Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik VI
Prof. Dr.-Ing. Hermann Ney

Seminar Computer Vision im WS 2004/2005

Segmentierung durch Modellanpassung

Carsten Dolch

Matrikelnummer 229 444

14.01.2005

Betreuer: Daniel Keysers

Inhaltsverzeichnis

1 Einleitung

Eine häufige Problemstellung in der Computer Vision besteht darin, dass man Bilder in bestimmte Bereiche segmentieren möchte, wobei bekannt ist, welche Form diese Bereiche in etwa haben. So kann es zum Beispiel sein, dass man weiß, dass es in einer Aufnahme einen Bereich gibt, der ein Buch darstellt. Ein solcher Bereich wird in etwa eine rechteckige Form haben, wobei die Ausrichtung dieses Rechtecks unerheblich ist. Bei der Segmentierung durch Modellanpassung wird dieses Rechteck nun erst als abstraktes mathematisches Modell beschrieben, dann an die entsprechenden Datenpunkte angepasst und schließlich wird das Bild in die entsprechenden Teile aufgeteilt. Etwas allgemeiner kann man das Verfahren wie folgt beschreiben:

mathematisches Modell (Rechteck, Ellipse, Kurve, ...)

$\Downarrow$

Anpassung an die gemessenen Bildpunkte

$\Downarrow$

Segmentierung des Bildes in die entsprechenden Bereiche

Im Folgenden sollen verschiedene Methoden vorgestellt werden, um Bilder nach dieser Vorgehensweise zu verarbeiten. Hierbei werden die Verfahren nur für recht einfache geometrische Modelle, wie Linien und Kurven, beschrieben. Jedoch lassen sie sich in der Regel auch auf komplexere Formen erweitern.

2 Linienanpassung

2.1 Die Hough-Transformation

2.1.1 Das Prinzip der Hough-Transformation

Eine der einfachsten, aber auch effizientesten Techniken um Linien in Punktmengen zu finden ist die in [3] vorgestellte Hough-Transformation. Sie basiert auf der Tatsache, dass man jede Linie durch die folgende Gleichung beschreiben kann:

$$x \cos\theta + y \sin\theta - r = 0, \theta \in [0, 2\pi].$$

Das heißt, alle Punkte (x, y), welche diese Gleichung erfüllen, liegen auf der entsprechenden Gerade. Dabei beschreibt r den Abstand der Linie zum Ursprung des Koordinatensystems und θ den Winkel zwischen diesem Abstandsvektor und der $x-$Achse.
Das bedeutet aber auch, dass man jede Linie durch die Parameter (θ, r) vollständig beschreiben kann. Geht man andererseits davon aus, dass man einen Punkt (x_0, y_0) betrachtet, so bildet die Menge aller Linien durch diesen Punkt einen sogenannten *Linienraum*.
Löst man nun diese Geichung nach r auf, so ergibt sich für jeden Punkt (x_0, y_0) eine Kurve in einem Koordinatensystem mit den Dimensionen θ und r. Wobei die Punkte auf der jeweiligen Kurve die Menge aller Linien beschreibt, die durch (x_0, y_0) gehen. Und es ist klar, dass nun im Schnittpunkt der Kurven, die zu jedem Datenpunkt (x_n, y_n) bestimmt werden können, die Parameter θ und r abgelesen werden können, welche die Linie beschreiben, die durch alle (x_n, y_n) führt. Die obige Gleichung nach r aufgelöst lautet dann:

$$r = -x_0 \cos\theta - y_0 \sin\theta \tag{1}$$

Da das zu untersuchende Bild nur eine bestimmte Größe hat, existiert außerdem ein R, so dass man nicht an $r > R$ interessiert ist, da diese Linien außerhalb des Bildes liegen

würden. Um die Menge von Linien, die durch die verschiedenen Punkte (x_n, y_n) gehen, zu diskretisieren wird nun ein so genanntes Akumulatorarray mit den Dimensionen (θ, r) festgelegt. Man kann es sich wie ein Gitter vorstellen mit den Feldern des Arrays als Körbe. Für jeden Punkt (x_n, y_n) wird dann die Menge aller durch ihn gehenden Linien bestimmt, wobei jede Linie definiert ist durch das Paar (θ, r). Dann wird in dem entsprechenden Feld (θ, r) des Arrays ein Zähler um eins erhöht. Oder etwas anschaulicher: es wird eine Stimme in den entsprechenden Korb gelegt. Das ganze ist dann mit einer Art Wahlprozess vergleichbar. Der Korb mit den meisten Stimmen bestimmt dann durch seine Position (θ, r) in dem Gitter die Linie, welche am besten an die Datenpunkte angepasst ist. Zur besseren Illustration ist dieser Wahlprozess in der Abbildung 1 grafisch dargestellt. $r \in [0; 1,55]$ ist in vertikaler Richtung angetragen, θ in horizontaler Richtung. Es gibt in jeder Dimension 200 Felder. Die ideale Linie wird durch $x \cos\theta + y \sin\theta - r = 0$ mit $\theta = \frac{\pi}{2} = 45°$ und $r = \sqrt{0,5}$ dargestellt. Für jeden Datenpunkt in den linken Bildern ergibt sich eine durch die Gleichung (1) definierte Kurve der Parameter (θ, r) aller durch diesen Punkt gehenden Linien. Man erkennt in dem oberen rechten Bild, dass die Linien sehr regelmäßig sind und sich in genau einem Punkt alle Kurven schneiden. Dies ist klar, da die untersuchten Datenpunkte genau auf einer Linie liegen. Der Schnittpunkt aller Kurven bestimmt genau die Paramter der Geraden durch die Punkte, da hier die meisten Stimmen zusammengekommen sind. In den unteren beiden Bildern sieht man nun Datenpunkte, die nicht mehr genau auf einer Linie liegen und entsprechend schneiden sich die Linien auch nicht mehr alle in einem Punkt. Aber man erkennt eine Region in der sich die Schnittpunkte häufen. Innerhalb dieser Region wird sich eine Position mit den meisten Stimmen finden, welche eine bezüglich der gegebenen Datenpunkte optimale Linie bestimmt, da diese Linie durch die größmögliche Anzahl aller Datenpunkte führt.

2.1.2 Probleme in der Praxis

In dieser Form ergeben sich in der Praxis allerdings häufig noch folgende Probleme:

- **Quantisierungsfehler:** Es stellt sich häufig als schwierig heraus, eine adäquate Gittergröße zu wählen. Wird das Raster zu grob gewählt, so kann dies dazu führen, dass sich in einem Korb die Stimmmen vieler recht verschiedener Linien sammeln. Wird es hingegen zu fein gewählt, so kann es passieren, dass sich in keinem Korb signifikant mehr Stimmen ansammeln als in anderen, weil auch für sehr ähnliche Linien die Stimmen nicht im gleichen Korb landen.

Lösung: in der Praxis wird das Gitter gewöhnlich mittels „Trial and Error “angepasst. Das heißt, man wendet die Hough-Transformation für ein Bild an, dass die Klasse von Bildern repräsentiert, welche untersucht werden sollen. Dann wird die Gittergröße so lange angepasst, bis das Ergebnis zufriedenstellend ist. Außerdem kann es vor allem im Falle der zu klein gewählten Gittergröße sinnvoll sein, dass man jedem Nachbarn eines Feldes auch eine Stimme gibt, wenn man dem Feld eine Stimme gibt. Auf diese Weise bekommen auch alle nahe bei dieser Linie liegenden Linien eine Stimme.

