

Visuelle Erkennung erfolgreicher Greifversuche für die mobile Robotik mittels Pattern-Matching-Algorithmus

DIPLOMARBEIT

Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines
Diplom-Ingenieurs (Dipl.-Ing.)

unter der Leitung von

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Markus VINCZE
Dipl.-Ing. Mag. Dr.techn. David FISCHINGER

eingereicht an der

Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik
Institut für Automatisierungs- und Regelungstechnik

von

Christopher KÜNZEL

Matrikelnummer: 0526486

Mantlergasse 13

1130 Wien, Österreich

Wien, im Oktober 2015

Gruppe Vision für Robotics (V4R)

Gusshausstrasse 27-29, A-1040 Wien, Internet: <http://www.acin.tuwien.ac.at>

Schon seit meiner Kindheit war ich, wie viele Andere, von Robotern fasziniert. Zu dieser Faszination ist nach einiger Zeit reges Interesse an dem „Inneren“ dieser Maschinen aufgekommen, welches die Wahl meines Studiums deutlich beeinflusst hat. Beginnend mit der Aktorik hat sich das Augenmerk deutlich in Richtung Sensorik verschoben. Dem zufolge hatte ich seit dem Ende meines Bachelor-Studiums an der Technischen Universität Wien das Bestreben meine Diplomarbeit im Bereich der Robotik zu schreiben. Das Thema dieser Diplomarbeit beschäftigt sich mit einem Teilbereich der (mobilen) Robotik, der Greifererkennung.

Diese Arbeit wäre jedoch nicht ohne die Unterstützung und Hilfe einiger Personen entstanden, die ich hier kurz erwähnen möchte. Zu aller erst möchte ich mich für die fachliche Unterstützung und gute Betreuung bei der gesamten Vision-for-Robotics-Gruppe bedanken. Vor allem meine Betreuer, Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Markus VINCZE und Dipl.-Ing. Mag. Dr.techn. David FISCHINGER, will ich hier herausstreichen. Durch zahlreiche Diskussionen, Tipps und – nicht zuletzt – Verbesserungsvorschläge konnte diese Arbeit komplettiert werden. In diesem Zusammenhang möchte ich mich auch bei der Firma Blue Danube Robotics, vor allem bei Herrn Dipl.-Ing. Dr.techn. Walter WOHLKINGER, für die Unterstützung und Zusammenarbeit bedanken. Meinen Studien- und Arbeitskollegen will ich ebenfalls für die guten Ratschläge danken, die mir unterschiedliche Sichtweisen näher gebracht haben. Zu guter letzt möchte ich mich für die enorme Unterstützung meiner Familie, während meines gesamten Studiums, bedanken. Diese Unterstützung diente mir stets als gute Basis für meine Studienzeit.

Wien, 2014

Christopher Künzel

In der heutigen Zeit finden Roboter in den verschiedensten Bereichen unseres Lebens ihre Anwendung. Sei es beispielsweise ein massiver Industrieroboter in einer Fertigungsstraße, welcher Karosserieteile bearbeitet, ein Portalroboter, welcher Sortieraufgaben für Objekte durchführt, ein militärisch eingesetzter Roboter, welcher beispielsweise ein Minenfeld räumen soll oder der Rasenmäroboter des Nachbarn. Die Robotik beschäftigt sich mit Themen verschiedenster Disziplinen.

Diese Diplomarbeit ist einem Unterthema der Robotik gewidmet, nämlich der Unterstützung von älteren Menschen in ihrem Alltag durch einen mobilen Roboter namens Hobbit. In diesem Sinne ist es das Ziel dieser Arbeit einen Weg zu finden um die Erkennung eines erfolgreichen Greifversuches zu beurteilen. Diese soll dabei rein auf optischen Informationen, in diesem Fall die Daten einer Microsoft Kinect Kamera, beruhen.

Im Laufe der Arbeit werden mehrere Lösungsansätze präsentiert, wobei jener Ansatz der Kantenerkennung, des Bereichsvergleichs mittels *SIFT*¹ und der Erkennung von *AR-Markern*² implementiert und experimentell bewertet wurde. Für den planaren Greifer des Hobbit hat sich die Variante der Greifererkennung mit Hilfe der AR-Marker als besonders effektiv herauskristallisiert, da diese selbst bei kleinen Objekten robust, schnell, einfach zu implementieren und zuverlässig ist. Die Implementierung, welche durch ein *ROS*³-Paket erfolgte, hat sich in der Praxis bewährt und kann für alle planaren Greifer verwendet werden.

¹Scale Invariant Feature Transform [ein größen- und rotationsinvarianter Algorithmus zum Vergleich von Bildmerkmalen] (*SIFT*)

²Augmented Reality [Erweiterte Realität] (*AR*)

³Robot Operating System [ein Betriebssystem für Roboter] (*ROS*) www.ros.org

Nowadays, robots are applicable in various areas of our life. To name some examples: A massive industrial robot, which works in a production line to deal with car body panels, a gantry robot, which sorts different objects, a military robot, which cleans a mine field for instance and last but not least the neighbours' robotic lawn mower. Robotics deal with topics of varied disciplines.

This diploma thesis is dedicated to a subtopic of robotics, namely the assistance of old people in their everyday life by an auxiliary mobile robot named Hobbit. In this context the goal of this thesis is to find a way of evaluating the detection of successful grasping. This has to be accomplished only on the base of visual information, which is provided by the Microsoft Kinect camera.

In the course of the thesis, different approaches of solving this problem will be presented, whereby the edge detection approach, the *SIFT*⁴ approach of comparing regional features and the approach of detection with *AR-Markers*⁵ were implemented. All these solutions were experimentally evaluated. Applying the Hobbits' planar grippers, the option of using AR-markers for grasp detection emerged as especially effective, as it is robust, fast, easy to implement and reliable, even if small objects are being investigated. The implementation, which took place through a *ROS*⁶-package, has proven itself in practice and can be used and ported for all planar grippers.

⁴Scale Invariant Feature Transform (a scale- and rotation invariant algorithm zu compare image features)

⁵Augmented Reality

⁶Robot Operating System (an operating system for robots) www.ros.org

Inhaltsverzeichnis

1. Einleitung	1
1.1. Motivation	2
1.2. Problemstellung	4
1.3. Aufbau der Arbeit	5
2. Das Problem der Greiferkennung	6
2.1. Stand der Technik	6
2.2. Möglichkeiten der Greiferkennung	8
2.2.1. Optisch	8
2.2.2. Mechanisch/Elektrisch	11
2.2.3. Modellbasiert	12
2.2.4. Kombinierte Methoden	13
3. Grundlagen	14
3.1. SIFT	14
3.2. AR-Marker	19
3.3. Optischer Fluss	20
4. Lösungsansätze	22
4.1. 3D-Erkennung des Greifers und/oder des Objektes	22
4.2. Kurven-/Kantenerkennung bzw. Konturerkennung	24
4.3. Flächenerkennung/-berechnung	26
4.4. SIFT-Vergleich	27
4.5. Graphenbasiertes Verfahren	28
4.6. AR-Marker	29
4.7. Optischer Fluss	30
4.8. Modellierung des Greifprozesses	31
5. Hardwaremodifikation	33
5.1. Prototypen	35
5.2. CAD-Modell	37

5.3.	3D-Druck	39
5.4.	Plastische Modellierung	40
6.	Implementierung	43
6.1.	Kantenerkennung	43
6.2.	SIFT	49
6.3.	Punktwolkenanalyse und Verbesserungen	54
6.3.1.	Analyse der Punktwolken	54
6.3.2.	Verbesserungen	57
6.4.	Pattern Matching mit AR-Marker	61
7.	Experimente und Ergebnisse	68
7.1.	Vorbereitung und Überblick	68
7.2.	Experimente des statischen Algorithmus	71
7.2.1.	Umgebungseinflüsse	72
7.2.2.	Objekteinflüsse	76
7.3.	Ablauf des dynamischen Algorithmus	81
7.3.1.	Algorithmus mit relativer Bewegung	81
7.3.2.	Algorithmus mit absoluter Bewegung	88
7.3.3.	Algorithmusvalidierung und -tests	91
8.	Schlussfolgerung und Ausblick	98
A.	Programmcode ROS package	100
A.1.	checkgrasp.py	100
A.2.	checkMarkerName.py	103
A.3.	checkSub.py	104
A.4.	startClient.py	105
A.5.	startup.launch	106

Abbildungsverzeichnis

1.1. Übersicht Greifproblematik	3
1.2. Übersicht Hobbit	5
2.1. Positionsbasiertes Visual Servoing	9
2.2. Bildbasiertes Visual Servoing	9
3.1. SIFT-DoG Funktionen und Nachbarschaftsvergleich	15
3.2. SIFT-Deskriptorbestimmung	16
3.3. SIFT-Beispiel	18
4.1. Verformung des Greifers und Krümmungsänderung	25
5.1. Festo-Greifer	34
5.2. Greiferzustände	34
5.3. Durch Objekte verdeckter Greifer	35
5.4. Prototyp Überzug V1	36
5.5. Gussform mit Gipsabdruck	37
5.6. Prototyp Überzug V2	38
5.7. CAD-Modell V1	38
5.8. CAD-Modell V2	40
5.9. 3D-Druck der Modelle V1 und V2	41
5.10. Prototyp Überzug V3	42
6.1. Kantendetektion-Greifererkennung in Bodennähe vs. Tischnähe	45
6.2. Parametereinfluss auf die Kantenerkennung	46
6.3. Größeneinfluss auf die Kantenerkennung	47
6.4. Bewegungseinfluss auf die Kantenerkennung	48
6.5. SIFT-Vergleich Greifer in Bodennähe	51
6.6. SIFT-Vergleich Greifer in Tischnähe	53
6.7. Punktwolkenanalyse mit Box, ohne Überzug	54
6.8. Punktwolkenanalyse offener Greifer, ohne Überzug	55
6.9. Punktwolkenanalyse gedrehter Greifer mit Box, ohne Überzug	56

Abbildungsverzeichnis

6.10. Punktwolkenanalyse der Seitenansicht mit Box, ohne Überzug	56
6.11. Punktwolkenanalyse offener Greifer mit Überzug V1	57
6.12. Punktwolkenanalyse offener Greifer ohne vs. mit Überzug V1	58
6.13. Punktwolkenanalyse mit Box, mit Überzug V2	58
6.14. Punktwolkenanalyse mit Objekten, mit Überzug V2	59
6.15. Flächennormalen der Punktwolke des überzogenen Greifers	60
6.16. Canny-Kantenerkennung des Greifers mit Überzug V1	61
6.17. Quadratische Test-Patterns	62
6.18. Rechteckige Test-Patterns	62
6.19. Matching der Testpatterns mittel AR-Marker	63
6.20. V4R Pattern und TU Pattern	63
6.21. Patternerkennung bei zerschnittenem Pattern	64
6.22. Patternmatching bei geöffnetem Greifer	65
6.23. Abgrenzungsprobleme	66
6.24. Verwendete Patterns	67
7.1. Kontrollausgabe AR-Marker-Erkennung	70
7.2. Probleme beim Schließen des Greifers	71
7.3. Greiferansicht aus Hobbitperspektive	71
7.4. Markererkennung mit ersten Testobjekten	72
7.5. Probleme mit Spiegelungen am Pattern	73
7.6. Beleuchtungsprobleme	73
7.7. Markererkennung bei Schrägsicht des Greifers mit Objekt	75
7.8. Greifererkennung eines Bleistifts bei Draufsicht	77
7.9. Erkennung eines USB-Sticks bei Draufsicht	77
7.10. Analyse der Greifererkennung eines Bleistifts	78
7.11. Ablaufdiagramm Greifererkennung V1.0	84
7.12. Ausschnitte bei geringer Rotation des geschlossenen Greifers	85
7.13. Ausschnitte bei starker Rotation des geschlossenen Greifers	85
7.14. Ausschnitte bei mittlerer Rotation des geschlossenen Greifers	86
7.15. Ausschnitte bei Rotation des Greifers mit Objekt (Dose)	87
7.16. Ausschnitte bei Rotation des Greifers mit Objekt (USB-Stick-liegend) .	88
7.17. Ausschnitte bei Rotation des Greifers mit Objekt (USB-Stick-stehend)	88
7.18. Ablaufdiagramm Greifererkennung V2.0	90
7.19. Analyseposition Greifszenario in Bodennähe	92
7.20. Erste Analyseposition Greifszenario in Tischnähe	93
7.21. Analyse der Bleistifterkennung mit verschiedenen Analysepositionen in Tischnähe	94
7.22. Probleme bei Endposition in Tischnähe	95
7.23. Fehlersimulation beim Greifszenario Tisch	96

Tabellenverzeichnis

6.1. SIFT-Vergleich in bodennaher Position	50
6.2. SIFT-Vergleich in tischnaher Position	52
7.1. Ergebnissübersicht der Experimente	97

ACIN

Automation and Control Institute [Institut für Automatisierungs- und Regelungstechnik] 23

AR

Augmented Reality [Erweiterte Realität] ii

CAD

Computer Aided Design [Computerunterstütztes Design] 22

CAN

Controller Area Network [Ein Feldbussystem, welches beispielsweise in der Automobilindustrie verwendet wird] 4

CCD

Charge Coupled Device [Halbleiterbauelemente, die mit dem Photoeffekt arbeiten und beispielsweise als Bildsensor in Kameras verwendet werden] 12

CPU

Central Processing Unit [Zentralrecheneinheit] 29

CUDA

Compute Unified Device Architecture [eine Programmier Technik, die von der Firma NVIDIA entwickelt wurde um Programmteile auf die Grafikkarte auszulagern und dort parallel abzuarbeiten] 29

DoF

Degree of Freedom [Freiheitsgrad] 4

DoG

Difference of Gaussian [die Differenz zweier Gaussfunktionen] 14

GPU

Graphics Processing Unit [Grafikprozessor] 29

HMD

Head-Mounted-Display [ein Display welches direkt vor den Augen eines Menschen angebracht ist, ähnlich einer Brille; in dieser Anwendung ist es durchsichtig, sodass darauf Informationen projiziert werden können] 19

LoG

Laplacian of Gaussian [die Laplace-Ableitung der Gaussfunktion] 43

PCL

Point Cloud Library [eine Softwarebibliothek zur Verarbeitung von Punktwolken]
4

RGB-D

red-green-blue-depth [Rot-Grün-Blau-Tiefeninformation] 4

ROS

Robot Operating System [ein Betriebssystem für Roboter] ii

SIFT

Scale Invariant Feature Transform [ein größen- und rotationsinvarianter Algorithmus zum Vergleich von Bildmerkmalen] ii

TU Wien

Technische Universität Wien 23

USB

Universal Serial Bus [Universeller serieller Bus zur Kommunikation] 74

V4R

Vision for Robotics [Eine Arbeitsgruppe des ACIN, die sich mit maschinellern Sehen und Ähnlichem beschäftigt] 62

KAPITEL 1

Einleitung

Seit einigen Jahren gewinnt die mobile Robotik sowohl im privaten als auch im industriellen Bereich an Bedeutung. Seien es autonome Roboter, die in Lagerhallen ihre Arbeit verrichten oder Haushaltsroboter, die den Menschen den Alltag erleichtern, der Trend zur Entwicklung des „unterstützenden Roboters“ hält an und fordert technische Innovationen.

Für die Weiterentwicklung in der mobilen Robotik wird die „Intelligenz“ der Roboter immer notwendiger! Ein einfacher, dummer Staubsaugerroboter, der seine Bahnplanung nicht durch Einbeziehen von optischen Sensordaten verbessern kann, steht somit am Anfang der Roboterevolution. Im Vergleich dazu sollen Roboter entwickelt werden bzw. werden Roboter entwickelt, die Arbeitsvorgänge beschleunigen, Fehlerfälle reduzieren, Präzision und Effektivität verbessern, Kosten einsparen oder in Tätigkeitsfeldern arbeiten, wo der Mensch nur eingeschränkt oder gar nicht agieren kann. Hierzu zählen Arbeiten unter extremen Bedingungen wie hohem Umgebungsdruck, hoher Temperatur, chemischer Belastung, das Arbeiten mit großen Lasten, aber auch die hochpräzise Arbeit mit Endeffektoren, Lasern, uvm. Roboter finden in verschiedenen Disziplinen, unter anderem der Medizintechnik, Industrie bzw. Fertigung, Logistik, im Haushalt und in der Weltraumtechnik, ihre Anwendung.

In welchem Bereich auch immer ein Roboter eingesetzt wird, es darf nicht vergessen werden, dass viele Lösungsansätze für Problemstellungen, die durch einen Roboter gelöst werden sollen, vom Menschen abgeschaut und für die Robotik adaptiert werden. Die jahrelange Erfahrung eines (erwachsenen) Menschen dient hierfür oft als Vorlage. Verhaltensregeln und Lösungsansätze werden aber nicht nur vom Menschen kopiert oder imitiert, sondern sowohl die gesamte Fauna wie auch Flora dienen als Vorbild für Roboter. Aktuell wird dieser Ansatz stark von der Firma *Festo Ges.m.b.H.*, mit dem *Bionic Learning Network*, verfolgt. Da Erfahrung stark mit Lernfähigkeit in Verbindung steht, wird der Intention nachgegangen lernfähige Maschinen zu entwickeln.

In dieser Arbeit wird nur ein Teilbereich der mobilen Robotik herausgegriffen nämlich jener der den Menschen das Leben im Alltag (auch bei zunehmendem Alter) erleichtern

1. Einleitung

soll. Das *Hobbit*-Projekt der Technischen Universität Wien ist nur ein namhaftes Projekt, das die Entwicklung in diesem Bereich vorantreibt. Diese Diplomarbeit behandelt einen Teil dieses Projektes, nämlich das Erkennen von erfolgreichen Greifversuchen.

1.1. Motivation

In der mobilen Robotik spielt das Greifen von Objekten eine zentrale Rolle, vor allem, wenn der zuvor genannte Bereich der Betreuung und Unterstützung von älteren Menschen bedacht wird.

Damit jedoch ein Roboter auf einen Befehl eines Menschen, wie zum Beispiel „Bring mir einen Apfel aus der Küche!“, mit dem gewünschten Ergebnis reagieren kann, müssen einige Schritte getan und Probleme gelöst werden. Diese einfache Aufgabe dient jedoch nur der Veranschaulichung eines Problems und soll nicht von der allgemeinen Sicht ablenken.

Die erfolgreiche Greifererkennung findet in mehreren Bereichen der Industrie ihre Anwendung. Als weiteres Beispiel dient ein Industrieroboter in einer Fertigungsstraße, welcher Bauteile oder Objekte verladen oder platzieren muss. Auch hierfür muss das Objekt erst erkannt und ergriffen werden.

Bleiben wir jedoch bei der vereinfachten Anschauung anhand der Aufgabe „Bring mir den Apfel aus der Küche!“. Die Aufgabenkette fängt bei der Spracherkennung und Aufgabenerfassung an. Der Roboter muss sozusagen „verstehen“, was seine Aufgabe ist. Anschließend muss die Navigation durch einen Raum bzw. durch die ganze Wohnung erfolgen, um an den gewünschten Ort, die Küche, zu kommen. Sind auch die Probleme des Findens eines Apfels und des Ergreifens erledigt – hierzu gehört die Berechnung passender Angriffspunkte –, wird das eigentliche Problem dieser Arbeit sichtbar. Die Erkennung ob der Apfel wirklich ergriffen wurde und zum Menschen gebracht werden kann. Da dieses Problem für das Hobbit-Projekt eine Rolle spielt, findet die Entwicklung des Lösungsalgorithmus auf der Hobbit-Plattform statt. Näheres hierzu ist im nächsten Unterkapitel zu finden. Die komplette Problemstellung ist vereinfacht in 1.1 dargestellt.

1. Einleitung

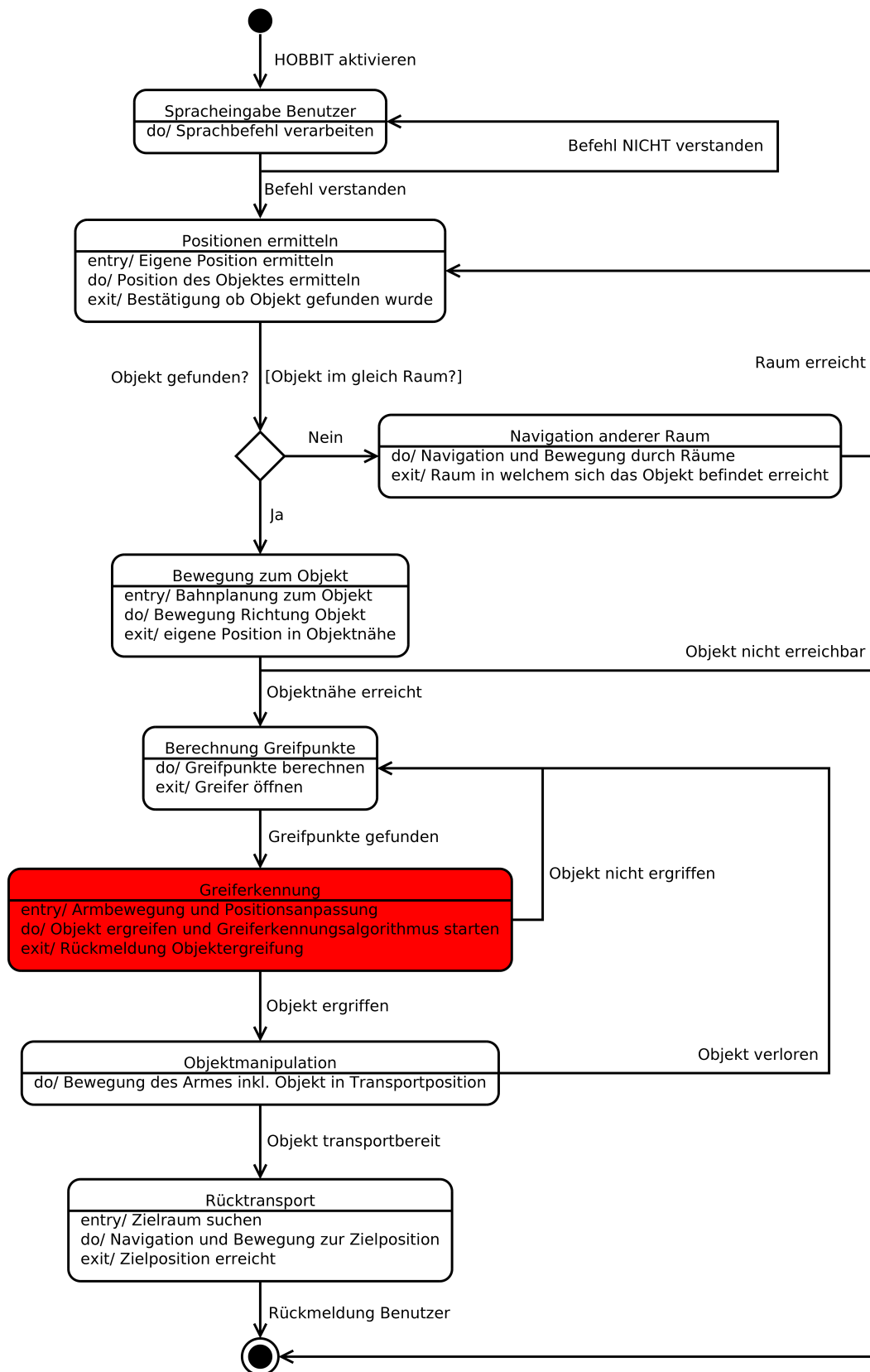


Abbildung 1.1.: Simple Übersicht über den Ablauf der Objektergreifung

1.2. Problemstellung

Wie schon in Abschnitt 1.1 kurz umrissen gibt es einige Vorarbeit zu leisten, bevor das eigentliche Problem behandelt werden kann. Diese Themen, wie etwa die Wegfindung oder die Lokalisation des gesuchten Objektes, werden in dieser Arbeit nicht behandelt.

Es existiert eine Vielzahl von Möglichkeiten, um das Ergreifen eines Objektes sicherzustellen. Diese werden in Kapitel 2.2 behandelt. Die zugrunde liegende Problematik wird in diesem Kapitel näher beleuchtet. Auf die Robustheit und Leistungsfähigkeit der Erkennung soll allerdings schon hier verwiesen werden. Es muss – möglichst unabhängig von Einflussfaktoren wie der Beleuchtung, der Position und der Größe des Objektes – erkannt werden, ob sich tatsächlich ein Objekt im Endeffektor (dem Greifer) des Roboters befindet. Ohne sicherzustellen, dass das gesuchte Objekt erfasst wurde, hat dessen Ergreifung bzw. Transport kaum einen (keinen) Sinn.

Das Ziel dieser Arbeit ist es diese Greifererkennung rein optisch zu bewerkstelligen. Dazu steht eine Kamera im Kopf des Hobbit-Roboters zur Verfügung, die sowohl über einen 2D Sensor, als auch über einen Sensor für die Tiefeninformation, verfügt. Es können somit RGB-D¹ Daten zur Problemlösung herangezogen werden. Diese werden mittels PCL² als Punktwolke gespeichert bzw. weiterverarbeitet. Die Erkennung sollte möglichst universell anwendbar sein, der Fokus liegt jedoch eindeutig auf einer Lösung für das Hobbit-Projekt und ist somit vorrangig. Es müssen verschiedene Greifpositionen und (Selbst-)Okklusion³ berücksichtigt werden. Im idealen Fall sollte der Manipulator nicht bewegt werden. Die Lösung dieses Problems muss mittels eines ROS Paketes in den Programmablauf des Hobbit implementiert werden.

Um die genaueren Umstände zu erfassen, sei kurz auf den Hobbit selbst, im Speziellen auf dessen Arm eingegangen werden. Das Hobbit-Projekt ist ein länderübergreifendes Projekt mit mehreren Kooperationen, an dem sich das Institut für Automatisierungs- und Regelungstechnik stark beteiligt. Das Projekt wurde im Rahmen des „EU's 7th Framework Programme“ (FP7) ins Leben gerufen. Nähere Informationen zum Projekt können unter [1] gefunden werden. Der Hobbit selbst ist ein autonomer Roboter, welcher im Bereich der Betreuung älterer Menschen eingesetzt werden soll. Der Aufbau des Hobbit soll übersichtsmäßig in Abbildung 1.2 dargestellt sein.

Es wird hier nur auf die, für die Arbeit relevanten, Teile des Hobbit eingegangen. Am Kopf des Roboters befindet sich eine Kamera, die Microsoft Kinect, welche zur visuellen Erfassung der Umgebung dient. Der Arm besitzt fünf Freiheitsgrade (5 DOFs⁴). Die einzelnen Achsen werden über Seilzüge bewegt und werden mittels CAN-Botschaften⁵ gesteuert. An der Spitze des Armes befindet sich der Greifer, welcher über einen Elektromotor bewegt werden kann. Dieser befindet sich im Ausleger der Achse 5 des Hobbit. Wird der Arm des Roboters mit dem eines Menschen verglichen, können die Achsen 1 und 2 als eingeschränktes Schultergelenk, die Achse 3 als Oberarm, die Achse 4 als Ellenbogen und die Achse 5 als Unterarm angesehen werden. Die Ansteuerung der

¹red-green-blue-depth [Rot-Grün-Blau-Tiefeninformation] (RGB-D)

²Point Cloud Library [eine Softwarebibliothek zur Verarbeitung von Punktwolken] (PCL) - point-clouds.org

³Verdeckung

⁴Degree of Freedom [Freiheitsgrad] (DoF)

⁵Controller Area Network [Ein Feldbussystem, welches beispielsweise in der Automobilindustrie verwendet wird] (CAN) Controller Area Network

1. Einleitung

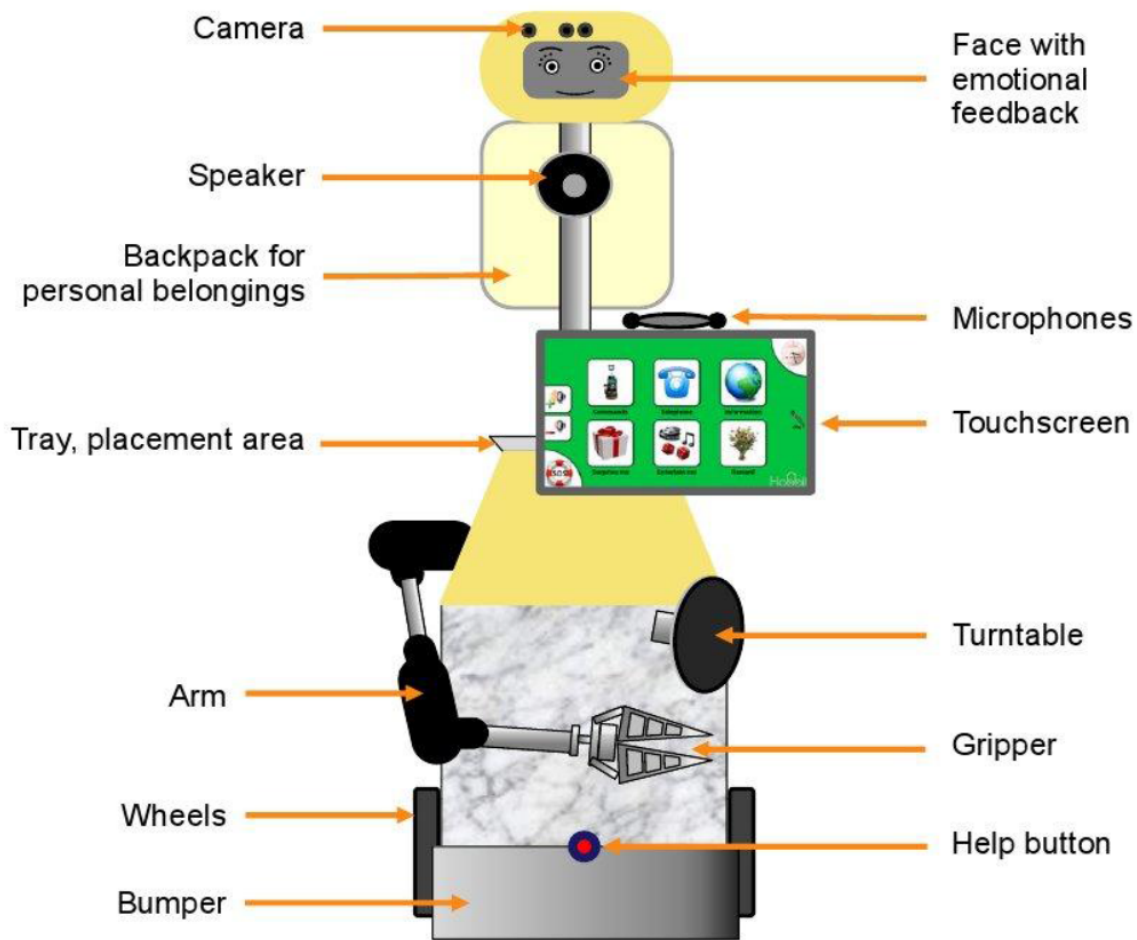


Abbildung 1.2.: Übersicht über die Funktion und den Aufbau des Hobbit-Roboters (Quelle [17])

einzelnen Achsen kann entweder mittels Tablet oder Zentralrechner des Hobbit erfolgen. Weitere Informationen über den Hobbit können unter [13] gefunden werden.

1.3. Aufbau der Arbeit

Um einen allgemeinen Überblick über die Problematik des Greifens bzw. der Greiferkennung zu bekommen, werden die prinzipiellen Möglichkeiten der Analyse des Greifvorganges in Kapitel 2 dargestellt. Im darauf folgenden Kapitel sollen die notwendigen Grundlagen erklärt werden, um die in Kapitel 4 aufgelisteten Lösungsansätze zu verstehen. Um die Funktionalität einiger Lösungsansätze sicherzustellen musste eine Hardwareänderung des Greifers durchgeführt werden. Diese ist in Kapitel 5 erklärt. Die Implementierung der Lösungsansätze und die daraus folgenden Erkenntnisse sind in Kapitel 6 und Kapitel 7 dargestellt. Kapitel 8 stellt ein Resümee und einen Ausblick auf zukünftige Projekte bereit.

Das Problem der Greiferkennung

In diesem Abschnitt wird zuerst kurz auf einige Details und Informationen eingegangen, die das Problem des Greifens und Ergreifens näher behandeln. Danach werden die Möglichkeiten dargestellt, wie ein Greifversuch analysiert bzw. beurteilt werden kann und als (nicht) erfolgreich erkannt werden kann. Die verschiedenen Techniken bzw. Lösungstypen werden mit ihren Vor- und Nachteilen beleuchtet, um einen Überblick zu bekommen.

2.1. Stand der Technik

Bei der Verwendung von Robotern zur Lösung von technischen Problemen gewinnt die Thematik des Greifens immer mehr an Bedeutung. Sei es auf der einen Seite bei industriellen Anwendungen, wie beispielsweise einer *Pick-and-Place*-Aufgabe¹ oder auf der anderen Seite in der mobilen Robotik, die Verwendung von Endeffektoren zum Ergreifen von Objekten ist unumgänglich geworden. Diese Endeffektoren reichen von einfachen Ansaugstutzen über Magnete bis hin zu menschenähnlichen Händen.

Als Inspiration für die Entwicklung von Lösungen für das Greifproblem hat sich der Mensch selbst herauskristallisiert. Anhand zahlreicher Studien wurde das Greifverhalten von Menschen analysiert, um eine Nachstellung, oder zumindest eine Anlehnung, der vom Menschen ausgeführten Prozesse zu schaffen [3]. Beispielsweise wurde die menschliche Bewegungssteuerung zum Ergreifen von Objekten in [19] untersucht. Dabei ist zwischen einfachen (bspw. planaren) und komplizierten (bspw. Greifer mit mehreren Fingern) Greifern zu unterscheiden. Prinzipiell gilt es zu überlegen ob eine komplexe Hand für die jeweilige Aufgabe überhaupt notwendig ist oder auf eine einfachere Variante zurückgegriffen werden kann.

Was aber können Roboter manipulatoren vom Menschen bezüglich des Greifens lernen? Die interessantesten Eigenschaften der menschlichen Hand sind das Erkennen bzw.

¹Nimm ein Objekt und platziere es an einem anderen Ort

2. Das Problem der Greiferkennung

Ertasten, die Abgrenzung von Objekten gegenüber ihrer Umwelt und die Beeinflussung dieser Objekte. Technisch gesehen ist es durchaus möglich einer Roboterhand all diese Fähigkeiten zu verschaffen! Die Frage ist bloß, ob die Anwendung den technischen und finanziellen Aufwand rechtfertigt.

Zu Beginn sollen ein paar Begriffe erklärt werden, die einen Einblick in die Greiffunktion und den damit verbundenen Problemen gewährt. Wird ein Objekt beispielsweise von einem Menschen ergriffen, ist eine wichtige Information die Art des Kontaktes zwischen dem Objekt und der Hand. Der Mensch erkennt alleine anhand der Oberfläche des Objektes und den Kontaktpunkten meist schon ob dieser Kontakt geeignet ist, um ein Objekt hochzuheben. Für Endeffektoren gibt es einige Definitionen für Kontaktarten und Grifftypen, die im Folgenden kurz erläutert werden. Ein reibungsloser Kontakt ist ein Kontakt zwischen Finger und Objekt an dem durch den Greifmanipulator (bspw. den Finger) rein eine Normalkraft entlang des Greifkontaktes aufgebracht werden kann. Bei einem Reibkontakt kann zusätzlich eine tangentielle Kraft aufgebracht werden. Die höchste Form eines Greifkontaktes ist der weiche bzw. elastische Kontakt, bei dem außer einer Normal- und einer Tangentialkraft ein Drehmoment von den Fingern ausgeübt werden kann.

Des Weiteren wurden verschiedene Unterscheidungen hinsichtlich der Art des Greifens, dem umschließenden Greifen oder dem Greifen mit Fingerspitzen und bezüglich der Griffart, einem formschlüssigen² oder einem kraftschlüssigen³ Griff gemacht. Unter einem kraftschlüssigen Griff wird das Gleichgewicht zwischen externen Einflüssen, beispielsweise Stößen, und der Handkraft verstanden. Ein formschlüssiger Kontakt lässt sich als kraftschlüssiger Griff mit reibungslosen Kontakten definieren [3]. Werden diese Definitionen auf den Hobbit-Greifer umgelegt, kann von kraftschlüssigen Kontakten gesprochen werden. Obwohl der Greifer selbst nur eine Normalkraft auf die Kontaktpunkte aufbringen kann, kann auf Grund der Oberfläche der Gelpads, welche sich an der Innenseite der Greiferfinger befinden, auf einen Reibkontakt geschlossen werden.

Diese Definitionen werden verwendet, um die Stabilität eines Griffes, also dessen Qualität, zu charakterisieren. Dabei kommen Berechnungen anhand von Modellen und Ähnlichem zur Anwendung. Ein stabiler Griff kann beispielsweise durch die Berechnung der Definitheit der Steifigkeitsmatrix nachgewiesen werden [3].

Bei der Verwendung von modellbasierter Greiferkennung (siehe Kapitel 2.2.3) spielt der mathematische Beweis der Stabilität eines Griffes eine zentrale Rolle. Diese ist jedoch noch von weiteren Parametern, wie beispielsweise der Nachgiebigkeit der Finger und der Oberflächenbeschaffenheit, etc., abhängig. Auch wenn diese Definitionen eine mathematische Behandlung von Kontakt- und Greifproblemen zulässt, wird sehr oft auf einfachere Methoden zurückgegriffen.

Selbst wenn die menschliche Hand als Vorbild für Robotergreifer gesehen wird, kann deren Funktion nur mit hohem Aufwand angenähert werden. Der Mensch kann bei einem Greifprozess auf unterschiedliche Sensoren zurückgreifen. Würden für einen Robotergreifer all diese Sensoren (haptisch und optisch) in einer ungefähr gleichen Qualität zur Verfügung gestellt werden, würde das die Kosten des Roboters extrem steigern. Die Kosten müssen in Relation zur gestellten Komplexität der Aufgabe betrachtet werden und mit denen der Funktionalität abgewogen werden. Um Projektkosten zu

²form-closure

³force-closure

2. Das Problem der Greiferkennung

sparen werden Abstriche bei den Sensoren und bei den Greifalgorithmen gemacht. In industriellen Anwendungen ist es beispielsweise oft ausreichend die Greifkontakte von (meist vordefinierten) Objekten offline zu berechnen und die visuelle Erkennung rein auf die Berechnung der Orientierung des Werkstückes zu beschränken [43].

Um zwischen den anwendungsspezifischen Lösungen des Greifproblems unterscheiden zu können, werden die Möglichkeiten der Greiferkennung in drei Kategorien unterteilt. Erwähnt soll dabei sein, dass sich die Grenzen zwischen diesen Kategorien auf die Erkennung eines (nicht) erfolgreichen Greifversuches beziehen und nicht auf den kompletten Greifvorgang. Beispielsweise ist für die modellbasierte Greiferkennung zumindest anfangs ein optischer Sensor notwendig, um das Modell zu errechnen. Da die eigentliche Erkennung jedoch aufgrund des Modells und nicht aufgrund des optischen Sensors erfolgt, ist die Kategorisierung eindeutig.

2.2. Möglichkeiten der Greiferkennung

Wie zuvor erwähnt, sollen in diesem Kapitel die Möglichkeiten der Erkennung eines (nicht) erfolgreichen Greifversuches zusammengefasst werden. Die einzelnen Arten der Greiferkennung wurden anhand der ausschlaggebenden Erkennungsart – optisch, mechanisch oder modellbasiert – kategorisiert. Es wird außerdem auf die Voraussetzungen, Vor- und Nachteile der einzelnen Verfahren eingegangen. Wichtig anzumerken ist, dass die Art der Objekterkennung und die der Greiferkennung unterschiedlich sein kann, aber nicht sein muss.

2.2.1. Optisch

Wie schon zuvor erwähnt ist für jedes Greifsystem in der mobilen Robotik ein optischer Sensor notwendig. Ein Gegenstand muss zumindest visuell erkannt oder gefunden werden um mit dem Greifvorgang beginnen zu können. Bei Roboterarmen in der Industrie muss dies beispielsweise nicht der Fall sein, da die Positionierung des Endeffektors anhand von vordefinierten Koordinatenpunkten erfolgen kann. Um auf das Thema der Adaption von menschlichen Abläufen zurückzukommen ist es interessant, dass die wichtigste Ressource des Menschen für das Greifen von Objekten die 3D-Daten sind und nicht die 2D-Projektionen der räumlichen Szenen [19]! Es werden jedoch nicht nur die visuellen Charakterisierungsvorgänge, sondern auch die haptischen Eigenschaften des Menschen von der mobilen Robotik nachempfunden, vor allem wenn es darum geht Objekte zu ertasten, von der Umwelt abzugrenzen und zu beeinflussen [3].

Ein eigenständiges System – ein autonomer Roboter – kann ein Objekt nur dann richtig und effizient greifen, wenn das Objekt zumindest einmal visuell erfasst wurde. Für die Erkennung eines (nicht) erfolgreichen Greifversuches wird allerdings kaum ein rein optisches System verwendet. Es werden zwar häufig Angreifpunkte am Objekt anhand visueller Informationen berechnet, jedoch werden diese nicht zur Entscheidung des erfolgreichen Greifens verwendet. Nichts desto trotz sollen ein paar Beispiele erwähnt werden, die für diese Problemlösung als Vorbild dienen können.

Oft wird das Ergreifen von Objekten im Zusammenhang mit der Aufgabe des *Visual*

2. Das Problem der Greiferkennung

*Servoing*⁴ gelöst und nicht als separates Problem gesehen. Wie in [21] beschrieben ist Visual Servoing der Vorgang bei dem maschinelles Sehen dazu verwendet wird, um den Aktor eines Roboterarmes zu steuern. Die Steuerung erfolgt hierbei durch eine geschlossene Regelschleife, welche über die Daten der Kamera die Position des Greifers mit der gewünschten Endposition abgleicht. Dabei muss zwischen bildbasiertem und positionsbasiertem Visual Servoing unterschieden werden. Beim positionsbasierten Ansatz wird anhand der Kameradaten die Position des Manipulators im Vergleich zur Kamera selbst geschätzt und anhand dieser Daten eine Minimierungsaufgabe für die Positionierung des Armes gelöst. Für die Schätzung der Position sind hierbei ein Umgebungs- und ein Kameramodel notwendig. Im Vergleich dazu benötigt der bildbasierte Ansatz diese Daten nicht. Es wird direkt anhand von Bildmerkmalen die Steuerung des Armes durchgeführt. Dies ist in Abbildung 2.1 und Abbildung 2.2 dargestellt.

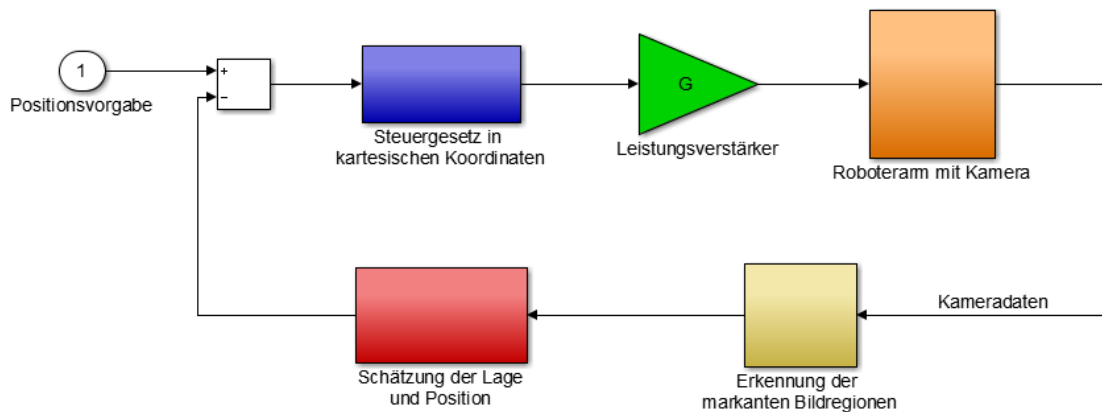


Abbildung 2.1.: Positions-basiertes Visual Servoing nach Quelle [21]

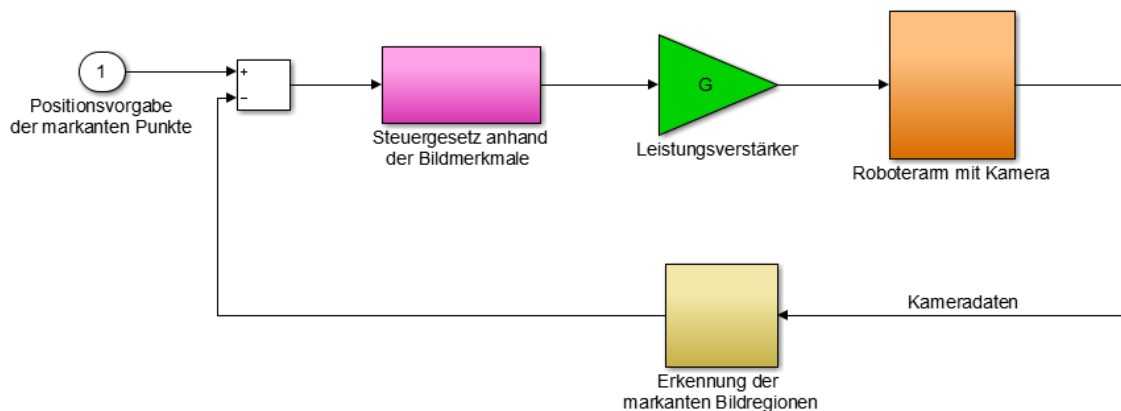


Abbildung 2.2.: Bildbasiertes Visual Servoing nach Quelle [21]

Werden einfache Objekte, wie beispielsweise Bälle, als Versuchsobjekte in Betracht gezogen, kann der Greifprozess mit einem simplen Farberkennungsprozess zur Segmen-

⁴Visuelle Bewegungssteuerung

2. Das Problem der Greiferkennung

tierung, einer Kantendetektion und einem Matching Algorithmus vollzogen werden, wie dies bei [20] durchgeführt wurde. Wie schon hier bemerkt werden kann, spielt die Art und Form der Objekte eine zentrale Rolle. Bei [20] wurde das gesamte, allerdings simple, Greifproblem, welches aus dem Erkennen des Objektes, über die Positionierung des Armes bis hin zum eigentlichen Zugreifen, rein optisch gelöst. Wie bei allen visuellen Systemen ist auch hier die Position und Qualität des Sensors von entscheidender Bedeutung!

Die Greiferkennung kann auch mit *Shape Tracking*⁵ in Verbindung gebracht werden. Um zu erkennen, ob ein Objekt ergriffen wurde, wird ein Kalmanfilter⁶ für den Verfolgungsalgorithmus verwendet und danach die Bilderkennung mittels Optimierungsproblem gelöst [31]. Dieser Trackingalgorithmus kann verwendet werden, um nach dem Schließen des Greifers das Objekt weiterhin zu verfolgen, während der Arm bewegt wird. Falls sich das Objekt mit dem Arm mitbewegt, kann es als erfolgreich gegriffen kategorisiert werden. Ein großes Problem stellen hierbei jedoch kleine Objekte dar. Dieser Ansatz hat Ähnlichkeit mit der Thematik des optischen Flusses. Dies soll jedoch zu einem späteren Zeitpunkt (siehe Kapitel 3.3) näher beleuchtet werden.

Beim Ergreifen von Objekten stellt sich außerdem immer die Frage, ob eine komplette Analyse der Szene notwendig ist oder ob ein rein heuristisches Verfahren⁷ ausreicht [43]. Werden beispielsweise einige industrielle Anwendungen betrachtet, kann erkannt werden, dass der Greifprozess häufig offline entschieden wird. Dies bedeutet, dass – unter der Voraussetzung, dass die zu ergreifenden Objekte bekannt sind – das Objekt lediglich in der Szene gefunden werden muss und der Greifer in der gewohnten, zuvor festgelegten, Weise agieren muss, um für ein positives Greifergebnis zu sorgen.

