

# Detektion, Verfolgung und Entfernungsschätzung von Objekten in Robot-Vision Systemen mittels Online Adaption von Modellen

Von der Fakultät für Ingenieurwissenschaften  
der Universität Duisburg-Essen  
zur Erlangung des akademischen Grades einer  
Doktorin der Naturwissenschaften (Dr. rer. nat.)  
genehmigte Dissertation

von  
Maria Sagrebin-Mitzel  
aus Sankt-Petersburg (Russland)

Referent: Prof. Dr. Josef Pauli

Korreferent: Prof. Dr. Daniel Cremers

Tag der mündlichen Prüfung: 19. Januar 2011



# Danksagung

Die vorliegende Dissertation wurde am Lehrstuhl 'Intelligente Systeme' unter der Leitung von Herrn Prof. Dr. Josef Pauli an der Fakultät der Ingenieurwissenschaften der Universität Duisburg-Essen angefertigt.

An dieser Stelle möchte ich mich bei all denen bedanken, die mich auf vielfältige Weise bei meiner Arbeit unterstützt haben:

- Bei meinem Doktorvater Prof. Dr. Josef Pauli für die hervorragende Betreuung, fachliche Unterstützung und die zahlreichen Hinweise und Ratschläge bei der Bearbeitung des Themas.
- Bei Prof. Dr. Daniel Cremers, der sich bereit erklärt hat die Rolle des Zweitgutachters bei dieser Arbeit zu übernehmen.
- Bei allen Mitarbeitern des Lehrstuhls 'Intelligente Systeme', die während der letzten Jahre tapfer meine Anwesenheit am Lehrstuhl ertragen haben.  
...Ihr wart super!! ;))
- Bei Anastasia Noglik, die nicht nur durch ihr mathematisches Fachwissen entscheidend zum Verlauf der Arbeit beigetragen hat. In Konfliktsituationen sorgte sie immer dafür, dass unterschiedliche Ansichten stets gleichstark beleuchtet wurden. Ihre moralische Unterstützung war für die erfolgreiche Fertigstellung dieser Arbeit unentbehrlich.

Mein Dank gilt nicht zuletzt meinen Eltern, meiner Familie und meinen Freunden, die mich während meiner gesamten Ausbildung unterstützten und mir zu jeder Zeit mit Rat und Tat zur Seite standen.

Meinem Mann möchte ich an dieser Stelle besonders danken: 'Ohne Dich, wäre Vieles nicht möglich gewesen.'



# Zusammenfassung

Bildsequenzen, aufgenommen mittels einer einfachen oder professionellen Kamera, bilden mittlerweile in vielen Anwendungen eine wichtige Informationsquelle. Sei es Verkehrsüberwachung, Personen-Monitoring oder Roboternavigation. In allen diesen Bereichen ist eine verlässliche Auswertung der Kamerabilder von fundamentaler Bedeutung.

Im Rahmen dieser Arbeit wurden in den drei wichtigen Bereichen der Objektdetektion, Objektwiedererkennung und Rekonstruktion der lokalen Umgebung neue Ansätze und Verfahren entwickelt. Den Grundgerüst aller Verfahren bildet dabei die Konstruktion eines anwendungsspezifischen Modells und eine online stattfindende Adaption der entsprechenden Parameter.

Zur Detektion von bewegenden Objekten im Bild wurde ein neues Vordergrund-Hintergrund Separationsverfahren konzipiert. Anders als bei den auf Farb- oder Grauwerten der Pixel basierenden Ansätzen liegt hier der Schwerpunkt auf der Grauwertstruktur des Bildes. Zur Extraktion der Struktur wird bei diesem Verfahren die diskrete Kosinustransformation angewandt. Die auf unterschiedlichen Bildbereichen berechneten Koeffizienten bilden hierbei die Basis für das entsprechende Hintergrundmodell. Dabei werden im Sinne der Rauschunterdrückung feinere Details des Bildausschnittes vernachlässigt, indem nur die relevanten Koeffizienten zur Modellbildung herangezogen werden. Folgend der Idee der multimodalen Gaußverteilung zur Modellierung des dynamischen Hintergrunds werden hier mehrere Koeffizientenvektoren verwendet. Das Hinzufügen neuer Vektoren oder das Entfernen bereits vorhandener erfolgt online und hängt entscheidend von den Gegebenheiten der abgebildeten Szene ab. Die Detektion von bewegten Objekten im Bild erfolgt mittels eines Koeffizientenvergleichs zwischen den Koeffizienten des betrachteten Bildausschnittes im aktuellen Kamerabild und denen im Hintergrundmodell gespeicherten Werten. Im Unterschied zu bereits bestehenden Verfahren werden hier für die endgültige Entscheidung über das Vorhandensein eines bewegten Objektes auch Informationen benachbarter Bildbereiche herangezogen. Dieses Vorgehen resultiert in einem sehr stabilen Vordergrundbild, welches verglichen mit den Ergebnissen anderer Verfahren auch bei sich stark ändernden Umgebungsparametern robust bleibt.

Im Bereich der Objektverfolgung und -wiedererkennung wurde im Rahmen dieser Arbeit ebenfalls ein neues Verfahren entwickelt. Im Gegensatz zu vielen anderen Ansätzen konstruiert dieses bereits im Online-Betrieb ein 'multiview appearance' Modell von dem interessierenden Objekt und verfeinert das Modell auch während der darauf folgenden Objektverfolgung immer weiter. Die anfängliche Detektion des

interessierenden Objektes basiert auf der Erkennung eines vorher spezifizierten Bewegungsmusters, welches von dem Objekt während der Initialisierungsphase durchgeführt wird. Dazu werden zunächst mittels des oben beschriebenen Vordergrund-Hintergrund Separationsverfahrens alle bewegten Objekte der Szene detektiert und deren Trajektorien auf das spezifizierte Bewegungsmuster untersucht. Zur Berechnung der Trajektorien wird hier ein Ansatz zur Objektverfolgung verwendet, welches auf der Bestimmung von korrespondierenden SIFT-Merkmalen basiert.

Die so gewonnene erste Objektansicht von dem interessierenden Objekt bildet die Basis für dessen 'multiview appearance' Modell. Abhängig von der darauf folgenden Bewegung des Objektes und den bereits im Modell gespeicherten Ansichten werden entweder neue Objektansichten in das Modell aufgenommen oder bereits vorhandene neu sortiert. Die verschiedenen Objektansichten werden in der Datenbank als eine Ansammlung von entsprechenden SIFT-Merkmalen gespeichert. Ein Vergleich zweier Ansichten erfolgt mittels der Korrespondenzfindung der dazugehörigen SIFT-Merkmale.

Ein entscheidender Vorteil dieses Ansatzes ist, dass anders als bei den bereits etablierten Verfahren hier eine offline stattfindende Trainingsphase nicht benötigt wird. Auch sind die im Modell gespeicherten Ansichten des Objektes optimal an die jeweilige Anwendungsumgebung angepasst. Abhängig von der Position und dem Orientierungswinkel der Kamera werden nur solche Ansichten im Modell gespeichert, die auch von dieser Kamera während der Objektbewegung aufgenommen werden können.

Ein weiterer wissenschaftlicher Beitrag dieser Arbeit liegt im Bereich der Entfernungsschätzung. Hier wurde ein Verfahren entwickelt, welches ausgehend von der Bildinformation und der Bewegung der Kamera die verbleibende Distanz zu den bevorstehenden Hindernissen online ausrechnet. Der Ansatz basiert auf der Berechnung von Time-To-Contact (Zeit bis zum Zusammenstoß). Um einen hohen Grad an Stabilität und Robustheit in den berechneten Werten zu erzielen, wurden hierzu Modellgleichungen hergeleitet, die die theoretisch richtige Positionen der Merkmale in den Bildern der Kamera während der Bewegung vorhersagen. Die Herleitung der Gleichungen basiert ausschließlich auf den Gesetzen der projektiven Geometrie und den zugrunde liegenden Annahmen zur Kamerabewegung. Während in der Basisversion eine geradlinige Vorwärtsbewegung der Kamera mit einer konstanten Geschwindigkeit eine wichtige Voraussetzung war, wurden im Rahmen dieser Arbeit noch zusätzliche Erweiterungen entwickelt, die es erlauben, das Verfahren auch bei einer variierenden Kamerabewegung einzusetzen. Ausgehend von der Bildinformation der bewegten Kamera war es damit möglich die Entfernung zu den im Kamerabild sichtbaren Hindernissen zu schätzen und damit eine wichtige Basis für

Hinderniserkennung und -umgehung zu schaffen.

Alle im Rahmen dieser Arbeit entwickelten Ansätze wurden zur Performanzevaluierung umfangreichen Experimentenreihen unterzogen. In einer Fallstudie zur Simulation eines Lagerhausszenarios wurde auch der kombinierte Einsatz der Verfahren untersucht. Alle hierbei erzielten Ergebnisse zeugen von der praktischen Anwendbarkeit dieser Verfahren.





# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Fokus der Arbeit . . . . .	2
1.2	Kapitelübersicht und Aufbau der Arbeit . . . . .	4
<b>2</b>	<b>Detektion von bewegten Objekten in einer Bildsequenz</b>	<b>7</b>
2.1	Objektdetektion mittels Hintergrundbeseitigung . . . . .	7
2.2	Diskrete Kosinustransformation . . . . .	10
2.3	Vordergrund-Hintergrund Separation . . . . .	13
2.3.1	Online Modellierung des Hintergrunds . . . . .	14
2.3.2	Vordergrund Separierung . . . . .	18
2.4	Evaluierung und Ergebnisse . . . . .	21
2.4.1	Zum Vergleich herangezogene Verfahren . . . . .	22
2.4.2	Testvideos zur Evaluierung der Verfahren . . . . .	24
2.4.3	Ergebnisse der durchgeführten Experimente . . . . .	27
2.4.4	Laufzeit des Verfahrens . . . . .	40
2.5	Diskussion und Zusammenfassung . . . . .	41
<b>3</b>	<b>Robuste Objektverfolgung in der Bildsequenz</b>	<b>45</b>
3.1	Stand der Technik im Bereich der Objektverfolgung . . . . .	46
3.2	Bewegungsbasierte Objekterkennung im Bild . . . . .	50
3.2.1	Einfache Methode zur Objektverfolgung . . . . .	51
3.2.2	Detektion einer periodischen Bewegung . . . . .	54
3.3	Online Bildung eines Objektmodells . . . . .	58
3.4	Ergebnisse der Objektverfolgung . . . . .	61
3.4.1	Konstruktion einer initialen Objekttrajektorie . . . . .	61
3.4.2	Erkennung einer periodischen Bewegung . . . . .	63
3.4.3	Online Konstruktion eines Robotermodells . . . . .	67
3.4.4	Laufzeit des Verfahrens . . . . .	68
3.5	Diskussion und Zusammenfassung . . . . .	69
<b>4</b>	<b>Entfernungsschätzung bei geradliniger Kamerabewegung</b>	<b>71</b>
4.1	Theorie zur Berechnung von Time-To-Collision . . . . .	72
4.2	Darstellung der Problematik . . . . .	77
4.2.1	Leistungsfähigkeit des Standardverfahrens . . . . .	77
4.2.2	Ursache für schlechte TTC-Werte . . . . .	80

---

4.2.3	Existierende Ansätze zur TTC Berechnung . . . . .	81
4.3	Bewegungsmodell basierte TTC Berechnung . . . . .	83
4.3.1	Konstruktion der Modellgleichungen . . . . .	83
4.3.2	Verfahren zur Online Adaption der Modellparameter . . . . .	87
4.4	Ergebnisse im Rahmen einer Simulationsumgebung . . . . .	92
4.5	Diskussion und Zusammenfassung . . . . .	101
<b>5</b>	<b>Entfernungsschätzung bei einer 2D Kamerabewegung</b>	<b>103</b>
5.1	Erweiterung des Ansatzes auf verschiedene Bewegungsarten . . . . .	104
5.1.1	Gerade Bewegung mit variierender Geschwindigkeit . . . . .	104
5.1.2	Punktrotation in der zur Grundfläche parallelen Ebene . . . . .	106
5.1.3	Kurvenartige Bewegung . . . . .	108
5.2	Versuchsumgebung und technische Ausstattung . . . . .	111
5.3	Einfluss der Position der Merkmale im Bild . . . . .	113
5.4	Einfluss der Messgenauigkeit der Merkmale . . . . .	117
5.5	Ergebnisse im Rahmen einer Laborumgebung . . . . .	127
5.5.1	Versuchsdurchführung . . . . .	127
5.5.2	Ergebnisse der durchgeführten Experimente . . . . .	128
5.5.3	Laufzeit des Verfahrens . . . . .	130
5.6	Diskussion und Zusammenfassung . . . . .	131
<b>6</b>	<b>Durchführung und Ergebnisse einer Fallstudie</b>	<b>133</b>
6.1	Implementierung und Durchführung . . . . .	134
6.1.1	Programmetechnische Aspekte der Implementierung . . . . .	134
6.1.2	Aufbau der Versuchsumgebung . . . . .	137
6.1.3	Durchführung der Fallstudie . . . . .	139
6.2	Ergebnisse . . . . .	139
6.2.1	Bewegungsbasierte Roboterdetektion . . . . .	140
6.2.2	Online Generierung eines Robotermodells . . . . .	142
6.2.3	Konstruktion der Robotertrajektorie . . . . .	147
6.2.4	Randbedingungen . . . . .	149
6.3	Diskussion und Zusammenfassung . . . . .	151
<b>7</b>	<b>Zusammenfassung und Ausblick</b>	<b>153</b>
	<b>Abbildungsverzeichnis</b>	<b>159</b>
	<b>Tabellenverzeichnis</b>	<b>166</b>
	<b>Literaturverzeichnis</b>	<b>169</b>

# Einleitung

---

## Kapitelinhalt

---

<b>1.1 Fokus der Arbeit . . . . .</b>	<b>2</b>
<b>1.2 Kapitelübersicht und Aufbau der Arbeit . . . . .</b>	<b>4</b>

---

In den letzten Jahren hat die Entwicklung mobiler, autonom agierender Roboter rasant zugenommen. Heute werden solche Systeme bereits in sehr vielfältigen Bereichen eingesetzt: mobile Plattformen optimieren die Abläufe der industriellen Produktion, Einkaufsroboter navigieren die Kunden durch den Baumarkt und autonome Gabelstapler erleichtern Logistikprozesse.

Vor allem Industrieroboter haben sich in den letzten Jahren als ein unentbehrlicher Bestandteil vieler Produktionsabläufe etabliert. So werden diese erfolgreich u.a. in der Auto-, Lebensmittel-, Kunststoff- und Pharmaindustrie eingesetzt. Solche Roboter haben den Zweck verschiedene Arbeiten zu erledigen und dadurch für geregelte, schnelle und sichere Produktionsabläufe zu sorgen. Dabei werden oft nach der einmaligen Programmierung die Bewegungsabläufe ohne zeitliche Begrenzung immer wieder durchgeführt.

Die Einsatzmöglichkeiten von Industrierobotern beschränken sich oft auf genau beschreibbare Aufgaben in kontrollierten Umgebungen. Die hohe Präzision der Arbeitsabläufe wird üblicherweise auf Kosten der Flexibilität erreicht. In einer Vielzahl von aktuellen Problembereichen lassen sich jedoch weder die Umgebungsparameter noch die Aufgabe scharf definieren. Dies betrifft z.B. Verkehr und Transport, Gebäudereinigung, Kanalisation, Service, Pflege und alle Bereiche, in denen eine effiziente Mensch-Maschine-Kommunikation durch Nicht-Spezialisten erreicht werden soll. Diese marktträchtigen Anwendungen stellen Anforderungen an die Sensorik, Motorik und Steuerung von Robotern, für die bislang nur erste Ansätze vorliegen.

Der allgemeine Trend in letzter Zeit geht in die Entwicklung von mobilen Systemen, die weniger starr an ihre Anwendungsumgebung gebunden sind. Verfahren zur Robotersteuerung und Navigation, die nur wenige Annahmen über das Umfeld voraussetzen, haben ein breites Anwendungsgebiet und können viel flexibler mit den sich ändernden Umweltbedingungen umgehen.

Ein weiteres wichtiges Kriterium ist eine schnelle und einfache Montage des Gesamtsystems. Immer mehr Firmen und auch Endverbraucher fordern eine einfache und damit auch eine kostengünstige Installation des Systems, welches zusätzlich flexibel genug ist sich optimal in das jeweilige Anwendungsgebiet zu integrieren. Billigere und oft auch mit einfachen Sensoren ausgestattete Roboterkonstruktionen haben ebenfalls ein hohes Potenzial für das breite Publikum der Privatverbraucher interessant zu sein. Vor allem im Service- und Unterhaltungsbereich sehen viele Firmen gute Absatzmöglichkeiten.

Zur Einsparung der Kosten wird oft auf eher einfache Sensormodelle zurückgegriffen. Aufgrund der in letzter Zeit stark gestiegenen Massenproduktion an Web-Kameras bilden diese eine wertvolle Alternative. Das große Interesse an der Verwendung der Kameras als sensorischem Input spiegelt sich auch wieder in der rasanten Entwicklung unterschiedlicher Algorithmen zur Extraktion relevanter Informationen aus den Bildern der Kamera.

Der Fokus auf die 'Extraktion relevanter Informationen' bildet dabei den Kern eines noch relativ jungen Forschungsgebiets [BY92], [FA95], [Alo93], [BB92], [BC92] und [AWB87]. Dieses wird mit dem allgemeinen Begriff 'Aktives Sehen' bezeichnet und stellt eine Alternative zu dem bis zum Ende der achtziger Jahre die Bildverarbeitungsforschung dominierenden 'computational theory of vision' von D. Marr [D.82] dar. Hier wird, im Gegensatz zur Marr'schen Theorie das Sehen nicht als ein rein informationsverarbeitender Rekonstruktionsprozess aufgefasst. Im Gegenteil, aktive Sehsysteme können dadurch charakterisiert werden, dass sie über Mechanismen verfügen, aktiv Kameraparameter wie Position, Orientierung oder Fokus in Abhängigkeit von den Anforderungen der aktuellen Aufgabe und den externen Stimuli zu regeln. Eine hohe Priorität wird dabei auch den s.g. Aufmerksamkeitsmechanismen [Sch93], [OAVE93] zugeteilt. Sie erlauben eine kontextsensitive Figur-Hintergrund Trennung und dienen damit als eine Art Filter zur Extraktion der für die Aufgabenstellung relevanten Informationen aus den Bilddaten.

Die jüngsten Entwicklungen auf dem Gebiet der bildbasierten Robotik zeigen, dass eine Lockerung der Annahmen über das Anwendungsgebiet der Verfahren und eine gleichzeitige Präzision der zu lösenden Aufgabe ein hohes Potenzial haben, die Architektur der zukünftigen Anwendungen stark zu beeinflussen. Die vorliegende Arbeit ist in diesem Rahmen anzusiedeln.

## 1.1 Fokus der Arbeit

Der Fokus der Arbeit liegt in der Entwicklung eines bildbasierten Systems zur robusten Navigation des Roboters. Die Einsatzgebiete einer solch bildgestützten Navi-

gation sind sehr vielfältig und reichen von Anwendungen auf dem Mars hin bis zum Einsatz im privaten Haushalt. Erkundungsarbeiten, Lieferservice oder Transport sind dabei die üblichen Aufgaben.

Die Grundlage der vorliegenden Arbeit bildet das im folgenden beschriebene abstrakte Szenario: der Roboter navigiert autonom zu einem vorher festgelegten Zielpunkt, basierend auf den Bildinformationen einer direkt auf dem Roboter montierten Kamera und einer externen Kamera. Die externe Kamera überschaut dabei einen größeren Bereich der Umgebung und hat sowohl den Roboter als auch die Zielposition im Blickfeld. Die Umgebungsinformation und die eigene Position des Roboters relativ zum Zielpunkt sind dem Roboter nicht bekannt. Eine mögliche Erweiterung dieses Szenarios wäre die Verwendung mehrerer externer Kameras mit überlappenden Sichtbereichen.

In praktischen Anwendungen lässt sich dieses Szenario durch den Einsatz des Systems in einem Lagerhaus oder Bürokomplex realisieren. Eine an der Decke angebrachte Kamera würde dabei als die oben erwähnte externe Informationsquelle fungieren. Die Navigation des Roboters zu einem Zielpunkt würde dann entsprechend der Anwendung dem Zweck dienen, entweder ein Paket in einem bestimmten Büro abzuliefern oder einen schweren Container an einen anderen Ort zu transportieren.

Die fundamentalen Bausteine eines solchen Systems lassen sich gut durch die folgenden Teilaufgaben beschreiben:

- Erkennung des Roboters im Bild der Deckenkamera.
- Verfolgung des Roboters im Bild der Deckenkamera.
- Entfernungsschätzung zu den bevorstehenden Hindernissen basierend auf den Bildern der Roboterkamera.
- Korrektur der Robotertrajektorie ausgehend von den Informationen der Deckenkamera.

Die erste Aufgabe, die Erkennung des Roboters, dient dem Zweck der anfänglichen Positionsbestimmung des Roboters im Kamerabild. Anhand dieser Information wird auch die initiale Entfernung des Roboters zum Zielpunkt in Bildkoordinaten der Deckenkamera bestimmt. Mittels der Verfolgung des Roboters wird die 2D-Trajektorie des Roboters im Kamerabild ermittelt. Damit können eventuelle Abweichungen vom richtigen Kurs dem Roboter mitgeteilt werden und so seine Bewegungsrichtung korrigiert werden. Das Modul 'Entfernungsschätzung' sorgt dafür, dass der Roboter auf seinem Weg zum Zielpunkt bevorstehende Hindernisse als solche erkennt, um dann basierend auf dieser Information diesen auszuweichen.

Die Entfernungsschätzung zu den bevorstehenden Hindernissen basiert hier auf den Bildern der monokularen Roboterkamera, die in Fahrtrichtung des Roboters gerichtet ist. Im Rahmen dieser Arbeit wurde explizit auf die Verwendung eines Stereo-Kamerasystems verzichtet. Im Gegensatz zu Standardkameras sind der Kosten- und der Installationsaufwand eines Stereo-Kamerasystems deutlich höher. Die Deckenkamera liefert entsprechend ihrer statischen und technischen Eigenschaften auch nur eine grobe Richtung, in die sich der Roboter zum Erreichen des Ziels bewegen sollte. Die erfolgreiche Navigation zum Ziel erfolgt dann auf der Auswertung der Informationen beider Kameras.

Den Schwerpunkt dieser Arbeit bilden die ersten drei der oben genannten Teilaufgaben. Dabei ist anzumerken, dass bei der Entwicklung dieser Bausteine darauf geachtet wurde, dass diese einen weiten Anwendungsbereich haben und deren Performanz nur wenig von den anwendungsspezifischen Umweltparametern abhängt. Alle im Rahmen dieser Arbeit entwickelten Ansätze basieren auf der Verarbeitung von Bildsequenzen und können damit unter dem Oberbegriff der Bildfolgen Analyse zusammengefasst werden.

## 1.2 Kapitelübersicht und Aufbau der Arbeit

**Kapitel 2** dieser Arbeit beschäftigt sich mit der Aufgabe der Detektion von nicht statischen Objekten in einem abgebildeten Szenenbereich. Die Detektion basiert dabei auf der Vordergrund-Hintergrund Separierung der Szene. Als Vordergrund sind alle bewegten Objekte der Szene definiert.

Zur Modellierung des Hintergrunds der Szene wird hier die Strukturinformation des Bildes ausgenutzt. Das Kamerabild wird dabei zunächst in gleichgroße und nicht überlappende Bildbereiche aufgeteilt, auf welchen dann die Koeffizienten der diskreten Kosinustransformation berechnet werden. Die Struktur des jeweiligen Bereiches wird mittels dieser Koeffizienten beschrieben. Zur Rauschunterdrückung werden zur Modellgenerierung nur die Koeffizienten ausgewählt, die die grobe Struktur eines Bildbereichs darstellen. Die feinen Details werden hierbei vernachlässigt. Die Detektion von sich bewegenden Objekten im Bild erfolgt dann mittels eines Vergleiches zwischen den neu berechneten Koeffizienten und denen, die bereits im Modell gespeichert wurden.

Das zweite Kapitel gibt eine detaillierte Beschreibung wie das Hintergrundmodell verwaltet und aktualisiert wird und präsentiert gleichzeitig auch die mit diesem Verfahren erzielten Ergebnisse. Diese zeugen davon, dass die Ausnutzung der Grauwertstruktur des Bildes enorme Vorteile gegenüber farb- oder grauwertbasierten Ansätzen bringt. In vielen Situationen war dieser Ansatz denen in der Literatur

oft verwendeten Verfahren überlegen.

Die Detektion von bewegten Objekten im Bild bildet gleichzeitig auch die Grundlage für die im **Kapitel 3** vorgestellten Ansätze zur Erkennung und robusten Verfolgung eines Objektes in einer Bildsequenz. Die Grundidee hier war Verfahren zu entwickeln, die ohne jegliches Vorwissen und bereits im Online-Betrieb ein Modell von dem interessierenden Objekt erstellen und dieses dann darauf basierend in den folgenden Bildaufnahmen robust verfolgen.

Die anfängliche Selektion des betreffenden Objektes aus allen bewegenden Objekten der Szene basiert auf der Erkennung eines vorher spezifizierten Bewegungsmusters, welches von dem Objekt zu Beginn ausgeführt wird. Die dabei gemachte Restriktion auf periodische Bewegungsabläufe hat den Vorteil der leichteren Wiedererkennbarkeit dieser Bewegung und der einfachen Durchführung. Fast jedes bewegliche Objekt ist in der Lage diese Bewegungsart durchzuführen. Bei einem autonomen Roboter würde beispielsweise eine gleichmäßige Vor- und Rückwärtsbewegung oder eine kreisförmige Bewegung durchaus ausreichen.

Nach einer erfolgreichen Markierung des entsprechenden Bildbereichs startet dann das Modul 'Online Modellgenerierung'. Hierbei wird ein 'multiview appearance' Modell erzeugt. Die unterschiedliche Ansichten des Objektes werden als eine Ansammlung der entsprechenden SIFT-Merkmalen in der Datenbank abgelegt. In Kapitel 3 wird detailliert beschrieben wie neue Ansichten dem Modell hinzugefügt werden und unter welchen Bedingungen zwei Modelle in der Datenbank zu einem Modell verschmolzen werden müssen.

Das auf diese Weise bereits im Online-Betrieb konstruierte Modell dient in späteren Bildfolgen der Wiedererkennbarkeit des Objektes und damit auch der Konstruktion der entsprechenden Objekttrajektorie.

**Kapitel 4** dieser Arbeit beschreibt die Funktionsweise des Moduls 'Entfernungsschätzung'. Dieses basiert auf der Berechnung von Time-To-Contact (TTC, Zeit bis zum Zusammenstoß) Werten. Die Bilder der monokularen Kamera, die in Fahrtrichtung des Roboters montiert ist, liefern dazu den nötigen Input. Das im Rahmen dieser Arbeit entwickelte Verfahren zur robusten Berechnung von TTC-Werten basiert auf den Gesetzen der projektiven Geometrie und verwendet zur genaueren TTC-Schätzung die theoretisch richtigen Verläufe der 2D-Projektionen von 3D-Punkten während der Kamerabewegung.

In Kapitel 4 werden zum einen die dazu benötigten Modellgleichungen hergeleitet, zum anderen werden hier auch zwei Verfahren vorgestellt, die dem Zweck der zeitgerechten Adaption der entsprechenden Parameter dienen. Basierend auf den Ergebnissen der Experimente in der dafür programmierten Simulationsumgebung wurde eines der Verfahren zur späteren Evaluierung im Rahmen einer Laborumge-

bung ausgewählt.

In **Kapitel 5** werden zunächst Erweiterungen des im vierten Kapitel präsentierten Ansatzes vorgestellt. Diese dienen dem Zweck die bei diesem Verfahren sehr einschränkende Annahme einer konstanten Geradeausbewegung der Kamera aufzulösen. Die Erweiterungen beziehen sich zum einen auf rotierende Bewegungen der Kamera in der zur Grundfläche parallelen Ebene. Zum anderen werden damit auch Geradeausbewegungen mit variierender Geschwindigkeit ermöglicht.

Den größeren Teil des fünften Kapitels bildet die praxisnahe Evaluierung der Verfahren. In praktischen Experimenten wurde hier im Rahmen einer Laborumgebung untersucht, welche Merkmale sich besser für das hier entwickelte Verfahren eignen. Gleichzeitig wurde auch analysiert, in wie fern die Position der Merkmale im Bild Einfluss auf die Güte der berechneten TTC-Werte hat.

In einer weiteren Experimentenreihe wurde untersucht, wie stabil ein Roboter Hindernissen ausweichen kann, wenn er nur die aktuell berechneten TTC-Werte zur Verfügung hat. Zeitgleich wurden hierbei auch die bereits erwähnten Erweiterungen des Verfahrens getestet.

In **Kapitel 6** wurde im Rahmen einer Fallstudie zur Simulation eines Lagerhaus-szenarios der kombinierte Einsatz der hier entwickelten Verfahren untersucht. Die während dieser Experimentenreihe erzielten Ergebnisse zeugen von der praktischen Anwendbarkeit dieser Verfahren.

**Kapitel 7** schließt die Arbeit mit einer Diskussion über die hierbei erzielten Einsichten und einem ausführlichen Fazit ab.



# Detektion von bewegten Objekten in einer Bildsequenz

---

## Kapitelinhalt

---

<b>2.1</b>	<b>Objektdetektion mittels Hintergrundbeseitigung . . . . .</b>	<b>7</b>
<b>2.2</b>	<b>Diskrete Kosinustransformation . . . . .</b>	<b>10</b>
<b>2.3</b>	<b>Vordergrund-Hintergrund Separation . . . . .</b>	<b>13</b>
<b>2.4</b>	<b>Evaluierung und Ergebnisse . . . . .</b>	<b>21</b>
<b>2.5</b>	<b>Diskussion und Zusammenfassung . . . . .</b>	<b>41</b>

---

Eine robuste Detektion von bewegten Objekten in einer Videosequenz bildet in vielen Anwendungen oft einen fundamentalen Baustein. In vielen Bereichen, wie zum Beispiel der Überwachung von öffentlichen Verkehrsszenen müssen die Trajektorien der Fahrzeuge erfasst werden. Dazu ist eine sichere Extraktion von bewegenden Objekten vom Hintergrund der Szene eine wichtige Voraussetzung.

