Westfälische Wilhelms-Universität Münster
Institut für Informatik

Seminar: „Unterstützung von Landminendetektion durch Bildauswertungsverfahren und Robotereinsatz“ (WS 03/04)
Dozenten: Dr. D. Lammers, Dipl. Inf. S. Wachenfeld
Bearbeiter: Christian Große Lordemann, Martin Lambers

Objekterkennung in Bilddaten

Inhalt

Abbildungen

1. Einführung

1.1. Ziele

Ziel unseres Vortrags ist eine Einführung in einige grundlegende Verfahren und Vorgehensweisen der Objekterkennung. Dabei werden wir jeweils auf die Ideen, die sich hinter den Ansätzen verbergen, eingehen und die Probleme, die auftreten können, besprechen. Die Datengrundlage sind Bilddaten, wobei wir uns größtenteils auf 2D-Bilddaten beziehen werden. Wir wollen keineswegs verschweigen, dass auch 3D-Objekterkennungssysteme existieren. Allerdings haben wir in Anbetracht der knappen Zeit und der Fülle des Materials beschlossen, uns auf 2D-Objekterkennungssysteme zu beschränken und die dritte Dimension nur zu betrachten, wenn es gerade sinnvoll ist.

Ziel unseres Vortrags ist es nicht, *das* Objekterkennungssystem vorzustellen. Vielmehr ist es unser Ziel zu zeigen, dass man abhängig von vielen verschiedenen Parametern ein Objekterkennungssystem individuell zusammenstellen muss.

1.2. Was ist Objekterkennung oder wie sehen Computer?

Um diesen Teil ein wenig unwissenschaftlicher anzugehen, begeben wir uns in den Film „2001 – Odyssee im Weltraum“. Mit an Bord ist der HAL-9000. HAL ist ein Supercomputer mit einer hoch entwickelten künstlichen Intelligenz, der menschliche Züge annehmen kann und sogar eine eigene Meinung besitzt. Von HALs Kamera aufgenommen Bilder werden in digitale Signale umgewandelt und in seinem „Gehirn“ verarbeitet. Er konnte Bilder bewerten und sogar Personen erkennen. Mit solchen Fähigkeiten ist HAL heutigen Systemen natürlich weit überlegen, trotzdem besitzen heutige Systeme einen ähnlichen Anspruch. Mittlerweile wird auf vielen Gebieten der Einsatz von Objekterkennungssystemen erforscht. Darunter zum Beispiel in der Medizin (unter anderem zur Erkennung von Tumoren oder Anomalien), in der optischen Qualitätskontrolle, in der Dokumentanalyse (unter anderem zur automatischen Erkennung von Handschrift), in der Industrie (zum Beispiel zur Steuerung von Industrierobotern), in der Entwicklung von intelligenten Benutzerschnittstellen, in der Biometrik (unter anderem zur Gesichtserkennung) oder in der Entwicklung von autonomen Fahrzeugen bzw. intelligenten Fahrhilfen. Es gibt allerdings auch schon einige Objekterkennungssysteme in der Anwendung, zum Beispiel bei der automatischen Mülltrennung.

Am Anfang jedes Objekterkennungsprozesses steht immer die Datensammlung. Dazu braucht der Computer ein Medium, mit dem er die Daten sammeln kann. Bei HAL ist das eine normale Kamera, allerdings gibt es Systeme, die mit anderen Medien arbeiten, zum Beispiel mit Infrarotkameras, Radargeräten oder Röntgengeräten. Liegen die Daten dann vor, so werden Algorithmen auf ihnen ausgeführt um nähere Informationen zu sammeln, die auf das Objekt hindeuten.

Uns Menschen macht es keine Probleme Objekte zu erkennen, aber auch bei uns werden Da-

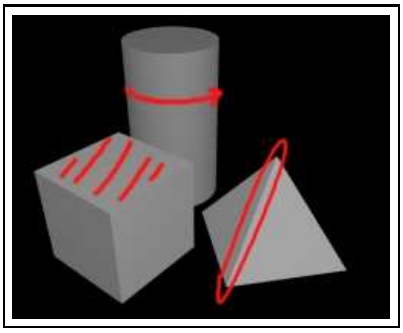

Abbildung 1 (Quelle: [4])

ten/Informationen an das Gehirn gesendet, wo sie den Objekterkennungsprozess auslösen. Bei Menschen ist dieser Prozess intuitiv, der Computer dagegen muss die Objekte erst aus den Eingabepixeln extrahieren. Doch bevor man sich mit der Extraktion auseinandersetzt, muss man sich zunächst näher mit den Pixeln beschäftigen, die die wichtigste Informationsquelle für den Computer darstellen.

Jedes Pixel ist im eigentlichen Sinne eine Repräsentation des Lichts, das die Kamera aus der Richtung des Objekts erreicht. Das Pixel enthält also Informationen über Helligkeit und Farbe einer durch mehrere Pixel repräsentierten Fläche, wobei die Helligkeit und die Farbe von der Beleuchtung und dem Reflexionsvermögen des Objekts abhängt. Willkürliche Änderungen in der Leuchtkraft, Reflexionskraft und Ausrichtung lassen nicht auf das Vorhandensein eines Objekts schließen, fließende Übergänge dagegen schon[4].

Starke Änderungen des Pixelwertes sind ein Hinweis für den Übergang von einem Objekt zum nächsten, stufenweise Änderungen (Schattierungen) dagegen sind ein Hinweis für die Änderungen der Oberflächenausrichtung (siehe Abbildung 1). Durch diese Informationen kann der Computer das Bild in Regionen einteilen, welche Objekte entsprechen. Diese Eigenschaften werden zum Beispiel auch bei der Kantendetektion ausgenutzt, worauf aber in späteren Kapiteln noch näher eingegangen werden soll.

Wenn wir im vorherigen schon auf das Sehen beim Menschen und das „Sehen“ beim Computer eingegangen sind, so stellt sich die Frage: Kann der Computer genauso gut sehen wie wir? Zu dem jetzigen Zeitpunkt kann man diese Frage auf jeden Fall mit einem klaren Nein beantworten. Objekterkennung beim Menschen ist auch heute noch ein Feld, in dem geforscht wird. Allerdings kann man feststellen, dass die Erkennung beim Menschen ein komplexer Vorgang ist, der eine Vernetzung verschiedenster Fähigkeiten voraussetzt. Er ist in der Lage in hohem Maße Erfahrung und Kontextwissen mit in den Erkennungsprozess einfließen zu lassen.

Zum Beispiel ist in Abbildung 2 im ersten Augenblick keine Struktur zu erkennen, die auf ein Objekt schließen lässt. Weiß man allerdings, dass ein Dalmatiner zu sehen ist, so kann man ihn im Bild sofort erkennen. Dieser Erkennungsprozess setzt voraus, dass wir vorher schon mal einen Dalmatiner gesehen haben. Wir haben dadurch ein Bild des Dalmatiners in unserem Kopf und können so trotz fehlender Umrisse das Bild imaginär vervollständigen. Eine weitere ähnliche Fähigkeit ist das Erkennen von perspektivischen Strukturen in Bilddaten. Zum Beispiel erkennen wir in Abbildung 3 genau, dass der linke Baum weiter vorne stehen muss als

Abbildung 2 (Quelle: http://www.mindfake.com/illusion_22.html, 23.12.2003)

Abbildung 3 (Quelle: Windows XP Desktopbilder)

die Bäume in der Bildmitte. Wir können also aus unserer 3D-Realität in „flache“ Bilddaten abstrahieren.

In Abbildung 4 sind eigentlich nur drei Kreise mit Aussparungen zu erkennen. Dass wir trotzdem ein Dreieck zwischen den Kreisen sehen können, liegt daran, dass unsere Vorstellung die drei Basisecken imaginär vervollständigen kann.

Ein Mensch erkennt auch Objekte wieder, wenn sie gedreht oder teilverdeckt sind. Er scheint außerdem Funktion und Eigenschaften von Objekten direkt erkennen zu können und ist somit in der Lage Objekte schnell und sicher zu klassifizieren.

