Mathematische Aspekte von fig2lat

Dirk Krause

3. September 2016

Inhaltsverzeichnis

1 Gedrehte Ellipsen

1.1 Abstand eines Punktes zu einer Geraden durch den Nullpunkt

Für Berechnungen an gedrehten Ellipsen wird eine Formel für den Abstand l eines Punktes $P(x_P; y_P)$ zu einer Geraden durch den Koordinatenursprung mit dem Winkel β benötigt.

Für die Gerade gilt:

$$\tilde{k}: \quad \vec{r}(t) = \begin{pmatrix} t\cos\beta \\ t\sin\beta \end{pmatrix}$$

Für den Abstand zwischen P und einem durch t gegebenen Geradenpunkt gilt:

$$l(t) = \sqrt{(t\cos\beta - x_\mathrm{P})^2 + (t\sin\beta - y_\mathrm{P})^2}$$

Der Abstand des Punktes zur Geraden ist gleich der Länge des Lotes vom Punkt auf die Gerade. Da das Lot rechtwinklig zur Gerade steht, muss das Skalarprodukt von Lotvektor und Geradenvektor 0 ergeben.

$$\begin{aligned}
0 &= (\vec{r}(t_\mathrm{L}) - \vec{r_\mathrm{P}}) \cdot \vec{r}(t_\mathrm{L}) \\
&= \begin{pmatrix} t_\mathrm{L}\cos\beta - x_\mathrm{P} \\ t_\mathrm{L}\sin\beta - y_\mathrm{P} \end{pmatrix} \cdot \begin{pmatrix} t_\mathrm{L}\cos\beta \\ t_\mathrm{L}\sin\beta \end{pmatrix} \\
&= t_\mathrm{L}^2\cos^2\beta - x_\mathrm{P}t_\mathrm{L}\cos\beta + t_\mathrm{L}^2\sin^2\beta - y_\mathrm{P}t_\mathrm{L}\sin\beta \\
&= t_\mathrm{L}^2 - t_\mathrm{L}(x_\mathrm{P}\cos\beta + y_\mathrm{P}\sin\beta) \\
&= t_\mathrm{L}(t_\mathrm{L} - (x_\mathrm{P}\cos\beta + y_\mathrm{P}\sin\beta))
\end{aligned}$$

Die Lösung

$$t_\mathrm{L} = x_\mathrm{P}\cos\beta + y_\mathrm{P}\sin\beta$$

ergibt beim Einsetzen die Länge des Lotes:

$$l = \sqrt{(y_P - (y_P\sin\beta + x_P\cos\beta)\sin\beta)^2 + (x_P - (y_P\sin\beta + x_P\cos\beta)\cos\beta)^2}$$

Die Scheinlösung $t_\mathrm{L} = 0$ ergibt zwar ebenfalls ein Skalarprodukt mit dem Wert 0 aber nicht wegen eines rechten Winkels zwischen den Vektoren sondern weil ein Vektor die Länge 0 hat.

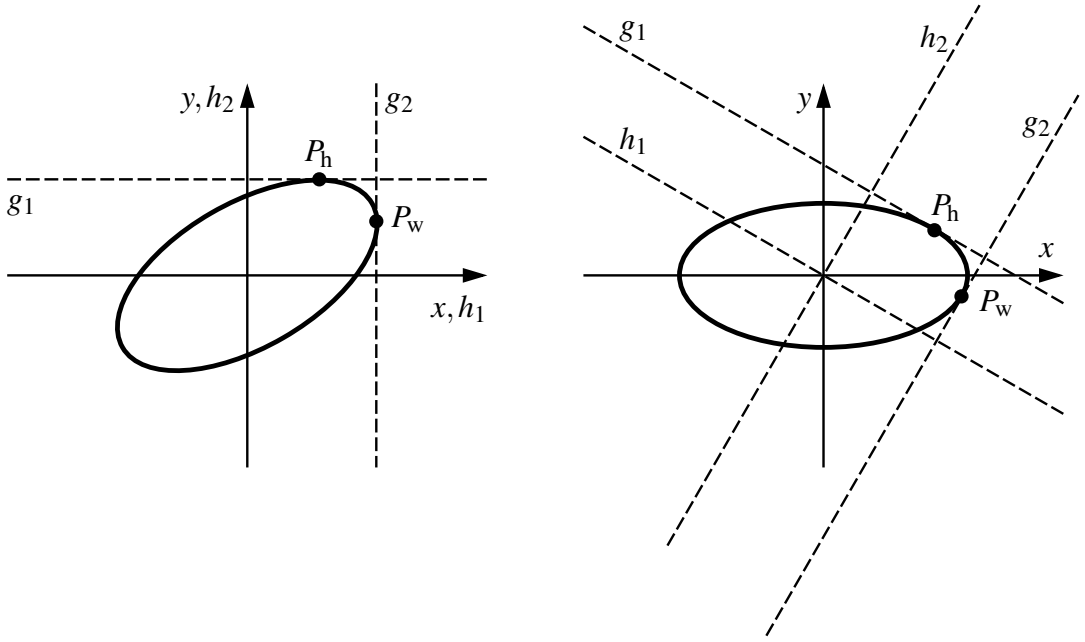

Abbildung 1: Berechnungen an der gedrehten Ellipse

1.2 Bounding-Box für gedrehte Ellipsen

Um die Bounding-Box einer um α gedrehten Ellipse zu bestimmen, wird die Höhe h des höchsten Punktes und die Weite w des am weitesten rechts befindlichen Punktes benötigt.
In der linken Darstellung ist g_1 die obere horizontale Tangente an die gedrehte Ellipse, g_2 die rechte vertikale Tangente. Die Höhe h ergibt sich als Abstand des Berührungspunktes P_h zur x-Achse, die Weite w ist der Abstand des Berührungspunktes P_w zur y-Achse.
Für die Berechnung wird die gesamte Anordnung im Koordinatenursprung um $-\alpha$ zurückgedreht, die Ellipse ist wieder an den Achsen ausgerichtet. Die Geraden g_1 und g_2 sind jetzt nicht mehr parallel zu den Achsen, die Hilfsgeraden h_1 und h_2 werden aus der Drehung der Achsen gewonnen. Die Höhe h ist nun der Abstand des Punktes P_h zur Hilfsgerade h_1 (ehemals x-Achse), die Weite w der Abstand des Punktes P_w zur Hilfsgerade h_2 (ehemals y-Achse).

