Technik
Informatik & Medien
Hochschule Ulm
University of
Applied Sciences

Erkennen und Greifen von Alltagsgegenständen mittels Katana-Manipulatorarm:

Umgebungserkennung mittels 3D-Entfernungsdaten

Bachelorarbeit an der
Hochschule Ulm
Fakultät Informatik
Studiengang Technische Informatik

vorgelegt von
Manuel Wopfner

August 2009

1. Gutachter: Prof. Dr. Christian Schlegel
2. Gutachter: Prof. Dr. Rüdiger Lunde

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst und keine anderen als die im Literaturverzeichnis angegebenen Hilfsmittel verwendet habe. Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht und mit genauer Quellenangabe dargelegt habe.

Ulm, den 27. August 2009 Manuel Wopfner

Zusammenfassung

In dieser Arbeit wird ein einfaches System zur Objekterkennung vorgestellt. Die dreidimensionale Umgebungserfassung erfolgt mit Hilfe eines am Katana-Arm montierten Laserscanners. In der damit erstellten Punktwolke wird zuerst die Tischplatte, sowie alle Objekte, die sich nicht im Aktionsradius des Katana-Arm befinden, entfernt. Anschließend wird die verbleibende Punktwolke in Segmente aufgeteilt, die je ein Objekt repräsentieren. Anhand der aus den Segmenten extrahierten Merkmale findet die Klassifizierung der Objekte statt, indem diese mit den in der Objektdatenbank enthaltenen Objekte verglichen werden. Es wird gezeigt, dass dieses Verfahren eine ausreichende Klassifizierungssicherheit für primitive Alltagsgegenstände bietet.
Zusammen mit der Arbeit von J. Brich [Bri09], die das kollisionsfreie Manipulieren von Alltagsgegenständen zum Inhalt hat, wird ein Verfahren zur Mobilen Manipulation vorgestellt, welches einfache Alltagsprobleme lösen kann.

Danksagung

Bedanken möchte ich mich bei Herrn Prof. Dr. C. Schlegel für die hilfreichen Hinweise, sowie für die Diskussionen, die mir bei meiner Arbeit halfen. Gleichzeitig möchte ich mich bei Herrn M. Sc. S Hochdorfer bedanken, der mir bei allen Fragen mit Rat und Tat zur Seite stand.
Weiter möchte ich mich bei Jonas Brich für die Zusammenarbeit während des Studiums und dieser Arbeit bedanken. Die Diskussionen waren immer anregend und haben geholfen, so manches Problem zu lösen.
Zu guter Letzt möchte ich denjenigen danken, die mich durch Korrekturlesen oder andersweitig bei dieser Arbeit unterstützt haben.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Der Bereich der Servicerobotik gewinnt immer mehr an Bedeutung. Mittlerweile gibt es mehrere Projekte, die dieses Thema behandeln. Dabei ist es das Ziel, den Menschen durch das Lösen von Alltagsproblemen, wie das Decken und Abräumen eines Tisches, zu unterstützen. So gibt es z. B. mit der *Rollin' Justin* Platform vom DLR [BWS+09] ein System, mit dem grundlegende Aufgaben im Bereich der Servicerobotik demonstriert werden können. Die Lösung von Alltagsproblemen ist aber weiterhin ein großes Forschungsgebiet. Speziell im Bereich der *Mobilen Manipulation*, bei der es um das Greifen und Bewegen von Objekten geht, gibt es noch Forschungsbedarf. Ein Grund dafür ist, dass sich im Gegensatz zur Industrierobotik, die Umgebung ständig verändert. Des Weiteren befinden sich im Aktionsraum des Roboters Menschen, die durch dessen Arbeit nicht gefährdet werden dürfen.

In Abbildung 1.1 findet sich ein Überblick über die Komponenten eines Systems zur *Mobilen Manipulation*. Das System besteht aus mehreren Komponenten, die es dem Roboter zum einem ermöglichen, sich im Raum zu orientieren und zu bewegen. Zum anderen gibt es die *Objekterkennung* sowie die *Bahnplanung* des Manipulators, die beide bei der Mobilen Manipulation eine große Rolle spielen.

Durch die Montage des Greifarms (Manipulator) auf einer mobilen Roboterplattform können an allen Orten, die der Roboter erreicht, Manipulationsaufgaben durchgeführen werden. Die Umwelt wird mittels mehrerer Sensoren, meist Kameras oder Lasersensoren, die einzeln oder auch zusammen verwendet werden können, erfasst. Anhand dieser Daten erfolgt die Objekterkennung, welche die Teilbereiche *Hintergrund entfernen*, *Objekte finden* und *Objekte klassifizieren* umfasst. Aufbauend auf diesen Daten erfolgt die Bahnplanung für das kollisionsfreie Greifen der Objekte. Falls eine kollisionsfreie Bahn gefunden wurde, wird diese ausgeführt.

Ziel dieser Arbeit ist die Entwicklung eines Systems zur *Objekterkennung*, mit dem einfache Objekte aus dem Alltagsleben erkannt werden können. Unter einfachen Objekten versteht man dabei Objekte, die durch einen Zylinder oder Quader angenähert werden können. Zu solchen Objekten gehören z. B. Becher, Schüsseln, Flaschen, Dosen und Schachteln. Es soll gezeigt werden, dass durch ein einfaches Verfahren ein System realisierbar ist, welches Alltagsprobleme lösen kann. Für die Umgebungserfassung wird ein 2D-Laserscanner verwendet, der an einem Roboterarm montiert ist und somit ein dreidimensionales Scannen der Umgebung ermöglicht. Mittels dieser Daten findet anschließend die Objekterkennung in mehreren Schritten statt.

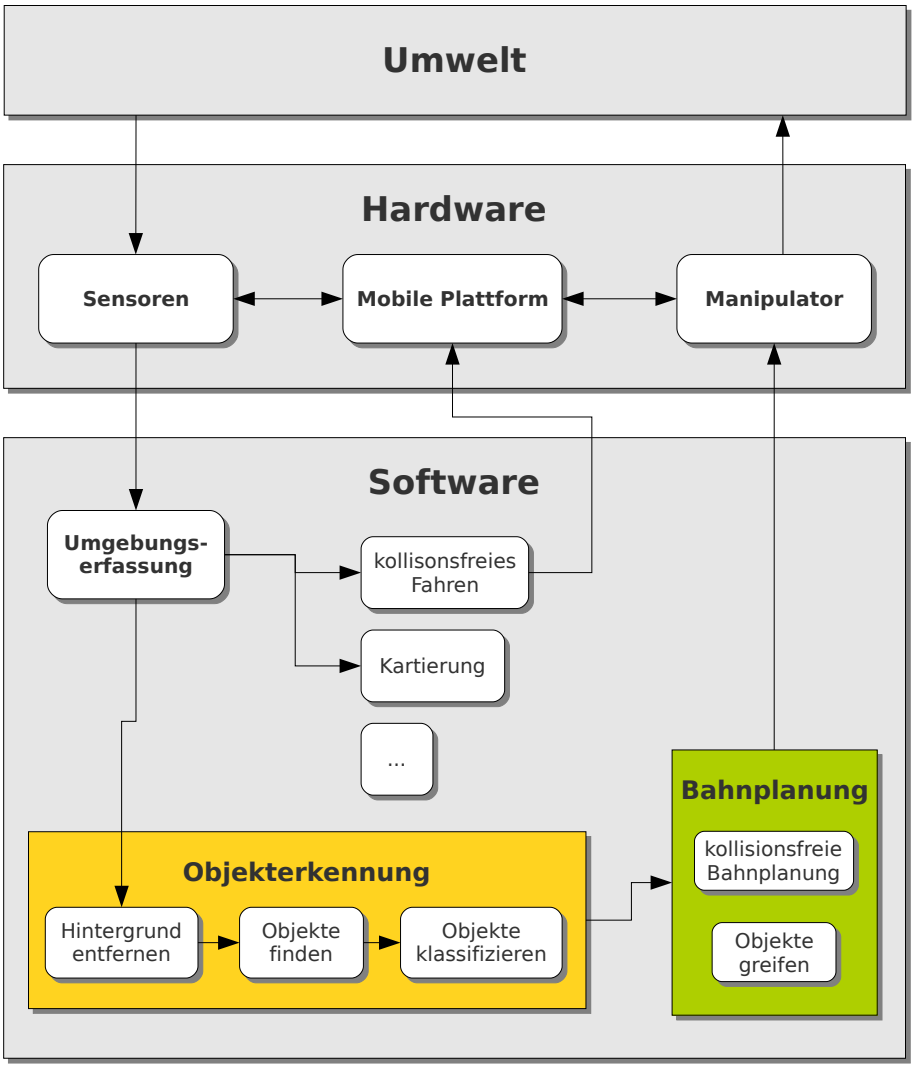


Abbildung 1.1: Komponenten eines Systems zur *Mobilen Manipulation*.

Kapitel 2

Stand der Technik

2.1 Mobile Manipulation

Unter *Mobiler Manipulation* versteht man das Durchführen von Handhabungsaufgaben mittels eines Manipulators, der auf einer mobilen Roboterplattform montiert ist. Dadurch können an beliebigen Orten, die der Roboter erreicht, Handhabungsaufgaben durchgeführt werden. Mittlerweile gibt es unterschiedliche Projekte z. B. vom *DLR*[1] und *Fraunhofer IPA*[2], die das Thema der *Mobilen Manipulation* zum Inhalt haben.

Abbildung 2.1: DLR's *Rollin' Justin.* [BWS+09]

Der Roboter *Rollin' Justin* vom DLR [BWS+09] ist eine fortschrittliche Roboterplattform. In seinem Kopf ist eine Stereokamera sowie ein Laserscanner untergebracht. Durch die Kombination dieser Sensoren ist es möglich, ein exaktes Umgebungsmodell zu erstellen. Dieses System markiert den Stand der Technik und zeigt, was im Bereich der *Mobilen Manipulation* grundsätzlich möglich ist. Es

[1] Deutsches Zentrum für Luft- und Raumfahrt
[2] Fraunhofer-Institut für Produktionstechnik und Automatisierung

ist jedoch komplex und teuer und wird deshalb nicht in naher Zukunft im alltäglichen Einsatz Verwendung finden. Aus diesem Grund wird in dieser Arbeit die Realisierung eines einfacheren Systems angestrebt.

In [Pau08] wird solch ein einfacheres System vorgestellt. Dabei kommt ein Katana-Arm für die Umgebungsmanipulation zum Einsatz. Die Umgebungserfassung erfolgt mittels einer Stereokamera, wodurch die Position des zu greifenden Objektes bestimmt werden kann. Der Roboter, siehe Abbildung 2.2, wird vom Team *b-it-bots* beim *RoboCup@Home* verwendet. Dort wurde gezeigt, dass diese Konfiguration für das Lösen von einfachen Alltagsproblemen ausreichend ist.
Der Katana-Manipulator ist kostengünstig und für den Einsatz im Umfeld von Menschen zugelassen, weshalb er sich auch für diese Arbeit anbietet. Die durch die Stereokamera ermittelten Entfernungswerte sind für die kollisionsfreie Bahnplanung in Szenarien mit mehreren Objekten jedoch zu ungenau, weswegen in dieser Arbeit keine Stereokamera verwendet wird.

Abbildung 2.2: *Johnny Jackanapes des Teames b-it-bots.* [Pau08]

2.2 Hardware

2.2.1 Laserscanner

Laserscanner finden heute im Bereich der Robotik einen breiten Einsatz. Hier werden sie vor allem zur Objektverfolgung, zum Erstellen von 2D-Umgebungskarten, zum kollisionsfreien Fahren und zur Selbstlokalisierung verwendet. Zunehmend werden sie aber auch bei der Erstellung von 3D-Umgebungskarten eingesetzt. Dafür wird entweder ein 3D-Laserscanner verwendet oder es wird ein 2D-Laserscanner so am Roboter angebracht, dass dieser schräg auf den Boden blickt. Während sich der Roboter bewegt, kann dadurch eine 3D-Karte erstellt werden. Eine weitere Lösung ist es, einen Laserscanner auf eine Pan-Tilt Einheit, welche das Schwenken und Neigen des Laserscanners ermöglicht, zu montieren und die Umgebung abzutasten.

Laserscanner bieten, im Vergleich zu anderen Sensoren wie beispielsweise Kameras, den Vorteil, dass sie eine genaue Entfernungsangabe des Objektes liefern. Es gibt die Möglichkeit, die Entfernung mittels Stereokameras zu ermitteln, die Entfernung ist jedoch nur in einem Bereich von wenigen Metern einigermaßen präzise. Eine Objekterkennung anhand dieser Entferungsdaten ist jedoch wegen ihrer Ungenauigkeit nicht möglich. Seit einiger Zeit gibt es auch PMD[3]-Kameras, die mit einer Aufnahme ein 3D-Bild der Umgebung aufnehmen können. Dabei wird die Szene durch Lichtimpulse ausgeleuchtet und anhand der Lichtlaufzeit wird die Entfernung des Objektes ermittelt. Die Kameras haben jedoch den Nachteil, dass sie einen kleinen Öffnungswinkel von $30°$ bis $40°$ haben und dass ihre Genauigkeit noch nicht an die eines Laserscanners heranreicht.

Der Aufbau eines Laserscanners ist in Abbildung 2.3 veranschaulicht. Die meisten Laserscanner funktionieren nach diesem oder einem ähnlichen Prinzip. Dabei wird von einer Laserdiode ein Laserstrahl auf einen rotierenden Spiegel geschickt. Dieser Strahl wird von einem getroffenen Objekt reflektiert und das reflektierte Licht wird von einer Fotodiode detektiert.

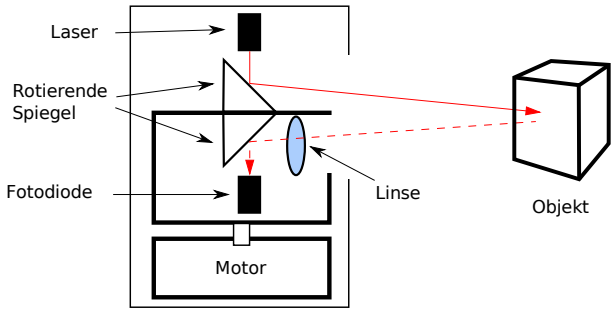


Abbildung 2.3: Schematische Zeichnung eines Laserscanners. [OYB09]

Bei Laserscannern unterscheidet man zwischen zwei Arten der Entfernungsmessung. Beim *Impulslaufzeitmessverfahren* wird der Laserstrahl in kurzen Impulsen verschickt und das reflektierte Licht mittels eines Sensors detektiert. Anhand der benötigten Zeit wird die Entfernung des Objektes ermittelt. Eine Alternative ist das *Phasendifferenzverfahren*. Hierbei wird ein kontinuierlicher Laserstrahl ausgesandt, dessen Amplitude mit mehreren sinusförmigen Wellen unterschiedlicher Wellenlänge moduliert wird. Anhand der Phasendifferenz, die man durch das gleichzeitige Betrachten der Phasenlage des gesendeten und des empfangenen Signals erhält, kann man den Objektabstand ermitteln. Dieses Verfahren wird in allen modernen Laserscannern verwendet.

Die bekanntesten Vertreter von Laserscannern sind in der Robotik die Produkte von *Sick* und *Hokuyo*. Dabei stellt *Sick* hauptsächlich große, robuste Laserscanner, die auch für den Outdoor-Bereich geeignet sind her. *Hokuyo* hingegen produziert kleinere Scanner für den Indoor-Bereich.

Sick Die Laserscanner von *Sick* galten viele Jahre lang als Standard im Bereich der Robotik. Vor allem die LMS200 Serie wird in vielen Projekten verwendet. Aufgrund seiner Leistungsmerkmale ist der Laserscanner für viele Bereiche in der Robotik einsetzbar.

Hokuyo Die Laserscanner von *Hokuyo* sind kleiner und günstiger als die von *Sick*. Durch die geringeren Abmessungen können auch mehrere dieser Geräte auf kleineren Roboterplattformen, wie sie in der Servicerobotik üblich sind, montiert werden. Dabei ist die Leistung der Scanner in den meisten Fällen ausreichend.

[3] **P**hotonic **M**ixer **D**evice

(a) Sick LMS200. [HP-09a] (b) Hokuyo URG-04X. [HP-09b]

Abbildung 2.4: Unterschiedliche Laserscanner.

Tabelle 2.1 stellt zwei der meist verwendeten Laserscanner der Firmen *Sick* und *Hokuyo* mit ihren Leistungsdaten gegenüber. Der *LMS200* von *Sick* hat eine größere Reichweite und eine höhere Scanfrequenz als der *URG-04X*, er ist jedoch auch um ein Vielfaches größer als dieser.
Aufgrund der kleineren Abmessungen und der ausreichenden Leistung bietet sich für diese Arbeit der *URG-04X* von *Hokuyo* an.

	Sick LMS200	Hokuyo URG-04X
Öffnungswinkel:	$180°$	$240°$
Scanfrequenz:	$75Hz$	$10Hz$
Arbeitsbereich:	$0m \ldots 80m$	$0{,}06m \ldots 5{,}6m$
Winkelauflösung:	$0{,}25°; 0{,}5°; 1°$	$0{,}36°$
Fehler:	$\pm 15mm$	$0{,}06m \ldots 1m$: $\pm 10mm$ $1m \ldots 5{,}6m$: $\pm 1\%$
Betriebsspannung:	$\leq 24VDC, \pm 15\%$	$5VDC \pm 5\%$
Leistungsaufnahme:	$20W$	$2{,}5W$
Gewicht:	$4{,}5kg$	$0{,}16kg$
Abmessungen:	$156mm \times 155mm \times 210mm$	$50mm \times 50mm \times 70mm$

Tabelle 2.1: Leistungsmerkmale von Sick LMS200 und Hokuyo URG-04X.

2.2.2 Manipulatoren

Mittlerweile gibt es viele unterschiedliche Manipulatoren, die im Bereich der Robotik zum Einsatz kommen. Die Herausforderung für Manipulatoren in der Servicerobotik ist, dass sie im Umfeld von Menschen betrieben werden. Daher muss gewährleistet sein, dass sie bei einer Kollision mit Objekten und vor allem mit Menschen diese nicht verletzen. Aus diesem Grund sind Manipulatoren aus der Industrierobotik meist nicht für die Servicerobotik geeignet.
Ein bekannter Manipulator, vor allem auch im Bereich des *RoboCup@Home*, ist der Katana-Arm der Firma *Neuronics*, den es mittlerweile in unterschiedlichen Ausführungen gibt. Dieser Manipulator ist kostengünstig und für den Einsatz im Umfeld von Menschen zugelassen. Aus diesem Grund kommt er in dieser Arbeit zum Einsatz. Eine ausführlichere Beschreibung des Katana-Manipulators findet sich in [Bri09].

Abbildung 2.5: Katana-Manipulator.

2.3 3D-Objekterkennung

Für die Objekterkennung in der realen Welt bietet sich eine dreidimensionale Repräsentation des Objektes und der Umgebung an. Durch sie ist es nicht nur möglich, das Objekt zu klassifizieren, sondern auch, seine genaue Lage im Raum zu bestimmen. Diese Positionsangabe ist für die Mobile Manipulation wichtig, da die Art des Objektes, wenn es gegriffen werden soll, ohne seine Position nutzlos ist. Besonders Laserscanner bieten sich für die Erstellung des 3D-Modells an, da sie exakte Entfernungsdaten liefern.

2.3.1 Objekterkennung in 3D-Entfernungsdaten

Die *Objekterkennung in 3D-Entfernungsdaten* ist schon seit den 90iger Jahren ein Thema. In diese Zeit fallen die bekannten *Spin Images* von Andrew E. Johnson, die er in seiner Doktorarbeit [Joh97] vorstellte. Ein anderer Ansatz [SM92] beruht auf 3D-Kurven und Splashes, mit denen die Form von 3D-Objekten beschrieben wird. In den darauffolgenden Jahren finden sich weniger Publikationen zu diesem Thema. Mittlerweile ist das Thema wieder aktuell. Dies wird auch durch die Publikationen auf der ICRA[4] in den letzten zwei Jahren deutlich. Es scheint sich aber noch kein Verfahren bei der Analyse von 3D-Entfernungsdaten als Standard zu etablieren. Es zeichnet sich jedoch eine Tendenz ab, bei der die Objekte über die Beziehung der einzelnen Oberflächenpunkte zueinander beschrieben werden.

Bei der Analyse von Entfernungsdaten stellen sich dabei folgende Probleme:

1. Zerlegung der Punktwolke in relevante Teile. (Segmentierung)

2. Lernen von Objekten.

3. Erkennen von gelernten Objekten in einer Szene. (Objekterkennung)

[4] International Conference on Robotics and Automation

Nachfolgend wird ein Einblick in aktuelle Publikationen gegeben. Dadurch soll ein Eindruck der aktuellen Forschung und Ansätze zur Lösung der unterschiedlichen Probleme vermittelt werden. Die Analyse von 3D-Entfernungsdaten wird sicher ein Forschungsthema für die nächsten Jahre bleiben.

2.3.1.1 Segmentierung

In [KWB09] wird ein Verfahren zur Segmentierung von kontinuierlich gelieferten Entfernungsdaten beschrieben. Dabei wird ein verfeinertes RBNN Verfahren [KWB08] verwendet, welches mit einem Datenstrom umgehen kann. Zur Bestimmung der nächsten Nachbarn zu einem Punkt p wird die Tatsache herangezogen, dass die nacheinander eintreffenden Daten einen räumlichen Bezug zueinander haben. Die Suche nach dem nächsten Nachbarn beschränkt sich somit auf eine gewisse Anzahl an Datensätzen, vor und nach dem aktuellen Scan, je nachdem wie groß die Nachbarschaft gewählt ist. Die gefundenen Nachbarn werden zur Berechnung des Normalenvektors des Punktes p verwendet, anhand derer die Segmentierung durchgeführt wird. Kriterien für die Segmentierung sind dabei der Winkel, sowie der Abstand zweier Punkte p und q.
Dieses Verfahren ist für kontinuierlich gelieferte Entfernungsdaten, wie bei einem Laserscanner, der durch die Bewegung des Roboters die Umgebung in 3D erfasst, geeignet. Dabei entstehen große Datenmengen, aus denen schon während des Scanvorgangs nützliche Daten gewonnen werden können.
In der gegebenen Aufgabenstellung, bei der nur eine kleine Datenmenge während eines abgeschlossenen Scanvorgangs anfällt, ist dieses Verfahren nicht notwendig. Die Segmentierung erfolgt deshalb in dieser Arbeit nach dem Scan mittels einer statischen Datenstruktur.

