Bézier Splines

Historie, mathematische Grundlagen, Anwendung und Programmierung

Semesteraufgabe im Fach Multimedia- & Webtechnologien
Prof. Dr.-Ing. Stefan Gössner
20.Dezember 2005

Johanna Jeuken Benjamin Stüttgen Roman Hagen

Historie

mathematische Grundlagen

Anwendung & Programmierung

Semesteraufgabe im Fach Multimedia- & Webtechnologien Prof. Dr.-Ing. Stefan Gössner

Historie

- Hierachie der Splines
  - Natural Splines
  - Bézier Splines
  - B- Splines
- Unisurf
- Bézier – Casteljau

Was ist ein Spline?

Historie

mathematische Grundlagen

Anwendung & Programmierung

- Ursprünglich aus Schiffsbau
- Eine Reihe von Punkten verbinden

Was ist ein Spline?

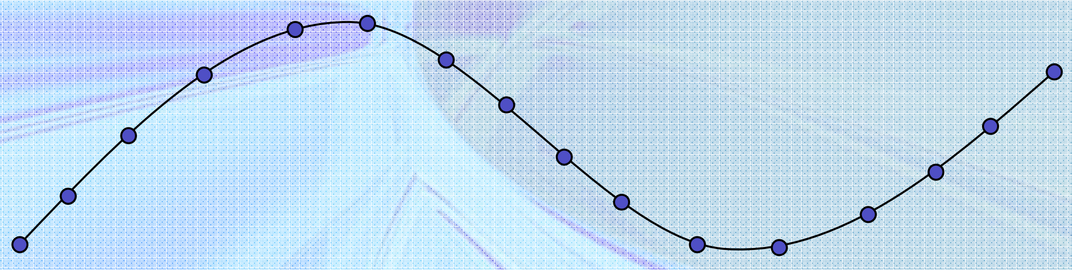
- Im Schiffsbau: dünne, elastische Metall- oder Holzlatten (auch Straklatten genannt) um Konturen zu bilden; Sie wurden unter Spannung an mehren Punkten fixiert
- Ein Spline wird genutzt um eine Reihe von Punkten mit einer möglichst eleganten Kurve zu verbinden
- Bei Interpolation eines Polynoms zwischen Punkten entstehen starke Schwankungen durch viele Maxima und Minima
- Somit werden bei einem Spline Polynome niederen Grades interpoliert; weitere Bedingung ist, dass bei zwei aufeinander treffenden Kurven, die ersten Ableitungen der beiden Kurven gleich sein müssen
- Gebrauch: z.B. Schiffsbau, aerodynamische Karosserieformen im Flugzeug- bzw. Automobilbau, Rastern von Vektorschriften

Hierarchie der Splines

- **Natural Splines**
- Bézier- Splines
- B- Splines

Natural Splines

- Alle Stützpunkte auf der Kurve
- Nur leichte Krümmungen möglich



Hierarchie der Splines

- Natural Splines
- **Bézier- Splines**
- B- Splines

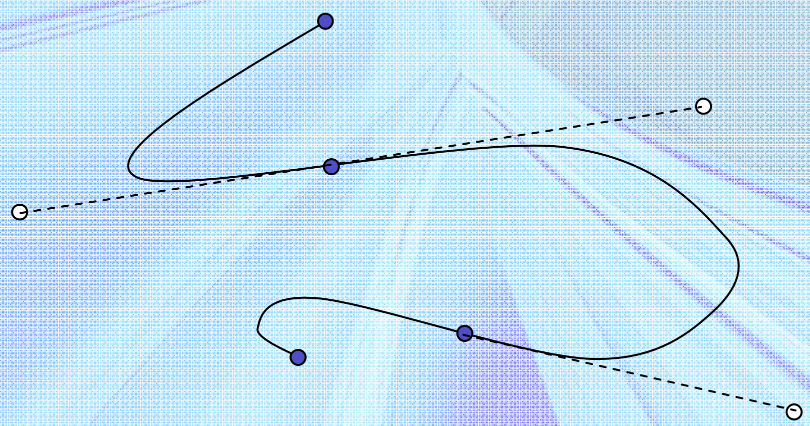
Bézier- Splines

Historie

mathematische Grundlagen

Anwendung & Programmierung

- Stützpunkte auch neben, nur Anfangs- und Endpunkt auf der Kurve
- Flexibler als natural Splines



Semesteraufgabe im Fach Multimedia- & Webtechnologien Prof. Dr.-Ing. Stefan Gössner

Pierre Etienne Bézier

Historie

mathematische Grundlagen

Anwendung & Programmierung

- 1. September 1910 geboren
- 25. November 1999 gestorben
- Französischer Ingenieur
- 1933 – 1975 als Ingenieur bei Renault

Freiformflächen - Autokarosserie

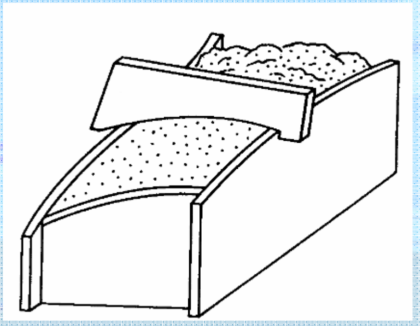
- Bis 1960
  - Skizzen Tonmodelle
  - Master-Model
  - Stanzvorlage
- Gewünscht 1960
  - Verbesserte Datenübertragung
  - Definition der Freiformflächen durch numerische Daten

Freiformflächen - Autokarosserie

- Bis 1960
  - Entwicklung der Freiformflächen der Karosserieteile durch Tonmodelle und Skizzen, nach Korrekturen Anfertigung eines Mastermodells; danach Fertigung der Stanzformen; durch dieses komplizierte Verfahren, Entwicklungszeit bis zu einem Jahr

- Gewünscht 1960
  - Da große Fortschritte in der Computerentwicklung, Datenübertragung vom Zeichner bis hin zum Modellschreiner als numerische Daten gewünscht
  - Bézier begann UNISURF, ein CAD / CAM Programm zu entwickeln

UNISURF



- Grundidee: Prozess der Gießereien
- Definition Fläche: geometrischer Ort einer Kurve, welche zur selben Zeit bewegt und verformt wird
- Dies Prinzip wurde in mathematische Lösung übertragen
- Erste Anwendung 1968
- Vollständiger Einsatz 1975

UNISURF

- Erste Lösungen orientiert an Kinematik und Mechanik
- Bézier leitete sein Konzept von einem Prozess der Abdruckerstellung in Gießereien ab; Sand wurde in eine Form gepresst und der Überschuss mit einer Holzplanke abgetragen
- Diese Form hat Bézier mit vier Polynomen definiert; somit war auch die Fläche bekannt
- Dies war dann das Grundprinzip für sein Programm UNISURF

- Ab 1955 erste Pressmaschinen mit numerischer Eingabe
- Mit UNISURF gesamte Entwicklung von Karosserieteilen mit numerischen Daten möglich

