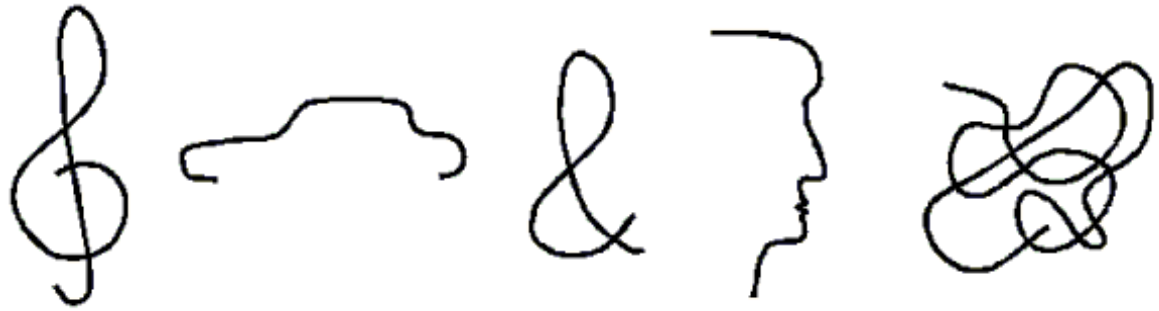
Bézierkurven

1. Einführung

Wir haben bisher gelernt, wie man Kurven durch Parameterdarstellungen beschreiben kann und wie man aus solchen Parameterdarstellungen Eigenschaften der Kurve (Tangenten, Bogenlänge, Krümmung etc.) ermitteln kann. In vielen "konstruktiven" Anwendungsbereichen (Design von Zeichensätzen, architektonische Entwürfe, Karosseriebau etc.) kommt man aber mit der Benutzung "bekannter" Kurven (Kreise, Spiralen, Zykloiden u.v.a.) natürlich nicht aus, sondern man will **beliebige** (etwa von einem Designer visuell vorgegebene) Kurven formelmäßig beschreiben können. Es ist klar, dass es bei Kurven folgender Art



nahezu aussichtslos (und auch praktisch wenig sinnvoll) wäre, sie durch jeweils eine einzige, notwendigerweise hochkomplizierte Formel beschreiben zu wollen. Außerdem möchte man solche "Freihandkurven" als Vektorgrafik-Objekte speichern können. So wie man in der Vektorgrafik für ein Rechteck nur zwei gegenüberliegende Eckpunkte speichert, für eine Ellipse nur das umschreibende Rechteck etc., soll auch eine solche Kurve nur durch einen mehr oder weniger umfangreichen Datensatz beschrieben werden, aus dem die Kurve dann jederzeit nach fest programmierten Formeln gezeichnet werden kann. Die "Wunschliste" an das zu entwickelnde Verfahren sieht also etwa wie folgt aus:

- Es sollen nahezu beliebige Kurven darstellbar sein.
- Die Parametrisierungen sollen immer über dieselben festen Formeln erfolgen, so dass die einzelnen Kurven nur durch Datensätze beschrieben werden.
- Die Kurven sollen nachträglich bearbeitbar und insbesondere "lokal modellierbar" sein, d. h. dass an jeder Stelle ein "Fine Tuning" erfolgen kann, ohne weiter entfernt liegende Kurventeile zu beeinflussen.
- Für die einfache Umsetzung und intuitive Bedienung wäre es außerdem wünschenswert, wenn die abzulegenden Daten einen anschaulichen Bezug zu den visuellen Vorstellungen vom Aussehen der Kurve hätten.

Diese Forderungen klingen zunächst sehr anspruchsvoll und zum Teil geradezu widersprüchlich. Sie lassen sich aber mit Hilfe der sogenannten **Bézierkurven** auf erstaunlich einfache Weise erfüllen. (Pierre Étienne Bézier, 1910-1999, war in den 60er Jahren und noch bis 1975 Chefingenieur bei Renault in Paris.)

Die Lösung sieht im Prinzip wie folgt aus:

- Die Kurven werden aus vielen kleinen Stücken ("Segmenten") zusammengesetzt.
- Für die Parameterdarstellungen jedes einzelnen Segments werden kubische Polynome benutzt. Die Koeffizienten dieser Polynome sind die abzulegenden Daten.
- Die einzelnen Segmente werden "glatt" zusammengesetzt, so dass optisch der Eindruck einer "sauberen" Kurve entsteht.