Auch wenn HAL ähnlich effektiv in der Erkennung ist wie seine menschlichen Begleiter, so sind heutige Computer von unseren Fähigkeiten noch weit entfernt. Es existieren heute zwar

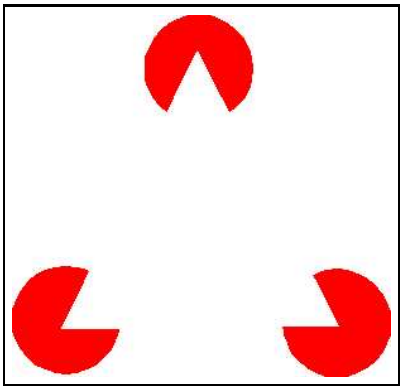

Abbildung 4 (Quelle: http://www.uebi.de/pages/optik/optik_11.htm, 23.12.2003)

Abbildung 5 (Quelle: [6])

schon Sensoren, die dem menschlichen Auge weit überlegen sind und dem Computer so eine bessere Datengrundlage verschaffen. Trotzdem ist es schwer den Computer in diesen Daten Objekte erkennen zu lassen, da wir einfach nicht wissen wonach wir ihn dann suchen lassen sollen. Ebenfalls nachteilig ist, dass heutige Systeme, im Gegensatz zu HAL, nur begrenzt lernfähig sind. Sie sind immer noch im hohen Maße von Konfigurationen des Menschen abhängig. Lässt man zum Beispiel einen Computer ein Objekt in ungewohnter Umgebung erkennen, in der die Objekte dazu noch gedreht und teilverdeckt sind, so versagen viele Systeme. Ein ähnliches Problem stellen ständig variierende Daten und Bedingungen, wie Änderungen der Lichtintensität, des Hintergrundes oder der Aufnahmeentfernung da.

Abbildung 5 fasst sehr gut mögliche Probleme, die bei einer Objekterkennung auftreten können, zusammen. Wir haben hier viele ähnliche Objekte, Teilverdeckungen, wechselnde Lichtintensität, verschiedene Objektausrichtungen und noch weitere Probleme mehr.

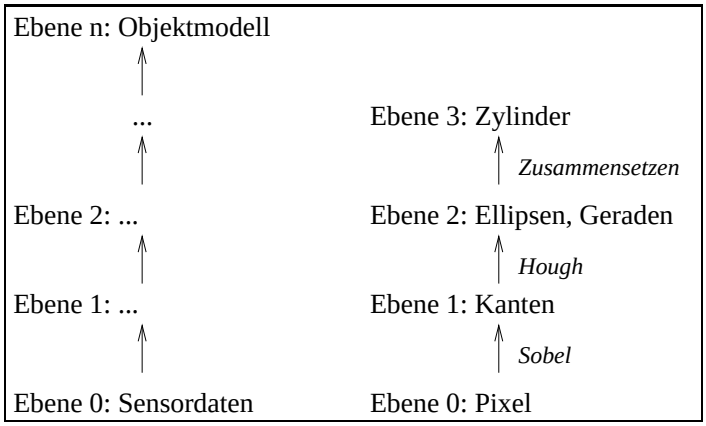

Abbildung 6

Trotz der vielen Probleme, die auftreten können, gibt es viele Ansätze und Methoden in der Objekterkennung, die robust gegenüber einige dieser Probleme sind und Objekte sicher in Bilddaten erkennen können. In den folgenden Betrachtungen wollen wir uns mit einigen dieser Methoden beschäftigen.

2. Ein Aufbau eines Objekterkennungssystems

Zunächst wollen wir uns aber mit einem möglichen Aufbau eines Objekterkennungssystems auseinandersetzen. Dazu stellt sich die Frage: Wie will man in dem Objekterkennungsprozess vorgehen?

Die einfachste Methode ein Objekt zu erkennen wäre das zu erkennende Objekt zu fotografieren und dieses Objektbild dann pixelweise mit einem unbekannten Bild, das das Objekt enthält, zu vergleichen. Diese Methode wäre allerdings sehr fehleranfällig. Sie wäre nicht robust gegenüber Größenänderungen des Objekts im Bild, gegenüber Drehungen, gegenüber Änderungen des Blickwinkels, gegenüber Belichtungsänderungen, etc. Des Weiteren wäre sie sehr ineffizient, da zum Beispiel auch Objekte verglichen werden, die eckig sind, obwohl man nach runden Objekten sucht. Um diese Nachteile zu vermeiden bildet man Informationsebenen. Das Prinzip hinter den Informationsebenen ist, dass man versucht das unbekannte Bild in verschiedene Komplexitätsebenen einzuteilen. Die unterste Ebene enthält die Sensordaten, die nächste Ebene baut auf dieser Ebene auf und versucht nähere Informationen zu extrahieren, sie erhöht sozusagen die Komplexität des Informationsgehalts. In der obersten Ebene ist die Komplexität am höchsten, wir besitzen also genug Informationen, um das Objekt zu erkennen. Ein Objekterkennungssystem muss sich also von der untersten Ebene nach oben durcharbeiten, um dann Aussagen über das Vorhandensein eines Objektes machen zu können.

Abbildung 6 zeigt schematisch den Aufbau von Informationsebenen (links) und Informationsebenen am Beispiel einer Erkennung eines Zylinders (rechts). Auf die jeweiligen Methoden (Hough und Sobel) wird später noch genauer eingegangen.

Im Allgemeinen müssen folgende Schritte durchlaufen werden: Zuerst wählt man einen Sensor aus, der je nach Art des zu erkennenden Objekts und je nach Bedingung unterschiedlich sein kann (Schritt 0). Die Bilder, die so gewonnen werden, müssen vorverarbeitet werden, um Aufnahmestörungen zu minimieren (Schritt 1). Dieser Schritt fällt noch in die Bildverarbeitung, weswegen für nähere Informationen auf den vorherigen Vortrag verwiesen wird. Anschließend wird in dem aufbereiteten Bild nach vorher festgelegten Merkmalen gesucht (Schritt 2). Diese Merkmale können je nach Objekt und Vorgehen verschieden sein, umfassen aber unter anderem Kanten, Ecken, Farbe oder Reflektanz. Zur Extraktion werden die festgelegten Merkmale in den Attributraum (bzw. Merkmals- oder Eigenschaftsraum) transferiert (Schritt 3). Der Attributraum steht in keiner Verbindung mit dem Ausgangsbild, sondern enthält nur die extrahierten Merkmale. Nähere Erläuterungen folgen in den anschließenden Kapiteln. Da die Merkmale jetzt im Attributraum vorhanden sind, können sie mit Merkmalen von bekannten Objekten, die zum Beispiel in einer Datenbank stehen, verglichen werden (Schritt 4). Dieses Verfahren wird auch Matching genannt, Näheres dazu später. Der letzte Schritt beinhaltet die Ausgabe der erkannten Objekte mit ihren jeweiligen Positionen im Bild.

Wie schon vorher angedeutet ist dieser Aufbau nicht statisch. Es gibt viele Möglichkeiten solche Systeme aufzubauen. Manche weichen komplett von diesem Aufbau ab, doch viele operieren auf einer ähnlichen Struktur. Wichtig ist bei allen Arten von Erkennungssystemen die Planung. Fragen die man sich hierzu stellen muss sind: Welche Vorkenntnisse besitze ich? Ist das Objekt rund, eckig, farblich stark textuiert? Wie ist der Hintergrund aufgebaut? Wie kann ich diese Vorkenntnisse dem Rechner zur Verfügung stellen, zum Beispiel durch Extraktionsmethoden oder durch spezifische Eigenheiten, wie Histogramme bei Farben? All diese Überlegungen stehen unter dem Aspekt, Mangel an Kontextwissen und Erfahrung auszugleichen.

Im nächsten Kapitel werden einige grundlegende Methoden der Objekterkennung vorgestellt. Diese teilen sich in zwei Hauptarten auf und zwar in modellbasierte Erkennung und erscheinungsbasierte Erkennung. Beide Arten basieren im Groben auf oben vorgestelltem Aufbau, sind aber von ihrer Vorgehensweise völlig verschieden.