Zeitgleiche Entwicklungen

Historie

mathematische Grundlagen

Anwendung & Programmierung

- Casteljau → Citroên
- Amerikanische Flugzeugindustrie
- Bericht von S.A. Coons

Zeitgleiche Entwicklungen

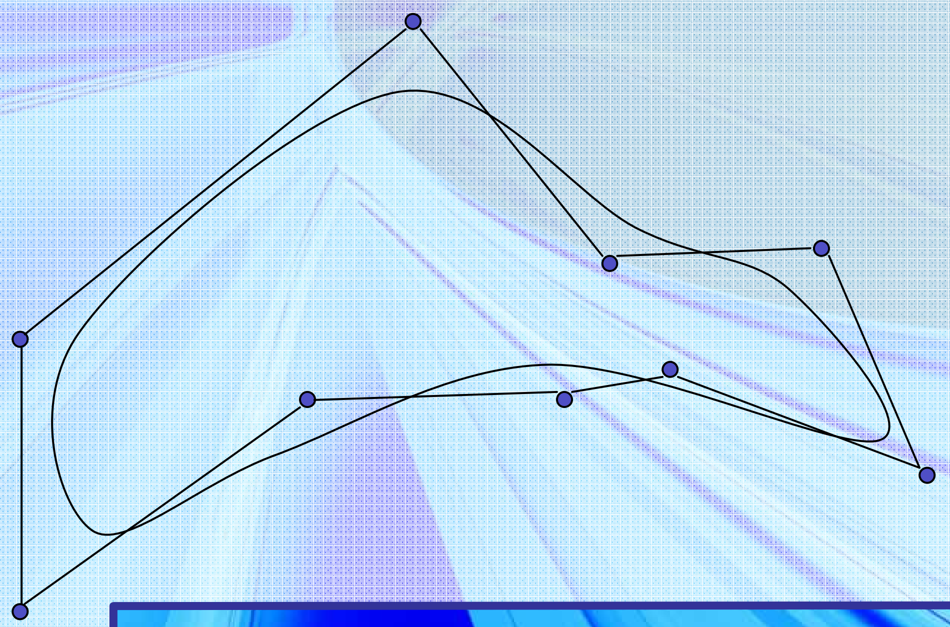
- Parallel zu Béziers Entwicklungen gab es auch Forschungen bei anderen Automobilherstellern und in der amerikanischen Flugzeugindustrie
- Paul de Faget de Casteljau entwickelte die gleiche Theorie wie Bézier schon früher; Citroên, sein Arbeitgeber, hielt seine Entwicklungen aber zu lange geheim, und die Kurven wurden nach Bézier benannt
- Auch der Bericht von S.A. Coons, der die gleiche Thematik behandelt, wurde erst 1967 veröffentlicht

Hierarchie der Splines

- Natural Splines
- Bézier- Splines
- **B- Splines**

B- Spline

- bei geschlossenen Konturen kein Stützpunkt direkt auf Kurve
- Sehr feine Krümmungen realisierbar



NURBS

- **N**on **U**niform **R**ational **B**- **S**pline
- Elementares Werkzeug in CGI
  (**C**omputer **G**enerated **I**magery)

NURBS

- B- Splines sind die Grundlage für NURBS (**N**on **U**niform **R**ational **B**- **S**plines)
- NURBS ist elementares Werkzeug in der **C**omputer **G**enerated **I**magery; somit wichtig bei der Erzeugung von Bildern in der Filmtechnik durch 3D-Computergrafik
- NURBS wird auch in CAD angewendet

Historie

mathematische Grundlagen

Anwendung & Programmierung

Semesteraufgabe im Fach Multimedia- & Webtechnologien Prof. Dr.-Ing. Stefan Gössner

Mathematische Grundlagen

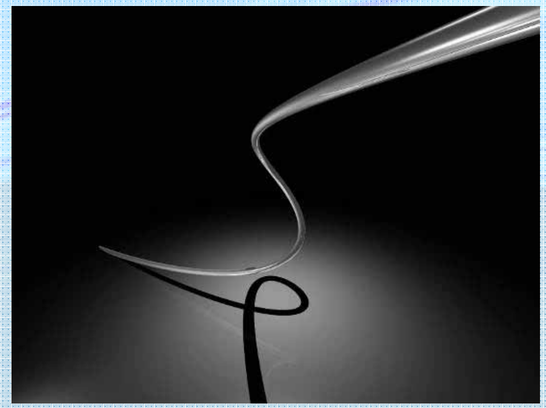
- analytisch beschreibbare Geometrien
- analytisch nicht beschreibbare Geometrien
- parametrische Kurven
- Definition Bézierkurve
- Bernsteinpolynome
- de Casteljau- Algorithmus

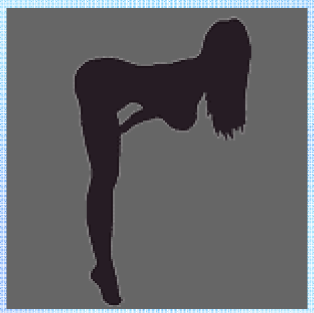
Kurvendiskussion

Historie

mathematische Grundlagen

Anwendung & Programmierung





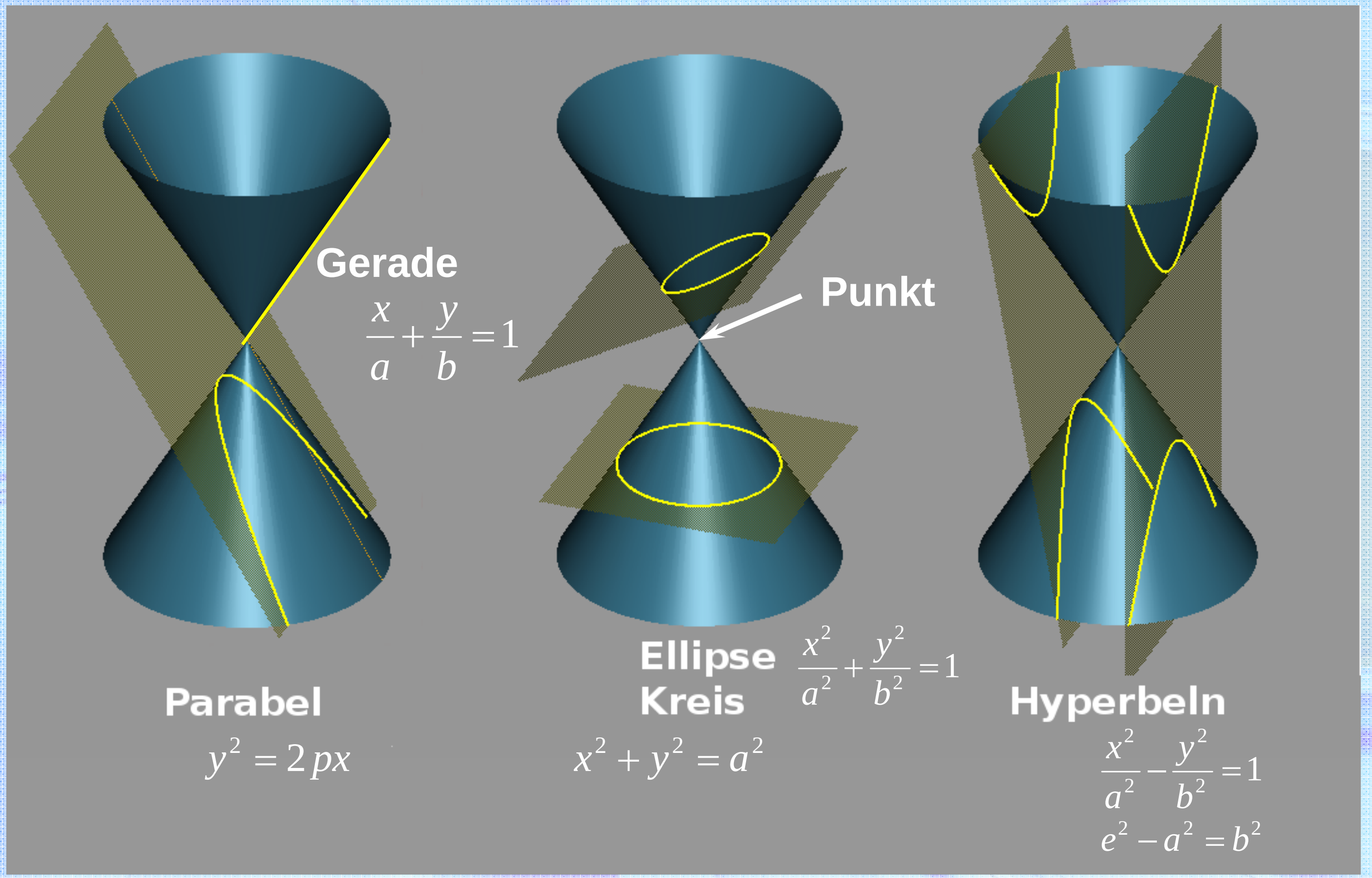
Semesteraufgabe im Fach Multimedia- & Webtechnologien Prof. Dr.-Ing. Stefan Gössner

