Hochschule für Angewandte Wissenschaften Hamburg
*Hamburg University of Applied Sciences*

# Bachelorarbeit

**Manuel Bösch**

## Schneller, robuster Algorithmus zur Ellipsenlokalisierung in Grauwertbildern

*Fakultät Technik und Informatik*
*Studiendepartment Informatik*

*Faculty of Engineering and Computer Science*
*Department of Computer Science*

Manuel Bösch

# Schneller, robuster Algorithmus zur Ellipsenlokalisierung in Grauwertbildern

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Meisel
Zweitgutachter: Prof. Dr. Fohl

Eingereicht am: 15.07.2014

**Manuel Bösch**

**Thema der Arbeit**
Schneller, robuster Algorithmus zur Ellipsenlokalisierung in Grauwertbildern

**Stichworte**
Merkmalsextraktion, Ellipsendetektion, Ellipsenlokalisierung, RANSAC

**Kurzzusammenfassung**
Diese Arbeit befasst sich mit der Entwicklung eines Algorithmus zur Lokalisierung von Ellipsen in Grauwertbildern.
Der erarbeitete Algorithmus umfasst eine kantenbasierte Vorverarbeitung, um potentiell zu Ellipsen führende Kantenverläufe zu ermitteln. Aus den verbleibenden Verläufen werden mit Hilfe einer fehlerrobusten Methode, die sich an dem RANSAC-Prinzip orientiert, mögliche Ausgleichsellipsen berechnet.
Als Basis diente ein bestehendes Verfahren, das an mehreren Stellen optimiert wurde.
**Manuel Bösch**

**Title of the paper**
Fast and robust algorithm for ellipse detection in gray scale images

**Keywords**
feature extraction, ellipse fitting, ellipse detection, RANSAC

**Abstract**
This paper deals with ellipse detection in grey scale images.
The developed algorithm contains an edge based preprocessing to extract potentially elliptic edges segments. With the aid of an RANSAC based procedure ellipses get fitted to all remaining segments.
An already existing algorithm provided a basis for an optimized solution. …

# Inhaltsverzeichnis

# Tabellenverzeichnis

# Abbildungsverzeichnis

# 1. Einleitung

Digitale Bildverarbeitung ist ein bedeutsames Themengebiet der modernen Informatik und findet somit Anwendung in den verschiedensten Bereichen.
Autonome Systeme wie Roboter oder Fahrzeuge benötigen bildverarbeitende Algorithmen zur Verarbeitung von Informationen über ihre unmittelbare Umgebung, um zum Abgleich ihrer Position oder zum Vermeiden von Hindernissen (vgl. Tön05, S.16).
Auch die Sicherheitsbranche befasst sich mit dieser Thematik, um bildgestützte Zugangskontrollen zu realisieren, die Personen anhand der Eigenschaften der menschlichen Iris zu identifizieren (vgl. SK12, S.1).
Ein drittes Anwendungsgebiet stellt die "visuelle Inspektion"(Tön05, S.16) von Objekten als Teil eines industriellen Fertigungsprozesses dar. Werkstücke werden auf festgelegte Kriterien untersucht, wie zum Beispiel Durchmesser von Bohrungen, und gegebenenfalls aussortiert, wenn sie der gewünschten Qualität nicht entsprechen. In dieses Anwendungsgebiet fällt auch die Detektion von Ellipsen, mit der sich diese Arbeit beschäftigt.

## 1.1. Problembeschreibung

Kreise und Ellipsen sind neben rechteckigen Formen bedeutsame Charakteristika, wenn es um die Auswertung abgebildeter Objekte geht. Dabei ist der Kreis eine Sonderform der Ellipse, weshalb im weiteren Verlauf der Arbeit nur noch von Ellipsen gesprochen wird. Diese liegen immer dann vor, wenn kugel- und kreisförmige Objekte in zweidimensionalen Bildern abgebildet werden. Die Berechnung und Auswertung von Ellipsen ist beispielsweise für die zuvor erwähnte Überprüfung von Bohrungen, aber auch für die automatisierte Erkennung von Straßenschildern von Bedeutung.

Um die Auswertung vorzunehmen müssen zwei Probleme gelöst werden. Hauptaugenmerk liegt im ersten Schritt auf den Kanten der abgebildeten Objekte, wobei nur jene Kanten von Interesse sind, die eine elliptische Struktur aufweisen. Zunächst gilt es also, die tatsächlich zu elliptischen Kanten gehörenden Pixel zu ermitteln. Erschwert wird dies dadurch, dass Ob-

jektkanten selten vollständig und geschlossen aus den Bilddaten ermittelt werden können. Ursache dafür sind unter anderem komplexe Hintergründe (Abb. 1.1) und Rauschen in Bildern.

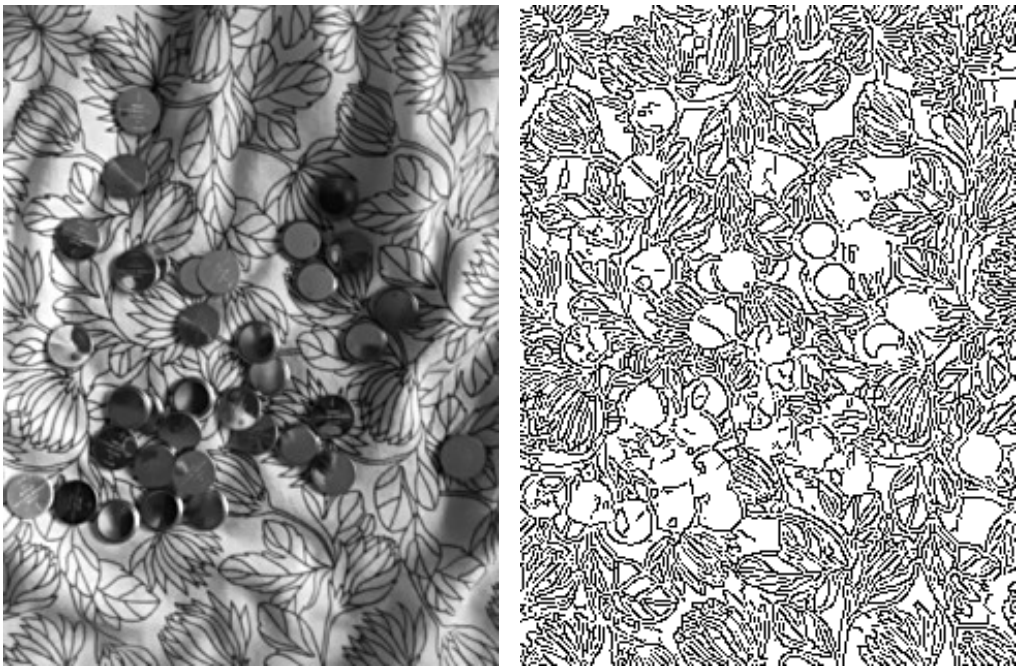


Abbildung 1.1.: Auswirkung komplexer Hintergründe

Es kommt jedoch auch vor, dass abgebildeten Objekte sich teilweise verdecken, wodurch ebenfalls Informationen im Bild verloren gehen. Gleiches resultiert aus Schattenwürfen und Reflexionen, die besonderen Einfluss auf die Zuordnung der Kanten zu abgebildeten Dinge hat. Somit liegen die Kanten eines Objekts in den meisten Fällen in mehreren unzusammenhängenden Teilstücken vor. Bevor also die Berechnung mit zufriedenstellenden Ergebnissen durchgeführt werden kann, sollten die Pixel unterschiedlicher Ellipsen ermittelt und gruppiert werden.

Die zweite Herausforderung ist die eigentliche Parameterberechnung. Die vorliegenden Daten müssen durch Ellipsen approximiert werden. Dabei gilt es zu festzulegen, mit welcher Abweichungstoleranz ein berechnetes Modell als ausreichend korrekt akzeptiert wird. Hinzu kommt, dass der hohe Parameterraum und der daraus resultierende Aufwand für die Berechnung berücksichtigt werden muss.
Die Vorverarbeitung im ersten Schritt entscheidet somit maßgeblich, ob und wie korrekt die Berechnung stattfindet. Grobe Fehler müssen daher größtenteils eliminiert werden, damit keine

Beeinträchtigung der Ergebnisse stattfindet. Da sich das Vorhandensein von abweichenden Werten, auch groben, nicht komplett vermeiden lässt, benötigt die Berechnung ein Verfahren, das selbst unter diesem Umstand korrekte Ellipsen aus den Daten berechnet.

## 1.2. Ziel der Arbeit

Diese Arbeit setzt es sich zum Ziel einen Algorithmus zu entwickeln, der schnell und robust Ellipsen in Grauwertbildern lokalisiert und berechnet. Dazu wird eine Lösung für die beiden in Kapitel 1.1 erwähnten Hauptproblematiken vorgestellt.
Vorliegende Informationen in den Bildern sollen durch die Vorverarbeitung soweit reduziert und gruppiert werden, dass nur so viele Pixel in die Berechnung der Ellipsen einfließen, wie gerade nötig sind, um korrekte, der Abbildung entsprechende Ergebnisse zu erhalten. Teilstücke, die ein und die selbe Ellipse repräsentieren, sollen anhand ihrer Eigenschaften als solche erkannt und zusammengefasst werden, um zum Beispiel die mehrfache Berechnung gleicher Modelle anhand einzelner Teilstücke zu verhindern. Für die Parameterberechnung sollen dadurch effektiv verwertbare Hypothesen zu potentiellen Ellipsen vorliegen.
Für die Hypothesenbildung wird eine bereits entwickelte echtzeittaugliche kantenbasierten Vorgehensweise von Thanh Minh Nguyen, Siddhant Ahuja und Q. M. Jonathan Wu (vgl. NAW09) als Ausgangspunkt verwendet und optimiert. Insbesondere die im original Verfahren verwendete Berechnung der Ellipsen, die auf potentielle elliptische Kantenstücke direkt die Methode der minimalen Fehlerquadrate anwendet (vgl. NAW09, S.3283), wird durch eine den erwähnten Ansprüchen der Robustheit gegenüber fehlerhafter Werten entsprechenden Ansatz ausgetauscht. Die verwendete Methode orientiert sich von M.A. Fischler, und R.C. Bolles vorgestellten Algorithmus RANSAC zur fehlertoleranten Berechnung mathematischer Modelle aus experimentellen Daten (vgl. FB81, S.381).
Die Implementierung wird für unterschiedlichste Bilder getestet, um eine Aussage darüber treffen zu können, unter welchen Umständen und mit welchen Einschränkungen der Algorithmus effektiv eingesetzt werden kann.

## 1.3. Struktur der Arbeit

Um ein leichteres Verständnis der für die Auswertung der Bilder angewandten Mathematik in dieser Arbeit zu ermöglichen, liefert Kapitel 2 die nötigen mathematischen Grundlagen von Ellipsen.

Es gibt die unterschiedlichsten Ansätze, die sich mit der Auswertung elliptischer Strukturen in Bildern befassen. Um einen groben Überblick über deren Funktionsweise zu erhalten, stelle ich in Kapitel 3 zwei Klassen von Algorithmen vor. Die meisten Lösungen basieren entweder auf der Hough-Transformation (Abschnitt 3.1) oder lassen sich in die Ausgleichsrechnung (Abschnitt 3.2) einordnen.

In Kapitel 4 erläutere ich detailliert den Ablauf und die Probleme des als Basis gewählten optimiertes Verfahren. Dabei orientiere ich mich an den einzelnen Schritten: Kantenextraktion (Abschnitt 4.1), Kantensegmentierung (Abschnitt 4.2), Zusammenführung von Kurvenstücken Teilen potentieller Ellipsen (Abschnitt 4.3) und die Bestimmung der Parameter der gefundenen Ellipsenteile durch Ausgleichsrechnung (Abschnitt 4.4).

Für die in Kapitel 4 festgestellten Probleme, liefere ich in Kapitel 5 Verbesserungsvorschläge. Diese umfasst eine optimierte Kantensegmentierung in Abschnitt 5.1 und eine robustere RANSAC basierte Berechnung der Ellipsen in Abschnitt 5.2. Die von der Optimierung gelieferten Ergebnisse werden in Abschnitt 5.3 für Beispielbildern dargestellt.

Kapitel 6 enthält eine kurze Erläuterung zu den in der Implementierung verwendeten Bibliotheken (Abschnitt 6.1) sowie den realisierten Klassen und deren Funktionalitäten (Abschnitt 6.2).

Den Abschluss der Arbeit bildet ein Ausblick für die Anwendung des Algorithmus und weitere Verbesserungsmöglichkeiten in Kapitel 7.

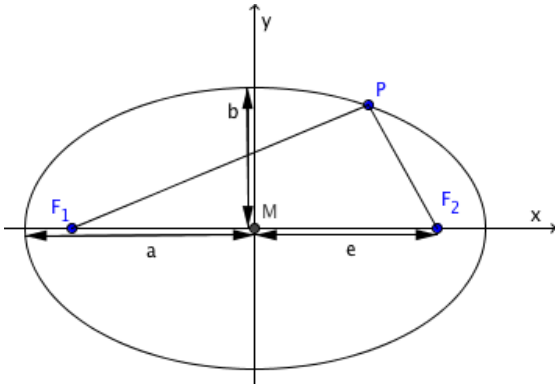
# 2. Mathematische Grundlagen von Ellipsen

Die folgenden Abschnitte liefern die mathematisch relevanten Grundlagen für den Umgang mit Ellipsen. Dabei beschränke ich mich ausschließlich auf für das Verständnis der Arbeit notwendigen Aspekte.

## 2.1. Ellipse als Punktmenge

Eine Ellipse ist durch folgende Definition festgelegt:
*"Die Ellipse ist definitionsgemäß die Menge aller (ebenen)Punkte P, für die die Summer der Entfernung von zwei festen Punkten, den sog. Brennpunkten $F_1$ und $F_2$, konstant ist"* (Pap08, S.219):

$$\overline{F_1P} + \overline{F_2P} = const. = 2a \tag{2.1}$$




$M$: Mittelpunkt (0,0)
$F_1$, $F_2$: Brennpunkte
$a$: große Halbachse
$b$: kleine Halbachse
$e$: Brennweite

mit $a = \sqrt{e + b}$

Abbildung 2.1.: Ellipse mit Mittelpunkt im Ursprung

Die in Abb. 2.1 dargestellte Ellipse lässt sich durch die Mittelpunktgleichung darstellen.

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{2.2}$$

Für die Betrachtung einer Ellipse mit dem Mittelpunkt $M(x_0, y_0)$ gilt die Hauptform der Ellipsengleichung

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1 \tag{2.3}$$

Wie in der Einleitung erwähnt, ist der Kreis eine spezielle Sonderform der Ellipse und lässt sich somit ebenfalls durch die Gleichungen 2.2 und 2.3 darstellen. Für diesen Sonderfall gilt: $a = b$.

Eine weitere Möglichkeit zur mathematischen Beschreibung ist die Parameterdarstellung (Pap09, S.117):

$$\begin{aligned} x(\varphi) &= x_0 + a * \cos\varphi \\ y(\varphi) &= y_0 + b * \sin\varphi \end{aligned} \tag{2.4}$$

Zur mathematischen Beschreibung beliebig verdrehter nicht achsenparalleler Ellipsen, de-

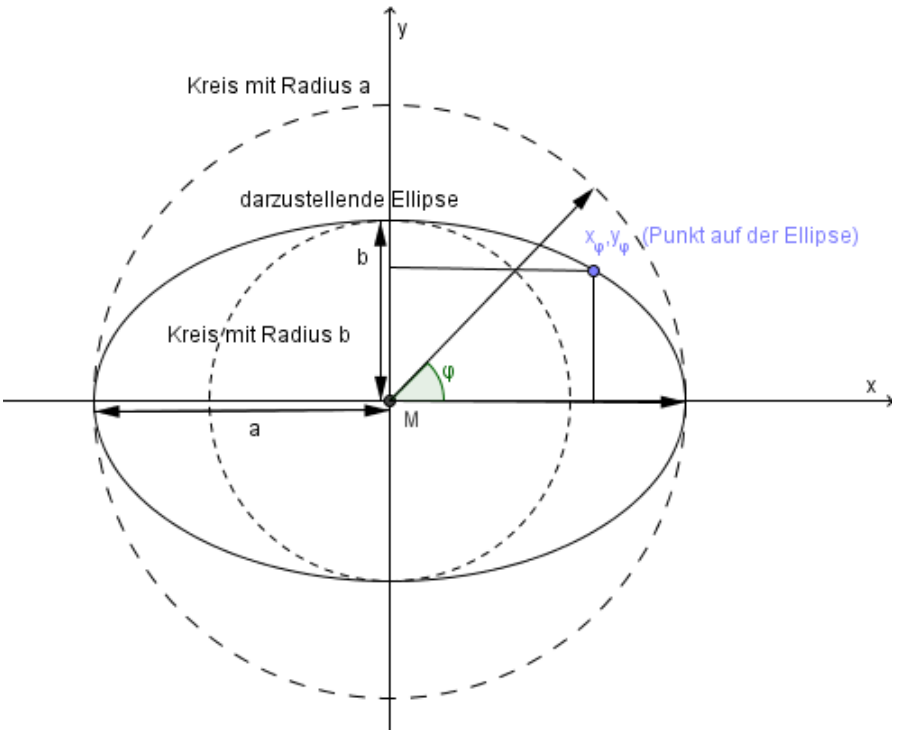



Abbildung 2.2.: Parameterdarstellung einer Ellipse

ren Hauptachse um den Winkel $\alpha$ bezüglich der x-Achse rotiert ist, wird eine allgemeinere Parameterdarstellung gewählt (Wik14):

$$\begin{aligned} x(\varphi) &= x_0 + a\cos\varphi\cos\alpha - b\sin\varphi\sin\alpha \\ y(\varphi) &= y_0 + b\cos\varphi\sin\alpha + b\sin\varphi\cos\alpha \end{aligned} \tag{2.5}$$

## 2.2. Ellipse als Kegelschnitt

Jede Ellipse lässt sich zusätzlich durch die allgemeine Kegelschnittgleichung darstellen

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0 \tag{2.6}$$

Ellipsen stellen neben Hyperbel und Parabeln eine Art der *regulären* Kegelschnitte dar. Welcher Art eine konkrete Gleichung angehört, ergibt sich aus den Koeffizienten $A, B, C, D, E, F$. Eine Ellipse liegt immer dann vor, wenn gilt (FPF99):

$$4AC - B^2 > 0 \tag{2.7}$$

Aus der allgemeinen Kegelschnittgleichung lässt sich die Hauptform der Ellipsengleichung herstellen. Gilt $B \neq 0$, muss der vorliegende Kegelschnitt zunächst in achsenparallele Lage transformiert werden. Dies wird durch eine Drehung des Koordinatensystems um den Winkel $\alpha$ erreicht. $\alpha$ ist der Winkel zwischen der Hauptachse einer Ellipse und der x-Achse. Für neuen Koordinaten $(\dot{x}, \dot{y})$ im gedrehten System gilt (Pap09, S.43) (Gut05):

$$\begin{aligned} x &= \dot{x}\cos\alpha - \dot{y}\sin\alpha \\ y &= \dot{x}\sin\alpha + \dot{y}\cos\alpha \\ \text{mit } 2\alpha &= \arctan\left(\frac{B}{A-C}\right) \end{aligned} \tag{2.8}$$

Durch diese Hauptachsentransformation wird das gemischte Glied $Bxy$ eliminiert und es ergibt sich eine vereinfachte Formel:

$$\begin{aligned} \dot{A}\dot{x}^2 + \dot{C}\dot{y}^2 + \dot{D}\dot{x} + \dot{E}\dot{y} + \dot{F} &= 0 \\ \dot{A} &= A\cos^2\alpha + B\cos\alpha\sin\alpha + C\sin^2\alpha \\ \dot{C} &= A\sin^2\alpha - B\cos\alpha\sin\alpha + C\cos^2\alpha \\ \dot{D} &= D\cos\alpha + E\sin\alpha \\ \dot{E} &= -D\sin\alpha + E\cos\alpha \end{aligned} \tag{2.9}$$

Aus Gleichung 2.9 lässt sich die aus Gl. 2.3 bekannten Parameter der Hauptform der Ellipsengleichung bestimmen:

$$a = \sqrt{\frac{-4\dot{F}\dot{A}\dot{C} + \dot{C}\dot{D}^2 + \dot{A}\dot{E}^2}{4\dot{A}\dot{C}^2}} \tag{2.10}$$

$$b = \sqrt{\frac{-4\dot{F}\dot{A}\dot{C} + \dot{C}\dot{D}^2 + \dot{A}\dot{E}^2}{4\dot{A}^2\dot{C}}} \tag{2.11}$$

$$\dot{x}_0 = \frac{-\dot{D}}{2\dot{A}} \tag{2.12}$$

$$\dot{y}_0 = \frac{-\dot{E}}{2\dot{C}} \tag{2.13}$$

Der ursprüngliche Mittelpunkt $M(x_0, y_0)$ vor der Transformation ergibt sich durch einfache Rücktransformation von $(\dot{x}_0, \dot{y}_0)$(Gl. 2.8)

# 3. Überblick zu bestehenden Verfahren

Es gibt zahlreiche Ansätze die sich mit der Detektion von Ellipsen in Bildern beschäftigen. Neben exotischen Vorgehensweisen, wie beispielsweise einem genetischen Ansatz (YKG05) oder einem sich an kollektiven Tierverhalten orientierendenden Algorithmus (CGZPC14), gibt es im Groben zwei Grundvorgehensweisen.
Ein großer Teil basiert auf der generalisierten Hough Transformation, deren Funktionsprinzip ich in Abschnitt 3.1 kurz erläutern werde. Der zweite häufig verwendete Ansatz stellt die Anwendung von Ausgleichsrechnung auf vorverarbeiteten Daten dar, zu der Abschnitt 3.2 einen groben Überblick enthält.