3. Methoden

3.1. Modellbasierte Methoden

3.1.1. Kantendetektion

Ziel der Kantendetektion ist es, in einem Bild die Konturen von Objekten zu finden. Man will also von der Informationsebene der Pixel zur Informationsebene der Kantenpunkte kommen.

Eine Kante wird dabei als Änderung der Grauwerte betrachtet. Je stärker die Änderung, desto höher die *Kantenstärke*. Auch die *Kantenrichtung* ist interessant. Abbildung 7 zeigt eine starke Kante in horizontaler Richtung und eine schwache Kante in vertikaler Richtung.

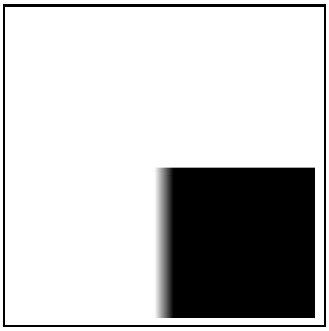

Abbildung 7

Es gibt verschiedene Möglichkeiten, Kanten zu detektieren. Hier soll der Gradientenansatz vorgestellt werden: Betrachte eine zweidimensionale Funktion $f(x,y)$ (differenzierbar). Der Gradient dieser Funktion ist $g(x,y) = (f_x, f_y)$. Er zeigt in die Richtung der höchsten Steigung und sein Betrag ist ein Maß für die Stärke der Steigung.

Überträgt man die Gradientenberechnung auf Bilder, liefert sie also sowohl Kantenstärke als auch Kantenrichtung (die Kantenrichtung ist senkrecht zur Gradientenrichtung).

Dazu betrachtet man ein Bild als diskrete zweidimensionale Funktion $f(x,y)$, die den Grauwert des Punktes (x,y) liefert. Jetzt braucht man Näherungen für die partiellen Ableitungen f_x und f_y. Davon gibt es viele; das weitverbreitete Verfahren von Sobel benutzt die Masken

$$f_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \text{ und } f_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}.$$

Für jeden Bildpunkt (x,y) liefert dann $|s(x,y)| = \sqrt{f_x^2(x,y) + f_y^2(x,y)}$ ein Maß für die Kantenstärke.

Die Gradientenrichtung lässt sich aus $d(x,y) = \arctan(f_x/f_y)$ berechnen (dabei sind noch Fallunterscheidungen zu machen). Aus ihr erhält man die Kantenrichtung des Punktes (x,y).

Störungen im Urpsrungsbild können zu falsch detektierten Kanten führen. Um ihre Anzahl zu verringern, empfiehlt sich ein Glättung des Bildes vor der Kantendetektion. Glättungsmethoden wurden im vorherigen Vortrag vorgestellt.

Anschließend sollten die Kanten noch verdünnt werden, damit sie besser Konturlinien repräsentieren.

Mit einer Schwellwertoperation kann danach entschieden werden, welche Kantenpunkte im Weiteren betrachtet werden und welche nicht – so kann man zur Vereinfachung von vielen Kantenpunkten unterschiedlicher Stärke zu einem binären Kantenbild übergehen.

Abbildung 8 zeigt oben links das Originalbild, oben rechts das Glättungsergebnis (Gaussfilter, $\sigma = 1.2$), unten links das Ergebnis der Sobel Kantendetektion und unten rechts das nach-

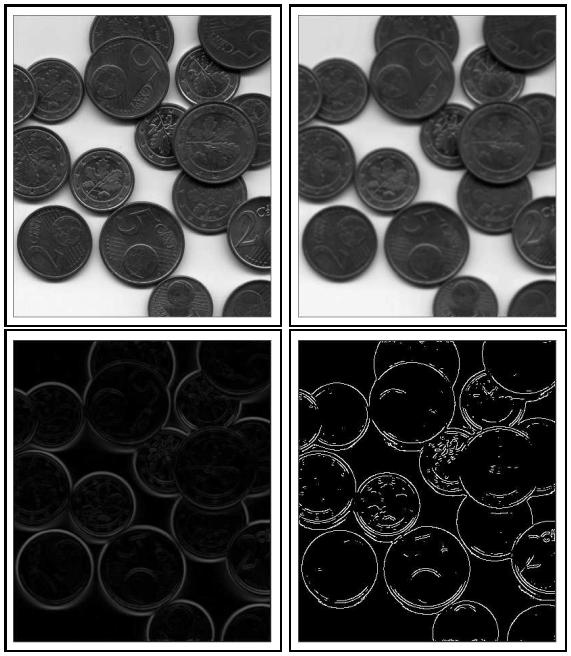

Abbildung 8

bearbeitete Endergebnis. Die Schritte Gaussglättung, Sobel Kantendetektion, Verdünnung und Schwellwertoperation werden zusammengefaßt auch Canny Kantendetektion genannt.

3.1.2. Houghtransformation

Die Houghtransformation baut auf der Kantendetektion auf: Sie sucht Formen, die von Kantenpunkten gebildet werden. Die Houghtransformation ist sehr allgemein verwendbar; Formen können im einfachsten Fall Geraden oder Kreise sein, es können aber auch Ellipsen und andere geometrische Figuren detektiert werden. Mit der verallgemeinerten Houghtransformation können sogar beliebige Formen gefunden werden.

Die Houghtransformation benutzt ein einfaches Grundprinzip: Man untersucht alle Kantenpunkte auf *Hinweise* auf eine gegebene Form, die man detektieren möchte. Diese Hinweise werden im *Akkumulatorraum* gespeichert. Nachdem alle Kantenpunkte untersucht wurden, wertet man die gesammelten Hinweise im Akkumulatorraum aus.

Die Untersuchung der Kantenpunkte und die Dimension des Akkumulatorraums hängen dabei von der Form ab, die man detektieren will. Dies soll am Beispiel von Kreisen mit bekanntem Radius verdeutlicht werden:

Ein Kreis mit Radius r_0 kann durch seinen Mittelpunkt (x_0, y_0) charakterisiert werden.

In Abbildung 9 sieht man einen Kantenpunkt (dargestellt durch ein Kreuz). Die Mittelpunkte der Kreise mit Radius r_0, zu denen dieser Punkt gehören könnte (zwei davon sind gestrichelt eingezeichnet), liegen auf einem Kreis mit Radius r_0 um ihn herum. Diese möglichen Mittelpunkte

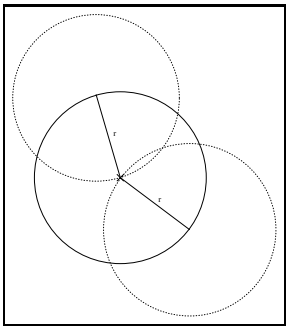

Abbildung 9

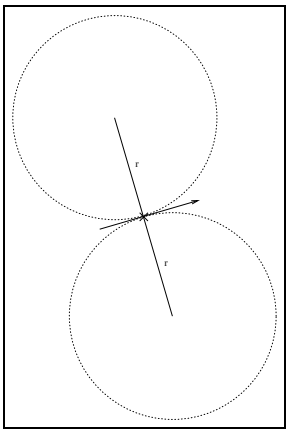

Abbildung 10

sind die Hinweise; man sammelt sie im Akkumulator, indem man für jeden Hinweispunkt (r,c) die Akkumulatorzelle (r,c) hochzählt.

Auf diese Weise sammelt man Hinweise aus allen Kantenpunkten. Kantenpunkte, die tatsächlich auf einem Kreis mit Radius r_0 liegen, sorgen für einen hohen Akkumulatorwert am Mittelpunkt dieses Kreises.

Nachdem die Hinweise gesammelt sind, muss der Akkumulatorraum ausgewertet werden. Dazu werden lokale Maxima gesucht. Dies sind die gesuchten Kreismittelpunkte.

Die Kantendetektion liefert im Allgemeinen aber nicht nur Kantenpunkte, sondern auch die zugehörige Kantenrichtung. Diese Information kann genutzt werden: Es bleiben nur noch zwei mögliche Kreise übrig (siehe Abbildung 10). Dadurch wird der Aufwand drastisch reduziert und gleichzeitig der Akkumulatorraum leichter auswertbar.

