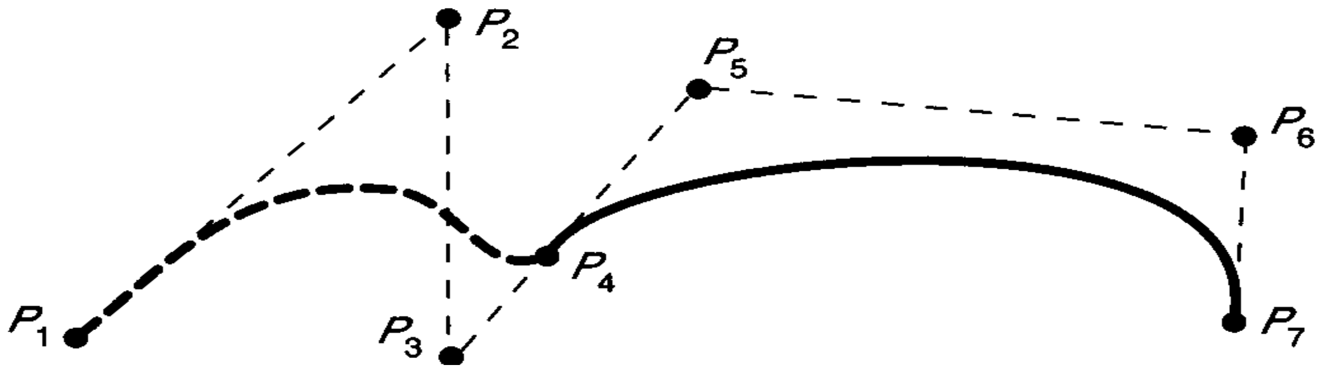
wobei $0 \leq t \leq 1$ den Wertebereich festlegt.

1.2.1.1 Kubische Bezierkurven

Gleichung: $C(t) = P_0 (1-t)^3 + 3 P_1 t (1-t)^2 + 3 P_2 t^2 (1-t) + P_3 t^3$

Bedingungen:
2 Endpunkte (P_1, P_4) und 2 Zwischenpunkte (P_2, P_3), wobei P_1P_2 und P_3P_4 die Tangenvektoren implizit angeben.

Eigenschaften:
- C_0-Stetigkeit ist erreichbar durch gemeinsamen Kontrollpunkt.
- C_1-Stetigkeit erreicht man dadurch, dass $P_4 = P_5$ (letzter Punkt 1. Kurve = 1. Punkt 2. Kurve) und die Strecke P_4P_6 den Mittelpunkt P_5 hat (der Tangentenvektor beider Kurven im Punkt 4 ist gleich –> siehe Bild.