Analytisch beschreibbare Geometrien

Historie

mathematische Grundlagen

Anwendung
&
Programmierung



Analytisch nicht beschreibbare Geometrien

- **Bézier- Splines**

$$C(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t) \quad mit \quad t \in [0,1]$$

$$B_{i,n}(t) = \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i}$$

- **B- Splines**

$$C(t) = \sum_{i=1}^{n} P_i \cdot B_{i,k}(t) \quad mit \quad t \in [0, n+k-2]$$

$$B_{i,1}(t) := \begin{cases} 1 \; wenn \;\; t_i \le t < t_{i+1} \\ 0 \; sonst \end{cases}$$

$$B_{i,k}(t) := \frac{(t - t_i) \cdot B_{i,k-1}(t)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - t) \cdot B_{i+1,k-1}(t)}{t_{i+k} - t_{i+1}}$$

- **NURBS** = **N**on **U**niform **R**ational **B**- **S**plines

Analytisch nicht beschreibbare Geometrien

Historie

mathematische Grundlagen

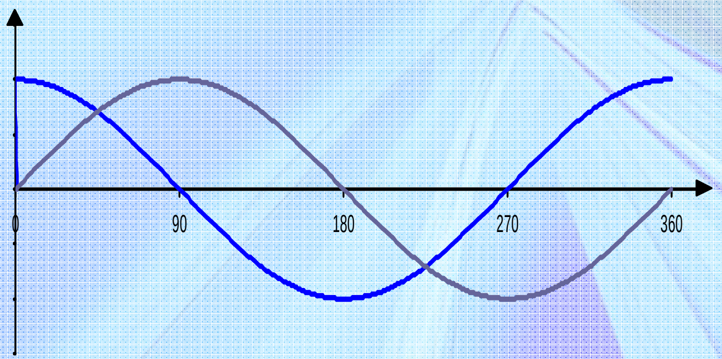
Anwendung & Programmierung

- **Bézier- Splines**
  - •Durch Anwender intuitiv veränderbar
  - •Definition durch Polygon
  - •Approximierend
  - •Grad hängt von Anzahl der Stützstellen ab
  - •Stützstellen haben globalen Einfluss
- **B- Splines**
  - •Definition durch Polygon
  - •Approximierend / Interpolierend
  - •Jeder Stützpunkt hat eigene Gewichtsfunktion
  - •Stützstellen habe lokalen Einfluss
- **NURBS** = **N**on **U**niform **R**ational **B**- **S**plines

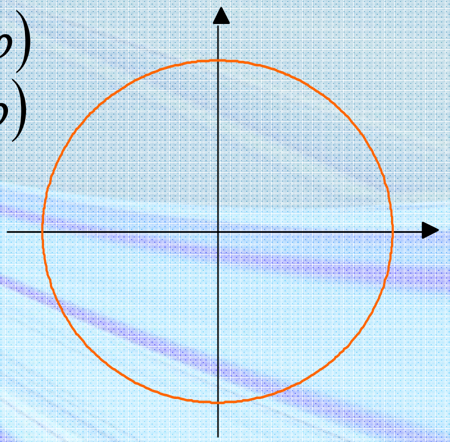
- Approximierend: (v. lat.: *proximus, -a, -um* = der/die/das Nächste)
  - Annäherung > Kurve befindet sich in der Nähe der vorgegebenen Punkte/ Stützstellen.
- Interpolierend: (v. lat.: interpolare = einschieben, zwischenschalten)
  - Stützpunkte liegen auf der Kurve und werden linear, polynomisch oder hier durch Spline verbunden.
- Bézier- Splines:
  - der Designer kann die Gestalt der Kurve leicht verändern, ohne sich mit Ableitungen oder Tangentenvektoren zu befassen
  - Grad der Kurve wird durch Anzahl der Stützpunkte bestimmt
- B- Splines (= Basis- Splines)
  - Stützpunkte hier sog. *De Boor- Punkte*
  - Grad ist unabhängig von der Anzahl der Stützstellen > Teilweise ist höherer Grad notwendig, aber die Anzahl der Stützstellen soll nicht erhöht werden
  - Interpolierte Form wird für Geometrien verwendet, bei der die Punkte genau vermessen wurden, dabei soll natürlich die Kurve durch die Punkte verlaufen, nicht daneben.
  - Gewichtsfunktionen werden so gewählt, dass sie nur in der Nähe des jeweiligen Stützpunktes ungleich Null sind, dh nur der eine Stützpunkt hat in diesem Bereich einen Einfluss auf den Kurvenverlauf.
- NURBS
  - Allgemeinste Kurvenformulierung
  - B- Spline mit unterschiedlicher Gewichtung der Stützstellen
  - Findet Anwendung in CAD- Systemen
  - Kegelschnitte bis Bézierkurven beschreibbar

Parametrische Kurven

- Splines liegen in parametrischer Form vor
- Parameterformulierung: $f(t) = x$, $g(t) = y$
- x und y werden durch unabhängige Funktionen von t berechnet
- keine Schwierigkeiten durch Mehrdeutigkeiten
- auch vertikale Tangenten möglich
- Beispiel > Kreisfunktion:



$$x = r \cdot \cos(\varphi)$$
$$y = r \cdot \sin(\varphi)$$



Definition Bézierkurven

Historie

mathematische Grundlagen

Anwendung & Programmierung

- Eine Bézierkurve n- ten Grades wird definiert durch:

$$C(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t) \quad mit \quad t \in [0,1]$$

und dem Bernsteinpolynom:

$$B_{i,n}(t) = \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i}$$

$$P_i = Stützpunkte$$

Binominalkoeffizient $\binom{n}{i}$

Definition: $$\binom{n}{i} = \frac{n!}{i! \cdot (n-i)!}$$

Abzulesen im pascalschen Dreieck:

Spalte i=0

Zeile n=0 → 1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

Bernsteinpolynome

Historie

mathematische Grundlagen

Anwendung & Programmierung

- reelle Polynome mit ganzzahligen Koeffizienten
- Stammen aus der Approximationstheorie

$$B_{i,n}(t) = \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i} \qquad hier: B_{i,4}\ , 0 \le i \le n$$