## 3.1. Hough Transformations basierte Vorgehensweisen

Das Grundprinzip der Hough Transformation ist es aus dem Kantenbild eines Quellbildes zuerst alle möglichen Ausprägungen eines gesuchten Modells zu ermitteln und im Anschluss die mit den meisten Übereinstimmungen als Ergebnis auszuwählen. Für die Detektion von Geraden stellt die sie ein effektives und robustes Standardverfahren dar.
Eine Gerade wird durch die Hessesche Normalform mit dem Winkel $\phi$ und dem euklidischen Abstand $d$ dargestellt. In Abhängigkeit zu den vorliegenden Datenpunkten wird ein Parameterraum $(\phi, d)$ festgelegt, den man sich als zweidimensionale Häufigkeitsmatrix vorstellen kann. Ein Punkt $P$ im Kantenbild kann durch unendlich vielen Geraden dargestellt werden. Für jede möglichen Geraden mit ($phi$,$d$) wird im Parameterraum an der zugehörigen Stelle inkrementiert. Am Ende werden die $(\phi, d)$-Paarungen ausgewählt, die Häufungen aufweisen.

Wendet man diese Prinzip auf Ellipsen an, erhält man einen fünfdimensionalen Parameterraum für den jeder Kantenpunkt unendlich viele Ellipsen bestimmt. Daraus resultiert ein sehr hoher Rechen- und Speicheraufwand um allein den Parameterraum aufzustellen. Zusätzlich muss im Anschluss der gesamte Raum auf Häufungen untersucht werde. Mit zunehmender Bildgröße würde also bei direkter Anwendung der Hough Transformation die Bearbeitungszeit förmlich in ungeahnte Höhen schwinden.
Dennoch ist die Hough Transformation aufgrund ihrer robusten Ergebnisse, die sie selbst unter

hohen Störeinflüssen im Bild liefert, auch für die Ellipsendetektion von Bedeutung. Es gibt zahlreiche Ansätze die eine deutliche Beschleunigung der Berechnung durch beispielsweise Berücksichtigung der Gradienten und Tangenten der Kantenpixel (GZ97) oder Reduzierung des effektiven Parameterraums aufgrund geometrischer Eigenschaften von Ellipsen (CLER07).

## 3.2. Ausgleichsrechnungs basierte Verfahren

Eine zweite Gruppe von Verfahren verwendet Ausgleichsrechnung, um aus vorverarbeiteten Kanteninformationen Ellipsen zu berechnen. Prinzip der Ausgleichsrechnung ist es, für die betrachteten Daten das am besten übereinstimmende Modell zu erhalten. Darin liegt der grundlegende Unterschied zur Hough Transfromation.

Das Hauptproblem der Ausgleichsrechnung ist die Sensibilität auf abweichende Werte oder sonstige Störeinflüsse im Bild, wodurch die ermittelten Ergebnisse mitunter stark verfälscht werden können. Daher geht der Ausgleichsrechnung immer eine Kantenvorverarbeitung voraus. Beispielsweise werden gerade Kantenverläufe und Rauschen aus dem Kantenbild entfernt und für die verbliebenen unterbrechungsfreien Abschnitte jeweils Ausgleichsellipsen berechnet (JRW06). Zwei andere Algorithmen unterteilen die vorliegenden Kantenverläufe in Kurvenstücke, die potentiell eine Ellipse ergeben können und berechnen für die unterschiedlichen Kurven die Ellipsen (NAW09) (MHZS07).

# 4. Überprüfung eines bestehenden kantenbasierenden Verfahren

In diesem Kapitel wird der kantenbasierte Lösungsansatz von T.M. Nguyen, S. Ahuja und Q.M.J. Wu (NAW09) erklärt und auf effektive Verwendbarkeit untersucht. Ich habe mich für diesen Ansatz als Basis für eine Weiterentwicklung entschieden, weil er gute Ergebnisse für die Detektion von mehreren Ellipsen sogar in Echtzeit erwarten ließ und zudem anschaulich und gut verständlich beschrieben wird.

Das Verfahren gliedert sich in vier Bearbeitungsschritte, die gemäß ihrer Ablaufreihenfolge beschrieben werden:

1. **Kantenextraktion**:
   Aus dem Quellbild werden mittels des Canny-Operators die Kanten der abgebildeten Objekte extrahiert.
   (Abschnitt 4.1)
2. **Kantensegmentierung**:
   Aus dem gewonnenen Kantenbild werden zusammenhängende Segmente gebildete. Diese Kantensegmente werden durch Linienstücke approximiert und danach in Kurvenstücke unterteilt.
   (Abschnitt 4.3)
3. **Zusammenführung zu potentiellen Ellipsenstücken**:
   Die Kurvenstücke werden auf potentielle Zusammengehörigkeit zu gleichen Ellipse überprüft und gegebenenfalls zu entsprechenden Ellipsenstücken zusammengefasst.
   (Abschnitt 4.3)
4. **Berechnung der Ausgleichsellipsen**:
   Auf Basis der durch die Zusammenführung gelieferten Ergebnisse werden die jeweils am besten zu den Ellipsenstücken passenden Ellipsen durch Ausgleichsrechnung bestimmt.
   (Abschnitt 4.4)

Die einzelnen Abschnitt weisen jeweils die gleiche Struktur auf: Für jeden Teilschritt werden zunächst Funktionsweise und Ablauf erklärt. Anschließend wird die Anwendbarkeit anhand konkret gewählten Ausgangssituationen und den daraus resultierenden Ergebnissen kritisch bewertet.
Alle als verbesserungswürdig festgestellten Aspekte werden abschließend in Hinblick auf die Optimierung in Kapitel 5 zusammengefasst (Abschnitt 4.5).

## 4.1. Kantenextraktion

Wie die Bezeichnung des Algorithmus als kantenbasiertes Verfahren bereits erkennen lässt, ist es als erstes notwendig, die Kanten aller im Quellbild befindlichen Objekte zu ermitteln. Hierfür wird der Canny Operator von J. Canny (Can86) vorgesehen. Aufgrund der Popularität dieses Kantendetektors werde ich sein grundlegendes Funktionsprinzip nur kurz beschreiben.

Der Canny Algorithmus folgt er drei grundlegenden Kriterien:

1. **Gute Detektion**:
   Gleichermaßen soll die Wahrscheinlichkeit ein Kantenpixel nicht zu erkennen und die, ein nicht zu einer Kante gehörendes Pixel fälschlicherweise als Kantenpixel zu markieren, sehr gering sein.
2. **Gute Positionierung**:
   Die ermittelten Kantenpixel sollen so genau wie möglich der Mitte der tatsächlichen Kante im Bild entsprechen.
3. **minimales Ansprechverhalten**:
   Für eine Kante im Bild soll auch nur eine Kante im Kantenbild entstehen.

### 4.1.1. Bildglättung

Zur Reduzierung von Rauschen im Quellbild, das beispielsweise durch schlechte Lichtverhältnisse bei der Aufnahme entstehen kann, wird das Bild unter Anwendung eines Gauss-Filters

$$G = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.1}$$

geglättet.

### 4.1.2. Gradientenbestimmung

Eine Kante liegt genau dann vor, wenn direkt benachbarte Pixel im Quellbild stark abweichende Grauwerte vorweisen. Betrachtete man den Grauwertverlauf eines Bildes als diskrete Funktion

$$f(x, y) = z \tag{4.2}$$

bei der $x$ und $y$ die Position und $z$ den Grauwert des Pixels beschreibt, weißt $f$ bei einer Kante einen Gradient $\nabla f(x, y)$ mit hohem Wert auf. Um Einflüsse durch Restrauschen zu minimieren wird ein Mittelwert über die Gradienten drei übereinander liegender Bildzeilen gebildet. Diese Vorgehensweise ist bekannt als Anwendung des Sobel Filters und ist beschrieben durch die beiden Faltungsmasken $G_x$ und $G_y$ für die Faltung in x- bzw. y-Richtung.

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.3}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{4.4}$$

Nach der separaten Faltung in x- und y-Richtung ergibt sich für die Stärke $G$ (Betrag) des Gradienten

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{4.5}$$

und für die Richtung $\alpha$

$$\alpha = \arctan(\frac{G_y}{G_x} \tag{4.6}$$

Die Richtung $\alpha$ wird auf einen der vier möglichen Winkel gerundet: $0°$,$45°$,$90°$,$135°$.

### 4.1.3. Non-Maxima Surpression

Die Kanten im Quellbild werden durchlaufen und alle Kantenpixel gemäß der "Non-Maxima Surpessionäussortiert. Pixel, deren Gradientenstärke kein Maximum in ihrer Richtung aufweisen, werden verworfen. Dadurch werden die Kanten auf eine Breite von einem Pixel reduziert.

### 4.1.4. Hystersis Thresholding

In diesem letzten Schritt werden als zu schwach angesehene Kanten verworfen. Dazu werden zwei Grenzwerte für die Stärke der Gradienten festgelegt, die über die Akzeptanz einer Kante entscheiden:
$th_u$ : oberer Grenzwert
$th_l$ : unterer Grenzwert
($th_l < th_u$)
Für eine effektive Aussortierung der Kantenpixel schlägt Canny $th_u : th_l$ ein Verhältnis zwischen $2:1$ und $3:1$ vor (vgl. Can86, S.690).
Es ergeben sich drei mögliche Fälle:

1. $|G| > th_u$:
   Zugehöriges Kantenpixel wird beibehalten
2. $|G| < th_l$:
   Zugehöriges Kantenpixel wird verworfen
3. $th_l < |G| < th_u$:
   Zugehöriges Kantenpixel wird nur beibehalten, wenn es mit einem Kantepixel verbunden ist, dass das den ersten Fall erfüllt

### 4.1.5. Bewertung der Ergebnisse

Da der Canny-Algorithmus ein Standardverfahren zur Kantendetektion ist, habe ich an dieser Stelle keine Korrekturvorschläge. Die Abbildung 4.2 zeigt exemplarisch die aus dem Quellbild (Abb. 4.1) ermittelten Kantenbilder für unterschiedlich gewählte Grenzwerte. Hier lässt sich erkennen, dass niedrigere Grenzwerte wie beschrieben auch deutlich schwächere Kanten noch im Kantenbild verbleiben lässt. Durch Herabsetzen der Grenzwerte erhält man nur die stärkeren Kanten, allerdings nimmt dadurch auch die Verfälschung der Kantenverläufe durch durch die von den Objekten geworfenen Schatten zu.

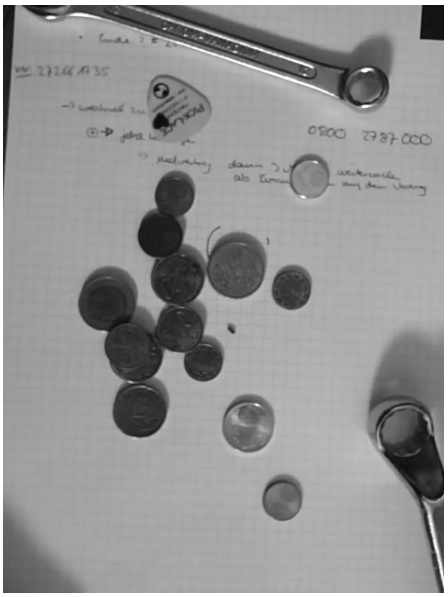
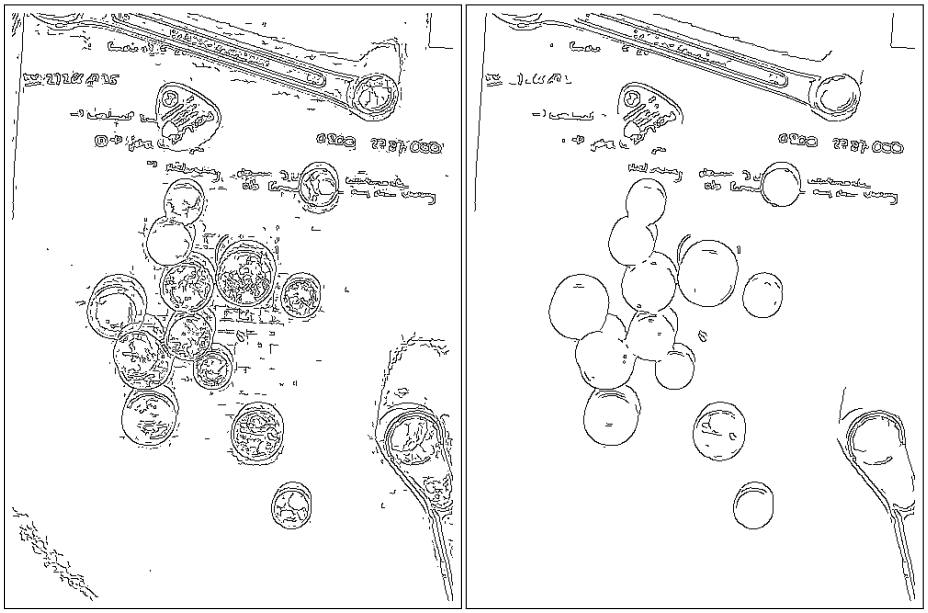


Abbildung 4.1.: Quellbild



(a) $th_u = 30$ und $th_l = 20$ (b) $th_u = 150$ und $th_l = 100$

Abbildung 4.2.: Zu einem Quellbild erzeugte Kantenbilder

## 4.2. Kantensegmentierung

Im Folgenden wird das aus der Kantenextraktion hervorgehende Kantenbild weiterverarbeitet. Zur folgenden Untersuchung der Kanten werden unterbrechungsfrei verbundenen Pixel zu separaten Kantensegmenten zusammengefasst (Abschnitt 4.2.1). Um für die Anzahl der zu berücksichtigen Pixel im weiteren Bearbeitungsverlauf sinnvoll zu reduzieren, werden die einzelnen Segmente durch Linienstücke approximiert (Abschnitt 4.2.2). Dadurch können die Kantenverläufe anschließend effektiver auf elliptische Strukturen untersucht werden und weiter in Kurvenstücke unterteilt werden (Abschnitt 4.2.3).

### 4.2.1. Gruppieren Zusammenhängender Kanten

Für die Bildung von zusammenhängenden Kantensegmenten wird im originalen Artikel, in dem der hier untersuchte Algorithmus vorgestellt wird, keine genauer spezifizierte Vorgehensweise beschrieben. Natürlich gibt es zahlreiche Ansätze, die sich mit dieser Thematik befassen. Ich habe mich für eine leicht vereinfachte Version *edgeLinking*-Algorithmus von Peter Kovesi entschieden (Kov13), da die später folgende Approximation sich an einer von ihm entworfenen Vorgehensweise orientiert. Die durch die Vereinfachung eingesparte Einarbeitungszeit ließ sich somit für die intensivere Untersuchung und Optimierung der Ellipsendetektion aufwenden.
Im Folgenden veranschauliche ich den Ablauf, den ich den online veröffentlichen Matlab Quelldateien entnommen habe (Kov13).

#### Edge Thinning

Zur effektiven Segmentaufsammlung ist vorausgesetzt, dass die Kanten durchgehende eine Breite von genau einem Pixel aufweisen, da sonst unnötige Verkürzungen stattfinden können (Kov13, einführende Quellcodekommentare). Die verwendete Canny-Implementierung von openCV stellt dies nur diagonal betrachtet sicher, jedoch nicht horizontal oder vertikal (siehe Abb. 4.3).
Da sich dieses Problem auf ein leicht zu erkennendes Muster beschränkt(Abb. 4.4), verzichte ich an dieser Stelle auf die Verwendung eines aufwändigen Algorithmus, um eine einheitliche Kantenbreite von einem Pixel sicherzustellen. Unter den Stichwörtern *edge skeletonizing* und *edge thinning* lassen sich viele Verfahren finden, die abgebildete Objekte aller Art (nicht nur Kantenverläufe) auf ein maximal ein Pixel breites Skelett reduzieren (NO94) (HSCP87).
Von jedem Pixel einer Kante wird einfach auf das Muster aus Abb. 4.4 in einer der vier möglichen Variationen zutrifft. Ist dies der Fall, wird das überflüssige Pixel (Abb. 4.4 markiert durch

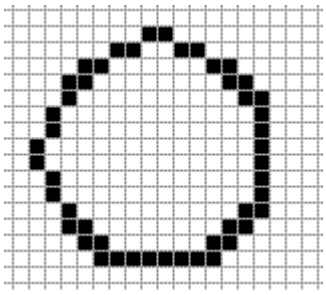


Abbildung 4.3.: Kantenstück mit Breite größer als ein Pixel

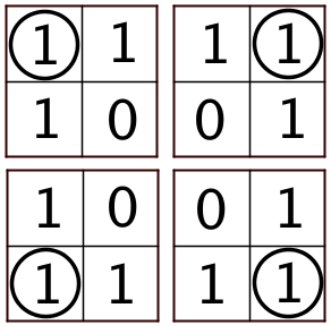



Abbildung 4.4.: Muster zur Erkennung überflüssiger Kantenpixel

Kreis) aus dem Kantenbild entfernt. Abbildung 4.5 zeigt das Ergebnis der Anwendung auf das Kantenstück aus Abb. 4.3.

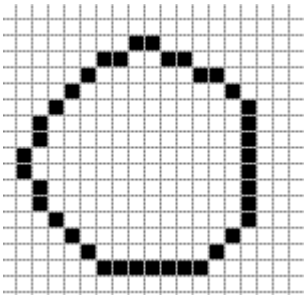
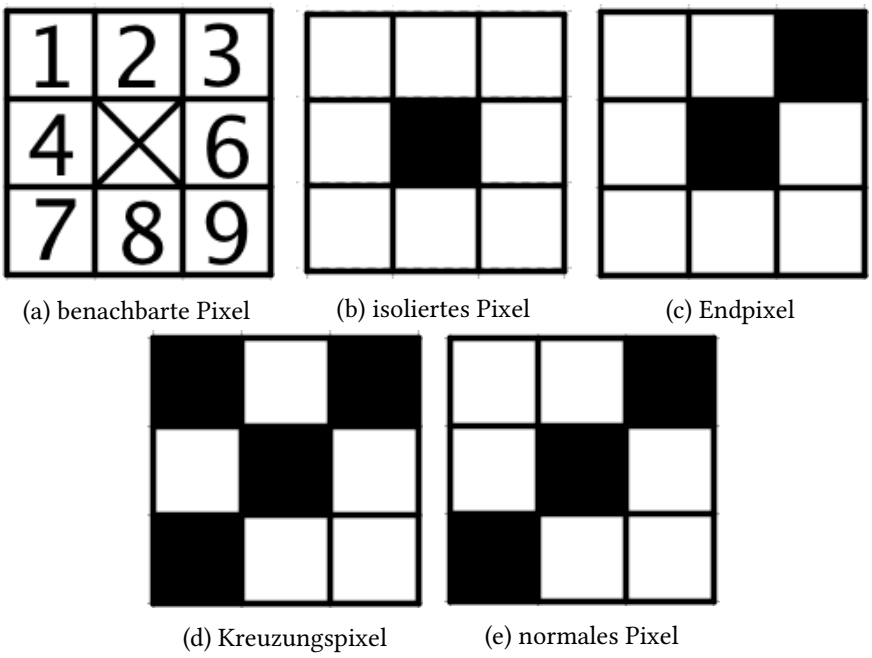


Abbildung 4.5.: Ergebnis des Thinning Algorithmus

### Behandlung unterschiedlicher Kantenpixel

Nach der Kantenbilderzeugung liegen vier Typen von Kantenpixel vor. Der Typ ergibt sich aus der Anzahl der Schwarz-Weiß-Übergänge, also der Anzahl der Wechsel von Kantenpixeln zu nicht gesetzten Pixeln, in der direkten Nachbarschaft des jeweiligen Pixels. Als direkt benach-

bart gelten alle Pixel, die direkt vertikal, horizontal oder diagonal aneinander grenzen. Somit ist die Nachbarschaft als eine 3x3 Matrix aufzufassen, von welcher der zu charakterisierende Punkt die Mitte darstellt (Abb. 4.6a).