Wie [43] zeigt, kann jedoch auch 2D-Information ausreichend sein, um ein gutes Greifergebnis zu erreichen. Dabei werden durch Verarbeitung der Daten einer Kamera, welche am Handgelenk des Greifers montiert ist, markante Punkte entlang der Kontur des Objektes errechnet und als Angreifpunkte markiert. Die Stabilität der Greifpunkte wird anhand der Krümmung, des Reibmodells und anhand des Abstands zwischen der Verbindung der Greifpunkte und des Schwerpunkt des Objektes bestimmt. Der Greifvorgang selbst wird durch visuelles Feedback anhand der Lösung einer Optimierungsaufgabe gesteuert. Alternativ kann die Stabilität von Greifpunkten auch anhand von sogenannten *Attractive Regions*⁸ nachgewiesen werden. Dazu stehen mehrere Varianten zur Verfügung.

Die Vor- und Nachteile der optischen Greiferkennung liegen auf der Hand, wenn allgemein an optische Systeme gedacht wird. Die Abhängigkeit von der Beleuchtung, der Qualität der Kamera, der Oberfläche bzw. des Materials der einzelnen Objekte (Glas, Spiegel, etc.) sind nur einige direkte Nachteile der optischen Greiferkennung. Hinzu kommen die Probleme der (Selbst-)Okklusion. Dem gegenüber steht eine hohe Portabilität der Greiferkennung auf unterschiedliche (Roboter-)Systeme und die einfache Wartbarkeit bei Ausfall des Kamerasystems. Da ein optisches System ohnehin in beinahe jedem Greifprozess zu tragen kommt, muss keine zusätzliche Hardware am Greifer angebracht werden und die Kosten werden reduziert.

⁵Form-/Gestaltverfolgung

⁶Eine Erklärung des Kalmanfilters ist in [28] zu finden

⁷Ein Verfahren, welches eine Näherungslösung findet

⁸Aussagekräftige Bereiche

2.2.2. Mechanisch/Elektrisch

Jede Art der Greiferkennung, die direkt mit dem Kontakt zwischen Objekt und Greifer zu tun hat, wird hier als mechanische Greiferkennung kategorisiert. Da der Kontakt zwischen einem Objekt und dem Greifer jedoch meistens auf elektrischem Wege, mit elektrischen Sensoren, analysiert wird, umfasst diese Kapitel sowohl mechanische als auch elektrische Methoden um einen (erfolgreichen) Greifversuch zu erkennen. Dies vereinfacht die Unterteilung. In den meisten Fällen wird ein Feld aus Sensoren am Greifer angebracht, um den Kontakt mit Objekten zu analysieren. Es werden in der Folge einige Sensoren aufgezählt, die in praktischen Systemen zur Anwendung kommen. Für eine genauere Beleuchtung der verschiedenen Sensortechnologien sei auf die Literatur verwiesen [8].

Neben resistiven, piezoresistiven, piezoelektrischen – einem Spezialfall von resistiven Sensoren – dem Kraft-sensitiven Widerstand [29] und kapazitiven Sensoren werden auch optische Sensoren unter der Roboterhaut zur Detektion von Kräften und Kontaktpunkten verwendet. Etwas seltener vertreten finden Ultraschall-, Magnet- und solche Sensoren, die anhand des Quantentunneleffektes Berührungen analysieren, Einzug in die breite Palette der taktilen Sensoren [8]. Neben Anwendungen von kapazitiver Kraftmessung [50, 18] und optischer Kontaktanalyse [7, 36] werden auch Dehnmessstreifen verwendet [49]. Der größte Vorteil der mechanisch-elektrischen Erkennung liegt vor allem in der hohen Auflösung und der Genauigkeit der Greiferkennung, vorausgesetzt die Signale werden in guter Qualität übertragen. Je nach verwendeter Technik kann dabei sogar ein genaueres „Tastgefühl“ erreicht werden als beim menschlichen Finger! Es ist des Weiteren möglich dynamisch während der Armbewegung die Angreifkontakte zu überwachen, um beispielsweise ein Herausrutschen des Objektes aus dem Greifer frühzeitig zu detektieren. Große Nachteile von Sensorfeldern sind einerseits die aufwändige Implementierung, die schlechte Wartbarkeit bei beschädigten Sensoren und andererseits die Rauschanfälligkeit der Signale.

Die Bestimmung der Position des Roboterarms im Raum anhand der Motordaten, Achs- bzw. Gelenksstellungen mithilfe von inverser Kinematik ist eine weitere Möglichkeit einen Greifversuch zu analysieren. Ist die Position des Objektes bekannt und wurden exakte Angreifpunkte berechnet, kann über diese Informationen der Arm gesteuert werden, um den Endeffektor des Armes an der richtigen Stelle zu positionieren und den Greifvorgang durchzuführen. Die Vorteile sind die sehr einfache Implementierung, da ohnehin die meisten Motoren und Gelenke ihre Positionsinformationen, beispielsweise durch Messung mittels Hall-Sensoren, an den Zentralrechner übertragen.

Bei der Berechnung der Achspositionen anhand der Position des Objektes im Raum – diese Thematik behandelt die inverse Kinematik – ergibt sich das Problem, dass die Lösung der Berechnung der Achswerte oft nicht eindeutig ist. Hinzu kommt der Fakt, dass Messwerte der Gelenke oft ungenau sind, vom Referenzierungsvorgang abhängen und fehleranfällig sind. Außerdem bedarf es eines großen Aufwands um die optimalen Angreifpunkte an einem Objekt zu berechnen.

Ein weiterer großer Nachteil ist, dass alleine durch die richtige Positionierung des Greifers noch lange keine klare Aussage bzgl. des erfolgreichen Greifens getätigt werden kann. Es kann lediglich bestätigt werden, dass der Greifer die gewünschte Position erreicht hat. Kann zusätzlich ein Stabilitätsnachweis der Greifpunkte erbracht werden,

kann sehr wohl von einer erfolgreichen Greifanalyse gesprochen werden. Für einfache *Pick-and-Place*-Aufgaben mit einem Ansaugendeffektor ist diese Art der Greiferkennung, gemeinsam mit dem rückgemessenen Luftstrom im Luftkanal des Endeffektors, jedoch ausreichend.

2.2.3. Modellbasiert

Modellbasierte Ansätze verwenden verschiedene Modelle für das zu ergreifende Objekt, die Umgebung und den verwendeten Endeffektor, um den Greifvorgang, offline oder online, zu modellieren bzw. zu simulieren. Beispielsweise kann ein Objekt mittels umhüllenden Rahmen-Modells⁹ verarbeitet werden. Oft kann sogar die gleiche Modelltype für den Greifer und für das Objekt verwendet werden (bspw. konvexes Rechteckmodell¹⁰). Auch das häufig verwendete Hertz-Modell, welches vor allem für den Kontakt zwischen einer Kugel und einer Fläche verwendet wird, dient zur Modellierung des Kontaktes [15].

Die Greifanalyse findet, wie schon erwähnt, offline und/oder online statt. Bei [15] werden beispielsweise drei Schritte offline ausgeführt, bevor in der Onlinephase die tatsächliche Planung des Greifvorganges stattfindet. Genauer gesagt werden in der Offlinephase die Oberflächen in Cluster¹¹ unterteilt, die Berührungspunkte am Objekt anhand der Bedingung für stabile kraftschlüssige Griffe berechnet und der Annäherungsvektor des Greifers berechnet. In der Onlinephase wird dann die Pose des zu ergreifenden Objektes mittels visuellen Sensors festgestellt und es werden jene Cluster, Berührungspunkte und Annäherungsvektoren gesucht, die eine vorgegebene Optimierungsfunktion minimieren. Danach kann der Greifvorgang durchgeführt werden.

Eine kurze Übersicht über das Greifproblem und Kontakte kann in [3] gefunden werden. Eine ähnliche Vorgangsweise ist in [53] zu finden. Hier wird mittels eines Lasers und CCD¹² Kameras das Objekt gescannt und danach eine virtuelle Objektrekonstruktion durchgeführt. Ist die Rekonstruktion beendet, folgt ebenfalls eine Berechnung und Stabilitätsüberprüfung der Angreifpunkte. Sind diese gefunden, wird der Vorgang des Ergreifens simuliert und eine Bewertung des Vorganges durchgeführt. Nach der Simulation werden jene Greifpunkte für den eigentlichen Steuerungsvorgang verwendet, die das beste Simulationsergebnis lieferten.

Eine weitere Möglichkeit ist die Lageschätzung – die Lage eines Objektes besteht aus dessen Position und Orientierung – anhand eines Modells des Gegenstandes. Diese kann beispielsweise durch ein sogenanntes bilineares Modell erfolgen, bei dem die Gier- und Nickwinkel eines Objektes auf Grund von verschiedenen Objektansichten berechnet bzw. geschätzt werden [37] oder auch mittels einfacher Kantenerkennung. Ist die Lage des Objektes auf Grund des Modelles bekannt, kann ein Optimierungsproblem, in den meisten Fällen ein Minimierungsproblem, in Anlehnung an das Visual Servoing Prinzip erfolgen. Ein Beispiel hierfür ist in [37] wiedergegeben. Allerdings bezieht sich diese Anwendung auf industrielle Förderbänder und müsste für mobile Roboter adaptiert

⁹bounding-box-model

¹⁰convex-rectangular-model

¹¹Kleine Bereiche

¹²Charge Coupled Device [Halbleiterbauelemente, die mit dem Photoeffekt arbeiten und beispielsweise als Bildsensor in Kameras verwendet werden] (CCD)

und erweitert werden!

Ein Nachteil an der Verwendung von Modellen ist der meist enorme Rechenaufwand und die damit verbundene Rechnerleistung. Der Grund dafür liegt vor allem in der notwendigen Exaktheit der möglichst detailgetreuen Abbildung der Realität – der Objekte, des Greifers, der Kontakte – durch die Modelle. Dies führt zu langen Berechnungsvorgängen und starker Belastung des Rechenkernes des Roboters.

Ein weiterer Nachteil ist, dass, nach der Berechnung der Angreifpunkte und Stabilitätskontrolle, der Greifvorgang ohne Rückmeldung von der Umwelt (durch andere Sensoren) stattfindet und sich rein auf die Exaktheit des Modells verlassen wird. Auf unvorhergesehene Änderungen nach der Offline-Phase kann somit nicht mehr reagiert werden [15, 47]. Außerdem ist eine exakte Positionierung des Endeffektors notwendig. Idealerweise sind bei den Berechnungen der Angreifpunkte sowohl die geometrischen Informationen als auch sämtliche Materialparameter des Objektes und auch des Greifers, wie Oberflächenbeschaffenheit, Materialart, etc., mit einzubeziehen. Als wichtige Vorteile seien die Wiederverwendbarkeit von Modellen und die Portabilität erwähnt.

2.2.4. Kombinierte Methoden

Je nach Anwendungsfall werden die einzelnen Möglichkeiten miteinander kombiniert. Dabei sind so gut wie alle Kombinationen möglich! Beispielsweise wird in [51] sowohl ein Stereokamerasystem als auch Messwerte von Kraftsensoren und Motordaten für die Analyse des Greifprozesses verwendet. Für den gesamten Greifprozess, also die Objekterkennung, Greiferpositionierung, Greiferkennung und Objektmanipulation werden jedoch ohnehin meist kombinierte Verfahren angewandt. Da der Mensch sowohl seine optische Wahrnehmung als auch seinen Tastsinn für die Handhabung von Objekten verwendet ist, diese Kombination in der mobilen Robotik ebenfalls eine gute und zuverlässige Variante zur Greiferkennung wie beispielsweise [35] zeigt.

In diesem Kapitel werden drei wichtige Konzepte für maschinelles Sehen erklärt. Dies dient dem besseren Verständnis der Algorithmen und soll die Grundlage für die, in Kapitel 4 vorgestellten, Lösungsansätze sein.

3.1. SIFT

Im Folgenden soll eine Zusammenfassung von [32] erfolgen, die zur näheren Erklärung des SIFT-Algorithmus dient. SIFT ist die Abkürzung für *Scale Invariant Feature Transform*. Der Name lässt sich davon ableiten, dass Bilddaten in größenunabhängige Koordinaten transformiert werden, welche auf Grund von lokalen Merkmalen gefunden wurden. Der reine SIFT-Algorithmus bezieht sich nur auf das Finden der markanten Punkte und die Charakterisierung dieser. Es wird meistens ein zusätzlicher Übereinstimmungsalgorithmus verwendet, um die gefundenen Punkte mit denen in Referenzbildern zu vergleichen. Der Ablauf zum Finden der markanten Punkte, auch Features oder Frames genannt, besteht aus folgenden Teilschritten:

- Detektion der Extremwerte im Skalierungsraum¹:
Der Skalierungsraum ist hierbei nicht als Raum zu verstehen, sondern als kontinuierliche Funktion bzgl. des Maßstabes, welche nach stabilen Merkmalen über sämtliche Größenordnungen von Bildern sucht. Das Ziel der Suche ist es Schlüsselpunkte bzw. Merkmale zu finden, welche durch ihre Position und Größenordnung definiert sind und von verschiedenen Perspektiven des Objektes erkannt werden können. Zu diesem Zweck wird die DoG²-Funktion verwendet. Sie besteht, wie der Name schon sagt, aus der Differenz der Produkte eines Bildes mit unterschiedlichen Gauss-Funktionen. Diese sind variabel skalierbar mittels eines Parameters. Für die mathematische Definition sei auf die Referenzliteratur [32] verwiesen. Diese

¹scale-space

²Difference of Gaussian [die Differenz zweier Gaussfunktionen] (DoG)

3. Grundlagen

Funktion wird auf unterschiedliche Größen des Bildes, eingeteilt in sogenannte Oktaven, angewandt. Um die Extremwerte der DoG-Funktion zu finden wird jeder Bildpunkt mit all seinen Nachbarn, sowohl auf der gleichen, wie auch den beiden benachbarten Skalierungsebenen, verglichen. Der aktuell betrachtete Wert wird nur dann als Minimum oder Maximum gespeichert, wenn er größer bzw. kleiner als alle seine Nachbarn ist. Dies zeigt 3.1.

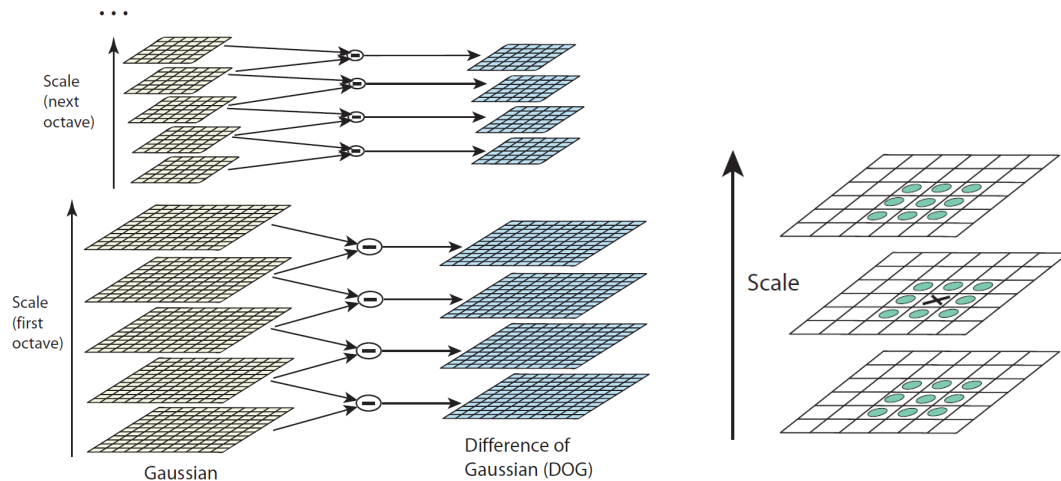


Abbildung 3.1.: Bildung der DoG Funktionen und Nachbarschaftsvergleich nach Quelle [32]. Dabei wird die Gaussfilterung in verschiedenen Skalierungsebenen (Oktaven) durchgeführt und aus jeweils zwei die Differenz gebildet (rechtes Bild). Danach findet ein Vergleich mit den direkten Nachbarn statt. Die direkten Nachbarn befinden sich in der aktuellen sowie in den beiden angrenzenden Oktaven. Daraus resultieren 26 direkte Nachbarn (neun pro angrenzender Oktave und acht auf der eigenen Oktave).

- **Lokalisieren der Schlüsselpunkte:**
Konnten, durch die Extremwertdetektion, Kandidaten für Schlüsselpunkte extrahiert werden, muss ein Vergleich mit der näheren Umgebung durchgeführt werden und anhand eines Schwellwertes eine Selektion der relevanten Kandidaten stattfinden. Bei dieser Analyse kommt es zu einer Filterung hinsichtlich Position, Maßstab und Hauptkrümmung des untersuchten Kandidaten. Dadurch werden weniger markante Extremwerte herausgefiltert sowie Rechenleistung eingespart und Fehler minimiert.
- **Zuweisung der Orientierung:**
In diesem Schritt wird jedem Schlüsselpunkt eine konsistente Orientierung auf Grund seiner lokalen Bildeigenschaften zugewiesen. Der im Späteren definierte Deskriptor³ kann relativ auf diese Orientierung zugeschnitten werden und ermöglicht somit die Unabhängigkeit von Bildrotationen. Es wird für jeden Punkt die Amplitude und Richtung des Gradienten berechnet. Auf Grund dieser Gradienten kann in der Umgebung des Schlüsselpunktes ein Orientierungshistogramm erstellt

³Charakteristischer Wert, welcher als Merkmal für einen Bereich verwendet werden kann (Beispiel: der Umfang eines Objekts)

3. Grundlagen

werden. Spitzenwerte des Histogramms deuten dabei auf eine vorherrschende Richtung der lokalen Gradienten hin. Alle lokalen Spitzenwerte, die mindestens 80% des höchsten Spitzenwertes ausmachen, werden für die Orientierungszuweisung verwendet. Jene, welche darunter liegen, werden aussortiert.

- Bestimmen des Deskriptors für die Schlüsselpunkte:

Da nun sämtliche Parameter in den vorbereitenden Schritten bestimmt wurden, muss ein Deskriptor für den lokalen Bildbereich bzw. Schlüsselpunkt berechnet werden. Dazu wird erneut die Amplitude und Richtung des Gradienten in einem Bereich um die einzelnen Schlüsselpunkte verwendet, welche durch eine Gauss-Fenster-Funktion gewichtet werden. Es entsteht wiederum ein Orientierungshistogramm mit jeweils mehreren orientierten Richtungen. Diese werden als Vektor, dem Feature-Vektor, dargestellt.

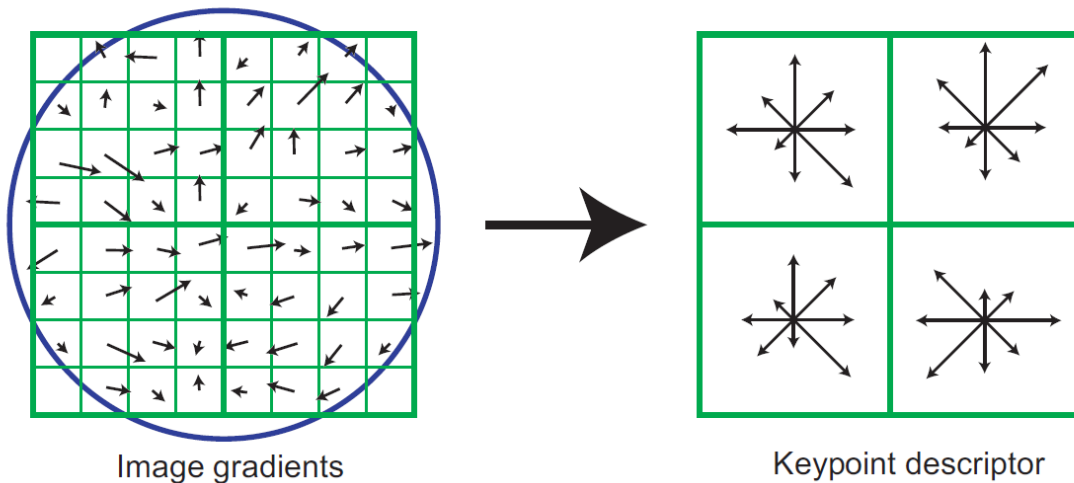


Abbildung 3.2.: Deskriptorbestimmung nach Quelle [32]. Die, im linken Bild dargestellten, Orientierungen und Amplituden der Gradienten werden zu einem Schlüsselpunkt zusammengefasst. Dies wird mittels einer Gauss-Fenster-Funktion durchgeführt.

Damit die gefundenen Schlüsselpunkte in verschiedenen Bildern verglichen werden können, wird folgender Algorithmus zusätzlich durchgeführt:

- Finden des nächsten Nachbarns⁴:

Um die beste Übereinstimmung von allen gespeicherten Kennzeichen mit einem Kennzeichen der aktuellen Szenerie zu finden wird das nearest-neighbour-Prinzip angewandt. Dabei wird jener Schlüsselpunkt mit der minimalen euklidischen Distanz zu den Feature-Vektoren gesucht. Oft werden auch die zweit-nächsten Features hinzugezogen. Da der Vergleich von allen Features notwendig ist, wird ein Suchalgorithmus benötigt. Dazu wird ein k-d-tree⁵ ähnlicher Suchalgorithmus verwendet. Grob gesprochen wird beim k-d-Baum die Umgebung, der k-dimensionale

⁴nearest neighbour, [Nachbarschafts-/Umgebungsvergleich] eine kurze Erklärung und einige Beispiele sind in [48] zu finden.

⁵k-d-Baum, ein mehrdimensionaler Suchbaum

3. Grundlagen

Raum, in Punkte aufgeteilt um die Suche zu beschleunigen. Eine Definition von k-d-Bäumen ist [48] angegeben.

- Gruppierung anhand der Hough-Transformation:
Dieser Schritt dient dazu ein Objekt mit der geringsten möglichen Anzahl an Matches zu erkennen. Die Gruppierung erfolgt anhand der Hough-Transformation. Die Hough-Transformation identifiziert Mengen von Features dadurch, dass jedes Feature eine Art Voting-Prozess für alle möglichen Objektpositionen durchführt. Dabei wird das gleiche Prinzip angewandt, wie beim Finden kollinearere Geradenelemente. Jedes Geradenstück wird parametrisiert, in diesem Fall durch den Normalabstand und den Winkel zwischen der horizontalen Achse und der Normalen durch den Ursprung, und die Werte in den Akkumulator eingetragen. Dieser besitzt die beiden Geradenparameter als Achsenwerte. Es liegt auf der Hand, dass kollineare Geradenstücke auf den gleichen Wert im Akkumulator abgebildet werden. Auf ähnliche Art und Weise können auch zusammengehörige Features gefunden werden. Für eine genauere Beschreibung der Hough-Transformation sei auf [48] verwiesen.
- Übereinstimmungsfindung⁶ anhand der Lösung für kleinste Quadrate⁷:
Die, durch die Hough-Transformation gefundenen, Mengen an ähnlichen Features werden nun mit einem Least-Squares-Verfahren verglichen. Dabei werden die Featuremengen aller Referenzbilder einbezogen. Das Least-Squares-Verfahren ist in [28] erklärt.

In Abbildung 3.3 ist exemplarisch das Ergebnis eines in Matlab implementierten SIFT-Vergleiches dargestellt. Obwohl es zu einigen falschen Matches bzgl. der Umgebung bzw. anderer Objekte kommt, kann das Modellauto eindeutig in der unteren Szene gefunden werden.

⁶Matching

⁷least-squares-solution, Least-Squares-Verfahren [ein Optimierungsverfahren]

3. Grundlagen

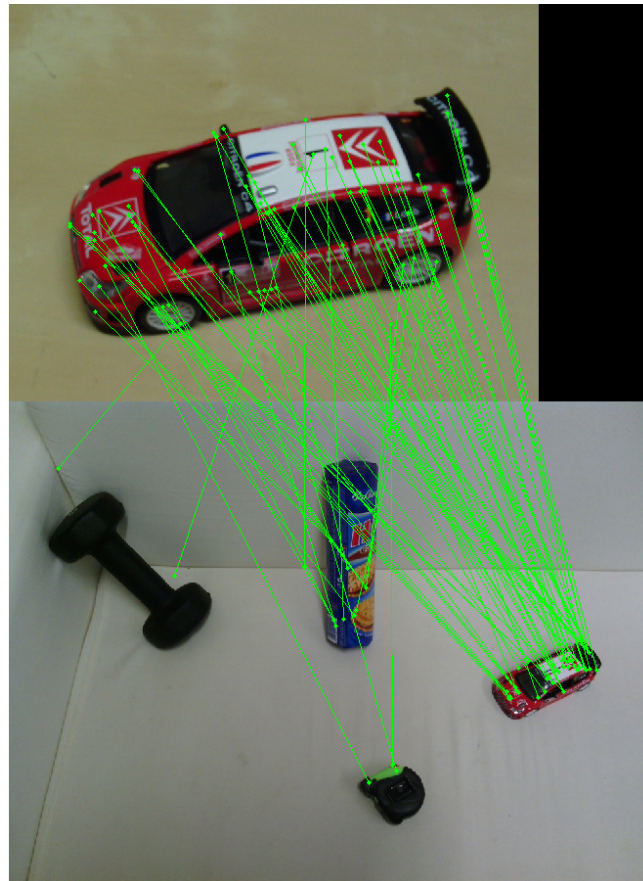


Abbildung 3.3.: SIFT-Vergleich zweier Bilder. Das, im oberen Teil dargestellte, Modellauto wird in der unteren Szene eindeutig gefunden. Obwohl ein paar wenige Übereinstimmungen auf anderen Objekten bzw. in der Umgebung gefunden werden, lässt sich eindeutig bestätigen, dass der SIFT-Algorithmus korrekt arbeitet.

3.2. AR-Marker

Die ursprüngliche Verwendung von AR-Markern dient der Unterstützung des Konferenzleiters innerhalb eines Videokonferenzsystems. Der Konferenzleiter wird auf seinem Arbeitsplatz durch ein HMD⁸ unterstützt. Die Kommunikation mit den anderen Konferenzteilnehmern erfolgt durch ein Kamerasystem. Des Weiteren besitzt der Konferenzleiter ein virtuell geteiltes Notizboard, welches für alle Teilnehmer sichtbar ist. Auf diesem sind Marker bzw. Patterns für die anderen Teilnehmer oder für Aktionen abgebildet.

Das vorgestellte Konzept beschäftigt sich einerseits mit der Berechnung der Position und Lage der Marker und andererseits mit der Kalibrierung der Kamera und des HMDs. Da die Kalibriermethode irrelevant für diese Arbeit ist, wird diese nicht behandelt, kann jedoch in [25] nachgelesen werden. Ziel des Algorithmus ist es Patterns zu finden, zu identifizieren und ihre Position bzw. Lage zu berechnen. Zu diesem Zweck wird eine Transformationsmatrix benötigt, welche die Marker-Koordinaten ins Kamerakoordinatensystem umrechnet. Der Ablauf sei anhand von Referenz [25] dargestellt:

- * Zu Beginn wird das Bild mit einem Schwellwertfilter bearbeitet.
- * Im zweiten Schritt werden alle Regionen gesucht, welche von vier Linien begrenzt werden. Die Linien und Eckpunkte werden für einen späteren Zeitpunkt gespeichert und die Bereiche werden einer Normalisierung unterzogen.
- * Nun werden jene Ebenen berechnet in welchen je zwei parallele Linien bzw. Seiten des Markers liegen. Die Normalvektoren auf diese Ebenen (n_1 bis n_4) werden ebenfalls berechnet.
- * Um die Richtungsvektoren zweier paralleler Seiten zu berechnen wird das Kreuzprodukt $u_1 = n_1 \times n_2$ verwendet. Im Idealfall sollten die zwei resultierenden Vektoren orthogonal aufeinander stehen. Falls dem nicht so ist, sondern eine Abweichung auftritt, wird diese korrigiert (v_1, v_2). Es kann außerdem jener Richtungsvektor v_3 berechnet werden, welcher normal auf die beiden anderen steht. Durch die drei Vektoren v_1 bis v_3 ist die Rotationskomponente der gesuchten Matrix berechnet. Die Translationskomponente kann auf Grund der Gleichungen für die zuvor gespeicherten vier Eckpunkte und dem Zusammenhang zwischen Kamerakoordinaten und Markerkoordinaten berechnet werden.
- * Zur Charakterisierung der Marker werden kleine Regionen innerhalb der gefundenen vier Linien gebildet, beispielsweise ein 16×16 px⁹ Abtastgitter, und ein Korrelationsvergleich¹⁰ durchgeführt.
- * Abschließend wird ein Vertrauenswert anhand des Vergleiches mit Referenzmustern durchgeführt und anhand einer Schwellwertüberschreitung festgelegt, wann ein Pattern richtig erkannt wurde.

⁸Head-Mounted-Display [ein Display welches direkt vor den Augen eines Menschen angebracht ist, ähnlich einer Brille; in dieser Anwendung ist es durchsichtig, sodass darauf Informationen projiziert werden können] (HMD)

⁹Pixel

¹⁰Mit der Korrelation wird der Zusammenhang bzw. die Ähnlichkeit zweier oder mehrerer Merkmale, Funktionen oder Ähnlichem gemessen

3. Grundlagen

Nachdem die Funktionalität des Algorithmus beschrieben wurde, soll auf ein paar wichtige Eigenschaften der Patterns eingegangen werden. Grundlage dafür stellt Referenz [38] dar. Ein gutes Pattern muss:

- ▷ dementsprechend entworfen sein, damit ein Rückschluss auf die mehrdeutige Position und Orientierung relativ zur Kamera gezogen werden kann.
- ▷ sämtliche Orientierungen gleich behandeln. Es darf keine Vorzugsorientierung geben. Allerdings soll die Orientierung erkannt werden können.
- ▷ gut von anderen Patterns unterscheidbar sein. Dies bedeutet, dass die Korrelation zwischen den einzelnen Patterns minimal sein sollte.
- ▷ gut von der Umgebung bzw. Alltagsgegenständen unterscheidbar sein.
- ▷ schnell in der Szene auffindbar sein.
- ▷ einfach zu identifizieren sein.
- ▷ gut sichtbar sein, selbst bei größeren Distanzen zur Kamera. Dies inkludiert die Anforderung einer möglichst großen Fläche für die Identifikation des Patterns.
- ▷ möglichst unabhängig vom Kamerarauschen oder Verdunkelung sein.
- ▷ einen Kompromiss bzgl. der Rahmengröße des Patterns haben. Einerseits sollte das Pattern einen möglichst großen Rahmen haben, damit die Abgrenzung von der Umgebung ausreichend ist. Andererseits sollte jedoch der Rahmen möglichst klein sein, damit genügend Platz für das Pattern vorhanden ist. Laut oben genannter Referenz sollte der Rahmen mindestens 13% der Patternbreite ausmachen.

Bezüglich der Form und Farbe kann gesagt werden, dass für das einfache und schnelle Auffinden des Patterns zumindest vier markante Punkte vorhanden sein müssen. Werden vier Linien als Begrenzung verwendet ist diese Bedingung optimal erfüllt. Darüber hinaus bietet ein dadurch entstehendes Rechteck eine ausgezeichnete Basis zur Erkennung der Außenkanten und somit der Abgrenzung von der Umgebung. Wird ein monochromes Pattern gewählt hat dies den Vorteil der geringeren Rechenzeit und der Unabhängigkeit von der Farbauflösung bzw. Farbinterpretation einzelner Kameras. Außerdem bietet sich dadurch ein Vorteil hinsichtlich der Kontrastunterscheidung. Zu dieser Thematik müssen die Farben der Umgebung des Patterns berücksichtigt werden.

3.3. Optischer Fluss

Die Methode der Berechnung des optischen Flusses ist eine beliebte Variante zur Verfolgung von Objekten. Sie stellt eine der Möglichkeiten dar, um eine Bewegungsschätzung durchzuführen. Für die Schätzung der Bewegung eines Objektes muss zu anfangs eine sogenannte Fehlermetrik bestimmt werden und danach ein geeigneter Suchalgorithmus angewandt werden. Eine Fehlermetrik dient zur Bestimmung der Unterschiede zweier Bilder. Die Minimierung dieser Unterschiede kann als Beweis der Bewegung in einer

3. Grundlagen

Szene verwendet werden. Bewegt sich ein Objekt beispielsweise innerhalb von zehn Bildern um einen Zentimeter, kann auf Grund des minimalen Unterschiedes der Objekt-position, auf die Bewegung des Objektes geschlossen werden. Der Suchalgorithmus hat die Aufgabe jene Punkte oder Bildbereiche zu finden, welche verglichen werden sollen.

Es liegt auf der Hand, dass die einfachste, aber aufwändigste Möglichkeit hierfür ein kompletter Abgleich von jedem einzelnen Pixelwert der beiden Bilder ist. Beim optischen Fluss findet ebenfalls ein Pixelvergleich statt. Zu diesem Zweck werden die Unterschiede der Helligkeitswerte, manchmal auch der Farbwerte, zwischen zwei korrespondierenden Pixeln minimiert. Es wird jedoch kein kompletter Suchvorgang durchgeführt. Stattdessen wird das Bild zu Beginn bei einer niedrigen Auflösung durchsucht, um durch diese groben Übereinstimmungen den Suchbereich einzugrenzen. Danach wird bei einer höheren Auflösung in der eingegrenzten Region weitergesucht.

In dem, als Referenz angegebenen, Artikel [33] wird mit einem Startwert für die Differenz eines Punktes im Bilde gestartet. Anhand des räumlichen Gradienten der Intensität von jedem Punkt wird dieser geschätzte Startwert verbessert, bis das Gütekriterium, also die Differenz, minimiert ist. Grundsätzlich findet diese Gradientenbildung auf der Basis der linearen Approximation der ersten Ableitung der Bildfunktion statt. Die rekursive Ausführung dieses Prozesses, bis zum Auffinden eines Minimums, erfolgt ähnlich wie eine Iteration nach dem Newton-Verfahren.

Neben dieser einfachen Methode gibt es andere Implementierungen des Optischen Flusses. In [41] beispielsweise findet eine Minimierung eines Energiefunktionals, welches von der geschätzten Geschwindigkeit abhängt, statt. Eine separate Segmentierung des Greifers wäre beim Ansatz des optischen Flusses nicht unbedingt notwendig, würde allerdings Rechenzeit einsparen. Als themenbezogenes Anwendungsbeispiel sei hier Referenz [40] genannt. Hier kann anhand von optischem Fluss erkannt werden, welche Bildpunkte bzw. Gegenstände Teil des Roboters sind. Die Grundidee dabei ist, dass eine Änderung im optischen Fluss durch eine Bewegung des Armes erkannt werden kann. Bei zwei identen Bildern kommt es zu keinem optischen Fluss. Der Ansatz beruht, wie schon erwähnt, auf der Bewegung in einer Szene.

Im Folgenden werden einige Lösungsansätze vorgestellt, die als Entscheidungsgrundlage für die Greiferkennung dienen sollen. Dabei werden auch die Nach- und Vorteile kurz erwähnt und auf einige Schwierigkeiten eingegangen.

Allgemein ist zu sagen, dass ein paar Ansätze ohne Armbewegung auskommen. Eine zuverlässige Aussage über den erfolgreichen Greifversuch, anhand der vorgestellten Methoden, kann jedoch nur mittels einer Bewegung und anschließender Detektion getätigt werden. Dies liegt daran, dass die Stabilität des Griffes nur schlecht anhand einer optischen Analyse erfolgen kann. Obwohl sich ein Objekt innerhalb des Greifers befindet und umschlossen wurde, kann es sein, dass es durch eine Bewegung zum Herausgleiten des Gegenstandes aus dem Greifer kommt. Dabei ist es irrelevant, ob es sich um eine Armbewegung oder um die Bewegung des kompletten Roboters handelt. Da in den Anforderungen möglichst keine Bewegung festgelegt wurde, wird darauf geachtet eine notwendige Bewegung anzugeben und zu begründen.

4.1. 3D-Erkennung des Greifers und/oder des Objektes

Einer der ersten Lösungsansätze beschäftigt sich mit der direkten Objekterkennung. Die zu erkennenden Objekte müssen zu diesem Zweck als 3D-Modell in einer Datenbank vorhanden sein. Für genauere Informationen sei auf [54] und [56] verwiesen. In diesen beiden Referenzen sind die Grundlagen zur 3D-Objekterkennung mittels CAD¹-Modell zu finden. Es muss dementsprechend entweder ein CAD-Modell des Gegenstandes vom Benutzer entworfen werden oder per visueller Aufnahme und Oberflächenrekonstruktion erzeugt werden.

Nun gibt es zwei Möglichkeiten um diesen Ansatz für eine Implementierung zu verwenden. Erstens kann die Objekterkennung auf den Greifer selbst angewandt werden.

¹Computer Aided Design [Computerunterstütztes Design] (CAD)

4. Lösungsansätze

Zu diesem Zweck muss ein 3D-Modell des offenen und des (teils-)geschlossenen Greifers, ohne und eventuell mit einem darin befindlichen Objekt, vorhanden sein. Durch Vergleich der Greifermodelle mit dem realen Zustand des Greifers soll festgestellt werden, ob sich ein Objekt im Greifer befindet. Dabei ist nach dem Zugreifen durch den Endeffektor eine Bewegung des Armes von Vorteil um eine fehlerhafte (Nicht-)Erkennung zu verhindern. Die Erkennung könnte jedoch auch ohne Bewegung auskommen. Wird der geöffnete Greifer im aufgenommenen Szenario gefunden, muss der Greifer erneut geschlossen werden oder es befindet sich ein sehr großes Objekt im Greifer. Wird hingegen der geschlossene Greifer erkannt, kann davon ausgegangen werden, dass kein oder ein sehr kleines Objekt im Greifer vorhanden ist. Wird keines der beiden Modelle erkannt bzw. der verformte Greifer erkannt, muss sich, abgesehen von Verdeckungen und Ähnlichem, ein Objekt im Greifer befinden.

Zweitens kann von der Objekterkennung des zu ergreifenden Objektes ausgegangen werden. Dabei soll ein Objekt vor dem Zugreifen identifiziert werden und nach dem Zugreifen überprüft werden, ob das Objekt immer noch gefunden werden kann. Für diesen Fall muss ein Modell des Objektes, und nicht des Greifers, vorhanden sein. Aber auch dieser Ansatz kann noch einmal unterteilt werden. Einerseits könnte das Objekt ergriffen werden und der Arm inkl. Greifer aus dem Sichtbereich der Kamera gebracht werden. Solange sich kein zweites gleiches Objekt in der Szene befindet, wird das Objekt nicht mehr erkannt und es kann in gewisser Weise von einem erfolgreichen Greifversuch ausgegangen werden. Das große Problem hierbei liegt in dem Spezialfall, wenn das Objekt während der Bewegung aus dem Greifer fällt. Dieser Fehlerfall müsste durch eine Extrabehandlung bearbeitet werden. Allgemein wird bei dieser Idee von einem Negativvergleich gesprochen, weil nicht direkt überprüft wird, ob ein Objekt im Greifer oder der Greifer selbst gefunden wurde, sondern überprüft wird ob dies nicht gefunden wird. Andererseits könnte es auch zu einem Positivvergleich kommen. Hierbei soll das Objekt selbst vor und nach dem Greifvorgang erkannt werden. Befindet sich das Objekt tatsächlich im Greifer, muss dieser nach einer Bewegung von der Umgebung segmentiert werden und nach dem Objekt gesucht werden. Wird das gesuchte Objekt im Greifer gefunden, kann von einem erfolgreichen Greifversuch ausgegangen werden.

Es kann auch durch die Erkennung des Gegenstandes zu einer Positionsschätzung kommen. Verändert sich die Position des Objektes, kann von einem erfolgreichen Ergreifen gesprochen werden. Eine Ausnahme ist das Herausfallen oder Umfallen des Objektes. Für diese Art ist auf jeden Fall eine Bewegung des Armes notwendig. Der Vorteil dieser Variante liegt eindeutig im geringen Softwareaufwand.

Da die 3D-Objekterkennung schon laut [55] an der TU-Wien² implementiert wurde, müsste lediglich die Einbindung in ROS vorgenommen werden. Außerdem handelt es sich um einen simplen Programmablauf. Der Nachteil dieser Methode liegt jedoch in dessen Genauigkeit. Alleine auf Grund der Distanz zur Kamera kann eine Objekterkennung erschwert werden. In diesem Zusammenhang wurde der am ACIN³ entwickelte Algorithmus vom Autor in einer Projektarbeit getestet und arbeitete sehr zufriedenstellend. Bei diesem Szenario war jedoch der Kameraabstand weitaus geringer und die vordefinierten Objekte waren alle von größerem Volumen in der Dimension eines Apfels oder Modellautos. Kleinere Objekte wie beispielsweise ein Bleistift führen zu Problemen.

²Technische Universität Wien (TU Wien)

³Automation and Control Institute [Institut für Automatisierungs- und Regelungstechnik] (ACIN)

Der Algorithmus entscheidet sich für ein Objekt aus der vorhandenen Datenbank durch einen Wahrscheinlichkeitswert. Dieser gibt an, wie hoch die Ähnlichkeit zu einem CAD-Modell ist. Wird der Ansatz der Unterscheidung zwischen geschlossenem und geöffnetem Greifer verfolgt, stellt sich die Frage in wie weit und wie genau zwischen diesen beiden Zuständen anhand der 3D-Modelle unterschieden werden kann bzw. wie ähnlich welcher Zustand zu einem verformten Greifer ist. Zu Fehlentscheidungen könnte es, wie schon oben erwähnt, außerdem kommen, wenn sich kleine Objekte im Greifer befinden. Ein weiterer Nachteil ist die notwendige Rechenleistung und Verarbeitungsdauer. Zwei Punkte sind diesbezüglich zu erwähnen. Auf der einen Seite wird für die Erkennung sowohl 2D als auch Tiefeninformation verwendet. Auf der anderen Seite ist die Dauer des Vergleichs von der Anzahl der in der Datenbank vorhandenen Modelle abhängig. Je nach Art der Implementierung ist auch die notwendige Bewegung hinsichtlich der Anforderungen als nachteilig zu werten.

4.2. Kurven-/Kantenerkennung bzw. Konturerkennung

Der nächste Ansatz soll lediglich durch Kanten- oder Kurvendetektion einen Rückschluss auf den Erfolg des Greifversuches zulassen. Diese Idee umfasst mehrere Möglichkeiten. Vorweg muss erwähnt werden, dass diese Art der Erkennung für planare Greifer vorgesehen ist. Für komplexere Greifer ist dieser Ansatz weitgehend ungeeignet.

Als erste große Unterkategorie soll alleine die Entscheidung anhand von Kantendetektion diskutiert werden. Dem gegenüber steht eine Erkennung einer kompletten Kontur bzw. der Eigenschaften dieser Kontur, wie beispielsweise die Fläche. Die Erkennung anhand einer Flächenberechnung wird im nächsten Unterkapitel besprochen.

Wird der Greifer von der Umgebung segmentiert, soll anhand der Außenkanten bzw. deren Lage zueinander oder dem Normalabstand in der Greifermite zur Innenkante entschieden werden, ob sich ein Objekt im geschlossenen Greifer befindet oder nicht. Beispielsweise ist bei Verformung des Greifers durch ein Objekt der Abstand zwischen Außen- und Innenkante geringer als im geschlossenen Zustand ohne Objekt. In diesem Zusammenhang kann auch die Anzahl der geraden Linien als Entscheidungskriterium verwendet werden. Beispielsweise können alle parallelen Greifergräten gezählt werden. Ist der Greifer geschlossen und es befindet sich kein Objekt in ihm, verlaufen die Gräten jedes Fingers parallel zueinander. Außerdem wird nur eine Mittelkante zwischen den beiden Greifern erkannt. Diese Überlegung führt auch zur nächsten Rückschlussmöglichkeit.

Die Winkel zwischen den gefundenen Kanten können als Kriterium verwendet werden. Dies führt zu einer Art Kurvenerkennung. Es liegt auf der Hand, dass sich sowohl die Winkel zwischen Handgelenk und Innen- bzw. Außenkante, zwischen Innen- und Außenkante und zwischen den Gräten und einer der Kanten durch eine Greiferverformung ändern. Diese Winkel können ebenfalls als Entscheidungsmerkmal verwendet werden. Hierfür sei auf den Winkel β in Abbildung 4.1 verwiesen.

Alternativ kann, anstatt direkt über die Kanten eine Entscheidung zu treffen, die Krümmung einzelner Fingerteile betrachtet werden. Sind die Außenkanten nicht gekrümmt, befindet sich kein Objekt im Greifer. Die Ausnahme hierfür bilden (große) keilförmige Objekte. Sobald die Krümmung der Außen- oder Innenkante eines Fingers

4. Lösungsansätze

zunimmt und einen Schwellwert überschreitet, muss sich ein Objekt im Greifer befinden. In Abbildung 4.1 sind die Unterschiede der Krümmungen anhand der roten Linien veranschaulicht.

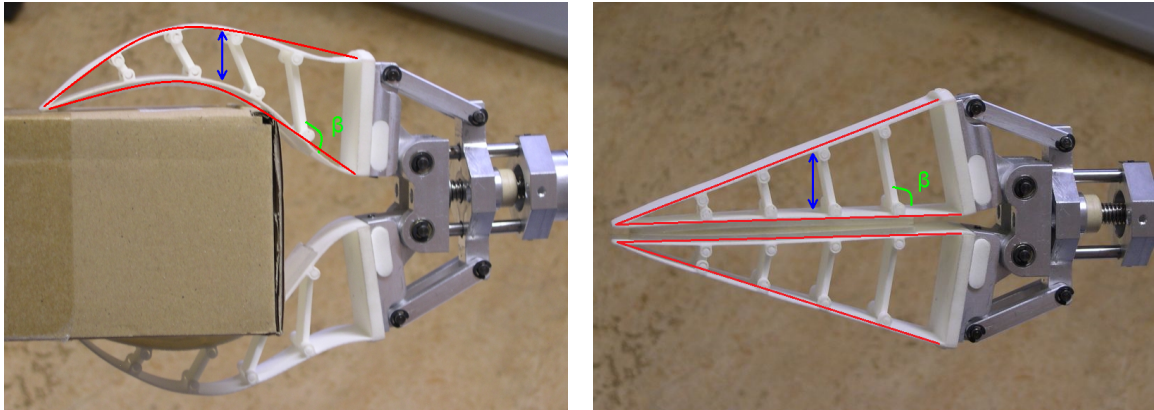


Abbildung 4.1.: Veränderung der Lage der Kanten zueinander, der Winkel zwischen den Einzelteilen und der Krümmung der Kanten auf Grund der Verformung des Greifers. In grün ist der Winkel β zwischen einer Greifergräte und der -innenkante dargestellt. Mit zunehmender Krümmung der Innenkante wird dieser größer. In blau ist der lotrechte Abstand zwischen Innen- und Außenkante dargestellt. Dieser wird mit steigender Krümmung kleiner. Die Verläufe der Außen- und Innenkanten der Greiferfinger sind mit roten Linien hervorgehoben.

Sämtliche dieser Möglichkeiten können anhand von Vergleichsaufnahmen mit einer Art Datenbank oder anhand von Schwellwerten agieren. Ein Vorteil dieser Methode ist die schnelle Datenverarbeitung, da nur 2D Daten verwendet werden. Außerdem gibt es eine Vielzahl an Kantenerkennungsalgorithmen die sich in der Praxis bewährt haben. Fraglich ist jedoch, ob die Größe der Bilder, also des segmentierten Greifers, für die Erkennung ausreichend ist. Unter diesem Nachteil leidet auch die Berechnung der Krümmung. Dies läuft daraus hinaus, dass kleine Objekte nur eine marginal unterschiedliche Krümmung der Greiferkanten verursachen, welche sogar nicht messbar auf Grund der Kameraqualität sein kann. Keilförmige Objekte erschweren die Erkennung zusätzlich, da diese, bei passendem Winkel der Keilform, zu keiner Verformung des Greifers und somit zu keiner Veränderung einer Entscheidungseigenschaft führen. Ein Anwendungsbeispiel ist in [5] zu finden. Hier wird anhand eines Konturvergleichs, zwischen einer Referenzansicht und der aktuellen Szene, ein Objekt gesucht. Dabei wird zuerst die Kontur des Gegenstandes segmentiert und anschließend findet der Vergleich anhand der Schätzung der Kollineationsmatrix⁴ statt. Da anhand der Kontur ein Visual Servoing Prozess ausgeführt wird, ist diese Vorgehensweise auch auf die hier gestellte Problematik anwendbar. Eine ähnliche Vorgehensweise ist in [6] beschrieben.