Abbildung 11 zeigt ein komplettes Beispiel: Zunächst das Kantenbild, dann den Akkumulator (einmal ohne und einmal mit Nutzung der Richtungsinformation) und das Ergebnis der Houghtransformation.

Das Vorgehen besteht also aus folgenden Schritten:
– Wahl einer geeigneten Darstellung $(x_1,\ldots,x_n)$ der zu detektierenden Formen (im Fall von

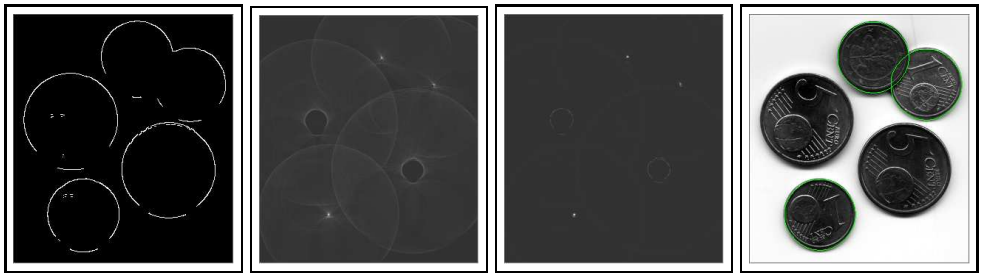

Abbildung 11

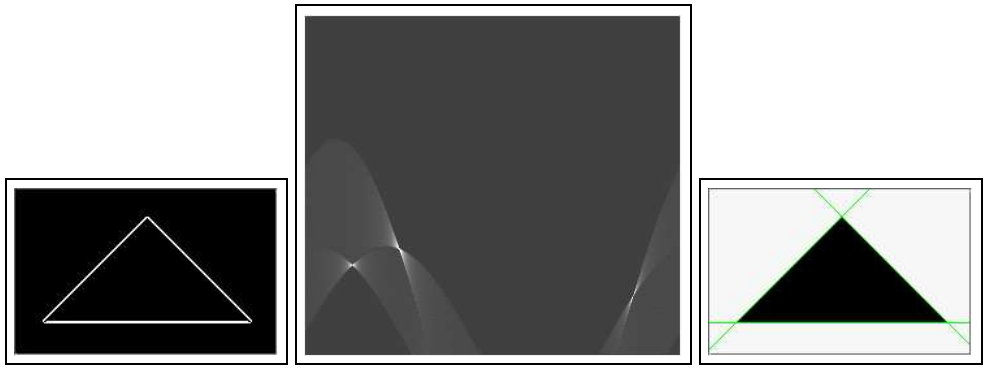

Abbildung 12

Kreisen mit bekanntem Radius: Mittelpunkte (x,y); im Fall von Kreisen mit unbekanntem Radius: (x,y,r)). Diese Darstellung bestimmt die Dimension des Akkumulatorraums.
– Bildpunkte in Akkumulatorraum transformieren (Wenn dieser Bildpunkt zur gesuchten Form gehört, welche Hinweise auf diese Form muss ich im Akkumulatorraum dann sammeln?)
– Suche nach lokalen Maxima im Akkumulatorraum.
– Ausgabe der gefundenen Formen.

Will man Geraden finden, zeigt sich schnell, dass die naheliegende Darstellung (m,n) (aus $y = mx+n$) ungeeignet ist, da m und n (und damit der Akkumulatorraum) beliebig groß werden können (z. B. bei senkrechten Geraden). Stattdessen geht man zur Hesseschen Normalenform $l = x\cos\rho + y\sin\rho$ über. Die Gerade ist die Menge aller Punkte (x,y), die diese Gleichung erfüllen. ρ ist der Winkel zwischen der y-Achse und der Gerade, l ist der Abstand der Geraden zum Ursprung. Man hat also einen zweidimensionalen Akkumulator (ρ,l): $\rho \in [0,\ldots,2\pi]$, $r \in [0,C]$ (wobei C die Diagonallänge des Bildes ist).

Ein Kantenpunkt kann auf unendlich vielen Geraden liegen. Also müssen für jeden Kantenpunkt alle sinnvollen Werte für ρ durchlaufen werden, um daraus l zu berechnen und den Hinweis im Akkumulatorraum speichern zu können. So entsteht für jeden Kantenpunkt eine sinusförmige Kurve im Akkumulatorraum.

Lokale Maxima im Akkumulatorraum sind dann (ρ,l)-Paare, die eine Gerade festlegen.

Abbildung 12 zeigt ein simples Beispiel: links das Kantenbild, in der Mitte der Akkumulatorraum und rechts das Ergebnis. Das Akkumulatorbild zeigt den Winkel in horizontaler Richtung

Abbildung 13 (Quelle: [5])

und die Entfernung zum Ursprung (der links oben liegt!) in vertikaler Richtung. In diesem sehr einfachen Beispiel kann man gut den Zusammenhang zwischen Akkumulatorraum und Ergebnis erkennen: Das erste Maximum entspricht der Gerade im 45° Winkel (linke Schräge), das zweite der Gerade im 90° Winkel (Unterseite) und das dritte der Gerade im 315° Winkel (rechte Schräge).

Auch bei der Detektion von Geraden ist es möglich, Kantenrichtungsinformation auszunutzen, um nicht alle Werte von ρ berücksichtigen zu müssen.

Für die Detektion beliebiger Formen steht die verallgemeinerte Houghtransformation zur Verfügung. Sie ermöglicht das Auffinden von Formen, für die keine mathematische Beschreibung vorliegen muss. Stattdessen genügt zur Beschreibung der Form eine Folge von diskreten Punkten[5].

3.1.3. Snakes

Snakes sind auch bekannt unter den Namen deformable contours, active contours oder minimum energy contours[5]. Wie man aus den alternativen Namen schon erahnen kann, beschreiben Snakes eine Klasse von Verfahren, die Konturen von Objekten erkennen kann. Diese Kontur wird ausgehend von einer initialen Kontur unter Minimierung einer so genannten Energiefunktion bestimmt.

Das linke Bild in Abbildung 13 zeigt die intiale Kontur um eine Cola-Dose, während das rechte Bild die detektierte Kontur zeigt.

Die parametrische Beschreibung einer Kontur ist $V(s) = (x(s), y(s)), \quad s \in [0,1]$.
$x(s)$ und $y(s)$ stehen für die Koordinaten entlang der Kontur.

Die Energiefunktion besteht aus einer Kombination interner und externer Kräfte und ist folgendermaßen aufgebaut:

$$E_{Kontur} = \int [\underbrace{(\alpha(s) \cdot E_1 + \beta(s) \cdot E_2)}_{\text{interne Kräfte}} + \underbrace{\gamma(s) \cdot E_3}_{\text{externe Kräfte}} \ ]$$

Wobei $E_1 = \|dv/ds\|^2$, $E_2 = \|d^2v/ds^2\|^2$ und $E_3 = -\|\nabla f(x,y)\|^2$. Ein großes $\alpha(s)$ bewirkt ein Zusammenziehen der Kontur, während ein großes $\beta(s)$ das Streben zu einer kreisförmigen

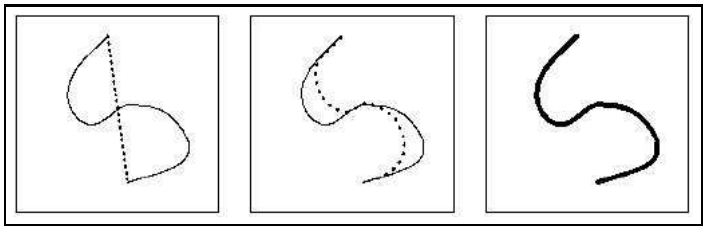

Abbildung 14 (Quelle: [5])

