Karlsruher Institut
für Technologie (KIT)
Institut für Prozessrechentechnik,
Automation und Robotik (IPR)
der Fakultät für Informatik

Markerbasierte Objektverfolgung für die Mensch-Roboter-Kooperation

Diplomarbeit
von
Beibei Cao

Stand: 21. Juni 2012

Referenten: Prof. Dr.-Ing. Heinz Wörn
Betreuer: Dipl.-Inform. Stephan Puls

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Roboter haben ihre Stärke in der Wiederholung von einfachen Handhabungstätigkeiten. Menschen dagegen sind mit ihren kognitiven Fähigkeiten einzigartig, etwa mit ihrem Verständnis der Aufgabe. Die Kombination von Mensch und Roboter kann Aufgaben stark rationalisieren, sofern jedem die optimalen Arbeitsteile zugewiesen sind. Die Anwendungsbereiche der Mensch-Roboter-Kooperation vergrößern sich heutig immer schneller auf dem Feld der Medizin sowie der Industrie. Damit Mensch und Roboter in einer geringen Entfernung sicher und effizient zusammenarbeiten können, ist die Erkennung bzw. die Verfolgung von Menschen und Objekten für ein Mensch-Roboter-Kooperation-System notwendig. Die Menschenerkennung garantiert die Sicherheit für den Menschen und liefert gleichzeitig Informationen über die Blickrichtung, um Aussagen über die Aufmerksamkeit des Menschen treffen zu können. Die Objekterkennung vereinfacht die Kommunikation zwischen Mensch und Roboter, dadurch können die Fremdobjekte ohne weitere Programmierung direkt vom Roboter erkannt werden. Außerdem vermeidet die Objektverfolgung auch die Kollision zwischen Roboter und anderen Anlagenteilen.

1.2 Aufgabenstellung

Das Rahmenwerk MAROCO wird am IPR entwickelt, damit Menschen und Roboter in einer gemeinsamen Umgebung sicher zusammenarbeiten können. Die Erfassung des Mensch und Handlungsanalyse in der Szene erlaubt es, die Gefahr von der Roboterbewegung für Menschen zu minimieren. Jedoch ist die Erkennung zurzeit auf das Menschmodell beschränkt. Alle anderen Objekte werden von dem System als Zylinder dargestellt. Das Ziel der Arbeit ist, die verschiedenen Objekte zu kalibrieren und die entsprechenden geometrischen Charakteristika in dem System zu speichern, dann die

Objekte mit Hilfe der gespeicherten Informationen zu erkennen und zu verfolgen. Beide Schritte sollen in Echtzeit durchgeführt werden.

Folgende Ziele sollen erreicht werden:

- Objektkalibrierung,
- Darstellung und Speichern des charakteristischen Modells der Objekte,
- Objekterkennung und Verfolgung,
- Einhaltung der Echtzeitbedingungen (Framerate $\geq$ 30fps).

Kapitel 2

Stand der Forschung

3D Objekterkennung und Verfolgung kommt in vielen Anwendungsbereichen zum Einsatz. Daher wurden viele Algorithmen bzw. Systeme dafür in den vergangenen Jahrzehnten entwickelt. Ein gutes Beispiel ist das kommerzielle Erkennungssystem von VICON [VICON]. In dem System werden 8 Kameras benutzt, die von verschiedenen Richtungen das Zielobjekt bzw. Person beobachten. Einige weiße Marken werden vorher am Ziel angebracht, damit seine Positionen und Bewegungen von den Kameras gut erkannt werden können. Nach Vergleichen der Bilder von verschiedenen Kameras kann ein 3D Modell des Ziels in Echtzeit erzeugt werden. Das System wird im Bereich von Computerspielen und Filmindustrie sehr oft benutzt. Ein anderes Beispiel ist das neue Gerät Kinect von Microsoft XBox360 [Kinect]. Eine 3D Kamera kann die 3D Daten von Spielern ansammeln, damit die Spieler das Spiel direkt mit ihren Körpern statt des traditionellen Kontroller steuern können. Die Analyseverfahren der Objekterkennung basieren auf unterschiedlichen Charakteristika der Objekte und sind für verschiedenen Typen von Objekten geeignet. In der Arbeit von Lepetit und Fua sind die aktuelle Verfolgungsverfahren in zwei große Gruppen unterschieden worden: auf Marken basierte Objektverfolgungen und auf natürlichen Merkmale basierte Objektverfolgungen. Die Verfahren in der zweiten Gruppe können weiter in kantenbasiertes Verfahren, optischer Fluss basiertes Verfahren, Templatebasiertes Verfahren, Punktebasiertes Verfahren und das SLAM-Verfahren unterteilt werden [Lepetit & Fua 2005]. Im folgenden Abschnitt werden kurz die Details jedes Verfahrens erklären.

2.1 Standardmarkenbasierte Verfahren

Die Verfolgungsverfahren können in zwei Schritte unterteilt werden: zuerst der Informationen von Bildsequenzen ansammeln um dann die Position des erkannten Objekts zu bestimmen. Die vordefinierte Marken können in beiden Schritten mehr Information liefern, damit die Objekte schneller und einfacher verfolgt werden können. Deshalb sind in diesem Bereich viele Systeme für die Erweiterte Realität implementiert worden. Ein

Echtzeitsystem für Erweiterte Realität wurde von Zhang und Navab für Objektverfolgung in einer Industrieumgebung realisiert [Zhang & Navab 2000]. Sie haben eine Gruppe von 4 Vierecken als eine Marke benutzt. Die Marke wird durch Farbe und weiße Flecken innerhalb der Vierecke kodiert.

Ein anderes System heißt ARToolKit, was vom HITLab der Universität Washington entwickelt wird. Es ist eine bekannte Software-Bibliothek zur Entwicklung von Anwendungen für die Erweiterte Realität [ARToolKit]. In ARToolKit wird ein Viereck mit schwarzer Umrandung als Marke benutzt. Das Muster in der Mitte kodiert die Marke und kann frei gewählt werden. Das Eingabebild wird zuerst in ein Binärbild umgewandelt und dann alle verbundenen schwarzen Pixel extrahiert. Die Figur innerhalb der schwarzen Umrandung wird segmentiert und mit dem früheren definierten Muster verglichen. Durch den Vergleich kann man die Projektivität zwischen Kamerakoordinatensystem und Musterkoordinatensystem bestimmen.

ARToolKit liefert eine hohe Frame-Rate mit bis zu 30 fps bei niedrigem CPU-Bedarf. Eine dicke schwarze Umrandung garantiert die Stabilität des Systems und die Marke kann in niedriger Auflösung sehr gut erkannt werden. Ein anderer wichtige Vorteil ist, dass die Verfolgung der ARToolKit keine Initialisierung braucht. Dadurch wird nicht nur die Laufzeit am Anfang des Verfahren gespart, kann aber auch Chaos vermeiden, wenn die eingegebene Bildsequenz abgebrochen wird.

2.2 Verfahren basierend auf natürlichen Merkmalen

2.2.1 Kantenbasiertes Verfahren

Das kantenbasierte Verfahren wurde in früheren Objektverfolgungssystemen häufig benutzt, weil es effizient und einfach zu realisieren ist [Lepetit & Fua 2005]. Die Hauptidee dieses Verfahrens ist entweder die Kanten des Objekts direkt von dem Bild herauszufinden und zu verfolgen, oder den Teil des Bildes mit starkem Gradient zu betrachten, damit man die Konturen des Objekts zum nächsten Zeitpunkt vorhersagen kann. RAPiD war eines der frühesten 3D Verfolgungsverfahren, das in Echtzeit laufen konnte [Harris 1992]. Vacchetti und Lepetit haben ein neues, effizienteres Verfolgungsverfahren entwickelt, was mehr als eine Voraussagen für die ausgewählten Steuerpunkte darstellen [Vaccetti, Lepetit & Fua 2004]. Diese Erweiterung verstärkt die Stabilität der Verfolgung und erfüllt weiterhin die Echtzeitbedingung.

2.2.2 Optischer Fluss basiertes Verfahren

Der Optische Fluss ist ein Vektorfeld, das die Bewegungsrichtung und Bewegungsgeschwindigkeit für jeden Bildpunkt einer Bildsequenz bezeichnet. Die Berechnung des Optischen Fluss kann als eine Differentialgleichung zusammengefasst werden und das Lösungsverfahren wurde von Horn und Schunck entwickelt [Horn & Schunck 1981]. Black und Yacoob benutzten reine Optische Fluss basierte Verfahren für die Verfolgung kleiner Veränderungen auf menschlichem Gesicht, um den Gesichtsausdruck zu bestimmen [Black & Yacoob 1997]. Außerdem wurde ein Verfolgungssystem für den Innerstadt Verkehr von Haag und Nagel durch die Verknüpfung der Information von Optischem Fluss und Kanten des Objekts implementiert [Haag & Nagel 1999].

2.2.3 Templatebasiertes Verfahren

Im Templatebasierten Verfahren wird ein Objekt nicht durch lokale Merkmale z.B. Kanten oder Punkte, sondern durch das globale Charakteristikum erkannt und verfolgt. Das Verfahren ist geeignet für komplexe Objekte, die nicht einfach durch lokale Merkmale bezeichnet werden können [Lepetit & Fua 2005]. Der Lucas-Kanade Algortihmus wurde anfänglich entwickelt, um den Optischen Fluss zu berechnen [Lucas & Kanade 1981], ist aber auch für die 2D templatebasierte Verfolgung nutzbar. Jurie und Dhome haben einen Algorithmus für die Verfolgung von ebenen Objekten mithilfe von Hyperebenen entwickelt [Jurie & Dhome 2001]. In ihrer Arbeit wurde die Approximation der Abbildung des Objekts auf Hyperebenen abgeschätzt, dadurch die Translation des Objekts bestimmt werden kann.

2.2.4 Punktebasiertes Verfahren

Der Unterschied zwischen dem punktebasierten Verfahren und den oben beschriebenen Verfahren ist, dass nur lokale Merkmale betrachtet werden. Im Vergleich zum Verfahren, das globale Merkmale behandelt, ist das Verfolgungsverfahren mit lokalen Merkmalen viel stabiler, wenn es Kollision für mehr Objekte gibt, oder die Messung der Merkmalen stark stört wird [Lepetit & Fua 2005]. Ein Verfahren wurde von Zhang et al. im Jahre 1995 realisiert, was die nicht kalibrierten Bilder als Eingabe benutzen kann [Zhang et al. 1995]. In dem Verfahren wird kein Modell der Epipolargeometrie verwendet, wodurch viele komplexen Berechnungen vermieden werden. Eine andere Möglichkeit für die Punkteverfolgung ist der Kanade-Lucas-Tomasi(KLT) Tracker, was auf der Arbeit von [Lucas & Kanade 1981] begründet wurde. Sie haben eine Approximation für den Unterschied zwischen zwei Bildern definiert. Mithilfe des Iterationsverfahrens von Newton-Raphson wird die Approximation minimiert. Dadurch kann die Translation des Objekts bestimmt werden. Tomasi erweiterte den Algorithmus von Lucas und Kanade mit einer besseren Strategie zur Auswahl von Merkmalen [Tomasi & Kanade]. Der dritte Schritt wurde von Shi und Tomasi vervollständigt [Shi & Tomasi 1994]. Sie verbesserten

weiter die Auswahl der Punkte mit der Ähnlichkeit zwischen dem Anfangsbild und das aktuelle Bild. Diese Ähnlichkeit wird durch einem Modell von affiner Abbildung bestimmt. Außerdem benutzen sie gleichzeitig zweites Modell von reiner Translation, um das Objekt mit hoher Seriosität und Präzision zu verfolgen.

2.3 Markenbasierte Objekterkennung

Rhijn und Mulder haben ein markenbasiertes Verfahren entwickelt, was das Verdeckungsproblem behandelt [Rhijn & Mulder 2005]. Einige runde, hoch reflektierende Marken werden auf dem Objekt angebracht. In der Initialisierungsphase speichert das System die Charakteristika des Objekts als einen 3-dimensionalen vollständigen Graph. Wenn das Objekt wiedererkannt werden soll, führt das System zuerst einen Kalibrationsalgorithmus durch, um die verschiedenen Objekte zu differenzieren. Dann vergleicht das System für jedes Objekt die sichtbaren Marken mit dem in der Initialisierungsphase gespeicherten vollständigen Graphen.