Das vorliegende Kapitel gibt zunächst eine Einführung in die grundlegende Problematik und beschreibt kurz schon bestehende Ansätze, sowie die allgemein gültigen Anforderungen an diese. Den zweiten Teil des Kapitels stellt eine detaillierte Beschreibung des im Rahmen dieser Arbeit entwickelten Verfahrens dar. Abschließend findet eine Evaluierung dieses Verfahrens statt. Dabei werden die Ergebnisse mit drei weiteren Verfahren verglichen, die in der Literatur bereits sehr verbreitet sind. Eine Zusammenfassung rundet das Kapitel ab und gibt einen Ausblick über die weiterführende Entwicklungsmöglichkeiten.

## 2.1 Objektdetektion mittels Hintergrundbeseitigung

Zur Detektion von bewegten Objekten in einer Bildsequenz werden oft Verfahren der Klasse Vordergrund-Hintergrund Separation eingesetzt. Diese Verfahren zielen darauf ab, das Bild in einen Hintergrund- und einen Vordergrundbereich aufzuteilen. Den Hintergrund der Szene bilden alle statischen Objekte. In Freilandanwendungen

werden zusätzlich noch die leichten periodischen Bewegungen der Bäume und das sanfte Rauschen der Blätter dem Hintergrund zugeordnet. Zu dem Vordergrundbereich gehören entsprechend alle bewegende Objekte der Szene.

Zu den elementaren Verfahren dieser Klasse gehört das so genannte 'frame differencing', [RE95]. Dabei wird das aktuelle Kamerabild von dem vorherigen Bild subtrahiert. Die Unterschiede zwischen den beiden Bildern werden üblicherweise in dem resultierenden Bild durch hellere Bereiche gekennzeichnet. Besitzt das bewegte Objekt kaum homogene Flächen, so ist dieses gut im Ergebnisbild zu sehen. Im anderen Fall ist das Objekt oft nur aufgrund seiner Umrandungen zu erkennen. Gleichzeitig ist dieser Ansatz sehr anfällig gegenüber Bildrauschen. Schon leichte Änderungen der Aufnahmebedingungen sorgen für ein sehr verrauschtes Ergebnis. Die helleren Bereiche im resultierenden Bild entsprechen dann nicht unbedingt den bewegten Objekten der Szene.

Einer anderen Vorgehensweise dagegen folgen Verfahren, die zunächst anhand von Trainingsbildern ein Hintergrundmodell der jeweiligen Szene erlernen und dann Unterschiede zwischen dem Modell und jedem neu aufgenommenen Bild feststellen. Abbildung 2.1 zeigt dazu ein Beispiel mit Aufnahmen aus der PETS Datenbank [PET05]. Hier besteht das Hintergrundmodell aus einem Abbild der Szene ohne

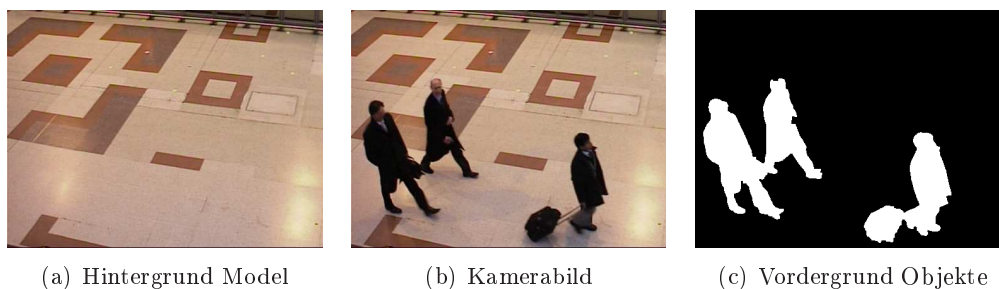


Abbildung 2.1: (a) anhand von Trainingsbildern erstelltes Hintergrundmodell der Szene; (b) aktuelles Kamerabild; (c) Ergebnisbild, welches nur die Vordergrund Objekte enthält.

jegliche bewegte Objekte. Durch Subtraktion des Kamerabildes (mitte) von dem Hintergrundmodell (links) erhält man das rechts gezeigte Bild, welches nur die bewegten Objekte enthält.

In Freiland Anwendungen, wenn die Umgebung nicht kontrollierbar ist, sind entsprechende Szenenbilder ohne bewegte Objekte kaum oder nur schwer zu bekommen. Daher wurden Verfahren entwickelt, welche mit dieser Problematik umgehen. Das Hintergrundmodell wird hier nicht einmalig gelernt, sondern online während des gesamten Aufnahmeprozesses adaptiert. Eine gute Übersicht über die mittlerweile sehr breite Palette der Verfahren findet sich in [SKS08]. Elhabian et.al. beschreiben

hier wie gut die jeweiligen Verfahren den folgenden Bedingungen genügen:

- Robust gegenüber Änderungen der Lichtverhältnisse, sowohl plötzlichen (Einschalten der Lampe) als auch sukzessiven (Änderungen der Tageszeit).
- Robust gegenüber plötzlichen Szenenwechsel aufgrund der Kameraverschiebung.
- Stabil gegenüber leichten periodischen Bewegungen im Hintergrund, z.B. Blätterrauschen der Bäume oder Flickern eines Monitors.
- Stabil gegenüber zeitweisen Verschiebungen der Hintergrundobjekte. Das Hintergrundmodell soll schnell an die geänderten Bedingungen angepasst werden.

Die einzelnen Verfahren unterscheiden sich in der Art und Weise wie das Hintergrundmodell erzeugt wird, und wie Abweichungen von diesem Modell festgestellt werden. Einige der Verfahren erzeugen anhand der vorherigen Bildsequenz ein statistisches Modell über die Werte eines jeden Pixels und berechnen dann die Wahrscheinlichkeit für das Auftreten des neuen Pixelwertes im aktuellen Bild. Ist diese zu klein, wird das entsprechende Pixel als zum Vordergrund gehörend markiert. Andere Verfahren benutzen wiederum Hidden-Markov Modelle, Kalman-Filter oder den Wiener-Filter zur Prädiktion von möglichen Werten des jeweiligen Pixels.

Ein, in der Literatur sehr verbreitetes Verfahren wurde 2000 von Stauffer und Grimson [SG00] vorgestellt. Sie modellieren die Werte eines jeden Pixels im Bild durch eine geeignete Komposition von unterschiedlichen Gaußfunktionen. Die Anzahl sowie der Mittelwert und die Varianz der einzelnen Gaußfunktionen werden online anhand der aufgenommenen Bildsequenz bestimmt und immer wieder neu adaptiert. Dieses Verfahren wird oft als Maßstab zur Evaluierung ähnlicher Verfahren herangezogen. Aufgrund der multimodalen Modellierung der möglichen Werte eines Pixels, werden leichte Bewegungen im Hintergrund sehr gut in das Hintergrundmodell eingebunden. Auch die online Adaption der Parameter der Gaußfunktionen sorgt dafür, dass dieses Verfahren sich in vielen praktischen Anwendungen bewährt hat. Besonders gut eignet es sich für Anwendungen wo sicher gestellt ist, dass sich die Lichtverhältnisse nur langsam ändern. Bei plötzlichen Lichtänderungen wie zum Beispiel Einschalten der Lampe oder das plötzliche Auftauchen der Sonne aus einer Wolke, wird dagegen oft das gesamte Bild in einen durchgehenden Vordergrundbereich transformiert. Dies liegt daran, dass aufgrund der vorherigen Bildsequenz die neuen Pixelwerte des Hintergrunds nicht durch das Modell abgedeckt werden.

Um die Auswirkungen der plötzlichen Lichtänderungen zu mindern, wurden in der letzten Zeit viele Ansätze zur kantenbasierten Detektion von bewegten Objekte

im Bild vorgestellt [CXS<sup>+</sup>07], [HDC07] oder [KO09]. Bei Lichtänderungen bleibt die Kanteninformation im Bild erhalten. Die meisten dieser Verfahren basieren auf den Ergebnissen des Sobel-Operators [ZT98] oder des Canny-Kantendetektors [Can86]. Um eine optimale Performanz dieser Verfahren zu erreichen, müssen oft der jeweiligen Anwendung entsprechend unterschiedliche Parameter und Schwellenwerte richtig gesetzt werden. Bei falschen Parameterwerten kann es passieren, dass die relevanten Kanten nicht extrahiert werden, und damit die Objektdetektion fehlschlägt.

Im Rahmen dieser Arbeit wurde daher ein Verfahren entwickelt, welches die Grauwertstruktur des Bildes ausnutzt. Nach der gleichmäßigen Aufteilung des Bildes in gleichgroße quadratische Bereiche, wird mittels der diskreten Kosinustransformation zunächst die Struktur des jeweiligen Bereiches ermittelt. Die Ergebnisse dieser Transformation bilden die Basis für das Modell des entsprechenden Bereichs.

Zur Detektion von Bewegungen im Bild wird die aktuell berechnete Struktur des Bereichs mit der im Modell gespeicherten Werten verglichen. Aufgrund des Modellaufbaus und der hier entwickelten Vergleichsmethode, ist dieses Verfahren sehr stabil gegenüber plötzlichen wie auch sukzessiven Lichtänderungen. Zusätzlich wird noch die räumliche Information einer Objektbewegung ausgenutzt. Anders als bei herkömmlichen Verfahren wird hier ein Bereich nur dann als zum Vordergrund gehörend markiert, wenn zusätzlich noch die Struktur eines seiner benachbarten Bereiche im vorherigen Bild von dem Modell signifikant abgewichen ist.

In nachfolgenden Abschnitten werden als erstes die Grundlagen der diskreten Kosinustransformation erklärt und dann basierend darauf das im Rahmen dieser Arbeit entwickelte Verfahren vorgestellt. Die Ergebnisse dieses Verfahrens werden anschließend mit den Resultaten der bereits etablierten Verfahren verglichen.

## **2.2 Diskrete Kosinustransformation**

Die diskrete Kosinustransformation, erstmalig erwähnt in 1974 von Ahmed, Natarjan und Rao [RY90], ist eine lineare, orthogonale Transformation, die ein zeitdiskretes Signal vom Ort- in den Frequenzbereich transformiert. Im Bereich der Bildverarbeitung wird diese Transformation gern zur Redundanzreduktion in Bildern verwendet. Ein bekanntes Beispiel ist dabei die JPEG-Kompression. Hier wird ein Bild so in seine Frequenzanteile zerlegt, dass eine Aufteilung in 'relevante' und 'weniger relevante' Komponenten ermöglicht wird. Die 'weniger relevanten' Komponenten werden dann bei der Kompression eher vernachlässigt.

Bei der Transformation wird ein Bild zunächst als eine diskrete Funktion interpretiert, welche jeder Pixelposition  $(x, y)$  einen entsprechenden Wert aus dem

Intervall  $[0, 255]$  zuordnet:

$$b : [0 \dots n] \times [0 \dots m] \rightarrow [0 \dots 255]$$

wobei  $n$  die Breite und  $m$  die Höhe des betrachteten Bildes bezeichnen. Diese Funktion wird dann auf die einzelnen Komponenten einer vorher definierten orthonormalen Funktionenbasis projiziert. Die Projektion auf solch eine Basisfunktion erfolgt mittels des Skalarprodukts, welches im diskreten Fall wie folgt berechnet werden kann:

$$s_{i,j}(b, f_{i,j}) = \sum_{x=0}^n \sum_{y=0}^m b(x, y) \cdot f_{i,j}(x, y)$$

wobei  $f_{i,j}(x, y)$  hier die  $i, j$ -te Funktion der verwendeten orthonormalen Funktionenbasis bezeichnet. Die Verwendung der Doppelindizes dient hier nur zum besseren Verständnis der später folgenden Überlegungen. Die so berechneten Skalarwerte  $s_{i,j}$  bilden die gesuchten Koeffizienten der Transformation. Diese geben an, wie stark die jeweilige Basisfunktion in der zu untersuchenden Bildfunktion enthalten ist.

Wie der Name der Transformation vermuten lässt, besteht die Funktionenbasis hier aus Kosinusfunktionen. Diese haben jeweils unterschiedliche Frequenzen und können im zweidimensionalen Fall, der Abbildung 2.2 entsprechend dargestellt werden. Insgesamt sind hier 64 solcher Basisfunktionen abgebildet. Ein kleines Quadrat

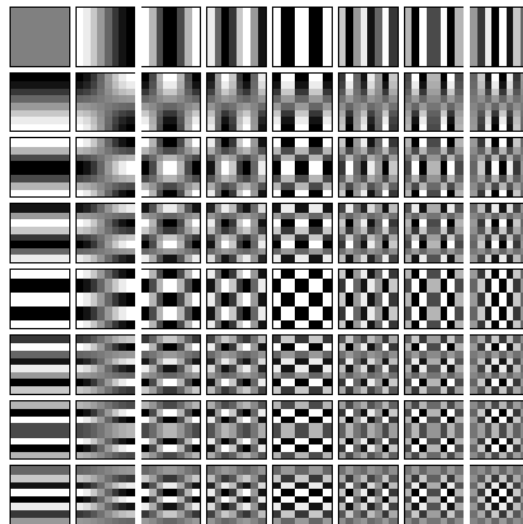


Abbildung 2.2: Graphische Darstellung der zweidimensionalen Basisfunktionen der diskreten Kosinustransformation.

entspricht einer Basisfunktion. Diese hat üblicherweise die  $8 \times 8$  Größe. Daher wird in vielen Anwendungen, wie auch bei der JPEG Kompression, das ursprüngliche Bild vor der Anwendung der Kosinustransformation in  $8 \times 8$  Blöcke aufgeteilt.

## 12 Kapitel 2. Detektion von bewegten Objekten in einer Bildsequenz

Das Muster links oben hat die niedrigste 'Frequenz' und ist nur ein Einheitsblock. Weist die Bildfunktion keinerlei Struktur auf, so erhält der entsprechende Koeffizient nach der Transformation einen sehr hohen Wert, während die anderen Koeffizienten nahe bei der Null bleiben. Von links nach rechts nimmt die Anzahl der 'Zyklen' zwischen hell und dunkel in die vertikale Richtung zu. Diese 'Zyklen' repräsentieren vertikal zunehmende räumliche Frequenz. Von oben nach unten nimmt hingegen die Anzahl der 'Zyklen' in die horizontale Richtung zu. In diagonalen Richtung nehmen sowohl die horizontalen als auch die vertikalen Frequenzen zu.

Die gesuchten Koeffizienten der Kosinustransformation können damit wie folgt berechnet werden:

$$s_{i,j} = C(i) \cdot C(j) \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} b(x,y) \cdot \cos[h(x,i)] \cdot \cos[h(y,j)]$$

$$\text{mit } h(a,b) = \frac{\pi}{4} \left(a + \frac{1}{2}\right) b$$

wobei die Indizes  $i$  und  $j$  hier die horizontalen und vertikalen räumlichen Frequenzen repräsentieren und  $C(\bullet)$  eine Normalisierungsfunktion bezeichnet. Bezugnehmend auf das bereits erwähnte Skalarprodukt von Funktionen, ist hier die Funktion  $f_{i,j}(x,y)$  wie folgt definiert:

$$f_{i,j}(x,y) = \cos[h(x,i)] \cdot \cos[h(y,j)]$$

Abbildung 2.3 zeigt beispielhaft eine 3D Visualisierung der originalen Werte eines  $8 \times 8$  Blocks und der entsprechenden Koeffizienten bezüglich der jeweiligen Basisfunktion. Die vertikale Kante im Originalbild wird im Frequenzbereich nur durch

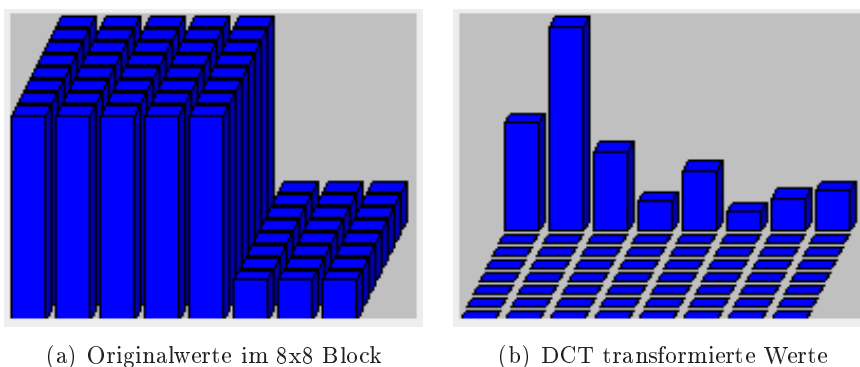


Abbildung 2.3: 3D Visualisierung der (a) Originalwerte im  $8 \times 8$  Block und (b) der entsprechenden Koeffizienten der diskreten Kosinustransformation.

Basisfunktionen der obersten Reihe dargestellt. Entsprechend müssen nur diese 8 Ko-

effizienten für die weitere Verarbeitung gespeichert werden. Im Unterschied zu den 64 Werten des Originalbildes ist dies eine enorme Einsparung. Andere Kantenorientierungen lassen sich ebenfalls durch geeignete Kombinationen der Basisfunktionen darstellen.

Es sei an dieser Stelle angemerkt, dass Basisfunktionen, die in Abbildung 2.2 in der linken oberen Ecke dargestellt sind, eher für die grobe Struktur des Bildes verantwortlich sind. Basisfunktionen in der unteren rechten Ecke dagegen stellen eher die feineren Details des Bildes dar.

Wie bereits am Anfang des Abschnittes erwähnt, wird die diskrete Kosinustransformation auch bei der JPEG-Kompression verwendet. Die hierbei manchmal an Rändern der Bildbereiche auftretenden Bildartefakte spielen allerdings im Rahmen des hier vorgestellten Ansatzes keine Rolle. Die diskrete Kosinustransformation dient hier lediglich zur Beschreibung der Struktur des jeweiligen Bildbereichs. Eine anschließende Rekonstruktion des Bildes findet nicht statt.

Eine mögliche Alternative zur der diskreten Kosinustransformation wäre die diskrete Wavelet-Transformation [Mal99]. Diese dient ebenfalls zur Strukturbeschreibung des Bildes und wird bereits bei JPEG 2000 erfolgreich eingesetzt. Bei der Verwendung dieser Transformation können die bereits erwähnten Artefakte bei der anschließenden Rekonstruktion des Bildes gemindert werden. Weil dies allerdings im Rahmen dieser Arbeit kaum von Bedeutung ist, wurde hier die diskrete Kosinustransformation beibehalten. Ähnlich der JPEG-Kompression werden auch hier zur Redundanzreduktion nur einige ausgewählte Koeffizienten bei der Modellbildung berücksichtigt.

## 2.3 Vordergrund-Hintergrund Separation

Wie bereits erwähnt wird bei diesem Verfahren für das Hintergrundmodell die Struktur unterschiedlicher Bildbereiche ermittelt. Dazu wird über das Bild zunächst ein Raster der Größe  $n \times n$  gelegt und dann mittels der diskreten Kosinustransformation die Struktur des jeweiligen Quadrats berechnet. Anders als bei herkömmlichen Verfahren wird hier das Hintergrundmodell nicht für jedes einzelne Pixel des Bildes bestimmt, sondern für die jeweiligen Quadrate. Dies hat den Vorteil, dass plötzliche Ausreißer in den Werten eines Pixels kaum Auswirkungen auf das resultierende Vordergrundbild haben. Bei dem Verfahren von Stauffer et. al. würden zum Beispiel solche Ausreißer zu einem Salz und Pfeffer Rauschen in den Ergebnissen führen.

Die Größe des Rasters kann der Anwendung entsprechend gewählt werden. Dabei ist zu beachten, dass die Größe des kleinsten Objekts von Interesse im Bild nicht viel kleiner sein soll als die gewählte Rastergröße. Dabei hängt die Größe des Objek-

tes im Kamerabild nicht nur von der eigentlichen Größe des Objektes ab, sondern auch von der Kameraauflösung und dem Abstand der Kamera zum betrachteten Objekt. Diese Faktoren sollten bei der Wahl der Rastergröße ebenso berücksichtigt werden. Bei den durchgeführten Experimenten im Rahmen der Parkplatzüberwachung war zum Beispiel die  $4 \times 4$  Rastergröße durchaus ausreichend. Objekte wie Kinder oder Katzen konnten hier robust extrahiert werden. Für die Überwachung von Büroräumen reichte dagegen auch eine  $8 \times 8$  Rastergröße aus.

### 2.3.1 Online Modellierung des Hintergrunds

Als Modell für den jeweilige Bildbereich wird bei diesem Verfahren eine Auswahl der 'wichtigen' Koeffizienten der diskreten Kosinustransformation gespeichert. Diese sind in Abbildung 2.4 durch Kreuze markiert. Wie dieser Darstellung zu entnehmen

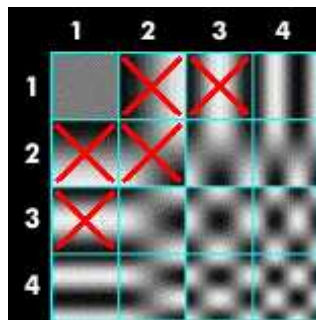


Abbildung 2.4: Koeffizienten, die durch Kreuze markierten Basisfunktionen, werden in einem Vektor als das Hintergrundmodell des jeweiligen Bildausschnitts gespeichert.

ist, repräsentieren die entsprechenden Basisfunktionen die grobe Struktur des jeweiligen Bildausschnittes. Die Nichtberücksichtigung anderer Koeffizienten führte zu der bereits erwähnten Rauschunterdrückung. Plötzliche Ausreißer in den Pixelwerten würden sich hier in den Koeffizienten höherfrequenter Basisfunktionen widerspiegeln.

Die Beschränkung auf die oben markierte Basisfunktionen ist eher experimenteller Natur. Versuche in denen zusätzlich weitere Koeffizientendiagonalen zur Modellbildung verwendet wurden, konnten keine erhebliche Verbesserung der Ergebnisse liefern. Im Gegenteil die Hinzunahme von Koeffizienten, die den Basisfunktionen aus der unteren rechten Ecke der obigen Abbildung 2.4 entsprechen, führte zu einer Minderung des positiven Effektes der Rauschunterdrückung.

Abbildung 2.5 zeigt beispielhaft eine Aufnahme eines Büroraumes und das entsprechende Hintergrundmodell. Zur Visualisierung des Hintergrundmodells wurde, nur unter Berücksichtigung der ausgewählten Koeffizienten eine inverse Kosinustrans-



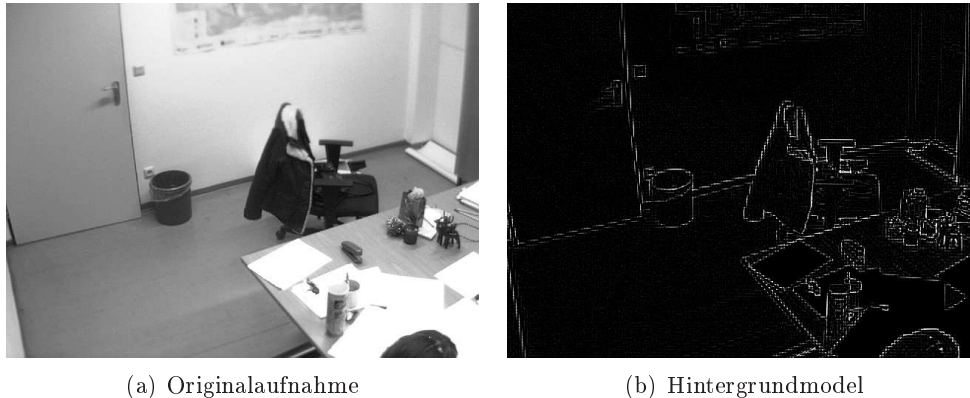


Abbildung 2.5: (a) Originalaufnahme eines Büroraumes; (b) Visualisierung des entsprechenden Hintergrundmodells mittels der inversen Kosinustransformation und nur unter Berücksichtigung der ausgewählten Koeffizienten.

formation berechnet. Es ist deutlich zu sehen, dass das Hauptaugenmerk des Modells auf der Grauwertstruktur des Bildes liegt. Alle helleren Bereiche des rechten Bildes entsprechen den Objektkanten, während die schwarzen Bereiche die homogenen Flächen widerspiegeln. Das plötzliche Aufhellen des Büros durch Einschalten einer Lampe würde hier nichts an der Struktur und damit an dem Hintergrundmodell ändern. Damit sind die Voraussetzungen für die Stabilität gegenüber sich ändernden Lichtverhältnissen gesetzt. Doch anders als beim Einsatz des Canny-Kantendetektors müssen hier zur Extraktion der Bildstruktur keine Parameter oder Schwellenwerte angepasst werden.

Die ausgewählten Koeffizienten des entsprechenden Bereiches werden in Form eines Vektors gespeichert. Abweichungen von dem Modell werden festgestellt, indem der neu berechnete Koeffizientenvektor mit dem des Modells verglichen wird. Dabei wird der Winkel zwischen den beiden Vektoren berechnet. Ist dieser größer als ein bestimmter Schwellenwert  $S$ , wird der entsprechende Bereich als 'vom Hintergrund abweichend' markiert. Die Länge der jeweiligen Vektoren wird hier nicht berücksichtigt. Dies hat den Vorteil, dass Hintergrundstrukturen, die aufgrund der starken Beleuchtung im Bild nur noch schwach ausgeprägt sind, vom Verfahren immer noch robust dem Hintergrund zugeordnet werden.

Abhängig von der Wahl des erwähnten Schwellenwerts, kann ein Bereich aus zwei Gründen als 'vom Hintergrund abweichend' markiert werden:

- Der Bereich repräsentiert ein bewegtes Objekt und gehört zum Vordergrund.
- Die neu berechneten Koeffizienten stimmen aufgrund des Bildrauschens nicht mit denen des Modells überein.

Um die beiden Fälle voneinander unterscheiden zu können, werden noch zusätzlich Informationen der benachbarten Bildbereiche herangezogen. Ein Bildausschnitt wird nur dann als 'zum Vordergrund gehörend' markiert, wenn mindestens einer seiner Nachbarn im letzten Bild als 'vom Hintergrund abweichend' markiert wurde. Im anderen Fall wird der Bereich weiterhin dem Hintergrund zugeordnet.

Damit wird an dieser Stelle angenommen, dass die Aufnahme Frequenz der Kamera hoch genug ist, um auch die schnellen Bewegungen in der Szene gut zu erfassen. Ein wichtiger Punkt dabei ist, dass beim Betrachten der Aufnahme nicht der Eindruck entsteht, als ob das bewegende Objekt von einer Stelle zur anderen springen würde. Eine im Kamerabild fließende Objektbewegung ist hier eine der Voraussetzungen für die gute Performanz des Verfahrens. Ist diese Annahme gegeben, so können damit bei der Objektdetektion noch zusätzlich die räumlichen Eigenschaften der Objektbewegung ausgenutzt werden.

Wird ein Bereich aufgrund des Bildrauschens als 'vom Hintergrund abweichend' markiert, so wird das entsprechende Modell um einen weiteren Koeffizientenvektor erweitert. Auf diese Weise kann auch ein leicht dynamischer Hintergrund einer Szene modelliert werden. Ein Modell  $BM$  (Background Model) für ein Bildausschnitt  $k$  ist damit wie folgt definiert:

$$BM_k = \{v_1, \dots, v_n, \alpha_1, \dots, \alpha_n\}$$

wobei  $v_i$  hier den  $i$ -ten Koeffizientenvektor des Modells und  $\alpha_i$  das Gewicht des jeweiligen Vektors bezeichnen. Um sicherzustellen, dass bei einem längeren Einsatz des Verfahrens die Modelle der unterschiedlichen Bildbereiche aus nicht zu vielen Vektoren bestehen und immer aktuell bleiben, werden Vektoren mit niedrigen Gewichten von dem Modell entfernt. Das Gewicht des jeweiligen Koeffizientenvektors repräsentiert damit seine Relevanz in dem Modell. Wird ein neuer Vektor dem Modell hinzugefügt, bekommt sein Gewicht den initialen Wert  $T_{inc}$  zugewiesen. Gleichzeitig werden die Gewichte anderer Vektoren um den ebenfalls vorher definierten Wert  $T_{dec}$  verkleinert. Wenn allerdings ein neu berechneter Koeffizientenvektor einem Vektor des Modells zugeordnet werden konnte, wird das Gewicht desjenigen Vektors vergrößert und damit auch seine Relevanz für das Modell erhöht. Die Zuordnung basiert dabei auf dem Winkel zwischen den beiden Vektoren und dem vorher festgelegten Schwellenwert  $S$ . Die Gewichte anderer Vektoren werden in diesem Fall um den Wert  $T_{dec}$  verkleinert. Damit ergibt sich folgende Aktualisierungsvorschrift:

$$\alpha_i = \begin{cases} \alpha_i + T_{inc}, & \text{wenn } v_i = \arg \max_{v_j} \{s(v_{new}, v_j) \mid j = 1, \dots, n\} \\ \alpha_i - T_{dec}, & \text{sonst} \end{cases}$$

wobei  $s(v_{new}, v_j)$  das Ähnlichkeitsmaß der beiden Vektoren bezeichnet. Diese Funktion ist umgekehrt proportional zum Winkel der beiden Vektoren definiert:

$$s(v_{new}, v_j) = \frac{1.0}{\angle(v_{new}, v_j)}$$

Hat der neu berechnete Koeffizientenvektor  $v_{new}$  zu mehr als einem Vektor des Modells einen Winkel kleiner als der bereits oben erwähnte Schwellenwert  $S$ , so wird nur das Gewicht desjenigen Vektors erhöht zu dem der Winkel am kleinsten ist. Damit bleiben dem Modell nur die Vektoren erhalten, die für die Modellierung des Hintergrunds am relevantesten sind. Die weniger wichtigen Vektoren werden nach einiger Zeit entfernt. Damit das Gewicht eines Vektors nicht unkontrolliert immer weiter ansteigt, wurde im Rahmen dieser Arbeit die obere Grenze für ein Gewichtswert willkürlich auf 20 gesetzt.