Gestalt unterstützt. Beide Parameter müssen aber abhängig von der Form der zu detektierenden Kontur festgelegt werden[5]. Zur näheren Betrachtung schauen wir uns ein Grauwertbild $f(x,y)$ mit Kantenstärke $s(x,y)$ und einer initialen Kontur $p_1,\ldots,p_n$ an. Zur Berechnung der Snake betrachtet man zuerst die Nachbarschaft $N(p_i)$ von p_i und sucht den Punkt p, der die Energiefunktion minimiert. Ist p gefunden, so ersetze p_i durch p. Dann sucht man nach möglichen Ecken, die sich als lokale Maxima der Krümmung k_i mit $k_i = \|p_{i-1} + p_{i+1} - 2p_i\|$ zeigen. Ist eine Ecke gefunden, so wird $\beta(s)$ für die nächste Iteration auf 0 gesetzt, wodurch man stückweise glatte Kurven bekommt[5]. Die Iteration endet, wenn keine Veränderung der Punkte mehr festzustellen ist. Der Nachteil von Snakes ist, dass man die initiale Kontur sozusagen „von Hand“ festlegen muss. Wollte man eine Snake zum Beispiel in ein automatisches System einbauen, so bräuchte man noch ein Programm, das die Form und Lage der initialen Kontur festlegt. Man müsste also vorher schon ein Objekterkennungssystem über das Bild laufen lassen, um das Objekt, dessen Kontur bestimmt werden soll, erst einmal zu finden.

Im praktischen Einsatz befinden sich Snakes unter anderem in der Objektverfolgung und in der Medizin.

3.2. Erscheinungsbasierte Methoden

Die Entwicklung von erscheinungsbasierter Erkennung ist aus der Frage hervorgegangen, ob die Form, also die geometrischen Eigenschaften, eines Objekts zur Erkennung dessen ausreichen. Vorherige Ansätze haben nur geometrische Merkmale wie Ecken, Kanten oder Flächeninhalt berücksichtigt. Doch was ist mit wichtigen nicht-geometrischen Merkmalen wie Farbe oder Reflektanz? Erscheinungsbasierte Methoden nutzen genau diese Merkmale, betrachten dafür aber keine geometrischen Eigenschaften. Grundlage hier ist das ganze Objekt. Sie umgehen so Nachteile der modellbasierten Erkennung, wie Segmentierungsfehler und Informationsverlust aufgrund der Beschränkung auf geometrische Merkmale. Ein weiterer Vorteil der erscheinungsbasierten Methoden ist, dass keine Modelle generiert werden müssen. Dies ist vor allem bei Objekten mit komplexer Oberfläche vorteilhaft, da hier eine Modellgenerierung sehr komplex und zeitaufwendig wäre.

Von uns vorgestellte erscheinungsbasierte Methoden sind Eigenspaces und Histogramme.

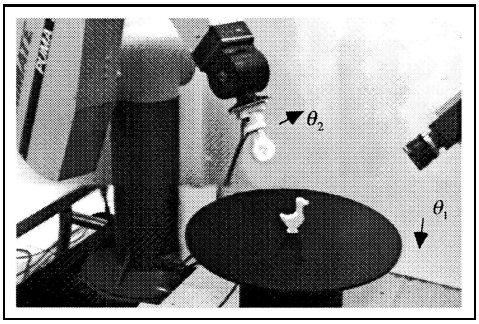

Abbildung 15 (Quelle: [9])

3.2.1. Eigenspace

Wir suchen einen Ansatz, der das Objekt als Ganzes repräsentiert, also als eine Kombination aus Reflektanz, Form, Belichtungsbedingungen und Ausrichtung im Bild. Die Repräsentation eines Objekts ist natürlich abhängig von dessen Ausrichtung zur Kamera. Um aus diesem Grund eine gute Datengrundlage zu schaffen, wird das Objekt auf eine rotierende Scheibe gestellt und aus allen Richtungen von einer Kamera aufgenommen (siehe Abbildung 15)[9].

Wichtig ist, dass die aufgenommenen Bilder hier keine Matrizen sind, wie man es von Bildern gewöhnt ist, sondern Vektoren. Ein Bild mit $N \times N$ Pixeln ist also ein Punkt im N^2-Raum. Ein Bild mit der Auflösung 256×256 Pixel wäre also ein Punkt im Raum der Dimension 65536. Jede Bilderfolge wird mit unterschiedlichen Freiheitsgraden aufgenommen. Ein Freiheitsgrad beinhaltet zum Beispiel die Objektausrichtung oder den Grad der Beleuchtung. Da zu jedem Freiheitsgrad viele Bilder aufgenommen werden, ist der Speicherverbrauch ziemlich groß. Wir brauchen also eine Kompression, welche gerade durch unseren Eigenspace geliefert wird. Die Idee hierzu ist die kompletten Folgen von Bildern in einen speziellen Raum (den Eigenspace) zu transformieren, in dem Bilder, die eine hohe Korrelation zu anderen Bildern aufweisen weggelassen werden. Hierzu werden zu jeder Bilderfolge die Eigenvektoren generiert, die dann den Eigenspace aufspannen. Der Eigenspace ist also praktisch der kleinste Raum, der das Objekt enthält. Anschließend wird jedes Bild aus der Bilderfolge mit den Eigenvektoren multipliziert, wodurch jedem Bild ein Punkt im Eigenspace zugeordnet wird. Der Eigenspace ist jetzt mit Punkten gefüllt, die diskrete Bilder aus der Bilderfolge repräsentieren. Um jetzt eine kontinuierliche Funktion zu bekommen, interpoliert man über die diskreten Werte.

Abbildung 16 zeigt solche Funktionen, die zur Veranschaulichung dreidimensional dargestellt sind. Wir besitzen jetzt eine Repräsentierung des Objekts.

Wollen wir nun ein Objekt in einem unbekannten Bild erkennen, so müssen wir zuerst das zu bestimmende Objekt extrahieren. Das so bearbeitete Bild wird ebenfalls in den Eigenspace transformiert. Der Punkt im Eigenspace, der am nächsten zu dem des aufgenommenen Bildes liegt, ist dann das Objekt im aufgenommenen Bild.

Vorteil der Eigenspace-Methode ist, dass das ganze Auftreten des Objekts berücksichtigt wird, also kein Informationsverlust vorliegt, der zum Beispiel beim Extrahieren von geometrischen Merkmalen auftreten würde. Nachteilig ist, dass diese Methode anfällig für partielle Über-

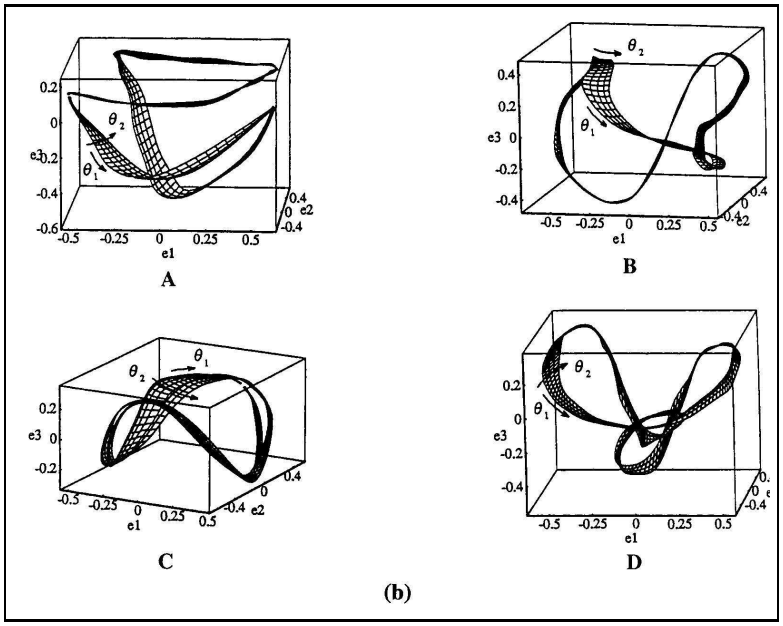

Abbildung 16 (Quelle: [9])

deckungen ist und dass das zu bestimmende Objekt erst aus dem Suchbild extrahiert werden muss.