2.3.1 Markenerkennung

Es gibt heutzutage viele benutzte Standardalgorithmen der Markenerkennung. Die Harris Matrix wird von der Summe der partiellen Ableitungen des Eingabebildes dargestellt. Eine Ecke wird genau dann als ein Merkmal erkannt, wenn die beiden Eigenwerte der Harris Matrix die positive und genug groß sind [Harris & Stephens 1988]. Ein anderer Algorithmus für diese Eckenerkennung heißt ,,Features from Accelerated Segment Test"(FAST) und wurde von Rosten und Drummond im Jahre 2006 veröffentlicht [Rosten & Drummond 2006]. Außer der Eckenerkennung können die Marken sich aber auch als den Klecks annähern. Das Ziel dieser Art der Verfahren ist, die Punkte zu extrahieren, die unterschiedliche Eigenschaften zu der Umgebung haben, z.B Helligkeit und Farbe. Lowe hat seinen Algorithmus ,,Scale-invariant feature transform"(SIFT) im Jahre 2004 veröffentlicht, was starke Stabilität gegen Transformation, Beleuchtungsvariation bzw. Bildrauschen hat [Lowe 2004]. Bay et al. erweiterte die Arbeit von Lowe mit dem ,,Speeded Up Robust Feature"(SURF) [Bay et al. 2006]. Mithilfe des Integralbild wird der Algorithmus stark beschleunigt. Der Algorithmus ,,Center Surround Extremas"(CenSurE oder STAR) benutzt sogenannten ,,Center-surround" Filter, um den Laplace-Operator zu approximieren, damit die Betrachtung der Merkmale in allen Skalar-Räumen schnell durchgeführt werden kann [Agrawal, Konolige & Blas 2008]. Wegen diesen Veränderungen ist der CenSurE Algorithmus viel effizienter und stabiler. Die vorhandenen Realisierung obiger Algorithmen können in der freien Programmbibliothek OpenCV gefunden werden. Anhand von OpenCV hat Odessa in seinem Blog die Geschwindigkeit, Stabilität bzw. die Genauigkeit dieser Algorithmen verglichen [Odessa 2011]. Seinem Ergebnis zufolge scheint, dass der STAR Algorithmus die niedrigste durchschnittliche Fehler-Rate hat bzw. den geringsten Berechnungsaufwand

benötigt.

2.3.2 Markenverfolgung

Nach Bestimmung der Marken, sollen diese Marken in einer Bildsequenz verfolgt werden. Scott und Longuet-Higgins haben einen eleganten und einfachen Algorithmus entwickelt [Scott & Longuet-Higgins 1991]. Sie haben das Problem zu einer Matrix zusammengefasst, deren Elemente als die Distanz zwischen verschiedenen Merkmalen definiert werden. Durch eine Singulärwertzerlegung und Matrixersetzung werden die Abbildungen der Marken zwischen zwei Bildern bestimmt. Rhijn und Mulder verbesserten den Algorithmus von Scott und Longuet-Higgins. Sie haben die Elemente der Matrix neu definiert und fügten die Beschränkung der Epipolargeometrie ein [Rhijn & Mulder 2005].

2.3.3 Schätzung der Transformation

Für ein 3D Objekt ist die Markenverfolgung innerhalb von Bildern nicht ausreichend, um die komplette geometrische Information zu rekonstruieren. Deshalb soll die Pose des Objekts gleichzeitig bestimmt werden, damit ein räumlicher charakteristischer Graph für das Objekt erstellt werden kann. Natürlich kann man mit Hilfe der Ergebnisse der Markenverfolgung die relative Rotation und Translation des Objekts zwischen zwei Zeitpunkten berechnen, aber außerdem gibt es Verfahren, die durch Vergleich der Punktwolken von zwei Bildern direkt die Transformation des Objekts bestimmen können. Diese Verfahren wurden von Eggert et al. in ihrer Arbeit durch Merkmale und Lösungsverfahren in verschiedene Typen unterteilt [Eggert, Lorusso & Fisher 1997]. Die Merkmale könnten die Oberfläche, die Kanten bzw. die Punkte sein. Die Verfahren, die auf Punkte basieren, werden in der Praxis häufig benutzt und sind geeignet für die markenbasierte Objekterkennung [Eggert, Lorusso & Fisher 1997]. Die Lösungsverfahren können in iterative Verfahren und geschlossene Form unterschieden werden.

2.3.3.1 Geschlossene Form

Ein effizientes Verfahren mit geschlossener Form für das Berechnen der Transformation wurde von Arun et al. zuerst am Jahr 1987 veröffentlicht [Arun, Huang & Blostein 1987]. Die Hauptidee des Verfahrens ist, eine approximierte Rotations- bzw. Translationsmatrix zwischen zwei aufeinander folgenden Bildern zu bestimmen, damit die Summe des Unterschieds zwischen den durch approximierte Transformation berechnete Positionen der Punkte und die genaue Positionen der Punkte minimiert wird. Dieses Verfahren basiert auf der Singulärwertzerlegung einer Korrelationsmatrix. Die Rotations- und Translationsmatrix werden am Ende ausgegeben. Wenn die zwei eingegebenen Punktwolken auf gleich Oberfläche liegen, liefert das Verfahren leider keine richtige Rotationsmatrix. Deshalb soll eine korrigierte Matrix darauf aufbauen, die

von Umeyama [Umeyama 1991] und Kanatani [Kanatani 1994] vorschlagen wurde. Ein anderes Verfahren wurde von Horn entwickelt, was die relative Rotation durch Einheitsquaternionen beschreibt [Horn 1987]. Im Vergleich zu der Standard-Beschreibung der Rotation als Matrix ist die Quaterniondarstellung viel effizienter und stabiler. Die verbesserte Stabilität kann den Fehler vermeiden, wenn der Winkel der Rotation zur Singularität wird, z.B. 0° oder 180°. D.h. dieses Verfahren braucht keine Maßnahme für Behandlung des speziellen Winkels, was aber im Verfahren von Arun nötig ist [Arun, Huang & Blostein 1987].

Eggert et al. haben in ihrer Arbeit die obengenannten zwei Verfahren mit zwei anderen geschlossene Form-Verfahren verglichen [Eggert, Lorusso & Fisher 1997]. Wenn die Anzahl der betrachteten Punkte weniger als 100 ist, braucht das Verfahren mit Einheitsquaternionen weniger Zeit als die anderen Verfahren. Auf der anderen Seite, wenn die Anzahl der betrachteten Punkten weniger als 10 ist, liefert das Verfahren von Arun den kleinsten Fehler.

2.3.3.2 Iterative Closest Point Algorithmus

Der Iterative Closest Point Algorithmus ist ein Algorithmus, der es ermöglicht, Punktwolken aneinander anzupassen [ICP Wiki]. In jedem Iterationsschritt wird der korrespondierende Punkt für jeden Punkt einer Punktwolke aus anderer Punktwolke gefunden. Die Transformation zwischen beiden Punktwolken werden so bestimmt, dass die Summe des Abstands der korrespondierenden Punkte minimiert wird. Dieser Vorgang wird wiederholt, bis die Veränderung des mittleren quadratischen Fehler zwischen zwei folgenden Schritten unter einer Schranke liegt. Der Algorithmus wurde erst von Chen und Medioni [Chen & Medioni 1991] vorgestellt und ICP wurde als Name des Algorithmus von Besl und McKay in ihrer Arbeit erstmals benutzt [Besl & McKay 1992]. Doria et al. erweiterten den Algorithmus mit gewichtetem Kriterium für das Rauschen [Dorai, Weng & Jain 1997]. Ein Vergleich der verschiedene ICP Algorithmen vor dem Jahr 2001 wurde von Rusinkiewicz und Levoy erstellt [Rusinkiewicz & Levoy 2001]. Der grundlegende ICP Algorithmus wurde von ihnen in 6 Schritte unterteilt. In jedem Schritt wurde die Leistung des Algorithmus verglichen und ihr Einfluss auf den ganzen Algorithmus diskutiert. Eine andere Veränderung wurde von Chavarria und Sommer vorgeschlagen, in der die Kontur des Objekts auch in der Schätzung der Pose betrachtet wird [Chavarria & Sommer 2007].

2.3.4 Objektkalibrierung

2.3.4.1 Statistisches Verfahren

Im Ablauf der markenbasierten Objekterkennung werden alle Objekte durch einige Marken einzig definiert. Deshalb kann für jedes Bild die Kalibrierung der Objekte als

Clusteranalyse der Marken zusammengefasst werden. Josiger und Kirchner haben einen Vergleich über drei moderne Clusteralgorithmen erstellt [Josiger & Kirchner 2003]. Sie sind BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies), DBSCAN (Density Based Spatial Clustering of Applications with Noise) und CHAMELEON-Algorithmus. BIRCH-Algorithmus basiert auf dem Clustering-Feature-Tree, was ein balancierter Baum mit Verzweigungsfaktor und Schwelle als zusätzlichen Parametern ist [Zhang et.al 1996]. Ein vorhandener Clusteralgorithmus wird für die Blätter des Baums durchgeführt, damit die angeforderte Gruppen ausgesucht werden können. In CHAMELEON werden zuerst die Nachbarn des betrachteten Objekts bestimmt. Dann wird die gesamte Menge der Daten durch die Ähnlichkeit in M Teile unterteilt, wobei M ein Eingabeparameter ist. Nach der Verschmelzung der M Teile werden die Daten schlussendlich in K Gruppen verteilt. Der Parameter K ist vordefiniert und zeigt die vom Benutzer gewünschte Anzahl der Gruppen [Karypis et.al 1999]. Bei BIRCH oder CHAMELEON muss man die Anzahl der Cluster als Eingabe festlegen. Manchmal ist jedoch diese Zahl nicht bekannt oder schwer vorherzusagen. Der DBSCAN-Algorithmus kann dieses Problem vermeiden, was von Ester et.al in Jahr 1996 veröffentlicht wurde [Ester et.al 1996]. Statt der gewünschten Anzahl der Cluster sollen ein Bereich ϵ und die mindeste Anzahl der Nachbarn in diesem Bereich, was die sogenannte Grenzdichte beschreibt, vordefiniert werden. Das Objekt, das nicht genug Nachbarn im anforderten Bereich hat, wird als Rauschen markiert. Die bleibende Objekte werden im gleichen Cluster zusammengefasst, wenn sie nicht weiter als ϵ von mindesten N von ihren Nachbarn liegen, wobei N ein Eingabeparameter ist. In der Arbeit von Josiger und Kirchner werden die Ergebnisse der drei Algorithmen von verschiedenen Testdaten verglichen. Entweder für die Punktmenge einfacher Gestalt oder die Punktmenge nicht-sphärischer Gestalt können das DBSCAN und CHAMELEON Algorithmen die zufriedene Lösung liefern. In der Situation, wo die Testdaten verrauscht sind, kann der DBSCAN Algorithmus mit geeignetem Parameter auch ein zufriedenes Ergebnis ausgeben. Das Ergebnis bleibt stabil, obwohl der Anteil des Rauschens stark einsteigt.

2.3.4.2 Dynamisches Verfahren

Das Problem der Kalibrierung der Objekte ist ähnlich zum Problem der Segmentierung der Bewegungen in einer langen Bildsequenz. Eine traditionelle Lösungsstrategie ist, die Positionen der Marken im nächsten Zeitpunkt mit Kalman Filter vorherzusagen. Dann werden alle Marken anhand der kinematischen Parameter in verschiedene Gruppen eingeteilt. Mills und Novins haben eine andere Möglichkeit geliefert, damit die Objekte direkt mit 2D Graphen kalibriert werden können [Mills & Novins 2000]. Am Anfang ihres Algorithmus wird jede Marke mit einander verbunden. Dann werden alle Marken und die Kanten dazwischen zusammen als ein vollständiger Graph dargestellt. Zwischen den Bewegungen der Objekte verändert sich die Länge der Kanten. Die Kanten, die länger als eingesetzte Beschränkung sind, werden aus dem Graphen Schritt um Schritt gelöscht. Eine Marke gehört einem Objekt, genau dann wenn das Dreieck, das von dieser Marke und anderen Marken in diesem Objekt erzeugt wird, mindestens eine gleiche

Kante mit den anderen Dreiecken des Objekts hat. Am Ende des Algorithmus wird der ursprüngliche vollständige Graph in viele Teilgraphen zerlegt, die genau den kalibrierten Objekten entsprechen. Es gibt jedoch die Einschränkung in dem Algorithmus von Mills und Novins, dass zwei Objekte nicht auseinanderzuhalten sind, wenn ihre Marken mit einigen besonderen Strukturen angebracht werden [Rhijn & Mulder 2005]. Rhijn und Mulder haben dieses Problem gelöst, indem sie die Voraussetzung des Algorithmus veränderten. In der neuen Voraussetzung darf die Marke in einem Objekt erkannt werden, nur wenn sie mit anderen drei Marken in dem Objekt zusammen eine Pyramide erzeugen kann. Die Pyramide hier bezieht sich auf einen Körper der Geometrie, der vier Knoten hat und jede zwei davon eine Kanten erzeugen. Die neue stärkere Beschränkung erhöht die Erfolgsquote deutlich.