Die Ellipsenkurve kann beschrieben werden durch:

$$\tilde{k}: \quad \overrightarrow{r}(t) = \begin{pmatrix} r_x \cos t \\ r_y \sin t \end{pmatrix} \qquad 0 \leq t \leq 2\pi$$

Für die Ableitungen nach t ergibt sich

$$\dot{\overrightarrow{r}}(t) = \frac{\mathrm{d}\overrightarrow{r}}{\mathrm{d}t} = \begin{pmatrix} -r_x \sin t \\ r_y \cos t \end{pmatrix}$$

Im Berührungspunkt P_h gilt

$$\tan(\pi - \alpha) = -\frac{r_y \cos t_h}{r_x \sin t_h}$$

Damit ergibt sich t_h zu:

$$t_h = \arctan -\frac{r_y}{r_x \tan(\pi - \alpha)}$$

Analog erhält man für P_w

$$\tan\left(\tfrac{\pi}{2} - \alpha\right) = -\frac{r_y \cos t_w}{r_x \sin t_w}$$
$$t_w = \arctan -\frac{r_y}{r_x \tan\left(\frac{\pi}{2} - \alpha\right)}$$

Mit t_h und t_w können die Punkte P_h und P_w berechnet werden. Dann ergibt sich h als Abstand von P_h zur Hilfsgeraden h_1 mit $\beta_1 = -\alpha$ und w als Abstand von P_w zur Hilfsgeraden h_2 mit $\beta_2 = \frac{\pi}{2} - \alpha$.

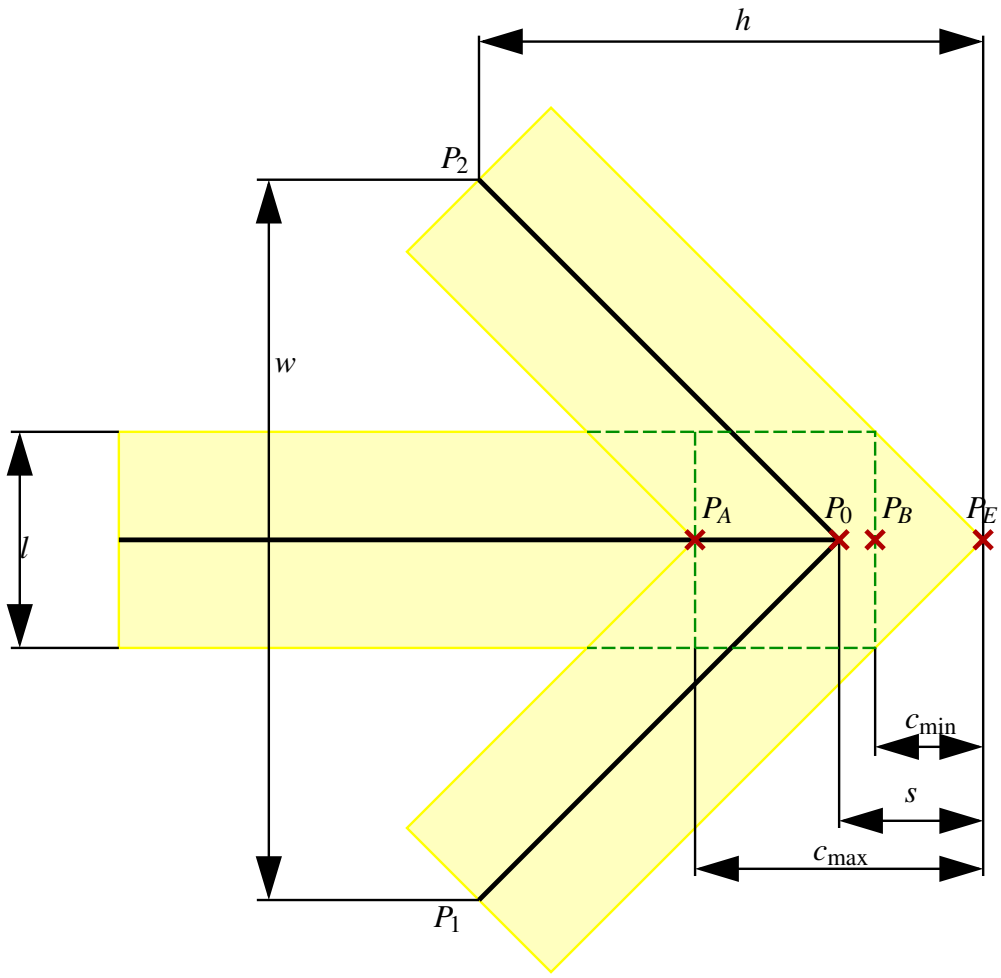

Abbildung 2: Längenkorrektur für Pfeilspitzen

2 Pfeilspitzen

2.1 Notwendigkeit der Längenkorrektur

Für Linien mit Pfeilspitzen gibt der in der Fig-Datei gegebene Endpunkt den Punkt an, in dem die Pfeilspitze endet. Die entsprechende Linie muss dann etwas kürzer gezeichnet werden.
Abb. 2 zeigt eine von links kommende Linie, die in Punkt P_0 endet. An diese Linie wird eine Pfeilspitze angefügt, die als Polylinie $P_1 \ldots P_0 \ldots P_2$ gezeichnet wird. Zum Zeichnen der Pfeilspitze wird dieselbe Linienbreite l verwendet wie für die eigentliche Linie. Wie man sieht, endet die Pfeilspitze im Punkt P_E, der um die Länge s von P_0 entfernt ist.

