Erstellung eines E-Learning-Moduls zum Hopf'schen Umlaufsatz

Raffael Hagger

Frühjahrssemester 2010

Inhaltsverzeichnis

1 Einleitung

Diese Arbeit gibt einen Einstieg in die ebene Kurventheorie. Es werden alle wichtigen Begriffe erläutert und erste wichtige Sätze, insbesondere der Hopf'sche Umlaufsatz bewiesen. Begleitend dazu wurde ein Applet auf Basis von LEMUREN erstellt, das insbesondere die Krümmung illustriert. Dieses Applet bietet die Möglichkeit, eigene Kurven zu zeichnen und diese nach Krümmung und Umlaufzahl zu untersuchen. Ausserdem enthält diese Arbeit eine Dokumentation der Entstehung des Applets, sowie MATLAB-Implementierungen der Algorithmen. Das Applet und die MATLAB-Codes können auf der Homepage `http://www.math.ethz.ch/~lemuren/bsc/haggerr/` heruntergeladen werden.

Es empfiehlt sich, das Applet während des Lesens dieser Arbeit offen zu haben, da einige Illustrationen und vor allem die Aufgaben am Ende jedes Kapitels darauf verweisen. Dies soll den Lerneffekt verstärken und die ebene Kurventheorie gut verständlich einführen. Einige Begriffe aus der Analysis wie die des Wegintegrals oder der Differenzierbarkeit werden als bekannt angenommen. Die Vorlesungen Analysis I und II sind daher die Grundvoraussetzung für das Lesen dieser Arbeit.

2 Die Krümmung

Wir haben alle eine Intuition dafür, wann eine Kurve gekrümmt ist. Dieser Abschnitt versucht den Begriff der Krümmung zu formalisieren und erste explizite Rechenbeispiele zu geben. Ein Skript, welchem dieser Abschnitt teilweise folgt, sowie weiterführende Themen findet man unter [DG1] (allerdings ohne Beweise und Beispiele).

2.1 Definition (C^2-Kurve)

Sei $I \subset \mathbb{R}$ ein Intervall. Eine Abbildung $c\colon I \to \mathbb{R}^2$ heisst eine *C^2-Kurve*, falls sie zweimal stetig differenzierbar ist.

2.2 Definition (regulär)

Eine C^2-Kurve $c\colon I \to \mathbb{R}^2$ heisst *regulär*, falls $\dot{c}(t) \neq 0$ für alle $t \in I$ gilt.

2.3 Definition (Krümmung)

Sei $c\colon I \to \mathbb{R}^2$ eine reguläre C^2-Kurve. Wir definieren die beiden Vektorfelder e_1 und e_2 wie folgt:

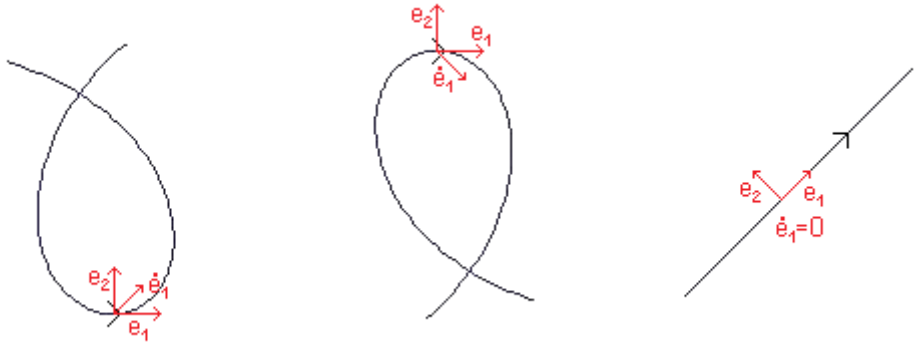
$$e_1(t) := \frac{\dot{c}(t)}{|\dot{c}(t)|},$$
$$e_2(t) := \begin{pmatrix} -e_1^2(t) \\ e_1^1(t) \end{pmatrix}.$$

Die Funktion $\kappa(t) := \frac{1}{|\dot{c}(t)|} \langle \dot{e_1}(t), e_2(t) \rangle$ heisst dann die *Krümmung* von c an der Stelle t.

Da die Ableitung einer regulären Kurve nirgends verschwindet und wir von der Kurve C^2 gefordert haben, ist das Einheitstangentialvektorfeld e_1 überall stetig differenzierbar. Das Einheitsnormalenvektorfeld e_2 ist demnach ebenfalls überall stetig differenzierbar. Daraus folgt, dass die Krümmung an jedem Punkt wohldefiniert und stetig ist.

2.4 Illustration

Um die Krümmung besser zu verstehen betrachten wir zunächst eine Illustration.



Im ersten Bild ist der Winkel zwischen $\dot{e_1}$ und e_2 kleiner als 90 Grad, was bedeutet, dass die Krümmung positiv ist. Die Schlaufe wird gegen den Uhrzeigersinn durchlaufen und erfährt im eingezeichneten Punkt eine Linkskurve. Im zweiten Bild ist der Winkel gerade grösser als 90 Grad und die Krümmung damit negativ. Die Schlaufe wird mit dem Uhrzeigersinn durchlaufen und dreht nach rechts. Im dritten Bild verläuft die Kurve geradeaus und die Krümmung ist Null. Vergleiche auch mit der Aufgabe 2.9 i).

2.5 Bemerkung (alternative Definition)

Oft definiert man die Krümmung auch wie folgt:

$$\kappa := \frac{1}{|\dot{c}(t)|^3} \det(\dot{c}(t), \ddot{c}(t)).$$

Diese Definition erweist sich in der Praxis meist als nützlicher, weil wir so die Krümmung κ direkt aus der Kurve c und ihren Ableitungen bestimmen kann. Die Äquivalenz dieser Definitionen zeigt man wie folgt:

$$
\begin{aligned}
\kappa(t) &= \frac{1}{|\dot{c}(t)|}\,\langle \dot{e_1}(t), e_2(t)\rangle \\
&= \frac{1}{|\dot{c}(t)|}\,\langle \frac{|\dot{c}(t)|\,\ddot{c}(t) - \dot{c}(t)(|\dot{c}(t)|)^{\cdot}}{|\dot{c}(t)|^2}, e_2(t)\rangle \\
&= \frac{1}{|\dot{c}(t)|^2}\,\langle \ddot{c}(t), e_2(t)\rangle \qquad (\text{wegen } \langle \dot{c}(t), e_2(t)\rangle = 0) \\
&= \frac{1}{|\dot{c}(t)|^2}\,|\ddot{c}(t)|\,|e_2(t)|\,\cos\sphericalangle(\ddot{c}(t), e_2(t)) \\
&= -\frac{1}{|\dot{c}(t)|^2}\,|\ddot{c}(t)|\,\sin\sphericalangle(\ddot{c}(t), e_1(t)) \quad (\text{weil } e_1 \text{ senkrecht zu } e_2 \text{ steht}) \\
&= \frac{1}{|\dot{c}(t)|^2}\,\det(e_1(t), \ddot{c}(t)) \\
&= \frac{1}{|\dot{c}(t)|^3}\,\det(\dot{c}(t), \ddot{c}(t)).
\end{aligned}
$$

2.6 Definition (Umparametrisierung)

Sei $c\colon [a,b] \to \mathbb{R}^2$ eine reguläre C^2-Kurve und $f\colon [\tilde{a},\tilde{b}] \to [a,b]$ ein C^2-Diffeomorphismus mit $\dot{f}(t) \neq 0$ für alle $t \in [\tilde{a},\tilde{b}]$. Dann heisst $c \circ f$ eine *reguläre Umparametrisierung* von c.

Falls zusätzlich $f(\tilde{a}) = a$, $f(\tilde{b}) = b$ gilt, dann heisst die Umparametrisierung *orientierungserhaltend*.

2.7 Lemma

Die Krümmung ist invariant unter regulären, orientierungserhaltenden C^2-Umparametrisierungen.

Beweis. Sei $c\colon [a,b] \to \mathbb{R}^2$ eine reguläre C^2-Kurve und $e_1 = \frac{\dot{c}(t)}{|\dot{c}(t)|}$ das zugehörige Tangetialvektorfeld. Sei weiter $\tilde{c} := c \circ f$ eine reguläre Umparametrisierung. Da $\dot{f}(t)$ keine Nullstelle hat und $\dot{f}(\tilde{a}) > 0$ ist, folgt aus dem Zwischenwertsatz, dass $\dot{f}(t) > 0$ für alle $t \in [\tilde{a},\tilde{b}]$ gilt. Für die Krümmung folgt nun also:

$$
\begin{aligned}
\tilde{\kappa}(t) &= \frac{1}{\left|\dot{\tilde{c}}(t)\right|} \langle \dot{\tilde{e}_1}(t), \tilde{e_2}(t) \rangle \\
&= \frac{1}{|(c \circ f)\dot{}(t)|} \langle (e_1 \circ f)\dot{}(t), (e_2 \circ f)(t) \rangle \\
&= \frac{1}{\left|\dot{c}(f(t))\, \dot{f}(t)\right|} \langle (\dot{e_1}(f(t))\, \dot{f}(t), e_2(f(t)) \rangle \\
&= \frac{\dot{f}(t)}{|\dot{c}(f(t))| \left|\dot{f}(t)\right|} \langle (\dot{e_1}(f(t)), e_2(f(t)) \rangle \\
&= \frac{1}{|\dot{c}(f(t))|} \langle (\dot{e_1}(f(t)), e_2(f(t)) \rangle \\
&= (\kappa \circ f)(t)
\end{aligned}
$$

□

Aus diesem Lemma folgt, dass die Krümmung wirklich eine Eigenschaft der Kurve selbst und nicht etwa eine Eigenschaft der Parametrisierung ist. Nehmen wir eine Umparametrisierung, die nicht orientierungserhaltend ist, das heisst, durchlaufen wir die Kurve in die entgegengesetzten Richtung, so ändert die Krümmung in jedem Punkt ihr Vorzeichen. Der Betrag der Krümmung bleibt aber erhalten. Dies nachzurechnen sei dem Leser als Übung überlassen.

2.8 Beispiele

Wir betrachten nun einige einfache Beispiele, um das Konzept der Krümmung etwas besser zu verstehen. Diese Beispiele sind sehr wichtig. Wir werden sie später immer wieder verwenden, um uns gewisse Begriffe besser vorstellen und Sätze verifizieren zu können.

2.8.1 Gerade

Sei $c\colon \mathbb{R} \to \mathbb{R}^2$ eine Gerade:

$$
c(t) := \begin{pmatrix} t \\ t \end{pmatrix}
$$

$$
e_1(t) = \tfrac{\dot{c}(t)}{|\dot{c}(t)|} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}, e_2(t) = \begin{pmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}.
$$

Damit lässt sich die Krümmung für alle $t \in \mathbb{R}$ berechnen:

$$
\kappa(t) = \tfrac{1}{|\dot{c}(t)|} \langle \dot{e_1}(t), e_2(t) \rangle = 0.
$$

2.8.2 Graphen

Ein Graph ist eine Kurve $c\colon I \to \mathbb{R}^2$ der folgenden Form:

$$c(t) := \begin{pmatrix} t \\ c_2(t) \end{pmatrix}.$$

$$e_1(t) = \frac{\dot{c}(t)}{|\dot{c}(t)|} = \begin{pmatrix} \frac{1}{\sqrt{1+\dot{c}_2(t)^2}} \\ \frac{\dot{c}_2(t)}{\sqrt{1+\dot{c}_2(t)^2}} \end{pmatrix},\ e_2(t) = \begin{pmatrix} -\frac{\dot{c}_2(t)}{\sqrt{1+\dot{c}_2(t)^2}} \\ \frac{1}{\sqrt{1+\dot{c}_2(t)^2}} \end{pmatrix}$$

$$\dot{e}_1(t) = \begin{pmatrix} -\frac{\dot{c}_2(t)}{(1+\dot{c}_2(t)^2)^{\frac{3}{2}}} \\ \frac{\ddot{c}_2(t)}{\sqrt{1+\dot{c}_2(t)^2}} - \frac{\dot{c}_2(t)^2}{(1+\dot{c}_2(t)^2)^{\frac{3}{2}}} \end{pmatrix}$$

Für einen Graph gilt also:

$$\kappa(t) = \frac{1}{|\dot{c}(t)|}\langle \dot{e_1}(t), e_2(t)\rangle = \frac{\ddot{c}_2(t)\,(1+\dot{c}_2(t)^2)}{(1+\dot{c}_2(t)^2)^{\frac{5}{2}}} = \frac{\ddot{c}_2(t)}{(1+\dot{c}_2(t)^2)^{\frac{3}{2}}}.$$

Oder direkt mit der alternativen Definition:

$$\kappa(t) = \frac{1}{|\dot{c}(t)|^3}\,\det(\dot{c}(t), \ddot{c}(t)) = \frac{1}{|\dot{c}(t)|^3}\,\det\begin{pmatrix} 1 & 0 \\ \dot{c}_2(t) & \ddot{c}_2(t) \end{pmatrix} = \frac{\ddot{c}_2(t)}{(1+\dot{c}_2(t)^2)^{\frac{3}{2}}}.$$

2.8.3 Kreis

Sei $c\colon [0, 2\pi] \to \mathbb{R}^2$ ein Kreis:

$$c(t) := \begin{pmatrix} \cos t \\ \sin t \end{pmatrix}.$$

$$\kappa(t) = \det\begin{pmatrix} -\sin t & -\cos t \\ \cos t & -\sin t \end{pmatrix} = 1$$

2.8.4 Acht

Sei $c\colon [0, 2\pi] \to \mathbb{R}^2$ wie folgt definiert:

$$c(t) := \begin{pmatrix} \sin t\,\cos t \\ \sin t \end{pmatrix}.$$

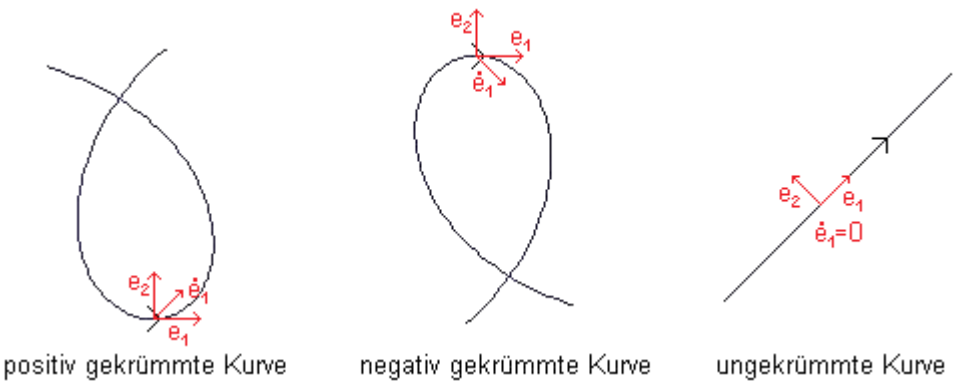
Anschaulich beschreibt c eine stehende Acht mit Mittelpunkt 0. Die Krümmung können wir auch in diesem Fall berechnen:

$$\kappa(t) = \frac{1}{\left|(\cos^2 t - \sin^2 t)^2 + \cos^2 t\right|^3} \det \begin{pmatrix} \cos^2 t - \sin^2 t & -4 \sin t \cos t \\ \cos t & -\sin t \end{pmatrix}$$
$$= \frac{\sin^3 t + 3 \sin t \cos^2 t}{\left|(\cos^2 t - \sin^2 t)^2 + \cos^2 t\right|^3}.$$

2.9 Aufgaben

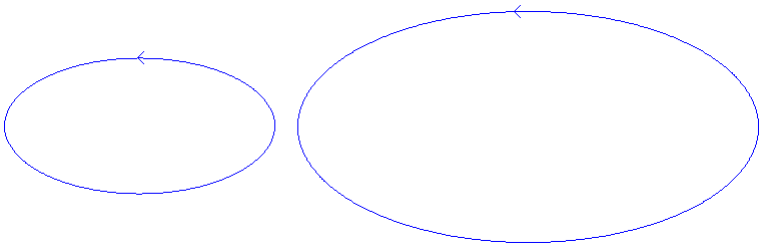
Um einige der Aufgaben lösen zu können, sollte man nebenbei das Applet offen haben. Das Applet steht auf der Homepage `http://www.math.ethz.ch/~lemuren/bsc/haggerr/` zum Download bereit.

i) Betrachte nochmal Illustration 2.4:



Verifiziere mit Hilfe des Applets, dass eine Links- bzw. eine Rechtskurve immer eine positive bzw. negative Krümmung bedeutet.