Bemerkung: Es gibt auch Varianten mit Polynomen anderen Grades, die aber deutlich seltener benutzt werden. Wir behandeln im Folgenden nur den "klassischen" Fall der kubischen Polynome.

2. Die Bézierdarstellung eines kubischen Polynoms

Bei der erwähnten Parametrisierung der einzelnen Segmente mittels kubischer Polynome sind noch einige, für das Verfahren sehr wichtige Besonderheiten zu beachten:

- Als Parameterintervall wird immer das Intervall $[0, 1]$ benutzt.
- Die kubischen Polynome werden nicht in der gewohnten, nach x-Potenzen sortierten Darstellung geschrieben, sondern in der nachfolgend erläuterten *Bézierdarstellung*.

Ist $f : [0, 1] \to \mathbb{R}$ ein Polynom dritten Grades in der Variablen t, so wäre die übliche Schreibweise

$$f(t) = at^3 + bt^2 + ct + d\ .$$

Die Bézierdarstellung eines solchen Polynoms hat dagegen die Gestalt:

$$f(t) = b_0(1-t)^3 + 3b_1(1-t)^2t + 3b_2(1-t)t^2 + b_3t^3\ .$$

Durch Ausmultiplizieren der Klammern und Sortieren, könnte man die "gewohnten" Koeffizienten a, b, c, d aus den "Bézier-Koeffizienten" b_0, b_1, b_2, b_3 errechnen (und umgekehrt). Solche Umrechnungen sind jedoch i. a. weder erforderlich noch sinnvoll, da bei dem ganzen Verfahren *ausschließlich* mit den Bézier-Koeffizienten gearbeitet wird.

Die Bézierdarstellung sieht zunächst aus wie eine unnötig komplizierte Art, ein Polynom hinzuschreiben. Ihre Vorteile - insbesondere auf dem Definitionsbereich $[0, 1]$ - sind aber bereits zu erkennen. Im linken Randpunkt ($t = 0$) verschwinden die letzten drei Summanden, im rechten Randpunkt ($t = 1$) die ersten drei. Das heißt, dass b_0 und b_3 direkt die Funktionswerte in den Intervall-Enden sind. Allgemeiner gilt, dass der Funktionsverlauf für t-Werte in der Nähe von 0 (linke Intervallhälfte) überwiegend durch die beiden ersten Koeffizienten bestimmt wird, für t-Werte in der Nähe von 1 (rechte Intervallhälfte) dagegen überwiegend durch die beiden letzten Koeffizienten. Das liefert später den gewünschten anschaulichen Zusammenhang zwischen Datensatz und Aussehen der Kurve.

3. Béziersegmente

Ein einzelnes Béziersegment ist eine Kurve über dem Parameterintervall $[0, 1]$, für deren Parameterdarstellung sowohl in der x- als auch in der y-Koordinate ein kubisches Bézierpolynom benutzt wird:

$$\vec{p}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} b_0(1-t)^3 + 3b_1(1-t)^2t + 3b_2(1-t)t^2 + b_3t^3 \\ c_0(1-t)^3 + 3c_1(1-t)^2t + 3c_2(1-t)t^2 + c_3t^3 \end{pmatrix}, \quad t \in [0, 1].$$

Dies ist die (wie gewünscht) fest programmierbare Formel für die Parameterdarstellung. Der abzulegende Datensatz besteht jeweils genau aus den acht Koeffizienten

$$b_0,\ b_1,\ b_2,\ b_3,\ c_0,\ c_1,\ c_2,\ c_3\ .$$

Da x- und y-Kompononte formal völlig gleich aussehen, schreibt man obige Darstellung meistens lieber in folgender Form:

$$\vec{p}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} b_0 \\ c_0 \end{pmatrix}(1-t)^3 + 3\begin{pmatrix} b_1 \\ c_1 \end{pmatrix}(1-t)^2t + 3\begin{pmatrix} b_2 \\ c_2 \end{pmatrix}(1-t)t^2 + \begin{pmatrix} b_3 \\ c_3 \end{pmatrix}t^3, \quad t \in [0, 1].$$

Die vier "Koeffizienten" dieses Polynoms nennt man die "Bézierpunkte" des Segments. (Ein Polynom, dessen Koeffizienten Vektoren aus dem $\mathbb{R}^2$ sind, ist vielleicht zunächst etwas ungewohnt. Diese Schreibweise ist aber sehr praktisch und nach der ersten Verblüffung eigentlich auch sehr eingängig.)

Zusammenfassend kann man also sagen: **Das Vektorgrafik-Objekt "Béziersegment" wird beschrieben durch Angabe seiner vier Bézierpunkte**.