Aufgrund des dynamischen Aufbaus des Modells hat die Wahl des Schwellenwerts  $S$  keine schweren Konsequenzen für die Performanz des Verfahrens. Bei einem kleineren Wert um die  $10^\circ$  bis  $20^\circ$  Grad reagiert das Verfahren sensitiver auf Änderungen im Bild. Die anfängliche online Adaption des Verfahrens dauert etwas länger und die Modelle der unterschiedlichen Bildbereiche enthalten mehr Vektoren, als dies bei einem höheren Schwellenwert der Fall wäre. Die allgemeine Performanz ist allerdings mit der eines höher gesetzten Schwellenwerts vergleichbar. Im Rahmen der hier durchgeführten Experimente wurde dieser Wert auf  $S = 45^\circ$  gesetzt.

Mittels der beiden Parameter  $T_{inc}$  und  $T_{dec}$  kann kontrolliert werden, wie lange die weniger relevanten Vektoren dem Modell erhalten bleiben sollen. Ein kleinerer  $T_{inc}$  Wert und gleichzeitig ein höherer  $T_{dec}$  Wert führen zu einem eher dünn besetzten Modell. Vektoren mit dem anfänglichen Gewicht  $T_{inc}$  werden vom Modell entfernt, wenn sie nicht innerhalb der nächsten  $\left\lceil \frac{T_{inc}}{T_{dec}} \right\rceil$  Bilder ein höchstes Ähnlichkeitsmaß zu einem, der neu berechneten Koeffizientenvektoren vorweisen können.

Sind zum Beispiel die beiden Parameter wie folgt gewählt:  $T_{inc} = 1.0$  und  $T_{dec} = 0.3$ , so wird ein neu hinzugefügter Vektor  $v_i$  mit dem initialen Gewicht  $\alpha_i = T_{inc} = 1.0$  nach genau  $\left\lceil \frac{T_{inc}}{T_{dec}} \right\rceil = \left\lceil \frac{1.0}{0.3} \right\rceil = 4$  Bildern aus dem entsprechenden Modell entfernt, wenn er die oben erwähnte Bedingung nicht erfüllt. In diesem Fall wird mit jedem neu aufgenommenen Bild das Gewicht  $\alpha_i$  des Vektors  $v_i$  um 0.3 verringert. Nach genau 4 Bilder erreicht das Gewicht einen negativen Wert und der Vektor wird aus dem Modell entfernt.

Der komponentenweise Aufbau des Modells hat Ähnlichkeiten mit dem Verfahren von Stauffer et.al. [SG00]. Hier werden die Werte eines Pixels mittels einer Komposition unterschiedlicher Gaußfunktionen modelliert. Damit ist es möglich auch einen leicht dynamischen Hintergrund zu erfassen. Die Anzahl der hierbei verwen-

deten Gaußfunktionen wird allerdings vorher festgelegt und ist für jeden Bildbereich identisch. Dagegen wird bei dem hier vorgestellten Verfahren die Anzahl der Koeffizientenvektoren für den jeweiligen Bildbereich dynamisch an die Gegebenheiten des Hintergrunds angepasst. Dies hat den Vorteil des verminderten Rechenaufwands ohne gleichzeitigen Qualitätsverlust.

### 2.3.2 Vordergrund Separierung

Zur Detektion von sich bewegenden Objekten im Bild müssen für die jeweiligen Bildbereiche die folgenden zwei Bedingungen erfüllt sein:

- Der neu berechnete Koeffizientenvektor muss von allen Vektoren des Modells signifikant abweichen.
- Mindestens einer der benachbarten Bereiche muss in dem vorherigen Bild als 'vom Hintergrund abweichend' markiert worden sein.

In diesem Fall wird der entsprechende Bereich als 'zum Vordergrund gehörend' markiert und in dem resultierenden Bild weiß gekennzeichnet. Die zweite der obigen Bedingungen sorgt dafür, dass neben der Abweichungen vom Hintergrund auch noch die temporalen und räumlichen Charakteristika einer Objektbewegung im Bild ausgenutzt werden. Dies trägt ebenfalls zur Reduzierung des Rauscheinflusses bei. Abbildung 2.6 zeigt beispielhaft eine Originalaufnahme einer Straßenszene und das resultierende Vordergrundbild ohne Berücksichtigung der zweiten Bedingung. Die



(a) Originalaufnahme



(b) Vordergrundbild

Abbildung 2.6: (a) Originalaufnahme einer Straßenszene; (b) Resultierendes Vordergrundbild, ohne Berücksichtigung der zweiten Bedingung. Während der Aufnahmen waren in der Szene keine bewegenden Objekte vorhanden.

Aufnahme erfolgte an einem regnerischen Tag mit kurzen aber häufigen Windstößen. Das im rechten Bild zu sehende Salz und Pfeffer Rauschen war permanent

vorhanden. Erst unter Einbezug der zweiten Bedingung wurden nur relevante Bildänderungen detektiert.

Es sei an dieser Stelle angemerkt, dass zeitgleich mit der ersten 'zum Vordergrund gehörend' - Markierung auch ein so genanntes Vordergrundmodell (FM, foreground model) des jeweiligen Bildausschnittes erzeugt wird. Ähnlich dem Hintergrundmodell besteht das Vordergrundmodell aus Koeffizientenvektoren mit den entsprechenden Gewichten:

$$FM_k = \{v_1, \dots, v_n, \alpha_1, \dots, \alpha_n\}$$

wobei  $k$  hier wieder den betrachteten Bildbereich bezeichnet. Dem Vordergrundmodell werden nur die Vektoren hinzugefügt, die zum einen die beiden obigen Bedingungen erfüllen und zum anderen signifikant von den bereits im Vordergrundmodell enthaltenen Vektoren abweichen. Als 'signifikant abweichend' werden diejenigen Vektoren bezeichnet, die zu allen Modellvektoren einen größeren Winkel haben als der bereits erwähnte Schwellenwert  $S$ . Die Gewichte werden hier ähnlich denen des Hintergrundmodells aktualisiert.

Um Adaptivität des Verfahrens zu gewährleisten, werden nach einer gewissen Zeitspanne nicht mehr bewegende Objekte wieder in den Hintergrund der Szene eingebaut. Das neue Hintergrundmodell wird dabei durch die Vereinigung der beiden Mengen erzeugt:

$$BM_k^{neu} = BM_k^{alt} \cup FM_k$$

Abbildung 2.7 verdeutlicht anhand eines Beispiels den Zusammenhang zwischen den beiden Modellen. Hierbei werden zwei Fälle unterschieden. Im Fall (a) bewegt sich das Objekt, der Smily, pausenlos durch das Kamerabild und gehört damit durchgehend zum Vordergrund der abgebildeten Szene. Im Fall (b) bleibt das Objekt ab dem zweiten Bild in der Mitte des Bildes stehen und wird in den darauf folgenden Bildern in das Hintergrundmodell der Szene eingebaut.

Die Vordergrund- und Hintergrundmodelle neben den jeweiligen Bildern entsprechen den Modellen des rot umrandeten Bildbereichs  $k$  zu den jeweiligen Zeitpunkten. Im zweiten Bild wird in diesem Bereich eine signifikante Änderung festgestellt. Der neu berechnete Koeffizientenvektor  $v_2$  kann nicht dem, im Hintergrundmodell bereits enthaltenen Vektor  $v_1$  zugeordnet werden. Unter der Annahme, dass auch die zweite der beiden oben erwähnten Bedingungen erfüllt ist, wird der Vektor  $v_2$  dem Vordergrundmodell des Bildbereichs  $k$  zugeordnet.

In den darauf folgenden Bildern bleibt im Fall (a) das Hintergrundmodell unverändert, während das Gewicht  $\alpha_2$  des Vordergrundmodells nach und nach verringert wird, bis der entsprechende Vektor aus dem Modell entfernt wird. Im Fall (b) dagegen werden nach einiger Zeit die beiden Modelle miteinander verschmolzen, so dass

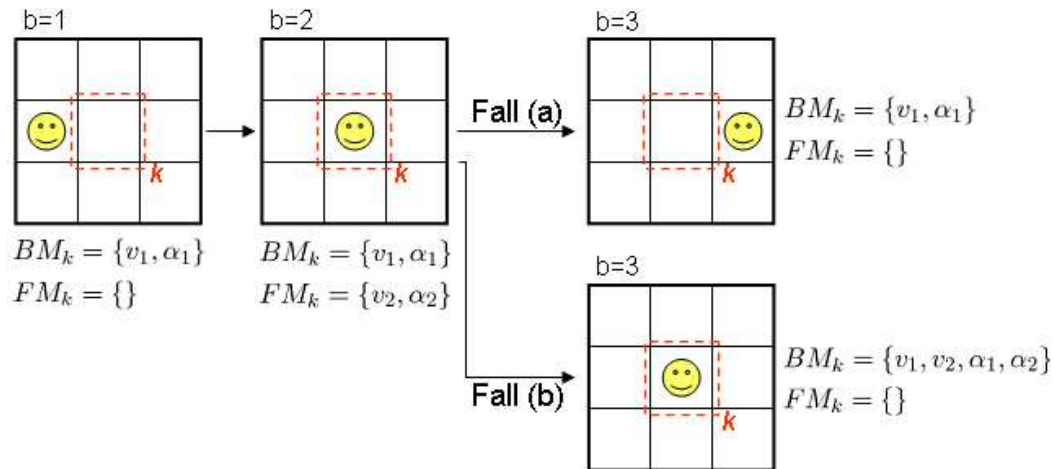


Abbildung 2.7: Graphische Darstellung des Zusammenwirkens zwischen dem Vordergrund- und dem Hintergrundmodell des Bildbereichs  $k$ . Die Zusammenlegung beider Modelle zum 'neuen' Hintergrundmodell des Bereichs ist situationsabhängig.

das 'neue' Hintergrundmodell des Bereiches nun Vektoren des Vordergrundmodells und Vektoren aus dem 'alten' Hintergrundmodell enthält. Der nicht mehr bewegende Smily wird so in das Hintergrundmodell der Szene eingebaut und nicht mehr als ein Vordergrundobjekt detektiert.

Die Zusammenlegung beider Modelle zum 'neuen' Hintergrundmodell erfolgt dabei unter den folgenden Bedingungen:

- Der betrachtete Bildbereich wurde in den letzten  $N$  Bildern kontinuierlich als 'zum Vordergrund gehörend' markiert.
- Mindestens die Hälfte der benachbarten Bereiche wurden im letzten Bild dem Hintergrund zugeordnet.

Auf diese Weise wird sichergestellt, dass die Zusammenlegung beider Modelle erst dann geschieht, wenn sich das Objekt nicht mehr bewegt. Die zweite Bedingung sorgt dafür, dass Teile größerer und gleichzeitig langsam bewegender Objekte nicht in das Hintergrund der Szene eingebaut werden, während sich das Objekt immer noch in Bewegung befindet. Die Abbildung 2.8 veranschaulicht graphisch einen solchen Fall. Aufgrund der langsamen Objektbewegung wird hier der rot umrandete Bildbereich kontinuierlich als 'zum Vordergrund gehörend' markiert. Wählt man in diesem Beispiel  $N = 3$ , so wird die erste der obigen Bedingungen bereits nach dem dritten Bild erfüllt sein. Ohne Berücksichtigung des zweiten Kriteriums, würden im vierten Bild beide Modelle des betrachteten Bildbereichs zum 'neuen' Hintergrundmodell

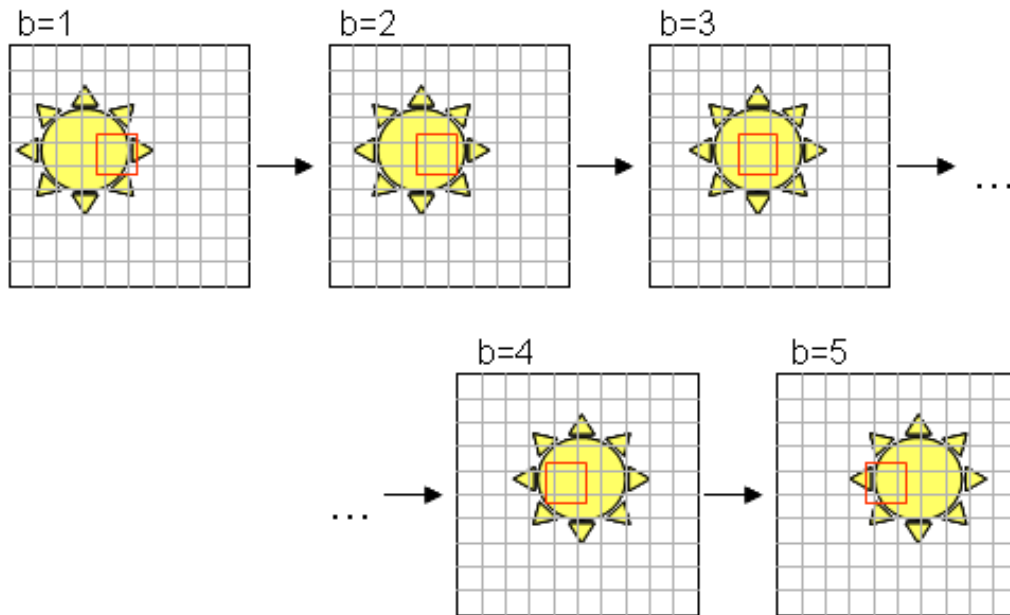


Abbildung 2.8: Graphische Darstellung eines sich langsam bewegenden Objektes. Ohne Berücksichtigung der zweiten Bedingung würden bei einer Wahl von  $N = 3$  Teile dieses Objektes bereits im vierten Bild in den Hintergrund der Szene eingebaut.

zusammengelegt werden, ohne dass das Objekt zum Stehen gekommen ist. Erst das Zusammenspiel beider Bedingungen sorgt dafür, dass Objekte in den Hintergrund der Szene nur dann eingebaut werden, wenn sich diese nicht mehr bewegen.

Der Vorteil des separaten Vordergrundmodells besteht darin, dass neue Vektoren nicht sofort in das Hintergrundmodell eingefügt werden. Im Falle von großen, sich langsam bewegenden Objekten kann es vorkommen, dass das Vordergrund eines Bereiches aus Vektoren besteht, die unterschiedliche Teile des Objekts repräsentieren und im Grunde nichts miteinander zu tun haben. Weil diese Vektoren nicht sofort dem Hintergrundmodell zugeordnet werden, bleibt dieses nach der Objektbewegung szenenecht.

## 2.4 Evaluierung und Ergebnisse

Zur Evaluierung der Performanz des vorgestellten Verfahrens, wurde dieses zusammen mit drei weiteren Verfahren auf unterschiedlichen Testvideos getestet. Dabei wurde untersucht, wie gut die Verfahren den in der Einleitung dieses Kapitels beschriebenen Anforderungen genügen. Die nächsten beiden Abschnitte geben einen Überblick über die zum Vergleich herangezogenen Verfahren und beschreiben kurz

die zur Evaluierung ausgewählten Videosequenzen. Die dabei erzielten Ergebnisse finden sich im letzten Abschnitt.

### **2.4.1 Zum Vergleich herangezogene Verfahren**

Wie bereits im Vorfeld erwähnt findet man in der Literatur eine sehr breite Palette an unterschiedlichen Verfahren zur Vordergrund-Hintergrund Separation. Ein Vergleich dieser Verfahren ist schwierig, denn meistens werden unterschiedliche Sequenzen zur Evaluierung verwendet. Im Rahmen dieser Arbeit wurden insgesamt drei Verfahren ausgewählt, welche sich im Laufe der Zeit bewährt haben und die bereits in vielen Anwendungen eingesetzt werden.

#### **2.4.1.1 Gaußscher Ansatz**

Wie bereits in der Einleitung zu diesem Kapitel erwähnt, erlangte das Verfahren von Stauffer und Grimson [SG00] zur Detektion bewegender Objekte innerhalb weniger Jahre eine erstaunliche Popularität. Hierbei werden die Werte eines jeden Pixels mittels einer Komposition von unterschiedlichen Gaußfunktionen modelliert. Die Parameter der einzelnen Funktionen, Mittelwert und Varianz, werden dabei im Online-Betrieb adaptiert. Dies ermöglicht eine dynamische Anpassung des Modells an die sich ändernde Umstände der abgebildeten Szene und erlaubt eine Modellierung eines nicht statischen Hintergrunds.

Vordergrundobjekte werden detektiert, indem die aktuellen Werte der einzelnen Pixel mit den jeweiligen Modellen verglichen werden. Weichen diese signifikant von dem Modell ab, so wird das Pixel als Vordergrund markiert. Ähnlich wie bei den meisten anderen Verfahren wird hier der minimale Wert einer als signifikant anzusehenden Abweichung mittels eines vorher definierten Schwellenwerts festgelegt. Abhängig von dem gesetzten Wert reagiert das Verfahren sehr sensitiv auf neue, in der vorherigen Bildsequenz noch nicht vorhandene Werte eines Pixels. Leichte, vom Objekt geworfene Schatten werden sofort dem Vordergrund zugeordnet, auch dann wenn das eigentlich Objekt noch nicht im Blickfeld ist.

Um dieses Problem zu mildern, kombinierten KaewTraKulPong und Bowden [KB01] dieses Verfahren mit dem von Horprasert et al. [HHD99]. In seinem Verfahren zur Eliminierung von Schatten, differenziert Horprasert den Farbwert eines Pixels in seinen Farb- und Helligkeitsanteil. Verändert sich nur der Helligkeitsanteil so wird der Pixel nicht dem Vordergrund zugeordnet, sondern als Schatten markiert. Um Rechenkapazitäten zu sparen werden die Farbwerte im RGB-Format beibehalten. Die Details der Symbiose der beiden Verfahren sind in dem Artikel [KB01] gut beschrieben. Zusätzlich zeigen die Autoren, dass das kombinierte Verfahren bessere



Ergebnisse erzielt als das von Stauffer und Grimson. Daher werden die Ergebnisse des hier vorgestellten Verfahrens mit denen des von KaewTraKulPong und Bowden entwickeltem Verfahren verglichen.

#### 2.4.1.2 Bayesscher Ansatz

Das zweite zur Evaluierung verwendete Verfahren wurde 2003 von Li et al. [LHGT03] vorgestellt. Die Vordergrund-Hintergrund Separation erfolgt hier auf der Grundlage, einer dafür formulierten bayesschen Entscheidungsregel. Entscheidend für die Klassifizierung ist eine, über eine Bildsequenz gewonnene Statistik, über die Auftrittswahrscheinlichkeit verschiedener Merkmalsvektoren eines Pixels. Hierbei unterscheiden die Autoren zwischen zwei Arten von Vektoren. Die einen dienen zur Repräsentation eines statischen, die anderen zur Modellierung eines dynamischen Hintergrunds. Die Detektion von Vordergrundobjekten erfolgt mittels der bereits erwähnten Entscheidungsregel. Dabei wird berücksichtigt, in wie fern die neu berechneten Merkmalsvektoren dem statischen oder dem dynamischen Hintergrund zugeordnet werden können.

Die in [LHGT03] präsentierten Ergebnisse dieses Verfahrens sind beeindruckend. Der Ansatz erlangte aufgrund seiner Stabilität ebenfalls einen großen Zuspruch. Unter anderem wurde es auch in der frei verfügbaren OpenCV Bibliothek <sup>1</sup> zur Bildverarbeitung implementiert. Dies trug ebenfalls entscheidend zu seiner Verbreitung in der Bildverarbeitungsgemeinde bei.

#### 2.4.1.3 Kantenbasierter Ansatz

Als dritter Ansatz wurde im Rahmen dieser Arbeit ein kantenbasiertes Vordergrund-Hintergrund Separationsverfahren ausgewählt. Ähnlich wie viele andere Verfahren dieser Klasse basiert auch dieses auf dem erstmalig im Jahr 1986 präsentierten Canny-Kantendetektor [Can86]. Chaohui et al. verbinden in ihrer Arbeit [CXS<sup>+</sup>07] die Ergebnisse des Canny-Detektors mit der einfachen und schnellen Methode des Frame Differencing. Dabei wird zunächst auf jedem neu aufgenommenen Bild mittels des Canny-Algorithmuses Kanten extrahiert. Die Detektion von bewegten Objekten erfolgt anschließend durch Subtraktion solch zweier aufeinander folgender Kantensbilder.

Das Verfahren ist einfach und benötigt wenig Implementierungsaufwand. Auch die von Autoren mehrmals erwähnte geringe Rechenkapazität ist ein weiterer dem Verfahren zu zurechnender Vorteil. Die Performanz der dabei erzielten Ergebnisse basiert allerdings sehr stark auf den für die Kantendetektion gewählten Parame-

---

<sup>1</sup>Die OpenCV Bibliothek ist verfügbar unter dem Link: <http://opencv.willowgarage.com/wiki/>

tern. Eine schlechte Parameterwahl kann zu einem zu sehr vielen kleinen und oft unerwünschten Kanten führen. Zum anderen kann es vorkommen, dass bei falsch gesetzten Parametern relevante Kanten im Bild nicht extrahiert werden. Aufgrund der in diesem Verfahren durchgeführten Differenzbildung der zwei aufeinander folgenden Kantenbildern, wären die sehr vielen kleinen Kanten im Hintergrund nicht problematisch. Ein Problem stellt dagegen die Instabilität dieses Verfahrens gegenüber leichten Änderungen in den Pixelwerten. Aufgrund des Rauschens ist oft die Anzahl oder die Länge der extrahierten Kanten in den aufeinander folgenden Bildern nicht gleich. Dies resultiert in einem verrauschten Vordergrundbild. Zur Behebung dieses Problems schlagen die Autoren vor, das resultierende Bild in gleichgroße Bereiche aufzuteilen und über die Anzahl der weißen Pixel in dem jeweiligen Bereich zu entscheiden, ob dieser eine bewegte Kante darstellt. Die Setzung des entsprechenden Schwellenwerts hat sich allerdings als sehr schwierig herausgestellt. In den nachfolgenden Abschnitten wird dieser Punkt noch genauer erläutert.

Der in letzter Zeit allgemeine Trend in Richtung der kantenbasierten Ansätze für die Vordergrund-Hintergrund Separation liegt in deren Stabilität gegenüber sich ändernden Lichtverhältnissen. Trotz der erwähnten Nachteile bei dem zuletzt vorgestellten Verfahren, ist dieses weitaus robuster gegenüber plötzlichen Lichtänderungen als die ersten beiden Verfahren. Bei Beleuchtungsänderungen in der betrachteten Szene bleibt die Kanteninformation erhalten. Die ersten beiden Verfahren haben allerdings den Vorteil des kaum vorhandenen Rauschens in den resultierenden Vordergrundbildern.

Die Performanz des im Rahmen dieser Arbeit entwickelten Verfahrens wird mit den Ergebnissen der drei beschriebenen Ansätze verglichen. Die frei wählbaren Parameter der Verfahren wurden entsprechend den Empfehlungen der Autoren gesetzt. Bei einigen Verfahren wurden nach einem groben Durchlauf der gesamten Experimentenreihe die Parameter noch weiter angepasst, um ein bestmögliches Ergebnis zu erzielen. Die Evaluierung der Ergebnisse während des groben Durchlaufs basierte nicht auf einem Vergleich der von den Verfahren erzielten Resultate, sondern folgte lediglich der subjektiven Wahrnehmung der resultierenden Vordergrundbilder. Die während dieses Prozederes festgelegten Parameter wurden in den darauf folgenden Experimenten nicht mehr verändert.

Der nächste Abschnitt gibt einen Überblick über die dazu verwendeten Videos.

### 2.4.2 Testvideos zur Evaluierung der Verfahren

Die Auswahl der Testsequenzen basierte überwiegend auf den bereits in der Einleitung erwähnten Anforderungen an das Verfahren. Jede einzelne Sequenz entspricht

einem solchen Präzedenzfall.

Anhand eines eigenständig erstellten Videos aus dem Bereich der Parkplatzüberwachung wurde untersucht, wie gut das Verfahren das entsprechende Hintergrundmodell bei zeitweisen Verschiebungen von Objekten aktualisiert. Dazu wurden Aufnahmen des Parkplatzes verwendet, in denen einige der Stellplätze bereits mit Fahrzeugen besetzt waren. Die Abbildung 2.9 zeigt einige Bilder der hierzu verwendeten Sequenz. Nach einigen Sekunden startet das weiße Auto rechts im Bild und verlässt

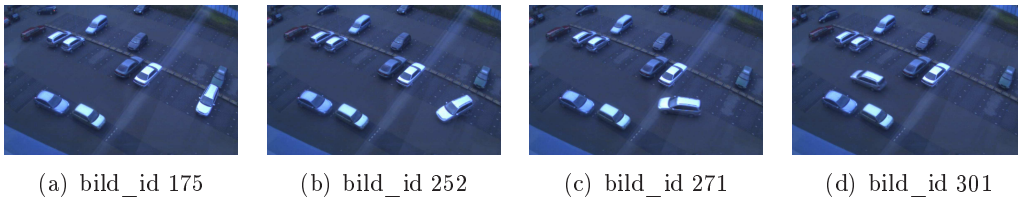


Abbildung 2.9: Einige Aufnahmen aus dem Video der Parkplatzüberwachung. Anhand dieses Videos wurde untersucht wie gut das hier vorgestellte Verfahren das Hintergrundmodell der Szene bei zeitweisen Verschiebungen von Objekten aktualisiert.

langsam den Stellplatz. Das Video beginnt mit der Aufnahme des Parkplatzes zu dem Zeitpunkt als das Auto noch auf dem Parkplatz steht.

Zur Evaluierung der Verfahren während sich ändernder Lichtverhältnisse wurden zwei Videosequenzen verwendet. Das erste Video [LiT] wurde mittels einer Deckenkamera aufgenommen und zeigt einen Aufenthaltsbereich in einem Bürogebäude. Zur Simulation plötzlicher Lichtänderungen wird hier während der Videoaufnahme die Deckenbeleuchtung aus- und eingeschaltet. Abbildung 2.10 zeigt einige Bilder dieser Sequenz. Ab und zu sind auf dem Video auch Personen zu sehen, die sich von

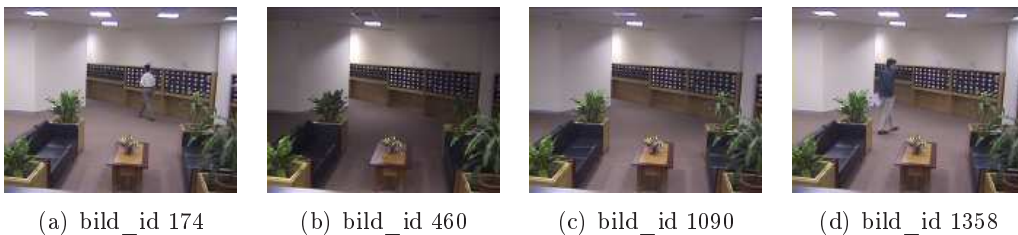


Abbildung 2.10: Einige Bilder aus den Aufnahmen eines Aufenthaltsbereich in einem Bürogebäude. Anhand dieses Videos [LiT] wurde untersucht wie stabil die Verfahren gegenüber plötzlichen Lichtänderungen sind.

einer Ecke in die andere bewegen. Bei der Evaluierung der Verfahren wurde daher auch untersucht wie gut die Personen bei schwacher Beleuchtung extrahiert werden können.

Das zweite Video [KT99] zeigt eine Aufnahme eines Wohnzimmers, wobei sich hier die Beleuchtung nur sukzessive ändert. Abbildung 2.11 zeigt einige Bilder dieser Sequenz. Das Video startet und endet mit einem schwarzen Bild auf dem nichts



Abbildung 2.11: Einige Bilder aus den Aufnahmen eines Wohnzimmers. Anhand dieses Videos [KT99] wurde die Robustheit der Verfahren gegenüber sukzessiven Lichtänderungen untersucht.

zu erkennen ist. Zwischenzeitlich wird die Beleuchtungsintensität erhöht und dann wieder reduziert. Wie den Bildern (c) und (d) zu entnehmen ist, müssen auch hier Personen vom Hintergrund separiert werden.

Um zu testen wie gut die Verfahren mit einem dynamischen Hintergrund der Szene umgehen können, wurde eine Sequenz [KT99] mit einem relativ stark schwankenden Baum im Hintergrund ausgewählt. Abbildung 2.12 zeigt dazu einige Bilder. Einen Großteil des Videos bildet der in der Mitte des Bildes schwankender Baum.

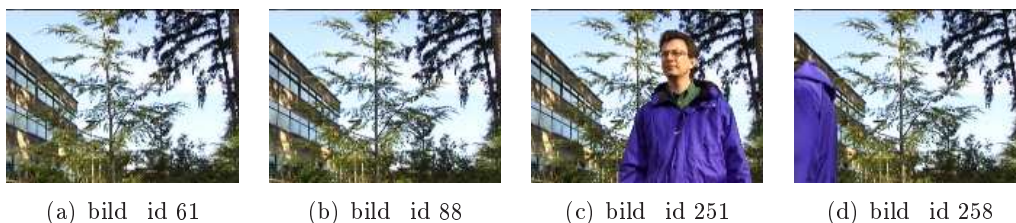


Abbildung 2.12: Einige Aufnahmen aus dem Video mit einem schwankenden Baum. Anhand dieses Videos [KT99] wurde getestet, wie gut die Verfahren mit einem dynamischen Hintergrund einer Szene umgehen können.

Weil seine Bewegung zeitweise periodisch ist, müssten die Verfahren diesen in das Hintergrundmodell einbauen. Gegen Ende der Sequenz taucht eine Person auf und verschwindet dann wieder aus dem Bild. Es ist zu erwarten, dass die getesteten Verfahren die Person sauber aus dem Bild extrahieren.