Bezeichnet man den Öffnungswinkel der Pfeilspitze (Winkel zwischen den zwei Schenkeln der Pfeilspitze) mit α, so erhält man:

$$\frac{l}{2} = s \cdot \sin\left(\frac{\alpha}{2}\right)$$

Sind Endpunkt P_E, Linienbreite l und Öffnungswinkel α vorgegeben, so muss der Punkt P_0 für das Zeichnen der Pfeilspitze den Abstand

$$s = \frac{\frac{l}{2}}{\sin\left(\frac{\alpha}{2}\right)} = \frac{l}{w}\sqrt{h^2 + \tfrac{1}{4}w^2}$$

zu P_E haben. Die Punkte P_1 und P_2 können dann über die Pfeilspitzenlänge und -breite berechnet werden.
Die eigentliche Linie – die noch vor der Pfeilspitze gezeichnet werden sollte – muss zwischen den Punkten P_A und P_B bzw. auf einem dieser Punkte enden. Die Entfernung von P_A zu P_E beträgt $2s$, die Entfernung von P_B zu P_E ergibt sich als $c_{\min}$ zu:

$$\frac{\frac{l}{2}}{c_{\min}} = \tan\left(\frac{\alpha}{2}\right) = \frac{w}{2h}$$

$$c_{\min} = \frac{\frac{l}{2}}{\tan\left(\frac{\alpha}{2}\right)} = \frac{l \cdot h}{w} \qquad c_{\min} \leq s$$

$$c_{\max} = 2s$$

$$c = \tfrac{1}{2}(c_{\max} + c_{\min})$$

$$\Delta c = \tfrac{1}{2}(c_{\max} - c_{\min})$$

2.2 Längenkorrektur für Pfeilspitzen an Linien

Die Längenkorrektur um die Länge c an einer Linie von $P_0(x_0, y_0)$ nach $P_1(x_1, y_1)$ wird mit Hilfe folgender Formeln durchgeführt:

$$l = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} \qquad q = \frac{l - c}{l}$$

für nichtnegative q gilt dann:

$$x_1' = x_0 + q(x_1 - x_0) \qquad y_1' = y_0 + q(y_1 - y_0)$$

3 X-Splines und Bezier-Splines

3.1 Kurven

Kurven in $\mathbb{R}^2$ können durch die Gleichung

$$\tilde{k}: \quad \vec{r}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} \qquad\qquad t_\mathrm{s} \leq t \leq t_\mathrm{e}$$

beschrieben werden.
Im Bereich der Computergraphik wird dabei häufig $t \in [0;1]$ für die gesamte Kurve oder für ein Kurvensegment verwendet.
Ein Spline ist eine Beschreibung für eine Kurve, dabei wird eine kontinuierliche Kurve durch einen Satz diskreter Werte bestimmt.

3.2 Kubische Bezier-Splines

3.2.1 Formel

Für kubische Bezier-Splines wird jedes Segment durch Startpunkt $P_s(x_s, y_s)$ und Endpunkt $P_e(x_e, y_e)$ sowie zwei Kontrollpunkte $P_{cs}(x_{cs}, y_{cs})$ und $P_{ce}(x_{ce}, y_{ce})$ beschrieben. Dabei bezeichnet P_{cs} den Kontrollpunkt „rechts“ vom Startpunkt. P_{ce} bezeichnet den Kontrollpunkt „links“ vom Endpunkt.
In der Literatur wird häufig auch die Schreibweise P_i und P_j für Start- und Endpunkt sowie P_{i+} bzw. P_{j-} oder auch P_i^+ bzw. P_j^- für die Kontrollpunkte verwendet.

$$\begin{aligned}
0 \leq t \leq 1 \\
\vec{r}(t) &= (1-t)^3 \cdot \vec{r_\mathrm{s}} + 3 \cdot (1-t)^2 \cdot t \cdot \vec{r_\mathrm{cs}} + 3 \cdot (1-t) \cdot t^2 \cdot \vec{r_\mathrm{ce}} + t^3 \cdot \vec{r_\mathrm{e}} \\
\vec{r}(t) &= (1-t)^3 \cdot \overrightarrow{0P_\mathrm{s}} + 3 \cdot (1-t)^2 \cdot t \cdot \overrightarrow{0P_\mathrm{cs}} + 3 \cdot (1-t) \cdot t^2 \cdot \overrightarrow{0P_\mathrm{ce}} + t^3 \cdot \overrightarrow{0P_\mathrm{e}} \\
\dot{\vec{r}} = \tfrac{\mathrm{d}}{\mathrm{d}t}\vec{r}(t) &= (-3+6t-3t^2)\vec{r_\mathrm{s}} + 3(1-4t+3t^2)\vec{r_\mathrm{cs}} + 3(2t-3t^2)\vec{r_\mathrm{ce}} + 3t^2\vec{r_\mathrm{e}}
\end{aligned}$$

An den Randpunkten $t = 0$ und $t = 1$ gilt:

$$\begin{aligned}
\dot{\vec{r}}(0) &= -3\vec{r_\mathrm{s}} + 3\vec{r_\mathrm{cs}} \\
\dot{\vec{r}}(1) &= -3\vec{r_\mathrm{ce}} + 3\vec{r_\mathrm{e}}
\end{aligned}$$

Dies kann umgestellt werden, so dass bei bekannten Anstiegen in Start- und Endpunkt die Kontrollpunkte berechnet werden können:

$$\begin{aligned}
\vec{r_\mathrm{cs}} &= \vec{r_\mathrm{s}} + \tfrac{1}{3}\dot{\vec{r}}(0) \\
\vec{r_\mathrm{ce}} &= \vec{r_\mathrm{e}} - \tfrac{1}{3}\dot{\vec{r}}(1)
\end{aligned}$$

Ein Nachteil der Bezier-Splines ist, dass es für unerfahrene Nutzer schwierig ist, abzuschätzen, wie sich eine Manipulation der Kontrollpunkte auf den Kurvenverlauf auswirkt.