- **Probleme mit Rauschen:** Der Reiz der Hough-Transformation liegt unter anderem auch darin, dass dieses Verfahren relativ weit voneinander entfernt liegende Punkte als potenzielle Teile einer Linie betrachtet, da die durch die beiden Punkte gehende

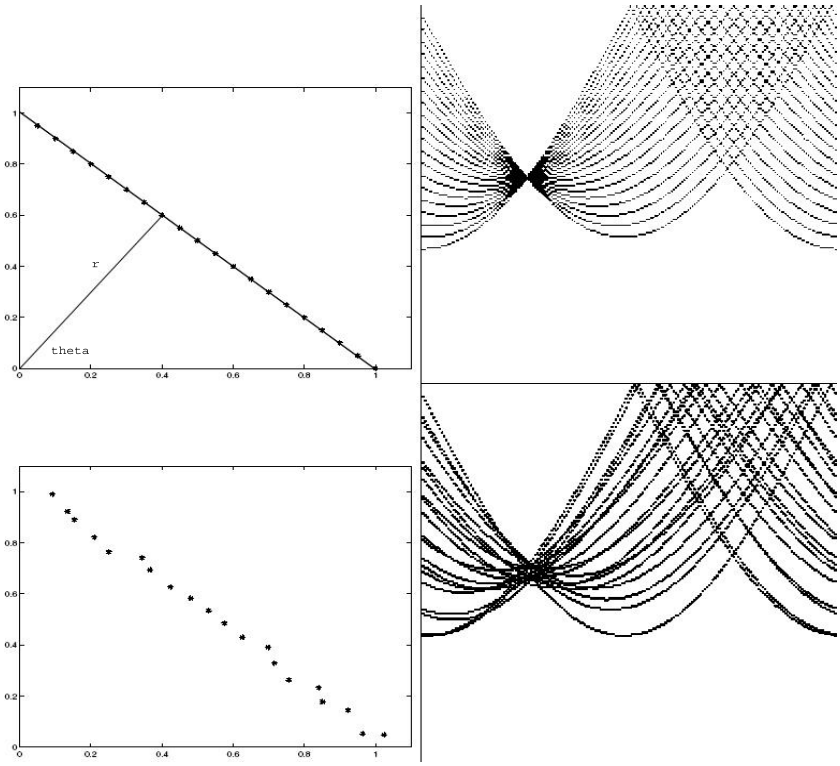

Abbildung 1: Beispiel zur Hough-Transformation (aus [1]).

Linie die gleichen Parameter θ und r hat und sie deswegen zwei Stimmen im entsprechenden Feld des Akumulatroarrays bekommt. Dieser Vorteil beinhaltet aber auch einen Nachteil: in einer gleichverteilten, genügend großen Menge von Bildpunkten, wie es zum Beispiel Texturen sind, findet man eine große Anzahl von Phantomlinien. Dies kann dazu führen, dass in einem Bild in Texturen Linien gefunden werden, welche mehr Stimmen bekommen, als die Kanten dieser Texturen, welche ja die Linien darstellen, nach denen eigentlich gesucht wurde.

Lösung: man sollte die Anzahl irrelevanter Bildpunkte so stark wie möglich reduzieren. Dies kann zum Beispiel dadurch geschehen, dass man den Kantendetektor so einstellt, dass Texturen aus dem Bild entfernt werden und nur scharf gezeichnete Kanten im Bild bleiben.

Trotz all dieser Nachteile kommt die Hough-Transformation bei bestimmten Problemen häufig zum Einsatz. Zumal sie auch auf komplexere Modelle als Linien angewendet werden kann. So kann die Hough-Transformation auch auf Objekte wie Kreise oder Ellipse angesetzt werden. Die wird zum Beispiel bei einigen Verfahren zur Iriserkennung gemacht.

2.2 Distanzbasierte Verfahren

Dieses und das folgende Verfahren gehören zu anderen Klasse von Methoden. Es wird nun angenommen, dass alle Datenpunkte zu genau einer Linie gehören. Das Ziel ist nun die Parameter der Linie so an die gegebenen Datenpunkte anzupassen, dass der Abstand zwischen den Datenpunkten und der Linie minimal wird.

Im weiteren wird zur besseren Lesbarkeit die Schreibweise $\overline{u}$ eingeführt, um den Mittelwert über alle Werte u_n zu beschreiben. Es gilt:

$$\overline{u} = \frac{1}{N} \sum_{n=1}^{N} u_n.$$

2.2.1 Die Methode der kleinsten Quadrate (Least Squares)

Eines der ältesten Verfahren mit einer langen Tradition ist die Methode der kleinsten Quadrate (engl. Least Squares). Hierbei wird eine Linie in der Form $y = ax + b$ und die entsprechenden Datenpunkte aus dem Bild als Zweiertupel (x_n, y_n) repräsentiert. Nun ist man an einer Liniendarstellung interessiert, die einem bezüglich aller gemessenen x_n-Werte, den am nächsten zu dem entsprechenden y_n-Wert liegenden Wert vorhersagt. Bezogen auf die oben gewählte Linienrepräsentation ist dies gleichbedeutend mit der Wahl der Linienparameter so, dass die Linie den folgenden Ausdruck minimiert:

$$\sum_{n=1}^{N} (y_n - (ax_n + b))^2. \tag{2}$$

Durch Differenzierung sind die Paramter a und b einer solchen Linie als Lösung der folgenden Gleichung gegeben:

$$\begin{pmatrix} \overline{xy} \\ \overline{y} \end{pmatrix} = \begin{pmatrix} \overline{x^2} & \overline{x} \\ \overline{x} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}. \tag{3}$$

Berechnet man nun die inverse Matrix

$$\begin{pmatrix} \overline{x^2} & \overline{x} \\ \overline{x} & 1 \end{pmatrix}^{-1}$$

so kann man a und b bestimmen.
Allerdings ergeben sich mit diesem Verfahren unter Umständen auch Probleme. Eines ist, dass die Fehlerberechnung von dem gegebenen Koordinatenrahmen abhängt. Verändert sich die Ausrichtung der $y-$Koordinate so verändert sich auch die Größe der Fehler. Zudem bewirkt die Tatsache, dass die vertikalen Verschiebungen der Punkte als Fehler gemessen werden, dass vertikal ausgerichtete Linien zu relativ großen Fehlern führen. Es ist sogar so, dass komplett vertikale Linien gar nicht erkannt werden. Dieses Verfahren wird insbesondere dann benutzt, wenn der Messfehler der Datenpunkte x_n sehr klein gegenüber den y_n ist, bespielsweise bei Zeitreihen.

2.2.2 Die Methode der insgesamt kleinsten Quadrate (Total Least Squares)

Um dieses Problem zu beheben, besteht die Möglichkeit anstatt der vertikalen Distanz zwischen dem Punkt und der Linie die direkte Distanz zwischen dem Punkt und der Linie zu wählen. Hierzu wird die oben gewählte Darstellung einer Linie zu $\phi(x, y) = ax + by + c = 0$ verändert. Eine Linie wird nun durch ein Tupel (a, b, c) dargestellt, mit der Einschränkung, dass gelten muss $a^2 + b^2 = 1$. Diese Darstellung entspricht der Darstellung in der Hough-Transformation, nur dass hier die Variablen a, b und c heißen, und nicht $\sin\theta, \cos\theta$ und r. Man beachte hier, dass für ein $\lambda \neq 0$ die durch (a, b, c) repräsentierte Linie mit der durch $\lambda(a, b, c)$ dargestellten Linie identisch ist, was aber durch die Normierung

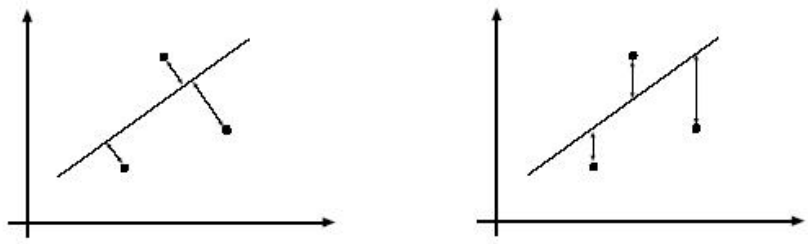

Abbildung 2: Links: Total Least Squares, rechts: Least Squares (aus [1]).