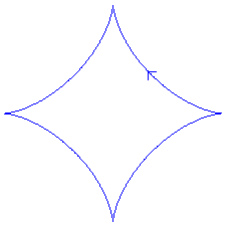
ii) Betrachte folgende zwei Ellipsen. Welche hat eine grössere Krümmung?



iii) Betrachte die Krümmung κ für die vorgegebenen Kurven im Applet. Welche Kurven haben eine konstante Krümmung $\kappa \equiv C$? Rechne explizit nach, dass diese Kurven wirklich eine konstante Krümmung haben.

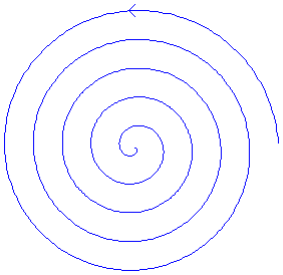
iv) Berechne die Krümmung der Astroide:

$$c\colon [0, 2\pi] \to \mathbb{R}^2,\ c(t) = \begin{pmatrix} \cos^3 t \\ \sin^3 t \end{pmatrix}.$$



Ist die Krümmung überall definiert?

v) Betrachte diese Spirale mit der Parametrisierung unten. Zeige dass die Krümmung monoton wachsend ist.



$$c\colon [0, \pi] \to \mathbb{R}^2,\ c(t) = \begin{pmatrix} (\pi - t)\ \cos(10t) \\ (\pi - t)\ \sin(10t) \end{pmatrix}$$

2.10 Lösungen

i) -

ii) Ohne näher auf die Ellipsen einzugehen, zeigen wir ein allgemeineres Resultat, nämlich dass die Skalierung einer Kurve um den Faktor $K > 0$ gerade einer Skalierung der Krümmung um den Faktor $\frac{1}{K}$ entspricht. Sei $c(t)\colon [0, L] \to \mathbb{R}^2$ eine reguläre C^2-Kurve mit Krümmung $\kappa\colon [0, L] \to \mathbb{R}$ und $c_K(t) := K\, c(t)$ die um den Faktor $K > 0$ skalierte Kurve. Für die Krümmung κ_K von c_K gilt dann:

$$\kappa_K(t) = \frac{1}{|\dot{c_K}(t)|}\, \langle \dot{e_1}(t), e_2(t) \rangle = \frac{1}{|K\, \dot{c}(t)|}\, \langle \dot{e_1}(t), e_2(t) \rangle = \frac{1}{K}\, \kappa(t),$$

weil e_1 und e_2 für c und c_K übereinstimmen. Die Krümmung ist also nicht skalierungsinvariant, sondern umgekehrt proportional zur Skalierung der Kurve. Damit besitzt die kleinere Ellipse in jedem Punkt eine grössere Krümmung als die grössere Ellipse.

iii) Mit Hilfe des Applets sehen wir, dass unter den angegebenen Kurven nur Geraden und Kreise konstante Krümmung haben. Dies haben wir schon in den Beispielen 2.8.1 für Geraden und 2.8.3 für den Einheitskreis gezeigt. Mit Aufgabe ii) wissen wir, wie sich die Krümmung unter Skalierung verhält. Damit haben wir das Resultat, dass ein regulär parametrisierter Kreis mit Radius R konstante Krümmung $\pm\frac{1}{R}$ hat. Wir bekommen ein positives Vorzeichen, wenn wir den Kreis gegen den Uhrzeigersinn durchlaufen und ein entsprechend negatives Vorzeichen sonst. Man kann sogar zeigen, dass jede reguläre C^2-Kurve mit konstanter Krümmung in einer Gerade oder einem Kreis enthalten ist.

iv) Die Astroide ist definiert durch:

$$c(t) = \begin{pmatrix} \cos^3 t \\ \sin^3 t \end{pmatrix}.$$

Daraus berechnen wir:

$$\begin{aligned}
\dot{c}(t) &= \begin{pmatrix} -3\,\cos^2 t\,\sin t \\ 3\,\sin^2 t\,\cos t \end{pmatrix}, \\
\ddot{c}(t) &= \begin{pmatrix} 6\,\cos t\,\sin^2 t - 3\,\cos^3 t \\ 6\,\sin t\,\cos^2 t - 3\,\sin^3 t \end{pmatrix}, \\
\kappa(t) &= \frac{1}{|\dot{c}(t)|^3}\,\det(\dot{c}(t), \ddot{c}(t)) \\
&= \frac{1}{(9\,\cos^4 t\,\sin^2 t + 9\,\sin^4 t\,\cos^2 t)^{3/2}}\,(-18\,\cos^4 t\,\sin^2 t + 9\,\cos^2 t\,\sin^4 t \\
&\qquad\qquad - 18\,\sin^4 t\,\cos^2 t + 9\,\sin^2 t\,\cos^4 t) \\
&= \frac{1}{27\,|\sin t\,\cos t|^3}\,(-9)\,\sin^2 t\,\cos^2 t \\
&= -\frac{1}{3\,|\sin t\,\cos t|}.
\end{aligned}$$

Die Krümmung ist demnach an den Stellen $t = 0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi$ nicht definiert. Das sind genau die Spitzen der Astroide. An diesen Stellen ist die Kurve nicht regulär und daher das Tangentialvektorfeld nicht wohldefiniert.

v) Wir haben:

$$c\colon [0,\pi] \to \mathbb{R}^2, c(t) = \begin{pmatrix} (\pi - t)\,\cos(10t) \\ (\pi - t)\,\sin(10t) \end{pmatrix}$$

und damit:

$$
\begin{aligned}
\dot{c}(t) &= \begin{pmatrix} -\cos(10t) - 10\,(\pi - t)\,\sin(10t) \\ -\sin(10t) + 10\,(\pi - t)\,\cos(10t) \end{pmatrix}, \\
\ddot{c}(t) &= \begin{pmatrix} 11\,\sin(10t) - 100\,(\pi - t)\,\cos(10t) \\ -11\,\cos(10t) - 100\,(\pi - t)\,\sin(10t) \end{pmatrix}, \\
\kappa(t) &= \frac{1}{|\dot{c}(t)|^3}\,\det(\dot{c}(t), \ddot{c}(t)) \\
&= \frac{1}{(1 + 100\,(\pi - t)^2)^{3/2}}\,(11\,\cos^2(10t) + 1000\,(\pi - t)^2\,\sin^2(10t) \\
&\qquad\qquad + 210\,(\pi - t)\,\sin(10t)\,\cos(10t) + 11\,\sin^2(10t) \\
&\qquad\qquad + 1000\,(\pi - t)^2\,\cos^2(10t) - 210\,(\pi - t)\,\sin(10t)\,\cos(10t)) \\
&= \frac{1}{(1 + 100\,(\pi - t)^2)^{3/2}}\,(11 + 1000\,(\pi - t)^2) \\
&= \frac{10}{(1 + 100\,(\pi - t)^2)^{1/2}} + \frac{1}{(1 + 100\,(\pi - t)^2)^{3/2}}.
\end{aligned}
$$

Beide Terme sind monoton wachsend für $t \in [0, \pi]$. Daher ist die Krümmung der Spirale ebenfalls monoton wachsend.

3 Winkelfunktionen

Dieser Abschnitt beschäftigt sich als Vorbereitung für den Hopf'schen Umlaufsatz mit Winkelfunktionen und den Eigenschaften von geschlossenen Kurven. Der Abschnitt folgt dem Skript [WI].

3.1 Definition (geschlossen)

Wir betrachten eine reguläre C^2-Kurve $c\colon [0, L] \to \mathbb{R}^2$ als *geschlossen*, falls es ein $\epsilon > 0$ und eine C^2-Kurve $\tilde{c}\colon [0, L + \epsilon] \to \mathbb{R}^2$ mit

i) $c(t) = \tilde{c}(t)$ für alle $t \in [0, L]$,

ii) $\tilde{c}(t) = \tilde{c}(t - L)$ für alle $t \in [L, L + \epsilon]$

gibt.

Eine reguläre C^2-Kurve $c\colon [0, L] \to \mathbb{R}^2$ heisst *einfach geschlossen*, falls sie geschlossen und auf $[0, L)$ injektiv ist.

Oft bezeichnet man eine Kurve auch als geschlossen, falls lediglich der Anfangspunkt mit dem Endpunkt übereinstimmt. Dies ist aber für unsere Zwecke nicht nützlich, da wir die Krümmung auf der gesamten Kurve definieren wollen und wir damit auch in diesem Punkt Regularität und C^2 fordern müssen.

3.2 Definition (Winkelfunktion)

Sei $V\colon [0,L]\times[0,K]\to S^1 := \{(x)\in\mathbb{R}^2 : |x|=1\}$ ein Einheitsvektorfeld, dann heisst $\varphi\colon [0,L]\times[0,K]\to\mathbb{R}$ die zu V gehörige *Winkelfunktion*, falls gilt:

$$V(t,s)=\begin{pmatrix}\cos\varphi(t,s)\\ \sin\varphi(t,s)\end{pmatrix},$$

Ist ein Vektorfeld in nur einer Variable $V\colon [0,L]\to\mathbb{R}^2$ gegeben, wie zum Beispiel bei einem Tangentialvektorfeld an eine reguläre Kurve, so betrachten wir das Vektorfeld $\bar{V}\colon [0,L]\times\{0\}\to\mathbb{R}^2$ mit $\bar{V}(t,0) := V(t)$. Analog sind dann auch Winkelfunktionen für das entsprechende Einheitsvektorfeld definiert.

3.3 Lemma (Existenz von Winkelfunktionen)

Für jedes differenzierbare Einheitsvektorfeld $V\colon [0,L]\times[0,K]\to\mathbb{R}^2$ existiert eine zugehörige differenzierbare Winkelfunktion φ.

Beweis. Sei $V(t,s)=\begin{pmatrix}x(t,s)\\ y(t,s)\end{pmatrix}$, dann gilt $x(t,s)^2+y(t,s)^2\equiv 1$ und wir können φ_0 wählen, so dass:

$$\begin{pmatrix}x(0,0)\\ y(0,0)\end{pmatrix}=\begin{pmatrix}\cos\varphi_0\\ \sin\varphi_0\end{pmatrix}$$

gilt.
Definiere nun $\varphi\colon [0,L]\times[0,K]\to\mathbb{R}$ wie folgt:

$$\begin{aligned}\varphi(t,s) := &\int_0^t x(t',0)\frac{\partial y}{\partial t'}(t',0)-\frac{\partial x}{\partial t'}(t',0)y(t',0)\,\mathrm{d}t'\\ &+\int_0^s x(0,s')\frac{\partial y}{\partial s'}(0,s')-\frac{\partial x}{\partial s'}(0,s')y(0,s')\,\mathrm{d}s'+\varphi_0\end{aligned}$$

Damit ist φ als Summe von differenzierbaren Funktionen ebenfalls differenzierbar. Betrachte nun die folgende Funktion von t und s:

$$F(t,s) := (x(t,s)-\cos\varphi(t,s))^2+(y(t,s)-\sin\varphi(t,s))^2.$$

Wegen $\varphi(0,0)=\varphi_0$ und der Wahl von φ_0 folgt direkt, dass $F(0,0)=0$ ist. Als nächstes zeigen wir, dass $dF\equiv 0$ und damit $F\equiv 0$ gilt. In der folgenden Rechnung werden wir

für eine bessere Übersicht alle Abhängigkeiten von s und t weglassen. Natürlich sind x, y und φ immer noch Funktionen von s und t und die Ableitungen auch entsprechend zu verstehen. Folgende zwei Gleichungen werden wir verwenden, um $dF \equiv 0$ zu zeigen:

i) $\dot{x}x + \dot{y}y = \frac{1}{2}(x^2 + y^2)^{\cdot} = 0$,

ii) $\dot{\varphi} = (x\dot{y} - \dot{x}y)$ (Hauptsatz der Differential- und Integralrechnung).

Damit folgt:

$$\begin{aligned}
\frac{1}{2}\frac{\partial F}{\partial t} &= (x - \cos\varphi)(\dot{x} + \dot{\varphi}\sin\varphi) + (y - \sin\varphi)(\dot{y} - \dot{\varphi}\cos\varphi) \\
&= \dot{x}x - \dot{x}\cos\varphi + x\dot{\varphi}\sin\varphi - \dot{\varphi}\sin\varphi\cos\varphi + \dot{y}y - \dot{y}\sin\varphi - y\dot{\varphi}\cos\varphi + \dot{\varphi}\sin\varphi\cos\varphi \\
&= (\dot{x}x + \dot{y}y) - (\dot{x}\cos\varphi + \dot{y}\sin\varphi) + \dot{\varphi}(x\sin\varphi(t) - y\cos\varphi) \\
&= 0 - (x^2 + y^2)(\dot{x}\cos\varphi + \dot{y}\sin\varphi) + (x\dot{y} - \dot{x}y)(x\sin\varphi - y\cos\varphi) \\
&= -x\cos\varphi(x\dot{x} + y\dot{y}) - y\sin\varphi(x\dot{x} + y\dot{y}) \\
&= -(x\cos\varphi + y\sin\varphi)(x^2 + y^2)^{\cdot} \\
&= 0.
\end{aligned}$$

Analog folgt auch $\frac{\partial F}{\partial s} = 0$.

Zusammen mit $F(0,0) = 0$ haben wir also $F(t,s) \equiv 0$ und damit $x(t,s) \equiv \cos\varphi(t,s)$ und $y(t,s) \equiv \sin\varphi(t,s)$. □

3.4 Definition (Umlaufzahl)

Sei $c\colon [0,L] \to \mathbb{R}^2$ eine geschlossene, reguläre C^2-Kurve, $e_1(t) = \frac{c(t)}{|\dot{c}(t)|}$ das Einheitstangetialvektorfeld und φ die zugehörige Winkelfunktion, dann ist die *Umlaufzahl* von c wie folgt definiert:

$$\rho_c = \frac{1}{2\pi}(\varphi(L) - \varphi(0)).$$

3.5 Bemerkung

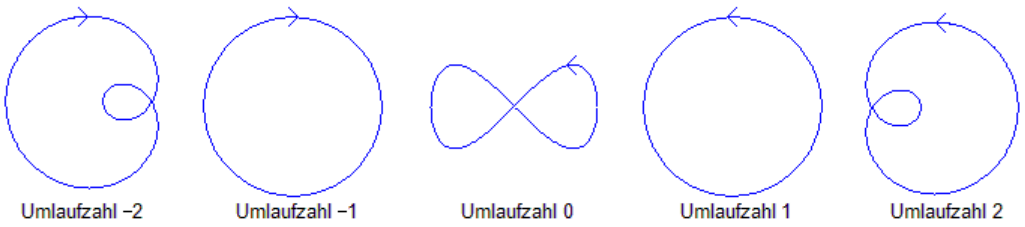
Ist $\varphi(t,s)$ eine Winkelfunktion für das Einheitsvektorfeld V, so ist für jedes $k \in \mathbb{Z}$ auch $\varphi(t,s) + 2\pi k$ eine Winkelfunktion für V.

Haben wir zwei Winkelfunktionen φ und $\bar{\varphi}$ für V, so gibt es ein $k \in \mathbb{Z}$, sodass $\varphi - \bar{\varphi} \equiv 2\pi k$.

Demnach hängt die Umlaufzahl nicht von den Wahl der Winkelfunktion ab und ist damit wie die Krümmung eine Eigenschaft der Kurve. Wegen $e_1(L) = e_1(0)$ ist ρ_c ausserdem immer ganzzahlig.

Wir können uns die Umlaufzahl als die Anzahl Drehungen um die eigene Achse vorstellen, während wir entlang der Kurve laufen. Auch hier betrachten wir wieder gerichtete Drehungen. Eine Drehung gegen den Uhrzeigersinn zählen wir positiv, eine Drehung gegen den Uhrzeigersinn entsprechend negativ. Folgende kleine Beispiele illustrieren diesen Umstand.

3.6 Beispiele



3.7 Definition (Isotopie)

Seien c_0 und $c_1 \colon [0, L] \to \mathbb{R}^2$ zwei geschlossene, reguläre C^2-Kurven. Eine stetige Funktion $F \colon [0, L] \times [0, 1] \to \mathbb{R}^2$ heisst *Isotopie* zwischen c_0 und c_1, falls die folgenden Eigenschaften erfüllt sind:

i) $F(t, 0) = c_0(t)$,
ii) $F(t, 1) = c_1(t)$,
iii) $F(\cdot, s) \colon [0, L] \to \mathbb{R}^2$ ist eine geschlossene, reguläre C^2-Kurve für alle $s \in [0, 1]$.

Die beiden Kurven heissen dann *isotop*.

3.8 Satz

Seien c_0 und $c_1 \colon [0, L] \to \mathbb{R}^2$ zwei isotope C^2-Kurven, dann gilt $\rho_{c_1} = \rho_{c_2}$.