Zusätzlich wurde die Performanz der Verfahren anhand einiger Videos aus der PETS Datenbank [PET05] evaluiert. Die Videos wurden mittels einiger Deckenkameras auf einem Bahnhof aufgenommen. Abbildung 2.13 zeigt beispielhaft einige Bilder einer solchen Sequenz. Anhand dieses Videos wurde zum einen untersucht

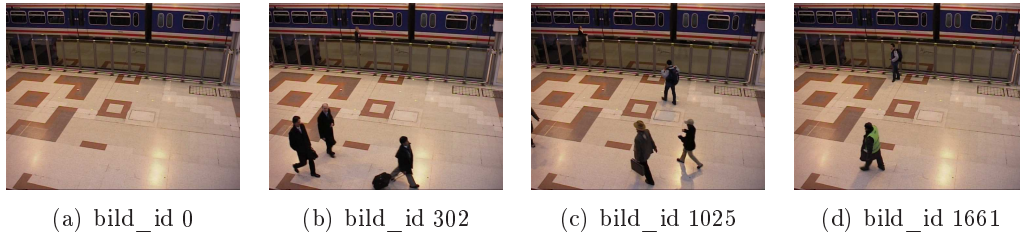


Abbildung 2.13: Einige Aufnahmen aus dem Bereich der Bahnhofüberwachung. Anhand dieses Videos [PET05] wurde untersucht wie gut die Verfahren Personen vom Hintergrund der Szene extrahieren können, auch wenn nur wenig Kontrast vorhanden ist.

wie stabil die Verfahren Personen extrahieren können, auch wenn kaum Kontrast zwischen dem Hintergrund und der sich bewegenden Person vorhanden ist. Während im unteren Bildbereich die Personen deutlich zu erkennen sind, weichen im oberen Bereich des Bildes die Farbwerte deutlich weniger von einander ab. Nur die zwei weißen Streifen am Zug sorgen hier für hellere Farbtupfer. Zum anderen wurde anhand dieses Videos überprüft, wie gut die von Personen geworfene Schatten ausselektiert werden. Aufgrund der Beleuchtung sind in dieser Sequenz in fast jedem Bild Schatten vorhanden. Besonders gut sind diese in Abbildung 2.13 (c) zu sehen.

Die bei den Experimenten erzielten Ergebnisse werden im nachfolgenden Abschnitt präsentiert.

### 2.4.3 Ergebnisse der durchgeführten Experimente

Dieser Abschnitt gliedert sich in die folgenden fünf Unterabschnitte: 1. Adaptivität des Hintergrundmodells bei zeitweisen Verschiebungen von Objekten, 2. Stabilität der Verfahren bei plötzlichen Lichtänderungen, 3. Stabilität der Verfahren bei sukzessiven Beleuchtungsänderungen, 4. Robustheit der Verfahren bei einem dynamischen Hintergrund der Szene und 5. Stabilität der Verfahren beim kaum vorhandenen Kontrast zwischen den Vordergrundobjekten und dem Hintergrund der Szene. Jedem dieser Abschnitte wurde jeweils eines der bereits oben erwähnten Testvideos zugrunde gelegt.

#### **Adaptivität des Hintergrundmodells bei zeitweisen Verschiebungen von Objekten.**

Der Videoausschnitt von der Parkplatzüberwachung wurde herangezogen, um zu untersuchen wie gut der im Rahmen dieser Arbeit entwickelte Ansatz das Hintergrundmodell der Szene bei einer Objektbewegung aktualisiert. Dabei liegt die Schwierigkeit bei diesem Szenario in der nichtvorhandenen Trainingsphase, d.h. hier

wird der Fall simuliert, dass keine Aufnahmen von dem nackten Hintergrund der Szene vorhanden sind. Abbildung 2.14 zeigt dazu einen Ausschnitt der mit dem hier entwickelten Verfahren erzielten Ergebnisse. Die als Vordergrund markierte Bereiche

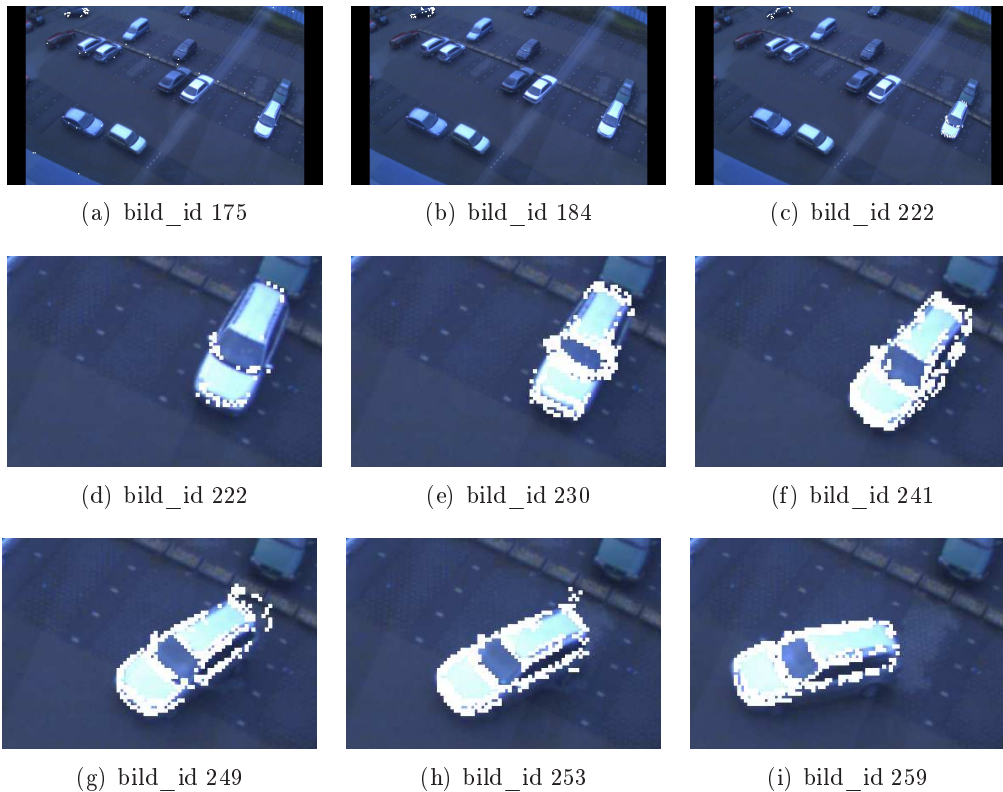


Abbildung 2.14: Ergebnisse des im Rahmen der Arbeit entwickelten Ansatzes im Bereich der Parkplatzüberwachung. Die Vordergrundobjekte bilden hier alle durch weiße Rechtecke markierte Bereiche.

sind in dieser Abbildung durch weiße Quadrate gekennzeichnet. Weil die in diesem Experiment betrachtete Sequenz mit der Bildnummer 175 startet, ist im Bild (a) noch ein wenig Rauschen zu sehen. Nach etwa zehn Bildern hat sich das Verfahren an die Gegebenheiten der Szene angepasst und markiert nur das in der linken oberen Ecke bewegte Fahrzeug durch weiße Quadrate. Mit der Bildnummer 222 setzt sich das rechte hellere Auto in Bewegung. Um einen besseren Einblick zu erhalten, zeigen Abbildungen (d)-(i) eine Nahaufnahme dieses Fahrzeugs.

Weil das Verfahren auf Strukturänderungen im Bild reagiert, sind die homogenen Flächen des Autos wie Dach oder die Motorhaube nicht markiert. Stattdessen bilden die weißen Quadrate die Umrandungen des Fahrzeugs. Im Bild (g) und (h) sind noch einige weiße Flecken auf dem Stellplatz zu sehen. Doch schon nach weiteren sechs Aufnahmen hat sich das Verfahren an die geänderten Bedingungen angepasst.

Der vorher vom Auto verdeckte Stellplatz wurde robust in das Hintergrundmodell eingebaut. In Abbildung (i) ist deutlich zu erkennen, dass auf dem freigewordenen Platz keine weißen Quadrate mehr zu sehen sind, obwohl sich das Fahrzeug noch halb auf dem Stellplatz befindet.

Zwischen den Bildnummern 184 und 195 wurden noch an einigen Stellen Ausreißer registriert. Insgesamt tauchten auf die elf Bilder verteilt etwa fünf weiße Quadrate an falschen Stellen auf. Dies ist auf die, zu diesem Zeitpunkt noch stattfindende Adaption des Verfahrens zurückzuführen. Bei den durchgeführten Experimenten wurde festgestellt, dass das Verfahren bei einem statischen Hintergrund etwa 20 bis 30 Bilder zur Adaption benötigt. Nach dem Abschluss dieser online stattfindenden Trainingsphase wurden üblicherweise keine Ausreißer mehr registriert.

Der von Li et. al. präsentierte bayessche Ansatz liefert auf dieser Sequenz ähnlich gute Ergebnisse. Bei den restlichen zwei Verfahren konnte deren spezifisches Rauschen in den Vordergrundbildern beobachtet werden. Im Folgenden wird auf dieses genauer eingegangen. Weil bei diesem Testszenario in den resultierenden Ergebnissen nichts Außergewöhnliches zu finden war, wird an dieser Stelle auf den expliziten Vergleich der Ergebnisse aller vier Verfahren verzichtet.

#### **Stabilität der Verfahren bei plötzlichen Lichtänderungen.**

In dem Testszenario der sich plötzlich wechselnden Lichtverhältnisse konnten dagegen interessante Unterschiede in den Ergebnissen der vier Verfahren beobachtet werden. In diesem Experiment wurde ein Video einer Deckenkamera in einem Bürogebäude verwendet. Die Sequenz startet mit den Aufnahmen eines gleichmäßig beleuchteten Aufenthaltsraums. Die ersten 150 Bilder zeigen den Raum ohne bewegte Objekte. Somit haben alle Verfahren ausreichend Information um sich ein entsprechendes Hintergrundmodell aufzubauen. Abbildung 2.15 zeigt die Ergebnisse der vier Verfahren kurz nachdem das Licht ausgeschaltet wurde.

Abbildungen (b)-(e) zeigen die resultierenden Vordergrundbilder der jeweiligen Verfahren. Es ist deutlich zu sehen, dass bessere Ergebnisse mit dem hier vorgestellten Verfahren und dem kantenbasierten Verfahren erzielt wurden. Es ist allerdings anzumerken, dass obwohl das von Chaohui et al. [CXS<sup>+</sup>07] vorgeschlagene Verfahren sehr robust gegenüber Lichtänderungen ist, wurde trotzdem in allen Bildern der Sequenz ein gewisser Rauschanteil beobachtet. Dieser ist in Abbildung (e) zu sehen. Das Ein- oder Ausschalten der Lampen führt hier zu keinen spürbaren Änderungen in dem Rauschanteil. Wie bereits erwähnt, schlagen die Autoren vor das Rauschen über einen bestimmten Schwellenwert über die Anzahl der vom Schwarz abweichenden Pixel in einem bestimmten Bereich zu unterdrücken. Das Bestimmen

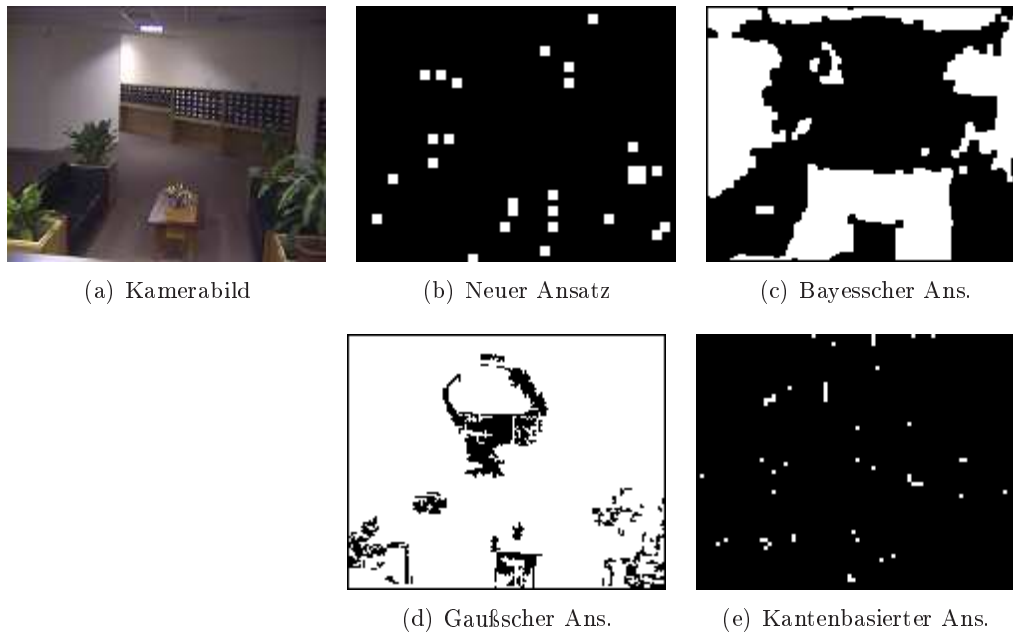


Abbildung 2.15: Vergleich der Ergebnisse kurz nachdem das Licht ausgeschaltet wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

des richtigen Schwellenwerts hat sich allerdings als sehr schwierig erwiesen. Teilt man das Bild in  $n \times n$  große Bereiche auf, so ist oft die Anzahl der durch Rauschen verursachten weißen Pixel in einem Bereich vergleichbar mit der Anzahl der kantenrepräsentierenden Pixel in einem anderen Bereich. Mit dem Setzen eines höheren Schwellenwerts kann zwar Rauschen unterdrückt werden, doch hat dies oft zur Folge, dass die bewegenden Objekte als solche von dem Verfahren nicht detektiert werden. Ein kleinerer Schwellenwert führt zwar zu einer Reduzierung des Rauschanteils, doch ist das Ergebnis für reale Anwendungen oft nicht ausreichend. Um die Ergebnisse des Verfahrens mittels eines Schwellenwerts nicht zu manipulieren, wurde hier darauf verzichtet dieses anzuwenden. Sowohl in Abbildung 2.15 (e) als auch in den nachfolgenden Abbildungen, sind die Ergebnisse dieses Verfahrens ohne Berücksichtigung eines Schwellenwerts dargestellt.

Aufgrund seiner Einfachheit demonstriert das Verfahren von Chaohui ein grundlegendes Problem der kantenbasierten Ansätze. Eine robuste und exakte Extraktion der Kanten ist eine der fundamentalen Voraussetzungen für eine gute Performanz des Verfahrens. Viele Algorithmen zur Kantenextraktion erfordern allerdings oft eine sehr sorgfältige und teilweise auch manuelle Feinabstimmung der entsprechenden Parameter. Mehr noch abhängig von der Anwendung und dem verwendeten Ver-



fahren sind die Ergebnisse der Kantenextraktion sehr abhängig von den gewählten Parametern. Im Falle des Canny-Kantendetektors und der hier verwendeten Testvideos konnte dies sehr genau beobachtet werden.

Wie der Abbildung 2.15 zu entnehmen ist, sind die beiden Ansätze von Li und KeawTraKulPong nicht für den Einsatz unter solchen extremen Bedingungen geeignet. Bei dem erweiterten Gaußschen Ansatz führte das plötzliche Ausschalten des Lichtes zu einer fast kompletten Konvertierung des Bildes in ein Vordergrundbereich. Ähnliches kann auch in Abbildung (c) beobachtet werden. Bei dem bayesschen Ansatz von Li wurde zwar nicht das komplette Bild, jedoch schon größere Bereiche des Bildes in ein Vordergrundbereich konvertiert.

Außer dem Verfahren von Chaohui besitzen die restlichen drei Ansätze einen Adaptionsmechanismus. Die Anpassung der Verfahren an geänderte Bedingungen dauerte unterschiedlich lange. Der im Rahmen dieser Arbeit entwickelte Ansatz benötigte hierfür nur 13 weitere Bilder. Das Verfahren von KeawTraKulPong brauchte deutlich länger, 201 Bilder. Die meiste Zeit benötigte allerdings das Verfahren von Li, insgesamt 260 Bilder. Die Adaptionsphase dieses Verfahrens endete erst mit der Bildnummer 1709.

Abbildung 2.16 zeigt die Ergebnisse der Verfahren bei dem 1648 Kamerabild. Es ist deutlich zu sehen, dass hier die Adaptionsphase des Verfahrens von Li noch nicht abgeschlossen ist. Auch das Verfahren von KeawTraKulPong braucht zu diesem Zeitpunkt noch zwei weitere Bilder zur vollständigen Anpassung.

Die Abbildung demonstriert sehr gut die Fähigkeit der Verfahren bewegte Objekte auch auf einem dunklen Hintergrund zu extrahieren. Während das hier vorgestellte Verfahren zusammen mit dem von KeawTraKulPong sehr gute Ergebnisse liefert, kann bei den beiden anderen Vordergrundbildern die laufende Person kaum erkannt werden. Nur unter Einbezug einiger vorherige Vordergrundbilder können im Bild (c) aufgrund der Bewegung die zwei kleinen mittleren weißen Flecken als die Person identifiziert werden. Ist ist allerdings anzumerken, dass das Verfahren von Li in diesem Fall es nicht schafft die Person als Ganzes zu extrahieren. Nur einige Bereiche am Kopf und an den Beinen werden hier vom Hintergrund separiert.

Bei dem Ergebnis des kantenbasierten Verfahrens ist die bereits erwähnte Problematik deutlich zu erkennen. Das in der Mitte des Bildes etwas dichtere Cluster an Punkten bildet die sich bewegende Person. Schon ein geringer Schwellenwert zur Unterdrückung des umliegenden Rauschen würde hier dazu führen, dass auch die wichtigen Teile der Person ausselektiert werden.

Ähnliches kann auch in Abbildung 2.17 beobachtet werden. Die Adaptionsphase ist hier bei allen Verfahren abgeschlossen. Wieder kann die Person im Kamerabild nur in den beiden Vordergrundbildern (b) und (d) deutlich erkannt werden. Während

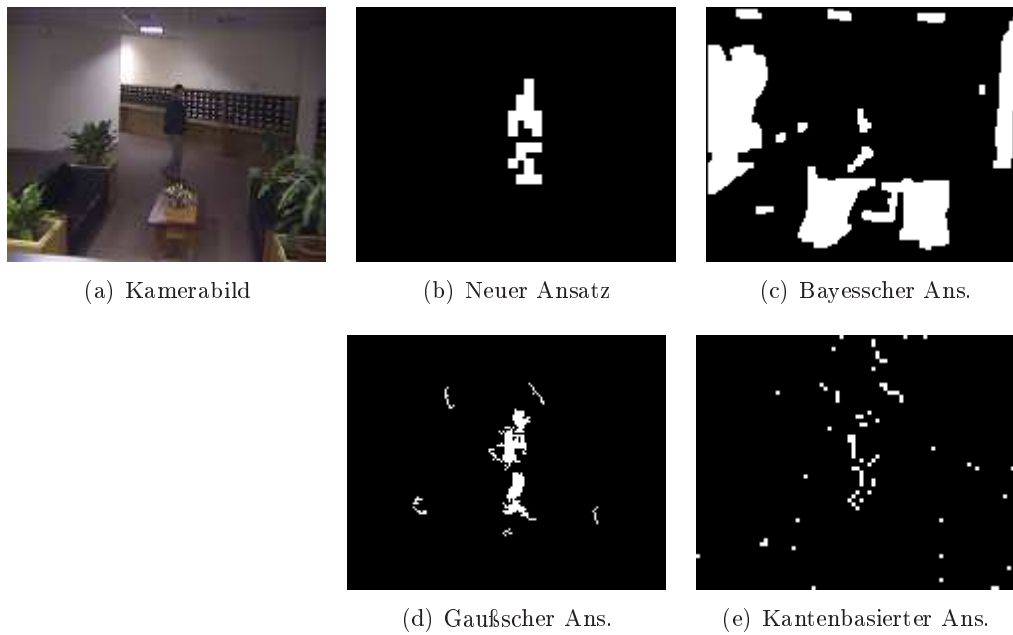


Abbildung 2.16: Detektion von Personen im abgedunkelten Raum, kurz nachdem das Licht ausgeschaltet wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

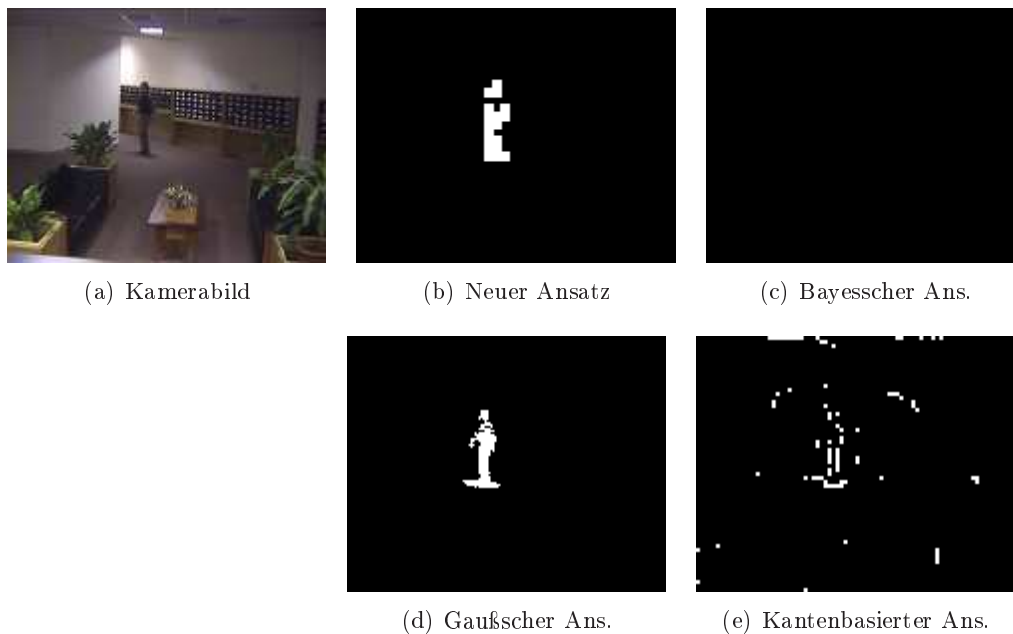


Abbildung 2.17: Detektion von Personen im abgedunkelten Raum nachdem die Adaptionsphase bei allen Verfahren abgeschlossen wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

im Bild (e) sich nur ein schwaches Cluster an Punkten abzeichnet, ist die Abbildung (c) komplett schwarz. Aufgrund des mangelnden Kontrastes im Kamerabild schafft das Verfahren von Li nicht die Person zu extrahieren. In den beiden vorherigen Vordergrundbildern konnte lediglich jeweils ein kleiner weißer Fleck beobachtet werden.

Abbildungen 2.18 und 2.19 zeigen die Situation nach dem erneuten Einschalten der Lampen. Im Gegensatz zu anderen Verfahren, bleibt bei dem hier vorgestellten

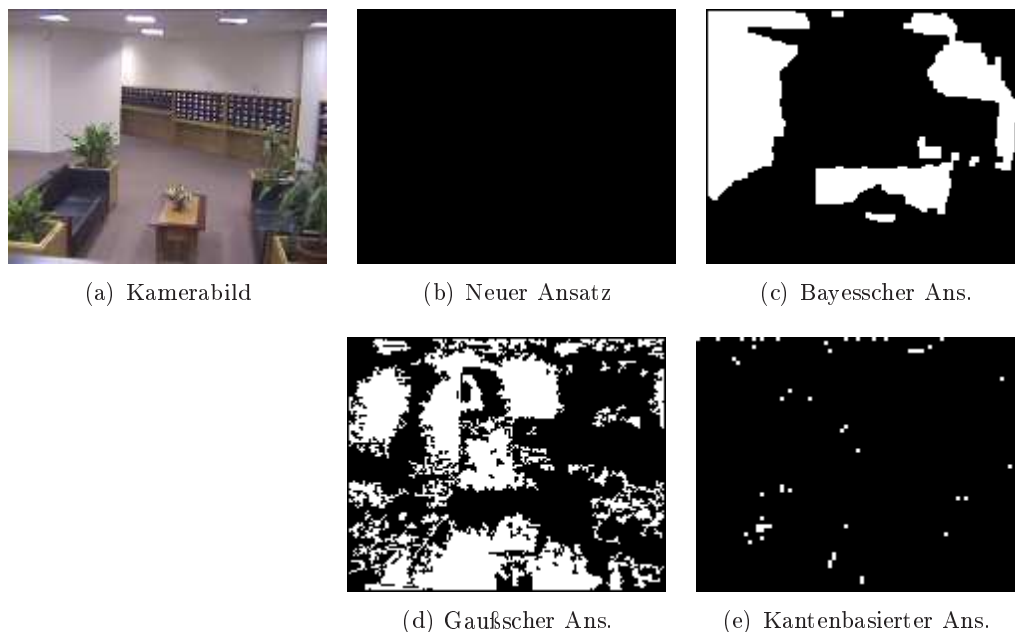


Abbildung 2.18: Vergleich der Ergebnisse kurz nachdem das Licht wieder eingeschaltet wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

Ansatz das Vordergrundbild schwarz. Erneute plötzliche Änderung der Lichtverhältnisse führt hier nicht wie bei den Verfahren von KeawTraKulPong und Li zu fast kompletten Weißfärbung des Bildes. Auch bei der erneuten Detektion von Personen ist der neue Ansatz den anderen überlegen. Während das Verfahren von KeawTraKulPong ein immer noch sehr verrauschtes Vordergrundbild liefert, erkennt das Verfahren von Li nur drei nicht zusammenhängende Flecken. Wie sich auch bei späteren Experimenten zeigen wird, schwächelt das Verfahren von Li immer dann wenn in einer Bildsequenz in den dunkleren Farbbereichen zwischen dem bewegenden Objekt und dem Hintergrund nicht genug Kontrast vorhanden ist. Das kantenbasierte Verfahren von Chaohui liefert hier und auch in den nachfolgenden Experimenten nur ein sich leicht abzeichnendes Cluster in der Mitte des Bildes.

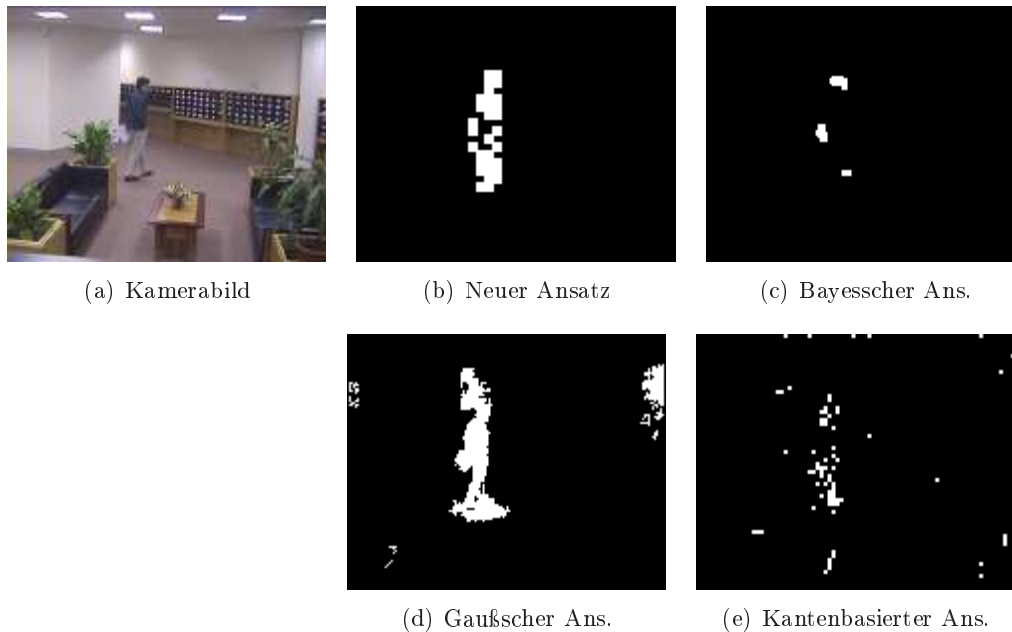


Abbildung 2.19: Erneute Detektion von Personen in dem nun wieder beleuchteten Raum: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

### Stabilität der Verfahren bei sukzessiven Beleuchtungsänderungen.

Im nächsten Testszenario wurde die Performanz der Verfahren bei sich nur sukzessive ändernden Lichtverhältnissen getestet. Die erzielten Ergebnisse des hier vorgestellten Verfahrens und des bayesschen Ansatzes von Li sind bei diesem Experiment vergleichbar. Beide konnten sehr gut mit sich langsam ändernden Lichtbedingungen umgehen. Bei beiden Verfahren änderte sich das Vordergrundbild nur wenn eine Person in der Szene erschien. Abbildung 2.20 zeigt dazu beispielhaft die Ergebnisse der Verfahren bei der Bildnummer 1845. Im Bild (b) und (c) ist die sich bewegende Person deutlich zu erkennen. Aufgrund des pixelbasierten Modells sind die Konturen bei dem Verfahren von Li deutlich besser ausgebildet als bei dem hier vorgestellten Verfahren. Es ist allerdings anzumerken, dass im Bild (c) wegen des nicht vorhandenen Kontrastes die Beine der Person abgeschnitten sind. Der neue Ansatz dagegen detektiert diese genau so robust wie den Rest des Körpers.