(a) benachbarte Pixel (b) isoliertes Pixel (c) Endpixel

(d) Kreuzungspixel (e) normales Pixel

Abbildung 4.6.: Beispiele für die vier Kantenpixeltypen

1. **isolierte Pixel**:
   Das sind all jene Bildpunkte, um die herum keine Übergänge stattfinden. Diese Pixel haben keinen Anschluss an andere Kantenpixel (Abb. 4.6b).
2. **Endpunkte**:
   Der Punkt, der einen zusammenhängenden Kantenverlauf abschließt. Diese lassen sich daran erkennen, dass in ihrer direkten Nachbarschaft exakt zwei Übergänge von gesetzten Punkten (=Kantenpixel) und nicht gesetzten Punkten vorliegen (Abb. 4.6c)
3. **Kreuzungspunkte**:
   Ein Punkt, an dem sich unterschiedliche Kantenverläufe überschneiden, ist ein Kreuzungspunkt. Um diese Pixel liegen mehr als fünf Übergänge vor (Abb. 4.6d).
4. **normaler Kantenpunkt**:
   Dies beschreibt alle Kantenpixel, die Teil einer Kantenstücks sind und auf die keine der vorherigen Beschreibungen zutrifft. Sie weisen somit zwischen drei und fünf Wechsel vor (Abb. 4.6e).

Das gesamte Quellbild wird durchlaufen, um für jedes Pixel den Typ zu bestimmen. Alle isolieren Punkte, werden aus dem Kantenbild gelöscht, weil sie für die weitere Betrachtung der Kanten keine Bedeutung haben. Außerdem werden die Positionen aller Endpunkte gespeichert. Ursprünglich ist auch die Speicherung der Kreuzungspunkte vorgesehen, um auf Basis dieser genauer über die Zugehörigkeit von normalen Kantenpunkten zu den sich kreuzenden Kantenstücken zu entscheiden. Dadurch soll die Wahrscheinlichkeit der Zusammenfassung von Kantenzügen unterschiedlicher in ein Kantensegment verringert werden. Allerdings verzichte ich auf diese Einbeziehung, um an dieser Stelle unnötige Komplexität zu vermeiden. Ich vereinfache die Behandlung der Kreuzungspunkte durch folgenden Schritt: Kreuzungspunkte werden wie isolierte Pixel aus dem Kantenbild entfernt. Zusätzlich werden die angrenzenden Pixel, die bereits zuvor als normale Kantenpunkte aufgefasst wurden, als Endpunkte gespeichert. Kantenverläufe werden also an Kreuzungen unterbrochen, wodurch bei der anschließenden Segmentbildung deutlich effektiver und effizienter vermieden wird, dass Kanten unterschiedlicher Objekte ein Segment bilden.

### Segmentaufsammlung

Um die verschiedenen Kantenstücke aufzusammeln wird von jedem freien Endpunkt aus einem Kantenverlauf solange gefolgt, bis ein weiterer Endpunkt erreicht wird. Ein Segment wird dann durch eine Liste der aufgesammelten Punkte repräsentiert. Die passierten Pixel werden als besucht markiert.
Nachdem alle Endpunkte zugeordnet sind muss das Bild zusätzlich noch nach isolierten Kantenschleifen durchsucht werden. In der Abb. 4.7 sieht man, dass diese speziellen Kantenverläufe keine Endpunkte (farblich hervorgehoben) aufweist, weshalb sie durch den vorangegangenen Schritt nicht aufgesammelt werden können.
Es muss also das gesamte Bild erneut durchlaufen werden. Trifft man auf ein Kantenpixel wird von diesem aus solange der aktuellen Kante gefolgt, bis kein weiterer gesetzter Bildpunkt mehr erreichbar ist. Damit ist die Schleife komplett aufgesammelt.
An dieser Stelle lässt sich abweichend vom originalen Algorithmus in einigen Fällen Bearbeitungszeit einsparen, wenn alle Kantenpixel zu Beginn gezählt werden. Die Suche nach isolierten Kanten ließe sich dann vorzeitig abbrechen, wenn die Anzahl der aufgesammelten Punkte der aller vorhandenen Kantenpunkte entspricht. Besonders bei großen Quellbildern hat sich diese Abbruchbedingung als sinnvoll erwiesen.

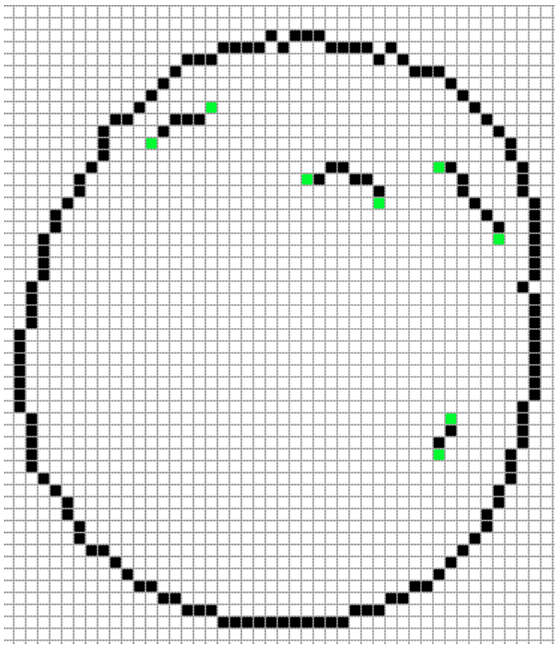


Abbildung 4.7.: Ausschnitt mit gefundenen Endpunkten und einem isolierten Kantenverlauf

### Bewertung der Ergebnisse

Abbildung 4.8 zeigt die ermittelten Segmente aus dem Kantenbild 4.2b des Abschnitts 4.1.5. Die einzelnen Segmente lassen sich durch die unterschiedliche Färbung von den sie direkt umgebenden unterscheiden. Die gefundenen Segmente liefern eine solide Basis für die weitere Untersuchung.

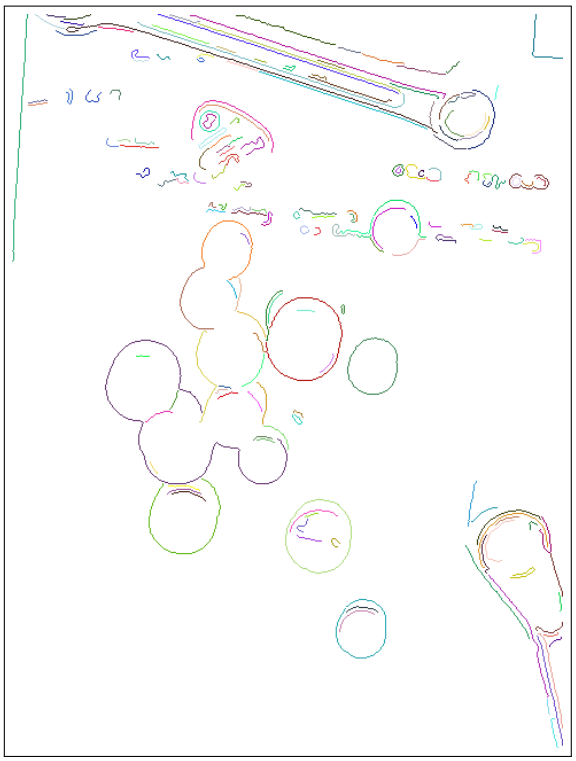


Abbildung 4.8.: Ergebnis der Segmentaufsammlung für das Kantenbild aus Abb. 4.2b

### 4.2.2. Approximation durch Linienstücke

Wie bereits erwähnt, weisen die vorliegenden Segmente deutlich mehr Punkte auf, als für die weitere Bewertung sinnvoll sind. Für eine verlustfreie Reduktion der Punktanzahl wird eine Segmentierung aller Kantenstücke durch gerade Linienstücke angewandt (vgl. NAW09, S.3281).
Nach einer Beschreibung des Ablaufs werden die Ergebnisse der Approximation exemplarisch veranschaulicht.

#### Ablaufbeschreibung

Peter Kovesi liefert die Vorlage in Form veröffentlichter kommentierter Matlab-Quelldateien (Kov06, vlg).
Die Punkte jedes Kantensegments werden durch Linienstücke approximiert, deren jeweilige Endpunkte so gewählt werden, dass alle Punkte des ursprünglichen Segments, die zwischen diesen Endpunkten liegen, maximal um $d_{tol}$ von der Linie abweichen.

Die Gerade $G$, die zwei Punkte $P_1 = (x_1, y_1)$ und $P_2 = (x_2, y_2)$ verbindet, wird durch

$$x * (y_1 - y_2) + y * (x_2 - x_1) + y_2 x_1 - y_1 x_2 = 0 \tag{4.7}$$

berechnet.
Der Abstand eines beliebigen Punktes $P_i = (x_i, y_i)$ zu $G$ ist definiert durch:

$$d_i = \frac{x_i(y_1 - y_2) + y_i(x_2 - x_1) + y_2 x_1 + y_1 x_2}{\sqrt{(x_1 - x_2)^2 + ((y_1 - y_2)^2}} \tag{4.8}$$

Somit gilt für den Punkt $P_m$ mit maximalem Abstand:

$$d_{P_m} = max(d_j, d_k, ...) \tag{4.9}$$

Der Ablauf für ein Segment $S$ verhält sich wie folgt:
Zunächst wird der Punkt $P_m$ mit maximalem Abstand zur Gerade zwischen den beiden Endpunkten $P_0$ und $P_{n-1}$ (n=Anz. Punkte in S) berechnet. Solange $d_{P_m} > d_{tol}$ gilt, wird die maximale Abweichung zur Geraden zwischen $P_0$ und dem zuletzt ermitteltem $P_m = (P_p)$ bestimmt. Sobald $d_{P_m} \leq d_{tol}$ gilt, ist der Bereich zwischen $P_0$ und $P_p$ zufriedenstellend approximiert und alle Punkte zwischen $P_0$ und $P_n$ werden aus dem Segment entfernt. Das gleiche wird nun für den Bereich von $P_p$ bis $P_{n-1}$
Der Ablauf wird solange durchgeführt, bis $S$ komplett approximiert ist. Ist die Anzahl der verbleibenden Punkte in $S$ kleiner als ein festgelegter Grenzwert, wird es für den weiteren Ablauf nicht berücksichtigt.

### Bewertung der Ergebnisse

Abbildung 4.9 veranschaulicht den Ablauf der Approximation für ein kurzes Kantensegment (Abb. 4.9a). Es ist jeweils die für den aktuellen Schritt (Abb. 4.9b-4.9d) betrachtete Gerade $G$ und der von ihr maximal abweichende Punkt $P_m$ hervorgehoben. Für das gewählte Segment verbleiben vier Kantenpixel (Abb. 4.9e).

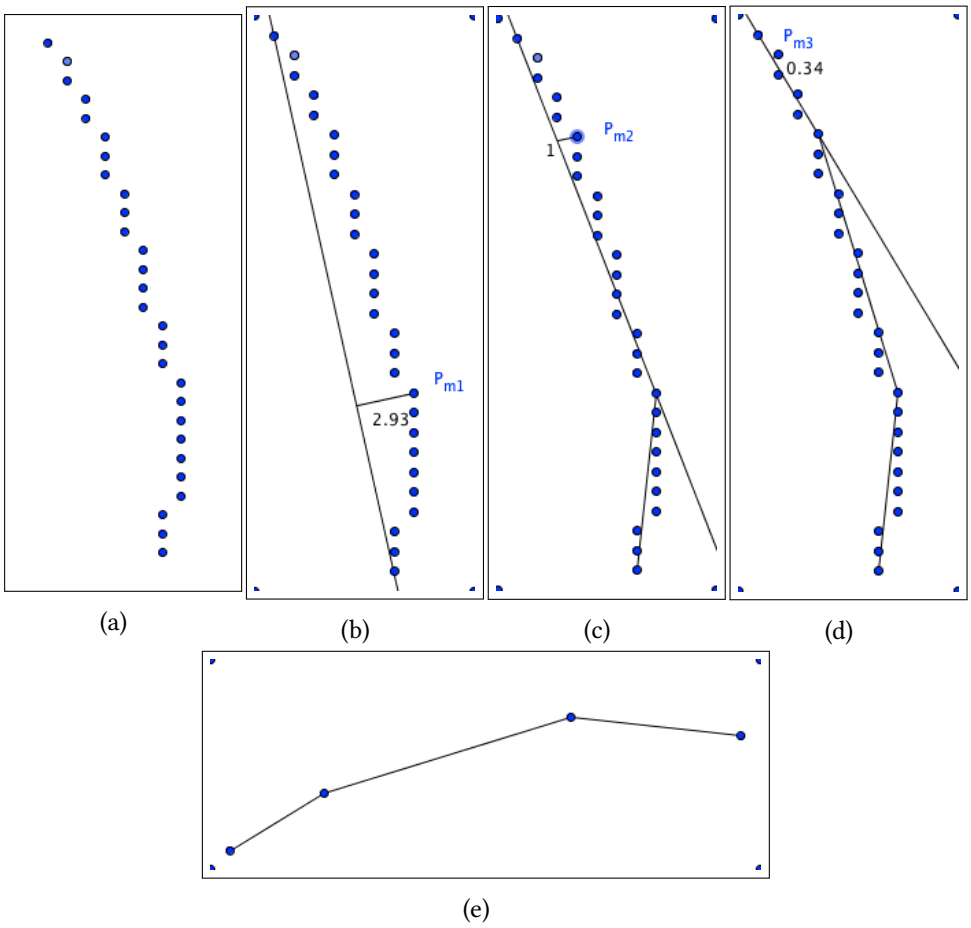



Abbildung 4.9.: Ablauf der Approximation für ein Kantensegment ($d_{tol} = 1$)

### 4.2.3. Unterteilung in Kurvenstücke

Bis zu diesem Schritt wurden die einzelnen Kantensegmente aufgesammelt und anschließend approximiert.
In diesem Schritt folgt eine genaue Untersuchung der Verlaufseigenschaften eines jeden Segments. Es liegt auf der Hand, dass bei weitem nicht alle der ermittelten Segmente zu auch zu einer sinnvollen Ellipse führen. Ein Grund dafür ist, dass durch die Segmentaufsammlung im vorherigen Schritt einfach nebeneinander liegende Pixel aufgesammelt werden und dabei zunächst nicht überprüft wird, ob die zu einem Segment zusammengefassten Pixel auch wirklich nur von der Kante eines Objekts stammen. Es werden also möglicherweise Kantenstücke unterschiedlicher Objekte und somit auch potentiell unterschiedlicher Ellipsen zusammengefasst.

Jedes Segment muss daher darauf untersucht werden, ob es potentiell Teil **einer** Ellipse ist. Dazu wird an jedem Punkt geprüft, ob in diesem Bereich drei Eigenschaften (Krümmungsbedingung, Längenbedingung und Winkelbedingung) eines Kurvenstücks erfüllt sind (vgl. NAW09, S.3281 f), welche im folgenden Verlauf einzeln erklärt und veranschaulicht werden:

- **Krümmungsbedingung** für den Verlauf der Linienstücke
- **Längenbedingung** benachbarter Linienstücke
- **Winkelbedingung** zwischen benachbarten Linienstücken

An allen Stellen, an denen einer dieser Bedingungen nicht erfüllt ist, wird angenommen, dass hier zwei unterschiedliche Kurvenstücke verbunden sind. Das ursprüngliche Segment wird daher an diesen Punkten in unabhängige Kurven aufgespalten.

### Krümmungsbedingung

Die umfangreichste der drei Bedingungen stellt die Krümmungsbedingung dar. Die Linienstücke, die auf einer Ellipse liegen, folgen immer einem Kreisbogen. Somit haben die Winkel zwischen angrenzenden Linien durchgehend die selbe Drehrichtung bzw. das gleich Vorzeichen (Abb.4.10).

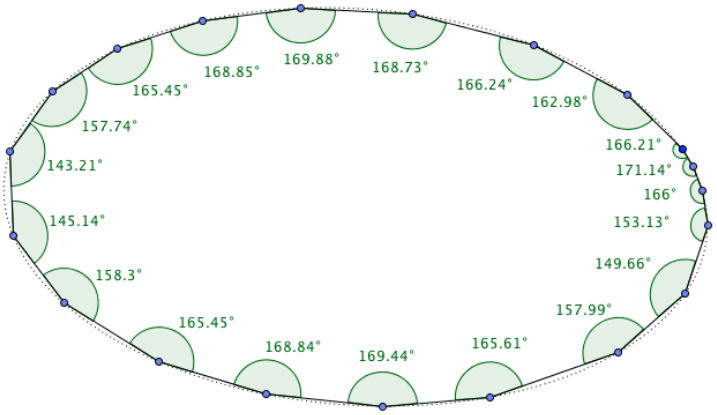



Abbildung 4.10.: Drehrichtung der Winkel benachbarter Linienstücke einer Ellipse

Aus dieser Eigenschaft ergeben sich vier Fälle, in denen ein Segment an Punkt $P$ kein gültiges Kurvenstück darstellen kann. Trift mindestens eine für die Abfolge von vier Linienstücken zu, ist $P$ als Verbindungspunkt zwischen zwei unterschiedlicher Kurvenstücke und das Segment

wird aufgespalten.