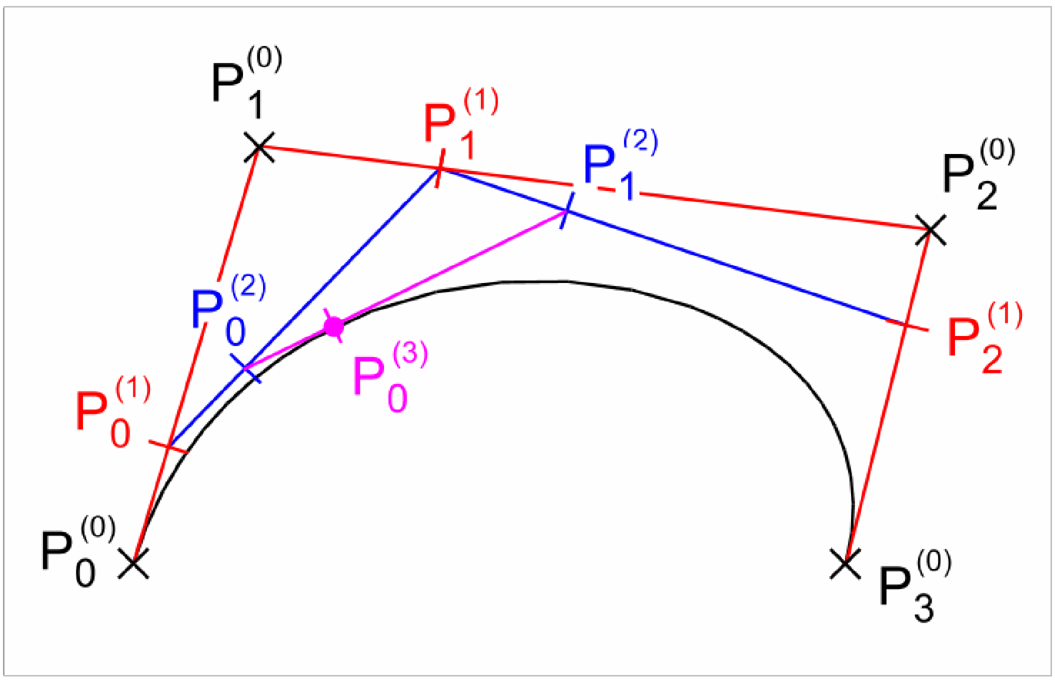


- Die Kurve durchläuft die beiden Endpunkte.
- Affine Transformationen (Verschiebung, Skalierung, Rotation, Scherung) können auf die Bézierkurve durch Transformation der Kontrollpunkte angewendet werden ("affine Invarianz").
- Sie ist als gewichtete Summe polynomialer Basisfunktionen darstellbar.
- Die Summe der Basisfunktionen hat an einer bestimmten Stelle den Wert 1.
- => Die Kurve liegt in der konvexen Hülle der Kontrollpunkte.

Darstellung (Casteljau-Algorithmus)

Um eine Bézierkurve darzustellen, kann man natürlich einfach die Kurve berechnen für bestimmte Werte von t, aber dies dauert länger als der Casteljau-Algorithmus (–> siehe Programm).
Mit dem Casteljau-Algorithmus kann man relativ einfach einen Punkt auf der Kurve bestimmen. Man kann dann diese Punkte zu einem Polygonzug verbinden, um eine Näherungslösung für die Kurve zu erhalten, oder aber man ermittelt so einfach nur die Funktionswerte zur „normalen“ Zeichnung.

Arbeitsweise (speziell für kubische):
Man stellt sich vor, dass die 4 Kontrollpunkte durch Geraden verbunden sind, wobei ein Punkt P^0_i immer mit seinem Nachfolger P^0_{i+1} verbunden ist (i = 0, 1, 2). Man teilt nun diese 4 Strecken im Verhältnis t : (1-t) (t ϵ [0, 1]) und erhält damit 3 neue Punkte P^1_i (i = 0, 1, 2). Teilt man diese wiederum im gleichen Verhältnis, erhält man wieder 2 Punkte P^2_i (i = 0,1). Diese Strecke teilt man noch mal und erhält P^3_i, der genau der Funktionswert von der Kurve an der Stelle t ist. Also muss man nur Grundrechenarten ausführen und keine Fakultäten bzw. Potenzen berechnen.



1.2.2 Hermite-Kurven

Sie sind nach dem Mathematiker Hermite benannt.

Bedingungen (kubisch):
2 Endpunkte (Kurve läuft durch sie hindurch) und 2 Tangenvektoren in diesen Punkten.

Eigenschaften
siehe Bezier-Kurven

1.2.3 Splines

Ein Spline ist ein flexibler Metallstreifen, mit denen technische Zeichner die Flächen von Flugzeugen, Autos und Schiffen konstruiert haben. An den Enden der Streifen werden kleine Gewichte angebracht, um ihnen eine Krümmung zu geben. Solange sie nicht extrem belastet werden, sind sie C^2-stetig.
Das mathematische Äquivalent sind natürliche kubische Splines. Sie interpolieren (durchlaufen) alle Kontrollpunkte. Die Koeffizienten der Kurven hängen von allen n Kontrollpunkten ab. Für ihre Berechnung muss eine (n+1) * (n+1) Matrix invertiert werden, was mit großem Aufwand verbunden ist. Außerdem hat die Änderung eines Kontrollpunktes Auswirkungen auf die gesamte Kurve, was oft unerwünscht ist.

