Diplomarbeit

Griffplanung für eine visuell geführte Multifinger-Hand eines mobilen Service-Roboters

Alexander Stahlberg

Universität Hamburg - Fachbereich Informatik

Arbeitsbereich Technische Aspekte Multimodaler Systeme

März 2006

Betreut von:

Prof. Dr. Jianwei Zhang
Prof. Dr. Bernd Neumann

Zusammenfassung

Die Planung von Greifaufgaben in der Robotik ist eine komplexe Aufgabe. Das Finden von stabilen Griffen spielt dabei eine wichtige Rolle am Ende einer Kette von Teilaufgaben. Die zu greifenden Objekte müssen zunächst lokalisiert und erkannt werden.

Das Planen wird deshalb in dieser Arbeit in drei Phasen unterteilt, welche aufeinander aufbauen. Sie geben der Planung von Griffen eine Rahmenstruktur, die unabhängig von den im Einzelnen verwendeten Verfahren ist. Es wird gezeigt, welche Methoden für die notwendigen Teilaspekte Anwendung finden können. Ein Ansatz wird dargelegt, der es ermöglicht, mit Hilfe einer Handkamera als Sensor die Lokalisierung und Erkennung von Objekten derart zu gestalten, dass die Ergebnisse für eine Griffplanung von beliebigen 3D-Objekten umgesetzt werden können.

Es wird sowohl darauf eingegangen, wie bekannte Objekte mittels statistischer Methoden anhand der Sensordaten klassifiziert werden können, als auch der Frage nachgegangen, inwieweit eine Planung von stabilen Griffen für unbekannte Objekte möglich ist.

Anhand einer Implementierung für die Planung von Griffen mit dem Service Roboter des Arbeitsbereichs TAMS wird gezeigt, inwieweit der vorgestellte Ansatz fähig ist, robuste Griffe zu finden und wo die Grenzen liegen. Dabei wird im Speziellen auf die Möglichkeiten der BarrettHand eingegangen.

An dieser Stelle möchte ich mich bei all jenen bedanken, die mich im Laufe dieser Arbeit mit wertvollen Ratschlägen unterstützt haben.

Ich danke Tim Baier, der immer ein offenes Ohr hatte und scheinbar nie müde wurde, meine ständigen Fragen zu beantworten oder Fehler in der Implementierung auszumerzen, sowie Prof. Zhang und Prof. Neumann für die konstruktive Kritik während der Ausarbeitung. Ich bedanke mich auch bei den anderen Diplomanten im AB TAMS, die die langen Stunden im Labor unterhaltsamer machten und mit denen so manche Idee diskutiert werden konnte.

Nicht zuletzt möchte ich meinen Eltern danken, die mich während all der Jahre meines Studiums unterstützt haben.

Inhaltsverzeichnis

1 Einleitung **11**

1.1 Problemstellung 12
1.1.1 Drei Phasen der Griffplanung 13
1.2 Forschungsstand 13
1.3 Hardware 15
1.3.1 Roboterplattform 15
1.3.2 Roboterarm PA10-6C 16
1.3.3 BarrettHand BH8-262 17
1.3.4 Handkamera CV-M2250 17

2 Grundlagen **23**

2.1 Sehen 23
2.1.1 Segmentierung 23
2.1.2 Rückprojektion 28
2.1.3 „visual servoing“ 33
2.2 Erkennen 34
2.2.1 Objektrepräsentation und Oberflächenanalyse 35
2.2.2 Principal Component Analysis 38
2.2.3 Klassifizierung unbekannter Objekte 43
2.3 Greifen 46
2.3.1 Optimale Griffe mittels „form-closure grasp“ Analyse 46
2.3.2 Suchen von „form-closure“ Griffen 51
2.3.3 Ermitteln von Kontaktpunkten 54

3 Implementierung **55**

3.1 Architektur 55
3.2 Versuchsaufbau 55
3.3 Programmablauf 56
3.4 Sehen 56
3.4.1 Bildaufnahme 56
3.4.2 Segmentierung 58
3.4.3 Objektlokalisierung 60
3.4.4 „visual servoing“ 63
3.5 Erkennen 64
3.5.1 Objektklassifizierung 64
3.5.2 Objektrepräsentation 67
3.6 Greifen 69
3.6.1 Griffbestimmung 69
3.6.2 Griffanalyse 70

3.6.3 Ausführen des Griffes 73

4 Ergebnisse & Erläuterungen **75**
4.1 Sehen 75
4.2 Erkennen 76
4.3 Greifen 78

5 Ausblick **83**

A Anhang **85**
A.1 Experimentelle Bestimmung des Öffnungswinkels einer CCD Kamera 85
A.2 Griffkonfigurationen 85

Notation

a Skalar

γ Winkel

$\vec{n}$ Vektor

$\underline{x}$ Basisvektor

$\widehat{T}$ Transformationsmatrix

$\breve{C}$ Menge

Anmerkung zur englischen Sprache

Die für diese Arbeit verwendete Literatur lag hauptsächlich in englischer Sprache vor. Die dort verwendeten englischen Fachausdrücke wurden in den meisten Fällen übersetzt. Lag allerdings kein gängiger deutscher Begriff vor, so wurde der englische verwendet, um Mißverständnissen vorzubeugen.

1 Einleitung

Für den Menschen ist es ein Leichtes, beliebige Objekte zu greifen. Er vereint dabei Steuerung und Regelung des Greifvorgangs in seinen Handbewegungen. Etwaige Ungenauigkeiten werden unmittelbar durch visuelles Feedback kompensiert [20]. Die Objektidentifikation läuft meist unterbewusst ab, aber selbst unidentifizierte Objekte kann der Mensch stabil greifen. Diese basalen Fähigkeiten eignet sich der Mensch bereits im frühen Kindesalter an [31]. Will man diesen Mechanismus einen Roboter lehren, stellt er sich als komplexe Problemstellung dar. Die Lokalisation und Klassifikation der zu greifenden Objekte muss nahtlos zusammenarbeiten, um eine robuste Griffplanung zu ermöglichen.

Viele Arbeiten haben sich mit den Aspekten dieser Thematik befasst, so wird in [1] ein Verfahren vorgestellt, welches es ermöglicht, Objekte nur anhand ihrer Silhouetten zu identifizieren, [10] und [9] präsentieren eine biologisch motivierte Griffplanung, welche anhand der Form eines Objekte optimale Greifpunkte für Zwei-Finger-Greifer finden kann. Die Kombination dieser Verfahren würde eine Griffplanung für erwähnte Zwei-Finger-Greifer ermöglichen, berücksichtigt aber nicht, dass reale Objekte eine dreidimensionale Form besitzen. Ein Ansatz dazu findet sich in [13]. Y. Liu und M. Lam stellen dort ein Verfahren vor, welches die Berechnung von Kontaktpunkten für generische Roboterhände ermöglicht. Dieses berücksichtigt allerdings nicht die Handphysik, welche bei realen Planungsaufgaben eine entscheidene Rolle spielt, da sie die möglichen Griffkonfigurationen einschränkt.

Diese Arbeit liefert einen Ansatz für den gesamten Vorgang des Greifens. Dies beinhaltet:

- die Identifikation, beziehungsweise Klassifikation, von Objekten
- die Erstellung von 3D-Repräsentationen der zu greifenden Objekte
- das Berechnen von Greifpunkten
- das Ausführen der berechneten Griffe auf der TAMS Roboterplattform (s. Kapitel 1.3)

Die dabei behandelten Fragen und Probleme werden in Kapitel 1.1 vorgestellt, worauf ein kurzer Überblick über den Stand der Forschung auf dem Gebiet der Griffplanung folgt.

In Kapitel 2 werden die verwendeten Verfahren für Objektklassifikation, Griffberechnung und Kinematik beschrieben. In Kapitel 3 wird im Einzelnen auf die nötigen

Anpassungen der verwendeten Verfahren an die Roboterplattform, speziell die BarrettHand, eingegangen.

Schließlich folgt in Kapitel 4 eine Präsentation der Ergebnisse der vorgestellten Griffplanung und es wird dargestellt, inwieweit der vorgestellte Ansatz zur Findung robuster Griffe geeignet ist.

1.1 Problemstellung

Interaktion mit der Umwelt ist eine der spannensten Aufgabenstellungen der autonomen Robotik. Feinmotorische Aufgaben, wie das Befüllen eines Glases, es zu greifen und anschließend zu servieren, ermöglichen eine vom Menschen als natürlich aufgefasste Interaktion mit künstlichen Intelligenzen.

Der Ansatz, diese Fähigkeiten allein auf optischen Sensoren wie CCD-Kameras zu basieren, versucht mit minimalem technischen Aufwand an diese Aufgabe heranzugehen. Üblicherweise verlassen sich moderne Robotersysteme auf Sensoren wie Laserscanner und taktile Sensoren, um die Umwelt zu erfassen und auszuwerten.

Der alleinige Einsatz von Kameras schließt zwar viele bekannte Verfahren zur Umwelterkennung aus, allerdings sind CCD-Kameras kostengünstig in der Anschaffung und lassen sich mit üblichen Computer-Systemen problemlos ansteuern. Außerdem sind sie an Manipulatorendpunkten wie Roboter-Händen eine der platzsparendsten Möglichkeiten.

Optische Sensoren bieten zudem den Vorteil, alle Dimensionen der Umwelt aus nur einer Sensoraufnahme extrahieren zu können. Bei der Projektion der dreidimensionalen Umwelt in ein digitales Bild geht zwar eine Raumdimension verloren, diese lässt sich aber, wenn nötig, mit entsprechenden Verfahren approximieren.

Mit Laserscannern ist es möglich, Entfernungen verlustfreier zu erfassen, als mit optischen Sensoren wie CCD-Kameras. Allerdings haben Laserscanner zwei entscheidende Nachteile:

- Sie sind, wie bereits zuvor erwähnt, baulich bedingt recht groß und können so schwerlich am Ende eines humanoid-großen Roboterarms angebracht werden.
- Laserscanner erfassen stets nur eine Ebene des Raums. Um sich ein mehrdimensionales Bild zu schaffen, muss mit mehreren Laserscannern gearbeitet oder der Laserscanner mit einer Einheit versehen werden, die den Scanner horizontal oder vertikal schwenkt.

In dieser Diplomarbeit wird nun der Frage nachgegangen, inwieweit eine einzelne CCD-Kamera als Sensor ausreichend ist, um eine Planung robuster Griffe zu ermöglichen. Es wird gezeigt werden, dass exakte Lokalisierung möglich ist, wie Handkameras eingesetzt werden können, um robuste Objektidentifikation zu gewährleisten und wie auf dieser Basis eine Planung von stabilen Griffen stattfinden kann.

1.1.1 Drei Phasen der Griffplanung

Das Greifen eines Objekts lässt sich, von der Objektlokalisierung bis zum Schließen der Finger um das Objekt an der optimalen Position, in drei Phasen unterteilen:

Die in einer Szene präsentierten Objekte müssen gesehen und erkannt werden, bevor sie gegriffen werden können. Dazu ist es nötig, die Position der Objekte in Weltkoordinaten zu bestimmen und diese anschließend zu klassifizieren, wodurch interne Repräsentationen der Objekte gebildet werden können. Anhand dieser Repräsentationen kann die Planung eines Griffes erfolgen.

Ziel der Unterteilung in drei Phasen ist es, eine strikte Trennung der einzelnen Aufgaben der Griffplanung zu schaffen. Dadurch soll gewährleistet werden, dass die in den jeweiligen Phasen verwendeten Methoden ausgetauscht werden können, ohne eine Änderung in anderen Phasen zu bedingen.

1.2 Forschungsstand

Für die Phasen „Sehen“ und „Erkennen“ ist es naheliegend, hinreichend bekannte Methoden der Bildverarbeitung zu verwenden. Kameras unterscheiden sich wenig, so dass dieses Feld sehr homogen ist und Methoden leicht allgemeingültig gemacht werden können. Diese Methoden sind seit Jahren in Gebrauch und ihre Robustheit wurde mehrfach gezeigt. Welche in dieser Arbeit verwendet wurden, wird in Kapitel 2 ausgeführt.

Roboterhände unterscheiden sich hingegen sehr voneinander. Das offensichtlichste Merkmal ist die Anzahl der Finger, aber auch die Freiheitsgrade der Finger und somit der gesamten Hand variieren stark. Viele Methoden, die in der aktuellen Forschung erwähnt werden, sind deshalb an bestimmte Hände adaptiert, oder so allgemein gehalten, dass eine der aufwendigsten Aufgaben ist, diese an die Kinematik der verwendeten Hand anzupassen.

Unter „Griff“ wird eine Menge von Punkten verstanden, welche den Kontakt zwischen Objektoberfläche und Manipulator (Roboterhand) beschreiben.Um bestimmen zu können, ob so ein Griff geeignet ist, das zu greifende Objekt stabil zu halten, wird zum einen ein Kriterium benötigt, welches entscheiden kann, ob ein Griff stabil ist und zum anderen ein Verfahren, welches die Handkonfiguration bestimmen kann, die zu den gewünschten Kontaktpunkten führt.

Y. Liu et al. präsentieren in [13] einen Algorithmus, der es ermöglicht, optimale Kontaktpunkte für 3D-Objekte zu finden. Das Qualitätsmerkmal für einen stabilen Griff ist hier die Eigenschaft „form-closure“. Die Suche beginnt mit sieben zufällig

gewählten Kontaktpunkten, da dies ein hinreichendes Kriterium für „form-closure“ ist, und verschiebt diese entlang der Oberfläche des Objekts, bis ein geeigneter Griff gefunden wurde. Dabei werden einige Eigenschaften von „form-closure“-Griffen ausgenutzt, um den Suchraum mehrfach zu teilen und eine Heuristik zu schaffen, welche die Punktauswahl in jedem Schritt auf das gewünschte Ziel hin optimiert. Es wird allerdings keine Lösung dargeboten, wie die Kinematik einer verwendeten Hand bei der Auswahl von Oberflächenpunkten berücksichtig werden könnte. Im Besonderen die in vorliegender Arbeit verwendete BarrettHand wird in den Freiheitsgraden stark eingeschränkt, sobald sieben Kontaktpunkte gefordert sind.

Werden die möglichen Griffkonfigurationen eingeschränkt, so lassen sich auch einfachere Qualitätsmerkmale für Griffe finden. Ein in [15] vorgestelltes Verfahren zur Griffplanung zeigt, wie auf Grundlage visueller Eigenschaften von Objekten Griffe für die BarrettHand geplant werden können. Dabei wird das Modell der BarrettHand vereinfacht und es werden nur „precision grasps“ betrachtet, also Griffe, bei denen nur die Fingerspitzen in Kontakt mit dem Objekt treten. Alle Objekte werden von oben gegriffen, was bei den zuvor genannten Bedingungen zu genau drei Kontaktpunkten führt. Anhand des von diesen Punkten aufgespannten Dreiecks findet die Griffoptimierung statt.

Die Grundidee ist, dass der Griff dann besonders stabil ist, wenn dieses Dreieck gleichseitig ist, bzw. einem gleichseitigen Dreieck besonders nahe kommt. Der Schnittpunkt der Winkelhalbierenden des Dreiecks und der Masseschwerpunkt des Objektes sollen zudem möglichst nahe beeinander liegen. Die in den Experimenten verwendeten Objekte waren speziell für diese Aufgabe konstruiert und unterschieden sich in ihrem Gewicht und ihrer Oberfläche. Sie besaßen alle dieselbe Höhe und waren „oben“ und „unten“ eben, hatten aber unterschiedliche Silhouetten. Die Ergebnisse der Experimente waren überzeugend, allerdings lässt sich dieses Verfahren nicht direkt auf „echte“ 3D-Objekte übertragen.

Eine Methode, die in abgewandelter Form auch für die in dieser Arbeit vorgestellte Simulation verwendet wurde, ist die Überprüfung auf Schnittpunkten zwischen Roboterhand und Objekt. Dies ermöglicht die Beurteilung, ob ein Griff für eine spezielle Hand durchführbar ist, da sowohl Kontaktpunkte, als auch Handkonfiguration berücksichtigt werden können. In [6] wird diskutiert, wie dies für Mehr-Finger-Hände und 3D-Objekte möglich ist. Es wird eine Verfahren vorgestellt, welches es ermöglicht, für eine gegebene Handkinematik und Objekte zu überprüfen, ob eine beliebige Griffkonfiguration erreichbar ist. Objekte und Hand werden dabei als eine Menge aus Punkten, Kanten und Ebenen aufgefasst, welche sich bei bestimmten Konfigurationen schneiden. Zusätzlich werden für die Hand die möglichen Gelenkwinkel betrachtet. Eine typische Hand besteht aus einigen Fingerspitzen und mindestens einem Gelenk für jede Fingerspitze und die dazugehörigen Glieder. Wird nun eine Menge an Kontaktpunkten gewählt, wird berechnet, welche Winkel die jeweiligen Gelenke einnehmen müssen, um diese Position zu erreichen. Treten dabei keine Schnittpunkte oder -geraden auf, welche innerhalb der jeweiligen Objekte liegen und wurde kein Gelenk über den zulässigen Bereich belastet, so ist der Griff anwendbar. Diese Methode ist für beliebige Hände und Objekte anwendbar, da sie keinerlei Vorbedingungen an die Beschaffenheit der Hand und der Objekte stellt.

Die Planung von Griffen für Zwei-Finger-Greifer ist einfacher, wenn auch bei weitem nicht so flexibel wie bei Multifingerhänden, da der Handkinematik kaum Beachtung geschenkt werden muss. Das Finden von Kontaktpunkten beschränkt sich dabei auf zwei beliebige Punkte, welche für die Hand erreichbar sind und das Objekt in einer stabilen Lage halten können. A. Hauck et al. stellen in [9] eine biologisch motivierte Greifstrategie für Zwei-Finger-Greifer vor. Die Objekte werden mit einem Stereo-Kamera-Aufbau lokalisiert und entsprechend einer soliden Strategie für Zwei-Finger-Greifer Greifpunkte berechnet: 3D-Objekte werden anhand ihres Umrisses bewertet, die Greifpunkte sollen dabei an Stellen liegen, die das Objekt in eine Gleichgewichtslage bringen. Sollte dies nicht möglich sein, werden die Punkte so gewählt, dass der Schwerpunkt unterhalb der Kontaktstellen liegt. Beim Anfahren der Greifpunkte wird mittels visuellem Feedback überprüft, ob die Lage korrekt erreicht wird und bei Abweichung korrigiert. Dabei wird ein Ansatz verfolgt, welcher der Greifstrategie des Menschen ähnlich kommt. Der Greifer nähert sich der Endposition in einem Bogen. Dies gewährleistet „flüssige“ Korrektur, da zunächst nur wenige Gelenke verwendet werden müssen. Korrekturen können so meist in geringen Winkelkorrekturen der übrigen Gelenke erfolgen. Der in [9] vorgestellte Ansatz für die Wahl der Greifpunkte lässt sich nicht direkt auf Hände mit mehr Fingern, die dreidimensionale Objekte greifen sollen, übertragen, allerdings kann die grundsätzliche Idee Anwendung finden. Objekte so zu greifen, dass sie sich im Gleichgewicht befinden, schränkt die Vorauswahl bei Griffen ein und reduziert somit die Komplexität.

In dieser Arbeit wurde „form-closure“ als Qualitätsmerkmal für Griffe verwendet. Ob diese Eigenschaft für einen bestimmten Griff erfüllt ist, wurde durch einen von Y. Liu et al. in [12] vorgestellten Algorithmus überprüft. Es wurden einige erfolgsversprechende Handkonfigurationen ausgewählt und mittels Simulation an internen Repräsentationen der zu greifenden Objekte getestet. Sollte ein solcher Griff die Eigenschaft „form-closure“ erfüllen, wird er als geeignet angesehen und ausgeführt.

1.3 Hardware

Dem Arbeitsbereich *Technische Aspekte Multimodaler Systeme* der Universität Hamburg steht eine mobile Roboterplattform zur Verfügung, welche mit diversen Sensoren und Manipulatoren bestückt ist.

Der Arm samt Hand, sowie eine an dieser befestigte Kamera, wurden im Laufe dieser Arbeit verwendet.

Im Folgenden sollen die Komponenten dieses Service-Roboters im Detail vorgestellt werden.

1.3.1 Roboterplattform

Bei der Roboterplattform des Arbeitsbereichs *TAMS* handelt es sich um einen mittels zweier Differentialmotoren angetriebenen Unterbau neobotix MP-L655, auf welchem eine Basis für die Montage der Manipulatoren und Sensoren angebracht ist.

Auf Höhe der Antriebsräder befinden sich zwei Laserscanner der Firma SICK, die auf der Längsachse vorne und hinten angebracht wurden. Dadurch kann ein Bereich von nahezu 180° vor und hinter dem Roboter abgetastet werden. Die Scanradien sind überlappungsfrei und links und rechts des Roboters befindet sich jeweils ein kleiner Bereich, welcher nicht erfasst werden kann.

An der Manipulatorbasis montiert sind ein Roboterarm der Firma Mitsubishi Heavy Industries, ein nach vorne gerichteter Stereokamera-Aufbau auf einer Pan-Tilt-Unit, sowie ein omnidirektionaler Kameraaufbau, welcher Rundumsichtaufnahmen der Roboterumgebung ermöglicht.

Am Ende des Roboterarms ist eine Roboterhand der Firma Barrett Technologies und eine Kamera, die auf den Greifbereich fokussiert, angebracht.

Abbildung 1.1: Roboterplattform des Arbeitsbereichs *TAMS*

1.3.2 Roboterarm PA10-6C

Für den Service-Roboter wird ein Roboterarm PA10-6C der Mitsubishi Heavy Industries, Ltd. verwendet. Er ist links der Fahrtrichtung montiert und es wurden Vorkehrungen getroffen, die es ermöglichen, den Roboter mit einem zweiten Arm auf der rechten Seite zu bestücken.

Der Arm besitzt sechs Freiheitsgrade (s. Abbildung 1.2), gewährleistet durch jeweils drei Rotations- (R) und Schwenkgelenke (S). Das Roboterkoordinatensystem ist rechtshändig, wobei die x-Achse in Fahrtrichtung und die z-Achse nach oben zeigt. Rotationsgelenke erlauben Rotation um die x-Achse, wohingegen Schwenkgelenke eine Rotation um die y-Achse ermöglichen. Rotation um die z-Achse ist durch eine Kombination dieser Gelenke möglich. Montiert sind diese, mit der Schulter beginnend, in der Reihenfolge: R-S-S-R-S-R.

Der in den Abbildungen 1.3 und 1.4 dargestellte maximale Arbeitsraum des PA10-6C wird auf der Roboterplattform durch den Roboterkörper eingeschränkt.

Zudem verhindert die Hand, und vor allem die an dieser montierte Kamera, ein volles Ausspiel des vorletzten Gelenks, da der Arm Gelenkkonfigurationen zulässt, die zu Kollisionen der beschriebenen Bauteile mit dem Arm selbst führen.

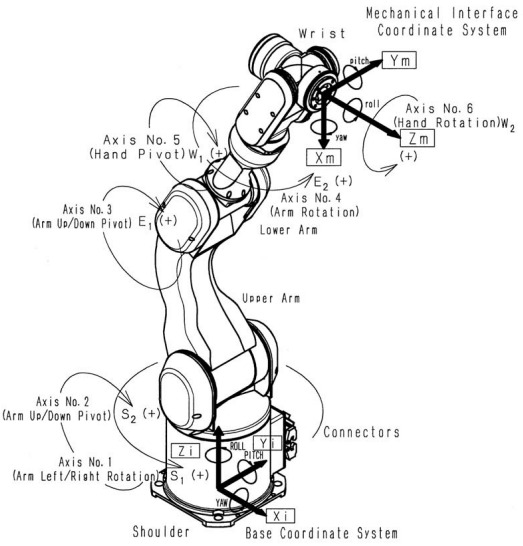


Abbildung 1.2: PA10-6C Freiheitsgrade

1.3.3 BarrettHand BH8-262

Am Endpunkt des Roboterarms ist eine Roboterhand des Typs BH8-262 montiert, eine Drei-Finger-Hand der Firma Barrett Technologies.

Jeder Finger besteht aus zwei Gliedern. Ein Finger dieser Hand, als „Daumen“ bezeichnet, ist starr montiert und lässt sich lediglich öffnen und schließen. Die übrigen zwei Finger sind rotationssymetrisch zum Ballen befestigt und lassen sich jeweils um 180° um diesen rotieren, wobei der Öffnungswinkel beider Finger zum Ballen stets gleich ist.

Eine Besonderheit dieser Roboterhand ist der sogenannte TorqueSwitch, ein Patent von Barrett Technologies. Dieser bewirkt, dass sich das letzte äussere Glied eines Fingers automatisch schliesst, sobald am ersten Glied eine definierte Kraft anliegt, welche dieses am weiteren Schließen hindert.

1.3.4 Handkamera CV-M2250

An der BarrettHand der TAMS Roboterplattform ist eine CV-M2250 Micro Head Color Camera der JAI CV-M2000 Series mit halbzölligem CCD befestigt und fokussiert in den Bereich ca. 20 cm vor dem Objektiv. Sie schliesst mit der Längsachse des Arms einen Winkel $\gamma >> 0$ ein und gewährleistet somit, dass der von der Hand abdeckbare Bereich im Bild zu sehen ist.