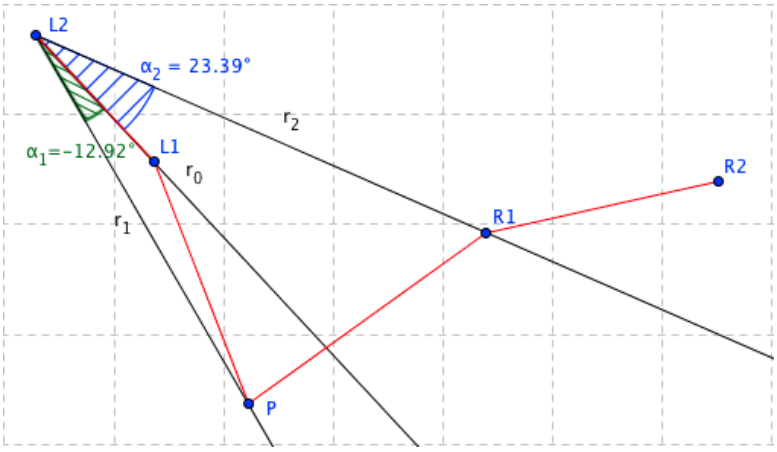



Abbildung 4.11.: Fall 1

- **Fall 1**:
  $P$ und $R_1$ liegen nicht auf der gleichen Seite von $r_0$ und können somit kein Kurvenstück bilden (Abb.4.11).

$$\alpha_1 * \alpha_2 < 0 \tag{4.10}$$

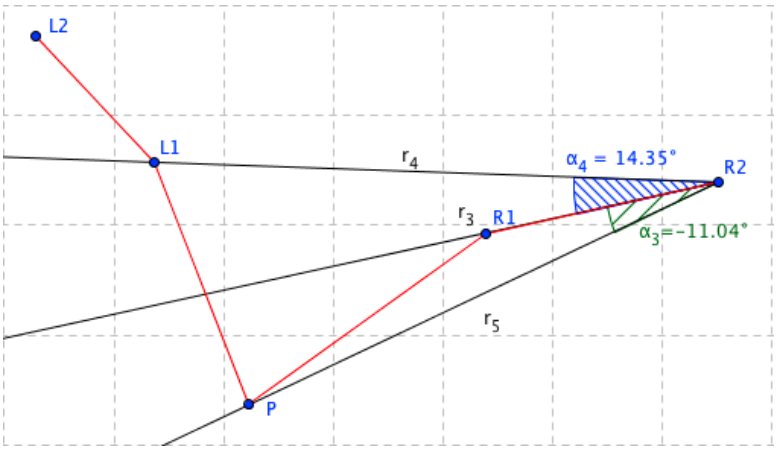



Abbildung 4.12.: Fall 2

- **Fall 2**:
  Analog zu Fall 1 nur aus der anderen Richtung für $r_3$, $P$ und $L_1$ betrachtet (Abb.4.12).

$$\alpha_3 * \alpha_4 < 0 \tag{4.11}$$

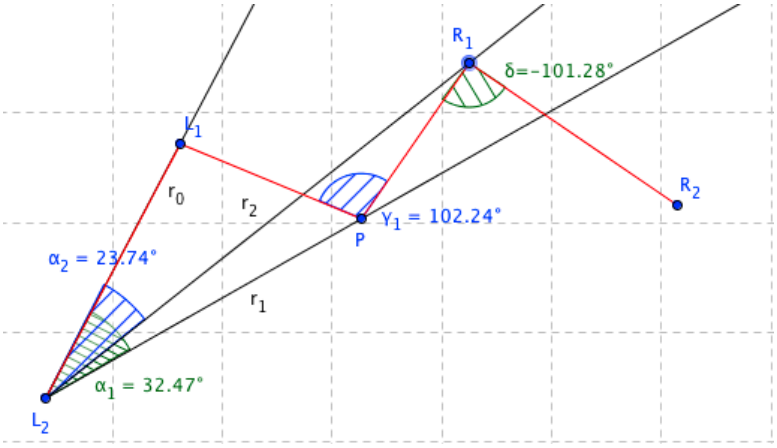



Abbildung 4.13.: Fall 3

- **Fall 3**:
  $R_1$ und $P$ befinden sich zwar auf der selben Seite von $r_0$, doch liegt bei $P$ ein Knick im Verlauf vor, wodurch die Drehrichtung von $\gamma_1$ entgegengesetzt zu $\delta$ verläuft (Abb.4.13).

$$|\alpha_2| - |\alpha_1| < 0 \tag{4.12}$$

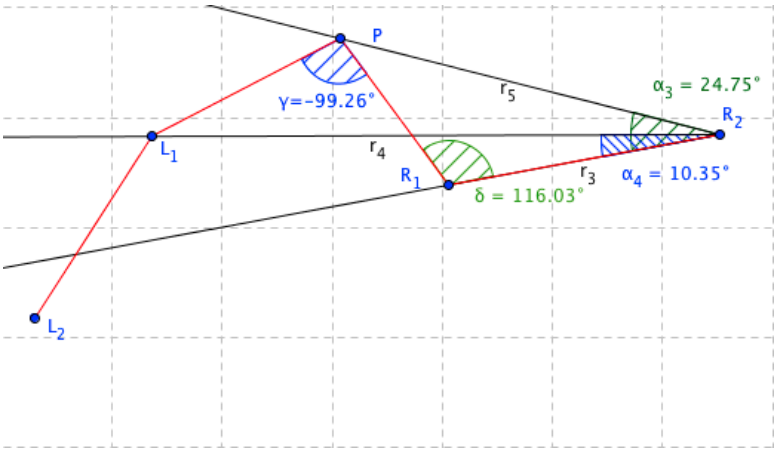



Abbildung 4.14.: Fall 4

- **Fall 4**:
  Wie Fall 3 nur aus der anderen Richtung für $P$ und $L_1$ betrachtet (Abb.4.14).

$$|\alpha_4| - |\alpha_3| < 0 \tag{4.13}$$

### Längenbedingung

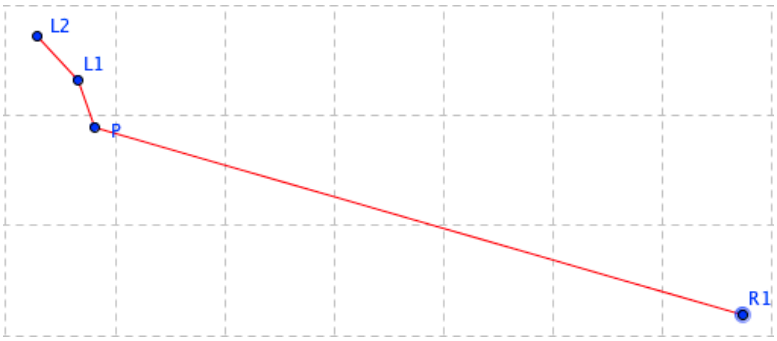



Abbildung 4.15.: Längenbedingung

$$\left(\overline{R_1P} > 4\overline{PL_1}\right) \vee \left(\overline{R_1P} < 4\overline{PL_1}\right) \tag{4.14}$$

Unterscheiden sich zwei benachbarte Linienstücke eines Segments gravierend in ihren Längen, wird an deren Verbindungspunkt $P$ eine Aufteilung in zwei Kurvenstücke vorgenommen.

### Winkelbedingung

$$(||\beta_1|| - ||\beta_2|| > TH_0) \wedge (||\beta_3|| - ||\beta_2|| > TH_0) \tag{4.15}$$

Basierend auf der Annahme, dass die Gradienten der Tangenten benachbarter Linienstücke sich kaum unterscheiden (vgl. NAW09, S.3282), wird ein Segment bei $P$ in zwei Kurvenstücke aufgeteilt, wenn sich die Winkel zwischen den benachbarten Linienstücken zu stark unterscheiden (Abb. 4.16). Für $TH_0$ gilt $TH_0 = 40°$. Hierdurch lassen sich spitz zulaufende Kantenverläufe erkennen.

### Bewertung der Ergebnisse

Abbildung 4.17 zeigt das Ergebnis der Anwendung auf vorliegende Kantensegmente. Besonders gut lässt sich die Auswirkung auf das lange Segments ganz links erkennen. Hier liegt der zuvor erwähnte Fall vor, bei dem die Kantenverläufe unterschiedlicher Objekte zusammengefasst

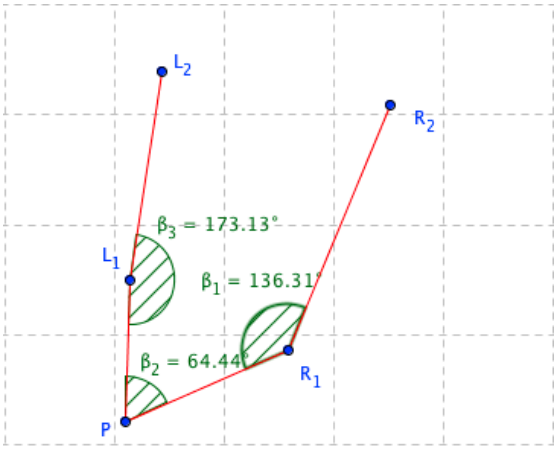
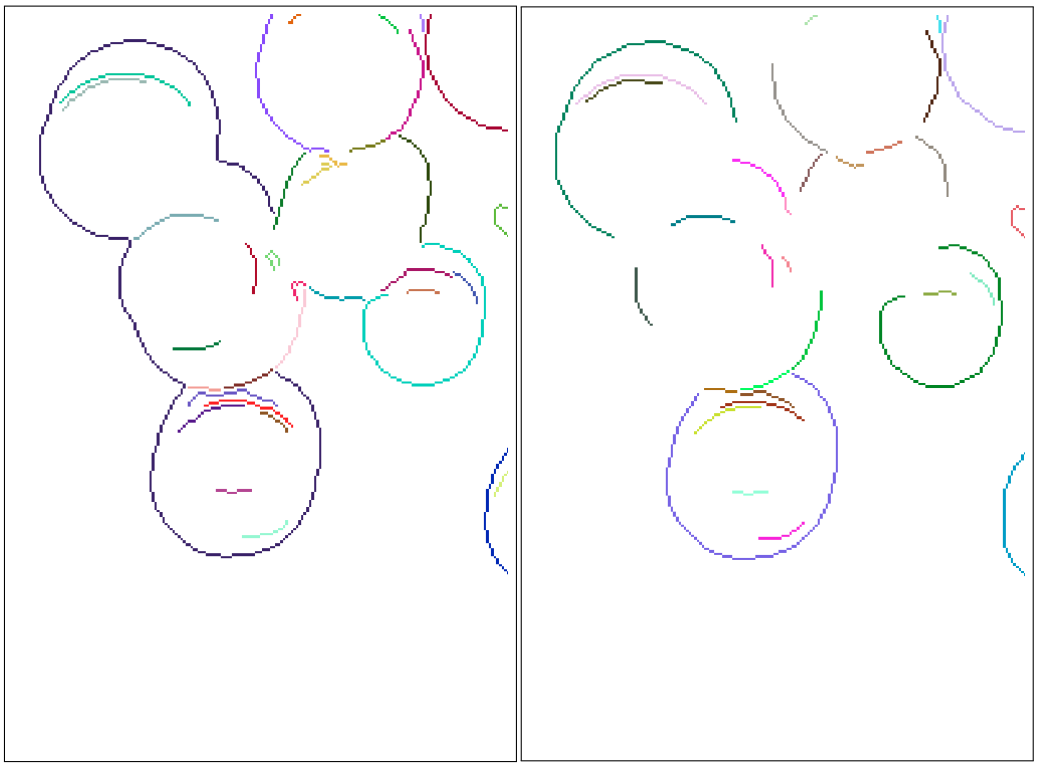



Abbildung 4.16.: Winkelbedingung

wurden. Durch die Unterteilung liegen Abschnitte der einzelnen Kantenverläufe in einzelnen Kurvenstücken vor. Die fehlenden Teile des Segments wurden verworfen, weil sie mit weniger als drei Punkten zu kurz für eine weitere Verwendung sind.



(a) Kantensegmente (b) Kurvenstücke

Abbildung 4.17.: Anwendung der Segmentierung

Es können jedoch ungültige Segmente nach der Unterteilung vorliegen, die durch keine der drei Bedingungen erkannt werden. Abbildung 4.18 zeigt ein solches Beispiel. Da $|\beta_1 - \beta_2| < 40°$ gilt, ist der erste Teil von Gl. 4.15 nicht erfüllt. Ebenso greifen weder die Längenbedingung, weil die an $P$ verbunden Linienstücke sich nicht stark in der Länge unterscheiden, noch ein Fall der Krümmungsbedingung, da ein gleichmäßiger Verlauf ohne Richtungswechsel vorliegt. Es verbleibt somit ein recht großer ungültiger Teil am Segment. Bei einer späterem Ausgleichsrechnung würde in ungünstigen Fällen also keine gute Ellipse mehr berechnet werden können.

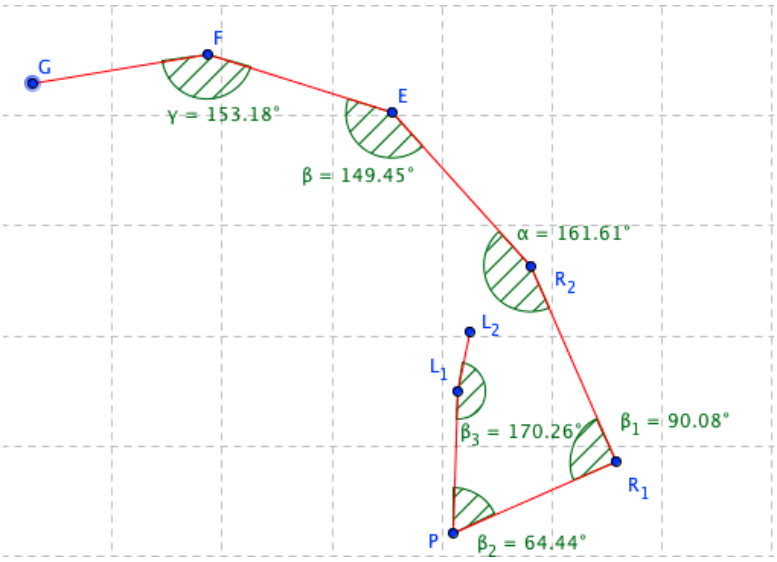



Abbildung 4.18.: Nicht unterteiltes Segment

## 4.3. Zusammenführung zu potentiellen Ellipsenstücken

Die aus der Kantensegmentierung hervorgehenden Kurvenstücke werden in diesem Schritt zu elliptischen Bögen zusammengefasst. Einzelne Kurven, die zu einer gemeinsamen Ellipse gehören, sollen so zusammen an die Ausgleichsrechnung übergeben werden, damit nicht für jede einzelne eine Ellipse berechnet wird.
Die Zusammenführung gliedert sich in zwei Teilschritte. Zuerst werden benachbarte Kurvenstücke zu Bögen zusammengefasst. Im zweiten Schritt werden diese mit weiteren, der gleichen Ellipse entsprechenden, Bögen aus dem gesamten Bild vereint.

### 4.3.1. Nachbarschaftsbasierte Zusammenführung

Zwei Kurvenstücke gelten als benachbart, wenn sie höchstens zehn Pixel voneinander entfernt liegen ($d_0 = 10$ Pixel). Die Entfernung $D_{m,n}$ der Kurven $m$ und $n$ ist der minimale Abstand zwischen den jeweiligen Endpunkten.

$$D_{m,n} = \min(\overline{P_{m,1}P_{n,1}}, \overline{P_{m,1}P_{n,N_n}}, \overline{P_{m,N_m}P_{n,1}}, \overline{P_{m,N_m}P_{n,N_n}}) \tag{4.16}$$

$N_m$ bzw. $N_n$ steht dabei für den letzten Punkt der entsprechenden Kurve $m$ bzw. $n$.

### Ablaufbeschreibung

Für jedes Kurvenstück $m$ ($m = 1...K$ *K=Anz. Kurvenstücke*) wird der Abstand zu allen anderen $n$ ($n = 1, ..., K; n \neq m$) bestimmt. Gilt

$$D_{m,n} \leq d_0 \tag{4.17}$$

wird der Winkel $\Theta_{m,n}$ zwischen den Gradienten der angrenzenden Enden von $m$ und $n$ berechnet.

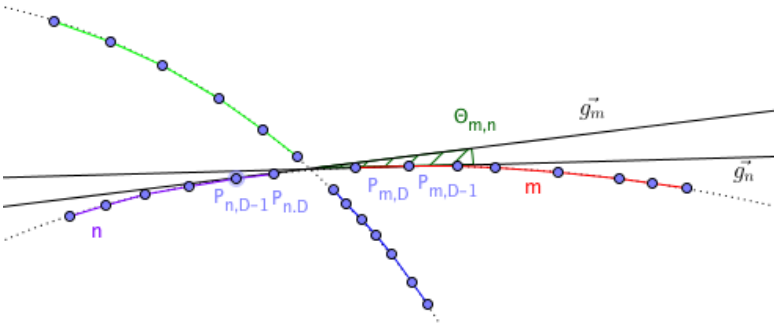



Abbildung 4.19.: Zusammenführen benachbarter Kurvenstücke

$$\Theta_{m,n} = \arccos\left(\frac{\vec{g_m} * \vec{g_n}}{|\vec{g_m}||\vec{g_n}|}\right) \tag{4.18}$$

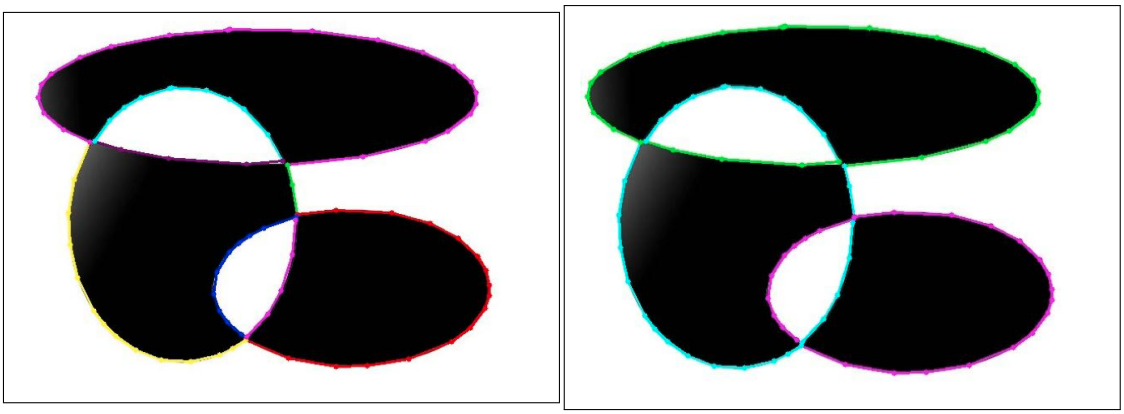
mit $\vec{g}_m = P_{m,D-1} - P_{m,D}$, $\vec{g_n} = P_{n,D} - P_{n,D-1}$ wobei $P_{m,D}$ und $P_{n,D}$ die Endpunkte mit minimalem Abstand und $P_{m,D-1}$ und $P_{n,D-1}$ jeweiligen Punkte vor $P_{m,D}$ bzw. $P_{n,D}$ darstellen. $g_m$ und $g_n$ sind die Richtungsvektoren der Linienstücke an den entsprechenden Enden der Kurven (Abb.4.19).
Ein Kurvenstück $m$ wird mit dem $j$ zu einem elliptischen Bogen zusammenfasst, das den kleinsten Winkel zwischen den Gradienten aufweist.

$$\Theta_{m,j} = \min_{1 \leq n \leq K} (\Theta_{m,n}) \tag{4.19}$$

Nachdem die Zusammenführung für jedes $m$ durchgeführt wurde, liegen $L$ elliptische Bögen vor ($L \leq K$).

### Bewertung der Ergebnisse

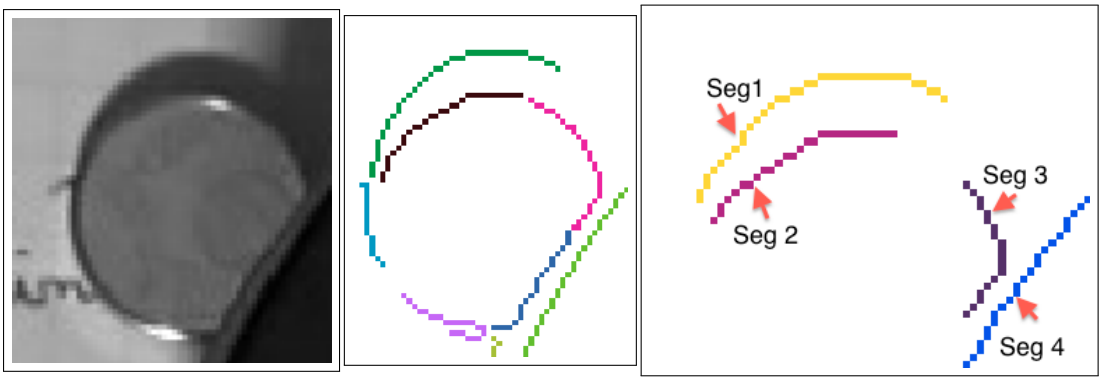
Die Zusammenfassung von zwei benachbarten Kurvenstücken funktioniert vom Grundgedanken einwandfrei. Abbildung zeigt ein aus der Veröffentlichung entnommenes synthetisches Beispiel. Für jeden der Bögen liegt in diesem Fall ein passendes benachbartes Stück vor, dass auch wie erwünscht durch die Berechnung des minimalen Winkels $\Theta_{m,n}$ gefunden wird. In allen Bildern, die nur solche eindeutigen Situationen wie in Abb. 4.20 aufweisen, ist nichts an der Nachbarschaftsbasierten Zusammenführung auszusetzen.



(a) Kurvenstücke (b) Elliptische Bögen

Abbildung 4.20.: Zusammenführung für eindeutige Anordnung der Kurvenstücke (NAW09, S.3282)

Betrachtet man jedoch reale Bilder, liegt in den wenigsten Fällen ein eindeutiger Fall vor. Strenggenommen stellt das Kurvenstück $j$, das Gl. 4.19 für das Kurvenstück $m$ erfüllt, lediglich das am wahrscheinlichsten zu einer gemeinsamen Ellipse gehörende Gegenstück dar. Das Problem ist, dass es keinerlei Einschränkung für eine maximale Differenz der Winkel zwischen $m$ und $j$ gibt. Liegt nun kein passendes Kurvenstück für $m$ vor - passend bedeutet, dass zwei gefundene Kurvenstücke auch wirklich Teile einer gemeinsamen Ellipse darstellen -, dafür aber mindestens eines, das nur die Nachbarschaftsbedingung 4.17 erfüllt, werden zwei falsche Kurven zusammengefasst.




