

# Diplomarbeit

## Stereoskopische Lokalisierung und Erkennung von Objekten im Greifraum eines autonomen Montageroboters

Christoph Theis

Bochum, 15.01.2001

Prof. Dr.-Ing. W. v. Seelen  
Institut für Neuroinformatik  
Lehrstuhl für Theoretische Biologie  
Ruhr-Universität Bochum

Prof. Dr.-Ing. Y. Tüchelmann  
Fakultät für Elektrotechnik  
Lehrstuhl für Datenverarbeitung  
Ruhr-Universität Bochum



Hiermit versichere ich, daß die vorgelegte Arbeit von mir selbständig verfaßt wurde und ich keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

---

Ort, Datum

---

Christoph Theis



# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
<b>2</b>	<b>Cora - Ein autonomes Assistenzsystem</b>	<b>3</b>
2.1	Zielstellung . . . . .	3
2.2	Der Aufbau . . . . .	5
2.3	Anforderungen an die Umgebung . . . . .	5
2.4	Grenzen dieser Arbeit . . . . .	6
<b>3</b>	<b>Filter und Modifikatoren</b>	<b>8</b>
3.1	Kantenfilter . . . . .	8
3.1.1	Einfaches Schwellwertverfahren . . . . .	8
3.1.2	Prewitt Operator . . . . .	10
3.2	Erosion und Dilatation . . . . .	10
3.3	Opening und Closing . . . . .	12
3.4	Medianfilter . . . . .	14
3.5	Linsenkorrektur . . . . .	15
3.6	Das räumliche Sehen . . . . .	17
3.6.1	Das Sehen im biologischen System Mensch . . . . .	17
3.6.2	Augenbewegung und Tiefensehen . . . . .	17
3.6.3	Sehen in technischen Systemen . . . . .	18
3.6.4	Disparitätsfindung . . . . .	19
3.7	Clustern . . . . .	21
<b>4</b>	<b>Objekte lernen und erkennen</b>	<b>25</b>
4.1	Szenenbeschränkung . . . . .	26
4.1.1	Bereiche der Szene . . . . .	26
4.1.2	Wissensbasierte Bereichsextraktion . . . . .	26
4.2	Farbbasierte Filterung . . . . .	29
4.2.1	Der HSI-Farbraum . . . . .	29
4.2.2	Ermittlung der Farbeigenschaften eines Objektes . . . . .	30
4.2.3	Anwendung der Farbeigenschaften eines Objektes . . . . .	31

4.3	Berechnung der Raumkoordinaten aus den Disparitätswerten . . . . .	32
4.4	Bilden eines gegenstandsbezogenen Objektes . . . . .	33
4.4.1	Ausrichtung bestimmen . . . . .	34
4.4.2	Objekt durch Drehung normieren . . . . .	35
4.5	Bestimmung der Objektlage . . . . .	37
4.5.1	Bereichsbewertung durch Unschärfe . . . . .	37
4.5.2	Bestimmung der Position . . . . .	38
4.6	Ablauf einer Lernphase . . . . .	39
4.7	Ablauf einer Suchphase . . . . .	40
<b>5</b>	<b>Beispiele</b>	<b>43</b>
5.1	Beispiel Lernobjekt Schwarze Zange . . . . .	43
5.2	Beispiel Lernobjekt Schraubendreher . . . . .	48
5.3	Beispiel Lernobjekt Tesarolle . . . . .	52
5.4	Beispiel Suchbereich A . . . . .	55
5.5	Beispiel Suchbereich B . . . . .	60
5.6	Beispiel Suchbereich C . . . . .	64
5.7	Beispiel Suchbereich D . . . . .	68
<b>6</b>	<b>Zusammenfassung und Ausblick</b>	<b>73</b>
<b>A</b>	<b>Anhang</b>	<b>75</b>
A.1	Programmbausteine . . . . .	75
A.1.1	Despeckle.h . . . . .	75
A.1.2	DespeckleLinear.h . . . . .	75
A.1.3	Dilatation.h . . . . .	76
A.1.4	ErkenneDingDa.h . . . . .	76
A.1.5	Erosion.h . . . . .	79
A.1.6	FaerbDisp.h . . . . .	79
A.1.7	FindDisp.h . . . . .	80
A.1.8	FindeFarben.h . . . . .	80
A.1.9	FindeKanten.h . . . . .	81
A.1.10	LadeDaten.h . . . . .	81
A.1.11	LeseBildDaten.h . . . . .	82
A.1.12	LinseEntzerren.h . . . . .	82
A.1.13	MinimiereAusschnitt.h . . . . .	83
A.1.14	ObjektDrehen.h . . . . .	83
A.1.15	PicPoint.h . . . . .	84
A.1.16	PositionBestimmen.h . . . . .	85
A.1.17	TischAusblenden.h . . . . .	85
A.1.18	TischMaskieren.h . . . . .	85

A.2	Kameras . . . . .	87
A.3	Benutzte Rechner . . . . .	89
A.3.1	Rechner für Testläufe . . . . .	89
A.3.2	Rechner Cora . . . . .	89



# Kapitel 1

## Einleitung

Rechnerbasierte Verarbeitung von visuellen Daten ist bereits seit vielen Jahren ein intensives Forschungsthema. Ziel vieler Projekte ist eine Erfassung der Umwelt, um mit dieser interagieren zu können (siehe [1] und [2]). Ein solches Vorhaben wurde am Institut für Neuroinformatik der Ruhr Universität Bochum bereits im Projekt NEUROS durch den autonomen und mobilen Serviceroboter ARNOLD realisiert. Die Aufgabenstellung begrenzte sich hierbei auf die Wahrnehmung der Umgebung mittels eines Stereokamerapaars, kombiniert mit einer Sprachbefehlssteuerung zur Erfassung von Kommandos. Auf sonst übliche Zusatzsensoren, wie Laserscanner oder Ultraschallarrays, wurde verzichtet, da die Umsetzung mit den Beschränkungen des menschlichen Vorbildes erfolgen soll.

Im Folgeprojekt MORPHA wurde CORA (COoperative Robot Assistant) entwickelt. Dieser Roboter entspricht in seinen Grundzügen dem Aufbau von Arnold im Projekt NEUROS, nur daß es sich hier um einen stationären Aufbau handelt. Ziel ist die Entwicklung eines mit seiner Umwelt autonom und interaktiv agierenden stationären Assistenzsystems. Der geplante Einsatzort ist ein Fließband, an dem ein Arbeiter unterstützt werden soll. CORA soll dabei selbstständig entscheiden, welche Handgriffe zur Unterstützung seines menschlichen Kollegen nötig sind.

Als Element dieses komplexen Szenarios behandelt diese Arbeit die Entwicklung eines Softwaremoduls, das den Roboter in die Lage versetzt, Objekte, die separiert auf der Arbeitsfläche präsentiert werden, zu lokalisieren, ihr Aussehen in Bezug auf Form, Farbe und Textur zu speichern und diese Objekte später in komplexeren Szenen wiederzuerkennen und dort ebenfalls exakt zu lokalisieren.

Zur Erkennung eines Gegenstandes auf einer Arbeitsfläche wurde ein Ansatz gewählt, der einige Voraussetzungen an den Aufbau des Roboters und die Gegenstände auf der Arbeitsfläche stellt. Durch eine einfache Frontansicht

ließe sich ein Gegenstand, nicht eindeutig beschreiben, da je nach Drehwinkel um die Hochachse des Gegenstandes für die Kamera ein völlig neues Bild entsteht. Darüber hinaus wird der Gegenstand durch die Änderung der Entfernung skaliert. Dies zeigt, daß eine Erkennung direkt über die Ansicht mit vielen Problemen behaftet ist. Daher erfolgt die hier beschriebene Objekterkennung über eine berechnete Aufsicht aller Gegenstände, die sich auf der Tischfläche befinden. Dabei wird angenommen, daß wesentliche Teile zur Ermittlung einer Objektauf sicht unter jedem Rotationsgrad um die Hochachse sichtbar sind und somit die Kameras von schräg oben auf die Gegenstände herabblicken. Der Vorteil der Analyse der Tischszene über eine Aufsicht liegt in der einfachen Beschreibung eines Objektes. Bei jeder Position und konstanter Standfläche eines Gegenstandes auf der Tischfläche ergibt sich die gleiche Aufsicht, die lediglich rotiert sein kann.

Im Kapitel 2 wird näher auf den Aufbau von CORA eingegangen. Neben einer Erklärung der Funktionen werden auch die Grenzen aufgezeigt, die Aufbau, Umgebung und Software setzen.

Im Kapitel 3 werden Grundlagen der Bildverarbeitung, die in dieser Arbeit zur Anwendung kamen, beschrieben. Dazu gehören Kantenfilter, die zum einen auf einem einfachen Schwellwertverfahren und zum anderen auf maskenbasierter Kantenextraktion beruhen. Weiter wird auf Wachstums- und Schrumpfungsalgorithmen, wie Erosions- und Dilatationsfiltern, und deren Zusammenspiel eingegangen. Danach werden Methoden zur Bildkorrektur, wie Linsenentzerrer, zur Bildoptimierung und zur Bildgliederung erörtert. Im letzten Teil des Kapitels wird eine Methode zur Disparitätsfindung im Vergleich zum biologischen Sehprozeß erörtert.

Das Kapitel 4 bezieht sich verstärkt auf die einzelnen Verarbeitungsstufen, die zur Erkennung und Ermittlung der Position eines Gegenstandes nötig sind. Dazu gehören die Extraktion der Objekte aus den Kamerabildern, die Ermittlung und Anwendung einer Farbfilterstufe, die Ermittlung der aus Disparitätswerten resultierenden Raumpunkte, die Bildung eines Objektmodells und die Bestimmung der Lage eines Gegenstandes auf der Tischfläche. Abschließend werden die Abläufe eines Lern- und Suchvorganges dargestellt.

Im Kapitel 5 werden für drei Lern- und vier Suchphasen Beispiele gegeben, die die Leistungsfähigkeit und Grenzen der Algorithmen zeigen sollen. Dabei werden die einzelnen Phasen der Verarbeitung durch Bilder dokumentiert.

Im Kapitel 6 werden die Ergebnisse dieser Arbeit zusammengefaßt und bewertet.

Im Anhang finden sich die Beschreibungen der verwendeten Programme, die Kameraparameter und die Angabe der verwendeten Rechner.

# Kapitel 2

## Cora - Ein autonomes Assistenzsystem

### 2.1 Zielstellung

Mit dem mobilen Serviceroboter ARNOLD (siehe [1] und [2]) wurde ein autonomes System geschaffen, welches sich eigenständig in seiner Umwelt bewegen kann. Zur Orientierung in seiner Umwelt dient ein Stereokamerapaar und per Sprachbefehl können Verhaltensmuster initiiert werden. CORA (Cooperative Robot Assistent) ist eine Weiterentwicklung von ARNOLD, jedoch in Form eines stationären Aufbaus. Ziel soll es sein, daß Cora mit seinem Roboterarm interaktiv in einen Produktionsprozeß mit menschlicher Beteiligung eingreifen kann. Im Rahmen des MORPHA-Projektes wird zur Zeit ein solches Produktionsszenario geschaffen (siehe Abb. 2.1). CORA befindet sich dabei am Ende eines Fließbandes und soll darauf liegende Gegenstände bei Bedarf aufnehmen, um diese interaktiv mit seinem menschlichen Partner weiter zu verwerten. Eine solche Interaktivität stellt jedoch hohe Anforderungen an die Handlungsplanung des Greifarms. Zu jedem Zeitpunkt muß die Szene vollständig erfaßt sein, um Kollisionen mit der eingesetzten Hardware, besonders aber mit dem Menschen, zu vermeiden, bzw. um auf dem Arbeitsbereich liegende Gegenstände aufnehmen zu können. Die vorliegende Arbeit beschäftigt sich mit der Erzeugung eines Bildverarbeitungsmoduls, welches in der Lage ist, einzelne Gegenstände im Arbeitsbereich anhand von Merkmalen zu erlernen, diese in komplexeren Szenen wiederzuerkennen und exakt zu lokalisieren.

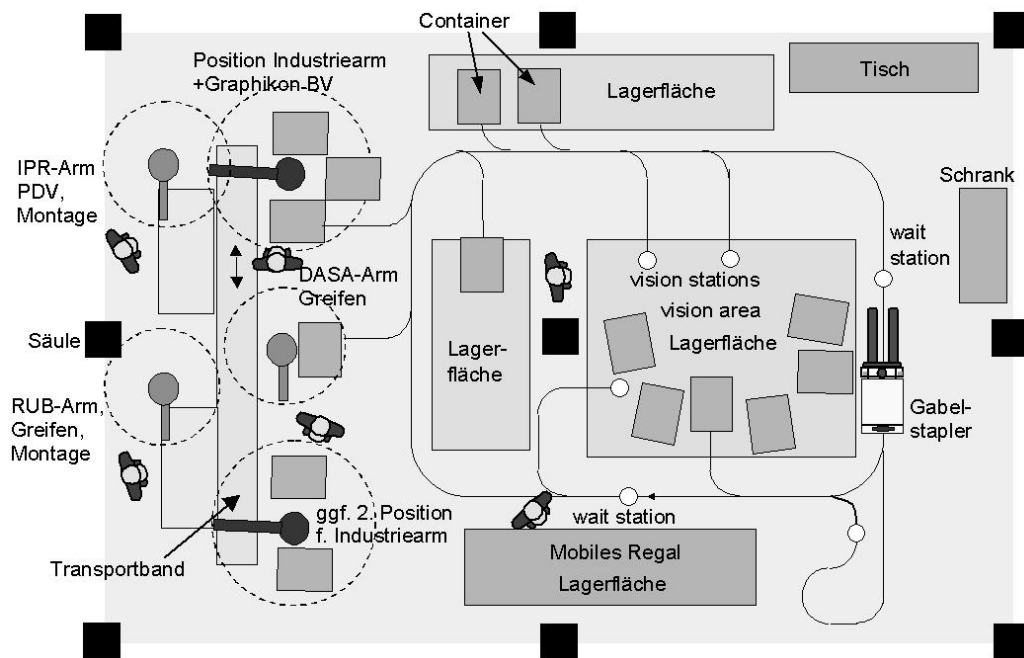


Abbildung 2.1: Darstellung des geplanten Arbeitsumfeldes von CORA, Daimler Chrysler AG.

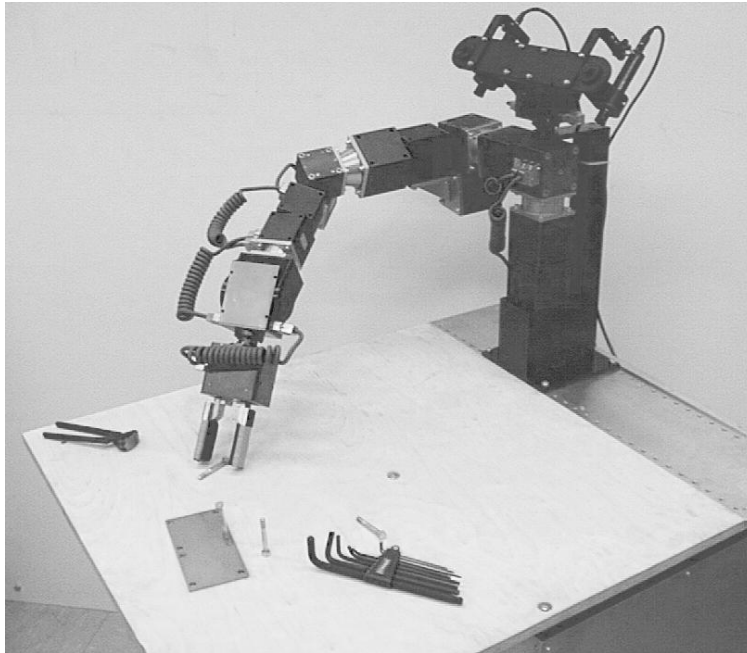


Abbildung 2.2: Der Aufbau von CORA.

## 2.2 Der Aufbau

Im derzeitigen Stadium ist Cora (siehe Abb. 2.2) ein großer Rollwagen mit aufgesetzter Arbeitsplatte. Auf dieser Arbeitsplatte ist der Arm und der Kamerakopf montiert. Das Stereokamera paar erfäßt dabei den Arbeitsbereich. Die Blickrichtung kann horizontal und vertikal variiert werden. Ein positiver Drehwinkel beschreibt ein Drehen nach rechts bzw. ein Schwenk nach oben (siehe Abb. 2.3). Beide Kameras bleiben jedoch immer parallel zueinander. Der modulare Arm des Roboters besitzt sieben Freiheitsgrade. So ist es möglich, Objekte aus verschiedenen Richtungen zu greifen, den "Ellenbogen" zu drehen oder durch einen weiteren Freiheitsgrad im Rumpf sogar den gesamten Arm auf die andere Körperseite zu schwenken, so daß die Anordnung für Rechts- und Linkshänder gleichermaßen geeignet ist.

## 2.3 Anforderungen an die Umgebung

Um ein möglichst gutes Ergebnis der Szenenanalyse zu erhalten, ist es sinnvoll, die Randbedingungen der Szene sorgfältig zu wählen bzw. Vorwissen über die Szene einfließen zu lassen:



gabe eines Objektes, welches zuvor gelernt wurde, kann dieses auf der Arbeitsplatte gefunden und in seiner Lage bestimmt werden.

Um eine vollständige Interaktivität des Systems zu erreichen, reicht es jedoch nicht aus, nur eine statische Szene zu analysieren. Ist ein System interaktiv, so ist es dynamisch. Sowohl die Umwelt als auch das System selber ist ständig in Bewegung. Daher ist es notwendig, Kollisionen auszuschließen, d.h. zukünftige Bewegungsabläufe zu planen und entsprechend einer Szene diese Bewegungsabläufe zu bewerten.

# Kapitel 3

## Filter und Modifikatoren

Die Analyse eines Bildes basiert auf der Extraktion einzelner Merkmale. Solche Merkmale besitzen bestimmte Eigenschaften, die durch einen Filter extrahiert oder unterdrückt werden können. In anderen Fällen ist es nötig, bestimmte Bildeigenschaften zu korrigieren oder zu ändern, welches durch Modifikatoren geschieht. In den folgenden Abschnitten wird zunächst die Extraktion von Bildkanten, d.h. von Bereichen mit starken Grauwertgradienten, anhand zweier Verfahren beschrieben, dem einfachen Schwellwertverfahren und als Beispiel einer anderen Klasse dem Prewitt-Operator. Anschließend wird auf Erosions- und Dilatationsverfahren eingegangen, die zur Reduktion bzw. Expansion von Bildbereichen benötigt werden. Der darauf folgende Abschnitt behandelt die Eliminierung von Bildfehlern durch einen Medianfilter im Vergleich zur Funktion eines Mittelwertfilter. Dann wird ein Verfahren zur Korrektur von Linsenfehlern vorgestellt. Mit der Ermittlung von Disparitätswerten im Vergleich zum biologischen Sehen befaßt sich der letzte Abschnitt dieses Kapitels.

### 3.1 Kantenfilter

#### 3.1.1 Einfaches Schwellwertverfahren

Der hier benutzte Kantenoperator beruht auf einem einfachen Schwellwertverfahren. An jeder Stelle des Bildes wird der Grauwert an der aktuellen Position mit dem eines oder mehreren Nachbarn verglichen. Ist die Grauwertdifferenz eines dieser Pixelpaare größer als ein zuvor gesetzter Schwellwert, handelt es sich an dieser Stelle um eine Kante mit einer durch den Schwellwert vorgegebenen Mindeststärke. Dieser Kantenpunkt wird daraufhin an der gleichen Position in das Ausgabebild des Kantenfilters übertragen

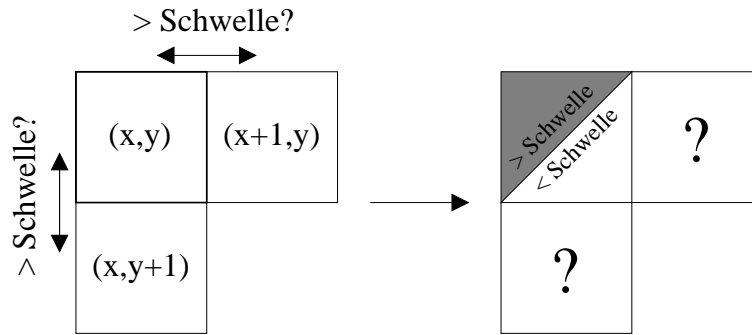


Abbildung 3.1: Schema der Erzeugung eines Kantenbildes (Extraktion von horizontalen und vertikalen Kanten) am Beispiel eines einzelnen Pixels.

(siehe Abb. 3.1). Die Umsetzung stellt drei Kantenmodi zur Verfügung:

- Kantendetektion horizontal
- Kantendetektion vertikal
- Kantendetektion horizontal und vertikal

Zur Ausgabeart können insgesamt drei Arten gewählt werden:

- Die Ausgabe erfolgt in Form von Bilddaten, in welcher diese auch betrachtet werden können.
- Alle Kantenpunkte werden durch ihre Koordinaten beschrieben und nacheinander in ein Feld geschrieben.
- Alle Kantenpunkte werden durch ihre Koordinaten beschrieben und nacheinander in ein Feld geschrieben. Durch Ausblendung von Bildbereichen durch Zuweisung des Grauwertes 0, können sich an den Grenzen von ausgeblendetem und nicht ausgeblendetem Bereichen Kanten bilden. Diese werden hier nicht berücksichtigt.

Der Vorteil dieses Verfahrens liegt in der einfachen und daher schnellen Ermittlung der Kanten. Darüber hinaus ist der Algorithmus universell einsetzbar, da einzelne Richtungen beliebig hinzugenommen und unterschiedlich gewichtet werden können, doch mit steigender Komplexität sinkt der Geschwindigkeitsvorteil.