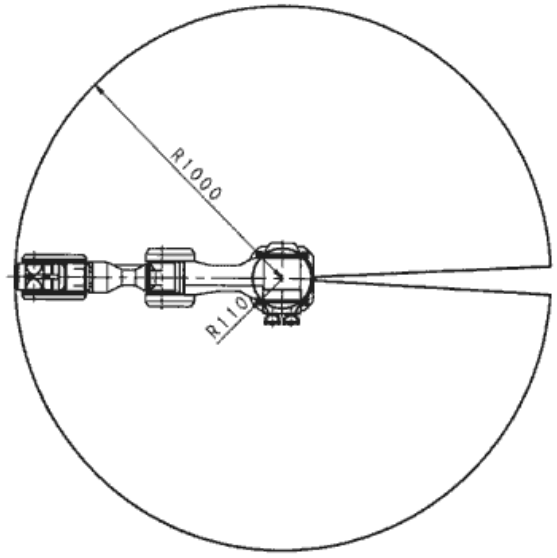


Abbildung 1.3: PA10-6C Arbeitsbereich - Draufsicht (Einheit: mm)

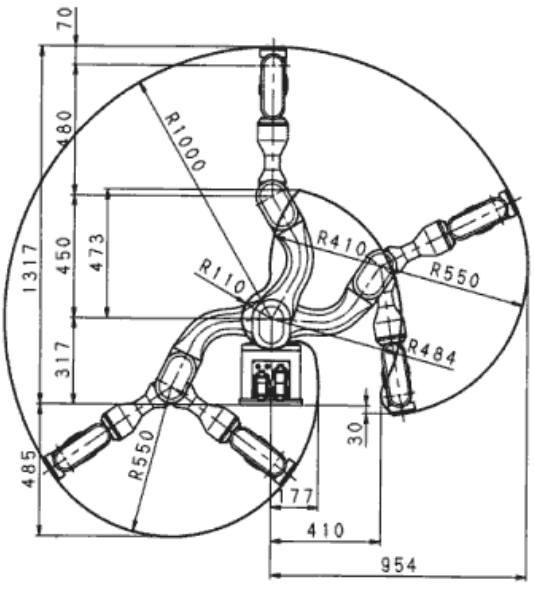


Abbildung 1.4: PA10-6C Arbeitsbereich - Seitenansicht (Einheit: mm)

Main Body		
Model		6 degrees of freedom
Mass of main body		38kg
Load capacity		10kg
Joint operating range	S1(rotation)	+−177 deg
	S2(swing)	−64 to +124 deg
	E1(swing)	−107 to +158 deg
	E2 (rotation)	+−255 deg
	W1(swing)	+−165 deg
	W2(rotation)	+−255 deg
Max.Operating speed	S1,S2 axis	57 deg/s
	E1axis	114 deg/s
	E2,W1,W2 axis	360 deg/s
Cleanliness		Class 1(0.2micrometer)
Configuration		Dust−proof/drop−proof construction(IP54)
Ambient environment		Temperature 0 to 50° C, Humidity under 90%RH(no condensation)
Drive method, Brake		AC servo motor, Electromagnetic brake
Position Sensor		Absolute angle sensor(resolver)
Positional repeatability		+−0.1mm
Max.lengthbetween joint		930mm
Air tubes for the user		2(Outside diameter 4mm,inside diameter.5mm)
Signal lines for the user		6(0.3sq)、3(0.75sq)

Abbildung 1.5: PA10-6C Spezifikationen

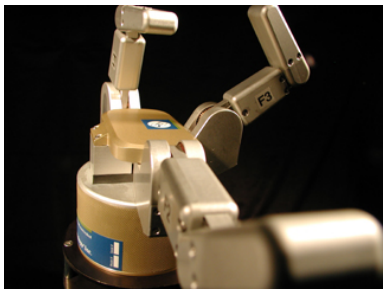


Abbildung 1.6: BarrettHand BH8-262 mit einem Spread von 90°

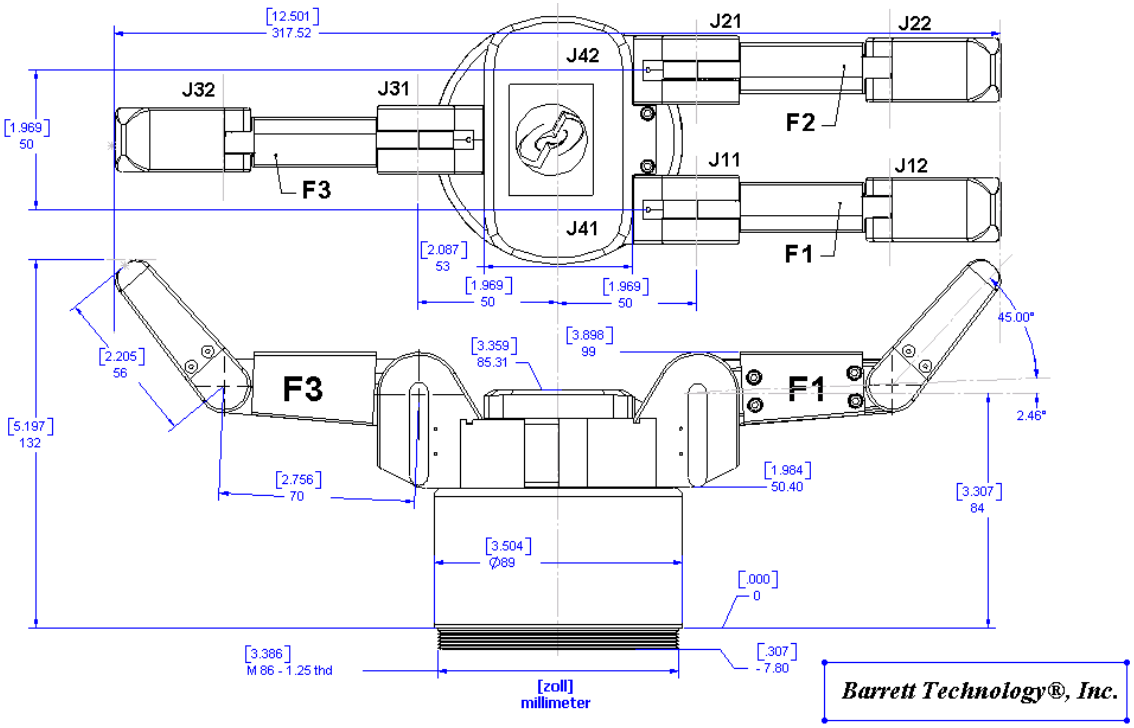


Abbildung 1.7: BarrettHand BH8-262 - Dimensionen

Specifications	PAL	NTSC
Scanning system Interlaced	625 lines 25 frames/sec. 2:1 Interlaced	525 lines, 30 frames/sec. 2:1 Interlaced
Scanning frequency (h) (v)	15.625 kHz 50 Hz	15.734 kHz 59.94 Hz
CCD sensor	Color 1/2" or 1/3" Hyper HAD IT CCD	
Effective pixels	752 (h) x 582 (v)	768 (h) x 494 (v)
Sensing area 1/2" CV-M2200/2250 1/3" CV-M2300/2350	 6.5 (h) x 4.8 (v) mm 4.9 (h) x 3.7 (v) mm	
Cell size 1/2" CV-M2200/2250 1/3" CV-M2300/2350	 8.6 (h) x 8.3 (v) µm 6.5 (h) x 6.25 (v) µm	 8.4 (h) x 9.8 (v) µm 6.35 (h) x 7.4 (v) µm
Resolution (horizontal)	>450 TV lines	
Sensitivity on sensor 1/2" CV-M2200/2250 1/3" CV-M2300/2350	 0.25 Lux, Max gain, 50% video 0.34 Lux, Max gain, 50% video	
S/N ratio	>48 dB (Min. gain, Gamma 1)	

Abbildung 1.8: JAI CV-M2250 Spezifikationen (Auszug)

Die Kamera ist über eine Framegrabber-Karte mit dem in den Roboter integrierten PC verbunden und liefert maximal PAL-Auflösung (752×582 Pixel).

Der Fokus ist mechanisch einstellbar, erfordert aber eine Demontage der Kamera, da die Halterung die dafür nötige Vorrichtung abdeckt.

2 Grundlagen

Die Teilaspekte der vorgestellten drei Phasen lassen sich auf das Lösen bekannter Probleme zurückführen. Aufgaben wie die Segmentierung von digitalen Bildern kommen in vielen Bereichen der Bildverarbeitung immer wieder vor. Die dazu verwendeten Methoden sind hinlänglich bekannt und ihre Zuverlässigkeit wurde mehrfach bewiesen.

Nachfolgend sollen die Verfahren vorgestellt werden, welche in den drei Phasen Anwendung fanden.

2.1 Sehen

Unter dem Begriff „Sehen“ soll das Betrachten einer Szene ohne semantische Interpretation verstanden werden. In dieser Phase wird die Anzahl der Objekte und deren Positionen bestimmt, ohne auf deren Bedeutung einzugehen. Dabei werden auch Bildstörungen zunächst als Objekt angesehen. Die Unterscheidung in interessante und uninteressante Objekte ist Aufgabe des „Erkennens“. In einem Schritt zwischen diesen Phasen können einige Objekte aufgrund sehr eindeutiger Merkmale allerdings häufig bereits als uninteressant markiert werden.

Das „Auge“ eines intelligenten Systems ist meist eine CCD Kamera. Die nachfolgend erläuterten Methoden arbeiten auf diskreten Bildern einer solchen Kamera. Anhand dieser Bilder soll erkannt werden, ob sich überhaupt Objekte in der betrachteten Szene finden lassen [1] und an welcher Position der Welt, bzw. relativ zum Greifer, sich diese befinden.

Dies sind zwei Schritte. Im ersten Schritt müssen die Objekte im Bild vom Hintergrund separiert werden, im zweiten wird die Position für jedes dieser Objekte bestimmt.

2.1.1 Segmentierung

Segmentierung von diskreten Bildern kommt in nahezu allen Bereichen der Bildverarbeitung vor. Es ist ein notwendiger Schritt zur Interpretation einer Szene.

Die Anforderungen an die Segmentierung in dieser Arbeit ist, dass Objekte vom Hintergrund freigestellt werden können, um ihren Masseschwerpunkt zu berechnen.

[1] Im folgenden wird davon ausgegangen, dass sich in den betrachteten Bildern zumindest ein interessantes Objekt befindet.

Dazu ist harte Segmentierung geeignet, die auf Binärbildern arbeitet. Dies spiegelt die Grundsätzlichkeit dieser Phase wider, da es zunächst vielmehr interessiert, was innerhalb des Pixelrasters Teil eines Objekts ist, als wie sich das Objekt darstellt.

Pixelorientierte Segmentierungsverfahren versuchen, die in einer aufgenommenen Szene enthaltenen Objekte zu separieren. Geläufig ist dafür auch der zuvor genannte Ausdruck: freistellen. Ziel ist es, die Objekte vom Hintergrund und von anderen Objekten zu separieren, wobei alle dem Objekt zugehörigen Pixel seiner Klasse zugeordnet werden. Hierbei wird von Labeling gesprochen.

In Szenen mit mehreren Objekten kommt es häufig zu partiellen Verdeckungen, welche die an die Segmentierung anschließende Klassifikation erschwert. Gleason et al. stellen dazu in [5] eine Möglichkeit vor, Objekte anhand ihrer Form in anderen Szenen wieder zu erkennen, auch wenn diese partiell verdeckt sind. Dies setzt natürlich voraus, dass das Objekt zu einem Zeitpunkt bereits unverdeckt zu sehen oder im Vorfeld bekannt war. Dies ist nicht immer gegeben.

Ziel dieser Arbeit ist es allerdings, die grundsätzliche Möglichkeit einer Griffplanung auf Grundlage von Handkameras zu zeigen, weshalb solche Probleme der Segmentierung durch entsprechend präparierte Szenen ausgeschlossen wurden. Es kann deshalb ein einfaches Labeling-Verfahren angewendet werden.

Zunächst wird die Aufnahme der Szene in ein Binärbild umgewandelt. Dazu wird auf das Bild ein Schwellwert angewendet, welcher es in Hintergrund- und Objektpixel unterteilen soll. Die Bestimmung des Schwellwertes ist ausschlaggebend für die zu erhaltenen Informationen aus einem Binärbild und muss den Anforderungen entsprechend angepasst werden. Ist er zu hoch, werden aus dem Binärbild möglicherweise zu viele Informationen eliminiert, ist er zu niedrig, kann das Resultat unbrauchbar für die weitere Analyse werden, da sich die einzelnen Bildelemente zu einem einzigen großen, unförmigen Objekt zusammenschließen.

Um dieses Problem zu behandeln, wurde der „optimale Schwellwert“ für das Bild bestimmt und angewendet. Dieser wird durch die Verteilung von Hintergrund- und Objektintensitäten bestimmt. Ist deren Verteilung für einen bestimmten Schwellwert identisch, so wird dieser Schwellwert „optimal“ genannt. Die Verwendung dieses Schwellwerts minimiert die Wahrscheinlichkeit für Segmentierungsfehler [25]. Ein Algorithmus, der den optimalen Schwellwert eines Bildes bestimmt, ist in Abbildung 2.2 zu finden.

Auf das erhaltene Binärbild wird ein Labeling-Verfahren nach [25] angewendet, welches die einzelnen Pixelcluster separiert. Hintergrund-Pixel werden einer gesonderten Klasse zugeordnet, Objekt-Pixel Objekt-Klassen (s. Abbildung 2.5). Zu welcher Klasse ein Objekt-Pixel zugehörig ist, wird durch dessen Umgebung bestimmt. Es wird von einem Zweierverbund ausgegangen, das heißt, befindet sich neben oder über dem Pixel ein Objekt-Pixel, so wird er der Klasse des Nachbarpixels zugeordnet; befindet sich links und über dem Pixel Hintergrund, so ist dieser Pixel einer neuen Klasse zugehörig.

Ein Sonderfall liegt dann vor, wenn sich links und über dem Pixel unterschiedliche Klassen befinden (s. Abbildung 2.4), was bedeutet, dass die Klasse des linken Pixels zur Klasse des oberen Pixels gehört, zuvor allerdings falsch markiert wurde. Es

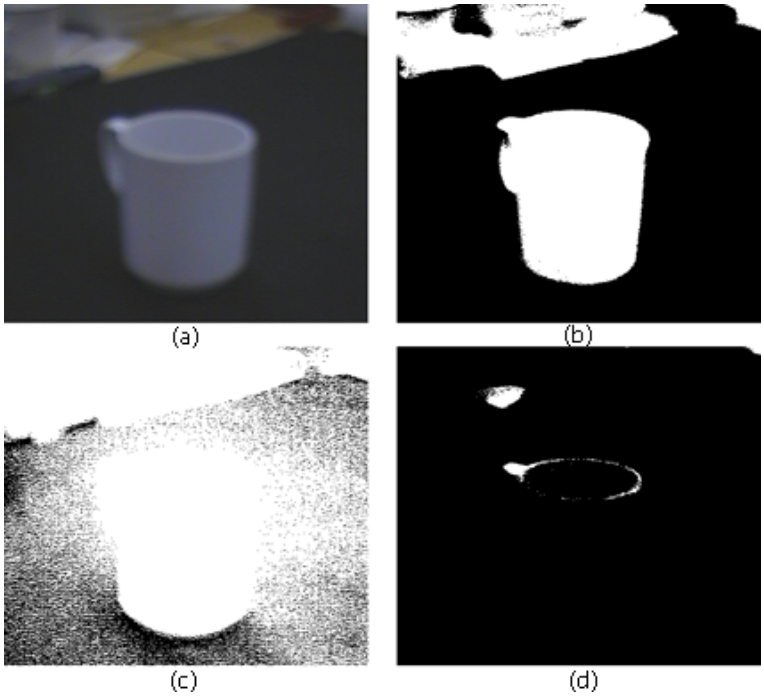


Abbildung 2.1: Schwellwerte: (a) Originalbild - (b) „brauchbarer“ Schwellwert - (c) Schwellwert zu niedrig - (d) Schwellwert zu hoch

Algorithmus: Optimaler Schwellwert

1. Eingabe sei ein Grauwertbild. Unter der Annahme, dass über das Bild keinerlei Informationen vorliegen, werden die vier Eckpixel als Hintergrund angenommen, die übrigen Pixel als Objekte.

2. Schritt t: ${\mu_B}^t$ und ${\mu_O}^t$ repräsentieren den Durchschnitt der Hintergrunds- bzw. Objektsgrauwerte, nach Segmentierung mittels des Schwellwerts T^t (s. Gleichung 2.1), mit:

$$ {\mu_B}^t := \frac{\sum_{(x,y)\in Hintergrund} I(x,y)}{\|HintergrundPixel\|} \qquad {\mu_O}^t := \frac{\sum_{(x,y)\in Objekte} I(x,y)}{\|ObjektPixel\|} $$

3. Setze

$$ T^{t+1} := \frac{{\mu_B}^t + {\mu_O}^t}{2} \tag{2.1} $$

   T^{t+1} ist neuer Schwellwert.

4. Wenn $T^{t+1} = T^t$, stoppe. Sonst gehe zu 2.

5. T^t ist „optimaler Schwellwert“.

Abbildung 2.2: Algorithmus - Optimaler Schwellwert

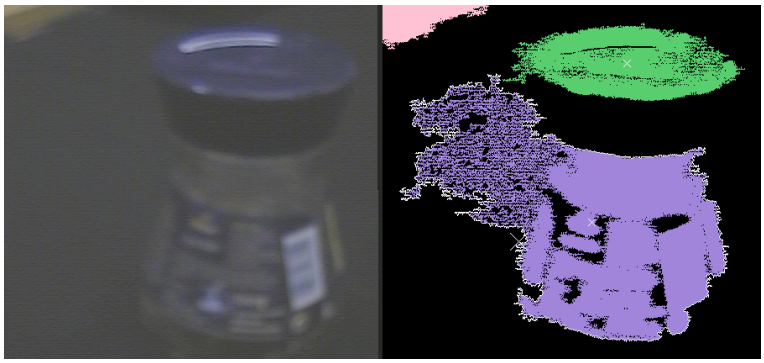


Abbildung 2.3: Der optimale Schwellwert führt nicht immer zu optimalen Ergebnissen

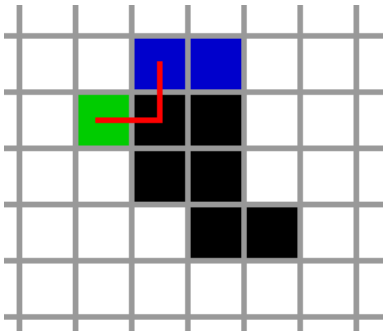


Abbildung 2.4: Fehl-Labeling wird bei späteren Pixeln korrigiert

müssen an dieser Stelle alle Pixel der linken Klasse zur Klasse des oberen Pixels hinzugefügt und die linke Klasse entfernt werden.

Das Ergebnis dieses Verfahrens ist eine Menge von Klassen, die jeweils die Pixel zusammenhängender Objekte beinhalten. Es werden nun all jene Klassen als „uninteressant“ markiert, deren Größe unter einen bestimmten Schwellwert fällt. Dieser wurde auf 1000 Pixel festgelegt, was ungefähr 0,2% des Gesamtbildes entspricht. Dadurch fallen alle Klassen heraus, die kleineren Bildstörungen entsprechen.

Alle „interessanten“ Klassen stellen potentiell greifbare Objekte dar. Um bestimmen zu können, welche dieser Klassen ein greifbares Objekt darstellt, müssen sie identifiziert werden. Damit dies möglich ist, muss deren Lage in Weltkoordinaten bestimmt werden.

Algorithmus: Labeling

1. Eingabe ist ein Binärbild, 0 markiert den Hintergrund, 1 markiert Objekte. Durchlaufe das Bild zeilenweise, beginnend oben links.
2. Setze P_n als das aktuelle Pixel. P_l das Pixel links von P_n und P_o oberhalb von P_n
3. $n = 0$, $class = 1$, $\widehat{C} = \emptyset$, function $classof(P_n) : N$
4. Wenn $P_n = 0$, $P_n \in \widehat{C}_0$
5. Wenn $P_n = 1$,
   a) Wenn $P_l = 0$ und $P_o = 0$, $\widehat{C}_{class}$ ist neue Klasse mit $P_n \in \widehat{C}_{class}$, $class := class + 1$
   b) Wenn $P_l = 0$ und $P_o = 1$, $P_n \in \widehat{C}_{classof(P_o)}$
   c) Wenn $P_l = 1$ und $P_o = 0$, $P_n \in \widehat{C}_{classof(P_l)}$
   d) Wenn $P_l = 1$ und $P_o = 1$
      i. Wenn $classof(P_l) = classof(P_o)$, $P_n \in \widehat{C}_{classof(P_l)}$
      ii. Wenn $classof(P_l) <> classof(P_o)$, $P_n \in \widehat{C}_{classof(P_l)}$ und $\widehat{C}_{classof(P_l)} := \widehat{C}_{classof(P_l)} \cup \widehat{C}_{classof(P_o)}$, $\widehat{C}_{classof(P_o)} := \emptyset$
6. $n := n + 1$
7. Wenn Bild noch nicht durchlaufen. gehe zu 4., sonst beende.
8. Alle $\widehat{C}_N <> \emptyset$ beinhalten die zusammenhängenden Pixel der Klasse N und repräsentieren die Objekte des Bildes.

Abbildung 2.5: Algorithmus - Labeling

2.1.2 Rückprojektion

Nachdem bekannt ist, welche Teile des Bildes zu Objekten gehören, kann deren Position bestimmt werden. Die potentiellen Objekte liegen nach der Segmentierung als eine Menge von Pixelclustern vor. Als Position des Objekts wird der Masseschwerpunkt des Pixelclusters angenommen, dieser liegt in Bildkoordinaten vor. Um bestimmen zu können, ob ein solcher Pixelcluster einem greifbaren Objekt entspricht, muss zunächst dessen Position in Weltkoordinaten bestimmt werden, um das Objekt für die Identifikation aus einer bekannten Position betrachten zu können. Das am häufigsten dazu angewandte Verfahren ist die Rückprojektion [25]. Diese ist sowohl bei stationären als auch bei mobilen Kameras anwendbar. In der Robotik lässt sich die Objektposition auch mittels „visual servoing“ bestimmen. Diese Regelungsmethode ist, wie der Name nahe legt, visuell basiert und ermöglicht es, die Zielposition eines mobilen Systems, wie einem Roboterarm, auf Grundlage von visuellen Sensoren zu erreichen.

Typischerweise werden visuell gesteuerte Greifsysteme in zwei Klassen unterteilt:

- die Lokalisation erfolgt einmalig und der Manipulator wird an die entsprechende Position gefahren („open-loop“)
- es findet eine ständige Kontrolle statt, an welcher Stelle sich der Manipulator momentan befindet und bei Abweichung entsprechend nachgeregelt („closed-loop“)

Siebel et al. haben in [23] gezeigt, wie eine robuste Positionierung eines Manipulators über einem Objekt mittels einer Handkamera möglich ist. Aus der Position des Manipulators lässt sich dann auf die Position des Objekts schließen.

In [8] wird vorgeschlagen, „open-loop“ und „closed-loop“ Verfahren zu kombinieren. Dies hat zur Folge, dass die Präzisionsanforderungen an beide Teilsysteme drastisch reduziert werden können. Dabei wird eine einzelne stationäre Kamera verwendet, um die Position des Manipulators zu bestimmen. Für die Kalibrierung dieses Systems wird der Manipulator an einige bekannte Positionen gefahren und von der Kamera verfolgt. Aus diesen Daten lässt sich auf die Dimensionen der Umwelt schliessen. Die Kombination von „open-loop“ und „closed-loop“ Steuerung wurde in dieser Arbeit auf eine Handkamera übertragen.

Dem Thema „visual servoing“ widmet sich das nächste Kapitel, zunächst wird auf Rückprojektion eingegangen. Wie diese beiden Verfahren kombiniert wurden, ist in Kapitel 3.4.3 beschrieben.

Die Pixel eines zweidimensionalen Bildes wieder zurück in Weltkoordinaten zu transformieren, wird als Rückprojektion bezeichnet. Der Name rührt daher, dass der Vorgang, welcher bei der Bildaufnahme stattfindet, einer Projektion gleichkommt. Projektion ist eine Abbildung von 3D nach 2D.

$$\begin{aligned} \text{Projektion } \widehat{T}_P\text{:} \quad & Dim \times Dim \times Dim \quad \rightarrow Dim \times Dim \\ \text{Rückprojektion } \widehat{T}_R\text{:} \quad & Dim \times Dim \quad \rightarrow Dim \times Dim \times Dim \end{aligned}$$

Es gibt verschiedene Arten der Projektion. Kameras bilden die Welt durch eine perspektivische Projektion ab. Die Parameter dieser Projektion werden hauptsächlich durch das verwendete Objektiv bestimmt.

Die wohl bekanntesten Ansätze für Rückprojektion verwenden Stereo-Kamera-Aufbauten. Das dabei auftretende Korrespondenz-Problem, die Identifikation desselben Merkmals in zwei unterschiedlichen Aufnahmen, macht die Verwendung dieses Systems allerdings alles andere als trivial. Dafür ist es mit Hilfe von Stereo-Kameras allerdings möglich, die Rückprojektion ohne zusätzliche Information über die Umwelt durchzuführen. Wird nur eine Kamera verwendet, so ist für eindeutige Rückprojektion zusätzliches Wissen über die Welt erforderlich.

Rückprojektion beruht auf dem Lochkameramodell. Dieses idealisiert die Linse einer Kamera durch einen dimensionslosen Punkt und liefert so ein sehr einfaches geometrisches Modell.

Um einen Bildpunkt in Weltkoordinaten zurückführen zu können, ist es nötig, einige Daten der verwendeten Kameras zu kennen. Dies beinhaltet unter anderem die Größe des CCD[2], die Brennweite oder den vertikalen sowie horizontalen Öffnungswinkel der Kamera. Welche dieser Daten von Nöten sind, wird durch das verwendete Verfahren und die vorhandenen Informationen über die Welt bestimmt.

Das verwendete Lochkameramodell und dessen einfache Geometrie reduziert die Rückprojektion nun auf das Berechnen einiger Schnittpunkte von Vektoren. Diese Vektoren werden aufgespannt durch den Linspunkt und dem zum entsprechenden Pixel korrispondierenden Punkt auf dem CCD.

Zur groben Lokalisierung des Objekts wird der Masseschwerpunkt des Pixelclusters in Weltkoordinaten projiziert. Da bei der Projektion Informationen verloren gehen, müssen diese bei der Rückprojektion unter Zuhilfenahme zusätzlicher Informationen rekonstruiert werden. In dieser Arbeit war die Höhe, auf welcher sich die Objekte befinden, im Vorfeld bekannt.

Die Lage der Objekte in der Welt wird durch einen dreidimensionalen Punkt $P_W = (x_W, y_W, z_W)$ bestimmt. Dieser steht in Relation zu einem Bezugspunkt $P_{W0} = (0, 0, 0)$. Die Lage der verwendeten Kamera ist in der Regel auch in Bezug auf dieses Koordinatensystem gegeben. Die Umwelt wird mithilfe von CCD Kameras erfasst, welche zweidimensionale Sensordaten liefern. Jedes Pixel des von der Kamera gelieferten Bildes ist ein Punkt $P_I = (x_I, y_I)$, deren Bezugspunkt $P_{I0} = (0, 0)$ meist der Pixel in der linken oberen Ecke des Bildes ist. Aus diesen Daten soll die Position eines Objekts in Weltkoordinaten bestimmt werden.

$$\begin{aligned} \widehat{T}_P: \quad & P_W(x_W, y_W, z_W) && \rightarrow P_I(x_I, y_I) \\ \widehat{T}_R: \quad & P_I(x_I, y_I) && \rightarrow P_W(x_W, y_W, z_W) \end{aligned}$$

Eine einfache Form der Projektion ist die sogenannte orthogonale Parallelprojektion auf die xy-Ebene. Bei dieser wird durch die Abbildung nur die z-Komponente

[2]Rückprojektion ist natürlich auch mit analogen Kameras möglich. Da in der Robotik aber hauptsächlich CCD Kameras verwendet werden, ist hier von der Größe des CCD die Rede.

gestrichen, es gilt also $P_I = (x_W, y_W)$.