Die Autoren von [30] hingegen arbeiten mit einem Fourierdeskriptor zur Konturerkennung. Allgemein gesprochen wird ein Deskriptor verwendet, um ein möglichst eindeutiges Merkmal für einen bestimmten Bildbereich oder ein Objekt darzustellen. In diesem Fall wird die gefundene Kontur im Frequenzbereich dargestellt. Der Konturvergleich selbst wird anhand der Summe der quadrierten Abweichungen, dadurch werden

⁴Siehe auch Projektion und Kollineation in [16]

auch Informationen über die Rotation des Objektes berücksichtigt, durchgeführt. Der Vergleich der Konturen wird oft auch mittels Optimierungsalgorithmen gelöst. Als Beispiel sei hier [24] genannt, bei dem die Korrespondenz als Problem der kombinatorischen Optimierung formuliert wird.

Ein weiterer Ansatz ist die Verwendung von verschiedenen affinen Transformationen, deren Ergebnis Parameter sind, welche auf unterschiedliche Eigenschaften von geometrischen Objekten, wie Kurven oder Ähnlichem, hinweisen. Dazu sei auf die Literatur [39, 44, 45] verwiesen. Weitere Möglichkeiten zu Konturerkennung, beispielsweise durch Vergleich von Eigenschaften oder Kennzeichen einzelner Punkte der Kontur, sind in [2] zu finden. Dazu zählen auch die Eigenschaften der direkten Umgebung der Kontur. Als Grundlage der Erkennung und Berechnung von planaren Formen soll [34] als wichtige Referenz angegeben werden.

4.3. Flächenerkennung/-berechnung

Auch ein Algorithmus zur Flächenberechnung kann zur Entscheidung über einen erfolgreichen Greifversuch herangezogen werden. Die Grundlage dafür können 2D oder auch 3D-Daten sein. Es können mehrere Ansätze verfolgt werden. Erstens kann der Greifer von seiner Umgebung segmentiert werden, um die Fläche des Objektes, welches sich im Greifer befindet, zu berechnen. Da der Greifer nur aus seinen dünnen Konturen besteht, wird dieser selbst nicht als Fläche erkannt. Somit kann die Berechnung der gefundenen Fläche jedenfalls zur Bestimmung des Objektes benutzt werden. Eine Bewegung des Armes ist hier eindeutig notwendig. Falls der Arm, aus welchem Grund auch immer, ein Objekt beim Greifen verfehlt hat, steht dieses neben dem geschlossenen Greifer. Wird nun bei der Segmentierung nicht nur der Greifer, sondern auch das Objekt selbst, ausgewählt, kommt es zu einer Fehlentscheidung, da eine Fläche berechnet werden kann. Der gleiche Fehler würde bei (Selbst-)Okklusion passieren. Wird der Greifer bewegt, kann diese Problematik ausgeschlossen werden. Selbst der Fall eines teilweise geöffneten Greifers, dies kann durch Herausgleiten des Objektes aus dem Greifer passieren, wird korrekt behandelt. Auf Grund der nicht vorhandenen Daten kann keine Fläche gefunden werden.

Für den zweiten Ansatz kann die höchste Fläche in der Umgebung des Greifers, nach dessen Segmentierung, gesucht und berechnet werden. Als Ausgangspunkt dient hier jedoch die Annahme, dass ein Objekt gegriffen werden soll, welches höher ist als der Greifer selbst. Werden niedrigere Objekte ergriffen, führt dies zurück zum vorigen Ansatz. Die zuvor genannten Probleme hinsichtlich der Notwendigkeit einer Armbewegung sind ident.

Eine weitere Möglichkeit ist die Berechnung der Greiferfläche selbst. Dabei soll die vom Greifer aufgespannte Fläche in der Berechnung berücksichtigt werden. Zu diesem Zweck muss der Greifer jedoch über eine ausgefüllte Oberfläche verfügen. Dies ist ohne eine Veränderung der Hardware nicht gegeben. Eine Armbewegung ist hier, aus den gleichen Gründen wie zuvor genannt, ebenfalls notwendig. Bei diesem Ansatz wird, durch die Segmentierung des Greifers, jedenfalls eine Fläche gefunden und berechnet. Somit gilt es nur noch den Schwellwert einzustellen, welcher festlegt, ab wann die Fläche groß genug ist, um ein Objekt zu signalisieren.

Verwandt mit dieser Problematik ist auch die Oberflächenrekonstruktion von Objekten. Für dieses Problem sei auf [27, 42] verwiesen. Die Oberflächenrekonstruktion kann als Grundlage für die Berechnung der Flächen dienen. Sie kann ebenfalls für den unter Abschnitt 4.1 vorgestellten Ansatz verwendet werden.

Die Berechnung des Volumens kann ebenfalls in Betracht gezogen werden. Zu erwähnen ist allerdings, dass eine aufwändige Flächenrekonstruktion für die zu lösende Problemstellung einen unnötig hohen Aufwand darstellt und daher nur der Vollständigkeit halber erwähnt werden soll. Im Gegensatz dazu kann aus der aufgenommenen Punktwolke eine Schätzung der Oberflächennormalen durchgeführt werden. Wurde der Greifer, und das eventuell in ihm vorhandene Objekt, aus der Punktwolke segmentiert, sollen die Flächennormalen berechnet werden. Nun kann beispielsweise auf Grund der Anzahl der berechneten Flächennormalen eine Aussage über die Größe der Fläche getätigt werden. Zu beachten ist, nicht alle Normalen zu zählen, sondern lediglich jene, welche parallel zu den Normalen der Greiferoberfläche verlaufen oder mit diesen einen geringen Winkel einschließen. Für die Schätzung der Oberflächennormalen sei auf [9] verwiesen.

Als Vorteil dieser Methode ist die universelle Verwendung eindeutig hervorzuheben. Der Algorithmus kann sogar weitgehend unabhängig vom verwendeten Greifer arbeiten. Wird ein komplizierterer Greifertyp verwendet, muss lediglich die Entscheidungsfindung verändert werden. Die Berechnung der Greiferfläche fällt hierfür meistens weg, da beispielsweise bei einer menschenähnlichen Hand kaum parallele Oberflächennormalen gefunden werden. Trotzdem kann die Größe der gesamten Oberfläche der leeren Hand und der befüllten Hand verglichen werden und eine Schwellwertentscheidung implementiert werden. Wird die komplette Oberflächenrekonstruktion von Gegenständen und dem Greifer miteinbezogen, kann das Volumen des gesamten Körpers, also der Hand inklusive eines Objektes, als Entscheidungskriterium gewählt werden. Es sind keine Referenzaufnahmen notwendig, welche in einer Datenbank gespeichert werden müssen.

Ein großer Nachteil ist die leistungsintensive Berechnung der Oberfläche. Allein die Verwendung von 2D Informationen und der zusätzlichen Tiefeninformation ermöglicht zwar eine genauere Berechnung, jedoch fallen dadurch mehrere Rechenschritte an. Dies trifft vor allem bei einer kompletten Rekonstruktion der Oberfläche zu. Außerdem muss, wie in vielen anderen Lösungsansätzen, der Greifer segmentiert werden. Dies stellt eine zusätzliche Fehlerquelle dar. Zuletzt muss auch die Genauigkeit der Flächenberechnung hinterfragt werden. Die Abhängigkeit von der verwendeten Kamera kann zu Problemen bei der Erkennung von kleinen Objekten führen. Durch die notwendige Bewegung des Armes kann dies jedoch verbessert werden, da der Greifer näher bei der Kamera positioniert werden kann, und dadurch die Anzahl der aufgenommenen Punkte, die den Greifer repräsentieren, erhöht wird.

4.4. SIFT-Vergleich

Die Verwendung des SIFT-Algorithmus kann ebenfalls für die Greifererkennung verwendet werden. Es ist das Ziel auf Grund der gefundenen Features (Kennzeichen, Eigenschaften) in der aufgenommenen Szenerie einen Vergleich mit den Charakteristika von zuvor

aufgenommenen Bildern zu tätigen. Es soll ebenfalls der Greifer von der Umgebung selektiert werden. Die Referenzbilder müssen bereits in der Datenbank vorhanden sein und deren Kennzeichen berechnet worden sein. Alle in der aktuellen Szene aufgenommenen Features können somit online mit denen der Vergleichsbilder gegenüber gestellt werden. Als Vergleichsbilder sollen dabei unterschiedliche Greiferpositionen dienen. Diese beinhalten den teilweise bzw. komplett geöffneten und den geschlossenen Greifer, sowie einige Bilder mit unterschiedlicher Verformung des Greifers. Dafür können sowohl Aufnahmen des Greifers mit bestimmten Objekten, welche auch im Anwendungsfall zu ergreifen sind, wie auch mit Hilfsobjekten verwendet werden. Mit Hilfsobjekten sind einfache geometrische Objekte gemeint, welche nur zur Verformung des Greifers, nicht aber zur Repräsentation eines Greif szenarios dienen. Anschließend soll auf Grund eines Vergleichs mit allen vorhandenen Greiferstellungen festgestellt werden, welche die Ähnlichste ist. Die Referenzbilder können eindeutig einem Ergebnis zugeordnet werden – ob ein Objekt ergriffen wurde oder nicht – und dienen somit als Grundlage der Entscheidungsfindung.

Der Grundgedanke war es den Greifer nicht verändern zu müssen um zufriedenstellende markante Punkte für die Analyse zu erhalten. Sollte sich dies als nicht möglich herausstellen, besteht zusätzlich die Möglichkeit den Greifer insofern zu modifizieren, dass markante Punkte gefunden werden können. Ein Modifikationsbeispiel ist die Färbung von Greiferteilen. Außerdem kann eine Struktur auf die Oberfläche des Greifers angebracht werden um über die Textur markante Punkte zu finden. Eine Möglichkeit der Anpassung ist in Kapitel 5 dargestellt. Werden die markanten Punkte direkt am Greifer gefunden, kann auf Grund derer gegenseitiger Lage auf das Vorhandensein eines Objektes im Greifer rückgeschlossen werden.

Vorteile dieses Ansatzes sind einerseits die geringeren Leistungsanforderungen an den Computer und die einfache Implementierung des SIFT-Algorithmus, da dieser in der Praxis für andere Anwendungen vorhanden ist, und andererseits die gute Portabilität und Wiederverwendbarkeit des Algorithmus. SIFT kann für alle Greifertypen verwendet werden, solange ein paar wenige Anpassungen hinsichtlich der Hardware getätigt werden. Im Weiteren könnte der SIFT-Algorithmus für andere Aufgaben in der mobilen Robotik und im Hobbit Projekt verwendet werden. Fraglich ist allerdings, ob die Qualität der Kamera ausreichend ist und ob aussagekräftige Features gefunden werden können. Eine notwendige Armbewegung ist zusätzlich als Nachteil anzuführen.

4.5. Graphenbasiertes Verfahren

Die Anwendung von graphenbasierten Verfahren zur Erkennung von Objekten steht im engen Zusammenhang mit der Kantendetektion. Ohne Kanten können keine Graphen erstellt werden. Einfach ausgedrückt bestehen Graphen aus Linien und Knotenpunkten. Dabei gibt es im Allgemeinen zwei verschiedene Ansätze.

Beim ersten Ansatz wird die komplette Kontur eines Objektes verwendet und die sogenannten Kosten, ein Wert, der die Ähnlichkeit der Kontur mit einer Referenzkontur beschreibt, werden berechnet. Dies kann auf unterschiedliche Weisen passieren. Anhand der Kosten wird entschieden, welcher Referenzfigur das aufgenommene Objekt am

meisten ähnelt. Dadurch kann eine Entscheidung getroffen werden.

Im Unterschied dazu wird beim zweiten Ansatz lediglich ein Teil der gesamten Kontur verglichen. Somit wird jedes Fragment der aufgenommenen Figur mit den Fragmenten der Referenzbilder verglichen. Dies hat mehrere Vorteile. Es ist dadurch weitaus besser möglich auf das Problem von Verdeckungen einzelner Objekte zu reagieren. Außerdem sind bessere Ergebnisse zu erwarten, falls der Unterschied zwischen der aktuellen Gegenstands- und der Referenzaufnahme nur in einer Rotation besteht. Obwohl die beiden Verfahren verwandt sind, ist die Vorgehensweise im Vergleich zur Kantendetektion jedoch komplett unterschiedlich. Das zu Beginn erfasste Szenario wird nicht mit einem Kantenfilter bearbeitet, sondern es findet eine Segmentierung kleiner Bereiche statt. Es kommt zu einer sogenannten Übersegmentierung, weil sehr viele kleine Bereiche, die Segmente, entstehen, die aneinander grenzen. An den Grenzstellen werden Sprungstellen bzw. Unterschiede festgestellt.

Ein Graph besteht nun einerseits aus den Knotenpunkten, dies sind Punkte an denen drei oder mehrere Segmente aneinander grenzen, und den Kanten, dies sind die Verbindungen zwischen zwei Knotenpunkten. Für den Vergleich mit den Graphen der Referenzbilder kommen mehrere Möglichkeiten in Frage. Eine sehr einfache Art und Weise ist die euklidische Distanz (siehe [22]). Kompliziertere Ansätze verwenden beispielsweise einen Algorithmus der die Kosten des kürzesten geschlossenen Umlaufes eines sogenannten Produktgraphes berechnet. Für nähere Informationen sei hier auf die Literatur [23] verwiesen.

Ein großer Vorteil dieses Verfahrens besteht in der Möglichkeit der Behandlung von verdeckten Objekten. Zusätzlich kann dieser Ansatz auf verschiedene Endeffektoren bzw. Systeme portiert werden. Die Form des Greifers ist nicht ausschlaggebend. Nachteilig ist erneut die starke Abhängigkeit der Kamera zu erwähnen. Sind die einzelnen Konturen sehr dünn oder weit entfernt, wird es zu Leerstellen im Graph kommen. Diese beeinträchtigen das Ergebnis. Mit zunehmender Komplexität des Konturvergleiches steigt zwar die Qualität der Erkennung, allerdings auch die notwendige Rechenleistung und Berechnungsdauer. Ein Extrembeispiel ist in [4] gegeben. Hier wird mit einem hochqualitativen Konturdetektor gearbeitet, der allerdings eine leistungsstarke CPU⁵ und GPU⁶ benötigt. Zur Abarbeitung der Detektion wird die Programmieretechnik CUDA⁷ verwendet, welche auf die parallele Abarbeitung von Prozessen auf einer GPU spezialisiert ist.

4.6. AR-Marker

Als nächster Ansatz sei die Verwendung des in Kapitel 3.2 vorgestellten AR-Marker-Algorithmus zur Greifererkennung präsentiert. Die Grundidee liegt auf der Hand. Es werden Patterns auf dem Greifer angebracht, welche, auf Grund ihrer Identifikation, einen Rückschluss auf den Status des Greifvorgangs ermöglichen.

⁵Central Processing Unit [Zentralrecheneinheit] (CPU)

⁶Graphics Processing Unit [Grafikprozessor] (GPU)

⁷Compute Unified Device Architecture [eine Programmieretechnik, die von der Firma NVIDIA entwickelt wurde um Programmteile auf die Grafikkarte auszulagern und dort parallel abzuarbeiten] (CUDA)

4. Lösungsansätze

Da der Greifer ursprünglich aus einem dünnen flexiblen Gerüst besteht, muss für diese Lösungsvariante eine fundamentale Modifikation der Hardware vorgenommen werden. Der Greifer muss mit einem Material überzogen werden, damit die Patterns problemlos auf dem Greifer platziert werden können. Hier sind erneut zwei Möglichkeiten vorhanden. Es sollen je ein Pattern auf der Oberfläche einer Greiferhälfte angebracht werden und über deren Position soll bestimmt werden, in welchem Zustand sich der Greifer befindet. Daraus kann geschlossen werden, ob sich ein Gegenstand im Greifer befindet oder nicht. Auf Grund der Form des Greifers wurde diese Variante als unbrauchbar empfunden, da die zu verwendenden Patterns sehr länglich oder sehr klein sein müssen, um an der Oberseite des Greifers Platz zu finden. Vor allem bei größeren Distanzen zur Kamera kann dies zu Problemen führen.

Deswegen wurde eine andere Möglichkeit ins Auge gefasst, bei der ein einziges Pattern an der Oberseite des Greifers angebracht wird. Das Pattern ist sozusagen zweigeteilt und schließt beim geschlossenen Greifer bündig mit der Kante ab. Befindet sich nun der Greifer in seiner geschlossenen Position, wird ein Pattern erkannt und die Lage dieses zurückgeliefert. Befindet sich allerdings ein Gegenstand im Greifer oder ist der Greifer geöffnet, kann das Pattern nicht gefunden werden und der Erkennungsversuch schlägt fehl.

Obwohl dieser Algorithmus nach Wissen des Autors noch nie für eine derartige Anwendung verwendet wurde, eignet sich dieser für sämtliche planaren Greifer und hat deutliche Vorteile gegenüber anderen Algorithmen. Beispielsweise kommt dieser ohne jegliche Zusatzberechnungen und -schritte, wie der Segmentierung des Greifers, aus. Es handelt sich um ein simples aber leistungsfähiges Prinzip, was sich in der Ausführungszeit niederschlägt. Selbst für kleine Objekte ist mit guten Ergebnissen zu rechnen. Als Nachteil muss die, wie bei vielen anderen Ansätzen, notwendige Bewegung und Qualität der Kamera genannt werden. Die Veränderung der Hardware kann, auf Grund des Aufwandes, als Nachteil gesehen werden, konnte aber durch einige damit verbundene Verbesserungen bzgl. der Sichtbarkeit des Greifers als durchaus positiv gewertet werden. Details hierzu sind im Abschnitt 5 zu finden.

4.7. Optischer Fluss

Im Unterschied zu allen bisher vorgestellten Lösungsansätzen wird beim Optischen Fluss vorausgesetzt, dass eine Bewegung stattfindet. Ohne Bewegung des Armes bzw. des Objektes kann der Algorithmus, wie in Kapitel 3.3 vorgestellt, nicht arbeiten. Die Grundlage ist allerdings erneut der Vergleich von Bildaufnahmen.

Im Gegensatz zu einem Vergleich mit einem Offline-Referenzbild, also einem in einer Datenbank gespeicherten Bild, findet ein Vergleich zweier Online-Bilder statt. Dabei wird das aktuell von der Kamera gelieferte Bild mit dem zuvor gelieferten verglichen. Durch die Unterschiede in den beiden Bildern kann auf eine Bewegung geschlossen werden. Für die Greifererkennung ist es nun wichtig, dass die Bewegung des Armes eine Bewegung des Gegenstandes im Arm zu Folge haben muss, falls ein Objekt ergriffen wurde. Es liegt auf der Hand, dass die direkte Sicht auf den Gegenstand möglichst gut gegeben sein muss. Kann die Bewegung des Objektes, auf Grund dessen Größe oder Lage im Greifer, nicht nachvollzogen werden, kann keine Aussage über den Erfolg des

Greifprozesses gemacht werden. Der große Vorteil dieser Methode ist die Portabilität des Verfahrens. Unabhängig von der Art des Armes und Greifers kann dieser Algorithmus überall eingesetzt werden. Ein Nachteil ist jedoch die schon angesprochene Sicht auf den Gegenstand. Um während der Bewegung eine möglichst unveränderte Sicht auf diesen zu haben ist eine Trajektorienplanung für den Arm notwendig. Dies bedeutet, dass der Arm nicht einfach von jeder Position auf beliebigem Wege in eine vordefinierte Endposition bewegt werden kann, sondern der genaue Bewegungsablauf berechnet werden muss, um die Sicht auf den Gegenstand während der Bewegung, zu gewährleisten. Kleine Gegenstände werden außerdem schwierig zu erkennen sein.

Da der Vergleichsprozess oft in Verbindung mit Helligkeitswerten getätigt wird, ist die Beleuchtungsabhängigkeit ein weiterer Nachteil. Hierzu ist zu erwähnen, dass alle visuellen Systeme von der Helligkeit abhängig sind, sie aber gerade beim Optischen Fluss eine sehr große Rolle spielt. Die Variante der Kantenerkennung ist hier wesentlich unempfindlicher.

4.8. Modellierung des Greifprozesses

Der letzte vorzustellende Ansatz behandelt die Abbildung einer realen Greifszene in einer Modellierungs- bzw. Simulationsumgebung. Bei diesem Vorgang ist das Ergreifen des Objektes inkludiert. Das heißt, es muss nicht auf eine Berechnung der Greifpunkte gewartet werden. Es wird ein Szenario möglichst genau durch eine oder mehrere Kameras aufgenommen. Wird nur eine Kamera verwendet, soll die Kamera im Raum bewegt werden. Ist die Aufnahme der realen Umgebung in möglichst vielen Perspektiven abgeschlossen, muss mit der Rekonstruktion bzw. der Modellierung begonnen werden. Ist die komplette Szene inklusive des Roboters in die Simulationsumgebung geladen, werden einerseits die besten Greifpunkte und andererseits die Bahnplanung zum Greifvorgang berechnet.

Sowohl die Angreifpunkte, als auch die Bahnplanung ist nicht eindeutig. Deswegen kann jede Möglichkeit vorab mittels Simulation getestet werden und das Ergebnis anhand einer Kostenfunktion bewertet werden. Diese beinhaltet eine Bewertung für die Geschwindigkeit und Stabilität des Vorganges. Ist der beste Pfad inklusive der besten Angreifpunkte gefunden, wird der simulierte Ablauf in die Realität umgesetzt.

Diese Methode setzt erstmals auf die Bewegung des gesamten Roboters zur Aufnahme der kompletten Szene. Der endgültige Greifvorgang kann zusätzlich durch eine Visual Servoing Technik verbessert werden. Dabei soll eine zweite Kamera am Arm des Roboters verwendet werden. Auf Grund der genauen Offlineberechnung und Simulation kann die Entscheidung über den Erfolg des Greifvorganges vorweg genommen werden. Dieser beruht allerdings rein auf mathematischen Berechnungsschritten. Es besteht immer noch die Möglichkeit, dass das Objekt auf Grund von Toleranzen und Berechnungsfehlern nicht ergriffen wird.

Ein Vorteil dieser Methode ist die Unabhängigkeit von Beleuchtungsänderungen und Verdeckungen. Verdeckungen werden von vornherein berücksichtigt. Auf spontan entstandene Probleme wie die Änderung der Lichtverhältnisse, neuerliche Verdeckung oder Ähnliches kann nicht eingegangen werden. Ist der Vorgang einmal gestartet wird

4. Lösungsansätze

nicht mehr auf Veränderungen reagiert. Ein weiterer Vorteil dieser Technik ist die Plattformunabhängigkeit und die Einbeziehung mehrerer Modelle. Beispiele hierfür sind die Referenzen [15] und [53]. Es muss sowohl ein Modell der Umgebung, wie auch des Objektes und des Greifers vorhanden oder erstellt werden. Dies ist gleichzeitig ein Nachteil dieses Ansatzes. Neben dem hohen Berechnungsaufwand, für die Modelle, Bahnplanung und Auswertung, existiert eine lange Wartezeit bis der Vorgang eingeleitet wird, da die Umgebung erst visuell erfasst werden muss. Außerdem kann auf Fehler jeglicher Art nicht mehr reagiert werden, nachdem der Greifvorgang gestartet wurde. Wie allerdings schon in Kapitel 2.2.3 erwähnt, kann dies ohne Modell der Umgebung und unter Einbeziehung einer optischen Regelschleife für den Greifprozess ermöglicht werden.

Wie schon zu Anfangs erwähnt, musste eine Modifikation am Endmanipulator des Hobbit vorgenommen werden, um mit dem in späterer Folge gewählten Ansatz Erfolg zu haben. Dies ist notwendig, da für die Rekonstruktion des Greifers in der Punktwolke zu wenige Daten vorhanden sind. Der Greifer selbst – ohne der Hardwaremanipulation – ist in den wichtigsten Perspektiven (Draufsicht, etc.) nicht sichtbar in der Punktwolke. Dieser Sachverhalt wird vor allem in Kapitel 6.3. veranschaulicht. Im Falle des ursprünglichen Greifers sind die 2D Daten sogar relevanter als die Tiefeninformation. Dies liegt hauptsächlich an den feinen Konturen des Greifers.

Zuerst soll kurz auf die Verformung des Greifers durch ein Objekt und deren Auswirkungen eingegangen werden. Der Greifer selbst besteht aus mehreren Teilen und ist entlang seiner Mittelachse symmetrisch. Im Grunde kann dieser in drei Stücke eingeteilt werden. Der erste Teil wird im Weiteren als Handgelenk bezeichnet, weil er den mechanischen Teil beinhaltet, der die beiden anderen Teile, die sogenannten Finger (Finray-Finger der Firma Festo) aufnimmt und bewegt. Die beiden Finger bestehen aus feinen Streifen aus Polyamid und sind dadurch sehr flexibel, jedoch trotzdem äußerst stabil [11]. Diese Stabilität wird durch je vier Querverstrebungen, zukünftig Fischgräten genannt, unterstützt. Diese Gräten stehen im Grundzustand, unabhängig davon ob der Greifer geöffnet oder geschlossen ist, parallel zu einander. Auf Grund dieser Konstruktion ist ein sicheres Umfassen bzw. Greifen verschiedenster Gegenstände möglich. Die Finger passen sich an die Kontur des Objektes an. Um ein Herausgleiten der Objekte aus dem Greifer – dies ist in der harten, glatten Oberfläche des Polyamids begründet – zu vermeiden, werden Gel-Pads an der Innenseite angebracht. Durch eine Motorspindel kann der Greifer innerhalb seines Arbeitsbereiches stufenweise geöffnet und geschlossen werden! Der geschlossene Zustand des Greifers ist in Abbildung 5.1 dargestellt.

Im Vergleich dazu sollen weitere Zustände des Greifers besprochen werden. Ein wichtiger Vergleichszustand ist der geöffnete Greifer. Für diesen Zustand ist besonders wichtig, dass die beiden Finger selbst ihre Form nicht verändern, sondern lediglich ein keilförmiger Spalt zwischen ihnen entsteht. Dies gilt für jede geöffnete Position,



Abbildung 5.1.: Geschlossener Greifer mit Festo-Fingern. Die Finger werden durch die, von einem Elektromotor angetriebene, Spindel am Handgelenk geöffnet und geschlossen. Sie passen sich an die, sich im Greifer befindlichen, Objekte an.

unabhängig davon ob der Greifer komplett oder nur teilweise geöffnet ist.

Befindet sich allerdings ein Objekt im Greifer und wird dieser geschlossen, kommt es, je nach Gestalt des Objektes, zu einer Verformung der Greiferfinger. Eine Gegenüberstellung der verschiedenen Greiferzustände ist in Abbildung 5.2 gegeben.

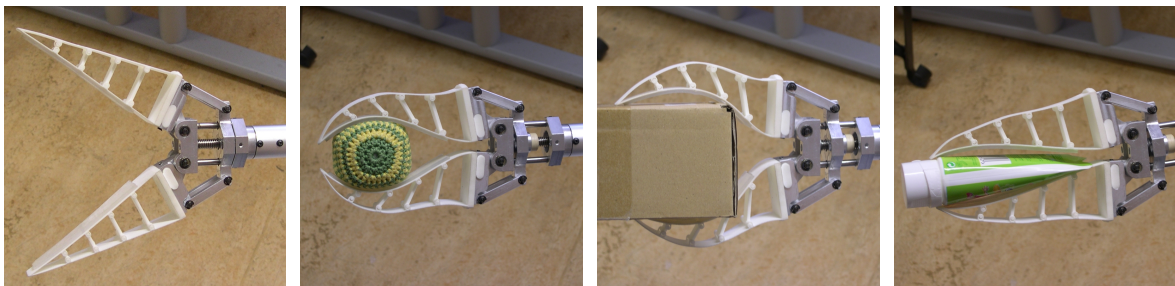


Abbildung 5.2.: Greiferzustände mit verschiedenen Objekten. Dem komplett geöffneten Greifer steht der geschlossene Zustand mit Objekten, wie einem Hacky-Sack, einer Kartonbox und einer Zahnpastatube, gegenüber.

In den mittleren beiden Abbildungen kann erkannt werden, dass sich die vier Fischgräten durch das Umschließen eines Objektes unterschiedlich neigen. Je nach Objekt wird auch die Innen- und Außenkante der Finger gekrümmt. Wird ein Blick auf das rechte Bild in Abbildung 5.2 geworfen, ist ersichtlich, dass keilförmige Objekte die Greiferfinger weitaus weniger bis gar nicht, je nach Winkel der Keilform, krümmen. Dieser Fakt ist vor allem hinsichtlich der Idee, die Greifererkennung anhand der Krümmung der Finger zu kontrollieren, interessant. Hätte ein Objekt exakt jene keilförmige Gestalt, sodass der Greifer nicht gekrümmt wird, wäre eine Entscheidung über das (erfolgreiche) Greifen eines Objektes anhand der Krümmung der Greiferfinger nicht zielführend. Unterstützend müsste eine zweite Instanz zu Rate gezogen werden. Dies kann beispielsweise durch die

5. Hardwaremodifikation

Kontrolle der Lage der beiden Außenkanten der Finger zueinander ermöglicht werden.

Werden sehr kleine Objekte verwendet, würde auch dies, wie schon weiter oben erwähnt, nicht zu einem korrekten Ergebnis führen. Eine weitere wichtige Beobachtung konnte anhand von Objekten getätigt werden, welche durch ihre Form oder Höhe die Sicht auf den Greifer (teilweise) verdecken. Dazu seien ein kleiner Holzhammer und eine große Kartonrolle, welche vom Hobbit ergriffen wurden, betrachtet. Durch die teilweise oder komplette Verdeckung des Greifers können einige Merkmale, wie beispielsweise die Krümmung der Außenkanten, nicht mehr gefunden bzw. berechnet werden. Diesen Sachverhalt unterstreicht Abbildung 5.3.



Abbildung 5.3.: Teilweise durch ein Objekt verdeckter Greifer. Je nach Form und Größe des Gegenstandes werden die Finger des Greifers oder sogar der komplette Greifer verdeckt. Im Falle des kleinen Holzhammers im linken Bild findet nur eine geringe Okklusion des Greifers statt. Im Vergleich dazu sind im rechten Bild durch die massive, lange Kartonrolle beinahe beide Greiferfinger zur Gänze verdeckt.

5.1. Prototypen

Kommen wir zurück zum ursprünglichen Problem, nämlich jenem der schlechten Sichtbarkeit des Greifers aufgrund seiner Struktur. Es liegt nahe, dass der Greifer durch eine Schicht abgedeckt werden soll, damit dieser eine zusammenhängende Fläche bildet und nicht nur von Konturen umrandet wird. Die einfachste Variante um dies zu bewerkstelligen war zwei Luftballone über den Greifer zu ziehen. Dabei wurde jeder Finger mit einem Luftballon abgedeckt. Auf Grund der deutlich unterschiedlichen Form eines Greiferfingers im Vergleich zu einem Luftballon, wurde der Greifer leicht verformt. Da diese Variante jedoch lediglich der Demonstration diene um somit eine kostenarme, flexible und schnelle Durchführungsmöglichkeit für diesen Ansatz bereitstellte, war im Vorhinein klar, dass mit einigen Abstrichen hinsichtlich der Gestalt zu rechnen ist. In Abbildung 5.4 ist dies veranschaulicht.

5. Hardwaremodifikation

Es ist klar zu erkennen, dass beim Umfassen von Objekten kein gravierender Unterschied der Greiferform im Vergleich zum – nicht-überzogenen – Greifer vorhanden ist. Allerdings kann für den geöffneten und den geschlossenen Greifer eine Deformierung festgestellt werden, welche ihren Ursprung in der Form des Luftballonüberzuges findet.

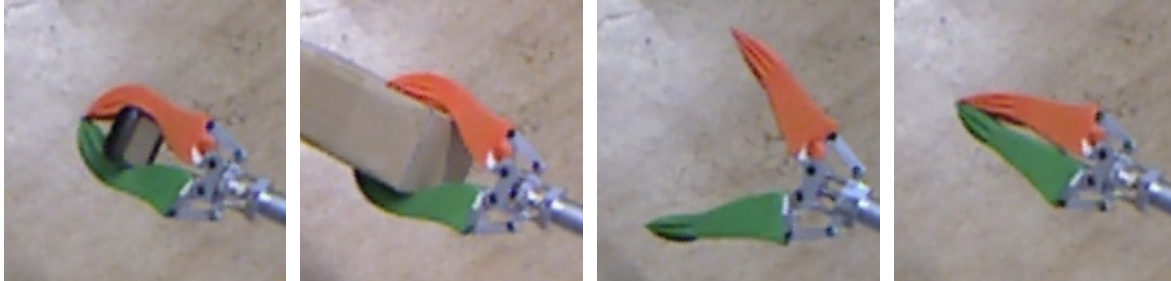


Abbildung 5.4.: Greifer mit Prototypüberzug V1. Im Anfangsstadium wurden zwei unterschiedliche farbige Luftballone verwendet um den Greifer zu überziehen. Verglichen wurden die Aufnahmen mit zwei Objekten, einem Handy und einer Kartonbox, dem geöffneten und dem geschlossenen Greifer. Da die runde Form der Ballone nicht mit der Kontur der Finger übereinstimmt, kommt es zu einer geringen Verformung des Greifers.

Wird der geschlossene Greifer betrachtet, kann eine eindeutige Krümmung der Außenkante und in diesem Zusammenhang ein Spalt zwischen den beiden Fingern festgestellt werden, welcher beim „nackten“ Greifer nicht vorhanden war. Da sich der Luftballonüberzug als Prototyp im Hinblick auf die Tatsache, dass der Greifer in der Punktwolke nun sichtbar war, bewährt hatte, wurde nach einer Möglichkeit gesucht einen möglichst passgenauen Überzug für den Greifer herzustellen. Es musste allerdings beachtet werden, dass ein Material welches zum Überziehen des Greifers verwendet werden soll, einige Eigenschaften aufweisen muss. Einerseits muss das Material reißfest und robust sein. Andererseits sollte es ebenso flexibel und anpassungsfähig sein und sich, trotz Verformung, nach Möglichkeit wieder in seine Ausgangsgestalt zurückbilden.

Bevor eine aufwändige, kostenintensive Herstellung eines hochpräzisen Überzuges durchgeführt wurde, sollte ein möglichst einfacher Weg gefunden werden um einen adäquaten Überzug herzustellen. Als erstes Material kam dem Autor Latex in den Sinn. Dieses Material erfüllt all diese Eigenschaften und wird häufig für verschiedene alltagsübliche Anwendungen verwendet. Als Paradebeispiel sollen Latexhandschuhe erwähnt werden, welche sowohl im Haushaltsbereich, wie auch in der Medizin oft verwendet werden. Auch dies beweist erneut die Nähe zu menschlichen Anwendungsfällen und Problemstellungen, wenngleich die Verwendung von Latexhandschuhen einen anderen Hintergrund hat als die Verwendung eines Latexüberzuges für den Hobbitgreifer.

Zu Beginn musste jedoch geklärt werden, wie Latex verarbeitet wird. Nach kurzer Recherche wurde mit der Herstellung eines Gipsmodells in der Form eines Greiferfingers begonnen. Dazu wurde eine mehrteilige Gussform aus lackiertem Holz hergestellt. Sämtliche Dimensionen des Greifers sind in [11] zu finden. Anschließend konnte der dickflüssige Gips in die Form gegossen werden. Nachdem der Gips ausgehärtet war, wurde das Fingermodell aus der Form genommen und mit sehr feinem Schleifpapier nachbearbeitet. Dargestellt sind die Gussform und das Gipsmodell in Abbildung 5.5.

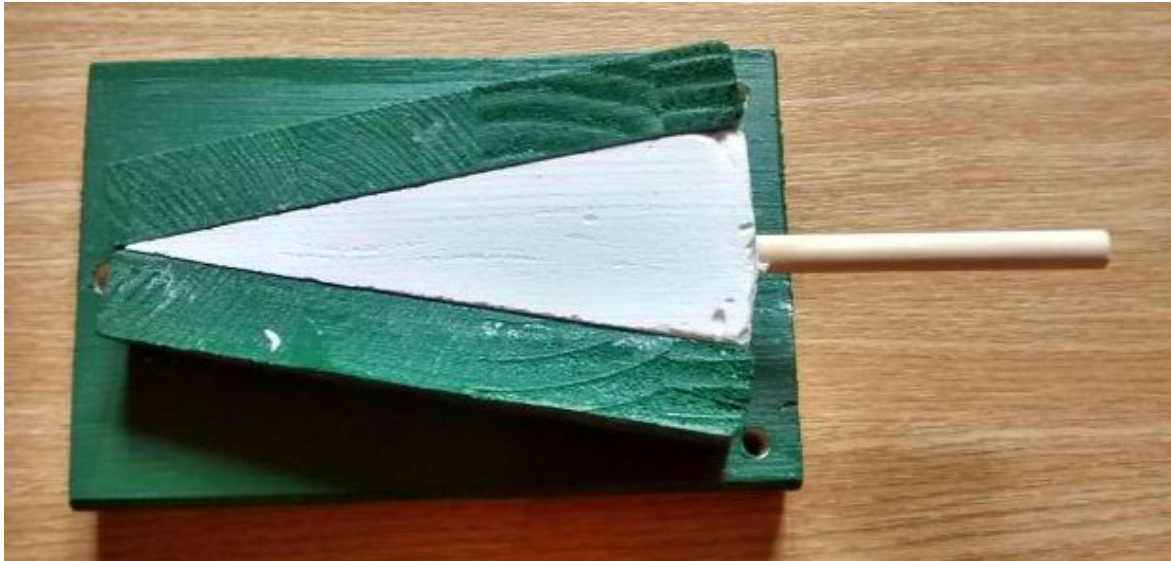


Abbildung 5.5.: Gussform mit Gipsabdruck. Die grün lackierten Holzwände wurden durch drei Gewindestangen miteinander verbunden und zusammengepresst um ein Herausfließen des Gips aus der Form zu verhindern. Zusätzlich wurde ein Rundholzstab eingegossen, damit das Eintauchen in die Überzugsmasse, beispielsweise Latex, vereinfacht wird.

Es liegt auf der Hand, dass die Form, auf Grund einiger Toleranzen in der Fertigung, nicht exakt den Abmessungen des Greiferfingers entsprach. Das Ergebnis war, in Anbetracht der Tatsache, dass jeder Arbeitsschritt händisch durchgeführt wurde, sehr zufriedenstellend. Die Gipsform wurde in ein Naturlatexbad getaucht und somit überzogen. Nach der Trockenzeit konnte eine weitere Schicht Latex aufgetragen werden. Es wurden Überzüge mit unterschiedlicher Schichtdicke erzeugt, wobei drei Schichten ungefähr 0.7mm entsprechen und als passend empfunden wurden. Alle dünneren Überzüge hatten eine zu geringe Reißfestigkeit und wurden bei der Bewegung des Greifers beschädigt.

Die ersten Testreihen mit den Latexüberzügen verliefen einwandfrei und es wurde ein AR-Muster, wie in Kapitel 6 erwähnt, auf der Oberfläche des Greiferüberzuges angebracht. Da es sich immer noch um einen Prototyp handelt, wurde das Pattern mittels Schablone und Permanentmarker auf die Oberseite gezeichnet. Abbildung 5.6 zeigt das Ergebnis des Prototyps.

Wie im späteren erklärt, sollte nach Bewährung des Ansatzes und der Modifikation der Hardware der komplette Entwurfsprozess und Fertigungsprozess präziser unter Verwendung eines CAD-Programmes, Werkzeugmaschinen oder Ähnlichem erfolgen.

5.2. CAD-Modell

Wie oben erwähnt sollte der Entwurfsprozess auf eine höhere Ebene gehoben werden. Dazu sollte eine neue Gussform hergestellt werden, welche die Herstellung des Überzuges in einer vordefinierten Dicke ermöglichen soll. Zu diesem Zweck wurde der Greifer selbst, nach den Vorgaben und Außenabmessungen der Firma *Festo Ges.m.b.H.* (siehe Referenz [11]), als fester Block und eine Außenschale mit Hilfe des CAD-Programms *SolidEdge*

5. Hardwaremodifikation

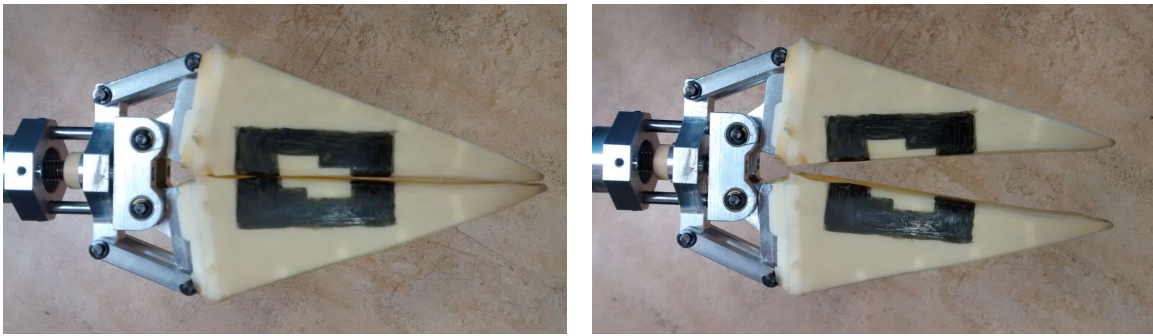


Abbildung 5.6.: Latexüberzug inkl. Pattern. Die fertigen Latexüberzüge wurden auf den Greifer gespannt und das oben definierte top-Pattern mittels permanenten Stiften aufgetragen. Die Zustände des teilgeöffneten und geschlossenen Greifers sind dargestellt.

entworfen. Die Außenschale hat zwei wichtige Zwecke. Auf der einen Seite werden die Abmessungen exakt begrenzt, sodass ein Überzug mit konstanter Dicke entsteht, auf der anderen Seite sind dadurch glatte Oberflächen gewährleistet.

Die in SolidEdge entworfenen Bauteile sollten anschließend mittels 3D-Druck hergestellt werden. Dazu mehr im nächsten Kapitel. Als erster CAD-Entwurf wurde ein Modell aus zwei Teilen erstellt. Ein Teil stellt den Greifer inklusive der Aufnahme für die Zentrierung in der Schale dar, der zweite die komplette Schale. Mittels Zentrierbolzen wurde die exakte Passung zwischen der Schale und der Greiferaufnahme ermöglicht. Abbildung 5.7 veranschaulicht die beiden Teile.

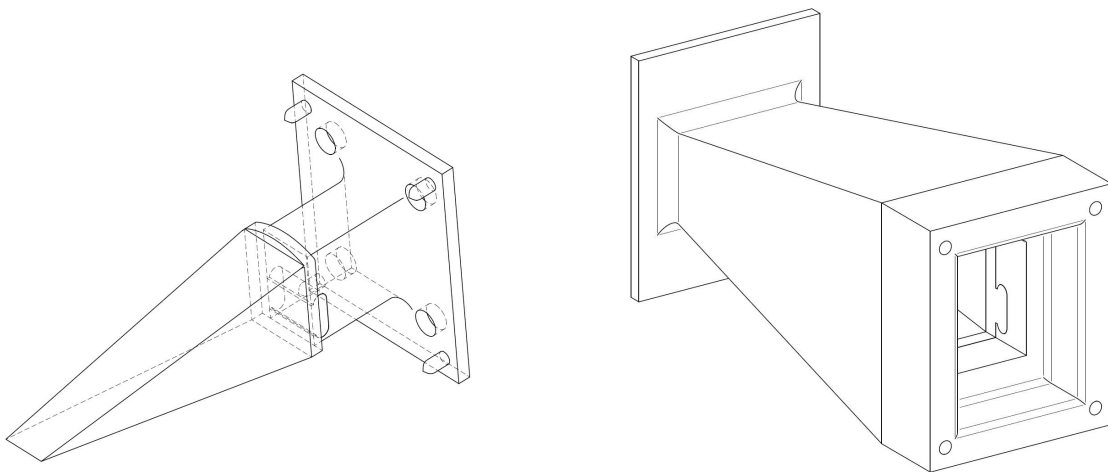


Abbildung 5.7.: CAD-Zeichnung der Gussform V1. Im linken Teil der Abbildung ist der Greiferfinger inklusive seiner Aufnahme und den Zentrierbolzen dargestellt. Der rechte Teil veranschaulicht den Korpus, das Negativ, in welchen die Überzugsmasse eingefüllt werden kann.

Die Idee war nun die Schale auf deren Fuß zu stellen, sie mit einem Material, Latex oder Ähnlichem, zu befüllen und diese mit dem Greifer und dessen Aufnahme zu verschließen. Dadurch ergibt sich eine optimale Passform des Überzuges mit definierter

Dicke und glatten Flächen. Das überschüssige Material kann über die vier Öffnungen an der Oberseite der Greiferaufnahme abrinnen.

Da zum ersten Mal mit 3D-Druck gearbeitet wurde, mussten ein paar Verbesserungen an der Gussform vorgenommen werden. Es kann viel Material beim Druck der Schale eingespart werden, da die Außenwand zu dick entworfen wurde. Außerdem stellte sich heraus, dass eine dünne Außenwand nicht nur Vorteile bzgl. der Stabilität hat sondern auch bessere Druckeigenschaften aufweist. Die höhere Stabilität beruht auf dem Fakt, dass die Drucksoftware so eingestellt ist, dass volle Objekte hohl gedruckt werden, wobei einzelne Stützstellen im Hohlraum des Objektes berechnet und gedruckt werden. Diese Einstellung ist durchaus sinnvoll wenn einerseits der Materialaufwand und somit das Gewicht und andererseits die Druckzeit bedacht wird. Dem gegenüber steht natürlich eine noch höhere Stabilität. Wird nun eine dünnere Wandstärke gewählt, kann die Stabilität bei gleicher Anzahl an Stützstellen deutlich erhöht werden, da der Abstand zwischen den beiden Flächen des Gegenstandes geringer ist.

Weiters konnte die Höhe des gesamten Modells verringert werden und somit erneut Material- und Zeitaufwand eingespart werden. Ein weiterer schwerwiegender Punkt ist, dass beim Herausnehmen des gegossenen Produktes, aus der ersten Version der Schale, nach der Aushärtezeit große Schwierigkeiten entstanden. Dies hängt einerseits mit den Klebeeigenschaften des jeweiligen Gussmaterials und andererseits mit der schweren Zugänglichkeit zum Endprodukt des Gusses zusammen. Das bedeutet, dass in der Schale nur eine verhältnismäßig kleine Öffnung an der Oberseite vorhanden ist. Dort wird der Greifer inklusive seiner Aufnahme eingesteckt, bzw. der Überzug entnommen. Auf Grund der guten Haftungs- und Klebeeigenschaften der meisten Materialien wird die Entfernung des Endproduktes erschwert. Deswegen wurde ein zweiter Entwurf gestartet. Um die zuvor genannten Komplikationen und Fehler zu beheben, wurde die Schale dünner gemacht, aus zwei Teilen gefertigt und der gesamte Aufbau in seiner Höhe verringert. Dies geschah durch Verkürzung der Greiferaufnahme und durch Verringerung der Wandstärke der Schale. Das Ergebnis ist in Abbildung 5.8 ersichtlich.

Das Herstellungsprinzip des Überzuges wurde im Vergleich zur ersten Version nicht verändert. Die beiden Schalenteile wurden aufgestellt, mit flüssigem Material befüllt und die Greiferaufnahme eingepasst.

5.3. 3D-Druck

Der 3D-Druck hat sich unter mehreren anderen Fertigungstechniken für diese Anwendung durchgesetzt, weil diese Methode der Herstellung von Objekten verhältnismäßig einfach und schnell funktioniert. Ein zusätzlicher Vorteil ist, dass die Firma *Blue Danube Robotics* eng mit der TU beim Hobbitprojekt zusammenarbeitet und über einen 3D-Drucker verfügt. Somit musste kein Kontakt zu einer externen Firma hergestellt werden und es konnten Kosten gespart werden.