Beweis. Sei F eine Isotopie zwischen c_0 und c_1 und definiere $c_s(t) := F(t, s)$. Sei weiter $\varphi \colon [0, L] \times [0, 1] \to \mathbb{R}$ die zu F gehörige Winkelfunktion. Die Funktion $\rho \colon [0, 1] \to \mathbb{Z}$ mit $\rho(s) := \rho_{c_s} = \frac{1}{2\pi}(\varphi(L, s) - \varphi(0, s))$ ist stetig und damit konstant. □

3.9 Korollar

i) Die Umlaufzahl einer Kurve ist unabhängig von der Wahl des Startpunkts.

ii) Die Umlaufzahl einer Kurve ist invariant unter regulären Umparametrisierungen.

iii) Die Umlaufzahl einer Kurve ist invariant unter Verschiebungen.

iv) Die Umlaufzahl einer Kurve ist invariant unter Drehungen.

v) Die Umlaufzahl einer Kurve ändert das Vorzeichen unter Spiegelungen.

Beweis. Wegen Satz 3.8 genügt es, wenn wir für i) bis iv) eine Isotopie angeben.

i) Sei $c_0\colon [0,L] \to \mathbb{R}^2$ eine geschlossene, reguläre C^2-Kurve und sei c_1 die gleiche Kurve mit Startpunkt $c_0(t_0)$:

$$c_1(t) = \begin{cases} c_0(t+t_0), & \text{für } t+t_0 \leq L \\ c_0(t+t_0-L), & \text{für } t+t_0 \geq L. \end{cases}$$

Eine Isotopie können wir dann direkt angeben:

$$F(t,s) = \begin{cases} c_0(t+st_0), & \text{für } t+st_0 \leq L \\ c_0(t+st_0-L), & \text{für } t+st_0 \geq L. \end{cases}$$

ii) Sei $c_0\colon [0,L] \to \mathbb{R}^2$ eine geschlossene reguläre C^2-Kurve und $c_1 := c_0 \circ f$ eine reguläre Umparametrisierung von c_0. Wie wir schon einmal gesehen haben, folgt damit bereits $\dot{f}(t) > 0$ für alle $t \in [0, \tilde{L}]$. Eine Isotopie ist dann gegeben durch:

$$F(t,s) = c_0((1-s)t + sf(t)).$$

Es bleibt zu zeigen, dass $F(\cdot, s)$ regulär ist. Dies folgt aus:

$$\frac{\partial F}{\partial t}(t,s) = ((1-s) + s\dot{f}(t))\, \dot{c_0}((1-s)t + sf(t)) \neq 0 \text{ für alle } s \in [0,1].$$

iii) Sei $c_0\colon [0,L] \to \mathbb{R}^2$ eine geschlossene, reguläre C^2-Kurve und c_1 die um den Vektor $a \in \mathbb{R}^2$ verschobene Kurve, dann können wir eine Isotopie wie folgt angeben:

$$F(t,s) = c_0(t) + sa.$$

iv) Sei $c_0\colon [0,L] \to \mathbb{R}^2$ eine geschlossene, reguläre C^2-Kurve und $A = \left(\begin{smallmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{smallmatrix}\right)$ eine Drehung um den Winkel ψ. Eine Isotopie zwischen c_0 und der bezüglich dem Ursprung um den Winkel ψ gedrehten Kurve $c_1 = A\, c_0$ ist gegeben durch:

$$F(t,s) = \begin{pmatrix} \cos(s\psi) & -\sin(s\psi) \\ \sin(s\psi) & \cos(s\psi) \end{pmatrix} c_0(t).$$

Eine allgemeine Drehung erhalten wir durch Komposition von Verschiebungen und Drehungen um den Ursprung. Wegen iii) ist die Umlaufzahl also auch invariant unter allgemeinen Drehungen.

v) Sei $c_0 \colon [0, L] \to \mathbb{R}^2$ eine geschlossene, reguläre C^2-Kurve und sei c_1 die um die x-Achse gespiegelte Kurve. Seien weiter φ_i die zu c_i gehörigen Winkelfunktionen:

$$\begin{aligned} \frac{c_0(t)}{|\dot{c_0}(t)|} &= \begin{pmatrix} \cos\varphi_0(t) \\ \sin\varphi_0(t) \end{pmatrix}, \\ \frac{c_1(t)}{|\dot{c_1}(t)|} &= \begin{pmatrix} \cos\varphi_1(t) \\ \sin\varphi_1(t) \end{pmatrix} = \begin{pmatrix} \cos\varphi_0(t) \\ -\sin\varphi_0(t) \end{pmatrix} = \begin{pmatrix} \cos(-\varphi_0(t)) \\ \sin(-\varphi_0(t)) \end{pmatrix}. \end{aligned}$$

Daraus folgt (bis auf ein Vielfaches von 2π) $\varphi_1 = -\varphi_0$ und damit:

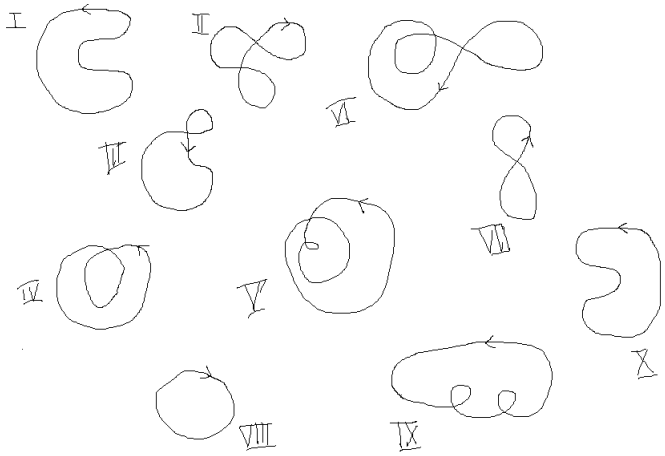
$$\rho_{c_1} = \frac{1}{2\pi}(\varphi_1(L) - \varphi_1(0)) = \frac{1}{2\pi}(-\varphi_0(L) + \varphi_0(0)) = -\rho_{c_0}$$

Eine allgemeine Spiegelung erhalten wir durch Komposition von Drehungen, Verschiebungen und der oben angegebenen Achsenspiegelung. Aus iii) und iv) folgt dann, dass die Umlaufzahl auch unter allgemeinen Spiegelungen das Vorzeichen wechselt.

□

3.10 Aufgaben

i) Betrachte folgende Skizze:



Ordne die Kurve aufsteigend nach der Umlaufzahl.

Hinweis: Bist Du Dir nicht sicher, so verwende das Applet, um die Umlaufzahlen zu verifizieren.

ii) Sei $c\colon [0, 2\pi] \to \mathbb{R}^2, c(t) = \begin{pmatrix} \cos t \\ \sin t \end{pmatrix}$ die Standardparametrisierung des Einheitskreises. Gib explizit eine Winkelfunktion an, die dem Tangetialvektorfeld entspricht und zeige, dass diese Kurve Umlaufzahl 1 hat. Zur Erinnerung: Wir suchen eine Funktion φ, die $e_1 = \frac{\dot{c}(t)}{|\dot{c}(t)|} = \begin{pmatrix} \cos\varphi(t) \\ \sin\varphi(t) \end{pmatrix}$ erfüllt.

iii) Wie ändert sich die Umlaufzahl, wenn wir eine geschlossene Kurve in die entgegengesetzte Richtung durchlaufen?

3.11 Lösungen

i) Umlaufzahl -1 haben: II, VI und VII.

Umlaufzahl 0 haben: III und VII.

Umlaufzahl 1 haben: I und X.

Umlaufzahl 2 hat: IV.

Umlaufzahl 3 haben: V und IX.

ii) Wir haben $e_1 = \begin{pmatrix} -\sin t \\ \cos t \end{pmatrix}$ und können damit sofort eine Winkelfunktion angeben: $\varphi(t) := t + \frac{\pi}{2}$. Die Umlaufzahl ρ ist dann gegeben durch $\rho = \frac{1}{2\pi}\left(\varphi(2\pi) - \varphi(0)\right) = 1$.

iii) Sei $c\colon [0, L] \to \mathbb{R}^2$ eine geschlossene Kurve. Die Kurve in entgegengesetzte Richtung zu durchlaufen, bedeutet gerade $\tilde{c}(t) := c(L - t)$ zu betrachten. Wenn φ eine Winkelfunktion für c ist, so ist $\tilde{\varphi}(t) := \varphi(L - t) - \pi$ eine Winkelfunktion für $\tilde{c}$,

denn wenn man die Kurve in entgegengesetzter Richtung durchläuft, dreht sich das Tangetialvektorfeld in jedem Punkt auf der Kurve gerade um π. Daraus folgt nun direkt:
$\tilde{\rho} = \frac{1}{2\pi}\left(\tilde{\varphi}(L) - \tilde{\varphi}(0)\right) = \frac{1}{2\pi}\left(\varphi(0) - \varphi(L)\right) = -\rho.$

4 Der Hopf'sche Umlaufsatz

4.1 Definition (Gesamtkrümmung)

Sei c eine geschlossene, reguläre C^2-Kurve, dann heisst $\int_c \kappa(t)\,\mathrm{d}s(t)$ die *Gesamtkrümmung* oder das *Krümmungsintegral* von c.

4.2 Satz (Hopf'scher Umlaufsatz)

Für eine geschlossene, reguläre C^2-Kurve c gilt:

$$\int_c \kappa(t)\,\mathrm{d}s(t) = 2\pi\rho_c,$$

wobei ρ_c die Umlaufzahl der Kurve ist.
Falls c einfach geschlossen ist, dann gilt sogar:

$$\int_c \kappa(t)\,\mathrm{d}s(t) = \pm 2\pi.$$

Beweis. Dieser Beweis wurde aus der Vorlesung Differentialgeometrie I Herbst 2009, gehalten von Prof. Urs Lang an der ETH Zürich, entnommen.

Sei $c\colon I \to \mathbb{R}^2$ eine geschlossene, reguläre C^2-Kurve und sei (e_1, e_2) die zugehörige positiv orientierte Orthonormalbasis in jedem Punkt. Schreibe das Vektorfeld $e_1(t)$ mit Hilfe einer Winkelfunktion φ:

$$e_1(t) = \begin{pmatrix}\cos\varphi(t)\\ \sin\varphi(t)\end{pmatrix}.$$

Ableiten von e_1 führt zu folgendem Ergebnis:

$$\begin{aligned}\dot{e_1}(t) &= \dot{\varphi}(t)\begin{pmatrix}-\sin\varphi(t)\\ \cos\varphi(t)\end{pmatrix}\\ &= \dot{\varphi}(t)\,e_2(t)\end{aligned}$$

und daher:

$$\begin{aligned}\kappa(t) &= \frac{1}{|\dot{c}(t)|}\,\langle \dot{e_1}(t), e_2(t)\rangle \\ &= \frac{1}{|\dot{c}(t)|}\,\langle \dot{\varphi}(t)\, e_2(t), e_2(t)\rangle \\ &= \frac{\dot{\varphi}(t)}{|\dot{c}(t)|}.\end{aligned}$$

Damit gilt für die Gesamtkrümmung:

$$\begin{aligned}\int_c \kappa(t)\,\mathrm{d}s(t) &= \int_c \frac{\dot{\varphi}(t)}{|\dot{c}(t)|}\,\mathrm{d}s(t) \\ &= \int_0^L \frac{\dot{\varphi}(t)}{|\dot{c}(t)|}\,|\dot{c}(t)|\,\mathrm{d}t \\ &= \int_0^L \dot{\varphi}(t)\,\mathrm{d}t \\ &= \varphi(L) - \varphi(0) \\ &= 2\pi\rho_c\end{aligned}$$

Sei nun $c\colon [0, L] \to \mathbb{R}$ einfach geschlossen. Mit Hilfe von Korollar 3.9 bringen wir c in folgende Position:

i) Die Kurve wird gegen den Uhrzeigersinn durchlaufen. Dies ist gerade der Fall $\rho_c \leq 0$ (Spiegelung).

ii) Das Bild von c ist vollständig in der oberen Halbebene enthalten (geeignete Wahl des Startpunktes), beginnt in $c(0) = \begin{pmatrix}0\\0\end{pmatrix}$ und hat Startrichtung $\frac{\dot{c}(0)}{|\dot{c}(0)|} = \begin{pmatrix}1\\0\end{pmatrix}$ (Verschiebung und Drehung).

Sei $A := \{(s,t) \in \mathbb{R}^2 : 0 \leqslant s, t \leqslant L\}$ und $f\colon A \to S^1 := \{(x) \in \mathbb{R}^2 : |x| = 1\}$ wie folgt definiert:

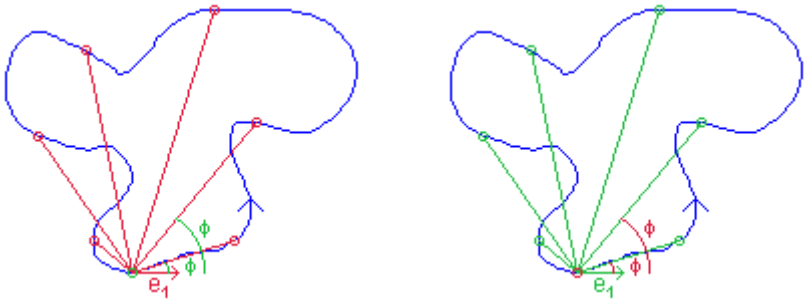
$$f(s,t) = \begin{cases} \frac{\dot{c}(s)}{|\dot{c}(s)|}, & \text{falls } s = t \\ \left(\begin{smallmatrix}-1\\0\end{smallmatrix}\right), & \text{falls } (s,t) = (0,L) \\ \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right), & \text{falls } (s,t) = (L,0) \\ \frac{c(t)-c(s)}{|c(t)-c(s)|}, & \text{sonst.}\end{cases}$$

Für f existiert nun eine stetige Winkelfunktion $\bar{\varphi}\colon A \to \mathbb{R}$ mit $\bar{\varphi}(0,0) = 0$.

Diese Winkelfunktion $\bar{\varphi}$ induziert eine Winkelfunktion $\varphi(t) := \bar{\varphi}(t,t)$ für $e_1(t) = f(t,t) = \frac{\dot{c}(t)}{|\dot{c}(t)|}$.

Wie oben gilt dann:

$$\begin{aligned}\int_c \kappa(t)\,\mathrm{d}s(t) &= \int_0^L \dot{\bar{\varphi}}(t,t)\,\mathrm{d}t \\ &= \bar{\varphi}(L,L) - \bar{\varphi}(0,0) \\ &= (\bar{\varphi}(L,L) - \bar{\varphi}(0,L)) + (\bar{\varphi}(0,L) - \bar{\varphi}(0,0)) \\ &= \pi + \pi \\ &= 2\pi.\end{aligned}$$



Wir durchlaufen dir Kurve zuerst in der zweiten Variablen und erhalten einen Winkel von $\bar{\varphi}(0,L) - \bar{\varphi}(0,0)$, was offensichtlich gerade π ist. Dann bleiben wir in der zweiten Variable stehen und durchlaufen mit der ersten Variablen die Kurve. Wir erhalten einen Winkel von $\bar{\varphi}(L,L) - \bar{\varphi}(0,L)$, was nochmal π ist. Daher haben wir als Umlaufzahl gerade $+1$.

Analog gilt $\rho_c =$ -1, falls die Kurve mit dem Uhrzeigersinn durchlaufen wird (vergleiche mit Korollar 3.9 v)).

Eine nette Animation zum besseren Verständnis des letzten Arguments findet man unter [HO].

□

Mit Hilfe dieses Satzes können wir in vielen Fällen die Umlaufzahl direkt ausrechnen, während wir uns mit der Winkelfunktion sehr schnell ziemlich schwer tun. In manchen Büchern wird das Krümmungsintegral auch direkt als Definition für die Umlaufzahl verwendet. Die Definition über die Winkelfunktion ist aber ebenfalls nützlich, zum Beispiel um eine geometrische Anschauung zu bekommen.

4.3 Beispiele

Betrachten wir nun einige Beispiele, um den Hopf'schen Umlaufsatz zu verifizieren:

4.3.1 Kreis

Für einen Kreis $c\colon [0, 2\pi] \to \mathbb{R}^2$, $c(t) = \begin{pmatrix} R\cos t \\ R\sin t \end{pmatrix}$ gilt:

$$\int_c \kappa(t)\,\mathrm{d}s(t) = \int_0^{2\pi} \frac{1}{R}\,|\dot{c}(t)|\,\mathrm{d}t = \int_0^{2\pi} \mathrm{d}t = 2\pi$$

und damit $\rho = 1$ wie erwartet.