NB: *Die nachfolgenden Transformationsmatrizen setzen die Verwendung von homogenen Vektoren bzw. Koordinaten voraus. Jeder dreidimensionale Punkt hat also noch eine vierte Komponente. Da diese in der Regel stets mit dem Wert 0 belegt ist, wurde sie in den aufgeführten Betrachtungen außen vor gelassen.*

$$\widehat{T}_{ortho_{xy}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

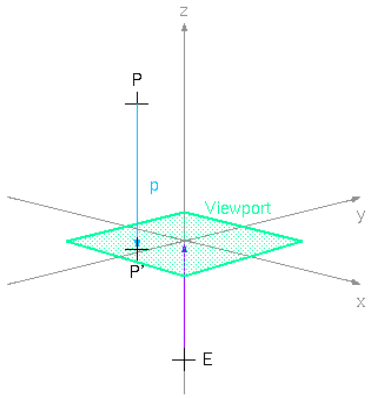


Abbildung 2.6: Orthogonalprojektion

Diese Art der Projektion ist in gewissem Maße verzerrungsfrei, die x- und y-Komponenten des Ursprungspunktes bleiben auch in der Abbildung erhalten. Allerdings bleiben nur vertikale und horizontale Kanten unverändert. Bei allen anderen Kanten kommt es zu einer Verzerrung der Winkel, mit denen diese aufeinandertreffen.

Anders ist es bei der perspektivischen Projektion. Bei dieser sind x- und y-Komponenten des Bildpunktes $P_I = (x_I, y_I)$ von der z-Komponente des Punktes P_W, sowie der Position des Betrachters, abhängig.

O.B.d.A. sei die Position des Betrachters durch den Punkt $E_W = (0, 0, d)$ gegeben, auch Augpunkt genannt. Die Projektion findet auf die xy-Ebene statt.

Der Bildpunkt ergibt sich dabei folgendermaßen: $P_I = (x_W \cdot \frac{d}{z_W}, y_W \cdot \frac{d}{z_W})$ [25]

$$\widehat{T}_{pers} = \begin{pmatrix} \frac{d}{z_W} & 0 & 0 & 0 \\ 0 & \frac{d}{z_W} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{d} \end{pmatrix} \tag{2.3}$$

Die Division durch z_W entspricht dabei einer Skalierung. Weiter entfernte Objekte erscheinen dadurch kleiner, näherliegende größer.

Das Bild, das eine Kamera liefert, ist eine perspektivische Projektion der Umwelt, modelliert durch die Aufnahme mittels einer Lochkamera.

Lochkameramodell

Das Lochkameramodell ist ein simplifiziertes Modell einer Kamera, welche die Linse einer Kamera als einen dimensionslosen Punkt annimmt. Es beruht auf der Camera Obscura. Diese entspricht einem dunklem Raum mit einem Loch in einer Wand, durch das Licht auf eine Projektionsfläche an der gegenüberliegenden Wand fallen kann.

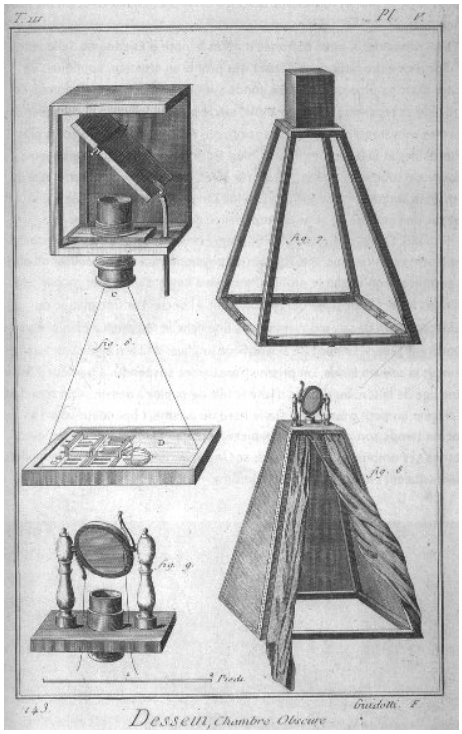


Abbildung 2.7: Camera Obscura

Der Augpunkt E liegt hier im Fokalpunkt und d entspricht dem Abstand zwischen Linse und Bildebene (CCD).

Der Vorteil dieses Modells ist seine einfache Geometrie. Spiegelt man die Bildebene in Bezug auf die xy Ebene, wie in Abbildung 2.9 zu sehen, so erhält man einen Bezug zwischen den Winkeln der Bildebene und der Welt ohne negative Koordinaten.

Das Modell entspricht somit einer perspektivischen Projektion, wie sie zuvor erläutert wurde. Auf dieser Grundlage kann eine Rückprojektion gefunden werden.

Wird nur eine Kamera verwendet, so ist diese Rückprojektion ohne zusätzliche Information über die Welt nicht eindeutig. Als Punkt P_I wird der Punkt des Bildes

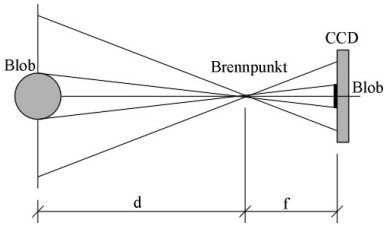


Abbildung 2.8: Lochkameramodell

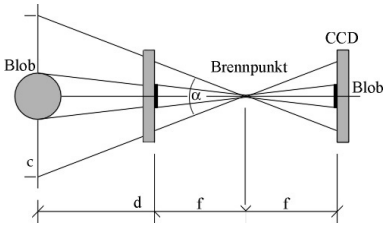


Abbildung 2.9: Lochkameramodell mit gespiegelter Bildebene

angenommen, der mit dem Mittelpunkt des CCD korrespondiert[3]. Diesem Punkt entsprechen alle Punkte P_W der Welt, welche auf der Normalen des CCD Mittelpunkts[4] liegen. Dieser Strahl wird häufig auch als n-Vektor bezeichnet und entspricht der in Blickrichtung zeigenden Achse des Kamerakoordinatensystems.

Soll für einen Punkt P_I nun die korrespondierende Weltkoordinate P_W bestimmt werden, muss eine der drei Komponenten von P_W bekannt sein. Eine Möglichkeit ist es, eine zweite Aufnahme, von einem anderen Ort aus aufzunehmen. Dabei muss das Korrespondenzproblem gelöst werden, die Frage, welcher Bildpunkt $P^2{}_I$ der zweiten Aufnahme, dem Punkt P_I entspricht. Ist dies bekannt, so entspricht der Schnittpunkt der Vektoren in die Bildebene dieser beiden Punkte P_W. In den meissten Fällen werden sich diese beiden Strahlen nicht kreuzen, weshalb der Ort ihrer höchsten Annäherungen P_W bestimmt.

Wird keine weitere Aufnahme verwendet, so muss die dritte Komponente im Vorfeld bekannt sein. Wie zuvor erwähnt, war in dieser Arbeit die Höhe z_W der Auflage der Objekte bekannt. Im einfachsten Fall gilt $z_W = 0$, die Objekte befinden sich also auf derselben Höhe wie der Ursprung des Weltkoordinatensystems. Der Weltpunkt entspricht dem Schnittpunkt des n-Vektors $\vec{n} = (n_x, n_y, n_z)$ mit der z-Ebene. Sei $\vec{p} = (p_x, p_y, p_z)$ die Position der Kamera in Weltkoordinaten.

$$\begin{aligned} x_W &= p_x + \lambda \cdot n_x \\ y_W &= p_y + \lambda \cdot n_y \end{aligned}$$

[3] P_I entspricht ohne Berücksichtigung der Verzerrung dem Mittelpunkt des Bildes.

[4] Häufig kommt es vor, dass das CCD einer Kamera „schief“ eingebaut wurde. Dabei kann es sich um nur wenige Bruchteile eines Grades handeln, allerdings wirkt sich diese Abweichung bei der Rückprojektion von weit entfernt liegenden Punkten dramatisch aus und muss mittels Kalibration ausgeglichen werden.

$$\lambda = \frac{z_W - p_z}{n_z}$$

Somit kann also der Mittelpunkt eines Bildes rücktransformiert werden. Für die übrigen Bildpunkte muss die Größe des CCD bekannt sein. Soll ein beliebiger Punkt P'_I rücktransformiert werden, so liegen dessen korrespondierenden Weltpunkte auf dem Strahl $\vec{S'}$, der durch den Linspunkt und dem zu P'_I korrespondierenden Punkt auf dem CCD verläuft.

Ist nun bekannt, wie groß das CCD ist, kann der Winkel γ_* zwischen $\vec{S'}$ und $\vec{n}$ berechnet werden. $\vec{S'}$ entspricht somit dem um γ_* rotiertem Strahl $\vec{n}$.

Mithilfe dieses Verfahrens ist es also möglich, jedem Bildpunkt seinen korrespondierenden Weltpunkt zuzuordnen.

2.1.3 „visual servoing“

Bei der Rückprojektion mit nur einer Kamera ergibt sich ein Problem: es können nur diejenigen Bildpunkte exakt in Weltkoordinaten transformiert werden, deren ursprüngliche Lage exakt auf der angenommenen Höhe z_W liegt. Liegt dieser ober- oder unterhalb z_W, so wird der rückprojizierte Punkt weiter entfernt bzw. näher an der Kamera berechnet, als er tatsächlich ist. Da der Masseschwerpunkt eines Objekts aus Seitenansicht oberhalb der Ebene liegt, auf der es steht, werden nahezu alle Punkte zu weit entfernt projiziert. Unter den in dieser Arbeit verwendeten Bedingungen wurde die Lage der Objekte im Mittel um 2 cm überschätzt. Um eine exakte Lokalisierung zu gewährleisten, wurde deshalb nach der Rückprojektion „visual servoing“ verwendet. Unter „visual servoing“ wird ein Regelmechanismus verstanden, der bildbasiert arbeitet.

Es soll nicht sehr detailiert auf diese Art der Regelung eingangen werden, da sie in dieser Arbeit mehr die Rolle einer unterstützenden Funktion wahrnimmt, als bestimmend für die Erlangung des Zieles zu sein. Eine Implementierung für „visual servoing“ kann in [23] gefunden werden.

Die Aufgabe eines Reglers ist es, aufgrund von Sensordaten den Ist-Zustand eines Systems zu bestimmen, daraus auf die notwendige Korrektur des Systems zu schließen, um einen gewünschten Soll-Zustand zu erlangen oder zu wahren, und diese auszuführen.

Beim „visual servoing“ liefert eine Kamera die nötigen Sensordaten. Soll ein System beispielsweise eine Greifhand über einem Objekt positionieren, so kann das dazu nötige „Endbild“ als Soll-Zustand in den Regler einfließen. Der Regler wertet die Aufnahmen der Kamera zu beliebigen Zeitpunkten - der Ist-Zustand - aus und berechnet daraus die notwendige Korrektur zur Erlangung des Soll-Zustands.

Befindet sich das gesamte Objekt stets im Blickfeld, so kann der Abstand eines bekannten Merkmals dieses Objekt zwischen Ist- und Soll-Zustand als Grundlage für die Regelung dienen. Die Güte der Regelfunktion bestimmt dann, auf welche

Weise sich dem Soll-Zustand genähert wird. In dieser Arbeit war dieses Merkmal der Masseschwerpunkt des Objekts. Die Kamera wird über der Position, welche die Rückprojektion liefert, positioniert, wodurch sich nur noch das zu lokalisierende Objekt im Blickfeld der Kamera befindet. Der Regler positioniert die Kamera dadurch schnell und zuverlässig über dem Masseschwerpunkt.

Typisches Fehlverhalten bei Reglern ist das sogenannte Überschwingen und das Hinauslaufen aus dem regelbaren Bereich. Dies kann durch geeignet gewählte Regelfunktionen vermieden werden (s. Kapitel 3.4.4).

2.2 Erkennen

Nachdem die Szene „gesehen“ wurde, müssen die in ihr vorhandenen Objekte auch „erkannt“ werden. Dies ist die Interpration der Szene. Ziel dabei ist es, die jeweilige Klassenzugehörigkeit der Objekte zu bestimmen, aus welcher auf ihre Gestalt geschlossen und eine interne 3D-Repräsentation des Objekts erstellt werden kann. Dies ist notwendig, um eine Griffplanung durchzuführen zu können.

Hier muss generell zwischen zwei Varianten unterschieden werden:

- die Objekte können im Vorfeld bekannt sein
- es sollen unbekannte Objekte gegriffen werden

Ein Kernproblem bei unbekannten Objekten stellt die Ungewissheit über deren Masse sowie Oberflächenbeschaffenheit dar. Sind diese beiden Eigenschaften nicht bekannt, ist es wahrscheinlich, dass ein als anwendbar bestimmter Griff nicht fähig ist, das Objekt tatsächlich zu greifen. Ist die Masse zu hoch, kann die an den Kontaktpunkten angebrachte Kraft zu gering sein und das Objekt wird demzufolge nicht stabil gegriffen werden können. Dasselbe wird passieren, wenn der Reibungskoeffizient geringer ist als erwartet[5].

Im Fall von bekannten Objekten liegen Kenntnisse über die Objektklassen im Vorfeld vor. Dies beinhaltet sowohl ihre Gestalt, als auch deren Masse und Oberflächenbeschaffenheit. Die Eigenschaften dieser Merkmale bestimmen, welche Verfahren zur Klassifikation Anwendung finden können. Da die Klassifikation in dieser Arbeit auf visuellen Merkmalen stattfinden soll, ist es nötig, solche für jedes bekannte Objekt vorliegen zu haben. Jedes Objekt soll durch eine Menge von visuellen Merkmalen eindeutig beschrieben werden können. Sie müssen also derart gewählt werden, dass sie die Objektklassen separieren können.

Ein gängiges Verfahren, um solche Merkmale aus Bilddaten zu extrahieren, stellt die Principal Component Analysis dar [24]. Aus den Methoden der Statistik findet

[5] Reibung und Kraft stehen in direktem Zusammenhang. Ist die applizierte Kraft zu gering, ist auch die Reibung zu gering, als dass das Objekt stabil gehalten werden könnte. Umgekehrt können Objekte, deren Oberfläche für einen hohen Reibungskoeffizienten zwischen Finger und Objekt sorgen, mit deutlich weniger Kraft gegriffen werden.

die Principal Component Analysis häufig Anwendung in der Bildverarbeitung. In diesem Verfahren wird versucht, aus Daten mit vielen Eigenschaften einige wenige latente Faktoren zu extrahieren, die für diese Eigenschaften bestimmend sind.

Als Trainingsdaten dienen unter bekannten Parametern aufgenommene Bilder der zu erkennenden Objekte. Die PCA extrahiert aus diesen eine Reihe von Merkmalen, welche für die Unterscheidung der Objekte bestimmend sind.

In [28] wurde ein auf PCA basierendes Verfahren namens „Eigenfaces“ entwickelt, welche es in einer Beispielanwendung ermöglicht, verschiedene Personen anhand von Gesichtsaufnahmen zu unterschieden. Dabei wurden mehrere Aufnahmen derselben Person zu einem Bild zusammengefügt. Dies sollte Schwankungen ausgleichen. So war es beispielsweise möglich, eine Brille tragende Person, auch dann noch zu erkennen, wenn sie keine Brille trägt. Dies wurde ermöglicht, indem in die Trainingsdaten Aufnahmen mit und ohne Brille einflossen.

Eine Weiterentwicklung dieses Verfahrens nach [27] wurde angewendet, um die zu greifenden Objekte zu klassifizieren. Dabei wurde jedes Objekt durch eine Reihe von Aufnahmen aus verschiedenen Positionen repräsentiert, was eine robuste Klassifikation ermöglichte.

2.2.1 Objektrepräsentation und Oberflächenanalyse

In der analytischen Geometrie werden Objekte häufig durch ihre parametrische Darstellung repräsentiert. Dies ist für viele analytische Verfahren eine geeignete Darstellung. Die in dieser Arbeit verwendeten Methoden setzten allerdings eine diskrete Punktdarstellung von Objekten voraus.

Ein Objekt O sei eine Menge von Punkten $P_n \in R^3$, welche dessen Oberfläche beschreiben.

Es ist zweckmäßig, komplexe Objekte als einen Verbund aus geometrischen Primitiven darzustellen. Geometrische Primitive zeichnen sich dadurch aus, dass sie durch eine sehr geringe Menge von Parametern eindeutig dargestellt werden können.

Kugel Radius

Zylinder Radius, Höhe

Quader Höhe, Breite, Tiefe

Anhand dieser Eigenschaften sollen die Oberflächenpunkte der Objekte generiert werden. Für einen Quader ist dies trivial, da er aus sechs Ebenen besteht. Die acht Eckpunkte des Würfels sind die kleinstmögliche Punktmenge für eine eindeutige Repräsentation des Objekts. Sind mehr Oberflächenpunkte notwendig, lassen sich diese einfach berechnen. Entsprechendes gilt für Polyeder mit beliebigen Ecken. Kugeln und Zylinder lassen sich als Polyeder mit einer hohen Anzahl an Ecken darstellen. Dazu ist eine parametrische Darstellung nötig, welche es ermöglicht, die Oberflächenpunkte zu berechnen. Diese lässt sich aus Polarkoordinaten herleiten.

Ein Punkt $P_P = (r_P; \phi_P)$ in Polarkoordinaten wird repräsentiert durch einen Abstand r_P zum Ursprung und einem Winkel ϕ_P. In kartesische Koordinaten lässt sich dieser umrechnen durch

$$\begin{aligned} x_P &= r_P \cdot cos(\phi_P) \\ y_P &= r_P \cdot sin(\phi_P) \end{aligned}$$

Wird diese Defintion durch eine Höhe h_Z erweitert, so führt dies zu einem zylindrischen Koordinatensystem, $P = (r_Z; \phi_Z; h_Z)$:

$$\begin{aligned} x_Z &= r_Z \cdot cos(\phi_Z) \\ y_ZP &= r_Z \cdot sin(\phi_Z) \\ z_Z &= h_Z \end{aligned}$$

Durch die Einführung eines zweiten Winkels θ, welcher sich orthogonal zu ϕ verhält, wird ein Kugelkoordinatensystem definiert, $P_K = (r_K; \phi_K; \theta_K)$

$$\begin{aligned} x_K &= r_K \cdot sin(\phi_K) \cdot cos(\theta_K) \\ y_K &= r_K \cdot sin(\phi_K) \cdot sin(\theta_K) \\ z_K &= r_K \cdot cos(\theta_K) \end{aligned}$$

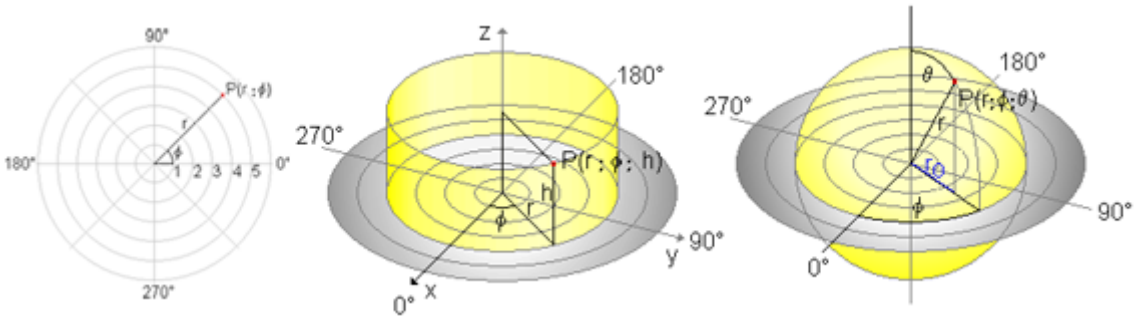


Abbildung 2.10: Von Polar- zu Kugelkoordinaten

Mithilfe dieser Definitionen ist es möglich, Oberflächenpunkte beliebiger Dichte für Kugeln und Zylinder zu berechnen, indem über ϕ_Z und h_Z bzw. ϕ_K und θ_K in diskreten Werten iteriert wird.

Normalen

Die Normale $\vec{n}$ eines Punktes der Oberfläche gibt Aufschluß über die Krümmung an diesem Punkt. Für den reibungslosen Fall der Griffberechnung entspricht die Normale am Kontaktpunkt der Wirkrichtung der Kraft an diesem Punkt.

Liegt eine parametrische Darstellung $V(s,t)$ der Oberfläche vor, so kann die Normale für jeden Punkt exakt bestimmt werden:

$$\vec{n} = \frac{\delta V}{\delta s} \times \frac{\delta V}{\delta t} \qquad [11]$$

Ist die parametrische Darstellung nicht mehr verfügbar, so lässt sich die Normale approximieren. Die Oberfläche kann anstatt als Punktmenge auch als eine Menge von Dreiecken aufgefasst werden. Dies ist in den meisten Polygon basierten 3D-Bibliotheken der Fall. Das Kreuzprodukt zweier Vektoren liefert einen Vektor, welcher orthogonal zu diesen liegt. Die Normale eines Dreiecks $D = (P_1, P_2, P_3)$ der Oberfläche lässt sich also aus den Kanten des Dreiecks berechnen. Diese werden als Vektoren aufgefasst.

$$\begin{aligned} N_1 &= \vec{P_1P_2} \times \vec{P_1P_3} \\ N_2 &= \vec{P_2P_1} \times \vec{P_2P_3} \\ N_3 &= \vec{P_3P_1} \times \vec{P_3P_2} \end{aligned}$$

Die drei Normalen eines Dreiecks sind parallel, es genügt also, nur eine Normale zu berechnen. Sie entspricht der Ausrichtung des Dreiecks.

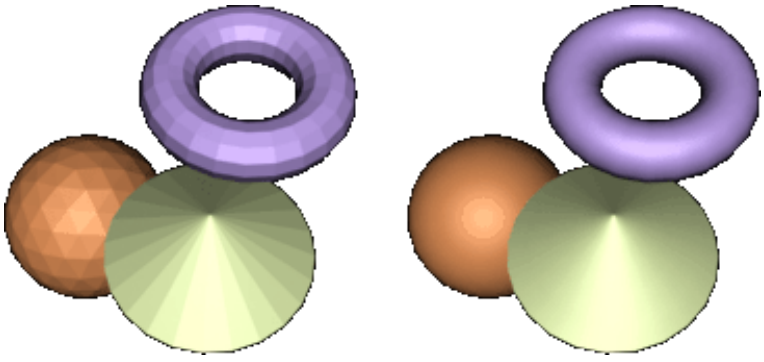


Abbildung 2.11: Dreiecksnormalen (links) und parametrische Normalen

Ein Nachteil dieser Methode ist die nicht Eindeutigkeit der Normalen an den Stellen, an denen sich mehrere Dreiecke berühren. An den Kanten sind dies nur zwei Normalen, an den Ecken stets mehr als zwei.

Wie damit umgegangen werden soll, ist Thema ständiger Diskussionen in der Bilderzeugung. Die Normalen eines Objekts sind entscheidend für Shading-Verfahren.

Ohne weitere Behandlung dieses Problems wird von „flat-shading“ gesprochen. Dies führt zu sehr „kantigen“ Objekten, wie in Abbildung 2.11 zu sehen ist.

Eine mögliche Variante ist es, an den Ecken der Dreiecke den Mittelwert aller Normalen zu berechnen und diesen als Normale für diesen Punkt zu setzen. Dies ist bei besonders „scharfen“ Kanten allerdings mit Vorsicht zu genießen, da sich die Normalen an solchen Stellen sehr stark unterscheiden können.

2.2.2 Principal Component Analysis

In dieser Arbeit wurde eine Vielzahl von bekannten Objekten verwendet, um Griffe für diese zu planen. Um diese anhand der Szenenbilder zu klassifizieren, wurde eine auf PCA aufbauende Klassifikation nach [27] verwendet. Im deutschen Sprachraum wird für „Principal Component Analysis“ auch der Ausdruck „Hauptkomponentenanlyse“ verwendet. Dieses statistische Verfahren kann verwendet werden, um bestimmende Eigenschaften aus einer Vielzahl von Eigenschaften für unterschiedliche Objekte zu bestimmen. Nachfolgend soll anhand eines Beispiels beschrieben werden, was die grundsätzliche Idee hinter PCA ist und wie diese Form der Merkmalsanalyse auf einen Datensatz angewendet werden kann.

	E_1	E_2	E_3
A	0,12	2,33	1,01
B	0,09	7,83	1,40
C	0,14	0,03	0,24
D	0,07	12,02	0,71

Tabelle 2.1: Beispieldaten

Als Beispiel sei der Datensatz E in Tabelle 2.2.2 betrachtet. Die Objekte A - D besitzen jeweils drei Eigenschaften. Diese unterscheiden sich in E_1 und E_3 wenig, wohingegen die Eigenschaft E_2 sehr deutliche Unterschiede aufweist.

Für die Separation der Objekte wäre ein 1-Tupel, welches nur Eigenschaft E_2 enthält, also ebenso aussagekräftig wie ein 3-Tupel aller Eigenschaften. Die Daten für die Klassifikation können somit um zwei Dimensionen gekürzt werden.

Bei Daten mit wesentlich mehr Eigenschaften wird dies nicht mehr so einfach zu erkennen sein. PCA bietet dafür eine Lösung.

Zunächst ist es nötig, die Kovarianzmatrix des Datensatzes zu bestimmen. Kovarianz ist ein Maß, welches Aufschluß darüber gibt, wie weit zwei Dimensionen X und Y eines Datensatzes in Bezug aufeinander vom Mittelwert abweichen.

$$cov(X,Y) = \frac{\sum^{n}{}_{i=1}(X_i - \overline{X})(Y_i - \overline{Y})}{n-1} \tag{2.4}$$

Für Daten mit mehr als zwei Dimensionen ergibt sich die sogenannte Kovarianzmatrix. Diese beinhaltet die Kovarianz zwischen allen Dimensionen. Sie ist stets quadratisch und besitzt bei n Dimensionen die Größe $n \times n$.

$$C^{n\times n} = (c_{i,j}, c_{i,j} = cov(Dim_i, Dim_j)) \tag{2.5}$$

Das angeführte Beispiel besitzt drei Dimensionen, die zugehörige Kovarianzmatrix besitzt demzufolge die Größe 3×3 und setzt sich folgendermaßen zusammen:

$$C = \begin{pmatrix} cov(E_1, E_1) & cov(E_1, E_2) & cov(E_1, E_3) \\ cov(E_2, E_1) & cov(E_2, E_2) & cov(E_2, E_3) \\ cov(E_3, E_1) & cov(E_3, E_2) & cov(E_3, E_3) \end{pmatrix} \tag{2.6}$$

Die Diagonale der Matrix besitzt Einträge der Gestalt $cov(X, X)$, dies entspricht der Varianz von X. Zudem gilt $cov(X, Y) = cov(Y, X)$, die Matrix ist also symmetrisch.

Damit die PCA korrekt funktioniert, muss der Mittelwert von jeder Dimension der Daten abgezogen werden [24]. Dies gewährleistet Eingabedaten, deren Mittelwert 0 ist. Die Kovarianzmatrix für die so behandelten Beispieldaten lautet:

$$C = \begin{pmatrix} 0,0156667 & 0,609967 & 0,110167 \\ 0,609967 & 70,4064 & 7,28557 \\ 0,110167 & 7,28557 & 1,1806 \end{pmatrix} \tag{2.7}$$

PCA basiert auf Eigenwerten und -vektoren. Der Eigenvektor $\vec{e}$ und die zugehörigen Eigenwerte λ definieren sich für eine quadratische Matrix C durch:

$$C \cdot \vec{e} = \lambda \cdot \vec{e} \tag{2.8}$$

Die Eigenwerte und -vektoren für die Kovarianzmatrix der Beispieldaten lauten:

$$\lambda = \begin{pmatrix} 0,00513143290 13 \\ 0,4273164211668 \\ 71,1702188459320 \end{pmatrix} \tag{2.9}$$

$$\vec{e} = \begin{pmatrix} -0,99382541037725 & 0,11061465201189 & 0,008686336756185 \\ -0,00286786045718 & -0,10386931953125 & 0,994586818652108 \\ 0,11091811874030 & 0,98842074200231 & 0,103545196495234 \end{pmatrix} \tag{2.10}$$

Diese Eigenvektoren haben zwei interessante Eigenschaften. Sie stehen rechtwinkelig zueinander und spannen ein Koordinatensystem innerhalb der normierten Daten auf. Die Achsen dieses Koordinatensystems geben Aufschluss über die Verteilung der Daten.