In den letzten Jahren wurde der 3D-Druck stark weiterentwickelt, sodass es mittlerweile eine Vielzahl an Druckverfahren gibt. Alle Verfahren sind Teil des Themengebiets des Rapid-Prototypings, also des schnellen Produzierens von Prototypen. Das hier verwendete Verfahren wird Schmelzschichtung, auch *Fused Deposition Modeling*, genannt. Dabei wird ein Kunststoff in Form einer dünnen Leitung mit einer Heizdüse

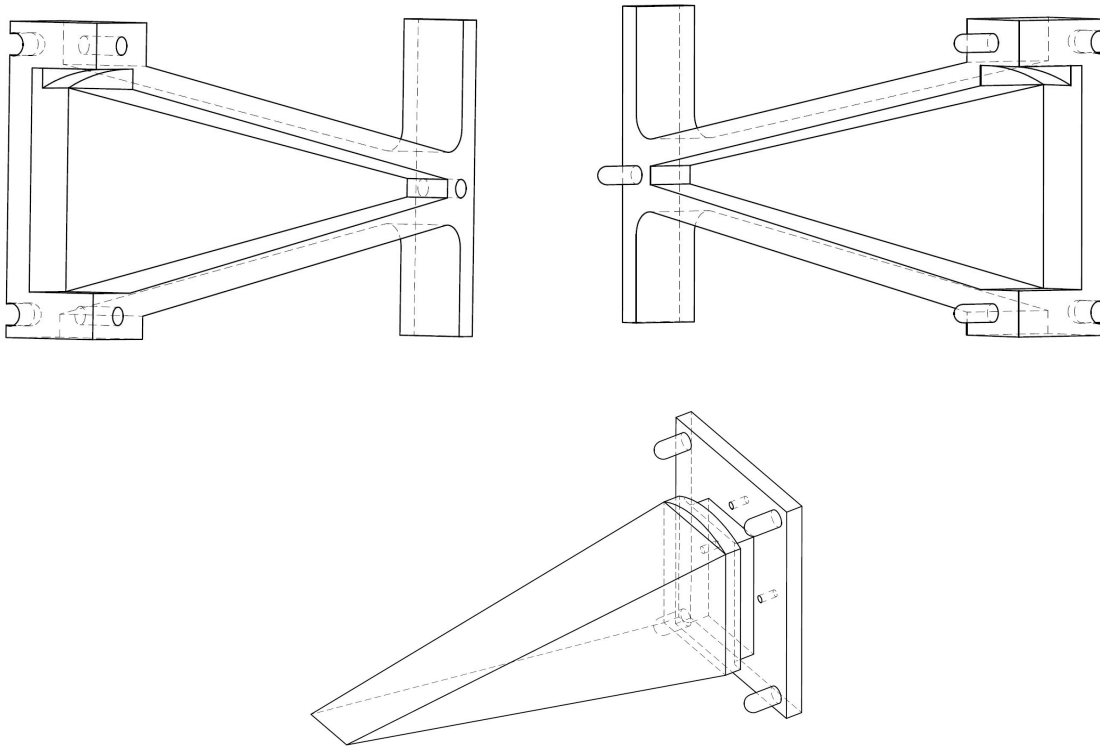


Abbildung 5.8.: CAD-Zeichnung der Gussform V2. In den oberen beiden Teilen der Abbildung sind die überarbeiteten Hälften des Korpus abgebildet. Dieser wird mit Überzugsmasse befüllt. Im unteren Teil ist der Greiferfinger inklusive der verkürzten Aufnahme ersichtlich.

geschmolzen und schichtweise auf das Druckbett aufgetragen. Die Kunststoffleitung ist hierbei auf einer Spindel aufgewickelt. Der Druckkopf, an dem die Düse angebracht ist, wird mittels zweier Motoren in der Ebene (x- und y-Achse) bewegt. Das Druckbett ist in der z-Achse beweglich. Ist eine komplette Schicht aufgetragen, fährt es ein Stück weiter nach unten. Somit kann die nächste Schicht aufgetragen werden. Für die Unterscheidung und genauere Erklärung des 3D-Drucks sei auf die Literatur [10] verwiesen.

Die zuvor vorgestellten Modelle für die Herstellung des Greiferüberzuges sind der Vollständigkeit halber in Abbildung 5.9 dargestellt.

5.4. Plastische Modellierung

Es wurde nun nach einem Material gesucht, welches ähnliche Eigenschaften wie Latex hat und zusätzlich einfacher zu verarbeiten ist. Naturlatex hat den Nachteil einer langen Aushärtezeit, kann kaum oder nicht gut gefärbt werden und hat eine begrenzte Lebensdauer auf Grund von Brüchigkeit. Durch den Kontakt bzw. die Kooperation des Instituts mit der Firma *Blue Danube Robotics* wurde ein geeignetes Material nach nur kurzer Zeit gefunden. Die sogenannte *Dragon Skin 20* der gleichnamigen Serie hat ideale Eigenschaften für diese Anwendung. Neben einer hohen Zug- und Reißfestigkeit

5. Hardwaremodifikation

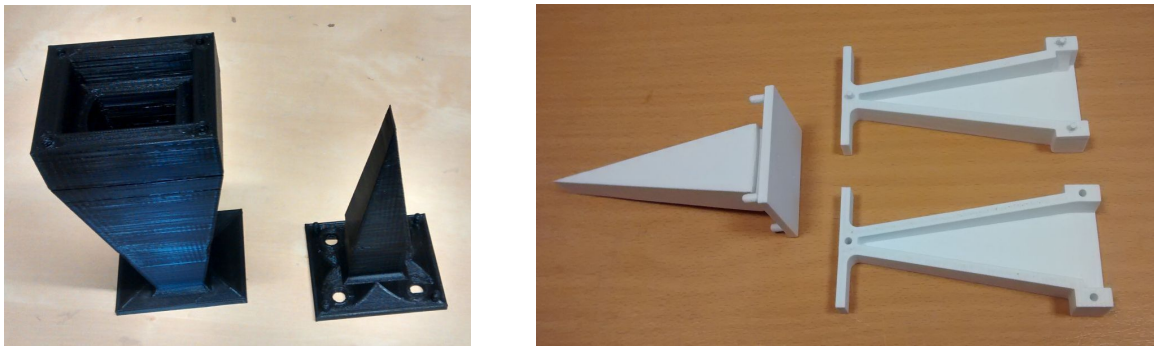


Abbildung 5.9.: 3D-Druck der beiden CAD-Modelle. Sowohl die schwarzen zwei Teile des ersten Ausdrucks, wie auch die weißen Teile der zweiten Version mussten nachbearbeitet werden.

ist eine Dehnung von mehr als 600%, laut Hersteller, möglich, bevor das Material reißt. Zusätzlich zu der deutlich kürzeren Aushärtezeit, ca. die Hälfte derer von Latex, ist auch der Einsatztemperaturbereich deutlich größer. Bei dem Produkt handelt es sich um ein Zweikomponentensilikon, welches nach Vermischung aushärtet und sowohl im Bereich des Formenbaus, im Theater- und Filmbereich für Hauteffekte oder ganze Kostüme, als auch in der Medizintechnik verwendet wird [26].

Nach zusätzlicher Recherche stellte sich heraus, dass es für dieses, annähernd durchsichtig-milchige, Material Farbzusätze gibt, welche schon im Mischprozess hinzugefügt werden können. Somit kann, falls notwendig, sogar die Farbe des Überzuges frei definiert werden. Außerdem kann auf die Oberfläche eine Silikonfarbe aufgetragen werden, welche den gleichen Beanspruchungen standhält wie die Dragon Skin selbst. Diese ist beim gleichen Anbieter, der Firma *KauPo Plankenhorn e.K.*, erhältlich und kann mittels Airbrushtechnik aufgetragen werden. Aus all diesen Gründen kann von einem industrietauglichen Material gesprochen werden, welches den Belastungen des „Roboter greifens“ jedenfalls standhält. Nachdem der erste Prototyp gegossen wurde, stellte sich ein Problem heraus. Da das Zweikomponentenmaterial händisch vermischt wurde, konnten Luftpneinschlüsse im Material nicht vermieden werden. Diese Luftblasen stellen Schwachstellen im Material bzgl. dessen Reißfestigkeit dar, vor allem wenn bedacht wird, dass die Dicke des Überzuges gering sein sollte. Falls dieses Problem Einfluss auf den Greifalgorithmus haben sollte, kann hier Abhilfe geschafft werden, indem eine Vakuummischung in einem dafür vorgesehenen Gerät stattfindet. Um ein problemloses Ablösen des Überzuges von den 3D-gedruckten Teilen zu ermöglichen, wurden deren Oberflächen mit einem Mold-Release-Spray eingesprüht. Das Ergebnis der Modellierung ist in Abbildung 5.10 abgebildet.

5. Hardwaremodifikation



Abbildung 5.10.: *Dragon Skin* Überzug. Der, aus zwei Teilen bestehende, Korpus wurde mit transparenter *Dragon Skin 20* befüllt. Anschließend wurde der 3D-gedruckte Greiferteil hinzugefügt. Nach der Aushärtezeit des Zwei-Komponenten-Materials konnte der Überzug von den Gussteilen abgezogen und verwendet werden.

Von den in Kapitel 4 vorgestellten Lösungsansätzen wurden drei implementiert. Dabei entschied sich der Autor nach Vergleich der oben genannten Vor- und Nachteile aller Lösungsmöglichkeiten für die Variante der Greiferkennung anhand Kantenerkennung, SIFT-Vergleich und AR-Marker-Matching. Auch das Thema Flächenberechnung fließt in dieses Kapitel ein, wurde jedoch nicht implementiert.

6.1. Kantenerkennung

Zu Beginn wurde der Canny-Algorithmus nach [52] bzw. [46] verwendet, um einerseits eine Vorstellung von der Qualität der, von der Kopfkamera des Hobbits aufgenommenen, Bilder zu erhalten und andererseits zu überprüfen ob Kantenerkennung alleine, oder in Kombination mit einer weiteren Bildverarbeitungstechnik, als Lösung der Problemstellung ausreicht.

Der Canny-Algorithmus wurde gewählt, weil mit diesem schon im Übungsteil der Vorlesung *Komponenten der Automation* gute Erfolge erzielt wurden und dieser nach der Meinung des Autors für diese Anwendung anderen Kantenalgorithmen, wie Prewitt, Sobel oder LoG¹, überlegen ist. Der Canny-Algorithmus ist einer einfachen Filteroperation mittels Prewitt- oder Sobel-Algorithmus insofern überlegen, weil dieser aus mehreren Teilschritten besteht. Dies wird im Folgenden etwas genauer erklärt. Der Canny-Algorithmus besteht prinzipiell aus drei bzw. vier Teilen. Beginnend mit einer Gauss'schen Glättung erfolgt im zweiten Schritt die eigentliche Kantenerkennung mit einem Filteroperator und abschließend eine Gratverfolgung mit Hysterese-Schwellwerten. Es handelt sich hierbei um einen Non-Maxima-Suppression-Algorithmus (NMS) und den Hysteresis-Threshold-Algorithmus (HT). Die Gauss-Filterung verbessert die Unabhängigkeit gegenüber Rauschen, während beim NMS jene Punkte, welche nicht Teil einer Kante sind, gelöscht werden. Der abschließende HT-Algorithmus verwendet zwei Schwellwerte, welche den Bereich angeben in dem eine Kante als „sicher“ erkannt wird.

¹Laplacian of Gaussian [die Laplace-Ableitung der Gaussfunktion] (LoG)

6. Implementierung

Die einzustellenden Parameter sind die Standardabweichung der Gauss-Filterung, Sigma genannt, und die beiden Schwellwerte, oberer und unterer, der Gratverfolgung. Detailliertere Informationen über den verwendeten Canny-Algorithmus sind in [46] zu finden.

Wird der Canny- mit dem LoG- Algorithmus verglichen, fällt zu Anfangs auf, dass bei LoG die zweite Ableitung, die Nullstellen, für die Kantendetektion verwendet wird. Bei Canny wird nach den Minimal- bzw. Maximalwerten, also nach der ersten Ableitung, gefiltert. Im LoG Algorithmus wird nach möglichst zusammenhängenden Formen gesucht, sodass teilweise Kantenelemente fälschlicherweise verbunden werden.

Natürlich muss die Auswahl des Kantenerkennungsalgorithmus immer vom Anwendungsfall abhängig entschieden werden. Bei dem in Matlab implementierten Algorithmus wurde der Sigma-Wert und der untere Schwellwert variiert, um geeignete Richtwerte für diese Parameter zu finden. Der obere Schwellwert wird, für diese Anwendung, nicht berücksichtigt, weil dieser lediglich die Grenze für stärkere Kanten angibt. Sämtliche Kanten die stärker sind als der obere Schwellwert, werden herausgefiltert. Die Schwellwerte können als Grenzwerte eines Bandpassfilters verstanden werden. Nur jene Kanten, deren Stärke innerhalb der beiden Grenzwerte liegt, werden akzeptiert.

Die Außenkanten des Greifers stellen meistens die stärksten Kanten im Vergleich zu deren (nahen) Umgebung dar und dürfen auf keinen Fall weggefiltert werden. Darum wurde der obere Schwellwert nicht berücksichtigt. Da diese Parameter jedoch stark vom Szenario abhängig sind, wurde die automatische Parameterermittlung von Matlab hinzugezogen. Folgende Erkenntnisse konnten aus der Anwendung des Canny-Algorithmus mit unterschiedlichen Parameterwerten gewonnen werden:

- Abhängigkeit der Algorithmusparameter:

Die Parameter des Algorithmus sind von der Beleuchtung und Position des Greifers abhängig. In der folgenden Abbildung (Figure 6.1) sind in der oberen Zeile Aufnahmen des Greifers in Bodennähe und in der unteren Zeile Aufnahmen des Greifers in Tischnähe ersichtlich. Die Werte der verwendeten Parameter, Sigma und unterer Schwellwert, sind unter den Abbildungen eingetragen. Die Namensgebung der Abbildungen bezieht sich auf die Ausgangsposition, das verwendete Objekt und auf die Parameterwerte. Beispielsweise ist mit „upTurn700Close t=0.06 $\sigma=1.1$ “ die obere Startposition des geschlossenen Greifers mit einer Rotation des Armes um 700 Motorschritte, einem Schwellwert von 0.06 und einem Sigma von 1.1 bezeichnet.

6. Implementierung

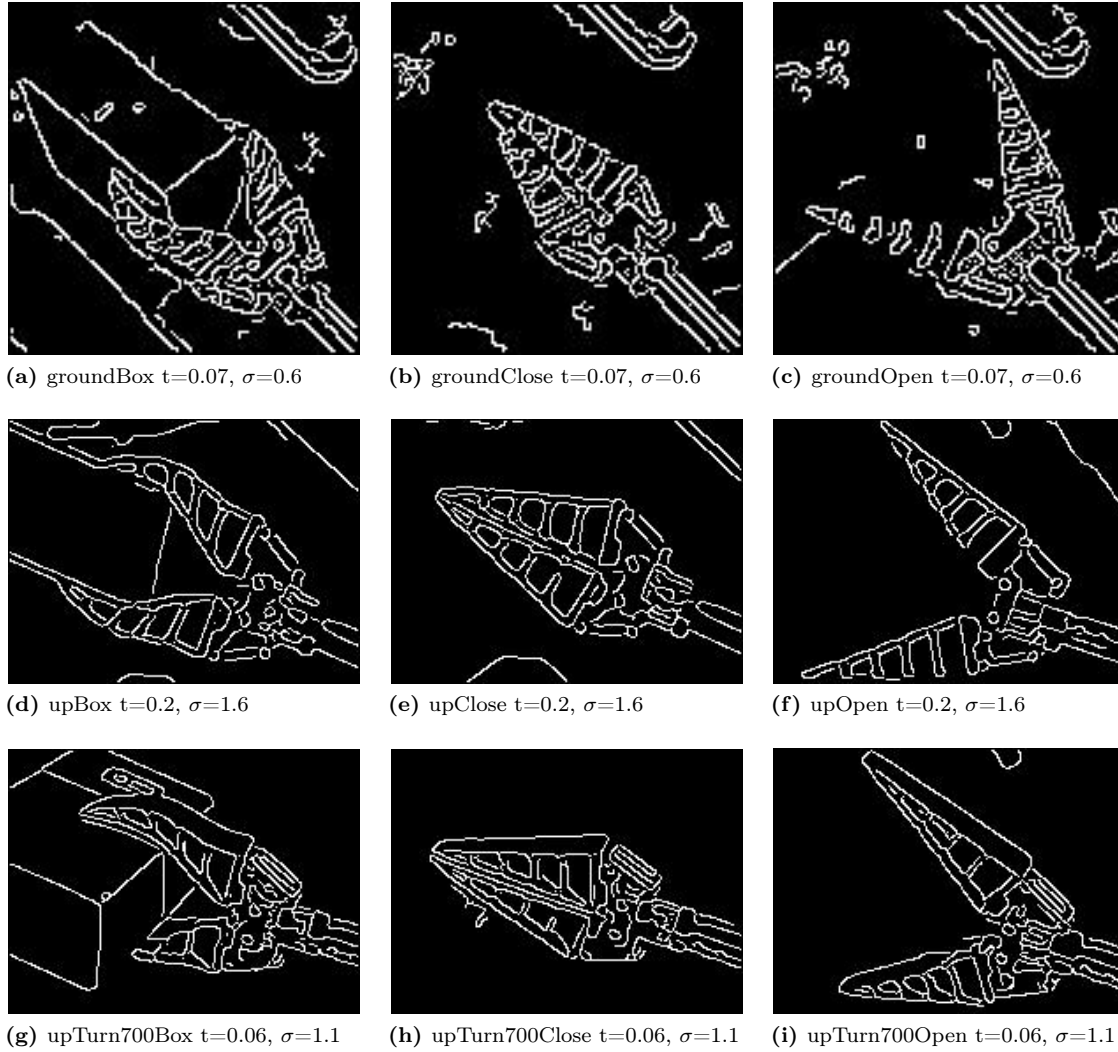


Abbildung 6.1.: Erkennungspositionen in Bodennähe vs. Tischnähe. Die Teilbilder 6.1a bis 6.1c veranschaulichen den Greifer in Bodennähe, die Teilbilder 6.1d bis 6.1i in Tischnähe. Dabei befindet sich in den linken drei Abbildungen (6.1a, 6.1d, 6.1g) jeweils eine Kartonbox im Greifer. In den mittleren Abbildungen (6.1b, 6.1e, 6.1h) ist der geschlossene und in den rechten drei Abbildungen (6.1c, 6.1f, 6.1i) ist der vollständig geöffnete Greifer dargestellt. Der Schwellwert und die Varianz sind unter den einzelnen Bildern angegeben.

6. Implementierung

- Parametereinfluss:

Der Glättungsparameter Sigma hat in diesem Fall weniger Einfluss auf das Ergebnis, als die Veränderung des Schwellwertes (siehe dazu Figure 6.2).

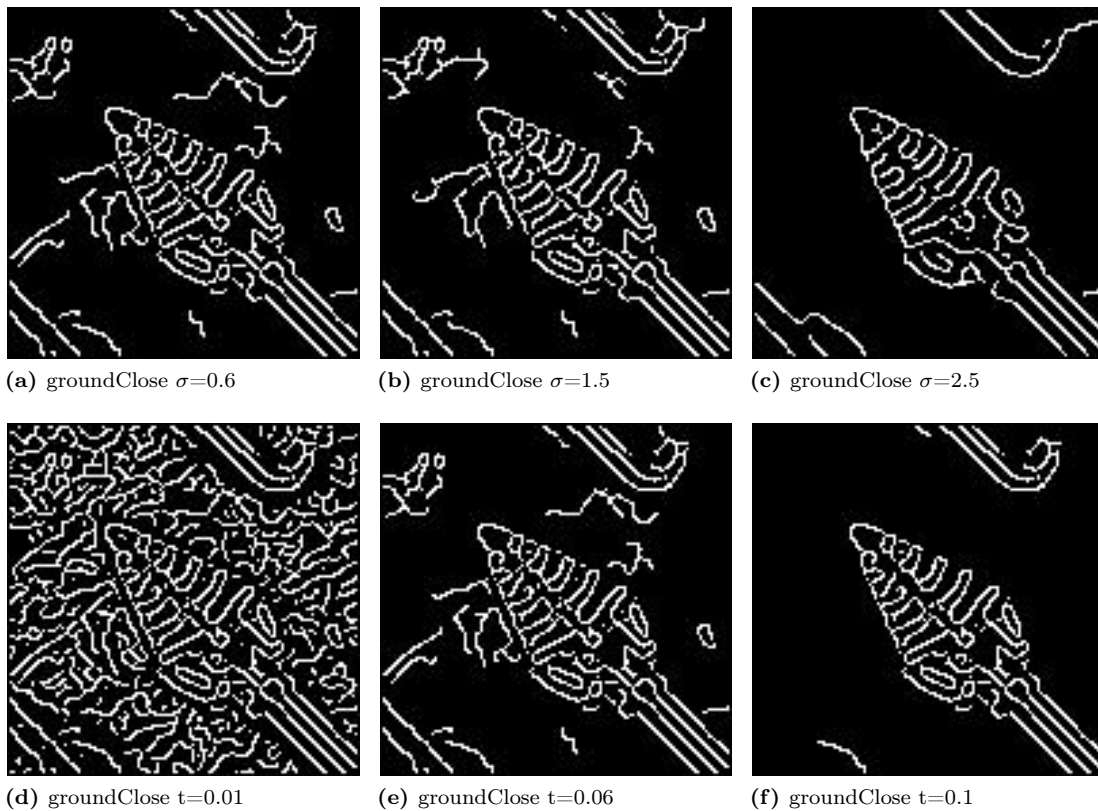


Abbildung 6.2.: Parametereinfluss Sigma vs. Schwellwert. Veranschaulicht wird der Einfluss der Varianz in den oberen drei (6.2a bis 6.2c) und der Einfluss des unteren Schwellwertes in den unteren drei (6.2d bis 6.2f) Teilbildern.

6. Implementierung

- Größe des Bildausschnittes

Die Abhängigkeit der Größe des Bildausschnittes bzw. der zu untersuchenden Region (meistens der Greifer selbst) von der Entfernung Kamera-Objekt erschwert die Greifererkennung. Dies liegt auf der Hand, da aufgrund der geringeren Distanz zu einem Objekt mehr Bildpunkte vorhanden sind. Figure 6.3 zeigt diesen Sachverhalt. In den linken beiden Abbildungen (6.3a, 6.3c) wurde eine Position in größerer, in den rechten beiden Abbildungen (6.3b, 6.3d) eine Position kleinerer Distanz zwischen Kamera und Greifer eingenommen. Auch hier sind die Parameterwerte unter den jeweiligen Bildern angegeben.



(a) groundHandy $t=0.07$, $\sigma=0.6$



(b) upHandy $t=0.2$, $\sigma=1.6$



(c) groundClose $t=0.07$, $\sigma=0.6$



(d) upClose $t=0.2$, $\sigma=0.6$

Abbildung 6.3.: Bildausschnittvergleich. Verglichen werden der geschlossene Greifer ohne Objekt und mit Handy in einer bodennahen und einer tischnahen Position. Es ist ein deutlicher Unterschied in der Qualität und der Größe des zu analysierenden Bildausschnittes feststellbar.

6. Implementierung

- Notwendige Armbewegung:

Der Arm muss, nach dem Zugreifen, in den meisten Fällen zumindest einer Rotationsbewegung ausgesetzt werden, damit die Kantenerkennung sinnvoll durchgeführt werden kann. Werden beispielsweise die Querstreben des Greifers als Entscheidungsmerkmal für die Greifererkennung verwendet, müssen diese gut sichtbar sein. Dieser Fakt ist in Figure 6.4 ersichtlich, da die Fischgräten (Querstreben) des Greifers durch die Armrotation im Bild verschwinden und die innere Greiffingerkante nicht mehr sichtbar ist. Unter den einzelnen Bildern ist der Rotationsgrad angegeben. Die Namensgebung erfolgt gleichermaßen wie schon oben erklärt.

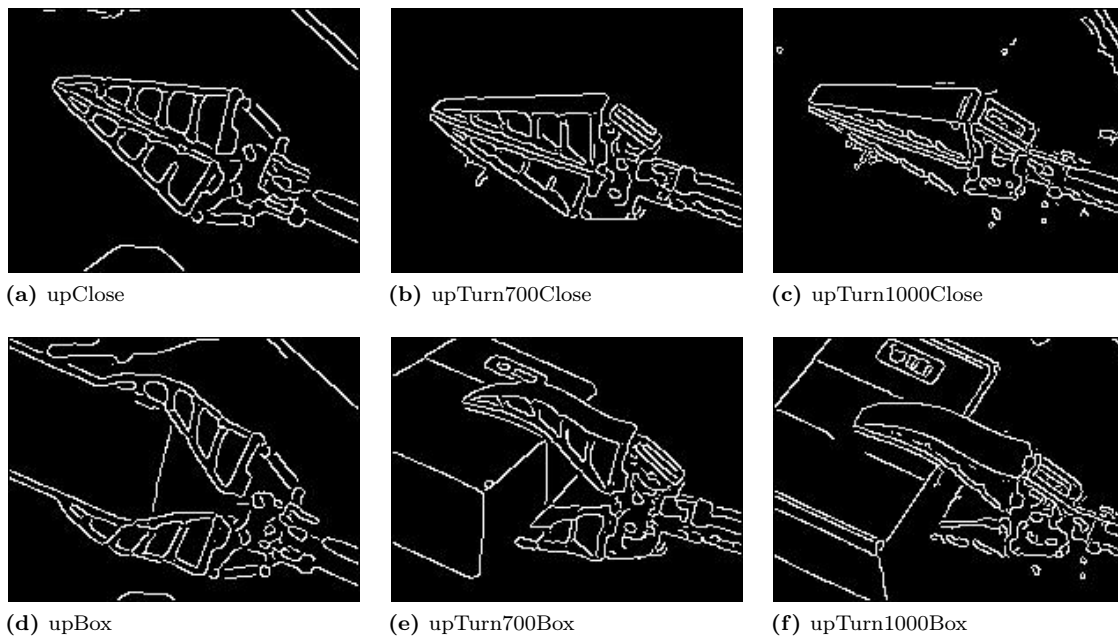


Abbildung 6.4.: Armrotation. Die Rotation des Armes, vor allem der Achse 5, hat starken Einfluss auf die Sichtbarkeit der Greifergräten und -kanten und somit auf die Analyse. In der oberen Bildreihe (6.4a-6.4c) ist der geschlossene Greifer ohne Objekt, in der unteren Bildreihe (6.4d-6.4f) der Greifer inklusive Kartonbox ersichtlich. Die Rotationswerte sind unter den einzelnen Bildern angegeben.

Der Lösungsansatz „Kantenerkennung“ musste nach obigen Ergebnissen aufgegeben werden. Dies liegt einerseits daran, dass die Ergebnisse der Filterung nicht zufriedenstellend waren und andererseits an der Tatsache, dass für diesen Ansatz eine Reihe von vor- und nachbereitenden Schritten notwendig ist. Um für diesen Anwendungsfall eine sinnvolle Kantenerkennung durchführen zu können, ist beispielsweise eine vorherige Segmentierung des Greifers unumgänglich. Nach erfolgter Kantenerkennung kann aber noch keine Auskunft darüber gegeben werden, ob die Objektergreifung erfolgreich war oder nicht. Um dies zu bewerkstelligen muss ein Entscheidungsalgorithmus implementiert werden, der bspw. anhand der Krümmung der inneren Fingerkante oder anhand der Lage der Querstreben des Greifers eine Aussage über den Greifstatus tätigen kann.

Durch die Verwendung mehrerer Algorithmen, die je einen Abschnitt in der Greifererkennung bewerkstelligen, ist einerseits die Leistungsfähigkeit, bezogen auf die Geschwindigkeit und Qualität, eingeschränkt und andererseits die Fehleranfälligkeit hoch.

Des Weiteren wird der Aufwand insofern höher als pro Algorithmus zusätzliche Parameter verwendet werden müssen, die ebenfalls angepasst und optimiert werden müssen, um ein brauchbares Ergebnis zu erzielen. Aus diesen Gründen wurde der Ansatz der Greifererkennung auf Kantenerkennungsbasis für diese Anwendung nicht weiter verfolgt.

6.2. SIFT

Der nächste Lösungsansatz wurde ebenfalls zuerst anhand einer Matlab-Implementierung getestet. Der Grund dafür war, dass die Matlab-Implementierung weitaus schneller möglich war, als die Implementierung in ROS bzw. mit PCL. Um Entwicklungszeit zu sparen, wurde somit die Matlab-Variante gewählt. Sollte der Algorithmus gute Ergebnisse liefern, würde die Implementierung in ROS mit PCL erfolgen. Im Gegensatz zur SIFT-Erkennung in Matlab, welche offline arbeitet, sollte die ROS-Implementierung online arbeiten. Eine detaillierte Erklärung des Algorithmus ist in Kapitel 3.1. gegeben. Dort ist ebenfalls der Verweis zur Fachliteratur vorhanden.

Bei dem verwendeten Algorithmus wurden die markanten Punkte, die Features bzw. Frames, von jedem Bild in einem Array gespeichert. Die errechneten Übereinstimmungen, die Matches, von beiden Bildern wurden ebenfalls in einem Array gespeichert. Des Weiteren wurde eine Übereinstimmungsrate, die Matchrate, errechnet. Diese stellt die gefundenen Features mit den gefundenen Matches in Relation. Abschließend wurde vom Autor selbst eine kurze Aussage darüber getroffen, ob die Übereinstimmung sinnvoll ist (✓) oder nicht (✗). Das bedeutet, dass abgewogen wurde, ob die Matches hauptsächlich am Greifer, bzw. auf dessen Finger, gefunden wurden oder in der Umgebung des Greifers und ob die Matches korrekt waren oder nicht.

Dabei wurde der Schwellwertparameter des DoG-Algorithmus verändert. Er gibt an wie viele markante Punkte gefunden werden, indem er die Maxima der Funktion des sogenannten Skalierungsraumes unterhalb dieses Schwellwertes herausfiltert. Eine kurze Definition des Skalierungsraumes bzw. dessen Funktion befindet sich in Kapitel 3.1. Obwohl mehrere Schwellwerte verwendet wurden, können die Ergebnisse anhand eines Schwellwertes exemplarisch beschrieben werden. Der eingestellte Schwellwert des DoG-Algorithmus betrug 0.005. In der nachfolgenden Tabelle (Tabelle 6.1) sind die Namen der Bilder, die gefundenen markanten Punkte, die Matches, die Matchrate und der Bewertungsgrad, welcher angibt ob die Matches als sinnvoll erachtet werden, angegeben. Dabei wurden alle Bilder in einer bodennahen Position aufgenommen.

Wird nur die Matchrate betrachtet, könnte der Trugschluss entstehen, dass die Werte eine gute Aussage über die Arbeitsweise des Algorithmus erlauben. Beispielsweise liegt es auf der Hand, dass ein Szenario in dem der Greifer ein Handy gegriffen hat, weitaus mehr Ähnlichkeit mit dem geschlossenen Greifer hat, als dies mit einer größeren Kartonbox der Fall ist. Selbst das Bild des offenen Greifers hat größere Ähnlichkeit mit dem des geschlossenen Greifers als jenes mit der Kartonbox, weil der offene Greifer keine Verformung aufweist. Bei genauerer Analyse ist jedoch zu erkennen, dass sowohl die gefundenen Features als auch die erkannten Matches hauptsächlich an nicht-relevanten Stellen, beispielsweise dem Handgelenk des Greifers oder in der Umgebung, gefunden wurden. Von den gefundenen Übereinstimmungspunkten waren bei diesem Vergleich maximal sechs Punkte richtig, welche jedoch nicht auf einem Finger des Greifers, sondern

6. Implementierung

Bild 1	Bild 2	Features 1	Features 2	Matches	MR	✓✗
groundClose	groundClose	166	166	166	100%	✓
	groundHandy	166	165	70	42.4%	✗
	groundBox	166	178	29	17.5%	✗
	groundOpen	166	175	40	24.9%	✗
	groundHandy	175	165	35	21.2%	✗
	groundBox	175	178	35	20.0%	✗
	groundOpen	175	175	175	100%	✓
groundHandy	groundBox	165	178	35	21.2%	✗

Tabelle 6.1.: SIFT-Vergleich in bodennaher Position

in der Umgebung oder auf dem hinteren Gelenk des Greifers zu finden waren.

Die Effizienz des verwendeten Algorithmus war hinsichtlich der Aufgabenstellung sehr schlecht. Die Ergebnisse werden von der nächsten Abbildung (Abbildung 6.5), welche zugehörig zur obigen Tabelle ist, unterstrichen. Es sei hier erwähnt, dass die Bildaufnahmen absichtlich mit der Microsoft Kinect getätigt wurden, um reale Bedingungen für den Erkennungsalgorithmus zu erstellen. Dadurch ist die Qualität der Bilder, in Zeiten von Kameras mit mehreren Megapixel, natürlich stark reduziert.

6. Implementierung

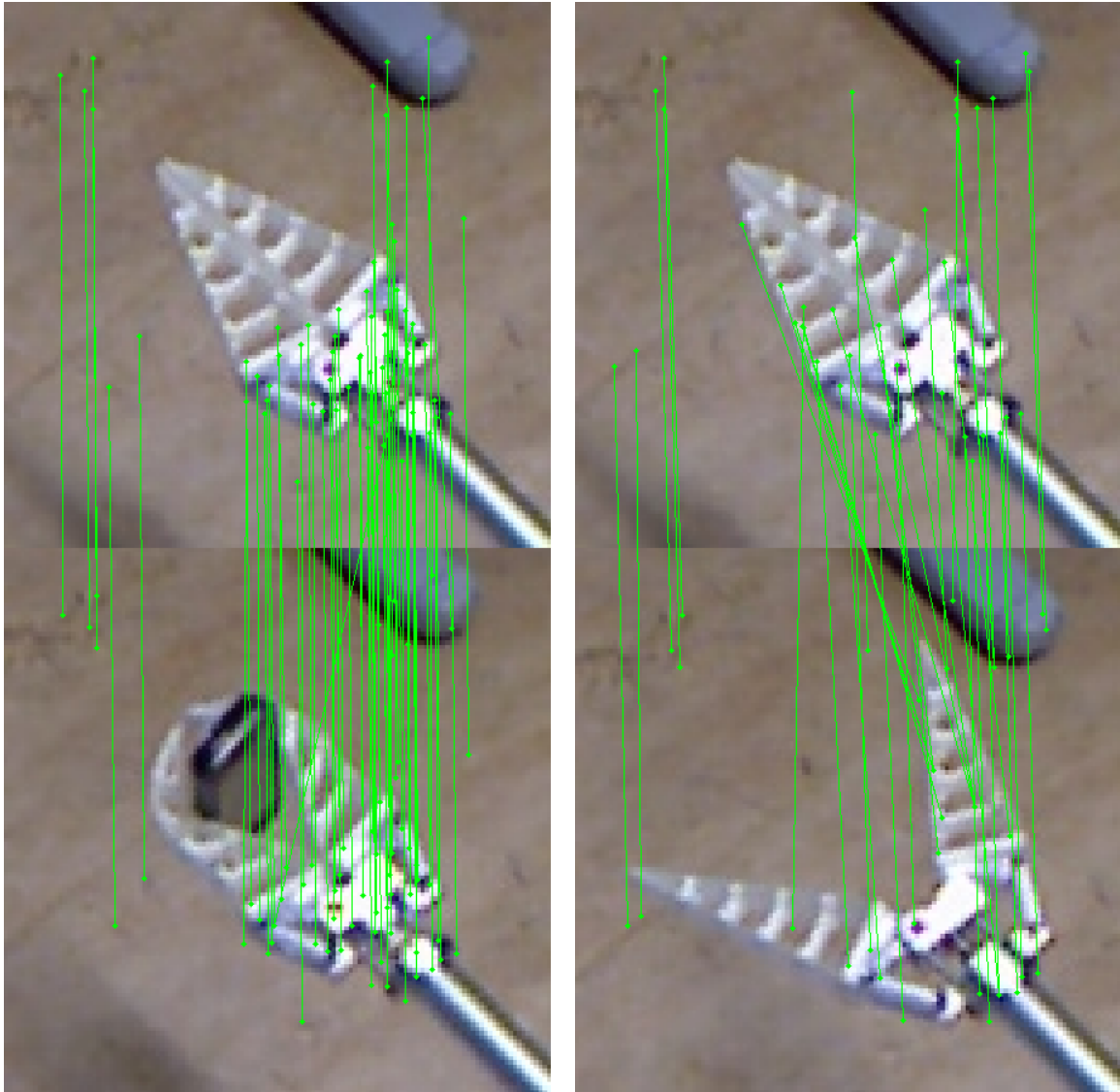


Abbildung 6.5.: SIFT-Vergleich in bodennaher Position zwischen einem ergriffenen Handy, dem geschlossenen und offenen Greifer. Die Übereinstimmung zweier Features (grüne Punkte), also ein Match, ist durch eine grüne Linien dargestellt.

6. Implementierung

Selbst in der oberen Greifposition, in welcher eine geringere Distanz zwischen dem Greifer und der Kamera besteht, waren die Ergebnisse nicht besser. Es konnten hier maximal zehn korrekte Matches gefunden werden, wovon nur einer direkt auf einem Finger des Greifer zu liegen kam. Die nachfolgende Tabelle (Table 6.2) und Abbildung (Abbildung 6.6) verdeutlichen das oben Gesagte.

Bild 1	Bild 2	Features 1	Features 2	Matches	MR	✓✗
upClose	upClose	312	312	312	100%	✓
	upHandy	312	344	136	43.6%	✗
	upBox	312	310	98	31.6%	✗
	upOpen	312	327	36	11.5%	✗
upOpen	upHandy	327	344	49	15%	✗
	upBox	327	310	44	14.2%	✗
	upOpen	327	327	327	100%	✓
upHandy	upBox	344	310	116	37.4%	✗

Tabelle 6.2.: SIFT-Vergleich in tischnaher Position

Um die Effizienz des Algorithmus zu steigern, wurden zwei Erweiterungen getätigt. Um den Einfluss der Umgebungsstruktur zu verringern wurde zuerst der SIFT-Algorithmus auf die kantengefilterten Bilder, ebenfalls mit dem Canny-Algorithmus, angewandt. Alleine durch die teils nicht zusammenhängenden Kanten im Kantenbild fiel die Rate der übereinstimmenden Punkte deutlich. Diese Erweiterung brachte somit keine Verbesserung.

Selbst durch händische Segmentierung des Greifers von seiner Umgebung mittels Photoshop und unter Verwendung besserer Ausgangsbilder, welche mit einer Digitalkamera aufgenommen wurden, konnte mit dem verwendeten Algorithmus keine markante Verbesserung erzielt werden. Wie schon während der Testreihe vermutet, bietet der Greifer zu wenig Struktur und markante Regionen, um den SIFT-Algorithmus zur zuverlässigen Greiferkennung zu verwenden. Es sei jedoch erwähnt, dass die Arbeitsweise des Algorithmus kontrolliert wurde, indem Szenarien mit einer Digitalkamera aufgenommen wurden. Dabei wurde beispielsweise ein Modellauto in unterschiedlichen Positionen gemeinsam mit anderen Objekten aufgenommen (siehe Abbildung 3.3). Trotz teilweiser Verdeckung des Autos konnte dieses zuverlässig in beiden Szenen anhand der übereinstimmenden markanten Punkte identifiziert werden.

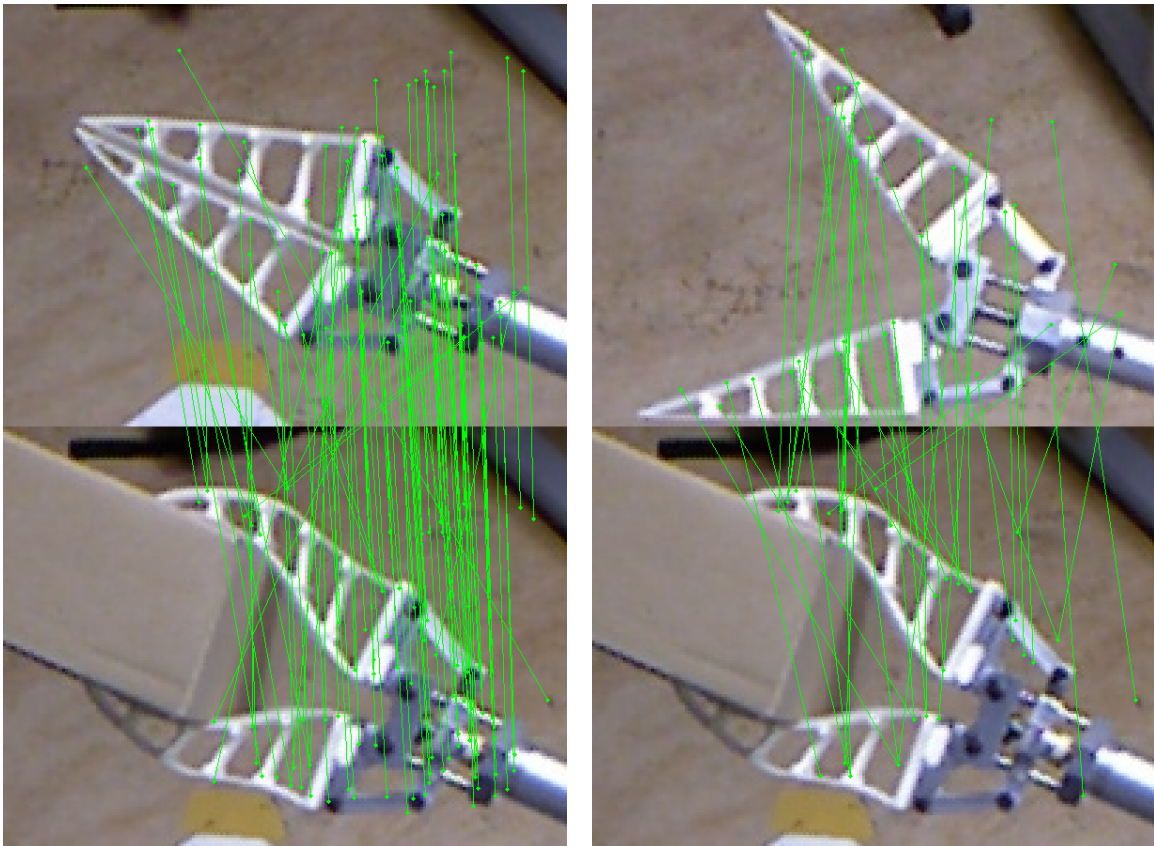


Abbildung 6.6.: SIFT-Vergleich in oberer Position zwischen einer ergriffenen Box, dem geschlossenen und offenen Greifer. Die Übereinstimmung zweier Features (grüne Punkte), also ein Match, ist durch eine grüne Linien dargestellt.

6.3. Punktwolkenanalyse und Verbesserungen

Nachdem die ersten beiden Ansätze anhand von Kantenerkennung und SIFT als unbrauchbar abgelegt wurden, wurde für den nächsten Ansatz, im Gegensatz zu den vorigen, das Augenmerk auf die 3D Daten, also die Tiefeninformation, und nicht auf 2D Daten der Szene gelegt.

6.3.1. Analyse der Punktwolken

Es wurden die Daten der Kopfkamera des Hobbits, in diesem Fall die Microsoft Kinect, als Punktwolke gespeichert und mit dem, von der PCL zur Verfügung gestellten, Pointcloud-Viewer (*PCD-Viewer*) angezeigt. Dabei ist es möglich die Daten unterschiedlich zu visualisieren. In den meisten Fällen wurde die Höhenlinienansicht verwendet, da anhand der Farbwerte direkt auf die Distanz zur Kamera geschlossen werden kann. Manchmal ist auch eine monochrome Farbansicht sinnvoll.

Der erste Gedanke war, dass ein Objekt allein durch Berechnung der höchsten (kleinen) Fläche in der direkten Umgebung des Greifers erkannt werden kann. Wird eine Fläche oberhalb der Greiferfinger, natürlich innerhalb der beiden Außenkanten der Finger, erkannt, wurde ein Objekt ergriffen. Dies gilt allerdings nur, wenn das Objekt nicht mehr auf einer Ebene (beispielsweise dem Boden oder einem Tisch) steht. Der eigentliche Kontakt zwischen Greifer und Objekt kann dadurch nicht bewiesen, sondern nur geschätzt, werden. Falls der Greifer beim Umschließen des Objektes nicht korrekt geschlossen wird, kann nur durch Bewegung des Armes nachvollzogen werden, ob ein Objekt tatsächlich ergriffen wurde. Dies erschien als einfache Lösungsmöglichkeit, brachte allerdings zusätzliche Schwierigkeiten hervor, wie anhand von Figure 6.7 ersichtlich ist.

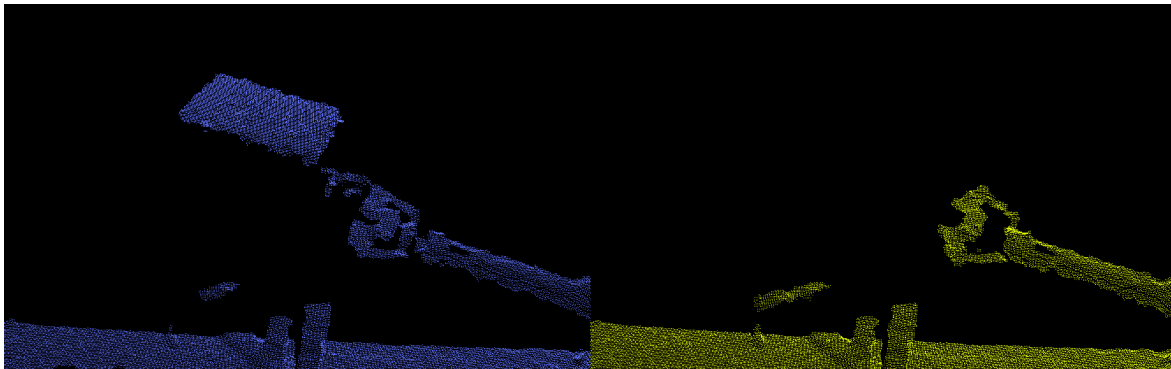


Abbildung 6.7.: Obere Greifposition (Monoansicht). Die Abbildung zeigt eine Aufnahme des Greifers im geschlossenen Zustand mit Kartonbox (linker Teil) und eine Aufnahme des offenen Greifers (rechter Teil). Es ist sofort zu erkennen, dass die Greiferfinger im rechten Bild für die Kamera nicht sichtbar sind.

Es sind vor allem zwei Probleme deutlich zu erkennen. Einerseits, und dies ist das schwerwiegendste Problem, können die Finger des Greifers, aufgrund ihrer schmalen Struktur nicht erkannt werden. Am Institut sind zwei verschiedene Greifer verfügbar, die sich nur in der Schichtdicke der Rahmen unterscheiden. Die Schichtdicken betragen 0.9 bzw. 1.1mm. Für die Kinect Kamera ist es dabei irrelevant, welcher der beiden Greifer

6. Implementierung

verwendet wird. Die Kamera ist aufgrund ihrer Auflösung nicht dafür geeignet solch dünne Strukturen zu erkennen. Zur Veranschaulichung des Greifers sei auf die Kapitel 1 und 5 verwiesen. Wird von der Sensorgröße der Kamera von 640×480 Pixel ausgegangen, kann leicht abgeschätzt werden, dass, bei der Aufnahme einer normalen Szene, selbst wenn sich der Greifer in einer annehmbar hohen Position befindet, deutlich weniger als ein Pixel/mm² für die Tiefeninformation übrig bleibt. Dabei wurde vorausgesetzt, dass die Kamera annähernd orthogonal auf die Oberseite, also jene Seite, wo die Querstreben sichtbar sind, blickt. Dies beweist, dass der Greifer keine oder nur sehr wenig Punkte in der Punktwolke besitzt. Auf der anderen Seite ist der Ansatz über die Höhendetektion zwar passend für größere Objekte, bei schmalen und vor allem kleinen Objekten wird dieser Ansatz jedoch nicht zum gewünschten Ergebnis führen. Dabei ist es annähernd unwichtig, ob sich der Greifarm in Bodennähe oder näher bei der Kamera befindet, wie Figure 6.8 zeigt.