Der erweiterte Gaußsche Ansatz von KeawTraKulPong schwächelte am Anfang des Videos. Wie dem Bild (d) zu entnehmen ist, ist im Vordergrundbild dieses Verfahrens noch sehr viel Rauschen enthalten. Die Geschwindigkeit der ansteigenden Lichtintensität entsprach in diesem Fall nicht den für die Adaption gewählten Parameterwerten. Diese wurden allerdings bewusst während der hier durchgeführten

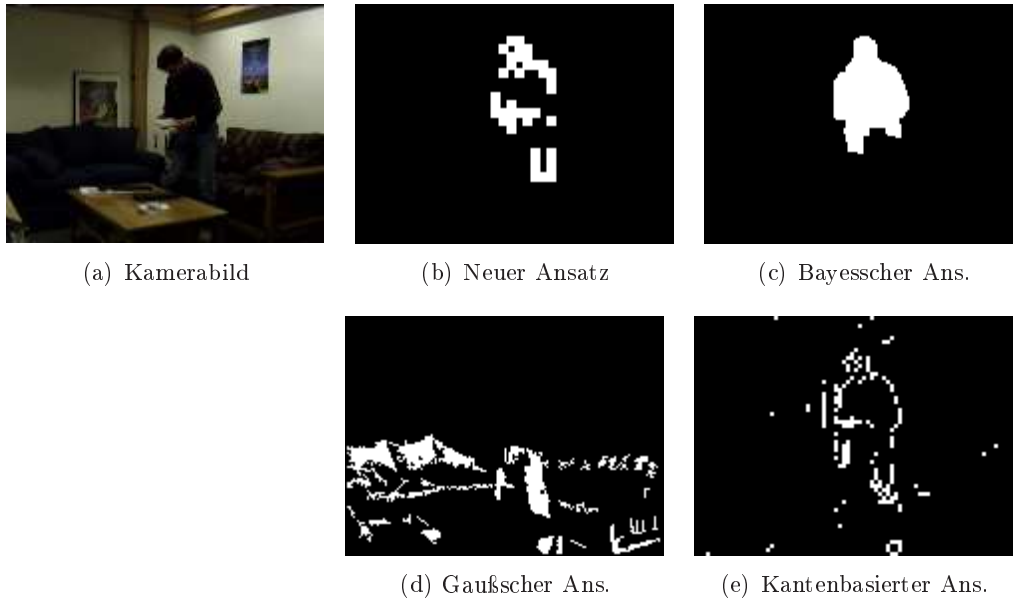


Abbildung 2.20: Vergleich der Detektionsergebnisse bei sukzessiven Lichtänderungen: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

Experimentenreihe nicht geändert. Zusätzlich ist anzumerken, dass die sich im Raum befindende Person von dem Verfahren nicht erfasst wird. Aufgrund des Lichtanfalls ist die Person im Bild fast ausschließlich durch dunklere Farbpigmente dargestellt. Die Intensität dieser Farbwerte stimmt in diesem Fall mit den Werten, die in den vorherigen Bildern als Hintergrundmodell gespeichert wurden. Wie bereits erwähnt fängt das Video mit einem schwarzen Bild an. Eine andere Wahl der Adaptionsparameter würde dieses Problem sicherlich beheben. Doch demonstriert diese Situation deutlich die Sensitivität des Verfahrens gegenüber der richtigen Kalibrierung der Parameter.

Beim kantenbasierten Ansatz kann aufgrund des starken Kontrastes zwischen der Person und dem Hintergrund so etwas wie eine Kontur in dem Vordergrundbild erkannt werden. Allerdings hat man auch hier das bereits erwähnte Problem des umliegenden Rauschens.

### **Robustheit der Verfahren bei einem dynamischen Hintergrund der Szene.**

Die folgende Abbildung 2.21 zeigt einen Teil der Ergebnisse des nächsten Experimentes. Hier wurde untersucht wie gut die Verfahren mit einem dynamischen Hintergrund umgehen können. Die dazu verwendete Sequenz besteht größtenteils aus

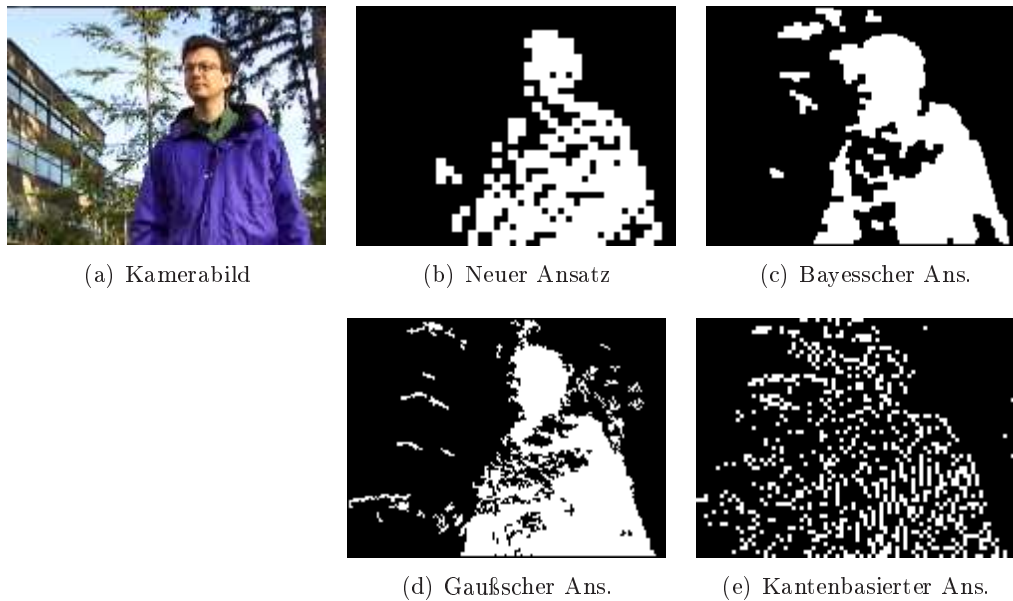


Abbildung 2.21: Performanz der Verfahren bei einem dynamischen Hintergrund der Szene: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

einem in der Mitte des Bildes schwankenden Baums. Die Abbildung 2.21 zeigt den Moment als eine Person im Bild erscheint. Bis zu diesem Zeitpunkt hatten alle Verfahren die gleiche Informationsmenge zur Adaption derer Parameter.

Das erweiterte Gaußsche Verfahren von KeawTraKulPong erreichte schon nach wenigen Bildern seinen stabilen Zustand. In jedem Bild wurde ungefähr immer der gleiche Rauschanteil beobachtet wie er im Bild (d) zu sehen ist. Eine Verbesserung konnte bei dieser Sequenz auch mit unterschiedlichen Parameterwerten nicht erzielt werden. Das Verfahren von Li zeigte hier einen vergleichbaren Adaptionsverlauf. Zwar konnte zwischenzeitlich, abhängig von dem Ausmaß der Baumschwingung eine Reduzierung des Rauschanteils beobachtet werden, doch hielt diese nicht lange an. Ähnliches gilt auch für das im Rahmen dieser Arbeit vorgestellte Verfahren. Allerdings konnte hier im Unterschied zum Verfahren von Li, zeitweise eine komplette Rauschunterdrückung erzielt werden. Auch in der als Referenz herangezogenen Abbildung 2.21 ist deutlich zu sehen, dass der Rauschanteil im Bild (b) geringer ist als bei den anderen Verfahren.

Um die Dynamik des Modells bei dem hier vorgestellten Verfahren zu demonstrieren, wurden die Modellgrößen einiger bei dieser Sequenz zufällig ausgewählter Bildbereiche protokolliert. Abbildung 2.22 zeigt die Größenänderungen der Modelle dreier solcher Bereiche. Abhängig von der Schwingung des Baumes steigt und

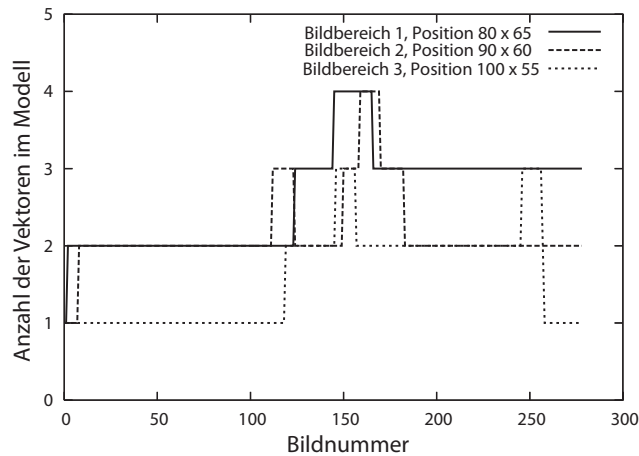


Abbildung 2.22: Graphische Darstellung der Größenänderung der Hintergrundmodelle einiger ausgewählter Bereiche. 'Position  $n \times m$ ' bezeichnet die Position des entsprechenden Bereiches im Bild. Alle ausgewählten Bereiche repräsentieren den dynamischen Teil des Hintergrunds.

sinkt die Anzahl der im Modell verwendeten Vektoren. Dies zeigt deutlich die bereits erwähnte Adaptivität des Verfahrens gegenüber sich ändernden Eigenschaften der abgebildeten Szene. Bei Aufnahmen einer statischen Szene besteht ein Modell aus durchschnittlich etwa 1.3 Vektoren. Diese Zahl hängt allerdings entscheidend von den Eigenschaften der verwendeten Kamera ab. Produziert diese eine schlechte Bildqualität, so werden bei gleicher Szene mehr Vektoren zur Modellierung des Hintergrunds benötigt als bei einer besseren Kamera der Fall wäre.

Aufgrund der adaptiven Anpassung des Modells ist auch die erforderliche Rechenkapazität bei diesem Verfahren nur schwer feststellbar. Diese hängt entscheidend von der jeweiligen Anwendung und der verwendeten Hardware ab.

### **Stabilität der Verfahren beim kaum vorhandenen Kontrast zwischen den Vordergrundobjekten und dem Hintergrund der Szene.**

Für den letzten Teil der durchgeführten Experimentenreihe wurde ein Testvideo aus der PETS Datenbank herangezogen. Abbildungen 2.23 und 2.24 zeigen einen Teil der hierbei erzielten Ergebnisse. Anders als der bayessche Ansatz von Li, brauchte das hier vorgestellte Verfahren eine anfängliche Adaptionsphase von insgesamt 19 Bildern. Nachdem allerdings alle Modelle für alle Bereiche aktualisiert wurden, wurde während der gesamten restlichen Sequenz kein Rauschen mehr detektiert. Das Verfahren von Li war dagegen bei dieser Sequenz sofort einsetzbar.

Wie der Abbildung 2.23 zu entnehmen ist, sind die Konturen der drei Personen im unteren Bereich des Bildes bei dem Ansatz von Li akkurater ausgebildet

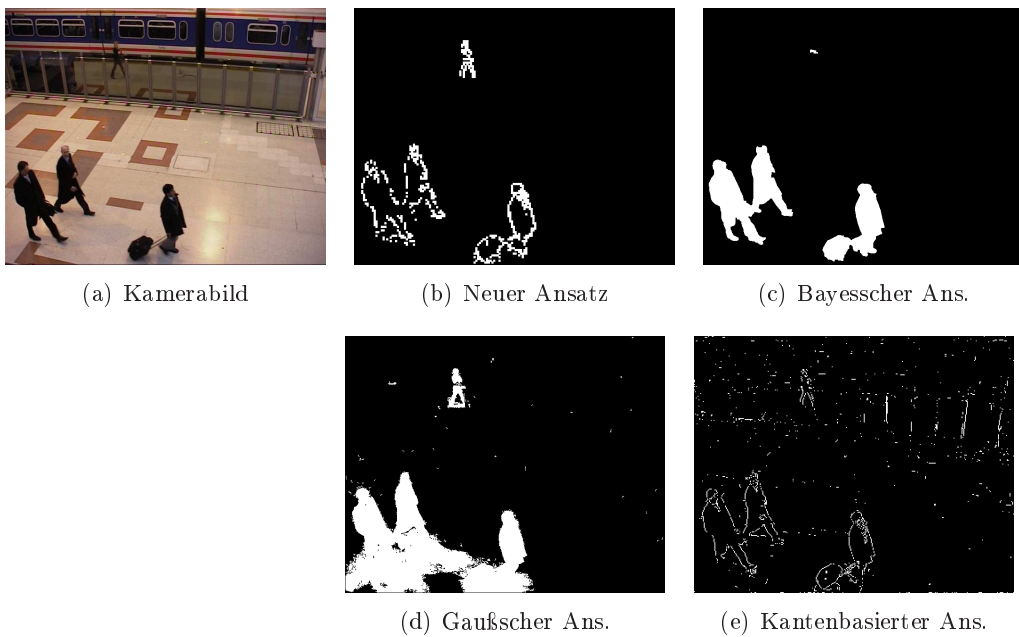


Abbildung 2.23: Performanz der Verfahren beim kaum vorhandenen Kontrast zwischen den Personen und dem Hintergrund der Szene im oberen Bereich des Bildes: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.



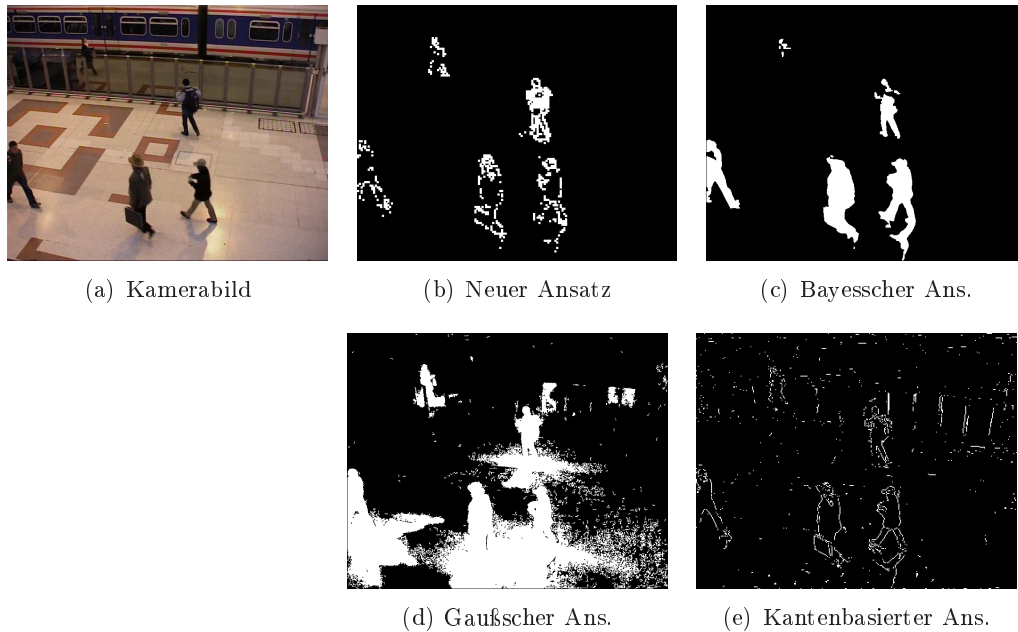


Abbildung 2.24: Performanz der Verfahren bei der Selektion von Personenschatten: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren.

als bei den restlichen drei. Ein sehr schwerwiegender Nachteil dieses Verfahrens ist allerdings sein Versagen die Person im oberen Bereich des Bildes zu detektieren. Während bei dem hier vorgestellten Verfahren im Bild (b) die Person deutlich zu erkennen ist, findet man im Bild (c) dagegen nur einen kleinen weißen Fleck. Dieser könnte schon fast als Rauschen angesehen werden.

Das erweiterte Gaußsche Verfahren von KeawTraKulPong liefert in beiden Abbildungen mangelnde Resultate. Das Konstrukt der Schattendetektion versagte hier während der gesamten Sequenz. Besonders schlechtes Ergebnis ist in Abbildung 2.24 (d) zu sehen. Fast das gesamte Bild ist hier weiß gefärbt. Es ist allerdings auch anzumerken, dass es sich hierbei um eine relativ schwierige Szene handelt. Auch das sonst, gegen Schatten sehr robuste Verfahren von Li schafft es bei dieser Aufnahme nicht den Schatten der rechten Person im Bild auszuwählen. Aufgrund des starken Kontrastes zwischen dem Hintergrund und dem Schatten wird die dabei erzeugte Kante auch von dem hier vorgestellten Verfahren als zum Vordergrund gehörend markiert.

Das kantenbasierte Verfahren von Chaohui versagt bei dieser Sequenz vollkommen. Wie den beiden Abbildungen zu entnehmen ist, können die Personen aufgrund des umliegenden Rauschens in keinem der beiden Bilder als solche erkannt werden.

Ähnliche Ergebnisse wurden auch während der gesamten Sequenz beobachtet.

#### 2.4.4 Laufzeit des Verfahrens

Die Laufzeit dieses Verfahrens hängt entscheidend von der konkreten Implementierung und der zugrunde liegenden Hardware ab. Im Rahmen des am Lehrstuhl 'Intelligente Systeme' durchgeführten Praxisprojektes wurde die anfängliche Implementierung dieses Verfahrens an die vorhandene Hardware so optimiert, dass folgende Laufzeiten erreicht werden konnten. Tabelle 2.1 fasst diese Ergebnisse zusammen.

Bildformat	Multi-threading	Intel Core *	AMD Phenom **
Grauwert (1 Kanal)	nein	2219 <i>ms</i> (454 <i>fps</i> )	6125 <i>ms</i> (164 <i>fps</i> )
Grauwert (1 Kanal)	ja	1078 <i>ms</i> (1000 <i>fps</i> )	2500 <i>ms</i> (400 <i>fps</i> )
Farbe (3 Kanäle)	nein	7921 <i>ms</i> (126 <i>fps</i> )	20406 <i>ms</i> (50 <i>fps</i> )
Farbe (3 Kanäle)	ja	5328 <i>ms</i> (188 <i>fps</i> )	9500 <i>ms</i> (105 <i>fps</i> )

Intel Core *	Intel Core 2 Quad-Core CPU Q9400 2.66GHz, RAM 3GB
AMD Phenom **	AMD Phenom 9600 Quad Core CPU 2.3GHz, RAM 2GB

Tabelle 2.1: Laufzeit Performanz des hier entwickelten Vordergrund-Hintergrund Separationsverfahrens in Abhängigkeit von der verwendeten Hardware. Als Evaluierungsgrundlage wurde eine Sequenz mit 1000 Bildern mit  $640 \times 480$  - Auflösung verwendet.

Als Evaluierungsgrundlage wurde hierbei eine Sequenz mit 1000 Bildern herangezogen. Die Auflösung der einzelnen Bilder betrug dabei  $640 \times 480$  Pixel.

Die folgenden Punkte fassen kurz die Eigenschaften der dabei verwendeten Implementierung zusammen:

- Die grobe Struktur des Algorithmuses wurde in C++ implementiert.
- Berechnungsschritte wie diskrete Kosinustransformation oder Berechnung der Kosinusfunktionen wurden in *x86* inline assembler implementiert. Dabei wurden SSE2 (Streaming SIMD Extensions 2) und SSSE3 (Super Streaming SIMD Extension 3) Erweiterungen der *x86*-Architektur verwendet.
- Die Koeffizienten der diskreten Kosinustransformation wurden im Speicher in Form von 16-Byte Datenelementen gespeichert. Mittels des SSE2 und SSSE3 konnte damit eine byteweise Adressierung des Speichers erfolgen und auf die 16-Bytes in einen Takt zugegriffen werden.
- Für Multithreading wurde die WINAPI genutzt.

- Die Implementierung einzelner Strukturen und Methoden erfolgte auch unter Zuhilfenahme der OpenCV Bibliothek.

Aufgrund des Zugriffs auf die SSE2 und SSSE3 Erweiterungen läuft diese Implementierung nur auf Intel Prozessoren. Die hierbei erzielte real-time Performanz des Verfahrens lässt sich allerdings auch auf anderen Prozessoren mit einer entsprechenden Implementierung erreichen.

## 2.5 Diskussion und Zusammenfassung

Das im Rahmen dieser Arbeit entwickelte Verfahren erzielte bei allen hier durchgeführten Experimenten bessere oder vergleichbare Ergebnisse. Im Video der Parkplatzüberwachung wurde die Adaptivität des hier vorgestellten Verfahrens gegenüber Änderungen im Hintergrund der Szene gezeigt. So wurde zum einen ein vom Stellplatz weggehendes Fahrzeug vom Verfahren als bewegendes Objekt detektiert, zum anderen wurde zeitgleich der nun sichtbare Asphalt robust in das Hintergrundmodell eingebaut. Ergebnisse dieses Experiments haben gezeigt, dass für den erfolgreichen Einsatz dieses Verfahrens eine vorherige offline stattfindende Trainingsphase nicht notwendig ist. Das Verfahren ist in der Lage das Hintergrundmodell der abgebildeten Szene online zu erstellen. Die zum Training verwendeten Aufnahmen können auch bewegende Objekte enthalten.

In dem Testszenario mit plötzlich ändernden Lichtverhältnissen war das hier vorgestellte Verfahren gegenüber den anderen dreien weit überlegen. Beim Ausschalten der Lampen in dem abgebildeten Aufenthaltsraum wurde zwar bei dem neuen Ansatz ebenfalls Rauschen detektiert, allerdings war der Umfang dieses Rauschens im Vergleich zu den Ergebnissen der anderen Verfahren deutlich geringer. Während bei den Verfahren von Li und KeawTraKulPong fast das komplette Bild in den Vordergrundbereich konvertiert wurde, waren es bei dem hier vorgestellten Verfahren nur einige über das Bild verteilte Punkte.

Bei der anschließenden Detektion von Personen im abgedunkelten Raum übertrug das hier neu vorgestellte Verfahren ebenfalls. Alle Personen wurden robust während der gesamten Sequenz detektiert. Im Gegenteil dazu wurde bei dem Verfahren von Li wegen des fehlenden Kontrastes nur einige wenige Bereiche der Person detektiert oder diese gar gänzlich falsch klassifiziert. Das erweiterte Gaußsche Verfahren von KeawTraKulPong detektierte zwar die Person, allerdings wurde während der gesamten Sequenz ein gewisser Rauschanteil beobachtet. Außerdem reagierte das Verfahren sehr sensitiv auf die von Personen geworfene Schatten.

Beim Wiedereinschalten der Lampen wurden bei dem hier neu vorgestellten Verfahren keine Änderungen in dem korrespondierenden Vordergrundbild erfasst. An-

ders war die Situation bei den Verfahren von Li und KeawTraKulPong. Hier wurden wieder große Bereiche des Vordergrundbildes weiß gefärbt. An diesem Punkt kommt die Überlegenheit dieses Verfahren gegenüber den anderen besonders deutlich hervor. Die Stabilität dieses Verfahrens gegenüber plötzlichen Lichtänderungen ist darauf zurückzuführen, dass der Schwerpunkt hierbei in der Detektion der Änderungen in den Grauwertstrukturen der Bilder liegt. Anders als bei den Verfahren von Li und KeawTraKulPong wird hier nicht auf Farbunterschiede geachtet. Das kantenbasierte Verfahren von Chaohui zeigte sich ebenfalls sehr robust gegenüber Lichtänderungen. Das Ein- und Ausschalten der Lampen hatte keine Auswirkungen auf die Menge des permanent vorhandenen Rauschens in dem korrespondierenden Vordergrundbild.

Bei dem Experiment der sich langsam ändernden Lichtverhältnisse waren die Ergebnisse des hier vorgestellten Verfahrens mit denen von Li vergleichbar. Beide Verfahren zeigten sich sehr stabil gegenüber sich ändernden Bedingungen. Bei der anschließenden Detektion der Person schwächelte allerdings das Verfahren von Li wieder wegen des mangelnden Kontrastes bei der Erfassung der Beine. Im Gegensatz dazu konnte von dem hier vorgestellten Verfahren die Person komplett erfasst werden. Es ist jedoch anzumerken, dass bezüglich der extrahierten Konturen der detektierten Person des Verfahren von Li den anderen deutlich überlegen ist. Das Verfahren von KeawTraKulPong versagte bei dieser Sequenz vollkommen. Die sich bewegende Person wurde aufgrund des überladenen Hintergrundmodells nicht detektiert. Das Verfahren schaffte auch nicht sich rechtzeitig an die sich ändernden Bedingungen anzupassen. Während des größeren Teils der Sequenz wurde ein permanentes Rauschen in dem Vordergrundbild beobachtet. Ähnlich wie bei den anderen Experimenten hatte das kantenbasierte Verfahren von Chaohui auch hier sein charakteristisches Rauschen in dem Vordergrundbild.

Bei der Modellierung des dynamischen Hintergrunds war das hier vorgestellte Verfahren den anderen ebenfalls überlegen. Zwar hat man auch hier abhängig von der Stärke der Baumschwingung hin und wieder mal Rauschen erfasst, doch konnte dieses im Gegenteil zu anderen Verfahren zwischenzeitlich komplett eliminiert werden. Der bei diesem Ansatz modulare Aufbau des Modells bringt wegen seiner dynamischen Anpassung an die Gegebenheiten der Szene weitere Vorteile mit sich.

Zusätzlich wurden die Ergebnisse der vier Verfahren auf den Testsequenzen der PETS Datenbank evaluiert. Dabei wurde festgestellt, dass das hier vorgestellte Verfahren viel robuster und sauberer die bewegenden Personen im Video detektiert. Während das Verfahren von Li einige Personen bei der Detektion auslöst, hat das Verfahren von KeawTraKulPong immense Probleme mit den von Personen geworfenen Schatten. Abhängig von den Aktivitäten in der Szene war zeitweise fast das gesamte Vordergrundbild weiß gefärbt.

Zusammenfassend lässt sich sagen, dass das im Rahmen dieser Arbeit entwickelte Verfahren bei der Detektion von Objekten in fast allen hier getesteten extremen Situationen den anderen Verfahren überlegen war. Es ist sehr robust gegenüber plötzlichen, sowie auch sukzessiven Änderungen in der Szenenbeleuchtung. Es ist adaptiv und passt sich an die Gegebenheiten der Szene an. Es kann sehr gut mit einem dynamischen Hintergrund umgehen. Leichtes Flickern eines Monitors oder periodische Bewegungen einer schwankenden Pflanze bereiten hier keine Probleme.

Der einzige bei diesem Verfahren beobachtete Nachteil ist seine nicht akkurate Extraktion der Objektumrandungen. Eine geschickte Kombination der Ergebnisse dieses Verfahrens mit denen des Verfahrens von Li, würde allerdings auch dieses Problem zum großen Teil beheben. In den durchgeführten Experimenten wurde festgestellt, dass das im Rahmen dieser Arbeit entwickelte Verfahren sehr stabil Bereiche erfasst in denen relevante Änderungen stattgefunden haben. Bei einer Überlappung dieser Bereiche mit den Ergebnissen des Verfahrens von Li, könnte man auf dessen Ergebnisse zurückgreifen und somit zumindestens in einigen Bereichen des Bildes eine akkurate Objektkontur erzielen.

Das hier vorgestellte Verfahren wurde im Jahr 2009 auf der sechsten 'IEEE International Conference on Advanced Video and Signal Based Surveillance' zum ersten mal vorgestellt. Weitere Details der Veröffentlichung können in [SP09b] entnommen werden.

Das folgende Kapitel beschreibt wie dieses Verfahren im Rahmen der Objektverfolgung erfolgreich eingesetzt werden kann.



# Robuste Objektverfolgung in der Bildsequenz

---

## Kapitelinhalt

---

<b>3.1</b>	<b>Stand der Technik im Bereich der Objektverfolgung . . . .</b>	<b>46</b>
<b>3.2</b>	<b>Bewegungsbasierte Objekterkennung im Bild . . . . .</b>	<b>50</b>
<b>3.3</b>	<b>Online Bildung eines Objektmodells . . . . .</b>	<b>58</b>
<b>3.4</b>	<b>Ergebnisse der Objektverfolgung . . . . .</b>	<b>61</b>
<b>3.5</b>	<b>Diskussion und Zusammenfassung . . . . .</b>	<b>69</b>

---

Verfolgung von Objekten und damit auch die Kenntnis über die Objekttrajektorien in einer Bildsequenz bilden in vielen Anwendungen eine wichtige Informationsquelle. Bezugnehmend auf das in der Einleitung vorgestellte Szenario ist die Kenntnis über die vom Roboter aktuell eingeschlagene Fahrtrichtung unentbehrlich für die zeitnahe Korrektur seiner Bewegungskurve. Andere Bereiche, in denen eine robuste Objektverfolgung von entscheidender Bedeutung ist, sind beispielsweise Verkehrsüberwachung oder Mensch-Maschine Interaktion.

Das vorliegende Kapitel stellt ein im Rahmen dieser Arbeit entwickeltes Verfahren dar, welches im Bereich modellbasierter Objektverfolgung anzusiedeln wäre. Der entscheidende Vorteil dieses Verfahrens gegenüber den bereits etablierten Ansätzen ist dessen Online-Einsatzbarkeit. Eine für diese Klasse der Verfahren üblicherweise benötigte Trainingsphase findet hier im Online-Betrieb statt.

Das Kapitel beginnt mit einer Einführung in die Problemstellung. Gleichzeitig werden hier auch bereits bestehende Ansätze vorgestellt und Vor- und Nachteile dieser kurz erläutert. Anschließend wird ein Verfahren zur bewegungsbasierten Objekterkennung vorgestellt. Dieses dient der anfänglichen Positionsbestimmung des Objektes im Bild und bildet damit einen grundlegenden Baustein für die Online-Bildung des Objektmodells. Das Verfahren zur Generierung des Modells wird im darauf folgenden Abschnitt präsentiert. Das Kapitel schließt mit einer Evaluierung dieses Ansatzes ab.

### 3.1 Stand der Technik im Bereich der Objektverfolgung

Im Unterschied zur Objektdetektion, beinhaltet Objektverfolgung noch zusätzlich die Aufgabe einer wahrheitsgetreuen Zuordnung von denselben Objekten in zwei aufeinander folgenden Bildern einer Kameraaufnahme. Im Falle einer erfolgreichen Zuordnung können die Trajektorien der entsprechenden Objekte berechnet und zur weiteren Verarbeitung an nachfolgende Applikationsmodule gereicht werden.

Ähnlich wie die Objektdetektion bildet auch eine robuste Objektverfolgung in vielen Anwendungen einen fundamentalen Baustein. Im Falle einer automatischen Auswertung der Videoinformation ist die Kenntnis über die Bewegungskurven der interessierenden Objekte oft unentbehrlich. Bei den Kameraaufnahmen zur Verkehrsüberwachung können damit beispielsweise von der Norm abweichende Situationen extrahiert werden, ohne dass eine menschliche Intervention erforderlich wäre. Auch bei der Überwachung von öffentlichen Parkplätzen kann ausgehend von den berechneten Trajektorien der sich bewegenden Personen auf deren eventuell kriminelle Absicht geschlossen werden. So wurde oft beobachtet, dass Menschen, die vorhaben ein fremdes Auto zu entwenden, sich vorher noch mehreren anderen Fahrzeugen nähern und diese dann wieder verlassen. Eine richtige Auswertung der Trajektorien könnte hier einer Prävention krimineller Handlungen dienen.

Im Bereich der Mensch-Maschine Interaktion funktioniert die Kommunikation oft über die entsprechenden Handbewegungen der Personen. Die Deutung einer Handbewegung setzt allerdings auch hier voraus, dass die Hand als solche von Bild zu Bild verfolgt werden kann. Die Genauigkeit der berechneten Trajektorien ist hier oft der Schlüssel zu einer barrierefreien Informationsvermittlung.