$$B_{0.4}(t) = 1 \cdot t^0 \cdot (1-t)^{4-0} = (1-t)^4$$

$$B_{1.4}(t) = 4 \cdot t^1 \cdot (1-t)^{4-1} = 4t(1-t)^3$$

$$B_{2.4}(t) = 6 \cdot t^2 \cdot (1-t)^{4-2} = 6t^2(1-t)^2$$

$$B_{3.4}(t) = 4 \cdot t^3 \cdot (1-t)^{4-3} = 4t^3(1-t)$$

$$B_{4.4}(t) = 1 \cdot t^4 \cdot (1-t)^{4-4} = t^4$$

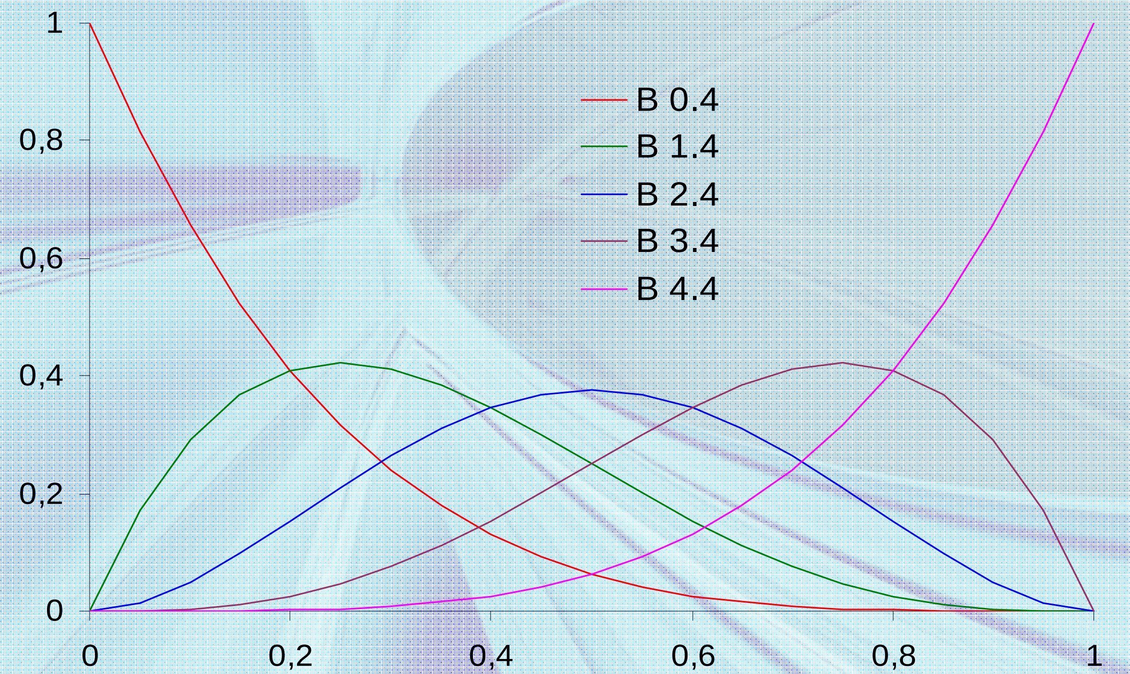
Bernsteinpolynome

Historie

mathematische Grundlagen

Anwendung & Programmierung

Aus den Bernsteinpolynomen ergeben sich für die Bézier- Splines:
Die Kurve verläuft genau durch den ersten (P_0) und letzten (P_n) Stützpunkt des Polygons, da nur genau das erste Polynom an der Stelle t=0 und das letzte Polynom an der Stelle t=1 den Wert 1 besitzen!



Berechnung der Kurve

$$C(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t) \quad z.B.\ \text{für}\ t = 0{,}5$$

$$\boxed{\text{Stützpunkte}: P_0(0{,}0{,}0);\ P_1(30{,}25{,}0);\ P_2(60{,}30{,}0);\ P_3(90{,}-10{,}0);\ P_4(150{,}0{,}0)}$$

$$\begin{aligned}C_x(0{,}5) &= B_{0.4}(0{,}5)\cdot P_{x0} + B_{1.4}(0{,}5)\cdot P_{x1} + B_{2.4}(0{,}5)\cdot P_{x2} + B_{3.4}(0{,}5)\cdot P_{x3} + B_{4.4}(0{,}5)\cdot P_{x4}\\ &= 0{,}0625\cdot 0 + 0{,}25\cdot 30 + 0{,}375\cdot 60 + 0{,}25\cdot 90 + 0{,}0625\cdot 150\\ &= 61{,}875\end{aligned}$$

$$\begin{aligned}C_y(0{,}5) &= B_{0.4}(0{,}5)\cdot P_{y0} + B_{1.4}(0{,}5)\cdot P_{y1} + B_{2.4}(0{,}5)\cdot P_{y2} + B_{3.4}(0{,}5)\cdot P_{y3} + B_{4.4}(0{,}5)\cdot P_{y4}\\ &= 0{,}0625\cdot 0 + 0{,}25\cdot 25 + 0{,}375\cdot 30 + 0{,}25\cdot(-10) + 0{,}0625\cdot 0\\ &= 15\end{aligned}$$

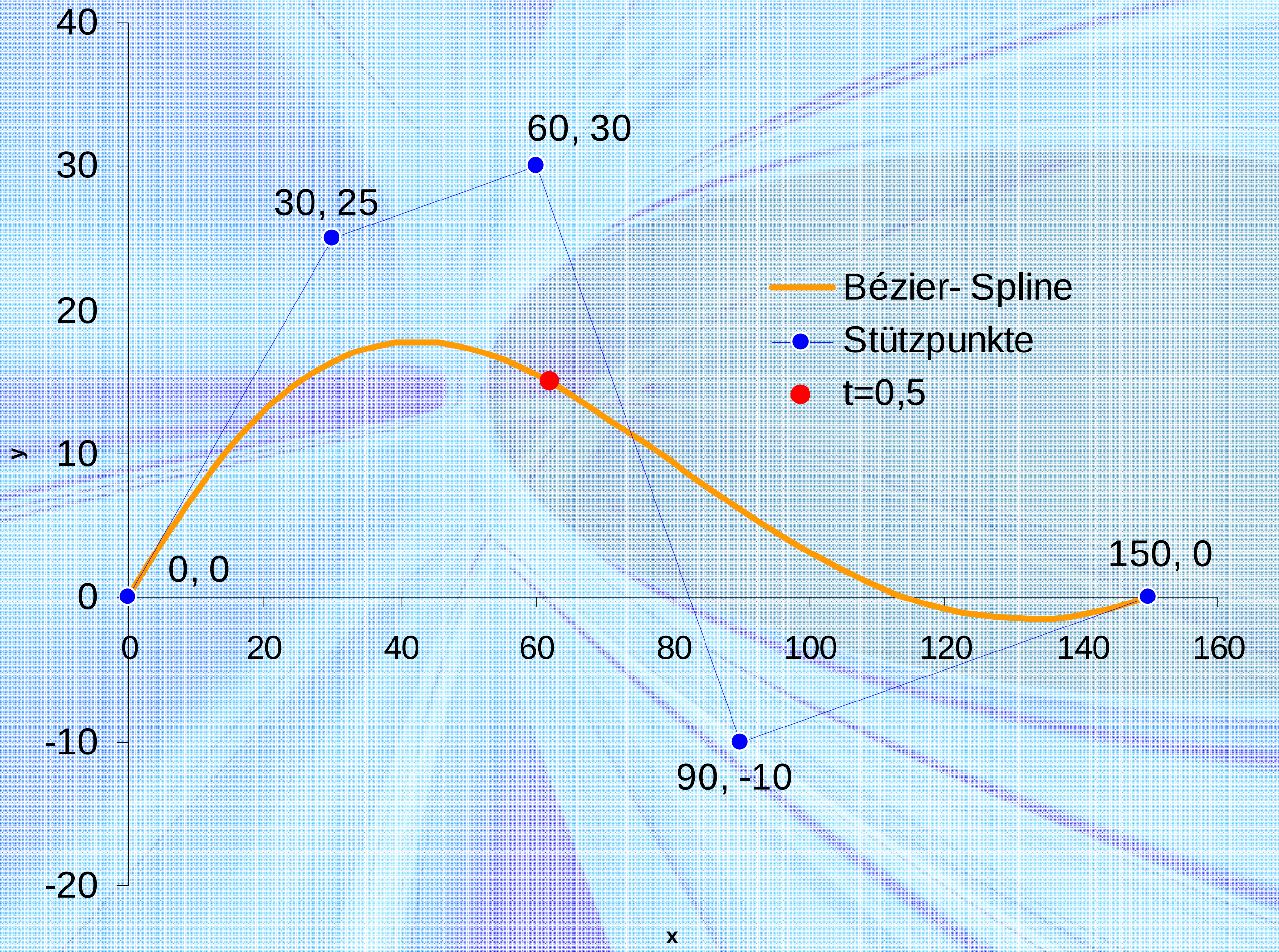
$$C_z(0{,}5) = 0$$

Bézier- Spline

Historie

mathematische Grundlagen

Anwendung & Programmierung



Semesteraufgabe im Fach Multimedia- & Webtechnologien Prof. Dr.-Ing. Stefan Gössner