(a) Ausschnitt des Quellbildes (b) Kantensegmente (c) Kurvenstücke

Abbildung 4.21.: Problematische Anordnung von Kurvenstücken

In Abbildung 4.21 ist das Ergebnis des Segmentierung für den Ausschnitt eines realen Bildes dargestellt. Aufgrund der Ungleichmäßigkeiten fallen einige Teile der Verläufe weg. Übrig bleiben die vier Kurvenstücke in Abb. 4.21c. Segment *Seg 1* und *Seg 2* gelten als benachbart, ergeben aber keine gemeinsame Ellipse. Da kein anderes benachbartes Segment vorliegt, weist *Seg 2* den minimalen (und einzigen) Winkel zwischen den einander zugewandten Enden auf und würde mit *Seg 1* zu einem elliptischen Bögen zusammengefasst werden. In der folgenden Ausgleichsrechnung würde für diesen Bogen keine gültige Ellipse ermittelt werden.
Für die Anwendung auf reale Bilder liefert die Nachbarschaftsbasierte Zusammenführung also je nach vorliegenden Kurvenstücken grobe Fehler und ist somit nicht in dieser Form verwendbar.

### 4.3.2. Globale Zusammenführung

Im vorherigen Schritt wurden nur benachbarte Kurvenstücke vereint. Einzelne Teile einer Ellipse können jedoch auch mehr als zehn Pixel voneinander entfernt liegen. Aus diesem Grund werden die $L$ elliptischen Bögen in diesem Schritt ebenfalls auf potentielle Zusammengehörigkeit untersucht.

#### Ablaufbeschreibung

Für jede elliptischen Bogen $m$ ($m = 1...L$ *L=Anz. elliptische Bögen*) wird der Abstand zu allen anderen $n$ ($n = 1, ..., K; n \neq m$) gemäß Gl. 4.16 bestimmt. Gilt

$$D_{m,n} > d_0 \tag{4.20}$$

wird anhand einer Orientierungsbedingung berechnet, ob $m$ und $n$ zusammengefasst werden.

$$\overline{P_{m,round(\frac{N_m}{2})}P_{n,round(\frac{N_n}{2})}} > \overline{C_m P_{n,round(\frac{N_n}{2})}}$$
$$and \tag{4.21}$$
$$\overline{P_{m,round(\frac{N_m}{2})}P_{n,round(\frac{N_n}{2})}} > \overline{C_n P_{m,round(\frac{N_m}{2})}}$$

mit $C_m = \frac{P_{m,1}+P_{m,N_m}}{2}$, $C_n = \frac{P_{n,1}+P_{n,N_n}}{2}$. $C_m$ und $C_n$ stellen jeweils den Mittelpunkt auf der Geraden zwischen den beiden Endpunkten eines Bogens dar. $P_{m,round(\frac{N_m}{2})}$ und $P_{n,round(\frac{N_n}{2})}$ sind die "mittleren"Punkte der von $n$ bzw. $m$. Einfacher lassen sich die Punkte aus Abbildungen 4.22 und 4.22 erkennen. Durch diese Orientierungsbedingung 4.21 wird überprüft, ob zwei Bögen $m$ und $n$ ihre Öffnungßueinander gerichtet haben, wie in Abb. 4.22 zu sehen, oder voneinander abgewandt verlaufen, wie in Abb. 4.23 zu erkennen ist.
Nachdem die Zusammenführung für jedes $m$ durchgeführt wurde, liegen $L$ elliptische Bögen und die aus diesem Teilschritt hervorgehenden $H$ Bögen-Paare vor.

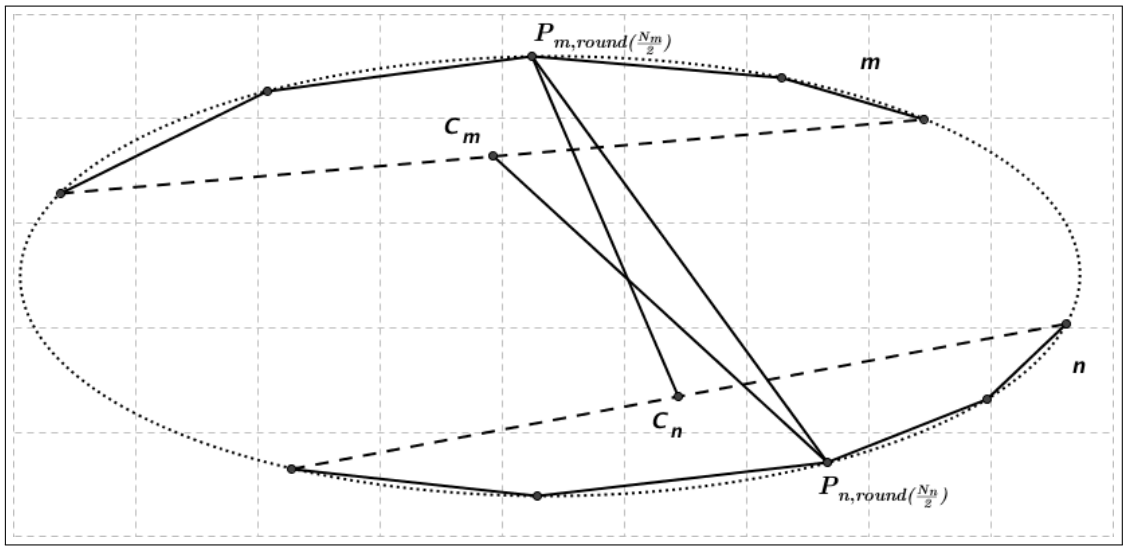



Abbildung 4.22.: Einander zugewandte Bögen

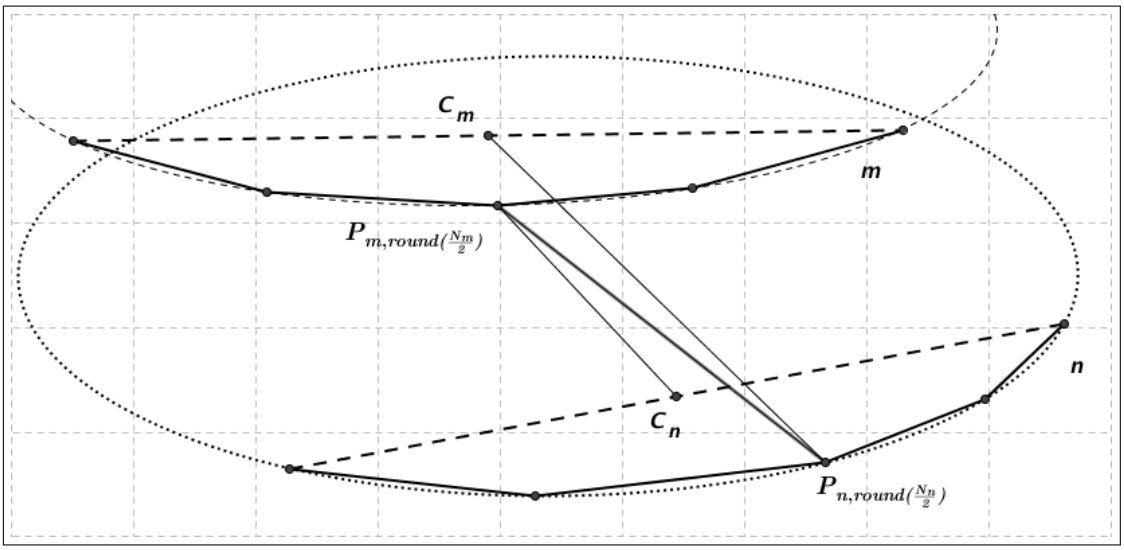



Abbildung 4.23.: Voneinander abgewandte Bögen

### Bewertung der Ergebnisse

Genau wie bei der Zusammenführung in der Nachbarschaft, liefert die globale Zusammenführung in der angewandten Form nur für eindeutige Fälle richtige Ergebnisse. Liegen nur zwei Bögen wie in Abb. 4.22 vor, werden diese auch zu einem Paar vereint. Sobald aber weitere Bogen dazu kommen, die ebenfalls die Orientierungsbedingung zu einem der anderen beiden erfüllt, hängt das Ergebnis der Paarbildung ausschließlich von der Realisierung in der Implementierung ab. Der erste Bogen, der die Bedingung erfüllt, wird gemäß des Ablaufs zur Zusammenfassung herangezogen. Es gibt keine weitere Überprüfung, ob ein anderer Bogen ebenfalls die Bedingung erfüllt oder vielleicht wahrscheinlicher auf einer gemeinsamen Ellipse liegt. Ein solcher Fall wird in Abbildung 4.24 dargestellt.

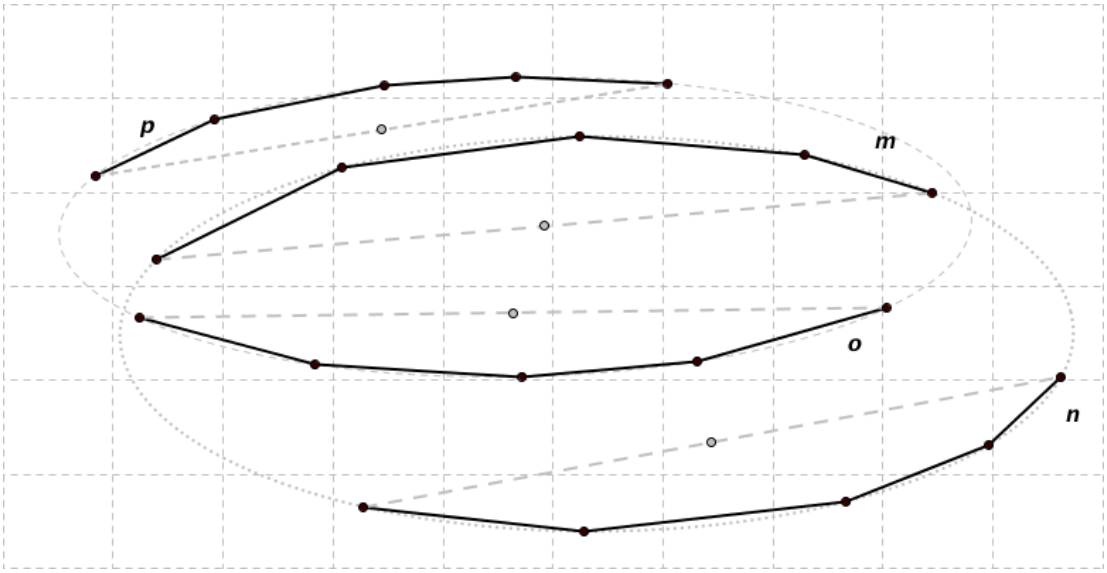



Abbildung 4.24.: Situation mit mehreren möglichen Paarbildungen

Die Bögen $o$ und $n$ erfüllen sowohl zu $m$ als auch zu $p$ die Orientierungsbedingung 4.21. Würde in der Implementierung zuerst das Verhältnis von $o$ zu $m$ berechnet, würden diese beiden als mögliches Paar zusammengefasst, obwohl eigentlich $m$ mit $n$ auf einer gemeinsamen Ellipse liegt. Würde anschließend das Verhältnis von $p$ zu $n$ betrachtet, würden auch diese beiden ein Bogenpaar bilden, das zu keiner sinnvollen Ellipse führt.
Hinzu kommt, dass mit der noch folgenden Ausgleichsberechnung die beiden angedeuteten Ellipsen nicht mehr berechnet werden könnten. Um eine passende Ausgleichsellipse zu erhalten, müssen ausreichend gute und möglichst keine falschen Datenpunkte in die Berechnung einfließen. In dem vorliegenden Fall würden die Paare $(m, o)$ und $(p, n)$ jeweils zur Hälfte falsche Werte aufweisen. Diese Problematik wird näher im Abschnitt 4.4.3 veranschaulicht.

Die globale lässt sich somit ebenso wenig auf reale Bilder anwenden, wie die nachbarschaftsbasierte Zusammenführung. Dort liegen nämlich sehr viele mögliche Kandidaten für eine Paarbildung vor. Daraus nur den erst passenden zu wählen, würde vielen Fällen zu dem oben beschriebenen Szenario führen und das Auffinden vieler vorhandenen Ellipsen im schlimmsten Fall verhindern.

## 4.4. Berechnung der Ellipsen

Im letzten Bearbeitungsschritt werden für die $L$ elliptischen Bögen aus der Nachbarschaftszusammenführung und die $H$ Bogenpaare aus der globalen Zusammenführung die jeweils am besten zu allen zugehörigen Pixel passende Ausgleichsellipse berechnet. Dafür wird eine speziell auf Ellipsen angepasste Methode der minimalen Fehlerquadrate verwendet.

### 4.4.1. Ellipsen spezifische Methode der Fehlerquadrate

Zur Ausgleichsrechnung wird die aus Kapitel 2.2 bekannte allgemeine Kegelschnittgleichung (Gl. 2.6) herangezogen. In Matrixschreibweise ergibt sich:

$$F_a(X) = X * a = 0 \tag{4.22}$$

mit $a = \begin{bmatrix} A & B & C & D & E & F \end{bmatrix}^T$ und $X = \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}$
Ziel ist es, die Ellipse zu finden, die für alle zu betrachtenden Punkte den kleinst möglichen Abstand zu dieser liefert. Der Abstand eines Punktes $(x_i, y_i)$ zu $F_a(X) = 0$ ist definiert durch die algebraische Distanz $d$.

$$d = |F_a(x_i, y_i)| \tag{4.23}$$

Der optimale Koeffizientenvektor $a$ liegt somit genau dann vor, wenn der quadratische Abstand aller Punkte minimal ist:

$$\min_a = \sum_{i=1}^{N} (F_a(x_i, y_i)^2) = \min_a = \sum_{i=1}^{N} (F_a(X_i)^2) = \min_a = \sum_{i=1}^{N} ((X_i * a)^2) \tag{4.24}$$

Wie in Kapitel 2.2 erwähnt, beschreiben Kegelschnitte nicht nur Ellipsen. Somit stellt der durch Gl. 4.24 ermittelte Vektor $a$ einen beliebigen optimalen Kegelschnitt für die vorliegenden Punkte dar. Es muss also festgestellt werden, ob die berechneten Koeffizienten die hinreichende Bedingung für eine Ellipse erfüllen.

$$4AC - B^2 > 0 \tag{4.25}$$

Durch direktes Einbeziehen der Bedingung 4.25 in die Ausgleichsrechnung lässt sich sicherstellen, dass die Daten wirklich nur durch eine Ellipse approximiert werden (FPF99). Dazu wird die Ungleichung 4.25 angepasst:

$$4AC - B^2 = 1 \tag{4.26}$$

Diese Anpassung ist möglich, da sich die berechnete Ellipse durch beliebige Skalierung der Koeffizientenmatrix $a$ durch einen Faktor $\alpha \neq 0$ nicht verändert (vgl. FPF99, S.600).

$$F_a(X) = F_{\alpha * a}(x)(,)\alpha \neq 0 \tag{4.27}$$

Somit lässt sich Gl. 4.24 für das ellipsenspezifische Minimierungsproblem umformulieren:

$$\min_a = ||Da||^2, \text{ mit } aC^T a = 1 \tag{4.28}$$

$D$ stellt die *Design-Matrix* für alle $N$ zu berücksichtigen Punkte dar:

$$D = \begin{bmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 & x_N y_N & y_N^2 & x_N & y_N & 1 \end{bmatrix} \tag{4.29}$$

Die Bedingung 4.26 für die Koeffizienten wird durch die *Constraint-Matrix* $C$ repräsentiert:

$$C = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.30}$$

Durch Anwenden der Lagrange Verfahrens ergibt sich für die optimale Lösung $a$:

$$Sa = \lambda Ca, \text{ mit } a^T Ca = 1 \text{ und } S = D^T D \text{ (Scatter-Matrix)} \tag{4.31}$$

Gl. 4.31 stellt ein generalisiertes Eigenwertproblem dar. Von allen Eigenvektoren $a$ liefert genau das $a_k$ die optimale Koeffizienten der Kegelschnittgleichung, dessen zugehöriges $\lambda_k$ den kleinsten positiven Eigenwert darstellt.

Das Problem dieser Vorgehensweise ist, dass für eine Menge von Punkten, die exakt auf einer Ellipse liegen, falsche oder keine Lösungen gefunden werden kann (FH98). Für diesen Fall wäre die *Scatter-Matrix* $S$ singulär, wodurch der Eigenwert $\lambda_k$ der optimalen Koeffizienten

$a_k$ nicht der kleinste positive Eigenwert ist, sondern $\lambda_k = 0$ gilt. Durch Ungenauigkeiten in der numerischen Berechnung der Eigenwerte kann kann $\lambda_k$ sogar ein sehr kleiner negativer Wert sein.
Aus diesem Grund verwendet der original Algorithmus zu Berechnung der am besten zu den Daten passenden Ellipse eine Optimierung der vorab erläuterten Vorgehensweise von J. Flusser (FH98). Da sich die Verbesserung ausschließlich auf den Problemfall der exakten Übereinstimmung aller Punkte mit einer Ellipse bezieht, möchte ich diese nicht weiter ausführen. Das Berechnungsprinzip bleibt grundlegend gleich und angesichts der mathematischen Komplexität der Anpassung, scheint mir eine Betrachtung und Erläuterung als nicht sinnvoll.

### 4.4.2. Verifizierung gültiger Ellipsen

Zwar liefert die Ausgleichsrechnung aus dem vorherigen Abschnitt immer eine Ellipse für alle gewünschten Punkte, jedoch muss überprüft werden, ob die betrachteten Daten überhaupt auch der Ausgleichsellipse entsprechen. Dazu werden für die jede berechnete Ellipse $E_m$ zunächst doppelt so viele auf ihr befindliche Punkte wie in Bogen/ Bogenpaar $m$ enthalten sind bestimmt. Für alle diese Ellipsenpunkte wird der Abstand eines jeden zum nächstliegenden Kantenpixel $P_i$ von $m$ berechnet. $E_m$ ist dann durch $N_{E_m} = 2N_m$ Punkte $P_{E_m,k}$ $(k = 1, 2, ..., N_{E_m})$ dargestellt. Die Summer der Abstände wird mit der kleinen Halbachse $b$ gewichtet.

$$D_{E_m,m} = \frac{1}{N_{E_m} b} \sum_{k=1}^{N_{E_m}} (P_{E_m,k} - P_i) \tag{4.32}$$

Für eine gültige Ellipse gilt:

$$D_{E_m,m} < TH_E \tag{4.33}$$

$TH_E$ ist dabei ein global festgelegter Grenzwert.

### 4.4.3. Bewertung der Ergebnisse

Für elliptische Bögen und Bogenpaare, deren enthaltenen Punkte einen großen Teil oder unterschiedliche Bereiche einer im Bild befindlichen Ellipse abdecken, liefert die Ausgleichsrechnung gute Ergebnisse. Dadurch das die beiden Bögen in Abbildung 4.25 gegenüberliegende Abschnitte der Ellipse abdecken, entspricht die Ausgleichsellipse (durchgezogene Linie) der tatsächlichen (gestrichelte Linie).

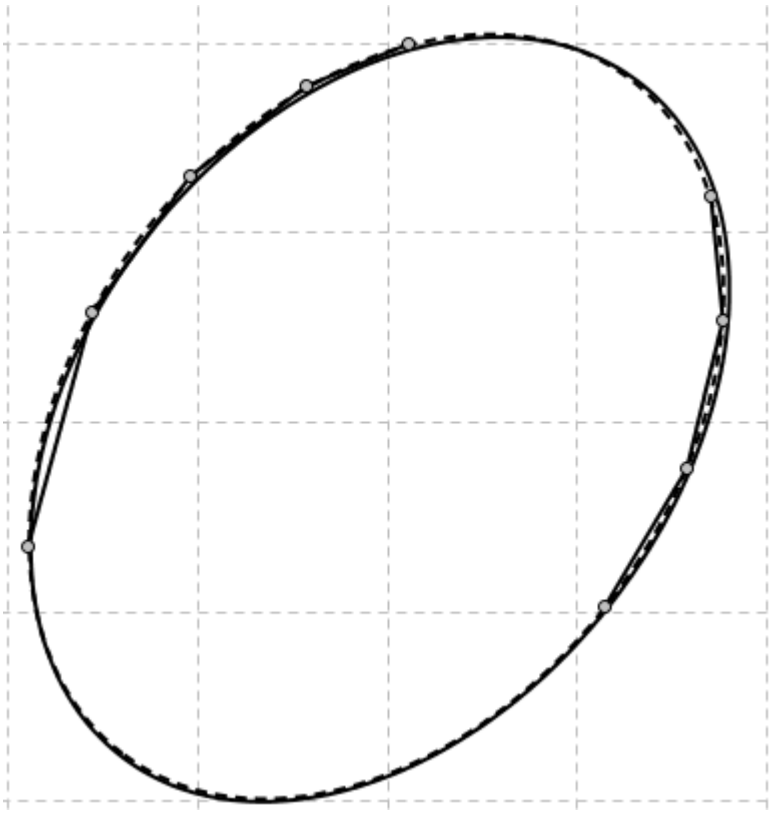


Abbildung 4.25.: Ausgleichsellipse für einen gutes elliptischen Bogenpaar

Zu beachten ist der bereits erwähnte starke Einfluss von falschen Werten. An dieser Stelle greife ich das Beispiel für eine unglückliche Zusammenfassung von elliptischen Bögen durch die globale Zusammenführung in Abb. 4.24 wieder auf. Hier wurden $m$ und $o$ sowie $p$ und $n$ fälschlicherweise zu Paaren zusammengeführt. Die beiden Paare werden jeweils an die Ausgleichsrechnung übergeben. Die beiden berechneten Ellipsen $E_{p,n}$ und $E_{m,o}$ haben keine Übereinstimmung mehr mit denen, auf denen sich $m$ und $n$ bzw. $o$ und $p$ eigentlich befinden (gestrichelt dargestellt).