Auch im Rahmen der sicheren Roboternavigation ist eine robuste Verfolgung des Roboters im Kamerabild essentiell. Zur Steuerung des Roboters zu einem vorher spezifizierten Zielpunkt muss dessen Position zu jedem Zeitpunkt der Bildaufnahme bekannt sein. Dies ist notwendig um eventuelle Abweichungen vom Kurs rechtzeitig korrigieren zu können.

Im Folgenden werden zunächst kurz die bei der Entwicklung der Verfahren auftretende Probleme diskutiert und anschließend einige bereits existierende Ansätze vorgestellt. Gleichzeitig werden dabei die Nachteile dieser Verfahren herausgearbeitet und darauf basierend der im Rahmen dieser Arbeit entwickelte Ansatz motiviert.

#### **Auftretende Probleme**

Aufgrund einer hohen Nachfrage nach stabilen und sicheren Algorithmen zur robusten Objektverfolgung in einer Videosequenz existiert bereits eine breite Palette von unterschiedlichen Ansätzen und Methoden zur Bewerkstelligung dieser Aufga-



be. Die dabei zu nehmenden Hürden lassen sich in den folgenden Punkten kurz zusammenfassen.

- Aufgrund der Beleuchtungsänderungen können sich die Farben und die Schattierungen eines Objektes im Bild sehr stark ändern.
- Abhängig von der verwendeten Kamera kann das dabei resultierenden Rauschen im Bild eine Objekt Verfolgung sehr erschweren.
- Ein Objekt kann teilweise oder auch komplett durch andere Objekte im Bild verdeckt werden.
- Aufgrund der komplizierten Objektstruktur, ist eine Konstruktion des entsprechenden Objektmodells oft nur bedingt möglich.
- Aufgrund der perspektivischen Abbildung auf das Kamerabild kommt es zum Informationsverlust.

### Grundlegende Vorgehensweise

Die meisten Ansätze zur Objektverfolgung sind auf einige wenige Anwendungsgebiete spezialisiert. Bei der groben Betrachtung dieser kristallisiert sich allerdings eine allgemein gültige Vorgehensweise.

Abhängig von dem Einsatzgebiet und der zu verfolgende Objekte wird zunächst ein brauchbares Objektmodell erstellt. Dabei werden oft Annahmen über die Anzahl, Größe, Aussehen und die Form der entsprechenden Objekte getroffen. Modelle bezüglich der Objektbewegung sind ebenfalls weit verbreitet. So gehen zum Beispiel viele Ansätze davon aus, dass die zu verfolgende Objekte eine gleichmäßige Bewegung, ohne abrupte Richtungsänderungen durchführen.

Im nächsten Schritt findet eine Objektdetektion statt. Dazu können auch, wie bereits im letzten Kapitel besprochen, Vordergrund-Hintergrund Separationsverfahren zum Einsatz kommen. Ein andere Möglichkeit ist ein modellspezifischer Abgleich (eng: matching) zwischen der aktuellen Bildinformation und dem vorhandenen Objekt Modell.

Die eigentliche Objektverfolgung findet dann durch ein Vergleich der relevanten Bildinformationen in den zwei aufeinander folgenden Bildern statt. Dabei können die beiden Aufgaben der Objektdetektion und der Korrespondenzbestimmung der in den beiden Bildern detektierten Objekte entweder getrennt oder zusammenwirkend ablaufen. Im Falle einer Punktrepräsentation ist der SIFT-Tracker (Scale Invariant Feature Transform, [Low04]) ein bekanntes Beispiel für die erste Möglichkeit. Hier werden zunächst in den beiden Bildern markante Merkmale extrahiert und

Eigenschaften dieser in einem so genannten ‘Feature’-Vektor gespeichert. Die Zuordnung zweier Merkmale zueinander erfolgt dann im nächsten Schritt mittels eines Vergleichs der jeweiligen ‘Feature’-Vektoren. Besteht ein Objektmodell aus einer Kombination solcher Merkmale, so kann auf diese Weise die Trajektorie des entsprechenden Objektes berechnet werden.

Im Falle der zusammenwirkenden Detektion und Korrespondenzbestimmung wird die Objektposition iterativ basierend auf der im vorigen Bild gesammelten Information berechnet. Dabei werden nach der anfänglichen Detektion des Objektes einige diese Region beschreibende Merkmale ausgewählt.

### Existierende Ansätze zur Objektverfolgung

Yilmaz et al. [YJS06] gibt einen guten Überblick über die sich bereits etablierten Methoden und einige neue Ansätze. Als Beispiel für eine zusammenwirkende Detektion und Korrespondenzbestimmung sei hier die Arbeit von Birchfield [Bir98] erwähnt. Weil die Farbhistogramme sehr instabil gegenüber Beleuchtungsänderungen sind, verwendet er ein Gradientenhistogramm. Die Zuordnung des entsprechenden Bereiches im nachfolgenden Bild erfolgt mittels einer Ähnlichkeitsfunktion. Viele Verfahren verfolgen dabei eine ausgeklügelte Strategie zur Einschränkung des Suchbereiches. So verwenden Comaniciu und Meer eine so genannte ‘mean-shift’ Strategie. Eine detaillierte Beschreibung dieses Vorgehens findet sich in ihrer Arbeit [CRM03]. Andere Verfahren machen sich die Methode des Optischen Flusses zu Nutze. Der Kanade-Lucas-Tomasi (KLT) Tracker [ST94] verwendet eine dem Optischen Fluss ähnliche Berechnungsvorschrift zur Schätzung der Merkmalsposition im nachfolgenden Bild.

Die Vorteile dieser Verfahren liegen in der online Berechnung der das Objekt beschreibenden Merkmale. Farb- oder Gradientenhistogramme, Silhouette und Objektkonturen werden nach der ersten Objektdetektion in jedem weiteren Bild neu berechnet. Die Verfolgung basiert damit auf der zuletzt erworbenen Beschreibung des Objektes. Dies setzt allerdings voraus, dass das Erscheinungsbild des Objektes sich von Bild zu Bild kaum ändert. Unabhängig von den äußeren Einflüssen wie Beleuchtung der Szene oder der Kameraperspektive ist eine gleichmäßige Objektbewegung hierzu eine wichtige Voraussetzung. Sollte sich die Richtung des Objektes abrupt ändern, so wird üblicherweise dessen Erscheinung im Bild sich ebenfalls sehr verändern. Die im vorherigen Bild akquirierte Information wird unbrauchbar und das Verfahren verliert das Objekt.

Eine andere Klasse von Methoden versucht dem entgegen zu wirken, indem das das Objekt beschreibende Modell mehrere unterschiedliche Ansichten von diesem

beinhaltet. Ein solches Modell macht das Verfahren zur Objektverfolgung unabhängig von der Objektbewegung. Auch beim abrupten Richtungswechsel und schnellen Änderungen der Objektansicht kann dieses üblicherweise auch weiterhin robust verfolgt werden. Entscheidend hier ist der Aufbau des Modells. Die Anzahl der unterschiedlichen Objektansichten und die Art und Weise wie diese gespeichert sind, hat einen starken Einfluss auf die Performanz des jeweiligen Verfahrens. Sehr wichtig ist hier auch eine robuste Zuordnung der aktuellen Bildinformation zu den in der Datenbank gespeicherten Ansichten.

Die meisten aktuell vorhandene Verfahren dieser Klasse konstruieren solche ‘multiview appearance’-Modelle während einer offline stattfindende Trainingsphase. Im Jahr 2001, schlug Avidan die Verwendung von Support Vektor Maschinen (SVM) zur Klassifizierung der Bildinformation in solche, die das Objekt beinhaltet, und solche die den Hintergrund der Szene darstellen. Zur Anpassung der entsprechenden Parameter verwendete er für die jeweilige Anwendung angepassten Trainingsdaten. Die positiven Beispiele dieser Datenmenge beinhalteten mehrere unterschiedliche Ansichten des interessierenden Objektes. Die Ergebnisse und weitere Details dieser Methode können seiner Arbeit [Avi04] entnommen werden.

Einen anderen Ansatz verfolgten Black und Jepson. Sie benutzten einen Eigenvektor-Unterraum zur Modellierung der unterschiedlichen Objektansichten. Zur Konstruktion dieses Unterraums verwendeten sie die Principal-Component-Analysis (PCA). Die Evaluierung der Performanz dieses Ansatzes findet sich ebenfalls in ihrer Arbeit [BJ96].

### Motivation und Grundidee des neuen Ansatzes

Obwohl die meisten Ansätze der modellbasierten Objektverfolgung stabile Ergebnisse liefert, ist die bereits erwähnte notwendige Trainingsphase oft ein nicht zu unterschätzendes Problem. In vielen Anwendungen können die dazu erforderlichen Trainingsdaten nicht zur Verfügung gestellt werden. Sei es aus Kostengründen oder weil von vornherein nicht klar ist welche Objekte verfolgt werden müssen. Falls allerdings so ein Modell erzeugt werden kann, so ist dieses nicht selten überladen. Oft enthält es Ansichten, die bei der anschließenden Anwendung aufgrund der gewählten Kameraperspektive nicht verwendet werden.

Wünschenswert wäre daher ein Verfahren, welches zum einen ein solches ‘multiview appearance’ Modell online erstellt und zum anderen auch automatisch feststellt von welchen Objekten dieses generiert werden soll. Was zunächst utopisch klingen mag, wird von uns Menschen tagtäglich bewerkstelligt. Bei einer Verabredung mit einer unbekannt Person auf einem belebten Platz, wird diese als erstes von uns

anhand vorher definierter Merkmale erkannt. Im Laufe der darauf folgenden Interaktion erstellen wir, oft unbewusst, eine Liste personenspezifischer Merkmale, die wir dann später für die Wiedererkennung dieser Person heranziehen. Dabei beziehen sich, die für die anfängliche Erkennung verwendete Merkmale selten auf das Äußere der unbekannt Person. Optische Charakteristika wie Geschlecht, Größe oder Farbe der Kleidung würden auf einem offenen und sehr belebten Platz mit hoher Wahrscheinlichkeit auf mehrere Personen gleichzeitig zutreffen. Viel spezifischer können dagegen Tätigkeiten formuliert werden, die die gesuchte Person während der Erkennungsphase durchführen soll. Winken mit der Hand oder möglichst hoch auf einer Stelle hüpfen sind nur einige Beispiele, die sich hervorragend dafür eignen würden.

Im Rahmen dieser Arbeit wurde eben dieses Prinzip für die anfängliche Objekterkennung im Kamerabild und die darauf folgende Erstellung des entsprechenden Objektmodells adaptiert. Der nächste Abschnitt beschreibt wie ein Objekt anhand seiner periodischen Bewegung im Bild von anderen bewegenden Objekten ausselektiert werden kann. Die so gewonnene erste Ansicht des Objektes bildet dann den Startpunkt zur Generierung eines ‘multiview appearance‘ Modells. Das dazu entwickelte Verfahren wird in diesem Kapitel in einem der nächsten Abschnitte beschrieben.

## 3.2 Bewegungsbasierte Objekterkennung im Bild

Bewegungsbasierte Erkennung von Objekten bildet eine gleichwertige Alternative zu der Objekterkennung mittels optischer Merkmale. In vielen Situationen ist dies sogar die einzige Möglichkeit robust relevante Objekte zu identifizieren. Ein modellbasiertes Verfahren würde bei der Erkennung eines Objektes versagen, wenn dessen spezifische äußere Merkmale aufgrund fremder Einflüsse zwischenzeitlich verändert worden wären. So können beispielsweise die markanten Merkmale einer Maschine auf einer Baustelle aufgrund schlechter Wetterbedingungen durch Dreck bedeckt worden sein. Die Erkennung würde auch versagen, wenn infolge eines Unfalls ein wichtiges Glied der Maschine plötzlich fehlt würde.

Ein weiteres interessantes Szenario ist das eines Postboten Roboters. Ein Roboter, der ein Paket in einem Bürogebäude bei einer bestimmten Person abliefern soll, kennt zwar die Bezeichnung des Aufenthaltsbereiches dieser Person, allerdings nicht den Weg dorthin. Für den erfolgreichen Abschluss des Auftrags, müsste ein in diesem Gebäude installiertes Navigationssystem den Roboter dorthin geleiten. Dies setzt allerdings voraus, dass das System den im Eingangsbereich befindenden Roboter als solchen erkennt. Ein modellbasiertes Erkennungsverfahren würde hier definitiv versagen, falls in der zuvor stattfindenden Trainingsphase keine Abbildun-

gen des entsprechenden Roboters in den Trainingsdaten vorhanden waren. Dieses Szenario bildet nur ein Beispiel für viele Anwendungen in denen nicht von vornherein klar ist, welche Objekte verfolgt werden müssen. Wie bereits erwähnt ist in solchen Situation eine bewegungsbasierte Objekterkennung oft die einzige Alternative.

Im Rahmen dieser Arbeit wurde ein Verfahren zur automatischen Erkennung von periodischen Bewegungen in einer Bildsequenz entwickelt. Die Einschränkung auf die periodische Bewegungen erfolgte dabei aus zwei Gründen. Erstens ist es die Einfachheit der Bewegung. Fast alle beweglichen Objekte sind in der Lage diese durchzuführen. Bei einem mobilen Roboter würde eine regelmäßige Kreisbewegung oder ein primitives Vor- und Rückwärtsfahren durchaus ausreichen. Eine Person könnte beispielsweise eine Auf- und Abbewegung der Hand durchführen, um auf sich aufmerksam zu machen. Den zweiten Grund bilden die sehr charakteristischen Eigenschaften dieser Bewegung. In einem chaotischen Treiben lässt sich diese Art der Bewegung aufgrund ihres geordneten und schon fast vorhersagbaren Verlaufs einfacher und stabiler erkennen, als dies bei anderen komplizierteren Bewegungskonformationen der Fall wäre.

Der Ablauf der hier entwickelten Methode lässt sich in den folgenden Punkten grob zusammenfassen:

- Detektion aller bewegenden Objekte mittels eines Vordergrund-Hintergrund Separationsverfahrens.
- Verfolgung dieser Objekte über einen gewissen Zeitraum mittels einer einfachen Methode zur Objektverfolgung.
- Selektion einer periodischen Bewegung aus allen berechneten Trajektorien.

Die hierbei zur Anwendung kommende Methode zur Objektverfolgung ist sehr einfach und ist daher nicht für anspruchsvolle Anwendungen geeignet. Für die hier benötigte Zwecke ist sie allerdings durchaus ausreichend. Die folgenden Abschnitte geben eine detaillierte Beschreibung der drei genannten Punkte.

### 3.2.1 Einfache Methode zur Objektverfolgung

Zur anfänglichen Detektion der bewegenden Objekte in der Bildsequenz wurde das im vorigen Kapitel vorgestellte Verfahren zur Vordergrund-Hintergrund Separation verwendet. Dabei wurde auf das resultierende Vordergrundbild noch zusätzlich ein Algorithmus zur Berechnung von zusammengesetzten Komponenten angewandt. In der Literatur findet man solche Algorithmen unter dem Fachbegriff ‘Connected Components Labeling’. Im Rahmen dieser Arbeit wurde ein Verfahren angewandt, welches die folgenden Schritte beinhaltet:

1. Berechnung von Objektkonturen mittels des Freeman Ketten-Kodes [Fre74].
2. Berechnung von achsenparallelen Rechtecken um die extrahierten Konturen.
3. Fusion zweier Rechtecke, falls der euklidische Abstand zwischen denen kleiner ist als der vorher festgelegt Schwellenwert *minDistance*.
4. Eliminierung aller Rechtecke, deren Fläche kleiner ist, als der Schwellenwert *minRectArea*.

Abbildung 3.1 zeigt exemplarisch die damit erzielten Ergebnisse. Das rechte Bild

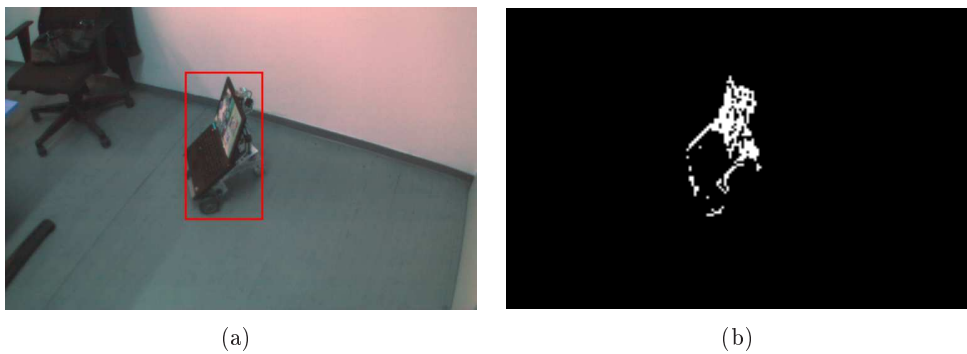


Abbildung 3.1: Ergebnis der Detektion des bewegenden Objekts: (a) der rote Rechteck wurde mittels des Moduls 'Komponenten Extraktion' berechnet; (b) Vordergrundbild des im Kapitel 2 vorgestellten Vordergrund-Hintergrund Separationsverfahrens.

zeigt hier das dabei resultierende Vordergrundbild. Die extrahierten Objekte sind durch örtlich verteilte Anhäufungen von weißen Rechtecken zu erkennen. Das linke Bild zeigt das dazu korrespondierende Kamerabild. Die roten Rechtecke um die bewegenden Objekte wurden mittels des oben beschriebenen Algorithmuses berechnet. Im Folgenden wird dieses Modul als 'Komponenten Extraktion' bezeichnet.

Die Bestimmung der Korrespondenzen zwischen den detektierten Objekten in zwei aufeinander folgenden Bildern erfolgte unter Zuhilfenahme der bereits erwähnten SIFT Merkmale. Der SIFT-Algorithmus zur Extraktion lokaler Merkmale im Bild wurde im Jahr 1999 von David. D. Lowe [Low04] formuliert. Die oft zitierten Vorteile der extrahierten Merkmale sind deren Unempfindlichkeit gegenüber Koordinatentransformationen wie Translation, Rotation und Skalierung. Außerdem sind die Merkmale robust gegen Beleuchtungsvariationen und Bildrauschen.

Jedem extrahierten Merkmal wird dabei ein Beschreibungsvektor zugeordnet deren Komponenten vor der lokalen Nachbarschaft des Merkmals abhängen. Die Zuordnung der korrespondierenden Merkmale in zwei aufeinander folgenden Bildern erfolgt durch den Vergleich der jeweiligen Beschreibungsvektoren.

Umfangreiche Studien über die Leistungsfähigkeit der SIFT-Merkmale belegen die bereits erwähnten Vorteile. Zusammen mit den GLOH-Merkmalen (Gradient Location and Orientation Histogram) besetzen die SIFT-Merkmale bei der Wiedererkennung von Objekten im Bild die höchsten Plätze [MS05]. Damit eignet sich der Algorithmus sehr gut zur Bestimmung von Punktkorrespondenzen in zwei aufeinander folgenden Bildern der Kamera.

Um den relativ hohen Aufwand zur Berechnung dieser Merkmale einzugrenzen, diente als Eingabe für den Algorithmus nicht das komplette Kamerabild, sondern nur die durch die roten Rechtecke umrandete Bereiche. Dabei wurde die Information über die Lage der entsprechenden Bereiche im Bild beibehalten, indem nur die nicht relevanten Bildausschnitte schwarz gefärbt wurden. Abbildung 3.2 (a) zeigt ein Beispiel für solch eine bearbeitete Eingabe. Das rechte Bild der Abbildung zeigt dagegen die Ergebnisse des darauf angewandten SIFT-Verfahrens. Die grünen Punkte kenn-

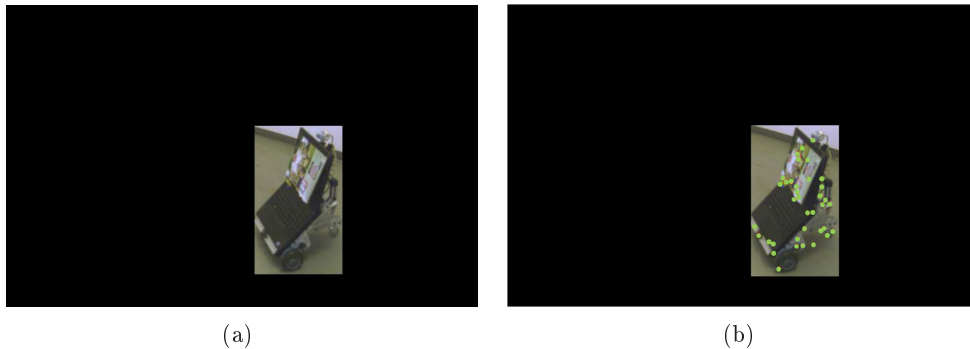


Abbildung 3.2: Eingabe und Ergebnis des SIFT-Verfahrens: (a) als Eingabe dienen ausschließlich Bildausschnitte, die die bewegenden Objekte der Szene repräsentieren; (b) die grünen Punkte kennzeichnen hier die Positionen der extrahierten SIFT-Merkmale.

zeichnen hierbei die Positionen der auf diesem Bild berechneten SIFT-Merkmale. Weil in den schwarzen Bereichen des Bildes keine Struktur vorhanden ist, werden diese Bildausschnitte von dem Verfahren ignoriert. Damit wurde sicher gestellt, dass ein Großteil der SIFT-Merkmale auf den bewegenden Objekten extrahiert wurde.

Korrespondenzen zwischen den Rechtecken werden nun auf der Basis der korrespondierenden SIFT-Merkmale berechnet. Rechtecke aus jeweils zwei aufeinander folgenden Bildern werden genau dann einander zugeordnet, wenn die SIFT-Merkmale, die sie umschließen zueinander korrespondierend sind.

Auf diese Weise werden die unterschiedlichen Objekttrajektorien berechnet. Es ist allerdings anzumerken, dass für komplexe Anwendungen dieses einfache Verfahren zur Objektverfolgung nicht ausreicht. So können oft bei einer schnellen Rota-

tionsbewegung eines Objektes keine SIFT-Korrespondenzen gefunden werden, weil das Objekt in diesem Fall viel zu schnell sein Abbild im Kamerabild ändert.

Für die Detektion einer einfachen periodischen Bewegung ist es allerdings gut geeignet. Ein Grund dafür ist das Nichtvorhandensein eines zeitlichen Limits. Sollte das Verfahren bei der Verfolgung eines Objektes versagen, so kann es sofort schon bei der nachfolgenden Detektion neu anfangen. Es wird damit vorausgesetzt, dass das interessierende Objekt so lange die entsprechende Bewegung durchführt bis das System genug Information gesammelt hat. Damit ist auch der Fall abgedeckt, dass sich zwischenzeitlich mehrere Objekte periodisch bewegen können. In diesem Fall verfolgt das System so lange die Objekte bis nur eines von diesen die charakteristische Bewegung durchführt.

### 3.2.2 Detektion einer periodischen Bewegung

Die Detektion einer periodischen Bewegung basierte auf der Evaluierung der Trajektorien der in der Szene bewegenden Objekte. Dazu wurden alle Trajektorien zunächst in eine, für den im Rahmen dieser Arbeit entwickelten Algorithmus brauchbare Form gebracht.

Dabei wurde zunächst die Position im Kamerabild, die das entsprechende Objekt bei der anfänglichen Detektion hatte, als die Startposition für dieses Objekt gewählt. Mit jedem nachfolgenden Bild wurde dann der euklidische Abstand zwischen der Start- und der neuen Position des Objektes berechnet. Führte das Objekt eine periodische Bewegung aus, so resultierte dies in der in Abbildung 3.3 nur beispielhaft gezeigten Kurve. Hierbei wurden die errechneten Abstände in Abhängigkeit von der jeweiligen Bildnummer aufgetragen. Die Nummerierung der Kamerabilder beginnt hier mit der erstmaligen Detektion des jeweiligen Objektes. Bei diesem Experiment wurde ein Roboter vor- und rückwärts bewegt. Ähnliche Kurven ergeben sich auch wenn das Objekt eine Kreisbewegung durchführt oder die Konturen eines Rechtecks kontinuierlich abfährt.

Die Glattheit der in Abbildung 3.3 gezeigten Kurve ist zum großen Teil auf das gewählte Vorgehen bei der Positionsbestimmung der Objekte zurückzuführen. Nach mehreren Experimenten hat sich herausgestellt, dass die Mitte des entsprechenden Rechtecks als die aktuelle Objektposition hier die bessere Wahl ist. Alternativ wurde die Position auch als Durchschnitt der in diesem Rechteck gefundenen SIFT-Merkmale berechnet. Die in Abbildung 3.4 gezeigte Graphik erklärt anschaulich den Grund warum dies keine gute Alternative war.

Auch bei einer zur Bildebene der Kamera parallelen Objektbewegung konnte nicht vorausgesetzt werden, dass bei allen Objektaufnahmen durchgehend immer



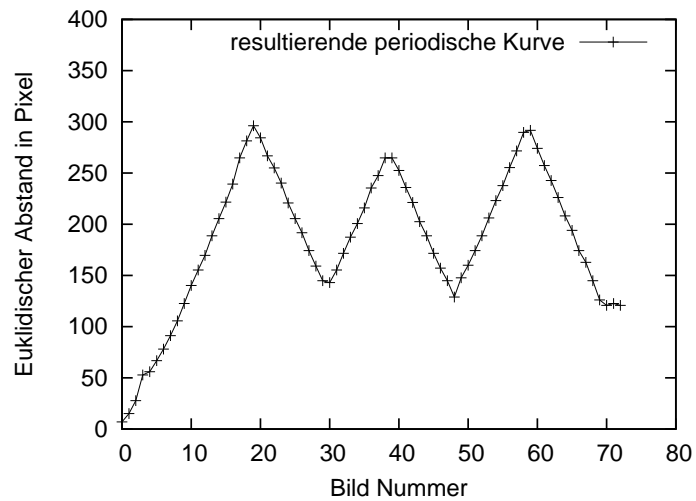


Abbildung 3.3: Beispiel für eine, während einer periodischen Bewegung resultierenden Kurve. Die ursprünglichen Daten der Trajektorie wurde hierfür in ein entsprechendes Format transformiert.

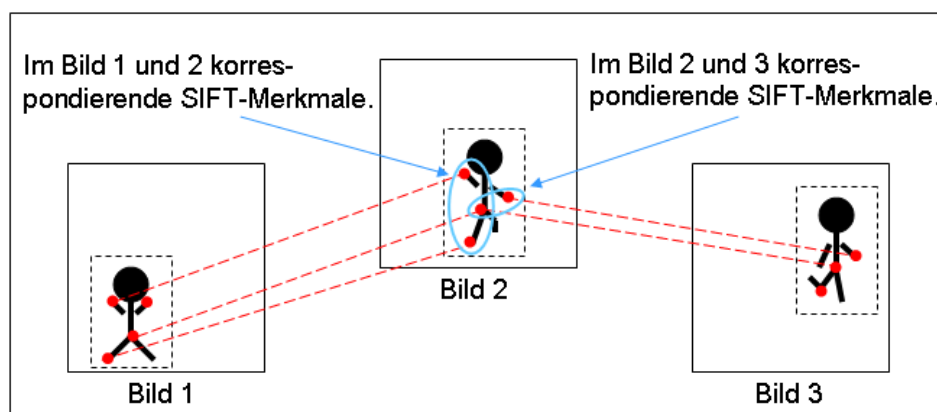


Abbildung 3.4: Bei den Experimenten oft beobachtete Situation: Die Anzahl und die Art, der auf einem Objekt extrahierten SIFT-Merkmale variieren von Bild zu Bild.

die gleichen SIFT-Merkmale extrahiert werden. Wie in der obigen Abbildung gezeigt wurde bei den Experimenten oft der Fall beobachtet, dass nur ein Teil der im aktuellen Bild extrahierten Merkmale zu den Merkmalen im vorigen Bild korrespondierend war. Die relativ hohe Fluktuation in der Anzahl und Position der auf einem Objekt gefundenen SIFT-Merkmale führte dazu, dass der daraus berechnete Durchschnitt keinesfalls der Objektmitte entsprach. Im Gegenteil, die so berechnete Position schwankte je nach Aufnahme um den Mittelpunkt des Objektes.

Der im Rahmen dieser Arbeit entwickelte Algorithmus zur Detektion von periodischen Bewegungen basiert auf den in Abbildung 3.3 gezeigten Daten und verarbeitet jeden neuen Wert einer solchen Kurve online. Dabei werden zur Laufzeit die auftretenden Maxima, Minima sowie die Periode der Schwingung erfasst und verarbeitet. Der Algorithmus lässt sich in zwei Phasen aufteilen:

**Initialisierungsphase:** Während dieser Phase werden zunächst Werte der folgenden drei Variablen geschätzt: *predicted\_maximum*, *predicted\_minimum* und *predicted\_period*. Diese Werte werden jeweils als Durchschnitt der bis zum aktuellen Zeitpunkt detektierten Extrema und der sich daraus ergebenden Periodenlängen berechnet. Als ein Maximum gilt ein Wert, wenn dieser höher ist als seine bezüglich der zeitlichen Achse benachbarten Werte. Entsprechendes gilt auch für ein Minimum. Die Länge der Periode wird durch den zeitlichen Abstand der zwei aufeinander folgenden Maxima oder Minima berechnet.

**Arbeitsphase:** Während dieser Phase wird bestimmt, ob die während der Objektbewegung resultierende Kurve einen periodischen Verlauf hat oder nicht. Bei jedem neu detektierten Maximum oder Minimum wird zuerst überprüft, ob dieser in einem vorher definierten Intervall der Breite *SE* um den bis dahin berechneten Wert der Variablen *predicted\_maximum* oder *predicted\_minimum* liegt. Bei einem positiven Ergebnis wird als nächstes verifiziert, in wie fern die aktuelle Periodenlänge dem bereits vorher berechneten Wert der Variable *predicted\_period* entspricht. Auch hier wird ein gewisser Schwellenwert *SP* für die maximal zulässige Abweichung festgelegt. Erst nachdem die beiden Test erfolgreich waren, wird eine Bewegung als periodisch markiert und das entsprechende Objekt für die anschließende Generierung eines Modells ausgewählt.

Weil sich die einzelnen Schritte des Algorithmuses kaum verständlich in einer sequentiellen Reihenfolge aufschreiben lassen, wurde in diesem Fall die Form des Pseudo-Codes gewählt. Dieses ist in Abbildung 1 gegeben.