2.3.1.2 Lernen von Objekten

Ein Verfahren zum Lernen von Objekten aus 3D-Punktwolken wird in [RSGB09] vorgestellt. Es ermöglicht das gleichzeitige Lernen von mehreren unterschiedlichen Objekten aus einer Szene. Dabei werden auch unterschiedliche perspektivische Ansichten des gleichen Objektes in einer Szene berücksichtigt. Das Verfahren entfernt im ersten Schritt den Hintergrund aus der Punktwolke. Dafür wird zuerst die Normale jedes Punktes mittels seiner Nachbarpunkte berechnet. Danach werden die Punkte zusammengefasst, die ähnliche Normalen besitzen. Waagrechte und senkrechte Ebenen, die Boden und Wände repräsentieren, werden entfernt. Die verbleibenden Punkte werden bezüglich ihrer Nachbarschaft gruppiert. Das Ergebnis ist eine Menge von Punktwolken, die jeweils ein Objekt aus einer bestimmten perspektivischen Sicht zeigen.
Um festzustellen, ob zwei Objekte miteinander fusioniert werden können, werden zuerst die möglichen Ausrichtungen berechnet. Hierfür wird ein auf Entfernungsbildern basierender Ansatz verwendet. Entfernungsbilder sind kompakte Repräsentationen von 3D-Daten. Der Wert eines Pixels ist die Länge des Strahls, der vom Beobachter ausgeht, das Pixel passiert und das nächstgelegenen Objekt in der Szene schneidet. Anhand dieser Bilder werden für jedes Modellpaar die möglichen Transformationen berechnet. Diese Transformationen werden mit einer Wertung versehen. Umso höher die Wertung ist, desto besser ist die Übereinstimmung. Die Wertungen aller Paare werden anschließend in eine Ähnlichkeitsmatrix eingetragen. Anhand dieser Matrix wird bestimmt, welche Modelle miteinander fusioniert werden. Nach der Fusionierung werden die Transformationen und Wertungen für die noch verbleibenden, nicht fusionierten Modelle neu berechnet.
Das Verfahren ist zum Lernen von komplexen Objekten geeignet. Für primitive Objekte, wie es in dieser Arbeit der Fall ist, kann ein einfacheres Verfahren verwendet werden. Die Hintergrundentfernung durch das Hineinlegen von unterschiedlichen Ebenen ist für das gestellte Problem anwendbar, da die Objekte auf einem Tisch stehen, welcher eine Ebene darstellt.

2.3.1.3 Objekterkennung

Die bekannten *Spin Images* wurden in [Joh97] vorgestellt. Eine kürzere Zusammenfassung des Themas findet sich in [JH99]. Bei *Spin Images* werden zuerst für alle Punkte die dazugehörenden Normalenvektoren mit Hilfe der Nachbarpunkte berechnet. Danach wird für jeden Punkt des Objektes ein Spin Image generiert. Hierbei werden alle Punkte auf eine 2D-Ebene abgebildet, indem man die Ebene um den Nomalenvektor des Punktes rotiert. Das Aussehen des Bildes kann mittels zweier Parameter beeinflusst werden. Die unterstützte Distanz D_s gibt an, wie weit ein Punkt entfernt sein darf, um noch ins Bild aufgenommen zu werden. Der unterstützte Winkel A_s spezifiziert, wie groß der Winkel zwischen Normalenvektor des Punktes und dem Normalenvektor des *Spin Images* maximal sein darf, damit er noch aufgenommen wird. Mithilfe dieser beiden Parameter kann der Einfluss von Rauschen und Verdeckung verringert werden. Um die große Datenmenge zu reduzieren wird ein PCA[5] Komprimierungsverfahren eingesetzt. Die aus der Komprimierung resultierenden Bilder werden *eigen-spin-images* genannt, da sie auf der Berechnung der Eigenvektoren und Eigenwerten basieren. Zum Erkennen der Objekte werden zufällig unterschiedliche Punkte der Szene ausgewählt und die dazu gehörenden *eigen-spin-images* berechnet. Diese werden anschließend mit den in der Datenbank hinterlegten *eigen-spin-images* verglichen und der beste Treffer ermittelt.
Dieses Verfahren ist komplex und das Erstellen und Modifizieren der Objektdatenbank sehr rechenaufwändig, da zum einen für jeden Punkt des Objektes ein *eigen-spin-image* berechnet werden muss, zum anderen danach das PCA Verfahren angewandt wird. Das Verfahren scheint trotz seiner Komplexität Anwendung zu finden, da die neue OpenCV[6] Version 2.0 eine Spin Image Implementierung enthalten wird.
Wegen seiner Komplexität ist das Verfahren für das gestellte Problem nicht geeignet. Stattdessen wird auf ein einfacheres Verfahren zurückgegriffen, bei dem das Erstellen der Datenbank, sowie die Suche in ihr effizienter ist.

Auf Grund der zeitintensiven Berechnung der *eigen-spin-images* und ihrem hohen Speicherverbrauch, wird in [ABBP07] ein alternatives Verfahren genannt, *Spin image signatures* zur Komprimierung von *Spin Images* vorgestellt. Dabei werden die *Spin Images* gleich berechnet, die Komprimierung erfolgt jedoch über Signaturen, die als n-dimensionaler Vektor definiert sind. Bei der Signatur handelt es sich um eine regionenbasierte Beschreibung der *Spin Images*. In der beschriebenen Lösung wurde das *Spin Image* in drei Regionen (siehe Abbildung 2.6) unterteilt. Die erstellten Signaturen werden

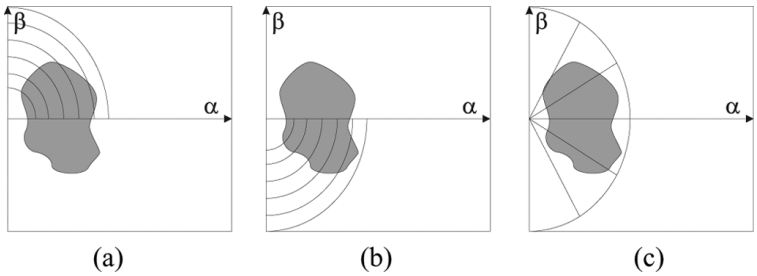


Abbildung 2.6: Spin image signatures: (a) Bereiche in der halben Ebene $\beta > 0$, (b) Bereiche in der halben Ebene $\beta < 0$, und (c) Sektoren. [ABBP07]

danach zu Clustern zusammengefasst und das Zentrum jedes Clusters wird in der Datenbank hinterlegt. Durch das Clustering erhält man die Signaturen, die das Objekt am besten beschreiben. Es wurde

[5] Principal component analysis
[6] Eine C++ Bibliothek für die Bildverarbeitung (`http://opencv.willowgarage.com/wiki/`)

gezeigt, dass die Zeit für die Objekterkennung sowie der Speicherbedarf um ein Vielfaches geringer ist als bei der Komprimierung mit PCA. Aufgrund dieser Tatsache kann dieses Verfahren auch auf große Objektdatenbanken angewandt werden.
Die Berechnungskomplexität der *Spin Images* bleibt jedoch bestehen, was gegen eine Verwendung dieses Verfahrens für die gestellte Aufgabe spricht. Die Repräsentation der Objektmerkmale durch einen n-dimensionalen Vektor in einem Merkmalsraum bietet sich jedoch an, da dies eine einfache und effiziente Art der Objekterkennung ermöglicht.

Die Technische Universität München stellt in [RMBB08b] ein Verfahren vor, welches für jeden Punkt ein Histogramm berechnet. Hierfür wird der Normalenvektor eines Punktes mit Hilfe seiner Nachbarpunkte berechnet. Anschließend wird dieser Normalenvektor mit den Normalenvektoren seiner Nachbarpunkte in Beziehung gesetzt. [RMBB08a] zeigt, wie mittels PFH[7] unterschiedliche Scans ausgerichtet werden können. Dabei werden in jedem Scan markante Punkte ermittelt und diese im Anschluss in Übereinstimmung gebracht. In [RBB09] wird eine weiterentwickelte Version der PFH, genannt FPFH[8], bei der die Berechnungszeit wesentlich verkürzt wurde, vorgestellt.
Dieses Verfahren scheint für unterschiedliche Probleme geeignet zu sein. Es wurde jedoch bisher keine Arbeit publiziert, welche die Objekterkennung mithilfe von PFH durchführt. Aus diesem Grund findet dieses Verfahren keine weitere Betrachtung, es zeigt jedoch eine aktuelle Forschungsrichtung auf.

In [HLLS01] wird ein Verfahren vorgestellt, das die Objekterkennung anhand recht einfacher Merkmale vornimmt und diese in einem n-dimensionalen Histogramm speichert. Das Verfahren beruht dabei wie [RSGB09] auf Entfernungsbildern. Anhand dieser Bilder werden die drei Merkmale *Pixel Tiefe*, *Oberflächennormalen* und *Krümmung* berechnet. Bei der *Pixel Tiefe* handelt es sich um den Intensitätswert des Pixels, welcher bei Entfernungsbildern mit der Entfernung des Objektes zusammenhängt. Dieses Merkmal ist sehr einfach zu berechnen, leidet jedoch an Aussagekraft, wenn die Werte durch andere Objekte, die sich im Bild befinden, beeinflusst werden. Die *Oberflächennormalen* können über die erste Ableitung des Bildes ermittelt werden. Da für das Histogramm eine zweidimensionale Repräsentation nötig ist, wird der Normalenvektor durch das Winkelpaar (ϕ, θ) in Kugelkoordinaten, siehe Abbildung 2.7, beschrieben. Die *Krümmung* kann entweder direkt anhand

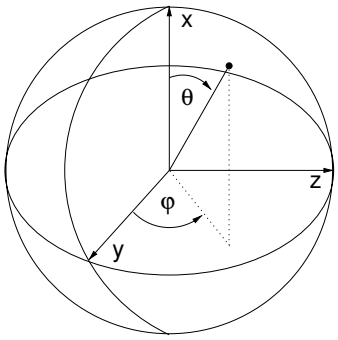


Abbildung 2.7: Normalenvektor beschrieben in Kugelkoordinaten. [HLLS01]

der ersten und zweiten Ableitung oder indirekt über die Änderung der Normalenvektoren berechnet werden. Für die Darstellung wurde die *shape index* Form, die von Koenderik [KD92] vorgestellt und

[7] Point Feature Histograms
[8] Fast Point Feature Histograms

von Dorai [DJ97] und Jain [JTLB04] modifiziert wurde, verwendet. Diese Repräsentation bewertet die Hauptkrümmung im Umfeld eines Punktes p und hat den Wertebereich $[0,1]$. Sie hat im Vergleich zur normalerweise verwendeten Gaußschen Krümmung K und der mittleren Krümmung H eine höhere Aussagekraft.
Die Objekterkennung erfolgt anhand eines statistischen Verfahrens, dem das Bayestheorem zu Grunde liegt. Ein statisches Verfahren hat den Vorteil, dass eine Aussage über die Qualität der Erkennung getroffen werden kann. Zudem können weitere Merkmale wie Farbinformationen berücksichtigt werden.
Die in der Arbeit vorgestellten Merkmale, die hauptsächlich auf den Normalenvektoren beruhen, sind wegen des Aufwandes zur Berechnung der Normalen in Punktwolken für diese Arbeit nicht geeignet. Die Objekterkennung mittels eines n-dimensionlen Vektors und eines statistischen Verfahrens bietet sich jedoch an, da dadurch weitere Merkmale einfach hinzugefügt werden können.

Kapitel 3

Grundlagen

3.1 RANSAC

Der RANSAC (**Ra**ndom **Sa**mple **C**onsensus)-Algorithmus ist für Messreihen gedacht, die mit Ausreißern und groben Fehlern behaftet sind. Dabei unterstützt er Verfahren, wie die *Methode der kleinsten Quadrate*, welche nur schlecht mit vielen Ausreißern umgehen können. RANSAC stellt eine von Ausreißern gesäuberte Datenbasis, das sogenannte *Consensus Set*, bereit. Der Algorithmus geht wie folgt vor:

1. Wähle zufällig aus der Messreihe so viele Punkte aus, um das Modell bestimmen zu können. (Drei Punkte für eine Ebene)
2. Ermittle mit Hilfe der gewählten Punkte die Modellparameter.
3. Bestimme die Teilmenge (*Consensus Set*) der Punkte, deren Abstand d zum Modell kleiner als eine definierte Fehlerschranke d_{max} ist. Wenn das *Consensus Set* eine gewisse Anzahl an Punkten beinhaltet, werden anhand dieser Punkte z. B. mit der *Methode der kleinsten Quadrate* die entgültigen Modellparameter bestimmt.
4. Falls das *Consensus Set* nicht genügend Punkte enthält, werden die Schritte 1-3 solange durchgeführt, bis ein passendes *Consensus Set* gefunden wurde oder die maximale Anzahl an Iterationen erreicht wurde. Es wird dann entweder das *Consensus Set* mit den meisten Punkten verwendet oder eine Exception geworfen.

3.2 Clustering

Die Anwendung eines Cluster-Verfahrens auf eine Punktmenge ist eine Möglichkeit, diese zu segmentieren. Es gibt viele Cluster-Verfahren für die unterschiedlichsten Problemstellungen. Einen Überblick und eine Kategorisierung einiger populärer Algorithmen findet sich in [HKT01]. Abbildung 3.1 zeigt die Einteilung der unterschiedlichen Verfahren mit den prominentesten Vertretern jeder Kategorie.

partitionierend *Partitionierende* Cluster-Verfahren teilen eine Menge von n Objekten in einem d-dimensionalen Raum in k Cluster auf und zwar so, dass die Abweichung des Objektes von seinem Clusterzentrum oder einer Clusterverteilung minimiert wird. Der Abstand kann dabei auf unterschiedliche Arten berechnet werden und wird allgemein als *Ähnlichkeitsfunktion* bezeichnet.

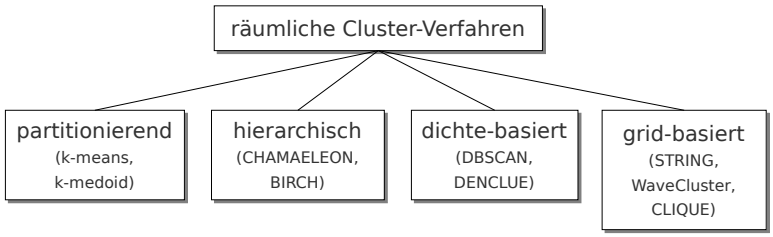


Abbildung 3.1: In Kategorien eingeteilte Cluster-Verfahren.

hierarchisch *Hierarchische* Verfahren teilen die gegebene Objektmenge in ein Dendrogramm[1] auf. Es wird zwischen zwei Verfahren, dem *bottom-up* und dem *top-down* Verfahren, unterschieden. Beim *bottom-up* Verfahren, auch *agglomeratives* Verfahren genannt, bildet jedes Objekt zu Beginn einen eigenen Cluster. Die Cluster werden nach und nach fusioniert bis ein Cluster entsteht oder ein Abbruchkriterium erreicht wird. Beim *top-down*, auch *divisives* Verfahren genannt, bilden alle Objekte zunächst ein Cluster, das in jeder Iteration in kleinere Cluster geteilt wird, bis ein Abbruchkriterium erreicht wird oder jedes Objekt einen eigenen Cluster bildet.

dichte-basiert *Dichte-basierte* Verfahren bieten den Vorteil, dass sie nicht, wie die meisten partitionierenden Verfahren, nur kreisförmige, sondern beliebig geformte Cluster bilden können. Zudem können sie mit Daten umgehen, die mit Rauschen behaftet sind. Sie fassen Cluster als Regionen mit einer hohen Objektdichte auf, die durch Regionen mit niedriger Dichte getrennt sind.

grid-basiert *Grid-basierte* Verfahren versuchen das Effizienzproblem von dichte-basierten Verfahren, wie DBSCAN, bei hochdimensionalen Räumen zu lösen. Dafür wird der Raum zu Beginn in eine definierte Anzahl an Zellen unterteilt. Anschließend wird ein Cluster-Verfahren nicht mehr auf die einzelnen Objekten sondern die Zellen angewendet. Der Vorteil dieses Verfahrens ist, dass es auch in hochdimensionalen Räumen effizient ist und die Geschwindigkeit nicht von der Anzahl der Objekte, sondern nur von der Anzahl der Zellen abhängig ist.

3.2.1 K-means

K-means [Mac67] ist einer der wohl bekanntesten Clustering-Algorithmen. Die Grundidee dabei ist, k Zentren, für jeden Cluster eines, zu definieren. Die Zentren sollten geschickt verteilt sein, da diese Einfluss auf die erzeugten Cluster haben. Nach der Verteilung der Zentren wird jeder Punkt aus der Punktmenge dem Zentrum zugeordnet, zu dem er den geringsten Abstand aufweist. Nachdem alle Punkte zugeordnet wurden, wird für jeden der k Cluster sein neues Zentrum berechnet. Nun wird wieder jeder Punkt einem Zentrum zugeordnet. Diese Schleife wiederholt sich so lange, bis sich die Position der Zentren nicht mehr ändert.

3.2.2 DBSCAN

Beim Betrachten von Abbildung 3.2 sind die unterschiedlichen Cluster zu erkennen, da hier die Punkte eine höhere Dichte aufweisen als die Punkte, die zum Rauschen gehören. Dies ist auch die Grundidee des ***D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise*-Algorithmus [EKJX96], bei dem jeder Punkt in einem Cluster eine minimale Anzahl an Nachbarpunkten innerhalb des Radius ε haben

[1] Ein Baum, der die Objektmenge rekursiv in kleine Teilmengen aufteilt.

muss. Der Algorithmus kann dabei auf 2D, 3D und n-dimensionale Räume angewandt werden. Als Abstandsmaß im dreidimensionalen Raum bietet sich die euklidische Distanz an, es kann aber auch jede andere Abstandsfunktion verwendet werden.

Abbildung 3.2: Einfache Cluster in 2D mit Rauschen.

Beim DBSCAN-Verfahren wird zwischen *Kernobjekten* und *Randobjekten* unterschieden. Ein *Kernobjekt* ist ein Punkt p, bei dem $|N_\varepsilon(p)| \geq minPts$ gilt. Ein Punkt q ist ein *Randobjekt*, wenn es einen anderen Punkt $p \in N_\varepsilon(q)$ gibt, für den $|N_\varepsilon(p)| \geq minPts$ gilt. Diese Bedingungen sind in den nachfolgenden Definitionen genauer ausgearbeitet. Eine detaillierte Ausführung hierzu findet sich in [EKJX96].

ε-Nachbarschaft Die *ε-Nachbarschaft* eines Punktes p, bezeichnet mit $N_\varepsilon(p)$, berechnet sich mit $N_\varepsilon(p) = \{p \in D | dist(p,q) \leq \varepsilon\}$, wobei D die gesamte Punktmenge bezeichnet.

Direkte Dichte-Erreichbarkeit Ein Punkt p ist *direkt dichte-erreichbar* von einem Punkt q bezüglich ε und $minPts$, wenn gilt:

1. $p \in N_\varepsilon(p)$ und
2. $|N_\varepsilon(p)| \geq minPts$ (Kernobjektbedingung)

Dichte-Erreichbarkeit Ein Punkt p ist *dichte-erreichbar* von einem Punkt q bezüglich ε und $minPts$, wenn eine Kette von Punkten $p_1, \ldots, p_n; p_1 = q; p_n = p$ existiert, so dass jeder Punkt p_{i+1} *direkt dichte-erreichbar* vom Punkt p_i ist.

Dichte-Verbundenheit Ein Punkt p ist *dichte-verbunden* mit einem Punkt q bezüglich ε und $minPts$, wenn es einen Punkt o gibt, so dass p und q *dichte-erreichbar* von o bezüglich ε und $minPts$ sind.

Cluster Ein *Cluster* C bezüglich ε und $minPts$ ist eine nicht-leere Teilmenge der Punktmenge D, für die folgende Bedingungen gelten:

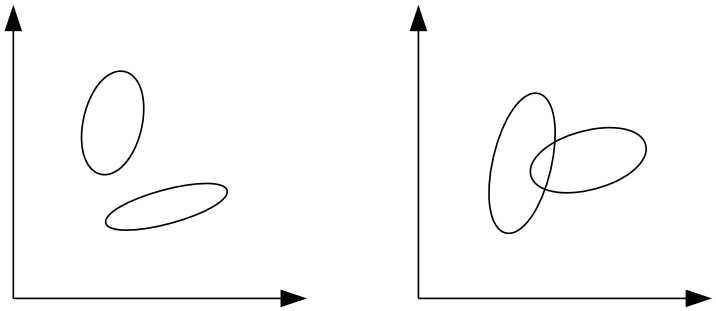
1. Maximalität: $\forall p, q \in D$: wenn $p \in C$ und q *dichte-erreichbar* von p bezüglich ε und $minPts$ ist, dann gilt $q \in C$.
2. Verbundenheit: $\forall p, q \in C$: p ist dichte-verbunden mit q bezüglich ε und $minPts$.

Rauschen Ein Punkt p gehört zum *Rauschen*, wenn er nicht in einem Cluster $C_1, \ldots, C_k$ der Punktmenge D liegt.

3.3 Klassifizierung von Objekten

3.3.1 Merkmalsraum

Eine Menge von P Eigenschaften bilden einen P-dimensionalen Raum $\mathbb{M}$, der als *Merkmalsraum* bezeichnet wird [JÖ5, S. 567ff]. Jedes Objekt in diesem Raum wird als *Merkmalsvektor* dargestellt. Bei einer guten Wahl der Merkmale liegen alle *Merkmalsvektoren* einer Objektklasse nahe beieinander. Unterschiedliche Objektklassen können in diesem Raum separiert werden, wenn die Cluster, die sie bilden, gut voneinander getrennt sind (siehe Abbildung 3.3(a)). Bei schlechten Merkmalen können sich Cluster überlappen (siehe Abbildung 3.3(b)) oder es existieren keine Cluster. Ist dies der Fall, so ist eine fehlerfreie Klassifizierung nicht möglich.



(a) Gut getrennte Cluster. (b) Überlappende Cluster.

Abbildung 3.3: Cluster im zweidimensionalen Raum.