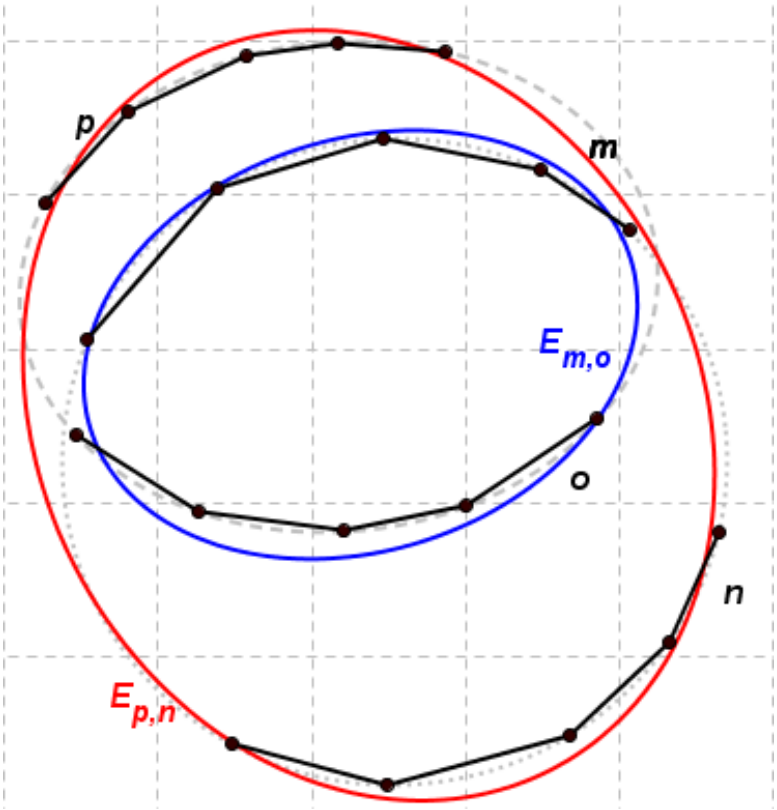



Abbildung 4.26.: Ausgleichsellipse für schlechte Bogenpaare

Damit die Ausgangsrechnung gute Ergebnisse liefern kann, ist es also wichtig, dass nur einer Ellipse entsprechende Daten zur Berechnung herangezogen werden.

Außerdem ist es schwer, den globalen Grenzwert $TH_E$ für die Verifizierung von gültigen Ellipsen zu bestimmen. Für große Ellipsen, die durch Bögen oder Bogenpaare mit vielen Punkten berechnet werden, ergibt sich bei der Aufsummierung der Abstände aller Ellipsenpixel zum nächstgelegenen Kantenpixel aus Gleichung 4.32 aufgrund der hohen Anzahl der zu betrachtenden Punkte, ein hoher Wert für $D_{E_m,m}$. Auch durch die zusätzliche Gewichtung mit der kleinen Halbachse $b$, unterscheidet sich die Abstandsumme deutlich von kleineren Ellipsen mit weniger Punkten. Es lässt sich anhand von $D_{E_m,m}$ nicht unterscheiden, ob ein elliptischer Bogen $m$ seiner Ellipse $E_m$ nicht entspricht oder ob $m$ und somit $E_m$ einfach nur sehr viele Punkte berücksichtigt. Für eine sinnvolle Bewertung der Ausgleichsellipsen muss ein anderes Kriterium gefunden werden.

## 4.5. Problemzusammenfassung

Dieser Abschnitte liefert einen zusammenfassenden Überblick zu den Problemen der zuvor analysierten Bearbeitungsschritte.
Die Unterteilung in Kurvenstücke kann für bestimmte Kantensegmente ungültige Kurven liefern, die noch aus zwei unterschiedliche Kurvenverläufe bestehen.
Sowohl die nachbarschaftsbasierte als auch die globale Zusammenführung ermöglichen in bestimmten Fällen die Vereinigung von nicht auf einer gemeinsamen Ellipse liegenden Kurven.
Die falschen Ergebnisse dieser beiden Schritte nehmen direkten Einfluss auf die Berechnung der möglichen Ellipsen. Daraus können falsche, nicht im Bild befindliche Ellipsen resultieren, aber auch ursprünglich vorhanden Ellipsen nicht mehr ermittelt werden.
Außerdem lässt sich kein allgemeingültiger Grenzwert $TH_E$ zur Verifizierung der Ausgleichsellipsen festlegen.

Alle anderen Bearbeitungsschritte haben bei der Überprüfung zufriedenstellende Ergebnisse geliefert und weisen keinen Optimierungsbedarf auf.

# 5. Optimierung des Verfahrens

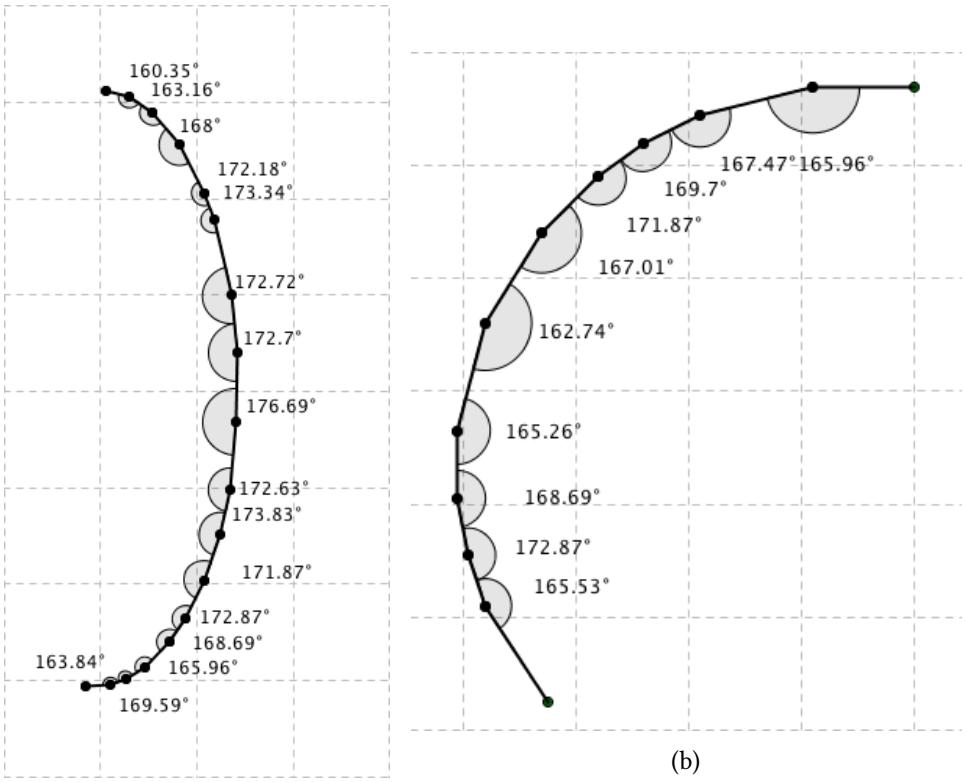
Dieses Kapitel umfasst die erarbeiteten Verbesserungen für die im vorherigen Kapitel festgestellten Probleme. Die Optimierung umfasst eine angepasst Aufspaltung der Kantensegmente in Kurvenstücke (Abschnitt 5.1) und eine robuste Vorgehensweise zur Berechnung aller im Bild befindlichen Ellipsen aus den vorliegenden Kurvenstücken (Abschnitt 5.2). Die Ergebnisse der Berechnung werden anschließend für ausgewählte Quellbilder dargestellt und erläutert (Abschnitt 5.3).

## 5.1. Verbesserte Unterteilung in Kurvenstücke

Die verbesserte Unterteilung der Kantensegmente basiert auf einer Modifizierung der Winkelbedingung, um die zuvor nicht erkannten Knicke im Segmentverlauf zu erkennen. Zusätzlich beinhaltet sie eine Anpassung der Krümmungsbedingung, die eine wesentlich einfachere Drehrichtungsbestimmung der Winkel zwischen benachbarten Linienstücken ermöglicht. Die Längenbedingung wird unverändert weiterverwendet.
Nach einer Beschreibung der neuen Vorgehensweise in Abschnitt 5.1.1 wird die Verbesserung zur vorherigen Unterteilung exemplarisch veranschaulicht (Abschnitt 5.1.2).

### 5.1.1. Ablaufbeschreibung

Betrachtet man die Winkelverläufe in zu Ellipsen führenden Kurvenstücke, die aus der Unterteilung des vorherigen Kapitels hervorgehen, lässt sich erkennen, dass die Winkel einer Kurve durchgehend ähnliche Werte haben. Die Winkel der Beispielkurve (a) befinden sich im Bereich $160.35° - 173.83°$ und die von (b) im Bereich $162.74° - 172.87°$ (Abb. 5.1).




(a) (b)

Abbildung 5.1.: Winkelverlauf in Kurvenstücken

Somit lässt sich das Ende eines elliptischen Verlaufs in einem Kantensegments durch den Vergleich zwei aufeinanderfolgender Winkel erkennen. Gilt für zwei Winkel $\beta_1$ und $\beta_2$

$$\beta_1 - \beta_2 > TH; \text{mit } TH = 25° \tag{5.1}$$

wird das Segment am Verbindungspunkt der zu $\beta_1$ gehörenden Linienstücke aufgeteilt.

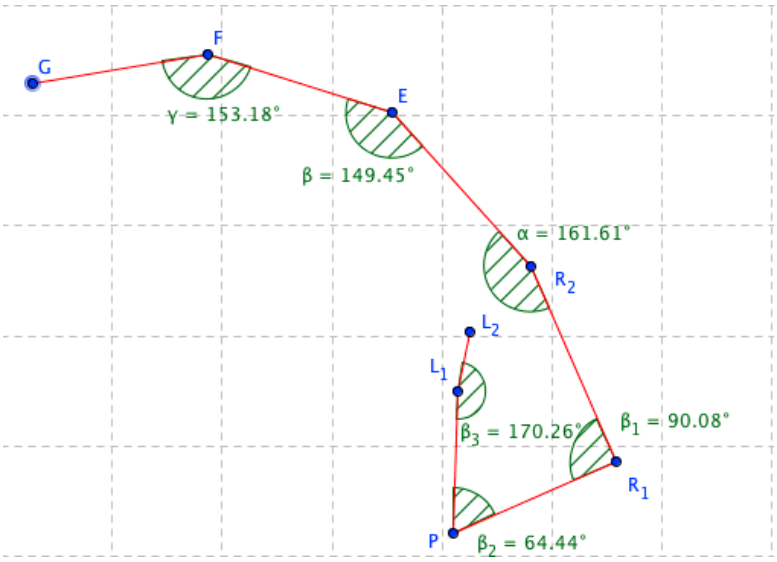



Abbildung 5.2.: Problemfall aus Abschnitt 4.2.3

Auf den aus Abbildung 4.18 in Abschnitt 4.2.3 bekannten Problemfall angewendet, wird das Segment nach dem Vergleich von $\alpha$ und $\beta_1$ am Punkt $R_1$ unterteilt. Die störenden Knicke in Kurvenverläufen werden also durch diese Überprüfung erkannt.

Die zuvor durch die Überprüfung der vier mögliche Fälle der Krümmungbedingung ermittelten Richtungswechsel im Kantenverlauf, lassen sich deutlich leichter erkennen. Hierzu betrachtet man das auf den zweidimensionalen Raum übertragene Kreuzprodukt für Vektoren.

$$K = \vec{a} \times \vec{b} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \times \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = a_1 * b_2 - b_1 * a_2 \tag{5.2}$$

Das Vorzeichen $K$ gibt Aufschluss über die Drehrichtung des Winkels zwischen $\vec{a}$ und $\vec{b}$. Diese Erkenntnis lässt sich zur Auswertung der Kantensegmente verwenden.

Dazu berechnet man das Kreuzprodukt $K_{a,b}$ zweier Linienstücke $a$ und $b$ anhand deren der Richtungsvektoren $\vec{r_a}$ und $\vec{r_b}$, die gemäß dem in Abbildung 5.3 dargestellten Schema bestimmt werden.

$$K_{a,b} = \vec{r_a} \times \vec{r_b} \tag{5.3}$$

Damit drei aufeinanderfolgende Linienstücke $a$, $b$ und $c$ eine gültige Kurve beschreiben,

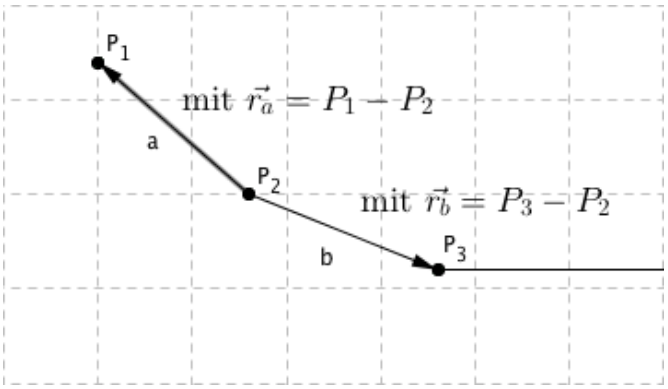


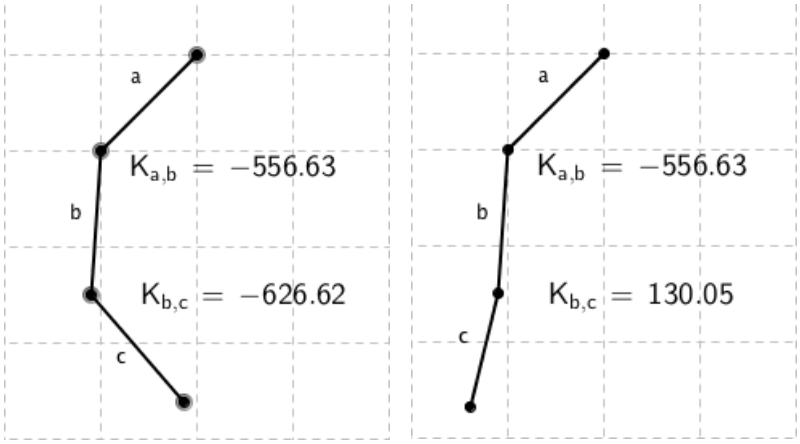

Abbildung 5.3.: Bestimmung der Richtungsvektoren

müssen die Vorzeichen der berechneten Produkte $K_{a,b}$ und $K_{b,c}$ übereinstimmen.

$$K_{a,b} * K_{b,c} > 0 \tag{5.4}$$

Gilt Gl. 5.4 nicht, wird das Segment am Verbindungspunkt von $a$ und $b$ aufgeteilt.

Abbildung 5.4 zeigt ein Beispiel für einen gültigen Verlauf und einen ungültigen Verlauf. Das Segment aus 5.4a erfüllt $K_{a,b} * K_{b,c} > 0$ und weist somit einen gültigen Verlauf auf. Für das zweite Segment5.4b gilt hingegen $K_{a,b} * K_{b,c} < 0$, weshalb es zwischen $a$ und $b$ aufgetrennt wird.




(a) Gültiger Verlauf (b) Ungültiger Verlauf

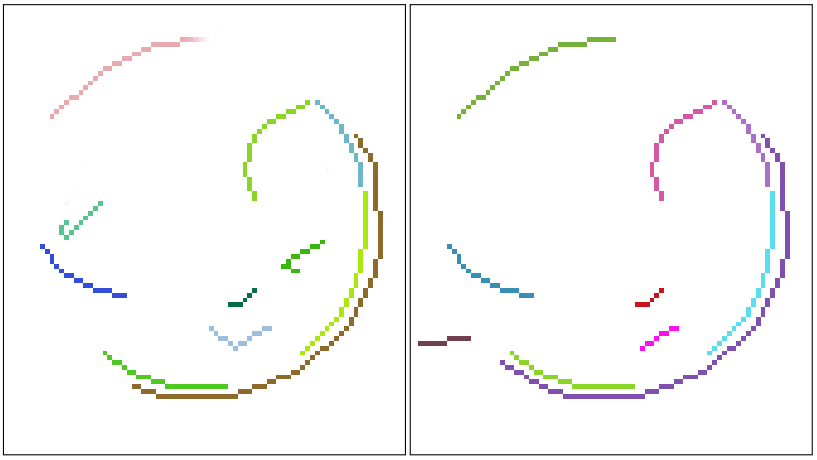
Abbildung 5.4.: Beispiele für Segmentverläufe

Nach der Unterteilung aller Segmente, liegen nur noch gültige Kurven vor, die mindestens vier Pixel aufweisen.

### 5.1.2. Vergleich mit der alten Unterteilung

Um den Unterschied zwischen der optimierten und der vorherigen Unterteilung besser nachvollziehen zu können, vergleiche ich Ergebnisse der beiden Abläufe für gleiche Ausgangssegmente (Abb. 5.5).

Abbildung 5.5.: Kantensegmente vor der Unterteilung



(a) Alte Unterteilung (b) Überarbeitete Unterteilung

Abbildung 5.6.: Kurvenstücke der beiden Vorgehensweisen

Das Ergebnis der alten Unterteilung (Abb. 5.6a) weist drei ungültigen Kurvenstücke mit Knick im Verlauf auf, welche alle von der optimierten Vorgehensweise (Abb. 5.6b) entdeckt und aufgetrennt werden.

## 5.2. RANSAC basierte Ellipsenberechnung

In diesem Abschnitt stelle ich eine neue Berechnung der im Bild befindlichen Ellipsen aus den vorliegenden Kurvenstücken, die aus der verbesserten Unterteilung hervorgehen. Dabei orientiere ich mich an einer fehlertoleranten Vorgehensweise *RANSAC* von M.A. Fischler und R.C. Bolles zur Berechnung von mathematischen Modellen aus experimentellen Daten (FB81). Zunächst beschreibe ich das zugrunde liegende Funktionsprinzip (Abschnitt 5.2.1) und anschließend die Anwendung auf die Ellipsenberechnung (Abschnitt 5.2.2).

### 5.2.1. Funktionsweise von RANSAC

Das herausgestelltes Problem der verwendeten Ausgleichsrechnung in Abschnitt 4.4 ist die starke Sensibilität auf bereits wenige vom Modell abweichenden Werten, das durch die Berücksichtigung aller vorliegenden Punkte resultiert. Um selbst gegen höhere Anzahlen von fehlerhaften Daten robust zu sein, betrachtet RANSAC nur genau so viele zufällig ausgewählte Werte, wie für die Berechnung eines Modells nötig sind. Der Ablauf folgt dabei einem einfachen Muster:

1. **Bilden der Menge** $S_1$:
   Auswahl von $n$ zufälligen Werten aus Menge aller vorhandenen Daten $P$
2. **Berechnen des Modells** $M$ **mit** $S_1$
3. **Bilden der *consensus set*** $S_1^*$ **zu** $M$:
   Bestimmung aller Werte aus $P$, die unter Berücksichtigung einer Fehlertoleranz $t$ mit dem zuvor berechneten Modell $M$ übereinstimmen
   - $|S_1^*| \geq n_{min}$:
     Neuberechnung des Modell $M$ mit $S_1^*$;
     Ende des Ablaufs
     ($|S_1^*|$= Mächtigkeit von $S_1$, $n_{min}$= Mindestgröße)
   - $|S_1^*| < n_{min}$:
     Wiederholen der Schritte 1-3

Liegt nach einer festgelegten Anzahl von Wiederholungen $T_{max}$ der Schritte 1-3 keine gültige *consensus set* vor, berechnet man $M$ entweder mit der größtmöglichen consensus set oder

meldet einen Fehler.