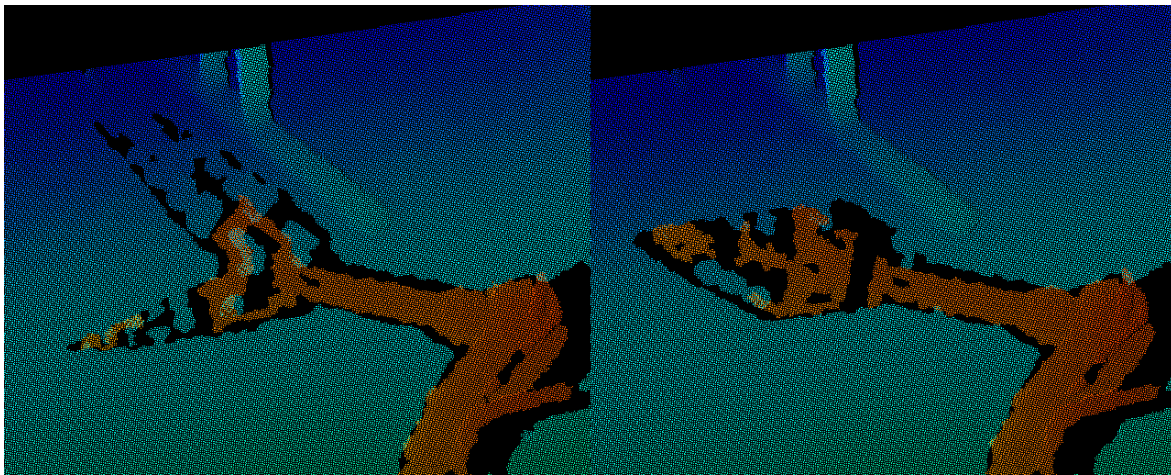


Abbildung 6.8.: Obere Greifposition mit leicht geneigtem Greifer in offener (linker Teil) und geschlossener (rechter Teil) Position (Höhenlinienansicht). Die Probleme der Sichtbarkeit der Greiferfinger werden hier verdeutlicht. Es ist an diesen Stellen zu wenig oder keine Tiefeninformation vorhanden.

Wird der Greifer allerdings gedreht, können die Querstreben und Außenkanten (besser) erkannt werden. Wie anhand von Figure 6.9 ersichtlich ist, ist der Greifer in der Punktwolke deutlich erkennbar. Selbst wenn sich eine kleine Kartonbox im Greifer befindet, kann aufgrund der Tiefeninformation auf den oberen Finger des Greifers geschlossen werden.

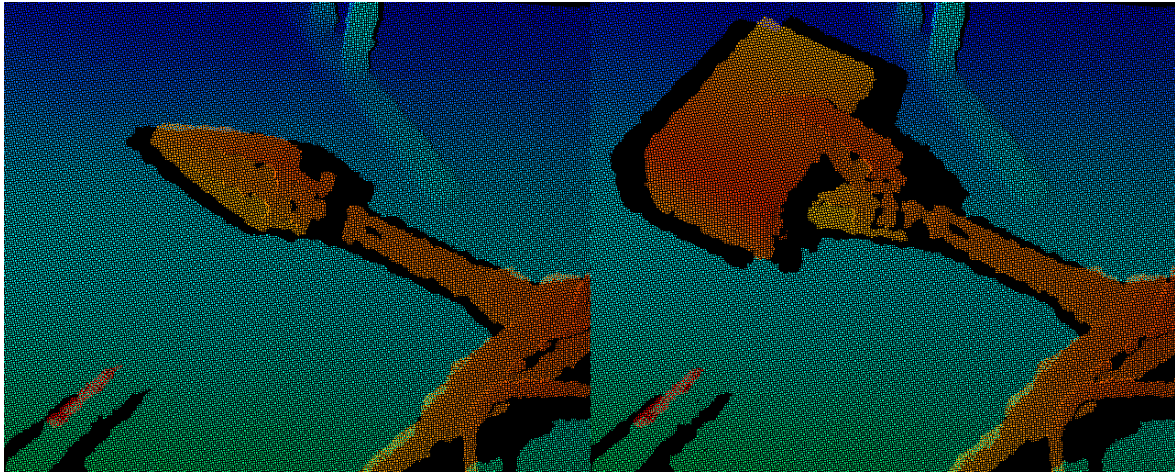


Abbildung 6.9.: Obere Greifposition mit deutlich gedrehtem Greifer in geschlossener Position (linker Teil) und mit einer Kartonschachtel (rechter Teil) in der Höhenlinienansicht. Der gesamte Greifer kann durch die Rotation, dies führt zu einer Veränderung der Perspektive, visuell erfasst werden.

Abbildung 6.10 verdeutlicht das oben gesagte hinsichtlich der besseren Erkennung des Greifers falls dieser bezogen auf die Kamerasicht geneigt wird.

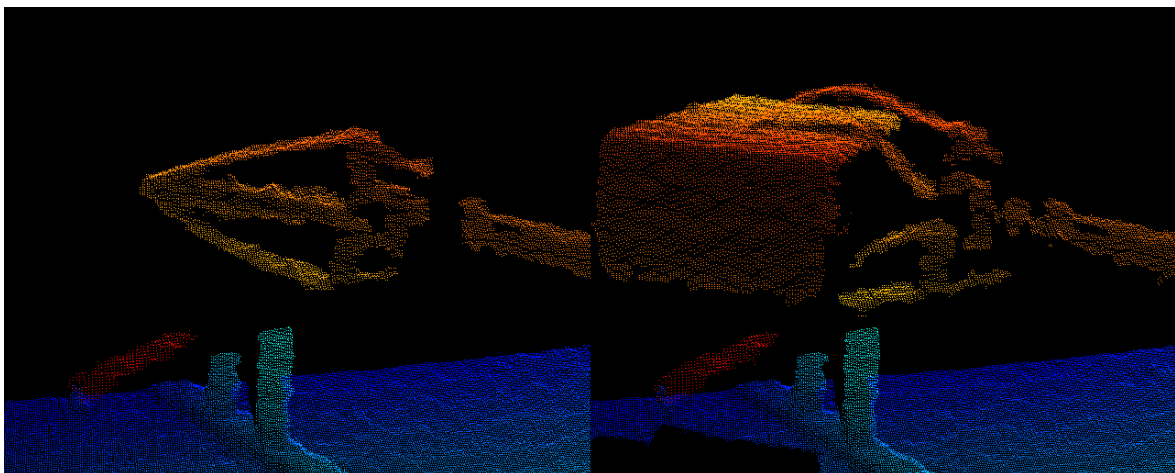


Abbildung 6.10.: Weitere Greiferposition mit geneigtem Greifer in geschlossener Position ohne Objekt (linkes Teilbild) und mit einer Kartonschachtel (rechtes Teilbild) in der Höhenlinienansicht.

Ein weiterer oder zusätzlicher Ansatz war es nicht bzw. nicht nur die höchste Fläche zu detektieren, sondern die Fläche selbst zu berechnen. Eine direkte Draufsicht auf die Oberseite des Greifers kann bei größeren Objekten genügen, um eine Aussage über den Greifstatus treffen zu können. Bei großen Objekten ist es ausreichend die Fläche des Greifers zu berechnen und anhand eines Schwellwerts, zu entscheiden ob ein Objekt ergriffen wurde. Der Schwellwert gibt an, ab welcher Flächengröße nicht nur der Greifer alleine sondern ein zusätzliches Objekt erkannt wurde. Jedoch ist auch hier eine Bewegung des Armes notwendig, um eine möglichst sichere Aussage treffen zu können. Die Problematik von kleinen Objekten besteht weiterhin. Außerdem muss, in Abhängigkeit des Objekts, entschieden werden von welcher Seite der Greifer betrachtet

wird (Seitenansicht oder Draufsicht) damit der Algorithmus brauchbare Ergebnisse liefert. Eine Seitenansicht ist bspw. ausreichend, falls sich ein großes Objekt, wie eine Kartonbox, im Greifer befindet. Im Gegensatz dazu muss für die meisten kleineren Gegenstände die Draufsicht gewählt werden.

6.3.2. Verbesserungen

Um das prinzipielle Problem zu beheben, dass vom Greifer selbst nur sinnvolle 2D Daten vorhanden sind und die Tiefeninformation fehlt, wurde eine Hardwaremodifikation vorgenommen. Diese ist in Kapitel 5 dargelegt. In diesem Unterabschnitt sollen lediglich die sich daraus ergebenden Verbesserungen bzgl. der visuellen Erfassung gezeigt werden.

Wie schon im vorigen Kapitel erwähnt, wurde eine Art Abdeckung für den Greifer konzipiert. Um zu Anfangs Kosten zu sparen wurden einfache Luftballone verwendet. Schon alleine durch diese nicht passgenauen aber gut formbaren Überzüge war der Greifer sehr deutlich in der Punktwolke zu sehen. Diese massive Verbesserung führte dazu, dass der Greifer als Fläche erkannt werden konnte (siehe Figure 6.11 und Figure 6.12). Für die Tiefeninformation ist die Verwendung der Abdeckung eine große Verbesserung, da der Greifer nun als Fläche erkannt wird und genügend Tiefeninformationen, sowohl für die untere Initialansicht als auch die obere Position, vorhanden sind.

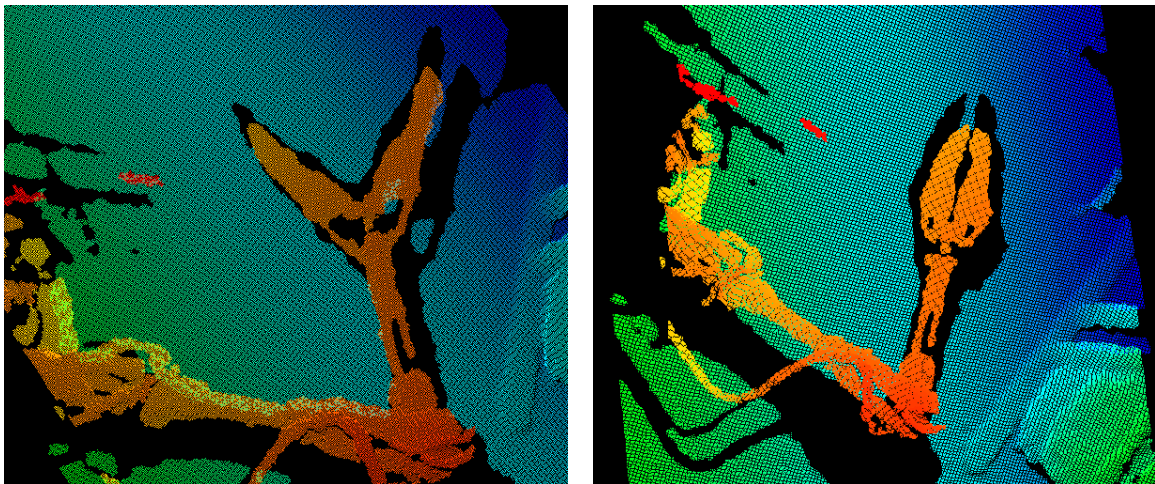


Abbildung 6.11.: Greifer mit simplem Überzug V1 in offener (links) und geschlossener (rechts) Position. In den Punktwolken ist die Verformung des Greifers auf Grund des Überzuges erkennbar.

Auf Grund der Form des Überzuges ist erkennbar, dass der Greifer nicht exakt überzogen wurde sondern, wegen seiner Flexibilität, leicht verformt wurde. Dies wurde jedoch durch Maßanfertigung eines Überzuges, wie in Kapitel 5.4 beschrieben, verbessert.

6. Implementierung

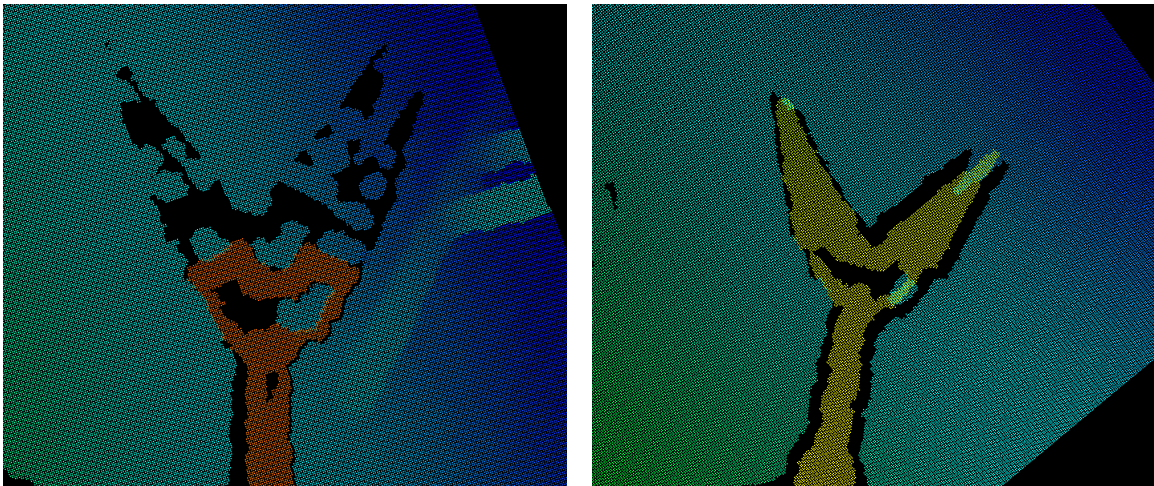


Abbildung 6.12.: Geöffneter Greifer ohne (links) und mit (rechts) Überzug V2. Direkter Vergleich der beiden Punktwolken in unterschiedlichen Positionen. Erst durch die Verwendung des Latex-Überzuges kann der Greifer und seine exakte Außenkontur ohne Verformung erkannt werden.

Mit der Abdeckung der Greiferoberfläche war der Grundstein für die weitere Arbeit gelegt. Nicht nur für diese Diplomarbeit sondern für mehrere Funktionen und Algorithmen des Hobbits, die mit dem Greifarm arbeiten, sollte diese Hardwaremodifikation Vorteile bringen. Nun konnte der Greifer in jeder Position, die im Sichtbereich der Hobbitkamera liegt, (Selbst-)Okklusion ausgeschlossen, erkannt werden. Die Außenkonturen des Greifers waren nun ebenfalls weitaus besser erkennbar. Anhand eines Vergleichs von Figure 6.7 mit Figure 6.13 ist ersichtlich, dass der Greifer eindeutig anhand der Tiefeninformation von der Kartonschachtel zu unterscheiden ist.

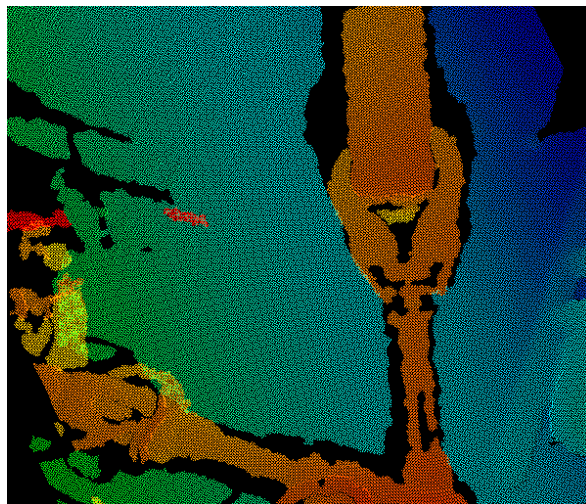


Abbildung 6.13.: Greifer in oberer Position mit Kartonbox und Überzug V2.

6. Implementierung

Nachdem der Greiferüberzug, wie in Kapitel 5.1 beschrieben, verbessert wurde – der Luftballon wurde durch einen handgefertigten Latexüberzug ersetzt – wurden erneut Punktwolken aufgenommen. Auf diesen ist die deutliche Verbesserung der Visualisierung des Greifers gut ersichtlich. Durch den verwendeten Latexüberzug, der im Vergleich zum Luftballon-Prototyp weitaus besser an den Greifer angepasst war, konnte der Greifer bei verschiedenen Rotationswinkeln gut erkannt werden. Die folgende Abbildung (Figure 6.14) veranschaulicht dies. Sie zeigt die Punktwolken (Höhenlinienansicht) des Greifers mit verschiedenen Objekten. Der Vergleich der Punktwolken des offenen und geschlossenen Greifers mit Objekt, wobei sich in den Teilbildern ein Kugelschreiber (links oben), eine Teebox (rechts oben) und ein Handy (rechts unten) im Greifer befinden, ist dargestellt.

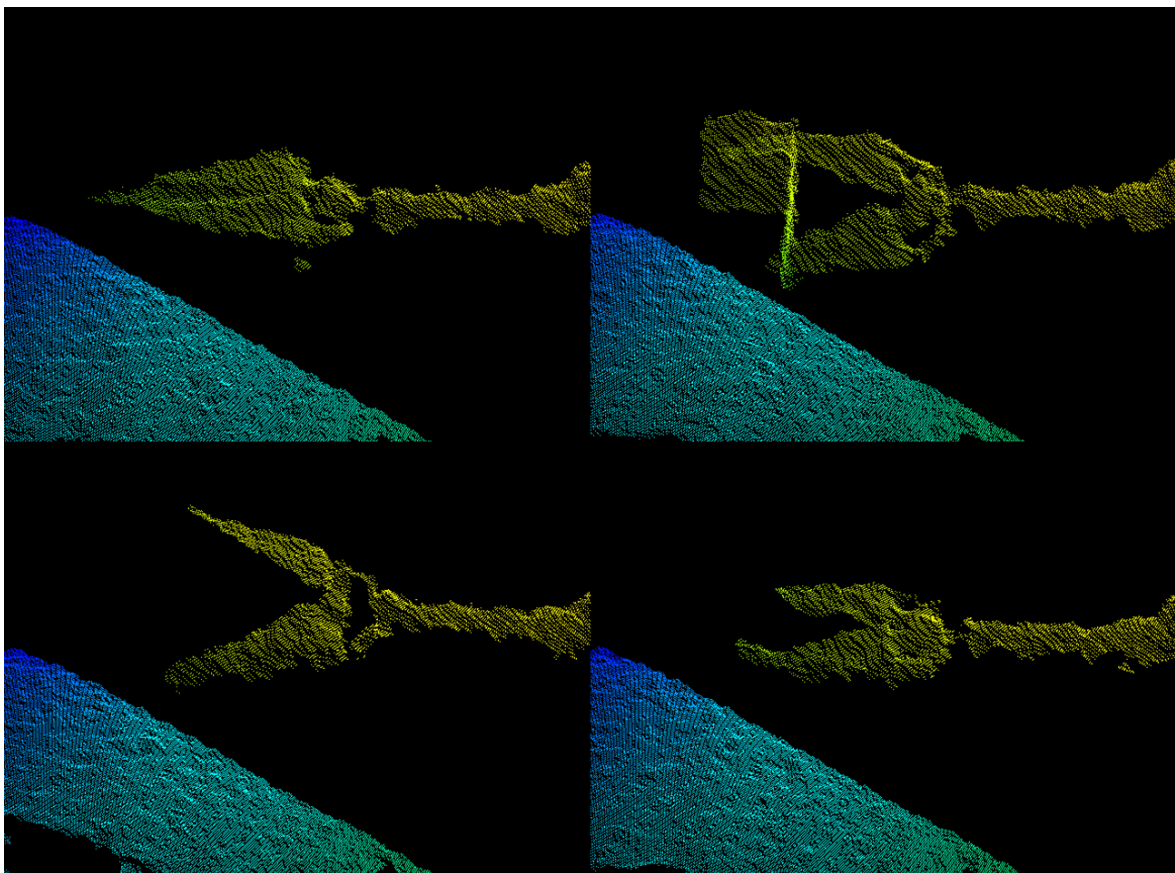


Abbildung 6.14.: Greiferpunktwolken mit geschlossenem (inkl. Objekten) und offenen Greifer (links unten). Der Kugelschreiber im linken oberen Teilbild ist kaum erkennbar. Im Vergleich dazu ist die Teebox im oberen rechten Teilbild gut sichtbar. Das rechte untere Teilbild zeigt das Ergreifen eines Handys.

Auf Basis dieser Verbesserung wurden die, in den vorigen Kapiteln vorgestellten, Ideen aufgegriffen und erneut einer Analyse unterzogen. Begonnen wurde mit der Berechnung der Flächennormalen anhand der Punktwolke (Figure 6.15). Die Berechnung wurde durch den, in der PCL inkludierten, PCD-Viewer durchgeführt. Dieser dient zur Anzeige von Punktwolken und kann mit unterschiedlichen Parametern aufgerufen werden. Trotz der Veränderungen wurde der Ansatz, die Greifererkennung auf Grund der Flächenberechnung oder der Flächennormalen der Punktwolke zu analysieren,

6. Implementierung

verworfen. Die Hauptgründe dafür wurden schon in Abschnitt 6.3.1 angegeben und haben ihren Ursprung darin, dass kleine und flache Objekte erhebliche Probleme für die Erkennung darstellen. Beispielsweise kann ein Radiergummi kaum erkannt werden, da die Flächenänderung des Greifers minimal ist (je nach Orientierung des Radiergummis) und dieser nicht aus der Greiferoberfläche herausragt. Nichts desto trotz kann diese Art der Greifererkennung für größere Objekte als simple und gute Variante verwendet werden.

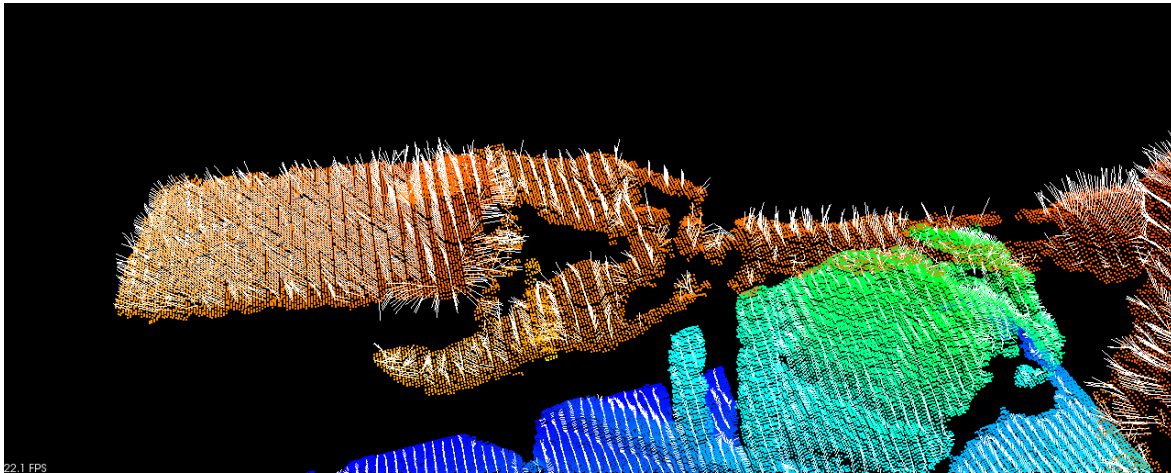


Abbildung 6.15.: Flächennormalen in der Punktwolke des überzogenen Greifers. Die Flächennormalen wurden durch den PCD-Viewer automatisch berechnet und können zur Greifererkennung genutzt werden. Dabei spielt deren Orientierung eine zentrale Rolle. Die Normalen sind durch kleine weiße Geraden mit einheitlicher Länge dargestellt.

Im nächsten Schritt wurde die oben vorgestellte Variante der Kantendetektion mittels Canny-Algorithmus erneut durchgeführt. Wie Abbildung 6.16 zeigt, war auch hier eine deutliche Verbesserung der Kantenbilder erkennbar. Der Parameter Sigma wurde hierbei automatisch eingestellt, der untere Schwellwert betrug 0.2. Sowohl die Außenkanten als auch die Innenkanten des Greifers sind in allen Szenarien erkennbar. Diese Kantenbilder könnten zur Greifererkennung mittels Berechnung der Krümmung der Innen- und/oder Außenkanten oder Ähnlichem verwendet werden. Bei genauerer Betrachtung ist ersichtlich, dass die Krümmungsunterschiede der Außenkante zwischen einem kleinen Objekt, wie beispielsweise dem verwendeten Handy, und dem geschlossenen oder offenen Greifer hauptsächlich im vorderen Bereich des Greifers auftreten. Bei größeren Objekten kommt es zu der Ausprägung eines Wendepunktes im Krümmungsverlauf, welcher markant für die Greifererkennung sein kann. Die Krümmung der Greiferkanten alleine kann allerdings nicht zur Unterscheidung zwischen offenem und geschlossenem Greifer herangezogen werden, da es zu keiner Verformung des Greifers kommt. Hierfür muss die Lage und Orientierung der Kanten berücksichtigt werden. Zusätzlich hat auch hier die Bildgröße einen erheblichen Einfluss auf das Resultat, sodass bei kleinen Objekten die Unterschiede in der Greiferkrümmung nur marginal sind. Aus diesen Gründen wurde diese Variante ebenfalls ad acta gelegt.

Letztlich wurde auch der SIFT-Ansatz aufgegriffen. Dabei wurden erneut Bilder mittels Kopfkamera des Hobbits aufgenommen und analysiert. Die Ergebnisse waren, auf Grund der bekannten Arbeitsweise des SIFT-Algorithmus und der vorigen Ergebnisse,



Abbildung 6.16.: Ergebnisse des Canny-Algorithmus bei überzogenem Greifer mit Überzug V1 (von links nach rechts, mit einer Kartonbox, geschlossener, offener Greifer und mit einem Handy).

schon zu anfangs klar und sollten durch ein paar wenige Experimente bestätigt werden. Durch die Abdeckung der Greiferoberfläche waren noch weniger markante Punkte am Greifer zu finden als ohne dieser. Dies liegt auf der Hand, da es sich bei der Abdeckung jedes Fingers um eine einfarbige und strukturlose Fläche handelt, welche selbst für das menschliche Auge keine markanten Punkte, außer deren Kanten, aufweist. Wie schon zuvor wurde der SIFT-Ansatz als Lösungsmöglichkeit aufgegeben.

Durch die Probleme und Nachteile des SIFT-Algorithmus für diesen Anwendungsfall wurde allerdings ein neuer Ansatz forciert bzw. entdeckt. Der Grundgedanke dazu ist, den Greifer mit markanten Punkten bzw. mit einer Struktur zu versehen und wird im folgenden Kapitel erklärt.

6.4. Pattern Matching mit AR-Marker

Die Ergebnisse der letzten Kapitel zeigten, dass für die Erkennung des Greifers und der damit verbundenen Greifererkennung markante Bereiche notwendig sind. Aus diesem Grund wurden verschiedene Überlegungen hinsichtlich der Strukturierung der Oberfläche des Greifers getätigt. Nach einiger Recherche und Überlegung, die von einfachen Linien, bunten Kanten und großen Farbpunkten bis hin zu komplizierten Strukturen und Marmorierungen der Oberfläche reichten, wurde der Ansatz der AR-Marker (siehe Kapitel 3.2) aufgegriffen. Am Anfang wurde eine Testreihe gestartet, um die prinzipiellen Möglichkeiten und Beschränkungen dieses Ansatzes zu überprüfen. Dazu wurde der provisorische Überzug auf dem Greifer belassen und ein Marker darauf angebracht. Wichtig dabei ist, dass die Anforderungen an das Muster eingehalten werden (siehe Kapitel 3.2). Zu Beginn wurden jeweils drei quadratische und drei rechteckige Patterns entworfen. Die Patterns sind in Abbildung 6.17 und Abbildung 6.18 dargestellt.

Nachdem diese Muster erstellt wurden und als PNG-Datei gespeichert wurden, musste eine Konvertierung durchgeführt werden, die die Bilddateien in PAT-Dateien umwandelt. Diese Notwendigkeit bestand, damit der Matching-Algorithmus die Muster interpretieren kann. Zu diesem Zweck wird das Originalbild in vier verschiedenen Positionen gespeichert. Dabei handelt es sich um das Originalbild, die um 90 Grad, 180 Grad und 270 Grad rotierte Ansicht. Alle vier Ansichten werden in einer Datei gespeichert. Dies dient der Rotationsinvarianz des Algorithmus. Die Konvertierung erfolgte mit einem Online-Tool im Internet (tarotaro.org). Die Optionen hierfür waren

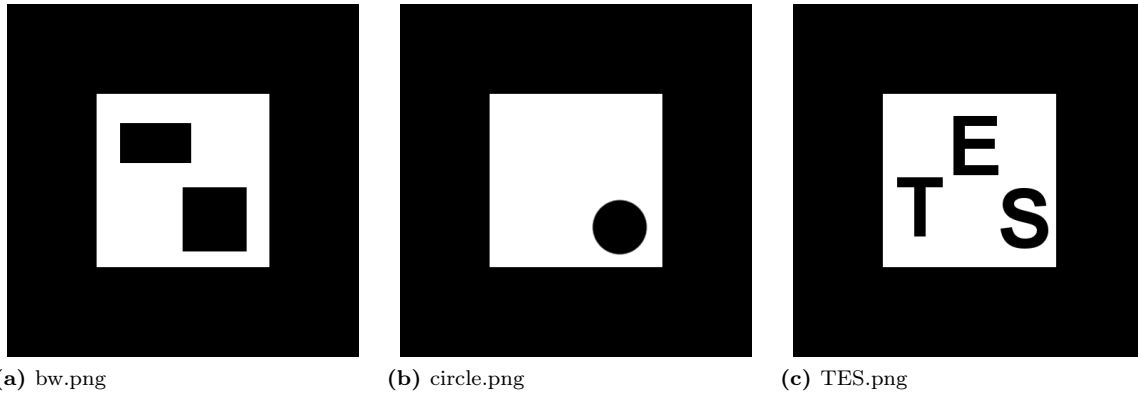


Abbildung 6.17.: Quadratische Test-Patterns

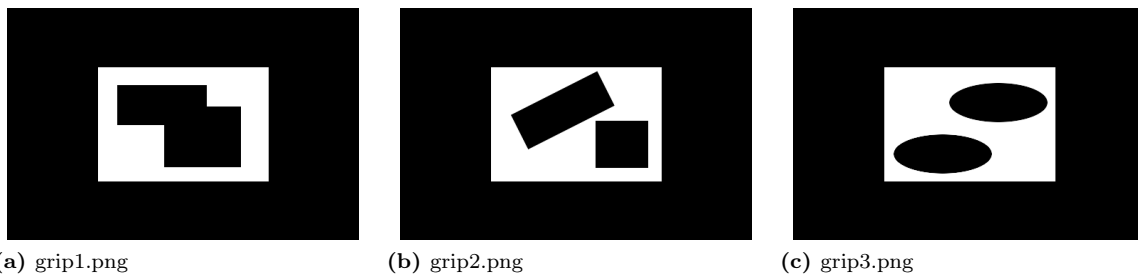


Abbildung 6.18.: Rechteckige Test-Patterns

16×16px als *Marker Segments* und eine *Marker Size* von 50%. Alle Patterns, welche für eine gewisse Aufgabe verwendet werden sollen, werden in einer lokalen Datenbank gespeichert auf welche das ROS-Package zugreifen kann. Beim Vorgang des Pattern-Matching werden zuerst sämtliche Muster in einer Szene bzw. Kameraaufnahme gesucht, erkannt und danach mit den Mustern der Datenbank verglichen. Der Vertrauensbereich, jener Bereich in dem ein Muster als sicher wiedererkannt gilt, kann dabei mittels eines Parameters eingestellt werden. Zusätzlich zu den neu entworfenen Mustern wurden auch das V4R², das TU und weitere Patterns in die Datenbank geladen. Die ersten Tests lieferten sehr gute Ergebnisse und zeigten, dass sogar Patterns – je nach Gestalt – bis zu einer Größe von 40×40mm am Boden liegend, also in einer ungefähren Entfernung von mindestens 1.5m erkannt werden konnten. Dies verdeutlicht Figure 6.19.

Es konnte gezeigt werden, dass das TU-Wien Logo (bzw. dessen Pattern) in der 40×40mm Variante am Boden liegend im Vergleich zum V4R Pattern erkannt wurde, weil die vergleichbar dünnen Linien der Schrift auf diese Entfernung von der Kamera nicht mehr aufgenommen werden konnten.

Ebenfalls ersichtlich ist, dass eine Rotation eines Patterns (beispielsweise des TU-Patterns auf dem Greifer selbst nach Figure 6.19) den Erkennungsalgorithmus nicht beeinträchtigt. Das Pattern-Matching ist somit rotationsinvariant und größeninvariant, soweit dies unabhängig von der Kameraqualität betrachtet wird. Die unterschiedlichen Ergebnisse des Matching-Algorithmus sind in Figure 6.19 ersichtlich. Wird ein Muster als

²Vision for Robotics [Eine Arbeitsgruppe des ACIN, die sich mit maschinellem Sehen und Ähnlichem beschäftigt] (V4R)

6. Implementierung

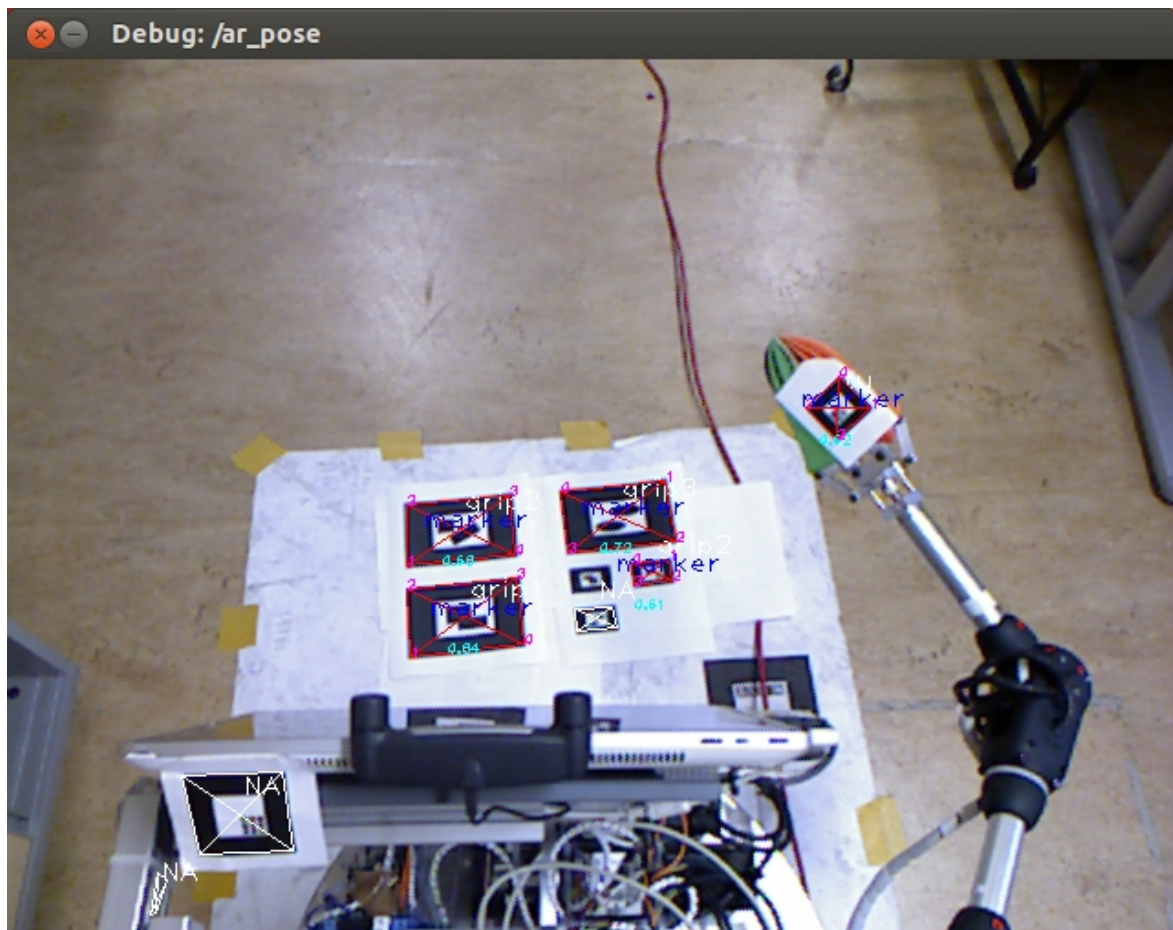


Abbildung 6.19.: Matching mehrerer Testpatterns am Boden und auf dem Greifer.

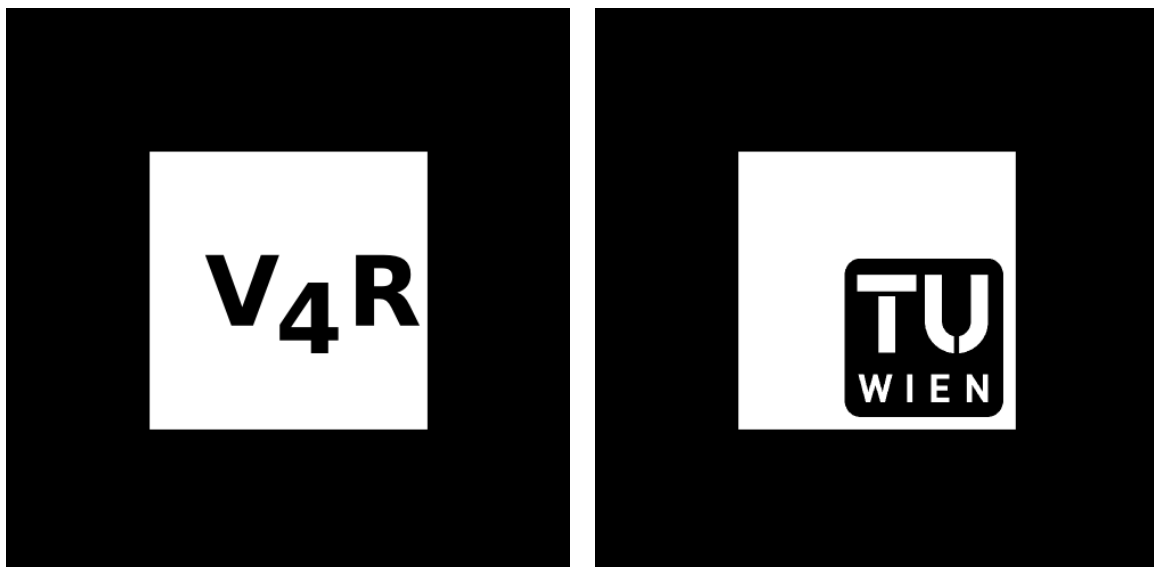


Abbildung 6.20.: V4R Pattern und TU Pattern

6. Implementierung

Pattern erkannt, wird es mit einem weißen Rahmen, einem Rechteck mit eingezeichneten Diagonalen, markiert. Falls das Muster durch Vergleich mit den, in der Datenbank gespeicherten, Offline-Mustern erkannt wurde, wird der Name des Patterns ebenfalls angezeigt. Anderenfalls wird *NA* angezeigt. Dies steht für „Not-Available“ (nicht in der Datenbank verfügbar). Je nachdem mit welcher Wahrscheinlichkeit das Pattern erkannt wurde, wird beim Überschreiten des Schwellwertes des Vertrauensbereiches ein roter Rahmen statt des weißen Rahmens über das Muster gelegt. Zusätzlich werden die vier Eckpunkte mit pinken Zahlen nummeriert und das Wort „marker“ in blau angezeigt. Diese Anzeige wurde im späteren Verlauf der Experimente angepasst.

Um die Greiferoberfläche möglichst gut mit einem Pattern auszufüllen, wurde der provisorische Überzug von der Greiferoberfläche entfernt und ein rechteckiges Pattern auf den Greifer montiert. Die Idee war es nun zu verifizieren, ob die Aufteilung des Musters Einfluss auf dessen Erkennung hat. Es wurde jeder Finger des Greifers mit einer Patternhälfte versehen, und kontrolliert, ob die Erkennung des Patterns trotzdem durchgeführt werden kann. Durch eine Öffnung des Greifers oder durch das Ergreifen eines Objektes sollte das Pattern nicht mehr erkannt werden. Dieser Fakt wurde als Ansatz für die Greifererkennung verwendet. Deswegen wurde das Pattern in der Hälfte geteilt und der Matching-Algorithmus gestartet. Tatsächlich wurde das Pattern bei geschlossenem Greifer erkannt. Um dies zu beweisen, wurden zwei Aufnahmen gemacht (das komplette und das zerschnittene Muster). Dies veranschaulicht Figure 6.21.

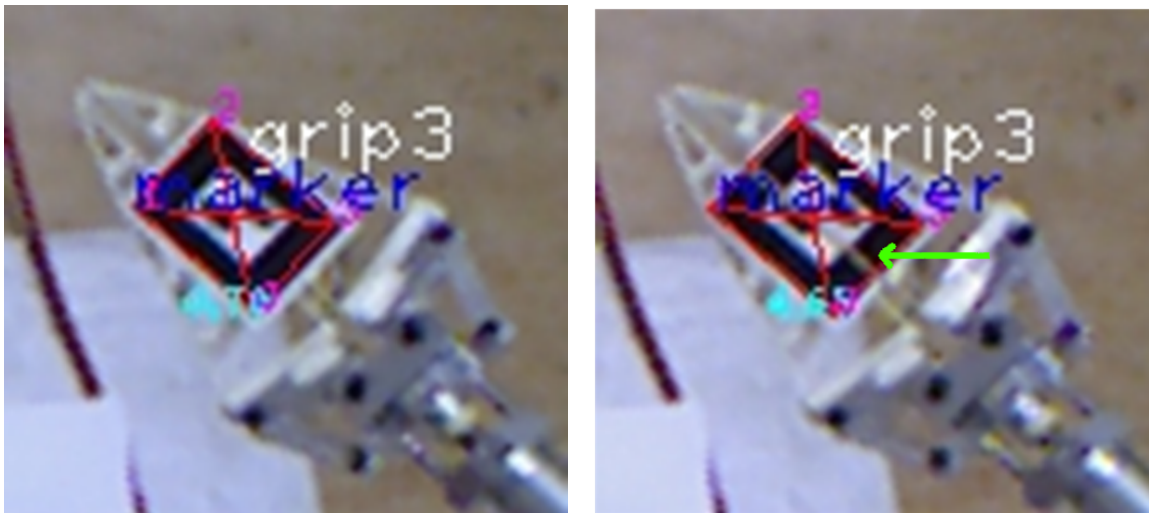


Abbildung 6.21.: Vergleich zwischen ganzem und zerschnittenem Pattern. Im rechten Bild ist ein kleiner Spalt zwischen den beiden Patternhälften erkennbar (grüner Pfeil). Trotzdem ist es möglich, das Pattern zu erkennen.

Abschließend wurde der Greifer in mehrere Positionen (von offen bis geschlossen) gebracht, um zu analysieren ob und wann das Muster auf dem Greifer erkannt wird. Nach den Erkenntnissen der ersten Testreihe wird es nur erkannt, wenn der Greifer komplett geschlossen ist (Figure 6.22).

Der Grundstein für die Greifererkennung wurde hiermit gelegt. Nun galt es herauszufinden in wie weit das Pattern Matching ausreichend ist, um zu erkennen, ob ein Objekt gegriffen wurde oder nicht. Die Grundidee ist ein Negativvergleich. Sobald der

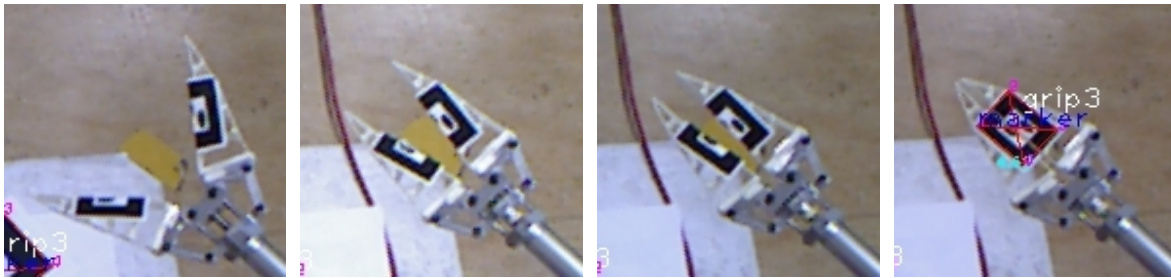


Abbildung 6.22.: Greifer mit halbiertem Pattern in mehreren (teil-)offenen und in geschlossener Position.

Greifer selbst nicht mehr erkannt wird bzw. dessen Pattern, kann davon ausgegangen werden, dass sich ein Objekt im Greifer befindet. Eine wichtige Voraussetzung für diesen Vergleich ist jedoch, dass der Greifer zuvor geschlossen wurde.

Dies führt zur nächsten Problematik. Der Greifer muss komplett geschlossen sein. Auf Grund von Hardware- und Steuerungstoleranzen kann es vorkommen, dass der Greifer trotz des Befehls „Greifer schließen“ nicht komplett geschlossen ist.

Eine weitere, aber zufällige, Erkenntnis konnte hinsichtlich der Umgebung des Patterns gewonnen werden. Der AR-Marker-Algorithmus beruht, wie in Kapitel 3.2 erklärt, auf der Suche nach Regionen, welche von vier Linien begrenzt sind. Befindet sich nun ein dunkles – annähernd schwarzes – Objekt direkt neben der Außenkante des schwarzen Randes des Patterns, werden diese vier Linien nicht mehr gefunden und das Muster wird nicht mehr als mögliches Pattern erkannt. Vor allem an den Eckpunkten des rechteckigen Patterns kann dies zu Fehlern führen, wie Figure 6.23 zeigt. Dieses Problem kann sehr einfach behoben werden, indem entweder ein schmaler weißer Rahmen an der Außenkante des Patterns hinzugefügt wird oder das Pattern ein wenig verkleinert wird. Damit kann der schwarze Rahmen des Patterns eindeutig von der Umgebung abgegrenzt werden und die Randlinien des Patterns können einwandfrei identifiziert werden. Wie auf den folgenden Abbildungen zu sehen ist, wird das Muster nicht erkannt, wenn ein ähnlich-farbenes Objekt (hier der dunkelblaue Verschluss einer Trinkflasche) direkt an den schwarzen Rahmen des Patterns, genauer gesagt an den linken oberen Eckpunkt, grenzt.

Zusätzlich muss die Möglichkeit bedacht werden, dass ein Objekt liegend gegriffen wird oder der Arm soweit bewegt wurde, dass das Oberflächen-Muster mit Hilfe der Kamera nicht aufgenommen werden kann. Um weiterhin unabhängig von den Motordaten des Armes zu sein, wurde auf der Seite des Greifers ein Pattern angebracht. Bei diesem musste allerdings beachtet werden, dass es sich um eine kleine Fläche handelt, die dem Muster zur Verfügung steht. Die ersten selbst erstellten Muster wurden, ohne direkt auf den theoretischen Hintergrund zu deren Design einzugehen, erstellt. Nachdem sich der Pattern-Ansatz positiv bewiesen hatte, wurden einige wichtige Eigenschaften für Patterns aus der Literatur gefunden. Zu diesem Zweck sei an das Kapitel 3.2. verwiesen.

Bei den neuen Patterns wurden diese Anforderungen berücksichtigt. Das Oberflächenpattern wurde so entworfen, dass der innere Bereich möglichst homogen ist. Hinzukommt, dass die Oberfläche des Greifers am besten weiß ist, um sich von den schwarzen Kanten des Musters möglichst gut abzuheben. Auch der Fakt, dass viele kleine

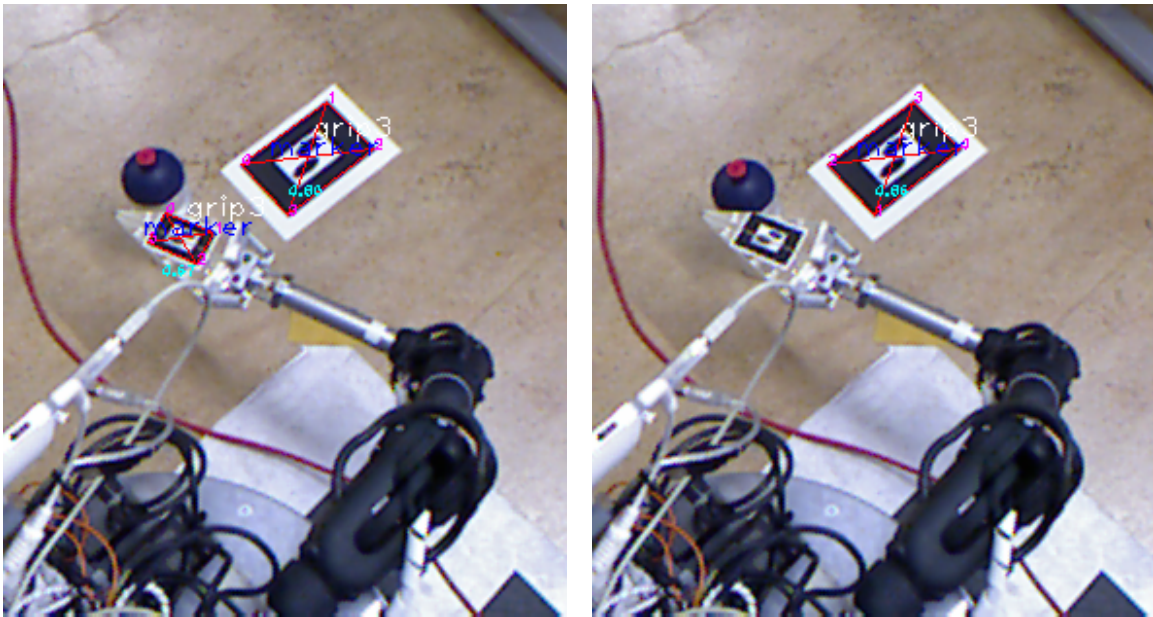


Abbildung 6.23.: Probleme durch ungenügende Abgrenzung des Patterns von der Umgebung. Der dunkelblaue Deckel der Trinkflasche wird im rechten Bild fälschlicherweise als Teil es Rahmens empfunden. Obwohl die Irritation nur an der linken oberen Ecke des Patterns auftritt, ist eine Erkennung der Außenkante nicht mehr möglich. Dadurch wird das Pattern nicht mehr gefunden. Im linken Bild besteht genug Abstand zwischen der Trinkflasche und den Außenkanten des top-Patterns.