$a^2+b^2=1$ berücksichtigt wurde. Zudem kann man beweisen, dass der senkrechte Abstand zwischen der Linie (a,b,c) und dem Punkt (u,v) berechnet werden kann durch:

$$dist(\phi(x,y),(y_n,x_n)) = |\phi(x_n,y_n)| = |ax_n+by_n+c| \text{ mit } a^2+b^2=1. \tag{4}$$

Das Ziel ist nun, analog zum vorherigen Verfahren, den quadratischen Abstand aller Punkte (x_n,y_n) zu der durch (a,b,c) definierten Graden zu minimieren. Dies schafft man durch die Minimierung des folgenden Ausdrucks:

$$\sum_{n=1}^{N}(ax_i+by_i+c)^2 \text{ mit } a^2+b^2=1. \tag{5}$$

Es existiert also ein Ausdruck der unter Beachtung einer Nebenbedingung minimiert werden soll. Ein solches Problem lässt sich mit Hilfe eines Lagrange-Multiplikators λ lösen. Für den hier vorliegenden Sachverhalt ergibt sich dann die folgende Gleichung:

$$\begin{pmatrix} \overline{x^2} & \overline{xy} & \overline{x} \\ \overline{xy} & \overline{y^2} & \overline{y} \\ \overline{x} & \overline{y} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \lambda \begin{pmatrix} 2a \\ 2b \\ 0 \end{pmatrix}. \tag{6}$$

Nun kann man c direkt durch Umformung bestimmen und dieses c dann in (6) resubstituieren. Mit $c=-a\overline{x}-b\overline{y}$ führt dies dann zu dem folgenden Eigenwert-Problem:

$$\begin{pmatrix} \overline{x^2}-\overline{x}\,\overline{x} & \overline{xy}-\overline{x}\,\overline{y} \\ \overline{xy}-\overline{x}\,\overline{y} & \overline{y^2}-\overline{y}\,\overline{y} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \mu \begin{pmatrix} a \\ b \end{pmatrix}. \tag{7}$$

Da dies ein 2D-Eigenwertproblem ist, können zwei Lösungen gefunden werden. Die beiden gefundenen Lösungen repräsentieren Vektoren, welche beide rechtwinklig zu der angepassten Linie stehen. Die Parameter eines dieser beiden Vektoren stellen die von uns gesuchte Lösung dar.
In Abbildung 2 ist noch einmal der prinzipielle Unterschied zwischen den Distanzmaßen der Least-Squares- und der Total-Least-Squares-Methode dargestellt.

2.3 Zuweisung der Punkte zu den Linien

Sowohl bei der Methode der kleinsten Quadrate, als auch bei der Methode der insgesamt kleinsten Quadrate wurde bis jetzt immer davon ausgegangen, dass alle Punkte des Bildes zu einer anzupassenden Linie gehören. Dies ist allerding im Bereich der Computer Vision nur in wenigen Fällen gegeben. In der Regel ist es so, dass entschieden werden muss, welchen Punkt man zur Anpassung welcher Linie wählt. Zur Lösung dieses Problems werden im Folgenden zwei Prinzipien betrachtet:

- die inkrementelle Anpassung
- Punktzuweisung mittels K-means

Die Hough-Transformation, die keine eindeutige Zuordnung erwartet wurde bereits im Kapitel 2.1 vorgestellt.

2.3.1 Die inkrementelle Anpassung

Bei diesem Algorithmus wird die Tatsache ausgenutzt, dass man nur selten isolierte Punkte vorfindet. In der Regel werden die Linien an Punkte angepasst, welche auf den Kanten liegen, die ein Objekt begrenzen. Solche Punkte werden Kantenpunkte genannt. Da diese Kantenpunkte Informationen über ihre Orientierung enthalten, können wir diese Information nutzen, um vorherzusagen, in welcher Richtung der nächste Kantenpunkt liegt. Der inkrementelle Anpassungsalgorithmus verwendet zusammenhängende Kurven von Kantenpunkten und passt die Linien an diese Menge von Punkten an. Solche Kantenpunktkurven sind durch einen Kantendetektor zu erhalten, welcher auch die Orientierung der Kantenpunkte ausgibt. Nun startet der Algorithmus an dem einen Ende der Kurve, passt die Linie entsprechend der Ausrichtung der Kantenpunkte an und enfernt die Punkte, welche zu der Linie gehörten. Kommt er an einen Punkt, an dem die Kantenpunktkurve nicht mehr zu der Linie passt, so beendet er die Anpassung der Linie, speichert diese und fährt mit der Anpassung einer neuen Linie an die verbleibenden Punkte in der Kantenpunktkurve fort. Der zu diesem Verfahren gehörende Algorithmus in Pseudocode lautet wie folgt:

Algorithmus: inkrementelle Anpassung

```
Schreibe alle Punkte in der Reihenfolge in der sie in der Kantenpunktkurve stehen in die
    KurvenListe

Leere die LinienPunkteListe

Leere die LinienListe

until zu wenige Punkte sind in der KurvenListe

    Verschiebe die ersten Punkte von der KurvenListe auf die LinienPunkteListe
    Passe die Linie an die Punkte auf der LinienPunkteListe an
    while die angepasste Linie gut genung ist do
        Verschiebe den nächsten Punkt von der KurvenListe auf die LinienPunkteListe
            und passe diese an
    end
    Verschiebe den letzten Punkt von der LinienPunkteListe auf die KurvenListe
    Passe die Linie an
    Füge die Linie zur LinienListe hinzu

end
```

Die inkrementelle Apassung funktioniert gut, trotz des Mangels eines zugrundeliegenden statistischen Modelles. Was diesen Algorithmus unter anderem attraktiv macht, ist der Fall, dass man nach einer abgeschlossenen Kurve sucht, da er Gruppen von Linien meldet, welche geschlossene Kurven bilden.

2.3.2 Punktzuweisung mittels k-means

Die obige Methode versagt allerdings, wenn die Punkte isoliert liegen, sie also keinerlei Anhaltspunkte über ihre Orientierung liefern. Man kann aber versuchen, mit einem k-Means-Algorithmus zu bestimmen, welcher Punkt auf welcher Linie liegt. In diesem Fall geht man von folgendem Modell aus: es existieren k Linien und jede von ihnen ist der Ursprung einer Teilmenge der Datenpunkte. Der k-Means Algorithmus versucht nun, eine Zuweisung zwischen den Datenpunkten und den angenommenen k Linien zu finden, indem die Gesamtdistanz für jede Linie bezüglich aller möglicherweise zu dieser Linie gehörenden Datenpunkte minimiert wird. Mathematisch wird dies wie folgt ausgedrückt:

$$\text{minimiere} \sum_{l_i \in \text{Linien}} \sum_{(x_n, y_n) \in l_i} dist(l_i, (x_n, y_n))^2. \tag{8}$$

Allerdings bedeutet die Überprüfung aller möglichen Zuweisungen ein Suche in einem unter Umständen sehr großen kombinatorischen Raum. Daher wird durch eine initiale Zuordung die Anzahl der möglichen zu untersuchenden Zuweisungen eingeschränkt.
Grundsätzlich wird folgendermaßen vorgegangen:

- weise jeden Punkt der Linie zu, die am nächsten zu ihm liegt
- passe die Linie unter Berücksichtigung der ihr zugewiesenen Punkte an diese an

Dies führt zu dem folgenden Algorithmus in Pseudocode. Die in diesem vorkommende Konvergenz kann dadurch getestet werden, dass man die Größe der Änderungen innerhalb der Linien beobachtet, ob die Zuordnung sich geändert haben, oder aber man beobachtet die Summe der orthogonalen Abstände der Punkte von ihren Linien.