2.3.5 Objekterkennung und Verfolgung

Das Ziel dieser Arbeit ist, dass ein Objekt nach Initialisierung von dem System wieder erkannt werden kann. Eine Wiedererkennung des 3D Objekts durch 2D Bildfolgen wird von Lamdan et al. in ihrer Arbeit erfolgreich durchgeführt [Lamdan et al. 1988]. Sie haben einige interessante Punkte ausgewählt, um das totale Objekt zu beschreiben. Alle drei Punkte, die nicht in gleicher Gerade liegen, definieren ein Koordinatensystem, auf dem die entsprechenden Koordinaten von anderen Punkten berechnet und in einem HashMap gespeichert werden. Die richtige Korrespondenz wird so bestimmt, dass das kleinste Quadrate Modell der Transformation zwischen dem neuen Koordinatensystem und 2D Bild die beste Lösung liefert.

Andererseits kann die charakteristische Information jedes Objekts einen einzigen vollständigen Graphen erzeugen. Alle sichtbaren Marken des erkannten Objektes können als ein Teilgraph des vollständigen Graphen definiert werden, wodurch das Problem der Objekterkennung bzw. Objektverfolgung als das Teilgraph Isomorphismus Problem abgeleitet werden kann. Es gibt eine große Menge an Algorithmen, um das Problem zu behandeln, da das Isomorphismus Problem nicht nur im Bereich der Bildanalyse, sondern auch im Vergleich der Struktur von chemischen Verbindungen oder in biometrischer Identifikation betrachtet wird. Conte et al. haben eine Zusammenfassung über diese Algorithmen veröffentlicht [Conte et al. 2004]. Durch ihre Taxonomie können alle diese Verfahren in zwei Gruppen von genauen bzw. ungenauen Graph Matching Algorithmen unterteilt werden.

In einem genauen Graph Matching wird die strenge Korrespondenz zwischen zwei Graphen bestimmt. Die Abbildung von einen Graphen zu einem anderen soll bijektiv sein. Ullmann hat ein rekursives Rücksetzverfahren beschrieben, das sehr bekannt ist und bis heute für genaues Matching häufig benutzt wird[Ullmann 1976]. Was von Rhijn und Mulder in ihrer Arbeit für die Objekterkennung implementiert wurde, ist auch ein genaues Graph Matching Verfahren [Rhijn & Mulder 2005]. Sie folgen der Idee

von Lamdan, aber verbessern das Verfahren mit einer Beschränkung für die Größe des Teilgraphs, die ausreichend für die Unterscheidung von zwei Objekten ist. Nach der Verbesserung ist der neue Algorithmus viel effizienter und stabiler. Cordella et al. haben einen Algorithmus mit Namen VF2 für das Graph und Teilgraph Isomorphismus Problem in großen Graphen entwickelt [Cordella et al. 2004]. In dem Vorgang des Matching haben sie einige Regeln definiert, wodurch die Komplexität des Rechnens stark reduziert wird. Eppstein konzentriert auf das Teilgraph Isomorphismus Problem von planaren Graphen [Eppstein 1999]. Der Graph wird in viele kleine Bäume unterteilt. Auf diesen wird mittels dynamischer Programmierung das Matching in linearer Zeit durchgeführt.

Das genaue Graph Matching ist manchmal ungeeignet, beispielsweise für nicht komplett fest definierte Graphen, was z.B. bei Rauschen oder instabilen Komponenten vorkommt. Wegen des Unterschieds zwischen dem beobachteten Modell und idealen Modell, soll das Matchingsverfahren tolerant sein. Dadurch kann eine Korrespondenz zwischen zwei Graphen gefunden werden, obwohl es keine strenge Transformation dazwischen gibt. Außerdem benötigt das genaue Graph Matching Verfahren exponentielle Laufzeit im Worst-Case, die durch eine Approximation in ungenauen Matchingsverfahren stark reduziert werden kann. Messmer und Bunke haben ein Fehler-tolerantes Verfahren für Teilgraph Isomorphismus mit unbekanntem Graph als Eingabe entwickelt [Messmer & Bunke 1998]. Die Modellgraphen werden durch eine Vorverarbeitung in kleine Teilgraphen unterteilt. Alle diese Teilgraphen werden so zusammengefasst, dass die oftmals vorkommenden Teilgraphen nur ein mal repräsentiert werden. Der eingegebene Graph wird mit diesen verdichteten Graphen verglichen. Dadurch hängt die Laufzeit nur von der Anzahl der Modellgraphen ab.

Kapitel 3

Grundlagen

3.1 Generierung der 3D-Daten

3.1.1 TOF-Sensor

3.1.1.1 TOF Kamera

Die im MAROCO-System verwendete Kamera gehört zur Klasse der TOF-Sensoren, die außer den normalen Graufarbenbildern auch Tiefbilder liefern kann. Die Tiefmessung basiert auf dem sogenannten Laufzeitverfahren. Dazu wird die Szene durch ein Lichtpuls ausgeleuchtet und für jeden Bildpunkt wird die Zeit gemessen, die das Licht bis zum Objekt und wieder zurück braucht. Die Distanz ist direkt proportional zu der benötigen Zeit und kann durch die folgende Formel berechnet werden:

$$d = \frac{t_d}{2c} \tag{3.1}$$

wobei t_d die gemessene Zeit bezeichnet. Die Konstante c steht für die Lichtgeschwindigkeit.

Im Vergleich zu anderen 3D Kamerasystemen hat die TOF Kamera viele Vorteilen [TOF-Kamera Wikipedia]. Zuerst kann die TOF Kamera einfach die interessierenden Bereiche aus einem Bild extrahieren und nur die Pixel nah vor der Kamera betrachten. Zweitens, kann die TOF Kamera eine hohe Bildrate bis zu 80 bps erreichen. Diese Eigenschaft ermöglicht somit Echtzeitanwendungen. Außerdem benötigt die TOF Kamera weniger Platz als z.B. das Triangulationssystem und hat niedrigere Abhängigkeit von der Systemstruktur gegenüber dem Stereosystem.

Parameter	Value*	Notes
Type of Sensor	PhotonICs®PMD 41k-S2 (200x200)	Incl. SBI (Suppression of Background Illumination)
Standard measurement Range	0.3 to 7 m	
Repeatability (1σ)	< 3 mm	Typical value, central sensor area @4m distance, 75% reflectivity
Frame Rate (3D)	40 fps @ 200x200 pixels 60 fps @ 176x144 pixels 80 fps @ 160x120 pixels	Typical value, depending on camera settings and ROI
Field of View	40° x 40°	CS mount lens: f = 12,8 mm F1,1
Illumination Wavelength	870 nm	Eye safety class 1
Operation modus	hardware / software trigger mode, free run mode (standard)	
Power Supply [V]	12V ± 10%	
Interface	USB2.0	
Operating Temperature	0°C to 50°C	
Storage Temperature	- 20°C to 85°C	

* data for basic camera configuration

Abbildung 3.1: Die grundlegende Parameter der PMD Kamera. [PMD CamCube Einführung]

3.1.1.2 PMD Sensor

Der PMD Sensor heißt CamCube ist eine wichtige Komponente der TOF Kamera. Er liefert eine hohe Auflösung bis zum 204x204 Pixel und maximale Bildrate bis zum 40 bps. Durch Formel (3.1) wird der maximale Distanzbereich zum 7 m festgelegt. Die andere wichtigen Parameter findet man in Abbildung 3.1 [PMD CamCube Einführung].

Das Hintergrundlicht, z.B. das Sonnenlicht, könnte die Messung der Distanz stark stören. Die PMD Kamera benutzt das aktive Sendersignal und einen Fremdlicht-Filter (SBI), um das Hintergrundsignal zu unterdrücken. Außerdem bietet die PMD Kamera die Möglichkeit, die Integrationszeit der Kamera für jede Messung individuell einzustellen. Die Integrationszeit bezieht sich auf die Zeitspanne, in der die Kamera zur Aufzeichnung eines Bildes dem reflektierten Licht ausgesetzt wird. Für ein schwach reflektierendes Objekt benötigt der Sensor längere Integrationszeit als ein stark reflektierendes Objekt, um genug Information anzusammeln. Andererseits wird aber ausreichendes Licht von hellen Objekten auf den Sensor reflektiert, wenn die Integrationszeit zu lang definiert wird. In Abbildung 3.2 wird ein Beispiel der Tiefbilder mit verschie-

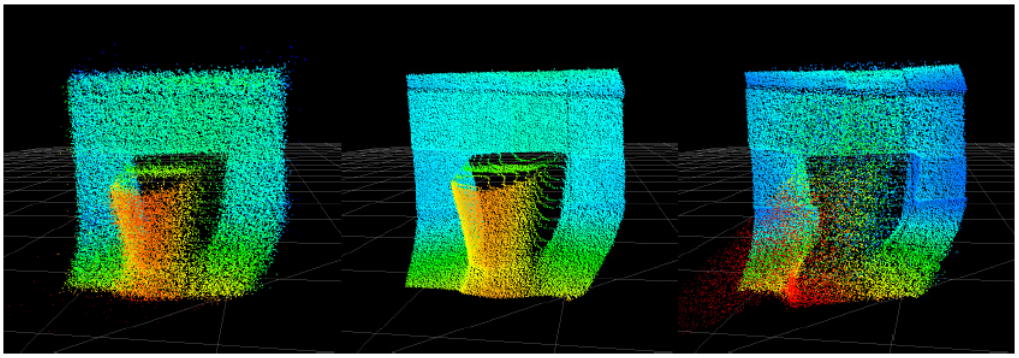

Abbildung 3.2: Integrationszeit von 140 μs, 1400 μs bzw. 14000 μs. Bitte beachten Sie die niedrige Signalstärke an der linken Seite und die Sättigung an der rechten Seite wegen der unangemessenen Integrationszeit. [PMD CamCube Entwicklungstutor]

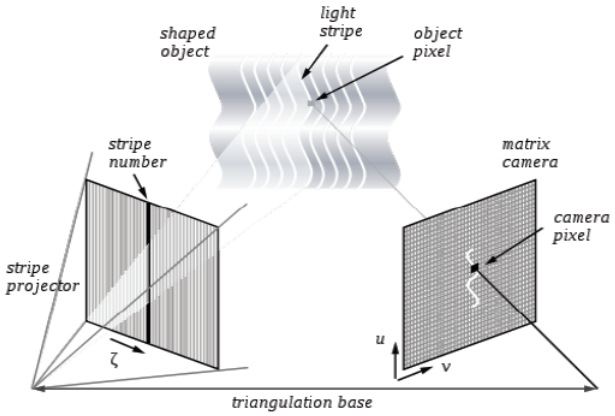

Abbildung 3.3: Das Arbeitsprinzip des Kinects. [Kinect-How it works]

denen Integrationszeiten gezeigt [PMD CamCube Entwicklungstutor].

3.1.1.3 Unterschied zwischen TOF Kamera und Kinect

Kinect ist eine Hardware zur Steuerung der Videospielkonsole Xbox360, die ein sogenanntes hands-free Kontrollieren liefert, wodurch die Spieler mit einigen bestimmten Gesten oder einer kurzen Bewegung ihres Körpers das Spiel spielen können [Kinect]. Um dieses Ziel zu erreichen, sammelt Kinect außer der normalen Bildeingabe aber auch die Tiefdaten der Szene an. Wegen dieser Eigenschaft wird das Gerät im Bereich von Computer Vision benutzt. Mithilfe des SDK ist die Programmierung der Kinect unter normalen Betriebssystemen z.B. Windows, Linux bzw. MacOS möglich.