Durch die zufällige Auswahl von nur so vielen Datensätzen, wie gerade zur Berechnung benötigt werden und der anschließende Abgleich des Modells mit allen vorhandenen Werten, besteht selbst bei einer hohen Anzahl von abweichenden Werten die Möglichkeit ein gutes Ergebnis zu erhalten. Man hat zwar einen höheren Rechenaufwand als bei einer direkten Ausgleichsrechnung, allerdings dafür eine Chance auf deutlich besserte Ergebnisse.

### 5.2.2. Ablauf der Ellipsenberechnung

Als Basis der Berechnung dienen die Kurvenstücke die aus der Unterteilung hervorgehen. Anders als zuvor werden diese nicht zuerst zu elliptischen Bögen und Bogenpaaren zusammengeführt, um die daraus resultierenden Probleme zu vermeiden. Die Zusammenführung der Kurven zu einer gemeinsamen Ellipse erfolgt im Laufe der Berechnung, die sich die beschriebenen Vorteile von RANSAC zu nutze macht, um alle im Bild befindlichen Ellipsen aus den möglichen Kombinationen der vorliegenden Kurvenstücke zu erhalten.
Zwei Kurven bilden eine gültige Kombination, wenn sie die Orientierungsbedingung 4.21 der globalen Zusammenführung erfüllen.

$$\begin{aligned} \overline{P_{m,round(\frac{N_m}{2})}P_{n,round(\frac{N_n}{2})}} &> \overline{C_m P_{n,round(\frac{N_n}{2})}} \\ &and \\ \overline{P_{m,round(\frac{N_m}{2})}P_{n,round(\frac{N_n}{2})}} &> \overline{C_n P_{m,round(\frac{N_m}{2})}} \end{aligned} \tag{5.5}$$

Der genaue Ablauf ist in Pseudocode 5.1 beschrieben.

Die Berechnung einer Ausgleichsellipse $M$ aus den gewählten Punkten erfolgt gemäß der in Abschnitt 4.4 beschriebenen Ausgleichsrechnung.

Zur Bestimmung der prozentualen Übereinstimmung einer Kurve $K$ mit einer Ellipse $E$ wird der Abstand der in $K$ enthaltenen Punkte zu $E$ betrachtet. Um den Abstand eines Punktes

$P(x_p, y_p)$ zu einer Ellipse $E$ zu bestimmen wird zunächst die Gerade $G_{P,M_E}$ berechnet, die durch $P$ und den Mittelpunkt $M_E(x_0, y_0)$ von $E$ verläuft.

$$\begin{aligned} G_{P,M_E}(x) &= m * x + k \\ &= \frac{y_p - y_0}{x_p - x_0} * x + (-\frac{y_p - y_0}{x_p - x_0} * x_0 + y_0) \end{aligned} \tag{5.6}$$

Für $G_{P,M_E}$ werden die beiden Schnittpunkte $S_1(x_1, y_1)$ und $S_2(x_2, y_2)$ mit $E$ berechnet.

$$x_1 = \frac{(b^2 x_0 + a^2 m y_0 + ab\sqrt{a^2m^2 + b^2 - k^2 - 2kmx_0 + 2ky_0 - m^2x_0^2 + 2mx_0y_0 - y_0^2} - a^2km)}{(a^2m^2 + b^2)} \tag{5.7}$$

$$x_2 = \frac{(b^2 x_0 + a^2 m y_0 - ab\sqrt{a^2m^2 + b^2 - k^2 - 2kmx_0 + 2ky_0 - m^2x_0^2 + 2mx_0y_0 - y_0^2} - a^2km)}{(a^2m^2 + b^2)} \tag{5.8}$$

$$y_1 = m * x_1 + k \tag{5.9}$$

$$y_2 = m * x_2 + k \tag{5.10}$$

Zu beiden beiden Schnittpunkten wird der Abstand bestimmt.

$$d_{S_1,P} = \sqrt{(y_1 - y_p)^2 + (x_1 - x_p)^2} \tag{5.11}$$

$$d_{S_2,P} = \sqrt{(y_2 - y_p)^2 + (x_2 - x_p)^2} \tag{5.12}$$

Der kleinere Abstand definiert den Abstand $d_{P,E}$ des Punktes $P$ zur Ellipse $E$.

$$d_{P,E} = \min\left(d_{S_1,P}, d_{S_2,P}\right) \tag{5.13}$$

Ein Punkt $P_i$ stimmt mit ein Ellipse $E$ überein, wenn für deren Abstand $d_{P_i,E}$ gilt:

$$d_{P_i,E} \leq TH_{Dist} \tag{5.14}$$

Die prozentuale Übereinstimmung $fitting_K$ von $K$ mit $E$ ergibt sich aus:

$$fitting_K = \frac{\text{Anzahl übereinstimmender Punkte}}{\text{Anzahl aller Punkte}} \tag{5.15}$$

**Algorithm 5.1** Ablauf der Ellipsenberechnung

```
1:  ellipses;                                          ▷ speichert die berechneten Ellipsen
2:  for all Kurvenstück S do
3:      wähle nicht zugeordnetes Kurvenstücke K;
4:      if K erfüllt Orientierungsbedingung zu S then
5:          validSet=false;
6:          tries=0;                    ▷ Anzahl der Versuche zur consensus set Bestimmung
7:          while !validSet & tries < T_max do
8:              S_1 ← 6 zufällige Punkte aus S und K;
9:              M ← berechne Ellipse aus S_1;
10:             S_1^* ← bestimme consensus set aus S und K zu M;
11:             fittingS ← berechne prozentuale Übereinstimmung von S zu M;
12:             fittingK ← berechne prozentuale Übereinstimmung von K zu M;
13:             if fittingS ≥ fit_part & fittingK ≥ 30% then
14:                 validSet ← true;
15:                 if fittingS ≥ fit_full then
16:                     markiere S als zugeordnet;
17:                 end if
18:                 if fittingK ≥ fit_full then
19:                     markiere K als zugeordnet;
20:                 end if
21:             else
22:                 tries++;
23:             end if
24:         end while
25:         if validSet then                          ▷ suche weitere passende Kurven
26:             for all nicht zugeordnete Kurvenstücke U do
27:                 if U erfüllt Orientierungsbedingung zu S oder K then
28:                     S_1^* ← S_1^*+ übereinstimmende Punkte aus U;
29:                     fittingU ← berechne prozentuale Übereinstimmung von U zu M;
30:                     if fittingU ≥ fit_part then
31:                         markiere U als zugeordnet;
32:                     end if
33:                 end if
34:             end for
35:             M ←berechne Ellipse aus S_1^*;
36:             ellipses ← füge m hinzu;
37:         end if
38:
```

Für alle Kurven, die nach dem Ablauf noch nicht zugeordnet wurden, wird jeweils eine einzelne Ausgleichsellipse bestimmt.

## 5.3. Bewertung gefundener Ellipsen

Für die Bilder in diesem Abschnitt wurde die optimierte Ellipsendetektion angewendet. Die ermittelten Ellipsen sind in Weiß über das Quellbild gezeichnet, um die Übereinstimmung mit den im Bild befindlichen elliptischen Objekten leichter erkennen zu können. Ein direkter Vergleich mit dem Ergebnis des ursprünglichen Verfahrens aus Kapitel 4 war an dieser Stelle leider nicht möglich, da mangels ausreichender Zeit eine vollständige Nachimplementierung des Ablaufs nicht umgesetzt werden konnte.

Von Bild zu Bild nimmt die Hintergrundskomplexität zu, wodurch das Detektionsergebnis aber kaum beeinflusst wird. Für die Bilder der Abbildungen 5.5 - 5.7 werden fast alle zu erkennenden elliptischen Objekte auch von einer Ausgleichsellipse beschrieben. Vereinzelt lassen sich platte, langgezogene Ellipsen erkennen, die keiner Struktur im Bild entsprechen. Diese resultieren aus der Kombination mehrerer kleiner Kurven, die nicht Teil einer gemeinsamen elliptischen Struktur sind, aber aufgrund ihrer geringen Pixelanzahl und ihrer Lage gerade noch zu 30% mit der berechneten Ellipse übereinstimmen.
Für ein Bild mit sehr komplexen Hintergrund wie in Abb. 5.10 ergeben sich neben einigen erwarteten Ellipsen noch sehr viele andere. Durch die hohe Anzahl an Kurvenstücken und die daraus resultierenden Kombinationsmöglichkeiten ergeben sich auch aus dem Hintergrund entstammende Kurven zahlreiche mögliche Ergebnisse. Mit der Erwartungshaltung nur die Batterien beschreibende Ellipsen zu erhalten, denkt man im ersten Moment an eine Fehldetektion. Jedoch ist das klare Ziel des Algorithmus, alle mögliche sich Bild befindlichen Ellipsen zu ermitteln. Während der Berechnung kann somit keine Entscheidung getroffen werden, ob ein Ergebnis einem komplexen Hintergrund entstammt und somit verworfen werden sollte. Die Entscheidung über die Sinnhaftigkeit einer Ellipse muss auf einer höheren Abstraktionsstufe getroffen werden und ist somit anwendungsfallspezifisch definiert.

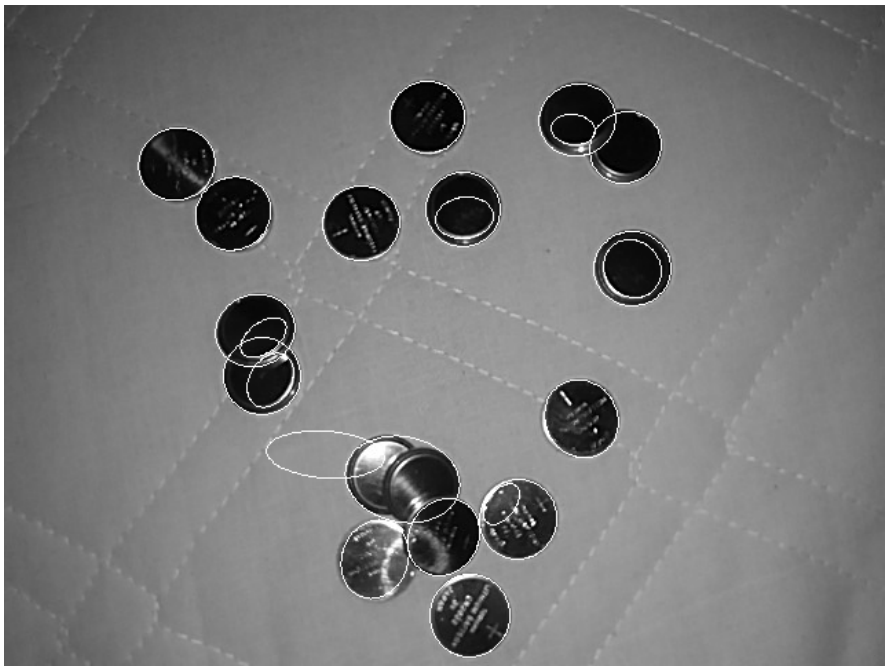


Abbildung 5.7.: $fit_{part} = 30\%$, $fit_{full} = 80\%$, Anzahl Ellipsen: 30


Abbildung 5.8.: $fit_{part} = 30\%$, $fit_{full} = 80\%$, Anzahl Ellipsen: 33

Abbildung 5.9.: $fit_{part} = 30\%$, $fit_{full} = 80\%$, Anzahl Ellipsen: 16

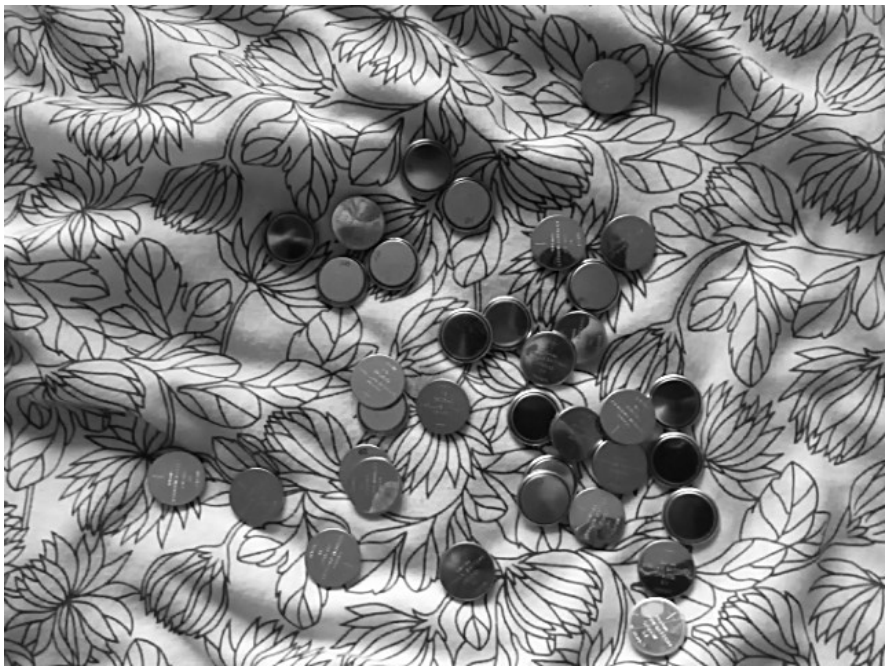


Abbildung 5.10.: Sehr Komplexes Quellbild

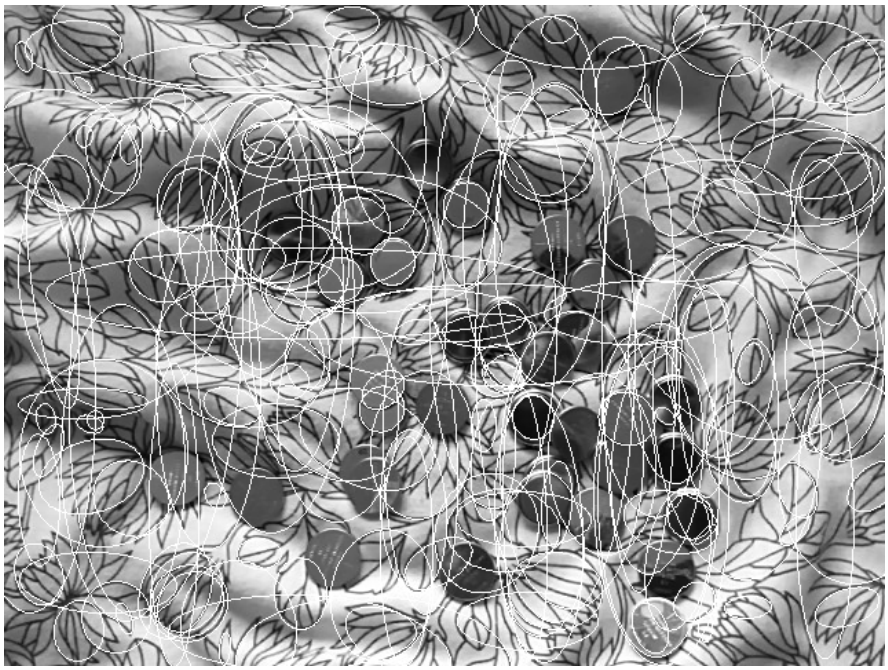


Abbildung 5.11.: $fit_{part} = 30\%$, $fit_{full} = 80\%$, Anzahl Ellipsen: 178

# 6. Implementierung

Dieses Kapitel enthält eine Überblick zur Implementierung und umfasst einen kleinen Abschnitt zu den verwendeten Bibliotheken (6.1) und eine Übersicht zu den relevanten Funktionalitäten der realisierten Klassen in Abschnitt 6.2.

## 6.1. Erläuterung zu verwendeten Bibliotheken

Für den effizienten und einfachen Zugriff auf Bilder nutze ich OpenCV 2.4.8 (its14). OpenCV ist eine frei verfügbare Open Source Bibliothek, die für aktuelle Programmiersprachen eine Vielzahl an Bildverarbeitungsalgortihmen anbietet. Unter anderem bietet sie auch eine Implementierung des Canny-Algorithmus, die ich in meinem Programm verwende.

Um die für die Ausgleichsrechnung benötigte Matrizenrechnung nicht selbst umsetzen zu müssen, verwende ich die C++ Template Bibliothek Eigen in der Version 3.2.1 (GJ+10), die eine ebenso effiziente wie intuitive Verwendbarkeit ermöglicht.

## 6.2. Überblick zu verwendeten Klassen

Dieser Abschnitt enthält eine kurze Übersicht zu den implementierten Klassen und den einstellbaren Parametern, die die beschriebenen Bearbeitungsschritte realisierten. Der gesamte Algorithmus wurde in C++ umgesetzt.

Klasse **Point**:

> Repräsentiert einen Bildpunkt als zweidimensionalen Vektor, der Addition, Subtraktion, Multiplikation und Normberechnung untersützt

Klasse **EdgeSegment**:

> Repräsentiert aus die dem Kantenbild ermittelten Kantensegmente und beinhaltet Methoden, die die zur Kantensegmentierung nötigen Schritte durchzuführen

**wichtige Methoden**:

- **... lineSegmentation(...)**:
  führt die in Abschnitt 4.2.3 beschrieben Approximation durch Linienstücke durch
- **... curveSegmentation(...)**:
  führt die Unterteilung in Kurvenstücke gemäß Abschnitt 4.2.3 durch
- **... curveSegmentationImproved(...)**:
  führt die optimierte Unterteilung in Kurvenstücke gemäß Abschnitt 5.1 durch

Klasse **Ellipse**:

Repräsentiert eine Ellipse mit den bekannten aus Kapitel 2.1 bekannten Parametern.
**wichtige Methoden**:

- **double calcDistance(Point* P**:
  berechnet den Abstand eines Punktes von der Ellipse gemäß der in Abschnitt 5.2.2 beschriebenen Methode
- **Ellipse* calcEllipse(std::vector<Point*> *points)**:
  berechnet eine Ausgleichsellipse für die angegebenen Punkte gemäß der ellipsenspezifischen Ausgleichrechnung in Abschnitt 4.4.1

Um die einzelnen Bearbeitunsschritte zu kapseln, steht für jeden eine Methode in *PreProcessing.cpp* zur Verfügung

- **... edgeDetection(...)**:
  Kantendetektion gemäß des in openCV implementierten Canny-Operators
- **... edgeLinking(...)**:
  Bildung der Kantensegmente aus einem Kantenbild gemäß Abschnitt 4.2.1
- **... curveSegmentation(...)**:
  Unterteilung der Kantensegmente in Kurvenstücke (wahlweise nach Abschnitt 4.2.3 oder 5.1)
- **... fitEllipses(...)**:
  Berechnet aus den vorliegenden Kurvenstücken die möglichen Ausgleichsellipsen gemäß Abschnitt 5.2.2

Die einzelnen Schritte lassen sich durch einige Parameter in *Defines.hpp* einstellen. Die Relevanten sind in folgender Tabelle dargestellt:

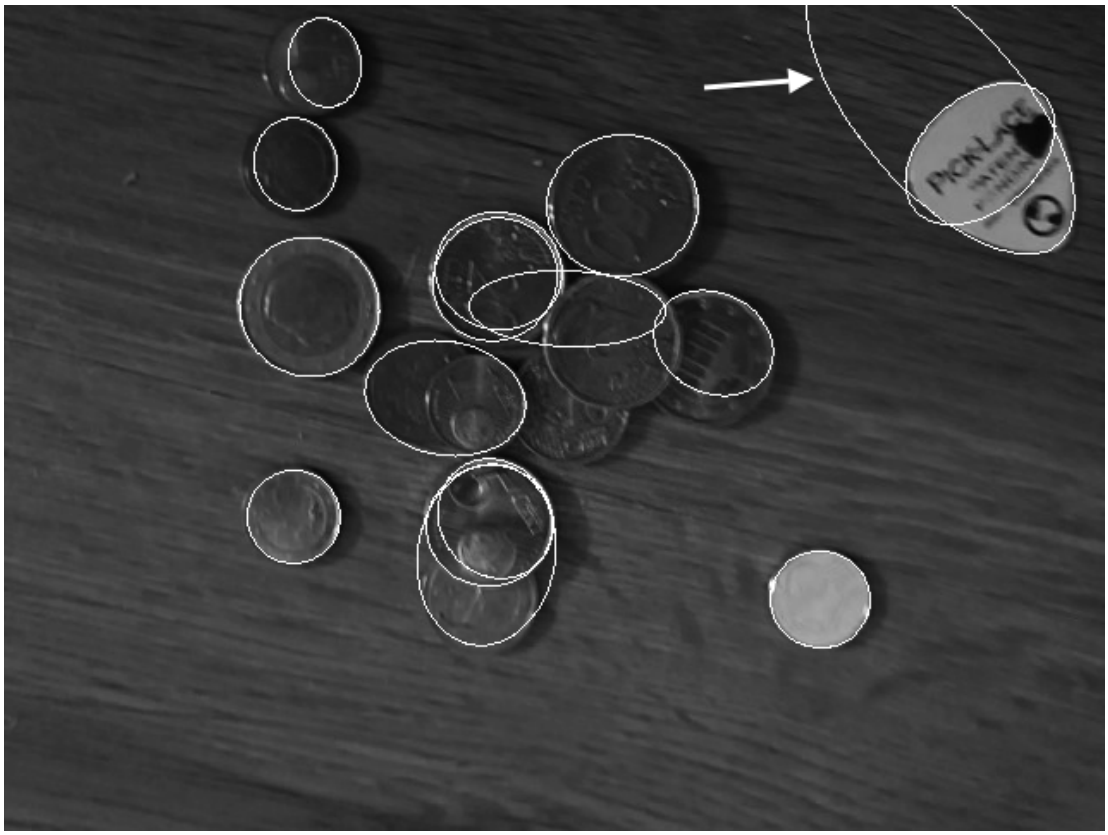
Tabelle 6.1.: Übersicht der Parameter

| **Parameter** | **Bedeutung** |
|---|---|
| TH_U | oberer Grenzwert für die Gradientenstärke des Canny Algorithmus |
| TH_L | unterer Grenzwert für die Gradientenstärke des Canny Algorithmus |
| MIN_LENGTH | minimale Länge eines Kantensegments für die weitere Berücksichtigung |
| D_TOL | maximaler Abstand eines Pixels zu einem Linienstück bei der Approximation |
| LTH | Faktor für die Längenbedingung zweier benachbarter Linienstücke |
| TH | maximale Differenz der aufeinander folgenden Winkel in einem Kurvenstück (in rad) |
| TH_DIST | maximaler Abstand bei dem ein Punkt mit einer Ellipse übereinstimmt |
| TH_FULLFIT | prozentualer Grenzwert für die komplette Übereinstimmung einer Kurve mit einer Ellipse |
| TH_PARTIALFIT | prozentualer Grenzwert für die teilweise Übereinstimmung einer Kurve mit einer Ellipse |

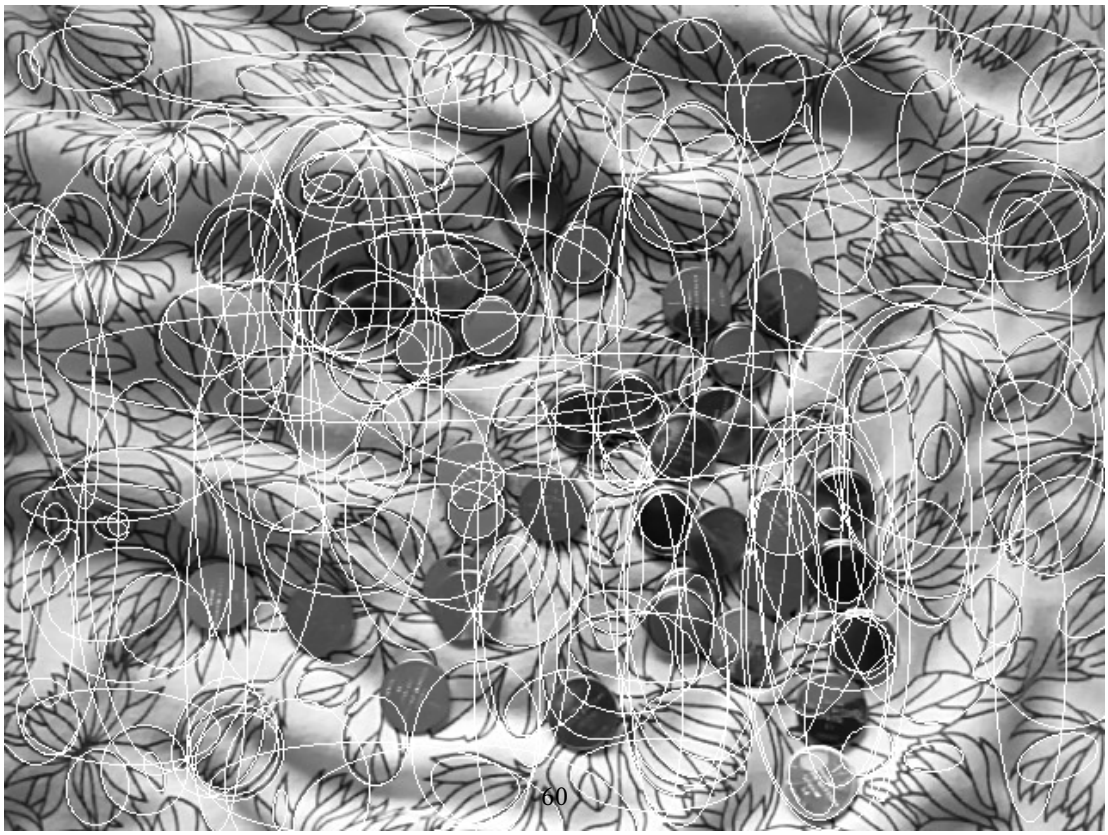
# 7. Ausblick

Mit dem in dieser Arbeit entwickelten Verfahren liegt bereits im aktuellen Entwicklungsstand eine recht robuste Ellipsendetektion vor. Momentan werden alle aus den Kombinationen der vorliegenden Kurvenstücke hervorgehenden Ellipsen als korrekt akzeptiert. Dadurch liegen gerade in komplexen Bildern sehr viele Ellipsen vor, die nicht den im Bild abgebildeten Objekten entsprechen. Für eine weitere Verbesserung des Algorithmus macht es somit Sinn, eine grobe Vorsortierung der Ergebnisse vorzunehmen. Dazu möchte ich zwei realisierbare Kriterien zur Erkennung unsinniger Ellipsen skizzieren.

Für die erste Möglichkeit geht man davon aus, das ein im Bild befindliches Objekt durchgehend ungefähr den selben Grauwert aufweist. Unter dieser Annahme sollten der von einer Ausgleichsellipse eingeschlossene Bildbereich unmittelbar an deren Kante für den gesamten Umlauf keine großen Veränderungen in den Grauwerten aufweisen. Die in Abbildung 7.1a durch den Pfeil markierte Ellipse ließe sich anhand dieser Untersuchung aussortieren. Die Kanten dieser Ellipse haben für den halben Umlauf einen sehr hellen und für den Rest einen dunklen Bildbereich und entspricht dadurch gemäß des Kriteriums kein Objekt im Bild.
Auch aus dem Bild in Abb. 7.1b würden eine Menge Ellipsen als ungültig erkannt werden.



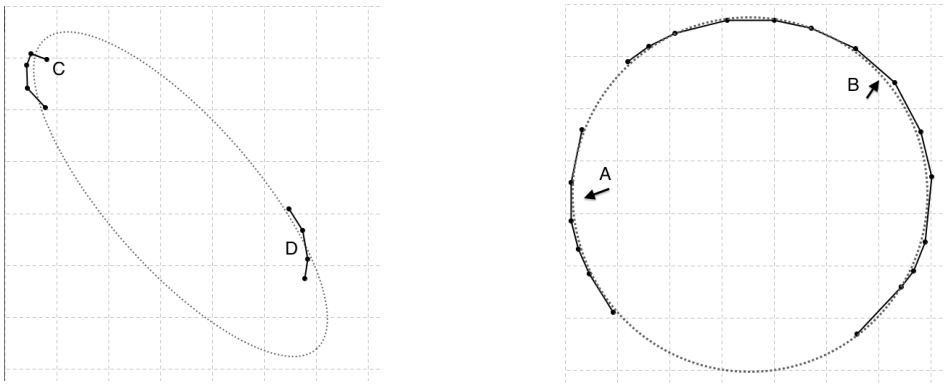
(a)



(b)

Abbildung 7.1.: Beispiele mit falsch detektierten Ellipsen

Die zwei Möglichkeit bestünde darin, anstatt wie bisher nur zu prüfen, ob Kurvenstücke mit einer Ausgleichsellipse übereinstimmen, ebenso zu untersuchen, wie die Ellipse den Kurvenstücken entspricht. Dazu könnte man ermitteln, wie viel des Ellipsenumfangs durch die beteiligten Kurven abgedeckt wird. Je Abdeckung vorliegt, umso höher kann die Qualität der Ellipse angesehen werden. Ergibt sich eine Ellipse wie in Abbildung 7.2a, weist diese eine deutlich geringere Übereinstimmung mit ihren Kurvenstücken $C$ und $D$ auf, als die Ellipse in Abbildung 7.2b, für die $A$ und $B$ fast die gesamte Ellipse beschreiben. Je mach Übereinstimmung könnte man Qualitätsstufen definieren und nur Ergebnisse ab einer festlegbaren Mindeststufe akzeptieren.




(a) (b)

Abbildung 7.2.: Übereinstimmung zwischen Ellipsen und Kurvenstücken

Auf jeden Fall ist es notwendig, den Algorithmus für spezielle Anwendungsfälle anpassbar zu machen. In den seltensten Fällen besteht das Ziel darin, alle möglichen Ellipsen aus einem Bild zu extrahieren. Viel häufiger werden bestimmte elliptische Objekte gesucht. Dadurch lassen sich die möglichen Ellipsenparameter einschränken. So hätte man die Möglichkeit, bereits einige Kurvenstücken auszusortieren, die auf keinen Fall zu den gesuchten Ellipsen führen können.
Würde man für die Kurvenstücke (Abb. 7.3) des Bildes in Abbildung 7.1b gezielt nach den Batterien suchen, könnte man die Suche auf Ellipsen mit ungefähr gleichgroßen Halbachsen $a$ und $b$ beschränken. Dadurch würden eine ganze Menge langgezogene Kurven nicht mehr für die Berechnung in Frage kommen.

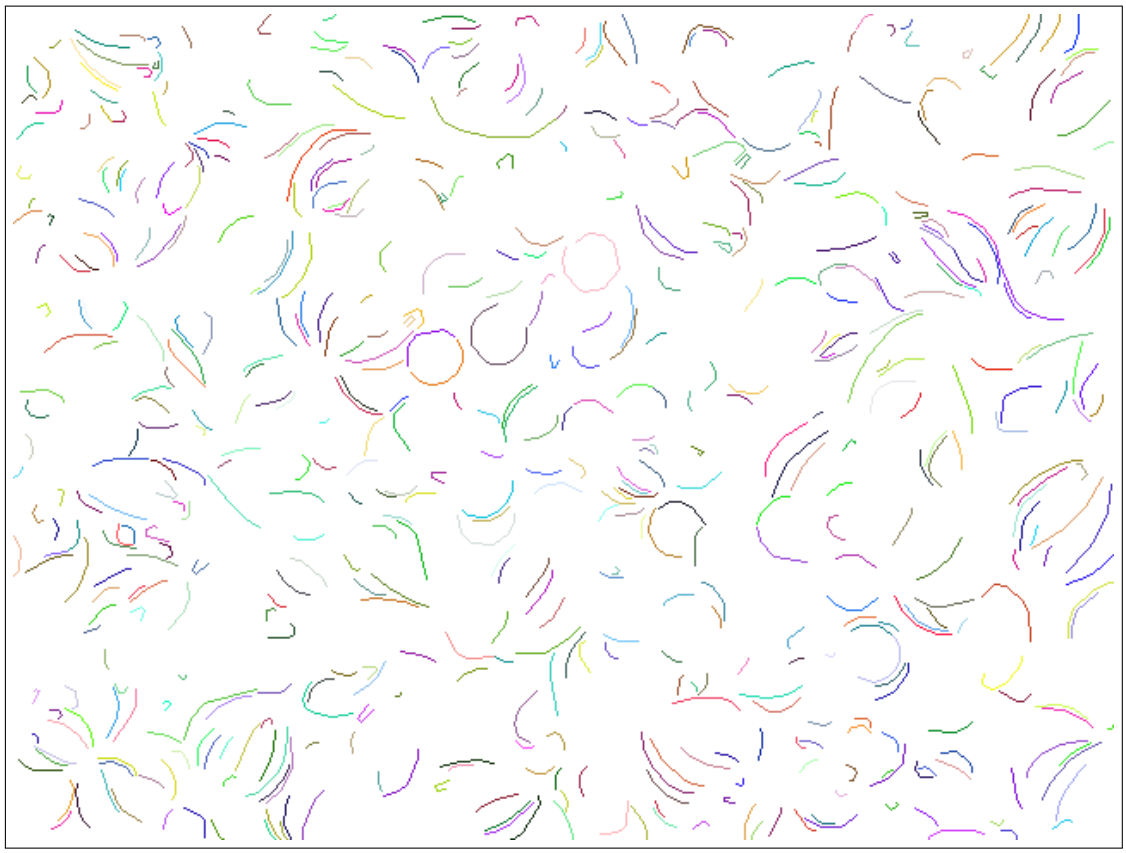


Abbildung 7.3.: Kurvenstücken zum Bild aus Abb. 7.1b

Die dargestellten Möglichkeiten lassen sich mit nicht allzu großem Aufwand vertiefen und auf die Implementierung übertragen. Insbesondere die Berücksichtigung von festlegbaren Einschränkungen für die gesuchten Ellipsen würde es ermöglichen den Algorithmus effektiv für unterschiedliche Problemfälle einzusetzen.
Der Aufwand für weitere Optimierungen ist somit als lohnend anzusehen.

# A. Inhalt der CD

Die beiliegende CD enthält unter anderem:

- **Quellcode der Implementierung**
- **Digitale Version dieser Arbeit**

# Literaturverzeichnis


[Can86] Canny, John: A Computational Approach to Edge Detection. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8 (1986), Nr. 6, 679-698. http://doi.ieeecomputersociety.org/10.1109/TPAMI.1986.4767851, Abruf: 13. Juli 2014. – ISSN 0162–8828

[CGZPC14] Cuevas, Erik ; González, Mauricio ; Zaldívar, Daniel ; Pérez-Cisneros, Marco: Multi-ellipses detection on images inspired by collective animal behavior. In: *Neural Computing and Applications* 24 (2014), Nr. 5, 1019-1033. http://dx.doi.org/10.1007/s00521-012-1332-4, Abruf: 13. Juli 2014. – ISSN 0941–0643

[CLER07] Chia, Alex Y. ; Leung, Maylor. K. H. ; End, How-Lung ; Rahardja, Susanto: Ellipse Detection with Hough Transform in One Dimensional Parametric Space. In: *IEEE International Conference on Image Processing, 2007 : ICIP 2007* Bd. 5, Piscataway, NJ : IEEE, 2007. – ISBN 978–1–4244–1437–6, V-333-V-336

[FB81] Fischler, Martin A. ; Bolles, Robert C.: Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartography. In: *Communication of the ACM* 24 (1981), Nr. 6, 381–395. http://dl.acm.org/citation.cfm?id=358692&CFID=345826157&CFTOKEN=73723239, Abruf: 26. Mai 2014

[FH98] Flusser, Jan ; Halir, Radim: Numerically stable direct least squares fitting of ellipses. In: *6th Int. Conf. in Central Europe on Computer Graphics and Visualization* (1998), 125-132. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.1.7559, Abruf: 13. Juli 2014

[FPF99] Fitzgibbon, A.W. ; Pilu, M. ; Fisher, R.B.: Direct least squares fitting of ellipses. In: *IEEE transactions on pattern analysis and machine intelligence* Bd. 21, New York, NY : IEEE, 1999. – ISSN 0162–8828, 476-480

[GJ+10] Guennebaud, Gaël ; Jacob, Benoît u. a.: *Eigen v3.* http://eigen.tuxfamily.org, 2010

[Gut05] Gut, Jutta: *Kegelschnitte in allgemeiner Lage*. http://members.chello.at/gut.jutta.gerhard/kegelschnitte9.htm. Version: 2005

[GZ97] Guil, N ; Zapata, E.L: Lower order circle and ellipse Hough transform. In: *Pattern Recognition* 30 (1997), Nr. 10, 1729 - 1744. http://dx.doi.org/10.1016/S0031-3203(96)00191-4, Abruf: 13. Juli 2014. – ISSN 0031–3203

[HSCP87] Holt, Christopher M. ; Stewart, Alan ; Clint, Maurice ; Perrott, Ronald H.: An Improved Parallel Thinning Algorithm. In: *Communications of the ACM* 30 (1987), Februar, Nr. 2, 156–160. http://doi.acm.org/10.1145/12527.12531, Abruf: 13. Juli 2014. – ISSN 0001–0782

[its14] itseez: *Open Source Computer Vision Homepage*. http://opencv.org. Version: 2014, Abruf: 10. Juli 2014

[JRW06] Jianfei, Mao ; Rong, Xiong ; Weilong, Ding: A Compound and Robust Algorithm for Ellipse Detection. In: *16th International Conference on Artificial Reality and Telexistence workshops, 2006 : ICAT '06*, Los Alamitos, Calif. : IEEE, 2006. – ISBN 0–7695–2754–X, 381-386

[Kov06] Kovesi, Peter: *MATLAB and Octave Functions for Computer Vision and Image Processing*. http://www.csse.uwa.edu.au/~pk/research/matlabfns/LineSegments/lineseg.m. Version: 2006, Abruf: 13. Juli 2014

[Kov13] Kovesi, Peter: *MATLAB and Octave Functions for Computer Vision and Image Processing*. Version: 2013. http://www.csse.uwa.edu.au/~pk/research/matlabfns/LineSegments/edgelink.m, Abruf: 13. Juli 2014

[MHZS07] Mai, F. ; Hung, Y.S. ; Zhong, H. ; Sze, W. F.: A Hierarchical Approach for Fast and Robust Ellipse Extraction. In: *IEEE International Conference on Image Processing, 2007 : ICIP 2007* Bd. 5, Piscataway, NJ : IEEE, 2007. – ISBN 978–1–4244–1437–6, V-345-V-348

[NAW09] Nguyen, Thanh M. ; Ahuja, Siddhant ; Wu, Q.M J.: A real-time ellipse detection based on edge grouping. In: *IEEE International Conference on Systems, Man and Cybernetics, 2009*. Piscataway, NJ : IEEE, 2009. – ISBN 978–1–4244–2793–2, S. 3280–3286

[NO94] Neusius, Christian ; Olszewski, Jan: A Noniterative Thinning Algorithm. In: *ACM transactions on mathematical software* 20 (1994), Nr. 1, 5–20. http://doi.acm.org/10.1145/174603.174604, Abruf: 13. Juli 2014. – ISSN 0098–3500

[Pap08] Papula, Lothar: *Mathematik für Ingenieure und Naturwissenschaftler Band 1.* 11.Aufl. Wiesbaden : Vieweg + Teubner Verlag, 2008 http://dx.doi.org/10.1007/978-3-8348-9220-1. – ISBN 978–3–8348–9220–1

[Pap09] Papula, Lothar: *Mathematische Formelsammlung für Ingenieure und Naturwissenschaftler.* Wiesbaden : Vieweg+Teubner, 2009 http://dx.doi.org/10.1007/978-3-8348-9598-1. – ISBN 978–3–8348–9598–1

[SK12] Shamsi, Mahboubeh ; Kenari, Abdolreza R.: Iris boundary detection using an ellipse integro differential method. In: *2nd International eConference on Computer and Knowledge Engineering (ICCKE), 2012.* Piscataway, NJ : IEEE, 2012. – ISBN 978–1–4673–4475–3, S. 1–5

[Tön05] Tönnies, Klaus D.: *Grundlagen der Bildverarbeitung.* München : Pearson Studium, 2005. – ISBN 3–8273–7155–4

[Wik14] Wikipedia: *Ellipse — Wikipedia, Die freie Enzyklopädie.* http://de.wikipedia.org/w/index.php?title=Ellipse&oldid=131791099. Version: 2014, Abruf: 13. Juli 2014

[YKG05] Yao, Jie ; Kharma, Nawwaf ; Grogono, Peter: A multi-population genetic algorithm for robust and fast ellipse detection. In: *Pattern Analysis and Applications* 8 (2005), Nr. 1-2, 149-162. http://dx.doi.org/10.1007/s10044-005-0252-7, Abruf: 13. Juli 2014. – ISSN 1433–7541

*Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.*

Hamburg, 15.07.2014 Manuel Bösch