### 3.1.2 Prewitt Operator

Die Kantenextraktion mit einem einfachen Schwellwertverfahren besitzt den Nachteil, daß z.B. ein einzelnes Störpixel große Kantenwerte in alle Richtungen erzeugt. Um solche falschen Kanten besser zu unterdrücken und die Richtungsdetektion breiter Kantengebiete zu verbessern, kann der Prewitt-Operator (siehe [13]) eingesetzt werden. Zur Detektion der Kanten dient eine Maske, die jeweils eine bestimmte Kantenrichtung extrahiert, wie in 3.1 anhand zweier Beispiele gezeigt.

$$p_1 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}, \text{ oder } p_2 = \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix} \quad (3.1)$$

Auf diese Weise können bis zu acht Richtungen einzeln extrahiert werden. Die Kantenstärke ergibt sich aus der Summe der durch die Maske gewichteten Bildpunktswerte. Andere Operatoren wie Sobel, Robinson oder Kirsch (siehe [13]) basieren auf dem gleichen Verfahren, nur daß hier die Wichtungen anders gewählt sind.

Durch diese Operatoren ist es zwar möglich, qualitativ bessere Kanten bei schlechten Bildern zu erhalten, doch die in dieser Arbeit verwendeten Bilder sind von hoher Qualität. Damit ist der Einsatz dieser komplexeren Kantenfilter nicht sinnvoll, da diese einen erhöhten Rechenaufwand fordern.

## 3.2 Erosion und Dilatation

Die Erosion (siehe [8]) beschreibt eine Methode, in der nur Bildpixel innerhalb einer dicht besetzten Pixelumgebung übernommen werden. Im Allgemeinen handelt es sich bei der Erosion um ein binäres Verfahren. Dieses kann jedoch erweitert werden, indem alle Bildpixel mit dem Grauwert 0 als nicht gesetzte und alle anderen Grauwerte als gesetzte Pixel interpretiert werden.

Ausgangspunkt der Erosion ist ein quadratischer Bereich fester Größe mit einer ungeraden Kantenlänge, der über das Bild bewegt wird. Ist der mittlere Pixel der Maske gesetzt, d.h. ungleich 0, wird die Anzahl aller gesetzten Pixel innerhalb dieser Maske gemessen. Überschreitet diese Anzahl eine zuvor gesetzte Schwelle, wird nur dann der mittlere Pixel an dieser Position im Ausgabebild gesetzt.

Das Beispiel (siehe Abb. 3.2) verdeutlicht dies. Sind innerhalb der Maske genügend Pixel gesetzt (hier acht), wird im Ergebnisbild das mittlere Pixel erhalten. Wird diese Schwelle nicht überschritten, wird kein Pixel im Ergebnisbild gesetzt.

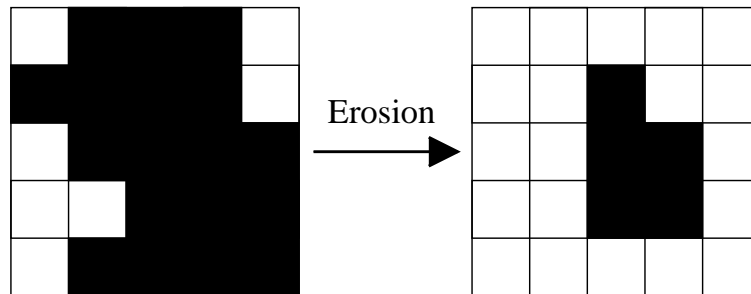


Abbildung 3.2: Erosionsverhalten bei einer Gebietsgröße von  $3 \times 3$  und einem Schwellwert von 8. Links dargestellt ist das Ausgangsbild und rechts das Ergebnis der Operation.

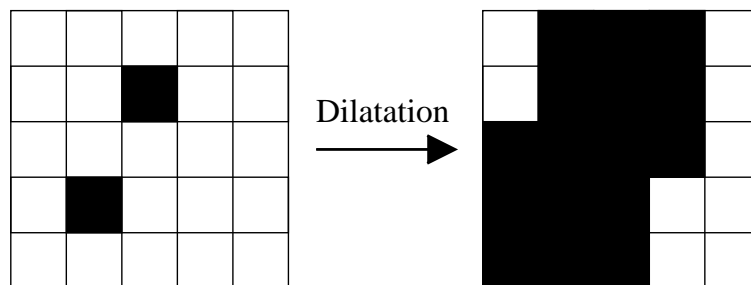


Abbildung 3.3: Dilatationsverhalten einer  $3 \times 3$  Matrix bei einem Schwellwert von 1.

Im allgemeinen genügt die Verwendung einer  $3 \times 3$  Maske. Bei größeren Masken entstehen kompaktere Gebiete. Dies kann in einigen Bereichen der Bildverarbeitung nützlich sein, doch hier wird zur Ortsauflösung nur ein sehr grobes Raster benutzt, um die Geschwindigkeit zu erhöhen, so daß bei einer großen Maske aussagekräftige Konturen verloren gehen können.

Die Dilatation beschreibt ein Verfahren, daß eine entgegengesetzte Wirkung wie die Erosion besitzt. Ausgangspunkt ist auch hier eine quadratische Maske fester Größe mit einer ungeraden Kantlänge, die über das Bild bewegt wird. Die Anzahl aller gesetzten Pixel innerhalb dieses Bereichs wird gemessen. Überschreitet diese Anzahl eine zuvor gesetzte Schwelle, wird der mittlere Pixel an dieser Position im Ausgabebild gesetzt (siehe Abb. 3.3).

Einen Spezialfall bildet in der vorliegenden Implementation der Schwellwert Eins. Nach herkömmlichen Verfahren müßte innerhalb jeder Maskenposition geprüft werden, ob und wie viele Pixel gesetzt sind. Stattdessen wird für diesen Fall das Verfahren modifiziert. Demnach wird nur dann der gesam-

te Bereich der Maske gesetzt, wenn im Quellbild der mittlere Pixel gesetzt ist. Die Ergebnisse beider Verfahren sind dabei identisch.

Der Vorteil liegt darin, daß nicht alle Pixel einer Matrix auf Belegung hin überprüft werden müssen und dadurch ein Geschwindigkeitsvorteil erzielt wird.

Da es sich bei der Erosion und bei der Dilatation um binäre Verfahren handelt, erzeugt die Ausgabe auch nur ein binäres Bild. Ein Graustufenbild kann durch einen Schwellwert wie ein binäres Bild ausgewertet werden. Später werden die Grauwerte im Ergebnisbild als den Grauwerten des Eingabebildes ergänzt.

Erosion und Dilatation sind, wie bereits gezeigt, nicht kommutativ. Durch eine Anwendung dieser beiden Methoden nacheinander kann eine spezielle Wirkung erzielt werden (siehe Kap. 3.3).

Ein neuer Ansatz dieser Verfahren wird durch eine Modifikation des Verfahrens der Erosion erreicht. Bisher mußten zwei Bedingungen bei einer Erosion erfüllt sein, wenn das Zielpixel gesetzt werden sollte:

- Der mittlere Maskenpixel mußte gesetzt sein.
- Die Anzahl der gesetzten Maskenpixel mußte einen Schwellwert übersteigen.

Das Außerkraftsetzen der ersten Bedingung führt zu einem Verfahren, welches sowohl Erosions- als auch Dilatationseigenschaften besitzt, dem Prinzip nach jedoch einer einfachen Dilatation entspricht. Ist der Schwellwert gering, so überwiegen die Dilatationseigenschaften, und das Verfahren ähnelt einem Closing (siehe Abb. 3.4 und Kap. 3.3), das durch eine Dilatation, gefolgt von einer Erosion, definiert ist. Ist der Schwellwert hoch, so überwiegen die Erosionseigenschaften, und das Verfahren ähnelt einem Opening (siehe Abb. 3.5 und Kap. 3.3), d.h. einer Erosion, gefolgt von einer Dilatation.

### 3.3 Opening und Closing

Aus den zuvor erläuterten Methoden einer Erosion und Dilatation ist ersichtlich, daß beide zwar eine entgegengesetzte Wirkung haben, sich diese aber nicht gegenseitig aufheben, d.h. die Verfahren sind nicht kommutativ. Daher resultiert aus einer Erosion gefolgt von einer Dilatation ein anderes Ergebnis als aus einer Dilatation gefolgt von einer Erosion. Jede Folge für sich besitzt dabei spezielle Eigenschaften und werden als Opening und Closing (siehe [8]) bezeichnet.

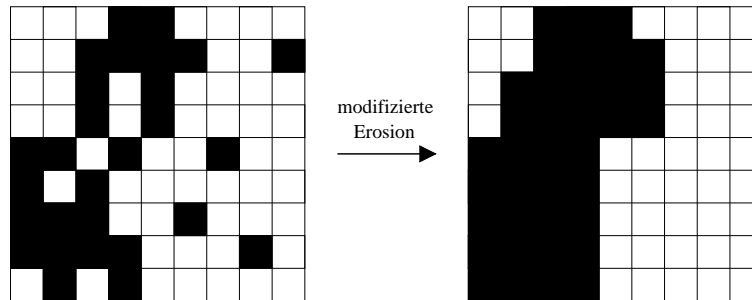


Abbildung 3.4: Modifizierter Erosionsalgorithmus mit Schwellenwert 3.

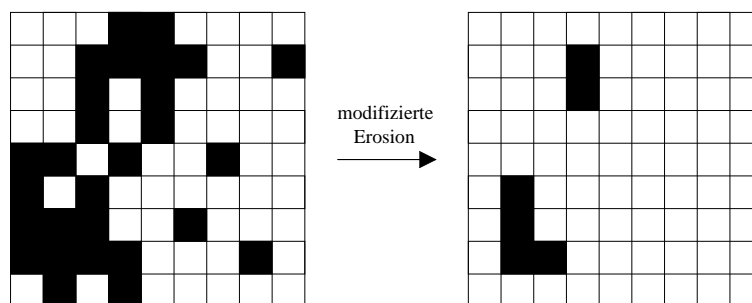


Abbildung 3.5: Modifizierter Erosionsalgorithmus mit Schwellenwert 7.

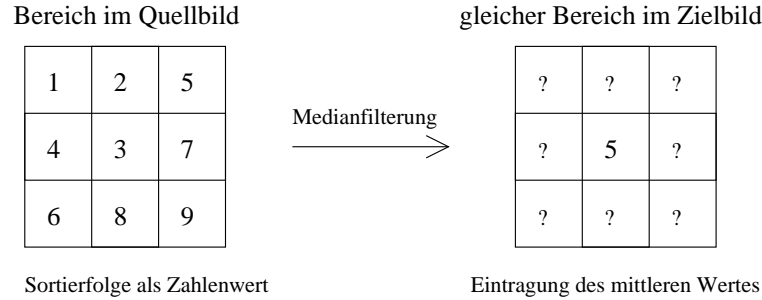


Abbildung 3.6: Medianfilterung eines 3x3 Bereichs. Der in der sortierten Reihenfolge in der Mitte stehende Pixelwert, hier durch den Zahlenwert 5 dargestellt, wird in den mittleren Pixel des Ergebnisbildes übertragen.

Die Erosion gefolgt von einer Dilatation wird als Opening bezeichnet. Die Wirkung ist eine Beseitigung kleiner Störungen und die Glättung von Kanten.

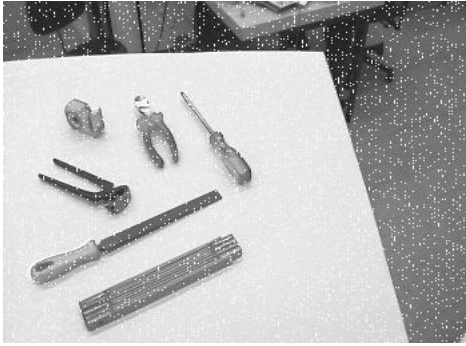
Die Dilatation gefolgt von einer Erosion wird als Closing bezeichnet. Diese dient zur Schließung von kleinen Lücken, z.B. bei einer unterbrochenen Geraden oder nicht vollständig besetzten Gebieten.

### 3.4 Medianfilter

Stark sichtbare Fehler in Bilddaten sind einzelne Pixel, die bezogen auf ihre Nachbarn, im Grauwert stark abweichen, wie Bildschnee oder Fische (siehe Abb. 3.7 links). Solche Abweichungen können in der weiteren Verarbeitung schwere Fehler verursachen. Ein Medianfilter kann in begrenztem Maße solche Fehler eliminieren. In einer quadratischen Maske mit ungerader Kantenzahl (siehe Abb.3.6) werden dabei alle Grauwerte der Einzelpixel der Größe nach sortiert. Der in der Sortierreihenfolge in der Mitte stehende Grauwert wird anschließend in das Ergebnisbild an die Position der Mitte der Maske übertragen.

Das Ergebnis einer solchen Filterung ist ein Bild (siehe Abb.3.7 rechts), welches stark von den oben genannten Fehlern befreit wurde. Das Besondere am Medianfilter ist die Eigenschaft, starke Kanten und Strukturen zu erhalten und kleine Störungen zu beseitigen.

Ein Mittelwertfilter glättet dagegen auch starke Kanten, so daß das Bild unscharf wird. Dieser operiert auf einem Bereich einer Maske mit ungerader Kantenzahl, wobei alle Grauwerte im Maskenbereich aufsummiert und durch die Anzahl der Maskenwerte geteilt werden. Der sich ergebene Wert,



(a) Bild mit Störungen



(b) Bild nach Medianfilterung

Abbildung 3.7: Medianfilter auf ein künstlich verrauschtes Bild. Die Anzahl der Fehlpixel liegt bei ca. 5%. Nach der Filterung sind diese Fehlstellen beseitigt und das Bild ist in einigen Teilen leicht geglättet.

der in die Mitte der Maske ins Ergebnisbild eingetragen wird, stellt dadurch den durchschnittlichen Grauwert seiner Umgebung dar. Diese Art der Filterung bewirkt eine gute Unterdrückung von Rauschanteilen, jedoch verliert das Bild dabei erheblich an Kontrast.

Der Vorteil des Medianfilters gegenüber des Mittelwertfilters ist die Erhaltung von Kanten, die in diesem Projekt von besonderem Interesse sind. Ebenso wie sich dieses Verfahren auf Bilddaten anwenden läßt, kann es auch zur Glättung von Funktionskurven oder ermittelten Spektren benutzt werden.

## 3.5 Linsenkorrektur

Für die Bestimmung von Disparitäten (siehe Kap. 3.6.4) ist es zwar nicht nötig, daß korrespondierende Punkte in derselben Zeile liegen, doch die Ermittlung wird dadurch erheblich vereinfacht. Um diese Voraussetzung nutzen zu können, müssen verschiedene Faktoren, die dies beeinträchtigen können, berücksichtigt werden:

- Durch eine leichte Fehljustage der Kameras, besonders aber durch eine nicht mittige Anordnung des CCD-Sensors hinter der Kameralinse, kann der Bildausschnitt leicht verschoben oder sogar gedreht sein. Eine Korrektur dieser Fehler kann durch eine Translation bzw. einer Rotation des Bildes erfolgen, in der Praxis genügt jedoch meist die mechanische Kalibrierung den Anforderungen.

- Durch die Krümmung der Linse entsteht eine radiale Verzerrung des Bildes. Diese Verzerrung kann dadurch korrigiert werden, daß der Abstand jedes Bildpunktes zum Bildhauptpunkt durch

$$x' = \frac{x}{1 + k|x|^2} \quad (3.2)$$

approximiert wird (siehe [11]).  $k$  ist dabei eine feste Linsengröße und beschreibt die radiale Verzeichnung.

Zur Berechnung des entzerrten Bildes wird eine inverse Berechnung verwendet, d.h. ausgehend von einer Position im entzerrten Bild wird die korrespondierende Position im verzerrten Bild berechnet. Diese ermittelte Position im verzerrten Bild ist nicht gerundet und befindet sich daher zwischen bis zu vier Pixelmittelpunkten (siehe Abb. 3.8). Abhängig von der Lage dieser Position gehen die beteiligten Pixel durch ihren Grauwert anteilig in den Grauwert des Pixels im entzerrten Bild ein. Dadurch ist eine lückenlose interpolierte Berechnung des entzerrten Bildes gewährleistet.

Eine direkte Berechnung des entzerrten Bildes ist nicht angebracht, da durch eine Linsenentzerrung das Bild geweitet wird und damit die im Bildbereich vorhandene Pixelanzahl abnimmt. Die dadurch entstehenden Lücken müßten interpoliert werden und dies aus Pixeln, die in ihrer Position bereits gerundet wurden.

Der Nebeneffekt der inversen Berechnung ist eine leichte Glättung der Bilddaten. Zum einen kann dies als negativ bewertet werden, da auf diese Weise Kontraste geschwächt werden. Auf der anderen Seite besitzt jedes Kamerabild einen Rauschanteil, der durch diese nur leichte Glättung stark verringert wird.

Ein weiterer Aspekt der Linsenkorrektur ist die verwendete Bildauflösung. Der Bildhauptpunkt, d.h. der Mittelpunkt der Linse projiziert durch deren optische Achse auf den CCD-Chip, ändert nicht seine physikalische Position, jedoch seine Koordinaten bei geänderter Bildauflösung. Bei einer Verringerung der vertikalen Auflösung werden die untersten Bildzeilen einfach weggelassen. Der Bildhauptpunkt behält daher seinen Wert bei, befindet sich jedoch, bezogen auf den Mittelpunkt des neuen Bildes, weiter unten. Anders verhält es sich bei einer Änderung der horizontalen Auflösung. Die Kamera liefert ein analoges Zeilensignal, welches erst später, entsprechend der horizontalen Auflösung, abgetastet wird. Daher entspricht die Pixelbreite des Chippixels nicht der Breite des Chipbereiches, der für die Erzeugung eines Bildpixels benutzt wird. Eine einfache Umrechnung per Dreisatz kann daher durch diese Breitenänderung die neue horizontale Position des Bildhauptpunktes bestimmen.



Für das Tiefensehen sind verschiedene sich ergänzende Systeme notwendig (siehe [9] und [14]), die zum einen rein visuell sind, zum anderen aber auch auf Wissen basieren.

- Ein wesentliches Mittel zur Abschätzung der Tiefe des Raumes ist die Disparität. Bei der Betrachtung eines Raumpunktes wird dieser im linken und rechten Auge auf unterschiedliche Bereiche der Retina abgebildet. Ziel der Bestimmung der Disparität ist es, die korrespondierenden Punkte auf der Retina beider Augen zu finden. Aus dem Betrag der Verschiebung ist es möglich, direkt die Entfernung des Punktes zu bestimmen.
- Auch die Bewegung des Auges in der Welt führt zu einer Bewertung der Tiefe durch die sogenannte Bewegungsparallaxe. Dabei werden temporal versetzte Bilder der Umwelt miteinander verglichen. Der Versatz des fixierten Objektes auf der Retina zusammen mit der in dieser Zeit zurückgelegten Strecke ist dabei ein Maß der Entfernung des Objektes.
- Schattierungen auf Objekten ermöglichen eine Schätzung der Tiefe. Dabei müssen Grundeigenschaften der Szene wie Position der Lichtquelle, Richtung des Betrachters und Reflexionseigenschaften des Objektes bekannt sein. Voraussetzung hierfür ist die Annahme einer homogenen Oberflächenbeschaffenheit des Körpers.
- Eine gleichmäßige Textur verkleinert sich in der Tiefe und erhöht dabei auch ihre Dichte. Durch die Annahme einer konstanten Umwelt kann daraus direkt auf die Entfernung geschlossen werden.
- Das Auge muß, wie jedes optisches Linsensystem, eine eigene Schärferegulierung besitzen. Die Schärfe ist demnach ein direktes Maß zur Tiefenmessung.

Eine Fusion korrespondierender Bildpunkte zur Disparitätsermittlung ist beim Menschen nur in einem eng begrenzten Bereich möglich bzw. nötig. Vergenzbewegungen des Auges reduzieren die absolute Disparität sowie damit verbunden auch die sonst nur mit einem Auge wahrgenommenen Bildanteile.

### 3.6.3 Sehen in technischen Systemen

Die Komplexität der neuronalen Netze und besonders die parallele Verarbeitung der Sinnesreize machen eine adäquate Nachbildung biologischer Strukturen mit dem heutigen Stand der Technik unmöglich. Eine sinnvolle Beschränkung der Systeme auf wesentliche Prinzipien ist daher unerlässlich.

Einige Beschränkungen können durch einen multisensorischen Aufbau, wie z.B. zusätzliche Infrarotsensoren, Ultraschallentfernungsmesser oder Laserscanner, zum Teil aufgehoben werden. Bei CORA wurde jedoch bewußt auf weitere Sensoren verzichtet, da die Umsetzung mit den Beschränkungen des menschlichen Vorbildes erfolgen soll.

Das hier verwendete stationäre Kamerasystem besitzt im Vergleich zum biologischen Vorbild große Einschränkungen. Die Schärfe ist nur manuell einstellbar und die Kamerabewegung ist auf konjugierte Bewegungen in der Vertikalen und eine Drehung um eine gemeinsame Achse in der Horizontalen beschränkt. Dadurch gestaltet sich, eine entsprechende Kalibrierung und Linsenentzerrung vorausgesetzt, auch die Disparitätssuche einfacher, da korrespondierende Punkte dann immer in der gleichen Zeile liegen.

Durch diese parallele Ausrichtung der Kameramodule und deren Abstand zueinander in Bezug auf den Abstand zur Szene, ist jedoch nur der durch beide Kameramodule erfaßte Bildanteil, beim Roboter CORA ca.  $\frac{2}{3}$  der Bildfläche, nutzbar.

### 3.6.4 Disparitätsfindung

Bei der Disparitätsfindung ist es das Ziel, korrespondierende Pixelbereiche der zwei Kameras eindeutig zu fusionieren, um deren durch die unterschiedliche Perspektive verursachte horizontale Verschiebung zu erhalten. Ein Pixelbereich der rechten Kamera muß daher einen Korrespondenzbereich im linken Kamerabild rechts von dieser Position besitzen. Dabei ist die Gefahr der Gewinnung von Fehlinformationen sehr groß. Gründe für eine Fehlmessung der Disparität sind folgende:

- Durch die gedrehte Perspektive können Objekte oder Objektteile verdeckt oder stark verzerrt sein. Dadurch können diese Bereiche nicht oder nur sehr schwach mit anderen korrespondieren.
- Beleuchtungseffekte wie Reflexionen treten in beiden Kamerabildern in unterschiedlicher Form auf. Die dadurch bedingte Differenz der korrespondierenden Bereiche führt meist zu einer Fehlinterpretation.
- Wiederkehrende Strukturen oder Bereiche homogener Farbe, lassen sich lokal nicht eindeutig zuordnen. Eine Zuordnung ist dann nur mit globalem Wissen über die Szene möglich.