Sensor	PMD CamCube	Kinect
Auflösung	204×204 Pixel	640×480 Pixel
Sichtfeld	40° × 40°	57° × 43°
Max Bildrate	40 fps	30 fps
Messbereich	0.3 → 7.0 m	1.2 → 3.5 m (mit Xbox Software)
Lange der Tiefdaten	8 bit (unsigned char)	11 bit

Tabelle 3.1: Die technische Daten von PMD Kamera und Kinect

Der Unterschied zwischen TOF Kamera und Kinect können auf dem Arbeitsprinzip zurückgeführt werden. In TOF Kameras wird die Tiefdaten durch dem Laufzeitverfahren berechnet, was im 3.1.1.1 erklärt wird. Das Abtastverfahren der Kinect heißt Light Coding. Eine große Menge von Streifen werden als Mustern auf die Szene bzw. die Objekte durch infrarotes Licht projiziert. Die ganz Szene mit diesen zusätzlichen Mustern wird von einer infraroten Kamera des Kinects aufgenommen. Durch die Verzerrung zwischen dem vordefinierten Muster im infraroten Licht und dem von der infraroten Kamera erkannten Muster kann das Tiefbild der Szene ausgerechnet werden. Die Abbildung 3.3 zeigt dieses Arbeitsprinzip der Kinect. Die weitere Information findet man im technischen Dokument des Firma Cadet [Kinect-How it works]. Der Vergleich über die genauen technischen Daten von PMD Kamera und Kinect wird in Tabelle 3.1 zusammengefasst. Obwohl Kinect eine bessere Auflösung und größeres Sichtfeld hat, ist die PMD Kamera wegen ihrer hohen Bildrate und großen Messbereich für das MAROCO-System geeignet. Außerdem mithilfe des SBI Systems ist die Arbeit der PMD Kamera unter schwieriger Umgebungsbedingung, z.B. außerhalb des Zimmers mit starker Störung von Sonneneinstrahlung, auch möglich.

3.1.2 Daten der PMD Kamera

Die PMD Kamera CamCube kann insgesamt 4 verschiedenen Vermessungsdaten ausgeben. Sie sind Amplitude, Intensität, Distanz und 3D Koordinaten. Die erste Zwei und letzte Zwei Typen der Daten können durch die Dimension in zwei Gruppen unterteilt werden.

3.1.2.1 2D Daten

Die Intensität bezieht sich auf Graustufen. Nach einer Abbildung können diese Graustufen auf das Intervall 0 bis 255 beschränkt werden und das Ergebnis ist das Bild einer normalen Schwarz-Weiß Kamera. Die Amplitude zeigt die Stärke der Beleuchtung, die vom Objekt wegen des aktiven Sendersignal von PMD Kamera selbst reflektiert wird.

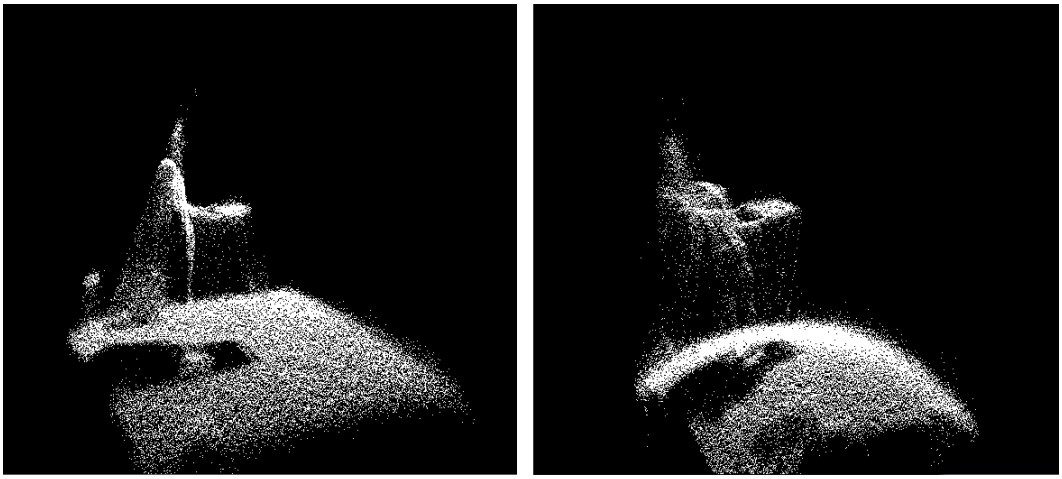

Abbildung 3.4: Die Visualisierung von 3D Koordinaten(link) und Distanzinformation(recht).

Dieser Wert kann die Qualität der Distanzinformation schätzen, d.h. das Objekt weit von Kamera liefert niedriger Amplitude als das Objekt in der Nähe von der Kamera, wenn sie mit identischem Material dargestellt werden. An der Gegenseite sind die Merkmalen mit höherem Rückstrahlvermögen durch Amplitudedaten einfacher betrachtet, was für die Erkennung der großen künstlichen Marken in unserer Arbeit sinnvoll ist.

3.1.2.2 3D Daten

Die PMD Kamera liefert 3D Daten in zwei Formen: die reine Distanzinformation und die 3D Koordinaten. Die Distanzinformation ist die gemessene Distanz zwischen der Kamera und Objekt, was direkt durch die Formel (3.1) berechnet wird. Bezüglich dieser Distanzinformation und der 2D Daten der normalen Kamera werden die 3D Koordinaten innerhalb der PMD Kamera berechnet und können durch die Schnittstelle abgefragt werden. Die Abbildung 3.4 zeigt die Visualisierung einer Szene mit jeweils der 3D Koordinaten und Distanzinformation von PMD Kamera. Eine Transformation ist notwendig, wenn man direkt die Distanzinformation im kartesischen Koordinatensystem visualisieren möchte.

3.1.3 Helligkeit und Sättigung

3.2 Markenerkennung

Wie im Abschnitt 2.3.1 beschrieben hat, sind viele Erkennungsalgorithmen in der Open Source Library OpenCV realisiert. Wegen vieler Benutzung von OpenCV, wird der

Abbildung 3.5: Die vier Beispielbilder. Von link nach recht sind Barbara, Lena, Peppers und Mandril. [Odessa 2011]

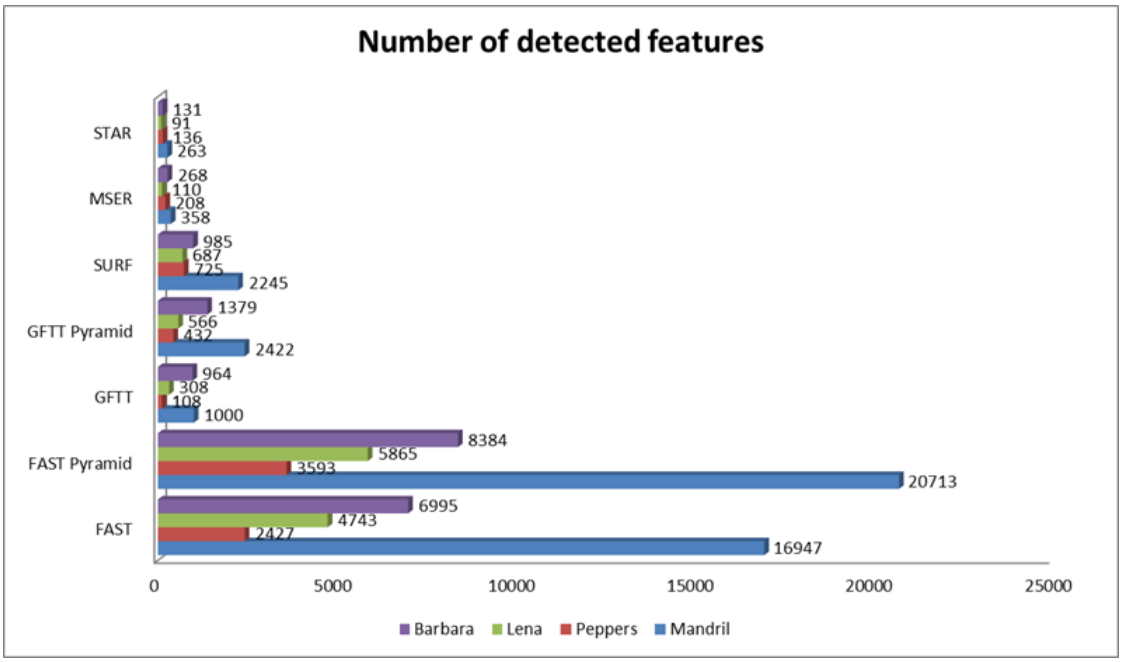

Abbildung 3.6: Anzahl der erkannten Merkmalen von alle vier Beispielbildern durch verschiedene Erkennungsalgorithmen. [Odessa 2011]

Vergleich dieser Algorithmen häufig gemacht. Im folgenden Abschnitt wird Ein davon zitiert, um den geeigneten Algorithmus für unsere Arbeit auszuwählen.

3.2.1 Auswahl des Erkennungsalgorithmus

Odessa hat in seinem Blog einen sehr gute Vergleich für alle Erkennungsalgorithmen von OpenCV durchgeführt [Odessa 2011]. Vier häufig benutzten Beispielbilder wurden betrachtet (s. Abb. 3.5).

Was besonders interessiert in seiner Arbeit ist der Vergleich über die Anzahl der be-

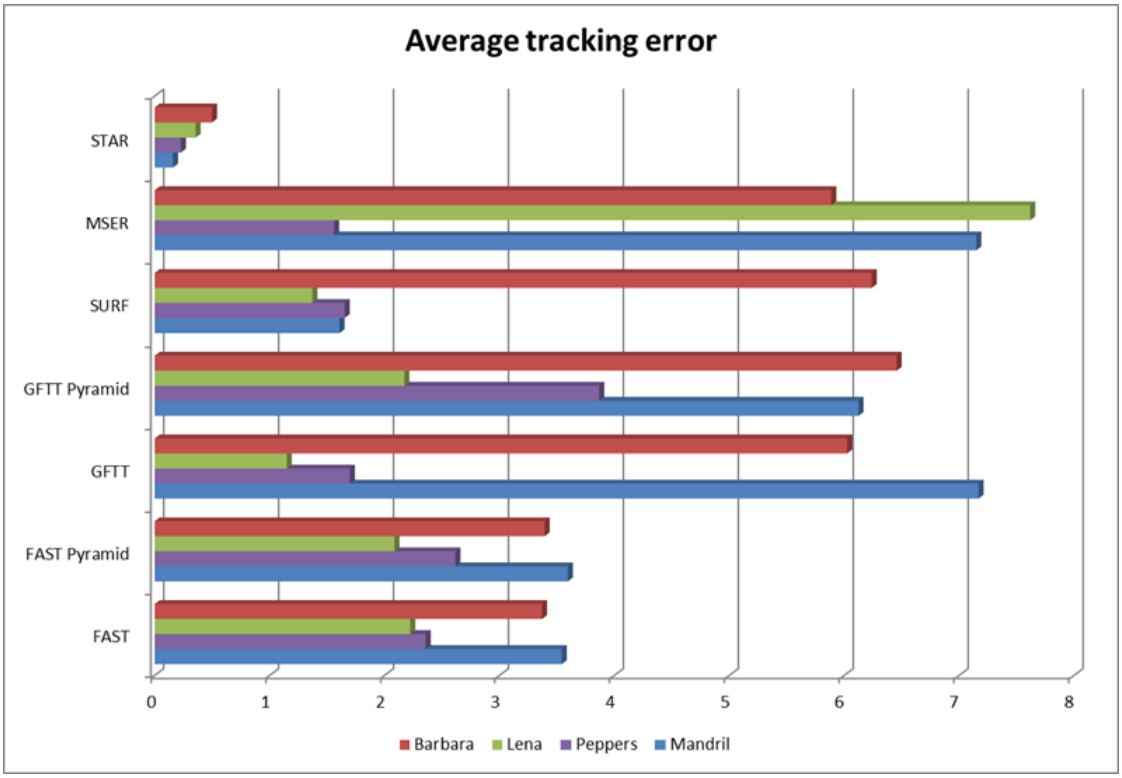

Abbildung 3.7: Der durchschnittliche Fehler (in Pixeln) zwischen der assoziierten Punkte von zwei verfolgten Bildern. [Odessa 2011]

trachteten Punkte bzw. der durchschnittlichen Fehler. Wegen der verschiedenen Prinzipien erkennt der Algorithmus FAST viel mehr Merkmale als SURF und STAR. Den Unterschied sieht man deutlich in der Abbildung 3.6. Je mehr Punkte betrachtet werden, desto mehr Rauschen wird in das System gebracht, weil viele normale Pixel auch als Merkmale erkannt werden. Das ist offensichtlich ein negativer Einfluss für die weitere Analyse der Daten. Abbildung 3.7 zeigt den durchschnittlichen Fehler in Pixeln von den Punktpaaren, was durch gleichen Erkennungsalgorithmus von Bezugsbildern erkannt wird. Der Algorithmus STAR erzeugt deutlich weniger Fehler in der Erkennung, und das Ergebnis hängt auch leicht von der Eingabe ab. Wegen der niedrigeren Fehlerquote und besserer Konzentration an großen Merkmalen wie Klecks ist der STAR Algorithmus für die Erkennung der Objekte mit künstlichen Marken sehr geeignet.