3.2.2 Bounding-Box von Bezier-Splines

Um die Bounding-Box eines Bezier-Spline-Segmentes zu bestimmen, müssen neben den Endpunkten möglicherweise auch Extremstellen der x- und y-Werte berücksichtigt werden. Die Extremstellen für x werden berechnet mit

$$0 = \left.\frac{\mathrm{d}x}{\mathrm{d}t}\right|_{t=t_\mathrm{e}}$$

Dies führt zu

$$t_\mathrm{e} = \frac{2x_\mathrm{cs} - x_\mathrm{ce} - x_\mathrm{s} \pm \sqrt{(x_\mathrm{ce} - 2x_\mathrm{cs} + x_\mathrm{s})^2 - (x_\mathrm{cs} - x_\mathrm{s})(x_\mathrm{e} - 3x_\mathrm{ce} + 3x_\mathrm{cs} - x_\mathrm{s})}}{x_\mathrm{e} - 3x_\mathrm{ce} + 3x_\mathrm{cs} - x_\mathrm{s}}$$

Die t_e-Werte mit

$$0 < t_\mathrm{e} < 1$$

müssen bei der Suche nach Extremstellen mit berücksichtigt werden.
Die Berechnung der Extremstellen für y erfolgt analog.

3.3 X-Splines

Cross-splines (X-Splines) wurden 1995 durch Schlick und Blanc vorgestellt. X-Splines werden hier nicht im Detail erklärt, hierzu sei auf die entsprechenden SIGGRAPH-Veröffentlichungen verwiesen, siehe [CB95]. Die Darstellung an dieser Stelle beschränkt sich auf die Verwendung von X-Splines in FIG-Dateien und die Behandlung im Programm fig2lat.[1]

Eine Funktion $f(u)$ mit dem Parameter p wird als Gewichtsfunktion für die Approximation verwendet:

$$f(u,p) = \begin{cases} u^3\Big(10-p+(2p-15)u+(6-p)u^2\Big) & \text{für } u \in [0;1] \\ 0 & \text{ansonsten} \end{cases}$$
$$= \begin{cases} (6-p)u^5+(2p-15)u^4+(10-p)u^3 & \text{für } u \in [0;1] \\ 0 & \text{ansonsten} \end{cases}$$

Zwei Teilfunktionen mit den Parametern p und q werden verwendet, um die Gewichtsfunktion $e(u)$ für die Interpolation zu bilden:

$$e(u,p,q) = \begin{cases} qu+2qu^2+(10-12q-p)u^3 & \\ \quad +(2p+14q-15)u^4+(6-5q-p)u^5 & \text{für } u \in [0;1] \\ qu+2qu^2-2qu^4-qu^5 & \text{für } u \in [-1;0) \\ 0 & \text{ansonsten} \end{cases}$$

Die Ableitung der Funktionen ergibt:

$$\frac{\partial f}{\partial u} = \begin{cases} 5(6-p)u^4+4(2p-15)u^3+3(10-p)u^2 & \text{für } u \in [0;1] \\ 0 & \text{ansonsten} \end{cases}$$

$$\frac{\partial e}{\partial u} = \begin{cases} 5(6-5q-p)u^4+4(2p+14q-15)u^3 & \\ \quad +3(10-12q-p)u^2+4qu+q & \text{für } u \in [0;1] \\ q+4qu-8qu^3-5qu^4 & \text{für } u \in [-1;0) \\ 0 & \text{ansonsten} \end{cases}$$

[1] In [CB95] lautet Gleichung (20) für die Interpolationsfunktion: $h(u) = qu+2qu^2-2qu^4+qu^5$. Dies ist meiner Meinung nach fehlerhaft und muss stattdessen $h(u) = qu+2qu^2-2qu^4-qu^5$ lauten, wie es auch in XFig und fig2dev implementiert ist. Die Version mit + würde auch die als Gleichungssystem (19) aufgestellten Forderungen nicht erfüllen, insbesondere die letzte nicht.

Bei der Verwendung von X-Splines in FIG-Dateien wird davon ausgegangen, dass äquidistante Kontrollpunkte vorliegen, fig2lat setzt hierfür $\Delta t = 1$.
Ein X-Spline wird durch n Kontrollpunkte mit Indizes im Bereich von 0 bis $n-1$ beschrieben, dabei ist jedem Kontrollpunkt P_k am Zeitpunkt $t = T_k$ ein zusätzlicher Parameter s_k zugeordnet. Es gilt $-1 \leq s_k \leq 1$.
Punkte mit $-1 \leq s_k < 0$ sind dabei Interpolationspunkte, Punkte mit $0 \leq s_k \leq 1$ sind Approximationspunkte.
Zu jedem Kontrollpunkt P_k gehört eine linksseitige Gewichtsfunktion (blending function) F_k^- und eine rechtsseitige Gewichtsfunktion F_k^+. Diese Gewichtsfunktionen geben an, wie stark der Kontrollpunkt sich auf den Verlauf in den jeweils angrenzenden Segmenten auswirkt. Die Gewichtsfunktionen erstrecken sich jeweils über bis zu 2 Segmente, ein Kontrollpunkt kann somit also insgesamt 4 Segmente beeinflussen.
Ein Kurvensegment wird jeweils von den zwei Kontrollpunkten an den Segmentenden und von deren Nachbarn beeinflusst.
Der Wert s_k eines jeden Kontrollpunktes P_k legt die Art und die Parameter der rechts- und linksseitigen Gewichtsfunktionen der *benachbarten* Punkte fest (*nicht* aber des Punktes P_k selbst).