4.3.2 Acht

Wir vergleichen mit Beispiel 2.8.4:

$$\begin{aligned} c(t) &= \begin{pmatrix} \sin t\,\cos t \\ \sin t \end{pmatrix}, \\ \dot{c}(t) &= \begin{pmatrix} \cos^2 t - \sin^2 t \\ \cos t \end{pmatrix}, \\ \kappa(t) &= \frac{\sin^3 t + 3\,\sin t\,\cos^2 t}{\left|(\cos^2 t - \sin^2 t)^2 + \cos^2 t\right|^3}. \end{aligned}$$

Damit folgt für die Gesamtkrümmung:

$$\int_c \kappa(t)\,\mathrm{d}s(t) = \int_0^{2\pi} \frac{\sin^3 t + 3\,\sin t\,\cos^2 t}{\left|(\cos^2 t - \sin^2 t)^2 + \cos^2 t\right|^2}\,\mathrm{d}t = 0.$$

Es sei dem Leser als Übung überlassen, sich davon zu überzeugen, dass das Integral verschwindet. Dazu zeigt man, dass der Integrand punktsymmetrisch um π ist.

4.3.3 Astroide

Wir vergleichen mit Aufgabe 2.9 iv):

$$\begin{aligned} c(t) &= \begin{pmatrix} \cos^3 t \\ \sin^3 t \end{pmatrix}, \\ \dot{c}(t) &= \begin{pmatrix} -3\,\cos^2 t\,\sin t \\ 3\,\sin^2 t\,\cos t \end{pmatrix}, \\ \kappa(t) &= -\frac{1}{3\,|\sin t\,\cos t|}. \end{aligned}$$

Wiederum berechnen wir die Gesamtkrümmung:

$$\begin{aligned}\int_c \kappa(t)\,\mathrm{d}s(t) &= \int_0^{2\pi} -\frac{1}{3\ |\sin t\ \cos t|}\ |\dot{c}(t)|\ \mathrm{d}t \\ &= \int_0^{2\pi} -\frac{1}{3\ |\sin t\ \cos t|}\ 3\sqrt{\cos^4 t\ \sin^2 t + \sin^4 t\ \cos^2 t}\,\mathrm{d}t \\ &= \int_0^{2\pi} -\frac{1}{|\sin t\ \cos t|}\ |\sin t\ \cos t|\ \mathrm{d}t \\ &= -2\pi.\end{aligned}$$

Wir haben also eine Kurve, die gegen den Uhrzeigersinn verläuft, aber eine Gesamtkrümmung von -2π aufweist. Betrachten wir die numerisch berechnete Gesamtkrümmung im Applet, so bekommen wir allerdings einen Wert von $+2\pi$. Wie kommt diese Differenz zu Stande? Der entscheidende Punkt ist, dass die Astroide keine reguläre C^2-Kurve ist. Wir werden dieses Problem im nächsten Satz lösen. Dazu benötigen wir aber erst einige Definitionen.

4.4 Definition (stückweise C^2)

Eine stetige Kurve $c\colon [0,L] \to \mathbb{R}^2$ heisst *stückweise* C^2, falls eine natürliche Zahl n und Punkte $0 = t_0 < ... < t_n = L \in [0,L]$ existieren, sodass $c_{|(t_{i-1},t_i)}$ zweimal stetig differenzierbar ist für alle $i = 1, ..., n$. Die Punkte t_i heissen Eckpunkte von c, sofern c in t_i nicht C^2 ist.

Eine solche Kurve heisst *regulär*, wenn sie auf allen Teilstücken $c_{|(t_{i-1},t_i)}$ regulär ist und sowohl der rechtsseitige wie auch der linksseitige Limes des Tangentialvektorfeldes an den Eckpunkten existiert und nicht verschwindet.

4.5 Definition (Winkelfunktionen für stückweise C^2-Kurven)

Sei $c\colon [0,L] \to \mathbb{R}^2$ eine stetige, reguläre, stückweise C^2-Kurve mit Eckpunkten $0 < t_1 < ... < t_n < L \in [0,L]$ und e_1 das fast überall definierte Einheitstangentialvektorfeld. Wir verallgemeinern den Begriff der *Winkelfunktion* auf stückweise C^2-Kurven wie folgt:

Wähle eine Winkelfunktion φ_0 auf $c_{|(0,t_1)}$, eine Winkelfunktion φ_i auf $c_{|(t_i,t_{i+1})}$ für $i = 1, ..., n-1$ und eine Winkelfunktion φ_n auf $c_{|(t_n,L)}$, sodass gilt:

$$\alpha_i := \lim_{t\to t_i} \varphi_i(t) - \lim_{t\to t_i} \varphi_{i-1}(t) \in [-\pi, \pi].$$

Dies ist möglich auf Grund der Eigenschaft der Winkelfunktionen, dass für jede Winkelfunktion φ und jedes $k \in \mathbb{Z}$ auch $\varphi + 2\pi k$ eine Winkelfunktion ist. Ausserdem sind die α_i ausserhalb von $\pm\pi$ wohldefiniert, da die Winkelfunktion auf $(-\pi, \pi)$ eindeutig ist (vergleiche mit Bemerkung 3.5).

Im Fall $\alpha_i = \pm\pi$ wählen wir $\alpha_i := \pi$, falls die Kurve nach dem Eckpunkt in Durchlaufrichtung links weg geht und $\alpha_i := -\pi$, falls die Kurve nach rechts weg geht. Man betrachte dazu die anschliessende Illustration.

Die aus den einzelnen Teilen zusammengesetzte (nicht stetige) Funktion heisst Winkelfunktion φ von c. Die α_i heissen *Aussenwinkel* von c.

Die Umlaufzahl einer stückweise C^2-Kurve ist dann analog über die Winkelfunktion definiert:

$$\rho_c = \frac{1}{2\pi}(\varphi(L) - \varphi(0)).$$

4.6 Illustration

Um besser zu verstehen, was mit den Aussenwinkeln gemeint ist, betrachten wir folgende Illustration.

Man muss sich den Aussenwinkel α als den Winkel in $[-\pi, \pi]$ vorstellen, um welchen das Tangentialvektorfeld an einem Eckpunkt gedreht wird. Im Fall $\alpha = \pm\pi$ wählen wir aus Konsistenzgründen $+\pi$, falls die Kurve einen Linksknick und $-\pi$, falls die Kurve einen Rechtsknick erfährt.

4.7 Satz (Umlaufsatz für stückweise C^2-Kurven)

Für eine stetige, stückweise C^2-Kurve $c\colon [0, L] \to \mathbb{R}^2$, die zusätzlich regulär und geschlossen ist, gilt:

$$\int_c \kappa(t)\,\mathrm{d}s(t) + \sum_{i=1}^{n} \alpha_i = 2\pi\rho_c,$$

wobei ρ_c die Umlaufzahl und die α_i die Aussenwinkel von c bezeichnen.

Analog wie für reguläre einfach geschlossene C^2-Kurven, gilt auch für entsprechende stückweise C^2-Kurven $\rho = \pm 1$.

Beweis. Sei φ eine Winkelfunktion für c und seien $0 < t_1 < ... < t_n < L$ die zu den α_i gehörigen Eckpunkte. Zur Vereinfachung schreiben wir $t_0 := 0$ und $t_{n+1} := L$. Ausserdem definieren wir $\varphi_i(t) := \varphi_{|(t_i,t_{i+1})}(t)$ wie in der Definition der Winkelfunktion für stückweise C^2-Kurven. Für C^2-Kurven wissen wir, dass $\kappa(t) = \frac{\dot{\varphi}(t)}{|\dot{c}(t)|}$ gilt, also können wir

$$\kappa(t)_{|(t_i,t_{i+1})} = \frac{\dot{\varphi}_{|(t_i,t_{i+1})}(t)}{\left|\dot{c}_{|(t_i,t_{i+1})}(t)\right|} \text{ für } i = 0, ..., n,$$

schreiben. Damit haben wir für die Gesamtkrümmung:

$$\begin{aligned}
\int_c \kappa(t)\,\mathrm{d}s(t) &= \sum_{i=0}^{n} \int_{c_{|[t_i,t_{i+1}]}} \kappa(t)_{|(t_i,t_{i+1})}\,\mathrm{d}s(t) \\
&= \sum_{i=0}^{n} \int_{c_{|[t_i,t_{i+1}]}} \frac{\dot{\varphi}_{|(t_i,t_{i+1})}(t)}{\left|\dot{c}_{|(t_i,t_{i+1})}(t)\right|}\,\mathrm{d}s(t) \\
&= \sum_{i=0}^{n} \int_{t_i}^{t_i+1} \frac{\dot{\varphi}_i(t)}{\left|\dot{c}_{|(t_i,t_{i+1})}(t)\right|} \left|\dot{c}_{|(t_i,t_{i+1})}(t)\right|\,\mathrm{d}t \\
&= \sum_{i=0}^{n} \int_{t_i}^{t_i+1} \dot{\varphi}_i(t)\,\mathrm{d}t \\
&= \sum_{i=0}^{n} (\lim_{t\to t_{i+1}} \varphi_i(t) - \lim_{t\to t_i} \varphi_i(t)) \\
&= \varphi(L) - \varphi(0) - \sum_{i=1}^{n} (\lim_{t\to t_i} \varphi_i(t) - \lim_{t\to t_i} \varphi_{i-1}(t)) \\
&= 2\pi\rho_c - \sum_{i=1}^{n} \alpha_i.
\end{aligned}$$

Für den einfach geschlossenen Fall betrachten wir zunächst eine Kurve $c\colon [0, L] \to \mathbb{R}^2$, die nur an einer Stelle t_0 nicht C^2 ist. Induktiv folgt die Aussage dann für beliebig viele Stellen.

Sei φ eine Winkelfunktion für c. Die Winkelfunktion verhält sich ausserhalb der Sprungstelle wie eine Winkelfunktion einer C^2-Kurve, das heisst wir können für jedes $\epsilon > 0$ eine einfach geschlossene C^2-Kurve c_ϵ mit Winkelfunktion φ_ϵ finden, sodass $c_{|[0,t_0-\epsilon]} = c_{\epsilon|[0,t_0-\epsilon]}$ und $c_{|[t_0+\epsilon,L]} = c_{\epsilon|[t_0+\epsilon,L]}$. Für die c_ϵ kann der Hopf'sche Umlaufsatz angewendet werden, also gilt $\rho_{c_\epsilon} = \pm 1$, wobei wir ohne Beschränkung annehmen können, dass alle c_ϵ Umlaufzahl $+1$ haben. Daraus folgt:

$$
\begin{aligned}
2\pi &= \int_{c_\epsilon} \kappa_\epsilon(t)\,\mathrm{d}s(t) \\
&= \int_{c_{\epsilon|[0,t_0-\epsilon]}} \kappa_\epsilon(t)\,\mathrm{d}s(t) + \int_{t_0-\epsilon}^{t_0+\epsilon} \dot{\varphi}_\epsilon(t)\,\mathrm{d}t + \int_{c_{\epsilon|[t_0+\epsilon,L]}} \kappa_\epsilon(t)\,\mathrm{d}s(t) \\
&= \int_{c_{|[0,t_0-\epsilon]}} \kappa(t)\,\mathrm{d}s(t) + \int_{t_0-\epsilon}^{t_0+\epsilon} \dot{\varphi}_\epsilon(t)\,\mathrm{d}t + \int_{c_{|[t_0+\epsilon,L]}} \kappa(t)\,\mathrm{d}s(t) \\
&= \int_{c_{|[0,t_0-\epsilon]}} \kappa(t)\,\mathrm{d}s(t) + \varphi_\epsilon(t_0+\epsilon) - \varphi_\epsilon(t_0-\epsilon) + \int_{c_{|[t_0+\epsilon,L]}} \kappa(t)\,\mathrm{d}s(t) \\
&= \int_{c_{|[0,t_0-\epsilon]}} \kappa(t)\,\mathrm{d}s(t) + \varphi(t_0+\epsilon) - \varphi(t_0-\epsilon) + \int_{c_{|[t_0+\epsilon,L]}} \kappa(t)\,\mathrm{d}s(t) \quad \text{für alle } \epsilon > 0.
\end{aligned}
$$

Mit $\epsilon \to 0$ folgt:

$$
2\pi = \int_c \kappa(t)\,\mathrm{d}s(t) + \alpha,
$$

wobei α den Aussenwinkel bei t_0 bezeichnet. Damit haben wir $\rho_c = +1$ gezeigt. Entsprechend hätten wir $\rho_c = -1$ bekommen, wenn wir c in die andere Richtung parametrisiert hätten. □

4.8 Bemerkung

Wir haben jeweils angenommen, dass der Startpunkt kein Eckpunkt der geschlossenen Kurve ist. Dies spielt aber natürlich keine Rolle, da die Umlaufzahl unabhängig vom Startpunkt ist und wir auch einfach in einem anderen Punkt starten können.

4.9 Beispiele

4.9.1 Acht

Betrachte die Acht wie in Beispiel 2.8.4: $c\colon [0, 2\pi] \to \mathbb{R}^2$, $c(t) := \begin{pmatrix} \sin t\, \cos t \\ \sin t \end{pmatrix}$. Wir wissen schon aus Beispiel 4.3.2, dass die Krümmung durch $\kappa(t) = \frac{\sin^3 t + 3 \sin t \cos^2 t}{\left|(\cos^2 t - \sin^2 t)^2 + \cos^2 t\right|^3}$ gegeben ist und die Umlaufzahl 0 beträgt. Mit Hilfe von Satz 4.7 können wir die Umlaufzahl jetzt auch anders bestimmen.

Die Teilkurven $c_1 := c_{|[0,\pi]}$ und $c_2 := c_{|[\pi,2\pi]}$ sind einfach geschlossen und haben je einen Eckpunkt mit Aussenwinkeln α_1 und α_2. Daraus folgt:

$$\begin{aligned}\int_c \kappa(t)\,\mathrm{d}s(t) &= \int_{c_1} \kappa(t)\,\mathrm{d}s(t) + \int_{c_2} \kappa(t)\,\mathrm{d}s(t)\\ &= 2\pi\rho_{c_1} + 2\pi\rho_{c_2} - \alpha_1 - \alpha_2\\ &= 2\pi - 2\pi - \alpha_1 - \alpha_2\\ &= -(\alpha_1 + \alpha_2).\end{aligned}$$

Ausserdem haben wir $\dot{c}(t) = \begin{pmatrix}\cos^2 t - \sin^2 t\\ \cos t\end{pmatrix}$. Der Winkel zwischen $\dot{c}(0) = \begin{pmatrix}1\\1\end{pmatrix}$ und $\dot{c}(\pi) = \begin{pmatrix}1\\-1\end{pmatrix}$ beträgt gerade $\frac{\pi}{2}$. Damit beträgt der Aussenwinkel α_1 ebenfalls $\pi - \frac{\pi}{2} = \frac{\pi}{2}$. Analog folgt auch $\alpha_2 = -\frac{\pi}{2}$. Somit haben wir:

$$\begin{aligned}\rho_c &= \frac{1}{2\pi}\int_c \kappa(t)\,\mathrm{d}s(t)\\ &= -\frac{1}{2\pi}(\alpha_1 + \alpha_2)\\ &= 0,\end{aligned}$$

wie erwartet.