1.2.3.1 Kubische B-Splines

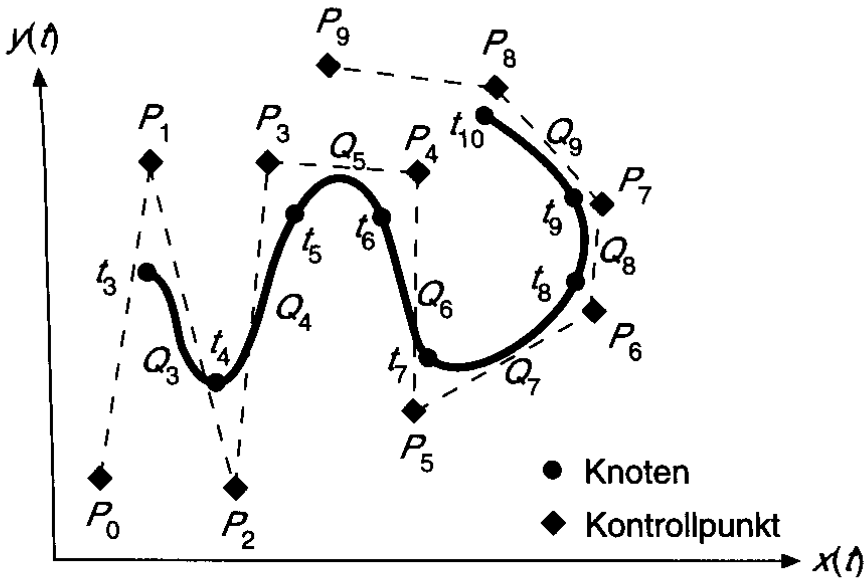
Das B steht für Basis, da man die B-Splines als gewichtete Summe polynomialer Basisfunktionen darstellen kann (wie auch Bézierkurven). Die Summe der Basisfunktionen hat an jeder Stelle den Wert 1. Dadurch liegt sie in der konvexen Hülle der Kontrollpunkte. Die relativ einfache Berechnung und die lokale Steuerung machen die B-Spline-Kurven und -Flächen zu den am weitest verbreiteten Kurven zur Beschreibung von gekrümmten Flächen.

Gleichung:

$$Q(t) = \frac{(1-t)^3}{6} P_0 + \frac{3t^3 - 6t^2 + 4}{6} P_1 + \frac{-3t^3 + 3t^2 + 3t + 1}{6} P_2 + \frac{t^3}{6} P3$$

(im Grunde wie Bézierkurven, nur statt Bernsteinpolynomen B-Spline-Funktionen)
(siehe Bild)

Eigenschaften:
Sie interpolieren die Kontrollpunkte nicht, aber dafür hängen die Koeffizienten nur von einigen Kontrollpunkten ab, wodurch man gewisse lokale Änderungen durchführen kann. Außerdem kann man die Koeffizienten wesentlich schneller berechnen.



Stetigkeiten entstehen durch gemeinsame Nutzung von Kontrollpunkten durch mehrere Segmente. Ein Kontrollpunkt hat Einfluss auf 4 Kurvensegmente. Bewegt man ihn in eine bestimmte Richtung, bewegen sich die 4 Kurvensegmente in die gleiche Richtung (lokale Steuerung).
Kubische Splines approximieren n Kontrollpunkte, wobei $n > 3$. Sie bestehen aus n-3 kubischen polynomialen Kurvensegmenten, die alle über einem Definitionsbereich $0 <= t <= 1$ definiert sind. Man kann durch eine Substitution $t = t+k$ einen gemeinsamen, stetigen Definitionsbereich für den gesamten Spline erhalten.
Eine geschlossene Kurve erhält man, indem man am Ende der Folge von Kontrollpunkten noch mal die ersten 3 Kontrollpunkte wiederholt.
Eine B-Spline-Kuve verläuft üblicherweise nahe an den Mittelpunkten der Strecken zwischen benachbarten Kontrollpunkten.