Für $0 \leq s_k \leq 1$ gilt:

$$T_{k-1}^+ = T_k + s_k \qquad \text{Ende des Wirkungsbereiches}$$

$$p_{k-1}^+ = 2(T_{k-1}^+ - T_{k-1})^2 \qquad \text{Parameter in der Gewichtsfunktion}$$

$$F_{k-1}^+(t) = f\left(\frac{t - T_{k-1}^+}{T_{k-1} - T_{k-1}^+}, p_{k-1}^+\right) \qquad \text{Gewichtsfunktion linker Nachbar}$$

$$T_{k+1}^- = T_k - s_k$$

$$p_{k+1}^- = 2(T_{k+1}^- - T_{k+1})^2$$

$$F_{k+1}^-(t) = f\left(\frac{t - T_{k+1}^-}{T_{k+1} - T_{k+1}^-}, p_{k+1}^-\right)$$

Für $-1 \leq s_k < 0$ gilt[2]:

$$
\begin{aligned}
p^+_{k-1} &= 2 \\
q^+_{k-1} &= -\frac{s_k}{2} \\
F^+_{k-1}(t) &= e\left(T_k - t, p^+_{k-1}, q^+_{k-1}\right) \\
p^-_{k+1} &= 2 \\
q^-_{k+1} &= -\frac{s_k}{2} \\
F^-_{k+1}(t) &= e\left(t - T_k, p^-_{k+1}, q^-_{k+1}\right)
\end{aligned}
$$

Nachdem für alle Punkte P_k die Funktionen F^-_k und F^+_k festgelegt wurden, können für jedes Kurvensegment mit den Randpunkten P_i und P_{i+1} Kurvenpunkte berechnet werden mit

$$
\begin{aligned}
z_x(t) &= x_{i-1}F^+_{i-1}(t) + x_i F^+_i(t) + x_{i+1}F^-_{i+1}(t) + x_{i+2}F^-_{i+2}(t) \\
z_y(t) &= y_{i-1}F^+_{i-1}(t) + y_i F^+_i(t) + y_{i+1}F^-_{i+1}(t) + y_{i+2}F^-_{i+2}(t) \\
n(t) &= F^+_{i-1}(t) + F^+_i(t) + F^-_{i+1}(t) + F^-_{i+2}(t) \\
z'_x(t) = \frac{\mathrm{d}z_x}{\mathrm{d}t} &= x_{i-1}\frac{\partial F^+_{i-1}(t)}{\partial t} + x_i\frac{\partial F^+_i(t)}{\partial t} + x_{i+1}\frac{\partial F^-_{i+1}(t)}{\partial t} + x_{i+2}\frac{\partial F^-_{i+2}(t)}{\partial t} \\
z'_y(t) = \frac{\mathrm{d}z_y}{\mathrm{d}t} &= y_{i-1}\frac{\partial F^+_{i-1}(t)}{\partial t} + y_i\frac{\partial F^+_i(t)}{\partial t} + y_{i+1}\frac{\partial F^-_{i+1}(t)}{\partial t} + y_{i+2}\frac{\partial F^-_{i+2}(t)}{\partial t} \\
n'(t) &= \frac{\partial F^+_{i-1}(t)}{\partial t} + \frac{\partial F^+_i(t)}{\partial t} + \frac{\partial F^-_{i+1}(t)}{\partial t} + \frac{\partial F^-_{i+2}(t)}{\partial t} \\
x(t) &= \frac{z_x(t)}{n(t)} \\
y(t) &= \frac{z_y(t)}{n(t)} \\
x'(t) = \frac{\mathrm{d}x}{\mathrm{d}t} &= \frac{n(t)\cdot z'_x(t) - z_x(t)\cdot n'(t)}{n^2(t)} \\
y'(t) = \frac{\mathrm{d}y}{\mathrm{d}t} &= \frac{n(t)\cdot z'_y(t) - z_y(t)\cdot n'(t)}{n^2(t)}
\end{aligned}
$$

[2] In fig2lat wird $q = -s_k/2$ verwendet, wie in [CB95] beschrieben. Dort wird $0 \leq q \leq \frac{1}{2}$ gefordert, um unerwünschte Oszillationen zu vermeiden. In XFig und fig2dev wird jedoch $q = -s_k$ verwendet, wodurch auch q-Werte mit $q > \frac{1}{2}$ möglich werden. Das Ergebnis ist ein übermäßig starkes Ausschwingen bei interpolierten Splines und z. T. Oszillationen, beispielsweise bei einer Anordnung entsprechend Abbildung 15 in [CB95] zwischen den Punkten P_2 und P_3.

Für das erste X-Spline-Segment entfällt der erste Summand, da der Anfangspunkt keinen „linken“ Nachbarpunkt hat. Analog entfällt für das letzte X-Spline-Segment der jeweils letzte Summand.
Für die Vereinfachung des Berechnungsprogrammes wird jedes X-Spline-Segment so in eine Anordnung mit vier Punkten A, B, C und D gelegt, dass der Kontrollpunkt für den Anfang des Segments auf B liegt und der Kontrollpunkt für den Endpunkt des Segmentes auf C. A entspricht dann dem „linken“ benachbartem Kontrollpunkt, D dem „rechten“.

3.4 Annäherung von X-Splines durch Bezier-Splines

Die für fig2lat vorgesehenen Ausgabeformate unterstützen Bezier-Splines. Es ist erforderlich, X-Splines näherungsweise durch Bezier-Splines nachzubilden.
Zunächst werden Kurvenpunkte berechnet, die den Kontrollpunkten entsprechen (d.h. Kurvenpunkte mit $t = T_1$, $t = T_2 \ldots$). Um die Kontrollpunkte für die Bezier-Splines zu errechnen, wird der Anstieg in den Kurvenpunkten benötigt. Da X-Splines Knicke in den Kurvenpunkten aufweisen können, die den Kontrollpunkten entsprechen, wird sowohl ein „linksseitiger“ als auch ein „rechtsseitiger“ Anstieg berechnet.
X-Splines sind Kurven fünften Grades – und weichen somit von Bezier-Splines dritten Grades ab – es ist deshalb sinnvoll, jedes X-Spline-Segment durch mehrere Bezier-Spline-Segmente darzustellen. Zu diesem Zweck werden Kurvenpunkte und Anstiege auch zwischen den im vorangegangenem Schritt berechneten Kurvenpunkten berechnet. Da hier keine Knicke auftreten, genügt jeweils eine Berechnung des Anstieges.
Erfolgt die Darstellung jedes X-Spline-Segmentes durch mehrere (m) Bezier-Spline-Segmente, ist eine Korrektur der Anstiege erforderlich. Bezier-Splines beginnen und enden jeweils bei $t = 0$ und $t = 1$. Die bisher errechneten Anstiege beziehen sich aber auf Bereiche mit $t = t_s$ und $t = t_s + \frac{1}{m}$. Die Verwendung der berechneten Kurvenpunkte als Anfangs- und Endpunkte von Bezier-Splines entspricht also einer Streckung entlang der t-Achse um den Faktor m. Damit geht eine Verringerung des Anstieges einher, alle bisher berechneten Anstiege müssen also korrigiert werden, indem sie durch m dividiert werden.