Der Rechner kann dieses Segment zeichnen, indem er das Intervall $[0, 1]$ in einer mehr oder weniger engen Schrittweite durchläuft, den jeweiligen Kurvenpunkt aus der obigen Formel berechnet und plottet.

Aufgrund der speziellen Bézierdarstellung sind der erste und letzte Bézierpunkt genau Anfangs- und Endpunkt der Kurve. Um die anschauliche Bedeutung der beiden übrigen Bézierpunkte zu erkennen, betrachten wir die Ableitung der Parameterdarstellung (d. h. die Tangentenvektoren des Segments):

$$\vec{p}'(t) = \begin{pmatrix} x'(t) \\ y'(t) \end{pmatrix} = -3\begin{pmatrix} b_0 \\ c_0 \end{pmatrix}(1-t)^2 + 3\begin{pmatrix} b_1 \\ c_1 \end{pmatrix}(1-t)(1-3t) + 3\begin{pmatrix} b_2 \\ c_2 \end{pmatrix}(2-3t)t + 3\begin{pmatrix} b_3 \\ c_3 \end{pmatrix}t^2$$

Die Tangentenvektoren im Anfangs- und Endpunkt sind also

$$\vec{p}'(0) = -3\begin{pmatrix} b_0 \\ c_0 \end{pmatrix} + 3\begin{pmatrix} b_1 \\ c_1 \end{pmatrix}$$

$$\vec{p}'(1) = -3\begin{pmatrix} b_2 \\ c_2 \end{pmatrix} + 3\begin{pmatrix} b_3 \\ c_3 \end{pmatrix}$$

Dies kann man einfach nach den beiden "inneren" Bézierpunkten (Indizes 1 und 2) auflösen. Zusammenfassend erhält man folgende Beziehungen zwischen den Bézierpunkten und den anschaulichen Eigenschaften der Kurve:

$$\begin{pmatrix} b_0 \\ c_0 \end{pmatrix} = \vec{p}(0) \qquad \text{Anfangspunkt}$$

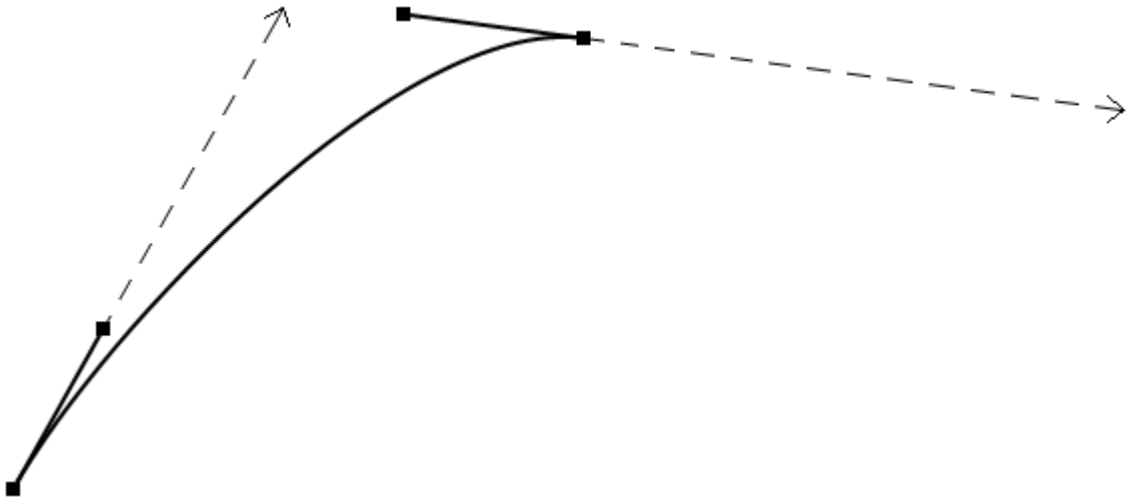
$$\begin{pmatrix} b_1 \\ c_1 \end{pmatrix} = \vec{p}(0) + \frac{1}{3}\vec{p}'(0) \qquad \text{Anfangspunkt + } \frac{1}{3}\text{Tangentenvektor}$$

$$\begin{pmatrix} b_2 \\ c_2 \end{pmatrix} = \vec{p}(1) - \frac{1}{3}\vec{p}'(1) \qquad \text{Endpunkt - } \frac{1}{3}\text{Tangentenvektor}$$