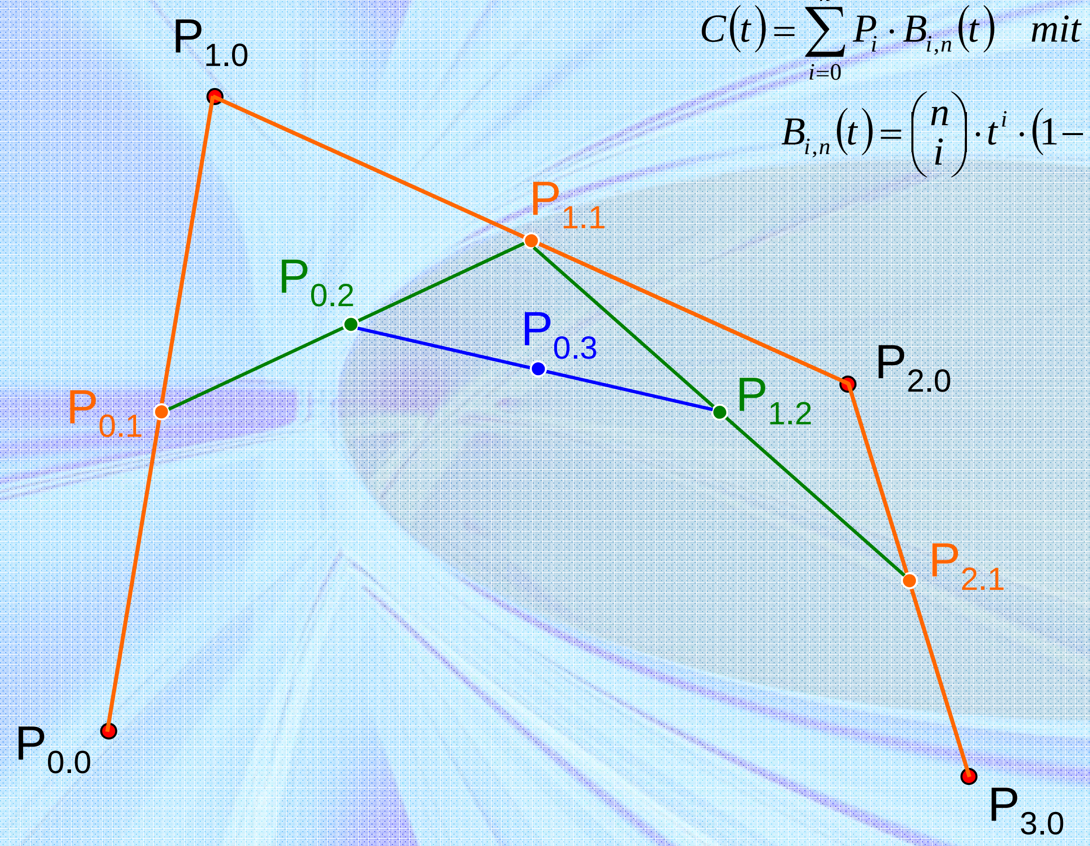
Geometrische Approximation

- Algorithmus von Casteljau [de Casteljau, 1959]
- Geometrische Konstruktion der Kurvenpunkte einer Bézierkurve
- Um einen Punkt auf der Kurve mit einem bestimmten Parameterwert t zu ermitteln werden die Polygonseiten im Verhältnis t zu 1-t geteilt. Die sich ergebenden Punkte werden miteinander verbunden und diese Strecke wiederum in dem gleichen Verhältnis geteilt. Dieses wiederholt sich nun genau n mal, dh bei 4 Stützstellen ist n=3 (0; 1; 2; 3), somit muss das Verfahren dreimal wiederholt werden. Bei der n- ten (hier: dritten) Teilung ergibt sich genau der Punkt an der Stelle t auf der Bézierkurve.
- Ebenfalls ist die letzte sich ergebene Strecke die Tangente an die Kurve in diesem Punkt.
- Das gesamte Verfahren kann nun für beliebig viele Werte von t wiederholt werden.
- Zwischen diesen Punkten wird die Kurve interpoliert.

De Casteljau- Algorithmus

$$C(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t) \quad mit \quad t \in [0,1]$$

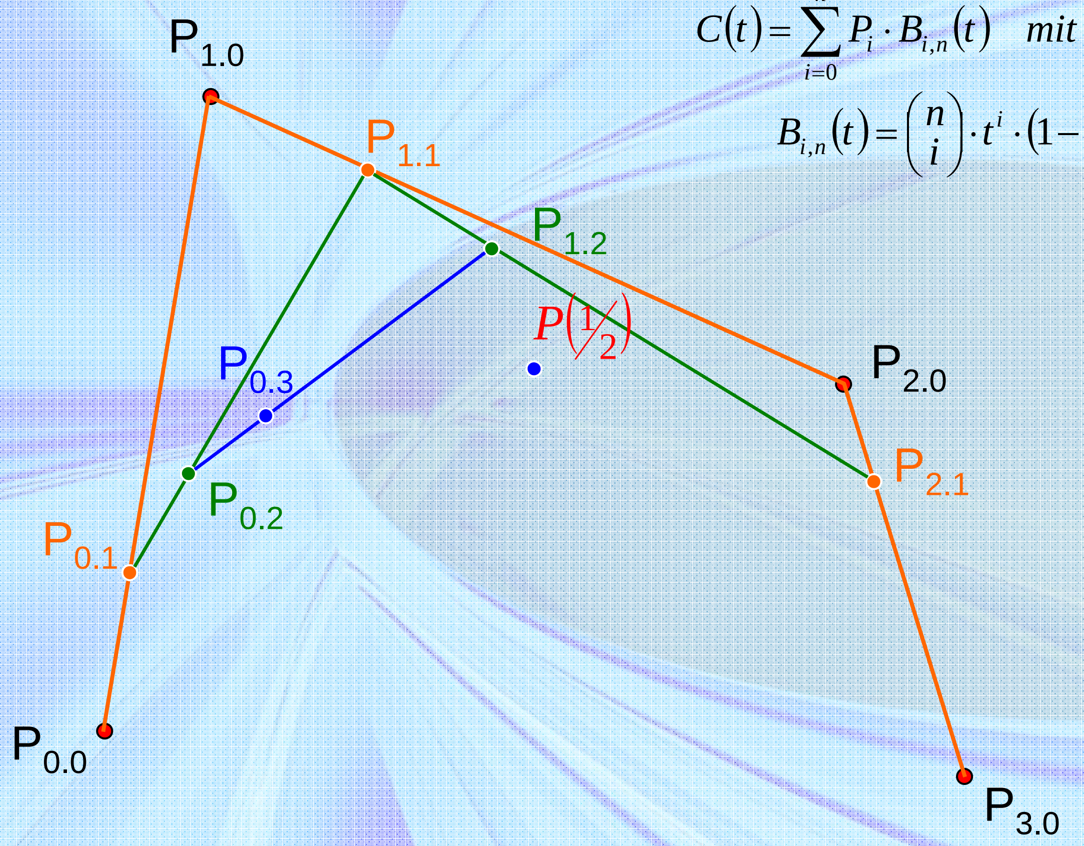
$$B_{i,n}(t) = \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i}$$



De Casteljau- Algorithmus

$$C(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t) \quad mit \quad t \in [0,1]$$