3.5 Längenberechnung an Splines

Die Längenberechung einer Kurve erfolgt üblicherweise durch das Integral

$$l = \int \sqrt{\left(\frac{\mathrm{d}x}{\mathrm{d}t}\right)^2 + \left(\frac{\mathrm{d}y}{\mathrm{d}t}\right)^2}\,\mathrm{d}t$$

Da ich weder für X-Splines noch für Bezier-Splines eine analytische Lösung des Problemes gefunden habe, wird die numerische Näherung

$$l \approx \sum_{i=1}^{n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$$

verwendet.
Das Spline oder das Spline-Segment wird in viele kleine Intervalle unterteilt, für jedes Intervall werden Anfangs- und Endpunkt berechnet, die daraus gewonnenen Intervalllängen werden aufaddiert.
In einer Iteration wird bei jedem Durchlauf die Intervallanzahl erhöht (verdoppelt). Die Iteration wird erfolgreich beendet, falls die Differenz zwischen dem Ergebnis des aktuellen Durchlaufes und dem Ergebnis des vorangegangenen Durchlaufes eine bestimmte Genauigkeitsschranke unterschreitet. Wurde nach einer bestimmten Anzahl Durchläufe die Iteration nicht erfolgreich beendet, so wird sie erfolglos abgebrochen.

3.6 Längenkorrektur für Pfeilspitzen an Splines

Die Längenkorrektur an Splines ist vergleichsweise schwierig, da hier die Formeln nicht analytisch so umgestellt werden können, dass man ein t für einen vorgegebenen Abstand zum Endpunkt erhält.
Um ein t zu bestimmen, für das die Länge vom zugehörigen Punkt bis zum Spline-Ende den vorgegebenen Wert s hat, wird das Intervallschachtelungsverfahren – auch Halbierungsverfahren genannt – benutzt. Zunächst wird überprüft, ob s kleiner ist als die Gesamtlänge des Splines. Falls ja, werden als Startwerte für die Intervallgrenzen das t für den ersten Punkt (0) und den letzten Punkt $(n-1)$ benutzt.

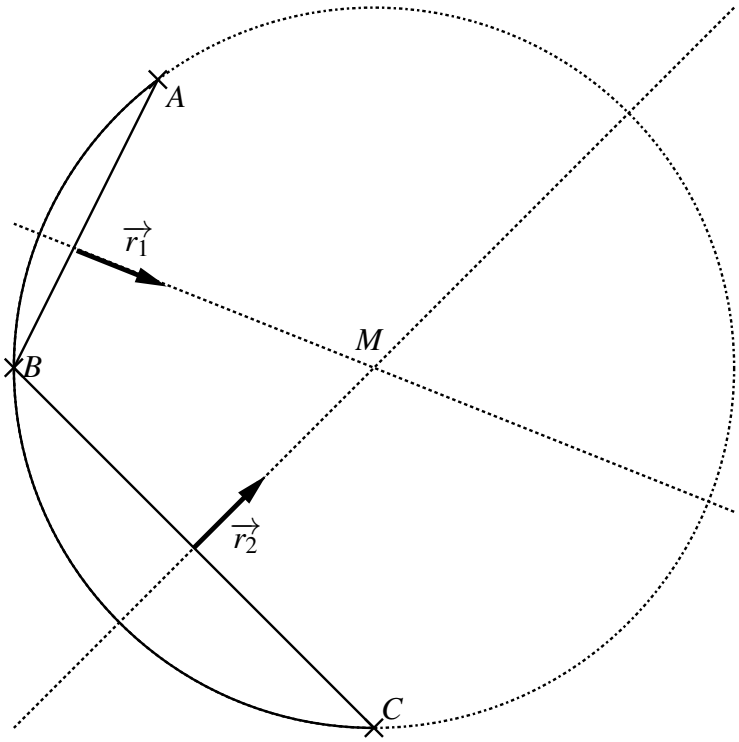

Abbildung 3: Berechnungen am Kreisbogen

4 Kreisbögen

4.1 Berechnung von Mittelpunkt und Radius

Kreisbögen sind in Fig-Dateien durch drei Punkte gegeben, die hier mit $A(x_A, y_A)$, $B(x_B, y_B)$ und $C(x_C, y_C)$ bezeichnet werden. Mit $\overrightarrow{r_A}$, $\overrightarrow{r_B}$ und $\overrightarrow{r_C}$ werden die Vektoren vom Koordinatenursprung zu den drei Punkten bezeichnet. A und C sind die Endpunkte des Bogens, B ist ein innerer Punkt.
Zum Zeichnen der Kreise und zur Berechnung der Bounding-Box werden der Mittelpunkt M und der Radius r benötigt.