Ihre zugehörigen Eigenwerte geben an, wie signifikant die entsprechende Achse in Hinsicht auf die Unterscheidung der Daten ist. Je höher der Eigenwert, desto signifikanter.

Der Eigenvektor mit dem höchsten Eigenwert wird Hauptkomponente, bzw. „principal component“, genannt. Daher der Name dieses Verfahrens.

Wie anhand des Beispiels gesehen werden kann, besitzt der dritte Eigenvektor den höchsten Eigenwert, er dominiert bei weitem. Dies spielt eine Rolle bei dem nächsten Schritt. Es wird zunächst ein sogenannter Merkmalsvektor f gebildet. Dies ist eine Matrix, deren Komponenten die Eigenvektoren sind, sortiert nach deren Eigenwert.

$$f = \begin{pmatrix} \vec{e}_1 & \vec{e}_2 & \dots & \vec{e}_n \end{pmatrix} \tag{2.11}$$

Dabei müssen nicht alle Eigenvektoren verwendet werden. Die Idee hinter der PCA war, die Dimension der Daten zu reduzieren und sie nur durch signifikante Eigenschaften zu repräsentieren.

Für das Beispiel kann der dritte Eigenvektor als einzige Komponente des Merkmalsvektors verwendet werden.

$$f = \begin{pmatrix} 0,008686336756185 \\ 0,994586818652108 \\ 0,103545196495234 \end{pmatrix} \tag{2.12}$$

Die Daten werden nun in das Koordinatensystem transformiert, welches durch den Merkmalsvektor aufgespannt wird, den sogenannten Eigenraum. Dazu wird dieser mit den angepassten Daten D_a multipliziert und liefert den neuen Datensatz P.

$$P = f^T \cdot {D_a}^T \tag{2.13}$$

$$P = \begin{pmatrix} 0,00868\dots \\ 0,99458\dots \\ 0,10354\dots \end{pmatrix}^T \cdot \begin{pmatrix} 0.015 & -0.015 & 0.035 & -0.035 \\ -3.2225 & 2.2775 & -5.5225 & 6.4675 \\ 0.17 & 0.56 & -0.6 & -0.13 \end{pmatrix} \tag{2.14}$$

$$= \begin{pmatrix} -3.18732304465 \\ 2.32302649447 \\ -5.55442880212 \\ 6.4187253523 \end{pmatrix}^T \tag{2.15}$$

Der neue Datensatz P besitzt lediglich eine Dimension und ist geeignet, alle vier Objekte voneinander zu unterscheiden. Dies deckt sich mit der Beobachtung, dass nur eine Eigenschaft maßgebend für die Objekte ist.

Soll nun ein neues Objekt klassifiziert werden, so muss dies nicht mehr in Bezug auf die Ursprungsdaten geschehen, sondern kann anhand der PCA Daten ausgeführt werden. Im Beispiel wurden die Daten auf eine Dimension reduziert, was das Berechnen des Abstandes eines neuen Objekts zu den vorhandenen Objekten trivial macht.

Anwendung auf die Bildverarbeitung

Wie erwähnt, wird in [27] ein auf PCA basierendes Verfahren vorgestellt, welches die Klassifikation von Gesichtern ermöglicht. Dieses wurde angewendet, um die in dieser Arbeit verwendeten Objekte zu klassifizieren.

Ein Rasterbild eines Objekts der Größe $n \times m = N$ kann als ein Vektor $\vec{\Gamma}$ aufgefasst werden, dessen Komponenten die Pixel x_n des Bildes darstellen.

$$\vec{\Gamma} = (x_1, x_2, \ldots, x_N) \tag{2.16}$$

Jede Komponente des Vektors stellt eine Dimension der Objekteigenschaften dar. Es werden M Aufnahmen von Objekten verwendet. Das durchschnittliche Objekt $\vec{\Psi}$ dieser Aufnahmen entspricht

$$\vec{\Psi} = \frac{1}{M} \sum_{i=1}^{M} \vec{\Gamma_i} \tag{2.17}$$

Der für die PCA angepasste Datensatz $\vec{\Phi}$ setzt sich demzufolge zusammen aus

$$\vec{\Phi_i} = \vec{\Gamma_i} - \vec{\Psi}, \;\; i = 1, \ldots, M \tag{2.18}$$

Die Kovarianzmatrix $C = \vec{A}\vec{A}^T$ dieses Datensatzes hat die Größe $N \times N$, mit $\vec{A} = [\vec{\Phi_1}, \ldots, \vec{\Phi_M}]$. Für eine derart Große Matrix die Eigenvektoren zu bestimmen, wäre sehr rechenintensiv. In einem Traingingsdatensatz werden in der Regel weniger Bildaufnahmen als Bildpixel vorhanden sein ($M < N$), daher wird es nur $M - 1$ statt N aussagekräftige Eigenvektoren geben. Aus diesem Grunde werden zunächst die Eigenvektoren $\vec{V}$ der $M \times M$ Matrix $L = \vec{A}^T\vec{A}$ berechnet. Die Eigenvektoren $\vec{U}$ der Matrix C entsprechen dann einer linearen Kombination von $\vec{\Phi}$, gewichtet mit $\vec{V}$

$$\vec{U} = \vec{A} \cdot \vec{V} \tag{2.19}$$

Für die Klassifikation der Objekte sind weniger als M Eigenvektoren ausreichend. Die Anzahl M' der benötigten Eigenvektoren wird über einen Schwellwert θ_λ bestimmt, welcher auf das Verhältnis der Summe der Eigenwerte angewendet wird

$$M' = \min_r \left\{ r | \frac{\sum_{i=1}^{r} \lambda_L}{\sum_{i=1}^{M} \lambda_L} > \theta_\lambda \right\} \tag{2.20}$$

Die Transformation in den Eigenraum erfolgt wie im vorherigen Abschnitt beschrieben und resultiert in einem Vektor $\vec{\Omega_k}$ für jedes Trainingsbild.

Um nun ein neues Bild $\vec{\Delta}$ zu klassifizieren, wird es zunächst in den Eigenraum projiziert und in den Vektor $\vec{\Omega}$ transformiert

$$\vec{\Omega} = \vec{U}^T \vec{\Delta} - \vec{\Psi} \tag{2.21}$$

Zum Vergleich mit den Trainingsdaten werden zwei Abstände berechnet. Der Abstand ϵ_k von $\vec{\Omega}$ zu den Vektoren $\vec{\Omega_k}$ der anderen Klassen und der Abstand ϵ des originalen Bildes $\vec{\Delta}$ mit dem aus dem Eigenraum rekonstruiertem Bild $\vec{\Delta_f}$. Letzteres ist nötig, um entscheiden zu können, ob das Bild bzw. Objekt überhaupt in diesem Eigenraum darstellbar ist, also zu den trainierten Klassen gehört.

$$\epsilon_k^2 = \left\| \vec{\Omega} - \vec{\Omega_k} \right\|^2 \tag{2.22}$$

$$\epsilon^2 = \left\| \vec{\Delta} - \vec{\Delta_f} \right\|^2 \tag{2.23}$$

$$\vec{\Delta_f} = \vec{U}\vec{\Delta} + \vec{\Psi} \tag{2.24}$$

Der maximale Abstand θ_c, den ein Vektor von dem Vektor einer anderen Klassen haben darf, ist definiert als die Hälfte des Abstandes zwischen den zwei am weitesten voneinander entfernten Vektoren des Eigenraums.

Für $\vec{\Omega}$ können nun drei Fälle eintreten:

1. $\epsilon \geq \theta_c$, das Eingabebild ist nicht Teil des Eigenraums und entspricht somit auch keiner bekannten Klasse

2. $\epsilon < \theta_c$ und $\forall k, \epsilon_k \geq \theta_c$, das Eingabebild ist Teil des Eigenraums, aber entspricht keiner bekannten Klasse

3. $\epsilon < \theta_c$ und $\epsilon_{k*} = \min_k \{\epsilon_k\} < \theta_c$, das Eingabebild gehört zu Klasse $k*$

Häufig wird auch eine „nächster Nachbar“ Klassifizierung verwendet. Dabei wird das zu klassifizierende Objekt derjenigen Klasse zugeordnet, zu der es den geringsten Abstand hat. Hierbei treten Fehlklassifikationen allerdings häufiger auf, als mit der zuvor beschriebenen Methode.

2.2.3 Klassifizierung unbekannter Objekte

Im Laufe dieser Arbeit wurde ein Verfahren überlegt, welches es ermöglichen soll, unbekannte Objekte anhand ihres Umrisses zu identifizieren. Aus diesen Daten wird ein internes Modell des Objekts erstellt. Die Überlegung ist, dass sich beliebige Objekte durch geometrische Primitive darstellen lassen. Dabei ist es für diese Arbeit nicht einmal nötig, das Objekt absolut präzise zu erfassen. Wie Versuche über biologisch motivierte Griffe in [10] gezeigt haben, ist es sinnvoll, Objekte an bestimmten Stellen zu greifen. Diese liegen in Bereichen, an denen das Objekt besonders „lang“ ist, da der Schwerpunkt innerhalb dieser Bereiche vermutet werden kann.

Ein Zylinder der Höhe h ist beispielsweise an der Stelle $\frac{h}{2}$ besonders „lang“. Ein Griff, welcher dort platziert wird, greift das Objekt genau in der Mitte. Oberhalb und unterhalb des Griffes befindet sich dann „besonders viel Objekt“.

Dies ist die Art und Weise, wie Menschen Objekte meist greifen.

Solche Stellen an Objekten zeichnen sich dadurch aus, dass sie in ihrer Form geometrischen Primitiven sehr nahe kommen. Es liegt also nahe, für die Klassifizierung solche Bereiche zu finden, deren Form zu bestimmen und als Objektrepräsentation zu verwenden.

Dies kann als eine Vorauswahl in der Griffplanung angesehen werden, da sich diese nur noch auf einen Bereich beschränkt, welcher implizit als günstig angenommen wird.

Die Klassifizierung der Form soll anhand zweier Bildaufnahmen geschehen, eine, welche das Objekt von oben zeigt und eine, welche es von der Seite zeigt.

Werden geometrische Primitive aus diesen Positionen betrachtet, fällt auf, dass sie sich aus einer Kombination von zwei 2D-Primitiven rekonstruieren lassen:

Kugel Kreis & Kreis

Zylinder Kreis & Rechteck

Quader Rechteck & Rechteck

Es werden also Methoden benötigt, welche die Form eines 2D-Objekts in einem Bild bestimmen können.

Üblicherweise geschieht dies durch eine Menge von Eigenschaften, die sich für ein solches Objekt bestimmen lassen. Für diese ist es nötig, den Umfang und die Masse des Objekts zu kennen. Die Masse ist die Anzahl der Pixel, welche zu diesem Objekt gehören. Der Umfang lässt sich mittels eines einfachen Algorithmus berechnen. Dieser „wandert“ entlang des Umrisses und findet alle Pixel, die zum Rand gehören.

Die Verwendung eines Kantendetektors wäre naheliegender. Auf Binärbilder angewandt, führt dieser zu sehr eindeutigen Ergebnissen. Allerdings ist es nicht garantiert, dass die gefundenen Kanten ausschließlich zum Rand des Objekts gehören. Nach der Segmentierung durch einen Schwellwert passiert es häufig, dass zwar der

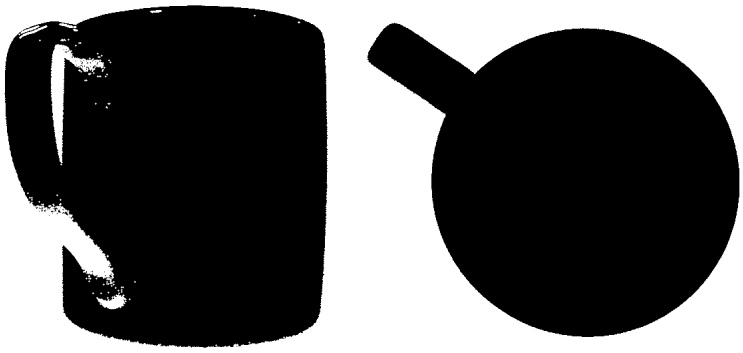


Abbildung 2.12: Front- und Topansicht eines Objekts

Algorithmus: Inner boundary tracing

1. Durchlaufe das Bild, beginnend oben links, bis eine neue Region gefunden wurde; dieses Pixel P_0 hat dann den kleinsten Spalten - sowie Zeilenwert - aller Pixel dieser Region und ist das Startpixel der Regionsgrenze. *dir* speichert die Richtung der vorherigen Bewegung entlang der Grenze und sei hier
   (a) $dir = 3$, wenn die Grenze im Viererverbund gefunden wurde
   (b) $dir = 7$, wenn die Grenze im Achterverbund gefunden wurde
2. Durchsuche die 3×3 Nachbarschaft des aktuellen Pixels entgegen dem Uhrzeigersinn, beginnend mit dem Pixel in Richtung
   (a) $(dir + 3)$ mod 4
   (b) $(dir + 7)$ mod 8, wenn *dir* gerade
   $(dir + 6)$ mod 8, wenn *dir* ungerade

   Das erste Pixel mit demselben Wert wie der aktuelle Pixel ist ein neues Element P_n der Grenze. Setze *dir* entsprechend.
3. Wenn $P_n = P_1$ und $P_{n-1} = P_0$, stoppe. Sonst wiederhole Schritt 2.
4. Die innere Grenze entspricht den Pixeln $P_0 \ldots P_{n-2}$.

Abbildung 2.13: Algorithmus - Inner boundary tracing

Rand eines Objekts geschlossen ist, innerhalb des Objekts allerdings Lücken auftreten.

Der in Abbildung 2.13 präsentierte Algorithmus „wandert“ entlang des äußeren Randes eines Objekts und umgeht somit das Problem der Lücken. Der Name „inner boundary tracing“ rührt daher, dass dieser Algorithmus die Randpixel liefert. Somit entspricht der Umfang der Entfernung zwischen diesen Pixeln. Im Gegensatz dazu entspräche „outer boundary“ den Pixeln, welche außerhalb der Randpixel liegen.

Erkennen der Form

Drei Eigenschaften eines Objekts werden besonders häufig verwendet, um dessen Form zu bestimmen:

- Rechteckigkeit, das Verhältnis zwischen Masse und des umschließenden Rechtecks eines Objekts
- Rundheit, das Verhältnis zwischen Masse des Objekts und eines Kreises mit derselben Fläche
- Kompaktheit, das Verhältnis zwischen dem Quadrat des Umfangs und der Masse des Objekts

Sei b die Breite und h die Höhe des das Objekt umschließenden Rechtecks, m = Masse, u = Umfang.

$$Rechteckigkeit = \frac{m}{b \cdot h} \tag{2.25}$$

$$Rundheit = \frac{u^2}{4\pi \cdot m} \tag{2.26}$$

$$Kompaktheit = \frac{u^2}{m} \tag{2.27}$$

Rechteckigkeit liefert Werte im Bereich $(0, 1]$, wobei 1 für eine perfektes Rechteck steht.

Rundheit und Kompaktheit liefern Werte im Bereich $[1, \infty)$. In der digitalen Bildverarbeitung ändert sich dieser Bereich allerdings in $[16, \infty)$ [25]. 1 bzw. 16 deuten auf einen perfekten Kreis hin.

Von 2D zu 3D

Aus den erkannten 2D-Primitiven ein 3D-Primitiv zu erstellen, ist ein einfacher Schritt. Die Größe der 2D-Primitive diktiert die Parameter des 3D-Primitivs.

Aus der Frontansicht lässt sich auf die Höhe und Breite schließen, aus der Topansicht auf die Breite und Tiefe.

Um die Größe der Objekte in Weltkoordinaten zu bestimmen, kann die Rückprojektion dienen. Dabei ist bei Rückprojektion mittels einer Kamera die z-Ebene entsprechend zu setzen. Geschieht dies nicht, so wird das Objekt aus der Topansicht größer berechnet als es tatsächlich ist.

Mit diesem Ansatz wurde im Laufe dieser Arbeit experimentiert und die Ergebnisse waren nicht überzeugend, weshalb die Klassifizierung von unbekannten Objekten lediglich als Ausblick gegeben werden kann. In Kapitel 4 wird auf die Probleme bei diesem Verfahren eingegangen.

2.3 Greifen

Das Berechnen von Griffen ist gleichzusetzen mit der Auswahl einer Menge von Oberflächenpunkten eines Objekts, welche für den Griff als Kontaktpunkte dienen sollen.

Es lässt sich entweder berechnen, welche Punkte optimal wären, wobei die möglichen Handkonfigurationen berücksichtigt werden müssen, oder es soll getestet werden, an welcher Stelle eine bestimmte Handkonfiguration ein Objekt berührt, um dann zu analysieren, ob diese Kontaktpunkte geeignet sind, um das Objekt stabil zu halten.

In diesem Abschnitt wird dargelegt, wie sich die Güte von Kontaktpunkten berechnen lässt und wie sich die Kontaktpunkte, die eine definierte Handkonfiguration mit einem Objekt haben wird, mittels Simulation bestimmen lassen. Letzteres wird auch Kollisionsdetektion genannt.

2.3.1 Optimale Griffe mittels „form-closure grasp“ Analyse

Einer der wichtigsten Aspekte des robotergestützten Greifens ist die Stabilität des Griffes. Beginnend bei Zwei-Finger-Greifern bis hin zur menschenähnlichen Fünf-Finger-Hand werden Verfahren benötigt, welche sichere Auskunft darüber liefern können, ob und wie das zu greifende Objekt stabil zu greifen ist.

Ein sicheres Indiz für die Robustheit eines Griffes ist die Eigenschaft „form-closure“. Ein Griff ist dann „form-closure“, wenn er ein Objekt - bei beliebiger von außen auf das Objekt einwirkender Kraft - halten kann.

Definition

Das zu greifende Objekt Ω sei repräsentiert durch N Oberflächenpunkte, mit $r_j = (x_j, y_j, z_j)^T,\ \ j = 1, \ldots, N$ und n_j als die Oberflächennormale an Punkt r_j.

Ein Griff G mit M Kontaktpunkten sei definiert durch $G = \{r_i | r_i \in \Omega,\ i = 1, \ldots, M\}$,

Im reibungslosen Fall liegen die Greifkräfte f_i in Richtung der Oberflächennormalen an den Kontaktpunkten. Die an r_i wirkende Kraft f_i und Drehmoment τ_i in Richtung des Massezentrums ist in diesem Fall gegeben durch

$$\begin{pmatrix} f_i \\ \tau_i \end{pmatrix} = \begin{pmatrix} f_i \\ r_i \times f_i \end{pmatrix} = \alpha_i \begin{pmatrix} n_i \\ r_i \times n_i \end{pmatrix} \tag{2.28}$$

mit α_i als nicht-negative Konstante, welche die Stärke der Greifkraft repräsentiert.

$$w_i = \begin{pmatrix} n_i \\ r_i \times n_i \end{pmatrix} \tag{2.29}$$

wird „primtive contact wrench“ genannt. $W = (w_1, w_2, \ldots, w_7) \in R^{6\times M}$ ist die korrespondierende „wrench matrix“.

Ein Griff sei dann „form-closure“, wenn es für beliebige auf das Objekt einwirkende externe Kraft $\underline{w}_{ext} \in R^6$ möglich ist, ein α mit allen $\alpha_i \geq 0$ zu finden, so dass

$$W\alpha + \underline{w}_{ext} = 0 \tag{2.30}$$

Qualitätstest

Gleichung 2.30 ist dann erfüllt, wenn der Ursprung O des „wrench space“ innerhalb der konvexen Hülle $H(W)$, der „primitive contact wrenches“, liegt. Die „primitive contact wrenches“ spannen dann den gesamten „wrench space“ auf.

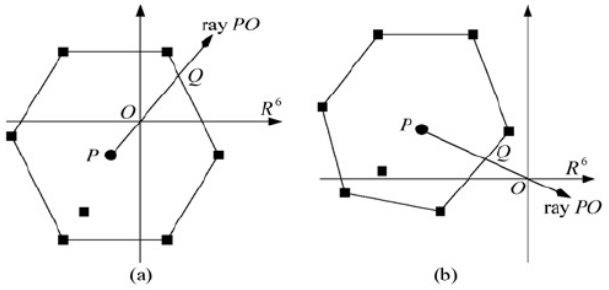


Abbildung 2.14: (a) - „form-closure grasp“, (b) - „non form-closure grasp“

Die Lösung dieses Problems ist komplex. In [12] wird deshalb ein Verfahren vorgestellt, welches die Dualität von konvexen Hüllen und konvexen Polyedern ausnutzt.

Um zu überprüfen, ob der Ursprung innerhalb $H(W)$ liegt, wird ein Punkt P gewählt, vom dem bekannt ist, dass er innerhalb der konvexen Hülle liegt und berechnet, an welchem Punkt Q der Strahl $\vec{PO}$ die konvexe Hülle schneidet.

Dies ist auch bekannt als das „rayshooting problem“. Der Punkt Q liegt auf einer Facette des konvexen Polyeders. Zu bestimmen, welche Facette dies ist, ist Teil des „rayshooting“ Problems.

Ein „form-closure“ Griff ist unter diesen Bedingungen äquivalent zu der Aussage, dass die Entfernung $||PQ||$ echt größer als die Entfernung $||PO||$ ist.

Das Finden des Schnittpunktes Q aus $H(W)$ mit $\vec{PO}$ lässt sich effektiv mithilfe der linearen Optimierung lösen [12]. Dazu wird zunächst die Facette der konvexen Hülle bestimmt, welche den Schnittpunkt beinhaltet. Ist dies bekannt, reduziert sich das Problem auf das Berechnen des Schnittpunktes einer Geraden mit einer Ebene.

Innerer Punkt der konvexen Hülle

Zur Anwendung des genannten Verfahrens, ist es zunächst notwendig, einen inneren Punkt der konvexen Hülle zu finden. Dieser lässt sich als Mittelwert aller „primitive contact wrenches“ festlegen.

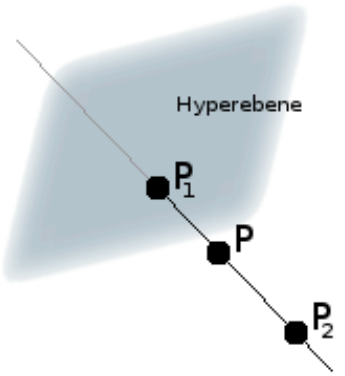


Abbildung 2.15: Eine Konvexkombination mit nichtnegativen Koeffizienten für P_1 und P_2 liegt nicht auf der Hyperebene S.

Jeder Punkt P, welcher in $H(W)$ liegt, lässt sich repräsentieren als

$$P = \sum_{i=1}^{N} \alpha_i w_i, \;\; \sum_{i=1}^{N} \alpha_i = 1, \;\; \alpha_i \geq 0, \;\; i = 1, 2, \ldots, N. \tag{2.31}$$

Die Facetten der konvexen Hülle $H(W)$ stellen fünf-dimensionale Hyperebenen dar. Sei V_1 die Menge aller w_i, die auf der Facette S liegen, V_2 die Menge aller w_i, die nicht auf S liegen.

Satz: ein Punkt P liegt in S. Nach Repräsentation (2.31) müssen alle Koeffizienten α_i in V_2 gleich 0 sein.

Beweis: angenommen P aus S beinhalte nichtnegative α_i in V_2. Seien k_1 und k_2 die Anzahl der Punkte in V_1 und V_2, $N = k_1 + k_2$ und die ersten k_1 „primitive contact wrenches“ gehören zu V_1, o. B. d. A. Der Punkt P kann dargestellt werden als

$$P = \sum_{i=1}^{k_1} \alpha_i w_i + \sum_{j=k_1+1}^{N} \alpha_j w_j \tag{2.32}$$

Sei

$$l_1 = \sum_{i=1}^{k_1} \alpha_i, \;\; l_2 = \sum_{j=k_1+1}^{N} \alpha_j, \;\; l_1 + l_2 = 1$$

Eingesetzt in (2.32) ergibt sich

$$P = l_1 P_1 + l_2 P_2 \tag{2.33}$$

mit den Punkten P_1 und P_2 als

$$P_1 = \sum_{i=1}^{k_1} \frac{\alpha_i}{l_1} w_i$$
$$P_2 = \sum_{j=k_1+1}^{N} \frac{\alpha_j}{l_2} w_j$$

Der Punkt P_1 ist eine Konvexkombination[6] der Punkte in Menge V_1 und liegt auf der Facette S. Der Punkt P ist eine entsprechende Kombination aus V_2 und liegt nicht auf S. Laut (2.33) ist P eine Konvexkombination aus P_1 und P_2. Da der Koeffizient l_2 nicht negativ ist, kann P nicht auf S liegen.

Liegt Punkt P also auf S, so muss l_2 gleich 0 sein, was impliziert, dass V_2 Koeffizienten α_i enthält, welche gleich 0 sind.

Theorem: Eine strikt positive Konvexkombination, mit allen $\alpha_i > 0$, der „primitive contact wrenches“ w_i ist ein innerer Punkt der konvexen Hülle $H(W)$.

Hat ein Punkt also folgende Form:

$$P = \sum_{i=1}^{N} \alpha_i w_i, \;\; \sum_{i=1}^{N} \alpha_i = 1, \;\; \alpha_i > 0, \;\; i = 1, 2, \ldots, N.$$

ist er ein innerer Punkt von $H(W)$.

Beweis: Nach dem Satz muss ein Punkt P, welcher auf einer Facette der konvexen Hülle liegt, mindestens einen Koeffizienten enthalten, der gleich 0 ist. Da kein a_i gleich 0 ist, muss der Punkt ein innerer Punkt von $H(W)$ sein.

Basierend auf dem Theorem lässt sich ein innerer Punkt P also wie folgt bestimmen:

$$P = \frac{1}{N} \sum_{i=1}^{N} w_i. \tag{2.34}$$

Dualität zwischen konvexer Hülle und konvexem Polyeder

Nachdem ein innerer Punkt der konvexen Hülle gefunden wurde, ist es nun nötig, den Schnittpunkt des Strahls, der von diesen Punkt und dem Ursprung gebildet wird, mit

[6] Eine Konvexkombination ist eine Linearkombination $x = \sum_{i=1}^{n} \alpha_i \cdot x_i$ von endlich vielen Elementen einer Menge M, mit $\alpha_i \geq 0$ und $\sum_{i=1}^{n} \alpha_i = 1$.

der konvexen Hülle zu finden. Dazu wird zunächst die Facette der konvexen Hülle herausgefunden, die diesen Schnittpunkt enthält. Der Schnittpunkt des Strahls mit der Facette ist dann der Schnittpunkt des Strahls mit der konvexen Hülle.