Die Wahl der richtigen Merkmale ist deshalb von großer Bedeutung. Wichtig ist dabei zu beachten, dass es nicht auf die Anzahl, sondern auf die Qualität der Merkmale ankommt. Bei einer Menge von zehn Merkmalen, bei der jedes Merkmal nur zwei Objektklassen unterscheiden kann, ergeben sich $2^{10} = 1024$ Objektklassen. Jedes Merkmal sollte neue Informationen enthalten, welche die bisherigen Merkmale ergänzen. Objektklassen, die sich in einem Merkmal nicht unterscheiden, sollten sich in einem anderen Merkmal gut unterscheiden. Allgemein kann gesagt werden, dass die Merkmale nicht miteinander korrelieren sollten. Die Korrelation von Merkmalen lässt sich dabei, wie in [JÖ5, S. 571ff] beschrieben, berechnen.

Die *Kreuzkovarianz* σ_{pq} zweier Merkmale m_p und m_q aus dem P-dimensionalen Merkmalsvektor einer Objektklasse ist dabei die entscheidende Größe:

$$\sigma_{pq} = \overline{(m_p - \overline{m_q})(m_q - \overline{m_p})} \tag{3.1}$$

Die beiden Merkmale sind unkorreliert oder orthogonal zueinander, falls die Kreuzkovarianz σ_{pq} Null ist. Die Größe

$$\sigma_{pp} = \overline{(m_p - \overline{m_p})^2} \tag{3.2}$$

ist ein Maß für die *Varianz* der Eigenschaft. Bei einem guten Merkmal ist die Varianz klein, was eine geringe Ausdehnung des Clusters in die entsprechende Richtung des Merkmalsraums bedeutet.

Aus P Merkmalen kann man die *Kovarianzmatrix* S mit den Koeffizienten σ_{pq} bilden:

$$S = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1P} \\ \sigma_{12} & \sigma_{22} & \cdots & \sigma_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{1P} & \sigma_{2P} & \cdots & \sigma_{PP} \end{bmatrix} \tag{3.3}$$

Die Diagonalelemente der Kovarianzmatrix enthalten die Varianz der P Merkmale, während die anderen Elemente jeweils für die Kreuzkovarianz zweier Merkmalen stehen.

3.3.2 Überwachte und unüberwachte Klassifizierung

Es gibt zwei Arten der Klassifizierung, die *überwachte* Klassifizierung und die *unüberwachte* Klassifizierung. Bei der *überwachten* Klassifizierung werden die Cluster im Merkmalsraum mit einer vorher bekannten Objektmenge bestimmt. Die Anzahl der Klassen, ihre Lage und Ausdehnung ist damit bekannt. Im Gegensatz dazu existiert bei der *unüberwachten* Klassifizierung keine Kenntnis über die zu klassifizierenden Objekte. Die Cluster im Merkmalsraum werden bei diesem Verfahren zuerst aus den zu klassifizierenden Objekten berechnet und anschließend analysiert. Die Anzahl der Cluster ist hierbei im Voraus nicht bekannt.

Man spricht von einem *selbstlernenden* Verfahren, wenn der Merkmalsraum durch jedes neu klassifizierte Objekt aktualisiert wird. Der Merkmalsvektor wird dabei zum Cluster, zu dem das Objekt gehört, hinzugefügt und die Clustergröße entsprechend angepasst. Dieses Verfahren kommt im industriellen Bereich zum Einsatz, bei dem eine wohldefinierte Umgebung existiert. Zeitliche Veränderungen des Umgebungslichtes, z. B. durch Alterung oder Verschmutzung des Beleuchtungssystems, was die Objekterkennung verfälscht, können so kompensiert werden. Es besteht jedoch die Gefahr, dass Clustergrenzen dadurch verfälscht werden und Cluster im schlimmsten Fall zusammenwachsen.

3.3.3 Einfache Klassifizierungsverfahren

Die Zugehörigkeit eines Merkmalsvektors zu einer Objektklasse kann auf unterschiedliche Arten ermittelt werden. Dabei kommt es auf die Repräsentation des Clusters im Merkmalsraum an [JÖ5, S. 574ff].

Quadermethode Bei der *Quadermethode* wird ein Cluster im zweidimensionalen Fall möglichst eng von einem Rechteck umschlossen. Um ein Objekt einer Objektklasse zuzuordnen, wird lediglich verglichen, ob der Merkmalsvektor in das entsprechende Rechteck fällt. Bei n-dimensionalen Räumen wird dementsprechned ein n-dimensionaler Quader verwendet.

Methode des geringsten Abstandes Die Klassifizierung mit der *Methode des geringsten Abstandes* ist eine weitere Möglichkeit für die Modellierung von Clustern. Jeder Cluster wird durch seinen Schwerpunkt repräsentiert. Ein Merkmalsvektor wird dabei der Objektklasse zugeordnet, zu der er den kleinsten Abstand hat. Die Größe der Cluster kann durch einen Skalierungsfaktor in die Entfernungsberechnung mit einfließen.

Methode der höchsten Wahrscheinlichkeit Ein anderes Verfahren ist die *Methode der höchsten Wahrscheinlichkeit*. Hierbei wird jeder Cluster als statistische Wahrscheinlichkeitsdichtefunktion modelliert. Im einfachsten Fall ist dies eine P-dimensionale Normalverteilung. Es wird nun nicht mehr eine binäre Entscheidung getroffen, ob ein Objekt zu einer bestimmten Klasse

gehört, sondern es wird die Wahrscheinlichkeit berechnet, mit der ein Merkmalsvektor zu einer der P Objektklassen gehört. Dies bietet die Möglichkeit, eine Aussage über die Qualität der Zuordnung zu treffen, was ein Vorteil gegenüber den anderen Verfahren ist.

3.3.4 Mahalanobis-Distanz

Die Mahalanobis-Distanz (nach P. C. Mahalanobis) ist ein einheitenloses Distanzmaß zwischen Punkten in einem n-dimensionalen Raum. Sie wird z. B. in der Statistik im Zusammenhang mit multivarianten Verfahren verwendet. Die Mahalanobis-Distanz d zwischen einem Merkmalsvektor $\vec{p}$ und dem Erwartungsvektor $\vec{\mu}$ berechnet sich dabei mit:

$$d(\vec{p}, \vec{\mu}) = \sqrt{(\vec{p} - \vec{\mu})^T S^{-1} (\vec{p} - \vec{\mu})} \tag{3.4}$$

wobei S^{-1} die Kovarianzmatrix der Objektklasse des Erwartungsvektors $\vec{\mu}$ ist. Ein Cluster wird dabei durch eine P-dimensionale Normalverteilung repräsentiert.

Die quadratische Mahalanobis-Distanz d^2 folgt dabei der Chi-Quadrat-Verteilung [Fil05]. Anhand dieser kann zu einer gegebenen Wahrscheinlichkeit p der maximalen Wert für die Entfernung d bestimmt werden. In Abbildung 3.4 ist $z_{(p;f)}$ das Quantil zur Wahrscheinlichkeit p bei f Freiheitsgraden. Die Anzahl der Freiheitsgrade ist dabei gleich der Anzahl an Dimensionen des Merkmalraums.

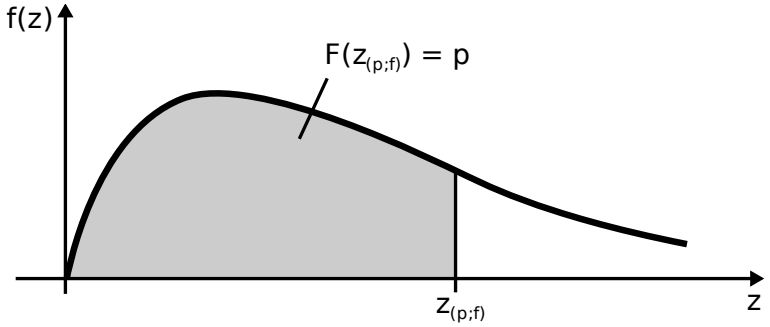


Abbildung 3.4: Chi-Quadrat-Verteilung.

Die Tabelle 3.1 enthält für spezielle Werte von p das jeweils zugehörige Quantil $z_{(p;f)}$ in Abhängigkeit vom Freiheitsgrad f [Pap06, S. 504]. Daraus ergibt sich z. B. bei 3 verwendeten Merkmalen und einer Wahrscheinlichkeit von $97{,}5\%$, eine maximale quadratische Mahalanobis-Distanz von $d^2 = z_{(0{,}975;3)} = 9{,}35$.

	p									
f	0,005	0,01	0,025	0,05	0,10	0,90	0,95	0,975	0,99	0,995
1	0,000	0,000	0,001	0,004	0,016	2,71	3,84	5,02	6,63	7,88
2	0,01	0,020	0,051	0,103	0,211	4,61	5,99	7,38	9,21	10,60
3	0,07	0,115	0,216	0,352	0,584	6,25	7,81	9,35	11,35	12,84
4	0,21	0,297	0,484	0,711	1,064	7,78	9,49	11,14	13,28	14,86
5	0,41	0,554	0,831	1,15	1,16	9,24	11,07	12,83	15,09	16,75

Tabelle 3.1: Quantile der Chi-Quadrat-Verteilung.

Kapitel 4

Umgebungserfassung

4.1 Anforderungsanalyse

Für die Ojekterkennung und die anschließende Bahnplanung ist ein dreidimensionales Umgebungsmodell notwendig. Anhand dieses Modells ist es möglich, die Objekte zu klassifizieren und ihre Position zu bestimmen. Des Weiteren kann durch die Lage und Ausdehnung der Objekte der Freiraum, in dem sich der Manipulator bewegen kann, ermittelt werden. Wichtig dabei ist, dass die Entfernungen im Modell mit denen in der realen Umwelt übereinstimmen, da ansonsten die Objekte später nicht gegriffen werden können.

4.2 Hardwarekonfiguration

4.2.1 Potentielle Konfigurationen

Aus den oben beschriebenen Anforderungen bietet sich die Verwendung eines Laserscanners an, da dieser die genauesten Entfernungswerte liefert. Aufgrund seiner kompakten Abmessungen und der damit verbunden flexibleren Anbringung am Roboter wurde der *Hokuyo URG-04X* ausgewählt. Dabei kommen folgende Montagemöglichkeiten des Laserscanners am Roboter in Frage:

Roboter Die einfachste Möglichkeit ist es, den Laserscanner direkt am Roboter zu befestigen und zwar um 90° auf die Seite geneigt, sodass er die Umgebung senkrecht scannt. Die Erfassung der Umgebung erfolgt durch Drehen der Roboterplattform. Ein Nachteil dieser Konfiguration ist, dass die aktuelle Lage des Scanners im Raum nicht genau bekannt ist, wodurch ein präzises Zusammenfügen der einzelnen Scans unmöglich ist. Auf Grund dieser Tatsache ist das gewonnene Umgebungsmodell für die Objekterkennung und die anschließende Manipulation unbrauchbar.

Pan-Tilt Einheit Eine bessere Methode ist es, den Laserscanner auf einer *Pan-Tilt Einheit* zu montieren. Dabei muss der Laserscanner so angebracht werden, dass er den Tisch erfassen kann, ohne durch den Katana-Manipulator in seiner Sicht behindert zu werden. Dies ist jedoch bei Objekten, die sehr nahe am Roboter stehen, nicht immer möglich.

Manipulator Eine weitere Möglichkeit ist es, den Laserscanner am Manipulator zu befestigen, wodurch ein flexibles Scannen der Umgebung möglich ist. Diese Konfiguration erlaubt es, ein Objekt aus einer anderen Perspektive zu scannen, falls es aus der ersten Perspektive nicht identifiziert werden konnte. Der Mensch agiert auf ähnliche Weise. Falls ein Objekt nicht genau

erkannt werden kann, wird die Perspektive gewechselt, um neue Informationen zu gewinnen. Zusätzlich wird durch die Montage am Manipulator die Sicht des Scanners durch diesen nicht behindert.

4.2.2 Gewählte Konfiguration

Die Befestigung des Laserscanners direkt am Manipulator bietet das größte Maß an Flexibilität. Dabei wird der Katana-Arm sowohl zur Umgebungsmanipulation-Ar als auch für das Scannen der Umgebung eingesetzt. Aufgrund der wenigen Freiheitsgrade des Katana-Arms [Bri09] ist die Wahl der Perspektive jedoch soweit eingeschränkt, dass durch weitere Scans keine neuen Informationen gewonnen werden können. Der erste Scan erfolgt dabei schräg von oben, woraus die Höhen- und Tiefeninformation des Objektes gewonnen werden kann. Der Aufbau mit den unterschiedlichen Koordinatensystemen K_a, K_s, K_t ist in Abbildung 4.1 veranschaulicht. Bei K_a handelt es sich um das Koordinatensystem des Armes, welches als Basiskoordinatensystem bezeichnet wird. K_s stellt das Koordinatensystem des Laserscanners, K_t das Koordinatensystem des TCP (**T**ool **C**enter **P**oint) dar.

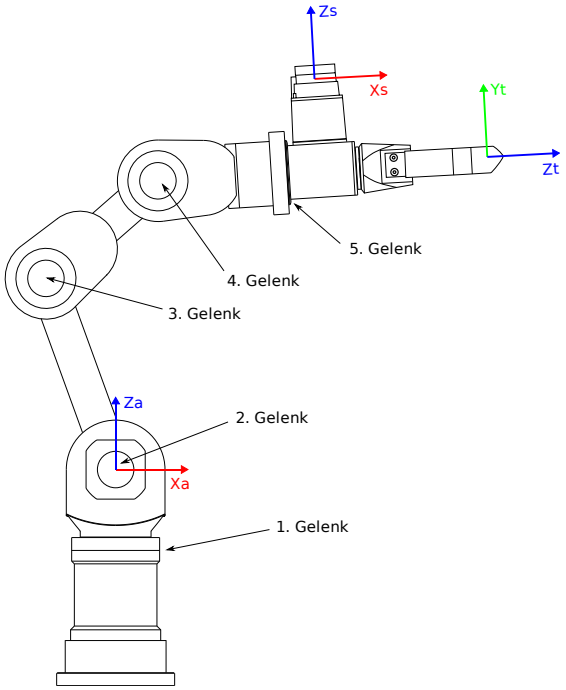


Abbildung 4.1: Katana-Arm mit URG-04LX mit den unterschiedlichen Koordinatensystemen.

4.3 Bestimmung der Scanposition

Da der Laserscanner am Arm befestigt wurde, muss dessen Position im Raum bestimmt werden. Der Scanner befindet sich am letzten Glied des Arms, wodurch seine Position relativ zum TCP angegeben werden kann. Die Scanposition P_a im Basiskoordinatensystem lässt sich dadurch leicht über die Verschiebund O des Lasers zum TCP und der Position M des TCP berechnen. Um einen Laserscan in die Umgebungskarte, die wie das Basiskoordinatensystem orientiert ist, einzutragen muss der Scan lediglich um Scanposition P_a transformiert werden.

Die Lage des TCP Koordinatensystems K_t zum Basiskoordinatensystem K_a wird mittels der ZXZ-Eulerwinkel ϕ, θ und ψ beschrieben. Dabei wird, wie in Abbildung 4.2 zu sehen ist, zuerst um die z-Achse mit dem Winkel ϕ gedreht. Als nächstes wird um die neu entstandene x'-Achse mit dem Winkel θ gedreht. Die letzte Rotation mit dem Winkel ψ erfolgt um die z''-Achse.

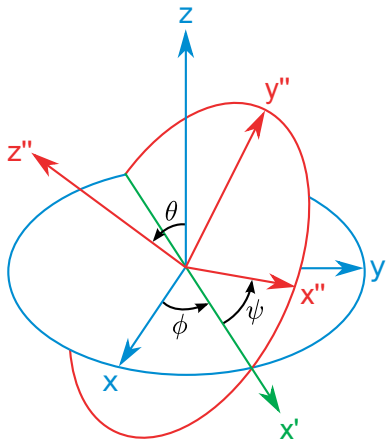


Abbildung 4.2: ZXZ-Eulerwinkel.

Die entsprechende Rotationsmatrix T_{zxz} berechnet sich wie folgt durch die drei Einzelrotationen um die entsprechenden Achsen:

$$T_{zxz} = \begin{bmatrix} cos\phi & -sin\phi & 0 \\ sin\phi & cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

$$= \begin{bmatrix} cos\phi\, cos\psi - sin\phi\, cos\theta\, sin\psi & -cos\phi\, sin\psi - sin\phi\, cos\theta\, cos\psi & sin\phi\, sin\theta \\ sin\phi\, cos\psi + cos\phi\, cos\theta\, sin\psi & -sin\phi\, sin\psi + cos\phi\, cos\theta\, cos\psi & -cos\phi\, sin\theta \\ sin\theta\, sin\psi & sin\theta\, cos\psi & cos\theta \end{bmatrix} \tag{4.2}$$

Die Rotationsmatrix T_{zxz} berücksichtigt jedoch nicht, dass der Laserscan in der XY-Ebene aufgenommen wird. Aus diesem Grund muss, bevor die Transformation durchgeführt wird, eine Korrektur der Lage des Koordinatensystems des Laserscanners bezüglich des Koordinatensystems des TCP vorgenommen werden. Die entsprechende Rotationsmatrix K_{zy} ist:

$$K_{zy} = \begin{bmatrix} cos(-\frac{\pi}{2}) & -sin(-\frac{\pi}{2}) & 0 \\ sin(-\frac{\pi}{2}) & cos(-\frac{\pi}{2}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(-\frac{\pi}{2}) & 0 & sin(-\frac{\pi}{2}) \\ 0 & 1 & 0 \\ -sin(-\frac{\pi}{2}) & 0 & cos(-\frac{\pi}{2}) \end{bmatrix} \tag{4.3}$$

$$= \begin{bmatrix} cos^2(-\frac{\pi}{2}) & -sin(-\frac{\pi}{2}) & cos(-\frac{\pi}{2})\, sin(-\frac{\pi}{2}) \\ sin(-\frac{\pi}{2})\, cos(-\frac{\pi}{2}) & cos(-\frac{\pi}{2}) & sin^2(-\frac{\pi}{2}) \\ -sin(-\frac{\pi}{2}) & 0 & cos(-\frac{\pi}{2}) \end{bmatrix} \tag{4.4}$$

Um die Scanposition P_a zu erhalten ist noch die Verschiebung O des Laserscanners zum TCP, sowie die x,y,z Position M des TCP, nötig. Aus diesen Werten berechnet sich die Position P_a des Laserscanners im Basiskoordinatensystem mit:

$$P_a = M \cdot T_{zxz} \cdot O \cdot K_{zy} \tag{4.5}$$

In der verwendeten Konfiguration beträgt die Verschiebung $O = \begin{pmatrix} 0 \\ 0{,}09 \\ -0{,}163 \end{pmatrix}$ Meter.

4.4 Scannen der Umgebung

Das Scannen der Umgebung erfolgt in einem Scandurchgang, der in Abbildung 4.4 veranschaulicht ist. Der Katana-Arm wird zuerst auf die Startposition gefahren und ist dort wie in Abbildung 4.3 orientiert. Die z-Achse des TCP ist dabei parallel zur x-Achse des Basiskoordinatensystems ausge-

Abbildung 4.3: Scanposition des Katana-Manipulators.

richtet. Der Laserscanner ist so geneigt, dass er die Umgebung senkrecht scannen kann. Das erste Gelenk wird anschließend um $80°$ gegen den Urzeigersinn gedreht. An dieser Position beginnt der eigentliche Scanvorgang, der $160°$ umfasst. Dabei wird jeweils ein Scan aufgenommen und anschließend der Arm im Uhrzgeigersinn gedreht. Insgesammt werden 220 einzelne Scans aufgenommen, die zusammen das dreidimensionale Umgebungsbild bilden.

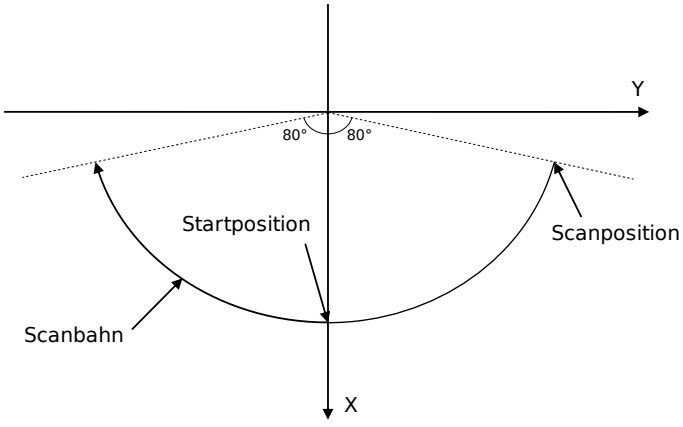
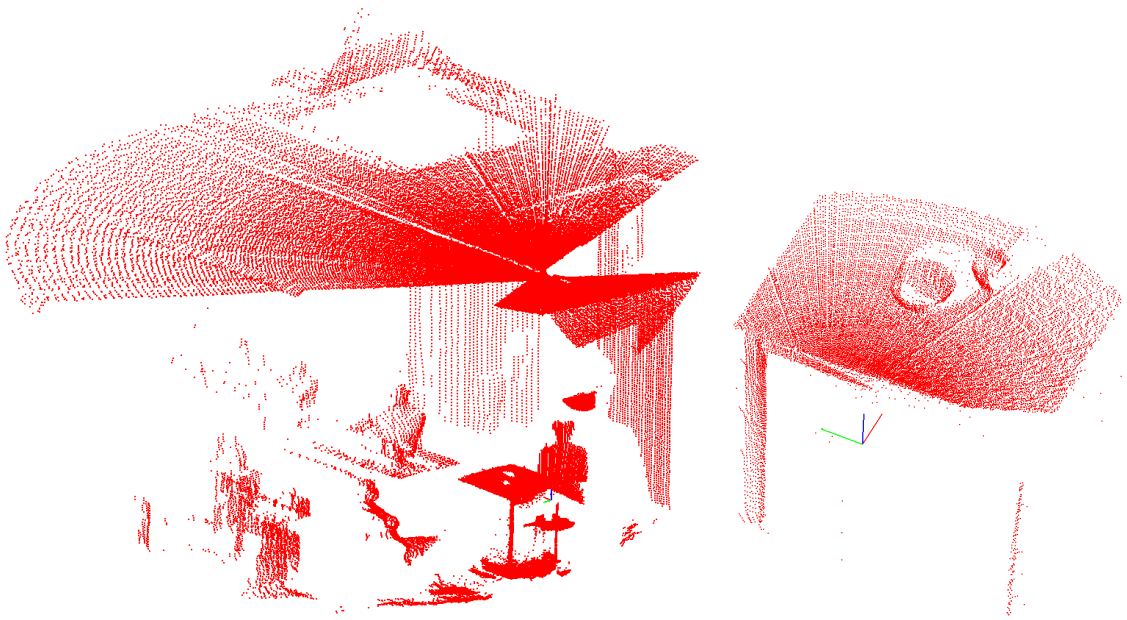


Abbildung 4.4: Scanbahn im Basiskoordinatensystem des Katana-Arms.