Der Mittelpunkt $M(x_M, y_M)$ ergibt sich als Schnittpunkt der Mittelsenkrechten auf den Strecken $\overline{BA}$ und $\overline{BC}$

$$g_1: \quad \overrightarrow{r} = \overrightarrow{r_B} + \tfrac{1}{2}\overrightarrow{BA} + t\,\overrightarrow{r_1} \qquad t \in \mathbb{R}$$
$$g_2: \quad \overrightarrow{r} = \overrightarrow{r_B} + \tfrac{1}{2}\overrightarrow{BC} + s\,\overrightarrow{r_2} \qquad s \in \mathbb{R}$$

Da die Mittelsenkrechten senkrecht auf den jeweiligen Strecken stehen, muss gelten:

$$0 = \overrightarrow{r_1} \cdot \overrightarrow{AB} \qquad\qquad 0 = \overrightarrow{r_2} \cdot \overrightarrow{CB}$$
$$0 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \begin{pmatrix} x_B - x_A \\ y_B - y_A \end{pmatrix} \qquad 0 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} \begin{pmatrix} x_B - x_C \\ y_B - y_C \end{pmatrix}$$

Dies wird erfüllt durch:

$$x_1 = y_B - y_A \qquad\qquad x_2 = y_B - y_C$$
$$y_1 = x_A - x_B \qquad\qquad y_2 = x_C - x_B$$

Somit ergibt sich für die Geradengleichungen:

$$g_1: \quad \overrightarrow{r} = \tfrac{1}{2}\begin{pmatrix} x_A + x_B \\ y_A + y_B \end{pmatrix} + t \begin{pmatrix} y_B - y_A \\ x_A - x_B \end{pmatrix}$$
$$g_2: \quad \overrightarrow{r} = \tfrac{1}{2}\begin{pmatrix} x_C + x_B \\ y_C + y_B \end{pmatrix} + s \begin{pmatrix} y_B - y_C \\ x_C - x_B \end{pmatrix}$$

Im Schnittpunkt M beider Geraden gilt:

$$x_B + \tfrac{1}{2}(x_A - x_B) + t_M(y_B - y_A) = x_B + \tfrac{1}{2}(x_C - x_B) + s_M(y_B - y_C)$$
$$y_B + \tfrac{1}{2}(y_A - y_B) + t_M(x_B - x_A) = y_B + \tfrac{1}{2}(y_C - y_B) + s_M(x_B - x_C)$$
$$\tfrac{1}{2}(x_A + x_B) + t_M(y_B - y_A) = \tfrac{1}{2}(x_C + x_B) + s_M(y_B - y_C)$$
$$\tfrac{1}{2}(y_A + y_B) + t_M(x_B - x_A) = \tfrac{1}{2}(y_C + y_B) + s_M(x_B - x_C)$$
$$t_M(y_B - y_A) + s_M(y_C - y_B) = \tfrac{1}{2}(x_C - x_A)$$
$$t_M(x_A - x_B) + s_M(x_B - x_C) = \tfrac{1}{2}(y_C - y_A)$$

Dieses Gleichungssystem hat entweder

- genau eine Lösung

$$t_M = \frac{\frac{1}{2}\Big((x_B - x_C)(x_C - x_A) - (y_C - y_B)(y_C - y_A)\Big)}{(y_B - y_A)(x_B - x_C) - (y_C - y_B)(x_A - x_B)}$$

- keine Lösung (die Punkte liegen auf einer Geraden) oder
- unendlich viele Lösungen (zwei oder drei Punkte sind identisch).

In den beiden letzten Fällen hat der Nenner den Wert 0. Ist der Nenner positiv, wird der Kreisbogen im mathematisch positiven Umlaufsinn gezeichnet. Ist der Nenner negativ, wird im mathematisch negativen Umlaufsinn gezeichnet.

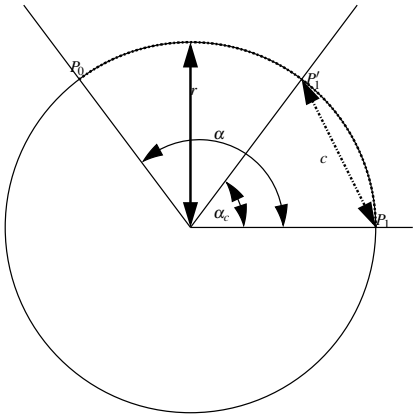

Abbildung 4: Korrektur am Kreisbogen

4.2 Längenkorrektur für Pfeilspitzen an Kreisbögen

Ist ein Kreisbogen von P_0 nach P_1 durch den Radius r und Winkel α gegeben und soll so verkürzt werden, dass c die Entfernung zwischen dem neuen Endpunkt P_1' und dem alten Endpunkt P_1 ist, so kann die Winkelverkürzung α_c folgendermaßen berechnet werden:

$$
\begin{aligned}
c &= \sqrt{(x_1' - x_1)^2 + (y_1' - y_1)^2} \\
&= \sqrt{(r\cos\alpha_c - r)^2 + (0 - r\sin\alpha_c)^2} \\
&= \sqrt{r^2 - 2r^2\cos\alpha_c + r^2\cos^2\alpha_c + r^2\sin^2\alpha_c} \\
&= r\sqrt{1 - 2\cos\alpha_c + \cos^2\alpha_c + \sin^2\alpha_c} \\
&= r\sqrt{2 - 2\cos\alpha_c} \\
\frac{c}{r} &= \sqrt{2 - 2\cos\alpha_c} \\
2 - 2\cos\alpha_c &= \left(\frac{c}{r}\right)^2 \\
2\cos\alpha_c &= 2 - \left(\frac{c}{r}\right)^2 \\
\cos\alpha_c &= 1 - \frac{c^2}{2r^2} \\
\alpha_c &= \arccos\left(1 - \frac{c^2}{2r^2}\right)
\end{aligned}
$$