Das Problem, die Facette zu finden, ist, wie bereits erwähnt, eng verwandt mit dem sogenannten „rayshooting problem“.

Definition ray-shooting problem: Sei M eine Menge von Punkten aus R^d. Unter der Voraussetzung, dass die konvexe Hülle $H(M)$ den Ursprung enthält, ist der Schnittpunkt der konvexen Hülle mit einem Strahl zu finden, der vom Ursprung ausgeht.

Dieses Problem lässt sich aufgrund der Dualität von konvexen Hüllen und konvexen Polyedern in ein lineares Optimierungsproblem umwandeln.

Enthält die konvexe Hülle $H(M)$ den Ursprung, so lässt sie sich laut [18] in einen konvexen Polyeder $CT(M)$ umformen, welcher folgendermassen definiert wird:

$$v^T x \leq 1, \forall v \in M$$

Die Rücktransformation ist durch eine duale Transformation T möglich, welche alle Punkte b aus R^d auf die Hyperebene

$$\sum_{i=1}^{d} b_i x_i = 1$$

abbildet.

Wie bereits erwähnt, soll dieses Verfahren prüfen, ob der Ursprung in der konvexen Hülle $H(W)$ liegt. Da dies verständlicherweise zu diesem Zeitpunkt noch unklar ist, die Nutzung der Dualität dies aber voraussetzt, wird auf die konvexe Hülle zunächst eine Translation angewandt. Sie wird um $-P$ verschoben. Der innere Punkt P wird also der neue Ursprung, wodurch dieser innerhalb der konvexen Hülle liegt.

Nach dieser Translation ist die konvexe Hülle dual zum konvexen Polyeder

$$(w_i - P)^T x \leq 1, \;\; i = 1, 2, \ldots, N. \tag{2.35}$$

Sei t der Richtungsvektor des Strahls PO, so ist das „rayshooting problem“ äquivalent zum Problem, folgende Zielfunktion zu maximieren:

$$z = t^T x \tag{2.36}$$

unter den Nebenbedingungen in (2.35).

Sei die optimale Lösung dieses linearen Optimierungsproblems $E = (e_1, e_2, \ldots, e_6)$, so ist laut Transformation T die Facette von $H(W)$, welche vom Strahl PO geschnitten wird

$$\sum_{i=1}^{6} e_i x_i = 1$$

und der Schnittpunkt der konvexen Hülle mit dem Strahl PO entspricht dem Schnittpunkt der Facette mit dem Strahl PO.

Der Strahl $\vec{PO}$ lässt sich darstellen als $\vec{PO} = \lambda P$ mit $P = (p_1, p_2, \ldots, p_6)$, wodurch sich λ bestimmen lässt als

$$\lambda = \frac{1}{\sum_{i=1}^{6} p_i e_i}$$

und liefert den Schnittpunkt $Q = (q_1, q_2, \ldots, q_6)$ mit $q_i = \lambda p_i$.

Algorithmus zur Überprüfung auf form-closure

In *Abbildung 2.16* wird ein in [12] vorgestellter Algorithmus beschrieben, welcher an den hier betrachteten reibungslosen Fall angepasst wurde und die oben genannten Bedingungen für „form-closure“ Griffe überprüfen kann.

Algorithmus: form-closure Analyse

1. Bestimme alle *primitive contact wrenches* w_i nach (2.29).
2. Bestimme nach (2.34) einen Punkt P, welcher innerhalb $H(W)$ liegt.
3. Finde die optimale Lösung E des linearen Optimierungsproblems, welches die Funktion z in (2.36) maximiert, unter den Nebenbedingungen in (2.35).

   Die zum Punkt E korrispondierende Hyperebene $T(E)$

   $$\sum_{i=1}^{6} e_i x_i = 1$$

   ist die Facette von $H(W)$, welche vom Strahl PO geschnitten wird.
4. Bestimme den Schnittpunkt Q des Strahls PO mit $T(E)$, welcher dem Schnittpunkt mit $H(W)$ entspricht.
5. Ist die Distanz zwischen P und Q größer, als die Distanz zwischen P und O, so ist der Griff *form-closure*, sonst ist er es nicht.

Abbildung 2.16: Algorithmus - form-closure Analyse

2.3.2 Suchen von „form-closure“ Griffen

Mithilfe des Algorithmus 2.16 kann überprüft werden, ob ein Griff die Kriterien des „form-closure“ erfüllt. Um einen solchen Griff zu finden, müssen die Kontaktpunkte also so gewählt werden, dass sie den Ursprung des „wrench space“ enthalten.

Die Oberfläche liegt als eine endliche Menge von Punkten vor. Existiert ein „form-closure“ Griff für ein beliebiges Objekt, so kann dieser bei sieben Kontaktpunkten

auf jeden Fall gefunden werden. Eine Möglichkeit wäre es, alle Kombinationen aus sieben Kontaktpunkten auf „form-closure“ zu überprüfen.

Ein performanter Ansatz ist in [13] zu finden. Bei der Definition der Oberflächenpunkte Ω wird dazu eine weitere Eigenschaft gefordert: jeder Punkt der Oberfläche ist mit vier adjazenten Punkten der Oberfläche verbunden.

Die Suche nach einem „form-closure“ Griff beginnt mit sieben zufällig gewählten Kontaktpunkten. Diese werden entlang der Oberfläche verschoben, mit dem Ziel, die konvexe Hülle mit jedem Schritt näher an den Ursprung zu verschieben.

Um das Problem zu vereinfachen, wird angenommen, dass sich in jeder Iteration nur jeweils ein Finger verschieben lässt. Dies lässt bei sieben Kontaktpunkten also je Schritt 28 mögliche Veränderungen zu. Um die konvexe Hülle in Richtung Ursprung O zu verschieben, muss der Abstand zwischen Q[7] und O verringert werden. $\|QO\|$ kann also als Heuristik dienen. Diejenige der 28 Möglichkeiten, welche die geringste Distanz $\|QO\|$ liefert, wird als neue Kontaktpunktmenge genommen.

Dabei spielt eine Beobachtung des „wrench space“ eine wichtige Rolle. Die „primitive contact wrenches“ eines „form-closure“ Grasp können durch eine Hyperebene separiert werden, welche durch den Ursprung O verläuft.

Dies bedeutet, dass bei der Auswahl eines neuen Kontaktpunktes all jene vernachlässigt werden können, welche dazu führen, dass alle zu diesen Punkten korrespondierenden „primitive contact wrenches“ auf einer Seite einer solchen Hyperebene liegen.

Sei diejenige Facette, welche den Punkt Q enthält, E. Als separierende Hyperebene $Y : \sum_{i=1}^{6} e_i x_i = 0$ wird diejenige gewählt, welche parallel zu E liegt und durch den Ursprung O verläuft.

Dies unterteilt die zu den möglichen Kontaktpunkten korrespondierenden „primitive contact wrenches“ in zwei Gruppen: Y^+ und Y^-. Y^+ ist dabei der Halbraum $\sum_{i=1}^{6} e_i x_i \geq 0$ und Y^- der Halbraum $\sum_{i=1}^{6} e_i x_i < 0$.

$\Omega(Y^-)$ und $\Omega(Y^+)$ seien die zu den „primitive contact wrenches“ jeweils korrespondierenden Punktmengen der Halbräume. Die Auswahl eines neuen Kontaktpunktes kann damit in sechs Teilprobleme zerlegt werden:

- ein Punkt aus $\Omega(Y^-)$, sechs Punkte aus $\Omega(Y^+)$
- zwei Punkte aus $\Omega(Y^-)$, fünf Punkte aus $\Omega(Y^+)$
- drei Punkte aus $\Omega(Y^-)$, vier Punkte aus $\Omega(Y^+)$
- vier Punkte aus $\Omega(Y^-)$, drei Punkte aus $\Omega(Y^+)$
- fünf Punkte aus $\Omega(Y^-)$, zwei Punkte aus $\Omega(Y^+)$
- sechs Punkte aus $\Omega(Y^-)$, ein Punkt aus $\Omega(Y^+)$

[7] Siehe Kapitel 2.3.1: Q ist ein Punkt auf der konvexen Hülle $H(W)$ und der Schnittpunkt zwischen dem Strahl $\vec{PO}$ und einer Facette der konvexen Hülle. P ist ein Punkt innerhalb $H(W)$.

Diese sechs Teilprobleme werden in einem unregelmäßigem Baum repräsentiert, dessen Wurzel die ursprüngliche Punktauswahl darstellt. Jeder Knoten repräsentiert eine mögliche Konfiguration des „wrench space“. Von den ersten sechs Knoten wird einer ausgewählt und in einer lokalen Suche, ausgehend von diesem Knoten, nach einer Lösung gesucht. Dabei können zwei Fälle eintreten:

- es wird ein „form-closure“ Griff gefunden, in diesem Fall terminiert die Suche und der Griff ist gefunden
- es wird ein lokales Minimum gefunden, in diesem Fall wird die separierende Hyperebene verwendet, um das Problem in drei bis vier weitere Teilprobleme zu unterteilen, dessen Wurzel der aktuelle Knoten ist

Abbildung 2.17: Der Suchbaum

Der Baum wird rekursiv durchsucht, bis ein „form-closure“ Griff gefunden oder er komplett durchlaufen wurde. In diesem Fall existiert kein „form-closure“ Griff für das Objekt.

Es besteht eine höhere Chance auf „form-closure“, wenn die „primitive contact wrenches“ gleichmäßig im Raum verteilt sind. Deshalb kann folgende heuristische Funktion verwendet werden, um den Knoten zu bestimmen, welcher zuerst durchsucht werden soll:

$$f\left(< a_1 \;\; a_2 \;\; \ldots \;\; a_S >\right) = \sum_{i=1}^{S} \left(\frac{7}{S} - a_i\right)^2 \tag{2.37}$$

Hierbei entspricht S der Anzahl der Teilmengen aus Ω des aktuellen Knotens und a_i der Anzahl der Punkte aus der Teilmenge i. Der Knoten, für den f den geringsten Wert liefert, wird zuerst expandiert.

Für den zweiten Knoten von Links aus *Abbildung 2.17* wäre a_1 beispielsweise 2 und $a_2 = 5$. Wird nun für eine Konfiguration aus $K(2,5)$ ein lokales Minimum erreicht, werden die Punkte dieser Konfiguration erneut durch eine Hyperebene geteilt und es werden entsprechende neue Knoten an den Baum angehängt.

2.3.3 Ermitteln von Kontaktpunkten

Das Berechnen von Kontaktpunkten wird auch als Kollisionsdetektion bezeichnet und ist dann nötig, wenn berechnet werden soll, an welchen Oberflächenpunkten eine Roboterhand ein Objekt bei einer bestimmten Konfiguration berührt.

An dieses Problem wird in dieser Arbeit nicht analytisch herangegangen, sondern eine Simulation verwendet. Die Schnittpunkte verschiedener Primitive im drei-dimensionalen Raum zu berechnen ist eine Fähigkeit, welche viele 3D-Bibliotheken bereits beinhalten.

Deshalb liegt es nahe, die verwendete Hand und das zu greifende Objekt mittels einer solchen 3D-Bibliothek zu modellieren und die vorhandenen Methoden zu verwenden.

Eine Möglichkeit, die Kollision zweier Objekte zu testen, basiert auf einem Theorem der algorithmischen Geometrie:

Zwei konvexe Polyeder, welche sich nicht schneiden, können durch eine Ebene getrennt werden, welche entweder parallel zu einer Facette eines der Polyeder ist oder jeweils eine Kante jedes Polyeders enthält.

Der Artikel [4] zeigt, wie diese Bedingung ausgenutzt werden kann, um mithilfe gerichteter umschliessender Quader das Problem der Kollision in eine Baumstruktur zu bringen, welche die Überprüfung beschleunigt.

Das Objekt wird dabei durch einen Baum präsentiert, bei dem jede Ebene eine feinere Repräsentation des Objekts darstellt. Die Knoten dieses Baums sind umschliessende Quader, wobei die Wurzel ein einzelner Quader ist, welcher das gesamte Objekt umschliesst. Die Blätter entsprechen den Dreiecken der Oberfläche, stellen also die exakte Geometrie des Objekts dar.

Soll nun geprüft werden, ob sich zwei Objekte schneiden, kann der Baum von der Wurzel an durchlaufen werden.

Diese Methode wurde motiviert durch die Entwicklung von 3D-Bibliotheken für sich bewegende Objekte. In einer Szene mit sich bewegenden Objekten kommt es wesentlich öfter vor, dass diese sich nicht schneiden, als dass sie sich schneiden, weshalb die Kollisionsdetektion daraufhin optimiert wurde, möglichst schnell zu erkennen, ob zwei Objekte nicht kollidieren.

3 Implementierung

Die im Kapitel „Grundlagen“ vorgestellten Verfahren sind teilweise nicht direkt auf die *TAMS* Roboter-Plattform übertragbar gewesen. Die dort Verwendung findende BarrettHand besitzt nicht die nötigen Freiheitsgrade, um eine optimale Griffplanung, wie es die „form-closure“ Analyse in ihren Grundlagen bietet, durchzuführen.

In der folgenden Vorstellung der Implementierung und Integration dieser Verfahren in ein Programm, das Griffplanung für die Roboter-Plattform ermöglicht, werden die nötigen Anpassungen dargelegt und begründet.

3.1 Architektur

Für Java-basierte Programme bietet der Roboter eine Serverarchitektur an, die es erlaubt, über sogenannte Roblets Programmteile mittels einer WLAN gestützen Netzwerkverbindung zur Laufzeit an den Roboter zu übertragen und dort ausführen zu lassen. Ergebnisse können abgefragt werden. Dadurch wird es ermöglicht, beispielsweise besonders rechenintensive Prozesse auf externer Hardware ausführen zu lassen.

Da die für die Hand- und Armsteuerung zuständigen Bibliotheken allerdings in Hardware nahem C++-Code implementiert wurden und noch keine geeignete Schnittstelle zu der Roblet-Architektur geschaffen werden konnte, wurde die Implementierung der in dieser Arbeit vorgestellten Verfahren in C++ vorgenommen und direkt auf dem in den Roboter integrierten PC unter Linux ausgeführt.

Die Steuerung des Arms und der Hand erfolgt dabei über eine von Tim Baier entwickelte Erweiterung der RCCL-Bibliothek, die an den TAMS Service Roboter angepasst wurde und auch Zugriff auf die Bibliotheken für die Hand- und Kamerasteuerung ermöglicht.

3.2 Versuchsaufbau

Der Roboter wurde vor einem Tisch aufgestellt, dessen Höhe bekannt war. Dabei wurde er relativ nahe an den Tisch herangefahren, um möglichst viel der Tischfläche in den Arbeitsbereichs des Arms zu bekommen. Eine exakte Ausrichtung auf den Tisch war dabei nicht nötig, da die Lokalisierung der Objekte in Bezug auf den Roboter stattfand.

Um die Segmentierung zu erleichtern, wurde der Tisch mit einem schwarzen Tuch abgedeckt, auf welchem die zu greifenden Objekte platziert wurden.

3.3 Programmablauf

Das im Laufe dieser Arbeit entstandene Programm implementiert eine Vielzahl der vorgestellten Verfahren. Es wurden die nötigen Anpassungen an die Roboter-Plattform vorgenommen, und das Programm ermöglicht das Planen von Griffen für diverse Objekte. Es orientiert sich an dem in *Abbildung 3.1* dargestellten Ablaufplan.

Zunächst wird eine Übersicht der Szene aufgenommen. Dazu fährt der Arm in eine Position, welche die Szene in einer Schrägansicht zeigt.

In dieser Aufnahme werden die dargestellten Objekte separiert und das Objekt von Interesse auf Grundlage von einigen Kriterien automatisch ausgewählt. Der Masseschwerpunkt dieses Objekts wird in Weltkoordinaten transformiert und die Kamera wird über diesem Punkt positioniert.

Da das Ergebnis der Rückprojektion in der Regel nicht genau genug ist, wird die genaue Position mittels „visual servoing“ ermittelt. Die gewünschte Genauigkeit beträgt 1 mm, kann aber auf beliebige Genauigkeit angepasst werden.

Nachdem das Objekt exakt lokalisiert wurde, wird versucht, es mittels der PCA zu klassifizieren. Schlägt dies fehl, so wird das Objekt durch eine Analyse der Umrisse klassifiziert. Solcherart klassifizierte Objekte werden gesondert markiert, da sie eine spezielle Form der Griffanalyse bedürfen. Da die verwendeten Methoden allerdings nicht geeignet waren, unbekannte Objekte mit hinreichender Güte zu klassifizieren, kann die derzeitige Implementierung unbekannte Objekte nicht stabil greifen.

Die Klassifikation liefert eine interne 3D Repräsentation der Objekte. Anhand dieser wird der geeignetste Griff bestimmt. Handelt es sich um ein mittels PCA klassifiziertes Objekt, beschränkt sich die Analyse auf einige wenige Griffe. Bei unbekannten Objekten werden alle für die Form des Objekts verfügbaren Griffe geprüft.

Der tauglichste Griff kann ausgeführt werden. Liegt kein geeigneter Griff vor, so ist das Objekt auf Grundlage der „form-closure“ Analyse nicht stabil greifbar.

3.4 Sehen

3.4.1 Bildaufnahme

Der Roboter stellt drei verschiedene Kamerasysteme zur Verfügung, einen Stereo-Kamera-Aufbau, eine Omnivisionskamera und eine Handkamera.

Für diese Arbeit fand ausschließlich die Handkamera Anwendung. Diese bietet den Vorteil, dass sie sich frei über der Szene platzieren lässt. Für einige Aufgaben, wie die Rückprojektion, wäre möglicherweise der Stereo-Kamera-Aufbau besser geeignet

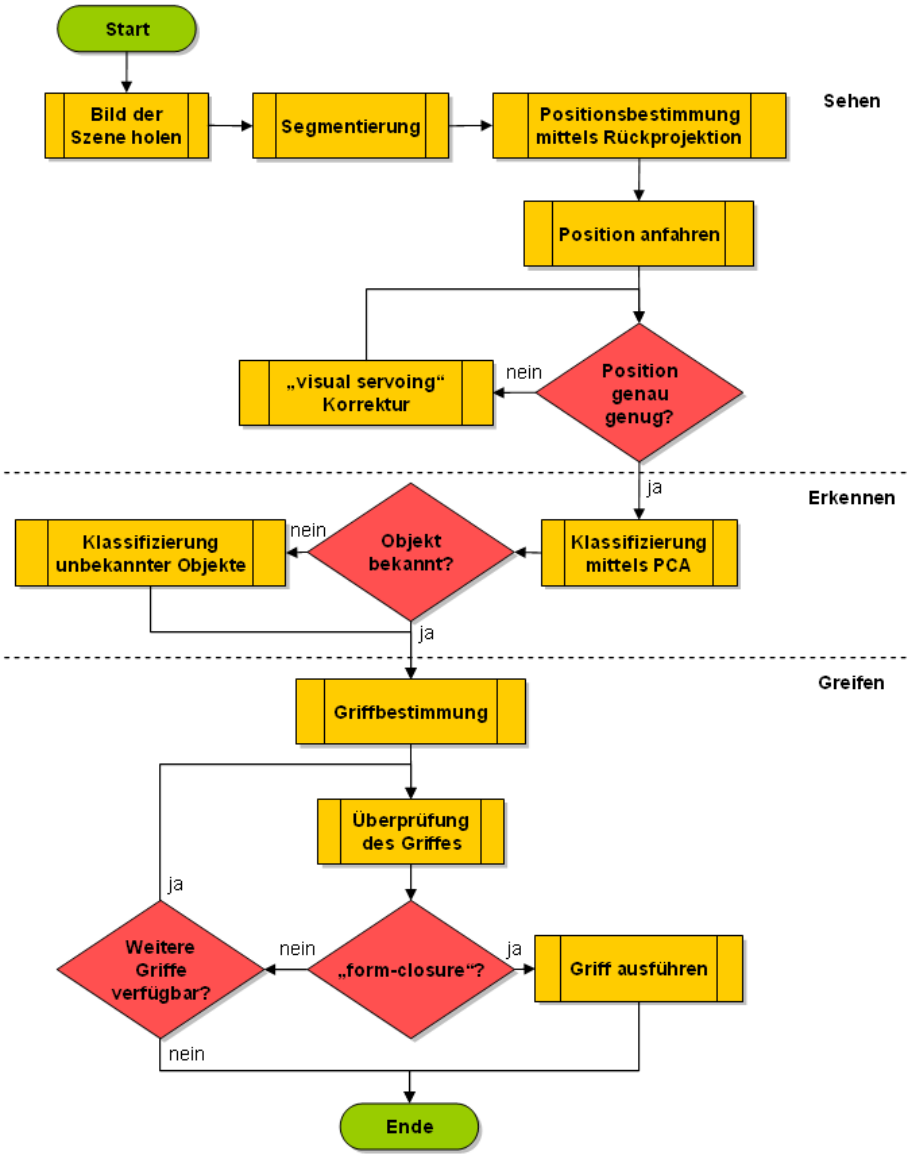


Abbildung 3.1: Programmablaufplan der Implementation

gewesen. Es wurde bewusst auf deren Einsatz verzichtet, um zu zeigen, dass eine visuell geführte Griffplanung mithilfe nur einer einzigen Kamera möglich ist.

Wie in vorherigen Kapiteln bereits erwähnt, ist der erste Schritt in der Griffplanung die Positionsbestimmung der Objekte. Zu diesem Zweck muss eine Aufnahme der Szene gemacht werden. Da bekannt ist, dass sich die Objekte unmittelbar vor dem Roboter befinden, erübrigt sich das automatisierte Suchen der Region von Interesse. Wie die Kamera zur Szene ausgerichtet werden soll, bedarf trotzdem einiger Vorüberlegungen.

Das Bild sollte keine Selbstverdeckung beinhalten. Das heißt, es sollen keine Teile des Roboters im Bild zu sehen sein, da diese den Klassifikationsprozess stören könnten.

Weiterhin ist es für die Rückprojektion von Vorteil, wenn der Winkel der Aufnahme nicht zu flach ist, die Randpixel also nicht Objekte in sehr großer Entfernung zeigen.

Außerdem sollte sich der Arm, an dem die Kamera befestigt ist, nach der Bildaufnahme in einer Position befinden, aus der es möglich ist, viele Konfigurationen zu erreichen.

Diese Überlegungen führten zu einer Kameraposition, welche sich „links über der Szene“ befindet. Die Kamera befindet sich an Punkt $P_{overview} = (800, 200, 900)$[1], wird 45° um z gedreht und 30° um y geneigt. Aus dieser Perspektive befindet sich in den unteren zwei Dritteln des Bildes mit sehr hoher Wahrscheinlichkeit die Tischoberfläche. Im oberen Drittel sind Teile der Laborumgebung zu sehen.

Es wären auch viele andere Positionen denkbar, aber diese hat sich in Versuchen als tauglich erwiesen.

Aus dieser Position wird ein Bild aufgenommen, das für die Objektlokalisierung Verwendung findet. Dabei wird die volle PAL Auflösung der Kamera verwendet, welche ein 752×582 Pixel großes Farbbild liefert.

3.4.2 Segmentierung

Auf dem von der Bildaufnahme gelieferten Farbbild werden die Objekte von Interesse gesucht. Dazu müssen die sich auf dem Bild befindlichen Objekte vom Hintergrund freigestellt werden.

Hierfür wurde ein Labeling-Algorithmus verwendet, der auf Binärbildern basiert (s. Abb. 2.5). Das Farbbild muss zunächst also in ein Binärbild gewandelt werden. Dazu wird der „optimale Schwellwert“ (s. Kapitel 2.1.1) auf das Bild angewendet. Dies liefert für gleichmäßig ausgeleuchtete Aufnahmen sehr brauchbare Ergebnisse. Schwierigkeiten machen bei diesem Verfahren Reflexionen auf stark spiegelnden Oberflächen.

[1] Der Roboter verwendet ein rechtshändiges Koordinatensystem, dessen positive x-Achse in Fahrtrichtung zeigt.

Als Grundlage für die Positionsbestimmung der Objekte dient deren Masseschwerpunkt. Dieser berechnet sich aus dem Mittelwert über die Gesamtzahl der zu einem Objekt gehörenden Pixel.

$$P_{centerofmass} = \frac{\sum P_n}{N} \tag{3.1}$$

Reflexionen spiegeln sich in einem diskreten Bild mit sehr hoher Intensität wieder, diese liegt häufig am Maximum des zulässigen Wertebereichs. Dadurch kommt es in diesen Objekten zu „Löchern“. Diese Löcher können unter Umständen bei der Berechnung des Schwerpunkts zu starken Abweichungen vom tatsächlichen Schwerpunkt des Objekts führen.

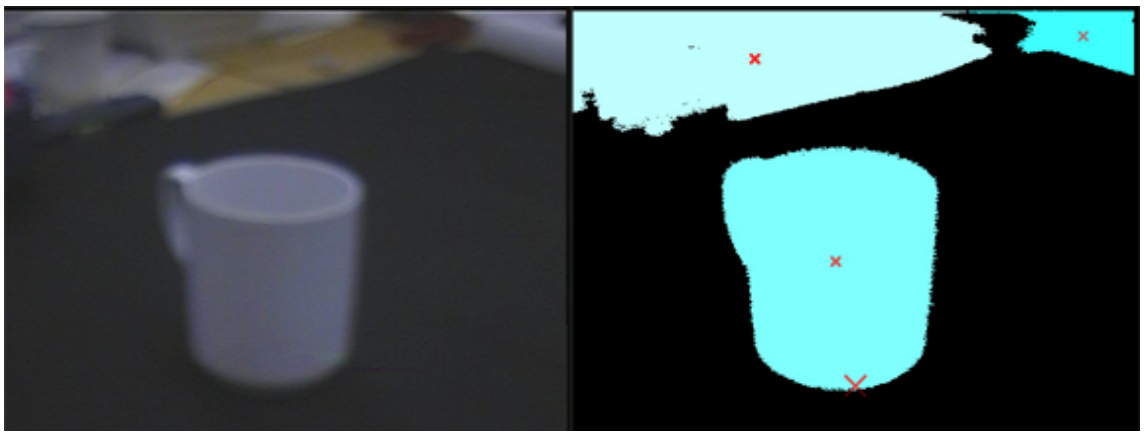


Abbildung 3.2: Ausschnitt aus einer Überblicksszene. Rechts sind die Masseschwerpunkte gekennzeichnet. Das „große X“ markiert den Mittelpunkt des Gesamtbildes.

Die in dem Binärbild freigestellten „Objekte“ sollen zunächst noch Pixelcluster genannt werden. Es ist noch nicht bekannt, ob die jeweiligen Pixelcluster Objekte repräsentieren, da noch keine Klassifikation stattfand.

Abhängig von der Ausleuchtung der Szene wird eine Vielzahl an Pixelclustern freigestellt. Aus diesen müssen nun jene ausgewählt werden, welche für den Greifvorgang von Interesse sind, also Objekte repräsentieren können.