Objekte und Gegenstände im Alltagsleben und in der Betreuung älterer Menschen kaum weiß sind, unterstützt diese Wahl. Aus diesem Grund wurde eine weiße Innenfläche gewählt. Dies sollte auch das Problem des nicht komplett schließenden Greifers relativieren bzw. verbessern. Da die Innenseite der Greiferfinger ebenfalls weiß ist, sollte der Spalt nicht sichtbar sein, wenn eine, von der direkten Draufsicht unterschiedliche, Perspektive gewählt wird. Sobald der Greifer nicht komplett geschlossen werden kann, weil sich ein Objekt darin befindet, wird das Muster nicht mehr erkannt und der Greifversuch kann als erfolgreich angesehen werden. Wird das Oberflächenpattern erkannt, kann davon ausgegangen werden, dass sich kein Objekt im Greifer befindet.

Der Entwurf des Oberflächenpattern wurde so gewählt, dass es einerseits nicht symmetrisch ist, jedoch eine einfache Struktur mit möglichst hohem Weißanteil gegeben ist. Hinzukommt, dass sämtliche Kurven und runde Objekte vermieden wurden, da bei Verformung des Greifers ohnehin aus den geraden gebogene Linien entstehen. Da die unterschiedlichen Krümmungen beim Marker-Matching nur schlecht unterschieden werden können, vor allem wenn sich das Pattern in größerer Entfernung von der Kamera befindet, wurden im Grundzustand gerade Linien gewählt, um möglichst klare Kanten zu erhalten. Dies sollte die Erkennung des Patterns verbessern. Für das seitliche Pattern wurde ein rotationsinvariantes, symmetrisches Pattern mit einer einfachen, klar erkennbaren Struktur verwendet. Diese Eigenschaften sind notwendig, da dieses Pattern relativ klein ist und es dadurch für den Algorithmus schwierig sein kann, es in größerer Entfernung zu erkennen. Außerdem wurde darauf geachtet, die Patterns

6. Implementierung

möglichst verschieden zu entwerfen, um einen niedrigen Korrelationswert zu erhalten. Beide Muster sind in der folgenden Abbildung (Figure 6.24) dargestellt.



Abbildung 6.24.: Greiferpatterns. Im linken Bild ist das top-Pattern (auf der Oberfläche) und im rechten das side-Pattern (an der Seite des Greifers) dargestellt.

Durch die später entworfene Latexoberfläche ist es möglich ein Pattern direkt auf die Greiferoberfläche aufzutragen. Dies wurde in erster Linie per Handzeichnung (mit Schablone) durchgeführt, soll aber in Zukunft mittels Sprühtechnik (Airbrush, etc.) aufgetragen werden. Zusammenfassend sei gesagt, dass der AR-Marker-Algorithmus sich wegen folgender Eigenschaften gegenüber den anderen Ansätzen durchgesetzt hat:

- ✓ Portabilität für planare Greifer
- ✓ Sehr gute Effizienz trotz geringer Kameraauflösung
- ✓ Eigenständigkeit (es ist kein weiterer Algorithmus zur Greifererkennung notwendig)
- ✓ Robustheit und Zuverlässigkeit
- ✓ Hardwaremodifikation, welche auch für andere Problemstellungen von Vorteil ist
- ✓ Beste Lösungsvariante in Anbetracht von kleinen Objekten
- ✓ Einfache Implementierung

Nachdem sich der AR-Marker-Algorithmus als Lösungsansatz des gestellten Problems herauskristallisiert hatte und sämtliche Hardwaremodifikationen vorgenommen wurden, konnte mit der Entwicklung der Software begonnen werden.

7.1. Vorbereitung und Überblick

Zur Entwicklung der Software ist die Steuerung des Roboterarmes notwendig. Bevor jedoch mit der Armsteuerung begonnen wird, muss die Referenzierungsfahrt des Armes überarbeitet werden. Die Referenzierungsfahrt ist notwendig, um die Positionen der Motoren der einzelnen Achsen zu initialisieren. Damit ist eine korrekte Positionierung überhaupt erst möglich. Wird keine Referenzierung vorgenommen, kann nicht festgestellt werden an welcher Position sich die einzelnen Achsen befinden. Wird die Steuerungseinheit des Armes abgeschaltet, geht die Positionsinformation verloren. Bei der Referenzierungsfahrt wird jede Achse solange bewegt, bis eine bestimmte Position erreicht wird. Dies kann durch ein Ende des Bewegungsraumes oder durch einen (End-)Schalter signalisiert werden.

Zu Beginn der Arbeit mit dem Hobbitarm verlief der Referenzierungsvorgang selten erfolgreich. Meistens hatte dies seinen Ursprung in der Nichterfüllung der Referenzierungsbedingung einer Achse. Dies ist beispielsweise das Erreichen einer bestimmten Achsposition, welche durch einen Endschalter angezeigt wird. Zur Konfiguration der Referenzfahrt wurde das Programm *mcTools* verwendet. Dies dient der Einstellung sämtlicher Parameter jeder einzelnen Achse. Dazu zählen beispielsweise die Kennwerte des Reglers, die Bewegungsart beim Referenzieren, die maximalen Stromwerte und die Positionslimits. Nach einigen Tagen Arbeit und der Unterstützung des Verantwortlichen für den Hobbit-Arm konnte die Zuverlässigkeit stark gesteigert werden. Bei den Verbesserungen spielte außerdem die Startposition, aus welcher der Referenzierungsalgorithmus ausgeführt wurde, eine zentrale Rolle.

Da das Referenzierungsproblem gelöst war, konnte mit der Programmierung des ROS

7. Experimente und Ergebnisse

Paketes begonnen werden. Nachdem die Tutorials auf der ROS Homepage ausgearbeitet wurden, galt es sich für eine der beiden Programmiersprachen, C++ oder Python, zu entscheiden. Die Wahl fiel auf Python. Zu Beginn wurde ein Testprogramm erstellt, der komplette Algorithmus inklusive Zustandsmaschine wird zu einem späteren Zeitpunkt erklärt.

Es wurde ein ROS Packet (ROS-Distribution *Fuerte*) erstellt, welches anhand des AR-Marker-Algorithmus (siehe Kapitel 3.2) die Patterns erkennen kann und als Ausgabe die Position(en) und Namen der erkannten Patterns ausgibt. Es wurden somit nicht nur die Rücktransformationen der Markerpositionen in Hinblick auf die Kamera ausgegeben, sondern auch die Namen der gefundenen Marker. Dieser sogenannte Publisher¹-Knoten ermöglicht allen weiteren ROS-Knoten, welche sich als sogenannter Subscriber² anmelden, die Weiterverwendung der Information. Auf Grund dieser Daten, des Namens und der Position, wurde ermittelt, ob es sich um das Pattern auf der Oberfläche³ oder um das Pattern auf der Seite⁴ des Greifers handelt. Dem entsprechend werden die Patterns im Weiteren als top- und side-Pattern benannt.

Zur Kontrolle wurde ein weiterer ROS-Knoten erstellt, der die Ausgabe des Paketes in einem Terminal anzeigt. Abbildung 7.1 zeigt diesen Sachverhalt. Getestet wurden die beiden ROS-Programme zuerst mit einer externen Kamera, um einerseits variabler mit der Kameraposition zu sein und andererseits, um eine komfortablere Programmierung – allein schon wegen der Computerleistung – zu ermöglichen. Dies ist ein großer Vorteil von ROS. Pakete können auf verschiedenen Systemen erstellt bzw. programmiert werden und ohne Umstände auf andere Systeme portiert werden.

Auf der linken Seite von Abbildung 7.1 sind die empfangenen Namen der gefundenen Patterns chronologisch aufgelistet. Diese werden nach Veröffentlichung durch den ROS-Knoten *ar-marker* mittels Knoten *checkSub* herausgefiltert und in einem Terminal angezeigt. Dies dient in erster Linie Debugging-Zwecken.

Um einer Verwechslung der Marker zu entgehen, wurden nur die, für diese Aufgabe notwendigen Marker, verwendet. Alle zuvor getesteten Patterns wurden aus der Datenbank entfernt. Dieses einfache ROS-Packet wurde verwendet, um einige Schwierigkeiten der Patternerkennung herauszuarbeiten ohne sich mit dem Zustandsablauf des kompletten Algorithmus zu beschäftigen. Das einfache Erkennungsprogramm wurde sowohl auf einem externen PC, wie auch am Hobbit direkt, getestet. Dies sollte den Fokus auf die Probleme der Erkennung selbst, das Kernstück des Programmes, und nicht auf den Ablauf lenken. Sämtliche Bewegungen des Greifers und Veränderungen in der Kameraansicht wurden manuell durchgeführt. Die einzelnen Achsbewegungen und das Öffnen und Schließen des Greifers wurden meist über das Tablet des Hobbit mit Hilfe des Programms *mcTools* gesteuert. Um konkrete Positionen des Armes einzunehmen, wurde eine synchronisierte Bewegung aller Achsen mittels vorgefertigtem Python-Script ausgeführt. Nachdem die prinzipielle Funktionsweise der Greifererkennung implementiert und die Steuerung des Armes hergestellt wurde, konnte mit den Experimenten begonnen werden.

¹Veröffentlicher

²Zuhörer

³top-Pattern

⁴side-Pattern

7. Experimente und Ergebnisse

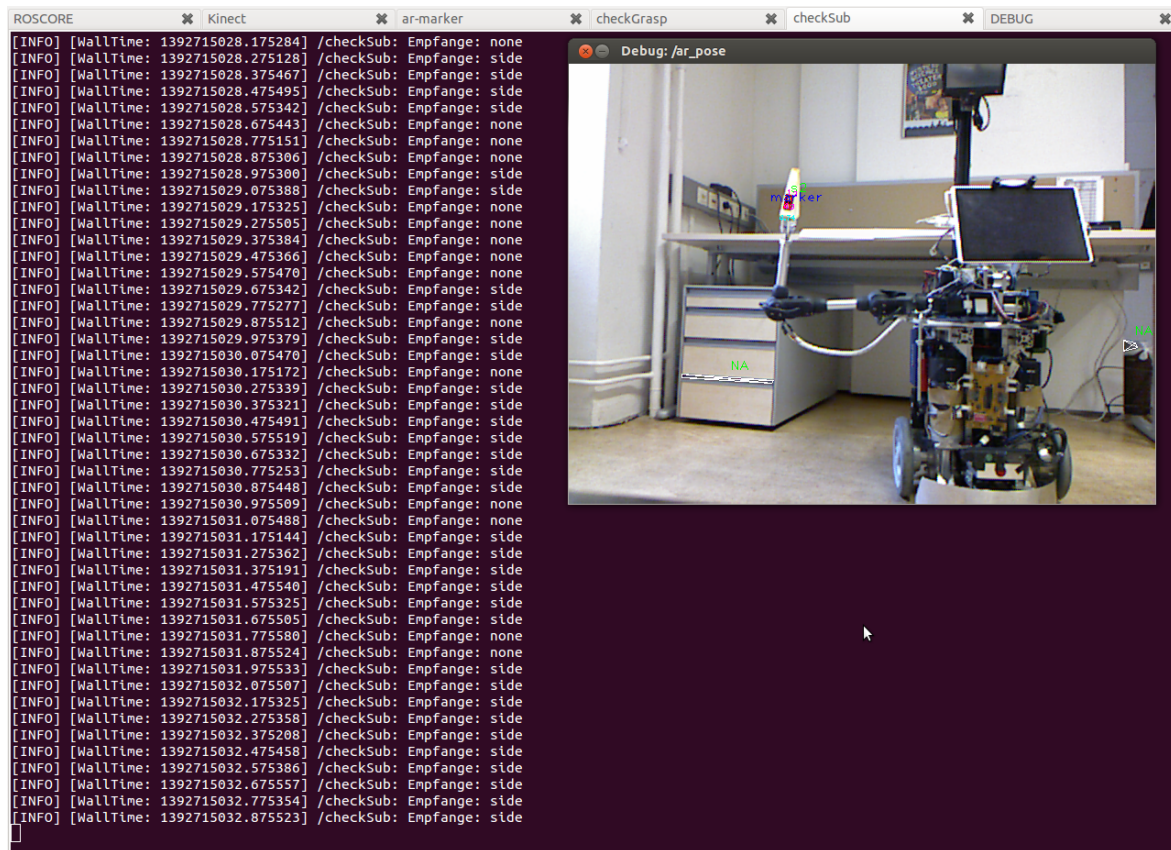


Abbildung 7.1.: AR-Marker Aufnahme und Kontrollausgabe. Die Reiter des Terminals zeigen alle gestarteten ROS-Pakete an. Neben dem Hauptknoten *ROSCORE* sind die Knoten der Kinect-Kamera, der AR-Marker, des checkGrasp-Paketes und des checkSub Paketes gestartet. Letzteres dient zur Anzeige der gefundenen Patterns bzw. deren Ausgabe am Terminal.

7.2. Experimente des statischen Algorithmus

Wie in Abbildung 7.2 erkennbar ist, gibt es Probleme beim Erkennen des Oberflächen-patterns, wenn der Kamerablickwinkel ungefähr 90 Grad auf die Fläche des Greifers (Draufsicht) beträgt.

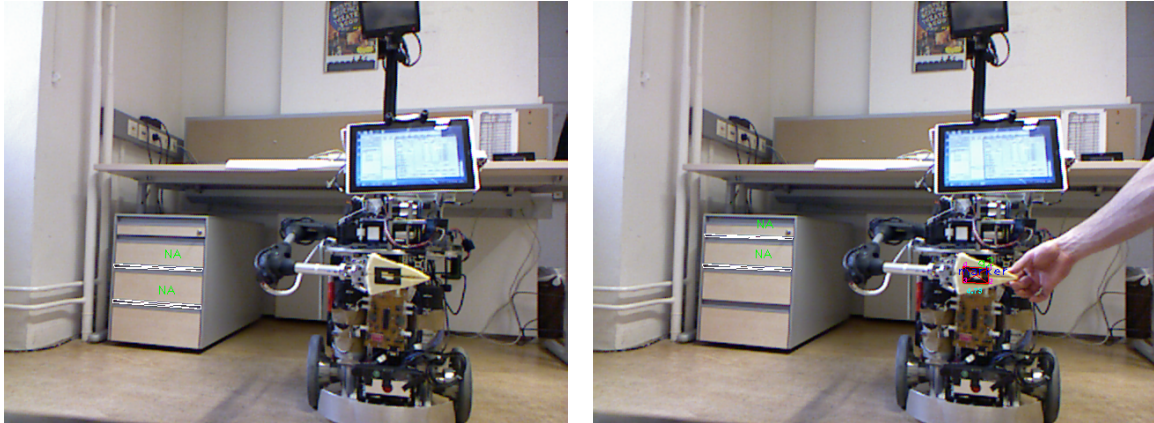


Abbildung 7.2.: Erkennungsprobleme des AR-Markers am Greifer. Im linken Bild wird der Greifer, auf Grund von mechanischen Problemen, nicht korrekt geschlossen und somit das Pattern nicht erkannt. Wird der Greifer manuell geschlossen (rechtes Bild) steht einer Erkennung nichts im Weg.

Der Grund dafür ist, dass der Greifer mechanisch nicht immer komplett schließt und sich somit ein Spalt zwischen den beiden Fingern des Greifers befindet. Dieses Hardwareproblem konnte bis zuletzt nicht komplett beseitigt werden. Um dies zu bewerkstelligen hätte ein Neudesign der Greifermechanik durchgeführt werden müssen. Es wurde jedoch, durch komplettes Zerlegen der Greifermechanik und Wartungsarbeiten, die Zuverlässigkeit des Schließens verbessert.

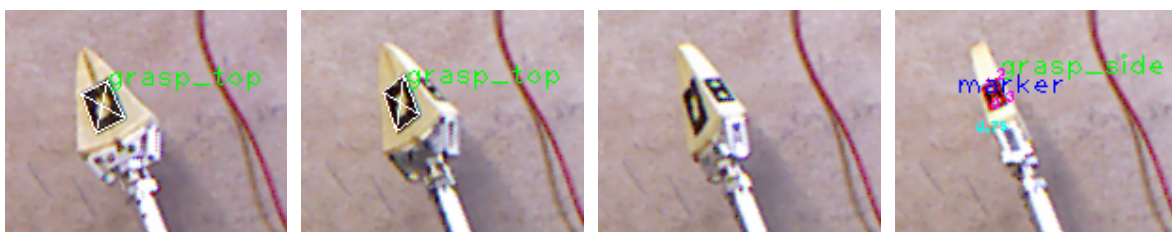


Abbildung 7.3.: Hobbitansicht beim bodennahen Greifen bei verschiedenen Rotationswinkeln (0°, 45°, 75°, 90°) der Achse 5.

Aus Sicht des Hobbit wurden verschiedene potenzielle Greifpositionen des Armes eingenommen. In Abbildung 7.3 sind die verschiedenen Ansichten bei einer Rotation um die Achse 5 des Greifers abgebildet. Dabei ist ersichtlich, dass das Muster auf der Greiferoberfläche beinahe bis zu einer Rotation von ca. 75° erkannt werden konnte. Der erste direkte Test mit verschiedenen Objekten (hier einem Handy und einem Whiteboard-Stift) fand in einem geschlossenen Raum bei konstanter Deckenbeleuchtung mittels Leuchtstoffröhren statt. Das natürliche Tageslicht wurde hierfür weitgehend

unterbunden. Dies war notwendig, um eine konstante Beleuchtungsszene zu erhalten. Wie anhand von Abbildung 7.4 nachvollzogen werden kann, verliefen die ersten Tests erfolgreich.

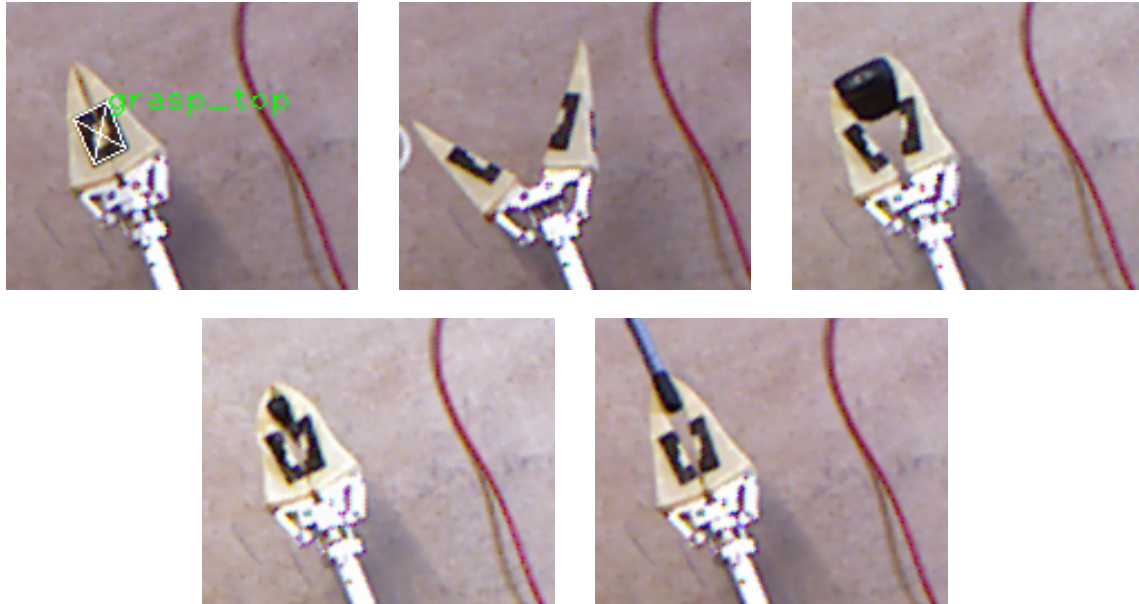


Abbildung 7.4.: Erster Test mit Objekten. Der geschlossene Greifer (mit und ohne Gegenstand) wurde mit dem geöffneten Greifer verglichen. Der Algorithmus liefert für alle Objekte (Handy, White-Board-Marker) ein korrektes Ergebnis.

Bei allen verwendeten Objekten wurde der Greifer durch das darin befindliche Objekt soweit verändert, entweder durch Verformung oder durch Freibleiben eines Spaltes zwischen den beiden Greiferteilen, dass das Pattern nicht mehr erkannt bzw. gefunden werden konnte.

7.2.1. Umgebungseinflüsse

Bei den weiteren Tests wurde festgestellt, dass die Mustererkennung – wie vermutet – beleuchtungsabhängig ist. Falls das Pattern zu stark reflektiert (dies hängt natürlich auch mit dem Glanz der verwendeten Farbe und Oberfläche zusammen), wird es nicht richtig erkannt. Wird ein künstlicher Schatten erzeugt (in Abbildung 7.5 wurde dies nur durch Vorhalten einer Hand bewerkstelligt), kann das Muster wieder problemlos erkannt werden.

Vor allem bei einer direkten Draufsicht auf das seitliche Pattern ist die Beleuchtungsabhängigkeit gut erkennbar. Es zeigt sich, dass die direkte Reflexion des Patterns (Glänzen) ein großes Problem für die Erkennung darstellt (siehe Abbildung 7.6). Dies ist darin begründet, dass durch die Reflexion an der schwarzen Farbe des Patterns für die Kamera ein heller Fleck erkennbar ist und somit die schwarze Fläche oder Kante des Patterns unterbrochen ist. Das bedeutet, dass nicht die Identifikation des Patterns, also der Vergleich mit den Patterns in der Datenbank und deren Korrelation, sondern meist schon das Finden des Patterns in der Szene das Problem ist, je nachdem wo die Spiegelung auftritt. Wird das Licht (Leuchtstoffröhren) abgeschaltet, so

7. Experimente und Ergebnisse



Abbildung 7.5.: Probleme der Mustererkennung durch Spiegelung am Pattern. Im linken Bild ist die Spiegelung in Form eines hellen Flecks erkennbar. Dies führt zu Problemen bei der Erkennung des Patterns. Wird die direkte Beleuchtung mit der Hand abgeschottet (rechtes Bild), kann das Pattern einwandfrei erkannt werden.

wird das Pattern ebenfalls kaum erkannt. Dies hat seinen Ursprung in der Tatsache, dass der Helligkeitsunterschied zwischen schwarzer und weißer Patternfläche zu gering ist. Erst wenn die direkte Beleuchtung des Patterns abgeschottet wird bzw. besser gesagt eine Spiegelung unterbunden wird, wird das Pattern einwandfrei erkannt (siehe Abbildung 7.6). Dies könnte jedoch ebenfalls durch Verwendung einer matten, nicht glänzenden, Farbe des Patterns bewerkstelligt werden. Allgemein sei betont, dass keine Tests mit verschiedenen Beleuchtungsmitteln durchgeführt wurden, da diese Thematik ein allgemeines Problem ist, welches nicht nur bei der Greifererkennung, sondern bei jedem Hobbit-Algorithmus, unter Verwendung der Kamera auftritt. Es sollte lediglich festgehalten werden, dass, wie bei fast allen visuellen Systemen, die Beleuchtung eine wichtige Rolle spielt, die nicht außer Acht gelassen werden darf.



Abbildung 7.6.: Beleuchtungsprobleme: In allen Szenen wurde das direkte Tageslicht abgeschottet. Links ist die reale Szene mit direkter künstlicher Beleuchtung (Leuchtstoffröhre) dargestellt. Im zweiten Bild von links wurde die künstliche Beleuchtung abgeschaltet. In der dritten und vierten Szenen wurde nur die direkte Beleuchtung durch ein Blatt bzw. ein dunkles Tuch unterbunden. Dies verhindert Spiegelungen an den Patterns und führt zur einwandfreien Identifikation (letztes Bild ganz rechts).

7. Experimente und Ergebnisse

Die Beleuchtungsabhängigkeit, welche hier dargestellt wurde, bezieht sich selbstverständlich nur auf die 2D-Daten der Kamera. Wird allerdings Tageslicht, vor allem direktes Sonnenlicht als Beleuchtungsszenario verwendet, kann daraus eine starke Beeinflussung der Farbinformation und der Tiefeninformation resultieren. Zusätzlich zu der Problematik für die Daten des CCD-Sensors kommt der Fakt hinzu, dass Sonnenlicht einen hohen Infrarotstrahlungsanteil besitzt, welcher die Projektionen der Infrarotquelle der Kinect-Kamera stört.

Ein großes Problem der Mustererkennung ist außerdem die Wahl des Blickwinkels zum Objekt. Beispielsweise kommt es, wie schon zuvor erwähnt, zu Fehlern bei der Erkennung eines Patterns am geschlossenen Greifer ohne Objekt, wenn die Blickrichtung der Kamera orthogonal zur Oberseite des Greifers ist. Dabei kann ein Spalt erkannt werden, welcher hauptsächlich mechanischen Ursprungs ist. Ist der Blickwinkel der Kamera jedoch sehr schräg zur Oberfläche des Greifers, kann selbst bei einem deutlichen Spalt zwischen den Greiferfingern, welcher durch ein Objekt verursacht wurde, das Pattern fälschlicherweise erkannt werden. Dieser Sachverhalt soll im Weiteren behandelt werden.

Es wurde schon erwähnt, dass sehr dünne Objekte, wie beispielsweise ein USB⁵-Stick, nicht aus jeder Kameraposition als Objekt im Greifer erkannt werden können. Anhand der vier Bilder in Abbildung 7.7 sind vier verschiedene Ergebnisse der Mustererkennung ersichtlich. Interessant hierbei ist, dass der Greifer nicht bewegt wurde, sondern der Algorithmus unterschiedliche Ergebnisse liefert. In allen Bildern wird das side-Pattern genau erkannt. Dies ist ersichtlich durch den roten Rahmen und die Benennung des Patterns in grüner Schrift (früher wurde der Name des Patterns in weißer Schrift dargestellt).

⁵Universal Serial Bus [Universeller serieller Bus zur Kommunikation] (USB)

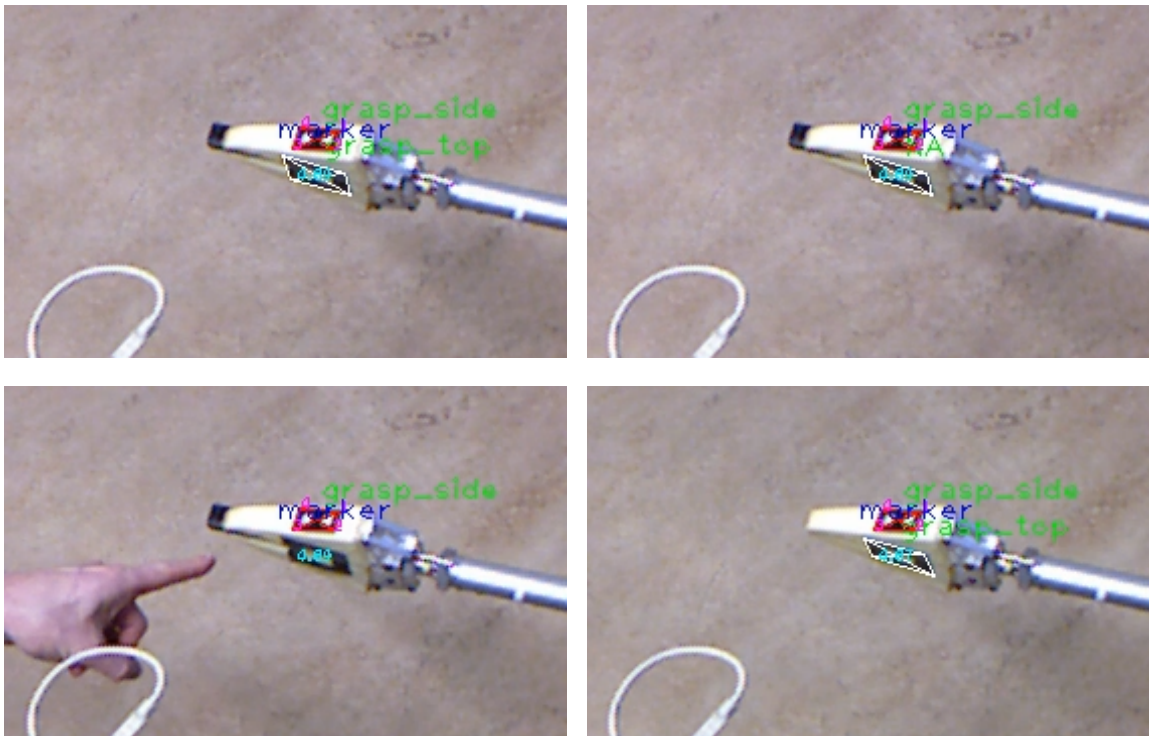


Abbildung 7.7.: Schrägsicht USB-Stick: Bei einem Vergleich zwischen den Schrägansichten des Greifers inklusive darin befindlichem USB-Stick fällt auf, dass das top-Pattern fehlerhafter Weise erkannt wird (erstes Bild oben links) oder zumindest ein Pattern an der Stelle des top-Patterns erkannt wird (zweites Bild oben rechts). Die dritte Teilabbildung (unten links) zeigt den korrekten Sachverhalt. Das Pattern wird nicht gefunden, da sich der USB-Stick im Greifer befindet. Als Referenz dient der vollständig geschlossene Greifer (unten rechts).

Im ersten Bild links oben ist zu erkennen, dass das top-Pattern mit dem USB-Stick erkannt wird (allerdings nur mit geringerer Sicherheit, was durch den weißen anstatt eines roten Rahmens ersichtlich ist). Im Bild rechts oben hingegen wird das top-Pattern überhaupt nicht gefunden, also nicht als Pattern erkannt. Durch Vergleich mit dem linken unteren Bild ist ersichtlich, dass zwar ein Pattern auf der Oberseite des Greifers gefunden wurde, es jedoch nicht identifiziert werden kann. Dies ist erkennbar durch den weißen Rahmen um das Pattern und die Benennung mit *NA* in grün. Um die Ansicht des Greifers mit gefasstem USB-Stick zu vergleichen, wurde der USB-Stick entfernt und der Greifer geschlossen. Im rechten unteren Bild wird das top-Pattern erkannt, wenn auch nur mit geringerer Sicherheit.

Es sei hier noch einmal erwähnt, dass die Schwelle der vollständigen Erkennung eines Patterns, dies wird durch den roten Rahmen angezeigt, durch einen Parameter im AR-Marker-Algorithmus verändert und optimiert werden kann. Wie schon zu Beginn gesagt sind die vorbereitenden Arbeitsschritte vor der Greifererkennung ausschlaggebend für den Erkennungsprozess. Dazu gehören die Themen wie sich der Hobbit in Bezug auf seine Umgebung und das zu ergreifende Objekt positioniert, welche anderen Objekte sich in der Umgebung befinden, ob das zu ergreifende Objekt richtig erkannt wird und welche Greifposition anhand der errechneten Angreifpunkte des Objektes eingenommen wird. Besonderes Augenmerk soll auf die letzte Thematik gelegt werden. Für den späteren Ablauf des Erkennungsprozesses wird diese Problemstellung ausschlaggebend und richtungsweisend sein.

7.2.2. Objekteinflüsse

Um die Thematik der Greifposition etwas näher zu beleuchten, wurden folgende Fragestellungen aufgeworfen:

- Wo steht das Objekt? (Am Boden, auf einem Tisch, auf einem Sessel, in einem Regal, etc.)
- In welcher Lage befindet sich das Objekt? (liegend, stehend)
- Wie wird das Objekt angegriffen? (Von der Seite, von oben, etc.)
- Können die Patterns nach der Positionierung des Greifers gesehen werden?
- Welche Unterschiede gibt es bei den verschiedenen Greiferpositionen und wie muss darauf reagiert werden?

Bevor diese Fragen anhand des kompletten Greifzyklus beantwortet werden, soll ein kleines Beispiel erläutern welche Probleme behandelt werden müssen. Dafür wurden zwei Greifsznarien simuliert. Beim ersten galt es am Boden stehende Objekte zu ergreifen. Die zweite Greifposition wurde so gewählt, dass ein, auf einem Tisch stehendes, Objekt ergriffen wurde. Bei diesen Szenarios wurde vorausgesetzt, dass ein Gegenstand von der Seite ergriffen wird und der Arm selbst nicht aus seiner Greifposition gebracht wird. Dies impliziert die freie Sicht auf das Objekt und vor allem auf den Greifer selbst. Es dürfen keine Hindernisse für die Abdeckung des Patterns sorgen.

7. Experimente und Ergebnisse

Nimmt der Greifer eine Position für bodennahes Greifen stehender Objekte ein, kann selbst ein Bleistift erkannt werden (siehe Abbildung 7.8). Genauer gesagt, wird das Pattern nicht mehr gefunden sobald sich der Bleistift im Greifer befindet. Wichtig zu erwähnen ist, dass in dieser Szene der Bleistift von der Spitze des Greifers ergriffen wurde und der Bleistift eine deutlich unterschiedliche Farbe – in diesem Fall grün – als der Greifer hat. Obwohl dieses Testszenario in der Realität eher selten bis garnicht auftreten wird, da ein Bleistift meistens in liegender Position vorgefunden wird, soll es zur Veranschaulichung der Effizienz des Algorithmus präsentiert werden. Hinzuzufügen ist, dass die Erkennung des Bleistifts in mehreren Fällen – abhängig von dessen Lage und Angriffspunkt – nicht funktioniert hat und es zu einer falschen Schlussfolgerung des Algorithmus kam. Wie sich später herausstellen wird, spielt die Position, an welcher sich der Gegenstand im Greifer befindet, bei kleinen Objekten, eine große Rolle. Zum Vergleich wurde eine Szene mit dem geschlossenen Greifer aufgenommen. Wie vermutet stellen größere Objekte, wie beispielsweise eine stehende Aluminiumdose, kein Problem für den Algorithmus dar.



Abbildung 7.8.: Greifererkennung eines Bleistifts in der unteren Greifposition (Bodennähe).

In einer erhöhten Greifposition, beispielsweise von einem kleinen Tisch oder einem Sessel, kann einerseits das Pattern weitaus zuverlässiger erkannt werden, als in der bodennahen Greifposition und andererseits können kleine Objekte, wie ein hochkant aufgestellter USB-Stick, problemlos erkannt werden. Dies veranschaulicht Abbildung 7.9.



Abbildung 7.9.: Greifererkennung eines USB-Sticks in der oberen Greifposition (Tischnähe).

7. Experimente und Ergebnisse

Wie schon bei der Behandlung der bodennahen Greifposition kurz angeschnitten, stellen sich bei der Erkennung eines stehenden Bleistiftes, wenngleich dieses Greifproblem selten vorkommen wird, einige Probleme in den Weg. Ein großer Nachteil ist die, schon öfters angesprochene, Qualität der Kamera. Durch die verhältnismäßig geringe Auflösung im Vergleich mit aktuellen Digitalkameras werden bald die Grenzen der Möglichkeiten aufgezeigt. Ein Leitspruch dazu ist: „Es können nur jene Strukturen erkannt werden, die auch visuell erfasst werden können?“. Auf Grund der schmalen Struktur des Bleistiftes kommt es nur zu einer kleinen Verformung der Greiferfinger. Diese Verformung und die, vom Greifer unterschiedliche, Farbe des Bleistifts erscheinen in dem Kamerabild lediglich als wenige Pixel. Dadurch kann es zu fehlerhaften Ergebnissen des Algorithmus kommen. Es zeigt sich, dass der Erfolg der Greifererkennung eines Bleistiftes erheblich von der Lage des Bleistiftes innerhalb des Greifers abhängt. Dies verdeutlicht, dass es eine große Rolle spielt wo ein Objekt ergriffen wird. Abbildung 7.10 zeigt diesen Sachverhalt und vergleicht diesen mit einem liegenden Bleistift.

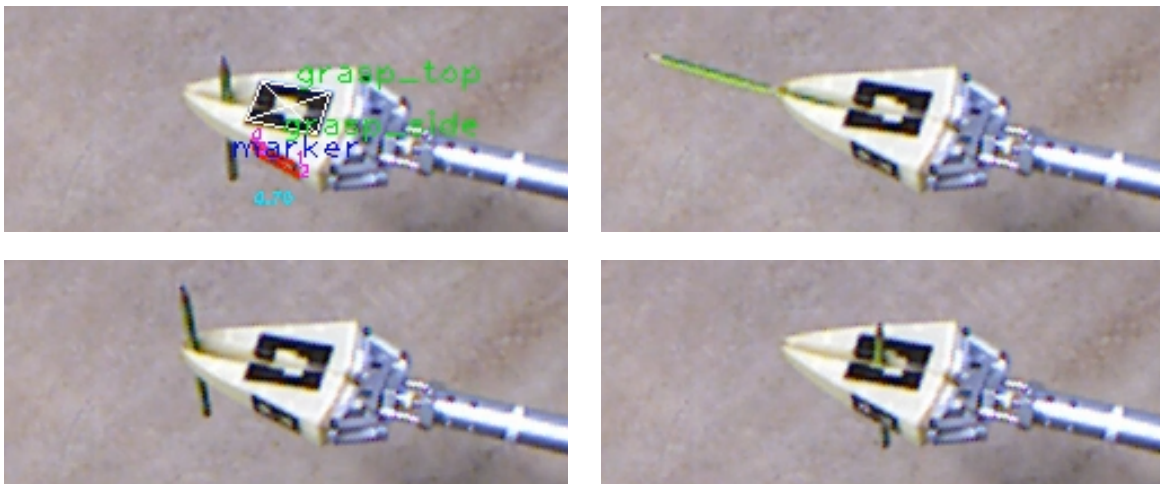


Abbildung 7.10.: Greifererkennung eines Bleistifts in der oberen Greifposition (Tischhöhe). Hier ist erkennbar, dass es bei der Erkennung zu Fehlern kommen kann (links oben). Je nachdem wo sich der Bleistift im Greifer befindet, kommt es zu unterschiedlichen Ergebnissen. Befindet sich der Angriffspunkt möglichst in der Mitte (rechts unten) des Greifers, kann das beste Ergebnis erzielt werden, solange der Bleistift nicht weiß ist und deutlich aus der Greiferoberfläche herausragt.

Anhand von Abbildung 7.10, ist eindeutig zu erkennen, dass sich der stehende Bleistift im Idealfall in der Mitte des Patterns befindet. Befindet er sich im vorderen Teil des Greifers, kann es zu Fehlern kommen. In diesem Fall (linkes oberes Bild) wurde sowohl das top- wie auch das side-Pattern erkannt. Der liegende Bleistift ergab weniger Probleme. Nicht bedacht wurde allerdings, dass ein liegender Bleistift nicht nahe der Oberfläche des top-Patterns ergriffen wird, sondern meistens im Bereich der Unterkante.

Eine Lösungsmöglichkeit für die, im letzten Abschnitt angesprochenen, Probleme ist, dass der Greifer nach dem Schließen um „den Unterarm“ des Hobbys, die Achse 5, gedreht wird und währenddessen das Pattern-Matching durchgeführt wird. Falls kein Pattern zu anfangs erkannt wird, kann selbst eine größere Rotation erfolgen, um eine genauere Aussage über den Greifstatus treffen zu können. Dies setzt voraus, dass das

7. Experimente und Ergebnisse

top-Pattern sowohl an der Ober- als auch an der Unterseite des Greifers angebracht wird. Des Weiteren könnte der Arm um eine weitere Achse, beispielsweise die Achse 1, die annähernd dem Schultergelenk entspricht, gedreht werden. Da die Anforderungen, möglichst ohne Bewegung des Armes auszukommen, erfüllt werden sollten, wurde dieser Gedanke zu anfangs nicht weiter verfolgt.

Wird die Umgebung des Menschen betrachtet, im Speziellen eine Wohnung, kann nicht von optimalen Bedingungen hinsichtlich der Sichtbarkeit von Objekten und der uneingeschränkten Zugänglichkeit zu jenen, ausgegangen werden. Dies bedeutet, es musste ein Kompromiss zwischen den gestellten Anforderungen und der Praxistauglichkeit erzielt werden. Neben der Problematik, den Unterschied zu erkennen, ob ein Objekt neben dem Greifer steht und diesen verdeckt oder ob es sich direkt im Greifer befindet, ist es wichtig zu wissen von welcher Position aus ein Objekt ergriffen werden soll. Am Boden befindliche Objekte werden im Fall des Hobbit meistens von oben ergriffen [12]. Weitere Angriffsversuche mit komplizierteren Greifern sind in [14] beschrieben.

Es kann durch alleinige Rotation um den Unterarm des Hobbits keine zuverlässige Analyse über den Greifstatus durchgeführt werden, weil der eigene Oberarm des Roboters den Greifer bzw. das Objekt verdecken kann. Somit liegt auf der Hand, dass alleine durch den statischen Pattern-Erkennungsansatz die Greiferkennung in vielen Szenarien nicht geklärt werden kann, da beispielsweise das top-Pattern oder der gesamte Greifer für die Kamera nicht sichtbar ist. Eine Bewegung des Roboterarms ist notwendig, um zuverlässige Ergebnisse zu erhalten. Aus diesem Grund wurde die Anforderung nach möglichst keiner Bewegung fallen gelassen. Dies liegt nicht an dem vorgestellten Lösungsansatz, sondern am Vorgang von welcher Richtung und Position ausgehend ein Objekt ergriffen werden soll. Sämtliche anderen Ansätze, die es auf rein visuellem Wege bewerkstelligen sollen über den Erfolg des Greifversuches zu entscheiden, könnten ebenso nicht agieren, wenn der Greifer und das Objekt selbst für die Kamera nicht sichtbar sind! Es wurde jedoch beachtet, dass möglichst geringe Bewegungen durchgeführt werden sollen.

Durch die Bewegung des Armes konnte außerdem sichergestellt werden, dass Spiegelungen der Beleuchtung im Pattern keinen entscheidenden Einfluss auf die Greiferkennung hat. Genauer gesagt sollten durch die Rotation des Unterarmes unterschiedliche Positionen zur Verfügung stehen in denen das Pattern-Matching durchgeführt werden kann. Diese Manipulation des Armes konnte des Weiteren dazu verwendet werden, um dem vorgestellten Problem der fälschlichen Erkennung eines Patterns auf Grund unterschiedlicher Perspektiven zu entgehen (siehe dazu das erste Bild von Abbildung 7.7). Dazu musste der Algorithmus jedoch einen Zähler für die gefundenen Patterns verwenden. Dieser wird im Programm ca. alle 70ms aktualisiert. Wird beispielsweise das top-Pattern zehn Mal erkannt, soll ein korrekter Match vorhanden sein. Dadurch kann erkannt werden, dass sich kein Objekt im Greifer befindet. Die Voraussetzung hierfür ist wiederum eine genau definierte Position des Manipulators in welcher möglichst wenige Fehlerkennungen stattfinden. Unterhalb des Schwellwerts für den Zähler gilt ein Muster als nicht erkannt und somit ein Objekt als ergriffen.

Um Informationen darüber zu erhalten, in welcher ungefähren Position sich der Greifer befindet, wurde beim Zustandsablauf des Algorithmus anfangs das seitliche Pattern berücksichtigt. Erneut könnte dies anhand der Daten der Armgelenke festgestellt werden. Wie jedoch schon weiter oben beschrieben, sollte die Abhängigkeit von

7. Experimente und Ergebnisse

(exakten) Achskoordinaten vermieden werden. Das side-Pattern kann sowohl im offenen wie auch im geschlossenen Zustand, unabhängig davon ob sich ein Objekt im Greifer befindet, erkannt werden, solange der Greifer nicht verdeckt ist. Die Rolle des seitlichen Patterns soll im nächsten Abschnitt diskutiert werden. Die beiden Muster auf dem Greifer sollten zur grundlegenden Bewegungssteuerung beitragen. Dabei ist der Fall zu betrachten, was passiert, wenn keines der beiden Pattern erkannt wurde.

Mehrere Ursachen sind hierfür vorstellbar:

- Ein oder mehrere Patterns sind von dem zu ergreifenden Objekt verdeckt
- Ein oder mehrere Patterns sind von der Umgebung oder anderen Gegenständen verdeckt
- Ein oder mehrere Patterns sind vom Arm des Roboters verdeckt
- Der Arm ist in einer Position in der das seitliche Pattern nicht sichtbar ist und es befindet sich ein Objekt im Greifer
- Die Ausrichtung der Kamera ist falsch
- Die Datenübertragung der Kamera, oder die Kamera selbst, ist fehlerhaft

Wird der erste Punkt betrachtet, liegt die Lösung jedenfalls in der Bewegung des kompletten Armes. Ist auch nach der Bewegung des Armes in eine bestimmte Position kein Pattern erkannt worden, kann getrost davon ausgegangen werden, dass sich ein Objekt im Greifer befindet, denn es hat sich mit dem Roboterarm mitbewegt. Ähnliches gilt für Punkt zwei der Aufzählung, solange der Greifer nicht von einem Tuch, einer Decke oder Ähnlichem abgedeckt ist. Dieser Fall sollte zwar kaum vorkommen, da dies durch die Festlegung der Angreifpunkte und andere vorbereitende Schritte gewährleistet ist, er soll jedoch trotzdem erwähnt werden.

Solange die Umgebung oder ein Gegenstand nicht beschädigt werden bzw. den Arm nicht beschädigen oder das Objekt aus dem Greifer „schlagen“, wird die Armbewegung in solcher Art ausgeführt, dass das top-Pattern jedenfalls sichtbar ist, außer es befindet sich tatsächlich ein Objekt im Greifer. Liegt eine unüberwindbare Barriere für den Arm vor, beispielsweise wurde der Arm unter einem Tisch positioniert und versucht sich, auf direktem Wege, zu einer Position über dem Tisch zu bewegen, wird der festgelegte Punkt vom Greifer nicht erreicht und es kommt zu einem Fehler der Greifersteuerung. Anhand dieser muss eine Fehlerbehandlung erfolgen. Diese ist jedoch nicht Thematik der Greifererkennung.

Die Punkte drei und vier der vorigen Liste werden ebenfalls durch die Bewegung des Manipulators behandelt. Einzig die letzten beiden Punkte werden hier nicht gelöst. Natürlich könnte eine Kamerabewegung oder eine Fehlerbehandlung der Kamera stattfinden. Diese ist jedoch ebenfalls nicht Thematik dieser Diplomarbeit. Da das zu ergreifende Objekt zuvor gefunden und somit gesehen wurde, kann von vereinfachten Bedingungen und von der korrekten Ausrichtung der Kamera ausgegangen werden.