In der Praxis erfolgt die Disparitätssuche auf einem Bereich fester Größe (siehe Abb. 3.9). 3x3 Pixel sind hierbei ausreichend und werden im rechten

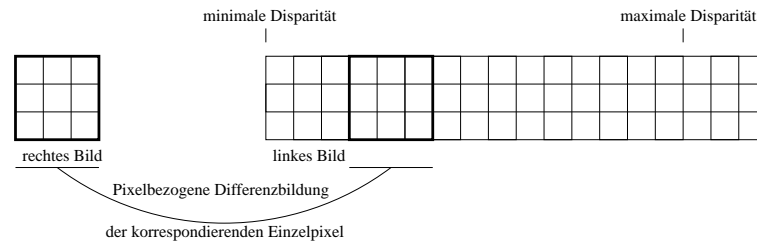


Abbildung 3.9: Schema der Disparitätsfindung.

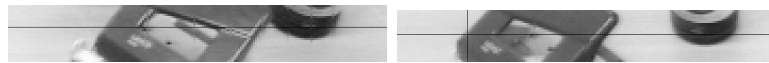


Abbildung 3.10: Rechtes Bild: Das Kreuz deutet die Position des Ausgangspixels an, zu dem im linken Bild ein korrespondierender Pixel gefunden werden soll. Linkes Bild: Entlang der Linie wird nach der minimalen Pixeldifferenzsumme und somit nach maximaler Korrespondenz gesucht.

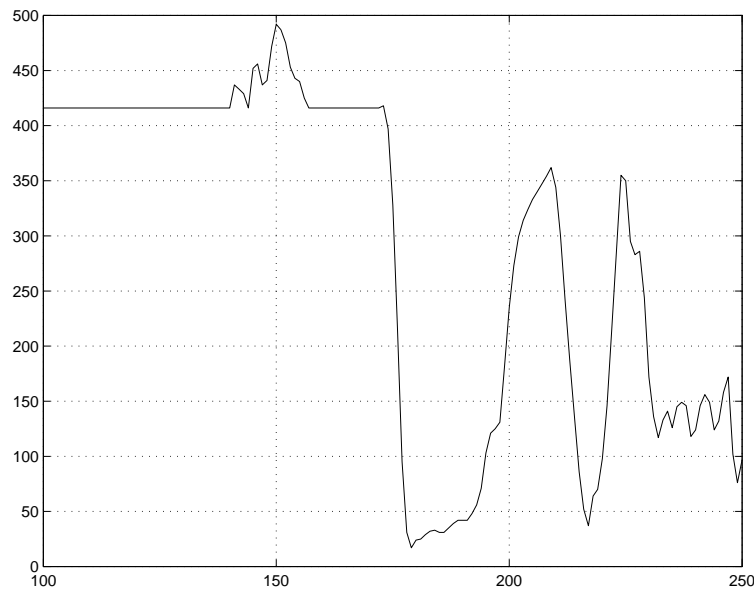


Abbildung 3.11: Die Darstellung der Pixeldifferenzsumme. Das Minimum bezeichnet die Stelle mit der wahrscheinlichsten Differenz.

Kamerabild extrahiert. Unter Bildung der Summen der pixelweisen Differenzen mit Bereichen der rechts gelegenen Restzeile im linken Bild wird ein Maß für die Übereinstimmung der jeweiligen Bereiche gewonnen.

$$\text{Differenz} = \sum_{i=1}^9 (\text{Maskenpixel}_i(\text{rechts}) - \text{Maskenpixel}_i(\text{links})) \quad (3.3)$$

Das Minimum ist demzufolge ein Indiz für eine maximale Übereinstimmung. Dieses Ergebnis ist aufgrund vorangegangener Überlegungen jedoch nur sinnvoll, wenn in einem Toleranzbereich um dieses Minimum herum keine weiteren Werte liegen, damit die Qualität des Treffers hoch genug ist. Doch auch wenn diese Bedingung erfüllt ist, muß zusätzlich eine gesetzte Schwelle unterschritten werden, d.h. es muß eine ausreichend gute Korrespondenz vorliegen (vgl. Abb. 3.12 und 3.13). Ergebnisse dieses Verfahrens zeigen jedoch, daß bei sich horizontal wiederholenden Strukturen Korrespondenzprobleme auftreten. Ursache dieses Problems ist die annähernde Übereinstimmung der Korrespondenzkandidaten. Dieser Mangel kann zum Teil durch eine Variation der Maskengröße vermieden werden, doch auch hier gibt es Grenzen.

Die Höhe einer Maske darf nicht sehr groß sein. Je nach Abstand beider Kameras und Entfernung der Objekte kann das Bildobjekt im Vergleich beider Kamerabilder stark verzerrt sein. Im vorliegenden Fall zeigte sich, daß eine Maskenhöhe von drei Pixeln sinnvoll ist. Bei der Breite einer Maske hat sich eine maximale Maskenbreite von zwölf Pixeln bewährt. Hierbei werden Kanten besser erkannt, als bei einer kleinen Maske, da auch die Umgebung wertvolle Informationen liefern kann.

Die neue Disparitätssuche besteht daher aus zwei Schritten. Zunächst werden Disparitäten mit einer 12x3 Maske gesucht. Dabei werden Gebiete, die arm an Gebietsstauchungen und Verdeckungen sind, sicher zugeordnet. Zur Verarbeitung der Restbereiche wird eine weitere Disparitätssuche, jetzt mit einer 3x3 Maske, durchgeführt. Durch die kleinere Maske können an dieser Stelle auch Bereiche starke Verzerrungen gut ausgewertet werden.

## 3.7 Clustern

Um in einem Bild verschiedene Bereiche, die voneinander getrennt sind, einzeln ansprechen zu können, müssen jedem einzelnen dieser zusammenhängenden Bereiche z.B. eine Nummer zugeordnet werden. Ein solches Verfahren ist das Clustern (siehe [4]). Dabei wird in einem ersten Durchlauf jedem gesetzten Bildpunkt eine Zahl zugeordnet. Diese Zahl wird immer dann inkrementiert, wenn ein ungesetzter Pixel einem gesetzten Pixel folgt. Damit werden



Abbildung 3.12: Disparitätsergebnis bei einer Ermittlung mit einer 3x3 Maske über das gesamte Bild ohne Bewertung der Qualität.



Abbildung 3.13: Disparitätsergebnis bei einer Ermittlung mit einer 3x3 Maske mit starker Bewertung der Qualität.

allen horizontal zusammenhängenden Pixel, die gesetzt sind, die gleiche Zahl zugeordnet. In einem zweiten Durchlauf wird überprüft, welche dieser Bereiche vertikal zusammenliegen. Das Ergebnis ist eine vollständige Durchnummerierung aller gesetzter zusammenhängender Pixelbereiche. Dadurch ist es möglich, jeden dieser Bereiche einzeln zu adressieren und Informationen über dessen Größe zu erhalten.



# Kapitel 4

## Objekte lernen und erkennen

Ein wesentliches Problem der Objekterkennung sind aufwendige Berechnungen, die einer geforderten Echtzeitfähigkeit des Algorithmus enge Grenzen steckt. Speziell in der hier gegebenen Hardwarelösung kann ein ergebnisoptimierter Ansatz nicht echtzeitfähig sein, da solche Berechnungen zu viel Zeit benötigen würden. Bei einer schnellen Verarbeitung müssen demnach Einschränkungen auf Kosten der Qualität in Kauf genommen werden oder für die Szene müssen frühzeitig unrelevante Anteile entfernt werden. Da eine Qualitätsverschlechterung nicht wünschenswert ist, werden hier Methoden zur frühzeitigen Reduktion der Datenmenge benutzt. Dazu wird zunächst die Szene auf die wesentlichen Bestandteile, also die auf der Tischoberfläche liegenden Gegenstände, begrenzt. Die Bereiche der Tischfläche und deren Umgebung werden ausgeblendet, indem sie den Grauwert 0 erhalten. Diese Ausblendungsmaske wird auf alle Farbbänder des HSI-Bildes übertragen. Dadurch wird die Datenmenge bereits erheblich reduziert. Ein weiterer Schritt der Datenreduktion wird in der Suchphase durch eine bereichsbezogene Farbfilterung erzielt, deren Farbwerte zuvor objektbezogen in einer Lernphase ermittelt wurden.

Im nächsten Schritt werden in diesem reduzierten Bildbereich die Disparitäten ermittelt. Diese können anschließend in Raumkoordinaten transformiert werden. Durch eine Aufsicht, d.h. einer Projektion der Raumpunkte von oben auf die Tischfläche, entsteht für jedes Objekt eine charakteristische Abbildung. Auf Grundlage zuvor gelernter Objektaufsichten wird die neu ermittelte Aufsicht analysiert, d.h. die Position von gewählten Objekten gefunden.



(a) linkes Kamerabild



(b) rechtes Kamerabild

Abbildung 4.1: Quellbilder einer Tischszene. Der Blickwinkel auf die Szene ist  $0^\circ$  horizontal und  $-45^\circ$  vertikal.

## 4.1 Szenenbeschränkung

### 4.1.1 Bereiche der Szene

Eine typische Szene des Arbeitsplatzes (siehe Abb. 4.1) beinhaltet drei zu unterteilende Regionen, welche:

- die auf der Tischoberfläche liegenden Gegenstände,
- die Tischoberfläche selber oder
- die Umgebung des Tischbereiches

repräsentieren. Ziel ist die Extraktion der Gegenstände bzw. die Unterdrückung der Tischfläche und der Umgebung. Durch eine einfache Bewertung der Bildregionen ist es nur schwer möglich, diese zu trennen. Daher müssen bekannte Eigenschaften des Roboters genutzt werden, um über Wissen diese Bereiche ausschließen zu können.

### 4.1.2 Wissensbasierte Bereichsextraktion

Die Geometrie des CORA-Aufbaus, d.h. die Abmessungen der Tischfläche und die Höhe und Ausrichtung der Kameras, ist bekannt. Dadurch, daß mit diesen bekannten Werten Raumpunkte eindeutig einem Bildpunkt des Kamerabildes zugeordnet werden können, ist eine Errechnung der Lage der Tischoberfläche im Kamerabild möglich.

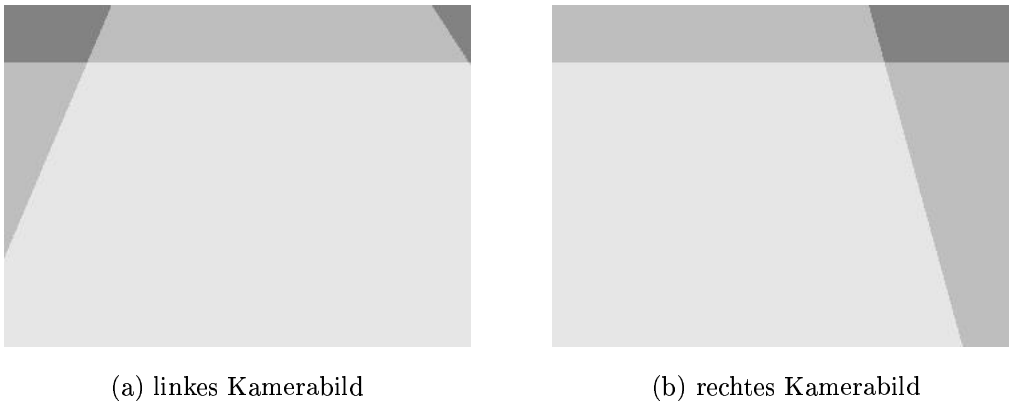


Abbildung 4.2: Darstellung der Maskenerstellung zur Tischflächenbestimmung. Die hellste Fläche stellt die Schnittmenge aller ermittelten Halbräume, und somit die Tischoberfläche, dar.

Dazu werden zunächst die Raumkoordinaten der Tischecken in Kamerakoordinaten transformiert. Über die Bestimmung des horizontalen und vertikalen Winkels, die jeder dieser erfaßten Punkte gegenüber der optischen Achse der Kamera aufspannt, kann der zugehörige Bildpunkt ermittelt werden. Diese Bildpunkte werden miteinander verbunden und die eingeschlossene Fläche als Maske zur Verfügung gestellt. Die Ermittlung dieser Fläche erfolgt durch die Überlappung vierer Halbräume (siehe Abb. 4.2), die jeweils durch zwei Bildpunkte definiert werden.

In dem nun betrachteten Restbereich befinden sich nur noch die Tischoberfläche und die Objekte. Um diese voneinander trennen zu können, ohne Informationen über die Tischfarbe zu besitzen, wird hier davon ausgegangen, daß:

- die Tischfarbe homogen und damit texturlos,
- der Tischflächenanteil wesentlich größer als der Objektflächenanteil und
- die Ausleuchtung aller Bereiche gleichmäßig ist.

Ausgehend von diesen Annahmen kann die Entfernung der Tischfläche in zwei Schritten erfolgen:

1. Ein Spektrum der im Bild vorkommenden Graustufen wird erstellt. Da die Tischoberfläche den Hauptteil der Bilddaten ausmacht, wird das Spektrum im Bereich der homogenen Tischfarbe ein Maximum ausbilden. Mit einer geringen Toleranz können alle Bildpixel, die sich im

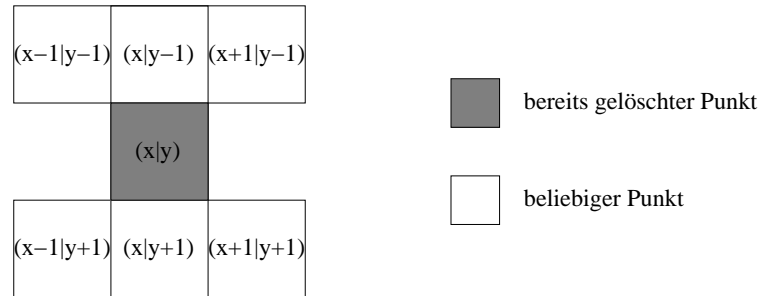
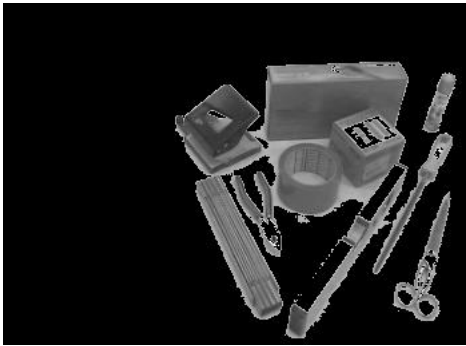


Abbildung 4.3: Schematische Darstellung der abgefragten Bildpunkte, um den gelöschten Bereich zu vergrößern.

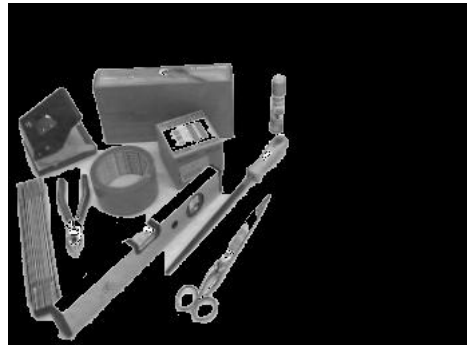
Bereich dieses Farb-Maximums befinden, gelöscht werden. Ein Großteil der Arbeitsfläche ist nach diesem Schritt bereits entfernt, doch besonders in Randbereichen des Bildes und nahe an oder zwischen Objekten ist die Tischoberfläche aufgrund von Schattenbildung, die durch eine geeignete Beleuchtung vermieden werden sollte, noch vorhanden.

2. Um diese Restflächen erfolgreich zu eliminieren, kann ein einfacher Wachstumsalgorithmus benutzt werden. Ziel ist es, Regionen, die an bereits gelöschte Bildpunkte grenzen, daraufhin zu überprüfen, ob es sich bei diesen Bildpunkten auch um die Tischfläche handeln könnte. Dazu werden hier die drei direkt benachbarten Pixel der nächsten bzw. vorhergehenden Zeile betrachtet (siehe Abb. 4.3). Ist ein Bildpunkt an der aktuellen Position bereits gelöscht, so wird getestet, ob die sechs benachbarten Pixel jeweils zur aktuellen Position einen hohen Grauwertgradienten aufweisen. Ist dies der Fall, so muß an dieser Stelle eine Kante vorliegen und eine Zugehörigkeit zur Tischfläche ist somit unwahrscheinlich. Ist der Grauwertgradient jedoch klein und weicht der Grauwert an der geprüften Stelle nur gering von der gemittelten Tischfarbe ab, so muß es sich weiterhin um die Tischoberfläche handeln und der getestete Bildpunkt kann ebenfalls gelöscht werden.

Das Ergebnis ist nicht immer optimal, aber genügt den Anforderungen (siehe Abb. 4.4). Darüber hinaus gewährleistet diese einfache Systematik eine schnelle Verarbeitungsgeschwindigkeit.



(a) linkes Kamerabild



(b) rechtes Kamerabild

Abbildung 4.4: Darstellung der extrahierten Objektanteile des Kamerabildes. Klar zu erkennen sind Restbereiche, die aufgrund einer starken Schattenbildung nicht als Tischfläche identifiziert wurden, oder Objektteile, die aufgrund ihrer Farbgebung ebenfalls der Tischoberfläche zugeordnet wurden.

## 4.2 Farbbasierte Filterung

In diesem Verarbeitungsschritt besteht der Bildbereich fast ausschließlich aus den auf der Arbeitsfläche liegenden Gegenständen. Zur weiteren Zeitoptimierung in der Suchphase wird diese Datenmenge wiederum reduziert. Dies geschieht mit einem Farbfilter, der nur solche Farben erhält, die den gesuchten Gegenstand maßgeblich charakterisieren, und getrennt erlernt werden müssen.

Dabei ist zu beachten, daß sich die Charakterisierung eines Gegenstandes anhand seiner Farbinformationen sehr schwierig gestaltet. Da ein einzelner Gegenstand bunt gemustert sein kann, besitzt er im Allgemeinen nie eine einzige charakteristische Farbe, Farbsättigung oder Intensität, sondern deckt im Farbraum unter Umständen diverse Bereiche ab. Durch Reflexionen, Schattenbildungen und Beleuchtungstoleranzen werden diese Bereiche weiter gestreut.

### 4.2.1 Der HSI-Farbraum

Die Farbbildverarbeitung in der vorliegenden Arbeit basiert auf dem HSI Farbraum. Das H steht für Farbwinkel (engl. Hue), S für Sättigung (engl. Saturation) und I für die Intensität (engl. Intensity). Geometrisch kann dieser Farbraum als Kegel (Abb. 4.5) dargestellt werden. Dies veranschaulicht, daß für geringe Sättigungswerte im Anwendungsfall keine zuverlässige Aussage

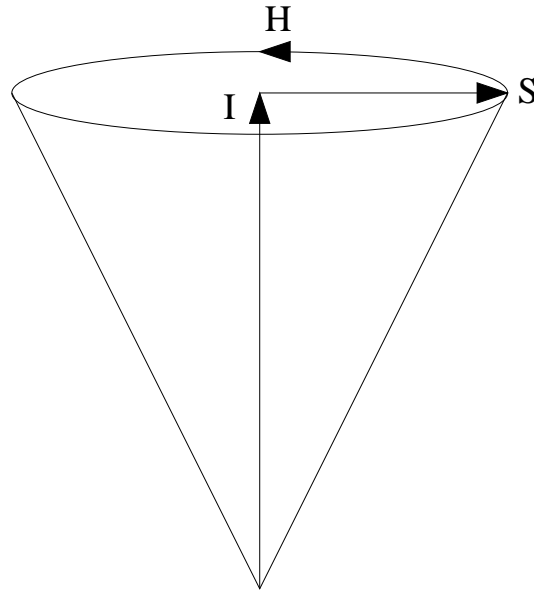


Abbildung 4.5: Darstellung des HSI-Farbraumes. Der Farbraum wird durch den Farbwinkel H (engl. Hue), die Sättigung S (engl. Saturation) und die Intensität I (engl. Intensity) aufgespannt.

über den Farbwert getroffen werden kann. Kleinste Änderungen, die z.B. durch Rauschen entstehen können, kippen dann den Farbwert in eine völlig andere Position.

Ein gutes Beispiel ist hierbei die weiße Tischoberfläche. Weiß hat eine hohe Intensität und liegt daher im Farbraum nahe oder sogar auf der Kegelgrundfläche (hier Oberseite). Die (Farb-)Sättigung ist sehr gering, da Weiß keine Farbe ist, so daß sich der Farbraumpunkt nahe oder sogar in der Mitte der Kegelgrundfläche befindet. Ist der Sättigungswert 0 zeigt sich, daß eine Angabe des Farbwinkels nicht möglich ist.

Zur Bildeingabe der Farbbilder dient das ppm-Format in abgewandelter Form (portable pixmap) und zur Bildausgabe das pgm-Format (portable graymap).

### 4.2.2 Ermittlung der Farbeigenschaften eines Objektes

Zur Ermittlung der Farbverteilung im Farb-, Sättigungs- und Intensitätsband wird zunächst ein Spektrum jedes einzelnen Farbbandes erzeugt (siehe Abb. 4.6).