Viele dieser Pixelcluster entsprechen lediglich kleineren Reflexionen, Staubpartikeln oder Bildstörungen.

In einem ersten Schritt können also jene Pixelcluster ausgeschlossen werden, die zu „klein“ sind, um greifbare Objekte darstellen zu können. Die Größe eines Pixelclusters ist die Anzahl der zu diesem Cluster gehörenden Pixel. Dies geschieht wiederum durch einen Schwellwert. Fällt die Größe eines Pixelclusters unter diesen Wert, wird er aus der weiteren Betrachtung ausgeschlossen.

Wie dieser Schwellwert bestimmt wird, ist variabel. Möglich ist eine Mittel- oder Medianwertbildung über die Masse aller Objekte. Um das Labeling zu beschleunigen, wurde willkürlich ein Wert festgelegt, welcher durch mehrere Testläufe auf

Tauglichkeit geprüft wurde. Solange dieser Wert relativ gering ist, spielt er für die folgende Klassifikation keine tragende Rolle und es bleibt lediglich beim Effekt der Beschleunigung.

Es hat sich herausgestellt, dass die Objekte von Interesse häufig auch die größten in einer Szene zu findenden Objekte sind. Das Programm folgt deshalb dem Weg des geringsten Widerstandes und betrachtet zunächst den größten gefundenen Pixelcluster genauer. Konnte das Objekt nicht klassifiziert werden, so stellt sich dies als Fehlentscheidung heraus und es können die übrigen Objekte einer genaueren Betrachtung unterzogen werden[2].

Dies löst zugleich das Problem, wenn sich mehrere interessante Objekte in der Szene befinden. Hier lässt sich ansonsten keine sinnvolle automatische Entscheidung treffen, und der Benutzer müsste gefragt werden.

3.4.3 Objektlokalisierung

Die Objektposition zu kennen, ist eine Bedingung für zwei weitere Schritte: die Klassifizierung und die Ausführung des Griffes.

Die Klassifizierung benötigt Bilder, die aus einer definierten Position relativ zum Objekt aufgenommen wurden. Für die Berechnung des Griffes ist die Position nicht nötig[3], allerdings für die Ausführung des Griffes, damit die Hand entsprechend platziert werden kann.

Für die Lokalisierung des Objekts wurde eine Rückprojektion verwendet. Diese transformiert Bildkoordinaten in Weltkoordinaten zurück.

Da die Kamera am Manipulatorendpunkt des Roboterarms befestigt ist, kann die Matrix für die Transformation in Kamerakoordinaten direkt abgefragt werden. Wie im Grundlagen Kapitel 2.1.2 bereits erwähnt, findet die Rückprojektion auf Grundlage des Lochkameramodells statt.

Kamera-Kalibrierung

Um eine möglichst präzise Rückprojektion zu ermöglichen, ist es nötig, einige Parameter der Kamera zu kennen. Dies ist die Fokallänge und die Größe des CCD. Zudem muss die Kamera kalibriert werden, um Verzerrungen ausgleichen zu können.

Bei der Rückprojektion mit nur einer Kamera ergibt sich ein fundamentales Problem: da zu diesem Zeitpunkt keine Objektklassifikation stattfindet, ist nicht bekannt, wie die Ausmaße der betrachteten Objekte beschaffen sind.

[2]Dies ist allerdings in keinem der Testläufe nötig gewesen. Die für die Bildaufnahme gewählte Position sorgt dafür, dass in den meisten Fällen nur der interessante Ausschnitt der Welt aufgenommen wird.

[3]Bei der Auswahl eines geeigneten Griffes spielt die Position des Objekts grundsätzlich eine Rolle, da es an einer Stelle liegen könnte, welche der Arm nicht erreichen könnte. Da in dieser Arbeit allerdings eine Handkamera verwendet wurde, welche bereits in der Vorphase der Klassifikation über dem Objekt platziert werden muss, ist davon auszugehen, dass die Position auch zu erreichen ist.

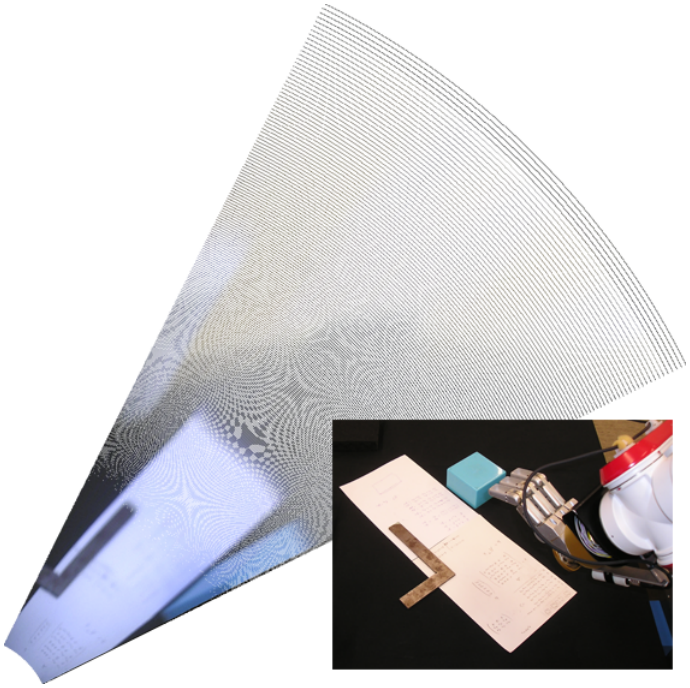


Abbildung 3.3: Die Rückprojektion wurde hier auf alle Pixel des Eingangsbildes angewendet (fehlt in der Darstellung) und deren x- und y-Koordinate in ein neues Bild eingefügt, um die Szene unverzerrt in einer Draufsicht darzustellen.

Wie in Kapitel 2.1.2 erwähnt, ist es für die Rückprojektion notwendig, eine Komponente der Weltkoordinaten bekannt zu machen, damit die Rücktransformation eindeutig wird. Da hier nur eine Kamera verwendet wird, muss diese Komponente zur Laufzeit bereits bekannt sein. Dazu ist die Höhe der Auflage, auf dem die Objekte stehen, als Parameter z mit in die Berechnung eingeflossen.

Es ist also die Höhenangabe (z-Komponente) der Objekte, welche im Vorfeld bekannt ist. Dadurch ist es ohne zusätzliches Wissen über die Objekte aber auch nur möglich, alle rücktransformierten Punkte auf diese z-Komponente festzusetzen.

Die Objekte besitzen allerdings selbst eine Höhe. Der Masseschwerpunkt der Objekte kann deshalb über dieser z-Komponente liegen. Dies führt dazu, dass die berechneten Objektkoordinaten nicht mit den tatsächlichen Koordinaten übereinstimmen. Sie werden einige Zentimeter hinter das Objekt projiziert.

Dies ist nur dann nicht mehr der Fall, wenn sich die Kamera direkt über dem Objekt befindet und der Masseschwerpunkt in den Bildaufnahmen mit dem realen Masseschwerpunkt übereinstimmt. Um diese Position erreichen zu können, wäre es allerdings nötig, die genaue Position des Objekts zu kennen.

Das Problem lässt sich also mit Rückprojektion allein nicht lösen. Darum wurde sie lediglich für eine auf wenige Zentimeter genaue Lokalisierung verwendet und die exakte Positionsbestimmung wurde über „visual servoing“ vorgenommen. Daher war es auch nicht nötig, die Kamera zu kalibrieren. Für die Feinpositionierung mittels „visual servoing“ muss lediglich gewährleistet werden, dass sich das Objekt im Bild befindet und dies konnte über Rückprojektion mittels einer unkalibrierten Kamera

erreicht werden.

Für die hier verwendete Rückprojektion war es allerdings nötig, die Öffnungswinkel der Kamera zu kennen. Diese wurden experimentell bestimmt. Anhang A.1 erläutert, wie der vertikale und horizontale Öffnungswinkel, γ_v bzw. γ_h, der Kamera berechnet wurden.

Mithilfe dieser beiden Winkel und des Wissens um die Bildgröße ($B_w \times B_h$) ist die Rückprojektion eines Bildpunktes $P_I = (x_I, y_I)$ in Weltkoordinaten $P_W = (x_W, y_W, z_W)$ möglich.

Für die Transformationen wurde die Implementierung der RCCL-Bibliothek verwendet [14]. Besonderes Augenmerk gilt der Transformationsmatrix:

$$\widehat{T} = \begin{pmatrix} n_1 & a_1 & o_1 & p_1 \\ n_2 & a_2 & o_2 & p_2 \\ n_3 & a_3 & o_3 & p_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

Die Vektoren $\vec{n}$, $\vec{a}$ und $\vec{o}$ spannen das lokale Koordinatensystem auf, während $\vec{p}$ die Translation angibt. Die homogenen Komponenten der Vektoren werden implizit wie in 3.2 zu sehen angenommen und sind nicht veränderbar.

Auf die Kamera angewendet, bedeutet dies: $\vec{n}$ entspricht dem Normalen-Vektor, also der in Blickrichtung gerichteten Komponente des Kamerakoordinatensystems, $\vec{p}$ der Kameraposition.

Da mit homogenen Koordinaten gerechnet wird, hat jeder Vektor vier Komponenten. Neben den Positionskomponenten x, y und z noch die homogene Komponente w. Für die folgende Berechnung der Weltkoordinaten werden die Pseudofunktionen normal($\widehat{T}$) und position($\widehat{T}$) eingeführt, die den Normalen-Vektor bzw. Positions-Vektor einer Transformationsmatrix zurückgeben, sowie $rot^{4\times4}(\alpha, A)$, welche die Rotationsmatrix für den Winkel α um die Achse A liefert.

$$\phi_h = \frac{\gamma_h}{B_w} \quad , \quad \phi_v = \frac{\gamma_v}{B_h}$$

$$x_\delta = x_I - \frac{B_w}{2} \quad , \quad y_\delta = y_I - \frac{B_h}{2}$$

$$\theta_{\delta h} = \phi_h \cdot x_\delta \quad , \quad \theta_{\delta v} = \phi_v \cdot y_\delta$$

$$\widehat{T}_R = rot^{4\times4}(\theta_{\delta v}, \underline{x}) \cdot rot^{4\times4}(\theta_{\delta h}, \underline{y}) \cdot \widehat{T}_{cam}$$

$$\vec{n} = \text{normal}(\widehat{T}_R) \quad , \quad \vec{p} = \text{position}(\widehat{T}_R)$$

$$\lambda = \frac{z - z_{\vec{p}}}{z_{\vec{n}}}$$

$$\begin{aligned} x_W &= x_{\vec{p}} + \lambda \cdot x_{\vec{n}} \\ y_W &= y_{\vec{p}} + \lambda \cdot y_{\vec{n}} \\ z_W &= z \end{aligned}$$

Es wird für einen beliebigen Punkt P_I der horizontale und vertikale Abstand zum Bildmittelpunkt berechnet. Dieser Pixelabstand liefert die Winkelabweichung in die zuvor beschriebenen Richtungen.

Das Kamerakoordinatensystem wird um diese Winkel rotiert und anschliessend der Schnittpunkt zwischen der Kameranormalen und der Ebene $E : \underline{x} \times \underline{y} \cdot \vec{z} = z$ berechnet. Dieser Schnittpunkt entspricht der zum Punkt P_I korrespondierenden Weltkoordinate P_W.

3.4.4 „visual servoing“

Da sich die Rückprojektion allein für die Objektlokalisierung nicht als ausreichend genau genug herausgestellt hat, wurde sie mit „visual servoing“ für die exakte Objektlokalisierung kombiniert.

Die Position des Manipulatorendpunktes kann auf den tausendstel Millimeter genau abgefragt werden. Um die Objektposition zu bestimmen, wird dieser in den Mittelpunkt der Kamera gelegt und diese über dem Masseschwerpunkt des Objekts platziert.

Dazu wird die Kamera in eine Position gebracht, die lotrecht zur Objektebene steht. Die Kamera schaut also direkt von oben auf die Szene. Der Soll-Zustand ist dann erreicht, wenn der Abstand zwischen Bildzentrum und Masseschwerpunkt des Objekts kleiner gleich ϵ ist. ϵ kann dabei beliebig gewählt werden, für die Testläufe kam in den meisten Fällen ein $\epsilon = 1mm$ zur Anwendung.

Das Ergebnis der Rückprojektion ist genau genug, so dass das Objekt stets im Bild zu sehen ist, wenn die Kamera einen bestimmten Abstand d_{cam} über der Objektebene wahrt. Dieser Abstand wurde so gewählt, dass er mit den Abstand, der bei den Aufnahmen zur Bestimmung der Öffnungswinkel verwendet wurde (s. Anhang A.1), übereinstimmt.

Für die „visual servoing“ Regelung muss eine Funktion geschaffen werden, die bestimmt, wie stark bei Abweichung vom Soll-Zustand geregelt werden soll. Hierbei wurde ein stark gedämpfter Regler verwendet. Mit jeder Iteration nimmt die Stärke der Regelung reziprok ab und nähert sich 0. Dadurch wird verhindert, dass sich der Regler aufschaukelt oder um einen bestimmten Punkt pulsiert.

Durch die spezielle Wahl des Abstands der Kamera zur Objektebene kann der Abstand D zwischen Bildzentrum und Massezentrum direkt aus dem Pixelabstand ($p_{\delta v}$, $p_{\delta h}$) abgelesen werden. Ein Pixel im Bild entspricht dem Abstand d_h in horizontaler Richtung in Weltkoordinaten, d_v in vertikaler Richtung.

$$D = \sqrt{(p_{\delta v} \cdot d_v)^2 + (p_{\delta h} \cdot d_h)^2}$$

Zur Regelung wird der Manipulatorendpunkt im Schritt i um $\frac{D}{i}$ in Richtung Masseschwerpunkt des Objekts bewegt.

Es liegt an der unkalibrierten Kamera, dass die genaue Position nicht in Schritt 1 erreicht wird. Je weiter der Masseschwerpunkt vom Bildzentrum entfernt ist, desto stärker wirkt sich die Radialverzerrung der Linse auf das Ergebnis aus.

In Testläufen hat sich gezeigt, dass der Regler bei einer Wahl von $\epsilon = 1mm$ nur vier bis sieben Schritte benötigt, um die gewünschte Genauigkeit zu erreichen. Dies ist für die weitere Verarbeitung genau genug.

Als Position des Objekts wird nun die x- und y-Komponente der Position des Manipulatorendpunktes gewählt. Die z-Komponente entspricht der Höhe der Objektauflagefläche.

3.5 Erkennen

3.5.1 Objektklassifizierung

Das Ziel der Klassifizierung ist es, die Ausmaße und Oberflächenbeschaffenheit der Objekte zu bestimmen und interne 3D Repräsentationen dieser Objekte zu erzeugen. Die Beschaffenheit der Oberfläche eines Objekts zu kennen, ist für die Bestimmung eines stabilen Griffes wichtig. Der Reibungskoeffizient an den Kontaktpunkten zwischen Roboterhand und Objekt ist maßgeblich bestimmend für die nötige Stärke der applizierten Kraft.

Die Oberflächenbeschaffenheit eines Objekts mittels Bildverarbeitung zu bestimmen ist sicherlich in gewissem Maße möglich, soll aber nicht Gegenstand dieser Arbeit sein. Da das Augenmerk auf der generellen Analyse von Griffen lag und die Beachtung von Reibung die nötigen Berechnungen lediglich verändert, aber nicht in sonderlichem Maße kompliziert, wurden nur die Fälle ohne Reibung betrachtet.

Was die Klassifizierung also leisten soll, ist das Schaffen einer 3D Repräsentation des Objekts. Diese beinhaltet die Menge der Oberflächenpunkte, sowie die Normalen an diesen Punkten. Im reibungslosen Fall werden die Normalen als die Richtung, in der die an diesem Punkt applizierte Kraft wirkt, angenommen.

Die Objektklassifizierung basiert auf einer PCA. Principal Component Analysis wurde in [30] erfolgreich für die Klassifizierung von Gesichtern verwendet. Hier soll sie in der Klassifizierung von greifbaren Objekten Anwendung finden.

Wird die PCA auf einen Satz von Daten angewendet, so wird ein neuer Satz von Daten geschaffen, dessen Elemente nicht mehr in Relation zu den ursprünglichen Dimensionen stehen, sondern zu einem Raum, der durch eine ausgewählte Menge

an Eigenvektoren aufgespannt wird. Werden diese Eigenvektoren geschickt gewählt, so hängen die Daten nur noch von Eigenschaften ab, welche maßgeblich für die Unterscheidung der einzelnen Datensätze sind.

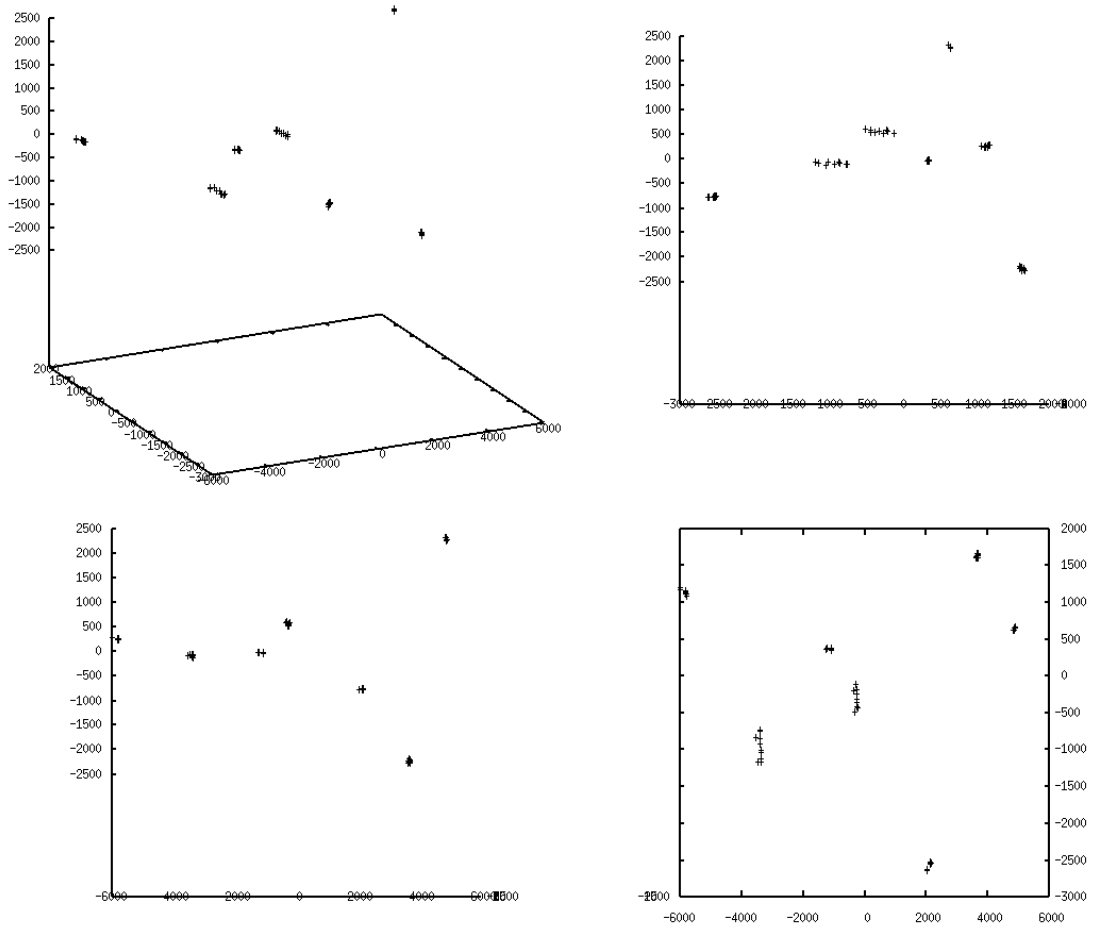


Abbildung 3.4: Sieben Objekte je zehn Bilder wurden in den Eigenraum transformiert und in einem Koordinatensystem dargestellt, das die drei Eigenvektoren mit den höchsten Eigenwerten aufspannen. Deutlich zu erkennen ist die Separation in sieben Anhäufungen, welche die Objektklassen darstellen.

Als Eingabe für die PCA wird eine Matrix benötigt, die die Datensätze enthält. Die Daten sollten so beschaffen sein, dass die gewünschten Unterscheidungsmerkmale der Objekte bereits in diesen enthalten sind. Als Eingabe sollen Bildaufnahmen der Objekte dienen. Diese müssen entsprechend aufbereitet werden.

Datenaufbereitung für die PCA

Soll PCA in der Bildverarbeitung verwendet werden, um verschiedene Objekte voneinander unterscheiden zu können, so müssen die Trainingsdaten in einer Art und

Weise erfasst werden, die die Auswahl einiger weniger „aussagekräftiger“ Eigenvektoren zulässt. Deshalb sollten sich die Testaufnahmen nur in den Merkmalen unterscheiden, die maßgeblich für die Unterscheidung der Objekte sind. Andere Unterscheidungsmerkmale sollten von vornherein ausgeschlossen werden.

So sollten sich die Objekte in den Trainingsaufnahmen, wenn möglich, stets an exakt derselben Stelle befinden. Dadurch wird verhindert, dass die Position des Objekts im Bild als markantes Merkmal extrahiert wird.

Um eine Klassifizierung mittels PCA zu ermöglichen, wurden die Trainingsaufnahmen unter folgenden Bedingungen aufgenommen:

1. die Kamera ist lotrecht zu einer ebenen Fläche positioniert
2. die Objekte werden unter der Kamera zentriert

In den Trainingsaufnahmen wurden die m zu trainierenden Objekte freigestellt und der Hintergrund schwarz gefärbt. Um die Objekte aus jeder Lage erkennen zu können, wurden sie mittels Bildverarbeitung um 360° rotiert, wobei bei jedem Rotationsschritt ein entsprechendes Bild erzeugt wurde. Bei n Rotationsschritten, erzeugt dies also n Bilder je Objekt bei einer Schrittweite von $(\frac{360}{n})°$.

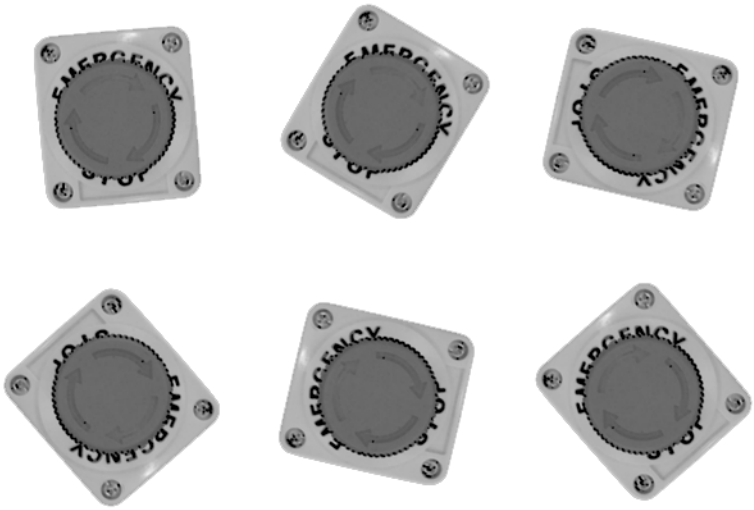


Abbildung 3.5: In sechs Schritten rotiertes Objekt

Diese $m \cdot n$ Bilder wurden in einem folgenden Schritt auf dieselbe Größe (Breite B_w × Höhe B_h) gebracht, ohne sie dabei zu verzerren.

Ein Bild B lässt sich repräsentieren als ein Vektor v der Größe $B_w \cdot B_h$. Die Gesamtheit aller Bildvektoren wird als Merkmalsvektor M repräsentiert, welcher die Eingabe für die PCA darstellt.

$$v_n = (x_1, x_2, \ldots, x_{B_w \cdot B_h})$$

$$M = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}$$

Jedes Element x_n stellt eine Eigenschaft der Objekte dar.

Zum Merkmalsvektor M muss nun die entsprechend Kovarianz-Matrix berechnet werden, auf deren Grundlage die Eigenwerte und Eigenvektoren bestimmt werden.

Ist K eine quadratische Matrix, so heißt $\vec{e} \neq \vec{0}$ Eigenvektor zum Eigenwert λ, wenn gilt

$$K \cdot \vec{e} = \lambda \cdot \vec{e}$$

Die Dimension des Merkmalsvektors entspricht der Bildgröße. Entsprechend viele Eigenvektoren werden für diese Eingabe berechnet werden.

Jener Eigenvektor mit dem höchsten Eigenwert hat die höchste Siginifikanz für die Eingabedaten, ist also das Hauptunterscheidungsmerkmal. Dieser Vektor wird „principle component“ genannt.

Je geringer der Eigenwert eines Eigenvektors, desto geringer ist seine Relevanz für die Unterscheidung der Daten.

Werden nun die Eigenvektoren mit geringer Relevanz fallen gelassen, wird die Dimension der Trainingsdaten reduziert. So lassen sich die ursprünglich $B_w \cdot B_h$ Eigenschaften auf einige wenige signifikante Unterscheidungsmerkmale reduzieren.

Klassifizierung mittels PCA

Soll nun ein Objekt klassifiziert werden, wird es ebenso aufbereitet wie die Trainingsdaten. Es wird vom Hintergrund freigestellt und auf die Größe der Trainingsbilder gebracht. Das Bild wird als Vektorrepräsentation ebenso in seine Merkmale zerlegt, wie es mit den Trainingsdaten geschah, und in Bezug auf die Eigenvektoren gebracht (s. Kapitel 2.2.2).

Der Abstand von den Merkmalen (den Eigenvektoren) klassifiziert das Objekt und identifiziert es als zugehörig zu einer der m Klassen. Sollte der Abstand zu gross sein, kann das Objekt nicht klassifiziert werden.

War die Klassifizierung mittels PCA erfolgreich, kann eine interne 3D Repräsentation des Objekts erzeugt werden, die für die Griffanalyse benötigt wird.

3.5.2 Objektrepräsentation

Wie einleitend im Kapitel der PCA bereits erwähnt, wird für die interne Objektrepräsentation ein 3D Modell benötigt, welches Punkte der Oberfläche und ihre zugehörigen Normalen liefern kann.

Um die Analyse der Griffe auf einfache Art im Programm sichtbar machen zu können, wurde die Objektrepräsentation der 3D Bibliothek OpenGL gewählt. Diese fasst Objekte als einen Verbund aus Dreiecken auf.

Im Vergleich mit der parametrischen Darstellung des Objekts, ist nur garantiert, dass die Ecken der Dreiecke auf der tatsächlichen Oberfläche liegen. Je höher die Anzahl der Dreiecke eines Objektes ist, desto präziser können also Punkte zwischen den Ecken berechnet werden. Dies kann bei stark gekrümmten Oberflächen eine Rolle spielen.

Die Objekte, die in dieser Arbeit verwendet wurden, bestehen aus einem geometrischen Primitiv und einer Position in Weltkoordinaten. Diese Position bezieht sich auf den Masseschwerpunkt des Objekts.