4.9.2 Astroide

Kommen wir zurück zur Astroide. Wir hatten da in Beispiel 4.3.3 das Problem, dass die analytisch berechnete Gesamtkrümmung deutlich vom numerisch berechneten Wert des Applets abgewichen ist. Wir hatten das auf die Nichtregularität der Parametrisierung zurückgeführt. Doch wo genau liegt das Problem und welchen Wert nimmt die Umlaufzahl der Astroide an? Diese Fragen können wir mit Hilfe von Satz 4.7 klären. Allerdings müssen wir dafür erst noch etwas arbeiten, da der Satz nur für reguläre stückweise C^2-Kurven gilt. Unsere Parametrisierung $c\colon [0, 2\pi] \to \mathbb{R}^2$, $c(t) := \begin{pmatrix}\cos^3 t\\ \sin^3 t\end{pmatrix}$ ist tatsächlich überall C^2, jedoch in $t = 0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi$ nicht regulär. Wir können die Astroide auf Kosten der Differenzierbarkeit regularisieren. Betrachte dazu folgende Parametertransformationen:

$$
\begin{aligned}
f_1 &\colon [0,1] \to [0, \frac{\pi}{2}], f_1(t) := \arcsin(\sqrt{t}),\\
f_2 &\colon [0,1] \to [\frac{\pi}{2}, \pi], f_2(t) := \arccos(-\sqrt{t}),\\
f_3 &\colon [0,1] \to [\pi, \frac{3\pi}{2}], f_3(t) := \arcsin(\sqrt{t}) + \pi,\\
f_4 &\colon [0,1] \to [\frac{3\pi}{2}, 2\pi], f_4(t) := \arccos(-\sqrt{t}) + \pi.
\end{aligned}
$$

Man beachte, dass die f_i keine regulären Parametertransformationen sind, da alle f_i in 1 nicht differenzierbar sind. Dies spielt aber hier keine Rolle, da die Krümmung in den Eckpunkten sowieso nicht definiert ist. In allen anderen Punkten sind die Parametertransformationen regulär und die Krümmung daher auch invariant unter Verknüpfung mit diesen. Insbesondere ändert sich die Umlaufzahl auch nicht. Betrachten wir also die Umparametrisierungen:

$$
\begin{aligned}
c_1(t) := (c \circ f_1)(t) &= \begin{pmatrix} (1-t)^{\frac{3}{2}} \\ t^{\frac{3}{2}} \end{pmatrix},\\
c_2(t) := (c \circ f_2)(t) &= \begin{pmatrix} -t^{\frac{3}{2}} \\ (1-t)^{\frac{3}{2}} \end{pmatrix},\\
c_3(t) := (c \circ f_3)(t) &= \begin{pmatrix} -(1-t)^{\frac{3}{2}} \\ -t^{\frac{3}{2}} \end{pmatrix},\\
c_4(t) := (c \circ f_4)(t) &= \begin{pmatrix} t^{\frac{3}{2}} \\ -(1-t)^{\frac{3}{2}} \end{pmatrix}.
\end{aligned}
$$

Die Vereinigung dieser Teilkurven bilden eine reguläre, stückweise C^2 Parametrisierung der Astroide und daher können wir den Satz 4.7 anwenden. Das Krümmungsintegral bleibt -2π, allerdings haben wir jetzt die Aussenwinkel $\alpha_{1,2,3,4}$ zu berechnen. Berechnen wir erst die Ableitungen der c_i:

$$
\begin{aligned}
\dot{c_1}(t) = \begin{pmatrix} -\frac{3}{2}\sqrt{1-t} \\ \frac{3}{2}\sqrt{t} \end{pmatrix}, \dot{c_2}(t) &= \begin{pmatrix} -\frac{3}{2}\sqrt{t} \\ -\frac{3}{2}\sqrt{1-t} \end{pmatrix},\\
\dot{c_3}(t) = \begin{pmatrix} \frac{3}{2}\sqrt{1-t} \\ -\frac{3}{2}\sqrt{t} \end{pmatrix}, \dot{c_4}(t) &= \begin{pmatrix} \frac{3}{2}\sqrt{t} \\ \frac{3}{2}\sqrt{1-t} \end{pmatrix},\\
\dot{c_1}(1) = \begin{pmatrix} 0 \\ \frac{3}{2} \end{pmatrix}, \dot{c_2}(0) = \begin{pmatrix} 0 \\ -\frac{3}{2} \end{pmatrix}, \dot{c_2}(1) &= \begin{pmatrix} -\frac{3}{2} \\ 0 \end{pmatrix}, \dot{c_3}(0) = \begin{pmatrix} \frac{3}{2} \\ 0 \end{pmatrix},\\
\dot{c_3}(1) = \begin{pmatrix} 0 \\ -\frac{3}{2} \end{pmatrix}, \dot{c_4}(0) = \begin{pmatrix} 0 \\ \frac{3}{2} \end{pmatrix}, \dot{c_4}(1) &= \begin{pmatrix} \frac{3}{2} \\ 0 \end{pmatrix}, \dot{c_1}(0) = \begin{pmatrix} -\frac{3}{2} \\ 0 \end{pmatrix}.
\end{aligned}
$$

Damit sind alle Aussenwinkel $\alpha_i = \pi$, da die Astroide in allen Eckpunkten einen Linksknick erfährt. Nun können wir die Umlaufzahl bestimmen:

$$\begin{aligned}\rho_c &= \frac{1}{2\pi}\,(\int_c \kappa(t)\,\mathrm{d}s(t) + \sum_{i=1}^{4}\alpha_i) \\ &= \frac{1}{2\pi}\,(-2\pi + 4\pi) \\ &= 1.\end{aligned}$$

Die Umlaufzahl der Astroide ist also $+1$, wie uns das Applet schon suggeriert hat. Das Applet berechnet das Krümmungsintegral nicht für exakte Kurven, sondern lediglich für beliebig genaue Approximationen. Das bedeutet gerade, dass bei der Astroide die Ecken abgerundet werden und es somit keine zu berücksichtigenden Winkel gibt. Wir erinnern uns, dass wir für den Beweis des Umlaufsatzes für stückweise C^2-Kurven ebenfalls etwas ähnliches gemacht haben. Wir haben da auch versucht die ursprüngliche Kurve durch beliebig genaue C^2-Kurven zu ersetzen und damit den Satz gezeigt.

4.10 Aufgaben

Öffne bitte das Applet, um die Aufgaben bearbeiten zu können.

i) Betrachte die Gesamtkrümmung für die vorgegebenen Kurven. Für welche Kurven gilt $\int_c \kappa(t)\,\mathrm{d}s(t) = 0$, 2π oder -2π?

ii) Zeichne eigene, einfach geschlossene Kurven und verifiziere damit den Hopf'schen Umlaufsatz.

iii) Wahr oder falsch? Überlege dir zuerst die Antworten und überprüfe sie dann mit dem Applet.

   1. Die Gesamtkrümmung ist unabhängig davon in welcher Richtung man die Kurve durchläuft.
   2. Die Gesamtkrümmung einer geschlossenen Kurve ist invariant unter Skalierungen.
   3. Man kann für jedes $\rho \in \mathbb{Z}$ eine geschlossene Kurve finden, sodass deren Gesamtkrümmung $2\pi\rho$ beträgt.

iv) Versuche einen rechten Winkel zu zeichnen. Was fällt dir im Bezug auf die Gesamtkrümmung auf? Entspricht das deiner Erwartung?

v) Berechne mit Hilfe des Hopf'schen Umlaufsatzes die Winkelsumme eines konvexen n-Ecks.

vi) Sei c eine geschlossene, reguläre C^2-Kurve, die in genau einem Punkt (ausser dem Anfangs- und Endpunkt natürlich) nicht injektiv ist. Ausserden habe dieser Punkt genau zwei Urbilder t_1 und t_2 und der Winkel zwischen $\dot{c}(t_1)$ und $\dot{c}(t_2)$ sei kein Vielfaches von π. Zeige, dass diese Kurve dann Umlaufzahl -2, 0 oder +2 hat.

vii) Finde mit Hilfe des Applets Gegenbeispiele zur Aussage von Aufgabe vi) im Fall, wo der Punkt mehr als zwei Urbilder hat und im Fall, wo der Winkel zwischen $\dot{c}(t_1)$ und $\dot{c}(t_2)$ ein Vielfaches von π ist.

4.11 Lösungen

i) -

ii) -

iii)
1. Falsch. Sie wechselt gerade das Vorzeichen. Für Kurven mit Umlaufzahl 0 gilt die Aussage natürlich.
2. Richtig. Dies kann man auf verschiedene Arten begründen. Skalieren wir die Kurve, so ändert sich nichts an der Winkelfunktion und daher auch nichts an der Umlaufzahl. Man kann es aber auch direkt über das Integral nachrechnen.
3. Richtig. Zum Beispiel kann man einfach ρ-mal den Kreis in positiver oder negativer Richtung durchlaufen. Für den Fall $\rho = 0$ haben wir mit der Acht schon ein Beispiel gesehen.

iv) Wenn man den rechten Winkel genug genau zeichnet, springt die Gesamtkrümmung um $\frac{\pi}{2}$. Das ist auch genau das, was wir erwartet und auf den letzten paar Seiten gezeigt haben. Bleibt nur zu sagen, dass das Applet die Aussenwinkel schon im Integral mit einberechnet, während wir analytisch die Aussenwinkel separat addieren müssen. Vergleiche dazu auch mit dem Beispiel 3.10.2.

v) Die Seiten den n-Ecks sind Geraden und damit ist die Krümmung überall 0. Ausserdem ist ein konvexes n-Eck einfach geschlossen, die Umlaufzahl also +1, wenn wir das n-Eck gegen den Uhrzeigersinn parametrisieren. Daraus folgt, dass die Summe der Aussenwinkel gerade 2π sein muss. Seien α_i die Aussenwinkel und β_i die Innenwinkel. Wegen $\beta_i = \pi - \alpha_i$ haben wir also $\sum_{i=1}^{n} \beta_i = \sum_{i=1}^{n} \pi - \alpha_i = n\pi - 2\pi = (n-2)\pi$.

vi) Die Bedingung, dass der Winkel zwischen $\dot{c}(t_1)$ und $\dot{c}(t_2)$ kein Vielfaches von π ist, stellt sicher, dass sich die Kurve in $P := c(t_1) = c(t_2)$ selbst schneidet und nicht bloss berührt. Da P genau zwei Urbilder hat, gibt es an diesem Punkt genau zwei Tangetialvektoren. Sei $\alpha \in [0, 2\pi]$ der Winkel zwischen diesen beiden Tangentialvektoren. Da die Kurve in genau einem Punkt P nicht injektiv ist, können wir die Kurve in zwei einfach geschlossene, stückweise C^2-Kurven c_1 und c_2 zerlegen. Die Aussenwinkel bei P sind dann gerade $\pi - \alpha$ und $-(\pi - \alpha)$. Daraus können wir nun die Umlaufzahl bestimmen:

$$\begin{aligned}\rho_c &= \frac{1}{2\pi}\int_c \kappa(t)\mathrm{d}s(t)\\ &= \frac{1}{2\pi}\left(\int_{c_1}\kappa(t)\mathrm{d}s(t)+\int_{c_2}\kappa(t)\mathrm{d}s(t)\right)\\ &= \frac{1}{2\pi}\left(2\pi\rho_{c_1}+2\pi\rho_{c_2}-(\pi-\alpha)+\pi-\alpha\right)\\ &= \rho_{c_1}+\rho_{c_2}.\end{aligned}$$

Da die Teilkurven c_1 und c_2 einfach geschlossen sind, folgt mit Satz 4.7, dass deren Umlaufzahl ± 1 sein muss. Damit gibt es für ρ_c nur die Möglichkeiten -2, 0 und +2.

vii) Diese beiden Kurven widerlegen die Aussagen:

Die linke Kurve hat Umlaufzahl 4, die rechte hat Umlaufzahl 1, wie man sich überlegen oder mit dem Applet nachprüfen kann.

5 Algorithmen und Dokumentation

Die Idee dieses Applets ist es, eine Visualisierung der Krümmung und des Krümmungintegrals von ebenen Kurven zu geben. Dazu soll man selbstständig Kurven zeichnen können und vom Applet dann die entsprechende Krümmung, sowie das Krümmungsintegral zurückbekommen. Ausserdem soll es vorgefertigte Kurven bieten, welche analog auf Krümmung und Gesamtkrümmung untersucht werden. Diese Dokumentation zeigt die Entstehung dieses Applets vom ersten Versuch bis zum fertigen Applet. Es werden alle benötigten Algorithmen in Pseudocode angegeben. MATLAB-Codes zur direkten Verwendung können von der Homepage heruntergeladen werden.

Problemstellung: Für jede Kurve, bestehend aus einer Menge von Pixeln, soll die Krümmungsfunktion sowie das Krümmungsintegral graphisch dargestellt werden.

5.1 Erste Probleme

Wir haben bisher immer auf dem $\mathbb{R}^2$ gearbeitet und dort unsere Begriffe wie C^2, *regulär* und *Krümmung* definiert. Doch ein Computer arbeitet nur mit Pixeln, also auf einer diskreten Menge. Zeichnen wir mit einem beliebigen Graphikprogramm eine Kurve auf den

Bildschirm, so sehen wir bei guter Auflösung auch wirklich eine Kurve. Der Computer sieht allerdings nur eine diskrete geordnete Menge aus Punkten mit ganzzahligen Koordinaten. Ein Pixel entspricht dann einer Einheit. Würden wir diese Punkte dann wiederum auf einem Blatt Papier als ausgefüllte Kästchen zeichnen, würden wir nur erahnen können, dass diese Menge aus Punkten eine Kurve darstellt. Entfernen wir uns allerdings von dem Blatt Papier, so sehen wir tatsächlich eine Kurve. So funktioniert auch die Anzeige bei einem Computer. Der Computer zeigt uns eine Menge von Punkten an und unser Auge fügt diese zu einer Kurve zusammen. Als Eingabe für unseren Algorithmus bekommen wir also bestenfalls eine geordnete Menge aus Punkten. Daraus ergeben sich direkt einige Fragen, die vor einer allfälligen Implementierung geklärt werden müssen:

i) Macht es bei einer Menge aus Pixeln überhaupt Sinn von Krümmung zu sprechen?

ii) Wie sind unsere Begriffe wie C^2, *Regularität* und *Krümmung* im Kontext der Pixel zu verstehen? Zum Beispiel: Was bedeutet es für eine solche Kurve C^2 zu sein?

iii) Über was genau integrieren wir, wenn wir das Krümmungsintegral bestimmen wollen?

Es geht also zunächst darum, unsere Begriffe diesem Problem entsprechend neu zu erklären.

5.2 Erster Lösungsansatz

Betrachten wir zunächst unsere Eingabe als geordnete Menge von Punkten in der Ebene. Verbinden wir diese Punkte in der Reihenfolge der Eingabe, erhalten wir ein Polygon. Begnügen wir uns mit der Eigenschaft *stückweise* C^2, so entspricht diese Kurve unseren Anforderungen und wir können die Krümmung explizit ausrechnen. Betrachten wir dieses Vorgehensweise genauer, stellen wir allerdings fest, dass sich dieser Ansatz als völlig untauglich erweist, denn die Krümmung eines Polygons ist ausserhalb der Ecken überall 0. Nehmen wir zusätzlich an, dass die Pixel sich jeweils entweder an einer Seite oder an einer Ecke berühren, so gibt es für die Aussenwinkel nur die Möglichkeiten 0, $\pm\frac{\pi}{4}$, $\pm\frac{\pi}{2}$, $\pm\frac{3\pi}{4}$ und $\pm\pi$. Dies wäre ein langweiliges Bild und würde nicht dem entsprechen, was wir uns unter der Krümmung vorstellen.

In unserem ersten Lösungsansatz hatten wir es gar nicht erst versucht, Begriffe wie C^2 oder *Regularität* für unser Problem zu interpretieren. Dies war einer der Gründe, warum dieser Ansatz schlussendlich untauglich war. Wir müssen also einen Ansatz finden, der diese Begriffe benutzt, um dann auch die Krümmung nach unserer Vorstellung zu definieren. Dies werden wir in unserem zweiten Lösungsansatz versuchen.

5.3 Zweiter Lösungsansatz

Zeichnen wir eine Kurve auf dem Bildschirm, so erhält der Computer eine Menge von Pixeln und verunmöglicht uns damit die Krümmung direkt über die Parametrisierung auszurechnen. Die zentralen Fragen die sich hier stellen, sind:

Ist dieser Vorgang in irgendeinem Sinne umkehrbar? Können wir eine Kurve aus den Pixeln zurückgewinnen?

Das Ziel wird also sein, aus der Menge von Pixeln eine Parametrisierung zu konstruieren, die der ursprünglichen Kurve möglichst nahe kommt. Haben wir eine solche Parametrisierung gefunden, so können wir Begriffe wie die *Krümmung* erklären, indem wir sie einfach auf diese Parametrisierung anwenden. Eine gezeichnete Kurve heisst dann beispielsweise C^2, falls die entsprechend berechnete Parametrisierung C^2 ist. Unser anfängliches Problem hat sich also darauf reduziert, anhand der Pixel eine Parametrisierung zu konstruieren. Damit hätten wir schonmal direkt die anfänglichen Fragen geklärt. Bleibt die Frage wie man eine solche Parametrisierung erhält. Eine typische Methode dafür ist die Polynominterpolation. Dabei gibt man sich zuerst den Grad eines Polynoms vor und sucht dann sowohl in x- wie auch in y-Richtung das Polynom m-ten Grades, welches den gegebenen Pixel $((x_1, y_1), ..., (x_n, y_n))$ am nächsten kommt. Als Kriterium benutzt man die sogenannte *Kleinste-Quadrate-Methode*, welche das Quadrat der Abstände zwischen Kurve und Pixel minimiert. Diese Polynome zu finden ist in der Tat nicht besonders schwierig. Es sind lediglich zwei lineare Gleichungssysteme zu lösen.

Betrachten wir zunächst alles nur in x-Richtung, die y-Richtung funktioniert analog. Die zu minimierende Funktion ist definiert durch $f(a_1, ..., a_m) := \sum_{i=1}^{n} (p(i) - x_i)^2$, wobei $p(X) := \sum_{j=0}^{m} a_j \, X^j$ das gesuchte Polynom ist.