3.3 Markenverfolgung

3.3.1 Kalman-Filter

3.3.1.1 Zustandsraummodellierung

Das Kalman-Filter ist basiert auf einem linearen dynamischen System auf diskretisiertem Zeitraum. Die Zustandsgleichung des Systems wird häufig durch eine Differenzengleichung beschrieben. In vielen Fällen werden die Zuständen nur durch einem voneinander getrennten Zeitpunkt bestimmt. Kalman hat den Sonderfall der lediglich linearen Abhängigkeit der Zustände untereinander betrachtet, und vereinfachte die Zustandsgleichung zur linearen Differenzengleichung:

$$X_k = F_{k-1}X_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \tag{3.2}$$

Der Index k bzw. $k-1$ beziehen sich auf dem Zeitpunkt t_k und t_{k-1}, wobei $t_k = t_0 + k\Delta t$ und t_0 der Anfangszeitpunkt und k eine natürliche Zahl von Interesse ist. Deshalb beschreibt der mehrdimensionale Vektor X_k den Zustand des Systems am Zeitpunkt t_k. Die Matrix F_{x-1} ist die Übergangsmatrix für die zeitlich aufeinanderfolgenden Zustände X_k und X_{k-1}. Die Matrix B_{k-1} und Kontrollvektor u_{k-1} stellen den deterministischen Anteil der weiteren äußeren Einflüsse auf das System dar. Die zufälligen, nicht erfassbaren Komponenten der äußeren Einflüssen werden durch der stochastischen Größe w_{k-1} geschätzt, die einer Normalverteilung mit Mittelwert 0 und Kovarianz Q_{k-1} folgt.

$$w_{k-1} \sim N(0, Q_{k-1})$$

Wegen dieser Zufallsvariable bilden die Menge aller Zustandsvektoren eine Markow-Kette, d.h. der Zustand zu einem Zeitpunkt k hängt lediglich vom unmittelbaren zeitlichen Vorgänger an $k-1$ ab.

Die Beobachtungen des Systems werden aus modellierbarer Verzerrung und unvorhersagbarem Messrauschen bestehlt.

$$Z_k = H_k X_k + v_k \tag{3.3}$$

Z_k bezieht sich auf die Messung am Zeitpunkt k. Die Multiplikation von der Beobachtungsmatrix H_k und Zustandsvektor X_k beschreibt die linear Approximation der Verzerrung des Systems. Das Rauschen v_k wird im Kalman-Filter als zeitlich unkorreliert und normalverteilt angenommen.

$$v_k \sim N(0, R_k)$$

3.3.1.2 Das Kalman-Filter

Die Idee des Kalman-Filter ist, eine rekursive Formulierung aufzubauen, die aber nur die Schätzung vorheriges Zeitpunkts und aktuelle Messung benötigt, um die Schätzung des aktuellen Zeitpunkt zu bestimmen. Es gibt hauptsächlich zwei Phase im Kalman-Filter.

Prädiktion
In dem ersten Schritt dieser Phase wird eine vorangegangene Schätzung $X_{k|k-1}$ für aktuellen Zeitpunkt vorausgesagt.

$$\hat{X}_{k|k-1} = F_{k-1}\hat{X}_{k-1} + B_{k-1}u_{k-1} \tag{3.4}$$

Die Indizierungsschreibweise $k|k-1$ bezieht sich auf der Bedingtheit zu den Zeitpunkten k und $k-1$. Die Kovarianz gilt:

$$\hat{P}_{k|k-1} = F_{k-1}\hat{P}_{k-1}F_{k-1}^T + Q_{k-1} \tag{3.5}$$

Korrektur
Die Vorhersagen von letztem Schritt werden hier durch die neue Messung korrigiert.

$$\hat{X}_k = \hat{X}_{k|k-1} + \hat{K}_k\tilde{y}_k \tag{3.6}$$

und

$$\hat{P}_k = \hat{P}_{k|k-1} - \hat{K}_k S_k \hat{K}_k^T \tag{3.7}$$

Die Hilfsgröße Innovation:

$$\tilde{y}_k = Z_k - H_k\hat{X}_{k|k-1}$$

beschreibt, wie genau die vorhergesagten Schätzungen mithilfe der Beobachtungsgleichung den aktuellen Messungen approximieren. S_k bezieht sich auf der Residualkovarianz, was gilt:

$$S_k = H_k\hat{P}_{k|k-1}H_k^T + R_k$$

und $\hat{K}_k$ ist die zugehörige Kalman-Matrix.

$$\hat{K}_k = \hat{P}_{k|k-1}H_k^T S_k^{-1}$$

3.3.2 Singulärwertzerlegung

Sei M eine komplexe $m \times n$ Matrix mit Rang r. Dann bezeichnet die Singulärwertzerlegung das Produkt:

$$M = U\Sigma V^* \tag{3.8}$$

wobei U eine Unitäre Matrix mit Größe $m \times m$ und V^* die Adjungierte einer Unitäre Matrix mit Größe $n \times n$ ist. Σ bezieht sich auf eine $m \times n$ Diagonalmatrix:

$$\Sigma = \left(\begin{array}{ccc|ccc} \sigma_1 & & & & \vdots & \\ & \ddots & & \cdots & 0 & \cdots \\ & & \sigma_r & & \vdots & \\ \hline & \vdots & & & \vdots & \\ \cdots & 0 & \cdots & \cdots & 0 & \cdots \\ & \vdots & & & \vdots & \end{array}\right)$$

mit $\sigma_1 \geq \cdots \geq \sigma_r > 0$, wobei $\sigma_i, i = 1, \ldots, r$ als die Singulärwerte von Σ genannt werden.

Mithilfe der Singulärwertzerlegung haben Scott und Longuet-Higgins einen Algorithmus zur Bestimmung der assoziierenden Merkmale entwickelt. Seien I und J zwei nachfolgende Bilder und haben jeweils m und n Merkmale, die als $I_i(i = 1, \ldots, m)$ und $J_j(j = 1, \ldots, n)$ bezeichnet werden. Dann wird eine $m \times n$ Matrix G mit den Elementen

$$G_{ij} = exp(-\frac{r_{ij}^2}{2\sigma^2})$$

definiert, wobei r_{ij} den Abstand zwischen Merkmale I_i und J_j beschreibt. σ wird als einen Standard für Abstand definiert, wodurch das Vergrößern oder Verkleinern der Verschiebung des Objekts geschätzt werden kann. Der Wert von G_{ij} nimmt durch die Erhöhung der Distanz von 1 bis 0 monoton ab. Der zweite Schritt des Algorithmus von Scott und Longuet-Higgins ist die Singulärwertzerlegung der Matrix G.

$$G = TDU$$

wobei T und U die Unitäre Matrix mit jeweils Größe $m \times m$ und $n \times n$ sind, und D eine Diagonalmatrix ist. Sei E eine neue Matrix mit gleicher Größe von D, in der aber jedes diagonale Element als 1 ersetzt wird. Nach Austausch der Matrix D durch Matrix E erhält man eine neue orthogonale Matrix:

$$P = TEU$$

Die Aufgabe des dritten Schritts ist das Element P_{ij} zu finden, was gleichzeitig das Maximum der Reihe und Spalte ist. Wenn P_{ij} diese Bedingung erfüllt, sagt man, dass es eine Eins zu Eins Korrespondenz zwischen den Merkmalen I_i und J_j gibt. Der ganze Algorithmus kann durch folgendem Pseudocode erklärt werden.

Algorithm 1 Bestimmung der Korrespondenz der Merkmalen von zwei Bildern

$I, J, \sigma, Result$
for $i = 1 \rightarrow m$, $j = 1 \rightarrow n$ **do**
  $r_{ij} \leftarrow Dis(I_i, J_j)$
  $G_{ij} \leftarrow exp(-\frac{r_{ij}^2}{\sigma^2})$
end for
$T, U \leftarrow$ Singulärwertzerlegung von G
$E \leftarrow m \times n$ Diagonalmatrix mit $E_{ii} = 1$
$P \leftarrow TEU$
for $i = 1 \rightarrow m$ **do**
  $MaxSpalteIndex[i] \leftarrow$ Index der Spalte des maximalen Elements an Reihe i.
end for
for $i = 1 \rightarrow m$ **do**
  if $P_{iMaxSpalteIndex[i]}$ ist Maximum der Spalte $MaxSpalteIndex[i]$ **then**
    $Result \leftarrow$ Punktpaar$(I_i, J_{MaxSpalteIndex[i]})$
  end if
end for

3.4 Schätzung der Transformation

3.4.1 Quaternion

Ein Quaternion besteht aus einem Vektor mit 4 Elementen, wobei ein Element ein Skalar bezeichnet und die anderen drei eine Richtung im 3D Raum beschreiben. Quaternionen können aber auch als eine Erweiterung der komplexen Zahlen betrachtet werden, deren Imaginärteil nach drei neuer Zahlen i, j und k entwickelt werden. Eine Normalform der Quaternion findet man im unten:

$$q = q_0 + iq_x + jq_y + kq_z$$

i, j und k erfüllen die sogenannte Hamilton-Regeln:

$$i^2 = j^2 = k^2 = ijk = -1$$

$$ij = k, \quad jk = i, \quad ki = j$$
$$ji = -k, \quad kj = -i, \quad ik = -j$$

Ein andere Form mit getrennten Realteil und Imaginärteil wird im Folgenden definiert:

$$q = (q_0, \vec{q}) \tag{3.9}$$

wobei $q_0 \in \mathbb{R}$ ein Skalar und $\vec{q} \in \mathbb{R}^3$ ein Vektor ist. Sei r ein andere Quaternion mit:

$$r = r_0 + ir_x + jr_y + kr_z$$

Analog zum Vektoren im $\mathbb{R}^4$ wird das Skalarprodukt zwischen zwei Quaternion definiert als:

$$\langle q, r \rangle := q \cdot r := q_0 r_0 + q_x r_x + q_y r_y + q_z r_z$$

Weiterhin kann die Quaternionmultiplikation mithilfe der Form (3.9) berechnet als:

$$qr = (q_0 r_0 - \vec{q} \cdot \vec{r}, \quad q_0 \vec{r} + \vec{q} r_0 + \vec{q} \times \vec{r}) \tag{3.10}$$
$$\begin{aligned} &= (q_0 r_0 - q_x r_x - q_y r_y - q_z r_z) \\ &+ i(q_0 r_x + q_x r_0 + q_y r_z - q_z r_y) \\ &+ j(q_0 r_y - q_x r_x + q_y r_0 + q_z r_z) \\ &+ k(q_0 r_z + q_x r_y - q_y r_x + q_z r_0) \end{aligned} \tag{3.11}$$

Die rechte Multiplikation von r in Formel (3.11) kann aber auch zum einen links multiplizierten Matrix umschrieben werden.

$$qr = \begin{pmatrix} r_0 & -r_x & -r_y & -r_z \\ r_x & r_0 & r_z & -r_y \\ r_y & -r_z & r_0 & r_x \\ r_z & r_y & -r_x & r_0 \end{pmatrix} = \mathbf{R}q \tag{3.12}$$

Das konjugierte Quaternion von q ist definiert als:

$$\overline{q} = q_0 - iq_x - jq_y - kq_z$$

Das Produkt eines Quaternion und dessen Konjugierte ist eine nicht negative reelle Zahl.

$$q \cdot \overline{q} = q_0^2 + q_x^2 + q_y^2 + q_z^2$$

Mithilfe des konjugierten Quaternion kann man die Länge des Quaternion $|q|$ definieren:

$$|q| = \sqrt{q \cdot \overline{q}}$$

Ist die Länge eines Quaternion gleich 1, nennt man das Quaternion ein Einheitsquaternion. Für ein Einheitsquaternion gilt:

$$q \cdot \overline{q} = 1 \iff \overline{q} = q^{-1}$$

D.h. die Inverse und Konjugierte sind identisch. Für jedes Einheitsquaternion $q \neq \pm 1$ gibt es eine entsprechende Polardarstellung:

$$q = \cos\alpha + v \cdot \sin\alpha \tag{3.13}$$

mit $\alpha = \arccos(q_0) \in (0, \pi)$ und $v = \frac{1}{\sin\alpha}(iq_x + jq_y + kq_z)$.