3.2.2. Histogramme

Ein anderer erscheinungsbasierter Ansatz ist die Erkennung mit Histogrammen[3]. Auch hier löst man sich komplett von der Repräsentierung eines Objekts anhand von Formen, also zum Beispiel durch Kanten, und nutzt stattdessen die farblichen Eigenschaften eines Objekts. Farbbasierte Erkennungsmethoden eignen sich hervorragend für Objekte mit einer komplexen Oberfläche, also solche Objekte, an denen schwer geometrische Merkmale wie Ecken oder Kanten zu extrahieren sind. Die Verwaltung der Farben erfolgt in einem Histogramm. Ein Farbhistogramm speichert die Anzahl von Pixeln eines Bildes, die alle einen bestimmten gleichen Farbwert besitzen. Für das Histogramm H_M eines Bildes M gilt: $\forall i \in [0, x_{\max}], j \in [0, y_{\max}] : M(i,j) = (r,g,b)^t \Rightarrow H_M(r,g,b) = H_M(r,g,b) + 1$. Im Prinzip werden also alle Pixel eines Bilds durchlaufen und zu dem Farbwert des Pixels wird im Histogramm die Anzahl um eins erhöht. In der Praxis ist ein so berechnetes Histogramm sehr groß: bei einer Farbtiefe von 8 bit hätte man $n = 256$ Einträge zu speichern. Aus diesem Grund werden die Farben meist zu Klassen gleicher Farben zusammengefasst. Dies geschieht häufig durch ein K-means clustering, in dem die repräsentativsten Farben extrahiert werden. Auf diese Weise wird die Größe des Histogramms reduziert und es wird gleichzeitig robuster gegenüber wechselnden Lichtbedingungen.

Abbildung 17 zeigt so ein Histogramm, das auf 14 Farben reduziert wurde.

Der eigentliche Erkennungsprozess ist dann ganz einfach. Es werden einfach die Histogramme eines Objekts M mit einem unbekanntem Bild I verglichen. Dieser Vergleich wird mit der so genannten Histogramm Intersection durchgeführt. Hierbei wird der Grad der Übereinstimmung

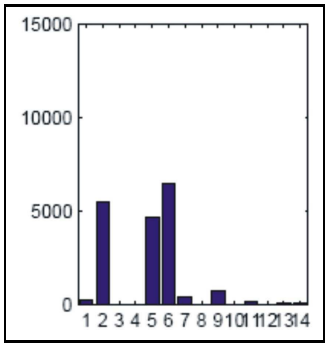

Abbildung 17 (Quelle: [10])

V zwischen den Histogrammen H_I und H_M berechnet:

$$V(H_I, H_M) = \frac{\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\min(H_I(i,j,k),H_M(i,j,k))}{\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}H_M(i,j,k)}$$

Bei anderer Auflösung oder Orientierung des Objekts im Bild und bei teilweiser Objektverdeckung ändert sich V in einem so geringen Maß, dass eine Objektidentifizierung noch möglich ist. Es ist also prinzipiell möglich ein dreidimensionales Objekt mit einer geringen Anzahl von Histogrammen (ca. 6), aufgenommen aus verschieden Blickwinkeln, zu bestimmen.

Vorteil der Histogramm-basierten Erkennung ist also: Sie ist rotations- und translationsinvariant und robust gegenüber Größenänderungen und teilweisen Verdeckungen. Des Weiteren ist sie leicht zu implementieren und hat ein gutes Laufzeitverhalten. Trotzdem sind aber auch große Nachteile zu nennen. So ist die Histogramm-basierte Erkennung im hohen Maß von Beleuchtungsbedingungen abhängig. Ändert sich die Beleuchtung, so ändert sich auch die Farbe. Es ändert sich also das Hauptvergleichsmerkmal und somit sinkt die Wahrscheinlichkeit der Erkennung rapide. Außerdem ist sie nur auf Objekte mit einer großen Farbvielfalt beschränkt. Objekte mit homogener Farboberfläche können zwar auch erkannt werden, allerdings darf die Farbe des Objekts nicht im Hintergrund vorkommen. Das größte Problem ist aber, dass selbst unter besten Bedingungen Objekte mit komplett unterschiedlichen Geometrien das gleiche Farbhistogramm haben können. Um diese Nachteile in Grenzen zu halten, hat man eine Erweiterung in die Histogramm-basierte Erkennung eingeführt: das Color Coocurrence Histogramm.

3.2.3. Color Cooccurrence Histograms

Das Color Coocurrence Histogramm (CCH) integriert einen Teil der Geometrie eines Objekts in sein Histogramm[1, 2]. Zu diesem Zweck wird der Abstand von Pixeln des Objekts in das Histogramm integriert. Ein CCH speichert die Anzahl von Paaren von Pixeln mit einer bestimmten Farbe und einem bestimmten Abstand zueinander. Es werden also die relativen geometrischen Positionen von Pixeln gespeichert.

Abbildung 18 zeigt ein Bildbeispiel hierzu. Das Pixelpaar $c_1 = (R_1, G_1, B_1)$ und $c_2 = (R_2, G_2, B_2)$ würde mit dem Abstandsvektor $(\Delta x, \Delta y)$ im CCH gespeichert werden. Man könnte das Color

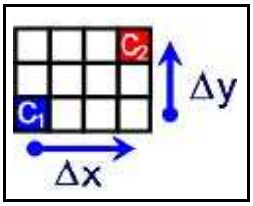

Abbildung 18 (Quelle: [1])

Abbildung 19 (Quelle: [2]

Coocurrence Histogramm CH also symbolisch als Funktion $CH(c_1, c_2, \Delta x, \Delta y)$ schreiben. Für den Fall $(\Delta x, \Delta y) = (0,0)$ und $c_1 = c_2$ entspricht das CCH einem einfachem Farb-Histogramm. Bevor die CCHs berechnet werden, werden auch hier die Farben klassifiziert. Grund für die Beschränkung auf die Hauptfarben ist wie bei dem Farb-Histogramm die einfachere Berechnung und der geringere Speicheraufwand. Entscheidend für den Erfolg der späteren Erkennung ist die Wahl der Anzahl der Farben und Distanzen. Sie können durch ein mathematisches Modell bestimmt werden.

Die eigentliche Erkennung läuft folgendermaßen ab: Zuerst tastet ein Suchfenster das Bild zeilenweise ab, wobei sich die Suchfenster bis zu 50% überlappen können. Zu jedem Suchfenster wird ein coocurrence Histogramm generiert. Dann wird das Histogramm h_s jedes Suchfensters mit dem Histogramm h_o des zu findenden Objekts verglichen, indem man zum Beispiel durch die oben schon erwähnte Histogram Intersection Methode den Grad der Übereinstimmung berechnet. Diesen Wert kann man dann in einer so genannten Vote-Matrix darstellen.

Abbildung 19 zeigt ein Beispiel. Hier soll die orange Reisverpackung gefunden werden. Das linke Bild ist das Eingabebild. Das rechte Bild zeigt die dazugehörige Vote-Matrix. Durch die Bestimmung des lokalen Maximums in der vote-Matrix wird ein Kandidatenfenster bestimmt, das das Objekt enthält. Allerdings enthält es nicht das komplette Objekt, sondern nur Teile davon, weswegen eine Expansion auf benachbarte Fenster durchgeführt wird. Der Expansionsprozess stoppt, wenn der Grad der Übereinstimmung einen bestimmten Schwellenwert unterschreitet.

In Abbildung 20 soll Woody aus dem Film Toy Story erkannt werden. An diesem Beispiel sieht man sehr gut die Vorteile der CCH. Sie sind robust gegenüber Größenänderungen bzw. Verzerrungen. Außerdem sieht man sehr gut, dass sie robust gegenüber teilweiser Verdeckung und Hintergrundstörungen sind. Die Nachteile sind ähnlich zu denen des einfachen Farbhistogramms.