Das Minimum dieser Funktion findet man durch Ableiten:

$$\begin{aligned} \frac{\partial f}{\partial a_k} &= 2 \sum_{i=1}^{n} \left(\sum_{j=0}^{m} a_j \, i^j - x_i \right) i^k \\ &\stackrel{!}{=} 0. \end{aligned}$$

Dies ist äquivalent zu folgendem linearen Gleichungssystem:

$$\begin{pmatrix} \sum_{i=1}^{n} i^0 & \sum_{i=1}^{n} i^1 & \cdots & \sum_{i=1}^{n} i^m \\ \sum_{i=1}^{n} i^1 & \sum_{i=1}^{n} i^2 & \cdots & \sum_{i=1}^{n} i^{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} i^m & \cdots & \cdots & \sum_{i=1}^{n} i^{2m} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} x_i \, i^0 \\ \sum_{i=1}^{n} x_i \, i^1 \\ \vdots \\ \sum_{i=1}^{n} x_i \, i^m \end{pmatrix}.$$

Dieses Gleichungssystem ist genau dann lösbar, wenn $m < n$ gilt. Der Fall $m \geq n$ ist aber auch nicht interessant, da wir beliebig viele Polynome m-ten Grades finden können, die genau durch diese n Punkte $x_1, ..., x_n$ verlaufen. Formal sieht der Algorithmus dann so aus:

Algorithm 1 Polynominterpolation

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$
- gewünschter Grad des Polynoms m

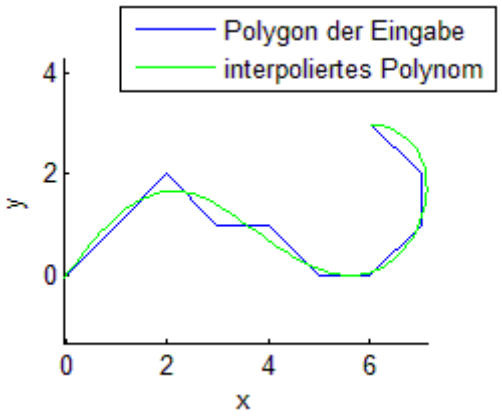
Output:

- je ein Polynom für x und y, welches den Pixeln bezüglich der *Kleinste-Quadrate-Methode* am nächsten kommt

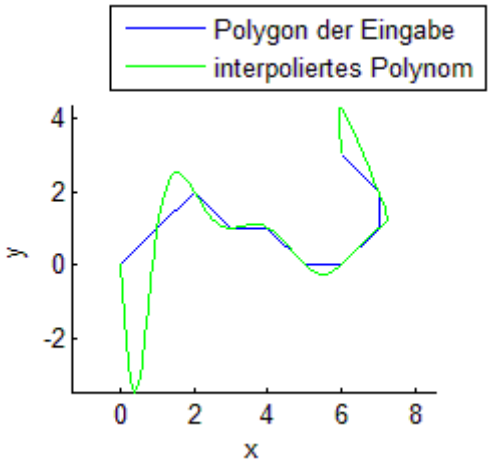
1: **for all** $j, k = 0$ **to** m **do**
2: $\quad A_{kj} := \sum_{i=1}^{n} i^{j+k}$
3: **end for**
4: **for all** $k = 0$ **to** m **do**
5: $\quad b_k := \sum_{i=1}^{n} x_i\, i^k$
6: $\quad c_k := \sum_{i=1}^{n} y_i\, i^k$
7: **end for**
8: Löse die linearen Gleichungssysteme $Ap = b$ und $Aq = c$.
9: **return** $p(t) := \sum_{k=0}^{m} p_k\, t^k$ **and** $q(t) := \sum_{k=0}^{m} q_k\, t^k$

Die Parametrisierung lautet dann $c\colon [1, n] \to \mathbb{R}^2$, $c(t) := \begin{pmatrix} p(t) \\ q(t) \end{pmatrix}$.

In der folgenden Grafik sehen wir ein kleines Beispiel für $n = 10, m = 5$. Zur besseren Veranschaulichung wurden die Pixel noch zusätzlich zu einem Polygon verbunden.



Die nächste Frage, die sich unweigerlich stellt, ist die nach dem optimalen Grad des Polynoms. Der Grad darf auf jeden Fall nicht zu klein sein, da das Polynom ja der Kurve möglichst folgen soll. Allerdings darf der Grad auch nicht zu gross sein, wie wir in der folgenden Grafik sehen ($n = 10, m = 9$).



Das Polynom geht jetzt zwar durch alle Punkte, aber sie weicht vor allem an den Enden sehr stark vom Polygon der Eingabe ab. In zahlreichen Tests hat sich $m \approx \frac{n}{2}$ als gute Wahl herausgestellt. Darauf wollen wir aber an dieser Stelle nicht näher eingehen.

5.4 Splines

Wir haben in letzten Abschnitt gesehen, dass sich Polynome als Approximation für kleine Eingabemengen relativ gut eignen. In der Praxis ist die Eingabe allerdings in der Regel viel grösser, als die, welche wir bisher betrachtet haben. Eine typische Eingabe wären zum Beispiel $n = 500$ Pixel. Um eine sinnvolle Approximation dafür zu finden, bräuchten wir ungefär ein Polynom vom Grad $m = 250$. Betrachten wir das zu lösende lineare Gleichungssystem, stellen wir aber fest, dass der Rechenaufwand für grosse m sehr schnell sehr gross wird. Ausserdem ist die Matrix für grosse m schlecht konditioniert, da die Einträge exponentiell anwachsen. Das bedeutet, dass eine numerische Lösung des Gleichungssystems grosse Fehler aufweisen kann. Daher wollen wir nun versuchen, die Kurve nur stückweise zu approximieren und dann an den Endpunkten zusammen zu setzen. Dies scheint auch im praktischen Sinne sinnvoll, denn wenn wir eine Kurve zeichnen, wollen wir nicht, dass sich die Approximation in den ersten Punkten noch ändert während wir schon die letzten Punkte zeichnen.

Die Lösung an dieser Stelle heisst *Spline*. Splines sind Kurven, die sich aus einzelnen Kurven zusammensetzen und an den Endpunkten gewisse Stetigkeitseigenschaften erfüllen. In unserem Fall wäre das C^2, denn wir wollen ja schlussendlich die Krümmung bestimmen. Sei p_{alt} das vorhergehende Polynom und p_{neu} das Polynom, das wir an p_{alt} ansetzen wollen. Der Grad der Polynome sei m und die Anzahl Pixel pro Polynom sei n. Damit wir am Verknüpfungspunkt C^2 haben, müssen wir folgende Anfangsbedingungen erfüllen:

$$
\begin{aligned}
p_{neu}(1) &= p_{alt}(n),\\
\dot{p}_{neu}(1) &= \dot{p}_{alt}(n),\\
\ddot{p}_{neu}(1) &= \ddot{p}_{alt}(n).
\end{aligned}
$$

Dies führt zu folgenden Lagrangefunktion, die es zu minimieren gilt:

$$
\begin{aligned}
L(a_1, ..., a_m, \lambda, \mu, \nu) = &\sum_{i=1}^{n} (p_{neu}(i) - x_i)^2 - \lambda \left(p_{neu}(1) - p_{alt}(n)\right)\\
&- \mu \left(\dot{p}_{neu}(1) - \dot{p}_{alt}(n)\right) - \nu \left(\ddot{p}_{neu}(1) - \ddot{p}_{alt}(n)\right).
\end{aligned}
$$

Daraus ergibt sich wieder ein lineares Gleichungssystem, das man analog wie in Algorithmus 1 lösen kann.

Essentiell für die spätere Spline-Interpolation ist folgender Algorithmus, der ähnlich funktioniert wie Algorithmus 1, nur dass wir eben am Anfangspunkt noch zusätzlich die Bedingung C^2 haben.

Algorithm 2 Polynominterpolation mit Anfangsbedingungen

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$
- gewünschter Grad des Polynoms m
- die Anfangsbedingungen $p(1), \dot{p}(1), \ddot{p}(1)$ und $q(1), \dot{q}(1), \ddot{q}(1)$

Output:

- je ein Polynom für x und y, welches den Pixeln bezüglich der *Kleinste-Quadrate-Methode* am nächsten kommt und die Anfangsbedingungen erfüllt

1: // *Lösen des Extremwertproblems mit Randbedingungen führt zu folgendem linearen Gleichungssystem:* //
2: **for all** $k = 0$ **to** m **do**
3: **for all** $j = 0$ **to** m **do**
4: $A_{kj} := 2 \sum_{i=1}^{n} i^{j+k}$
5: **end for**
6: $A_{k(m+1)} := -1$, $A_{k(m+2)} := -k$, $A_{k(m+3)} := -k(k-1)$
7: **end for**
8: **for all** $j = 0$ **to** m **do**
9: $A_{(m+1)j} := 1$, $A_{(m+2)j} := j$, $A_{(m+3)j} := j(j-1)$
10: **end for**
11: **for all** $j, k = m+1$ **to** $m+3$ **do**
12: $A_j k := 0$
13: **end for**
14: **for all** $k = 0$ **to** m **do**
15: $b_k := 2 \sum_{i=1}^{n} x_i \, i^k$
16: $c_k := 2 \sum_{i=1}^{n} y_i \, i^k$
17: **end for**
18: $b_{m+1} := p(1)$, $b_{m+2} := \dot{p}(1)$, $b_{m+3} := \ddot{p}(1)$
19: $c_{m+1} := q(1)$, $c_{m+2} := \dot{q}(1)$, $c_{m+3} := \ddot{q}(1)$
20: Löse die linearen Gleichungssysteme $Ap = b$ und $Aq = c$.
21: **return** $p(t) := \sum_{k=0}^{m} p_k \, t^k$ **and** $q(t) := \sum_{k=0}^{m} q_k \, t^k$

Algorithmus 2 erlaubt uns, die Kurven so zusammenzusetzen, dass die Krümmung überall definiert ist. Dadurch haben wir drei wesentliche Vorteile gegenüber der normalen Polynominterpolation gewonnen:

i) Die Matrizen der zu lösenden Gleichungssysteme sind nicht beliebig gross. Das Lösen der Gleichungssysteme erfolgt dadurch wesentlich schneller und grössere numerische Fehler auf Grund der schlechten Konditionierung können vermieden werden.

ii) Die Approximation hängt nur lokal von den eingegebenen Werten ab. Berechnen wir beispielsweise die Kurve in den ersten 10 Pixel, so hängt das Ergebnis nicht von den nächsten 10 gezeichneten Pixel ab.

iii) Wir können die Approximation besser kontrollieren, da sie gleichmässiger ist. Es kann dadurch nicht passieren, dass an manchen Stellen zu viele Freiheitsgrade verwendet werden. Wenn zu viele Freiheitsgrade verwendet werden, beginnt die Kurve um das Polygon zu oszillieren (vergleiche die Grafik im Abschnitt oben).

Als nächstes wollen wir geschlossene Eingaben betrachten. In diesem Kontext bedeutet das einfach nur $(x_1, y_1) = (x_n, y_n)$. Wir wollen, dass die Approximation auch in diesem Punkt C^2 ist. Mit dem Wissen über Splines ist dies aber kein Problem mehr, wir haben lediglich eine weitere Randbedingung für das letzte Stück.

Algorithm 3 Polynominterpolation mit Anfangs- und Endbedingungen

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$
- gewünschter Grad des Polynoms m
- die Anfangsbedingungen $p(1), \dot{p}(1), \ddot{p}(1)$ und $q(1), \dot{q}(1), \ddot{q}(1)$, sowie die Endbedingungen $p(n), \dot{p}(n), \ddot{p}(n)$ und $q(n), \dot{q}(n), \ddot{q}(n)$

Output:

- je ein Polynom für x und y, welches den Pixeln bezüglich der *Kleinste-Quadrate-Methode* am nächsten kommt und die Randbedingungen erfüllt.

1: // *Lösen des Extremwertproblems mit Randbedingungen führt zu folgendem linearen Gleichungssystem:* //
2: **for all** $k = 0$ **to** m **do**
3: **for all** $j = 0$ **to** m **do**
4: $A_{kj} := 2 \sum_{i=1}^{n} i^{j+k}$
5: **end for**
6: $A_{k(m+1)} := -1$, $A_{k(m+2)} := -k$, $A_{k(m+3)} := -k(k-1)$
7: $A_{k(m+4)} := -n^k$, $A_{k(m+5)} := -kn^{k-1}$, $A_{k(m+6)} := -k(k-1)n^{k-2}$
8: **end for**
9: **for all** $j = 0$ **to** m **do**
10: $A_{(m+1)j} := 1$, $A_{(m+2)j} := j$, $A_{(m+3)j} := j(j-1)$
11: $A_{(m+4)j} := n^j$, $A_{(m+5)j} := jn^{j-1}$, $A_{(m+6)j} := j(j-1)n^{j-2}$
12: **end for**
13: **for all** $j, k = m+1$ **to** $m+g$ **do**
14: $A_j k := 0$
15: **end for**
16: **for all** $k = 0$ **to** m **do**
17: $b_k := 2 \sum_{i=1}^{n} x_i \, i^k$
18: $c_k := 2 \sum_{i=1}^{n} y_i \, i^k$
19: **end for**
20: $b_{m+1} := p(1)$, $b_{m+2} := \dot{p}(1)$, $b_{m+3} := \ddot{p}(1)$
21: $b_{m+4} := p(n)$, $b_{m+5} := \dot{p}(n)$, $b_{m+6} := \ddot{p}(n)$
22: $c_{m+1} := q(1)$, $c_{m+2} := \dot{q}(1)$, $c_{m+3} := \ddot{q}(1)$
23: $c_{m+4} := q(n)$, $c_{m+5} := \dot{q}(n)$, $c_{m+6} := \ddot{q}(n)$
24: Löse die linearen Gleichungssysteme $Ap = b$ und $Aq = c$.
25: **return** $p(t) := \sum_{k=0}^{m} p_k \, t^k$ **and** $q(t) := \sum_{k=0}^{m} q_k \, t^k$

Jetzt müssen wir die Algorithmen nur noch zusammen setzen, um den gesamten Algorithmus für die Spline-Interpolation zu erhalten.

Algorithm 4 Spline-Interpolation

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$
- gewünschter Grad der einzelnen Polynomstücke m

Output:

- je einen Spline für x und y, welcher den Pixeln bezüglich der *Kleinste-Quadrate-Methode* am nächsten kommt

1: $l := \lfloor \frac{n}{10} \rfloor$
2: *// Die Länge der einzelnen Stücke ist hier willkürlich gewählt, man könnte auch jeden anderen Wert nehmen oder es als Variable der Eingabe freigeben. //*
3: Führe Algorithmus 1 für $((x_1, y_1), ..., (x_{10}, y_{10}))$ und Grad m aus und erhalte die ersten Teile $r_1(t)$ und $s_1(t)$ des Splines.
4: $d_{x1} := r_1(1)$, $d_{x2} := \dot{r}_1(1)$, $d_{x3} := \ddot{r}_1(1)$,
5: $d_{y1} := s_1(1)$, $d_{y2} := \dot{s}_1(1)$, $d_{y3} := \ddot{s}_1(1)$
6: $e_{x1} := r_1(10)$, $e_{x2} := \dot{r}_1(10)$, $e_{x3} := \ddot{r}_1(10)$
7: $e_{y1} := s_1(10)$, $e_{y2} := \dot{s}_1(10)$, $e_{y3} := \ddot{s}_1(10)$
8: **for all** $i = 2$ **to** $l - 1$ **do**
9: Führe Algorithmus 2 für $((x_{10(i-1)}, y_{10(i-1)}), ..., (x_{10i}, y_{10i}))$, Grad m und Anfangsbedingungen e_{x1}, e_{x2}, e_{x3} sowie e_{y1}, e_{y2}, e_{y3} aus und erhalte die Teile $r_i(t)$ und $s_i(t)$ des Splines.
10: $e_{x1} = r_i(11)$, $e_{x2} = \dot{r}_i(11)$, $e_{x3} = \ddot{r}_i(11)$
11: $e_{y1} = s_i(11)$, $e_{y2} = \dot{s}_i(11)$, $e_{y3} = \ddot{s}_i(11)$
12: **end for**
13: *// Falls die Kurve geschlossen ist, benutzen wir hier Algorithmus 3, ansonsten nochmal Algorithmus 2. //*
14: **if** $(x_1, y_1) = (x_n, y_n)$ **then**
15: *// Wir geben an dieser Stelle etwas mehr Freiheitsgrade, da wir hier im Allgemeinen eine grössere Eingabe und zusätzlich noch Endbedingungen zu erfüllen haben. //*
16: Führe Algorithmus 3 für $((x_{10(l-1)}, y_{10(l-1)}), ..., (x_n, y_n))$, Grad $m + 3$, Anfangsbedingungen e_{x1}, e_{x2}, e_{x3} sowie e_{y1}, e_{y2}, e_{y3} und Endbedingungen d_{x1}, d_{x2}, d_{x3} sowie d_{y1}, d_{y2}, d_{y3} aus und erhalte die letzten Teile $r_l(t)$ und $s_l(t)$ des Splines.
17: **else**
18: Führe Algorithmus 2 für $((x_{10(l-1)}, y_{10(l-1)}), ..., (x_n, y_n))$, Grad m, Anfangsbedingungen e_{x1}, e_{x2}, e_{x3} sowie e_{y1}, e_{y2}, e_{y3} aus und erhalte die letzten Teile $r_l(t)$ und $s_l(t)$ des Splines.
19: **end if**
20: **return** $(r_1(t), ..., r_l(t))$ **and** $(s_1(t), ..., s_l(t))$

Die Parametrisierung lautet dann $c\colon [1,n] \to \mathbb{R}^2$, $c(t) := \begin{pmatrix} r(t) \\ s(t) \end{pmatrix}$, wobei $r, s\colon [1,n] \to \mathbb{R}^2$ die zusammengesetzten Kurven seien:

$$
\begin{aligned}
&r(t) := r_1(t) \text{ für } t \in [1,10],\\
&r(t) := r_i(t - 10(i-1) + 1) \text{ für } t \in [10(i-1), 10i] \text{ und } i = 2, ...l-1,\\
&r(t) := r_l(t - 10(l-1) + 1) \text{ für } t \in [10(l-1), n],\\
&s(t) := s_1(t) \text{ für } t \in [1,10],\\
&s(t) := s_i(t - 10(i-1) + 1) \text{ für } t \in [10(i-1), 10i] \text{ und } i = 2, ...l-1,\\
&s(t) := s_l(t - 10(l-1) + 1) \text{ für } t \in [10(l-1), n].
\end{aligned}
$$

5.5 Implementierung der Krümmung

Mit Hilfe der gewonnenen Parametrisierung können wir die Krümmung explizit ausrechnen lassen. Da Polynome leicht analytisch abgeleitet werden können, brauchen wir dazu auch keine numerische Differentiation.