Die Wahl des geometrischen Primitivs entspricht einer Vorauswahl der für dieses Objekt möglichen Griffe. Wie in Kapitel 2.2.3 gezeigt wurde, ist es sinnvoll, Objekte an ihrer „längsten“ Stelle zu greifen. Griffe, die dort ansetzen, sind in der Regel stabiler als Griffe an allen anderen Positionen.

So ist es beispielsweise günstiger, ein Sektglas nicht am Stiel zu greifen, sondern am Kelch. Dieses Beispiel ist gut geeignet, um zu zeigen, wie die Vorauswahl vonstatten geht:

Ein Sektglas lässt sich mithilfe eines einzigen geometrischen Primitivs relativ exakt beschreiben. Der Fuß ist ein sehr flacher Zylinder mit großem Radius. Der Stiel ein hoher Zylinder mit kleinem Radius. Der Kelch ist wiederum ein hoher Zylinder mit großen Radius. Unter der Berücksichtigung, dass ein Objekt an seiner längsten Stelle gegriffen werden soll, bieten der Fuß und der Stiel keine zusätzliche Information für die Griffplanung. Sie können in der Repräsentation des Sektglases also ausgelassen werden.

Ein Sektglas entspricht somit einem „schwebenden“ Zylinder, dessen Masseschwerpunkt innerhalb des Kelches liegt.

Diese Art der Darstellung beschleunigt die Analyse möglicher Griffe sehr, da viele „unsinnige“ Konfigurationen gar nicht erst auftreten können.

3.6 Greifen

3.6.1 Griffbestimmung

Zunächst soll die verwendete BarrettHand noch einmal kurz ins Gedächtnis gerufen werden.

Die BarrettHand besitzt drei Finger, von denen einer starr ist und die anderen beiden symmetrisch um eine Achse rotieren können, welche vom starren Finger vorgegeben wird. Jeder Finger besitzt zwei Glieder, von denen das mit dem Handballen verbundene angesteuert werden kann, wohingegen das zweite Gelenk lediglich durch die TorqueSwitch-Funktionalität gesteuert wird. Diese schließt das zweite Gelenk, wenn am ersten Gelenk eine bestimmte Kraft anliegt und dieses am weiteren Schließen hindert.

Effektiv kann für einen Griff also über folgende Dinge entschieden werden:

- die Position der Hand, relativ zum Objekt
- die Kraft, welche an jedem Finger ausgeübt werden soll
- der Winkel, bis zu dem sich die Finger jeweils schliessen sollen (dies ist nicht vereinbar mit einer eingestellten Kraft für diesen Finger)
- der Spread, also der Öffnungswinkel zwischen den beiden rotierbaren Fingern

Um ein Objekt in „form-closure“ halten zu können, sind sieben Kontaktpunkte hinreichend [18]. Dies kann die BarrettHand nur erreichen, indem „power grasps“ durchgeführt werden. Also Griffe, bei denen alle Finger und der Handballen vollen Kontakt zum Objekt herstellen.

Für diese Art der Griffe schränken sich die Parameter bei der Griffkonfiguration auf Kraft und Spread ein.

Aufgrund dieser Einschränkungen durch die BarrettHand wurde eine Vorauswahl an möglichen Konfigurationen getroffen, welche entweder auf sieben Kontaktpunkte kommen, oder auch mit weniger Kontaktpunkten erwarten lassen, dass sie bei vielen Objekten zu „form-closure“ Griffen führen.

Dies führt zu fünf als „sinnvoll“ erscheinenden Griffkonfigurationen, die sich in zwei Eigenschaften unterscheiden. Sie können entweder von oben oder von der Seite ausgeführt werden und unterscheiden vier Spread Varianten. Alle Griffe, die von der Seite ausgeführt werden, sind „power grasps“. In Anhang A.2 finden sich Abbildungen dieser Konfigurationen.

Werden Objekte mit der PCA trainiert, so kann ihnen auch ein Satz Griffe zugeordnet werden, welcher für diese Art Objekt geeignet scheint.

Liegen keine Griffe vor - entweder wurden für das Objekt keine vergeben oder es handelt sich um ein unbekanntes Objekt - so werden bei der Griffanalyse alle Griffkonfigurationen berücksichtigt.

3.6.2 Griffanalyse

Die Griffanalyse berechnet, ob ein gewählter Griff in der Lage ist, das erkannte Objekt stabil zu greifen.

Das Qualitätsmaß dafür ist, wie bereits mehrfach erwähnt, die Eigenschaft „form-closure“. Die Überprüfung auf diese Eigenschaft erfolgt nach dem in Kapitel 2.3.1 vorgestellten Verfahren.

Dazu ist es nötig, die Kollisionspunkte der gewählten Griffkonfiguration mit dem Objekt zu bestimmen. Dies geschieht mittels Simulation.

Dabei kam eine Bibliothek zum Einsatz, welche auf die Kollisionsdetektion der WildMagic3 Engine zurückgreift.

Die BarrettHand und das zu greifende Objekt werden dazu in eine OpenGL Szene gestellt und der Greifvorgang simuliert. Die Hand wird zunächst an die entsprechende Position transliert, an welcher sie das Objekt greifen soll. In jedem weiteren Simulationsschritt werden die Finger der Hand inkrementell geschlossen und auf Kollision mit dem Objekt hin geprüft.

Für von der Seite ausgeführte Griffe wird die Hand so zum Objekt ausgerichtet, dass sie sich idealerweise mit dem Zentrum des Handballens auf halber Höhe des Objekts befindet. Dabei wird darauf geachtet, dass ein gewisser Sicherheitsabstand von der Tischoberfläche gewahrt bleibt, da es sonst zu einer Kollision zwischen diesem und dem Arm kommen könnte. Die Hand wird nun solange in Richtung Objekt verschoben, bis der erste Kontakt eintritt. Erst dann werden die Finger geschlossen.

Bei Griffen von oben wird die Hand in einer Höhe über dem Objekt platziert, die es wahrscheinlich macht, dass die Fingerspitzen auf halber Höhe des Objekts in Kontakt treten. Dazu wurde eine Konstante bestimmt, welche auf der Länge der Finger basiert, und auf die halbe Höhe des Objekts addiert. Sollte dies durch die Höhe des Objekts unmöglich sein, so wird die Hand möglichst nahe am Objekt platziert, wahrt dabei jedoch einen geringen Abstand, um ein „eintauchen“ der realen Hand in das Objekt zu vermeiden. Liegt der Arm weit aus, kann die Plattform bei Armbewegungen anfangen leicht zu schwanken.

Aufgrund der Dreiecks Repräsentation liefert die verwendete Bibliothek als Kollisionspunkt die beiden Dreiecke, an denen die beiden Objekte sich berühren. Dies liefert in den meisten Fällen eine Linie als Schnittpunkt[4]. Wie genau der tatsächliche

[4]Dies ist bedingt durch die diskrete Repräsentation. Eine Kollision tritt in der Regel erst dann ein, wenn Teile des einen Objekts in Teile des anderen Objekts „eintauchen“.

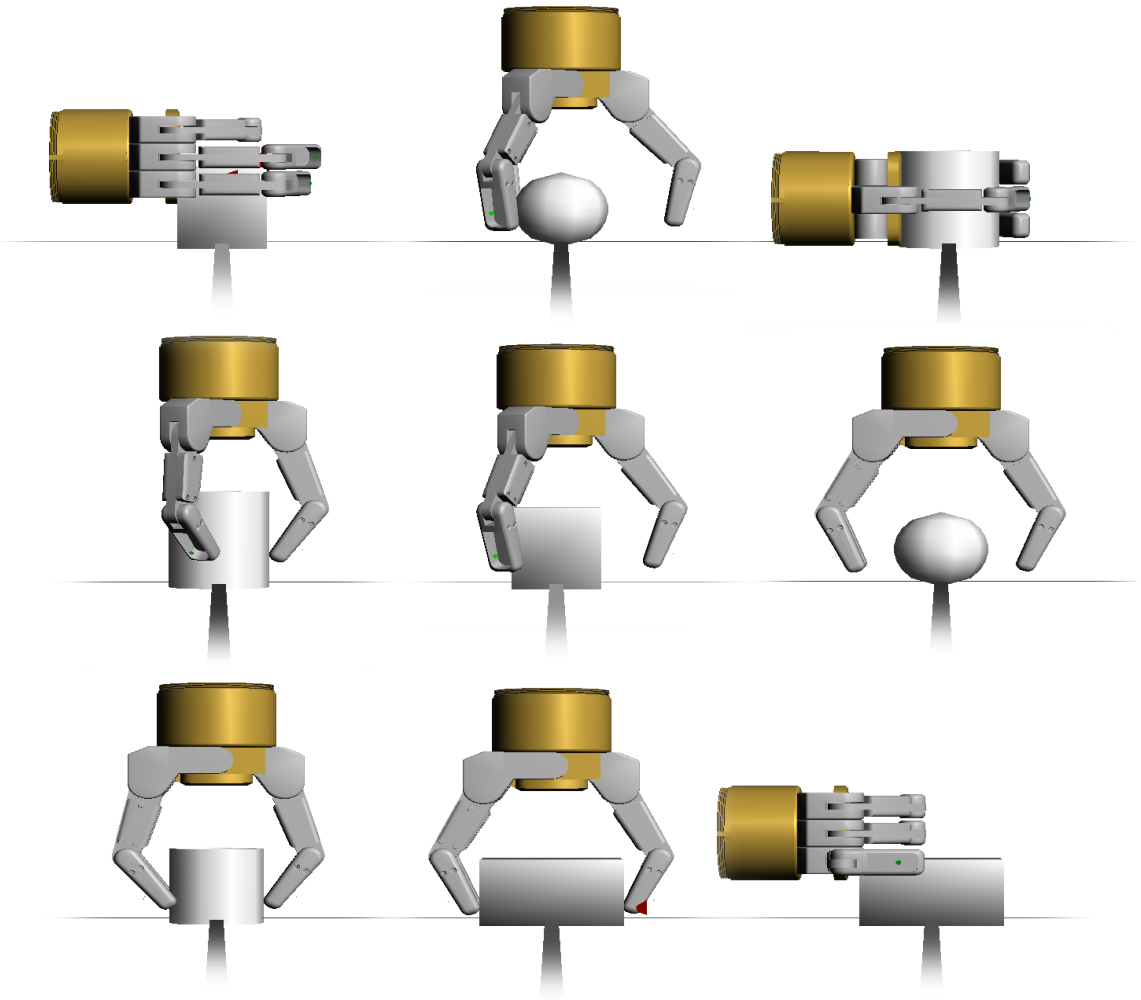


Abbildung 3.6: Visualisierungen der Simulation für unterschiedliche Griffe bei verschiedenen Objekten

Berührungspunkt der beiden Objekte bestimmt werden kann, hängt von der Größe dieser Dreiecke ab. Typischerweise bestehen Objekte an Stellen mit geringer Oberflächenkrümmung aus wenigen, großen Dreiecken. Beim Modell der BarrettHand sind dies im Besonderen die Oberflächen der Fingerglieder, da diese nicht gekrümmt sind. Die Krümmung an den Fingerspitzen ist stärker, sie bestehen also aus kleineren Dreiecken. Da die verwendeten „power grasps“ hauptsächlich zu Berührungen mit den Fingergliedern führen, sind die Ergebnisse teilweise nicht exakt, sollen für die Analyse aber ausreichend sein.

Sind alle Kollisionspunkte r_i bekannt, können diese einer „form-closure“ Analyse unterzogen werden. Die dafür notwendigen Normalen n_i entsprechen der Normalen des Dreiecks der Oberfläche des Objekts, welche diesen Punkt enthält.

Der Qualitätstest erfolgt nach Algorithmus 2.16. Es folgt eine kurze Wiederholung der dafür nötigen Schritte.

Es werden die „primitive contact wrenches“ w_i gebildet

$$w_i = \begin{pmatrix} n_i \\ r_i \times n_i \end{pmatrix}$$

Der Mittelwert der „primitive contact wrenches“ ist der Punkt P, welcher in der konvexen Hülle der w_i liegt.

Es wird ein lineares Optimierungsproblem folgender Gestalt definiert, mit n als der Anzahl der „primitive contact wrenches“ bzw. Kontaktpunkte:

Maximiere:
$-P_1e_1 - P_2e_2 - \ldots - P_6e_6$

Unter den Nebenbedingungen:
$$\begin{aligned}
(w_{11} - P_1)e_1 + (w_{12} - P_2)e_2 + \ldots + (w_{16} - P_6)e_6 &\leq 1 \\
(w_{21} - P_1)e_1 + (w_{22} - P_2)e_2 + \ldots + (w_{26} - P_6)e_6 &\leq 1 \\
&\vdots \\
(w_{n1} - P_1)e_1 + (w_{n2} - P_2)e_2 + \ldots + (w_{n6} - P_6)e_6 &\leq 1
\end{aligned}$$

$e_1, e_2, \ldots, e_6$ unbeschränkt.

Die Lösung $E = (e_1, e_2, \ldots, e_6)$ dieses Problems beschreibt durch Transformation $T : \sum_{i=1}^{6} e_i \cdot x_i = 1$ die Hyperebene, welche vom Strahl $\vec{PO}$ geschnitten wird[5].

Der Schnittpunkt Q mit der konvexen Hülle lässt sich berechnen durch $Q = \lambda \cdot P$ mit $\lambda = \left(\sum_{i=1}^{6} E_i \cdot P_i\right)^{-1}$.

Ist der Abstand zwischen P und Q echt größer als der Abstand zwischen P und O, so ist der Griff „form-closure“.

Für die lineare Optimierung wurde eine Implementierung des Soplex Algorithmus von R. Wunderling verwendet [29].

[5] O ist der Ursprung des „wrench space“, s. Kapitel 2.3.1

3.6.3 Ausführen des Griffes

Der aktuelle Endpunkt wird mittels einer Transformationsmatrix gesetzt. Der gewünschte Zielpunkt kann anhand dieser Matrix gesetzt werden. Die nötigen Gelenkgeschwindigkeiten werden durch inverse Kinematik ermittelt und umgesetzt. Wird dabei kartesische Interpolation verwendet, wird versucht, den Zielpunkt in einer geraden Linie anzufahren. Für kurze Distanzen ist dies brauchbar und wünschenswert, bei längeren Distanzen können zu hohe Gelenkgeschwindigkeiten auftreten, welche der Arm nicht mehr ausführen kann. Deshalb wird für große Positionsveränderungen (in der Regel größer als 20 cm) auf kartesische Interpolation verzichtet.

Für den Greifvorgang wird der Handballen als Endpunkt gesetzt. Die nötigen Translationen und Rotationen zur Erlangung des Zielpunktes ergeben sich aus der zu verwendenden Hand-Konfiguration. Größtes Unterscheidungsmerkmal ist hier, ob der Griff von oben oder von der Seite ausgeführt werden soll.

Der Spread der Hand wird auf Grundlage der Ergebnisse der Simulation gesetzt. Die Finger schließen sich über eine gesetzte Kraft. Diese bestimmt, wie schnell sich die Finger jeweils schließen müssen, um diese Kraft erreichen zu können.

Um manuell überprüfen zu können, ob der Griff erfolgreich war, wird das Objekt kurz angehoben und daraufhin wieder abgesetzt. Der Griff kann als „besonders gut“ gelten, wenn das Objekt dabei nicht verschoben wurde. Dies ist allerdings keine Anforderung, welche explizit an das System gestellt wurde. Ein Griff gilt deshalb als erfolgreich, wenn er das Objekt anheben konnte.

4 Ergebnisse & Erläuterungen

Der vorgestellte Ansatz hat sich als tauglich für die Planung von robusten Griffen für diverse Objekte herausgestellt. Objekte, die durch die interne Repräsentation relativ präzise dargestellt wurden, konnten nahezu immer stabil gegriffen werden.

Die Hauptschwäche des Systems liegt demzufolge in der Ungenauigkeit, mit der Objekte teilweise repräsentiert wurden.

Da der erläuterte Ablauf der Planung nicht von den einzelnen Methoden abhängig ist, könnten also andere Methoden verwendet werden, welche die Ergebnisse der Phasen verbessern und somit zu einem besseren Gesamtergebnis führten. Einige Ansätze dazu werden im Ausblick in Kapitel 5 geliefert.

4.1 Sehen

Die Ungenauigkeit der Rückprojektion mittels einer unkalibrierten Kamera konnte durch die Kombination mit „visual servoing“ effektiv kompensiert werden. Die Rückprojektion zunächst für die grobe Lokalisierung zu verwenden, ist gut geeignet, um das für die Feinpositionierung interessante Gebiet einzugrenzen und reduziert somit den Zeitaufwand der visuell geführten Regelung erheblich.

Um die Güte der Lokalisierung unabhängig von Griffplanung und internen Repräsentationen zu testen, wurde für einige Objekte mittels „visual servoing“ die dem Roboter am nähesten liegende Kante des jeweiligen Objekts bestimmt. Diese Position wurde mit der Handfläche des Roboters angefahren. Dabei wurden die Objekte sehr präzise angefahren, die Handfläche berührte die Objekte, verschob diese aber niemals mehr, als die für die Lokalisierung gewählte Präzision zuließ. Die Objekte wurden also korrekt und präzise lokalisiert.

Wie schnell die Position des Objekts dabei bestimmt wurde, hing maßgeblich davon ab, wie schnell die Szene segmentiert werden konnte.

4.2 Erkennen

Die Erkennungsraten der PCA hängen maßgeblich davon ab, wie die Trainingsbilder aufbereitet wurden. In Experimenten wurden zwei Grundsätzliche Verfahren verglichen:

- Rotierte Aufnahmen der Objekte direkt von oben (Rotationsbilder)
- Rundumaufnahmen der Objekte

Für die Rotationsbilder wurden die Objekte einmalig von oben aufgenommen und mittels Bildverarbeitung rotiert. Für die Rundumaufnahmen wurde die Kamera mit einer festen Schrittweite auf einer Kreisbahn um die Objekte bewegt und in jedem Schritt jeweils ein Bild aufgenommen (s. *Abbildung 4.2*).

Bei den Rundumaufnahmen als Trainingsdaten hat sich allerdings ein Nachteil gezeigt: wie erwähnt, sind die Ergebnisse der PCA sehr stark davon abhängig, unter welchen Bedingungen die Bilder der zu klassifizierenden Objekte aufgenommen wurden. Bei Verwendung der Rundumaufnahmen hätte das Objekt also aus einer ähnlichen Position aufgenommen werden müssen, wie sie auch bei der Erfassung der Trainingsdaten verwendet wurde. Dazu ist es allerdings nötig, die Position des Objekts zu kennen. Die Rückprojektion liefert die Position des Objekts auf 2-5 cm genau, was zu ungenau ist, um eine robuste Klassifikation mittels PCA zu gewährleisten. Da die exakte Lokalisierung mittels „visual servoing“ eine Positionierung über dem Objekt erfordert, war die Verwendung von Rotationsbildern naheliegender.

Dabei wurden zwei grundsätzliche Varianten der Aufbereitung der Rotationsbilder getestet:

- Ein aus mehreren Bildern zusammengesetztes Bild des Objekts
- Je Objekt eine Reihe von Bildern mit aufsteigendem Rotationswinkel

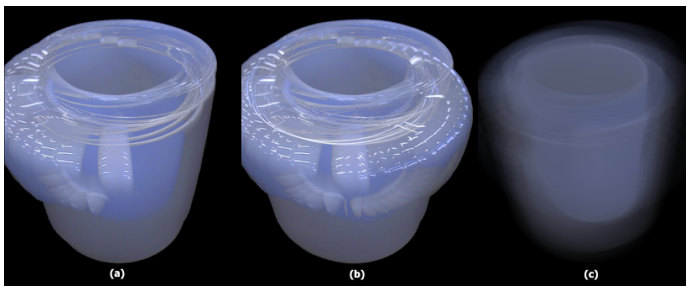


Abbildung 4.1: (a) - Median, (b) - Maximale Intensität, (c) - Mittelwert

Grundlage für die zusammengesetzten Bilder war ebenfalls eine Reihe von Bildern mit aufsteigendem Rotationswinkel. Für die PCA wurde für jedes Objekt aus diesen

Bildern ein einzelnes Bild generiert. Die Pixel dieses Bildes wurden als Mittelwert, Median oder höchster Intensität der korrispondierenden Pixel der jeweiligen Objektaufnahmen berechnet (*Abbildung 4.1* stellt dies für eine Serie von Rundumaufnahmen dar). Die Erkennungsraten der auf diese Art trainierten PCA waren allerdings verschwindend gering.

Für die zweite Variante wurde die PCA zunächst mit Bildern in voller PAL-Auflösung trainiert. Dabei wurden die Objekte in 36°-Schritten rotiert, es flossen also je Objekt zehn Bilder ein. Für einen Testlauf mit sieben Objekten, also 70 Bildern, benötigte die PCA über sieben Minuten, um den Eigenraum zu generieren. Die Erkennungsraten waren gut, solange die zu klassifizierenden Objekte sich in der Nähe des in der Trainingsphase eingeflossenen Rotatationswinkel der Objekte befanden.

Um die Trainingszeit für die PCA zu verkürzen, wurden die 70 Bilder deshalb in einem zweiten Versuch auf eine Auflösung von 150×150 Pixeln reduziert. Die Trainingsdauer der PCA reduzierte sich dadurch drastisch auf durchschnittlich 36 Sekunden. Die Erkennungsraten stiegen entgegen der ersten Erwartung an, so wurde im Versuch kein Trainingsobjekt falsch klassifiziert oder nicht erkannt. Dies ist sehr wahrscheinlich hauptsächlich zurückzuführen auf das durch die Skalierung der Eingangsbilder reduzierte Rauschen. Allerdings trat im Versuch eine Fehlklassifikation auf: ein nicht zu den Trainingsobjekten gehörendes Objekt wurde einer Klasse zugeordnet.

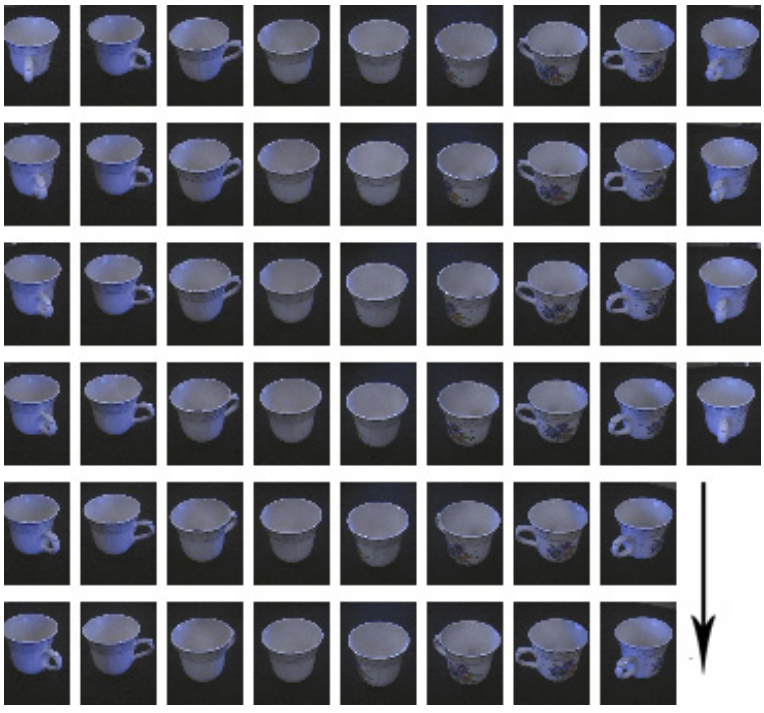


Abbildung 4.2: Rundumaufnahmen

Der vorgestellte Ansatz zur Erkennung unbekannter Objekte hat sich als unzureichend in Zusammenspiel mit der verwendeten Kamera gezeigt. Dadurch, dass sie nicht kalibriert wurde, waren die Objekte teilweise sehr verzerrt. Die theoretischen

Rechtecke wurden so in einigen Fällen als Elipsoide klassifiziert. Ein weiteres Problem stellten Objekte mit „Auswüchsen“ dar, wie beispielsweise der Henkel einer Tasse. Diese markanten Merkmale wären gut geeignet, eine höherstufige Art der Klassifizierung zu verwenden, störten aber die Erkennung von geometrischen Primitiven. Wie in Abbildung 2.12 zu sehen ist, liefert die Topansicht eine sehr saubere Form. Der Henkel könnte relativ einfach lokalisiert und bei der Formbestimmung ignoriert werden. In der Frontansicht haben Reflektionen allerdings dafür gesorgt, dass bei der Segmentierung mittels Schwellwert die Konturen sehr unsauber wurden. Die verwendeten Algorithmen zur Formerkennung waren nicht geeignet, solche Deformationen auszugleichen.

4.3 Greifen

Die mittels der Simulation ermittelten „form-closure“ Griffe waren in den meisten Fällen fähig, die Objekte auch zu greifen. Die durch die Darstellung aller Objekte durch geometrische Basisprimitive vorhandene Diskrepanz zwischen interner Repräsentation und tatsächlicher Objektoberfläche führte allerdings dazu, dass manche Griffe nicht zu den berechneten stabilen Griffen führten. Die Objekte konnten häufig dennoch angehoben werden, wenn auch nicht sehr stabil.

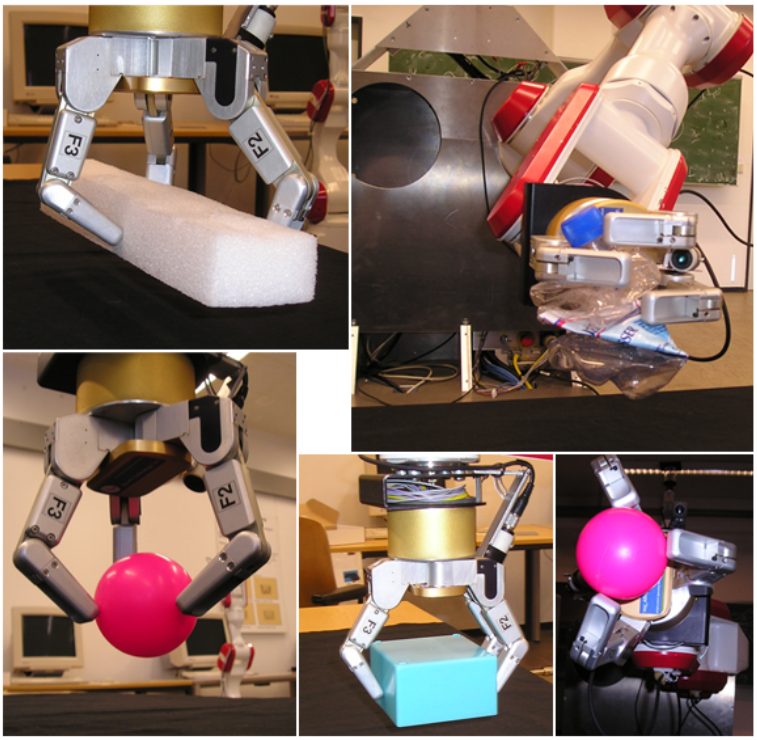
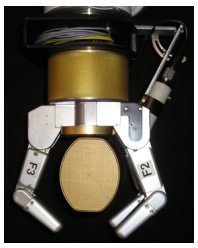


Abbildung 4.3: Einige erfolgreich durchgeführte Griffe

Ein durch die Simulation nicht abgedecktes Problem stellte die Motorik der Barrett-Hand dar. Um die Finger auf eine bestimmte Kraft zu bringen, müssen sie entspre-

chend beschleunigt werden. Dabei sind nicht alle Finger gleich schnell, sie erreichen ihre Zielposition demzufolge zu unterschiedlichen Zeitpunkten. Trifft ein Finger auf das Objekt, es liegt aber noch kein Gegendruck der anderen Finger vor, so kann das Objekt verschoben werden. Besonders bei von oben ausgeführten Griffen kam es so häufig dazu, dass die Objekte derart verschoben wurden, dass die nachfolgend schließenden Finger es noch weiter verschoben und es schließendlich nicht mehr innerhalb der Fingerspitzen lag - die Hand greift ins Leere.