7.3. Ablauf des dynamischen Algorithmus

Die Steuerung des gesamten Ablaufes soll anhand des Service-Call-Prinzip von ROS erfolgen. Während alle notwendigen Pakete ununterbrochen ausgeführt werden, wie der ROS-Hauptknoten, Kameraknoten und der AR-Marker-Knoten, soll der Steuerungsknoten, genannt *checkgrasp*, erst dann mit der Ausführung der Erkennung beginnen, wenn ein übergeordneter Prozess dies durch die einfache Mittelung „Start“ anordnet. Auf den Service-Call „Starte mit der Greifererkennung“ soll als Endergebnis ein „Objekt (nicht) ergriffen“ folgen. Der Erkennungsprozess soll somit erst durchgeführt werden, wenn dies explizit vom übergeordneten Programm gefordert ist. Bis zur Ausführung des Erkennungsalgorithmus verbleibt dieser in einer inaktiven Ruheposition, welche nur durch einen Startbefehl verlassen werden kann. Dies wurde insofern bewerkstelligt, als der Client beim Aufruf durch eine Eins, welche vom übergeordneten Prozess gesendet wird, gestartet wird und bei einer Null weiterhin im Ruhezustand arbeitet.

Nachdem die Greifererkennung durchgeführt wurde, verbleibt der Greiferkennungsalgorithmus im Ruhezustand. Das Startkommando sollte von dem Hauptprozess ausgehen, der die gesamte Hobbit-Steuerung bearbeitet. Nachdem die Greifpunkte errechnet wurden, der Hobbit in die Greifposition gebracht wurde und der Endeffektor zugegriffen hat, soll der Erkennungsprozess gestartet werden. Zu diesem Zweck wurde der einfache Knoten *startClient* erstellt, welcher rein für den Start des Erkennungsalgorithmus, *checkgrasp*, und für die Ausgabe des Endergebnisses zuständig ist. Die Form der Ausgabe war erneut Eins oder Null. Um dem Prinzip des Negativvergleiches treu zu bleiben signalisiert eine Null keine Erkennung des top-Patterns und somit, dass ein Objekt erkannt wurde. Eine Eins bedeutet, dass das top-Pattern erkannt wurde und sich somit kein Objekt im Greifer befinden kann.

7.3.1. Algorithmus mit relativer Bewegung

Der erste Entwurf des Ablaufes der Greifererkennung soll anhand des folgenden Diagrammes erklärt werden (Abbildung 7.11). Nachdem alle vorbereitenden Schritte für die Greifererkennung durchgeführt wurden, kann der Service-Call gestartet werden und es soll mit dem Pattern-Matching begonnen werden. Wird dabei das top-Pattern gleich zu Beginn beispielsweise zehn Mal erkannt (Überlauf des Patternzählers), kann darauf geschlossen werden, dass sich kein Objekt im Greifer befindet. Das Programm wäre beendet und die Rückmeldung des Service Calls fällt mit Eins positiv aus. Es wurde das top-Pattern gefunden. Somit ist kein Objekt im Greifer vorhanden. Sind beide Patterns vorhanden bzw. wurden diese erkannt, kann der gleiche Schluss gezogen werden. Das top-Pattern ist hier eindeutig das ausschlaggebende Pattern.

Wird nur das side-Pattern erkannt, kann davon ausgegangen werden, dass der Arm in einer Position ist, in der das top-Pattern nicht sichtbar ist oder sich ein Gegenstand im Greifer befindet. Nun bedarf es einer Rotation des Unterarm des Hobbit, also der Achse 5, damit auf jeden Fall die Möglichkeit gegeben ist, das top-Pattern zu sehen (in Abhängigkeit davon ob ein Objekt im Greifer ist oder nicht, ist dies natürlich nicht möglich). Das side-Pattern ist auf beiden Außenseiten, das top-Pattern auf der Ober- und Unterseite der Greiferfinger angebracht. Somit ist eine Rotation um 90 Grad in jedem Fall ausreichend um, zumindest während der Rotation, das top-Pattern in

7. Experimente und Ergebnisse

den Sichtbereich der Kamera zu bewegen. Während der Rotation wird die Erkennung selbst nicht gestoppt. Falls notwendig, wird lediglich der Zähler zurückgesetzt, um sporadische Fehlerkennungen zu vermeiden, welche bei einer größeren Rotation leicht vorkommen können. Nachdem die Rotation durchgeführt wurde (roter Pfad), wird der Zähler des top-Patterns zurückgesetzt und erneut nach Patterns gesucht. Wird in diesem Schritt das top-Pattern oft genug erkannt, oder werden beide Patterns erkannt, folgt gleiches wie oben gesagt. Wird erneut nur das seitliche Pattern erkannt, wird, um eine sichere Aussage treffen zu können, erneut eine Rotation der Achse 5 um 90 Grad vollzogen. Bringt auch dies keine Verbesserung im Sinne des Erkennens des top-Patterns, kann der Service-Call mit Null beantwortet werden. Um sicherzustellen, dass nur zwei Rotationszyklen der Achse 5 durchgeführt werden, werden Zähler (Rotationszähler A und B) für den Zustand nach der Rotation der Achse verwendet. Sobald der Zähler A auf zwei erhöht wurde und erneut nur das seitliche Pattern erkannt wurde, wird ein alternativer Weg, nämlich die Beendigung des Programmabschnittes, gewählt. Das top-Pattern wurde nicht gefunden und somit muss ein Objekt im Greifer vorhanden sein.

Zu guter Letzt wird der Fall behandelt, dass keines der beiden Patterns gefunden werden konnte. Ist dies der Fall, wird von einer verdeckten Greifposition, beispielsweise dem Greifen eines am Boden stehenden Objektes von oben, ausgegangen. Obwohl dies eine Verallgemeinerung ist und es andere Fälle gibt, in denen eine andere Greifposition eingenommen wurde, soll der Fall berücksichtigt werden. Ein Stichwort hierfür ist das Ergreifen eines stehenden Objektes von einem Tisch. Im ersten Schritt soll nur das Schultergelenk des Hobbits, die Achse 1 bewegt werden (rechtster Pfad). Auf Grund von theoretischen Überlegungen konnte die notwendige Bewegung auf ca. 30 Grad geschätzt werden. Es ist zu beachten, dass einerseits der Arm weit genug rotiert wird, um eine Erkennung zu ermöglichen und andererseits eine möglichst geringe Rotation durchzuführen, um einer Fehlerkennung vorzubeugen (Erklärung siehe Abbildung 7.12 und 7.14). Nach erfolgter Rotation (blauer Pfad) wird das Matching erneut gestartet. Dazu sei erwähnt, dass das Matching selbst ununterbrochen weiterarbeitet. Es werden, falls notwendig, lediglich die Zähler der einzelnen Zustände oder der Patterns zurückgesetzt. Wird selbst nach der Rotation um Achse 1 kein Pattern erkannt, startet der Hobbit eine Rotation der Achse 5, wie schon im vorigen Fall vorgenommen (zweiter Pfad von rechts). Es wird hier allerdings nur einmal um die Achse 5 rotiert, ebenfalls um 90 Grad. Hat auch diese Armbewegung keine Erkennung eines Patterns gebracht, wird davon ausgegangen, dass ein Objekt ergriffen wurde. Der Service-Call wird mit einer Null quittiert. Es sei hier erwähnt, dass dieser Fall sehr selten vorkommen kann und als Fehler behandelt werden könnte. Ursachen für dieses Verhalten wäre entweder ein Hindernis zwischen der Kamera und dem Greifer, welches den Greifer verdeckt oder ein Objekt im Greifer welches den kompletten – eigentlich reicht es wenn das side-Pattern abgedeckt wird – Greifer bedeckt.

Da ein Hindernis, wie schon oben erwähnt, im Sichtbereich zwischen Kamera und Greifer keinen adäquaten Fall darstellt, da die Berechnung der Greifpunkte nur mit Sicht auf das Objekt erfolgen kann, wird dieser nicht extra behandelt. Der Fall, dass die Kameradaten fehlerhaft sind, wird hier ebenfalls nicht diskutiert, da dies vom Kameraknoten behandelt werden sollte. Alle weiteren Möglichkeiten des zweiten Pattern-Matchings werden analog zu den oben erklärten Abläufen durchgeführt.

7. Experimente und Ergebnisse

Zwei wichtige Fakten sollen erneut aufgegriffen werden. Einerseits kann es, wie schon weiter oben gezeigt, durch die Bewegung des Armes oder eines Teils des Armes zu Spiegelungen der Beleuchtung am Pattern kommen. Dies muss in Programmablauf berücksichtigt werden. Andererseits kann es von Vorteil sein, dass die Position zwischen der Kopfkamera des Hobbit und dem Greifer beim eigentlichen Zugreifen möglichst gleich bleibt. Da die Greifpunkte schon zuvor errechnet wurden und der Greifer in eine Initialposition gebracht wurde, soll dieser lediglich geöffnet und geschlossen werden. Eine geringfügige Bewegung ist natürlich unumgänglich, um das Objekt zu umschließen. Diese Bewegung muss jedoch, wird der Fall des Ergreifens eines Objektes von einem Tisch bedacht, nicht durch Manipulation des Armes erfolgen. Vielmehr kann es ausreichend sein, den gesamten Hobbit auf das Objekt zufahren zu lassen. Somit wäre die Position zwischen Kamera und Greifer ident. Dies hätte des Weiteren den Vorteil, dass der Öffnungs- und Schließprozess bei Sicht auf das Pattern kontrolliert werden könnte.

Schon bei der ersten Testreihe stellte sich heraus, dass die zuvor verwendeten Rotationswinkel keine ausreichend gute Basis für die Greiferkennung darstellen. Dies resultiert aus der Verallgemeinerung der Zustände für die einzelnen Greiferpositionen. Es gibt eine Vielzahl an Spezialfällen (beispielsweise, dass sich der Greifer außerhalb des Sichtbereiches befindet), die nicht alle mit einer relativen Armbewegung abgedeckt werden können. Obwohl diese Vorgangsweise ein guter Schritt in die richtige Richtung war, sollte der Algorithmus weiter verbessert werden, um eine größere Sicherheit bei der Erkennung des Greifversuches zu gewährleisten! Eine Relativbewegung des Armes ist jedenfalls nicht ausreichend. Außerdem musste, kurz nachdem die ersten Testreihen des vorgestellten Ablaufes abgeschlossen wurden, eine Umstellung des ROS-Systems auf die neue Distribution *Hydro* erfolgen. Da diese Änderung nichts mit der Arbeitsweise des Algorithmus zu tun hat, wird darauf nicht näher eingegangen.

Um ein Gefühl für die notwendige Rotation der einzelnen Achsen zu erlangen, wurde unabhängig vom Zustandsablauf eine Videoanalyse gestartet. Aufbauend auf diesen Analysen sollte eine bessere Variante des Zustandsablaufs des Algorithmus entwickelt werden. Zu diesem Zweck wurde der „schlimmste“ Fall – das Objekt und vor allem der Greifer werden hierbei komplett vom Arm des Hobbit verdeckt – des Greifens von oben von einem Objekt, welches am Boden liegt bzw. steht, getestet.

Nach gründlicher Überlegung wurde festgestellt, dass eine Betrachtung der ersten und der fünften Achse ausreichend für die Analyse ist, da alle weiteren Achsen deutlich weniger Einfluss auf die Sichtbarkeit des Greifers haben. Außerdem entspricht diese Vorgehensweise der intuitiven Bewegung des menschlichen Armes. Wird beispielsweise angenommen, dass sich ein Objekt stehend am Boden befindet, kann leicht nachvollzogen werden, dass ein Mensch die Hand nicht seitlich orthogonal vom Körper streckt, also das Ellenbogengelenk ausstreckt und den Ober- und Unterarm in Verlängerung des Schlüsselbeines vom Körper platziert, sondern dass meistens nur der Ellenbogen abgewinkelt wird, sich die Schulter kaum bewegt und der Oberarm somit entlang des Brustkorbes in Richtung Hüfte zeigend verläuft. Falls notwendig wird die Schulter leicht rotiert, sodass der Oberarm Richtung Zehenspitzen zeigt und einen kleinen Winkel mit dem Brustkorb bildet. Auf Grund der unterschiedlichen Anatomie des Hobbitarms und des menschlichen Arms wurde eine Näherung ausgeführt. Die Achsen 1 und 2 sollen dabei das Schultergelenk nachbilden. Achse 1 wird für die leichte Rotation der Schulter

7. Experimente und Ergebnisse

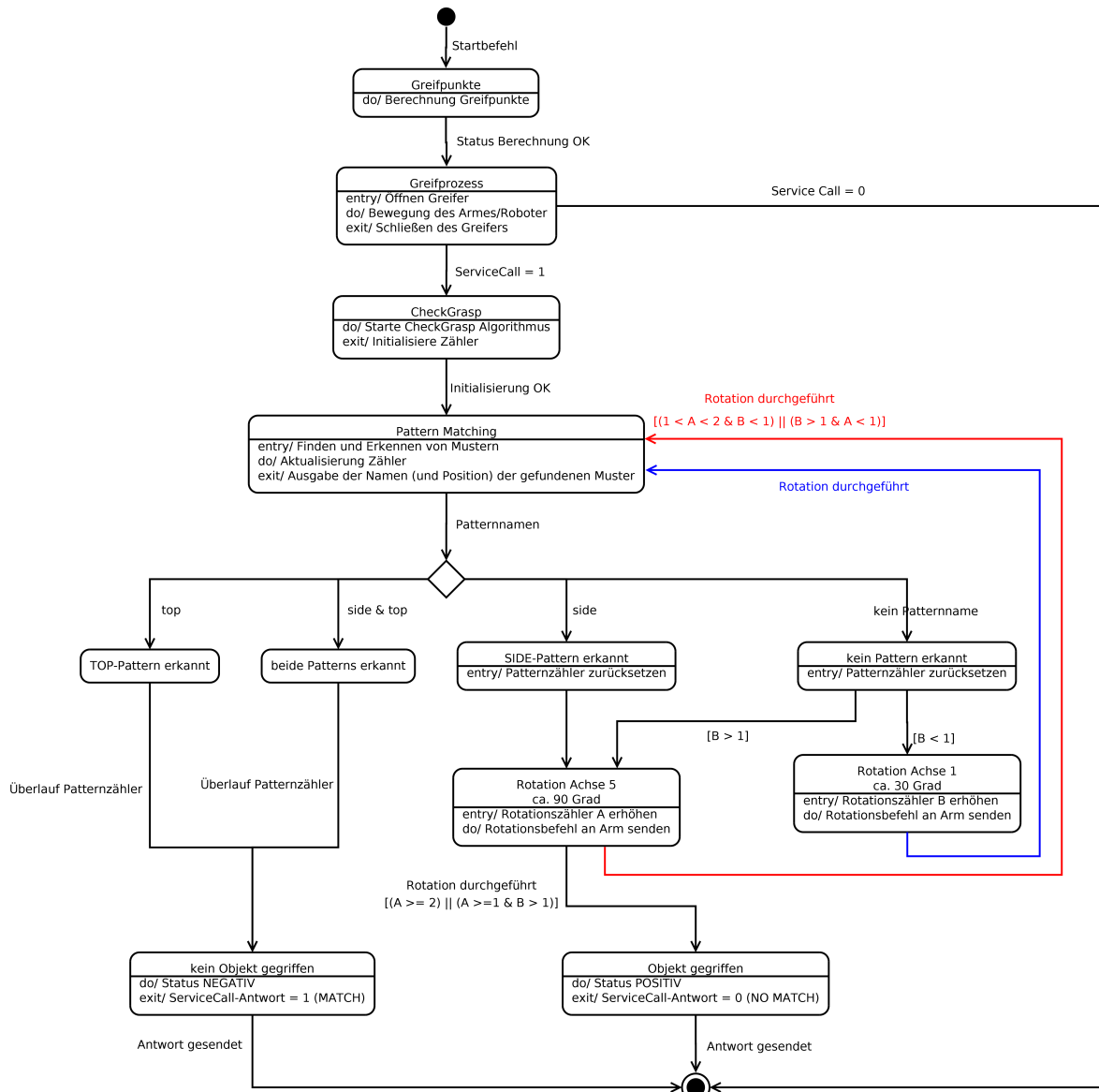


Abbildung 7.11.: Ablaufdiagramm Greifererkennung V1.0

7. Experimente und Ergebnisse

verwendet. Da bei der Ergreifung eines am Boden stehenden Objektes von oben die Position der Achse 4, welche am ehesten als Ellenbogengelenk bezeichnet werden kann, weitgehend vorgegeben ist, ist nur noch eine Rotation der Achse 5, also eine Rotation des Unterarmes, für die Patternerkennung nötig. Bei der Analyse der Rotationstest der einzelnen Achsen wurde der Arm von der Ausgangsposition in Schritten von ca. 4° (gemessen von der Ausgangsposition) um die Achse 1 gedreht, sodass festgestellt werden konnte, ab welchem Winkel der Greifer von der Kamera erfasst werden kann. In jeden Schritt wurde die Achse 5 um den kompletten Bewegungsbereich ($> 360^\circ$) gedreht, um die Möglichkeit der Patternerkennung bei ungehinderter Sicht auf den Greifer zu garantieren. Ab einem Winkel der Achse 1 von ca. 19° , gemessen zum Lot, konnte das top-Pattern bei geschlossenem Greifer erkannt werden. Abbildung 7.12 zeigt auf der linken Seite die Ausgangsposition. Auf der rechten Seite ist der Sachverhalt der Drehung der Achse 1 um 19° und dem erstmaligen Erkennen des top-Patterns dargestellt.



Abbildung 7.12.: Videoausschnitte der Rotation des geschlossenen Greifers (Achse 1 bei 0° bzw. bei 19°).

Steht die Achse 1 bei einem Winkel von ca. 60° , so wird das top-Pattern „perfekt“ erkannt (Matching oberhalb des Schwellwertes und Anzeigen der roten statt der weißen Umrandung). Das side-Pattern hingegen wird erst bei einem Winkel von ca. 68° perfekt erkannt. Ein Grund dafür ist die Entfernung zwischen Kamera und side-Pattern und die damit verbundene wahrgenommene Größe des Patterns. Veranschaulicht wird dies in Abbildung 7.13.



Abbildung 7.13.: Videoausschnitte der Rotation des geschlossenen Greifers (Achse 1 bei 60° für die „perfekte“ Erkennung des top-Patterns bzw. bei 68° für die „perfekte“ Erkennung des side-Patterns). Für die Erkennung des side-Patterns wurde ebenfalls eine Rotation der Achse 5 durchgeführt.

Wird der Greifer leicht geöffnet (Öffnungswinkel ca. 7 Grad) – dies soll den Fall simulieren, dass sich ein kleines Objekt im Greifer befunden hatte, es jedoch herausgefallen

7. Experimente und Ergebnisse

ist – so wird das top-Pattern kurzzeitig fälschlicher Weise bei einem Rotationswinkel der Achse 1 von ca. 42° erkannt. Da dieser Effekt bei den nächsten beiden Rotationsstufen nicht mehr auftrat (45° und 49°), kann von einem sporadischen Fehler gesprochen werden. Erst bei einem Winkel von 53° trat dieser Effekt öfters auf (siehe Abbildung 7.14).



Abbildung 7.14.: Videoausschnitte der Rotation des leicht geöffneten Greifers (Achse 1 bei 42° bzw. bei 53°).

Dieser Fakt stellt ein großes Problem der Bewegung des Armes dar. Es zeigte sich erneut, dass die Erkennung der einzelnen Patterns sehr von der Position und Lage der Patterns zur Kamera abhängt. Wird eine relative Bewegung einzelner Achsen im Vergleich zur Greifposition ausgeführt, kann es, durch Fehlerkennungen während der Bewegung, zur mehrfachen Erhöhung des Matchzählers kommen. Somit sollte dieser nach Beendigung der Rotation zurückgesetzt werden, um ein falsches Ergebnis der Greifererkennung zu verhindern.

Befindet sich eine Getränkedose im Greifer, kann eindeutig festgestellt werden, dass sich ein Objekt im Greifer befindet, weil das top-Pattern nie gefunden wird. Das seitliche Pattern kann hierbei schon ab einem Winkel von 49° erkannt werden. Grund dafür ist die Deformation des Greifers. Abbildung 7.15 zeigt einige Ausschnitte aus den aufgenommenen Videos um diesen Sachverhalt zu verdeutlichen.

7. Experimente und Ergebnisse

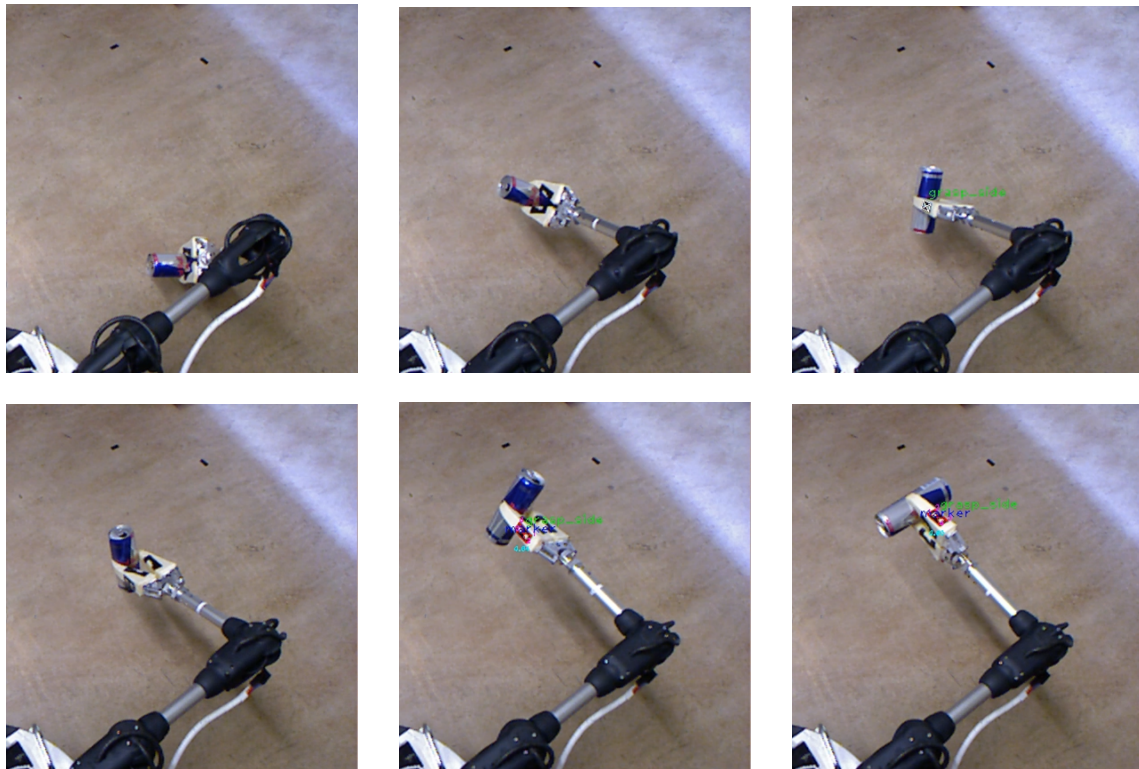


Abbildung 7.15.: Videoausschnitte der Rotation des Greifer mit Dose. Hier kommt es in jedem Fall zu einer korrekten Analyse des Greifzustandes.

Als nächstes Objekt wurde ein USB-Stick, der sowohl „stehend“ als auch seitlich aufgekantet vom Greifer erfasst wurde, verwendet. Befindet sich der USB-Stick liegend im Greifer, wird das top-Pattern erstmals bei einer Drehung der Achse 1 von 64° erkannt. Bei der Drehung der Achse 5 wurde zwar teilweise ein Pattern gefunden, aber nicht als top-Pattern erkannt. Deswegen wird der vorgestellte Algorithmus ein korrektes Ergebnis liefern (nämlich „NO Match“ => Objekt im Greifer), weil über die Höhe des Zählers sporadische Fehlerkennungen herausgefiltert werden. Außerdem kann, wie schon zuvor erwähnt, ein Zurücksetzen des Zählers nach dieser Rotation erfolgen. Erst ab einem Rotationswinkel der Achse 1 von 75° (letztes Bild von Abbildung 7.16) erkennt der Algorithmus fälschlicherweise, dass kein Objekt im Greifer ist, weil das top-Pattern identifiziert wurde. In Abbildung 7.16 können die einzelnen Positionen des USB-Sticks in der liegenden Variante veranschaulicht werden.

Falls sich der USB-Stick stehend im Greifer befindet, ist das Ergebnis besser. Lediglich bei einem Winkel von 86° und 90° kommt es zu sporadischen Erkennungen des top-Patterns. Dies wird jedoch, wie zuvor erwähnt, durch den Schwellwert des Zählers herausgefiltert. Somit kann im Falle des stehenden USB-Sticks eine korrekte Aussage über den Greiferfolg getätigt werden. Der Grund liegt hauptsächlich darin, dass die Breite des USB-Sticks größer ist als dessen Tiefe. Somit entsteht ein größerer Spalt zwischen den Fingern des Greifers. Dies ist in der folgenden Abbildung (Abbildung 7.17) gezeigt. In den letzten beiden Bildern ist ersichtlich, dass beide Patterns erkannt werden, wobei die Erkennung des top-Patterns hier sehr fragwürdig ist. Da diese beiden Greiferpositionen offensichtlich keine sinnvolle Referenz für die Erkennung des

7. Experimente und Ergebnisse

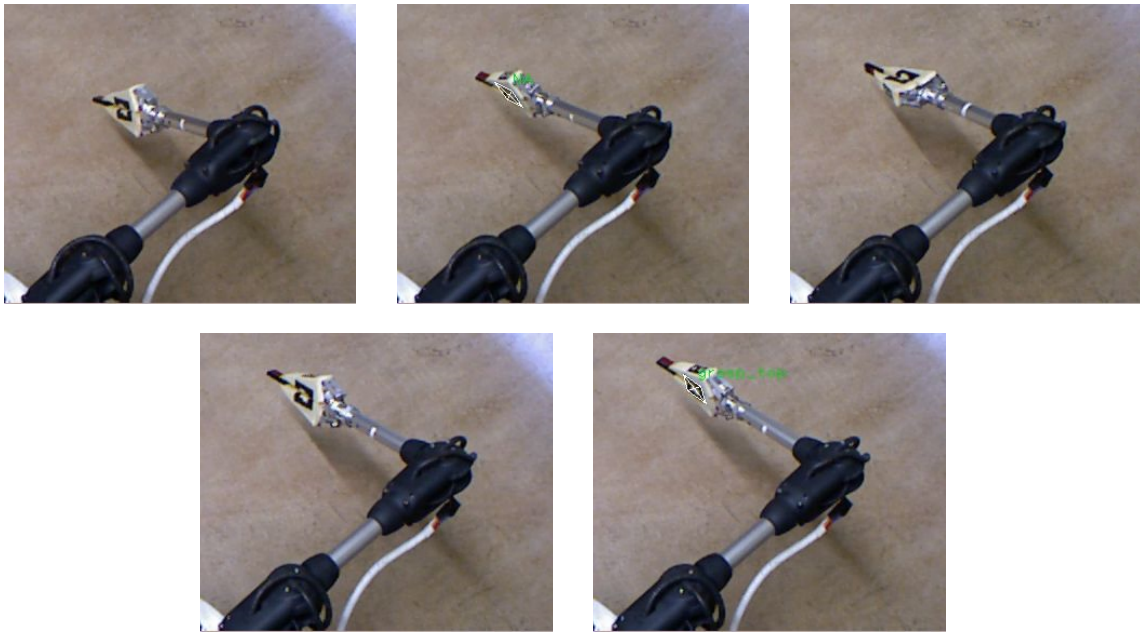


Abbildung 7.16.: Videoausschnitte der Rotation des Greifer mit USB-Stick (liegend). Eine Fehlererkennung ist bei einer Schrägansicht (wie auch schon weiter oben dargelegt, möglich (letztes Bild unten).

top-Patterns bieten, müssen diese beiden jedenfalls vermieden werden.



Abbildung 7.17.: Videoausschnitte der Rotation des Greifer mit USB-Stick (stehend). Bei zwei extremen Schräglagen kommt es auch hier zu einer falschen Erkennung des top-Patterns.

7.3.2. Algorithmus mit absoluter Bewegung

Aufgrund der Ergebnisse der letzten Analyse wurde der Ablauf des Programmes folgendermaßen angepasst: Anstatt von der Greifposition eine relative Bewegung (Drehung bspw. der Achse 1 um 30 Grad) durchzuführen, soll der Arm in eine fix vorgegebene Position gebracht werden. Dies hat den Vorteil, dass sämtliche Positionen des Greifers, welche durch die relative Bewegung bezüglich der Ausgangsposition entstanden sind und zu Fehlerkennungen führen können, vermieden werden. Dadurch wird die

7. Experimente und Ergebnisse

Erkennung zuverlässiger. Ein Nachteil dieser Vorgehensweise ist, dass in den meisten Anwendungsfällen für die Bewegung zu einer fix vorgegebenen Position des Greifers eine umfassendere Bewegung der einzelnen Achsen notwendig ist. Dies ist offensichtlich, da für die Bewegung eine synchronisierte Bewegung aller Achsen vorgenommen wird. Diesem Nachteil, der sich allerdings nur in einer etwas längeren Ablaufdauer des Algorithmus widerspiegelt, steht eine stark erhöhte Zuverlässigkeit des Algorithmus gegenüber. Dies alleine rechtfertigt eine Armbewegung zu einer vordefinierten Position. Natürlich muss auf die Vermeidung von Kollisionen geachtet werden.

Wurde diese Position erreicht und das top-Pattern wurde nicht gefunden, soll versucht werden den Greifer erneut zu schließen. Durch diese Maßnahme soll jener Fall behandelt werden, dass ein zuvor ergriffenes Objekt nach erfolgter Armbewegung aus dem Greifer gefallen ist. Eine Ursache hierfür wäre, dass das Objekt schlecht ergriffen wurde. Dies bedeutet, dass die Angreifpunkte nicht adäquat waren, das Objekt durch seine große Masse nicht bewegt werden kann oder ein anderer Gegenstand in der Umgebung bei der Armbewegung ein Hindernis darstellt. Wird das top-Pattern nach dem erneuten Schließen nicht gefunden, soll eine zusätzliche (geringe) Rotation des Armes um die Achse 5 stattfinden. Dies soll etwaigen Spiegelungen und dem Toleranzproblem der Mechanik des Greifers – hinsichtlich des kompletten Schließens – vorbeugen. Wird das top-Pattern auch während bzw. nach dieser Drehung nicht gefunden, gilt der Greifversuch als erfolgreich. Das Ablaufdiagramm ist in Abbildung 7.18 zu sehen. Der Ablauf wird erneut detailliert beschrieben.

Zu Beginn werden die gleichen Aufgaben erfüllt, wie schon in der ersten Version des Algorithmus. Nachdem die eigentliche Kontrolle des Greifzustandes das erste Mal durchgeführt wurde, wird erneut in Abhängigkeit der gefundenen Patterns vorgegangen. Wird das top-Pattern gefunden – unabhängig davon ob auch das side-Pattern gefunden wurde – so kann davon ausgegangen werden, dass kein Objekt im Greifer vorhanden ist. Ist nur das side-Pattern zu erkennen, soll der Arm in die vordefinierte Kontrollposition gebracht werden. In diesem Zustand wird immer noch nach Patterns gesucht und der Zähler weiterhin erhöht, falls ein Pattern gefunden wird.

Nachdem nun die gewünschte Position erreicht wurde, soll der Greifer erneut das Kommando „Greifer schließen“ erhalten um dem Problem des Fallenlassens eines Objektes Herr zu werden. Nach dem erneuten Schließen des Greifers (bzw. auch nicht, falls ein Objekt im Greifer ist), kommt es zur identen Entscheidungsfindung wie schon zu Beginn der Greifererkennung nach dem Service-Call. Wird das top-Pattern oder werden beide Patterns gefunden, konnte das Programm erfolgreich beendet werden. Es befindet sich kein Gegenstand im Greifer. Wird allerdings erneut nur das side-Pattern erkannt, was aufgrund der vorher definierten Kontrollposition nur in einigen speziellen Fällen vorkommen kann, erfolgt eine kleine Rotation des Unterarms des Hobbys. Somit soll sichergestellt werden, dass keine Spiegelung oder Ähnliches Grund für die „Nicht-Erkennung“ des top-Patterns ist. Die Drehung erfolgt hierbei zuerst um -10° und danach um insgesamt $+20^\circ$. Somit kann ein Bereich der Achse 5 von $-/+10^\circ$ um die Kontrollposition abgedeckt werden. Nun erfolgt die dritte Kontrolle welche Patterns gefunden wurden. Die Behandlung der ersten beiden Fälle, das Finden des top-Patterns und beider Patterns, ist ident zur vorigen. Wird erneut nur das side-Pattern erkannt, kann von einem ergriffenen Objekt ausgegangen werden. Der Service-Call wird mit „Null“ quittiert. Wird während der gesamten Prozedur kein Pattern erkannt,

7. Experimente und Ergebnisse

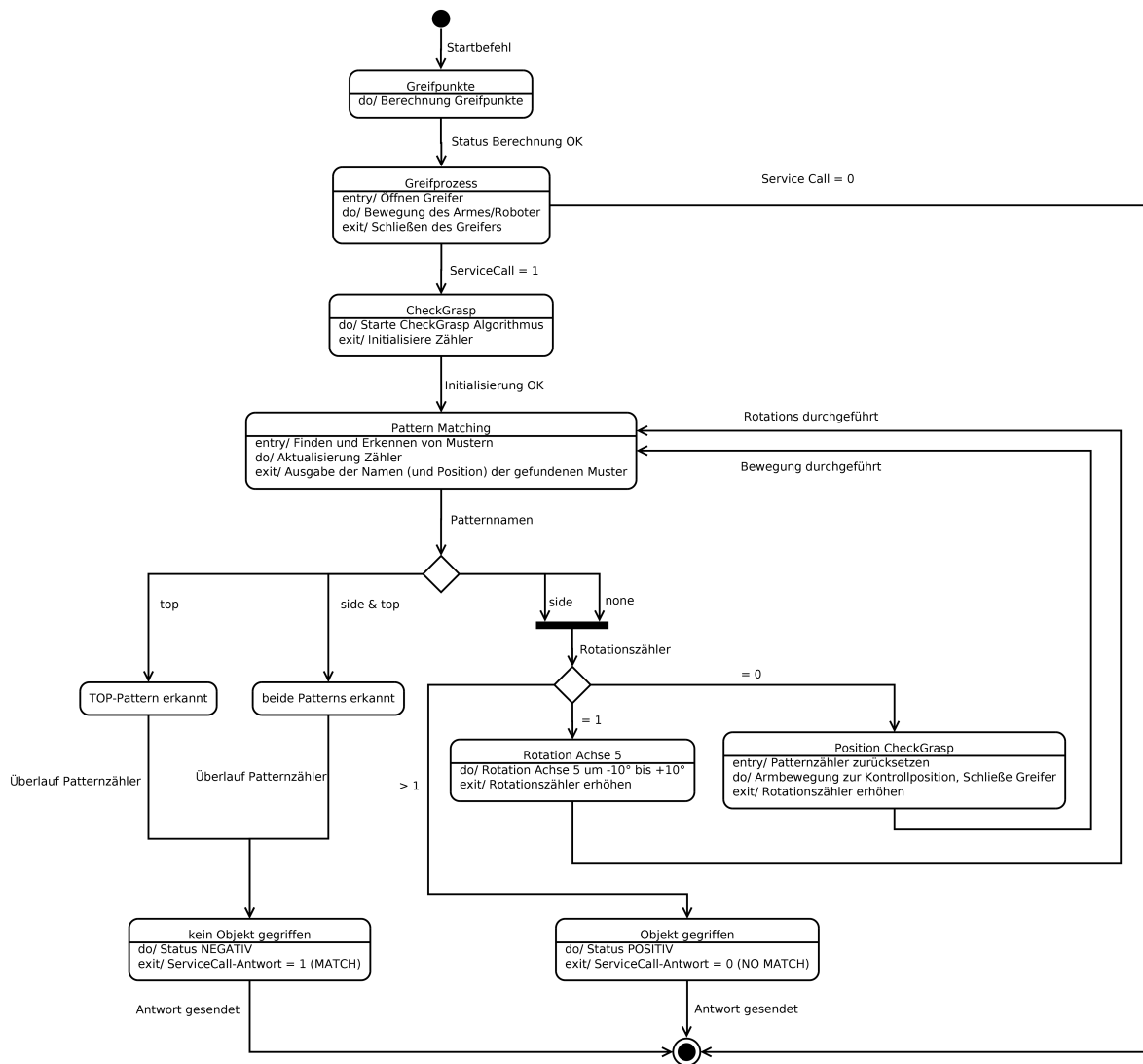


Abbildung 7.18.: Ablaufdiagramm Greifererkennung V2.0

muss sich, vor allem nach der Bewegung des Armes, ein Objekt im Greifer befinden. Fehlerfälle der Kamera bzw. der Datenübertragung der Kamera und Ähnliches sind davon ausgenommen und werden hier nicht behandelt.

7.3.3. Algorithmusvalidierung und -tests

Werden die beiden Abläufe verglichen, ist sofort ersichtlich, dass der zweite den ersten in seiner Effizienz bzw. Genauigkeit übertrifft. Durch die vordefinierte Kontrollposition sind annähernd idente Bedingungen, bis auf Spiegelungen der Beleuchtung, für jede Greiferkennung gegeben. Dies erleichtert die Parametrisierung des Algorithmus. Alle charakteristischen Werte, beispielsweise dem Schwellwert der sicheren Patternerkennung, können auf diese Position abgestimmt werden.

Eine zusätzliche (allgemeine) Problematik wurde durch das erneute Schließen des Greifers erkannt. Sollen rein optische Informationen für die Greiferkennung verwendet werden, ist auch eine Statusabfrage des Greifers nicht zulässig. Beispielsweise könnte nach dem Beginn der Armbewegung zur Kontrollposition geprüft werden ob diese auch wirklich erreicht wurde. Dazu sind im ROS-Paket, welches für die Armbewegung des Hobbots zuständig ist, Funktionen vorhanden. Die Position der Greiferfinger selbst könnte ebenfalls kontrolliert werden. Auf die Einbindung dieser Funktionen wurde jedoch, wegen der Anforderungen an den Lösungsansatz, verzichtet. Es sei trotzdem erwähnt, dass die Zuverlässigkeit der Greiferbewegung ausschlaggebend für den Erfolg der Greiferkennung ist. Als Ausgangsfall soll für ein kurzes Gedankenexperiment das Schließen des Greifers nicht ordnungsgemäß funktionieren. Ohne zusätzliche Überprüfung, sei es auf visuellem Wege unter Einbezug von Tiefeninformation oder unter Verwendung von Motorrückmeldungen, kann zwischen dem geöffneten Greifer und dem Fall eines Objektes im geschlossenen Greifer nicht unterschieden werden.

Die ersten Versuche mit dem aktualisierten Programmablauf konnten erfolgreich abgeschlossen werden. Dabei sollten mehrere Objekte (eine Getränkedose, eine kleine Wasserwaage, ein Bleistift und ein Radiergummi) aus zwei Positionen gegriffen werden und anschließend festgestellt werden, ob die Objekte auch tatsächlich ergriffen wurden oder nicht. In der ersten Position befinden sich die Objekte am Boden und werden vom Greifer von oben gegriffen. Dabei ist zu beachten, dass die Objekte durch den Greifarm selbst zu Beginn verdeckt sind. Bei der zweiten Position befinden sich die Objekte auf einem kleinen Tisch. Da der erste Fall einer bodennahen Greifposition schon intensiv vorgestellt wurde, soll hier hauptsächlich auf den Anwendungsfall „Ergreifen eines Objektes, welches auf einem Tisch steht bzw. liegt“ eingegangen werden.

Begonnen wurde mit der Verifikation des Algorithmus mit der Greifposition am Boden. Dabei wurden mehrere Objekte an eine bestimmte Stelle neben dem Hobbit abgelegt. Das Greifscenario bestand darin, diese Objekte von oben zu ergreifen. Die Armposition wurde so gewählt, dass der Greifer in der Greifposition fast komplett verdeckt war. Dies stellt gleichzeitig den Worst-Case dar. Kann der Greifer, auf Grund von Verdeckungen, zu Beginn nicht gesehen werden, muss der Arm unweigerlich bewegt werden. Die folgenden Bilder zeigen die Greifscenarios, bzw. die Endposition des Greifers nach Ablauf des Algorithmus, mit verschiedenen Objekten (Abbildung 7.19).

Trotz der zufriedenstellenden Ergebnisse traten Probleme bei der Durchführung auf. Der gravierendste Fehlerfall trat auf, als die Kommunikation des Armes in einem

7. Experimente und Ergebnisse

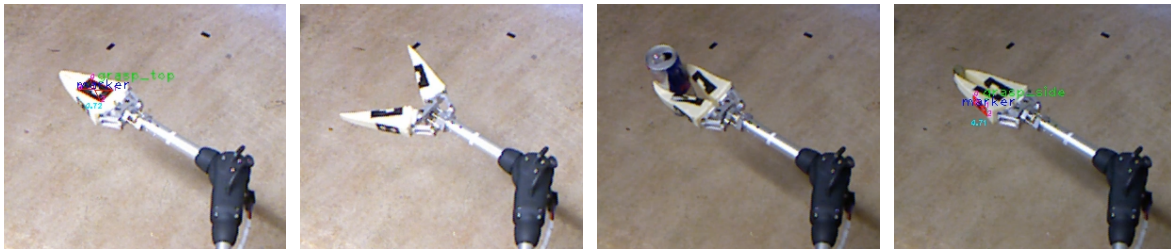


Abbildung 7.19.: Die Analyseposition des Greifszenarios am Boden. Der geschlossene Greifer ohne Objekt wird mit dem geöffneten und dem geschlossenen Greifer inklusive zweier Gegenstände, einer kleinen Wasserwaage und einer Dose, verglichen. Sämtliche Ergebnisse der Greifanalyse waren korrekt.

Endloszustand verharrte als dieser erneut geschlossen werden sollte. Dabei befand sich die Aluminiumdose im Greifer. Auf Grund des erheblichen Widerstandes der Dose sollte der Motor des Greifers den Versuch des Schließens abbrechen und, allerdings mit einer Fehlermeldung bei der Statusabfrage des Greifers, zur normalen Prozedur zurückkehren. Der Ablauf wurde jedoch nicht weiter ausgeführt! Selbst bei mehreren Versuchen zeigte sich das gleiche Problem. Auf Grund des Verharrens im Endloszustand überhitzte schlussendlich sogar der Motor des Greifarmes und musste ausgetauscht werden. Es konnte verifiziert werden, dass der Fehler in der Kommunikation durch die bereits vorhandene Greifersteuerung und nicht durch deren Einbindung in das ROS-Paket verursacht wurde. Dieses Kommunikationsproblem trat außerdem nicht auf, wenn der Greifer (ohne Objekt) zu anfangs leicht geöffnet in der bodennahen Position platziert wurde und nach der Bewegung geschlossen wurde. Da dieses Verhalten nicht der Spezifikation des Armes entsprach und die Behebung dieses Problems nur durch Veränderung der Armsteuerung bewerkstelligt werden konnte, musste von dem zusätzlichen Schließvorgang vorerst abgesehen werden. Ohne erneutes Schließen des Greifers konnte festgestellt werden, dass sowohl bei großen Objekten, wie beispielsweise der Aluminiumdose, als auch bei kleineren Objekten, wie der kleinen Wasserwaage (siehe letztes Bild von Abbildung 7.19), der Algorithmus die gewünschten Ergebnisse brachte.

Das zweite GreifszENARIO, welches gewählt wurde, behandelt das Ergreifen eines Objektes, welches sich auf einem niedrigen Tisch befindet. Der verwendete Algorithmus blieb unverändert. Als erste Testobjekte dienten erneut die Aluminiumdose und die kleine Wasserwaage. In Abbildung 7.20 ist eindeutig zu erkennen, dass Objekte als ergriffen erkannt werden können und der Algorithmus somit verifiziert werden konnte. Vor allem aus dieser Anfangsposition (siehe Abbildung 7.20 links oben) kann es bei kleinen Objekten sehr leicht zu Fehleinschätzungen kommen, da schon zu Beginn das top-Pattern zu sehen ist. Falls sich nun ein kleiner, schmaler Gegenstand im Greifer befindet und dieser in der unteren Hälfte der Greiferflächen erfasst wird, kann es sein, dass der Greifer nur so wenig geöffnet ist, dass das top-Pattern erkannt wird und somit kein Objekt im Greifer erkannt wird.

Es zeigt sich erneut deutlich, dass die Erkennung des Patterns, vor allem des top-Patterns, sehr positionsabhängig ist. Dies spielt für den Algorithmus insofern eine große Rolle, weil sowohl die Startposition, also die errechnete Greifposition, wie auch die

7. Experimente und Ergebnisse

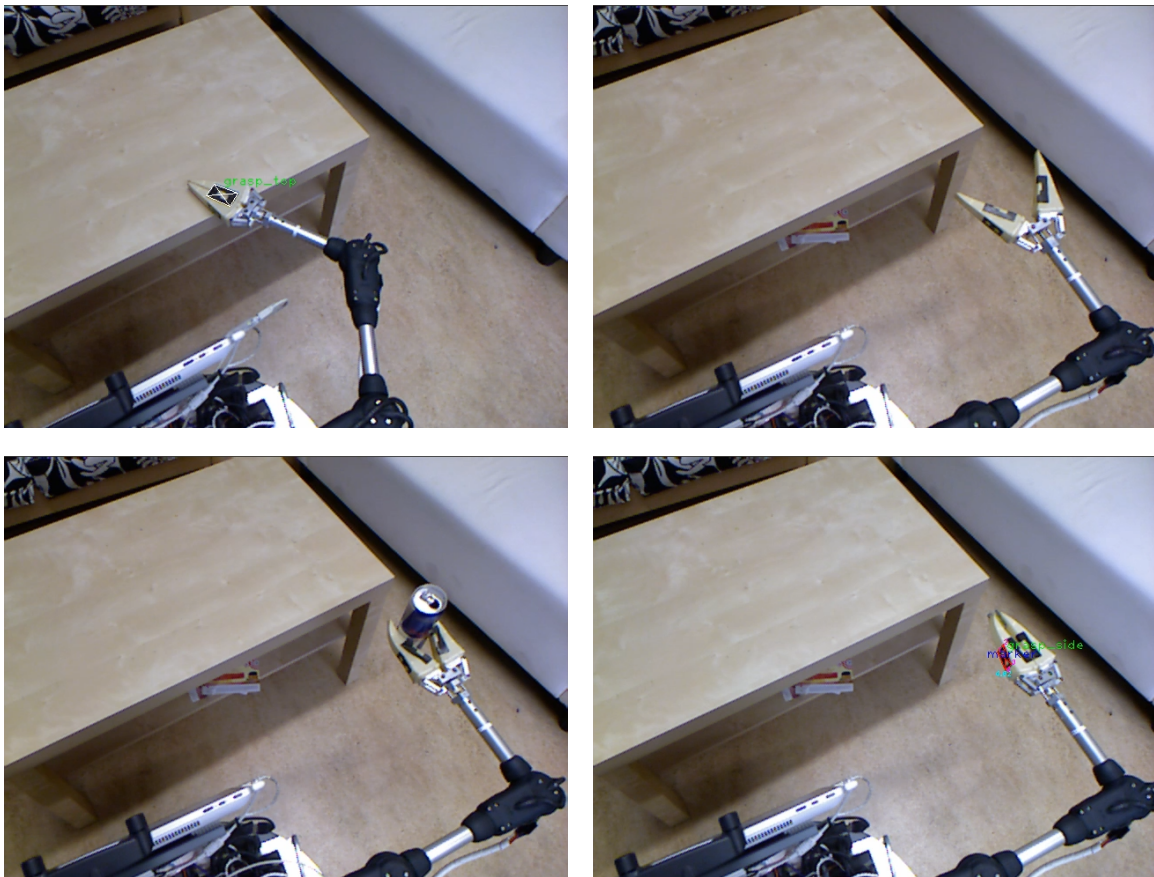


Abbildung 7.20.: Erste Analyseposition des Greifszenarios am Tisch. Als Testobjekte wurden erneut die kleine Wasserwaage (rechts unten) und eine Aluminiumdose (links unten) verwendet. Beim Vergleich mit dem geöffneten (rechts oben) und geschlossenen (links oben) Greifer ergaben sich ebenfalls korrekte Ergebnisse.