$$\begin{pmatrix} b_3 \\ c_3 \end{pmatrix} = \vec{p}(1) \qquad \text{Endpunkt}$$

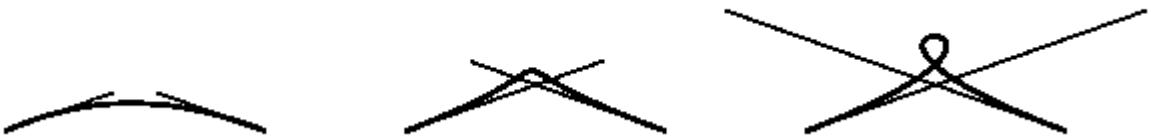
Diese Formeln bedeuten einerseits, dass man bei Kenntnis der Bézierpunkte (mit etwas Übung) den ungefähren Kurvenverlauf erahnen kann. Andererseits kann man aus anschaulichen Wunschvorstellungen vom Kurvenverlauf (Anfangs- und Endpunkt sowie Lage der zugehörigen Tangenten) sehr einfach die Bézierpunkte ermitteln.

Der praktische Umgang mit Béziersegmenten in Anwendungsprogrammen sieht fast immer so ähnlich wie in folgender Skizze aus:



Die (gestrichelten) Tangentenvektoren werden normalerweise **nicht** dargestellt, sondern nur die Bézierpunkte selbst und allenfalls noch die Verbindungsstrecken vom Anfangspunkt zum ersten Zwischenpunkt bzw. vom zweiten Zwischenpunkt zum Endpunkt. (Manchmal werden auch noch die beiden inneren Punkte verbunden - der entstehende Polygonzug stellt einen "vergröberten" Verlauf des Segments dar.) Üblicherweise kann man die Bézierpunkte mit der Maus "anfassen" und nach Belieben verschieben (siehe Beispiele im folgenden Abschnitt).

Bemerkungen:
(1) Aus der Form der Bézierdarstellung geht auch hervor, dass jeder Kurvenpunkt eine *konvexe Kombination* der vier Bézierpunkte ist (warum?). Die Kurve verläuft also immer innerhalb des von den Bézierpunkten gebildeten Vierecks.
(2) Wir wissen, dass die Länge des Tangentenvektors von der gewählten Parameterdarstellung abhängt und daher normalerweise keine direkte geometrische Bedeutung hat. Da die Parametrisierung eines Béziersegments aber fest vorgegeben ist, sind die Auswirkungen bloßer Längenänderungen des Tangentenvektors gut überschaubar: Je länger der Tangentenvektor ist, desto stärker ist die Kurve an die Tangente gebunden. Das sieht man an folgenden Beispielen, bei denen die Tangentenrichtung immer gleich bleibt und nur die Länge sukzessive vergrößert wird:



Im letzten (extremen) Beispiel zwingt man das Segment so stark an die Tangenten heran, dass es sich nur durch eine Schleifenbildung retten kann. Spätestens dann, wenn man es nicht mehr mit nur einem Segment zu tun hat, sondern viele Segmente zu einer Gesamtkurve zusammensetzt, sollte man es aber unbedingt vermeiden, die einzelnen Segmente derartig zu quälen.

4. Anwendungsbeispiele für Béziersegmente

Obwohl das einzelne Béziersegment noch nicht allzu viele Möglichkeiten bietet, gibt es auch dafür bereits praktische Anwendungen. Wir betrachten zwei Beispiele:

(A) MS Paint

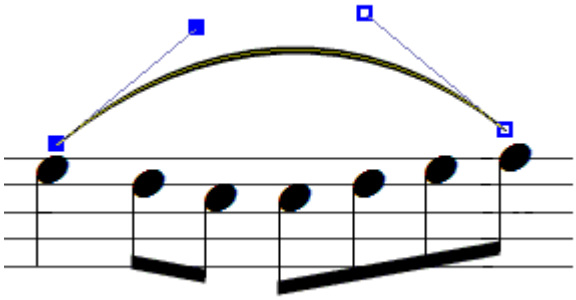
Zum Standard-Lieferumfang von Windows gehört das Zeichenprogramm **MS Paint**. Da es sich um ein einfaches Bitmap-Grafikprogramm handelt, kann es natürlich keine Bézierkurven speichern und verwalten. Das Kurvenwerkzeug benutzt aber Béziersegmente als Zeichenhilfe. Für jemanden, der sich nicht bereits mit Bézierkurven auskennt, ist die Bedienung nicht sehr intuitiv:

- Anklicken des Anfangspunktes, Ziehen mit gedrückter Maustaste, Loslassen am gewünschten Endpunkt. Gezeichnet wird dabei die Verbindungsstrecke.
- Anklicken des ersten "Steuerpunktes", das ist der zum Anfangspunkt gehörige innere Bézierpunkt. Solange man die Maustaste gedrückt hält, kann man diesen Punkt verschieben. Der Punkt selbst wird dabei nicht angezeigt, aber man sieht, wie sich die Kurve verformt.
- Anschließend kann man den zweiten Steuerpunkt genauso anklicken und verschieben, bis man mit der Kurve zufrieden ist.