Algorithm 5 Krümmung

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$
- gewünschter Grad der Approximation m

Output:

- die Krümmungsfunktion $\kappa(t)$

1: Wende Algorithmus 4 auf x und y mit Grad m an und erhalte den Spline c.
2: $\kappa(t) := \frac{1}{|\dot{c}(t)|^3} \det(\dot{c}(t), \ddot{c}(t))$
3: **return** $\kappa(t)$

Die Gesamtkrümmung erhalten wir durch eine Integration der Krümmungsfunktion. Die Krümmungsfunktion ist leider kein Polynom mehr, sodass wir eine numerische Integration anwenden müssen. Wir benutzen dazu die *Simpsonregel*, um eine *adaptive Quadratur* durchzuführen. Adaptiv bedeutet hier, dass wir die Schrittweite für die Simpsonregel der Krümmung anpassen. Dadurch vermeiden wir grössere numerische Fehler, falls die Krümmung an einigen Stellen stark ansteigt oder abfällt.

Algorithm 6 numerische Integration

Input:

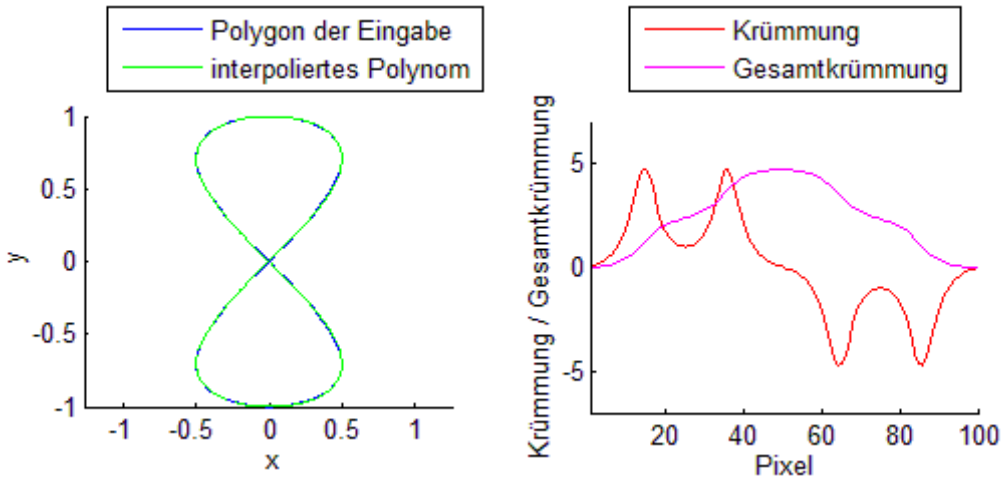
- die zu integrierende Funktion f
- die Integrationsgrenzen a und b
- die Toleranz *tol*, ein Mass für den maximal zulässigen Fehler der numerischen Integration
- einen Startwert Q_{in}, der bei der allerersten Ausführung beliebig gewählt werden kann (typischerweise 0 oder 1)

Output:

- das numerisch berechnete Integral von f über das Intervall $[a, b]$

1: $h := b - a$
2: *Unterteile das Integrationsintervall in zwei Hälften und verwende die Simpsonregel:*
3: $Q_L := \frac{h}{12}\left(f(a) + 4f(a + \frac{h}{4}) + f(a + \frac{h}{2})\right)$
4: $Q_R := \frac{h}{12}\left(f(a + \frac{h}{2}) + 4f(a + \frac{3h}{4}) + f(b)\right)$
5: $Q := Q_L + Q_R$
6: *Falls der Fehler klein genug ist, führe noch eine abschliessende Extrapolation durch. Falls nicht, unterteile das Intervall in zwei Hälften und beginne von vorne.*
7: **if** $|Q - Q_{in}| \leq tol$ **then**
8: $\quad Q_{out} := \frac{16Q - Q_{in}}{15}$
9: **else**
10: $\quad$ Führe den Algorithmus 6 für das Intervall $[a, a+\frac{h}{2}]$, die Funktion f und die Toleranz *tol* aus. Verwende Q_L als Startwert und nenne das Ergebnis wieder Q_L.
11: $\quad$ Führe den Algorithmus 6 für das Intervall $[a+\frac{h}{2}, b]$, die Funktion f und die Toleranz *tol* aus. Verwende Q_R als Startwert und nenne das Ergebnis wieder Q_R.
12: $\quad Q_{out} := Q_L + Q_R$
13: **end if**
14: **return** Q_{out}

Betrachten wir als Beispiel die Kurve $c\colon [0, 2\pi] \to \mathbb{R}^2$, $c(t) := \begin{pmatrix} \sin t \cos t \\ \sin t \end{pmatrix}$. Wir werden diese Kurve in Pixel umwandeln und dann mit unseren Algorithmen eine Approximation der Kurve zurückgewinnen, damit wir sehen was passiert. Ausserdem lassen wir die Krümmung und die Gesamtkrümmung berechnen. Daran sehen wir dann, ob unser Algorithmus funktioniert, da wir die Krümmung auch schon explizit berechnet haben. Zu bemerken wäre an dieser Stelle noch, dass die Pixel hier keine ganzzahligen Koordinaten haben, was wir aber in den Algorithmen auch nicht verwendet haben.

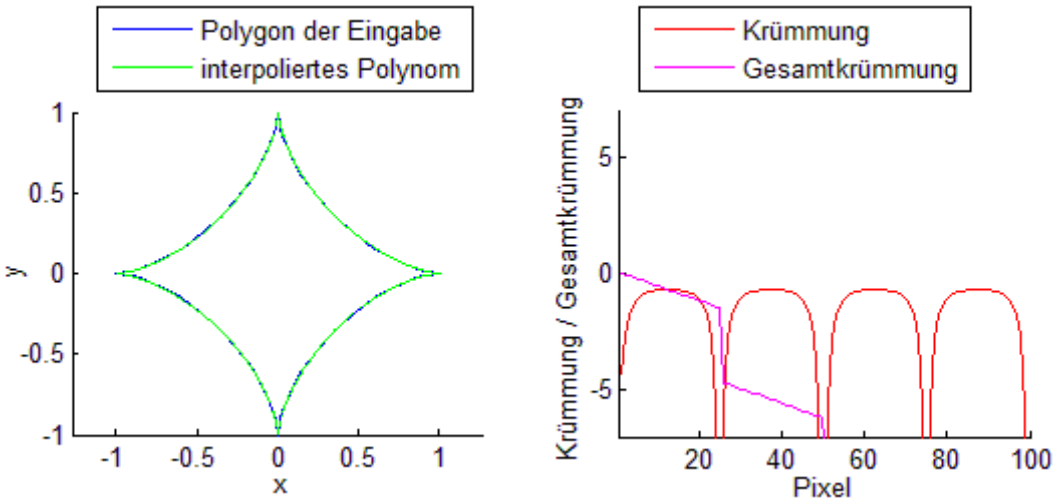


Als Pixel wurden die Punkte $x_i = \sin(\frac{i\pi}{50})\cos(\frac{i\pi}{50})$, $y_i = \sin(\frac{i\pi}{50})$ für $i = 1, ..., 101$ verwendet und der Grad der Polynome ist 5.

Man vergleiche nun mit Beispiel 1.8.4, um sich davon zu überzeugen, dass die Krümmung mit den analytisch berechneten Kurve in etwa übereinstimmt. Wir sehen auch, dass die Gesamtkrümmung über die ganze geschlossene Kurve gerade 0 ist, was wir schon in den Beispielen 3.3.2 und 3.10.1 gezeigt haben.

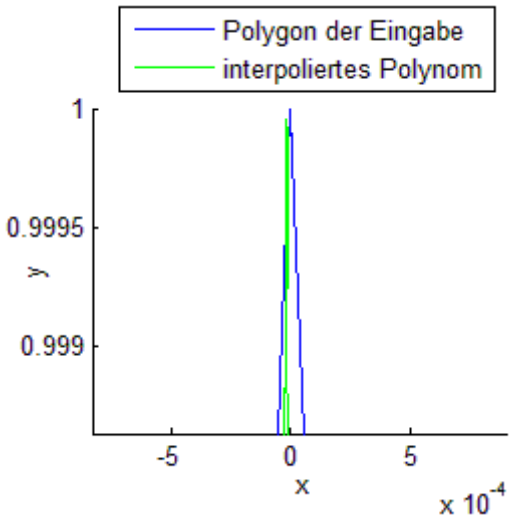
5.6 Eckpunkte

Wir haben gesehen, dass unsere Algorithmen für Kurven mit relativ schwacher Krümmung gut funktioniert. Jetzt wollen wir sehen was passiert, wenn wir eine Kurve mit sehr starker Krümmung oder gar Eckpunkten haben. Betrachten wir zunächst die Astroide:



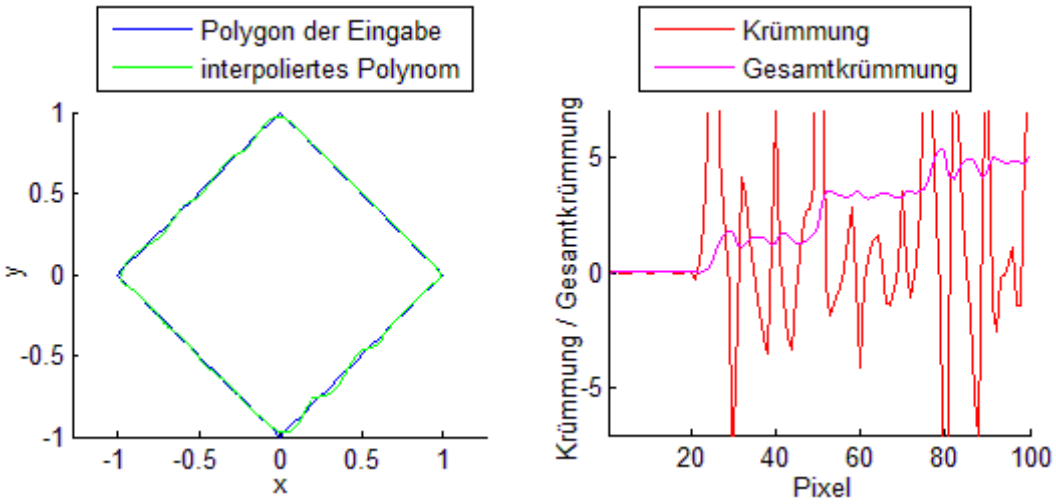
Als Pixel wurden die Punkte $x_i = \cos(\frac{i\pi}{50})^3$, $y_i = \sin(\frac{i\pi}{50})^3$ für $i = 1, ..., 101$ verwendet und der Grad der Polynome ist wieder 5.

Wir sehen direkt, dass die Gesamtkrümmung in der zweiten Ecke aus dem Bild läuft. Lassen wir uns die Gesamtkrümmung für die gesamte Kurve ausgeben, erhalten wir ungefär -6π. Wir hatten in Beispiel 3.10.2 allerdings eine Umlaufzahl von $+1$ berechnet. Zoomt man in einer Ecke näher ran, kann man auch erahnen, warum das passiert:



Die Approximation überschneidet sich in der Ecke, sodass aus einer Linkskurve eine Rechtskurve wird. Die Überschneidung ist leider so gering, dass man sie nicht deutlich sehen kann, aber man sieht auch an der Krümmungskurve, dass der Eckpunkt fälschlicherweise negativ gezählt wird.

Betrachten wir noch ein weiteres Beispiel, wo unsere Algorithmen versagen:



Als Pixel wurden die Punkte $x_i = |i/25 - 2| - 1$, $y_i = 1 - 2\,(1 - ||i/50 - 1/2| - 1|)$ für $i = 1, ..., 101$ verwendet und der Grad der Polynome ist wieder 5.

Das Polynom beginnt nach der ersten Ecke um das Quadrat zu oszillieren. Dadurch oszilliert auch die Krümmung um 0, dem exakten Wert zwischem den Ecken. Das ist natürlich nicht was wir wollen. Daher betrachten wir im nächsten Abschnitt eine neue Methode, mit welcher wir diese beiden Probleme nicht haben.

5.7 Bézierkurven

Bézierkurven sind spezielle Polynome, die zwischen 1960 und 1970 zum ersten mal von Pierre Bézier und Paul de Casteljau unabhängig voneinander verwendet wurden, um Kurven zu approximieren.

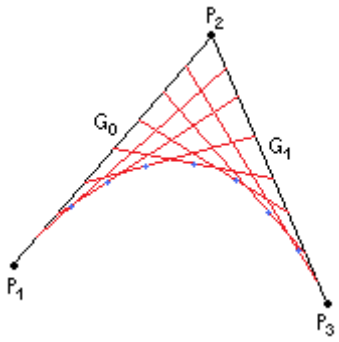
Wir werden quadratische Bézierkurven konstruieren und dann später kubische Bézierkurven implementieren. Der Grund dafür ist, dass die Konstruktion mit steigendem Grad immer mühsamer wird, man aber sehr schnell sehen kann was passiert.

Seien zwei Punkte P_1 und P_2 gegeben. Eine lineare Bézierkurve verbindet diese beiden Punkte einfach nur:

$$B^1(t) = P_1\,(1 - t) + P_2\,t.$$

Lineare Bézierkurven sind für uns also nicht weiter interessant.

Betrachten wir also quadratische Bézierkurven. Dafür seien drei Punkte P_1, P_2 und P_3 gegeben. Wir sehen uns erstmal ein Beispiel an, um zu sehen was passiert und rechnen dann die Kurve explizit aus.



Wir haben die beiden Strecken $G_0(t) = P_1\,(1-t) + P_2\,t$ und $G_1(t) = P_2\,(1-t) + P_3\,t$. Jetzt verbinden wir die beiden Punkte $G_1(t)$ und $G_2(t)$ für alle $t \in [0,1]$ wieder zu einer Strecke G_t. Die Funktion $B^2(t) := G_t(t)$ heisst dann die quadratische Bézierkurve von P_1, P_2 und P_3.

Die Bézierfunktion $B^2(t)$ kann explizit angegeben werden:

$$\begin{aligned} B^2(t) &= G_t(t) \\ &= G_0(t)\,(1-t) + G_1(t)\,t \\ &= (P_1\,(1-t) + P_2\,t)\,(1-t) + (P_2\,(1-t) + P_3\,t)\,t \\ &= P_1\,(1-t)^2 + 2\,P_2\,t(1-t) + P_3\,t^2. \end{aligned}$$

Für Bézierkurven höheren Grades wiederholen wir diese Konstruktion einfach. Für kubische Bézierkurven mit vier Punkten P_1, P_2, P_3 und P_4 berechnen wir beispielsweise die beiden quadratischen Bézierkurven $B_1^2(t)$ bzw. $B_2^2(t)$ für P_1, P_2 und P_3 bzw. P_2,P_3 und P_4. Danach verbinden wir wieder $B_1^2(t)$ und $B_2^2(t)$ für alle $t \in [0,1]$ mit einer Strecke G_t. Die kubische Bézierkurve ist dann definiert durch $B^3(t) := G_t(t)$. Es gilt

$$B^3(t) = P_1\,(1-t)^3 + 3\,P_2\,t(1-t)^2 + 3\,P_3\,t^2(1-t) + P_4\,t^3.$$

Den Beweis dieser Aussage sei dem Leser überlassen. Eine schöne Animation zur Konstruktion findet man unter [BE].