Algorithmus: Anpassung mit k-Means

wähle ein Zuweisung von Linien zu Punkten und passe diese Linien unter Benutzung dieser Zuweisung an

until Konvergenz

Teile jedem Punkt die am nächsten liegende Linie zu

Passe die Linien an

end

3 Kurvenanpassung

Im Prinzip verläuft die Anpassung von Kurven ähnlich wie die Anpassung von Linien. Es wird versucht, den quadratischen Abstand zwischen der Kurve und den Datenpunkten zu minimieren. Jedoch gestaltet sich hier die Berechnung des Abstandes ein wenig schwieriger. An den zwei Hauptrepräsentationsarten für Kurven, den impliziten und den parametrischen Kurven, soll dieses Problem exemplarisch erläutert werden.

3.1 Implizite Kurven

Die Punkte einer impliziten Kurven erfüllen eine Form von parametrischer Gleichung. Besteht diese Gleichung aus einem Polynom, so wird die Kurve algebraische Kurve genannt. Eine implizite Schreibweise wurde bereits bei der Behandlung der Total Least Squares-Methode verwendet. Dort wurde eine Linie in Form der folgenden Gleichung beschrieben:

$$ax + by + c = 0 \text{ mit } a^2 + b^2 = 1 \tag{9}$$

Ein Kreis mit dem Zentrum im Punkt (a, b) und mit dem Radius r würde durch die folgende Gleichung beschrieben:

$$x^2 + y^2 - 2ax - 2by + a^2 + b^2 - r^2 = 0 \tag{10}$$

Und als letztes Beispiel noch die Darstellung einer Ellipse in impliziter Form:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0, \text{ mit } b^2 - 4ac < 0 \tag{11}$$

Eine allgemeine Schreibweise ist $\phi(x, y) = 0$, wobei x und y die Koordinaten der auf der Kurve liegenden Punkte sind. Der Abstand zwischen einem Datenpunkt (x_n, y_n) und dem zu ihm am nächsten liegenden Punkt auf der Kurve (u, v) ist dann ein zu der Kurve normaler Vektor zwischen dem Punkt auf der Kurve und dem Datenpunkt. Man muss also nach allen Punkten (u, v) auf der Kurve suchen mit den folgenden Eigenschaften:

1. $\phi(u, v) = 0$, klar da (u, v) ein Punkt auf der Kurve sein muss
2. $\boldsymbol{s} = (x_n, y_n) - (u, v)$ ist normal bezüglich der Kurve

Von allen dadurch gegebenen Vektoren $\boldsymbol{s}$ ist die Länge des kürzesten die Distanz zwischen der Kurve und dem Datenpunkt (x_n, y_n).
Während die Berechnung der ersten Eigenschaft einfach ist, muss für die Bestimmung der zweiten Eigenschaft mehr Aufwand betrieben werden. Der Normalenvektor bezüglich einer Kurve hat die Orientierung in der man sich am schnellsten von der Kurve entfernt. Also ändert sich auch der Wert von ϕ in diese Richtung am schnellsten. Das bedeutet, dass sich der Normalenvektor an einer Stelle (u, v) durch die Berechung von

$$s(u, v) = \left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right)(u, v), \tag{12}$$

ergibt. Es existieren also die zwei Gleichungen (Punkt liegt auf der Linie und s normal zu ϕ) mit zwei Unbekannten, nämlich u und v, welche die gesuchten (u, v) erfüllen müssen. Das sieht nach einem grundsätzlich leicht lösbaren Problem aus. Allerdings stellt sich heraus, dass selbst in dem vergleichbar einfachen Fall eines Polynoms vom Grad zwei sich schon schwierig zu lösende Gleichungen ergeben können. Ein Polynom höheren Grades führt schließlich zu nicht mehr effizient lösbaren Gleichungen. Daher ist man in der Praxis dazu übergegangen, Näherungswerte für die Distanz zu bestimmen.
Die wohl bekannteste ist die algebraische Distanz. Hierbei wird die Distanz zwischen dem Punkt und der Linie durch die Auswertung des Polynoms an der Stelle des Datenpunktes näherungsweise bestimmt. Etwas formaler bedeutet das:

$$dist((x_n, y_n), \phi(x, y)) = \phi(x_n, y_n) \tag{13}$$

Die algebraische Distanz kann für nahe bei der Kurve liegende Punkte als ausreichend betrachtet werden, da $\phi(x_n, y_n)$ ansteigt, wenn (x_n, y_n) sich normal zu der Kurve bewegt, der Normalenvektor durch den Gradienten von ϕ definiert ist und gleich bleibt, wenn (x_n, y_n) sich tangential zur Kurve bewegt. Allerdings ist die algebraische Distanz nicht wohldefiniert, da es viele Polynome gibt, die zu derselben Kurve gehören. Denn eine durch $\mu\phi(x, y) = 0$ definierte Kurve ist identisch mit der durch $\phi(x, y) = 0$ definierten Kurve.
Eine mögliche Lösung könnte hier eine Form von Normalisierung der Koeffizienten bringen. Eine solche Art der Normalisierung wurde bereits in Kapitel 2.2.2 angewendet. Dort wurde die Linie $\phi(x, y) = ax+by+c = 0$ an die Datenpunkte angepasst unter der Einschränkung, dass $a^2 + b^2 = 1$ gelten muss. In diesem Fall entspricht die algebraische Distanz der euklidischen Distanz. Vor allem ist es wichtig, die Einschränkungen, die gemacht werden, gut zu überdenken. Würde man zum Beispiel bei der allgemeinen Formel für konische Körper $ax^2 + bxy + cy^2 + dx + ey + f = 0$ einschränken, dass $b = 1$ gelten muss, so wäre es nicht mehr möglich Kreise anzupassen. Daher hat man eine etwas flexiblere Form der Abschätzung gefunden, bei der keine normalisierende Konstante mehr verwendet wird:

$$dist((x_n, y_n), \phi(x, y)) = \frac{\phi(x_n, y_n)}{|\nabla\phi(x_n, y_n)|}. \tag{14}$$

Es wird also die algebraische Distanz durch die Länge des Normalenvektors dividiert, beziehungsweise normalisiert. Daher ist sie auch genauer als die algebraische Distanz. Ansonsten hat sie die gleichen Eigenschaften wie diese bezüglich der Bewegung in Richtung der Normalen und der Bewegung tagential zur Kurve. Durch das Dividieren durch die Länge der Normalen kann man die Ergebnisse dieser Abschätzung auch grob als prozentuale Werte der Entfernung des Punktes von der Kurve in Bezug auf die Länge der Normalen verstehen. In der Praxis wird diese Distanz allerdings nur selten eingesetzt, da die algebraische Distanz wesentlich leichter zu lösende numerische Probleme liefert. Zudem ist die algebraische Distanz auch auf höherdimensionale Probleme anwendbar. So zum Beispiel für die Berechnung der Distanz zwischen einem Punkt und einer impliziten Oberfläche. Daher findet man diese auch häufiger in praktischen Anwendungen wieder.
Wirklich problemlos ist keine der beiden Abschätzungen. Vor allem wenn die Punkte weiter von der Kurve entfernt liegen, ist das Verhalten dieser Aproximationen noch nicht besonders gut verstanden und daher nicht immer kontrollierbar.