Diese Schritte kann man nur genau einmal und in genau dieser Reihenfolge durchführen. Sobald ein Punkt einmal losgelassen wurde, kann man ihn nicht mehr erneut anfassen. Nach Loslassen des letzten Punktes ist die fertige Kurve dann (wie bei Bitmapgrafiken üblich) nur noch eine Menge von Pixeln, die nicht mehr als Objekt angewählt oder bearbeitet werden kann.

(B) Notensatz

Der volle Leistungsumfang eines einzelnen Béziersegments wird oft in Notensatzprogrammen für Bindebögen benutzt. Wenn man z. B. in **Capella** einen Bindebogen setzt und dann zum "Ändern" anwählt, erhält man folgendes typische Bild:



Durch Anfassen und Verschieben der Bézierpunkte lässt sich der Bindebogen nach Wunsch gestalten. Da Bindebögen normalerweise nur einfache Bögen sind und auch in Extremfällen höchstens einen einzelnen Wendepunkt benötigen, ist ein einzelnes Segment hier völlig ausreichend.

5. Bézierkurven aus mehreren Segmenten

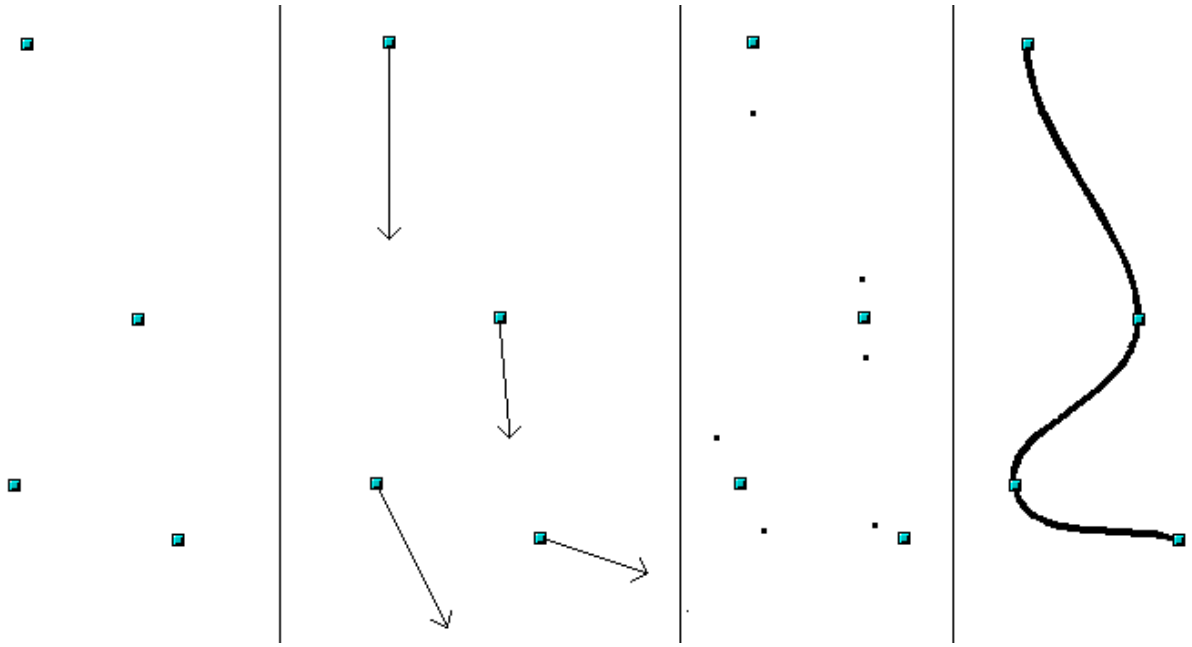
Die universelle Bedeutung der Bézierkurven für Grafikanwendungen entsteht erst dadurch, dass man solche Kurven aus einer beliebigen Anzahl einzelner Segmente zusammensetzen kann. Dieses Zusammensetzen erfolgt nach folgenden (naheliegenden) Grundsätzen:

- Der Endpunkt eines Segments ist gleichzeitig Anfangspunkt des nächstfolgenden.
- Die beiden in einem Punkt zusammenstoßenden Segmente haben dort denselben Tangentenvektor.