Der Katan-Arm wird bei diesem Scanvorgang direkt ohne kollisionsfreies Fahren angesprochen, da noch kein Umgebungsmodell existiert, das hierfür verwendet werden könnte. Aus diesem Grund wurde die Scanposition so gewählt, dass der Arm kaum in den Tisch hineinragt. Eine Kollision mit Gegenständen auf dem Tisch ist ausgeschlossen, da die Objekte auf dem Tisch, wie in [Bri09] beschrieben, einen gewissen Abstand zum Katana-Arm haben müssen.

Abbildung 4.5(a) zeigt ein mit diesem Verfahren aufgenommenes Umgebungsbild eines Tisches, auf dem eine Schüssel und ein Becher stehen. Es ist deutlich zu erkennen, dass der Großteil der enthaltenen Informationen, wie die Wände, die Decke und die Personen im Raum, irrelevant sind. Dies liegt am großen Öffnungswinkel des Laserscanners von $240°$ und einer Reichweite von $5{,}6m$ (vgl. Kapitel 2.2.1). Abbildung 4.5(b) zeigt das gleiche Szenario, wobei die irrelevanten Informationen entfernt wurden. Dabei werden alle Punkte, die weiter als $0{,}7m$ vom Ursprung des Basiskoordinatensystem entfernt sind, verworfen. Dieser Wert ergibt sich aus dem maximalen Aktionsradius des Katana-Arms von $0{,}65m$. Des Weiteren werden alle Punkte, die zum Katana-Arm oder dem Roboter gehören ignoriert. Dazu wird in der XY-Ebene ein Bereich definiert, der den Roboter umschließt. Alle Punkte, die in diesen Bereich fallen, werden verworfen. Im verwendeten Fall handelt es sich um ein Rechteck, das die Ausdehnung von $y = -y = 0{,}3m$, $x = 0{,}1m$ und $-x = 0{,}3m$ besitzt. Je nach Größe des Roboters muss dieser Bereich neu definiert werden. Durch diese Filterung wird die Datenmenge auf die relevanten Informationen begrenzt, was zu einfacheren Algorithmen und einer schnelleren Verarbeitung der Daten führt.



(a) Gesammte gescannte Umgebung.

(b) Gescannte Umgebung nach der Reduzierung auf die relevanten Informationen.

Abbildung 4.5: Mit dem Laserscanner aufgenommene Umgebungsbilder.

Kapitel 5

Objekterkennung

5.1 Anforderungsanalyse

Aus einem gegebenen Szenario, welches aus einem Tisch mit *einfachen* Objekten besteht, sollen die Objekte erkannt werden. Dabei wird die Umgebung mittels eines Laserscanners erfasst und die erhaltenen Punkte in ein globales Koordinatensystem als 3D-Punktwolke eingetragen. Anschließend müssen die Objekte vom Hintergrund getrennt und klassifiziert werden. Das Verfahren muss auf Szenarien mit einem Tisch, der parallel zur XY-Ebene des Basiskoordinatensystems liegt, anwendbar sein. Die Klassifizierungssicherheit sollte bei ca. 90% liegen. Weiterhin muss es möglich sein, auf einfachem Wege weitere Objekte in die Datenbank aufzunehmen und mehr als zwanzig Objekte in der Datenbank zu halten. Dabei sollte der Zeitaufwand für das Klassifizieren der Objekte im Sekundenbereich liegen.

5.2 Ablauf der Objekterkennung

Die Objekterkennung gliedert sich, wie in Abbildung 5.1 zu sehen ist, in drei Teilschritte. Zuerst wird der Hintergrund erkannt und entfernt. Die verbleibenden Punkte werden in Segmente zusammengefasst, wobei jedes Segment ein Objekt repräsentiert. Für jedes Segment werden seine Merkmale ermittelt und mit der Objektdatenbank verglichen. Wenn ein Objekt identifiziert wurde, wird es mit dem entsprechenden Label versehen. Falls es nicht erkannt wurde, wird es als statisches Hindernis angesehen.

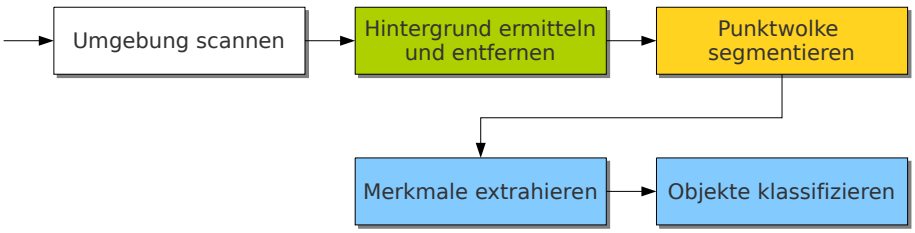


Abbildung 5.1: Teilschritte der Objekterkennung.

5.3 Potentielle Ansätze und Probleme

In der Literatur [Joh97], [HLLS01] gibt es mehrere Ansätze, die das Problem der Objekterkennung in unstrukturierten Punktwolken angehen. Dabei ist jedoch kein Verfahren auszumachen, das sich als Standard für dieses Problem etabliert hat. Vielmehr gibt es unterschiedliche Ansätze mit entsprechenden Vor- und Nachteilen.

In dieser Arbeit wird ein Verfahren vorgestellt, das die Objekterkennung in einfachen Alltagsszenarien ermöglicht. Dabei werden primitive Merkmale verwendet, die leicht aus der Punktwolke zu berechnen sind. Das Verfahren funktioniert für einfache Objekte, die durch Quader oder Zylinder angenähert werden können. Diese Annahme trifft für Becher, Schüsseln, Teller, Dosen, etc. zu. Des Weiteren wird davon ausgegangen, dass die Objekte immer senkrecht auf einer Ebene z. B. einem Tisch stehen und nicht etwa auf der Seite liegen.
Diese Annahmen werden in den meisten Alltagssituationen, wie z. B. einem gedeckten Tisch, erfüllt. Durch diese Einschränkungen vereinfacht sich die Objekterkennung, da eine einfache Trennung der Objekte vom Hintergrund mittels einer Flächenextraktion möglich ist. In Kapitel 7 wird gezeigt, dass das vorgestellte Verfahren eine ausreichende Leistung für einfache Alltagssituationen erbringt.

5.3.1 Entfernen des Hintergrundes

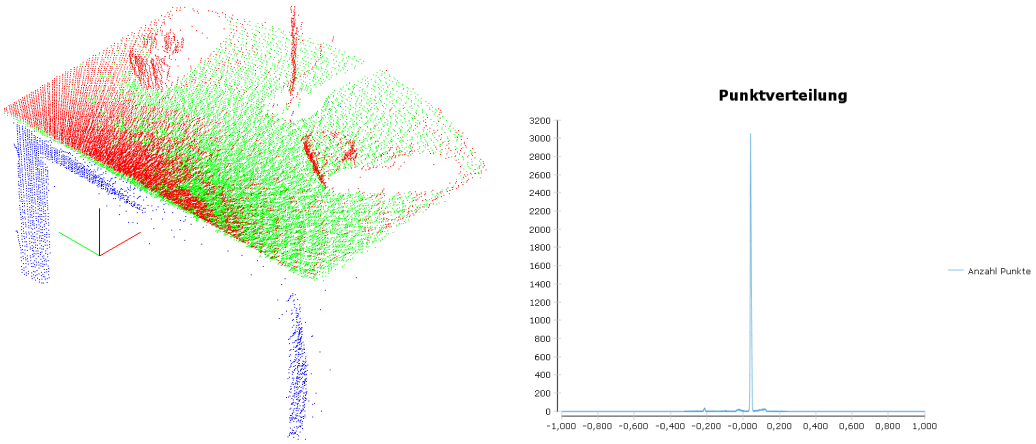
Der erste Schritt zum Erkennen der Objekte ist die Ermittlung und Entfernung des Hintergrundes. In der gegebenen Aufgabenstellung ist dies die Tischplatte. Das Verfahren kann aber auch auf alle anderen Probleme angewandt werden, bei denen der Hintergrund nur aus einer Ebene besteht. Die Hintergrundentfernung ist sinnvoll, da die nachfolgenden Algorithmen auf einer wesentlich kleineren Punktmenge agieren und somit effizienter arbeiten. Zur Ermittlung der Ebene bieten sich mehrere Verfahren an, die sich in Komplexität und Robustheit unterscheiden. Dabei ist bei allen Verfahren wichtig, dass nicht mehr als die Hälfte der Hintergrundebene durch Objekte verdeckt wird, da bei all diesen Verfahren davon ausgegangen wird, dass die Hintergrundebene die meisten Punkte enthält.

5.3.1.1 Erwartungswert in Z-Richtung

Die einfachste Art, die Hintergrundebene zu ermitteln, ist über deren Erwartungswert, das heißt die größte Punktdichte in der Z-Achse. Abbildung 5.2 zeigt die Punkteverteilung eines parallel zur XY-Ebene des Basiskoordinatensystems stehenden Tisches. Die grünen Punkte gehören dabei zur Tischebene. Die blauen Punkte liegen darunter und die roten Punkte darüber. Alle Punkte, bei denen der Abstand zur Tischebene kleiner als $1{,}5\,cm$ ist, das entspricht einer Tischplattendicke von $3\,cm$, gehören zu dieser. Der Abstand d berechnet sich mit der Formel 5.1, wobei die Ebene E durch die Gleichung $\vec{n} \cdot (\vec{r} - \vec{r}_1) = 0$ und der Punkt Q mit dem Ortsvektor $\vec{r}_Q$ gegeben ist.

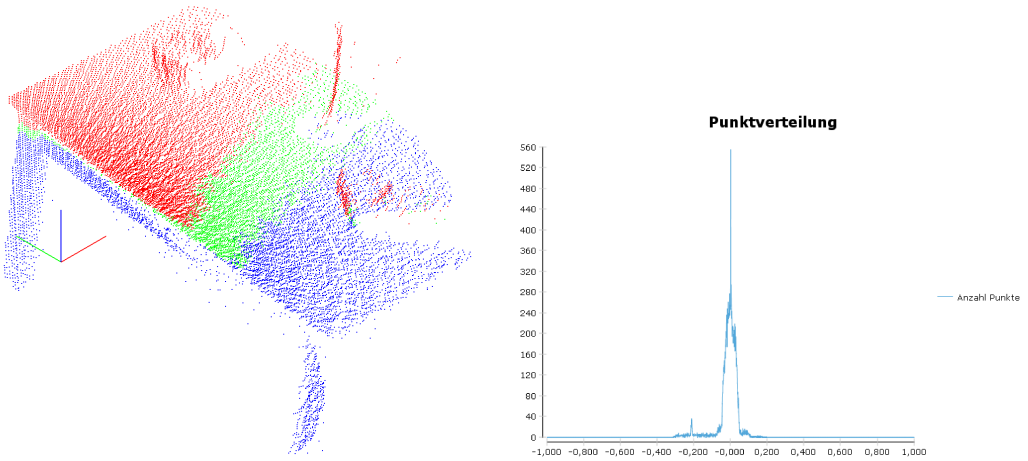
$$d = \frac{|\vec{n} \cdot (\vec{r}_Q - \vec{r}_1)|}{|\vec{n}|} \tag{5.1}$$

Dieses Verfahren ist einfach und effizient, leidet jedoch an Robustheit. In Abbildung 5.2 ist zu erkennen, dass schon eine minimale Abweichung von der Parallelität, die durch die Scanungenauigkeit verursacht werden kann, zu einem unbrauchbaren Ergebnis führt. Bei einer größeren Neigung, wie in Abbildung 5.3 ist dies noch deutlicher sichtbar.



(a) Tischebene parallel zum Koordinatensystem. (b) Histogramm der Punkteverteilung in Z-Richtung.

Abbildung 5.2: Erkannte Tischplatte bei einem parallel zur XY-Ebene des Basiskoordinatensystems stehenden Tisches.



(a) Tischebene um 7 Grad zum Koordinatensystem geneigt. (b) Histogramm der Punkteverteilung in Z-Richtung.

Abbildung 5.3: Erkannte Tischplatte bei einem um 7° zur XY-Ebene des Basiskoordinatensystems geneigt stehenden Tisches.

5.3.1.2 Segmentierung mittels Normalenvektoren

Bei diesem Verfahren wird die Punktwolke in einzelne Segmente zerlegt. Die Segmentierung wird dabei anhand zweier Parameter vorgenommen: dem Abstand p_e zwischen zwei Punkten und dem Winkel p_a ihrer Normalenvektoren. Durch die richtige Wahl der Parameter werden gekrümmte Oberflächen nicht in mehrere Segmente zerlegt. Die beiden Parameter p_e und p_a (siehe Abbildung 5.4) berechnen sich dabei mit den Formeln 5.2 und 5.3 wobei $\vec{n_a}$ der Normalenvektor auf $\vec{a}$ und $\vec{n_b}$ der Normalenvektor auf $\vec{b}$ ist.
Der Normalenvektor eines Punktes wird anhand seiner Nachbarpunkte berechnet, indem durch diese Punkte eine Ausgleichsebene gelegt wird. Der so ermittelte Normalenvektor der Ebene entspricht näherungsweise dem Normalenvektor des Punktes, weshalb dieser Vektor verwendet wird.

$$p_e = |\vec{a} - \vec{b}| \tag{5.2}$$

$$p_a = \arccos\left(\frac{\vec{n_a} \cdot \vec{n_b}}{|\vec{n_a}| \cdot |\vec{n_b}|}\right) \qquad mit: \ 0° \leq p_a \leq 180° \tag{5.3}$$

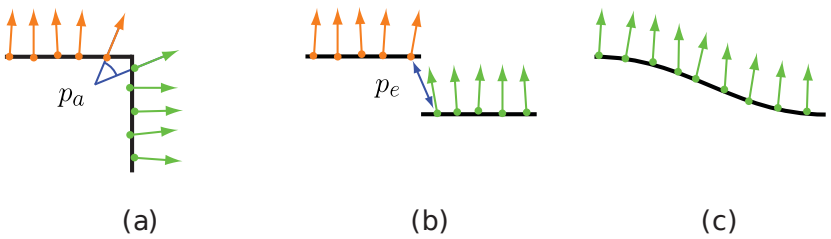


Abbildung 5.4: Auswirkung der zwei Parameter p_a und p_e auf die Segmentierung: (a) Kanten werden mit p_a erkannt. (b) Stufen werden mit p_e erkannt. (c) Gekrümmte Oberflächen werden nicht segmentiert. [KWB09]

Als Ergebnis erhält man eine Menge von Flächen, wobei die Fläche mit den meisten Punkten die Tischebene ist. Das Verfahren ist gegenüber der Neigung der Tischebene robust, die Berechnung der Normalenvektoren birgt jedoch einen gewissen Mehraufwand. Ein Problem ergibt sich, wenn, wie in Abbildung 5.5, die Tischebene in mehrere Segmente aufgeteilt wird. Da nur ein Cluster als Tischebene angenommen wird, kann es sein, dass ein anderes Segment mehr Punkte beinhaltet und somit fälschlicherweise als Tischebene erkannt wird.

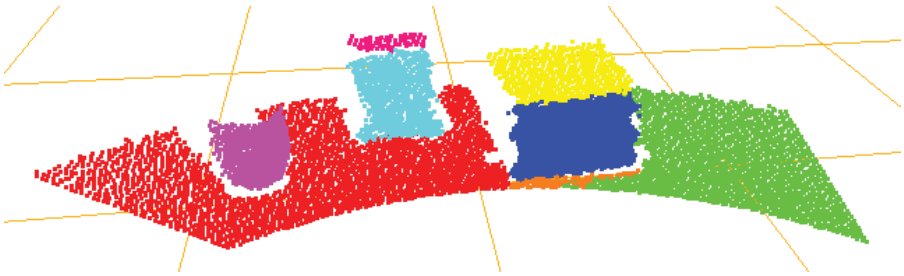
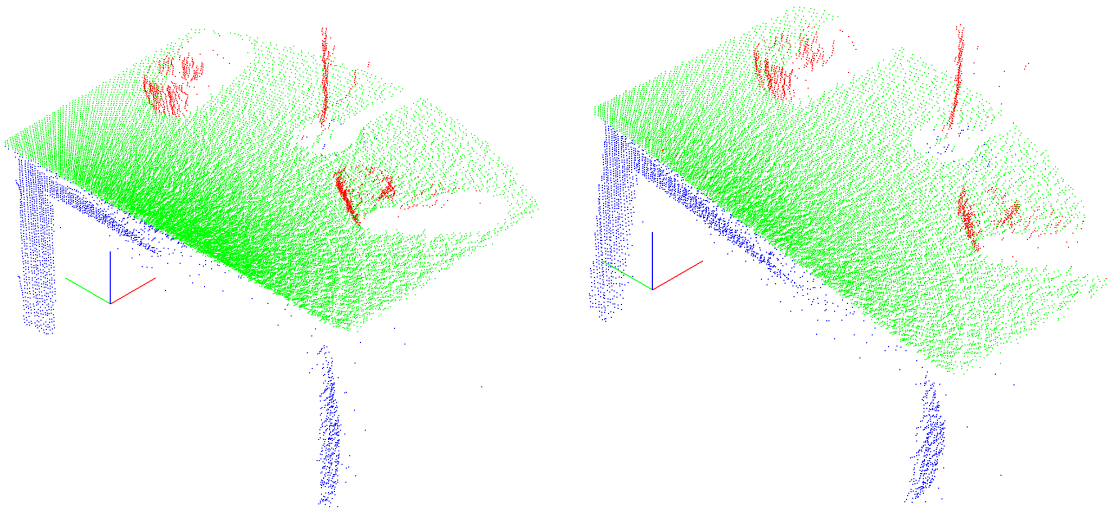


Abbildung 5.5: Segmentierter Tisch mit drei Objekten. [KWB09]

5.3.1.3 RANSAC

Abbildung 5.6 zeigt die mittels RANSAC ermittelte Tischebene (grün) bei einer Tischplatte, die einmal parallel und einmal um 7° zum Basiskoordinatensystem geneigt ist. Dabei wurde ein Wert von $d_{max} = 0{,}015\,m$ als Fehlerschranke gewählt (vgl. Kapitel 3.1). Zu erkennen ist, dass der Algorithmus in beiden Fällen die Tischebene exakt ermittelt und dadurch eine saubere Hintergrunderkennung ermöglicht.



(a) Tischebene parallel zum Koordinatensystem. (b) Tischebene um 7° zum Koordinatensystem geneigt.

Abbildung 5.6: Mittels RANSAC erkannte Tischplatte.

5.3.2 Segmentierung der Punktwolke

Im nächsten Schritt wird die verbleibende Punktwolke in Segmente unterteilt. Hierbei muss jedes entstehende Segment genau ein Objekt repräsentieren. Dieser Schritt erleichtert die spätere Merkmalsextraktion und somit auch die Objekterkennung erheblich.

5.3.2.1 Segmentierung über Normalenvektoren

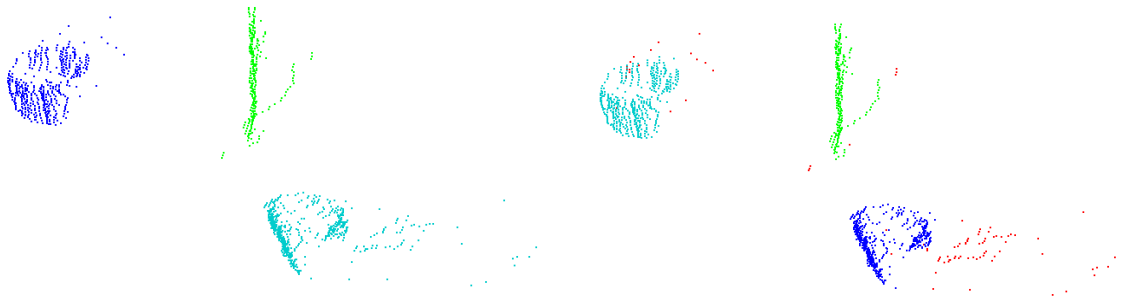
Es gibt mehrere Arbeiten ([KWB09], [RMBB08a]) die ein Segmentierungsverfahren beschreiben, das auf Normalenvektoren basiert. Die Segmente können, ebenso wie beim Verfahren für die Ermittlung der Tischebene (Kapitel 5.3.1.2), anhand der Parameter p_a und p_e ermittelt werden. Das Problem dieses Verfahrens liegt darin, dass nicht jedes Objekt nur durch ein Segment repräsentiert wird, sondern, wie Abbildung 5.7 zeigt, durch mehrere Segmente. Dies kommt durch eine zu große Winkeldifferenz, $p_a > maxAngle$, der Punkte eines Objektes zustande. Eine Berechnung der Objektmerkmale ist somit nicht möglich und das Verfahren kann nicht ohne zusätzliche Schritte verwendet werden.

Abbildung 5.7: Mittels Normalenvektoren segmentierte Szene. [RMBB08a]

5.3.2.2 Segmentierung mittels Clustering

Eine andere Möglichkeit ist es ein Cluster-Verfahren auf die Punktwolke anzuwenden. Das bekannteste Cluster-Verfahren ist wohl k-means aus der Kategorie der *partitionierenden* Verfahren. Der Algorithmus ist einfach und leicht zu implementieren. Um k-means verwenden zu können, muss jedoch schon im Voraus die Anzahl k der zu erstellenden Cluster bekannt sein. Weiterhin wird, wie in Abbildung 5.8(a) zu erkennen ist, jeder Punkt einem Cluster zugeordnet. Dies führt dazu, dass der Cluster durch Rauschen verfälscht wird. Die Laufzeitkomplexität beträgt $O(nkt)$, wobei n die Anzahl der Punkte, k die Anzahl der Cluster und t die Anzahl der Iterationen ist.