3.4.2 Beschreibung der Drehungen im Dreidimensionalen Raum mit Quaternionen

Die Drehungen im dreidimensionalen Raum können durch die Einheitsquaternionen sehr gut beschriebt werden. Eine Abbildung der Rotation ρ_q kann in folgender Form definiert werden:

$$\rho_q : x \to qx\overline{q}$$

wobei q ein Einheitsquaternion und $\overline{q}$ dessen Konjugierte ist. Mithilfe der Polardarstellung (3.13) kann die Abbildung ρ_q sich auf eine Drehung im $\mathbb{R}^3$ um die Achse v mit Winkel $2\alpha \in (0, 2\pi)$ beziehen. Die entsprechende orthogonalen Matrix von q ist

$$R = \begin{pmatrix} q_0^2 + q_x^2 - q_y^2 - q_z^2 & 2q_xq_y - 2q_0q_z & 2q_xq_z + 2q_0q_y \\ 2q_xq_y + 2q_0q_z & q_0^2 - q_x^2 + q_y^2 - q_z^2 & 2q_yq_z - 2q_0q_z \\ 2q_xq_z - 2q_0q_y & 2q_yq_z + 2q_0q_z & q_0^2 - q_x^2 - q_y^2 + q_z^2 \end{pmatrix} \tag{3.14}$$

was zur Drehgruppe SO(3) gehört und eine Drehung in der Matrixform repräsentiert.

3.4.3 Orientierung mit Einheitsquaternion

Das Verfahren für die Orientierung der Objekte mithilfe der Quaternionen wurde erst von Horn im 1987 veröffentlicht [Horn 1987]. Seien D und M zwei Punktmengen mit gleicher Größe n. Dann kann die Transformation zwischen den Punkten von zwei Menge formuliert werden als:

$$d_i = \mathbf{R}m_i + \mathbf{T} + e_i \tag{3.15}$$

wobei d_i und m_i die i-ten Punkte der Punktmengen D bzw. M bezeichnen. $\mathbf{R}$ ist die Rotationsmatrix und $\mathbf{T}$ ist die Translationsmatrix. e_i beschreibt den Fehler für die Transformation, und kann umformuliert werden als:

$$e_i = d_i - \mathbf{R}m_i + \mathbf{T} \tag{3.16}$$

Das Ziel des Verfahrens ist, eine Rotations- bzw. Transformationsmatrix mit minimalem Fehler zu finden, dadurch wird die Summer des Quadrats von e_i betrachtet.

$$\sum_{i=1}^{n} \|e_i\|^2 = \sum_{i=1}^{n} \|d_i - \mathbf{R}m_i + \mathbf{T}\|^2 \tag{3.17}$$

Seien $\overline{d}$ und $\overline{m}$ die Schwerpunkte jeweils der Punktmenge D und M. Dann gilt

$$\overline{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \qquad \overline{m} = \frac{1}{n}\sum_{i=1}^{n} m_i$$

Der Abstand von jedem Punkt zum Schwerpunkt wird berechnet als:

$$d'_i = d_i - \overline{d} \qquad m'_i = m_i - \overline{m}$$

und die Summe der Abstände erfüllt natürlich

$$\sum_{i=1}^{n} d'_i = 0 \quad \textit{und} \quad \sum_{i=1}^{n} m'_i = 0 \tag{3.18}$$

Dann kann der Fehler in Formel (3.16) mit den Abständen zum Schwerpunkt $\overline{d}$ und $\overline{m}$ umgeschrieben werden:

$$e_i = d'_i - \mathbf{R}m'_i + \mathbf{T}' \tag{3.19}$$

wobei $\mathbf{T}'$ als

$$\mathbf{T}' = \mathbf{T} - \overline{d} + \mathbf{R}\overline{m}$$

definiert wird. Analog kann die Summe des Quadrats des Fehlers neu formuliert werden.

$$\begin{aligned}\sum_{i=1}^{n}\|e_i\|^2 &= \sum_{i=1}^{n}\|d'_i - \mathbf{R}m'_i + \mathbf{T}'\|^2 \\ &= \sum_{i=1}^{n}\|d'_i - \mathbf{R}m'_i\|^2 - 2\mathbf{T}' \cdot \sum_{i=1}^{n}(d'_i - \mathbf{R}m'_i) + n\|\mathbf{T}'\|^2 \end{aligned} \tag{3.20}$$

Wegen (3.18) ist der zweite Term gleich 0. Der dritte Term kann nicht negativ sein und wird 0, wenn der gesamte Fehler minimiert wird. D.h.:

$$\begin{aligned}\mathbf{T}' &= \mathbf{T} - \overline{d} + \mathbf{R}\overline{m} = 0 \\ \Rightarrow \mathbf{T} &= \overline{d} + \mathbf{R}\overline{m}\end{aligned} \tag{3.21}$$

Die Formel (3.21) berechnet direkt die Transformationsmatrix durch die Rotationsmatrix und die Schwerpunkte der beiden Punktmengen. Der erste Term von (3.20) kann zu

$$\sum_{i=1}^{n}\|d'_i - \mathbf{R}m'_i\|^2 = \sum_{i=1}^{n}(d'^t_i d'_i + m'^t_i m'_i - 2d'^t_i \mathbf{R}m'_i) \tag{3.22}$$

weiter formuliert werden. Dann wird die Minimierung des Fehlers durch die Bestimmung des Maximum der Summe

$$\sum_{i=1}^{n} d'^t_i \mathbf{R}m'_i \tag{3.23}$$

erreicht. Durch Ersetzen der Rotationsmatrix mit Quaternion wird das maximierte Problem umformuliert als:

$$\sum_{i=1}^{n}(qm''_i\overline{q}) \cdot d''_i$$

wobei $m''_i = (0, m'_{i,x}, m'_{i,y}, m'_{i,z})$ und $d''_i = (0, d'_{i,x}, d'_{i,y}, d'_{i,z})$ die erweiterte Quaternion für Punkte m'_i bzw. d'_i sind. Dann gilt:

$$\begin{aligned}\sum_{i=1}^{n}(qm''_i\overline{q}) \cdot d''_i &= \sum_{i=1}^{n}(qm''_i\overline{q}) \cdot (d''_i q\overline{q}) \\ &= \sum_{i=1}^{n}(qm''_i) \cdot (d''_i q)\end{aligned} \tag{3.24}$$

Die beide Multiplikationen in Klammer können als Formel (3.11) zum Produkt von einem Quaternion und einer Matrix umschrieben werden.

$$qm_i'' = \begin{pmatrix} 0 & -m'_{i,x} & -m'_{i,y} & -m'_{i,z} \\ -m'_{i,x} & 0 & -m'_{i,z} & -m'_{i,y} \\ -m'_{i,y} & -m'_{i,z} & 0 & -m'_{i,x} \\ -m'_{i,z} & -m'_{i,y} & -m'_{i,x} & 0 \end{pmatrix} = \mathbf{M}_i q$$

und

$$d_i'' q = \begin{pmatrix} 0 & -d'_{i,x} & -d'_{i,y} & -d'_{i,z} \\ d'_{i,x} & 0 & -d'_{i,z} & d'_{i,y} \\ d'_{i,y} & d'_{i,z} & 0 & -d''_{i,x} \\ d'_{i,z} & -d'_{i,y} & d'_{i,x} & 0 \end{pmatrix} = \mathbf{D}_i q$$

Dann kann (3.24) weiter ableitet werden:

$$\begin{aligned} \sum_{i=1}^{n} (\mathbf{M}_i q) \cdot (\mathbf{D}_i q) &= \sum_{i=1}^{n} q^t \mathbf{M}_i^t \mathbf{D}_i q \\ &= q^t \Big(\sum_{i=1}^{n} \mathbf{M}_i^t \mathbf{D}_i \Big) q \\ &= q^t \Big(\sum_{i=1}^{n} \mathbf{N}_i \Big) q \\ &= q^t \mathbf{N} q \end{aligned} \tag{3.25}$$

wobei $\mathbf{N}_i = \mathbf{M}_i^t \mathbf{D}_i$ ist, und $\mathbf{N}$ die Summe von $\mathbf{N}_i$ beschreibt. Sei $\mathbf{H}$ die Summe der Kreuzprodukten des Punktpaars von Punktmengen D und M.

$$\mathbf{H} = \sum_{i=1}^{n} m_i d_i^t$$

Es ist deutlich, dass die Größe der Matrix $\mathbf{H}$ 3×3 ist, deshalb kann $\mathbf{H}$ auch als

$$\mathbf{H} = \begin{pmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix}$$

geschrieben werden, wobei

$$S_{xx} = \sum_{i=1}^{n} m'_{i,x} d'_{i,x} \quad S_{xy} = \sum_{i=1}^{n} m'_{i,x} d'_{i,y}$$

usw. Dann kann die Matrix **N** im (3.25) durch die Elemente von **H** dargestellt werden als:

$$\mathbf{N} = \begin{pmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & -S_{xx}+S_{yy}-S_{zz} & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & -S_{xx}-S_{yy}+S_{zz} \end{pmatrix} \tag{3.26}$$

Nach dem Beweis von Horn [Horn 1987] wird die Formel (3.25) Maximum, genau dann wenn q der zu dem maximalen positiven Eigenwert der Matrix **N** entsprechende Eigenvektor ist.

3.5 Objekterkennung

3.5.1 DBSCAN

DBSCAN, kurz von dem englischen Name Density-Based Spatial CLustering of Applications with Noise, ist ein auf Dichte basierter Data-Mining-Algorithmus. Die Hauptidee des Algorithmus hängt stark von dem Begriff ,,Dichteverbundenheit" ab, was durch folgenden Definitionen erklärt wird.

Seien D eine Punktmenge im Raum $\mathbb{R}^n$ und $Dist(p,q)$ eine darauf definierte Distanzfunktion. ϵ und MinPts sind zwei Eingaben des Algorithmus.

Definition 3.1 *ϵ-Umgebung $N_\epsilon(p)$ ist eine Menge der Punkte um p, die erfüllt*

$$N_\epsilon(p) = \{q \in D | Dist(p,q) \geq \epsilon\}$$

Definition 3.2 *Ein Punkt p ist direkt Dichte-erreichbar zum Punkt q, g.d.w:*

$$\begin{aligned} &1) \quad p \in N_\epsilon(q) \\ &2) \quad |N_\epsilon(q)| \geq MinPts \end{aligned}$$

Definition 3.3 *Ein Punkt p ist Dichte-erreichbar zum Punkt q, g.d.w es eine Kette von Punkten $p_1, \ldots, p_n$ mit $p_1 = p$ und $p_n = q$ gibt, wobei p_{i+1} direkt Dichte-erreichbar zum p_i für alle $i \in [1,n]$ ist.*

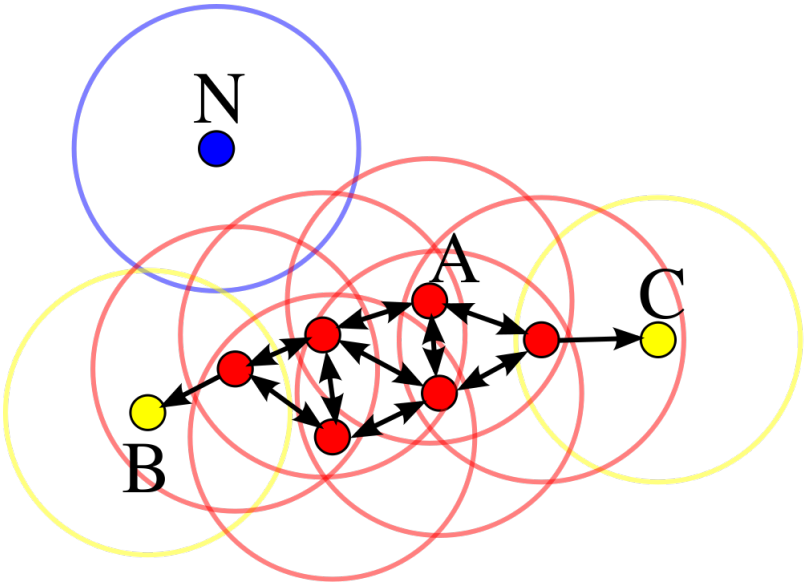

Abbildung 3.8: Ein Beispiel für die drei Typen der Punkte in DBSCAN-Algorithmus. Der rote Punkt A ist ein Kernpunkt. Die gelbe Punkte B und C sind die Grenzpunkte, die von roten Punkten Dichte-erreichbar sind. Wegen keine Dichte-Verbindung zwischen blau Punkt N und andere Punkte, wird N als Geräusch erkannt. [DBSCAN Wiki]

Definition 3.4 *Zwei Punkte p und q heißt Dichte-verbunden, g.d.w es ein Punkt o gibt, wobei p und q jeweils zum o Dichte-erreichbar sind.*

Mithilfe dieser Definitionen der Beziehung zwischen dem Punkten kann man jetzt eine einzige Definition des Clusters ausgeben.