Uniforme Splines:
Uniforme Splines liegen dann vor, wenn die Knoten (die Verbindungspunkte der einzelnen Kurven) auf gleich großen Intervallen des Parameters t angeordnet sind (gleiche Abstände zwischen den Knoten)

Interpolation:
Wenn Kontrollpunkte wiederholt verwendet werden (direkt hintereinander), kann die Interpolation bestimmter Punkte erzwungen werden. Allerdings entstehen so oft Geradenstücke oder Spitzen.
Ein anderes Verfahren verwendet Phantompunkte, wobei 2 neue Kontrollpunkte eingeführt werden. $P_{n+1} = 2P_n - P_{n-1}$ wird hinten eingefügt und $P_{-1} = 2P_0 - P_1$ wird vor dem ersten eingefügt. Damit durchläuft die Kurve die beiden Endpunkte.

Darstellung (Algorithmus von De-Boor):
Äquivalent zum Casteljau-Algorithmus. Bei einem B-Spline 3. Grades werden 2 Iterationen benötigt, wobei in der ersten Iteration 2 neue Punkte berechnet werden und in der zweiten Iteration ein neuer Punkt, welcher der gesuchte Punkt auf der Kurve ist.
Wenn man gewisse Parameter geeignet setzt, sieht man, dass eine Bézier-Kurve ist ein Spezialfall eines B-Splines ist.

1.2.3.2 NURBS

Non uniform rational B-Splines sind nichtuniforme B-Splines. Nicht-uniform bedeutet, dass die Knoten auf möglicherweise unterschiedlich großen Intervallen des Parameters t angeordnet sind. Rational bedeutet, dass die Kurve als Quotient von 2 B-Spline-Polynomen schreibbar ist. Durch die Nichtuniformität ist auch mehrfache Nutzung von Kontrollpunkten zulässig, wodurch zwar die Stetigkeit gesenkt wird, aber die Interpolation von Punkten ermöglicht wird. Ein Knoten darf allerdings nicht öfter als n (n ist Grad des Polynoms) auftreten, da sonst Unterbrechungen der Kurven auftreten.

Eigenschaften:

- Invariant bei Rotation, Skalierung, Translation und Zentralprojektion (nichtrationale Kurven sind bei Zentralprojektion nicht invariant); d.h. man muss die Transformation nur auf die Kontrollpunkte anwenden, um das gleiche Ergebnis zu erhalten, wie wenn man die Transformation auf jeden Punkt anwenden würde
- Man kann Bézierkuven sowie rationale B-Splines mit ihnen darstellen, indem man z.B. den Divisor 1 setzt –> nichtuniformer rationaler B-Spline; durch die Beschränkung auf gleiche Parameterintervalle erhält man uniforme Kurven und Bézierkurven lassen sich als Spezialfall von B-Splines auffassen.
- Man kann exakte Kegelschnitte sowie Kreise definieren, die häufig in CAD-Anwendungen benötigt werden.

______ ______ ______ ______ ______ ______

1.3 Interpolation

Man möchte zu einer gegebenen Menge von diskreten Punkten (z.B. Messwerten) eine stetige Funktion erhalten, die durch alle diese Punkte verläuft, also diese interpoliert.