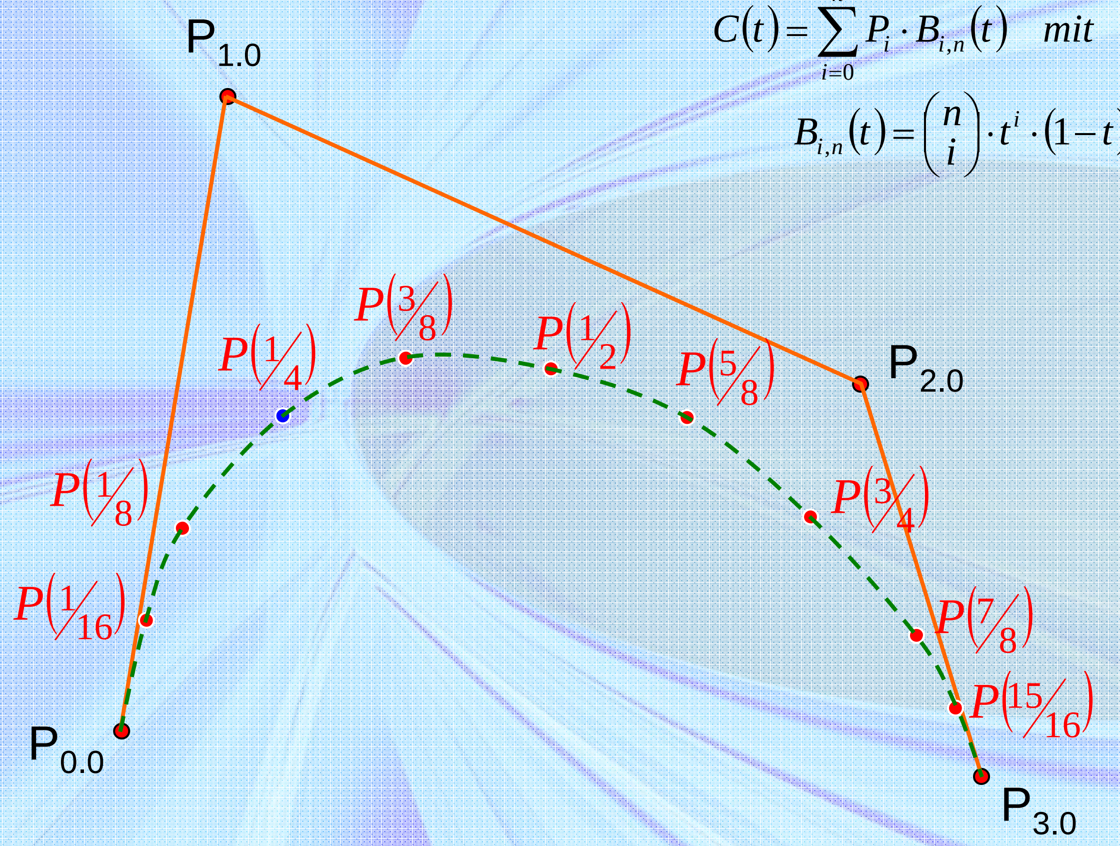
$$B_{i,n}(t) = \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i}$$



De Casteljau- Algorithmus

$$C(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t) \quad mit \quad t \in [0,1]$$

$$B_{i,n}(t) = \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i}$$



Historie

mathematische Grundlagen

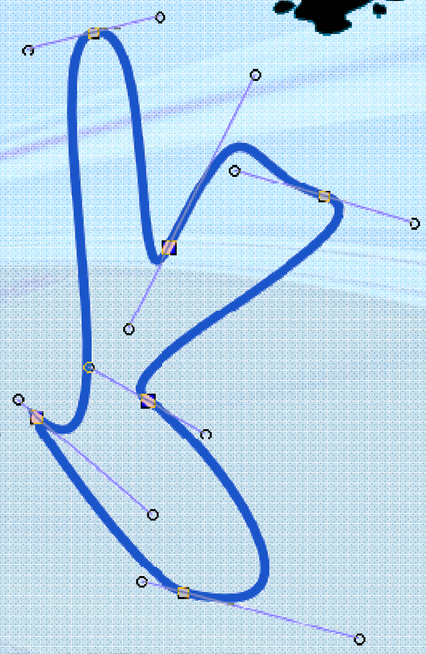
Anwendung & Programmierung

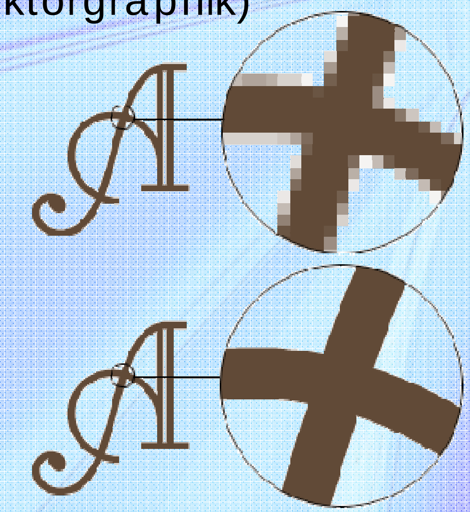
Anwendung und Programmierung

- 2D - Vektorgraphik (SVG – Inkscape)
- 3D – Organic Design / Bézier- Patch-Modelling
- Digitale Typografie
- Modelling im CAD
- Programmierung (C++)

2D: Vektorgraphiken – am Beispiel >*INKSCAPE*<

- Inkscape unterstützt den vom *W3C* vorgeschlagene „*Scalable Vector Graphics*“ – Standard (*SVG*); ist ein offener, zugänglicher Standard
- Inkscape (*SVG*) verknüpft zusätzlich *XML* und *CSS2* (hohe Kompatibilität im Internet durch reines Text-Format)
- Anwendung in elektronischen Medien, sowie Print-Illustrationen
- Bildaufbau über mathematische Funktionen mit Hilfe von Koordinaten (keine Pixel)
- Unterstützt Animation und Skalierbarkeit (Vorteil Verktorgraphik)





→ DURCH TEXTEINGABE, ABER EHER UNPRODUKTIV. ODER MIT WERKZEUGEN UND OPERATIONS- BZW. EDITIERFUNKTIONEN

(z.B. im gängigen Media-Prog. wie CorelDraw, Inkscape, Adobe Illustrator,…)

Download: inkscape.org

Erläuterung: 2D-Vektorgraphik (*Inkscape*)

Historie

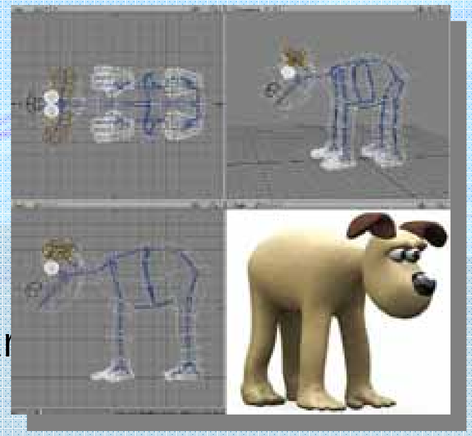
mathematische Grundlagen

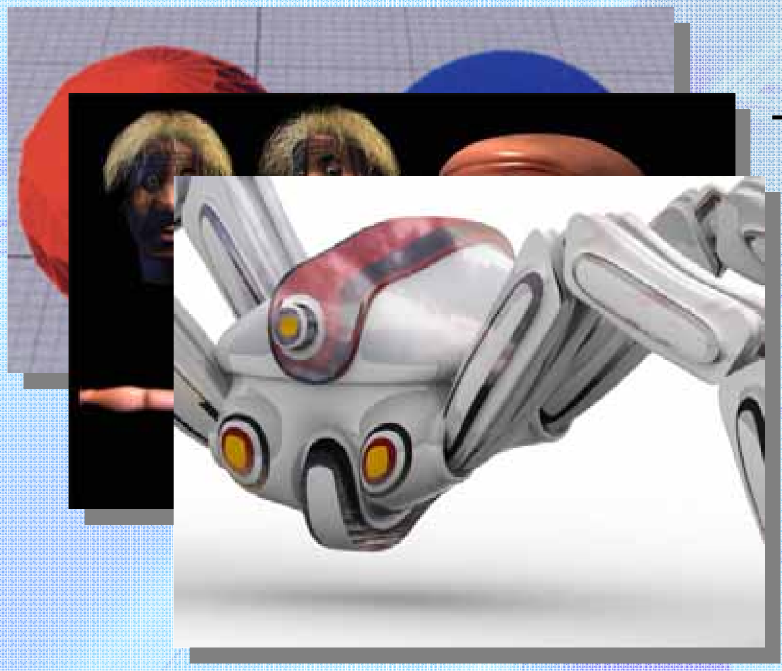
Anwendung & Programmierung

- Mit *Inkscape* lassen sich alle neben allen gängigen Maloperationen, Animationen und komplexe Pfadoperationen mit Hilfe von Bézierkurven generieren. Über die Drehpunkte und Referenzpunkte der Tangenten lassen sich die Kurven editieren.
- Vektorgraphiken beschreiben ein Bild durch mathematische Funktionen im 2D- oder 3D-Koordinatensystem (Ursprung oben links).
- Eigenschaften der Elemente (Linien, Kurven, Flächen,...) bleiben erhalten nach Verzerrung oder stufenloser Skalierung und können nachträglich noch verändert werden.
- Sind allerdings ungeeignet für Darstellung komplexer Bilder (z.B. Foto)
- Im *WWW* liegen Vektorgraphiken im offenen Format „*SVG*“ vor, dieses basiert auf *XML* und kann entweder durch einen Editor erstellt werden oder mittels beispielsweise eines OpenSource – Programms wie *Inkscape* erstellt werden.