Ausgeblendete Bereiche werden hierbei nicht berücksichtigt. Die Auswer-

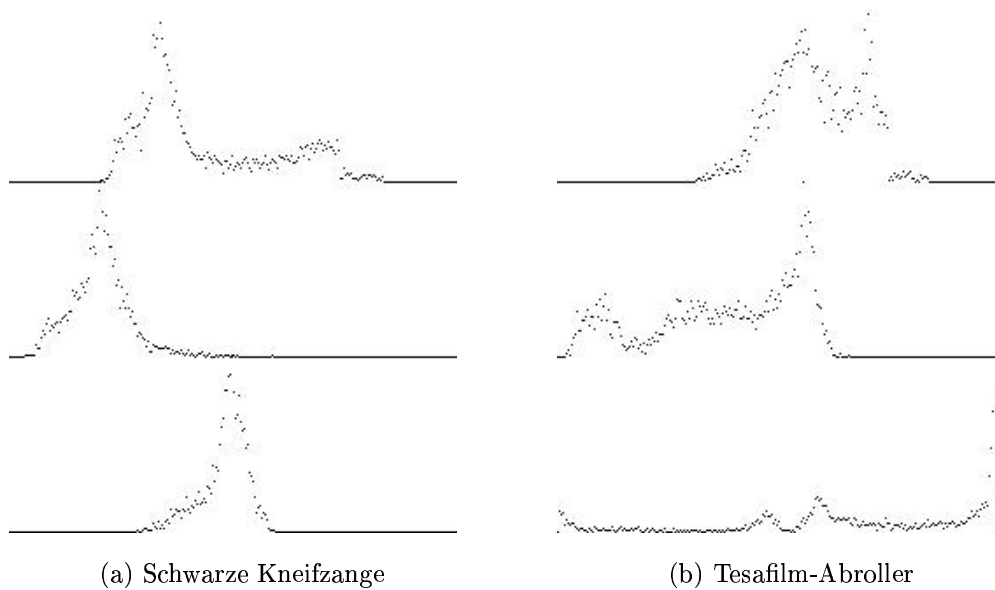


Abbildung 4.6: Darstellung der HSI-Bänder (H unten, S mittig, I oben) nach der Häufigkeit eines einzelnen Farbwertes. Deutlich erkennbar sind die unterschiedlichen Belegungen der einzelnen Bänder.

tung liefert für jeden Gegenstand ein charakteristisches Spektrum. Durch ein einfaches Schwellwertverfahren werden geringe Werte unterdrückt, so daß nur die meist vorkommenden Werte berücksichtigt werden. Diese werden, in Zuordnung zu diesem Gegenstand, zur weiteren Verwendung abgespeichert.

### 4.2.3 Anwendung der Farbeigenschaften eines Objektes

Um bei der Suche nach einem Gegenstand die Datenmenge und somit den in Frage kommenden Bildbereich möglichst klein zu halten, werden die zuvor ermittelten spektralen Eigenschaften eines Gegenstandes als Filterwerte genutzt. Für jedes Farbband, in dem der Farbwert eines Bildpunktes mit dem eines der Filterwerte übereinstimmt, wird die Filterantwort um eins erhöht. Ist dies in jedem Farbband der Fall, ist demnach die Filterantwort drei. Im nächsten Schritt wird ermittelt wie hoch der Anteil der maximalen Filterantwort innerhalb jedes Clusters ist. Wenn der Anteil sehr gering ist, muß davon ausgegangen werden, daß es sich bei dem an dieser Stelle abgebildeten Gegenstand nicht um den gesuchten handeln kann. Die Entscheidungsschwelle darf jedoch nicht zu hoch gewählt werden, da es möglich ist, daß sich mehrere Gegenstände im Bildbereich berühren, d.h. einen Cluster bilden, die eine

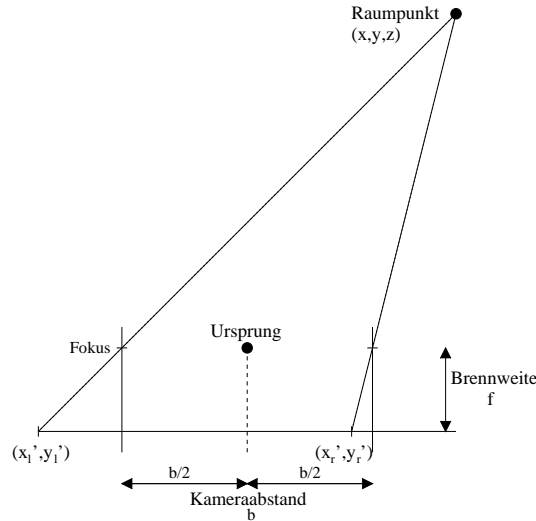


Abbildung 4.7: Darstellung der optischen Erfassung eines Raumpunktes durch ein Stereokamerapaar. Der Ursprung des Zielkoordinatensystems liegt in der Mitte zwischen den Brennpunkten der Linsen.

völlig andere Filterantwort besitzen.

### 4.3 Berechnung der Raumkoordinaten aus den Disparitätswerten

Eine Disparitätsberechnung (siehe Kap. 3.6.4) liefert in einer zweidimensionalen Darstellung Informationen über die Beschaffenheit des dreidimensionalen Raumes, der durch das Stereokamerapaar erfaßt wird. Die Kameras sind dabei entlang ihrer optischen Achse parallel ausgerichtet, im Abstand  $b$  voneinander montiert und die Brennweite ist durch  $f$  gegeben. Die Disparität entspricht dem horizontalen Abstand korrespondierender Bildpunkte. Aus der Darstellung (siehe Abb. 4.7) folgen die Beziehungen (siehe [12]):

$$\frac{x'_l}{f} = \frac{x + \frac{b}{2}}{z} \quad \text{und} \quad \frac{x'_r}{f} = \frac{x - \frac{b}{2}}{z} \quad (4.1)$$

Durch die parallele und kalibrierte Ausrichtung der Kameras ist gewährleistet, daß in beiden Kamerabildern ein Punkt jeweils in die gleiche Zeile projiziert wird, die Bildpunkte in einer epipolaren Zeile liegen. Somit gilt:

$$\frac{y'_l}{f} = \frac{y'_r}{f} = \frac{y}{z} \quad (4.2)$$

Durch eine Subtraktion der Gleichungen 4.1 entsteht eine neue Beziehung, mit Darstellung der Disparität  $d$ :

$$\frac{x'_l - x'_r}{f} = \frac{d}{f} = \frac{b}{z} \quad (4.3)$$

Somit können die Kamerakoordinaten des Raumpunktes bestimmt werden über:

$$x = b \frac{\frac{x'_l + x'_r}{2}}{x'_l - x'_r}, \quad y = b \frac{\frac{y'_l + y'_r}{2}}{x'_l - x'_r}, \quad z = b \frac{f}{x'_l - x'_r} \quad (4.4)$$

Durch Einsetzen der Disparität und unter der Annahme, daß in beiden Kamerabildern für einen Disparitätswert die Zeile identisch ist, und daß das linke Kamerabild die Disparitätswerte enthält, folgen die Kamerakoordinaten aus:

$$x = \frac{b}{d} \left( x'_l - \frac{d}{2} \right), \quad y = \frac{b}{d} y', \quad z = \frac{b}{d} f \quad (4.5)$$

Abschließend werden die Kamerakoordinaten in Weltkoordinaten umgewandelt. Der Nickwinkel der Kameras wird durch den Winkel  $\phi$  beschrieben. Eine Bewegung nach unten entspricht einem negativen Drehwinkel. In Drehrichtung entspricht einer Bewegung nach rechts dem positiven Drehwinkel  $\theta$ . Die Umrechnung erfolgt über:

$$\begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} = \begin{pmatrix} \cos(\phi) & -\sin(\phi)\sin(\theta) & -\sin(\phi)\cos(\theta) \\ \sin(\phi) & \cos(\phi)\sin(\theta) & \cos(\phi)\cos(\theta) \\ 0 & -\cos(\theta) & \sin(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (4.6)$$

Eine Projektion dieser Werte in die  $z$ -Ebene erzeugt eine Aufsicht.

## 4.4 Bilden eines gegenstandsbezogenen Objektes

Bevor Gegenstände einer Szene erkannt werden können, müssen diese in einem Lernschritt zuvor analysiert und die daraus resultierenden Ergebnisse für die weitere Verwendung gegenstandsbezogen abgespeichert werden. Um für die später Verarbeitung ein möglichst gutes Bild zu erhalten, sollte der einzelne Gegenstand mittig auf dem Tisch im Zentrum des Kamerabildes liegen. Eine Ausblendung des Tisch- und Umgebungsbereiches im Intensitätsbild extrahiert den Gegenstand aus dem Bildbereich. Innerhalb dieses extrahierten Bereiches wird in allen Farbbändern eine Farbanalyse (siehe 4.2) durchgeführt. Ferner werden in diesem Bereich die Disparitätswerte bestimmt, aus denen durch eine Umrechnung Raumpunkte errechnet werden. Um einen

Eindruck der Lage des Gegenstandes auf der Tischfläche zu erhalten, werden diese Raumpunkte in eine Aufsicht der Tischszene projiziert. Durch die Suche eines möglichst großen Clusters in der Nähe der Bildmitte wird die Aufsicht auf den Bereich dieses Clusters beschränkt. Hier wird nun die maximale Höhe des Objektes gemessen und anschließend die Ausrichtung der Aufsicht normiert. Die ermittelten Daten, d.h. die Farbbeschaffenheit, die Höhe und die Aufsicht, werden in einem Datenobjekt gegenstandsbezogen abgespeichert, so daß jeweils ein Datenobjekt einen Gegenstand beschreibt.

#### 4.4.1 Ausrichtung bestimmen

Um eine Ausrichtungsnormierung jeder Objektauf sicht zu erreichen, kann eine Hauptachsentransformation durchgeführt werden (siehe [5] und [8]), in der die Richtung der größten Ausdehnung einer Punktmenge, d.h. die größte Varianz, bestimmt wird. Demnach ist dieses Verfahren nicht eindeutig, da es zwei Lösungen gibt, jeweils um  $180^\circ$  gedreht. Dies ist jedoch für den vorliegenden Anwendungsfall ohne Bedeutung. Zur Berechnung der Hauptachse wird zunächst der Erwartungswert  $E$  der Punktmenge bestimmt. Dies entspricht dem Schwerpunkt und wird ermittelt durch:

$$\begin{pmatrix} E(x) \\ E(y) \end{pmatrix} = \frac{1}{A} \begin{pmatrix} \sum_{i=1}^A x_i \\ \sum_{i=1}^A y_i \end{pmatrix}, \text{ mit } A \hat{=} \text{Pixelanzahl} \quad (4.7)$$

Hiermit kann nun die Kovarianzmatrix  $K$  errechnet werden, die lautet:

$$\begin{aligned} K &= \begin{pmatrix} K_A & K_B \\ K_C & K_D \end{pmatrix} \\ &= \begin{pmatrix} \sum_{i=0}^A (P_{ix} - E(x))^2 & \sum_{i=0}^A (P_{ix} - E(x))(P_{iy} - E(y)) \\ \sum_{i=0}^A (P_{iy} - E(y))(P_{ix} - E(x)) & \sum_{i=0}^A (P_{iy} - E(y))^2 \end{pmatrix} \\ &\quad , \text{ mit } P = \begin{pmatrix} P_{ix} \\ P_{iy} \end{pmatrix} \hat{=} \text{Punkt } i \text{ der Punktmenge} \end{aligned} \quad (4.8)$$

Daraus ergeben sich die Eigenwerte  $EW_1$  und  $EW_2$  zu:

$$\begin{aligned} W &= \sqrt{\left(\frac{K_A + K_D}{2}\right)^2 + K_B K_C - K_A K_D} \\ EW_1 &= \frac{K_A + K_D}{2} + W \end{aligned} \quad (4.9)$$

$$EW_2 = \frac{K_A + K_D}{2} - W \quad (4.10)$$

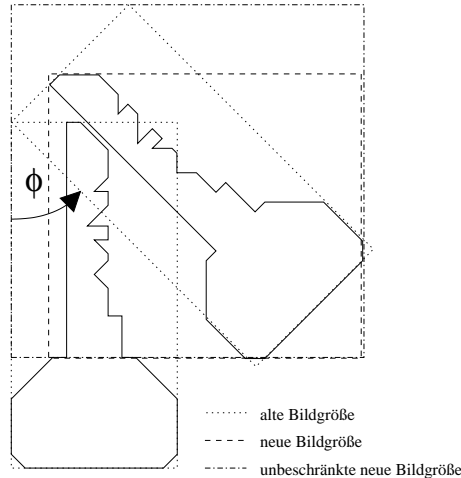


Abbildung 4.8: Drehung eines Objektes. Das Objekt wird aus seiner hier senkrechten Lage um  $45^\circ$  gedreht. Erkennbar ist die Vergrößerung des benötigten Bildausschnittes.

Um die erste Hauptachse, d.h. die Richtung mit der größten Varianz, zu erhalten, muß der Eigenvektor  $EV$  aus dem größeren Eigenwert  $EW_{max}$  der beiden Eigenwerte  $EW_1$  und  $EW_2$  berechnet werden.

$$EV_{EW_{max}} = \begin{pmatrix} 1 \\ \frac{1}{\frac{K_A + K_C - EW_{max}}{EW_{max} - K_B - K_D}} \end{pmatrix} \quad (4.11)$$

Die Hauptachse der Punktmenge, und somit des Objektes, ergibt sich aus einer Geraden, die in Richtung des Eigenvektors zeigt und durch den Schwerpunkt verläuft.

#### 4.4.2 Objekt durch Drehung normieren

Um bei der späteren Suche eine direkte Information über die Ausrichtung des Objektes zu erhalten oder ein iteratives Lernen zu ermöglichen, ist es sinnvoll, die abzuspeichernden Objektdaten normiert in einer definierten Ausrichtung abzulegen. Dadurch kann später durch eine Rotation um einen definierten Winkel direkt eine bestimmte Ausrichtung gesucht werden und bei Detektion eines Objektes dieser Ausrichtung direkt der Schwerpunkt und der Griffansatzpunkt genannt werden.

Als Rotationsbezugspunkt wird der Ursprung des Objektbildausschnittes, d.h. der Punkt oben links, gewählt (siehe Abb. 4.8). Das Objekt kann

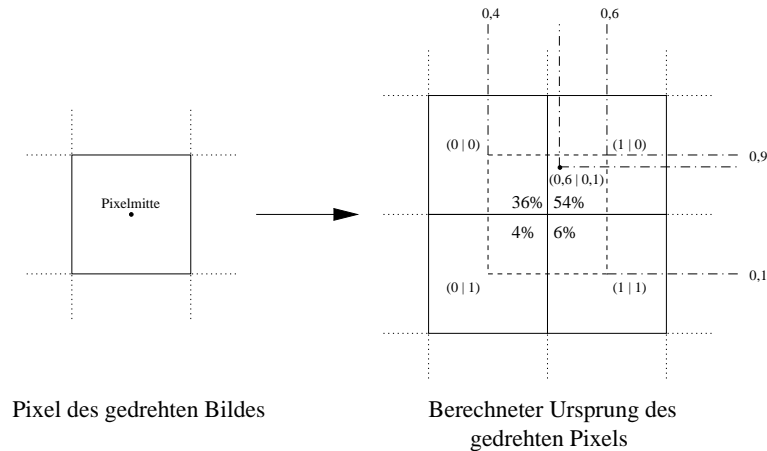


Abbildung 4.9: Interpolation des Zielpixels aus den vier Pixeln des Quellbereiches.

Pixelweise durch die Beziehung

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\Phi) & -\sin(\Phi) \\ \sin(\Phi) & \cos(\Phi) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (4.12)$$

gedreht werden. Durch Rundungsfehler entstehen jedoch Lücken im gedrehten Objekt. Für manche Anwendungszwecke reicht dies aus, jedoch ist eine inverse Berechnung

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\Phi) & \sin(\Phi) \\ -\sin(\Phi) & \cos(\Phi) \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \quad (4.13)$$

des gedrehten Objektes sinnvoller. Die inverse Bestimmung gliedert sich in drei Abschnitte.

Durch die Drehung ändert sich die Größe des Bildausschnittes, die das Objekt einnimmt (siehe Abb. 4.8). Für die inverse Berechnung muß daher zunächst durch eine normale Drehung der Eckpunkte die Größe des Bildausschnittes des gedrehten Objektes bestimmt werden. Erst dann kann die inverse Berechnung erfolgen. Ebenso wie bei der Linsenverzerrung (vgl. Abschnitt 3.5) muß dabei jedes Pixel aus vier Pixeln des Quellbereiches interpoliert werden (siehe Abb. 4.9).

Um einen möglichst kleinen Bildausschnitt des gedrehten Objektes zu erhalten, muß danach der Bildausschnitt auf das Objekt beschränkt werden, da durch die Drehung an den Rändern des neuen Bildausschnittes nicht besetzte Zeilen oder Spalten entstanden sein können (siehe Abb. 4.8). Nicht

besetzte Bereiche können das Bild so stark vergrößern, daß nur ein Viertel der Fläche nötig wäre. Für die weitere Verarbeitung würde dies durch die erhöhte Datenmenge einen enormen Anstieg der Rechenzeit bedeuten.

## 4.5 Bestimmung der Objektlage

Bereits zuvor wurde gezeigt, wie die Szenenaufsicht aus den Disparitätsdaten ermittelt werden kann. Auf Grundlage dieser Daten kann die Lage des Gegenstandes bestimmt werden, die sich aus zwei Elementen zusammensetzt:

- der Position des Objektes, angegeben durch die Koordinate der linken oberen Ecke eines Rechtecks, welches das Objekt umschließt um dessen Achsen parallel zu den Bildachsen verlaufen.
- dem Rotationswinkel des Objektes, gegenüber den das Objekt aus seiner normierten Richtung gedreht wurde. Die Rotation hat dabei keinen Einfluß auf die Bestimmung der Position, da dort das gedrehte Objekt zugrundegelegt wird.

Zur Suche eines Gegenstandes ist es daher nötig, an jeder Position des Suchbereiches durch eine Rotation der gesuchten Objektaufnahme eine möglichst gute Überdeckung beider Daten zu finden.

### 4.5.1 Bereichsbewertung durch Unschärfe

Bei der Suche nach einem Objekt besteht das Problem, daß die Objekte aus Such- und Quellbild nie ganz übereinstimmen. Ziel muß es demnach sein, auch die direkte Umgebung des Objekts zu bewerten. Dadurch können Abweichungen der Form und kleine Lageunterschiede, insbesondere Winkel-differenzen, abgeschwächt werden, so daß die Erkennung einer Überdeckung verbessert wird.

Dies geschieht hier unter Zuhilfenahme einer Unschärfewolke. Hierzu wird über jeden einzelnen gesetzten Pixel eine gewichtete Maske (siehe Abb.4.10) gelegt und die Maskenwerte in dem zu erzeugenden Unschärfebild an dieser Bildposition aufsummiert.

In anderen Verfahren kann einer Position im Unschärfebild maximal der höchste Wert einer Maske zugewiesen werden. Im Gegensatz dazu können hier durch die Summation von Maskenwerten verschiedener Bildpunkte wesentlich höhere Werte entstehen. Der Vorteil in diesem Verfahren liegt darin, daß dadurch Gebiete um so stärker gewichtet werden, desto dichter diese besetzt sind. Dadurch erhalten einzelne Pixel, die für die Suche absolut unbedeutend

1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

Abbildung 4.10: Maskenwerte zur Bewertung des Quellbildes. Die Werte der Maskenelemente sind dabei das Produkt der horizontalen und vertikalen Entfernung vom Maskenrand.

sind, nur eine sehr schwache Gewichtung. Ecken und Kreuzpunkte, die signifikante Merkmale eines Objektes darstellen, werden dagegen besonders stark gewichtet (siehe Abb. 4.11).

#### 4.5.2 Bestimmung der Position

Zur Positionsbestimmung von Objekten gibt es eine Reihe von Ansätzen. Die Bedeutendste ist hierbei die Positionsbestimmung mittels der Hausdorff-Distanz (siehe [6]). Diese bezeichnet den maximalen Abstand aus der Gesamtheit der minimalen Abstände der Punkte der Punktmenge  $A$  gegenüber allen Punkten der Punktmenge  $B$ . Der Vorteil ist eine gute Erkennung des gesuchten Bereiches, auch bei leichten Rotationen oder Verzerrungen. Der Nachteil ist jedoch eine sehr aufwendige Berechnung, da von jedem Punkt des Suchmusters zu jedem Punkt des Suchbereiches der Abstand berechnet werden muß und dies für jede mögliche Bildposition.

Erheblich schneller ist die Suche über die hier benutzten Unschärfewolken. Dabei wird das Suchobjekt über die Unschärfewolke des Suchbereichs geschoben. Als Qualität der Übereinstimmung werden dort die Werte der Unschärfewolke aufsummiert, an denen sich Punkte der Suchmaske befinden.

Ein hoher Wert dieser Unschärfesumme ist jedoch kein Garant für einen guten Treffer. Ist im Suchbild ein Bereich vorhanden, in dem sich auf engem Raum viele Objekte befinden, so wird jedes Objekt dort einen vermeintlich guten Treffer erzielen.

Zur Verifikation des Ergebnisses muß daher an dieser Stelle auch der inverse Schritt vollzogen werden, d.h. ein Bildausschnitt des Suchbereichs wird über die Unschärfewolke des Suchobjekts gelegt, die abgedeckten Werte der

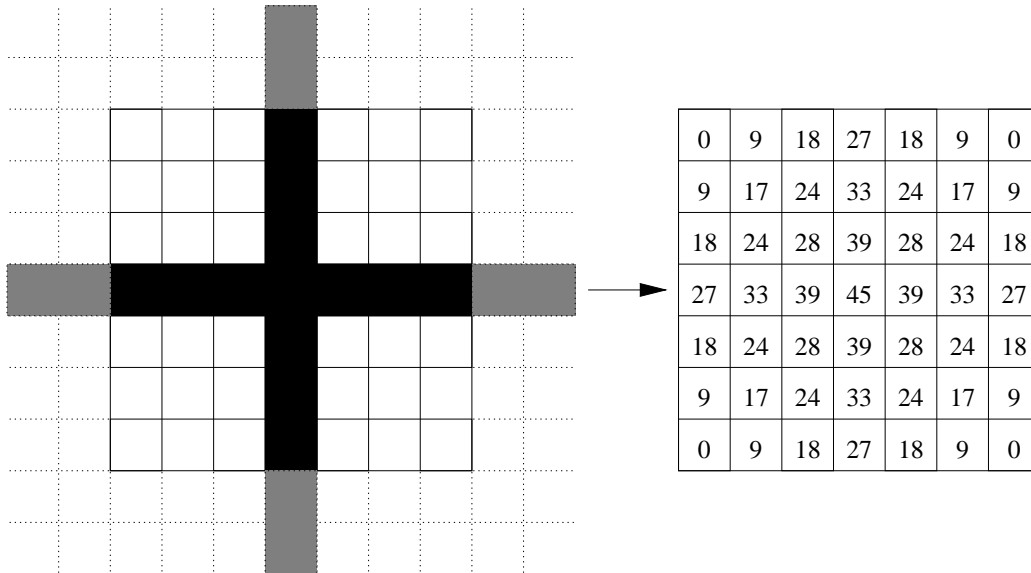


Abbildung 4.11: Die Bildung einer Unschärfe. Dieses Beispiel zeigt, daß bei zwei sich kreuzenden Geraden (links) durch Erzeugung der Unschärfe (rechts) besonders dicht besetzte Bereiche hoch gewichtet werden.