Beim DBSCAN-Verfahren aus dem Bereich der *dichte-basierenden* Verfahren muss die Anzahl k der Cluster im Voraus nicht bekannt sein, da die Cluster anhand der Punktdichte ermittelt werden. Die Laufzeitkomplexität beträgt $O(N \log(N))$, wobei N die Anzahl der Punkte in der zu clusternden Punktwolke ist. In Abbildung 5.8(b) ist deutlich zu erkennen, dass DBSCAN das Rauschen (rote Punkte) in der Punktwolke herausfiltert.



(a) Mittels k-means erstellte Cluster.

(b) Mittels DBSCAN erstellte Cluster. Die roten Punkte wurden als Rauschen klassifiziert.

Abbildung 5.8: Mittels unterschiedlicher Cluster-Verfahren erstellte Cluster.

5.3.3 Einfache geometrische Merkmale

In der 2D-Bildverarbeitung gibt es einfache Merkmale, anhand derer eine zuverlässige Objekterkennung möglich ist. Die in [NH04, S. 764ff] vorgestellten Merkmale werden im Folgenden für die Anwendung auf 3D-Modelle erweitert.

5.3.3.1 Abmessungen

Die *Abmessungen* eines Objektes sind ein primitives Merkmal. Dafür wird der minimale Quader, welcher das Objekt ganz umschließt, berechnet. Bei einem achsenparallelen Quader sind dessen Flächen parallel zu den Achsenebenen des Basiskoordinatensystems ausgerichtet. Ein rotationsinvarianter Quader mit minimalem Volumen kann berechnet werden, indem dieser nicht mehr parallel zum Basiskoordinatensystem ausrichtet wird. Die Ausrichtung kann stattdessen anhand der Eigenvektoren des Objektes erfolgen. Die Abmessungen des Objektes sind dann gleich den Abmessungen des Quaders.

5.3.3.2 Volumen und Oberfläche

Die Berechnung des *Volumens* und der *Oberfläche* gestaltet sich im dreidimensionalen Raum schwieriger als im zweidimensionalen, da kein vollständiges Modell des Objektes aufgenommen werden kann. Es ist zwar möglich, die Oberfläche des Objektes abzutasten, das Objektinnere bleibt jedoch verborgen.

Die sichtbare *Oberfläche* des Objektes kann näherungsweise als Summe der belegten Voxel (3D-Pixel) berechnet werden. Um aus der Punktwolke die Voxel zu bekommen, wird diese in ein festes Voxelraster unterteilt. Wenn ein Punkt in ein Voxel fällt, wird dieses als belegt markiert. Die Oberfläche S eines Objektes O berechnet sich demnach mit:

$$S = \sum_{x=1}^{L} \sum_{y=1}^{M} \sum_{z=1}^{N} O(x,y,z) \tag{5.4}$$

Bei einem vollständigen Objektmodell entspricht das *Volumen* dem von der Objektoberfläche eingeschlossenen Raum. Die Berechnung des *Volumens* gestaltet sich jedoch bei einem mit Laserscanner aufgenommenen Modell schwieriger, da durch die Selbstverdeckungen kein vollständiges Objektmodell existiert. Die Oberfläche weist Löcher auf, die vor der Berechnung geschlossen werden müssen. Dies gestaltet sich als problematisch, da die Oberfläche und die Löcher erst erkannt werden müssen, bevor sie behoben werden können. Aus diesem Grund ist die Berechnung des Objektvolumens anhand von 3D-Punktwolken nicht ohne zusätzlichen Aufwand möglich.

5.3.3.3 Kompaktheit

Die *Kompaktheit* K eines Objektes berechnet sich aus dessen Volumen und seiner Oberfläche mit der Formel:

$$K = \frac{Oberfläche^2}{Volumen} \tag{5.5}$$

5.3.3.4 Orientierung

Die *Orientierung* gibt die Lage des Objektes im Raum an. Eine einfache Möglichkeit, diese zu bestimmen, ist es, das Objekt durch einen Ellipsoid anzunähern, wobei die Hauptachsen des Ellipsoiden die Orientierung des Objektes angeben. Die Hauptachsen entsprechen dabei den Eigenvektoren des Ellipsoiden. Die Berechung der Eigenvektoren findet sich in [NH04, S. 768].

5.3.3.5 Schwerpunkt

Der *Schwerpunkt* $(\bar{x}, \bar{y}, \bar{z})$ eines Objektes mit N Punkten berechnet sich nach der Formel:

$$\bar{x} = \frac{\sum_{j=1}^{N} x_j}{N}, \bar{y} = \frac{\sum_{j=1}^{N} y_j}{N}, \bar{z} = \frac{\sum_{j=1}^{N} z_j}{N} \tag{5.6}$$

Die Berechnung aus 3D-Entfernungsdaten, die mit einem Laserscanner aufgenommen wurden, ist jedoch problematisch, da die Punktdichte über das Objekt hinweg nicht gleichbleibend ist. In Kapitel 7.1.3 wird diese Eigenschaft genauer betrachtet.

5.3.3.6 Füllungsgrad

Der *Füllungsgrad* wird aus dem Verhältnis des Objektvolumens V_O zu einem Referenzvolumen V_R berechnet. Dabei kann das Volumen des minimalen Quaders als Referenzvolumen verwendet werden. Der Füllungsgrad F_G berechnet sich demnach mit:

$$F_G = \frac{V_O}{V_R} \tag{5.7}$$

5.3.4 Klassifizierung von Objekten

Unter Klassifizierung versteht man das Problem der Zuordnung eines Objektes O zu einer Objektklasse K aus einer Menge von Objektklassen $K_1, \ldots, K_n$. Die Zuordnung erfolgt dabei anhand vorher extrahierter Merkmale $M_1, \ldots, M_l$. Diese Merkmale können als n-dimensionaler Vektor in einem n-dimensionalen Merkmalsraum aufgefasst werden. Jede Objektklasse ist dann durch einen Cluster im Merkmalsraum beschrieben. Der Cluster kann, wie in Kapitel 3.3.3 beschrieben, einfach durch ein *Rechteck*, durch den *Schwerpunkt* oder durch eine *Wahrscheinlichkeitsfunktion* beschrieben werden. Ein Objekt gehört zu einer Objektklasse, wenn der Merkmalsvektor des Objektes in den Bereich der Objektklasse fällt. Die Repräsentation durch ein Rechteck oder den Schwerpunkt stellt die einfachste Möglichkeit dar, wobei beim Schwerpunkt-Verfahren die Clustergröße extra berücksichtigt werden muss. Die Verwendung einer Wahrscheinlichkeitsfunktion bietet den Vorteil, dass auch eine Aussage über die Qualität der Zuordnung getroffen werden kann, was bei den beiden anderen Verfahren nicht der Fall ist.

Die Klassifizierung kann sowohl *überwacht* als auch *unüberwacht* erfolgen. Bei der *überwachten* Klassifizierung werden, wie in Kapitel 3.3.2 beschrieben, die Objektklassen anhand einer vorher ausgewählten Objektmenge bestimmt. Die Objektmenge wird dabei so gewählt, dass die Objektklasse eine möglichst große Aussagekraft besitzt. Eine hohe Aussagekraft bedeutet, dass die Sicherheit, mit der ein Objekt der richtigen Objektklasse zugeordnet wird, hoch ist. Im Gegensatz dazu werden bei der *unüberwachten* Klassifizierung die Objektklassen aus den zu klassifizierenden Objekten ermittelt. Der Vorteil hierbei ist, dass dafür keine vorher ausgewählte Objektmenge gebraucht wird. Es kann jedoch der Fall sein, dass die ermittelten Objektklassen eine geringe Aussagekraft besitzen.

Ein *selbstlernendes* Verfahren kann eingesetzt werden, um automatisch neue Objekte in den Merkmalsraum aufzunehmen oder Veränderungen der Objekte über die Zeit hinweg zu berücksichtigen. Eine Gefahr ist jedoch, dass die Objektklassen verfälscht werden und die Aussagekraft dadurch abnimmt.

5.4 Ausgewähltes Verfahren

5.4.1 Hintergrundentfernung mittels RANSAC

Als Verfahren für die Entfernung des Hintergrundes wird der RANSAC-Algorithmus, welcher gegenüber der Neigung der Tischebene robust ist, verwendet. Eine einfach zu verwendende C++ Implementierung des Algorithmus beinhaltet das MRPT[1] Projekt.

Im Rahmen dieser Bachelorarbeit kam das euklische Distanzmaß und eine Fehlerschranke von $d_{max} = 0{,}015\,m$ zum Einsatz. Für die Parameter der Ausgleichsebene werden dieselben Parameter verwendet, welche bei der Erstellung des *Consensus Set* mit den meisten Punkten zum Einsatz kamen. Die so ermittelte Ebene liefert für diese Anwendung ein ausreichendes Ergebnis, weshalb auf eine kompliziertere Ermittlung der Parameter z. B. mittels der *Methode der kleinsten Quadrate* verzichtet wird.

5.4.2 Segmentierung mittels DBSCAN

Für die Segmentierung der verbleibenden Punktwolke wird das *DBSCAN*-Clusterverfahren eingesetzt. Es kann, im Vergleich zur *Segmentierung mittels Normalenvektoren*, ohne zusätzliche Schritte direkt auf die Punktwolke angewendet werden. Bei der Segmentierung mittels Normalenvektoren müssen die Normalenvektoren zuerst aufwändig berechnet werden. Ein weiteres Problem ist, dass ein Objekt durch mehrere Segmente repräsentiert wird und diese Segmente zuerst zu einem Segment zusammengefasst werden müssen, bevor die Klassifizierung durchgeführt werden kann.

Das DBSCAN-Verfahren kann im Gegensatz zu k-means auch mit Punktwolken umgehen, die mit Rauschen behaftet sind. Rauschen kommt in allen Datensätzen, die in der realen Welt aufgenommen wurden, vor. Aus diesem Grund und der Tatsache, dass schon im Voraus die Anzahl k der Cluster bekannt sein muss (vgl. Kapitel 3.2.1), scheidet k-means für die Segmentierung der Punktwolke aus.

5.4.2.1 Ermittlung der DBSCAN-Parameter ϵ und $minPts$

Die Ermittlung der DBSCAN-Parameter ϵ und $minPts$ spielt für das erfolgreiche Bilden der Cluster eine große Rolle. In [TSK05, S. 529] wird ein Verfahren zur Ermittlung der beiden Parameter vorgestellt. Die Grundidee ist die Entfernung eines Punktes zu seinen k-Nachbarpunkten zu betrachten. Die Entfernung des k-ten Nachbarpunktes, wird dabei als $k\text{-}dist$ bezeichnet. Für Punkte desselben Clusters wird $k\text{-}dist$ klein sein, es sei denn k ist größer als die Punktmenge des Clusters. Im Gegensatz dazu ist $k\text{-}dist$ für Punkte, die zum Rauschen gehören, groß.

Um ϵ und $minPts$ zu bestimmen, wird bei einem definierten k für jeden Punkt $k\text{-}dist$ berechnet. Die Punkte werden anschließend nach der Größe von $k\text{-}dist$ sortiert in ein Diagramm eingetragen. Es wird erwartet, dass im $k\text{-}dist$ Diagramm ein Knick in der Kurve existiert, welcher einen sinvollen Wert für ϵ darstellt. Falls dieser Wert für ϵ und k für $minPts$ verwendet wird, werden alle Punkte, deren $k\text{-}dist$ kleiner als ϵ ist, als Kernobjekte, die anderen als Randobjekte oder Rauschen klassifiziert. Der Wert für ϵ ist vom gewählten k abhängig, wobei sich ϵ langsamer ändert als k. Wird k zu

[1] The Mobile Robot Programming Toolkit (`http://babel.isa.uma.es/mrpt/index.php/Main_Page`)

klein gewählt, werden nahe beieinander liegende Punkte, die zum Rauschen gehören, als eigenständiges Cluster erkannt. Wird k zu groß gewählt, werden Cluster, deren Punktmenge kleiner als k ist, fälschlicherweise als Rauschen erkannt.

In Abbildung 5.9 sind auf der X-Achse die einzelnen Punkte, sortiert nach ihrem $k\text{-}dist$ aufgetragen. An der Y-Achse kann der $k\text{-}dist$ Wert des Punktes abgelesen werden.

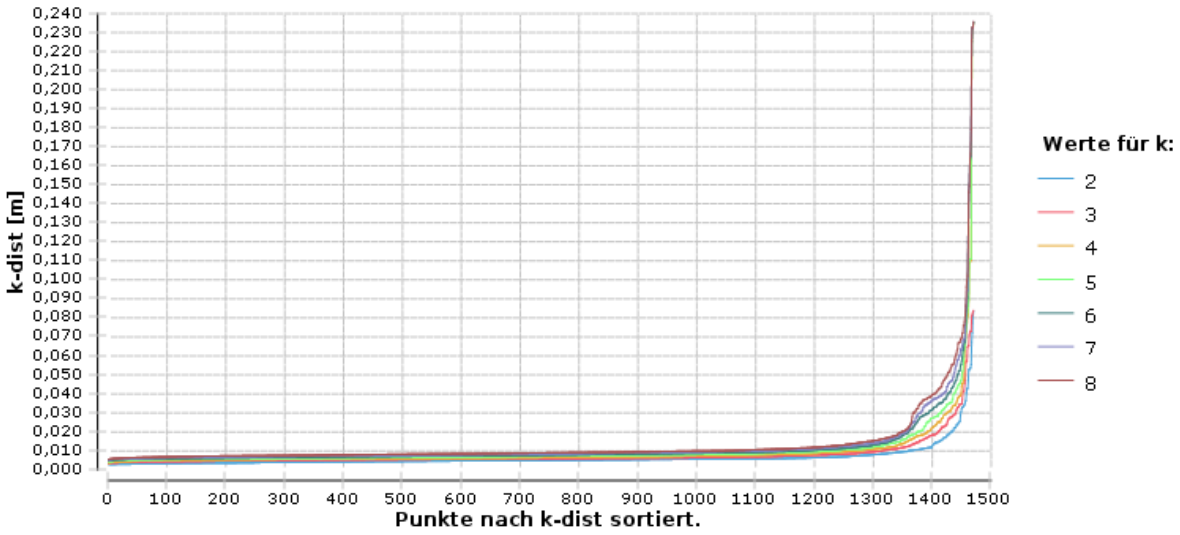


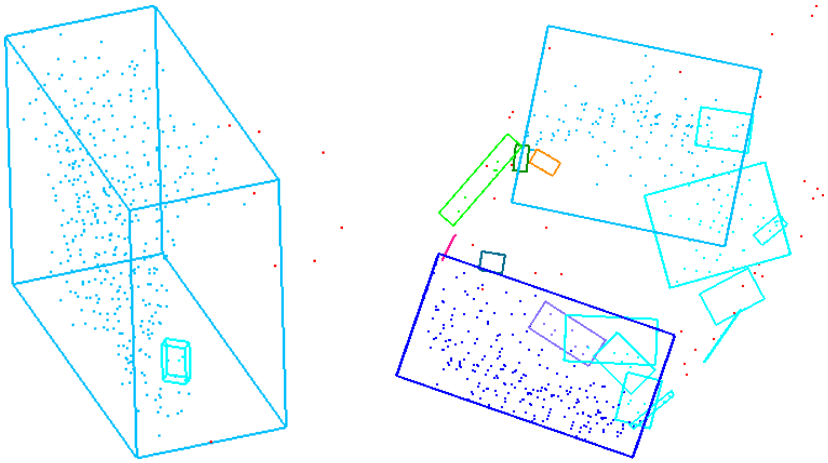
Abbildung 5.9: $k\text{-}dist$ Diagramm für unterschiedliche k.

Die Kurven wurden für die Werte $k = 2, \ldots, 8$ erstellt. Dieser Bereich wurde untersucht, da beim originalen DBSCAN-Algorithmus $k = 4$ als Wert, welcher für die meisten 2D-Anwendungen zutreffend ist, angegeben wurde. Wie anhand von Abbildung 5.9 zu erkennen ist, kommt ein Wert für ϵ zwischen $0{,}015$ und $0{,}02$ in Frage. Um eine möglichst gute Rauscherkennung zu erhalten, wurde $\epsilon = 0{,}015$ gewählt. Für $minPts$ wurden die Werte $minPts = 4, \ldots, 8$ anhand eines geclusterten Szenarios untersucht, wobei sich zeigte, dass die Werte von $minPts = 5, \ldots, 8$ die besten Ergebnisse lieferten. Aus diesem Grunde wurde $minPts = 6$ für diese Arbeit ausgewählt.

In Abbildung 5.10 sind zwei Objekte, die mit diesen Parametern geclustert wurden, abgebildet. Die Schachtel wurde als ein Cluster erkannt (vgl. Abbildung 5.10(a)), wobei der Becher in zwei große und mehrere kleine Cluster aufgeteilt wurde (vgl. Abbildung 5.10(b)). Die als Rauschen erkannten Punkte sind rot in den Bildern eingezeichnet.
Dieses Ergebnis ist für die anschließende Merkmalsextraktion nicht brauchbar, da hier jedes Objekt durch genau ein Cluster repräsentiert sein muss. Eine Erhöhung von ϵ löst dieses Problem nicht, da dadurch wieder die nahe am Objekt gelegenen Rauschpunkte als zum Objekt gehörend klassifiziert würden. Aus diesem Grund muss nach dem Clustering ein weiterer Schritt eingeführt werden, der erkennt, ob zwei oder mehrere Cluster zu einem zusammengefasst werden müssen.

5.4.2.2 Mergen von Clustern

Wie oben beschrieben muss nach dem Clustering ein weiterer Schritt eingeführt werden, wodurch der gesamte Clustervorgang in zwei Schritten vonstatten geht. Im ersten Schritt werden zunächst die Cluster mit den Parametern erstellt, die eine maximale Rauscherkennung bieten. Hierbei entstehen für ein Objekt, wie in Abbildung 5.11(a) zu sehen ist, mehrere Cluster. Im zweiten Schritt werden die Cluster, die zu einem Objekt gehören, gemerged. Als Resultat dieses Schrittes erhält man für jedes Objekt

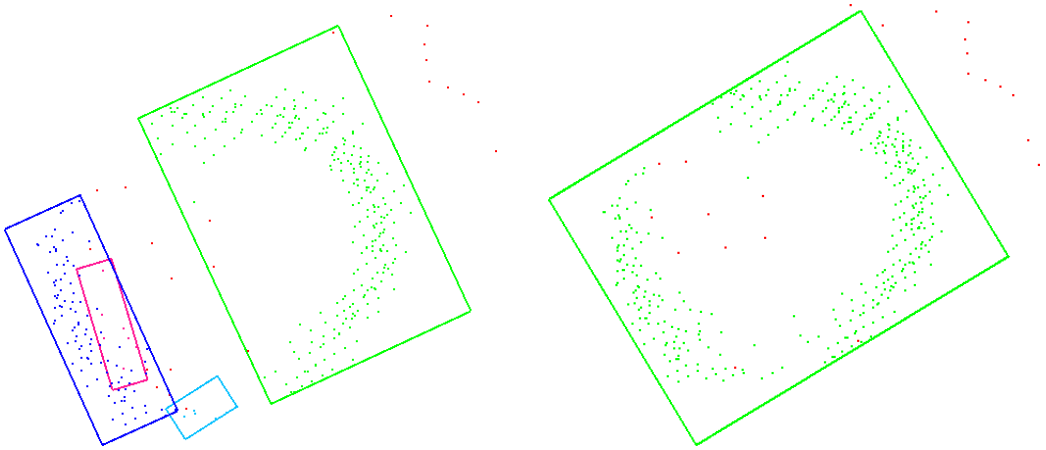


(a) Die Schachtel wurde als ein Cluster erkannt.

(b) Der Becher wurde in mehrere Cluster aufgeteilt.

Abbildung 5.10: Gebildete Cluster bei $\epsilon = 0.015\,m$ und $minPts = 6$.

ein Cluster. Der Parameter t für das Mergen zweier Cluster C_1 und C_2 ist der minimale Abstand der beiden Cluster. Es werden dabei zwei Cluster gemerged, wenn $\exists p \in C_1, \exists q \in C_2 : dist(p,q) \leq t$ gilt. Als Wert für t kann der minimale Abstand zwischen zwei Objekten, der für das Greifen der Objekte nötig ist, gewählt werden. Dieser Abstand ist nach [Bri09] für den Katana-Manipulator $0{,}05\,m$. Um etwas Sicherheit zu haben, wurde $t = 0{,}04\,m$ gewählt. Mit Hilfe dieses Parameters wird, wie in Abbildung 5.11(b) zu sehen ist, ein gutes Ergebnis erzielt.



(a) Cluster bei $\epsilon = 0.015\,m, minPts = 6, t = 0\,m$.

(b) Cluster bei $\epsilon = 0.015\,m, minPts = 6, t = 0.04\,m$.

Abbildung 5.11: Gebildete Cluster bei unterschiedlicher Parametrisierung.

5.4.3 Verwendete Merkmale

Die Anzahl der in Kapitel 5.3.3 vorgestellten Merkmale, die für die Objekterkennung in dreidimensionalen Punktwolken verwendet werden können, ist recht begrenzt. So scheidet der *Schwerpunkt* aus, da er durch die unterschiedliche Dichte der vom Laserscanner aufgenommenen Punkte stark beeinflusst wird.

Die Berechnung der *Oberfläche* und des *Volumens* gestaltet sich, wie in Kapitel 5.3.3.2 beschrieben, als äußerst schwierig, weshalb diese Merkmale hier keine Verwendung finden. Des Weiteren fallen dadurch die *Kompaktheit* und der *Füllungsgrad* weg, da für beide das Volumen bzw. die Oberfläche nötig ist.

Übrig bleibt die *Orientierung* und die *Abmessungen* des Objektes. Die Orientierung ist kein eigenständiges Merkmal, das zur Identifizierung von Objekten verwendet werden kann, da es nur die Lage im Raum beschreibt. Die Kenntnis der Orientierung eines Objektes ist jedoch von Vorteil, da alle weiteren Merkmale in einem für das Objekt einheitlichen Koordinatensystem berechnet werden können. Dadurch wird der Einfluss der Orientierung auf die anderen Merkmale eliminiert. Aus diesem Grund wird in dieser Arbeit zuerst die Orientierung ermittelt und das Objekt in ein einheitliches Objektkoordinatensystem gebracht, bevor die weiteren Merkmale extrahiert werden.
Die Abmessungen des Objektes lassen sich im dreidimensionalen Fall leicht berechnen und stellen das einzige Merkmal dar, welches für die Objekterkennung verwendet wird. Um die Abmessungen zu erhalten, wird der minimale umhüllende Quader im Objektkoordinatensystem bestimmt. Die Abmessungen des Objektes sind somit gleich den Abmessungen des Quaders.
In Kapitel 7.3.1 wird gezeigt, dass trotz der Beschränkung auf dieses Merkmal eine Erkennung von einfachen Alltagsgegenständen möglich ist.