3.2 Parametrische Kurven

Bei einer parametrischen Kurve werden die Koordinaten der Punkte, die auf der Kurve liegen, duch einen Parameter bestimmt, der sich entlang der Kurve, also innerhalb eines bestimmten Wertebereiches, verändert. Die allgemeine Schreibweise für parametrische Kurven lautet:

$$f(t) = (x(t), y(t)) = (x(t; \theta), y(t; \theta)) \ , \ t \in [t_{min}, t_{max}]. \tag{15}$$

So würde zum Beispiel ein Kreis mit dem Zentrum im Punkt (a, b) und dem Radius r wie folgt beschrieben:

$$(r\sin(t) + a, r\cos(t) + b) \text{ mit: } \theta = (r, a, b), t \in [0, 2\pi) \tag{16}$$

Und eine an den Koordinatenachsen ausgerichtete Ellipse mit den Ausdehnungen r_1 und r_2 mit dem Zentrum an der Stelle (a, b) hätte diese Repräsentation:

$$(r_1\sin(t) + a, r_2\cos(t) + b) \text{ mit: } \theta = (r_1, r_2, a, b), t \in [0, 2\pi) \tag{17}$$

Die Vorgehensweise, um nun die Distanz zwischen einem Punkt und der Kurve zu berechnen, ist ähnlich der bei impliziten Kurven. Es wird davon ausgegangen, dass man einen gegebenen Datenpunkt (x_n, y_n) hat und man sucht den durch den Parameterwert τ definierten Punkt auf der Kurve, der die geringste Distanz hat. Entweder liegt der Punkt an dem einen oder dem anderen Ende der Kurve, oder aber der Vektor von dem am nächsten zu dem Datenpunkt (x_n, y_n) liegende Punkt auf der Kurve, $(x(\tau), y(\tau))$ zu (x_n, y_n) ist normal zur Kurve. Also: $\boldsymbol{s}(\tau) = (x_n, y_n) - (x(\tau), y(\tau))$ ist normal zu dem Tangentialvektor $\boldsymbol{T}$. Wobei der Normalenvektor $\boldsymbol{s}(\tau)$ in Analogie zu Abschnitt 3.1 wie folgt berechnet wird:

$$s(\tau) = \left(\frac{dx}{dt}(\tau), \frac{dy}{dt}(\tau) \right). \tag{18}$$

Es existiert hier also nur eine Gleichung mit einer Unbekannten τ, im Gegensatz zu den impliziten Kurven, bei denen es ja zwei Gleichungen und zwei Unbekannte waren. Allerdings sind $x(t)$ und $y(t)$ fast immer Polynome, da bei diesen die Berechnung der Nullstellen einfacher ist.

4 Anpassung als probabilistisches Reduktionsmodell

Bis jetzt waren die Kriterien für die Anpassung an ein Modell geometrisch. Das heißt, man versuchte, die gemessenen Distanzen zwischen den Datenpunkten und der Linie zu minimieren. Auch wenn die vorgestellte Total Least Squares-Methode als ein recht gutes Kriterium erscheint, so sollte das Kriterium zur Anpassung doch auf einem Fehlermodell beruhen, welches die Abweichungen modelliert, so wie sie zu erwarten sind. Dies führt zurück zu der Anpassung von Linien an Punkte, von denen bekannt ist, dass sie auf Basis einer Linie erzeugt wurden und vor allem von welcher Linie sie stammen.
Im Folgenden soll gezeigt werden, dass Total Least Squares ein probabilistisches Kriterium ist. Zuerst wird hierzu ein Modell entworfen, welches darstellt, wie die Messwerte bezüglich einer Linie simuliert werden können.
Es wird angenommen, dass N-mal ein Punkt auf der Linie mit einer gleichverteilten Warscheinlichkeit ausgewählt, senkrecht verschoben und dann gemessen wird. Diese Verschiebung soll gaußverteiltem Rauschen entsprechen.
Auch wenn im Prinzip die Auswahl des zu messenden Punktes nicht gleichverteilt sein kann, da die Linie unendlich lang ist, so führt dies jedoch zu so minimalen Veränderungen, dass dieser Tatbestand außer Acht gelassen werden kann. Es existiert also eine Folge von N Messwerten (x_n, y_n) die man aus dem folgenden Modell erhält:

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} + m \begin{pmatrix} a \\ b \end{pmatrix}. \tag{19}$$

Wobei gilt: $m \sim N(0, \sigma)$, $au + bv + c = 0$, und $a^2 + b^2 = 1$.
Man definiert also eine Linie, die aus den Punkten (u, v) besteht. Um nun die Datenpunkte zu erhalten, werden diese Punkte orthogonal zur Linie verschoben, indem man die beiden Koeffizienten a und b multipliziert mit einer gaußverteilten Zufallsvariable m zu dem Punkt auf der Linie hinzuaddiert. Da davon ausgegangen werden kann, dass die generierten Datenpunkte stochastisch unabhängig sind, ergibt sich für ein so gewähltes Modell die

folgende Likelihood-Funktion:

$$P(x_1^N, y_1^N | a, b, c) = \prod_{n=1}^{N} P(x_n, y_n | a, b, c) \tag{20}$$

Wobei x_1^N für $x_1, ..., x_N$ steht. Nun kann man entweder die Werte wählen, an denen die Likelihood-Funktion ihr Maximum hat, oder aber man wählt die maximale *a posteriori* Linie bezüglich des gegebenen Modells. Da es keinen besonderen Grund gibt eine Linie vor einer anderen zu wählen und Maximum Likelihood ein gutes Kriterium ist, wird letzteres gewählt.
Da bekannt ist, dass die Warscheinlichkeit der Einzelmesswerte gaußverteilt ist, also:

$$\text{mit } P(x_n, y_n | a, b, c) \sim \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2\sigma^2}(ax_n + by_n + c)^2),$$

folgt daraus, dass die Log-Likelihood-Funktion wie folgt lautet:

$$-\frac{1}{2\sigma^2} \sum_n (ax_n + by_n + c)^2 \text{ mit } a^2 + b^2 = 1. \tag{21}$$

Berechnet man nun den Maximum Likelihood Wert, so führt dies zu dem Problem der Berechnung von senkrechten Distanzen zwischen Punkten und Linien, welches ja schon in Abschnitt 2.2.2 behandelt wurde.

Um mit diesem Kriterium arbeiten zu können, gibt es zwei wichtige Phänomene, die beachtet werden müssen:

- **Robustheit:** da bei dem Total Least Squares-Kriterium der quadratische Fehler betrachtet wird, werden weit außerhalb liegende Punkte reltiv stark gewichtet. Unter Umständen werden viele gute Datenpunkte durch wenige Ausreißer überlagert. Dies kann zu einer ziemlich starken Verschiebung der anzupassenden Linie führen. In dem folgenden Abschnitt werden zwei Ansätze vorgestellt um mit diesem Problem umzugehen.