Unschärfewolke werden aufsummiert und mit dem zuvor ermittelten Ergebnis verglichen. Hierbei wird eine geringe Diskrepanz beider Werte belohnt, da es sich in diesem Fall um einen guten Treffer handeln muß.

Daraus ergeben sich einige Vorteile, die sich gegenüber der Hausdorffdistanz hauptsächlich im Zeitbedarf bemerkbar machen:

- Die Suche im Suchbereich erfolgt nur auf dem Bereich, den das Suchobjekt abdeckt. Alle anderen Punkte fließen, im Gegensatz zur Hausdorff-Distanz, nicht in die Berechnung ein.
- Die Berechnung der Qualität ist schneller, da hier nur Pixelwerte aufsummiert werden müssen. Bei der Hausdorff-Distanz müßte für jeden Pixel des Suchobjekts der Abstand zu allen Pixeln des Suchbereichs bestimmt werden.

## 4.6 Ablauf einer Lernphase

In einer Lernphase werden zunächst die beiden Kamerabilder im HSI-Format akquiriert. Anschließend werden Linsenfehler und Ungenauigkeiten der Kamerakalibrierung durch Korrekturen beseitigt. Zur Einengung des Suchbereichs

ches werden im nächsten Schritt die Gegenstände extrahiert. Dazu wird die Lage der Tischfläche im Bildbereich ermittelt. Die Bereiche außerhalb der Tischfläche werden generell gelöscht. Im Bereich der Tischfläche wird durch eine Ermittlung des Grauwertes der Tischfarbe speziell dieser so gefärbte Bereich ausgeblendet. Die Restdaten des Bildbereiches, d.h. die Objektanteile, werden auf ihre spektralen Eigenschaften hin untersucht, um essentielle Farbeigenschaften zu speichern. Dieselben Daten werden anschließend als Maske für die Bestimmung der Disparitäten genutzt. Dadurch muß nur ein Bruchteil des Bildes ausgewertet werden. Die ermittelten Disparitäten enthalten ein Maß für die Entfernung des Abbildungspunktes zum korrespondierenden Raumpunkt. Dadurch können aus den Disparitäten Raumpunkte berechnet werden, die durch eine Projektion in eine Aufsicht konvertiert werden. Die Aufsicht des zu lernenden Objektes wird aus seiner Umgebung extrahiert, in seiner Ausrichtung entsprechend seiner Hauptachse normiert und schließlich ebenfalls gespeichert.

## 4.7 Ablauf einer Suchphase

In einer Suchphase ist es die Aufgabe des Programmes, einen Gegenstand, der zuvor in der Lernphase beschrieben wurde, unter vielen auf der Tischfläche zu finden und seine Position anzugeben. Dazu werden ebenso wie in der Lernphase die Kamerabilder soweit vorverarbeitet, daß nur Bildanteile der extrahierten Gegenstände erhalten bleiben. Diese Bilddaten werden mit den im Lernschritt gewonnenen Farbeigenschaften des gesuchten Objektes gefiltert. Nur Segmente des Bildbereiches, die eine genügend hohe Filterantwort liefert, werden übernommen. Anschließend werden die Restdaten wieder wie in der Lernphase verarbeitet, bis die Szenenaufsicht berechnet ist. Auf Grundlage der Daten dieser Szenenaufsicht und der Aufsicht des gesuchten Objektes wird für verschiedene Rotationsgrade des gesuchten Objektes eine möglichst gute Deckung beider Bildbereiche, und damit die Position im Bild, gesucht.

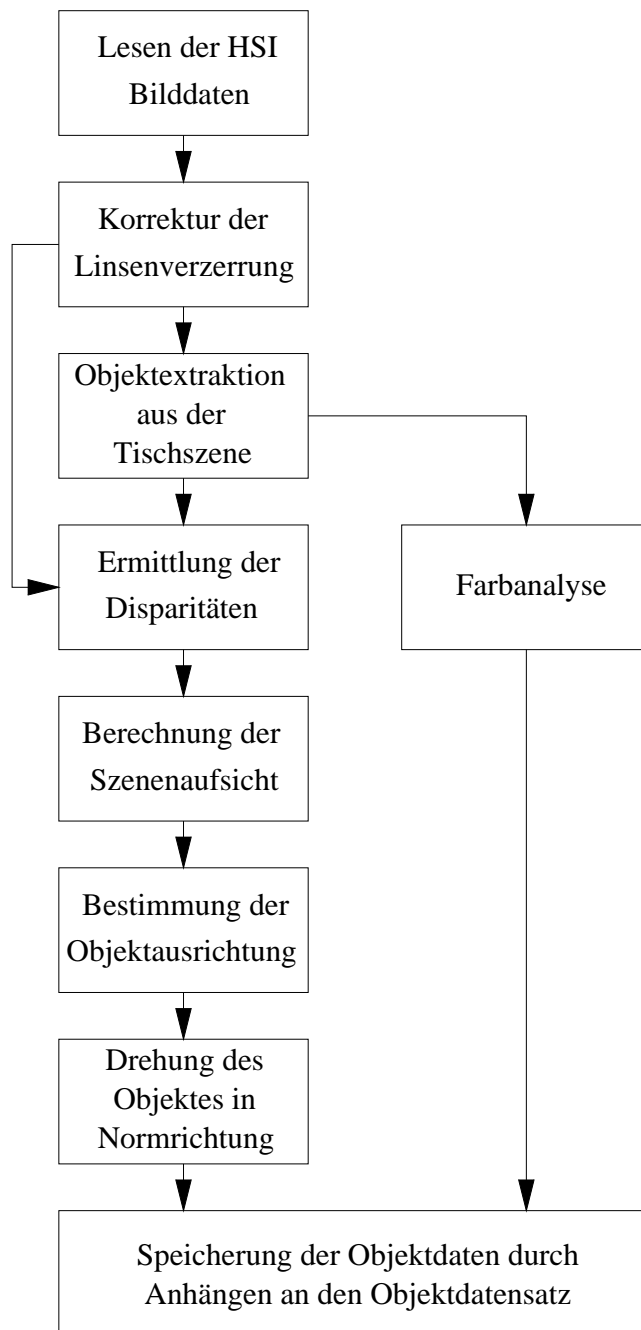


Abbildung 4.12: Schematische Darstellung eines Lernprozesses.

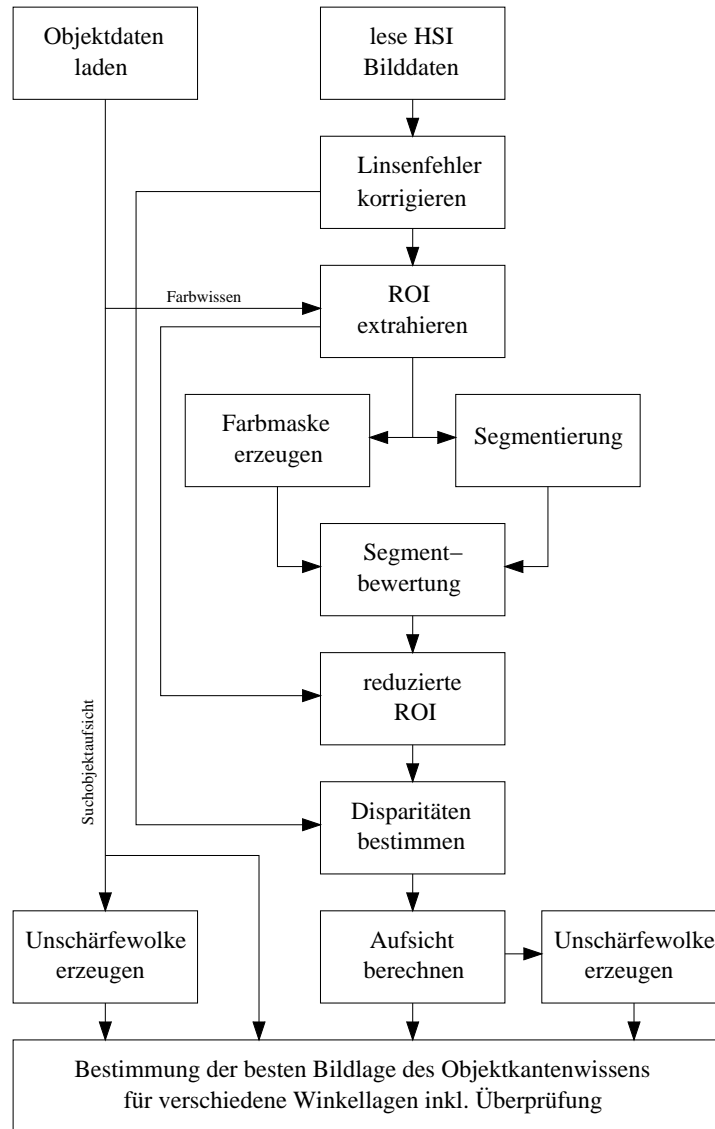


Abbildung 4.13: Schematische Darstellung eines Suchprozesses.

# Kapitel 5

## Beispiele

In diesem Kapitel wird anhand diverser Beispiele die Tauglichkeit der erarbeiteten Softwaremodule demonstriert. Die Darstellung der Ergebnisse erfolgt in zwei Schritten. Zunächst durchlaufen einige Objekte einen Lernprozeß. Später werden die darin gewonnenen Daten für diverse Objektsuchen verwendet. Zur Darstellung der Leistungsgrenzen werden diese explizit hervorgehoben und näher erläutert.

### 5.1 Beispiel Lernobjekt Schwarze Zange

Im Folgenden werden die typischen Phasen eines Lernprozesses dargestellt. Dabei wird auf spezielle Eigenarten und Probleme der verwendeten Methoden eingegangen.

Der Lernprozeß startet mit der Akquisition der Szene durch das Stereokamerapaar (siehe Abb. 5.1). Das hier dargestellte Bildpaar zeigt lediglich das Intensitätsband des HSI-Raumes.

Diese Darstellung ist jedoch für eine stereoskopische Weiterverarbeitung nicht geeignet, da Linsenfehler das Bild stark verändern. Diese Linsenfehler werden in Bezug auf den Bildhauptpunkt, der den Punkt beschreibt, der durch die optische Achse der Linse durchdrungen wird, beseitigt. Durch eine Verschiebung des Bildausschnittes zur Beseitigung der Fehler unterschiedlicher Bildhauptpunkte beider Bilder entstehen zusätzlich schwarze Ränder. (siehe Abb. 5.2).

Zur weiteren Verarbeitung müssen nicht relevante Bereiche gelöscht werden. Dazu zählen die Tischfläche und die Tischumgebung. Die Extraktion der Umgebung erfolgt durch Berechnung geometrischer Aspekte und die Tischextraktion erfolgt über Farbwissen, gekoppelt mit einem Wachstumsalgorithmus (siehe Abb. 5.3).



(a) linkes Bild



(b) rechtes Bild

Abbildung 5.1: Rohbilder der Kameras.



(a) linkes Bild



(b) rechtes Bild

Abbildung 5.2: Rohbilder nach der Linsenkorrektur.

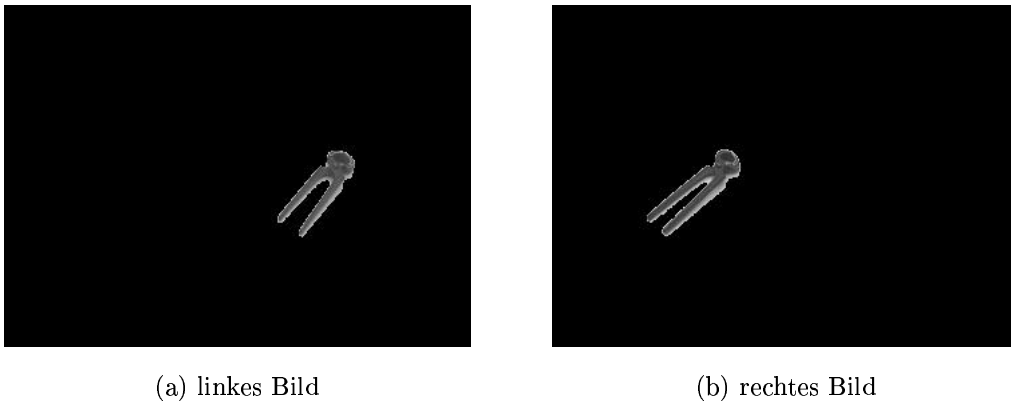


Abbildung 5.3: Die linsenkorrigierten Bilder nach der Tisch- und Hintergrundausbldung.

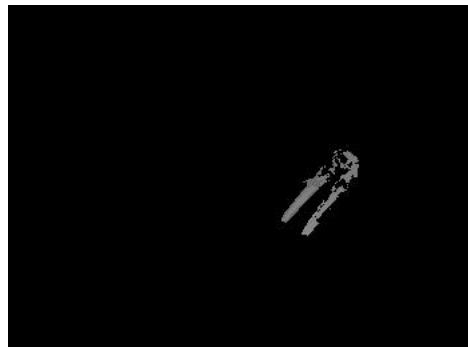


Abbildung 5.4: Ergebnis der Disparitätsbestimmung.

Durch diese Reduktion der Bilddaten müssen in den Folgeschritten weniger Bildpunkte bearbeitet werden, was zu einer erheblichen Beschleunigung der Berechnungen führt.

Zur Bestimmung der Entfernung wird an dieser Stelle die Verschiebung eines jeden Punktes, bezogen auf die beiden Kamerabilder, berechnet (siehe Abb. 5.4). Dieser Abstand, die Disparität, sinkt mit zunehmender Entfernung und beschreibt direkt die Entfernung des betrachteten Bildpunktes.

Diese Disparitätswerte werden dann unter Berücksichtigung geometrischer Gegebenheiten in eine Aufsicht umgerechnet (siehe Abb. 5.5). Diese Aufsicht ist charakteristisch für jedes Objekt und dient als Grundlage der späteren Objektsuche.

Eine Hauptachsentransformation normiert diese Datenpunkte in eine de-

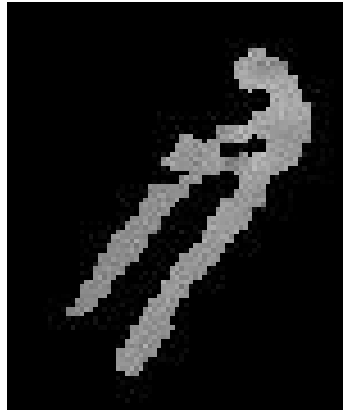


Abbildung 5.5: Berechnete Aufsicht (Ausschnitt).

finierte Richtung (siehe Abb. 5.6). Nur die gesetzten Punkte der Aufsicht werden gespeichert und dienen als Objektbeschreibung bei der Objektsuche. Als Zusatzinformation wird auch die maximale Höhe ermittelt.

Um den Suchbereich möglichst früh eingrenzen zu können, müssen weitere Eigenschaften des Objektes berücksichtigt werden. Die günstigste Möglichkeit ist die Hinzunahme der Farbinformationen, da jeder Gegenstand eine charakteristische Farbverteilung besitzt. Durch die Ermittlung des Farbspektrums des Gegenstandes (siehe Abb. 5.7) werden alle relevanten Farbwerte abgespeichert, um diese bei einer Objektsuche in einer Farbfilterung einsetzen zu können.

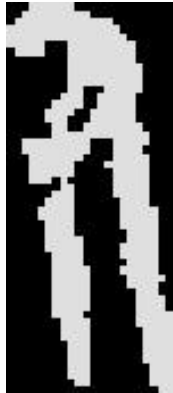


Abbildung 5.6: Darstellung der gewonnenen Objektauf sicht. Die Ausrichtung wurde durch eine Hauptachsentransformation gedreht.

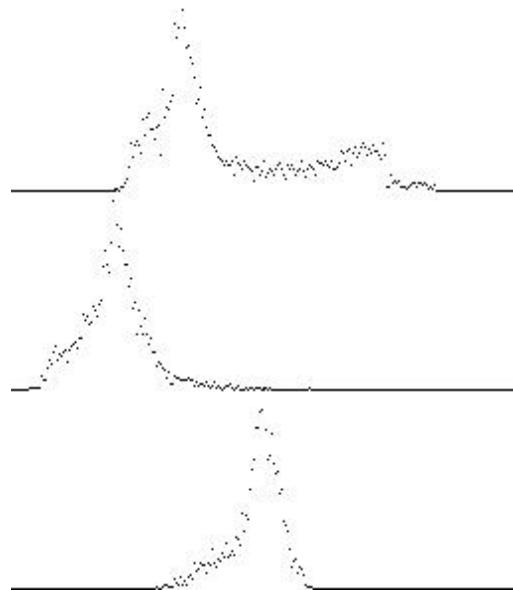
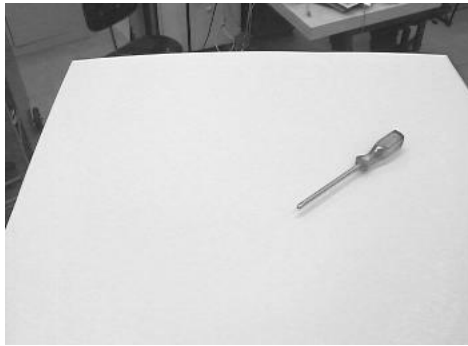
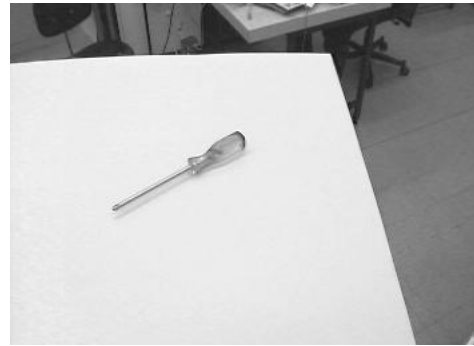


Abbildung 5.7: Spektrale Farbverteilung der Zange im HSI-Raum (H unten, S mittig und I oben).



(a) linkes Bild

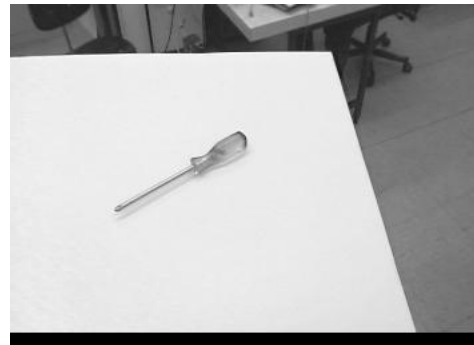


(b) rechtes Bild

Abbildung 5.8: Rohbilder der Kameras.



(a) linkes Bild



(b) rechtes Bild

Abbildung 5.9: Rohbilder nach der Linsenkorrektur.

## 5.2 Beispiel Lernobjekt Schraubendreher

Dieses Beispiel behandelt den Lernvorgang zur späteren Erkennung eines Schraubendrehers. Zu Beginn wird die Szene durch das Stereokamerapaar erfaßt (siehe Abb. 5.8).

Die Linsenfehler werden wie im vorherigen Beispiel beseitigt, so daß ein entzerrtes und in seiner Position verschobenes Bilderpaar entsteht (siehe Abb. 5.9)

Durch die Entfernung der Tischfläche und der Tischumgebung wird der Gegenstand aus der Szene extrahiert (siehe Abb. 5.10).

Die folgende Bestimmung der Disparitäten (siehe Abb. 5.11) liefert ver-

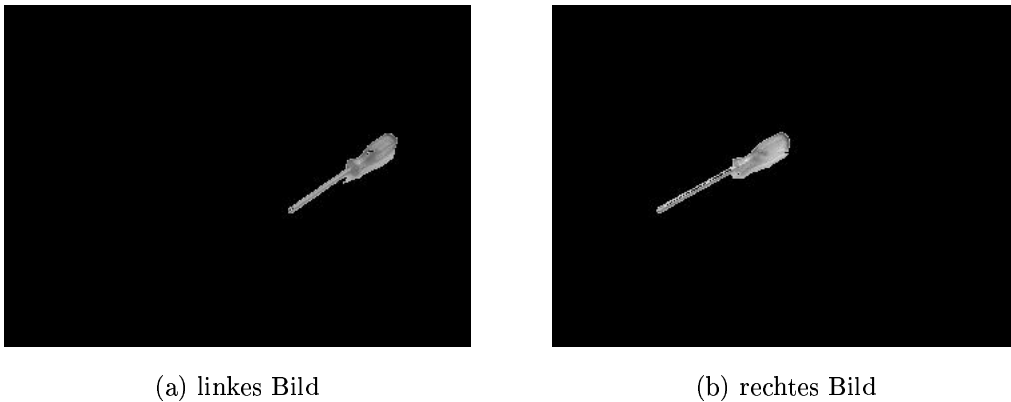


Abbildung 5.10: Die linsenkorrigierten Bilder nach der Tisch- und Hintergrundausbldung.



Abbildung 5.11: Ergebnis der Disparitätsbestimmung.

hältnismäßig wenig Disparitätswerte, was ein Zeichen dafür ist, daß viele der ermittelten Werte von zu schlechter Qualität waren, und daher nicht gesetzt wurden. Ist jedoch die Qualität der gefundenen Disparitätswerte schlecht, so werden auch einige der gesetzten Disparitäten fehlerhaft sein.

Der Grund für diese ungenaue Disparitätsfindung liegt hier in der Objektbeschaffenheit in Verbindung mit der benutzten Beleuchtung. Der Griff des Schraubendrehers ist hochglänzend und reflektiert stark das Licht. Durch die unterschiedliche Perspektive beider Kamerabilder treten diese Reflexionseffekte an verschiedenen Stellen mit unterschiedlicher Intensität auf. Dadurch wird eine exakte Bestimmung der Disparität dieser Bereiche im vorliegenden Fall nicht möglich.

Durch die fehlerhafte Bestimmung der Disparitäten wird auch die Szenen-

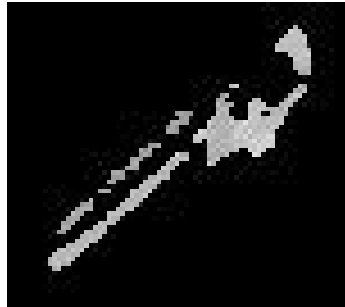


Abbildung 5.12: Berechnete Aufsicht (Ausschnitt).