Um dieses Problem zu umgehen, wurde von den Gelenkwinkeln, die die Simulation liefert, ein geringer Wert abgezogen und die Finger zunächst an diese Position gebracht. Dadurch wird gewährleistet, dass kein Finger das Objekt berührt. Wenn alle Finger diese Position erreicht haben, sind sie nur noch wenig vom Objekt entfernt. Erst dann werden sie vollends mit der benötigten Kraft geschlossen.

Durch diese Methode wurde die Erfolgsrate der Griffe signifikant erhöht.

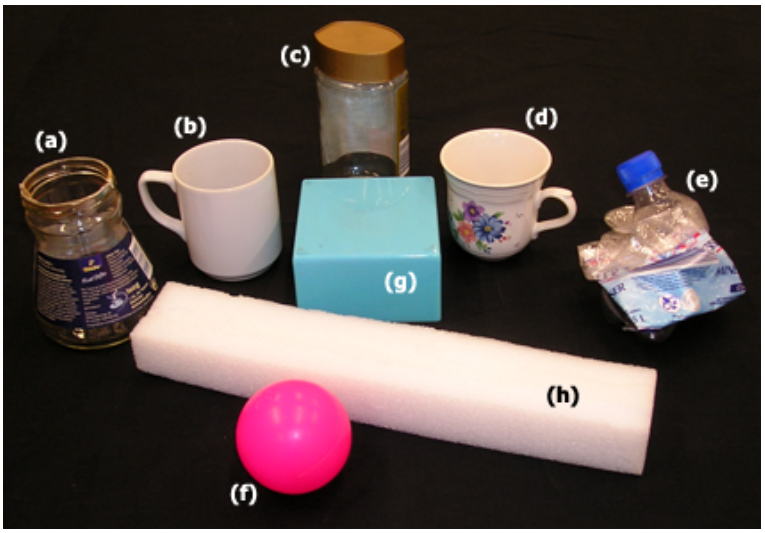
In *Abbildung 4.3* sind einige Objekte und die Erfolgsraten der für diese berechneten Griffe aufgeführt. Dabei wurden sechs Objekte verwendet, welche durch Basisprimitive relativ genau erfasst werden konnten und drei Objekte, deren tatsächliche Form von ihrer internen Repräsentation teilweise sehr stark abwich ((a), (c) und (e)).

Objekt (c) wurde dabei durch einen umschließenden Quader repräsentiert, Objekt (a) durch einen Zylinder, dessen Durchmesser dem Sockel des Objekts entsprach. Für das sehr unförmige Objekt (e) - eine zerdrückte Plastikflasche - wurden zwei verschiedene Repräsentationen gewählt: ein umschließender Quader und eine Kugel, deren Durchmesser der größten Ausdehnung des Objekts entsprach.

Der für (c) bestimmte Griff war fähig, das Objekt problemlos und stabil zu greifen. Für die interne Repräsentation von (a) wurde zwar ein „form-closure“-Griff gefunden, dieser war allerdings nicht geeignet, das Objekt zu greifen. Durch die konische Form rutschte es aus den geschlossenen Fingern.

Für die Quader-Repräsentation des Objekts (e) konnte kein „form-closure“-Griff gefunden werden. Die Kugelrepräsentation ergab, dass das Objekt von der Seite mit Griff S180 gegriffen werden sollte. Dies führte in zwei von drei Fällen auch zum Erfolg. Der Griff war dann erfolgreich, wenn der oberste Finger den Flaschenhals umschloss. War dies nicht der Fall, rutschte das Objekt aus den Fingern.

Objekt (g) war ein Sonderfall. Keiner der fünf Griffe wurde als „form-closure“ ausgegeben, obwohl ein derart gestaltetes Objekt typischerweise sehr einfach und stabil zu greifen ist. Ein manuell ausgelöster T180 für dieses Objekt führte zum Erfolg.



Objekt	ausgeführter Griff	Erfolg?
(a)	S180	Nein
(b)	T180	**Ja**
(c)	S180	**Ja**
(d)	T180	**Ja**
(e)	S180	s. Text
(f)	T120	**Ja**
(g)	-	s. Text
(h)	T120	**Ja**

Abbildung 4.4: Erfolgsrate einiger Griffplanungen

Δt_{sim} Δt_{fcc} form-closure	S0	S180	T120	T180	T90
Weisse Tasse	31,511483	31,255426	5,522343	5,105740	4,183311
Zylinder	0,4076	0,19759	0,2896	0,2493	0,16563
$h = 90,\ r = 37,5$	**fc**	**fc**	nfc	**fc**	nfc
Notaus	16,209292	16,125395	4,90534	4,501851	4,49514
Quader	0,17819	0,4280	0,15965	0,16558	0,37657
$b = 65,\ h = 70,\ t = 75$	**fc**	**fc**	nfc	nfc	nfc
Aibo Ball	30,558918	30,309252	6,335722	6,3430	5,535654
Kugel	0,17450	0,4937	0,2558	0,16851	0,2540
$r = 35$	nfc	nfc	nfc	**fc**	nfc
Blaue Box	8,2491	7,98294	4,365500	4,95921	4,350144
Quader	0,15735	0,17073	0,2294	0,2380	0,16094
$b = 105,\ h = 60,\ t = 105$	nfc	nfc	nfc	nfc	nfc
Blumen Tasse	23,217670	23,324081	5,826497	4,917632	5,108941
Zylinder	0,18879	0,2731	0,17499	0,983649	0,2551
$h = 70,\ r = 35$	nfc	nfc	**fc**	**fc**	nfc

Abbildung 4.5: Simulation Performanz in Sekunden (Objektmaße Millimeter). 1. Zeile je Objekt - Benötige Zeit für die Simulation, 2. Zeile - Benötige Zeit für Überprüfung auf „form-closure“, 3. Zeile - fc = „form-closure“, nfc = „non form-closure“

5 Ausblick

Von unbekannten Objekten interne Repräsentationen zu erstellen, um auch für diese Griffe finden zu können, ließ sich im Rahmen dieser Arbeit nicht mehr bewerkstelligen. Der vorgestellte Ansatz, aus Front- und Topaufnahmen auf die Form des Objekts zu schließen, war mittels einer unkalibrierten Kamera nicht umsetzbar. Um auf die tatsächliche Form eines Objekts aus einer Aufnahme schließen zu können, müsste diese von einer kalibrierten Kamera aufgenommen werden. Dies würde zum einen die Radialverzerrung als Störquelle ausschließen, zum anderen könnte aus der Aufnahme so die perspektivische Verzerrung herausgerechnet werden. Es ist allerdings fraglich, ob alle dazu nötigen Informationen aus nur einer einzigen Aufnahme extrahiert werden können.

Anhand der Ergebnisse der PCA lässt sich auf die Rotation der Objekte schließen. Die Idee dahinter beruht auf der Art und Weise, mit der die Trainingsdaten beschaffen sind. Für jedes Objekt liegt eine Folge von Rotationen dieses Objekts vor. Ein Eigenvektor wird also aller Voraussicht nach diese Eigenschaft beschreiben. Nach der Klassifikation des Objekts kann der Abstand dieses Eigenvektors des Objekts zu den Eigenvektoren der PCA dienen, um die Rotation des klassifizierten Objekts zu bestimmen. Die Güte des Ergebnisses ist dabei von der Anzahl der Trainingsbilder je Objekt abhängig. In anderen Experimenten wurde gezeigt, dass die Bestimmung der Rotation dabei gut funktioniert.

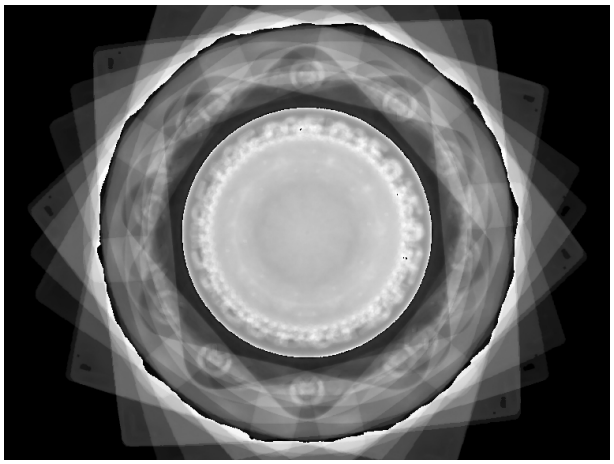


Abbildung 5.1: Darstellung eines Eigenvektors, welcher auf die Rotation eines Objekts hindeutet.

Das vorgestellte Verfahren zur Überprüfung auf „form-closure“ ist geeignet, auf beliebig geformte Objekte angewendet zu werden. Je präziser das reale Objekt dabei

erfasst wurde, desto besser können die Griffe an das Objekt angepasst werden. Die verwendete Methode, einige für die BarrettHand geeignete Griffe zu überlegen und mittels Simulation anhand von groben Repräsentationen der Objekte auf Tauglichkeit zu überprüfen, hat zu guten Ergebnissen geführt. Die Robustheit der Griffe würde durch präziser erfasste Objekte allerdings zunehmen.

Die Einschränkung der Handkonfigurationen führte dazu, dass für einige Objekte keine „form-closure“ Griffe gefunden werden konnten, obwohl diese eigentlich „form-closure“-greifbar sein sollten. Der in Kapitel 2.3.2 vorgestellte Algorithmus zum Finden von „form-closure“ Griffen könnte an die BarrettHand angepasst werden und würde es so ermöglichen, aktiv nach Griffen zu suchen. Die Idee des Algorithmus ist es, die Kontaktpunkte solange auf der Oberfläche zu verschieben, bis die Eigenschaft „form-closure“ erreicht ist. Da die BarrettHand nicht beliebige sieben Kontaktpunkte herstellen kann, sondern diese maßgeblich von der Position des Handballens abhängig sind, kann der Kontakt des Handballens mit dem Objekt als Bedingung gestellt werden. Dadurch lassen sich die übrigen Kontaktpunkte durch das Lösen eines Ungleichungssystems bestimmen. Der Spread gibt vor, an welcher Stelle die unteren Fingerglieder in Kontakt mit dem Objekt treten. Die Gelenkwinkel der unteren Glieder bei Kontakt geben vor, an welcher Stelle die Fingerspitzen mit dem Objekt in Kontakt treten. Um die Kontaktpunkte auf der Oberfläche zu verschieben, müssen also nur fünf Parameter betrachtet werden: der Kontaktpunkt des Handballens, der Spread und die drei Gelenkwinkel der unteren Glieder.

Anhang

A

A.1 Experimentelle Bestimmung des Öffnungswinkels einer CCD Kamera

Aus der Größe des CCD lässt sich berechnen, wie groß der Ausschnitt der realen Welt ist, welchen die Kamera aktuell zeigt.

Ist die Größe des CCD nicht bekannt, so kann auch der Öffnungswinkel als Grundlage dienen, indem er unter definierten Bedingungen bestimmt wird. Dazu wird die Kamera in der Entfernung E lotrecht auf eine Kalibrationsfläche ausgerichtet.

Die maximale Breite und Höhe des Ausschnitts der Kalibrationsfläche dient als Basis für die Berechnung des Öffnungswinkels. Es wird dabei in vertikalen und horizontalen Öffnungswinkel unterschieden.

Sei die Breite des Ausschnitts B_K und die Höhe H_K.

Der Mittelpunkt des CCD und die zwei äussersten Punkte der vertikalen und horizontalen Ebene bilden zwei Dreiecke. Die Hälfte eines jeden Dreiecks bildet durch die lotrechte Ausrichtung der Kamera ein rechtwinkeliges Dreieck.

Dadurch lassen sich der horizontale (γ_h) und vertikale (γ_v) Öffnungswinkel leicht bestimmen:

$$\sin \gamma_h = \frac{\frac{1}{2} B_K}{E}$$
$$\sin \gamma_v = \frac{\frac{1}{2} H_K}{E}$$

Die Güte des Ergebnisses hängt von der Kalibrierung der Kamera ab. Besonders die Radialverzerrung der Linse verfälscht das Ergebnis mitunter sehr stark. Auch leicht „schief“ eingebaute CCDs sind schon Ursache von Fehlberechnungen gewesen.

A.2 Griffkonfigurationen

Die Darstellungen der Griffkonfigurationen wurden rotiert, um die Konfiguration der Finger besser sichtbar zu machen. Ausgeführt werden sie lotrecht zur Objektebene.

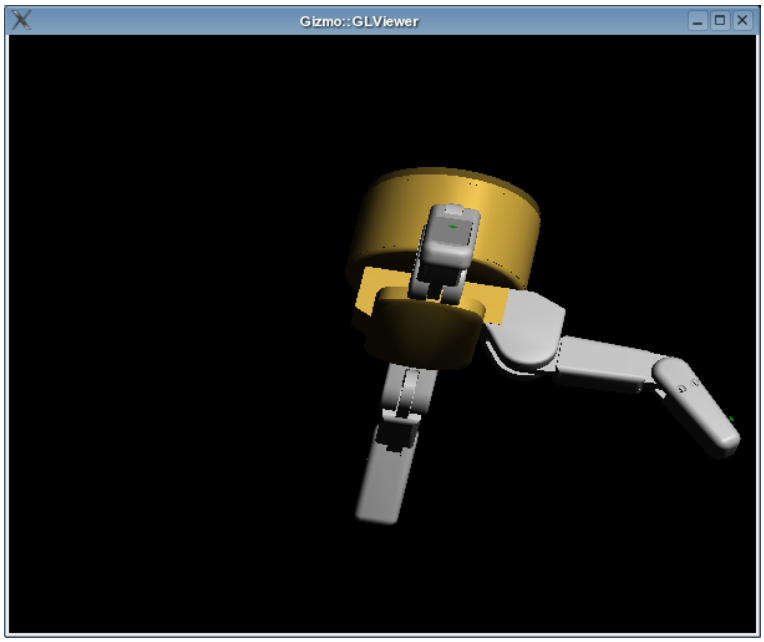


Abbildung A.1: T90 - Griff von Oben bei einem Spread von 90°

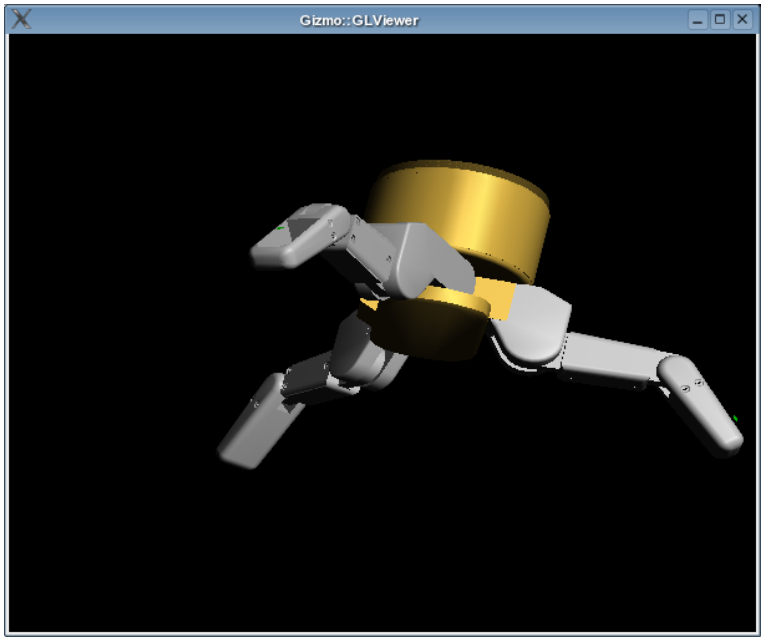


Abbildung A.2: T120 - Griff von Oben bei einem Spread von 120°

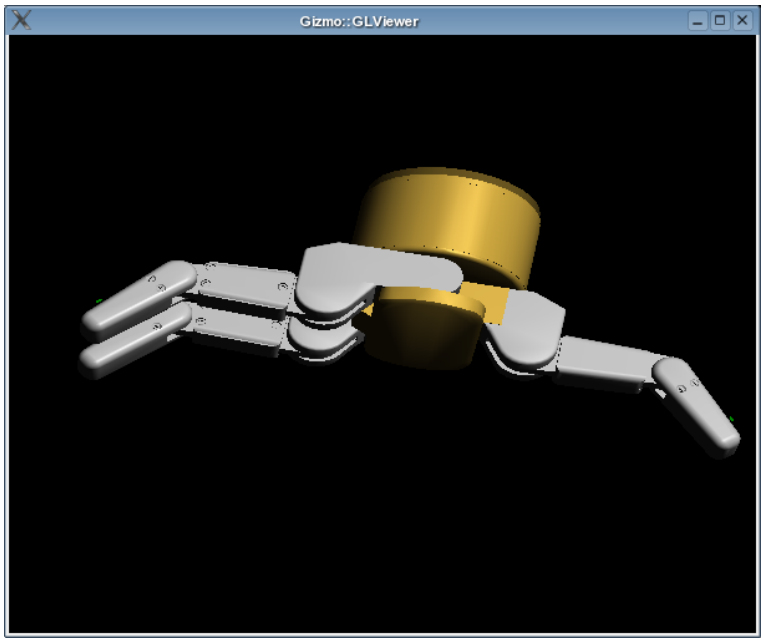


Abbildung A.3: T180 - Griff von Oben bei einem Spread von 180°

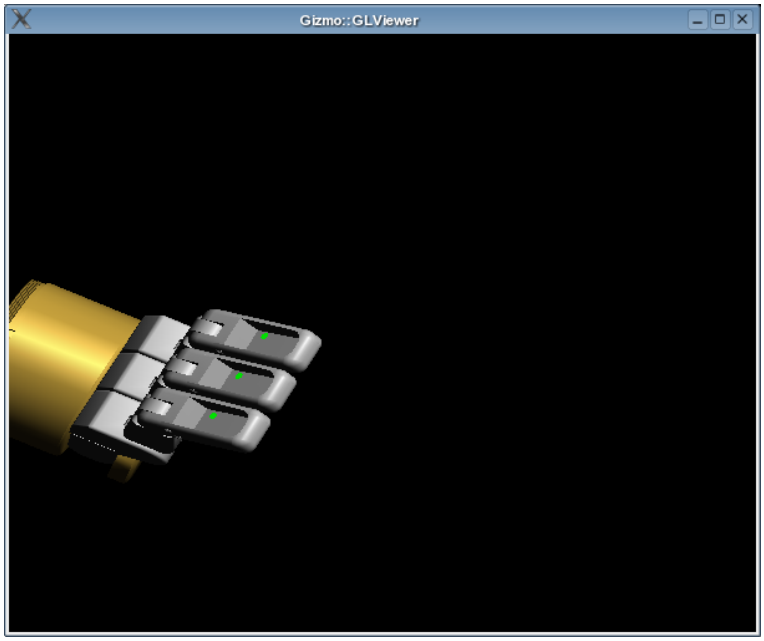


Abbildung A.4: S0 - Griff von der Seite bei einem Spread von 0°

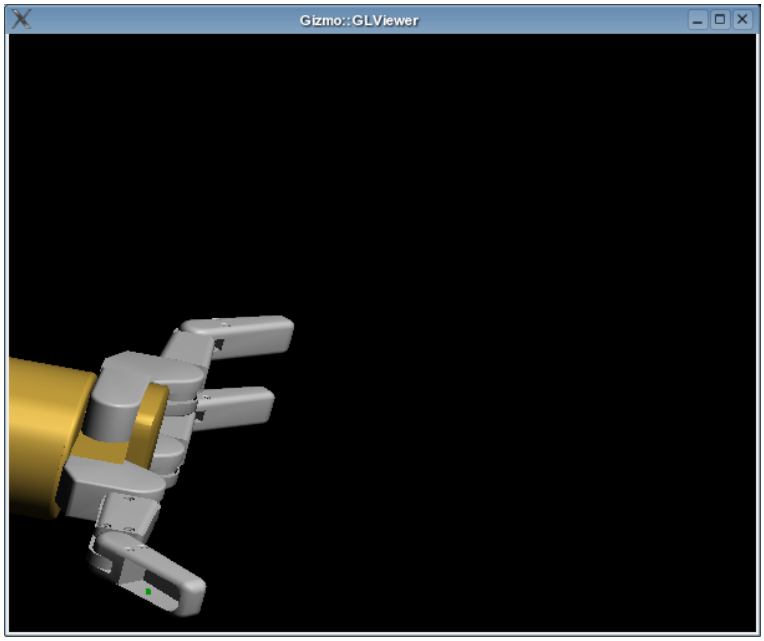


Abbildung A.5: S180 - Griff von der Seite bei einem Spread von 180°

Literaturverzeichnis

[1] Bandlow, Thorsten: *Objektidentifikation anhand von Silhouetten*, TU München, Diplomarbeit, 1997

[2] Borst, Ch. ; Fischer, M. ; Hirzinger, G.: Grasping the Dice by Dicing the Grasp. In: *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2003

[3] Cornella, Jordi ; Suarez, Raul: On 2D 4-Finger Frictionless Optimal Grasps. In: *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2003

[4] Eberly, David. *Dynamic Collision Detection using Oriented Bounding Boxes.* 1999

[5] Gleason, S. S. ; Abidi, M. A. ; Sarif-Sarraf, H.: Probabilistic Shape and Appearance Model for Scene Segmentation. In: *Proceedings of the 2002 IEEE International Conference on Robotics & Automation*, 2002, S. 2982–2987

[6] Guan, Yisheng ; Zhang, Hong: Kinematic Feasibilty Analysis of 3D Grasps. In: *Proceedings of the 2001 IEEE International Conference on Robotics & Automation*, 2001

[7] Harris, Chris ; Stephens, Mike: A combined corner and edge detector / Plessey Research Roke Manor. 1988. – Forschungsbericht

[8] Hauck, Alexa ; Färber, Georg: Hybrid Hand-Eye Coordination with a Single Stationary Camera / TU München. – Forschungsbericht

[9] Hauck, Alexa ; Passig, Georg ; Rüttinger, Johanna ; Sorg, Michael ; Färber, Georg: Biologically Motivated Hand-Eye Coordination for the Autonomous Grasping of Unknown Objekts / TU München. – Forschungsbericht

[10] Hauck, Alexa ; Passig, Georg ; Schenck, Thomas ; Sorg, Michael ; Färber, Georg: On the Performance of a Biologically Motivated Visual Control Strategy for Robotic Hand-Eye Coordination / Universität München. – Forschungsbericht

[11] Inc., Silicon G.: *OpenGL Programming Guide, 'The Red Book', 2nd Edition.* Addison-Wesley Publishing Company, 1997

[12] Liu, Yun-Hui: Qualitative Test and Force Optimization of 3-D Frictional Form-Closure Grasps Using Linear Programming. In: *IEEE Transactions on Robotics and Automation, Vol. 15, No. 1*, 1999

[13] Liu, Yun-Hui ; Lam, Miu-Ling: Searching 3-D Form-Closure Grasps in Discrete Domain. In: *IEEE Transactions on Robotics and Systems*, 2003

[14] Lloyd, John ; Hayward, Vincent: *Multi-RCCL User's Guide*, April 1992

[15] Morales, Antonio ; Chinellato, Eris ; Fagg, Andrew H. ; Pobil, Angel P.: Experimental prediciton of the performances of grasp tasks from visual features. In: *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2003

[16] Morales, Antonio ; Rectala, Gabriel ; Sanz, Pedro J. ; Pobil, Angel P.: Heuristic Vision-Based Computation of Planar Antipodal Grasps on Unknown Objects. In: *Proceedings of the 2001 IEEE International Conference on Robotics & Automation*, 2001

[17] Morales, Antonio ; Sanz, Pedro J. ; Pobil, Angel P.: Vision-Based Computation of Three-Finger Grasps on Unknown Planar Objects. In: *Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2002

[18] Mulmuley, K.: *Computational Geometry*. Prentice Hall, 1994

[19] Murase, Hiroshi ; Nayar, Shree K.: Visual Learning and Recognition of 3-D Objects from Appearance. In: *International Journal of Computer Vision* Bd. 14, 1995, S. 5–24

[20] Pentz, Christian v.: *Lokalisation von Objekten für die Hand-Auge-Koordination*, TU München, Diplomarbeit, 1997

[21] Priese, Lutz: Vergleich von Farbsegmentierungstechniken / Universität Koblenz-Landau, Abt. Koblenz. 1998. – Forschungsbericht

[22] Schlegl, Thomas ; Buss, Martin ; Omata, Toru ; Schmidt, Günther: Fast Dextrous Regrasping with Optimal Contact Forces and Contact Sensor-Based Impedance Control. In: *Proceedings of the 2001 IEEE International Conference on Robotics & Automation*, 2001

[23] Siebel, Nils ; Lang, Oliver ; Wirth, Fabian ; Gräser, Axel: Robuste Positionierung eines Roboters mittels Visual Servoing unter Verwendung einer Trust-Region-Methode / Zentrum für Technomathematik und Institut für Automatisierungstechnik der Universität Bremen. – Forschungsbericht

[24] Smith, Lindsay I. *A tutorial on Principal Component Analysis*. Feb. 2002

[25] Sonka, M. ; Hlavac, V. ; Boyle, R.: *Image Processing, Analysis and Machine Vision, Second Edition*. ITP, 1998

[26] Stappen, A. F. d. ; Wentik, Chantal ; Overmars, Mark H.: Computing Immobilizing Grasps of Polygonal Parts / Department of Computer Science, Utrecht University. – Forschungsbericht

[27] Sun, Wei: 308-766A Shape Analysis in Computer Vision Final Project Report: Face Recognition. (1998)

[28] Turk, M. ; Pentland, A.: Eigenfaces for recognition. In: *Journal of Cognitive Neuroscience* Bd. 3(1), 1991, S. 71–86

[29] Wunderling, Roland: *Paralleler und objektorientierter Simplex-Algorithmus*, Technische Universität Berlin, Diplomarbeit, 1996

[30] Zhang, Jun: Face Recognition: Eigenface, Elastic Matching and Neural Nets. In: *Proceedings of the IEEE, vol. 85, No. 9*, 1997

[31] Zimbardo, Philip G. ; Gerrig, Richard J.: *Psychologie, 7. Auflage.* Pearson Studium, 2001