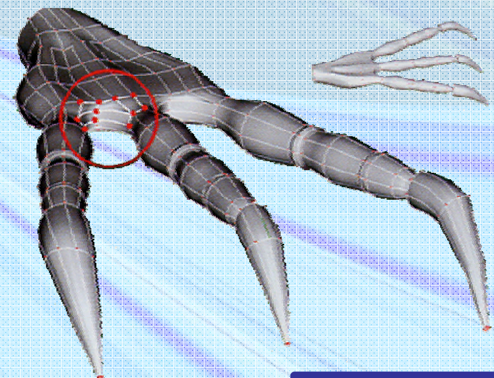
3D: Organic Design oder Bézier-Patch-Modelling

- Durch Bézierkurven wird die Polygonale Modellierung ersetzt (Punkt, Kante, Fläche)
- Bézierkurven als Randkurven definieren die tangentialen Patch-Übergänge
- Erstellen von gekrümmten Oberflächen, naht-losen Übergängen und organischen Körpern
- Auch Kamerapfade als Bézier-Splines definierbar





→ ÜBER TANGENTEN AN PUNKTEN DER RANDKURVEN
(z.B. in gängigen Media-Prog. wie 3D Studio Max, BodyPaint,...)



Historie

mathematische Grundlagen

Anwendung & Programmierung

Erläuterung: Bézier-Patch-Modelling

Historie

mathematische Grundlagen

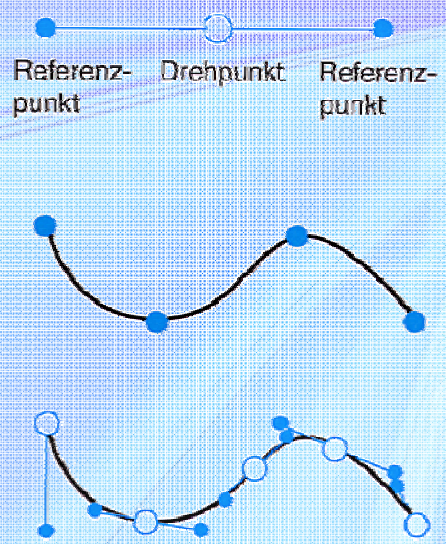
Anwendung & Programmierung

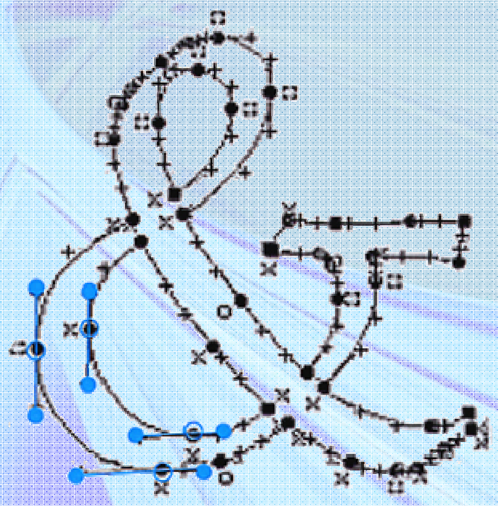
- *Bézier*-Patch-Modelling (oder auch *NURBS*-Patch-Modelling) ersetzen das Polygonale Modelling, wobei sich die Geometrien aus Flächen zusammen setzen (Drei- oder Vielecke) und ein Netz (mesh) bilden. Je detaillierter, desto mehr Aufwand nötig.
- *Bézier*-Patch-Modelling arbeitet dagegen mit Bézierkurven als Randkurven bei den Patch-Oberflächen (kleine Elemente des Netzes). Die beiden Tangenten eines Punktes der Randkurve werden über die beiden Béziergriffe, die außerhalb der Kurve liegen definiert. Randbedingungen steuern und manipulieren.
- Dadurch sehr weiche, sanfte Übergänge und Oberfächen möglich. Durch detaillierte Steuerung sind komplexe Geometrien möglich. Brauche keinen hohen Detailaufwand wie bei der polygonalen Modellierung.
- 3D-Patch ist oft ein Rechteck mit vier Kanten, vier Eckpunkten und ev. auch noch selbst ein Patch-Raster mit wiederum Patches. Die Eckpunkte besitzen zur Verformung Bézier-Tangenten-Griffe um die Kanten zu modifizieren und so eine Veränderung der Randkurve und damit der Fläche.

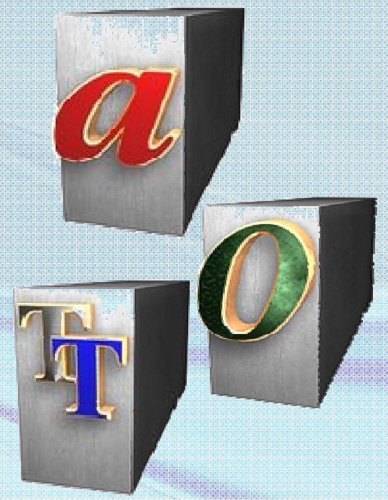
Konturbeschreibung in der Mikro- und Detailtypografie

- Digitale gestalterische Konstruktion von Buchstaben und Zeichen
- Jede gängige digitale Schrift wurde mittels Splines erstellt
- Bei *PostScript*- und *OpenType*-Formaten (Seitenbeschreibungssprache) vorwiegend Verwendung von Bézierkurven

→ DURCH REFERENZPUNKTLAGE WIRD DIE FORM / KONTUR BEEINFLUSST (z.B. im gängigen Illustratrionsprogrammen wie *Adobe Illustrator)*







Erläuterung: Typografie

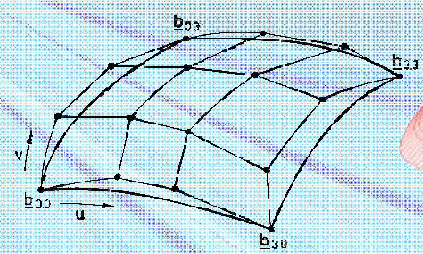
- Digitale Schriften werden in 3 Arten dargestellt. Als Bitmap, als Ikarus-Format (Koordinaten bezogene Bedeutung) oder als mathematische Beschreibung der Kontur.
- Jede beliebige Kurve lässt sich durch Polynomfunktionen darstellen und somit beschreiben.
- Digitale Schriften werden mit Hilfe von Splines erstellt. Bézier-kurven erfahren vorwiegend ihre Anwendung bei den *PostScript*- und *OpenType*-Formaten (Seitenbeschreibungs-sprachen). *OpenType* ist die Weiterentwicklung von *PostScript*, welches vorwiegend als Druckanwendung diente, und ist plattformübergreifend, d.h. Druck- sowie Bildschirmanwendung.
- Bei der digitalen Typografie werden die Konturen der Zeichen, Symbole, Figuren mittels der Drehpunkte, Tangenten und der Bézierstützstellen erstellt. Durch Verschiebung dieser Elemente in so genannten Illustrationsprogrammen (z.B. von Firma Adobe) erhält man die gewünschten Konturänderungen. Béziersplines sind dabei wesentlich sensibler und detaillierter als „normale“ Spline-Typen.

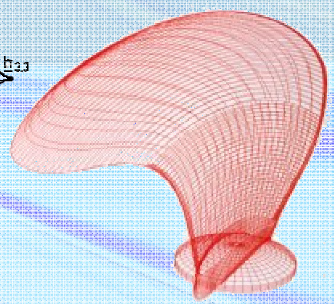
Modelling im Computer Aided Geometric Design

- Methode für nicht analytisch beschreibbare Kurven und Flächen (Sculptured Surfaces)
- Flugzeugbau, Automobilbau, Schiffsbau, Werkzeugbau, Formenbau, ...
- Vor allem im immer mehr verbreitertem 3D-Modelling; designerisch im „weicher“

→ ÜBER STÜTZSTELLEN WIRD SPLINE / FREIFORMFLÄCHE GENERIERT

(z.B. in gängigen CAD-Progr. wie *Catia*, *Solid Works*, ...)