Die zweite Forderung stellt sicher, dass die entstehende Gesamtkurve "glatt" aussieht, d. h. dass die "Flickstellen" nicht auffallen.

Konkret heißt das, dass man eine gewisse Anzahl von (durchnumerierten) Punkten vorgibt, durch die die Kurve verlaufen soll. Diese Punkte heißen auch "Stützstellen". Außerdem wird in jedem Punkt der Tangentenvektor vorgegeben. Daraus können dann für jedes Segment die beiden

inneren Bézierpunkte nach den bekannten Formeln berechnet und die gesamte Kurve gezeichnet werden.



Solche Bézierkurven gehören zum Grundumfang aller Vektorgrafik-Programme. Das gilt nicht nur für professionelle CAD-Systeme u. ä., sondern auch schon für die Zeichenprogramme der gängigen Office-Pakete wie MS Office, Open Office, Lotus Smart Suite etc. Der Bedienkomfort und die Hilfetexte lassen allerdings meistens sehr zu wünschen übrig, so dass man ohne Vorkenntnisse über Bézierkurven erstmal ziemlich hilflos ist. (Selbst vor echten Fehlfunktionen ist man nie ganz sicher.) Andererseits lässt sich die Funktionsweise und Leistungsfähigkeit von Bézierkurven damit ohne großen Aufwand demonstrieren.

Wählt man in solch einem Programm eine Bézierkurve zur Bearbeitung an, so erhält man typischerweise eine Darstellung, in der zunächst nur die Kurve mit allen Stützstellen sichtbar ist, die weiteren Bézierpunkte (zwei pro Segment) jedoch nicht. Erst wenn man eine Stützstelle anwählt, werden die beiden vorwärts und rückwärts benachbarten Bézierpunkten angezeigt und üblicherweise auch durch ein Geradenstück verbunden. Wegen der Glattheits-Bedingung müssen diese drei Punkte ja immer eine Gerade bilden (nämlich die Tangente!) und beide Teile dieser "Hantel" müssen jeweils gleichlang sein.

Die Bearbeitungsmöglichkeiten bestehen dann darin, dass man entweder Stützstellen oder Hantelenden anfasst und verschiebt. Im letzteren Fall wird immer die gesamte Hantel so bewegt, dass der glatte Übergang erhalten bleibt.

Anmerkung: Als Faustregel gilt, dass man bei mehrsegmentigen Bézierkurven die besten Ergebnisse erhält, wenn die Tangentenvektoren ungefähr genauso lang sind wie die zugehörigen Segmente. Das heißt, dass die Bézierpunkte die Segmente in etwa dritteln. Das geht natürlich nur, wenn benachbarte Segmente ungefähr gleich lang sind. Daher findet man in der Literatur manchmal die Empfehlung, die Stützstellen von vornherein möglichst äquidistant auf der Kurve zu verteilen, was aber aus anderen Gründen oft unpraktikabel ist. Das direkte Aneinanderstoßen von sehr langen und sehr kurzen Segmenten sollte aber in der Tat vermieden werden.

In den genannten Programmen gibt es neben der oben beschriebenen Standard-Vorgehensweise meistens auch noch weitere Varianten. Dazu kann man den einzelnen Stützstellen einen "Punkttyp" zuordnen. In OpenOfficeDraw gibt es etwa die folgenden drei Möglichkeiten:

- "Symmetrischer Übergang": Das entspricht genau dem oben beschriebenen Verfahren.
- "Glatter Übergang": Dabei hat man die zusätzliche Freiheit, beide Hantelseiten unterschiedlich lang zu machen. Das heißt, dass der Tangentenvektor beim Übergang von einem Segment ins nächste zwar die Richtung beibehält, aber seine Länge sprunghaft ändern darf. (Obwohl es also noch eine eindeutige Tangente gibt, wäre das im Sprachgebrauch der Differentialgeometrie bereits nicht mehr "glatt"!)
- "Eckpunkt": Hier darf die Hantel sogar geknickt sein. Das heißt, dass die Tangentenvektoren der beiden zusammenstoßenden Segmente völlig unabhängig voneinander festgelegt werden können und auch die Kurve dort einen Knick bekommt. (Als Mathematiker würde man in diesem Fall schon lieber von zwei Bézierkurven sprechen, die nur "zufällig" aneinanderhängen. Für die Praxis ist es aber natürlich sehr sinnvoll, das gesamte Gebilde mit allen Knicken als ein einziges Objekt verwalten zu können.)