- **Fehlende Daten:** bei der Least Squares- und der Total Least Squares-Methode wurde angenommen, dass bekannt ist, welche Punkte von welcher Linie stammen. Dies ist allerdings normalerweise nicht der Fall. In der Regel existieren neben den Punkten der Linie noch eine Menge anderer, durch Rauschen erzeugte Punkte. Wenn nun bekannt wäre, welcher Punkt von der Linie stammt, so könnte man sehr leicht bestimmen, welche Gleichung die Linie darstellt. Genauso könnte man, wenn man wüsste, welche Punkte bei der Messung welcher Linie gemessen wurden, sehr leicht bestimmen, welche Punkte von der Linie erzeugt wurden.

5 Robustheit

Die bisher beschriebenen Verfahren enthalten quadratische Fehler-Terme. Dies kann zu schlechten Anpassungen führen, die darin begründet liegen, dass einzelne Punkte, die weit von der Mehrzahl der anderen Punkte entfernt liegen, eine sehr viel größere Anzahl guter Punkte dominieren, und dadurch das Ergebnis verfälschen. Fehler bei der Erfassung oder der Übermittlung der Daten sind häufig die Gründe für solche Ausreißer. Ein weiterer Grund ist in vielen Fällen auch das zugrunde liegende Modell: eventuell wurden wichtige

Einflüsse nicht beachtete oder ihr Einfluss wurde unterschätzt.
Einer der möglichen Ansätze zum Umgang mit diesen Problemen gibt dem Modell die Schuld. Das Modell geht zum Beispiel davon aus, dass Ausreißer nur sehr selten auftreten, obwohl dies sehr viel öfter der Fall ist. Ein natürlicher Ansatz wäre hier, den Einfluss der Ausreißer zu reduzieren (siehe Abschnitt 5.1) oder aber man erlaubt ein explizites Ausreißermodell. Ein anderer Ansatz wäre die Suche nach Punkten, die für die Anpassung des Modells als gut erscheinen (siehe Abschnitt 5.2).

5.1 M-Schätzer

Eine Möglichkeit, den Einfluss von Aureißern auf den Anpassungsprozess zu verringern, sind die sogenannten M-Schätzer. Bei diesem Verfahren wird versucht eine Funktion durch die Anpassung von Parameterwerten zu minimieren. Die zu minimierende Funktion hat die folgende allgemeine Gestalt:

$$\sum_n \rho(dist((x_n, y_n)\theta); \sigma). \tag{22}$$

Mit θ als die Parameter des Modells, welches angepasst werden soll und mit $dist((x_n, y_n), \theta)$ als der Abstand des Datenpunktes (x_n, y_n) von dem durch θ beschriebenen Modell. Im Allgemeinen sieht $\rho(u, \sigma)$ aus wie u^2 für gewisse Teile des Wertebereichs, und wird dann nach außen hin flach. Eine gängige Wahl lautet:

$$\rho(u, \sigma) = \frac{u^2}{\sigma^2 + u^2} \tag{23}$$

In Abbildung 3 ist die Funktion für drei Werte von σ^2 dargestellt. Hierbei kontrolliert der Parameter σ^2 den Punkt, ab dem die Funktion abflacht. Je größer der Wert für σ^2 gewählt wird, umso größer ist der Einfluss, den Ausreißer auf die Anpassung des Modells haben, da selbst weiter außerhalb liegende Punkte einen nicht so hohen Funktionswert ergeben. Wählt man hingegen σ^2 klein, so veringert sich auch der Einfluss der Ausreißer, da, sobald ein Punkt nur einen geringen Abstand zum Modell hat, er schon einen großen Wert für die Minimierungsfunktion liefert. Außerdem kann man noch sehen, dass die Gewichtung für die Ausreißer ab einem gewissen Punkt immer konstant bei eins liegt. Ab einer bestimmten Distanz ist es also unerheblich wie weit der Punkt entfernt liegt, seine Gewichtung ist immer gleich hoch. Allerdings gibt es zwei Probleme die man bei der Benutzung von M-Schätzern beachten muss:

- Zum einen ist die Berechnung der Extrema nicht analytisch möglich und muss iterativ erfolgen. Hierbei können die folgenden Standard-Probleme auftreten:

  1. es kann mehr als ein lokalels Minimum existieren
  2. das Verhalten der Methode hängt vom Startpunkt ab, welcher für die Methode gewählt wird

  Eine gängige Lösung um dieses Problem zu behandeln ist folgende: zuerst wird eine Teilmenge aller Datenpunkte gewählt, dann wird unter Benutzung der Methode der kleinsten Quadrate an diese Teilmenge angepasst. Das damit gefundene Ergebnis wird nun als Startpunkt für den Anpassungsprozess gewählt. Diesen Prozess wiederholt man dann für eine große Anzahl verschiedener Teilmengen, um dadurch sicherzustellen, dass in dieser Menge eine Probe ist, die vollständig aus guten Datenpunkten besteht.

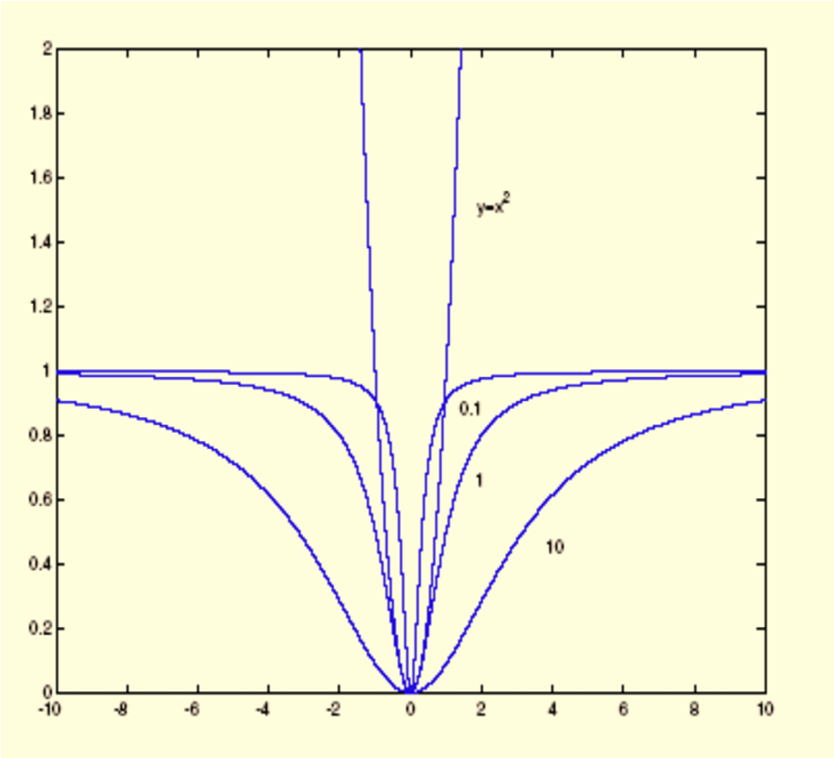

Abbildung 3: Die Funktion $\rho(u, \sigma)$ für $\sigma^2 \in \{0, 1; 1; 10\}$ (aus [1]).