4.3 Annäherung von Kreisbögen durch Bezier-Splines

Die nachfolgende Untersuchung wurde inspiriert durch die deutsche Wikipedia-Seite zum Thema „Bezierkurven“. Dort ist ein Verfahren angegeben, wie Kontrollpunkte für die näherungsweise Darstellung eines Viertelkreises durch ein Bezier-Spline-Segment gewonnen werden können. Dieses Verfahren wird hier verallgemeinert, um Kreisbögen mit beliebigem Öffnungswinkel darstellen zu können. In fig2lat wird die Formel für Kreisbogensegmente mit $|\alpha| \leq \frac{\pi}{2}$ verwendet, größere Kreisbögen werden dann durch mehrere Bezier-Segmente realisiert.
Im Abschnitt 3.2 auf Seite 9 wurde gezeigt, wie aus Start- und Endpunkt einer parametrischen Kurve und den Ableitungen $\mathrm{d}\overrightarrow{r}/\mathrm{d}t$ in Start- und Endpunkt die Kontrollpunkte für ein Bezier-Spline-Segment gewonnen werden können. Damit wird ein Bezier-Spline-Segment erzeugt, das den Verläufen $x(t)$ und $y(t)$ möglichst nahe kommt.
Hauptziel der Annäherung ist jedoch eine möglichst gute Annäherung an gewünschte x-y-Verläufe. Deshalb wird hier untersucht, ob eine Variation der Kontrollpunkte zu besseren Ergebnissen führt.
Die mit

$$\overrightarrow{r_{\mathrm{cs}}} = \overrightarrow{r_{\mathrm{s}}} + \tfrac{1}{3}\dot{\overrightarrow{r}}(0)$$
$$\overrightarrow{r_{\mathrm{ce}}} = \overrightarrow{r_{\mathrm{e}}} - \tfrac{1}{3}\dot{\overrightarrow{r}}(1)$$

gewonnenen Bezier-Kurven weisen zwar am Rand des Intervalles $[0;1]$ den geforderten Radius auf, im Intervall-Inneren verläuft die Kurve aber innerhalb des Kreises, d. h. der Radius wird unterschritten.
Um die Kurve zu ändern, werden die Kontrollpunkte etwas weiter von Start- und Endpunkt weggeschoben, sie bleiben aber auf den Tangenten in Start- und Endpunkt. Aufgrund der Symmetrie der Anordnung werden beide Kontrollpunkte in gleichem Abstand zum jeweiligen Endpunkt gehalten.

Für den Kreis gilt:

$$\tilde{k}_{\mathrm{k}}: \quad \overrightarrow{r_{\mathrm{k}}}(t) = \begin{pmatrix} r\cos\alpha t \\ r\sin\alpha t \end{pmatrix}$$

Die erste Ableitung ergibt

$$\dot{\overrightarrow{r_{\mathrm{k}}}}(t) = \begin{pmatrix} -r\alpha\sin\alpha t \\ r\alpha\cos\alpha t \end{pmatrix}$$

Als Start- und Endpunkt erhalten wir

$$\overrightarrow{r_\mathrm{s}} = \overrightarrow{r_\mathrm{k}}(0) = \begin{pmatrix} r \\ 0 \end{pmatrix}$$
$$\overrightarrow{r_\mathrm{e}} = \overrightarrow{r_\mathrm{k}}(1) = \begin{pmatrix} r\cos\alpha \\ r\sin\alpha \end{pmatrix}$$

Als Ansatz für die Kontrollpunkte wählen wir

$$\overrightarrow{r_\mathrm{cs}} = \overrightarrow{r_\mathrm{s}} + \varkappa\dot{\overrightarrow{r_\mathrm{k}}}(0)$$
$$\overrightarrow{r_\mathrm{ce}} = \overrightarrow{r_\mathrm{e}} - \varkappa\dot{\overrightarrow{r_\mathrm{k}}}(1)$$

und erhalten damit die Bezier-Kurve

$$\begin{aligned}\tilde{k}_\mathrm{b}: \quad \overrightarrow{r_\mathrm{b}}(t) = {} & (1-t)^3\overrightarrow{r_\mathrm{s}} + 3(1-t)^2 t\left(\overrightarrow{r_\mathrm{s}} + \varkappa\dot{\overrightarrow{r_\mathrm{k}}}(0)\right) \\ & + 3(1-t)t^2\left(\overrightarrow{r_\mathrm{e}} - \varkappa\dot{\overrightarrow{r_\mathrm{k}}}(1)\right) + t^3\overrightarrow{r_\mathrm{e}}\end{aligned}$$

Die oben aufgeführten Gleichungen werden in die Forderung

$$r = \left|\overrightarrow{r_\mathrm{b}}\left(\tfrac{1}{2}\right)\right|$$

eingesetzt und diese dann nach $\varkappa$ umgestellt:

$$\varkappa = \frac{2\sqrt{8(1-\cos\alpha)} - 4\sin\alpha}{3\alpha(1-\cos\alpha)}$$

Dabei ergeben sich recht lange Gleichungen. Zum Einsetzen, Umstellen und Lösen von Gleichungen wurde deshalb die Mathematik-Software wxMaxima mit folgenden Befehlen benutzt:

```
/* x- und y-Koordinaten des Kreises. */
X: R * cos(alpha * t);
Y: R * sin(alpha * t);

/* Start- und Endpunkt des Kreisbogens. */
xs: at(X, t=0);
ys: at(Y, t=0);
xe: at(X, t=1);
ye: at(Y, t=1);

/* Ableitungen der Kreis-Koordinaten. */
dXdt: diff(X, t);
dYdt: diff(Y, t);

/* Formeln fuer Kontrollpunkte, kappa ist noch unbekannt. */
xcs: xs + kappa * at(dXdt, t=0);
ycs: ys + kappa * at(dYdt, t=0);
xce: xe - kappa * at(dXdt, t=1);
yce: ye - kappa * at(dYdt, t=1);

/* x- und y-Koordinaten der Bezierkurve. */
xb: (1-t)^3*xs + 3*(1-t)^2*t*xcs + 3*(1-t)*t^2*xce + t^3*xe;
yb: (1-t)^3*ys + 3*(1-t)^2*t*ycs + 3*(1-t)*t^2*yce + t^3*ye;

/* Fuer t=0.5 soll Bezierkurve auch den Radius R haben.
   Wir suchen das dafuer passende kappa.
*/
res: solve( at(xb,t=1/2)^2 + at(yb,t=1/2)^2 = R^2, kappa);

/* Ergebnis vereinfachen. */
trigsimp(rhs(res[2]));
```

Literatur

[CB95] Christophe Schlick Carole Blanc. *X-Splines: A Spline Model Designed for the End-User*. SIGGRAPH proceedings 1995, 1995.