5.4.4 Objektklassifizierung mittels Merkmalsraum und Mahalanobis Distanz

Die Klassifizierung der Objekte erfolgt anhand der oben vorgestellten Merkmale. Die Merkmale werden dabei als Merkmalsvektor in einem dreidimensionalen Merkmalsraum aufgefasst. Jede Objektklasse ist in diesem Raum durch eine dreidimensionale Wahrscheinlichkeitsdichtefunktion modelliert. Die Funktion bietet den Vorteil, eine Aussage über die Qualität der Zuordnung zu treffen, weshalb dieses Klassifizierungsverfahren hier Verwendung findet.

Als Abstandsmaß zwischen einem Merkmalsvektor und dem Erwartungsvektor der Objektklasse dient die Mahalanobis Distanz (vgl. Kapitel 3.3.4). Die quadratische Mahalanobis Distanz folgt dabei der Chi-Quadrat-Verteilung, anhand welcher der Wert für die Distanz d^2 bestimmt werden kann. Bei einer gewünschen *Erkennungssicherheit* von z. B. $97{,}5\%$ ergibt sich eine Distanz von $d^2 = z_{(0{,}975;3)} = 9{,}35$. Dieser Wert sagt aus, dass $97{,}5\%$ der A-Objekte, eine kleinere Distanz als $9{,}35$ zur A-Objektklasse besitzen. Dieser Wert sagt jedoch nicht aus, dass ein A-Objekt zu $97{,}5\%$ der richtigen Objektklasse zugeordnet wurde. Um die *Zuordnungssicherheit* zu erhalten, muss die Wahrscheinlichkeit $1 - 0{,}975 = 0{,}025$ gewählt werden. Daraus ergibt sich eine Distanz von $d^2 = z_{(0{,}025;3)} = 0{,}484$. Wenn nun ein Objekt zur A-Objektklasse eine Mahalanobis Distanz kleiner $0{,}484$ aufweist, dann handelt es sich bei diesem Objekt mit einer Wahrscheinlichkeit von $97{,}5\%$ um ein A-Objekt.

Für die Objektklassifizierung kommt eine *überwachte* Klassifizierung zum Einsatz. Dadurch kann die Objektmenge so gewählt werden, dass die daraus generierte Objektklasse eine möglichst hohe Aussagekraft besitzt. Dies ist ein entscheidender Vorteil gegenüber der *unüberwachten* Klassifizierung. Neue Objekte können so durch einen definierten Lernprozess in die Objektdatenbank aufgenommen werden. Dieser Prozess besteht aus einer festgelegten Anzahl von einzelnen Scans des Tisches. Die Anzahl der Scans beeinflusst dabei die Aussagekraft der ermittelten Objektklassen, wobei eine

große Anzahl an Scans eine aussagekräftigere Objekteklasse als eine geringe Anzahl liefert. Die Anzahl der Scans darf jedoch nicht zu hoch gewählt werden, da das Lernen eines Objektes vom Benutzer in einer angemessenen Zeit durchführbar sein sollte. Eine Anzahl von 5 bis 15 Scans scheint deshalb für das Lernen eines Objektes geeignet zu sein.
Während des Scanvorgangs darf sich auf dem Tisch nur das zu lernende Objekt befinden, wobei es bei jedem Scan an einer anderen Position stehen sollte. In der Objektdatenbank werden die Merkmalsvektoren der einzelnen Scans zusammen mit dem Objektnamen gespeichert. Durch diesen Vorgang ist es einfach, neue Objekte in die Objektdatenbank aufzunehmen.

Kapitel 6

Greifen von Objekten

In der Mobilen Manipulation gibt es zwei große Teilbereiche: die *Objekterkennung* und die *Bahnplanung*. Die *Objekterkennung* ist für die Klassifizierung der in der Szene enthaltenen Objekte verantwortlich. Die Informationen über die Größe, die Lage und die Art des Objektes werden dabei von der *Objekterkennung* an die *Bahnplanung* übergeben (vgl. Abbildung 6.1). Die *Bahnplanung* versucht, anhand dieser Daten eine kollisionsfreie Bahn für das Greifen der Objekte zu planen. Falls eine solche Bahn existiert, wird diese ausgeführt.

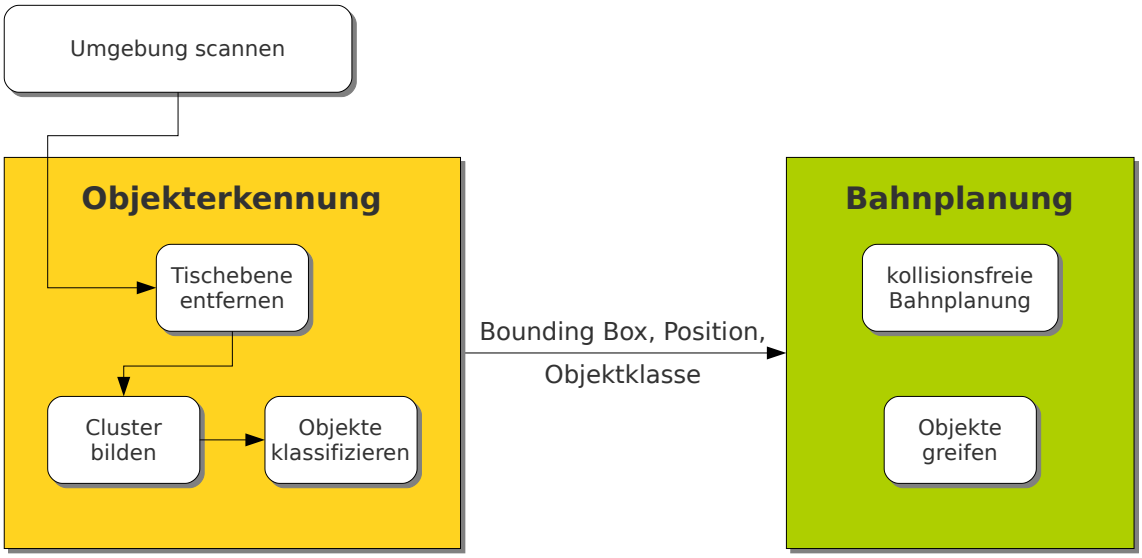


Abbildung 6.1: Komponenten und ihre Kommunikation.

In dieser Arbeit wird ein Verfahren zur *Objekterkennung* beschrieben, wobei die Arbeit von J. Brich [Bri09] die kollisionsfreie *Bahnplanung* umfasst. Die Information über die Größe der Objekte wird der *Bahnplanung* als minimaler Quader, der das Objekt einhüllt, übergeben. Die Informationen über Größe und Lage des Objektes werden an das für die *Bahnplanung* verwendete Framework *OpenRave* weitergegeben. Dieses plant anhand der Informationen die Bahnen und falls eine Bahn gefunden wurde, wird diese Bahn vom Katana-Manipulator ausgeführt. Die Performanz, sowie die Grenzen der *Bahnplanung*, werden in [Bri09] dargestellt.

Zusammen bilden die beiden vorgestellten Arbeiten ein einfaches System zur Mobilen Manipulation, mit dem grundlegende Alltagsprobleme gelöst werden können.

Kapitel 7

Ergebnisse

7.1 Bewertung der Laserdaten

Laserscanner werden in vielen Bereichen zur Umgebungserfassung eingesetzt. Der Laserscanner ist dabei gegenüber Änderungen des Umgebungs- und Fremdlichtes robust, da er selbst einen Laserstrahl aussendet, anhand dessen die Messung durchgefürt wird. Die ermittelten Entfernungsdaten werden jedoch durch andere Parameter beeinflusst. Eine ausführliche Betrachtung dieser Einflüsse auf den verwendeten Hokuyo URG-04LX findet sich in [OYB09]. Nachfolgend wird untersucht, wie diese Parameter die aufgenommenen Scans von unterschiedlichen Altagsgegenständen beeinflussen.

7.1.1 Einfluss des Blickwinkels

Ein Parameter, der den Laserscan eines Objektes beeinflusst, ist der Blickwinkel, aus welchem der Scan aufgenommen wurde (vgl. Abbildung 7.1). Nachfolgend wird dieser Einfluss anhand zweier Fälle untersucht.

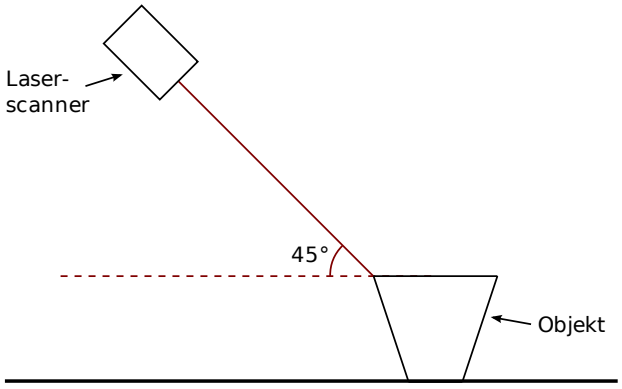


Abbildung 7.1: Laser-Blickwinkel von $45°$ auf ein Objekt.

Im ersten Fall wurde ein Becher aus einem Blickwinkel von ca. $35°$ mit dem Laser aufgenommen. Im 3D-Raum wird immer ein Teil des Objektes durch das Objekt selbst verdeckt, weshalb aus einem Blickwinkel nie ein komplettes Model des Objektes erstellt werden kann. Abbildung 7.2 zeigt diese Selbstverdeckung der Becherrückwand durch die Vorderseite. Dies führt dazu, dass bei komplexeren Objekten die extrahierten Merkmale aus unterschiedlichen Blickwinkeln, unterschiedliche Ergebnisse liefern und somit ein Objekt durch mehrere Merkmalsvektoren repräsentiert wird.

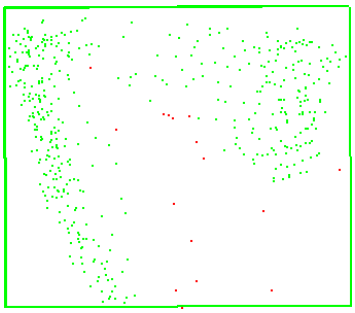
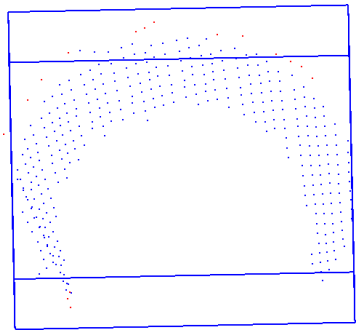


Abbildung 7.2: Selbstverdeckung der Becherrückwand (rechts) durch die Vorderseite.

Im zweiten Fall wurde eine Schüssel aus einem Blickwinkel von ca. $65°$ aufgenommen. Dieser Winkel entspricht dem Winkel der Schüsselvorderwand. Der Laserscanner bekommt, wie in Abbildung 7.3(a) zu sehen ist, nur die Kante der Wand, die wenige Millimeter dick ist, zu Gesicht. Aufgrund der Winkelauflösung des Hokuyo-Laserscanners von $0{,}36°$ (vgl. Kapitel 2.2.1) fällt die Kante zwischen zwei Punkte, oder der Laserstrahl wird nur teilweise reflektiert, wodurch kein sinvoller Messwert ermittelt werden kann und dieser somit verworfen wird. Dadurch wird, wie in Abbildung 7.3(b) zu sehen ist, die Vorderwand der Schüssel nicht erkannt, wodurch ein fehlerhaftes Modell der Schüssel entsteht. Um dieses Problem zu lösen, müsste die Szene aus mindestens zwei unterschiedlichen Perspektiven aufgenommen werden.

(a) Foto der Schüssel aus dem Blickwinkel des Laserscanners.



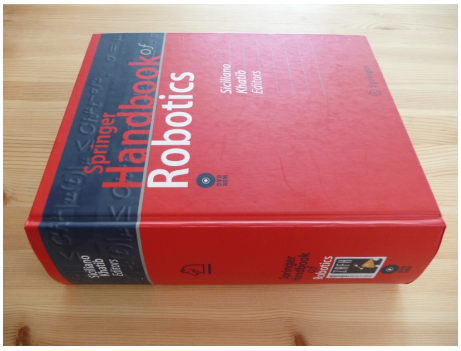
(b) Laserscan der Schüssel.

Abbildung 7.3: Einfluss des Blickwinkels auf die aufgenommen Schüssel.

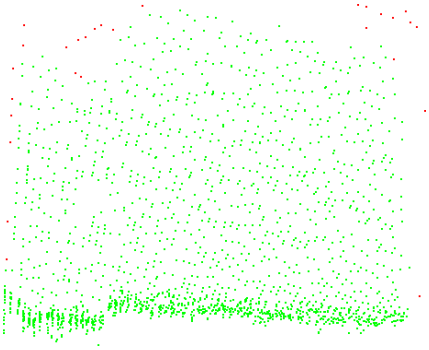
7.1.2 Einfluss der Objektoberfläche

Um den Einfluss der Oberflächenreflektivität zu untersuchen, wurde im ersten Versuch ein Buch, welches einen roten und dunkelblauen Einband (vgl. Abbildung 7.4(a) besitzt, gescannt. Es ist zu erwarten, dass für den dunkelblauen Streifen ein anderer Entfernungswert wie für den Rest des Buches ermittelt wird.
Diese Erwartung wurde, wie in Abbildung 7.4(b) zu sehen ist, bestätigt. Es ist deutlich zu erkennen, dass der dunkelblaue Teil des Buchrückens einen näheren Entfernungswert als der Rest des Buchrückens liefert. Daraus ergibt sich ein Modell, welches dem realen Objekt nicht genau entspricht. Hierdurch kann z. B. die Bahnplanung beeinflusst werden, da das Objekt größer erscheint und eine in

der Realität mögliche Bahn durch das Objekt verhindert wird. Die Objekterkennung wird durch das falsche Modell nicht beeinflusst, da das Objekt immer dieselben Merkmale aufweist.



(a) Buch mit dunkelblauem Streifen.

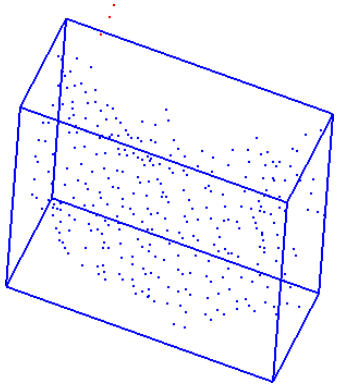


(b) Laserscan des Buches (von oben).

Abbildung 7.4: Einfluss der Materialfarbe auf die gemessene Entfernung.

Im zweiten Versuch wurde eine metallene Oberfläche, die den Laserstrahl vollständig reflektiert, untersucht. Dafür wurde, wie in Abbildung 7.5(a) zu sehen ist, der Boden einer Pringles Dose verwendet. Es wird erwartet, dass der Laserstrahl, der in einem Winkel von ca. $35°$ auf den Boden auftrifft, vollständig reflektiert wird und der Boden somit nicht erfasst werden kann.

(a) Dose mit reflektierendem Boden.



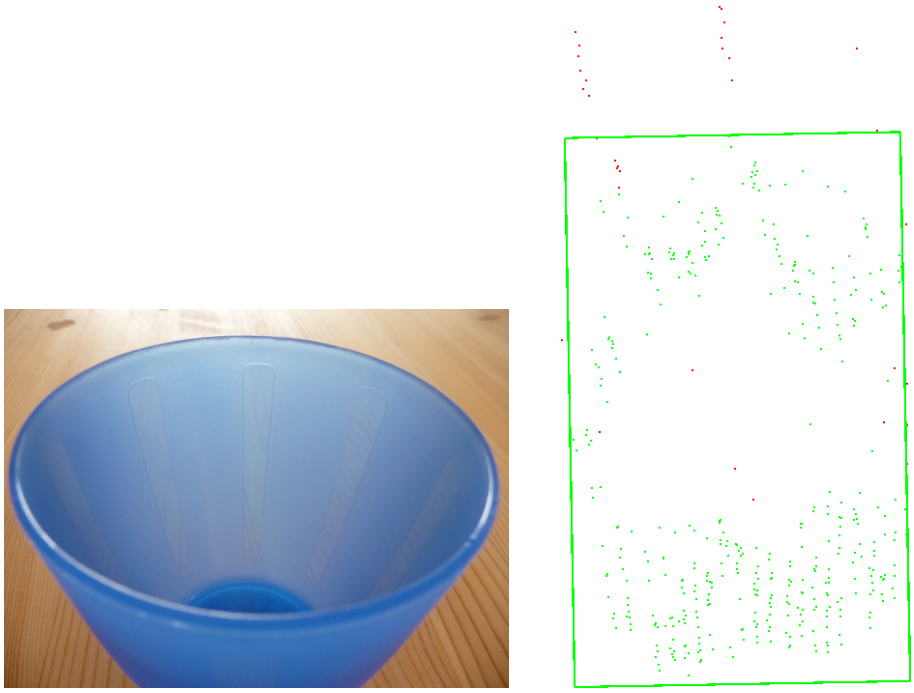
(b) Laserscan der Dose. Der Dosenboden wurde nicht erkannt.

Abbildung 7.5: Einfluss der Oberflächenreflektivität auf die gemessene Entfernung.

Abblidung 7.5(b) zeigt den aufgenommen Scan, in dem wie erwartet der Boden nicht erkannt werden konnte. Daraus ergibt sich, dass Objekte, die reflektierende Oberflächen aufweisen mit dem Laserscanner nicht zuverlässig erfasst werden können.

Einen weiteren Einfluss stellt die Lichtdurchlässigkeit dar. Hierfür wurde ein Becher[1] (vgl. Abbildung 7.6(a)), der teiltransparente Streifen besitzt, aufgenommen. Da diese Streifen den Laserstrahl nicht richtig reflektieren, wird ein Entfernungswert erwartet, der hinter dem Objekt liegt. Abbildung 7.6(b) zeigt den aufgenommenen Scan des Bechers.

[1] KALAS Becher von Ikea



(a) Becher mit transparenten Streifen.

(b) Laserscan des Bechers (von oben) *mit* transparenten Streifen.

Abbildung 7.6: Einfluss der Lichtdurchlässigkeit auf die Laserdaten.

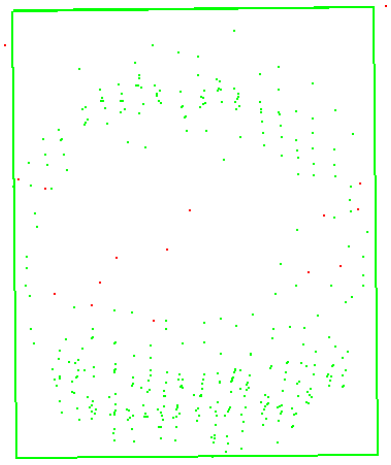


Abbildung 7.7: Laserscan des Bechers (von oben) *ohne* transparente Streifen.

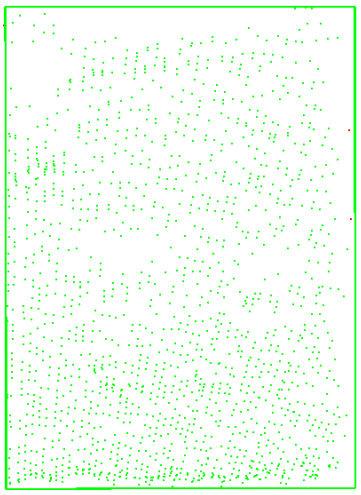
Anhand der Abbildung ist deutlich zu erkennen, dass der Becher in die Länge gezogen wird. Die transparenten Streifen liefern wie erwartet einen wesentlich höheren Entfernungswert (rote Punktlinien in Abbildung 7.6(b)) als der Rest des Bechers. Dies stellt ein ernsthaftes Problem dar, da vollständig transparente Objekte wie Flaschen oder Gläser durch den Laserscanner nicht erfasst werden können. Teiltransparente Objekte wie dieser Becher werden zwar erfasst, die Daten werden jedoch nicht konstant verfälscht, sondern sie sind bei jedem Scan anders. Aufgrund dessen ist eine sichere Objekterkennung ausgeschlossen.

Um dieses Problem zu lösen, wurde der Becher an der Innenseite mit einem Schleifpapier angeraut. Dadurch geht, wie in Abbildung 7.7 zu sehen ist, die Transparenz verloren, was zu einem wesentlich besseren Ergebnis führt.

7.1.3 Dichte der Scanpunkte

In diesem Versuch soll die Dichte der Scanpunkte, wie sie auf einer Ebene auftreten, untersucht werden. Aufgrund der Charakteristik von Laserscannern ist eine abnehmende Punktdichte des Scans mit der Entfernung zum Laserscanner zu erwarten. Des Weiteren ist von einer unregelmäßigen Verteilung der Punkte, aufgrund von Messungenauigkeiten, auszugehen. Als Ebene dient der Deckel eines Buches (vgl. Abblidung 7.8(a)). In Abbildung 7.8(b) ist der aufgenommene Scan des Buches zu sehen. Die Punktdichte nimmt wie erwartet von vorne nach hinten über den Buchdeckel ab. Des Weiteren ist zu erkennen, dass es Bereiche mit größerer und geringerer Punktdichte gibt. Dies ist auf die Messunsicherheit des Laserscanners zurückzuführen.
Diese Charakteristik birgt bei der Berechnung von Merkmalen, wie z. B. dem Schwerpunkt, ein Problem. Um dieses zu lösen, muss zuerst die Punktdichte des Objektes auf ein einheitliches Maß gebracht werden, was kein einfache Aufgabe darstellt.

(a) Aufgenommenes Buch.