- Zum anderen benötigt der Schätzer eine vernünftige Wahl des Parameters σ, genannt Skalierung. Das Problem ist folgendes: wird σ zu klein gewählt so führt dies dazu, dass zu wenige Punkte als einer Linie zugehörig eingestuft werden. Wird σ zu groß gewählt, so werden auch zu weit außerhalb liegende Punkte als der Linie zugehörig bewertet. In beiden Fällen wird das Ergebnis der Anpassung negativ beeinflusst. Um dieses Problem zu lösen, wird bei jeder Iteration der Lösungsmethode σ neu abgeschätzt. Eine verbreitete Schätzung für σ, welche im Rahmen des unten stehende iterativen Algorithmuses verwendet wird, lautet:

$$\sigma^{(i)} = 1,4826 \text{ median}_n |dist((x_n, y_n); \theta^{n-1})|. \tag{24}$$

Der Algorithmus in Pseudocode zur Anpassung eines probabilistischen Modells unter Benutzung eines M-Schätzers lautet dann wie folgt:

Algorithmus:

For $s = 1$ **to** $s = k$

Ziehe eine Teilmenge von n verschiedenen, gleichverteilt gewählten Punkten

Passe an diese Menge von Punkten, unter Benutzung von Maximum Likelihood (gewöhnlich kleinste Quadrate) an um θ_s^0 zu erhalten

Schätze σ_s^0 unter Benutzung von θ_s^0 ab

Until Konvergenz (gewöhnlich: $|\theta_s^r - \theta_s^{r-1}|$ist klein)

Mache einen Minimierungsschritt durch Anwendung von θ_s^{r-1}, σ_s^{r-1} um θ_s^r zu erhalten

Berechne nun σ_s^r

end

end

gib die beste Anpassung an die Menge zurück, unter Benutzung des Medians der Restfehler als Kriterium

5.2 Random Sample Consensus (RANSAC)

Eine Alternative zu dem Ansatz der Verringerung des Einflusses von Ausreißern auf die Anpassung des Modells ist die Menge von Datenpunkten nach Punkten zu durchsuchen, die für die Anpassung des Modells als gut erscheinen. Ein iterativer Prozess, wie der nachfolgende, wäre hierfür gut geeignet:

1. wähle zufällig eine kleine Teilmenge aller Punkte
2. passe bezüglich dieser Teilmenge an
3. nun sieht man wie viele der anderen Punkte zu dem daraus resultierenden Objekt passen und speichert dies als Gütekriterium für die Anpassung
4. wiederhole diesen Prozess so lange, bis eine hohe Warscheinlichkeit besteht, dass eine der Anpassungen nur auf Datenpunkten beruhte, die innerhalbe eines bestimmten Bereiches um das anzupassende Modell liegen

Zum besseren Verständniss ein kleines Beispiel: angenommen, dass eine Linie an eine Datenmenge angepasst werden soll, die zu etwa 50 % aus Ausreißern besteht. Wenn nun aus dieser Menge gleichverteilt Punktpaare gezogen werden, so dürften etwa ein viertel dieser Paare nur aus guten Datenpunkten bestehen. Solche guten Datenpunktpaare können dadurch erkannt werden, dass eine große Anzahl anderer Punkte in der Nähe der Linie liegen, welche wir mittels dieser Paare angepasst haben. Nun kann natürlich eine noch bessere Anpassung durch die Einbeziehung der Punkte, welche nahe bei der angepassten Linie liegen, in dem neuerlichen Anpassungsprozess erfolgen.
Ein solcher Ansatz führt zu einem in [3] beschriebenen Algorithmus, welcher hilft, eine zufällig gewählte Datenmenge zu finden, welcher zu einer Anpassung führt, die zu der eine großen Anzahl der Daten aus der Ursprungsdatenmenge passen. Dieser Algorithmus wird

RANSAC genannt. Dies steht für RANdom SAmple Consensus, was übersetzt soviel bedeutet wie Zufallsprobenübereinstimmung. Dieser Algorithmus zur Anpassung von Linien in Pseudocode lautet wie folgt:

Algorithmus:

Setze fest:

n - die kleinste Anzahl an benötigten Punkten
k - die Anzahl der benötigten Iterationen
t - die Grenze, mittels welcher bestimmt wird, ob ein Punkt nahe genug bei der Linie liegt
d - die Anzahl der benötigten in der Nähe liegenden Punkte um zuzusichern dass das Modell gut passt

Until k Iterationen wurden durchgeführt

Ziehe eine Teilmenge von n Punkten, gleichverteilt aus den Daten gewählt

Passe an diese Menge von n Punkten an

For each Datenpunkt außerhalb der Teilmenge

Prüfe die Distanz zwischen Punkt und Linie gegen t. Falls die Distanz zwischen dem Punkt und der Linie kleiner als t ist, dann liegt der Punkt nahe

end

Falls d oder mehr Punkte existieren, deren Abstand von der Linie kleiner als t ist, passe die Linie unter Benutzung aller dieser Punkte wieder an.

end

Benutze die beste Anpassung aus der Sammlung, unter Verwendung des Anpassungsfehlers als Kriterium

In diesem Algorithmus existieren drei nicht spezifizierte Paramter, nämlich:

1. die Fehlertoleranz d um zu bestimmen, ob ein Punkt zu dem Modell kompatibel ist
2. die maximmale Anzahl an Versuchen k um eine Übereinstimmungsmenge zu finden, beziehungsweise die Anzahl der Teilmengen, die getestet werde müssen
3. der Grenzwert t um zu bestimmen wann ein Punkt nahe bei der Linie liegt

Fehlertoleranz um zu bestimmen wie gut die Realwerte und das Modell zueinander passen
Um nun festzustellen ob die ausgewählten Datenpunkte zu dem Modell passen, wird gezählt, ob genug Punkte innerhalb eines gewissen Abstandes zu dem Modell liegen. Ist dies der Fall, so ist die Anpassung des Modells gut. Geht man davon aus, dass w dem Anteil der nahe genug beim Modell liegenden Punkte entspricht, so sollte man d so wählen, dass $(1-w)^d$ klein ist.

Maximale Anzahl an Versuchen um eine Übereinstimmungsmenge zu finden

Das hier gewählte Modell besteht aus Punkten, die gleichverteilt aus der Punktmenge ausgewählt wurden. Bei jeder Wahl wird nur die minimale Anzahl n der Punkte ausgewählt, die ausreicht um das gesuchte Modell vollständig zu beschreiben. Wird nach Linien gesucht, so werden zwei Punkte gewählt, bei einem Kreis drei, und so weiter. Nun gilt für die erwartete Anzahl der Iterationen k, die benötigt werden um einen guten Punkt zu erhalten: wenn man gewährleisten möchten , dass mit einer Warscheinlichkeit z eine der n-Elementigen Zufallsproben nur aus guten Datenpunkten bestehen soll, dann muss für das zu wählende k gelten:

$$(1 - z) = (1 - w^n)^k \tag{25}$$

So dass man k wie folgt berechnen kann:

$$k = \frac{\log(z)}{\log(1 - w^n)} \tag{26}$$

Im Allgemeinen ist jedoch nicht bekannt, welchen Wert w hat. Allerdings liefert jede neue Anpassung weitere Anhaltspunkte für den Wert von w. Man kann sagen, falls n Datenpunkte benötigt werden, so ist die Warscheinlichkeit einer erfolgreichen Anpassung w^n. Nach einer langen Folge von Versuchen, kann w aufgrund dieser Folge abgeschätzt werden. Es wird also mit einer relativ niedrigen Abschätzug von w angefangen und nach einer gewissen Anzahl von Anpassungsversuchen wird w dann neu abgeschätzt. Ab einem gewissen Punkt ist w dann so gut abgeschätzt dass keine weiteren Anpassungsversuche mehr durchgeführt werden müssen.