1.3.1 Polynominterpolation

Der Fundamentalsatz der Algebra garantiert, dass man zu n + 1 Datenpunkten genau ein Interpolationspolynom n-ten Grades finden kann. Die Bestimmung der Koeffizienten erfordert die Lösung eines linearen Gleichungssystems. Man erhält das Interpolationspolynom z.B. mit Hilfe der Formel von Lagrange:

$$p(x) = \sum_{i=0}^{n} f_i \prod_{k=0, k\neq i}^{n} \frac{x - x_k}{x_i - x_k}$$

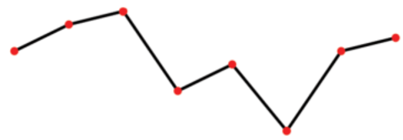
Allerdings haben Polynome den Nachteil, dass sie viele Extrema haben und deswegen bei hohem Polynomgrad recht stark zwischen den zu interpolierenden Punkten schwingen. Deswegen werden kaum Polynome mit Grad > 5 eingesetzt. Stattdessen interpoliert man einen großen Datensatz *stückweise*. Bei Polynomen vom Grad 2 oder 3 spricht man üblicherweise von Spline-Interpolation. Bei abschnittsweise definierten Interpolanten ist die Frage der Stetigkeit und Differenzierbarkeit an den Stützstellen von großer Bedeutung (Siehe Kapitel 2).

1.3.2 B-Spline Interpolation

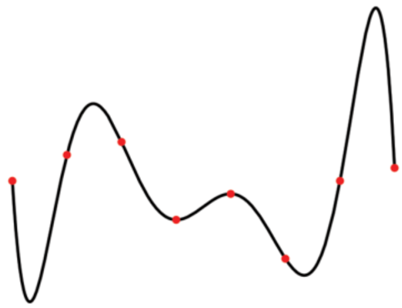
Um die Kontrollpunkte b_0 bis b_n zu interpolieren, braucht man n+1 neue Kontrollpunkte, die sich aus einem Gleichungssystem herleiten lassen. Das Gleichungssystem kann man als $Ax = b$ schreiben, womit $x = A^{-1}b$. Da A nicht von den Kontrollpunkten abhängt, lässt sich A^{-1} vorberechnen.

Soll z.B. diese Menge von Punkten interpoliert werden,

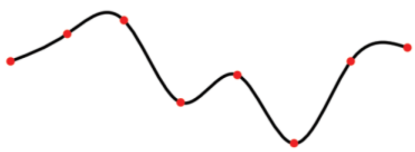
so erhält man mit stückweise linearer Interpolation eine Menge von Geradenstücken:



Eine Polynominterpolation 7.Grades erzeugt diese Kurve:

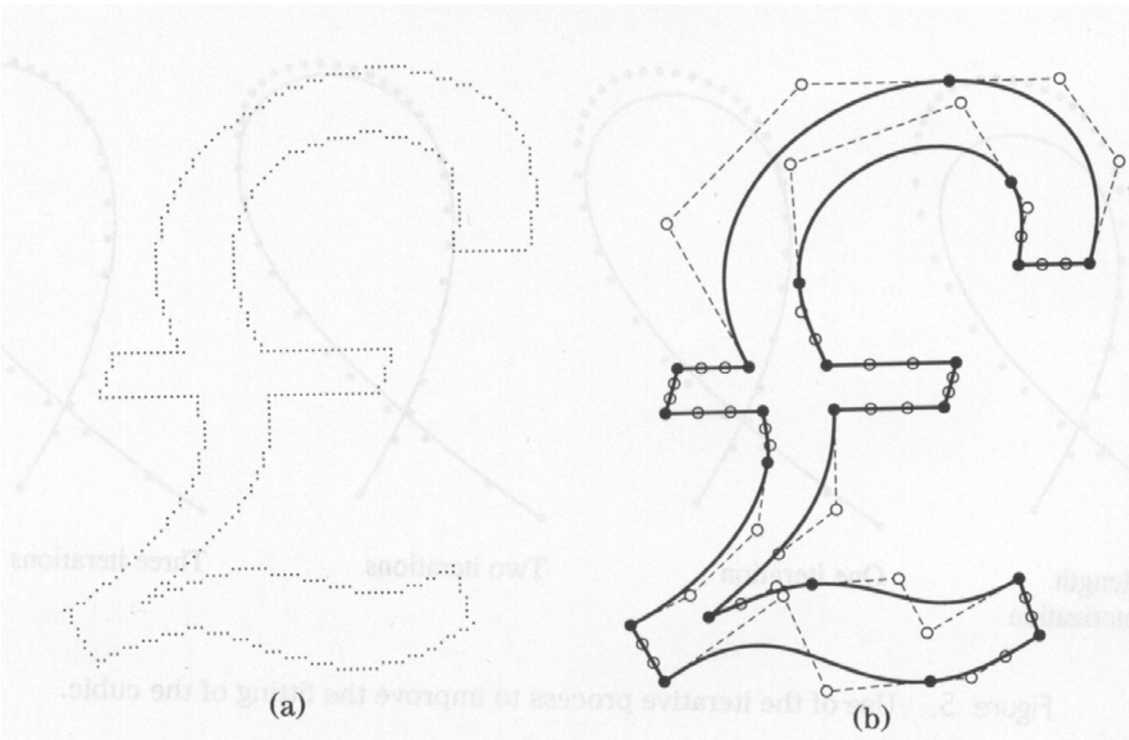


Eine kubische Spline-Interpolation diese:



1.3.3 Bézier Interpolation

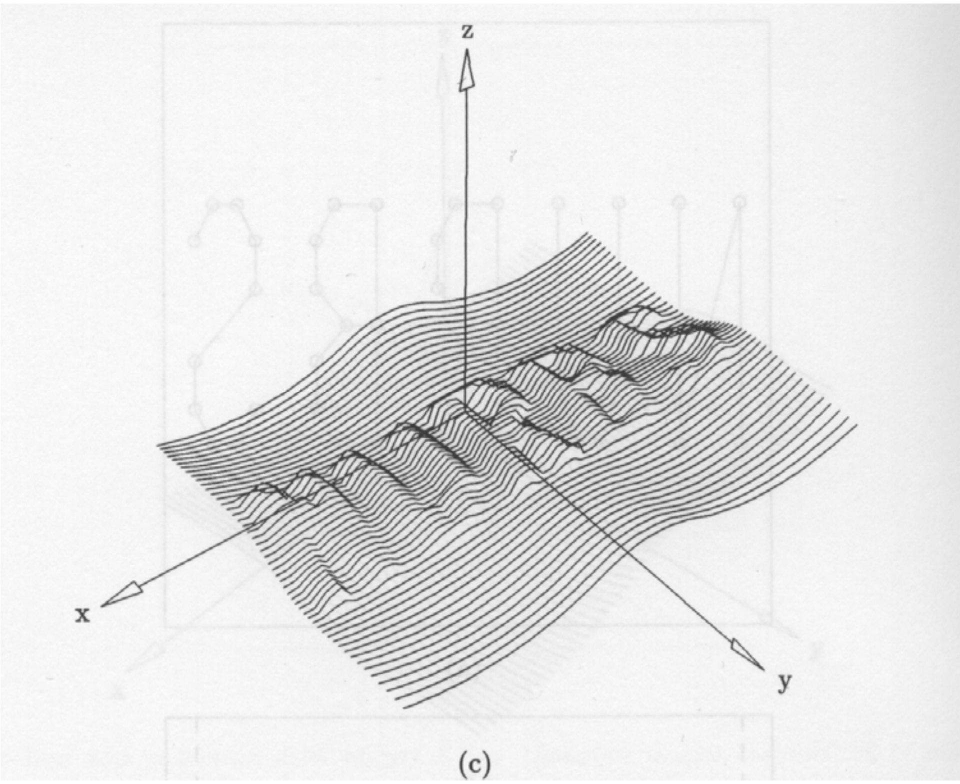
Philip J. Schneider hat einen Algorithmus entwickelt, der eine gegebene Menge von Punkten mit Hilfe von zusammengesetzten Bézierkurven interpoliert. Als Beispiel wurde ein Pfund-Zeichen digitalisiert und daraus eine Folge von Bézierkurven gemacht. (siehe Bild)