Abbildung 20 (Quelle: [1])

4. Matching und Lernen

Jedes Verfahren der Objekterkennung benötigt spezielle Verfahren zum Vergleich von Objektbildern und Suchbildern, da jedes Verfahren verschiedene Merkmale extrahiert. Trotzdem lassen sich einige Methoden mit Gemeinsamkeiten finden. Im Folgenden soll ein grobes gemeinsames Vorgehen skizziert werden. Dieses Vorgehen läßt sich zwar nicht auf alle Verfahren übertragen (auch nicht auf alle hier vorgestellten), aber es zeigt, dass in der Objekterkennung Methoden aus vielen Bereichen der Informatik angewendet werden.

4.1. Erscheinungsbasierte Methoden

Einige erscheinungsbasierte Methoden extrahieren zunächst Merkmale aus Trainingsbildern der Objekte, die erkannt werden sollen, und erzeugen für jedes Trainingsbild einen mehrdimensionalen Vektor in einem Merkmalsraum.

Mehrere Bilder desselben Objekts in verschiedenen Lagen bilden eine Punktwolke in diesem Merkmalsraum. Verschiedene Objekte bilden verschiedene Punktwolken.

Der aus dem Suchbild erstellte Merkmalsvektor liegt ebenfalls in diesem Raum. Es geht jetzt darum, ihn derjenigen Punktwolke zuzuordnen, deren Objekt dem Suchbild am ähnlichsten ist (will man nicht nur ein Objekt identifizieren, sondern auch seine Lage bestimmen, muss man den Vektor sogar dem nächsten Punkt zuordnen).

Man benötigt also ein Ähnlichkeitsmaß. Kann man für den Merkmalsraum ein Ähnlichkeitsmaß definieren, das die Eigenschaften eines Abstands im Merkmalsraum erfüllt, lassen sich für diese Aufgabe Methoden der →*Mustererkennung* anwenden (Nearest Neighbour, . . .).

4.2. Modellbasierte Methoden

Ein Modell eines Objekts kann z. B. ein Attributgraph sein, in dem jeder Knoten einen Teil des Objekts beschreibt. Ein Flächenknoten könnte z. B. die Art der Fläche (Rechteck, Kreis, Ellipse,

. . .) und den Flächeninhalt speichern. Die Kanten des Graphen würden dann Verbindungen einer Fläche zu einer anderen Fläche darstellen. Ein Würfel würde in diesem Fall aus sechs gleichen Flächenknoten und zwölf Kanten zwischen diesen Knoten bestehen.

Solche Modelle automatisch zu erstellen ist nicht einfach, wenn nur 2D-Daten zur Verfügung stehen. Zum Teil werden die Modelle deshalb aus aufwändig aufgenommenen 3D-Daten (→*3D-Messtechnik*; →*Modellerstellung*) oder aus →*CAD* Systemen erstellt.

Zum Vergleich der Modelle mit einem Suchbild gibt es viele Möglichkeiten; man kann z. B. versuchen, ein Objektmodell möglichst passend in das Suchbild zu projizieren. Gelingt das, gilt das Objekt als gefunden.

Naheliegender (wenn auch schwieriger) ist es, auch aus dem Suchbild ein Modell zu erstellen. Der Vergleich zwischen Objektmodell und Bildmodell muss dann mit Methoden des→*Graph-matching* erfolgen.

4.3. Lernen

Objekterkennungssysteme müssen die Objekte, die sie erkennen sollen, „lernen“. Oft geschieht das, indem sie Merkmale / Modelle aus einem Objektbild extrahieren (z. B. ein CCH) und speichern. Es gibt aber auch andere Wege.

Andere Objekterkennungsysteme „lernen“ keine konkreten Objekte im Voraus, sondern extrahieren (oft relativ komplexe und abstrakte) Merkmale aus den Suchbildern und entscheiden dann anhand einer Wissensbasis, ob das Objekt enthalten ist oder nicht.

Diese Wissensbasis kann z. B. aus einem neuronalen Netz bestehen, das zuerst trainiert werden muss (→*Neuronale Netze*). Dazu werden dem System Trainigsbilder vorgelegt. Das System berechnet eine Ausgabe zu jedem Bild. Aufgrund dieser Ausgabe und dem wirklichen Ergebnis ist das System in der Lage, sein Neuronales Netz anzupassen und zu verbessern. Der „Lernerfolg“ kann anhand von Testbildern überprüft werden. Das Wissen des Systems hat nicht mehr die Form von gespeicherten Merkmalen, sondern besteht im trainierten Neuronalen Netz.

Es können auch Methoden der →*Statistik* angewendet werden, z. B. Varianten des Bayes-Klassifikators[7].

5. Einsatz von Objekterkennungssystemen in der Landminendetektion

Es existieren einige Ansätze zur Objekterkennung in der Landminendetektion. Wir haben exemplarisch einen Ansatz ausgesucht, der mit Infrarotbildern arbeitet, die aus der Luft aufgenommen werden[8]. Die Idee hierbei ist, dass der Boden über einer Landmine über Tag wärmer

und über Nacht kälter ist als der umliegende Boden. Diese Wärmeunterschiede lassen sich sehr gut mit der Infrarotkamera nachvollziehen. Abbildung 21 zeigt ein Beispielbild hierzu.

Zur Merkmalsextraktion werden zwei Methoden benutzt. Zum einen die schon bekannte Hough-Transformation und zum anderen der Tophat-Filter. Der Tophat-Filter extrahiert lokale warme Objekte, die im aufgenommen Bild hell sind, und lokale kalte Objekte, die dunkel dargestellt werden. Es gibt einen weißen Tophat-Filter, der helle Stellen im Bild extrahieren kann und eine schwarzen Tophat-Filter, der dunkle Stellen im Bild identifizieren kann. Berechnet wird der weiße Tophat über $WTH(f) = f - f \circ B$, wobei f das Eingangsbild ist und B ein strukturierendes Element ist, das alle Bildstrukturen, die kleiner als B sind, aus dem Bild entfernt. Durch die Subtraktion werden dann genau die Bildstrukturen extrahiert. Der schwarze Tophat wird ähnlich definiert: $BTH(f) = f \circ B - f$.

Man nutzt drei Kostenfunktionen um drei verschiedene Parametereinstellungen für Hough und Tophat zu berechnen. Kostenfunktionen berechnen, wie der Name schon sagt, Kosten für gewisse Operationen, zum Beispiel beinhaltet eine Kostenfunktion die Kosten für einen falschen Alarm.

Daraus resultieren dann je drei Vertrauenswerte, die als Objekte in dem späteren Erkennungsprozess fungieren. Im Erkennungsprozess kombiniert man alle sechs Objekte und extrahiert die besten Kombinationen, was mit den zwei Methoden Mahalanobis Distances und Fisher Mapping durchgeführt wird, worauf aber nicht weiter eingegangen werden soll. Abbildung 22 zeigt den Ablauf schematisch während in Abbildung 23 von links nach rechts die Ergebnisse von Hough, Tophat, Mahalanobis und Fisher dargestellt sind.

6. Fazit

Wie wir in den oben stehenden Kapiteln sehen können, gibt es nicht *das* Objekterkennungssystem. Die Wahl des Systems ist von vielen Faktoren abhängig, unter anderem vom Ziel, von dem Objekt (Farbe, Form, Größe, etc.), von den Bedingungen (teilweise Verdeckung, Hintergrund, Belichtung, etc.) und von den Daten (Wie aufgenommen?, Welche Auflösung?). Es gibt viele verschiedene Ansätze, die jeweils komplett unterschiedlich sein können, zum Beispiel Hough-Transformation und Histogramme.

Obwohl schon einige Systeme entwickelt sind, muss immer noch Grundlagenforschung betrieben werden in Fragen wie: Wie funktioniert Objekterkennung beim Menschen? Was lässt sich von seiner Denkweise auf den Computer übertragen? Wie viele Ansichten müssen gespeichert werden bzw. gibt es einen besten Blickwinkel? Zu diesem Punkt haben Forscher zum Beispiel herausgefunden, dass es besondere kanonische Blickwinkel gibt, aus denen Menschen Objekte schneller erkennen als aus anderen.

Trotzdem gibt es auf diesem Gebiet aber auch Fortschritte. Hierzu sei auf schon bestehende Erkennungssysteme wie Sprach- oder Texterkennung oder die automatische Mülltrennung verwiesen.