Wann liegt ein Punkt nahe bei der Linie? Um diese Frage beantworten zu können, wird die Distanz zwischen dem Punkt und der angepassten Linie berechnet. Liegt diese Distanz unter der Schranke t, so liegt der Punkt nahe bei der Linie. Die Festlegung dieses Wertes ist Teil des Modellierungsprozesses. Als beispielsweise bei der Anpassung der Linie Maximium Likelihood verwendet wurde, gab es den Term der Standardabweichung σ in dem verwendeten Modell, welcher angab wie stark die Abweichungen von dem anzupassenden Modell im Durchschnitt abweichen.
Die Bestimmung eines Wertes für t wird oft heuristisch vorgenommen. Man schätzt in etwa die Größenordnung für t ab und wendet diesen dann einige male auf verschiedene Modelle an. Dies macht man mit einigen Werten für t und wählt anschließend den, der die besten Ergebnisse liefert. Ein anderer Ansatz wäre es, ein paar charakteristische Datenmengen zu betrachten, die Linie mit dem Auge anzupassen und dann die Standardabweichung abzuschätzen.

6 Automatischen Fahrbahnerkennung auf Autobahnen

Als abschließendes Beispiel für die Segmentierung durch Modellanpassung soll im Folgenden die automatische Erkennung von Fahrbahnen auf Autobahnen beschrieben werden. Die Fahrbahnerkennung ist der grundlegende Schritt um darauf aufbauend dann komplexere Systeme wie zum Beispiel Systeme zur automatischen Verkehrsüberwachung konstruieren zu können, wie sie zum Beispiel in [4] vorgestellt wurden. Gerade für solche Systeme bieten Autobahnen sich an, da hier eine reltiv stark strukturierte Umgebung vorliegt. Der ständig zunehmende Verkehr und die damit verbundene Zunahme des Aufwandes zur Überwachung dieses Verkehrs macht solche automatisierten Systeme zunehmend interessanter - nicht nur zur Überwachung sondern auch zur intelligenten Steuerung der Verkehrsflüsse.

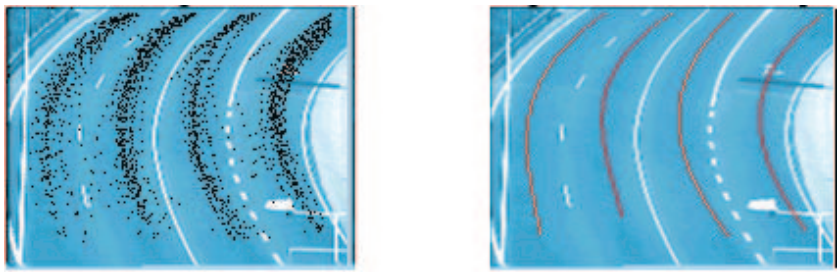

Abbildung 4: links: Bewegunsbahnen, rechts: die zentren der entsprechenden Cluster (aus [4]).

Als Basis für die Erkennung der Fahrbahnen dienen Fimaufnahmen des Verkehrsflusses. Diese werden statischen Bildern vorgezogen, da man hier nichtstationäre Pan-Tilt-Zoom-Kameras verwenden kann. Zudem ist ein solches Verfahren Blickwinkel und Skalenunabhängig und weniger empfindlich gegenüber sensorischem Rauschen und Lichtveränderungen. Das einzige, was bei diesem Verfahren bekannt sein muss, ist die ungefähre Fahrbahnbreite, da diese später im Algorithmus benötigt wird.
Grob gesagt geht der Algorithmus dann wie folgt vor: zuerst werden die sich bewegenden Fahrzeuge erkannt und eventuelle Überdeckungen der Fahrzeuge behandelt, so dass zwei Fahrzeuge, von denen das eine das andere überdeckt, auch als zwei Fahrzeuge erkannt werden. Danach werden die Bewegungen der Fahrzeuge analysiert, und abschließend können dann auf dieser Analyse aufbauend die Fahrbahnen bestimmt werden. Um die Fahrbahnen bestimmen zu können, wird zuerst von dem Benutzer eine Referenzsuche T_R spezifiziert. Diese wird als Polynom vom Grad M dargestellt. In der Regel ist als Referenzsuche ein Polynom vom Grad 2 ausreichend. Nun geht man wie folgt vor: Suche Bewegungsbahnen T_i, die eine ähnliche Ausrichtung haben wie die spezifizierte Referenzsuche T_R und die untereinander einen Abstand haben, der der Fahrbahnbreite entspricht. Fasse nun weitere Bahnen, die in der Nähe der T_i liegen zu Clustern zusammen und modelliere diese Cluster alle fünf Durchgänge mit `RANSAC` neu. Dies führt zu einer Least Squares-Polynomanpassung, da `RANSAC` robust ist gegenüber Ausreißern, die durch Fahrbahnwechsel, verdeckte Fahrzeuge oder Rauschen in der Videoaufnahme erzeugt wurden. In Abbildung 4 ist dieser Vorgang bildlich dargestellt. Links sieht man die Bewegungsbahnen und rechts die daraus reultierenden Cluster, die den Fahrbahnverlauf darstellen.

7 Zusammenfassung

Als Einstieg in diese Ausarbeitung wurde die Hough-Transformation vorgestellt. Danach folgten, als distanzbasierte Verfahren, die Methode der kleinsten Quadrate und die Methode der insgesamt kleinsten Quadrate. In diesem Zusammenhang trat dann das Problem auf, welche Punkte zu welcher Linie gehören. Dies wurde durch die Einführung der inkrementellen Anpassung sowie der Punktzuweisung mittels k-Means gelöst. Bis dahin wurden als anzupassende Modelle lediglich Linien verwendet, da hierdurch die Anschaulichkeit besser erhalten blieb. Um aber zu zeigen wie eine Erweiterung auf komplexere Modelle aussehen könnte, wurden die vorgestellten Verfahren auch für die Anpassung von Kurven, sowohl impliziter als auch parametrischer Kurven, erläutert. Um das geometrische Kriterium, was bis dahin immer verwendet wurde, durch ein probabilistisch motiviertes Kriterium zu er-

setzten, wurde nachgewiesen, dass die Methode der insegesamt kleinsten Quadrate an sich bereits ein probabilistisches Kriterium ist. Anschließend wurde die Robustheit der Verfahren durch die Verwendung von M-Schätzern und dem `RANSAC`-Algorithmus verbessert. Als letztes wurde dann noch ein Beispiel zur Erkennung von Fahrbahnen auf Autobahnen vorgestellt.
Abschließend kann man sagen, dass die Segmentierung durch Modellanpassung ein vielseitig einsetzbares Verfahren ist um Objekte in Bildern zu bestimmen und welches besonders durch die Erhöhung der Robustheit noch mehr an Attrktivität hinzugewinnt.

Literatur

[1] D. Forsyth, J. Ponce. *Computer Vision - A Modern Approach*, Prentice Hall, 2003, S. 329-352

[2] M. A. Fischler, R. C. Bolles. *Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography*, Comm. of the ACM, 24 (1981), S. 381-395.

[3] R.O. Duda, P.E. Hart. *Use of Hough transform to detect lines and curves in pictures*, Comm. ACM 15, 1972, S. 204-208

[4] J. Melo, A. Naftel, A. Bernardino, J. Santos-Victor. *Retrieval of Vehicle Trajectories and Estimation of Lane Geometry Using Non-Stationary Traffic Surveilance Cameras*, ACIVS (Advanced Concepts for Intelligent Vision Systems) 2004.