2.Flächen

Die allgemeine Formel für eine parametrisierte kubische Kurve ist Q(t) = G * M * T.
G ist Geometriematrix (Bézier: Kontrollpunkte). G = [G_1 G_2 G_3 G_4].
M ist Basis-Matrix (Koeffizienten des Polynoms).
T ist Parameter-Matrix (T = [t^3 t^2 t 1]).
Wenn nun aber die Geometriematrix nicht konstant ist (also die Kontrollpunkte selber eine Funktion in Abhängigkeit von s sind), ergibt sich: Q(t,s) = [G_1(s) G_2(s) G_3(s) G_4(s)] * M * T.
Wenn alle G_i(s) selber kubische Polynome sind, spricht man von einer kubischen Fläche. Für ihre Beschreibung braucht man nun 16 Unbekannte (4 pro Polynom bei 4 Polynomen).

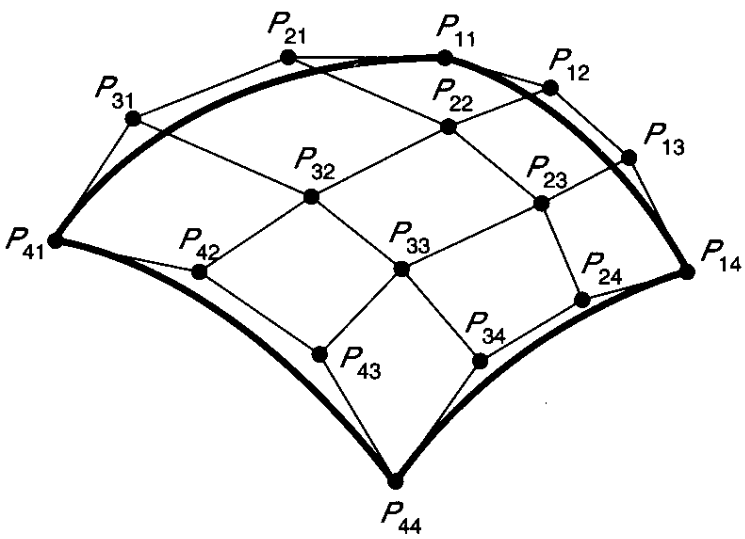
<u>Darstellung:</u>
Üblicherweise wählt man einige Werte für konstantes s und einige für konstantes t und berechnet mit ihnen die entstehenden Kurven. Diese verbindet man dann zu einer Fläche. Aufgrund des hohen Berechnungsaufwands wurden andere Methoden wie die Vorwärtsdifferenziation entwickelt.



(c)

2.1 Bézierflächen

Zur Beschreibung einer Bézierfläche benötigt man 16 Kontrollpunkte. Um C0-Stetigkeit zwischen 2 Flächen zu erhalten, müssen sie 4 Kontrollpunkte gemeinsam haben (also eine Kante). G1-Stetigkeit erhält man, indem die jeweils 4 Punkte neben den „Kantenpunkten“ auf einer Geraden liegen.



Quellen- und Literaturverzeichnis

- Foley, James (u.a.); Grundlagen der Computergraphik, Einführung, Methoden Konzepte; Bonn, 1994; ISBN 0-201-60921-5
- Glassner, Andrew (u.a.); Graphics Gems I; San Diego, 1990; ISBN 0-12-2861666-3
- Piegl, Les und Tiller, Wayne; The NURBS BOOK; Heidelberg, 1995; ISBN 3-540-61545-8
- Rauber, Thomas; Algorithmen in der Computergraphik; Stuttgart, 1993; ISBN 3-519-02127-7
- Wikipedia (http://de.wikipedia.org)