Beachte: Alle diese Varianten (und auch die im nächsten Abschnitt beschriebene) unterscheiden sich nur in der Art, wie die Bézierpunkte ermittelt werden. Speicherung und Anzeige der resultierenden Kurve erfolgen dagegen immer gleichartig. **Für eine Bézierkurve aus n Segmenten werden genau $3n + 1$ Punkte abgelegt, nämlich die $n + 1$ Stützstellen und $2n$ weitere Bézierpunkte (zwei pro Segment).** Das Zeichnen erfolgt dann durch Aufruf der in Abschnitt 3 angegebenen Funktionen für die x- und y-Komponente der Parameterdarstellung.

6. Spline-Interpolations-Kurven

Im Zeichenprogramm von MS Office ist sogar noch ein weiterer wichtiger Typ spezieller Bézierkurven implementiert. Dazu kann man den Stützstellen außer "glatt", "symmetrisch" und "Ecke" noch einen vierten Typ zuordnen, nämlich "AutoPunkt". Wenn man einen Punkt zum "AutoPunkt" erklärt, verschwindet die Hantel an dieser Stelle komplett. Die entsprechenden Bézierpunkte existieren intern natürlich weiter, sie dürfen aber nicht mehr vom Benutzer verändert werden. Vielmehr werden sie automatisch nach gewissen Kriterien berechnet, die einen besonders "guten" Kurvenverlauf ergeben sollen (und es meistens auch tun).

In den neueren Versionen von MS Office ist das sogar die Standardeinstellung, d. h. alle neuen Bézierkurven werden nur mit diesem Punkttyp angelegt. Das hat den Vorteil, dass die Ersteingabe, die sonst oft sehr kompliziert und verwirrend ist (siehe OpenOffice), sehr einfach wird: Man klickt einfach der Reihe nach auf die gewünschten Stützstellen und bekommt sofort eine brauchbare Kurve durch alle diese Punkte. Die Stützstellen lassen sich dann auch wie gewohnt verschieben, aber mit Tangenten wird man nicht weiter belästigt, was die meisten Benutzer wahrscheinlich eher als Vorteil sehen. (Das Programm soll ja auch kein professionelles CAD-System ersetzen, sondern nur zur Erstellung von hübschen PowerPoint-Folien u. ä. dienen.) Bei Bedarf kann man sich die weitergehenden Bearbeitungsmöglichkeiten natürlich jederzeit wieder verschaffen, indem man einzelne (oder alle) Punkte wieder auf einen anderen Typ setzt.

Diese speziellen Typen von Bézierkurven heißen **Spline-Interpolations-Kurven** (engl. *spline interpolation curve*s). Sie sollen in gewissem Sinne jeweils die "schönste aller möglichen Kurven" durch die gegebenen Punkte liefern. Das dafür benutzte Berechnungsverfahren beruht auf folgenden Kriterien:

- Die Forderung, dass die Tangentenvektoren der beiden in einem Punkt zusammentreffenden Segmente gleich sein sollen, wird beibehalten.

- Die Bézierpunkte werden aber außerdem so berechnet, dass auch die zweiten Ableitungen beider Segment im gemeinsamen Punkt übereinstimmen.

Die entstehenden Kurven sind also durchgehend zweimal differenzierbar. Anschaulich bedeutet das einen nahtlosen Übergang der Krümmung von einem Segment ins nächste. Mit bloßem Auge ist das lokal zwar nicht zu erkennen, auf den ästhetischen Gesamteindruck wirkt es sich oft aber doch sehr positiv aus. In den beiden Endpunkten, in denen die genannten Bedingungen natürlich nicht greifen, fordert man meistens, dass die Krümmung Null werden soll (es gibt aber auch andere Varianten).