Definition 3.5 *Sei C eine nicht leere Teilmenge von D. Dann heißt C ein Cluster, wenn die folgenden zwei Bedingungen erfüllt werden:*

1) *für $\forall p, q \in D$, wenn $p \in C$ und q ist Dichte-erreichbar zum p, dann gilt $q \in C$*
2) *für $\forall p, q \in C$, p ist Dichte-verbunden zum q*

Dann können alle Punkte von D in drei verschiedene Type zusammengefasst werden.

- Kernpunkt: Der Punkt, die Größe dessen ϵ-Umgebung größer als MinPts ist.
- Grenzpunkt: Der Punkt, dessen ϵ-Umgebung nicht genug groß (<MinPts), aber zu anderen Punkten Dichte-erreichbar ist.
- Geräusch: Der Punkt, der zu keinem Cluster gehört.

Abbildung 3.8 zeigt ein Beispiel für alle diesen drei Typen des Punkts, wobei die Kernpunkte mit Rot, Grenzpunkte mit Gelb und Geräusch mit Blau beschrieben werden. Die Kreise zeigen die ϵ-Umgebungen für die Punkte an ihrem Ursprüngen.

Das Algorithmus durchläuft für jeden Punkt von Punktmenge D und die Lösung hängt von der Reihenfolge der Punkte nicht ab. Ein Kernpunkt soll zuerst gefunden werden, und dann die andere Punkte in ihrer ϵ-Umgebung betrachtet werden. Zwei ϵ-Umgebung werden verknüpft, wenn sie mindesten einen identische Punkt haben. Der genaue Ablauf des DBSCAN-Algorithmus findet man unten.

Algorithm 2 DBSCAN

$D, \epsilon, MinPts$
Setzen C zum ersten Cluster
for jede $P_i \in D$ **do**
  if P_i ist nicht besucht **then**
    Markieren P_i als besucht
    $N = \{P_j \in D | Dist(P_i, P_j) < \epsilon\}$
    if Anzahl der Elemente von $N <$ MinPts **then**
      Markieren P_i als Geräusch
    else
      Fügen P_i in aktuellem Cluster C ein
      for jede $P_j \in N$ **do**
        if P_j ist noch nicht besucht **then**
          Markieren P_j als besucht
          $N' = \{P_k \in D | Dist(P_j, P_k) < \epsilon\}$
          if Anzahl der Elemente von $N' \geq$ MinPts **then**
            $N = N \cup N'$
          end if
        end if
        if P_j gehört zu keinem Cluster **then**
          Fügen P_j in aktuellem Cluster C ein
        end if
      end for
      Setzen C zum nächsten Cluster
    end if
  end if
end for

3.5.2 Teilgraph Isomorphismus

Die Objekterkennung bzw. Objektverfolgung werden vom Vergleich der Modellgraphen und Eingabegraphen realisiert. Ein für unsere Arbeit hilfreicher Algorithmus wurde von Rhijn und Mulder entwickelt [Rhijn & Mulder 2005]. Um das Problem von Graph Isomorphismus zu vereinfachen, werden nur ein Teilgraph S_{min} von Modellgraphen betrachtet. S_{min} soll ein vollständiges Graph sein und ein Modellgraph eindeutig bestimmen können.

Algorithm 3 Teilgraph Isomorphismus

Modellgraph:G_m, Eingabegraph:G_d, Menge von Kandidaten: K
for jede $p_i \in G_d$ **do**
  for jeder Nachbar p_j von p_i **do**
    Suchen alle Punktpaare $(v_k, v_l) \in G_m$, damit $dist(p_i, p_j) = dist(v_k, v_l)$
    if Anzahl der im letzten Schritt gefundenen Punktpaare
> die Größe der S_{min} **then**
      fügen den Punkt $< p_i, v_k >$ zum Kandidaten K ein
      if Drei Nachbarn von p_i können gefunden werden,
die mit p_i ein Pyramid aufbauen können
und die identische Kante zum p_i können in G_m gefunden werden **then**
        Teilgraph Isomorphismus finden
      end if
    end if
  end for
end for

Kapitel 4

Implementierung

Unsere Implementierung kann in 4 Hauptmodule unterteilt werden:

- Markenanalyse
- Objekteinlernen
- Objekterkennung und Verfolgung
- Bilder Steuerung

Die allgemeine Ablaufprozess des Programms findet man in Abbildung 4. Das ganz Programm ist eine Schleife. In jedem Schritt wird die Information von PMD Kamera angesammelt und als Eingabe eingegeben. Nach Bewertung der neuen Eingabe bzw. der Rückkopplung von letztem Ablauf wird die entsprechende Bilddaten von Modul Bilder Steuerung für die weiteren Schritten ausgewählt. Der Modul Markenanalyse analysiert

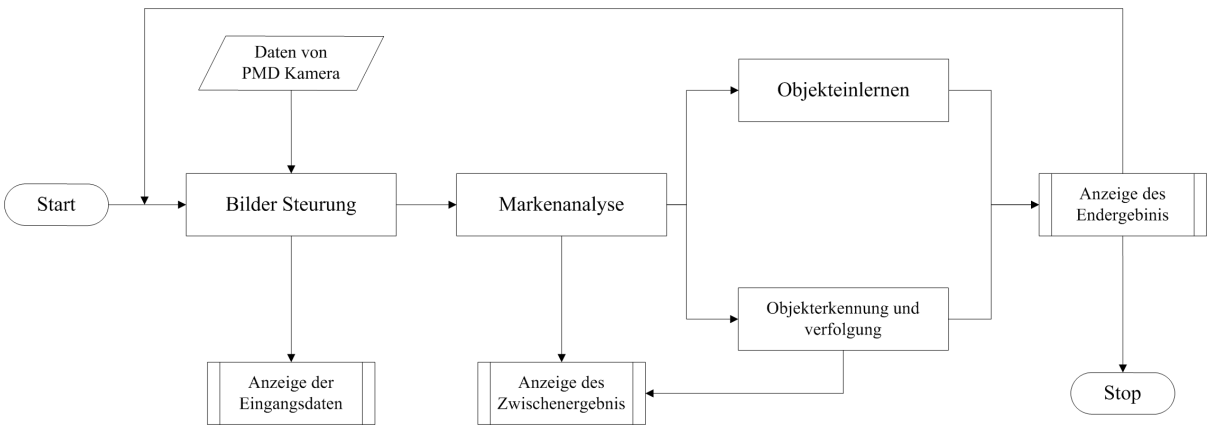

Abbildung 4.1: Ablaufdiagramm

die eingegebene Bilddaten und versucht die Marken zu finden und zu verfolgen. Die ausgefundene Marken werden von entweder dem Objekteinlernen oder der Objekterkennung und Verfolgen benutzt, welche am Anfang des Programms durch Benutzer entschieden wird. In der Einlernensphase wird zuerst die Transformation des Objekts bestimmt und dann das charakteristische Graph des Objekts durch die Marken dargestellt. Wenn man ein Objekt wieder erkennen möchte, werden die Marken von neuem Objekt zu den vorhandenen charakteristischen Graphen verglichen, die nach dem Objekteinlernen im Speicher gespeichert werden. Das Ergebnis wird in einem OpenGL Fenster ausgemalt und außerdem, als die Rückkopplung für den nächsten Schritt zurück zum Anfang der Schleife geschickt. Alle diese Module werden in folgenden Unterabschnitten genau erklären und ein ausführliches Ablaufdiagramm wird in Abbildung 4 gezeigt.

4.1 Markenanalyse

4.1.1 Markenerkennung

4.1.1.1 Auswahl der Größe der Marken

4.1.1.2 Kontrolle der Helligkeit

4.1.2 Markenverfolgung

4.2 Objekteinlernen

4.2.1 Markenanordnung

4.2.2 Kanteneinfügung

4.2.3 Speichern des Strukturgraph

4.3 Objekterkennung und Verfolgung

4.3.1 Vergleichung der Objekte

4.3.2 Bestimmung der Orientierung

4.4 Bilder Steuerung

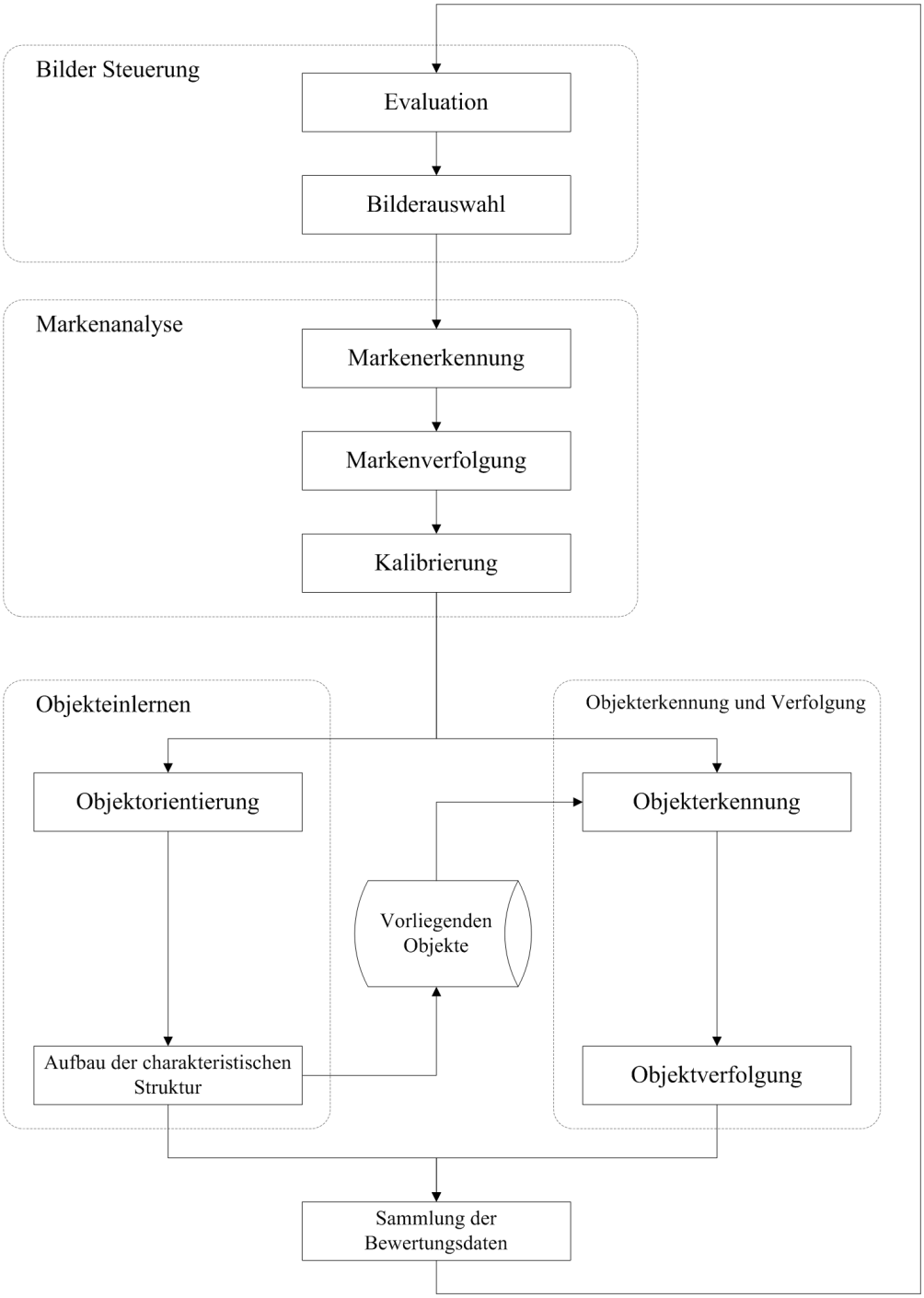

Abbildung 4.2: Ausführliches Ablaufdiagramm

Kapitel 5

Experimentelle Auswertung

Hier kommt die Auswertung ...