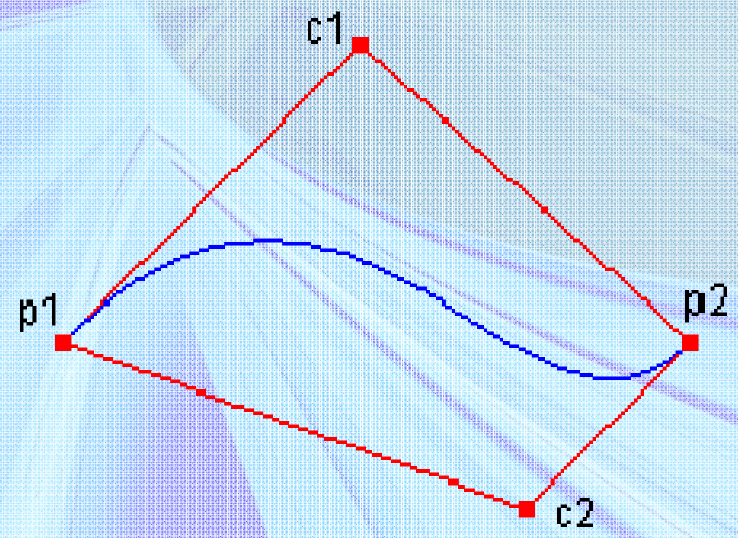
Erläuterung: Modelling im CAD

- Im CAD liegt der Ursprung bzw. die Erfindung der Bézier-Splines bzw. Bézierflächen (siehe auch Historie).
- Neben analytisch beschreibbaren Kurven und Flächen kommen beim Konstruieren häufig auch nicht analytisch beschreibbare vor, z.B. von Freihandlinien abgeleitete flächen, Strömungsprofilformen, Oberflächen von Gußteilen, usw.
- Freihandkurven oder nicht analytisch beschreibbare Linien werden als durch Stützstellen gelegte Linien (Splines) definiert. Über diese Stützstellen kann wiederum eine Fläche generiert werden, welche durch Extrusion, Erhebung oder Rotation in einen 3D-Körper gewandelt wird. Auch so genannte *Sweepings* (Pfadextrusion), wo Skizzen entlang eines 2D-/3D-Splines generiert werden sind gängige Werkzeuge im CAD
- Vorwiegend im Schiff-, Flugzeug- oder Karosseriebau
- Der Trend im CAD geht immer weiter zum 3D-Modelling, und hier designerisch wiederum zu organischen, weichen Kurven.

Programmierung von Bézier-Splines mit C++

- Für beispielsweise Microsoft Windows *Graphics Device Interface (GDI)* im 2D-Bereich (Darstellungssprache von MS-Windows am Bildschirm)

```
Point p1(10, 100);    // start point
Point c1(100, 10);    // first control point
Point c2(150, 150);   // second control point
Point p2(200, 100);   // end point
Pen pen(Color(255, 0, 0, 255));
graphics.DrawBezier(&pen, p1, c1, c2, p2);
```



Kubischer Bézier-Spline

Historie

mathematische Grundlagen

Anwendung & Programmierung

Semesteraufgabe im Fach Multimedia- & Webtechnologien Prof. Dr.-Ing. Stefan Gössner

Programmierung von Bézier-Splines mit C++

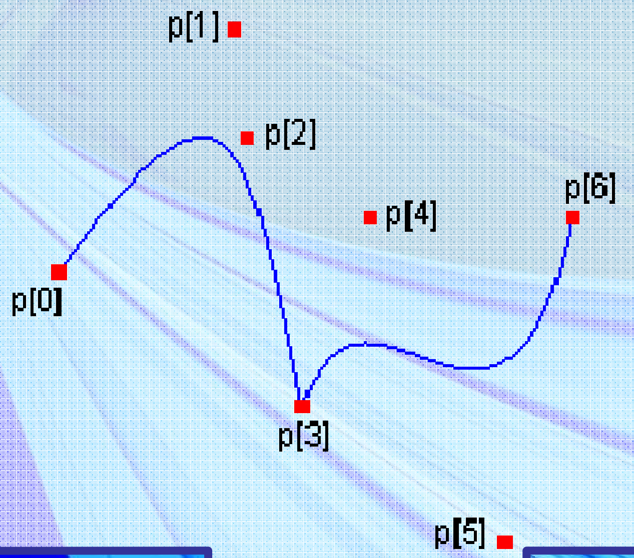
Historie

mathematische Grundlagen

Anwendung & Programmierung

- Für beispielsweise MS-Windows *Graphics Device Interface (GDI)* im 2D-Bereich (Darstellungssprache von MS-Windows am Bildschirm)

```
Point p[] = {
   Point(10, 100),    // start point of first spline
   Point(75, 10),     // first control point of first spline
   Point(80, 50),     // second control point of first spline
   Point(100, 150),   // end point of first spline and
                      // start point of second spline
   Point(125, 80),    // first control point of second spline
   Point(175, 200),   // second control point of second spline
   Point(200, 80)};   // end point of second spline
Pen pen(Color(255, 0, 0, 255));
graphics.DrawBeziers(&pen, p, 7);
```



2 Kubische Bézier-Splines

Erläuterung: Programmierung Bézier-Splines

Historie

mathematische Grundlagen

Anwendung & Programmierung

- Neben der Programmierung von *SVG*-Files, mittels eines *XML*-Editors oder eines Illustrationsprogramms gibt es weitere Sprachen die Bézierkurven programmieren können.
- Beispielsweise bedient sich die Darstellungssprache *GDI* (Graphics Device Interface) von MS-Windows mittels der Programmiersprache *C++* der Möglichkeit im 2D-Bereich Bézier-kurven am Bildschirm bzw. Drucker darzustellen.
- Dabei bestehen die Bézier-Splines immer aus einem Start- und Endpunkt, sowie zwei Kontrollpunkten. Werden mehrere Splines aneinander gesetzt wird der Endpunkt gleichzeitig zum Start-punkt des folgenden Splines (z.B. 2 Splines = 7 Punkte)
- Die Punkte werden mittels Koordinaten programmiert
- Der *C++* -Befehl bzw. die -Funktion *graphics.DrawBeziers* ist letztendlich der Indikator für die Erstellung der Splines am Bild-schirm bzw. am Drucker
- Natürlich lassen sich auch Bézier-Splines mittels anderer Sprachen programmieren, wie z.B. Javascript, Java, …

Literatur

Böhringer, J.; Bühler, P.; Schlaich, P.; Ziegler, H.-J.
Kompendium der Mediengestaltung; 2. Auflage;
Springer; Berlin, Heidelberg, New York; 2000; S.116 f.

Farin, G.
Kurven und Flächen im Computer Aided Geometric Design, 1. Auflage,
Braunschweig u.a., 1994, S.1-10

Grieger, I.
Graphische Datenverarbeitung, 2. Auflage
Springer; Berlin, Heidelberg, New York; 1992; S. 61-80

Mach, R.
3D Visualisierung; 1. Auflage;
Galileo Press; Bonn; 2000; S.54-57

Pahl, G.
Konstruieren mit 3D-CAD Systemen; 1. Auflage;
Springer; Berlin, Heidelberg, New York; 1990; S.74 f.

Peters, H.- F.
Rechnerunterstützte Gestaltung und Darstellung
Vieweg; Braunschweig; 1988; S. 45-50, 199-205

Spur, G.; Krause, F.- L.
Das virtuelle Produkt – Management der CAD- Technik
Hanser; München, Wien; 1997; S. 137-152

Von Koenigsmarck, A.
3D Character Design; 1. Auflage;
Galileo Press; Bonn; 2000; S.30 f.

http://www.fh-friedberg.de/users/mlutz/Javakurs/applets/Bezier/bezierhistorie.htm
http://www.fh-fulda.de/caelabo/inhalte/projekte/weber/.html
http://www.ibiblio.org/e-notes/Splines/
http://www.typografie.info/typowiki/index.php
http://www.wikipedia.de