In der Praxis werden hauptsächlich kubische Bézierkurven verwendet und auch wir werden einen Spline implementieren, der auf kubischen Bézierkurven basiert. Bézierkurven haben einige wichtige Vorteile gegenüber der Polynominterpolation:

i) Die Bézierkurve liegt immer innerhalb der konvexen Hülle der vorgegebenen Punkten. Dies folgt direkt aus der Konstruktion der Kurven. Grosse Oszillationen werden dadurch vermieden.

ii) Liegen die Punkte alle auf einer Gerade, so ist auch die Bézierkurve eine Gerade. Eine sehr nützliche Eigenschaft, die ebenfalls direkt aus der Konstruktion folgt.

iii) Die Kurven haben eine einfache Form. Es muss kein Gleichungssystem gelöst werden.

Obwohl uns Eigenschaft iii) erlauben würde, Bézierkurven beliebig hoher Ordnung zu berechnen, wollen wir lieber wieder auf Splines zurückgreifen. Der Grund dafür ist einerseits, dass auch Bézierkurven von allen Punkten abhängen und wir so wieder bei jedem zusätzlichen Punkt die Kurve ändern. Andererseits erfüllen die Bézierkurven keine Minimierungseigenschaften, sodass wir nicht kontrollieren können, was für sehr grosse Punktmengen passiert.

Bei den Bézierkurven haben wir leider keine Freiheitsgrade, die wir ausnützen könnten, sodass es nicht ganz klar ist, wie wir die einzelnen Kurven zusammensetzen sollen. Natürlich wollen wir aber wieder C^2 haben. Dafür gibt es einen Trick, den man für Splines oft verwendet. Man betrachtet die eingegebenen Punkte als Variabeln und berechnet die ersten zwei Ableitungen. Dadurch erhält man Bedingungen an die Punkte, die für C^2 erfüllt werden müssen. Diese sogenannten Stützpunkte kann man dann so wählen, dass die C^2-Bedingung erfüllt ist und die oben genannten Eigenschaften nicht verloren gehen. Eine explizite Herleitung dazu findet man unter [BE2]. Betrachten wir zunächst den Algorithmus für den geschlossenen Bézier-Spline.

Algorithm 7 geschlossener Bézier-Spline

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$

Output:

- je ein geschlossener kubischer Béziersplline für x und y

1: *// Definition aller Stützpunkte //*
2: $e_2 := \frac{2}{3}\, x_1 + \frac{1}{3}\, x_2$, $e_3 := \frac{1}{3}\, x_1 + \frac{2}{3}\, x_2$
3: $f_2 := \frac{2}{3}\, y_1 + \frac{1}{3}\, y_2$, $e_3 := \frac{1}{3}\, y_1 + \frac{2}{3}\, y_2$
4: $e_{3(n-1)+2} := \frac{2}{3}\, x_n + \frac{1}{3}\, x_1$, $e_{3(n-1)+3} := \frac{1}{3}\, x_n + \frac{2}{3}\, x_1$
5: $f_{3(n-1)+2} := \frac{2}{3}\, y_n + \frac{1}{3}\, y_1$, $f_{3(n-1)+3} := \frac{1}{3}\, y_n + \frac{2}{3}\, y_1$
6: $e_1 := \frac{e_2 + e_{3(n-1)+3}}{2}$
7: $f_1 := \frac{f_2 + f_{3(n-1)+3}}{2}$
8: $e_{3(n-1)+4} = e_1$
9: $f_{3(n-1)+4} = f_1$
10: **for all** $i = 1$ **to** $n - 2$ **do**
11: $\quad e_{3i+2} := \frac{2}{3}\, x_{i+1} + \frac{1}{3}\, x_{i+2}$, $e_{3i+3} := \frac{1}{3}\, x_{i+1} + \frac{2}{3}\, x_{i+2}$
12: $\quad f_{3i+2} := \frac{2}{3}\, y_{i+1} + \frac{1}{3}\, y_{i+2}$, $f_{3i+3} := \frac{1}{3}\, y_{i+1} + \frac{2}{3}\, y_{i+2}$
13: $\quad e_{3i+1} := \frac{e_{3i+2} + e_{3(i-1)+3}}{2}$
14: $\quad f_{3i+1} := \frac{f_{3i+2} + f_{3(i-1)+3}}{2}$
15: **end for**
16: $e_{3(n-1)+1} := \frac{e_{3(n-1)+2} + e_{3(n-2)+3}}{2}$
17: $f_{3(n-1)+1} := \frac{f_{3(n-1)+2} + f_{3(n-2)+3}}{2}$
18: *// Berechnung der Bézier-Splines für die Stützpunkte //*
19: **for all** $i = 0$ **to** $n - 1$ **do**
20: $\quad B_i(t) := e_{3i+1}\,(1-t)^3 + e_{3i+2}\, t(1-t)^2 + e_{3i+3}\, t^2(1-t) + e_{3i+4}\, t^3$
21: $\quad C_i(t) := f_{3i+1}\,(1-t)^3 + f_{3i+2}\, t(1-t)^2 + f_{3i+3}\, t^2(1-t) + f_{3i+4}\, t^3$
22: **end for**
23: **return** $(B_0, ..., B_{n-1})$ **and** $(C_0, ..., C_{n-1})$

Im ungeschlossenen Fall haben wie an den beiden Enden ungewollte Freiheitsgrade, die wir irgendwie ausfüllen müssen. Am einfachsten ist es, wenn man einfach je einen virtuellen Punkt einfügt. Im ungeschlossenen Fall haben wir also folgenden Algorithmus:

Algorithm 8 ungeschlossener Bézier-Spline

Input:

- Pixelkoordinaten $x := (x_1, ..., x_n)$ und $y := (y_1, ..., y_n)$

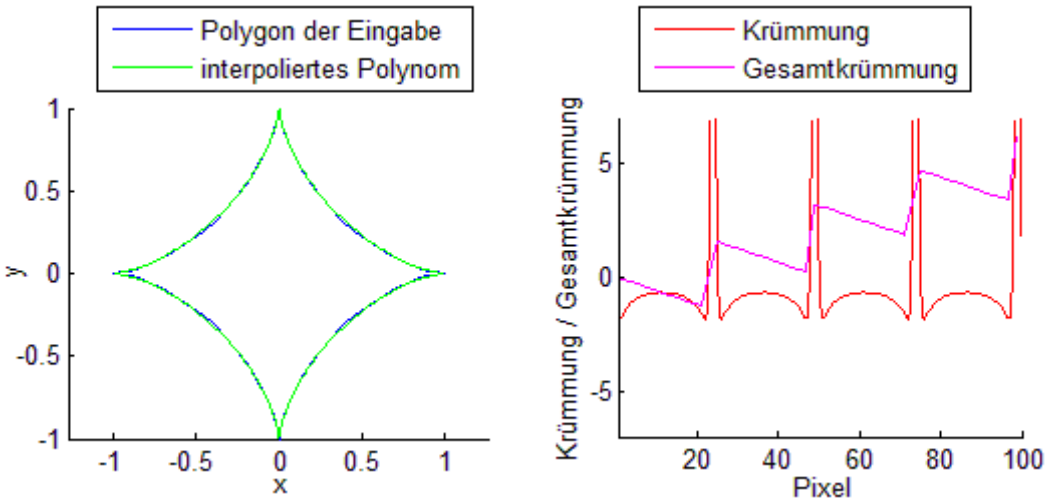
Output:

- je ein ungeschlossener kubischer Béziersplines für x und y

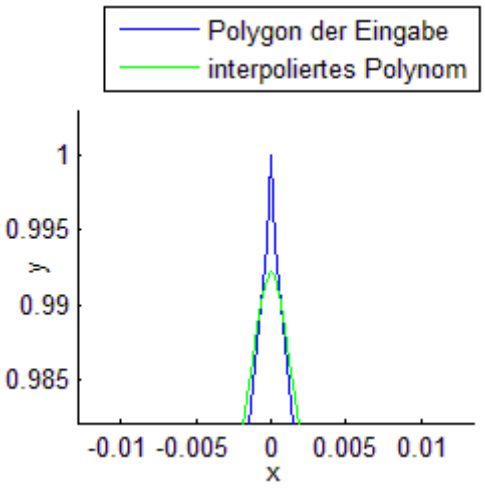
1: *// Einfügen zweier virtueller Punkte //*
2: $x_0 := 2x_1 - x_2$, $x_{n+1} := 2x_n - 2x_{n-1}$
3: $y_0 := 2y_1 - y_2$, $y_{n+1} := 2y_n - 2y_{n-1}$
4: *// Definition aller Stützpunkte //*
5: $e_0 := \frac{1}{3}\, x_0 + \frac{2}{3}\, x_1$
6: $f_0 := \frac{1}{3}\, y_0 + \frac{2}{3}\, y_1$
7: **for all** $i = 0$ **to** $n - 1$ **do**
8: $\quad e_{3i+2} := \frac{2}{3}\, x_{i+1} + \frac{1}{3}\, x_{i+2}$, $e_{3i+3} := \frac{1}{3}\, x_{i+1} + \frac{2}{3}\, x_{i+2}$
9: $\quad f_{3i+2} := \frac{2}{3}\, y_{i+1} + \frac{1}{3}\, y_{i+2}$, $f_{3i+3} := \frac{1}{3}\, y_{i+1} + \frac{2}{3}\, y_{i+2}$
10: $\quad e_{3i+1} := \frac{e_{3i+2} + e_{3(i-1)+3}}{2}$
11: $\quad f_{3i+1} := \frac{f_{3i+2} + f_{3(i-1)+3}}{2}$
12: **end for**
13: $e_{3n+1} := \frac{e_{3n+2} + e_{3(n-1)+3}}{2}$
14: $f_{3n+1} := \frac{f_{3n+2} + f_{3(n-1)+3}}{2}$
15: *// Berechnung der Bézier-Splines für die Stützpunkte //*
16: **for all** $i = 0$ **to** $n - 1$ **do**
17: $\quad B_i(t) := e_{3i+1}\,(1-t)^3 + e_{3i+2}\, t(1-t)^2 + e_{3i+3}\, t^2(1-t) + e_{3i+4}\, t^3$
18: $\quad C_i(t) := f_{3i+1}\,(1-t)^3 + f_{3i+2}\, t(1-t)^2 + f_{3i+3}\, t^2(1-t) + f_{3i+4}\, t^3$
19: **end for**
20: **return** $(B_0, ..., B_{n-1})$ **and** $(C_0, ..., C_{n-1})$

Bleibt zu erwähnen, dass man bei der Implementierung etwas vorsichtiger sein muss, als es hier vielleicht den Eindruck macht. Einerseits muss man bei der Approximation des ersten Stücks im geschlossenen Fall die Stützpunkte des letzten Stücks schon kennen, was natürlich nicht geht. Daher sollte man das erste Stück erst nach dem letzten Stück anfügen. Andererseits muss der Benutzer vor dem Zeichnen der Kurve angeben, ob er eine geschlossene Kurve zeichnen wird oder nicht, damit wir entscheiden können, welchen Algorithmus wir benötigen.

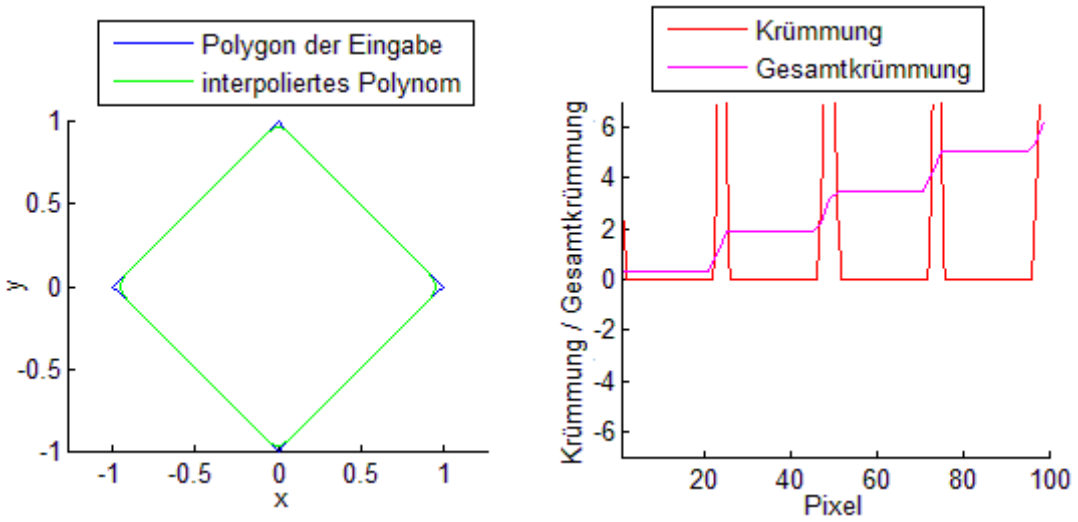
Die Algorithmen zur Berechnung der Krümmung und des Krümmungsintegrals können wir auch hier anwenden und uns so direkt mit den Problemfällen der Polynominterpolation beschäftigen. Betrachten wir also nochmal die Astroide, diesmal mit Algorithmus 7:



In der ersten Grafik hat sich im Vergleich zur Polynominterpolation auf den ersten Blick nichts geändert. In der zweiten Grafik sehen wir aber eine deutliche Veränderung. Die Krümmung läuft jetzt in positiver statt in negativer Richtung aus dem Bild und die Gesamtkrümmung läuft gegen 2π. Das ist genau das was wir wollten. Der Grund für dieses verbesserte Verhalten liegt tatsächlich in der ersten Grafik, wenn man sie genau betrachtet:



Die Approximation überschneidet sich nicht mehr, sondern rundet die Ecke schön ab. Daher bekommen wir in diesem Punkt eine stark positive Krümmung im Gegensatz zur Polynominterpolation, die uns eine stark negative Krümmung berechnet hat. Einen weiteren Vorteil des Bézier-Splines sehen wir in der nächsten Grafik, die wieder ein Quadrat zeigt.



Die Seiten den Quadrats werden exakt approximiert, da per Konstruktion beim Bézier-Spline vier Punkte, die auf einer Gerade liegen, durch eine Gerade approximiert werden. Dadurch zeigt auch die Krümmung das gewünschte Verhalten.

Mit Bézier-Splines sind wir also auch für extreme Eingaben gerüstet, während wir bei normalen Eingaben wie zum Beispiel Kreisen gegenüber der Polynominterpolation nichts verlieren. Daher sind Bézier-Splines auf jeden Fall vorzuziehen.

6 MATLAB-Codes

Die MATLAB-Codes, die während der Erstellung dieser Arbeit geschrieben und unter anderem dazu verwendet wurden, die Grafiken zu erstellen, können unter `http://www.math.ethz.ch/~lemuren/bsc/haggerr/` heruntergeladen werden. Sie sind unkommentiert und an manchen Stellen auch nicht optimal codiert. Dem Leser steht es frei diese Codes mit MATLAB zu testen und gegebenenfalls zu verwenden.

7 Kontakte

Diese Bachelor-Arbeit wurde vom Projekt LEMUREN unterstützt. Sie bildet die Grundlage zur Entwicklung neuer E-Learning Ressourcen im Rahmen des Projekts LEMUREN.
Weitere Informationen: `http://www.lemuren.math.ethz.ch`
Kontakt: lemuren@math.ethz.ch

Das Applet wurde von Heinz Rasched in JAVA implementiert und kann unter `http://www.math.ethz.ch/~lemuren/bsc/haggerr/` heruntergeladen werden.

Bemerkungen hierzu an: heinz.rasched@math.ethz.ch
Allgemeine Fragen oder Bemerkungen an: rhagger@student.ethz.ch

8 Literaturverzeichnis

[BE] http://de.wikipedia.org/wiki/Bézierkurve

[BE2] http://www.cfd.tu-berlin.de/Lehre/tfd_skript/node81.html,
Der kubische Bézier-Spline zur Approximation von Kurven

[DG1] http://www.math.ethz.ch/~lang/DG_d_11Jan08.pdf,
Prof. Urs Lang, *Skript zur Differentialgeometrie I Herbst 2007* (Abschnitt 1),
ETH Zürich

[HO] http://www.mathematik.com/Hopf/index.html,
Proof of the Hopf Umlaufsatz by deformation

[WI] http://www.imada.sdu.dk/Courses/MM08/umlaufsatz.pdf,
Andrew Swann, *Supplementary Notes for MM08 Geometry I* (Abschnitt 6.1)