Abbildung 5.13: Darstellung der gewonnenen Objektauf-sicht. Die Ausrichtung wurde durch eine Hauptachsentransformation gedreht.

aufsicht fehlerhaft berechnet (siehe Abb. 5.12). Trotz dieser Abweichung ist das resultierende Objekt (siehe Abb. 5.13) jedoch noch gut von anderen Objekten zu unterscheiden, zumal die Farbfilterung aus der Farbanalyse (siehe Abb. 5.14) andere Objekte teilweise unterdrücken wird.

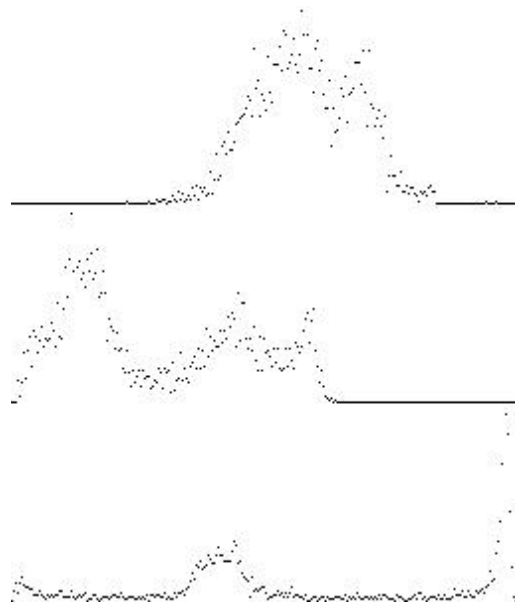
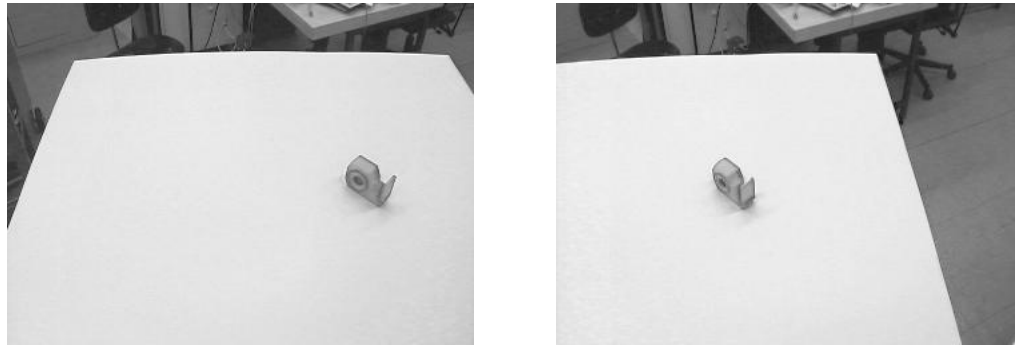


Abbildung 5.14: Spektrale Farbverteilung des Schraubendrehers im HSI-Raum (H unten, S mittig und I oben).



(a) linkes Bild

(b) rechtes Bild

Abbildung 5.15: Rohbilder der Kameras.

### 5.3 Beispiel Lernobjekt Tesarolle

In den vorangegangenen Beispielen konnte bisher nicht gezeigt werden, wie der Algorithmus auf stark strukturierte Objekte, in Bezug auf Form und Farbe, reagiert. Als Beispiel für diesen Fall dient ein Tesafilmabroller (siehe Abb. 5.15).

Nach Beseitigung der Linsenfehler und Entfernung der Tischfläche bzw. der Umgebung (siehe Abb. 5.16) werden wie bisher die Disparitäten ermittelt. Die hier ermittelten Werte (siehe Abb. 5.17) decken hauptsächlich Bereiche ab, die Objektkanten beinhalten. Diese Position von Disparitätswerten ist in der Regel ein Zeichen guter Qualität. Die Umrechnung dieser Daten in die Objektauf sicht (siehe Abb. 5.18) bestätigt, daß die gefundenen Werte von hoher Qualität sind, da das Modell der berechneten Aufsicht (siehe Abb. 5.19) sehr gut mit der Realität übereinstimmt.

Zur besseren Trennung von anderen Objekten in der Suchphase wird abschließend die Farbverteilung des Objektes gemessen (siehe Abb. 5.20).

Dieses Beispiel zeigt, daß trotz der geringen Größe und komplexen Struktur des Gegenstandes sehr gute Objektdaten ermittelt werden konnten.

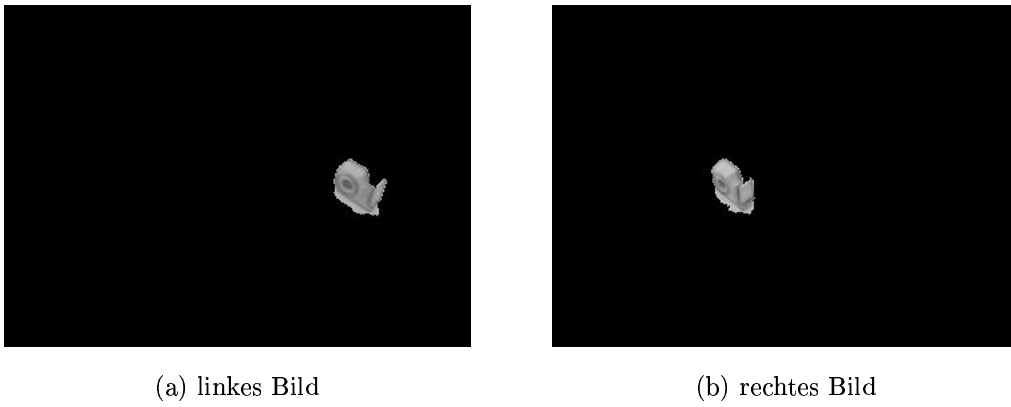


Abbildung 5.16: Die linsenkorrigierten Bilder nach der Tisch- und Hintergrundausbldung.

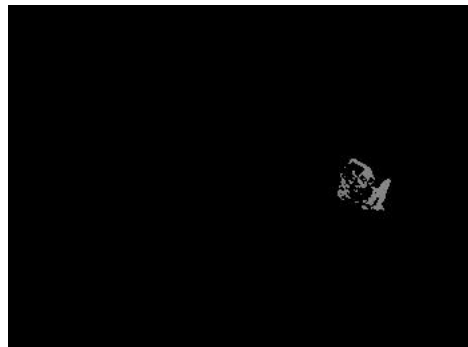


Abbildung 5.17: Ergebnis der Disparitätsbestimmung.

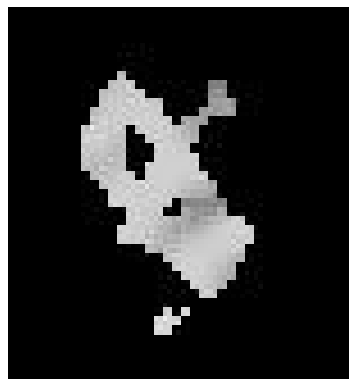


Abbildung 5.18: Berechnete Aufsicht (Ausschnitt).



Abbildung 5.19: Darstellung der gewonnenen Objektaufsicht. Die Ausrichtung wurde durch eine Hauptachsentransformation gedreht.

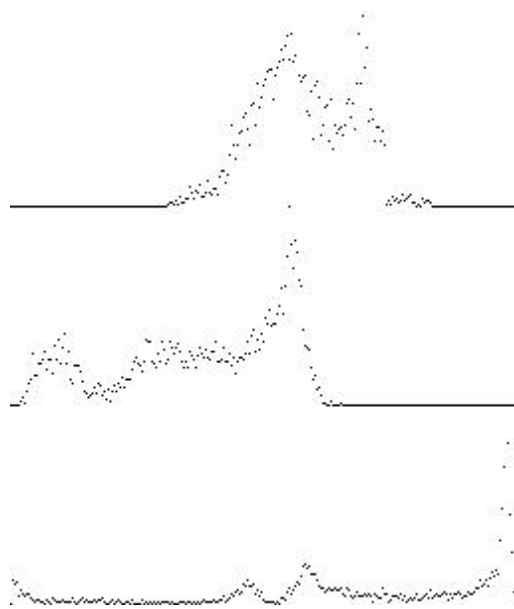
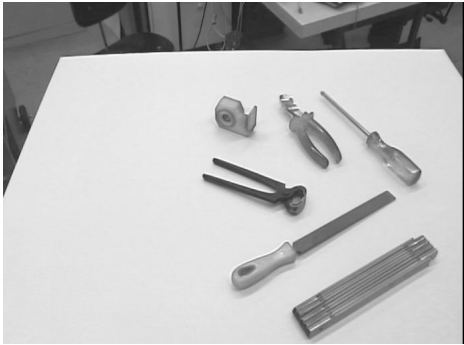
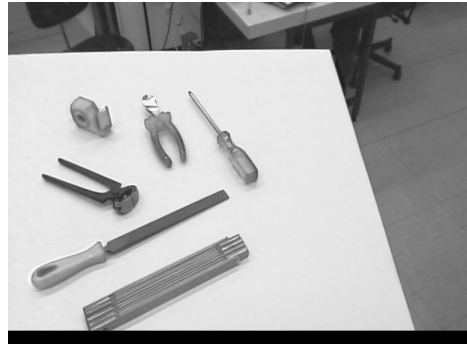


Abbildung 5.20: Spektrale Farbverteilung der Tesarolle im HSI-Raum (H unten, S mittig und I oben).



(a) linkes Bild



(b) rechtes Bild

Abbildung 5.21: Das Bilderpaar der Suchszene nach Korrektur der Linsenfehler.

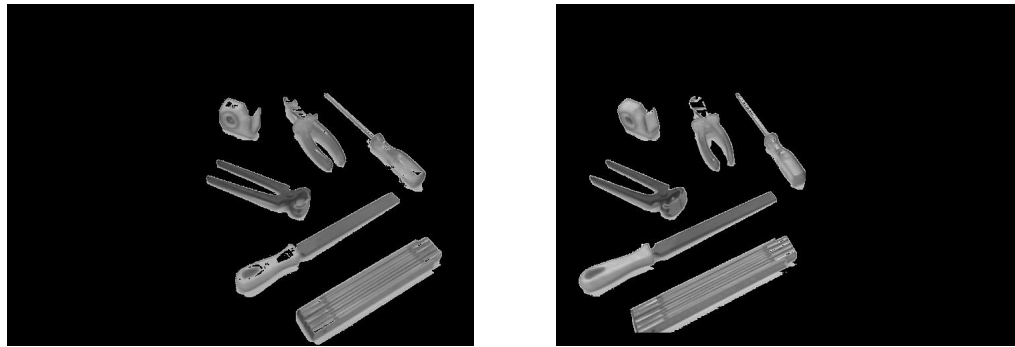
## 5.4 Beispiel Suchbereich A

In den vorangegangenen Beispielen wurden Lernabläufe in ihrer typischen Form beschrieben. Das darin über die Objekte abgelegte Wissen wird bei der Suche in einer komplexen Szene mit mehreren Gegenständen verwendet. Im Folgenden werden, wie auch schon in den Beispielen des Lernprozesses, stufenweise die Phasen einer Objektsuche (siehe Abb. 4.13) durchlaufen.

Nach der Akquisition der Bilddaten werden zunächst die Linsenfehler korrigiert (siehe Abb. 5.21). Die Szene beinhaltet in diesem Beispiel sechs verschiedene Gegenstände. Gesucht wird der Tesafilmabroller, der sich hinten links befindet. Um die Szene zu vereinfachen, werden zunächst alle Gegenstände extrahiert, indem die Umgebung unterdrückt und die Tischfläche herausgefiltert wird. Dadurch entsteht eine erheblich reduzierte Datenmenge (siehe Abb. 5.22).

Um eine weitere Datenreduktion zu erreichen, werden die extrahierten Objekte einer Farbfilterung unterzogen. Die zuvor im Lernschritt ermittelten Farbwerte des Teseabrollers im HSI-Raumes werden dabei mit denen der Bildcluster verglichen. Stimmt ein Pixel mit einer zuvor gelernten Farbe überein, erhöht sich an dieser Stelle die Filterantwort um eins. Werden an dieser Stelle in allen Bändern des Farbraums eine Übereinstimmung gefunden, erhält die Filterantwort an dieser Stelle demnach den Wert drei (siehe Abb. 5.23). Zur späteren Auswertung werden nur diese Stellen mit der maximalen Filterantwort weiterverwendet.

Diese Filterantworten werden dazu genutzt, zu überprüfen, ob innerhalb eines Clusters genügend Pixel, die den gelernten Farben entsprechen, vor-



(a) linkes Bild

(b) rechtes Bild

Abbildung 5.22: Die Szene, nachdem Hintergrund und Tischfläche entfernt worden sind.

handen sind. Bereiche geringer Filterantworten können nicht dem gesuchten Bereich entsprechen. Daher werden nur solche Cluster übernommen, die eine genügend hohe Filterantwortsdichte besitzen (siehe Abb. 5.24).

Ausschließlich in diesen Bereichen werden die Disparitätswerte (siehe Abb. 5.25) ermittelt, die danach in eine Szenenaufsicht (siehe Abb. 5.26) umgerechnet werden. Diese Szenenaufsicht wird für die Objektsuche verwendet. Hierbei wird, unter Zuhilfenahme einer Unschärfe, eine optimale Position der Suchobjektauf-sicht in den Daten der Szenenaufsicht gesucht. Das Suchobjekt wird dabei in diskreten Winkelschritten jeweils über die gesamte Szenenaufsicht bewegt. Eine Summenbildung der von dem Suchobjekt abgedeckten Unschärfewolkenpixel der Szenenaufsicht ist ein direktes Maß für die Wahrscheinlichkeit der Lage an dieser speziellen Position (siehe Abb. 5.27). Ein Rücktest kann diesen Wert dekrementieren, um eine zuverlässigere Aussage zu erhalten. Der Maximalwert aller Wahrscheinlichkeiten aller Suchobjekt-winkellagen gibt direkt die exakte Position des Suchobjektes an (siehe Abb. 5.28).

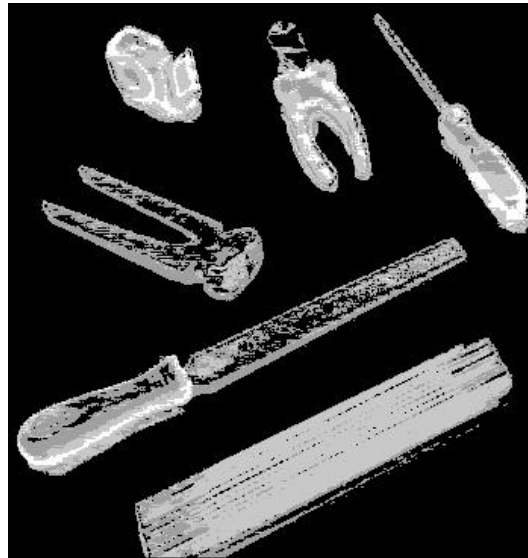


Abbildung 5.23: Die Darstellung der Filterantwort des Farbfilters. Je heller der Grauton, desto höher ist die Filterantwort.



Abbildung 5.24: Darstellung der reduzierten Objektmenge.



Abbildung 5.25: Die sich aus der reduzierten Objektmenge ergebenden Disparitäten.



Abbildung 5.26: Darstellung der berechneten Szenenaufsicht.

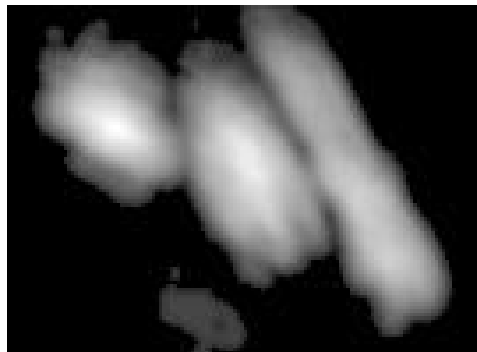
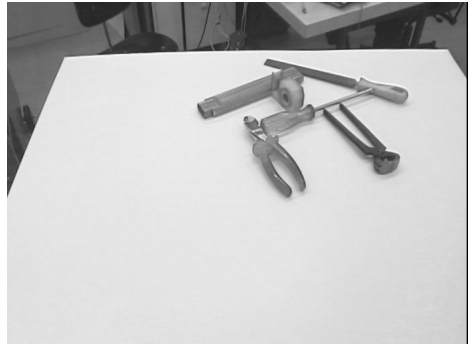


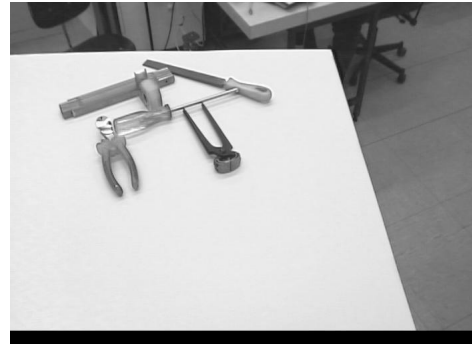
Abbildung 5.27: Darstellung der Qualität des Suchergebnisses. Je heller ein Bereich ist, desto wahrscheinlicher ist dort die Position des gesuchten Objektes.



Abbildung 5.28: Darstellung der gefundenen Position. Das hell dargestellte Objekt entspricht dem Suchdatenobjekt in seiner gefundenen Position.



(a) linkes Bild



(b) rechtes Bild

Abbildung 5.29: Darstellung der linsenkorrigierten Bilder.

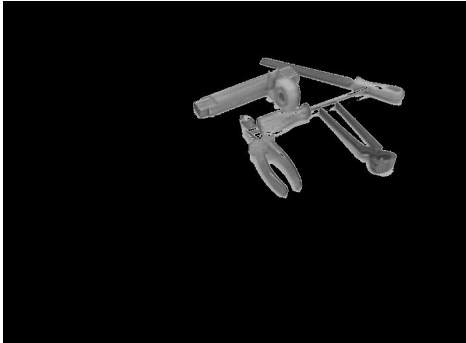
## 5.5 Beispiel Suchbereich B

Dieses Beispiel bezieht sich, wie auch das vorige Beispiel, auf die Suche nach der Tesarolle, jedoch bei einem anderen Aufbau der Szene. Durch eine hohe Packdichte der Objekte soll überprüft werden, ob das Programm invariant gegenüber leichten Verdeckungen und teilweise robust gegen direkte Nachbarschaft von Objekten ist.

Zunächst werden wie zuvor die Bilder der Szene gewonnen, von Linsenfehlern befreit (siehe Abb. 5.29) und die Gegenstände extrahiert (siehe Abb. 5.30). Die gespeicherten Farbdaten des Suchobjektes dienen zur Filterung der Szene (siehe Abb. 5.31). An dieser Stelle zeigt sich eine Besonderheit dieses Beispiels. Durch die enge Lage der Objekte ist eine Objekttrennung nicht möglich, so daß alle Bereiche übernommen werden müssen (siehe Abb. 5.32). Daher ist es nicht möglich, einzelne Bereiche auszuschließen. Dies wäre auch nicht sinnvoll, da es möglich wäre, daß Objektteile, die in der Lernphase durch die Ausrichtung des Gegenstandes nicht sichtbar waren, völlig neue Farbwerte besitzen.

Durch die Berechnung der Disparitäten (siehe Abb. 5.33) kann die Szenenaufsicht (siehe Abb. 5.34) berechnet werden.

Die Suche nach der wahrscheinlichsten Position (siehe Abb. 5.35) ergibt einen guten Treffer (siehe Abb. 5.36). Die Position ist zwar leicht nach rechts unten verschoben, doch genügt der geforderten Greifgenauigkeit.



(a) linkes Bild



(b) rechtes Bild

Abbildung 5.30: Die Szene, nachdem Hintergrund und Tischfläche entfernt worden sind.

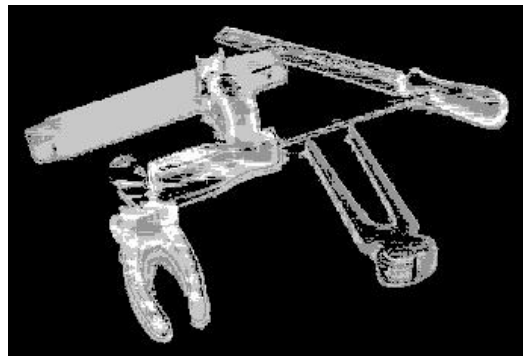


Abbildung 5.31: Die Darstellung der Filterantwort des Farbfilters. Je heller der Grauton, desto höher ist die Filterantwort.

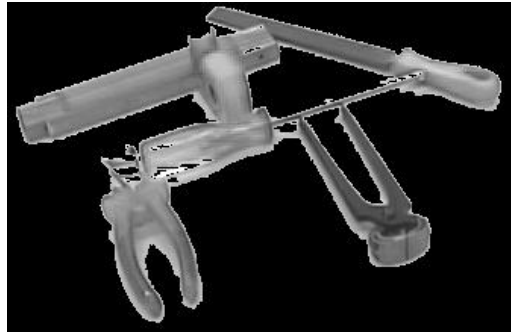


Abbildung 5.32: Darstellung der Objektmenge. Im Gegensatz zum vorherigen Beispiel konnten hier keine Bildanteile ausgeschlossen werden.

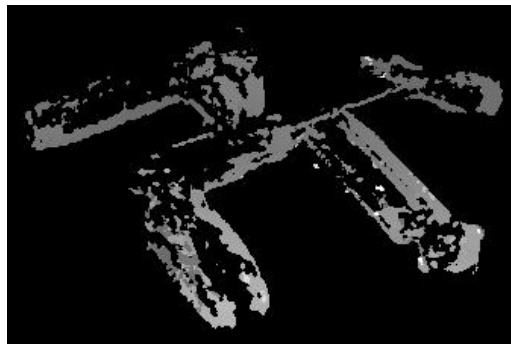


Abbildung 5.33: Die sich aus der reduzierten Objektmenge ergebenden Disparitäten.