Daraus ist ersichtlich, dass die absolute zeitliche Trennung in die beiden oben genannten Phasen nur bedingt möglich ist. Die Werte der erwähnten Variablen werden

---

**Algorithm 1** Algorithmus zur Bestimmung, ob eine Kurve eine periodische Form aufweist.

---

```
1: while neue Daten vorhanden do
2:   if neues Maximum detektiert then
3:     if predicted_maximum wurde nicht berechnet then
4:       berechne predicted_maximum;
5:     else
6:       if neues Maximum liegt nah am predicted_maximum then
7:         if predicted_period wurde nicht berechnet then
8:           berechne predicted_period;
9:         else
10:          berechne neue Periode;
11:          if neue Periode liegt nah am predicted_period then
12:            setze die Variable periodic_movement auf true;
13:          else
14:            setze die Variable periodic_movement auf false;
15:          end if
16:          aktualisiere predicted_period;
17:        end if
18:      end if
19:      aktualisiere predicted_maximum;
20:    end if
21:  end if
22:  ... führe ähnliche Schritte durch, wenn ein Minimum detektiert wurde
23: end while
```

---

fortwährend auch während der Arbeitsphase aktualisiert. In der im Rahmen der Arbeit erfolgten Implementierung wurden zur Durchschnittsbildung jeweils die letzten drei Maxima, Minima und die zuletzt berechneten Periodenlängen herangezogen.

Die in der Beschreibung erwähnten Schwellenwerte erlauben eine bequeme Regelung darüber wie genau das entsprechende Objekt die periodische Bewegung durchführen soll. In dem in Abbildung 3.3 gezeigten Beispiel ist deutlich zu sehen, dass die während der Bewegung resultierenden Maxima und Minima nicht die gleichen Werte haben. Je kleiner die gewählten Schwellenwerte sind, desto exakter und sauberer sollte die periodische Bewegung sein, um als solche erkannt zu werden.

Im Falle einer erfolgreichen Detektion wird für das entsprechende Objekt ein 'multiview appearance' Modell generiert. Das Vorgehen dazu erfolgt im Online-Betrieb und wird im nächsten Abschnitt ausführlich erläutert.

### 3.3 Online Bildung eines Objektmodells

Die Generierung eines 'multiview appearance' Modells basiert auf der Bildinformation, die mittels der einfachen Objektverfolgung gesammelt wurde. Die unterschiedlichen Objektansichten werden dabei sukzessive dem Modell hinzugefügt bis das entsprechende Modell vollständig ist und das Objekt robust in allen erdenklichen Positionen im Bild wiedererkannt werden kann. Mittels der so erhöhten Wiedererkennungsratesoll erreicht werden, dass das entsprechende Objekt viel stabiler in der darauf folgenden Bildsequenz verfolgt werden kann. Denn, wie bereits erwähnt, würde die oben beschriebene einfache Methode zur Objektverfolgung schon bei einer etwas schnelleren Objektrotation versagen. Unter Zuhilfenahme des entsprechenden Objektmodells sollen diese Probleme umgangen werden.

Das grundlegende Vorgehen der Modellgenerierung ist in Abbildung 3.5 graphisch dargestellt. Nach der Detektion des gesuchten Objektes in der Bildsequenz werden zunächst aus dem entsprechenden Bildausschnitt SIFT-Merkmale extrahiert. Diese werden später bei der Modellgenerierung verwendet. Wie bereits im Vorfeld erwähnt besteht hier das Modell aus unterschiedlichen Ansichten des interessierenden Objektes. Diese Ansichten werden in der Datenbank als eine Ansammlung der daraus extrahierten SIFT-Merkmale repräsentiert. Eine Indexierung der Ansichten sorgt dafür, dass diese eindeutig dem entsprechenden Modell zugeordnet werden.

Wie der Abbildung 3.5 zu entnehmen ist, können bei der Modellgenerierung unterschiedliche Fälle eintreten. Diese werden im Folgenden genauer beschrieben:

**Fall 1:** Dieser Fall tritt ein, wenn das Objekt zum ersten Mal in der Bildsequenz als das interessierende Objekt detektiert wurde. Zu diesem Zeitpunkt ist in

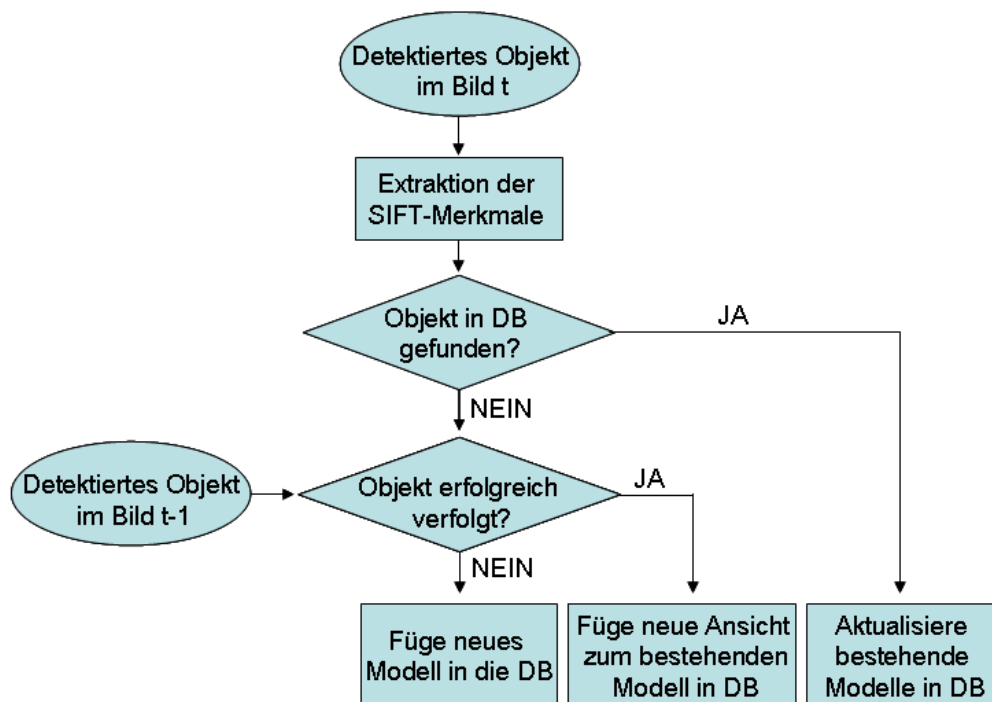


Abbildung 3.5: Grafische Repräsentation des prinzipiellen Vorgehens bei der online Modell Generierung.

der Datenbank noch kein Modell gespeichert. Die neu extrahierten SIFT-Merkmale können keiner Ansicht zugeordnet werden und es kann auch kein korrespondierendes Objekt in dem vorigen Bild gefunden werden, weil das Verfahren erst mit dem aktuellen Bild startet. In diesem Fall wird in der Datenbank ein neues Modell angelegt, und diesem die neu extrahierten Merkmale in Form einer neuen Objektansicht zugeordnet.

**Fall 2:** Es kann vorkommen, dass eine neue Objektansicht zwar mit keiner der, in der Datenbank gespeicherten Ansichten übereinstimmt und dass aber trotzdem ein korrespondierendes Objekt im vorigen Bild gefunden werden konnte. Der Grund dafür ist, dass bei der oben beschriebenen einfachen Methoden zur Objektverfolgung die Anzahl der gefundenen SIFT-Korrespondenzen für die erfolgreiche Objektzuordnung nicht relevant ist. Führt das Objekt eine langsame Rotationsbewegung aus, so reicht bereits eine SIFT-Korrespondenz aus, um die beiden Ansichten in zwei aufeinander folgenden Bildern einander zuzuordnen. Es sei an dieser Stelle angemerkt, dass für einen positiven Abgleich mit der Datenbank mindestens acht SIFT-Korrespondenzen notwendig sind. Tritt dieser Fall auf, so wird das bereits bestehende Modell in der Datenbank um die neue Objektansicht erweitert. Dabei wird das Modell ausgewählt, zu

welchem die Objektansicht im vorigen Bild zugeordnet wurde.

**Fall 3:** In diesem Fall wird ein detektiertes Objekt einem oder gleich mehreren Modellen in der Datenbank zugeordnet. Dabei wird in der ersten Situation zunächst überprüft, wie gut die neue Objektansicht zu den bereits gespeicherten Ansichten des Modells passt. Wird hier nur die minimal erforderliche Anzahl an SIFT-Korrespondenzen gefunden, so wird die neue Ansicht dem Modell hinzugefügt. Damit wird die Entscheidung über die Erweiterung oder Beibehaltung des alten Modells auf der Basis des dabei erzielten Übereinstimmungsgrads getroffen. Ist die Anzahl der gefundenen SIFT-Korrespondenzen höher als ein bestimmter Schwellenwert, so wird davon ausgegangen, dass das vorhandene Modell ausreichend ist. In diesem Fall wird keine neue Objektansicht dem Modell hinzugefügt.

Kann eine aktuelle Objektansicht zwei Modellen gleichzeitig zugeordnet werden, so werden die beiden Modelle zu einem zusammen verschmolzen. Alle Ansichten, die die beiden Modellen beinhalteten, werden nun einem Modell zugeordnet.

Die zweite Situation im oben beschriebenen dritten Fall kann auftreten, wenn mehrere Modelle in der Datenbank das gleiche Objekt repräsentieren. Wie bereits erwähnt wird ein weiteres Modell in der Datenbank angelegt, wenn die aktuelle Ansicht des Objektes mit denen in der Datenbank nicht übereinstimmt und auch kein korrespondierendes Objekt im vorigen Bild gefunden werden konnte. Dieser Fall kann durch eine plötzliche und sehr schnelle Rotationsbewegung des Objektes hervorgerufen werden. Nach solch einer Rotation wird üblicherweise die so gewonnene neue Objektansicht einem neuen Modell in der Datenbank zugeordnet.

Der Zeitpunkt des Zusammenführens mehrerer Modelle hängt entscheidend von der Art der Objektbewegung ab. Werden ausschließlich abrupte und immer die gleichen Rotationsbewegungen durchgeführt, so bestehen wenig Chancen für ein Zusammenschmelzen der Modelle. In den meisten Situationen kann allerdings davon ausgegangen werden, dass schon nach kurzer Zeit eine Objektansicht aufgenommen wird, die jeweils einer Ansicht der entsprechenden Modellen zugeordnet werden kann. Der dabei erzielte Übereinstimmungsgrad ist üblicherweise gering. Daher wird oft auch die aktuelle Objektansicht in das neu konstruierte Modell mit aufgenommen.

Mittels des bestehenden Objektmodells lässt sich die Verfolgung dieses Objektes viel robuster und stabiler durchführen. Falls die oben beschriebene einfache Methode zur Objektverfolgung versagt, kann die aktuelle Bildinformation immer noch mit der bereits bestehenden Datenbank verglichen werden. Der Umfang und der Informationsgehalt der Datenbank hängt allerdings entscheidend davon ab, wie lange

das System im Einsatz ist. Aufgrund des Online-Betriebs kann davon ausgegangen werden, dass je länger das System aktiv ist, desto umfangreicher und konsistenter die Datenbank sein wird.

Der folgende Abschnitt beschreibt die hierbei erzielten Ergebnisse und zeigt beispielhaft die mittels dieses Verfahrens generiertes Objektmodell.

## 3.4 Ergebnisse der Objektverfolgung

Das in diesem Kapitel vorgestellte Verfahren wurde in mehreren Experimenten auf seine Tauglichkeit untersucht. Als Versuchsobjekt diente dabei ein ER1 Roboter von 'Evolution Robotics'. Mittels eines manuellen Steuerungssystems ließ sich dieser komfortabel in die gewünschte Position navigieren oder eine vorher definierte Bewegung durchführen.

Um die Ergebnisse der einzelnen Module strukturiert darstellen zu können, wurde dieser Abschnitt in die folgenden drei Unterabschnitte aufgeteilt: 1. Konstruktion einer initialen Objekttrajektorie, 2. Erkennung einer periodischen Bewegung und 3. Online Konstruktion eines Robotermodells.

### 3.4.1 Konstruktion einer initialen Objekttrajektorie

Zur Testzwecken dieses Moduls wurde der Roboter während seiner Bewegung von einer höher positionierten Kamera aufgenommen. Abbildung 3.6 zeigt einige Aufnahmen dieser Bildsequenz. Die erste Reihe zeigt Originalbilder der Kamera zu unterschiedlichen Zeitpunkten. Diese können an der entsprechenden Bild\_id abgelesen werden. Die zweite Reihe besteht aus Bildern, die das Ergebnis des in Kapitel 2 vorgestellten Vordergrund - Hintergrund Separationsverfahrens sind. Die Konturen des Roboters sind hier deutlich zu erkennen. Darauf basierend wurden mittels des bereits erwähnten Algorithmuses zur Berechnung von 'Connected Components' die in der unteren Reihe dargestellten Ausschnitte des originalen Bildes berechnet.

Die auf diese Weise erfolgte Extraktion der bewegenden Objekte einer Szene funktionierte sehr stabil. Im Rahmen mehrerer durchgeführter Experimente konnten damit in den verwendeten Testsequenzen alle zum Vordergrund gehörende Objekte extrahiert und die rechtwinkligen Rechtecke um diese berechnet werden. Die Stabilität dieses Vorgehens geht allerdings zum großen Teil auf das hier verwendete Vordergrund-Hintergrund Separationsverfahren zurück.

Die Bilder in der letzten Reihe der Abbildung dienten zugleich auch als Eingabe für das bereits beschriebene Verfolgungsverfahren, welches auf der Bestimmung der SIFT-Korrespondenzen zwischen zwei aufeinander folgenden Bildern basiert. Damit genügend SIFT-Merkmale gefunden werden konnten, wurde der vordere Teil des

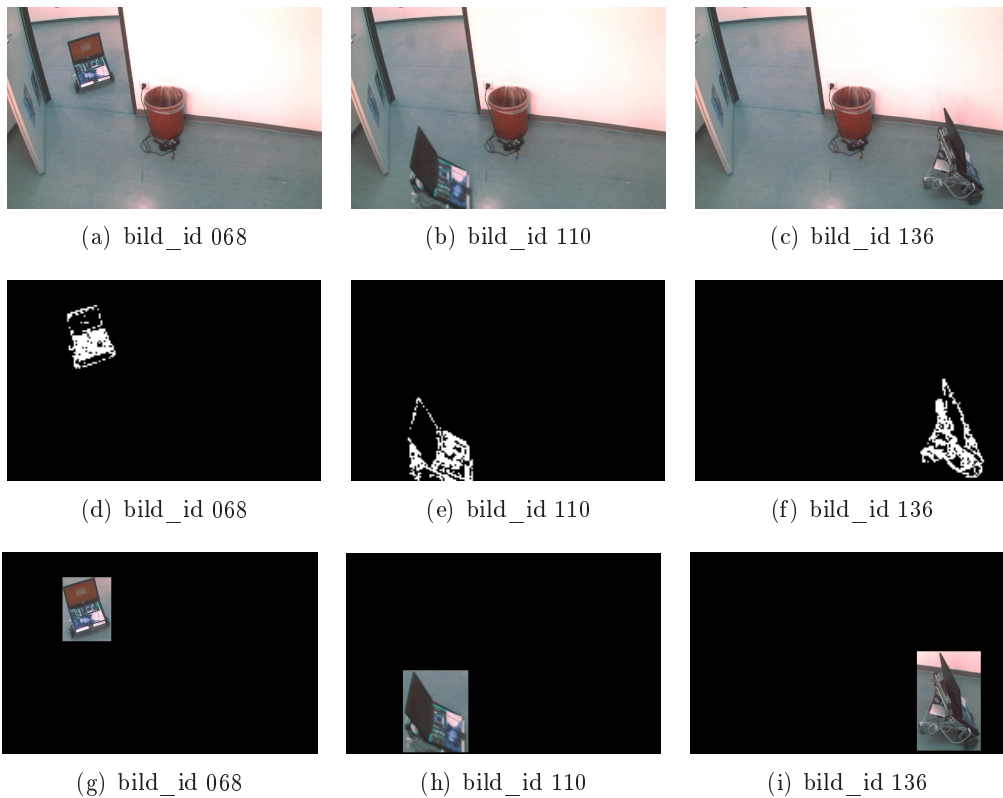


Abbildung 3.6: Einige Aufnahmen einer Bildsequenz während der Roboterbewegung; 1. Reihe: Originalaufnahmen der Kamera zu unterschiedlichen Zeitpunkten; 2. Reihe: Vordergrundbilder des im Kapitel 2 vorgestellten Separationsverfahrens; 3. Reihe: Ergebnisse des Moduls 'Komponenten Extraktion'.



Notebooks mit einem Zeitschriften Artikel beklebt. Besonders deutlich ist dies in Abbildungen 3.6 (a) oder (b) zu sehen. Die Ergebnisse der durchgeführten Experimente haben allerdings gezeigt, dass eine stabile Berechnung der Objekttrajektorien mit diesem Verfahren nur unter den folgenden Bedingungen möglich ist.

- Stabiles Kamerabild und eine robuste Merkmalsextraktion.
- Keine plötzliche und zugleich starke Änderung der Objektansicht im Kamerabild.
- Keine zeitweise Verdeckung des bewegenden Objektes durch andere Gegenstände.
- Keine Kreuzung zweier verschiedener Objekttrajektorien.

Ist eines dieser Voraussetzungen nicht erfüllt, so sind die berechneten Trajektorien oft zu kurz, um darauf basierend die Periodizität der zugehörigen Bewegung festzustellen. Immer wieder werden die bereits erzeugten Trajektorien unterbrochen, weil keine korrespondierende Merkmale zwischen zwei Bildern gefunden werden. Dies kommt immer dann vor, wenn das Objekt von anderen verdeckt wird, oder wenn es plötzlich seine Richtung ändert. Bei einem Zusammentreffen und erneutem Auseinanderdriften zweier Objekte können die entsprechenden Trajektorien oft nicht fortgesetzt werden, weil das Verfahren keine Information darüber, erhält welches Objekt welcher Trajektorie nach dessen Kreuzung zugeordnet werden soll.

Eine alternative Lösung zur initialen Objektverfolgung könnte auf der Überlappung der Rechtecke basieren. Rechtecke in zwei aufeinander folgenden Bildern wären dann als zueinander korrespondierend zu betrachten, wenn sich der Bildbereich dieser überlappt. Dieser Ansatz würde allerdings nur unter der Annahme funktionieren, dass die Aufnahmefrequenz der Kamera hoch ist und die Breite des Rechtecks um das betrachtete Objekt nicht zu klein.

Denkbar wäre auch ein Ansatz, welches sich die Histogramme der Rechtecke zu Nutze macht. Doch auch damit würde man die Probleme bei zeitweiser Verdeckung der Objekte oder Kreuzung zweier Trajektorien nicht lösen. Ähnliches gilt auch für die meisten in [YJS06] beschriebenen Verfahren, die unter den gegebenen Bedingungen anwendbar wären. Aus diesem Grund wurde im Rahmen dieser Arbeit der SIFT basierte Ansatz beibehalten.

### 3.4.2 Erkennung einer periodischen Bewegung

Zur Testzwecken des Moduls 'Erkennung einer periodischen Bewegung' wurden drei verschiedene Bewegungsmuster ausprobiert: 1. kontinuierliche Vor- und Rückwärtsbewegung; 2. kreisförmige Bewegung; 3. rechteckige Bewegung. Die Schwellenwerte

des Verfahrens wurden wie folgt initialisiert:  $SE = 20$  und  $SP = 20$ . Mit  $SE = 20$  wurde festgelegt, dass im Rahmen der folgenden Experimente die maximal mögliche Abweichung zwischen dem durchschnittlichen Maximum oder Minimum und der neu berechneten Extremstelle die Grenze von 20 Pixeln nicht überschreiten darf. Mit  $SP = 20$  wurde entsprechend die maximal mögliche Abweichung der neuen Periodenlänge zu der bis zum aktuellen Zeitpunkt berechneten durchschnittlichen Periodenlänge festgelegt.

Die hierbei erzielten Ergebnisse sind in den folgenden Unterabschnitten beschrieben.

### Erkennung einer kontinuierlichen Vor- und Rückwärtsbewegung

Die einzelnen Punkte der bei einer Vor- und Rückwärtsbewegung resultierenden Trajektorie im Kamerabild sind in Abbildung 3.7 (a) zusehen. Die Koordinaten der hier abgebildeten Punkte entsprechen den gemessenen Mittelpunkten der das Objekt umrandeten Rechtecke.

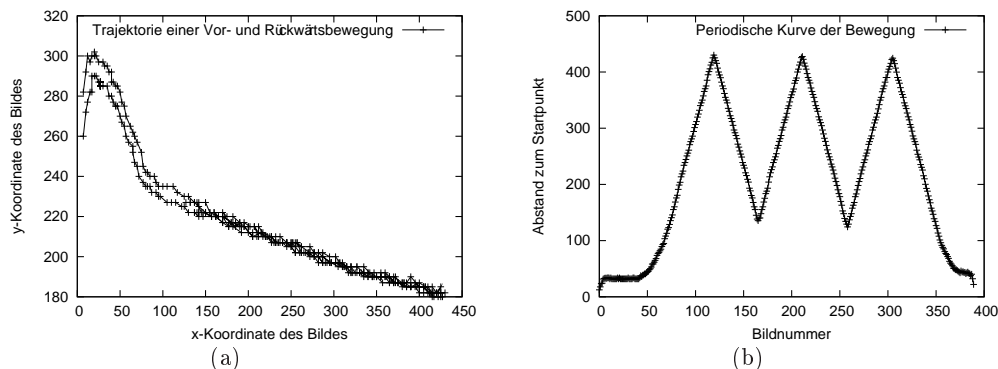


Abbildung 3.7: (a) zeigt im Kamerabild aufgenommene Objekttrajektorie einer Vor- und Rückwärtsbewegung, die insgesamt drei mal durchgeführt wurde; (b) zeigt die entsprechende periodische Kurve, die ausgehend von den Daten im Bild (a) berechnet wurde.

Die beiden Erhöhungen in dem linken Teil der Abbildung 3.7 (a) resultieren aus der Tatsache, dass sich in diesem Bereich des Kamerabildes der Roboter in das Sichtbereich der Kamera bewegt und diesen anschließend wieder verlässt. Während dieser beider Prozedere ist der Roboter nur teilweise sichtbar, weshalb sich auch die Größe des umrandeten Rechtecks hier sehr stark ändert. Zusammen mit der Größe des Rechtecks ändert sich auch die Position des jeweiligen Mittelpunktes. Dieses verschiebt sich zunächst etwas nach oben und anschließend wieder etwas nach unten. Die beiden Eckpunkte der Vor- und Rückwärtsbewegung haben bei diesem Experiment in etwa die folgenden Koordinaten: (150, 220) und (430, 180).

Abbildung 3.7 (b) zeigt die auf diesen Daten berechnete periodische Kurve. Die Maxima und Minima der Kurve entsprechen den Änderungen der Bewegungsrichtung des Roboters. Insgesamt wurde der Roboter hier drei mal hin und her bewegt. In der Abbildung lässt sich leicht erkennen, dass die oberen Extrema nicht alle den gleichen Wert haben. Obwohl bei diesem Versuch sehr auf die Gleichmäßigkeit der Bewegung geachtet wurde, konnte auch hier der ideale Fall, alle Maxima und Minima haben jeweils den gleichen Wert, nicht erreicht werden. Doch aufgrund der gesetzten Schwellenwerte, wurde die Bewegung trotzdem als periodisch erkannt.

### Erkennung einer kreisförmigen Bewegung

Im zweiten Versuch wurde zur Demonstrationszwecken der Roboter auf eine kreisförmige Bewegung vorprogrammiert. Die Abbildung 3.8 zeigt die dazu gehörige Trajektorie des Roboters und die dabei resultierende periodische Kurve. Sowohl bei

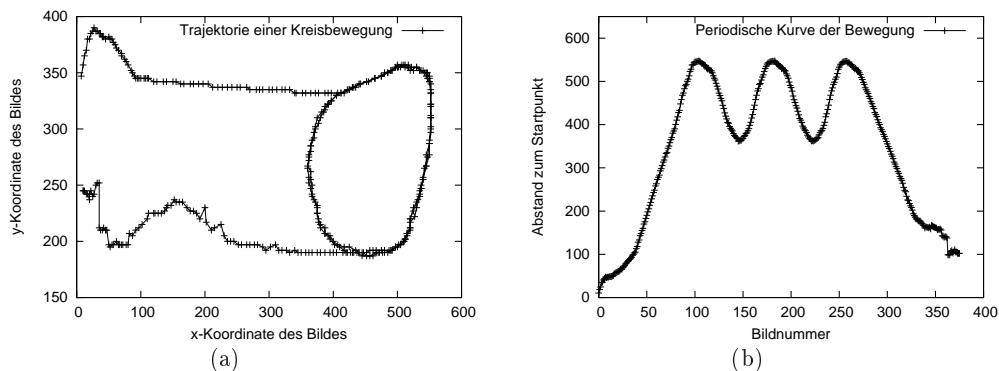


Abbildung 3.8: (a) zeigt im Kamerabild aufgenommene Objekttrajektorie einer kreisförmigen Bewegung, die insgesamt drei mal durchgeführt wurde; (b) zeigt die entsprechende periodische Kurve, die ausgehend von den Daten im Bild (a) berechnet wurde.

diesem als auch beim darauf folgenden Versuch wurde zur Bestimmung der Robotertrajektorie ein Verfolgungsansatz verwendet, welches auf der Überlappung zweier Rechtecke in zwei aufeinander folgenden Bildern basiert. Dies war notwendig, weil sich die Ansicht des Roboters im Bild aufgrund der gewählten Bewegungsart ständig änderte.

In Abbildung 3.8 (a) ist deutlich zusehen, dass die im Bild gemessene Trajektorie des Roboters nicht exakt einem Kreis entspricht. Dies ist allerdings kein Nachteil, sofern die auf den Daten berechnete Abstandskurve einen periodischen Charakter hat. Abbildung 3.8 (b) zeigt das Ergebnis dieser Berechnung. Im Gegensatz zur Abbildung 3.7 (b) ist hier die Kurve zwar etwas glatter, doch sind die Extrema auch hier deutlich zu erkennen.

### Erkennung einer rechteckigen Bewegung

Im dritten Versuch wurde der Roboter entlang eines Rechtecks manuell gesteuert. Die Abbildung 3.9 (a) zeigt die dabei im Kamerabild resultierende Trajektorie. Aufgrund der manuellen Steuerung weichen die Trajektorien der einzelnen Recht-

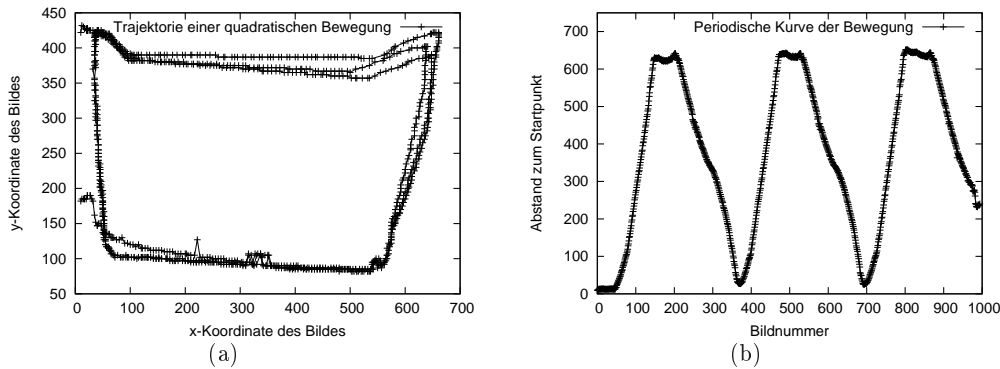


Abbildung 3.9: (a) zeigt im Kamerabild aufgenommene Objekttrajektorie einer rechteckigen Bewegung, die insgesamt drei mal durchgeführt wurde; (b) zeigt die entsprechende periodische Kurve, die ausgehend von den Daten im Bild (a) berechnet wurde.

eckumrandungen voneinander ab. An zwei Stellen sind auch die Ecken der resultierenden Rechtecke etwas verformt. Dies liegt daran, dass an diesen Stellen der Roboter während der Rotation nur teilweise im Bild der Kamera zu sehen war. Entsprechend haben sich auch die Mittelpunkte der Rechtecke nach Außen verschoben.

Die so resultierenden Fehler in der im Kamerabild gemessenen Objekttrajektorie spiegeln sich auch in der auf diesen Daten berechneten Kurve. Diese ist in Abbildung 3.9 (b) zu sehen. Anders als bei vorigen Experimenten sind hier an Stelle der Maxima kleine Plateaus zu sehen. In diesem Fall würde das Verfahren in seiner aktuellen Version die Periodizität dieser Kurve nicht erkennen. Die zur Erfassung der Maxima formulierte Bedingung geht vor der Definition aus, dass die Werte vor und nach der besagten Stelle kleiner sind als an dem potentiellen Maximum. Eine Umformulierung dieser Bedingung oder einfache Modifikationen der Kurve mittels eines zwischengeschalteten Verarbeitungsschrittes könnte dieses Problem allerdings leicht beheben. Als Beispiele für die Zwischenverarbeitung wäre ein Subsampling der Kurve oder eine gröbere Diskretisierung der Kurvenwerte denkbar.

Zusammenfassend lässt sich sagen, dass das hier angewendete Verfahren in der Lage ist eine beliebige sich ständig wiederholende Objektbewegung als periodisch zu erkennen, solange die auf der entsprechenden Trajektorie berechnete Kurve einen periodischen Charakter hat. Eine der wichtigen Bedingungen dafür ist, dass der Start-

und Endpunkt der Trajektorie ungefähr die gleichen Koordinaten haben. Sowohl die erforderliche Gleichmäßigkeit als auch die Genauigkeit der periodischen Bewegung kann mittels beider Schwellenwerte  $SE$  und  $SP$  der Anwendung entsprechend angepasst werden.

### 3.4.3 Online Konstruktion eines Robotermodells

Die erfolgreiche anfängliche Detektion des Roboters im Bild diene dann sogleich als Startpunkt für das Modul 'Online Modellgenerierung'. Während dieser Phase wurde der Roboter willkürlich durch den Raum bewegt. Dabei wurde darauf geachtet, dass dieser mehrere unterschiedliche Rotationen durchführt. Abbildung 3.10 zeigt beispielhaft die verschiedenen Ansichten eines Modells, welches während eines solchen Experimentes online erzeugt wurden.