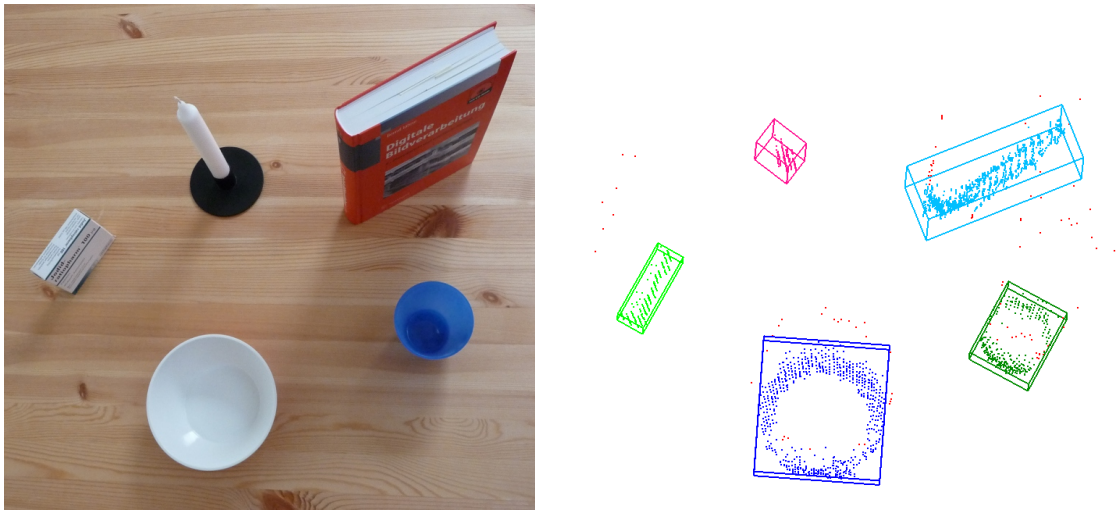


(b) Abnehmende Punktdichte von vorne nach hinten.

Abbildung 7.8: Abnehmende Punktdichte der Laserdaten.

7.2 Lernen von Objekten

Das Lernen von Objekten erfolgt wie in Kapitel 5.4.4 beschrieben anhand von mehreren Scans. Der Einfluss durch die Anzahl k der aufgenommenen Scans auf die quadratische Mahalanobis Distanz d^2 soll nachfolgend untersucht werden. Dafür wurde das in Abbildung 7.9(a) zu sehende Szenario mit einer *Schachtel*, einer *Schüssel*, einer *Kerze*, einem *Buch* und einem *Becher* verwendet. Alle in der Szene enthaltenen Objekte werden gegen die Becher-Objektklasse, die für die Werte von $k = 5,\ 10,\ 20,\ 50,\ 100$ ermittelt wurde, verglichen. Die *Erkennungssicherheit* soll gemäß den Anforderungen (Kapitel 5.1) bei einem Wert von 90% liegen, was einer quadratischen Mahalanobis Distanz von $d^2 = z_{(0,10;3)} = 0{,}584$ entspricht (vgl. Kapitel 5.4.4).



(a) Anordnung der Objekte auf dem Tisch.

(b) Laserscan des Szenarios.

Abbildung 7.9: Szenario zur Überprüfung des Einflusses der Anzahl der Scans.

Die quadratische Mahalanobis Distanz der Objekte zur Becher-Objektklasse ist für die Werte von k in der Tabelle 7.1 zu finden. Es ist zu erkennen, dass erst bei einem Wert von $k = 100$ die gewünschte *Erkennungssicherheit* von 90% erreicht wird. Diese große Anzahl an Scans und die damit verbundene Zeit für das Lernen eines Objektes durch den Benutzer ist für diesen nicht zumutbar.

k	*Schachtel*	*Schüssel*	*Kerze*	*Buch*	*Becher*
5	606,9	2136	640	12540	16,71
10	153,6	837,4	2014	2665	0,886
20	218,7	992,8	1587	2042	1,598
50	285,1	1236	1898	1697	0,864
100	245,1	906,2	1322	1898	0,488

Tabelle 7.1: Quadratische Mahalanobis Distanz der Objekte *Schachtel*, *Schüssel*, *Kerze*, *Buch* und *Becher* zur Becher-Objektklasse bei unterschiedlichem k.

Eine Alternative ist es, den Grenzwert für die Zuordnung eines Objektes zu einer Objektklasse nicht anhand der Chi-Quadrat-Verteilung zu ermitteln, sondern anhand der in der Datenbank enthaltenen Objektklassen. Im gegebenen Szenario ist aus der Tabelle zu erkennen, dass die Objekte schon bei

fünf Scans deutlich voneinander zu unterscheiden sind. Aus diesem Grund werden für die nachfolgende Erstellung der Objektdatenbank fünf Scans verwendet. Die Mahalanobis Distanz d für die Zuordnung eines Objektes zu einer Objektklassen wird anschließend anhand der erstellten Objektklassen ermittelt.

7.3 Erkennen von Objekten

Im Folgenden wird die im Kapitel 5 vorgestellte Objekterkennung auf ihre Einsatzfähigkeit in Alltagssituationen untersucht. Zuerst wird auf die verwendeten Objekte, sowie ihre Darstellung im Merkmalsraum eingegangen. Anschließend wird die Leistungsfähigkeit der Objekterkennung anhand von drei Szenarien überprüft. Zum Schluss werden die Grenzen des Systems anhand zweier Beispiele aufgezeigt.

7.3.1 Objektdatenbank

Für die Objektdatenbank wurden 6 Objekte, die im alltäglichen Leben vorkommen, ausgewählt. Dabei wurden 3 Objekte verwendet, die auch vom Katana-Manipulator gegriffen werden können. Die verwendeten Objekte sind in Abbildung 7.10 zu sehen.

Jedes Objekt wird durch fünf aufgenommene Scans in die Objektdatenbank eingefügt. Die Verteilung der einzelnen Objekte im Merkmalsraum ist in Abbildung 7.11(a) zu sehen. Die einzelnen Objektklassen sind deutlich voneinander getrennt, wobei die Objektklasse der *Dose* sowie der *Kerze* einen Ausreißer beinhalten. In Abbildung 7.11(b) ist die Ähnlichkeit, welche anhand des Erwartungsvektors der Objektklasse (linke Seite) mit der Wahrscheinlichkeitsdichtefunktion der anderen Objektklassen (oben), ermittelt wurde, zu erkennen. Für eine kompaktere Darstellung wurde für die Objektdatenbank nicht die quadratische Mahalanobis Distanz d^2 sondern die normale Mahalanobis Distanz d verwendet. In der Matrix ist dabei eine Mahalanobis Distanz von $d = 0$ als weiß und eine Mahalanobis Distanz größer/gleich $d \leq 100$ als schwarz kodiert.

In Tabelle 7.2 sind noch einmal die Mahalanobis Distanzen, die gleich wie bei der Ähnlichkeitsmatrix berechnet wurden, eingetragen. Es ist, wie in Abbildung 7.11(b), zu erkennen, dass jede Objektklasse mit sich selbst eine Mahalanobis Distanz von $d = 0$ besitzt, da der Erwartungsvektor genau im Zentrum der Wahrscheinlichkeitsverteilung liegt. Des Weiteren ist zu erkennen, dass die Werte nicht symmetrisch zueinander sind. So ist z. B. die Mahalanobis Distanz von *Becher/Kerze* = 163,95 und von *Kerze/Becher* = 28,29. Dies liegt an der unterschiedlichen Wahrscheinlichkeitsdichtefunktion der beiden Objektklassen, die anhand der Merkmalsvektoren berechnet wird. Aus der Tabelle ist abzulesen, dass die kleinste Mahalanobis Distanz d zwischen zwei Objektklassen $d = 12{,}33$ ist. Aus diesem Grund bietet sich ein Distanzwert von 10 als Grenzwert für die Zuordnung eines Objektes zu einer Objektklasse an.

	Becher	Flasche	Kerze	Dose	Schachtel	Schüssel
Becher	0	163,87	163,95	89,86	38,06	97,65
Flasche	45,04	0	21,83	19,59	39,02	88,11
Kerze	28,29	15,69	0	12,33	30,6	46,17
Dose	76,14	63,3	64,59	0	59,71	151,69
Schachtel	21,12	237,02	200,87	113,33	0	32,27
Schüssel	52,45	93,01	108,47	88,87	81,1	0

Tabelle 7.2: Gegenüberstellung der einzelnen Mahalanobis Distanzen.

(a) Red Bull Dose.

(b) 0,5 l PET-Flasche, von innen weiß eingefärbt.

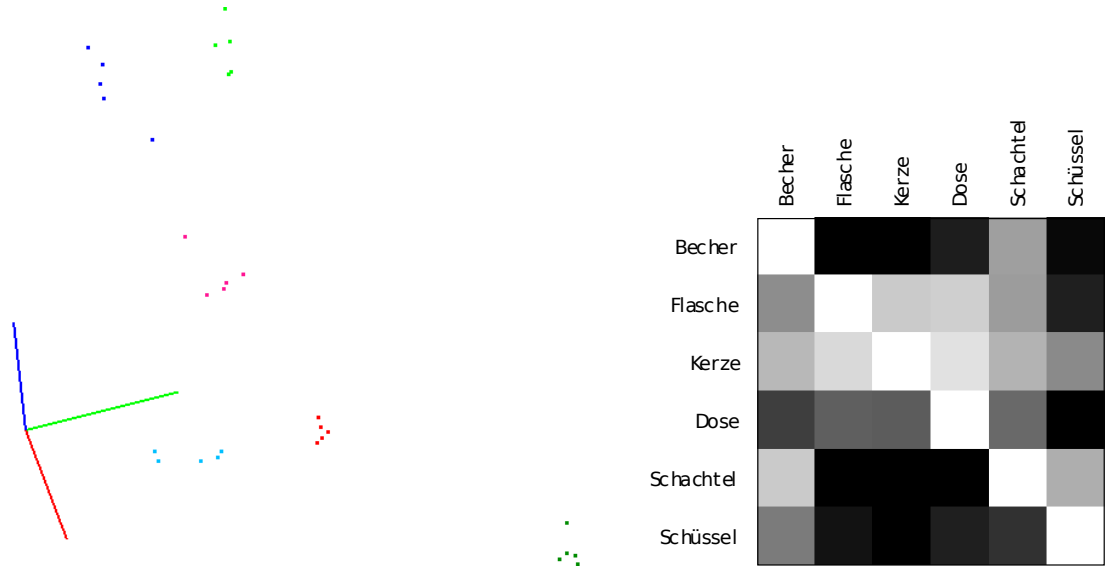
(c) VÄGHULT Kerzenständer von Ikea mit Kerze.

(d) FÄRGRIK TROLSK Schüssel von Ikea.

(e) KALAS Becher von Ikea.

(f) Tablettenschachtel.

Abbildung 7.10: Für die Datenbank verwendete Objekte.



(a) Im Merkmalsraum verteilte Objektklassen. rot = Becher, grün = Flasche; dunkelgrün = Schüssel, blau = Kerze, hellblau = Schachtel, pink = Dose

(b) Ähnlichkeitsmatrix der einzelnen Objektklassen.

Abbildung 7.11: Objektklassen im Merkmalsraum und als Ähnlichkeitsmatrix.

7.3.2 Szenarien

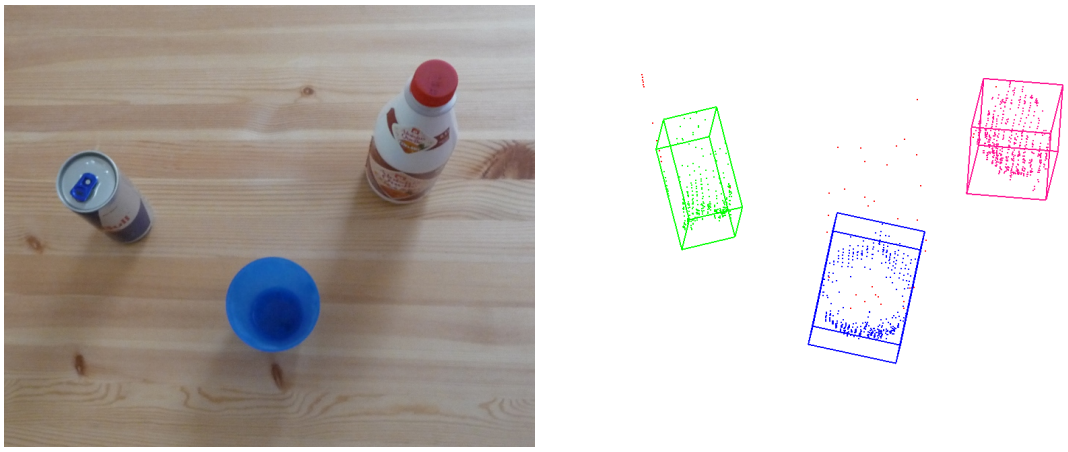
Es wurden drei Szenarien ausgewählt, in denen Objekte aus der Datenbank enthalten sind. Dabei wurde ein $73\,cm$ und ein $45\,cm$ hoher Tisch verwendet, auf dem die in der Datenbank enthaltenen Objekte beliebig positioniert wurden. Die Tischplatte befand sich, wie in der Anforderung beschrieben (vgl. Kapitel 5.1), in allen Szenarien parallel zur XY-Ebene des Basiskoordinatensytems. Ein Objekt wird als zu einer Objektklasse gehörend eingestuft, wenn seine Distanz zu dieser kleiner 10 ist.

Im Folgenden wird untersucht, ob die Objekterkennung in diesen Szenarien funktioniert und wo sie an ihre Grenzen stößt.

7.3.2.1 Hoher Tisch mit drei Objekten

Im ersten Szenario wurden eine *Dose*, ein *Becher* und eine *Flasche* beliebig auf einem $73\,cm$ hohen Tisch positioniert (vgl. Abbildung 7.12(a)). Der Scan des Tisches erfolgte aus einer Höhe von $26{,}5\,cm$ über der Tischplatte. Die Punktwolke wurde, wie in Abbildung 7.12(b) zu sehen ist, in drei Objekte segmentiert, wobei der *Becher* und die *Dose* gut von ihrer Bounding Box umschlossen werden. Das Cluster der *Dose* ist durch einige fehlerhafte Punkte in die Länge gezogen, was sich auch in der Mahalanobis Distanz niederschlägt (vgl. Tabelle 7.3).

In Tabelle 7.3 ist die Mahalanobis Distanz der Objekte im Szenario mit den Objektklassen der Datenbank eingetragen. Es ist zu erkennen, dass der *Becher* und die *Flasche* mit einer Distanz von $1{,}571$ und $3{,}44$ gut erkannt wurden. Die *Dose* liegt mit einer Distanz von $8{,}469$ im Vergleich zu den beiden anderen relativ nahe an der Grenze von 10. Dies liegt an den oben beschriebenen fehlerhaften Punkten, welche die Größe der Bounding Box verfälschen. Die Mahalanobis Distanz zu den anderen Objektklassen liegt bei den meisten Objekten deutlich oberhalb von 10, was eine große Distanz im Merkmalsraum bedeutet.



(a) Anordnung der Objekte auf dem Tisch.

(b) Laserscan des Szenarios.

Abbildung 7.12: Szenario mit *Dose*, *Becher* und *Flasche*.

	Dose	*Becher*	*Flasche*
Becher	80,91	1,571	150,9
Flasche	22,47	44,49	3,44
Kerze	14,81	28,14	12,11
Dose	8,469	74.93	51,94
Schachtel	93,63	22,34	212,8
Schüssel	90,71	52,7	94,63

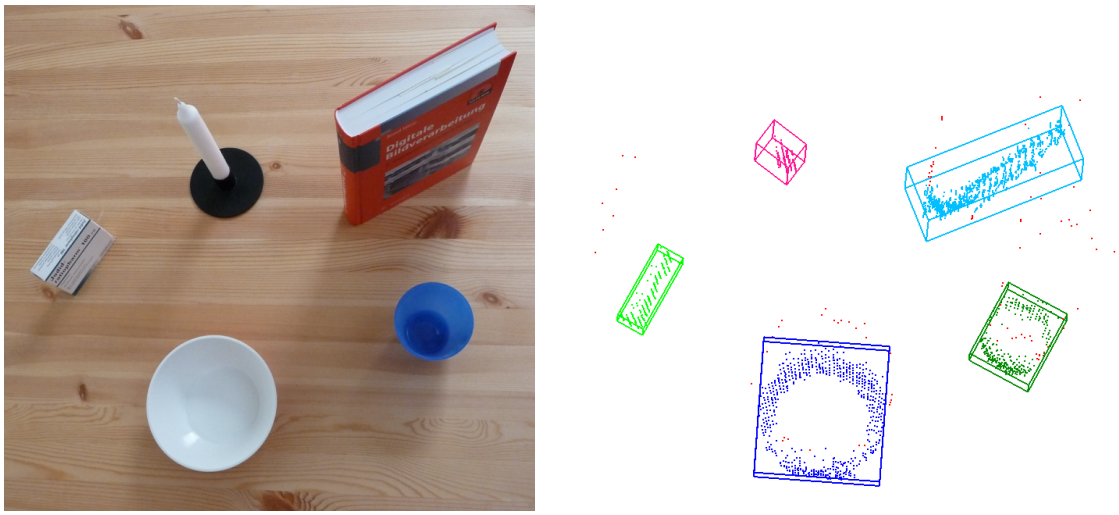
Tabelle 7.3: Mahalanobis Distanz der Objekte *Dose*, *Becher* und *Flasche* zu den Objektklassen.

7.3.2.2 Hoher Tisch mit fünf Objekten

Das zweite Szenario umfasst eine *Schachtel*, eine *Schüssel*, eine *Kerze*, ein *Buch* und einen *Becher* (vgl. Abbildung 7.13(a)). Es wurde derselbe Tisch und dieselbe Scanhöhe wie im ersten Szenario verwendet. Das *Buch* ist nicht in der Datenbank enthalten und sollte somit keiner Objektklasse zugeordnet werden.

In der segmentierten Punktwolke, siehe Abblidung 7.13(b), sind die einzelnen Objekte zu erkennen, wobei die *Kerze* und das *Buch* eine Auffälligkeit aufweisen. Bei der *Kerze* wurde der Kerzenständer nicht erkannt. Dies liegt an seiner niedrigen Höhe, die im Rauschen der Tischplatte untergeht und der Kerzenständer somit beim Entfernen der Tischplatte mitentfernt wurde. Am *Buch* ist zu erkennen, dass die Bounding Box das Objekt nicht immer optimal einhüllt. Die Ausrichtung der Bounding Box anhand der Eigenvektoren ist dafür verantwortlich, da die Diagonale eines Quaders die Hauptachse darstellt und die Bounding Box somit an dieser ausgerichtet wird.

Tabelle 7.4 zeigt die Mahalanobis Distanzen der Objekte mit den Objektklassen. Anhand einer Distanz kleiner 10 ist zu erkennen, dass alle Objekte, die in der Datenbank enthalten sind, der richtigen Objektklasse zugeordnet wurden. Das *Buch* wurde dabei, wie erwartet, keiner Objektklasse zugeordnet, da die Mahalanobis Distanz zu allen Objektklasse größer als 10 ist.



(a) Anordnung der Objekte auf dem Tisch.

(b) Laserscan des Szenarios.

Abbildung 7.13: Szenario mit *Schachtel*, *Schüssel*, *Kerze*, *Buch* und *Becher*.

	Schachtel	*Schüssel*	*Kerze*	*Buch*	*Becher*
Becher	40,97	96,78	163,5	165,1	7,933
Flasche	39,57	86,44	20,69	16,11	45,31
Kerze	31,59	45,73	0,5367	11,84	29,66
Dose	59,62	152,7	63,89	51,79	78,95
Schachtel	4,107	37,52	199	187,4	15,74
Schüssel	83,58	2,721	110,2	149,5	50,07

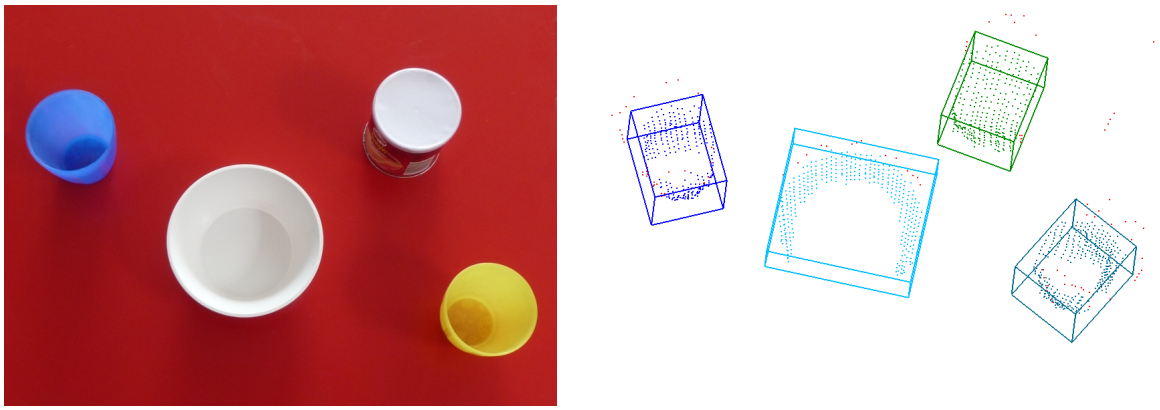
Tabelle 7.4: Mahalanobis Distanz der Objekte *Schachtel*, *Schüssel*, *Kerze*, *Buch* und *Becher* zu den Objektklassen.

7.3.2.3 Kleiner Tisch mit vier Objekten

Im dritten Szenario wurden zwei *Becher*, eine *Schüssel* und eine nicht in der Datenbank enthaltene *Pringles Dose* auf einem $45\,cm$ hohen Tisch plaziert (vgl. Abbildung 7.14(a)). Der Scan wurde aus einer Höhe von $54{,}5\,cm$ aufgenommen. Dieser Test soll zeigen, dass die Objekterkennung und vor allem das Erfassen der Umgebung nicht an eine bestimmte Tischhöhe gebunden ist. Des weiteren wird in diesem Szenario eine Grenze der Objekterkennung anhand der *Pringles Dose* aufgezeigt. Die Dose weist die gleichen Abmessungen wie der Becher auf, weshalb davon auszugehen ist, dass sie als Becher klassifiziert wird.

In Abbildung 7.14(b) ist die segmentierte Punktwolke mit den einzelnen Objekten zu erkennen. Auffällig dabei ist die *Schüssel*, da bei ihr die Vorderseite fehlt. Dies ist auf den Einfluss des Blickwinkels (vgl. Kapitel 7.1.1) zurückzuführen.

In Tabelle 7.5 finden sich die Mahalanobis Distanzen der Objekte zu den Objektklassen. Wie erwartet wurde die *Pringles Dose* als Becher erkannt (vgl. Kapitel 7.3.3.2). Die beiden *Becher* und selbst die unvollständige *Schüssel* wurden der richtigen Objektklasse zugeordnet.