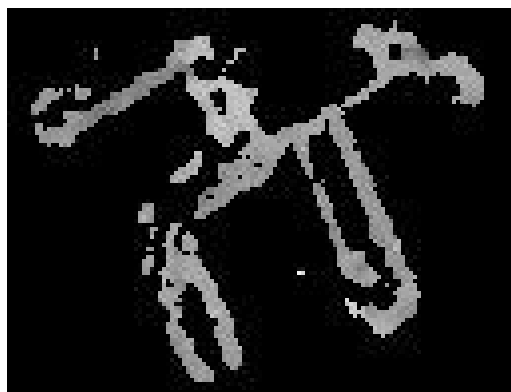


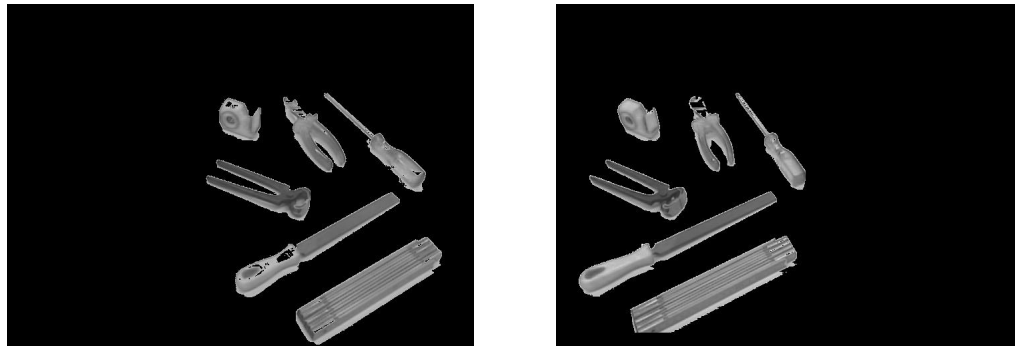
Abbildung 5.34: Darstellung der berechneten Szenenaufsicht.



Abbildung 5.35: Darstellung der Qualität des Suchergebnisses. Je heller ein Bereich ist, desto wahrscheinlicher ist dort die Position des gesuchten Objektes.



Abbildung 5.36: Darstellung der gefundenen Position. Das hell dargestellte Objekt entspricht dem Suchdatenobjekt in seiner gefundenen Position.



(a) linkes Bild

(b) rechtes Bild

Abbildung 5.37: Die Szene, nachdem Hintergrund und Tischfläche entfernt worden sind.

## 5.6 Beispiel Suchbereich C

In diesem und dem folgenden Beispiel werden die gleichen Szenen zur Objektsuche benutzt, wie in den beiden vorangegangenen Beispielen. Hier wird jedoch nicht nach der Tesarolle, sondern nach der schwarzen Zange (siehe 5.1) gesucht, die durch ihre Form und den völlig anderen und geringeren Anteilen im Farbraum ein völlig anders geartetes Problem darstellt. Da die ersten Schritte der Linsenkorrektur und der Objektextraktion bereits dargestellt wurden, werden an dieser Stelle lediglich die extrahierten Objekte nochmals dargestellt (siehe Abb. 5.37).

In diesem Beispiel besitzt die Farbfilterung (siehe Abb. 5.38) eine große Wirkung, da die Zange in ihrer Farbgebung sehr homogen ist. Dadurch ist der Filter sehr selektiv, so daß ein Großteil der anderen Gegenstände entfernt wird (siehe Abb. 5.39). Die errechneten Disparitäten (siehe Abb. 5.40) werden wie zuvor in eine Szenenaufsicht (siehe Abb. 5.41) umgerechnet.

Die Suche nach der wahrscheinlichsten Position der Zange (siehe Abb. 5.42) ergibt eine sehr genaue Übereinstimmung mit der tatsächlichen Position (siehe Abb. 5.43). Lediglich die Rotation des Objektes stimmt nicht exakt überein. Durch die Wahl kleinerer Winkelschritte bei der Objektsuche kann dies korrigiert werden.

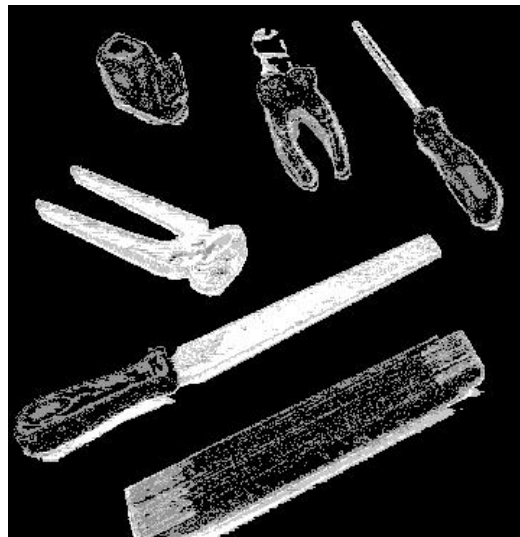


Abbildung 5.38: Die Darstellung der Filterantwort des Farbfilters. Je heller der Grauton, desto höher ist die Filterantwort.



Abbildung 5.39: Darstellung der reduzierten Objektmenge.

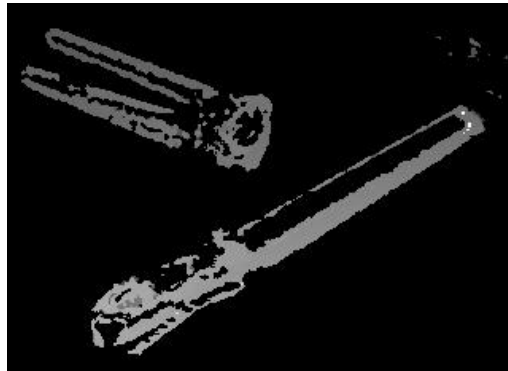


Abbildung 5.40: Die sich aus der reduzierten Objektmenge ergebenden Disparitäten.

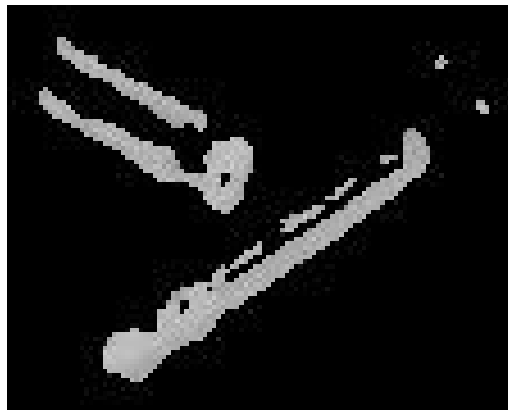


Abbildung 5.41: Darstellung der berechneten Szenenaufsicht.

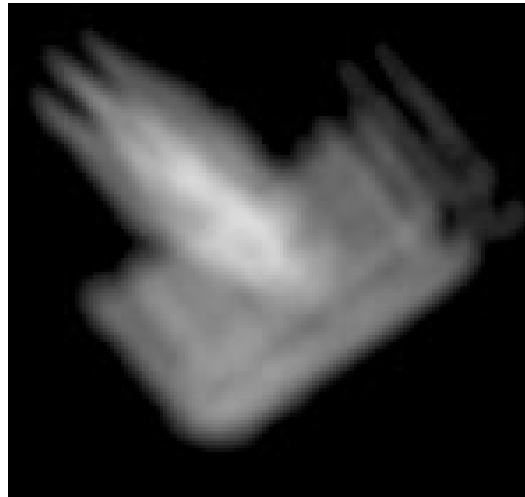


Abbildung 5.42: Darstellung der Qualität des Suchergebnisses. Je heller ein Bereich ist, desto wahrscheinlicher ist dort die Position des gesuchten Objektes.

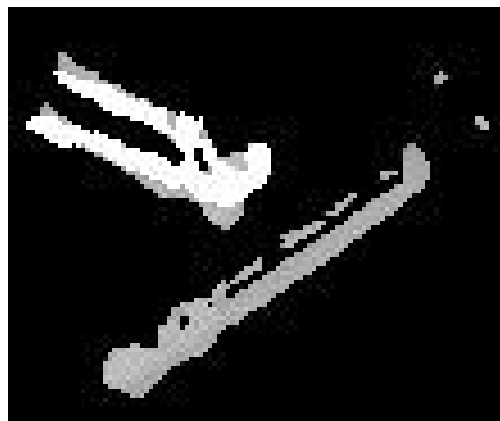
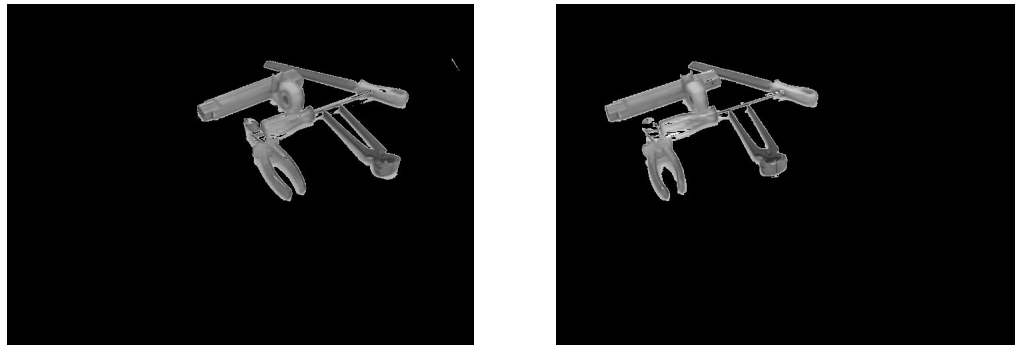


Abbildung 5.43: Darstellung der gefundenen Position. Das hell dargestellte Objekt entspricht dem Suchdatenobjekt in seiner gefundenen Position.



(a) linkes Bild

(b) rechtes Bild

Abbildung 5.44: Die Szene, nachdem Hintergrund und Tischfläche entfernt worden sind und Linsenfehler beseitigt wurden.

## 5.7 Beispiel Suchbereich D

Nachdem die Tesarolle auch in einer eng besetzten Szene gefunden wurde, soll die Robustheit des Programms durch die Suche nach der schwarzen Zange verifiziert werden. Die Szene (siehe Abb. 5.44) ist identisch mit der des zweiten Beispiels. Der Farbfilter kann die Zange zwar gut isolieren (siehe Abb. 5.45), doch da alle Gegenstände in einem Cluster zusammenhängen, kann kein Bereich entfernt werden. Somit unterscheidet sich das Folgebild (siehe Abb. 5.46) nicht von dem aus dem Beispiel, in dem nach der Tesarolle gesucht wurde. Die ermittelten Disparitäten (siehe Abb. 5.47) und die berechnete Szenenaufsicht (siehe Abb. 5.48) sind daher ebenfalls identisch.

Die Suche nach der Objektlage ist in diesem Beispiel nicht sehr eindeutig (siehe Abb. 5.49), da verschiedene Objekte oder Objektteile durch eine bestimmte Anordnung eine große Ähnlichkeit mit dem gesuchten Objekt besitzen. Das gesuchte Objekt wird zwar gefunden (siehe Abb. 5.50), jedoch um  $180^\circ$  gedreht.

Die Auswirkungen einer zu hohen Winkeliteration des Suchobjektes zeigt sich bei großen Objekten wie dieser Zange besonders gut. Da die Ausrichtung der Zange in der Szene genau zwischen zwei Winkelschritten liegt, wird die richtige Position nur schlecht erkannt. Andere Bereiche können in solchen Fällen bessere Übereinstimmungen bieten, als das eigentliche Objekt, so daß eine Fehlzuordnung wahrscheinlich ist (siehe Abb. 5.51).

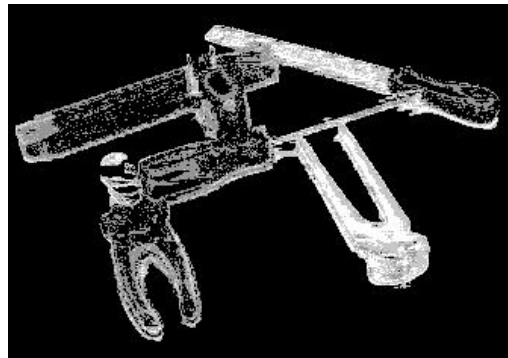


Abbildung 5.45: Die Darstellung der Filterantwort des Farbfilters. Je heller der Grauton, desto höher ist die Filterantwort.



Abbildung 5.46: Darstellung der reduzierten Objektmenge.



Abbildung 5.47: Die sich aus der reduzierten Objektmenge ergebenden Disparitäten.



Abbildung 5.48: Darstellung der berechneten Szenenaufsicht.



Abbildung 5.49: Darstellung der Qualität des Suchergebnisses. Je heller ein Bereich ist, desto wahrscheinlicher ist dort die Position des gesuchten Objektes.



Abbildung 5.50: Darstellung der gefundenen Position. Das hell dargestellte Objekt entspricht dem Suchobjekt in seiner gefundenen Position.



Abbildung 5.51: Darstellung der gefundenen Position mit einer zu großen Winkelschrittweite. Das hell dargestellte Objekt entspricht dem Suchobjekt in seiner gefundenen Position.



# Kapitel 6

## Zusammenfassung und Ausblick

Ziel ist die Entwicklung eines Bildverarbeitungsmoduls zur stereoskopischen Lokalisierung und Erkennung von Objekten im Greifraum eines autonomen Montageroboters. Um ein Objekt in einer Szene erkennen zu können, müssen diverse Objekteigenschaften bereits bekannt, d.h. gelernt worden sein. Daher wird im Rahmen dieser Arbeit die Erfassung von Objekteigenschaften eines Einzelobjektes und die Suche nach einem Objekt in einer komplexen Szene getrennt voneinander behandelt.

In einem ersten Schritt muß ein Objekt anhand von Objekteigenschaften erfaßt werden. Die Bilder des Stereokamerapaares werden dazu in geeigneter Form aufbereitet. Die Bilddaten im HSI-Format werden dabei zunächst von Linsen- und Kamerafehlern befreit. Um die Berechnungen auf das Wesentliche zu beschränken, wird das auf der Tischoberfläche liegende zu lernende Objekt aus der Szene extrahiert. Auf Grundlage dieser beschränkten Daten wird sowohl eine Farbanalyse des Objektes als auch eine Disparitätsermittlung durchgeführt. Die Farbanalyse ermittelt die charakteristischen Farbwerte jedes Bandes des HSI-Raumes. Dadurch können bei einer späteren Suche Bereiche allein aufgrund ihrer Farbverteilung ausgeschlossen werden. Die Ermittlung der Disparitätswerte wird weiter zur Berechnung einer Szenenaufsicht genutzt. Hierdurch entsteht eine charakteristische Aufsicht des zu lernenden Objektes, welches in seiner Größe, gegenüber der normalen Bildansicht für jede Position dieses Gegenstandes auf der Tischfläche, konstant ist. Diese Aufsicht wird später zur Suche in komplexeren Szenen genutzt.

Im zweiten Schritt kann aufgrund der gelernten Objekteigenschaften nach dem Gegenstand gesucht werden. Dazu werden die gleichen Schritte wie in der Lernphase vollzogen, bis die Gegenstände der Szene extrahiert sind. An dieser Stelle könnte bereits eine Disparitätssuche erfolgen, doch diese Suche ist sehr zeitaufwendig, so daß an dieser Stelle zuvor eine Datenreduktion erfolgt. Hierfür werden die bereits gelernten Farbwerte zur Maskierung

der extrahierten Objekte benutzt. Nur wenn ein Objektcluster ausreichend gut maskiert wird, d.h. die Farbgebung mit der des gelernten Objektes gut übereinstimmt, werden diese Bereiche übernommen. Auf den verbleibenden Bildbereichen wird eine Disparitätsermittlung durchgeführt und die Werte in eine Szenenaufsicht konvertiert. Zu diesem Zeitpunkt liegen die Aufsicht des gelernten Objektes und der reduzierten Objektszene vor. Indem von jeder Aufsicht ein Unschärfbild erzeugt wird, werden beide Aufsichten jeweils mit dem Unschärfbild der anderen Aufsicht korreliert und bewertet. Das Ergebnis ist eine Position mit der höchsten Wahrscheinlichkeit der Objektlage in der Suchszenen. Da das gesuchte Objekt gedreht sein kann, muß diese Korrelation für verschiedene Rotationswinkel durchgeführt werden. Je größer das Objekt ist, desto feiner müssen die Winkelschritte gewählt werden. Es genügt meist ein Winkel zwischen  $15^\circ$  und  $30^\circ$ .

Die Objektklassifikation wäre auch direkt über die Kameraansicht möglich gewesen. Der Nachteil eines solchen Verfahrens ist der wesentlich höhere Aufwand bei eingeschränktem Funktionsumfang. Zur Klassifizierung eines einzelnen Objektes wären für dieses Verfahren diverse Ansichten des Objektes nötig, um dieses vollständig zu beschreiben. Darüber hinaus wäre auch eine Skalierung der Objektgröße für verschiedene Entfernungen nötig. Ein einzelnes Objekt wäre somit durch eine große Anzahl von Objektdatensätzen definiert, da die Einzelansichten nicht oder nur bedingt fusionsfähig sind. Im Gegensatz dazu wird durch die Objektklassifizierung durch eine einzige Aufsicht der Aufwand zu Objektbestimmung drastisch reduziert. Der Vorteil dieses Verfahrens liegt demnach in der schnelleren und universelleren Einsatzmöglichkeit.

Besonders die Geschwindigkeit spielt bei einem solchen Verfahren eine große Rolle. Durch geeignete Maßnahmen zur Beschleunigung der Berechnungen wird bei einer Suche eines großen Objektes in einer stark besetzten Szene für ein Bild mit voller Auflösung eine Zeit von fast zehn Sekunden benötigt. Da dies jedoch die Ergebnisse eines Sun UltraSparc-Rechners sind, der wesentlich langsamer als ein einzelnes Rechnermodul von CORA ist und das Bild ohne weiteres auch mit halber oder sogar viertel Auflösung analysiert werden kann, ist eine Echtzeitfähigkeit, d.h. eine ausreichende Reaktionsgeschwindigkeit für den geplanten Anwendungsfall, möglich.

Ergebnis dieser Arbeit ist eine Objektsuche, die, bezogen auf die Erkennung von Gegenständen auf der Arbeitsfläche des Roboters CORA, sowohl translations- als auch rotationsinvariant ist. Die Genauigkeit der Positionsbestimmung von Objekten genügt dabei der geforderten Genauigkeit für einen Greifvorgang.

# Anhang A

## Anhang

### A.1 Programmbausteine

In diesem Abschnitt werden die einzelnen Programmodule alphabetisch geordnet kurz erklärt, indem die Objektschnittstelle und die Einsatzmöglichkeiten erläutert werden. Abschließend ist der Umfang des Moduls durch die Zeilenanzahl gegeben.

#### A.1.1 *Despeckle.h*

```
void Despeckle(unsigned char** k,  
               int* h,  
               int* v,  
               int g)
```

Die Prozedur *Despeckle* entfernt durch einen Medianfilter einzelne Störpixel eines Bildes wie Bildschnee oder auch Fische. Die Ein- und Ausgabe des Bildes erfolgt über die Variable *k*. Die Variablen *h* und *v* beinhalten die horizontale bzw. vertikale Auflösung des Bildes und *g* gibt die ungerade Kantlänge der quadratischen Maske an. Dabei sind nur die Werte 3,5,7 und 9 zulässig. Andere Maskengrößen wurden nicht implementiert.

Umfang: 80 Zeilen

#### A.1.2 *DespeckleLinear.h*

```
void DespeckleLinear(unsigned** k,  
                    int h,  
                    int g)
```

Die Prozedur *DespeckleLinear* entfernt durch einen Medianfilter einzelne Störpixel einer linearen Datenmenge und führt somit zu einer Glättung, z.B. einer Funktionskurve. Die Ein- und Ausgabe des Bildes erfolgt über die Variable *k*. *h* gibt die Anzahl der linearen Datenpunkte an und *g* bezeichnet die Größe der Filtermaske, die zur Zeit nur den Wert fünf annehmen darf.

Umfang: 50 Zeilen

### A.1.3 Dilatation.h

```
void Dilatation(unsigned char **p,
               int *h,
               int *v,
               int g,
               int Schwelle)
```

Die Prozedur *Dilatation* führt eine Dilatation aus. Die Ein- und Ausgabe des Bildes erfolgt über die Variable *k*. *h* und *v* bezeichnen die horizontale und vertikale Auflösung des Bildes *p*. Die Kantenlänge der quadratischen Maske ist durch die Variable *g* gegeben. Der für die Dilatation nötige Schwellwert wird durch *Schwelle* gegeben. Obwohl es sich bei einer Dilatation normalerweise um ein binäres Verfahren handelt, werden hier die Grauwerte eines Clusters rekonstruiert.

Umfang: 990 Zeilen

### A.1.4 ErkenneDingDa.h

```
void BeWolken(unsigned char **bild,
              int *h,
              int *v,
              unsigned char **Wolke)
```

Die Prozedur *BeWolken* erzeugt aus dem Eingabebild *bild* ein Unschärfebild und gibt dieses Bild durch *Wolke* aus. Die Variablen *h* und *v* bezeichnen die horizontale und vertikale Auflösung der Bilder.

Umfang: 50 Zeilen

```
void FindeRichtung(unsigned **Punkte,
                  unsigned *Anzahl,
                  unsigned *Winkel)
```

Die Prozedur *FindeRichtung* analysiert eine Anzahl von *Anzahl* Punkten, die in *Punkte* sequentiell jeweils mit x- und dann y-Koordinate abgelegt sind. Die Variable *Winkel* liefert den Winkel der ersten Hauptachse in Grad.

Umfang: 70 Zeilen

```
void FindeBildRichtung(unsigned char **Punkte,
                      unsigned *Hori,
                      unsigned *Verti,
                      unsigned *Winkel)
```

Die Prozedur *FindeBildRichtung* errechnet den Winkel *Winkel* der ersten Hauptachse aller gesetzten Punkte (Grauwert ungleich 0) des Bildes *Punkte*. Die horizontale und vertikale Bildauflösung ist durch die Variablen *Hori* und *Verti* gegeben.

Umfang: 80 Zeilen

```
void MacheBildWolke(unsigned **bild,
                   unsigned *AnzahlPunkte,
                   int *h,
                   int *v,
                   unsigned char **Wolke)
```

Die Prozedur *MacheBildWolke* erzeugt eine Unschärfewolke auf Grundlage binärer Bilddaten. Die Bilddaten liegen in Koordinatenform, d.h x-Koordinate gefolgt von der y-Koordinaten des Bildpunktes, sequenziell in der Variablen *bild*. Die Anzahl aller Bildpunkte ist durch *Anzahl* gegeben. Die Variablen *h* und *v* bezeichnen die benötigte horizontale und vertikale Auflösung des Bildes. Die Variable *Wolke* liefert die Unschärfewolke.