Kapitel 6

Schlusswort

Diese Arbeit implementiert die auf Marker basierte Objekterkennung bzw. Verfolgung im Grund auf dem Rahmenwerk MAROCO. Die Oberflächen der Zielobjekt werden mit retroreflektierenden Marker markiert. Eine PMD-Kamera beobachtet die ganze Szene von oben und liefert direkt die 3D-Daten. Die totale Laufzeit kann in zwei Phasen zusammengefasst werden. In der Initialisierungsphase wird die Fremdobjekt unter der Kamera gezeigt und die Marker darauf sollen herausgekannt und im System gespeichert werden. Wenn es mehr Objekte gibt, wird eine Kalibrierung am Anfang durchgeführt. Mills in [Mills & Novins 2000] hat eine kompakte Segmentierung der Bewegung mithilfe des sogenannten „Feature Interval Graph“dargestellt. [Rhijn & Mulder 2005] erweitert die Arbeit von Mills. Ein auf Pyramide basiertes Clustering-Verfahren wird statt des alten auf Dreiecke basierten Clustering-Verfahren vorgeschlagen. Nach der Initialisierungsphase wird das Objekt aus dem Gesichtsfeld der Kamera verschoben. Die Erkennungsphase fängt genau an, wenn das gleiche Objekt wieder unter der Kamera eingebracht wird. Die reflektierende Marker sollen nochmal gesammelt werden und ein von [Rhijn & Mulder 2005] repräsentierter Teilgraph-Tracker wird dann implementiert, um die Objekt zu kennen und die Position zu bestimmen.

Literaturverzeichnis

[Agrawal, Konolige & Blas 2008] M. Agrawl, K. Konolige and M. Blas: *CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching.* in Computer Vision-ECCV 2008, pp.102-115, Springer Berlin/Heidelberg (Zitiert auf Seite 6)

[ARToolKit] ARToolKit http://www.hitl.washington.edu/artoolkit/ Zuletzt besucht: 02.12.2011 (Zitiert auf Seite 4)

[Arun, Huang & Blostein 1987] K.S. Arun, T.S. Huang and S.D. Blostein: *Least-squares fitting of two 3-D point sets.* IEEE Trans Pattern Anal Machine Intell (1987) 9:698?700. (Zitiert auf Seiten 7 und 8)

[Bay et al. 2006] H. Bay, A. Ess, T. Tuytelaars and L.V. Gool: SURF:Speeded Up Robust Features. European Conference on Computer Vision, 2006 (Zitiert auf Seite 6)

[Besl & McKay 1992] P. Besl and N. McKay: *A Method for Registration of 3-D Shapes.* Trans. PAMI, Vol. 14, No. 2, 1992. (Zitiert auf Seite 8)

[Black & Yacoob 1997] M.J. Black and Y. Yacoob: *Recognizing Facial Expressions in Image Sequences Using Local Parameterized Models of Image Motion* International Journal of Computer Vision 25(1), 23-48 (1997) (Zitiert auf Seite 5)

[Chavarria & Sommer 2007] M.A. Chavarria and G. Sommer: *Structural ICP algorithm for pose estimation based on local features.* Computer Vision Theory and Applications - VISAPP , pp. 341-346, 2007 (Zitiert auf Seite 8)

[Chen & Medioni 1991] Y. Chen, G. Medioni: *Object Modeling by Registration of Multiple Range Images.* Proc. IEEE Conf. on Robotics and Automation, 1991. (Zitiert auf Seite 8)

[Conte et al. 2004] D. Conte, P. Foggia, C. Sansone and M. Vento: *Thirty years of graph matching in pattern recognition.* International Journal of Pattern Recognition and Artificial Intelligence Vol. 18, No. 3 (2004) 265-298 (Zitiert auf Seite 10)

[Cordella et al. 2004] L.P. Cordella, P. Foggia, C. Sansone and M. Vento: *A (Sub)Graph Isomorphism Algorithm for Matching Large Graphs.* IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 26, NO. 10, OCTOBER 2004 (Zitiert auf Seite 11)

[DBSCAN Wiki] DBSCAN Wikipedia: http://de.wikipedia.org/wiki/DBSCAN Zuletzt besucht: 11.05.2012 (Zitiert auf Seite 29)

[Dorai, Weng & Jain 1997] C. Dorai, J. Weng and A.K. Jain: *Optimal registration of object views using range data.* IEEE Transactions on Pattern Analysis and Machine Intelligence. 19(10):1131-1137, 1997 (Zitiert auf Seite 8)

[Eggert, Lorusso & Fisher 1997] D.W. Eggert, A. Lorusso, R.B. Fisher: *Estimating 3-D rigid body transformations: a comparison of four major algorithms.* Machine Vision and Applications (1997) 9: 272-290. (Zitiert auf Seiten 7 und 8)

[Eppstein 1999] D. Eppstein: *Subgraph Isomorphism in Planar Graphs and Related Problems.* Journal of Graph Algorithms and Applications, vol. 3, no. 3, pp. 1?27 (1999) (Zitiert auf Seite 11)

[Ester et.al 1996] M. Ester, H.P. Kriegel, J. Sander and X. Xu: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* Knowledge Discovery and Data Mining. Seite 226-231, 1996 (Zitiert auf Seite 9)

[Haag & Nagel 1999] M. Haag and H. Nagel: *Combination of Edge Element and Optical Flow Estimates for 3D-Model-Based Vehicle Tracking in Traffic Image Sequences.* International Journal of Computer Vision, Volume 35, Number 3, 295-319, 1999 (Zitiert auf Seite 5)

[Harris 1992] C. Harris: *Tracking with Rigid Objects.* MIT Press. 1992 (Zitiert auf Seite 4)

[Harris & Stephens 1988] C. Harris and M. Stephens: *A combined corner and edge detector.* Alvey Vision Conference, pp.147-151, 1988 (Zitiert auf Seite 6)

[Horn & Schunck 1981] B.K.P. Horn and B.G. Schunck: *Determining optical flow.* Artificial Intelligence Volume 17, Issues 1?3, August 1981, Pages 185-203 (Zitiert auf Seite 5)

[Horn 1987] B.K.P. Horn: *Closed-form solution of absolute orientation using unitquaternions.* J. Opt. Soc. Am. A, vol. 4, 629-642, 1987. (Zitiert auf Seiten 8, 24 und 28)

[ICP Wiki] ICP Wikipedia http://de.wikipedia.org/wiki/Iterative_ Closest_ Point_ Algorithm Zuletzt besucht: 02.02.2012 (Zitiert auf Seite 8)

[Josiger & Kirchner 2003] M. Josiger and K. Kirchner: *Moderne Clusteralgorithmen - eine vergleichende Analyse auf zweidimensionalen Daten.* Proc. FGML Workshop (FGML 2003), S.80-84, Karlsruhe 2003 (Zitiert auf Seite 9)

[Jurie & Dhome 2001] F. Jurie and M. Dhome: *A simple and efficient template matching algorithm.* International Conference on Computer Vision (ICCV 01) 2 (2001) 544-549. (Zitiert auf Seite 5)

[Kinect] Kinect http://www.xbox.com/en-US/kinect Zuletzt besucht: 02.12.2011 (Zitiert auf Seiten 3 und 14)

[Kinect-How it works] Kinect-How it works http://www.cadet.at/wp-content/uploads/2011/02/kinect_ tech.pdf Zuletzt besucht: 15.03.2012 (Zitiert auf Seiten 14 und 15)

[Kanatani 1994] K. Kanatani: *Analysis of 3-D rotation fitting.* IEEE Trans Pattern Anal Machine Intell (1994) 16:543-549 (Zitiert auf Seite 8)

[Karypis et.al 1999] G. Karypis, E.U. Han and V. Kumar: *CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling.* IEEE Computer, 32:8, S.68-75, 1999 (Zitiert auf Seite 9)

[Lamdan et al. 1988] Y. Lamdan, J.T. Schwatrtz and H.J. Wolfson: *On recognition of 3-D objects from 2-D images.* In Proceedings of IEEE International Conference on Robotics and Automation, 1988 (Zitiert auf Seite 10)

[Lepetit & Fua 2005] V. Lepetit and P. Fua: *Monocular Model-Based 3D Tracking of Rigid Objects: A Survey.* Foundations and Trends in Computer Graphics and Vision Vol.1, No 1(2005) 1-89. (Zitiert auf Seiten 3, 4 und 5)

[Lowe 2004] D.G. Lowe: *Distinctive Image Features from Scale-Invariant Keypoints.* Proceedings of the International Conference on Computer Vision. 2. pp. 1150?1157, 2004 (Zitiert auf Seite 6)

[Lucas & Kanade 1981] B.D. Lucas and T. Kanade: *An Iterative Image Registration Technique with an Application to Stereo Vision.* Proceedings of Imaging Understanding Workshop, pp. 121-130 (1981). (Zitiert auf Seite 5)

[Messmer & Bunke 1998] B.T. Messmer and H. Bunke: *A New Algorithm for Error-Tolerant Subgraph Isomorphism Detection.* IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 20, NO. 5, MAY 1998 (Zitiert auf Seite 11)

[Mills & Novins 2000] S. Mills and K. Novins: *Motion Segmentation in Long Image Sequences.* In Proceedings of the British Machine Vision Conference 2000, pp.162-171. (Zitiert auf Seiten 9 und 36)

[Odessa 2011] Odessa http://computer-vision-talks.com/2011/01/comparison-of-the-opencvs-feature-detection-algorithms-2/ Zuletzt besucht: 02.12.2011 (Zitiert auf Seiten 6, 17 und 18)

[PMD CamCube Einführung] PMDs CamCube Einführung: http://www.pmdtec.com/fileadmin/pmdtec/downloads/documentation/datenblatt_camcube3.pdf Zuletzt besucht: 13.03.2012 (Zitiert auf Seite 13)

[PMD CamCube Entwicklungstutor] PMDs CamCube Development Tutorial: http://www.pmdtec.com/fileadmin/pmdtec/downloads/documentation/camcube_softwaredevelopmenttutorial.pdf Zuletzt besucht: 14.03.2012 (Zitiert auf Seite 14)

[Rhijn & Mulder 2005] A. van Rhijn and J. D. Mulder: *Optical Tracking and Calibration of Tangible Interaction Devices.* IPT & EGVE Workshop, 2005. (Zitiert auf Seiten 6, 7, 10, 31 und 36)

[Rosten & Drummond 2006] E. Rosten and T. Drummond: *Machine learning for high-speed corner detection.* European Conference on Computer Vision, vol.1, 2006 (Zitiert auf Seite 6)

[Rusinkiewicz & Levoy 2001] S. Rusinkiewicz and M. Levoy: *Efficient Variants of the ICP Algorithm.* In Proceeding of the Third Intl. Conf. on 3D Digital Imaging and Modeling, pages 145-152, Quebec City, Canada (Zitiert auf Seite 8)

[Scott & Longuet-Higgins 1991] G.L. Scott and H.C. Longuet-Higgins: *An algorithm for associating the features of two images.* In Proc. Royal Society London, 1991, vol.B244, pp.21-26. (Zitiert auf Seite 7)

[Shi & Tomasi 1994] J. Shi and C. Tomasi: *Good features to track.* in IEEE Computer Society Conference: Computer Vision and Pattern Recognition, 1994. (Zitiert auf Seite 5)

[TOF-Kamera Wikipedia] TOF-Kamera Wikipedia: http://de.wikipedia.org/wiki/Time-of-flight-Sensor Zuletzt besucht: 02.04.2012 (Zitiert auf Seite 12)

[Tomasi & Kanade] C. Tomasi and T. Kanade: *Detection and Tracking of Point Features.* CiteSeerX - Scientific Literature Digital Library and Search Engine (United States) (Zitiert auf Seite 5)

[Ullmann 1976] J.R. Ullmann: *An Algorithm for Subgraph Isomorphism.* Journal of the ACM (JACM), Volume 23 Issue 1, Jan. 1976 (Zitiert auf Seite 10)

[Umeyama 1991] S. Umeyama: *Least-squares estimation of transformation parameters between two point patterns.* IEEE Trans Pattern Anal Machine Intell (1991) 13:376?380 (Zitiert auf Seite 8)

[Vaccetti, Lepetit & Fua 2004] L. Vacchetti, V. Lepetit and P. Fua: *Combining edge and texture information for real-time accurate 3D camera tracking.* Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality, 2004 (Zitiert auf Seite 4)

[VICON] http://www.vicon.com Zuletzt besucht: 01.12.2011 (Zitiert auf Seite 3)

[Zhang et.al 1996] T. Zhang, R. Ramakrishnan and M. Livny: *BIRCH: An efficient data clustering method for very large databases.* Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montnreal, 1996, S.103-144 (Zitiert auf Seite 9)

[Zhang & Navab 2000] X. Zhang and N. Navab: *Tracking and pose estimation for computer assisted localization in industrial environments.* Applications of Computer Vision, 2000, Fifth IEEE Workshop on. (Zitiert auf Seite 4)

[Zhang et al. 1995] Z. Zhang, R. Deriche, O. Faugeras and Q. Luong: *A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry.* Artificial Intelligence Volume 78, Issues 1?2, October 1995, Pages 87-119 (Zitiert auf Seite 5)