(a) Anordnung der Objekte auf dem Tisch.

(b) Laserscan des Szenarios.

Abbildung 7.14: Szenario mit zwei *Bechern*, *Schüssel*, und *Pringles Dose*.

	Becher 1	*Schüssel*	*Pringles Dose*	*Becher 2*
Becher	8,632	89,21	6,927	5,51
Flasche	43,91	81,31	42,76	43,47
Kerze	29,13	44,21	27,78	28,22
Dose	75,97	148,4	71,07	72,97
Schachtel	15,96	39,43	26,83	25,07
Schüssel	53,16	8,558	53	51,74

Tabelle 7.5: Mahalanobis Distanz der Objekte *Becher 1*, *Schüssel*, *Pringles Dose* und *Becher 2* zu den Objektklassen.

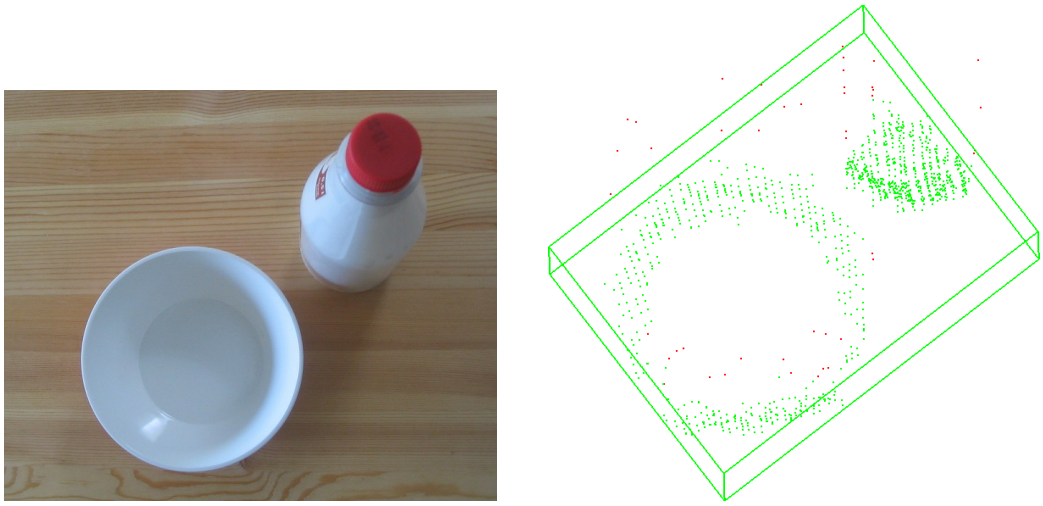
7.3.3 Grenzen der Objekterkennung

Es wurde gezeigt, dass die Objekterkennung in Alltagsszenarien anwendbar ist und ein brauchbares Ergebnis liefert. Das in Kapitel 7.3.2.3 beschriebene Szenario enthielt jedoch ein falsch klassifiziertes Objekt, was eine Grenze des vorgestellten Systems darstellt. Im Folgenden sollen die Grenzen des Systems und deren Bedeutung für die Objekterkennung betrachtet werden.

7.3.3.1 Abstand zweier Cluster

In diesem Szenario soll der Einfluss des Abstands zweier Objekte untersucht werden. Der Abstand beeinflusst dabei das Cluster-Verfahren, da hier die Objekte anhand ihres Abstands separiert werden. In Abbildung 7.15(a) ist das Szenario, bestehend aus einer *Schüssel* und einer *Flasche*, zu sehen. Der Abstand d zwischen den beiden Objekten beträgt $0{,}015\,m$. Es wird erwartet, dass die beiden Objekte durch ein Cluster repräsentiert werden, da der Abstand t für das Zusammenfassen von Clustern mit $t = 0{,}04\,m$ gewählt wurde (vgl. Kapitel 5.4.2.2).

In Abbildung 7.15(b) ist die geclusterte Punktwolke zu sehen. Wie erwartet werden die beiden Objekte durch ein Cluster repräsentiert. Für die Objekterkennung bedeutet dies, dass Objekte, die einen kleineren Abstand als $t = 0{,}04\,m$ aufweisen, nicht erkannt werden können und als Hindernis



(a) Anordnung der Objekte auf dem Tisch.

(b) Laserscan der beiden Objekte.

Abbildung 7.15: Schüssel und Flasche werden zu einem Cluster zusammengefasst.

klassifiziert werden. Diese Tatsache stellt für Alltagsszenarien jedoch keine Einschränkung dar, da Objekte, die gegriffen werden sollen, einen Mindestabstand von $0{,}05\,m$ zu anderen Objekten aufweisen müssen [Bri09].

7.3.3.2 Falsch klassifizierte Objekte

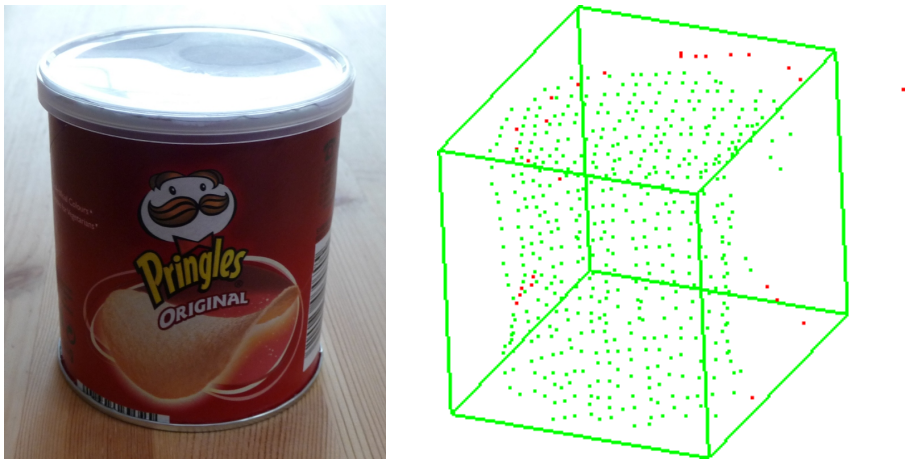
Ein entscheidendes Problem jeglicher Objekterkennung ist die falsche Klassifizierung von Objekten. Das vorgestellte System führt die Objekterkennung anhand der Abmessungen der Objekte durch. Aus diesem Grund wird für diesen Testfall eine *Pringles Dose*, welche dieselben Abmessungen wie der Becher besitzt, verwendet. Es wird erwartet, dass die Dose fälschlicherweise als Becher klassifiziert wird. Abbildung 7.16 zeigt die verwendete Dose, sowie deren Laserscan.

Die Mahalanobis Distanz der *Pringles Dose* zu den einzelnen Objektklassen der Datenbank ist in Tabelle 7.6 eingetragen. Die Dose besitzt eine Distanz von $4{,}336$ zur Objektklasse des Bechers, was unter die Grenze von 10 fällt. Aufgrund dessen wurde die Dose wie erwartet als Becher klassifiziert.

	Pringles Dose
Becher	4,336
Flasche	44,19
Kerze	27,91
Dose	72,6
Schachtel	30,77
Schüssel	49,74

Tabelle 7.6: Mahalanobis Distanz der *Pringles Dose* zu den Objektklassen.

Die Verteilung der beiden Objektklassen im Merkmalsraum ist in Abbildung 7.17 zu sehen. Deutlich zu erkennen ist, dass die Merkmalsvektoren der beiden Objekte nahe beieinander liegen wodurch sich fast zwei identische Objektklassen ergeben. Aus diesem Grund ist eine Unterscheidung der bei-



(a) Verwendete *Pringles Dose*. (b) Laserscan der *Prigles Dose*.

Abbildung 7.16: *Pringles Dose* mit denselben Abmessungen wie der Becher.

den Klassen nicht möglich. Die Einführung eines weiteren Merkmals und somit auch einer weiteren Dimension im Merkmalsraum würden dieses Problem lösen. Wichtig dabei ist natürlich, dass sich die beiden Klassen anhand dieses Merkmals gut unterscheiden lassen.

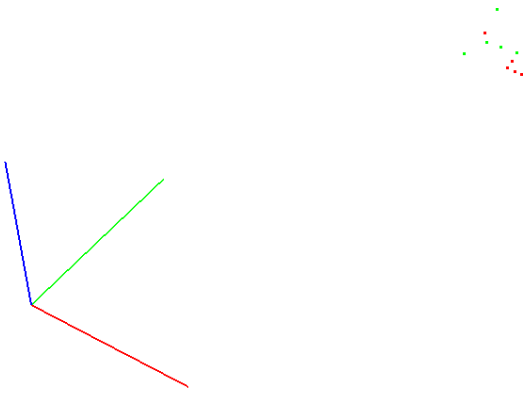


Abbildung 7.17: Im Merkmalsraum verteilte Objektklassen. rot = Becher, grün = *Pringles Dose*

7.3.3.3 Verdeckung durch andere Objekte

Ein weiteres Problem stellt die Verdeckung eines Objektes durch ein anderes dar. Um diesen Einfluss zu untersuchen, wurde das in Abbildung 7.18(a) abgebildete Szenario verwendet. Dabei wird der *Becher* teilweise durch die vor ihm stehende *Flasche* verdeckt, weswegen der *Becher* vom Laserscanner nur teilweise erfasst wird (vgl. Abbildung 7.18(b)). Aufgrund dessen wird erwartet, dass der *Becher* keiner Objektklasse zugeordnet werden kann.

Tabelle 7.7 zeigt die Mahalanobis Distanz der Objekte zu den einzelnen Objektklassen. Die *Flasche* wurde mit einer Distanz von $0{,}7125 \leq 10$ der richtigen Objektklasse zugeordnet. Der Becher konnte, wie erwartet, keiner Klasse zugeordnet werden, da die Distanz zu allen Klassen größer 10 ist.

(a) Anordnung der Objekte auf dem Tisch.

(b) Laserscan des Szenarios.

Abbildung 7.18: Durch *Flasche* verdeckter *Becher*.

	Flasche	*Becher*
Becher	166,2	48,31
Flasche	0,7125	41,43
Kerze	15,54	33,82
Dose	64,96	59,11
Schachtel	238,3	11,12
Schüssel	95,15	86,78

Tabelle 7.7: Mahalanobis Distanz der *Flasche* und des teilweise verdeckten *Bechers* zu den Objektklassen.

Daraus ergibt sich, dass Objekte, die durch andere Objekte verdeckt werden, nicht erkannt werden können. Diese Tatsache stellt in der gegebenen Aufgabenstellung jedoch keine Einschränkung dar, da wie in [Bri09] beschrieben, das Greifen von Objekten nur richtig möglich ist, wenn kein Hindernis den direkten Blick auf das Objekt versperrt.

7.3.4 Fazit

Anhand der beschriebenen Testfälle wurde gezeigt, dass die vorgestellte Objekterkennung in Alltagsszenarien funktioniert. Die Objektklassifizierung mittels Merkmalsraums und Mahalanobis Distanz erwies sich hierbei als zuverlässiges Kriterium, durch welches zudem eine qualitative Aussage über die Zuordnung getroffen werden kann. Unter Berücksichtigung der beschriebenen Grenzen ist das System für den Einsatz in einfachen Szenarien durchaus geeignet.

Kapitel 8

Zusammenfassung und Ausblick

8.1 Zusammenfassung

Das in dieser Arbeit vorgestellte Verfahren ist, wie Kapitel 7 zeigt, in einfachen Alltagszenarien einsetzbar. Dabei wird ein dreidimensionaler Laserscan der Umgebung mittels eines am Katana-Arm befestigten Laserscanners aufgenommen. In der erhaltenen Punktmenge wird zuerst die Tischplatte mit Hilfe des RANSAC-Algorithmus ermittelt und entfernt. Anschließend wird die verbleibende Punktmenge mitttels des DBSCAN-Verfahrens in Segmente aufgeteilt, die je ein Objekt repräsentieren. Die extrahierten Merkmale werden als Merkmalsvektor in einem dreidimensionalen Merkmalsraum aufgefasst. Die Zuordnung eines Objektes zu einer Objektklasse erfolgt dabei mittels der Mahalanobis Distanz.

Es wurde gezeigt, dass das eingesetzte Verfahren zur Erkennung der Tischplatte sowie dem Segmentieren der Punktwolke robust gegenüber Rauschen ist. Die Klassifizierung der Objekte anhand einfacher Merkmale erwies sich zusammen mit der Mahalanobis Distanz als ein zuverlässiges Verfahren. Daraus ergibt sich, dass das System eine ausreichende Erkennungsrate für primitive Szenarien bietet. Damit das System richtig funktioniert, muss die Tischplatte parallel zur XY-Ebene des Basiskoordinatensystems liegen und die Ojekte müssen einen Abstand zueinander von mindestens $5\,cm$ aufweisen.

Zusammen mit der von J. Brich [Bri09] vorgestellten kollisionsfreien Bahnplanung können Aufgaben wie das Abräumen eines Tisches zuverlässig gelöst werden. Diese Kombination stellt damit ein primitives System zur Mobile Manipulation dar, mit dem einfache Alltagsaufgaben gelöst werden können.

8.2 Ausblick

Das vorgestellte System stößt aufgrund der gewählten Merkmale bei Objekten mit den gleichen Abmessungen an seine Grenzen (vgl. Kapitel 7.3.3). Die Erweiterung um zusätzliche aus der 2D-Bildverarbeitung bekannten Merkmale stellt sich als schwierig dar, da sich diese nicht ohne weiteres auf den dreidimensionalen Fall anwenden lassen. Dies ist wohl auch der Grund, warum die in der Literatur beschriebenen Verfahren entweder die 3D-Repräsentation auf mehrere 2D-Bilder reduzieren, oder auf andere Merkmale wie die Oberflächenkrümmung zurückgreift.

Ein Verfahren zur Objekterkennung in 3D-Punktwolken sind die *Spin Images* [Joh97]. Diese Bilder sind relativ einfach anhand des Normalenvektors eines Punktes zu berechnen. Sie bieten dabei auch in Szenarien, in denen Objekte teilweise verdeckt sind eine gute Erkennungsrate. Kombi-

niert mit den in [ABBP07] vorgestellten *spin image signatures*, die den Speicheraufwand der *Spin Images* erheblich reduzieren und eine einfachere Objektklassifizierung erlauben, kann dieses Verfahren zur Verbesserung der Objekterkennung dienen. Die *spin image signatures* werden durch einen n-dimensionalen Vektor repräsentiert, welcher eine effiziente Suche in großen Datenbanken erlaubt. Durch diese Repräsentation ist eine einfach Integration in das eigene Verfahren möglich. Dafür muss der bestehende Merkmalsvektor lediglich um den Vektor der *spin image signatures* erweitert werden.

Ein anderer Ansatz, welcher auf Entfernungsbilder basiert, ist das in [HLLS01] vorgestellte Verfahren. Anhand der Entferungsbilder werden einfache Merkmale wie die Krümmung der Objektoberfläche ermittelt und als n-dimensionaler Vektor aufgefasst. Aufgrund dessen, kann auch dieses Verfahren einfach in das bestehende integriert werden. Zudem kann das Verfahren mit dem in [RSGB09] vorgestellte Verfahren zum Lernen von Objekten verbunden werden. Beide Verfahren beruhen auf Entfernungsbildern, weshalb die Bilder nur einmal aus der 3D-Punktwolke berechnet werden müssen.

Zusätzliche Informationen für die Objekterkennung können mit einer Kamera gewonnen werden. So lässt sich mit dieser die Farbe, sowie eventuelle Beschriftungen des Objektes erfassen. Aufgrund des verwendeten Merkmalsraumes können die so gewonnenen Merkmale, die eine weitere Dimension im Merkmalsraum darstellen, leicht in die bisherige Objektklassifizierung aufgenommen werden. Diese Erweiterung kombiniert die Vorteile der 2D und 3D-Objekterkennung miteinander, wodurch eine Objekterkennung auch in schwierigen Situationen ermöglicht wird.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[ABBP07] ASSFALG, J. ; BERTINI, M. ; BIMBO, A. D. ; PALA, P.: Content-Based Retrieval of 3-D Objects Using Spin Image Signatures. 9 (2007), April, Nr. 3, S. 589–599. http://dx.doi.org/10.1109/TMM.2006.886271. – DOI 10.1109/TMM.2006.886271

[Bri09] BRICH, J.: *Erkennen und Greifen von Alltagsgegenständen mittels Katana Manipulatorarm: Bahnplanung und -ausführung*. Hochschule Ulm, Bachelorarbeit, 2009

[BWS+09] BORST, C. ; WIMBÖCK, T. ; SCHMIDT, F. ; FUCHS, M. ; BRUNNER, B. ; ZACHARIAS, F. ; GIORDANO, P. R. ; KONIETSCHKE, R. ; SEPP, W. ; FUCHS, S. ; RINK, C. ; ALBU-SCHÄFFER, A. ; HIRZINGER, G.: Rollin' Justin - Mobile Platform with Variable Base. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, May 12-17*, 2009

[DJ97] DORAI, C. ; JAIN, A. K.: COSMOS-A Representation Scheme for 3D Free-Form Objects. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 19 (1997), Nr. 10, 1115–1130. http://dx.doi.org/10.1109/34.625113. – DOI 10.1109/34.625113. – ISSN 0162–8828

[EKJX96] ESTER, M. ; KRIEGEL, H. ; JÖRG, S. ; XU, X.: *A density-based algorithm for discovering clusters in large spatial databases with noise*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.71.1980. Version: 1996

[Fil05] FILZMOSER, P.: Identification of Multivariate Outliers: A Performance Study. In: *Austiran Journal of Statistics* Bd. 34. 2005, S. 127–138

[HKT01] HAN, Jiawei ; KAMBER, Micheline ; TUNG, Anthony K. H.: Spatial Clustering Methods in Data Mining: A Survey. In: MILLER, Harvey J. (Hrsg.) ; HAN, Jiawei (Hrsg.): *Geographic Data Mining and Knowledge Discovery, Research Monographs in GIS*, Taylor and Francis, 2001

[HLLS01] HETZEL, G. ; LEIBE, B. ; LEVI, P. ; SCHIELE, B.: 3D object recognition from range images using local feature histograms. In: *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2001* Bd. 2, 2001, S. II–394–II–399

[HP-09a] *https://mysick.com/saqqara/wrapper.aspx?id=im0006435*. August 2009

[HP-09b] *http://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx.html*. August 2009

[JÖ5] JÄHNE, B.: *Digitale Bildverarbeitung*. 6. Springer, 2005 http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/3540412603. – ISBN 9783540249993

[JH99] JOHNSON, A. E. ; HEBERT, M.: Using spin images for efficient object recognition in cluttered 3D scenes. 21 (1999), May, Nr. 5, S. 433–449. http://dx.doi.org/10.1109/34.765655. – DOI 10.1109/34.765655

[Joh97] JOHNSON, A.: *Spin-Images: A Representation for 3-D Surface Matching*. Pittsburgh, PA, Robotics Institute, Carnegie Mellon University, Diss., August 1997

[JTLB04] JAIN, A. K. ; TOPCHY, A. ; LAW, M. H. C. ; BUHMANN, J. M.: Landscape of clustering algorithms. In: *Proc. 17th International Conference on Pattern Recognition ICPR 2004* Bd. 1, 2004, S. 260–263

[KD92] KOENDERINK, J. ; DOORN, A. van: Surface shape and curvature scales. In: *Image and Vision Computing* 10 (1992), October, Nr. 8, 557–564. http://dx.doi.org/10.1016/0262-8856(92)90076-F. – DOI 10.1016/0262–8856(92)90076–F

[KWB08] KLASING, K. ; WOLLHERR, D. ; BUSS, M.: A clustering method for efficient segmentation of 3D laser data. In: *Proc. IEEE International Conference on Robotics and Automation ICRA 2008*, 2008, S. 4043–4048

[KWB09] KLASING, K. ; WOLLHERR, D. ; BUSS, M.: Realtime Segmentation of Range Data Using Continuous Nearest Neighbors. In: *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*. Kobe, Japan, 2009

[Mac67] MACQUEEN, J. B.: Some Methods for Classification and Analysis of MultiVariate Observations. In: CAM, L. M. L. (Hrsg.) ; NEYMAN, J. (Hrsg.): *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* Bd. 1, University of California Press, 1967, S. 281–297

[NH04] NISCHWITZ, A. ; HABERÄCKER, P.: *Masterkurs Computergrafik und Bildverarbeitung*. 1. Auflage. Vieweg, 2004

[OYB09] OKUBO, Y. ; YE, C. ; BORENSTEIN, J.: Characterization of the Hokuyo URG-04LX Laser Rangefinder for Mobile Robot Obstacle Negotiation, 2009

[Pap06] PAPULA, L.: *Mathematische Formelsammlung*. 9. Auflage. Vieweg-Verlag, 2006

[Pau08] PAULUS, J.: *Vision Based Mobile Manipulation*, Fachhochschule Bonn-Rhein-Sieg, Diplomarbeit, 2008

[RBB09] RUSU, R. B. ; BLODOW, N. ; BEETZ, M.: Fast Point Feature Histograms (FPFH) for 3D Registration. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, May 12-17*, 2009

[RMBB08a] RUSU, R. B. ; MARTON, Z. C. ; BLODOW, N. ; BEETZ, M.: Learning informative point classes for the acquisition of object model maps. In: *Proc. 10th International Conference on Control, Automation, Robotics and Vision ICARCV 2008*, 2008, S. 643–650

[RMBB08b] RUSU, R. B. ; MARTON, Z. C. ; BLODOW, N. ; BEETZ, M.: Persistent Point Feature Histograms for 3D Point Clouds. In: *Proceedings of the 10th International Conference on Intelligent Autonomous Systems (IAS-10), Baden-Baden, Germany*, 2008

[RSGB09] RUHNKE, M. ; STEDER, B. ; GRISETTI, G. ; BURGARD, W.: Unsupervised Learning of 3D Object Models from Partial Views. In: *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*. Kobe, Japan, 2009

[SM92] STEIN, F. ; MEDIONI, G.: Structural indexing: efficient 3-D object recognition. 14 (1992), Feb., Nr. 2, S. 125–145. http://dx.doi.org/10.1109/34.121785. – DOI 10.1109/34.121785

[TSK05] TAN, P. ; STEINBACH, M. ; KUMAR, V.: *Introduction to Data Mining*. US ed. Addison Wesley, 2005. – ISBN 0321321367