Umfang: 50 Zeilen

```
unsigned SucheBildObjekt(unsigned char **bild,
                        unsigned char **bildorig,
                        int *h,
                        int *v,
                        unsigned char **WolkBild,
                        unsigned char **KoorBild,
                        unsigned *hk,
                        unsigned *vk,
                        unsigned *BesterKoorx,
                        unsigned *BesterKoory)
```

Die Funktion *SucheBildObjekt* liefert die Position eines Suchobjektes innerhalb eines Bildes. Als Bilddaten liegt jedes Bild binär und als Unschärfewolke vor. Die Variable *bildorig* beinhaltet das Bild, in dem die Position des Objektes *KoorBild* gesucht wird. Die Variablen *bild* und *WolkBild* beinhalten die zuvor ermittelten Unschärfewolken der Bilddaten von *bildorig* und *KoorBild*. Durch *h* und *v* bzw. *hk* und *vk* sind die horizontalen und vertikalen Auflösungen der Bilder gegeben. Die gefundene Position wird über die Variablen *BesterKoorx* und *BesterKoory* ausgegeben. Der Rückgabewert der Funktion ist ein Qualitätsmaß der Treffergüte. Je höher dieser Wert ist, desto wahrscheinlicher ist die gefundene Position.

Umfang: 100 Zeilen

```
unsigned SucheObjekt(unsigned char **bild,
                    unsigned **bildorig,
                    unsigned *Menge,
                    int *h,
                    int *v,
                    unsigned char **WolkBild,
                    unsigned **KoorBild,
                    unsigned *Anzahl,
                    unsigned *hk,
                    unsigned *vk,
                    unsigned *BesterKoorx,
                    unsigned *BesterKoory)
```

Die Funktion *SucheObjekt* liefert die Position eines Suchobjektes innerhalb eines Bildes. Als Bilddaten liegt jedes Bild binär und als Unschärfewolke vor. Die Variable *bildorig* beinhaltet das Bild, in dem die Position des Objektes *KoorBild* gesucht wird. Beide dieser Bilder liegen sequenziell in Koordinatenform, d.h. jeweils x-Koordinate gefolgt von der y-Koordinaten, vor. Die Anzahl der gesetzten Bildpunkte ist im Bild *bildorig* durch *Menge* und im Bild *KoorBild* durch *Anzahl* gegeben. Die Variablen *bild* und *WolkBild* beinhalten die zuvor ermittelten Unschärfewolken der Bilddaten von *bildorig* und *KoorBild*. Durch *h* und *v* bzw. *hk* und *vk* sind die horizontalen und vertikalen Auflösungen der Bilder gegeben. Die gefundene Position wird über die Variablen *BesterKoorx* und *BesterKoory* ausgegeben. Der Rückgabewert der Funktion ist ein Qualitätsmaß der Treffergüte. Je höher dieser Wert ist, desto Wahrscheinlicher ist die gefundene Position.

Umfang: 100 Zeilen

### A.1.5 Erosion.h

```
void Erosion(unsigned char** p,
            int* h,
            int* v,
            int g,
            int Schwelle,
            char Mode)
```

Die Prozedur *Erosion* führt eine Erosion auf Binär- oder Graustufenbildern durch. Das Eingabebild wird durch die Variable *p* gegeben. Dessen horizontale und vertikale Auflösung werden mit *h* und *v* bezeichnet. Die Variable *g* entspricht der ungeraden Kantenlänge der quadratischen Erosionsmaske. Gültige Eingabewerte sind 3, 5, 7 und 9. Die Besetzungsschwelle zur Erosion ist durch *Schwelle* gegeben. Durch *Mode* kann durch *B* (Binärbild) oder *G* (Graustufenbild) gewählt werden, ob das Ergebnisbild ein Binär- oder Graustufenbild sein soll.

Umfang: 400 Zeilen

### A.1.6 FaerbDisp.h

```
void FaerbDisp(unsigned *ZahlHue,
              unsigned **WertHue,
              unsigned char **HueBild,
              unsigned *ZahlSat,
              unsigned **WertSat,
              unsigned char **SatBild,
              unsigned *ZahlInt,
              unsigned **WertInt,
              unsigned char **IntBild,
              unsigned char **IntFaerbBild,
              unsigned *h,
              unsigned *v)
```

Die Prozedur *FaerbDisp* benutzt eingegebene Farbwerte zur Erstellung einer Maske. Dabei werden von den Farbwerten belegte Stellen im Eingabebild in der Maske um eins inkrementiert. Da insgesamt drei Farbbänder vorhanden sind, ist der Maximalwert eines Maskenpunktes drei. Die Eingabe der einzelnen Farbbänder erfolgt über die Variablen *HueBild*, *SatBild* und *IntBild*, entsprechend den Bändern des HSI-Bildes. Die Anzahl der Farbwerte, die für jeweils ein Band genutzt werden, werden durch *ZahlHue*, *ZahlSat* und *ZahlInt*

gegeben. Die Variablen *WertHue*, *WertSat* und *WertInt* beinhalten sequentiell die einzelnen Werte der Farbwerte, die zur Filterung benutzt werden. Die Ausgabe der Filtermaske erfolgt über *IntFaerbBild*. Alle Bilder besitzen die horizontale und vertikale Auflösung *h* und *v*.

Umfang: 60 Zeilen

### A.1.7 FindDisp.h

```
void FindDisp(unsigned char** li,
              unsigned char** ri,
              unsigned char** lio,
              unsigned char** rio,
              int* h,
              int* v,
              unsigned char** d,
              float* scale)
```

Die Prozedur *FindDisp* ermittelt anhand eines stereoskopischen Bilderpaars die Disparitäten der Bildpunkte. Die Variablen *li* und *ri* stehen für die Eingangsgrößen des linken und rechten Kamerabildes. Damit nicht auf dem gesamten Bildbereich die Disparitäten bestimmt werden müssen, kann der Suchbereich über die Maske *rio*, die durch die Prozedur *FaerbDisp* erzeugt wurde, eingeengt werden. Die Maskeneingabe über *lio* steht nicht zur Verfügung. Die horizontale und vertikale Auflösung des Bildes werden durch die Variablen *h* und *v* gegeben. Die Ausgabe der Disparitätswerte erfolgt über *d*. Um eine konstante Größe des Disparitätsbildes zu gewährleisten, wird über *scale* ein Skalierungsfaktor ausgegeben.

Umfang: 210 Zeilen

### A.1.8 FindeFarben.h

```
void FindeFarben(unsigned char **hue,
                 unsigned char **sat,
                 unsigned char **inty,
                 int *h,
                 int *v,
                 unsigned **Daten)
```

Die Prozedur *FindeFarben* extrahiert dominante Farbanteile der einzelnen Farbbänder eines Bildes. Die Eingabe des Bildes erfolgt für jedes Band einzeln

über *hue*, *sat* und *inty*. Die Variablen *h* und *v* enthalten die horizontale und vertikale Bildauflösung. Die Ausgabe der ermittelten Farbwerte erfolgt durch die Variable *Daten*. Die ersten drei Stellen dieses Datensatzes beinhalten die Anzahl der Farbwerte des H-, S- und I-Bandes. In gleicher Reihenfolge folgen die Farbwerte der einzelnen Bänder.

Umfang: 120 Zeilen

### A.1.9 FindeKanten.h

```
unsigned FindeKanten(unsigned char **bild,
                    unsigned *h,
                    unsigned *v,
                    unsigned Schwelle,
                    unsigned Art,
                    unsigned Modus,
                    unsigned **KoorBild)
```

Die Funktion *FindeKanten* extrahiert Kanten aus einem Eingabebild, wobei verschiedene Optionen zur Art der Kantenfindung möglich sind. Die Eingabe des Bildes erfolgt über *bild*. Die horizontale und vertikale Auflösung des Bildes ist durch die Variablen *h* und *v* gegeben. Die für den Kantenoperator notwendige Schwelle wird durch *Schwelle* und die Art der Kantenfindung durch die Variable *Art* angegeben. Die Werte 1, 2 und 3 geben dabei die horizontale & vertikale, die nur horizontale und die nur vertikale Kantenfindung an. Durch *Modus* wird die Datenart der Ausgabedaten *KoorBild* festgelegt. Ist für *Modus* 1 gesetzt, so wird als Ausgabe ein Bild geliefert. Ist der Wert für *Modus* 3 oder 4, so werden nur die Kantenpixel als Koordinaten sequentiell ausgegeben. Ist für *Modus* 4 gesetzt, werden Kanten, die sich an Bereichen mit dem Grauwert 0 bilden, nicht berücksichtigt. Für die Fall, daß *Modus* den Wert 3 oder 4 besitzt, liefert die Funktion die Anzahl der gesetzten Kantenpixel.

Umfang: 530 Zeilen

### A.1.10 LadeDaten.h

```
void LadeDaten(PObjekt *Elementptr)
```

Die Prozedur *LadeDaten* lädt alle Daten zur Objektbeschreibung in eine lineare Liste.

Umfang: 170 Zeilen

### A.1.11 LeseBildDaten.h

```
void LeseBildDaten(char* Dateiname,
                   unsigned char **l,
                   int *h,
                   int *v)
```

Die Prozedur *LeseBildDaten* lädt eine Datei im pgm-Format. Die Variable *Dateiname* gibt den Namen der Datei an, die geladen werden soll. Durch die Variable *l* wird das Bild mit der horizontalen und vertikalen Auflösung *h* und *v* ausgegeben.

Umfang: 70 Zeilen

```
void LeseHSIDaten(char* Dateiname,
                  unsigned char **Hue,
                  unsigned char **Sat,
                  unsigned char **Int,
                  int *h,
                  int *v)
```

Die Prozedur *LeseHSIDaten* lädt eine Datei im ppm-Format. Die Variable *Dateiname* gibt den Namen der Datei an, die geladen werden soll. Durch die Variablen *Hue*, *Sat* und *Int* wird das HSI-Farbbild mit der horizontalen und vertikalen Auflösung *h* und *v* ausgegeben.

Umfang: 70 Zeilen

### A.1.12 LinseEntzerren.h

```
void LinseEntzerren(unsigned char **l,
                   unsigned char **r,
                   int *h,
                   int *v)
```

Die Prozedur *LinseEntzerren* behebt Linsenfehler. Als Ein- und Ausgabe-schnittstelle für das linke und rechte Kamerabild dienen die Variablen *l* und *r*. Die horizontale und vertikale Auflösung werden durch *h* und *v* gesetzt.

Umfang: 290 Zeilen

### A.1.13 MinimiereAusschnitt.h

```
void MinimiereAusschnitt(unsigned char **Bild,  
                        int *Hori,  
                        int *Verti,  
                        unsigned char **NeuBild,  
                        int **NeuH,  
                        int **NeuV)
```

Die Prozedur *MinimiereAusschnitt* beschränkt ein Bild, indem alle Ränder des Bildes abgeschnitten werden, die nur den Grauwert 0 beinhalten. Die Eingabe des Bildes mit der horizontalen und vertikalen Auflösung  $h$  und  $v$  erfolgt über die Variable *Bild*. Die Ausgabe des Ergebnisbildes mit der horizontalen und vertikalen Auflösung *NeuH* und *NeuV* erfolgt über die Variable *NeuBild*.

Umfang: 100 Zeilen

### A.1.14 ObjektDrehen.h

```
void ObjektDrehen(unsigned **Punkte,  
                 unsigned *Anzahl,  
                 unsigned *Hoehe,  
                 unsigned *Breite,  
                 unsigned *Winkel,  
                 unsigned **NeuePunkte,  
                 unsigned *NeueHoehe,  
                 unsigned *NeueBreite)
```

Die Prozedur *ObjektDrehen* dreht ein binäres Bild um einen gegebenen Winkel. Die Eingabe des Bildes erfolgt Punktweise. Jeder gesetzte Punkt ist sequentiell in der Variablen *Punkte* mit x- und y-Koordinate abgelegt und *Anzahl* gibt die Anzahl aller Punkte an. Die Variable *Winkel* enthält den Drehwinkel, um den die Punktmenge gedreht werden soll. Die Ausgabe erfolgt auch über die Angabe einer sequentiellen Koordinatenliste *NeuePunkte*. Die durch die Rotation geänderte horizontale und vertikale Auflösung wird durch *NeueBreite* und *NeueHoehe* ausgegeben.

Umfang: 70 Zeilen

```
void BildDrehen(unsigned char **Bild,  
               unsigned *Hoehe,
```

```

    unsigned *Breite,
    unsigned *Winkel,
    unsigned char **NeuesBild,
    unsigned *NeueHoehe,
    unsigned *NeueBreite)

```

Die Prozedur *BildDrehen* dreht ein Bild um einen gegebenen Winkel. Die Eingabe des Bildes mit der horizontalen und vertikalen Auflösung *Breite* und *Hoehe* erfolgt über die Variable *Bild*. Dieses Bild wird um den Winkel *Winkel* gedreht. Die Ausgabe erfolgt in ein Bild *NeuesBild* mit der horizontalen und vertikalen Auflösung *NeueBreite* und *NeueHoehe*.

Umfang: 180 Zeilen

### A.1.15 PicPoint.h

```

void PoiToPic(unsigned **Punkte,
              unsigned *Anzahl,
              unsigned char **Bild,
              unsigned *Hoehe,
              unsigned *Breite)

```

Die Prozedur *PoiToPic* wandelt ein durch Punkte definiertes binäres Bild in ein normales Bild um. Die Eingabe der sequentiellen Koordinatenpunkte, d.h. jeweils x- und y-Koordinate, mit der Anzahl *Anzahl* erfolgt über die Variable *Punkte*. Die Ausgabe liefert das Bild *Bild* mit der horizontalen und vertikalen Auflösung *Breite* und *Hoehe*.

Umfang: 20 Zeilen

```

void PicToPoi(unsigned char **Bild,
              unsigned *Hoehe,
              unsigned *Breite,
              unsigned **Punkte,
              unsigned *Anzahl)

```

Die Prozedur *PicToPoi* wandelt ein normales Bild in ein durch Punkte definiertes binäres Bild um. Punkte des normalen Bildes mit dem Grauwert 0 entsprechen dabei nicht gesetzten Punkten in der Punktdefinition. Die Eingabe der Bilddaten erfolgt über die Variable *Bild*. Die horizontale und vertikale Auflösung des Bildes ist *Breite* und *Hoehe*. Die Ausgabe der gesetzten Bildpunkte erfolgt sequentiell mit x- und y-Koordinate über die Variable *Punkte*

mit *Anzahl* Koordinatenpaaren.

Umfang: 30 Zeilen

### A.1.16 PositionBestimmen.h

```
void PositionBestimmen(unsigned char **k,  
                       int *h,  
                       int *v,  
                       float *scale,  
                       int *zEbeneOben)
```

Die Prozedur *PositionBestimmen* rechnet Disparitätswerte in eine Szenenaufsicht um. Die Eingabe des Disparitätsbildes mit der horizontalen und vertikalen Auflösung *h* und *v* erfolgt über die Variable *k*. Die Variable *scale* ist ein zuvor ermittelter Normierungsfaktor. Durch *zEbeneOben* wird die obere Grenze des Höhenwertes gesetzt. Dabei ist 255, d.h. 255 mm, die maximale Obergrenze. Die Ausgabe der Aufsicht *k* erfolgt mit der neu gesetzten horizontalen und vertikalen Auflösung *h* und *v*.

Umfang: 150 Zeilen

### A.1.17 TischAusblenden.h

```
void TischAusblenden(unsigned char **l,  
                     unsigned char **r,  
                     int *h,  
                     int *v,  
                     unsigned char **la,  
                     unsigned char **ra)
```

Die Prozedur *TischAusblenden* extrahiert die Objekte auf der Tischfläche. Andere Bereiche werden mit dem Grauwert 0 ausgeblendet. Die Eingabe des linken und rechten Kamerabildes mit der horizontalen und vertikalen Auflösung *h* und *v* erfolgt über die Variable *l* und *r*. Die Ausgabe erfolgt über *la* und *ra*.

Umfang: 180 Zeilen

### A.1.18 TischMaskieren.h

```
void TischMaskieren(int *h,
```

```
int *v,  
unsigned char **LinkesBild,  
unsigned char **RechtesBild)
```

Die Prozedur *TischMaskieren* errechnet die Bildbereiche der Tischfläche. Durch die Angabe der horizontalen und vertikalen Auflösung  $h$  und  $v$  der Kamerabilder liefert die Prozedur für jede Kamerablickrichtung eine Maske *LinkesBild* und *RechtesBild*, die Bereiche außerhalb der Tischfläche ausblendet.

Umfang: 260 Zeilen

## A.2 Kameras

Für eine Reproduzierbarkeit der Ergebnisse müssen die Kameradaten bekannt sein. Die erste Tabelle enthält die vom Hersteller gegebenen Grunddaten. In der zweiten Tabelle sind die Kameraparameter enthalten, die durch eine Kalibrierung (siehe [3]) ermittelt wurden.

Typ	Sony XC-999P (Pal)
Bildelemente	752 (H) x 582 (V)
Sensorgroße	6,4 x 4,8 mm
Minimale Beleuchtung	4,5 lux, F1,2 AGC: On
Sensitivität	2000 lux F5,6 AGC: Off (0 dB)
Video Ausgang	Y/C
Verschußzeit	1/50 s
Farbtemperatur	3200K (Innenraum)
Signal-Rausch Verhältnis	> 46 dB, AGC Off
Arbeitstemperatur	0°C bis 40°C
Arbeitsluftfeuchtigkeit	max. 80% (nicht kondensierend)
Linsen-Brennweite	6 mm

Kamera	rechts	links
Breite in Pixeln (Silizium)	752	752
Höhe in Pixeln (Silizium)	582	582
Breite in Pixeln (Bild)	700	700
Höhe in Pixeln (Bild)	512	512
Pixelbreite (Silizium)	0.0085106 mm	0.0085106 mm
Pixelhöhe (Silizium)	0.0082474 mm	0.0082474 mm
Pixelbreite (Bild)	0.0091429 mm	0.0091429 mm
Pixelhöhe (Bild)	0.0082474 mm	0.0082474 mm
Brennweite	6.0318443 mm	6.0373019 mm
Bildhauptpunkt x	347.64523	350.11509
Bildhauptpunkt y	274.67398	259.15769
radiale Verzerrung	0.0056723	0.0055524
Skalierung x	1.0256634	1.0229612
Fehlerbeurteilung der Kalibrierung		
Normalisierter Kalibrierungsfehler		
Mittelwert	1.16794 mm	1.07155 mm
Standardabweichung	0.779694 mm	0.621063 mm
Maximale Abweichung	7.54163 mm	4.42282 mm
Objektraum Kalibrierungsfehler		
Mittelwert	0.302013 mm	0.269844 mm
Standardabweichung	0.190145 mm	0.167753 mm
Maximale Abweichung	1.70749 mm	1.10732 mm

## A.3 Benutzte Rechner

### A.3.1 Rechner für Testläufe

Rechnername	streb
Model	Sun UltraSPARC 10 Model 300
CPU	UltraSPARC-IIi 300 MHz
physikalischer Speicher	256 MB
Virtueller Speicher	1,1 GB
Betriebssystem	SunOS Release 5.8

### A.3.2 Rechner Cora

Rechnername	max
CPU	Intel Celeron 500MHz
physikalischer Speicher	128 MB
Betriebssystem	QNX 4.2.5



# Literaturverzeichnis

- [1] Thomas Bergener, Carsten Bruckhoff, Percy Dahm, Herbert Janßen, Frank Joublin und Rainer Menzner, Arnold: An Anthropomorphic Autonomous Robot for Human Environments, SOAVE'97, Selbstorganisation von adaptivem Verhalten, 1997
- [2] Thomas Bergener, Carsten Bruckhoff, Percy Dahm, Herbert Janßen, Frank Joublin, Rainer Menzner, Axel Steinhage und Werner von See-len, Complex Behavior by means of Dynamical Systems for an Anthropomorphic Robot, Neural Networks - Special Issue, 1999
- [3] R. Tsai, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, IEEE Journal Robotics and Automation, 1987, S. 323-344
- [4] Christof Born und Bernd Völpel, Grouping bits to objects, Institut für Neuroinformatik, Ruhr-Universität Bochum, 1995
- [5] Michael Vetterling, Seminar Bildverarbeitung mit neuronalen Netzen WS 96/97, Adaptive Transform Coding, Johann Wolfgang Goethe Universität, <http://www.informatik.uni-frankfurt.de/~brause/SemWS96/transcodh.html>
- [6] Normand Grégoire und Mikael Bouillot, CS 507 Computational Geometry, Hausdorff distance between convex polygons, McGill University, Canada, 1998, <http://livia.etsmtl.ca/~normand/projets/hausdorff/>
- [7] Foley, van Dam, Feiner und Hughes, Computer Graphics - Principles And Practice, Second Edition, Addison Wesley Publishing Company 1990, S.203, S.591f
- [8] Rainer Steinbrecher, Bildverarbeitung in der Praxis, R. Oldenbourg Verlag GmbH, 1993, S. 108ff., S. 209ff.

- [9] Wolfgang Michael Theimer, VDI Fortschrittberichte, Vergenzsteuerung und Tiefenrekonstruktion, mit einem aktiven Stereokamerasystem, VDI Verlag, 1995, S. 30ff.
- [10] Claus-E. Liedtke und Manfred Ender, Wissensbasierte Bildverarbeitung, Springer Verlag, 1989, S. 50f.
- [11] Bernd Jähne, Digitale Bildverarbeitung, Springer Verlag, 1993, S. 23f.
- [12] Berthold Klaus Paul Horn, Robot Vision, MIT Press, 1986, S. 300f.
- [13] Milau Sonka, Vaclav Hlavac und Roger Boyle, Image Processing, Analysis and Machine Vision, Chapman & Hall Computing, 1993, S. 80f.
- [14] Robert F. Schmidt und Gerhard Thews, Physiologie des Menschen, 23. Auflage, Springer Verlag, 1986, S. 249ff.