Ein Nebeneffekt dieses Verfahrens besteht darin, dass die einzelnen Segmente nicht mehr völlig unabhängig voneinander sind. Änderungen an einer Stützstelle wirken sich jetzt im Prinzip auf alle Segmente aus. Für die eingangs erwähnte "lokale Modellierbarkeit" ist das dennoch kein gravierender Nachteil, da die Auswirkungen auf weiter entfernte Segmente nur sehr geringfügig sind. Die Berechnung der Bézierpunkte wird allerdings wesentlich aufwendiger, da jeder einzelne jetzt von allen Stützstellen abhängig ist. Letztendlich handelt es sich aber doch nur um das Lösen eines linearen Gleichungssystems, das außerdem eine besonders einfache Struktur hat. Es handelt sich um ein *tridiagonales LGS*, bei dem nur in der Hauptdiagonale und den Stellen direkt links und rechts daneben Werte $\neq 0$ stehen. Für solche LGS gibt es spezielle Algorithmen, bei denen die Rechenzeit nur linear mit der Anzahl der Unbekannten wächst (bei allgemeinen LGS geht das quadratisch!). Außerdem gehen die Stützstellen nur in die rechte Seite ein, während die Koeffizientenmatrix ausschließlich aus verfahrensbedingten Konstanten besteht. Das Lösen dieses LGS macht also auch bei sehr vielen Stützstellen keine Probleme - weder bezüglich der Rechenzeit noch bezüglich der numerischen Stabilität.

Anmerkungen zum Namen *Spline-Interpolations-Kurven*:

- Das Wort "Interpolation" bezieht sich auf die Tatsache, dass man **nur** die Stützstellen vorgibt und diese dann durch die Kurve verbinden oder "interpolieren" lässt. In der Tat nutzt man diese Vorgehensweise auch für die klassische Interpolation, bei der einzelne Werte eines Funktionsverlaufs vorgegeben sind und eine interpolierende Funktion gesucht wird. Für viele ingenieurwissenschaftliche Anwendungen ist die Spline-Interpolation in diesem Sinne von mindestens ebenso großer Bedeutung wie die allgemeinen Bézierkurven für Grafikanwendungen. Die "schwächeren" Varianten (ohne zweimalige Differenzierbarkeit) werden in diesem Zusammenhang dann oft etwas abfällig als "Subsplines" bezeichnet.
- Das englische Wort *spline* bezeichnet ursprünglich elastische Streifen aus Metall oder Holz, die als Bauelemente für flexible Konstruktionen eingesetzt werden. Wenn man einen solchen Streifen an einzelnen Punkten fixiert und ansonsten sich selbst überlässt, nimmt er eine Form an, die physikalisch dadurch charakterisiert ist, dass die innere Spannung minimiert wird. Mathematisch läuft das genau auf die genannte Bedingung an die zweiten Ableitungen hinaus. (Ehe sich die Bezeichnung "Spline" auch im Deutschen als Fachbegriff eingebürgert hatte, wurde sie manchmal als "Biegelinie" übersetzt.) Es handelt sich also eigentlich um eine physikalische Optimierung, die aber offenbar auch ästhetischen Wert hat.

7. Abschließender Hinweis

Das erwähnte MS Paint hat einen moderneren Nachfolger, der **Paint.NET** heißt und von Informatikstudenten der Washington State University entwickelt wurde (gesponsort von Microsoft). Es steht nicht nur auf der Microsoft-Website sondern auch z. B. auf *www.shareware.de* zum kostenlosen Download zur Verfügung.

Das Kurvenwerkzeug funktioniert im Prinzip wie im alten MS Paint, aber sowohl die Bedienung als auch die Funktionalität wurden stark verbessert. Man beginnt wieder mit dem Zeichnen eines

Geradenstücks vom Anfangs- zum Endpunkt. Diese beiden Punkte und zwei weitere dazwischen sind dann bereits markiert. Man kann diese Markierungen mit der **rechten**(!) Maustaste anfassen und verschieben, wodurch wieder ein Béziersegment gestaltet wird. Allerdings kann man jetzt alle vier Punkte beliebig oft neu anfassen und bearbeiten. Sie verschwinden erst nach Abschluss des Vorgangs. (Da es sich weiterhin um ein Bitmap-Programm handelt, ist die Kurve danach natürlich wieder nur noch als Ansammlung von Pixeln vorhanden.)

Wenn man die genannten Punkte dagegen mit der **linken** Maustaste anfasst und bewegt, entsteht eine Spline-Interpolations-Kurve durch die vier Punkte. Für die meisten unvorbelasteten Benutzer ist dies sicher ein natürlicheres Vorgehen zur Modellierung einer Kurve (daher wohl auch die linke Maustaste). Die Gestaltungsmöglichkeiten sind zwar einerseits eingeschränkt, da man keinen Einfluss mehr auf die Tangenten hat, dafür hat man andererseits aber auch drei Segmente (statt nur einem) zur Verfügung.