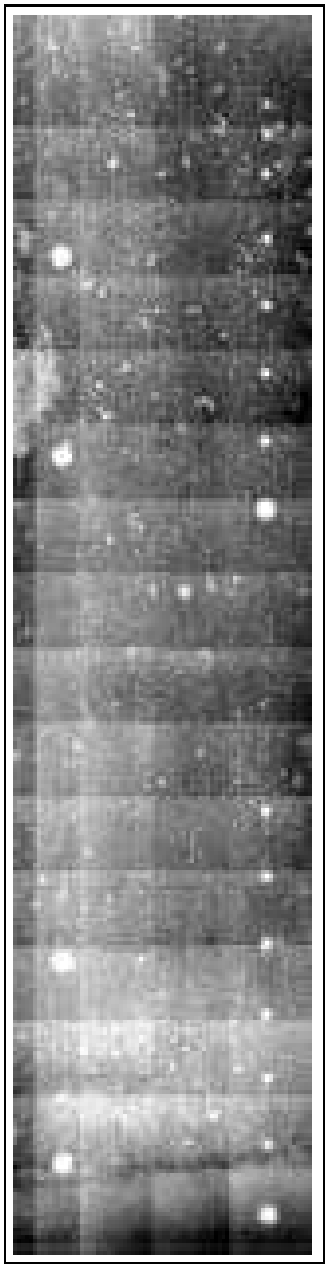

Abbildung 21 (Quelle: [8])

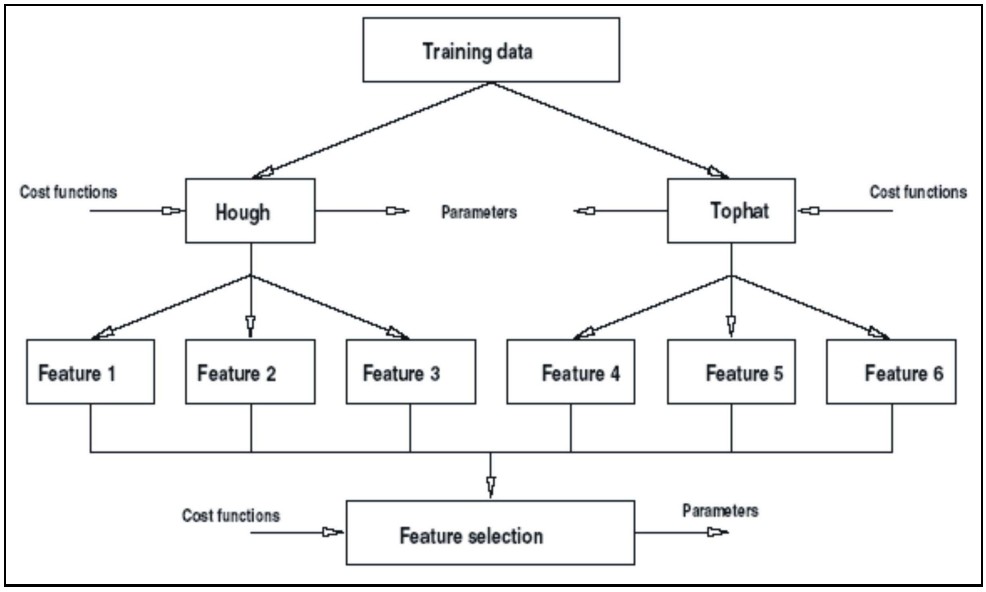

Abbildung 22 (Quelle: [8])

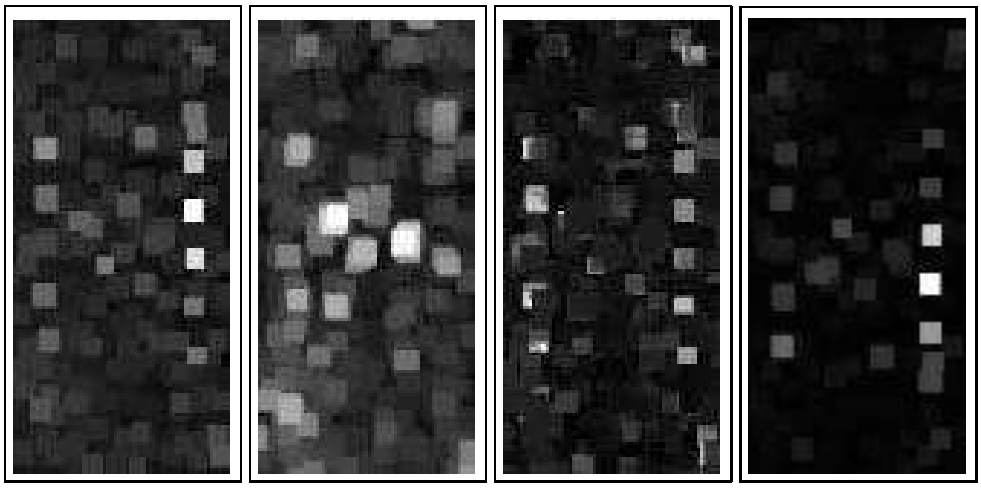

Abbildung 23 (Quelle: [8])

Mit einer sich verbessenderen Technik wird in Zukunft Objekterkennung eine immer größere Rolle spielen. Es wird auf absehbare Zeit zwar kein HAL erschaffen werden, aber wir können uns schon darauf einstellen, dass Computer immer weiter zu unserem Erkennungsvermögen aufschließen werden.

Literatur

[1] P. Chang, J. Krumm: Object Recognition with Color Cooccurrence Histograms, IEEE Conference on Computer Vision and Pattern Recognition, Fort Collins, CO, June 23 – 25, 1999, http://research.microsoft.com/users/jckrumm/Publications/Color%20Cooccurrence/CVPR1.pdf, 3.11.2003

[2] S. Ekvall, F. Hoffmann, D. Kragic: Object Recognition and Pose Estimation for Robotic Manipulation using Color Cooccurrence Histograms, keine näheren Angaben, http://esr.e-technik.uni-dortmund.de/Hoffmann/iros2003.pdf, 3.11.2003

[3] W. Hafner: Segmentierung von Video-Bildfolgen durch Adaptive Farbklassifikation, München, 1998, http://www.augusta.de/~hafner/diss/diss.html, 20.11.2003

[4] M. Höfler: Einführung – Wie Computer sehen können, Proseminar Neuroinformatik, Ulm 2002, http://www.informatik.uni-ulm.de/ni/Lehre/SS02/Proseminar_CV/ausarbeitungen2/mhoefler.pdf, 23.12.2003

[5] X. Jiang: Vorlesung Bildanalyse, Münster 2003

[6] J. C. Liter, H. H. Bülthoff: An Introduction to Object Recognition, Max-Planck-Institut für biologische Kybernetik, Technical Report No. 43, Nov. 1996, ftp://ftp.mpik-tueb.mpg.de/pub/mpi-memos/TR-043.ps

[7] D. Keysers, J. Dahmen, H. Ney, M. O. Güld: A Statistical Framework for Multi-Object Recognition, Informatiktage 2001 der Gesellschaft für Informatik, Bad Schussenried, Germany, pages 73 – 76, October 2001.

[8] W. Messelink, K. Schutte, A. Vossepoel, F. Cremer, J. Schavemaker, E. den Breejen: Featurebased detection of landmines in infrared images, PreprintProc. SPIE Vol. 4742, Det. and Rem. Techn. for Mine and Minelike Targets VII, Orlando FL, USA, Apr. 2002, http://citeseer.nj.nec.com/messelink02featurebased.html, 3.11.2003

[9] S. Nayar, H. Murase, S. Nene: Parametric Appearence Representation, Oxford University Press, 1996, http://citeseer.nj.nec.com/nayar96parametric.html, 15.11.2003

[10] S. Redfield, M. Nechyba, J. G. Harris, A. A. Arroyo: Efficient Object Recognition Using Color, Florida Conference on Recent Advances in Robotics, May 2001.