7. Experimente und Ergebnisse

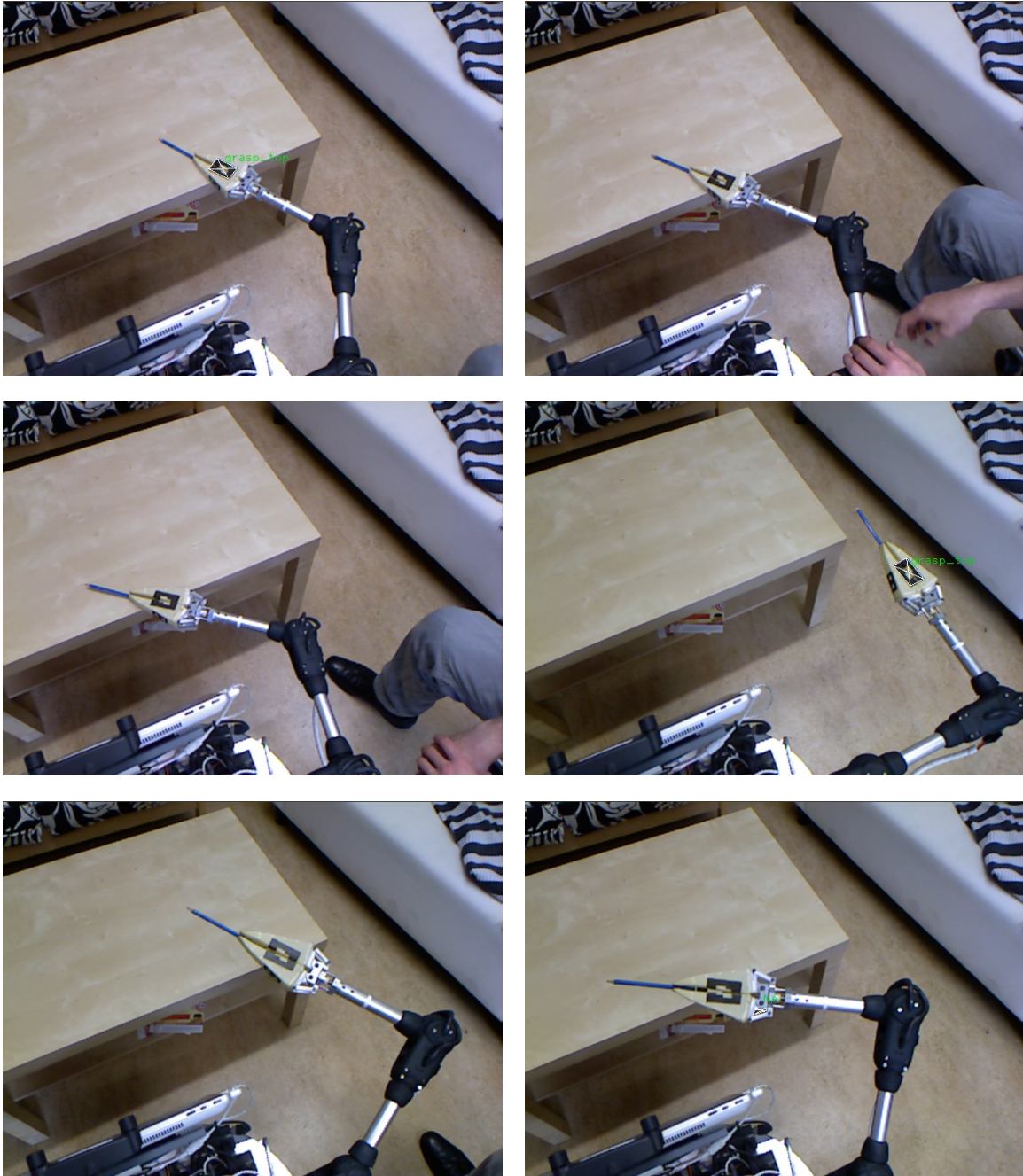


Abbildung 7.21.: Positionen eines Bleistifts im Greifer anhand des Greifszenarios am Tisch. Erneut wurde festgestellt, dass die Position des Bleistifts im Greifer von großer Bedeutung ist. Wird der Bleistift am unteren Ende der Innenkante der Greiferfinger erfasst, kann es zu Fehlerkennungen des top-Patterns kommen (links oben und rechts in der Mitte). Erst durch Variation der Analyseposition kann eine korrekte Analyse gewährleistet werden (rechtes unteres Bild).

7. Experimente und Ergebnisse

Endposition für die Markeranalyse ausschlaggebend sind. Ein paar Eindrücke zu dieser Problematik sind in Abbildung 7.21 zu sehen.

Für diese Abbildung bietet sich die Verwendung eines kleinen Objektes, wie beispielsweise einem Bleistift, besonders an, da große Objekte keine Hürde für den Algorithmus darstellen. Dies bedeutet aber auch, dass ein Bleistift hart an der Grenze zu nicht mehr erkennbaren Objekten auf Grund seiner Größe ist. Es wurden drei unterschiedliche Start- und Endpositionen willkürlich gewählt, um einen Vergleich anstellen zu können. Wie anhand des ersten Bildes in Abbildung 7.21 in der linken Spalte, als auch anhand des zweiten Bildes in der rechten Spalte erkennbar ist, müssen die Start- und Endposition optimiert werden, um gute Ergebnisse zu erreichen. In diesem Fall wurde an der Verbesserung der Positionen anhand empirischer Tests gearbeitet. Dabei wird das Hauptaugenmerk auf die Analyseposition, diese entspricht der Endposition, gelegt, da das Ergreifen von Objekten in anderen ROS-Paketen durchgeführt wird und somit kein direkter Einfluss auf die Anfangsposition genommen werden kann.

Abbildung 7.22 vergleicht zwei sehr ähnliche Endpositionen, welche jedoch für das Programm weitläufige Unterschiede bringt! Im ersten Fall wird kein Pattern erkannt, im zweiten sehr wohl. Um die Unabhängigkeit des Algorithmus von der Anfangsposition zu erreichen kann höchstens die Erkennung vor der Bewegung deaktiviert werden.

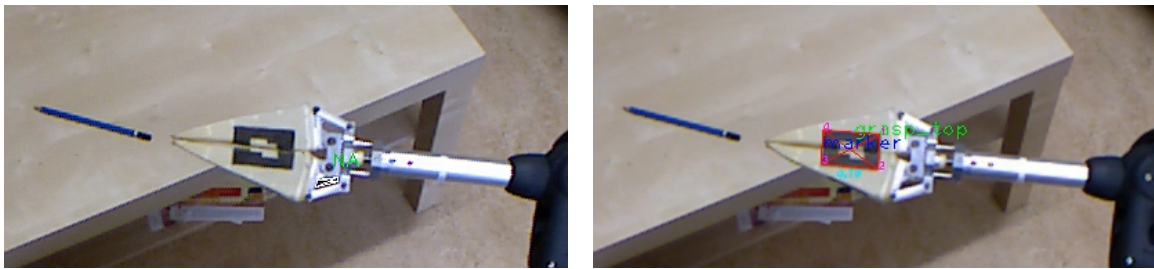


Abbildung 7.22.: Probleme in der Endposition des Greiferszenarios am Tisch. Der Bleistift wurde nicht vom Tisch ergriffen. Im linken Bild ist der Fall ersichtlich bei dem das Pattern, auf Grund von mechanischen Toleranzen beim Schließen, nicht erkannt wurde. Im rechten Bild, welches sich durch eine geringfügige Positionsänderung des Greifers unterscheidet, wurde das Pattern korrekt erkannt.

Da auf Grund der Beschädigung des Greifermotors, wie schon oben erwähnt, auf das erneute „Kontrollschließen“ des Greifers verzichtet wurde, sollte ein weiterer Fehlerfall analysiert werden. In dem Szenario wird ein Objekt zwar am Tisch ergriffen, jedoch beim Aufheben des Armes verloren. Wird der Greifer nach der Hebebewegung nicht geschlossen, sondern bleibt leicht geöffnet, wird fälschlicherweise das top-Pattern nicht erkannt und somit ein Objekt im Greifer festgestellt, welches allerdings nicht vorhanden ist (Abbildung 7.23). Dies soll allerdings nur veranschaulichen, wie wichtig der erneute Schließbefehl des Greifers ist. Ohne diesen kann die Greiferkennung nicht zuverlässig arbeiten!

7. Experimente und Ergebnisse

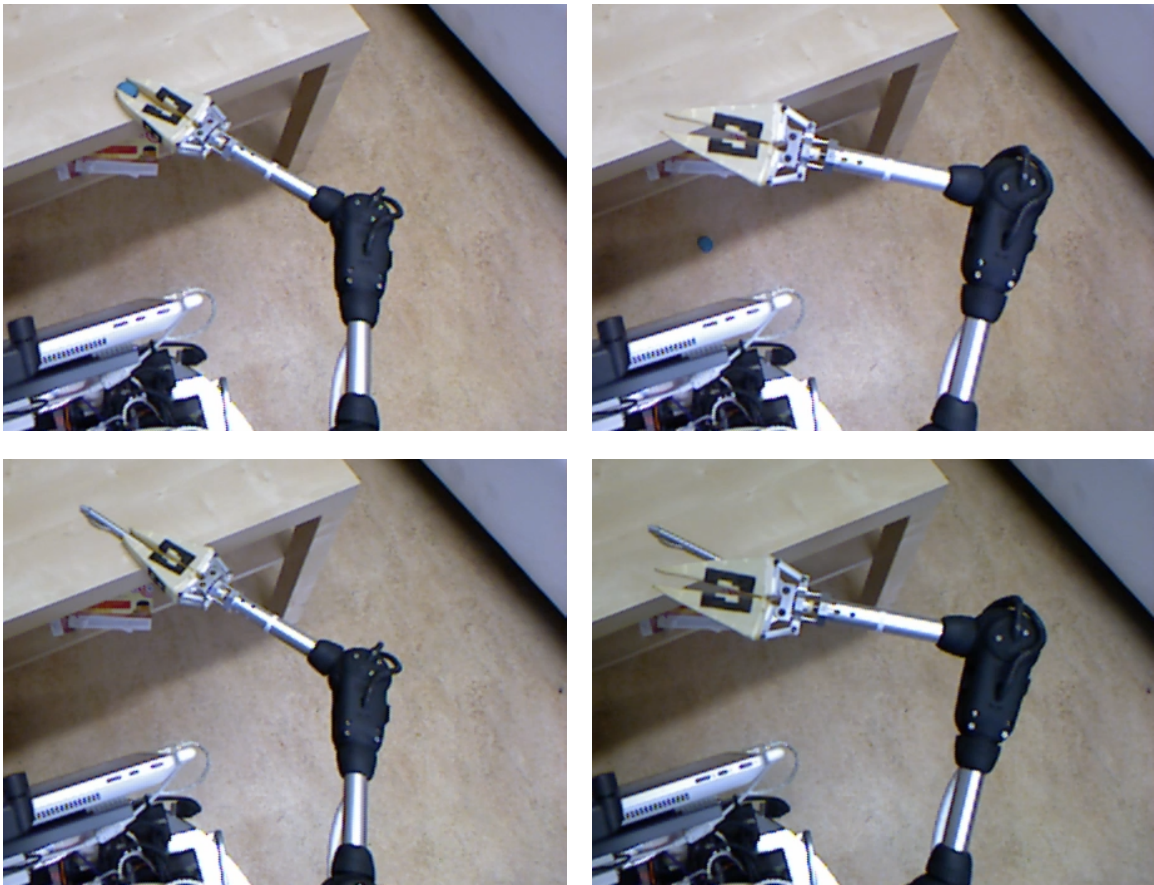


Abbildung 7.23.: Greifszenario Tisch – Simulation eines Fehlers bei Radierer und Kugelschreiber.

Es ist naheliegend, nach den in diesem Kapitel gewonnenen Erkenntnissen, den Greifer direkt nach seinem Schließen in die Analyseposition zu bringen, bevor mit der Mustererkennung begonnen wird. Dies hat jedoch den Nachteil, der längeren Ablaufdauer des Algorithmus im Falle, dass sich kein Objekt im Greifer befindet. Vorteilhaft ist jedoch die exaktere Aussagemöglichkeit über den Erfolg des Greifversuches. Die folgende Tabelle (Tabelle 7.1) fasst die Ergebnisse der Experimente zusammen. Dazu sind in jeder Zelle die Erfolgsrate und die Problemstellen angegeben.

Die Tests wurden in eine Umgebung mit künstlicher Beleuchtung (Leuchtstoffröhren) durchgeführt. Der Boden sowie der verwendete Tisch waren in einem hellen Braunton gehalten (Ahorn bzw. helle Buche), damit der Kontrast zwischen Greifer, bzw. dessen Latexüberzug, markant aber nicht allzu stark ist. Das natürliche Sonnenlicht wurde durch einen Vorhang abschottet. Der Hobbit wurde manuell vor dem jeweiligen Greifszenario positioniert. Eines der besagten Objekte wurde auf dem Tisch oder dem Boden platziert. Danach wurde der Arm in eine Initialposition gebracht, welche vom Entwickler ausgewählt wurde. Diese Position entsprach annähernd der Ruheposition des Armes nach der Referenzierung. Nun wurde der Greifer geöffnet und eine ungefähre Greifposition angesteuert. Dies erfolgte auf Basis von Koordinatenangaben mit einem ROS-Befehl (Kommandozeile).

7. Experimente und Ergebnisse

	Greifposition Boden	Greifposition Tisch
Dose	100% - keine Schwierigkeiten	100% - keine Schwierigkeiten
kl. Wasserwaage	100% - keine Schwierigkeiten	100% - keine Schwierigkeiten
USB-Stick(hockant)	95% - geringe Probleme bei seitlicher Ansicht auf den Greifer vor allem bei/nach einer Rotation um Achse 5	95% - Kleine Unsicherheiten bei schlechter Sicht direkt am Tisch
Bleistift	95% - geringe Probleme bei seitlicher Ansicht auf den Greifer vor allem bei/nach einer Rotation um Achse 5	90% - Schwierigkeiten abhängig von der Angreifposition des Bleistifts

Tabelle 7.1.: Ergebnisse der Experimente nach Ablauf V2.0 anhand von vier verschiedenen Objekten.

Es sei hier erwähnt, dass durch die idente Positionierung bei jedem Greifversuch nur eine geringe Variation der Angreifpunkte erfolgte. Diese Variation wurde durch eine leichte willkürliche Veränderung der Objektposition am Tisch oder Boden bewerkstelligt. Nachdem der Greifer die Endposition erreicht hatte, wurde der Greifer geschlossen und das eigentliche Programm der Greiferkennung gestartet. Anhand der Ausgabe des ROS-Packages am Ende des Erkennungsalgorithmus und einer visuellen Kontrolle durch den Autor wurde festgestellt, ob die Greiferkennung erfolgreich war.

Die angegebene Prozentzahl errechnet sich aus dem Verhältnis der tatsächlich richtig erkannten Ergebnissen der Greiferkennung zu den insgesamt durchgeführten Versuchen. Eine fälschliche oder eine „Nicht-Erkennung“ wurde als Fehler gewertet. Pro Testfall wurden 40 Versuche durchgeführt. Dabei musste beachtet werden, dass der Greifermotor regelmäßig auskühlen musste, was zu Wartezeiten führte. Nach der Hälfte der Versuche musste außerdem eine neue Referenzierungsfahrt gestartet werden, da die Detektionsposition nicht mehr exakt eingenommen wurde. Die Ursache hierfür sind Ungenauigkeiten der Motoren bzw. Drehgebersensoren und die Dehnung der Seilzüge.

Es sei noch einmal erwähnt, dass die Angreifpunkte im gesamten Ablauf willkürlich gewählt wurden. Es wurde nicht näher auf deren Definition eingegangen, da diese in einem vorbereitenden Schritt festgelegt werden und somit nicht Thema der Greiferkennung sind. Werden die Angreifpunkte insofern festgelegt, dass sich das zu ergreifende Objekt möglichst in der Mitte des Greifers befindet, können die Werte in Tabelle 7.1 weiter verbessert werden. Befindet sich ein Gegenstand in der Mitte des Greifers, so wird der weiße Bereich des top-Patterns unterbrochen. Dadurch wird eine Erkennung des Musters unterbunden und das korrekte Ergebnis des Algorithmus gewährleistet.

Schlussfolgerung und Ausblick

Es konnte gezeigt werden, dass sich der gewählte Ansatz, die Greifererkennung anhand des Findens und Erkennens von AR-Marker-Patterns durchzuführen, in der Praxis sehr gut bewährt hat. Im Gegensatz dazu konnten sowohl die Kantenerkennung mittels Canny-Algorithmus wie auch das SIFT-Matching für die Greifererkennung des Hobbit ausgeschlossen werden. Selbst bei kleinen Objekten konnten unter Berücksichtigung der Umstände, wie beispielsweise der Kameraqualität, gute Ergebnisse mit dem AR-Marker-Ansatz erzielt werden.

Ein Vergleich zwischen Aufgabenstellung und Ergebnis unterstreicht, dass das Hauptproblem gelöst wurde und die Rahmenbedingungen eingehalten wurden. Abstriche mussten lediglich hinsichtlich der ungewünschten Bewegung des Armes gemacht werden. Eine Nebenanforderung war es, den Arm möglichst wenig zu bewegen. Diese konnte von keinem Ansatz eingehalten werden. Die Tatsache, dass diese Forderung nicht zwingend war und dass durch deren Nichteinhaltung einige Vorteile entstanden sind, wie beispielsweise die Lösung des Problems der (Selbst-)Okklusion, rechtfertigt diese Entscheidung. Werden die Ergebnisse auf den direkten Anwendungsfall des Hobbit bezogen, kann mit Sicherheit gesagt werden, dass sämtliche anderen Anforderungen erfüllt wurden. Die vorgestellte Lösung ist robust, leistungsstark (schnelle Erkennung, wenig Rechenleistung), universell (für verschiedene Greifpositionen und Greifertypen) verwendbar und preiswert. Selbst kleine Objekte können erkannt werden. Die Erfüllung der Anforderungen bestätigt sich, wenn die Umstände der Betreuung älterer Menschen betrachtet werden. Bzgl. der Verwendung bzw. Greifererkennung von sehr kleine Objekte kann gesagt werden, dass diese (bei der Betreuung älterer Menschen) kaum benötigt werden. Da außerdem meist von einer geordneten Umgebung ausgegangen werden kann, spielt das Thema der Verdeckung eine untergeordnete Rolle. Anhand des eingangs erwähnten Problems „Bring mir einen Apfel aus der Küche!“ kann die Funktion des vorgestellten Ansatzes eindeutig verifiziert werden. Der Apfel kann als Objekt innerhalb des Greifers erkannt und für seine Weiterverwendung transportiert werden.

Es wurde eine rein optische Greifererkennung bewerkstelligt, welche performant, einfach und zuverlässig arbeitet. Die entwickelte Methode ist nicht auf das Hobbit-Projekt

8. Schlussfolgerung und Ausblick

beschränkt, sondern für planare Greifer universell einsetzbar. Für komplexere Greifer, welche der menschlichen Hand ähneln könnte dieses Prinzip beispielsweise insofern adaptiert werden, indem jede Fingerspitze mit einem kleinen Marker versehen wird und anhand der Positionen der Marker zueinander auf das Ergebnis des Greifversuches geschlossen werden kann. Dies stellt allerdings keine optimale Lösung für solche Endeffektoren dar und würde zusätzliche Verbesserungen hinsichtlich der verwendeten Komponenten (beispielsweise die Kamera) nach sich ziehen.

Wie bei allen visuellen Systemen spielt neben der Qualität und Position des Sensors, hier der Kamera, die Beleuchtung und natürlich die Mechanik des Greifers eine besonders große Rolle. Vor allem letztere bereitete zunehmend Probleme. Einerseits wirkt sich dies im nicht exakten Schließen des Greifers aus, andererseits resultierten daraus Beschädigungen des Motors und an Teilen des Seilzuges. Sämtliche Defekte konnte jedoch behoben werden.

Weiterführende Arbeiten könnten beispielsweise die Anpassung bzw. Optimierung der Analyseposition oder das Problem der Armbewegung behandeln. Falls eine Bewegung des Armes nicht zwingend notwendig ist, die direkte Sicht auf den Greifer demnach nicht versperrt ist, würde dies den Prozess beschleunigen. Zu diesem Zweck könnte am hinteren Ende der Greiferfinger je ein zusätzliches Pattern angebracht werden. Ist die direkte Sicht auf den Greifer vorhanden, muss keine Armbewegung durchgeführt werden, da durch das Finden zumindest eines der beiden kleinen zusätzlichen Fingerpatterns die Lage des Greifers identifiziert werden kann. Gesagtes gilt jedoch nur, falls sich kein allzu großer Gegenstand im Greifer befindet. Für diesen Ansatz wäre allerdings eine höhere Kameraauflösung notwendig.

Eine weitere Verbesserung, welche sich nicht direkt auf den Erkennungsalgorithmus bezieht, ist die Implementierung einer Kollisionskontrolle bei der Armbewegung. Ein zusätzlicher Anknüpfungspunkt ist, die Methode der Greifererkennung anhand von AR-Markern mit einer der in Kapitel 4 vorgestellten Lösungsarten zu vergleichen, welche noch nicht implementiert wurde. Dabei wären Vergleichstests hinsichtlich einer alternativen Konturerkennung des Greifers und hinsichtlich optischen Flusses besonders herauszustreichen. Wie schon zuvor erwähnt, kommt auch der erste der beiden Ansätze nur in wenigen Fällen ohne Bewegung des Greifers aus. Die Variante des Optischen Flusses ist immer mit Bewegung verbunden, ist aber aus der Sicht des Autors als gute Vergleichsmöglichkeit zum vorgestellten Lösungsweg in Betracht zu ziehen.

Programmcode ROS package

In diesem Anhang befinden sich die einzelnen Code-Dateien, die im ROS-Paket *check-grasp* enthalten sind. Dabei handelt es sich nur um die Quelldateien, welche im catkin-workspace von ROS im Ordner *src* zu finden sind. Das Paket *ar-marker* wird nicht angegeben. Die Anpassungen einiger ROS-Pakete, welche im Zuge der Umstellung auf die ROS-Distribution *Hydro* durchgeführt werden mussten, sind ebenfalls nicht dokumentiert.

A.1. checkgrasp.py

```

1  #!/usr/bin/python
2
3  # Author: Christopher Kuenzel
4  # Company: TU Wien
5  # Copyright (C) 2014 Ch. Kuenzel
6
7  PKG = 'check_grasp'
8
9  import roslib;
10 #roslib.load_manifest(PKG)
11 import rospy
12 import sys
13 import math
14 import tf
15 import numpy
16 import geometry_msgs.msg
17
18 from std_msgs.msg import String, Header
19 from check_grasp.srv import GraspState
20 from action_sequencer import McDSA
21
22 def startCheckG(status):
23     erg = 0
24     patq = 10 # neccessary quantity of patterncounts
25     status = str(status)
26     status = int(status[3:])
27     # print('datatype input: %s' %type(status))
28     # print('Status = %s' %status)
29     grasp_pos = 5 # 0 = ground, 5 = desk
30     if status >= 1:
31         print('-----')
32         graspChecker = GraspCheck()
33         #graspChecker.run()
34         rotstat_A = 0 # rotation status for axis 5
35         rotstat_B = 0 # rotation status for axis 1
36         rospy.sleep(2.0) # wait for 2 sec until PatternCounter has started
37         step = 0
38         while erg == 0:
39             rospy.sleep(2.)
40             topcount = graspChecker.getTopCount()

```

A. Programmcode ROS package

```

41 sidecount = graspChecker.getSideCount()
42 print('Top-Count=%d' %topcount)
43 if topcount >= patq:
44     print('Top-Counter->%d=>MATCH!!!' %patq)
45     print('_____\\n')
46     graspChecker.resetTopCount()
47     erg = 1
48     break
49
50 else:
51     # move arm
52     if step < 1:
53         # initial wait state
54         print('Initial_Waitstate_done...')
55         rospy.sleep(3.)
56         step = 1
57         erg = 0
58         #break
59
60     elif step >= 1 and step < 2:
61         # wait while arm moves
62         # Rotate axis Nr. 1
63         print('Moving_Arm_to_CheckGrasp_Position...')
64         if grasp_pos == 0:
65             graspChecker.arm.MoveAxisSynchronized(-1200,-850,-1200,-400,900)
66         else:
67             # desk grasp
68             graspChecker.arm.MoveAxisSynchronized(-1500,-370,-1270,40,900)
69         print('..._Arm_in_CheckGrasp_Position!!')
70         #rospy.sleep(10.)
71         step = 2
72         erg = 0
73         #break
74
75     elif (step >= 2 and step < 3):
76         print('Rotation_Axis_5_+/-_10_degree...')
77         if grasp_pos == 0:
78             graspChecker.arm.MoveAxisSynchronized(-1200,-850,-1200,-400,800)
79             rospy.sleep(1.)
80             graspChecker.arm.MoveAxisSynchronized(-1200,-850,-1200,-400,1000)
81         else:
82             # desk grasp
83             graspChecker.arm.MoveAxisSynchronized(-1500,-370,-1270,40,800)
84             rospy.sleep(1.)
85             graspChecker.arm.MoveAxisSynchronized(-1500,-370,-1270,40,1000)
86         print('..._DONE!!')
87         step = 3
88         erg = 0
89         #break
90
91     else:
92         erg = 0
93         print('Algorithm_DONE=>_NO_MATCH!!!')
94         print('_____\\n')
95         break
96
97 else:
98     erg = -1
99     print('Algorithm_terminated!!!')
100     #rospy.signal_shutdown('end')
101     return erg
102
103 class GraspCheck(object):
104     def __init__(self):
105         self.name = ""
106         self.countTop = 0
107         self.countSide = 0
108         #rospy.init_node('checkMarkerName', anonymous=False)
109
110         self.pub = rospy.Publisher('graspPub', String, latch=True)
111         rospy.Subscriber("markername", String, self.checkGrasp)
112
113         print('Initialise_ArmController...')
114         self.arm = McDSA.ArmController()
115         print('Initialisation_DONE!!!')
116
117     def checkGrasp(self, msg):
118         f_pat = msg.data # Namen der gefundenen Patterns
119         top_pat = "grasp_top"
120         side_pat = "grasp_side"
121         ind_top = f_pat.find(top_pat) # Stringindex Topmarker
122         ind_side = f_pat.find(side_pat) # Stringindex Sidemarker
123         #s = 'Topmarker at position ' + str(ind_top) + '; Sidemarker at Position ' + str(ind_side)
124         #print(s)
125         if ind_top >= 0 and ind_side >= 0:
126             # beide Patterns sind sichtbar
127             #print('Beide Patterns gefunden!!!')
128             self.name = 'both'
129             self.countSide += 1
130             self.countTop += 1
131             #print('Top-Counter: %d | Side-Counter: %d' % (self.getTopCount(), self.getSideCount()))
132
133         elif ind_top >= 0 and ind_side < 0:
134             # Pattern auf der Greiferoberflaeche sichtbar
135             #print('Pattern an der Oberflaeche mit Namen %s gefunden' %top_pat)
136             self.name = 'top'
137             self.countTop += 1
138             #print('Top-Counter: %d ' % (self.getTopCount()))

```


A. Programmcode ROS package

```

135
136     elif ind_top < 0 and ind_side >= 0:
137         # Pattern auf der Greiferseite sichtbar
138         #print('Pattern an der Seite mit Namen %s gefunden' %side_pat)
139         self.name = 'side'
140         self.countSide += 1
141         #print('Side-Counter: %d ' %(self.getSideCount()))
142
143     else:
144         #print('Kein Greiferpattern gefunden!!!')
145         self.name = 'none'
146         #print('Top-Counter: %d | Side-Counter: %d' %(self.getTopCount(), self.getSideCount()))
147
148     def run(self):
149         r = rospy.Rate(10)
150         #while not rospy.is_shutdown():
151         #if self.getRotCount() < 1:
152
153         topcount = self.getTopCount()
154         sidecount = self.getSideCount()
155
156         while ((topcount > 1 and topcount < 10) and (sidecount > 1 and sidecount < 10)):
157             self.pub.publish(self.name)
158             r.sleep()
159         #else:
160         #    while (self.getTopCount() < 10 and self.getSideCount() < 10):
161         #        self.pub.publish(self.name)
162         #        r.sleep()
163
164     def resetTopCount(self):
165         self.countTop = 0
166
167     def resetSideCount(self):
168         self.countSide = 0
169
170     def getTopCount(self):
171         return self.countTop
172
173     def getSideCount(self):
174         return self.countSide
175
176     def getRotCountA(self):
177         return self.countRotA
178
179     def getRotCountB(self):
180         return self.countRotB
181
182     def incRotCountA(self):
183         self.countRotA += 1
184
185     def incRotCountB(self):
186         self.countRotB += 1
187
188     def setSideCount(self, value):
189         self.countSide = value
190
191
192
193 # -----
194 def main(args):
195     rospy.init_node('checkMarkerName', anonymous=False)
196     #rospy.init_node('markerService', anonymous=False)
197     ser = rospy.Service('graspDONE', GraspState, startCheckG)
198
199     print
200     print ('Ready to check object grasping ...')
201     rospy.spin()
202
203 if __name__ == "__main__":
204     main(sys.argv)

```

A.2. checkMarkerName.py

```
1 #!/usr/bin/python
2
3 # Author: Christopher Kuenzel
4 # Company: TU Wien
5 # Copyright (C) 2014 Ch. Kuenzel
6
7 PKG = 'check_grasp'
8
9 import roslib;
10 roslib.load_manifest(PKG)
11 import rospy
12 import sys
13 import math
14 import tf
15 import numpy
16 import geometry_msgs.msg
17
18 from std_msgs.msg import String, Header
19
20 def callback(data):
21     rospy.loginfo(rospy.get_name() + ":_Empfange:_%s" % data.data)
22
23 def main(args):
24     rospy.init_node('checkMarkerName', anonymous=False)
25     rospy.Subscriber("markername", String, callback)
26
27     rospy.spin()
28
29
30 if __name__ == "__main__":
31     main(sys.argv)
```

A.3. checkSub.py

```
1 #!/usr/bin/python
2
3 # Author: Christopher Kuenzel
4 # Company: TU Wien
5 # Copyright (C) 2014 Ch. Kuenzel
6
7 PKG = 'check_grasp'
8
9 import roslib;
10 #roslib.load_manifest(PKG)
11 import rospy
12 import sys
13 import math
14 import tf
15 import numpy
16 import geometry_msgs.msg
17
18 from std_msgs.msg import String, Header
19
20 def callback(data):
21     rospy.loginfo(rospy.get_name() + ":_Empfange:_%s" % data.data)
22     #print "OK erhalten..."
23
24
25 def main(args):
26     rospy.init_node('checkSub', anonymous=False)
27     rospy.Subscriber("graspPub", String, callback)
28
29     rospy.spin()
30
31
32 if __name__ == "__main__":
33     main(sys.argv)
```

A.4. startClient.py

```

1  #!/usr/bin/python
2
3  # Author: Christopher Kuenzel
4  # Company: TU Wien
5  # Copyright (C) 2014 Ch. Kuenzel
6
7  PKG = 'check_grasp'
8
9  import roslib
10 import sys
11 import rospy
12 from check_grasp.srv import GraspState
13
14 def startChG(status):
15     rospy.wait_for_service('graspDONE')
16     try:
17         grasp = rospy.ServiceProxy('graspDONE', GraspState)
18         resp = grasp(status)
19         return resp.answer
20     except rospy.ServiceException, e:
21         print('Service call failed: %s' % e)
22
23
24 if __name__ == "__main__":
25     if len(sys.argv) == 2:
26         stat = str(sys.argv[1])
27         #print('datatype input: %s' % type(stat))
28     else:
29         print('%s [%s]' % sys.argv[0], sys.argv[1])
30         sys.exit(1)
31
32     print('\n-----')
33     print('Request: %s [0=OFF | 1=ON]' % stat)
34     temp = startChG(stat)
35     #print('datatype output: %s' % type(temp))
36     print('Answer: %d [0=NO_Match (none_OR_only_side-pattern) | 1=Match]' % temp)
37     print('-----\n')

```

A.5. startup.launch

```

1 <launch>
2
3   <node pkg="ar_marker" type="ar_pose" name="ar_pose" >
4     <param name="debug" type="bool" value="true"/>
5     <param name="frequency" value="15"/>
6     <param name="republish_last" type="bool" value="false"/>
7     <param name="marker_list" value="$(find ar_marker)/data/topkinect_hobbit_vienna_grasp.yml"/>
8     <param name="min_confidence" value="0.7"/>
9     <param name="pose_file_folder" value="/tmp"/>
10    <param name="pose_file_write_freq" value="50"/>
11    <param name="base_frame" value="base"/> <!-- the frame base must also be in the groups -->
12    <param name="groups" value="marker"/>
13    <remap from="image" to="/headcam/rgb/image_color" />
14    <remap from="camera_info" to="/headcam/rgb/camera_info" />
15  </node>
16
17  <node pkg="check_grasp" type="checkgrasp.py" name="GraspOK" >
18
19  </node>
20
21 </launch>

```

- 3D-Druck, 38–41
- Ablaufdiagramm, 84, 89, 90
- ACIN, 23
- AR, ii
- AR-Marker, 19, 29, 61, 65, 67–71, 76, 81, 98, 99
- Attractive Regions, 10
- Bounding-Box-Modell, 12
- CAD, 22, 24, 37, 38, 40, 41
- CAN, 4
- Canny, 98
- Canny-Algorithmus, 43, 44, 52, 60, 61
- CCD, 12, 74
- CPU, 29
- CUDA, 29
- Deskriptor, 15, 16, 25
- DoG, 14, 15, 49
- Features, 14, 16, 17, 27, 28, 49–53
- Flächenberechnung, 24, 26, 27, 43, 59
- GPU, 29
- Hertz-Modell, 12
- HMD, 19
- Hobbit, 2, 4, 5, 7, 28, 33, 36, 39, 43, 54, 58, 60, 73, 76, 78, 79, 81–83, 89, 91, 98
- Hough-Transformation, 17
- k-d-tree, 16, 17
- Kalmanfilter, 10
- Kantenerkennung, 12, 25, 31, 43, 44, 48, 49, 54, 98
- Kollineation, 25
- Kontaktart, 7
- konvexes Rechteckmodell, 12
- Korrelation, 19, 20, 67, 72
- least-squares, 17
- LoG, 43
- nearest neighbour, 16
- Optimierung, 10, 12, 26
- Optischer Fluss, 10, 20, 21, 30, 31, 99
- PCL, 4, 49, 54, 59
- Projektion, 8, 25, 74
- Prototyp, 36, 37, 39, 41, 59
- RGB-D, 4
- ROS, ii, 4, 23, 49, 62, 69, 70, 81, 83, 91, 92, 95
- Service-Call, 81, 82, 89
- Shape Tracking, 10
- SIFT, ii, 14, 17, 27, 28, 43, 49–54, 60, 61, 98
- Trajektorienplanung, 31
- TU, i, 23, 62
- V4R, i, 62
- Visual Servoing, 9, 12, 25, 31

- [1] ACIN - TU Wien. *Hobbit homepage*. URL: `hobbit.acin.tuwien.ac.at/`.
- [2] S. Belongie, J. Malik und J. Puzicha. „Matching shapes“. In: *Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001. Proceedings*. Bd. 1. 2001, 454–461 vol.1. DOI: 10.1109/ICCV.2001.937552.
- [3] A. Bicchi und V. Kumar. „Robotic grasping and contact: a review“. In: *IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA '00*. Bd. 1. 2000, 348–353 vol.1. DOI: 10.1109/ROBOT.2000.844081.
- [4] Bryan Catanzaro u. a. „Efficient, high-quality image contour detection“. In: *2009 IEEE 12th International Conference on Computer Vision*. Sep. 2009, S. 2381–2388. DOI: 10.1109/ICCV.2009.5459410.
- [5] G. Chesi, E. Malis und R. Cipolla. „Automatic segmentation and matching of planar contours for visual servoing“. In: *IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA '00*. Bd. 3. 2000, 2753–2758 vol.3. DOI: 10.1109/ROBOT.2000.846444.
- [6] G. Chesi, E. Malis und R. Cipolla. „Collineation Estimation From Two Unmatched Views of an Unknown Planar Contour for Visual Servoing“. In: 1999, S. 10.
- [7] A. Cirillo, P. Cirillo und S. Pirozzi. „A modular and low-cost artificial skin for robotic applications“. In: *2012 4th IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. Juni 2012, S. 961–966. DOI: 10.1109/BioRob.2012.6290685.
- [8] Ravinder S. Dahiya u. a. „Tactile Sensing - From Humans to Humanoids“. English. In: *IEEE Transactions on Robotics* IEEE Transactions on Robotics. Vol. 26 (Feb. 2010), S. 20.
- [9] T.K. Dey, G. Li und J. Sun. „Normal estimation for point clouds: a comparison study for a Voronoi based method“. In: *Point-Based Graphics, 2005. Eurographics/IEEE VGTC Symposium Proceedings*. Juni 2005, S. 39–46. DOI: 10.1109/PBG.2005.194062.

- [10] Petra Fastermann. *3D-Druck/Rapid Prototyping*. Springer. ISBN: 978-3-642-29224-8. URL: <http://www.springer.com/computer/information+systems+and+applications/book/978-3-642-29224-8>.
- [11] Festo AG & Co. KG. *Adaptiver Greifer DHDG*. Deutsch. 2011.
- [12] D. Fischinger und M. Vincze. „Empty the basket - a shape based learning approach for grasping piles of unknown objects“. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Okt. 2012, S. 2051–2057. DOI: 10.1109/IROS.2012.6386137.
- [13] David Fischinger u. a. „Hobbit, a care robot supporting independent living at home: First prototype and lessons learned“. In: *Robotics and Autonomous Systems* (), S. 19. ISSN: 0921-8890. DOI: 10.1016/j.robot.2014.09.029. URL: <http://www.sciencedirect.com/science/article/pii/S0921889014002140>.
- [14] Qiushi Fu und Marco Santello. „Towards a complete description of grasping kinematics: a framework for quantifying human grasping and manipulation“. eng. In: *Conference proceedings: ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2011 (2011), S. 8247–8250. ISSN: 1557-170X. DOI: 10.1109/IEMBS.2011.6092033.
- [15] K. Harada u. a. „Grasp planning for parallel grippers with flexibility on its grasping surface“. In: *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Dez. 2011, S. 1540–1546. DOI: 10.1109/ROBIO.2011.6181508.
- [16] Hans Havlicek. *Lineare Algebra für Technische Mathematiker: Dritte, korrigierte und erweiterte Auflage*. Deutsch. Auflage: 3., 3. Auflage. Lemgo: Heldermann, N, Okt. 2012. ISBN: 9783885381167.
- [17] Project Team Hobbit u. a. *Hobbit brochure*. English. Nov. 2011. URL: http://hobbit.acin.tuwien.ac.at/brochure/hobbit_brochure.pdf.
- [18] T. Hoshi und H. Shinoda. „Robot skin based on touch-area-sensitive tactile element“. In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*. Mai 2006, S. 3463–3468. DOI: 10.1109/ROBOT.2006.1642231.
- [19] Y. Hu, R. Eagleson und M.A. Goodale. „Human visual servoing for reaching and grasping: the role of 3D geometric features“. In: *1999 IEEE International Conference on Robotics and Automation, 1999. Proceedings*. Bd. 4. 1999, 3209–3216 vol.4. DOI: 10.1109/ROBOT.1999.774087.
- [20] Guo-Shing Huang, Hsiung-Cheng Lin und Po-Cheng Chen. „Robotic arm grasping and placing using edge visual detection system“. In: *Control Conference (ASCC), 2011 8th Asian*. Mai 2011, S. 960–964.
- [21] S. Hutchinson, G.D. Hager und P.I. Corke. „A tutorial on visual servo control“. In: *IEEE Transactions on Robotics and Automation* 12.5 (Okt. 1996), S. 651–670. ISSN: 1042-296X. DOI: 10.1109/70.538972.
- [22] H. Joo u. a. „Graph-based robust shape matching for robotic application“. In: *IEEE International Conference on Robotics and Automation, 2009. ICRA '09*. Mai 2009, S. 1207–1213. DOI: 10.1109/ROBOT.2009.5152594.

- [23] H. Joo u. a. „Graph-based shape matching for deformable objects“. In: *2011 18th IEEE International Conference on Image Processing (ICIP)*. Sep. 2011, S. 2901–2904. DOI: 10.1109/ICIP.2011.6116266.
- [24] Oliver van Kaick u. a. „Contour Correspondence via Ant Colony Optimization“. In: *15th Pacific Conference on Computer Graphics and Applications, 2007. PG '07*. Okt. 2007, S. 271–280. DOI: 10.1109/PG.2007.56.
- [25] H. Kato und M. Billinghurst. „Marker tracking and HMD calibration for a video-based augmented reality conferencing system“. In: *2nd IEEE and ACM International Workshop on Augmented Reality, 1999. (IWAR '99) Proceedings*. 1999, S. 85–94. DOI: 10.1109/IWAR.1999.803809.
- [26] KauPo Plankenhorn e.K. *DRAGON SKIN Serie Datenblatt*. 2013. URL: <http://www.kaupo.de/produkte/silikonkautschuk-additionsvernetzend/dragon-skin-serie/>.
- [27] Michael Kazhdan, Matthew Bolitho und Hugues Hoppe. „Poisson surface reconstruction“. English. In: 2006, S. 61–70. URL: <http://research.microsoft.com/en-us/um/people/hoppe/proj/poissonrecon/>.
- [28] Andreas Kugi und Wolfgang Kemmetmüller. „Prozessidentifikation Vorlesung und Übung“. Deutsch. TU Wien, 2010.
- [29] X. Lamy u. a. „Robotic skin structure and performances for industrial robot manipulation“. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2009. AIM 2009*. Juli 2009, S. 427–432. DOI: 10.1109/AIM.2009.5229975.
- [30] F. Larsson, M. Felsberg und P.-E. Forssen. „Patch Contour Matching by Correlating Fourier Descriptors“. In: *Digital Image Computing: Techniques and Applications, 2009. DICTA '09*. Dez. 2009, S. 40–46. DOI: 10.1109/DICTA.2009.17.
- [31] Peihua Li, F. Chaumette und O. Tahri. „A Shape Tracking Algorithm for Visual Servoing“. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005*. Apr. 2005, S. 2847–2852. DOI: 10.1109/ROBOT.2005.1570545.
- [32] David G. Lowe. „Distinctive Image Features from Scale-Invariant Keypoints“. en. In: *International Journal of Computer Vision* 60.2 (Nov. 2004), S. 91–110. ISSN: 0920-5691, 1573-1405. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: <http://link.springer.com/article/10.1023/B%3AVISI.0000029664.99615.94>.
- [33] Bruce D. Lucas und Takeo Kanade. „An Iterative Image Registration Technique with an Application to Stereo Vision“. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2. IJCAI'81*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981, S. 674–679. URL: <http://dl.acm.org/citation.cfm?id=1623264.1623280>.
- [34] Pablo Musé. „On the definition and recognition of planar shapes in digital images“. English. Diss. 61, avenue du Président Wilson, 94235 CACHAN CEDEX (France): Centre de Mathématiques et de Leurs Applications ENS CACHAN, Okt. 2004.

- [35] A. Namiki und M. Ishikawa. „Optimal grasping using visual and tactile feedback“. In: *IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems, 1996*. Dez. 1996, S. 589–596. DOI: 10.1109/MFI.1996.572234.
- [36] Y. Ohmura, Y. Kuniyoshi und A. Nagakubo. „Conformable and scalable tactile sensor skin for curved surfaces“. In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*. Mai 2006, S. 1348–1353. DOI: 10.1109/ROBOT.2006.1641896.
- [37] Zhicai Ou, Wei Liu und Jianhua Su. „A bilinear model based solution to object pose estimation with monocular vision for grasping“. In: *2011 International Conference on Mechatronics and Automation (ICMA)*. Aug. 2011, S. 501–506. DOI: 10.1109/ICMA.2011.5985613.
- [38] C.B. Owen, Fan Xiao und P. Middlin. „What is the best fiducial?“. In: *Augmented Reality Toolkit, The First IEEE International Workshop*. 2002, 8 pp. DOI: 10.1109/ART.2002.1107021.
- [39] Quasi-invariant Parameterisations, Jun Sato und Roberto Cipolla. „Quasi-Invariant Parameterisations and Matching of Curves in Images“. In: (1998). URL: <http://citeseerx.ist.psu.edu/viewdoc/versions?doi=10.1.1.28.5573>.
- [40] Giorgio Metta Paul Fitzpatrick. „Grounding vision through experimental manipulation.“ In: *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 361.1811 (2003), S. 2165–85. ISSN: 1364-503X. DOI: 10.1098/rsta.2003.1251.
- [41] Rafael A. B. de Queiroz u. a. „Determining Optical Flow using a Modified Horn and Schunck’s Algorithm“. English. In: National Laboratory for Scientific Computing - LNCC Petropolis-RJ, Brazil, S. 4.
- [42] Fabio Remondino. „From Point Cloud to Surface: The Modeling and Visualization Problem“. English. In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* Vol. XXXIV-5/W10 (2003), S. 11. URL: <http://www.isprs.org/proceedings/XXXIV/5-W10/papers/remondin.pdf>.
- [43] P.J. Sanz u. a. „Vision-guided grasping of unknown objects for service robots“. In: *1998 IEEE International Conference on Robotics and Automation, 1998. Proceedings*. Bd. 4. Mai 1998, 3018–3025 vol.4. DOI: 10.1109/ROBOT.1998.680889.
- [44] J. Sato und R. Cipolla. „Affine integral invariants and matching of curves“. In: , *Proceedings of the 13th International Conference on Pattern Recognition, 1996*. Bd. 1. Aug. 1996, 915–919 vol.1. DOI: 10.1109/ICPR.1996.546157.
- [45] V. Shantaram und M. Hanmandlu. „Contour based matching technique for 3D object recognition“. In: *International Conference on Information Technology: Coding and Computing, 2002. Proceedings*. Apr. 2002, S. 274–279. DOI: 10.1109/ITCC.2002.1000400.
- [46] Rainer Steinbrecher. *Bildverarbeitung in der Praxis*. München; Wien: Oldenbourg Wissenschaftsverlag, 1993. ISBN: 9783486223729.

- [47] Jianhua Su, Hong Qiao und Chuankai Liu. „A vision-based 3D grasp planning approach with one single image“. In: *International Conference on Mechatronics and Automation, 2009. ICMA 2009*. Aug. 2009, S. 3281–3286. DOI: 10.1109/ICMA.2009.5246278.
- [48] Richard Szeliski. *Computer Vision - Algorithms and Applications*. English. London: Springer, 2011. ISBN: 978-1-84882-934-3. URL: <http://www.springer.com/computer/image+processing/book/978-1-84882-934-3>.
- [49] Y. Tabuchi u. a. „Development of touch sensor with optical positional sensor“. In: *2004 IEEE Conference on Robotics, Automation and Mechatronics*. Bd. 1. Dez. 2004, 202–207 vol.1. DOI: 10.1109/RAMECH.2004.1438917.
- [50] John Ulmen und Mark Cutkosky. „A robust, low-cost and low-noise artificial skin for human-friendly robots“. In: *2010 IEEE International Conference on Robotics and Automation (ICRA)*. Mai 2010, S. 4836–4841. DOI: 10.1109/ROBOT.2010.5509295.
- [51] N. Vahrenkamp u. a. „Visual servoing for humanoid grasping and manipulation tasks“. In: *8th IEEE-RAS International Conference on Humanoid Robots, 2008. Humanoids 2008*. Dez. 2008, S. 406–412. DOI: 10.1109/ICHR.2008.4755985.
- [52] Markus Vincze und Michael Zillich. „Komponenten der Automation“. Deutsch. TU Wien, 2011.
- [53] B. Wang u. a. „Grasping unknown objects based on 3d model reconstruction“. In: *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. Juli 2005, S. 461–466. DOI: 10.1109/AIM.2005.1511025.
- [54] W. Wohlking und M. Vincze. „Ensemble of shape functions for 3D object classification“. In: *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Dez. 2011, S. 2987–2992. DOI: 10.1109/ROBIO.2011.6181760.
- [55] W. Wohlking und M. Vincze. „Shape-based depth image to 3D model matching and classification with inter-view similarity“. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sep. 2011, S. 4865–4870. DOI: 10.1109/IROS.2011.6094808.
- [56] W. Wohlking und M. Vincze. „Shape distributions on voxel surfaces for 3D object classification from depth images“. In: *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. Nov. 2011, S. 115–120. DOI: 10.1109/ICSIPA.2011.6144153.