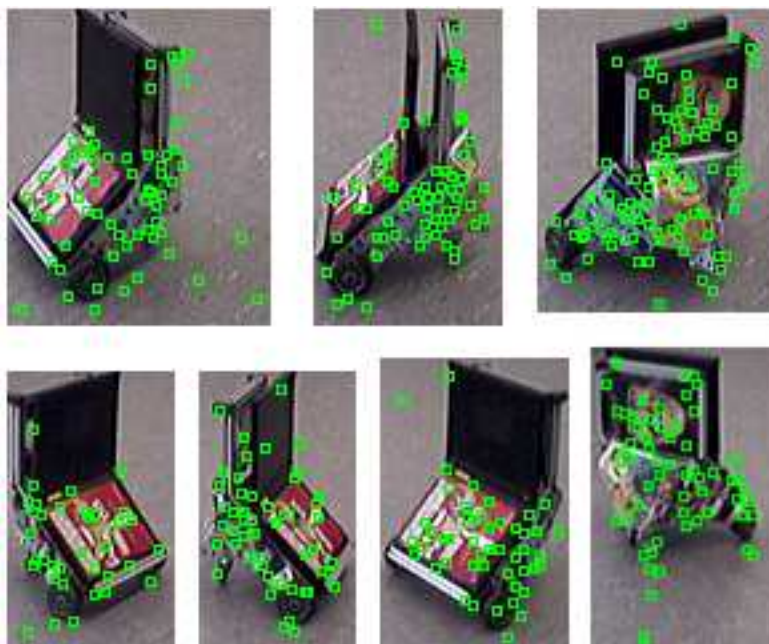


Abbildung 3.10: Unterschiedliche Ansichten eines Modells des Roboters, die online während der Roboterbewegung konstruiert wurden.

Die grünen Rechtecke in den unterschiedlichen Bildern entsprechen den Positionen den dabei extrahierten SIFT-Merkmale. An einigen Stellen ist deutlich zu sehen, dass diese nicht auf dem Roboter liegen, sondern auf dem Boden und damit auf dem Hintergrund der Szene. Diese Tatsache ist jedoch nicht als Nachteil zu beurteilen. Im Gegenteil, aufgrund des in einigen Ansichten vorhandenen Hintergrunds ist das generierte Modell optimal an die entsprechende Umgebung angepasst. Gleichzeitig werden auch nur solche Ansichten von dem Objekt gespeichert, die mit der

verwendeten Kamera aufgenommen werden können. So ist zum Beispiel in der obigen Abbildung die unterschiedliche Skalierung des Roboters in den verschiedenen Ansichten deutlich zu erkennen. Dies liegt daran, dass der Roboter während der entsprechenden Rotation eine Position etwas weiter weg von der Kamera hatte.

Unter Zuhilfenahme dieses Modells war es nun möglich den Roboter über einen längeren Zeitraum im Bild zu verfolgen. Die anfangs erwähnten Probleme bei einer plötzlichen Änderung der Bewegungsrichtung des Roboters oder eine zeitweise Verdeckung dessen durch andere Gegenstände spielten hier kaum eine Rolle. Mit jedem neu aufgenommenen Kamerabild wurden, die darauf extrahierten SIFT-Merkmale mit denen in der Datenbank gespeicherten Merkmalen verglichen und bei einem hohen Ähnlichkeitsgrad, der Roboter als solcher identifiziert.

#### 3.4.4 Laufzeit des Verfahrens

Die Echtzeitverarbeitung des hier vorgestellten Verfahrens zur Verfolgung von Objekten hängt von der Möglichkeit der Echtzeitverarbeitung seiner einzelnen Komponenten ab. Wie bereits in Kapitel 2 gezeigt, ist das Modul der Vordergrund - Hintergrund Separation für die Analyse der ankommenden Kamerabildern in Echtzeit geeignet. Die anschließende Berechnung der achsenparallelen Rechtecke um die bewegenden Objekte stellt ebenfalls kein Problem dar. Aufgrund der betrachteten Anwendung ist davon auszugehen, dass im Bild nicht viele bewegliche Objekte gleichzeitig zu sehen sein werden. Denn im anderen Fall würden sich die, ins Kamerabild projizierten Ansichten dieser Objekte überschneiden, womit eine klare Trennung der Objektkonturen nicht mehr möglich wäre. Aus diesem Grund ist die Anzahl der im vierten Schritt des Moduls 'Komponenten Extraktion' notwendigen Iterationen beschränkt. Die Echtzeitverarbeitung dieses Moduls ist damit gegeben.

Eine andere Situation ergibt sich bei der im Verfahren verwendeten Extraktion und dem Vergleich der SIFT-Merkmale. Im Rahmen dieser Arbeit wurde eine Implementierung aus dem ERSP 3.1 Software Development Kit<sup>1</sup> von Evolution Robotics verwendet. Dieses Software Paket bildet die Basis für alle am Lehrstuhl 'Intelligente Systeme' entwickelten Algorithmen zur Steuerung der eingesetzten Roboter und zur Verarbeitung der Kamerabilder. Aufgrund der sehr Ressourcen aufwendigen Rechenprozedur der SIFT-Merkmale war hiermit eine Echtzeitverarbeitung der Kamerabilder nicht möglich. Abhängig von der Anzahl der in der Datenbank gespeicherten Ansichten dauerte die Extraktion zusammen mit der Zuordnung der Merkmale bis zu 1,5 *sek* pro Bild. Allerdings ist an dieser Stelle anzumerken, dass im Moment keine CPU basierte SIFT-Implementierung existiert, die eine Echtzeit-

---

<sup>1</sup>Nähere Informationen dazu sind unter dem folgenden Link verfügbar: <http://www.evolution.com/products/ersp/>

verarbeitung der Bilder erlauben würde. Aus diesem Grund hat die Arbeitsgruppe der Abteilung Informatik der Universität von North Carolina eine GPU basierte SIFT-Implementierung entwickelt [Wu07]. Die einzelnen Schritte der Berechnungsprozedur werden hier parallel abgearbeitet. Die Evaluierungsergebnisse dieser Implementierung zeigen, dass bei einer Bildgröße von  $320 \times 240$  damit bis zu 45 *fps* verarbeitet werden können. Weitere GPU Implementierungen finden sich in [HMS<sup>+</sup>07] oder [SFPG07]. Zusammen mit der in letzter Zeit beobachteten rasanten Entwicklung der multi-core Prozessoren, existieren bereits auch Ansätze, die darauf basierend eine Parallelisierung der Berechnungsprozedur der SIFT-Merkmale ermöglichen [ZCZX08].

Damit ist zu erwarten, dass schon bereits in der nächsten Zeit ein frei verfügbares Software Paket entwickelt wird, welches unter Verwendung bestimmter Hardware Komponenten eine Echtzeitberechnung der SIFT-Merkmale erlauben wird. Damit wäre auch das im Rahmen dieser Arbeit entwickelte Verfahren echtzeitfähig.

### 3.5 Diskussion und Zusammenfassung

Das hier präsentierte Verfahren zur automatischen Generierung eines Objektmodells zeigte im Laufe der durchgeführten Experimente fortwährend eine gute Performanz. Die anfängliche Detektion der bewegenden Objekte mittels des im letzten Kapitel vorgestellten Verfahrens zur Vordergrund - Hintergrund Separation sorgte schon von vorn herein für ein sauberes Vordergrundbild. Das interessierende Objekt wurde dann aufgrund seiner periodischen Bewegung als das gesuchte Objekt erkannt und ausgehend von der so gewonnenen ersten Objektansicht ein 'multiview appearance' Modell von diesem erzeugt.

Die Identifikation einer periodischen Bewegung erfolgte mittels eines im Rahmen dieser Arbeit entwickelten Algorithmuses. Dabei wurden die anfänglichen Rohdaten zunächst in eine Form umgerechnet, so dass das sequentielle Auftragen der einzelnen Werte in eine Graphik in einer periodischen Kurve resultierte. Die Hauptaufgabe des Algorithmuses bestand dann in der Erkennung eines periodischen Musters einer solchen Kurve. Die Restriktion auf die Erkennung von periodischen Bewegungen erfolgte hier aus zwei Gründen. Zum einen ist es die Einfachheit dieser Bewegung. Jeder mobiler Roboter ist in der Lage diese auszuführen. Und zum anderen sind es die charakteristischen Eigenschaften, die diese Bewegungsart so besonders gut für die Wiedererkennung eignen lassen.

Die Berechnung der Objekttrajektorien im Bild erfolgte zunächst mittels eines einfachen Verfahrens zur Objektverfolgung. Neben dem Vordergrund-Hintergrund Separationsverfahrens aus dem letzten Kapitel wurden hier zur Korrespondenzfin-

dung SIFT-Merkmale verwendet. Dieses Verfahren lieferte nur dann eine gute Performanz wenn die Ansichten der bewegenden Objekte sich von Bild zu Bild wenig änderten. Bei schnellen Rotationsbewegungen versagte diese Methode. Zur Konstruktion einer periodischen Objekttrajektorie waren die Ergebnisse allerdings ausreichend. Dieses Verfahren diente auch als Basis für die online Generierung eines Objektmodells.

Bei dem Verfahren zur sukzessiven Konstruktion eines 'multiview appearance' Modells wurden mehrere Fälle unterschieden. Dadurch wurden auch solche Situationen abgedeckt, in denen die Objektverfolgung mittels des einfachen Verfahrens versagte. Während der durchgeführten Experiment konnte beobachtet werden, dass je länger das Verfahren aktiv war, desto umfangreicher und besser die dabei erzeugten Modelle waren. Die so konstruierten Objektmodelle waren optimal an die jeweilige Anwendungsumgebung angepasst. In der Datenbank wurde keine Ansicht gespeichert, die nicht aus der jeweiligen Kameraperspektive hätte aufgenommen werden können. Manchmal wurden auch solche Ansichten dem Modell hinzugefügt, die das Objekt durch einen anderen Gegenstand verdeckt zeigen. Damit konnte die Verfolgung auch dann fortgesetzt werden, wenn das entsprechende Objekt nur zum Teil sichtbar war.

Zusammenfassend lässt sich sagen, dass mittels eines so erzeugten Objektmodells die Objektverfolgung im Bild um einiges stabiler erfolgen konnte, als dies mit den anfangs erwähnten Verfahren zur Konstruktion einer initialen Objekttrajektorie möglich gewesen wäre.

Das Verfahren zur bewegungsbasierten Objekterkennung wurde zum ersten Mal auf dem zweiten 'International Robot Vision Workshop' im Jahr 2008 breiterem Publikum vorgestellt [SPH08]. Das Verfahren zur online Konstruktion eines Objektmodells wurde im Jahr 2009 auf der vierten 'International Conference on Computer Vision Theory and Applications' zum ersten Mal präsentiert [SCSP09].



# Entfernungsschätzung bei geradliniger Kamerabewegung

---

## Kapitelinhalt

---

4.1	Theorie zur Berechnung von Time-To-Collision . . . . .	72
4.2	Darstellung der Problematik . . . . .	77
4.3	Bewegungsmodell basierte TTC Berechnung . . . . .	83
4.4	Ergebnisse im Rahmen einer Simulationsumgebung . . . . .	92
4.5	Diskussion und Zusammenfassung . . . . .	101

---

In vielen Bereichen der Fahrzeugnavigation ist eine robuste Entfernungsschätzung zu bevorstehenden Hindernissen oft von unentbehrlicher Wichtigkeit. Steht dabei kein Stereo-Kamerasystem zur Verfügung, so wird die erforderliche Information aus den Bildern nur einer Kamera gewonnen. Voraussetzung dafür ist die Bewegung der Kamera. Wird diese zwischen zwei aufeinander folgenden Bildaufnahmen verschoben, so kann die Entfernungsschätzung basierend auf den dabei resultierenden Verschiebungen der Objekte im Kamerabild erfolgen. Ein dafür weit verbreiteter Ansatz ist die Berechnung der verbleibenden Zeit bis zum Zusammenstoß der Kamera mit den bevorstehenden Objekten. Der Fachbegriff dafür lautet 'Time-To-Collision' (TTC, Zeit bis zum Zusammenstoß).

Die Qualität und die Aussagekraft der berechneten TTC-Werte hängt dabei entscheidend von der Messgenauigkeit der entsprechenden Punktpositionen im Bild ab. Im Rahmen dieses Kapitels wird gezeigt, dass schon kleinere Abweichungen in den Messungen zu stark schwankenden TTC-Werten führen können. Zuversichtliche Aussagen über den Abstand zum entsprechenden Objekt sind damit nicht möglich.

Das vorliegende Kapitel beschreibt ein im Rahmen dieser Arbeit entwickeltes Verfahren zur robusten Berechnung der TTC-Werte. Das Verfahren liefert stabile Ergebnisse trotz verrauschter Messdaten. Es basiert auf den Gesetzen der projektiven Geometrie und verwendet zur genaueren TTC-Schätzung die theoretisch richtigen Verläufe der 2D-Projektionen von 3D-Punkten während der Kamerabewegung.

Das Kapitel beginnt mit einer Einführung in die Theorie zur Berechnung von Time-To-Collision. Dabei werden die dafür erforderlichen Grundlagen sowie die fundamentalen Rechenschritte anschaulich erklärt. Der zweite Abschnitt beschäftigt sich mit der Ursache für die sehr instabilen TTC-Werte bei der Anwendung des Standardalgorithmus. Gleichzeitig werden hier auch die bereits bestehenden Arbeiten auf diesem Gebiet vorgestellt. Im darauf folgenden Abschnitt wird dann der neue Ansatz vorgestellt. Hier werden ausgehend von den im zweiten Abschnitt gewonnenen Einsichten über die Ursachen instabiler TTC-Werte Modellgleichungen hergeleitet, mittels derer der theoretisch richtige Verlauf einer Punktsequenz im Bild geschätzt werden kann. Anschließend werden zwei Verfahren zur Adaption der entsprechenden Modellparameter vorgestellt. Das Kapitel schließt ab mit einer Evaluierung beider Adaptionansätze und einer ausführlichen Diskussion über deren Tauglichkeit in praktischen Einsätzen.

### **4.1 Theorie zur Berechnung von Time-To-Collision**

Die Rekonstruktion der Umgebung mittels Time-To-Collision (TTC) basiert auf den Bildinformationen der Kamera und deren Bewegung. Algorithmen zur TTC Berechnung bestimmen dabei die verbleibende Zeit bis zum Zusammenstoß zwischen der Kamera und den bevorstehenden Objekten. Der verbleibende Abstand kann unter Zuhilfenahme der Kamerageschwindigkeit ebenfalls ausgerechnet werden.

Die grundlegenden Berechnungen von Time-To-Collision basieren auf dem optischen Fluss. Daher wird im folgenden zunächst anschaulich erklärt wie dieser entsteht und wie die entsprechenden Informationen interpretiert werden können. Anschließend wird formal das Standardverfahren zur Berechnung von TTC-Werten hergeleitet. Gleichzeitig wird dieses auch etwas für die praktischen Anwendungen adaptiert.

#### **Optischer Fluss**

Alternativ zur 3D Rekonstruktion mittels Stereo Vision kann die Umgebung auch aus der Bildinformation nur einer Kamera rekonstruiert werden. Die entsprechenden Algorithmen findet man in der Fachliteratur unter dem Begriff 'structure from motion'. Wichtige Voraussetzung für diese Ansätze ist das Wissen über die Bewegung der Kamera. Aufgrund der Bewegung des Roboters und damit der Kamera ändert sich die Position der Hindernisobjekte im Kamerabild. Die Beträge der resultierenden Änderungen liefern dabei Hinweise auf die Entfernung des jeweiligen Objektes.

Den optischen Effekt, der hierbei oft ausgenutzt wird, bezeichnet man in der Wahrnehmungspsychologie als Bewegungsparallaxe. Dieser Effekt ergibt sich, wenn

man beispielsweise aus dem Seitenfenster eines fahrenden Autos oder Zugs schaut. Diejenigen Objekte, die nahe beim Betrachter sind, bewegen sich scheinbar schneller als die weiter entfernten Objekte.

Ähnliche Beobachtungen werden auch gemacht, wenn die Kamera nicht seitlich sondern in Fahrtrichtung des Roboters gerichtet ist. Die resultierenden Verschiebungsvektoren sind in diesem Fall nicht mehr parallel zu einander, sondern fließen radial von einem bestimmten Punkt aus in alle Richtungen. Dieser Punkt wird in der Literatur als 'focus of expansion' (FOE) bezeichnet, und repräsentiert die Fahrtrichtung des Roboters im Kamerabild. Abbildung 4.1 zeigt graphisch wie der eben beschriebener optischer Fluss entsteht.

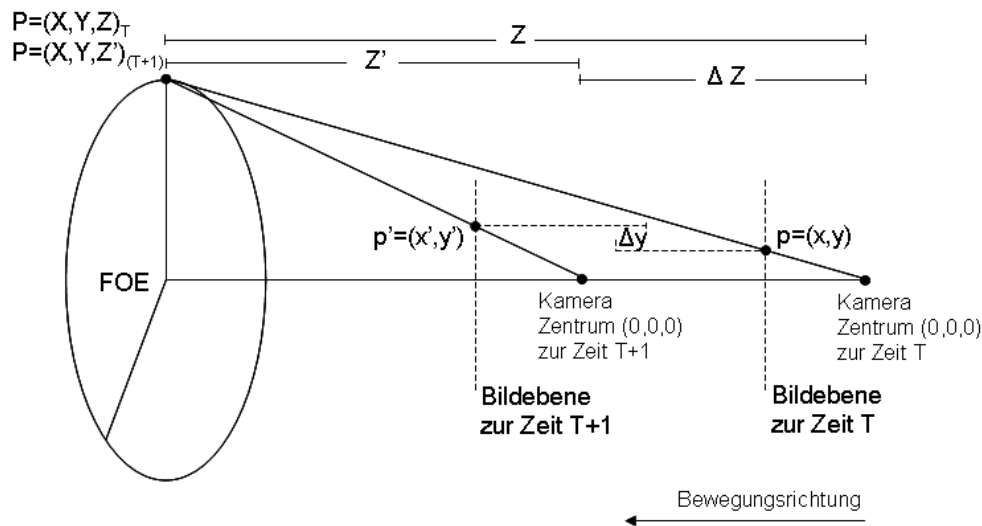


Abbildung 4.1: Graphische Darstellung zur Entstehung des optischen Flusses.

Zum Zeitpunkt  $T$  hat der 3D Punkt  $P$  im Kamerakoordinatensystem die Koordinaten  $P = (X, Y, Z)$ . Die  $Z$ -Achse des Koordinatensystems schneidet den in der Abbildung 4.1 eingezeichneten FOE Punkt. Unter der Annahme eines Lochkamera (pinhole camera) Modells wird der Punkt  $P$  auf den Bildpunkt  $p = (x, y)$  der Bildebene der Kamera zum Zeitpunkt  $T$  abgebildet. Der Abstand der Bildebene zum Koordinatenursprung hängt von der Brennweite (focal length)  $f$  des Kameramodells ab. Weil der genaue Wert dieser Größe im Weiteren nicht von Bedeutung ist, wurde hier zur Vereinfachung  $f = 1$  gesetzt.

Wenn die Kamera nun in Richtung des FOE bewegt wird, ändert sich der Abstand des 3D Punktes  $P$  zum Koordinatenursprung. Wie der obigen Abbildung 4.1 zu entnehmen ist, hat der Punkt  $P$  zum Zeitpunkt  $T + 1$  die Koordinaten  $P = (X, Y, Z')$ . Entsprechend ändert sich auch die Position des projizierten Punktes  $p' = (x', y')$ . Dieser wandert von dem projizierten FOE, welches in den beiden Bil-

ebenen gleich ist, nach außen. Durch solche Verschiebungen nach außen entstehen auch die schon erwähnten Verschiebungsvektoren, die radial von dem FOE in alle Richtungen fließen. Abbildung 4.2 zeigt beispielhaft den optischen Fluss, der bei einer Bewegung der Kamera in Richtung des FOEs entsteht.

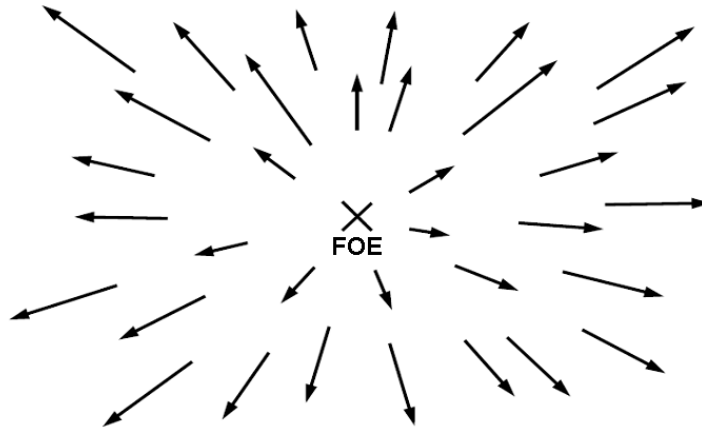


Abbildung 4.2: Optischer Fluss: die durch Pfeile gekennzeichneten Verschiebungsvektoren bilden den Optischen Fluss. Der hier dargestellte Fluss entsteht bei einer geraden Bewegung der Kamera in Richtung des FOEs.

Bei anderen Bewegungsarten werden entsprechend andere Muster des optischen Flusses beobachtet. Aus der Länge und der relativen Position des Vektors zum FOE lässt sich ableiten wie weit das entsprechende Objekt von der Bildebene der Kamera entfernt ist. Die dafür benötigte Berechnungsvorschrift wird im nächsten Abschnitt hergeleitet.

### Berechnung von Time-To-Collision

1976 formulierte Lee [Lee76] einen Algorithmus, der den optischen Fluss zur Berechnung der noch verbleibenden Zeit bis zum Aufprall ausnutzt. Die dabei berechnete Größe  $\tau$  (tau) gibt an wie lange die Kamera mit der aktuellen Geschwindigkeit in die bereits eingeschlagene Richtung bewegt werden muss, bis man damit auf ein bevorstehendes Hindernis trifft. Die folgende Herleitung der Berechnungsmethode basiert auf der von Camus [Cam95].

Sei  $P = (X, Y, Z)$  entsprechend der Abbildung 4.1 ein 3D-Punkt im Kamerakoordinatensystem. Unter der Annahme eines Lochkamera-Modells können die Bildkoordinaten  $p = (x, y)$  des Punktes  $P$  wie folgt berechnet werden:

$$x = f \frac{X}{Z} = \frac{X}{Z} \quad \text{und} \quad y = f \frac{Y}{Z} = \frac{Y}{Z} \quad \text{mit } f = 1$$

Mit der Bewegung der Kamera ändern sich auch die Koordinaten des projizierten Punktes  $p$ . Der Betrag der Änderung hängt sowohl von der Zeit zwischen zwei aufgenommenen Bildern ab als auch von der 3D-Position des Punktes  $P$ . Mittels der Quotientenregel gilt (die gleichen Überlegungen gelten auch für die  $x$ -Koordinate des Punktes  $p$ ):

$$\frac{dy(t)}{dt} = \frac{\frac{dY(t)}{dt} \cdot Z(t) - Y(t) \cdot \frac{dZ(t)}{dt}}{Z^2(t)}$$

Die Bestandteile  $\frac{dY(t)}{dt}$  und  $\frac{dZ(t)}{dt}$  bezeichnen hierbei die zeitliche Koordinatenänderung des 3D-Punktes  $P$ . Wie der Abbildung 4.1 zu entnehmen ist, ändert sich während der Kamerabewegung der Koordinatenursprung der Kamera und damit auch die  $Z$ -Koordinate des Punktes  $P$ . Die  $X$ - und  $Y$ -Koordinaten bleiben dagegen konstant. Daher gilt  $\frac{dY(t)}{dt} = 0$ .

Einfaches Umformen der obigen Gleichung resultiert in:

$$\frac{dy(t)}{dt} = \frac{\frac{dY(t)}{dt}}{Z(t)} - Y(t) \left( \frac{\frac{dZ(t)}{dt}}{Z^2(t)} \right)$$

Da  $\frac{dY(t)}{dt} = 0$  und  $Y = y \cdot Z$  gilt:

$$\frac{dy(t)}{dt} = -y(t) \left( \frac{\frac{dZ(t)}{dt}}{Z(t)} \right)$$

Durch Umformen folgt:

$$\frac{y(t)}{\frac{dy(t)}{dt}} = -\frac{Z(t)}{\frac{dZ(t)}{dt}} = \tau \quad (4.1)$$

Die so berechnete Größe  $\tau$  bezeichnet man in der Literatur als 'time-to-contact' oder 'time-to-collision' (TTC). Sie gibt die Zeit an, die noch verbleibt bis die auf dem Roboter montierte Kamera mit einem bevorstehendem Hindernis kollidiert. Der Wert dieser Größe kann ausschließlich aus der Bildinformation berechnet werden. Der linke Teil der Gleichung 4.1 gibt dazu eine einfache Anleitung.

Ist die Kamera in Fahrtrichtung des Roboters gerichtet, so können die  $\tau$ -Werte wie folgt berechnet werden:

1. Berechne den FOE aus der Bildsequenz der Kamera.
2. Finde in zwei aufeinander folgenden Bildern der Sequenz korrespondierende Punkte. Für diese Punkte werden die TTC-Werte berechnet.
3. Berechne die Länge der Verschiebungsvektoren, welche durch je zwei korrespondierende Punkte gebildet werden. Die Länge dieser Vektoren stellt eine

Schätzung für die Divergenz des jeweiligen Bildpunktes von dem FOE zum aktuellen Zeitpunkt dar.

4. Berechne den Abstand der Punkte zu dem FOE.
5. Für jeden Bildpunkt  $p_i$  berechne den TTC Wert nach der folgenden Formel:

$$\tau_i = \frac{\text{Abstand des Punktes von dem FOE}}{\text{Länge des Verschiebungsvektors}}$$

Für die Berechnung der Länge der Verschiebungsvektoren und für den Abstand des Punktes zum FOE wurde im Rahmen dieser Arbeit die  $L^2$ -Norm verwendet.

Wie der obigen Gleichung zu entnehmen ist, kann für den Punkt im FOE kein TTC Wert berechnet werden. In diesen Fall wären sowohl der Abstand des Punktes zum FOE als auch die Länge des entsprechenden Verschiebungsvektors Null. Für Punkte nahe dem FOE sind die Ergebnisse in den praktischen Anwendungen üblicherweise sehr instabil. Störungen in den Messungen der Punktpositionen haben hier einen viel größeren Einfluss auf die Berechnung der TTC-Werte. Gleiche Messfehler bei Punktpositionen, die im Bild weiter entfernt vom FOE sind, wirken sich stattdessen weniger stark auf die Güte der Werte aus. Diese Beobachtungen können schon an den Ergebnissen des nachfolgenden Abschnitts gemacht werden.

Der FOE wird üblicherweise als der Schnittpunkt der Verschiebungsvektoren berechnet. Dieses Vorgehen ist allerdings sehr instabil und funktioniert nur wenn die Position der Punkte im Bild exakt gemessen werden kann. Dies ist in der Praxis oft nicht der Fall.

Es ist an dieser Stelle anzumerken, dass der nach dem obigen Algorithmus berechnete TTC Wert nicht die verbleibende Zeit in Sekunden angibt, sondern viel mehr die Anzahl an Bildern, die unter den gleich bleibenden Systemvoraussetzungen noch bis zu Aufprall aufgenommen werden können. Um also die verbleibende Distanz zu den Hindernissen auszurechnen, reicht es nicht den berechneten TTC Wert mit der Geschwindigkeit des Roboters zu multiplizieren. Die Zeit, die zwischen zwei aufeinander folgenden Bildaufnahmen verstreicht, muss ebenfalls berücksichtigt werden. Die folgende Formel fasst dies zusammen:

$$A_i = V \cdot \tau_i \cdot t_{(m,m+1)}$$

$A_i$  bezeichnet hier den Abstand zu dem Objekt  $i$ ,  $V$  steht für die aktuelle Geschwindigkeit der Kamerabewegung und  $t_{(m,m+1)}$  ist die Zeit, die zwischen zwei aufeinander folgenden Bildaufnahmen  $m$  und  $m + 1$  verstreicht.

## 4.2 Darstellung der Problematik

Wie bereits zu Anfang des Kapitels erwähnt, hängt die Qualität der berechneten TTC-Werte entscheidend von der Messgenauigkeit der entsprechenden Punktpositionen im Bild ab. Schon kleine Abweichungen in den Messungen können zu stark schwankenden TTC-Werten führen. Um das Ausmaß dieses Problems besser demonstrieren zu können, wurde im Rahmen dieser Arbeit eine Simulationsumgebung programmiert. Die projizierten Bildpunkte wurden in der Simulation zunächst unterschiedlich stark verrauscht und dann die auf diesen Daten berechneten TTC-Werte mit den richtigen Werten verglichen. Die dabei erzielten Ergebnisse werden im Folgenden vorgestellt. Gleichzeitig werden auch die Ursachen für solch instabile TTC-Werte diskutiert. Der vorliegende Abschnitt schließt ab, mit einer kurzen Zusammenfassung über die bisherigen Arbeiten zur robusten Berechnung von Time-To-Collision.

### 4.2.1 Leistungsfähigkeit des Standardverfahrens

Um die Leistungsfähigkeit des Standardverfahrens unter kontrollierten Bedingungen testen zu können, wurde im Rahmen dieser Arbeit ein Simulationsprogramm geschrieben. Damit war es möglich in einem 3D-Weltkoordinatensystem exakt die relative Position und Orientierung der Kamera zu den bereits vorher spezifizierten 3D-Punkten zu bestimmen. Die Kamera wurde dabei mittels der intrinsischen und extrinsischen Kameramatrizen modelliert. Zur Bestimmung des optischen Flusses wurde die Kamera immer in jeweils einem Zeitintervall eine Streckeneinheit nach vorne bewegt. Dabei wurde vor jeder Verschiebung der Kamera die 3D- Weltpunkte auf die Bildebene der Kamera projiziert.

Im Rahmen des hier durchgeführten Experimentes wurde unter der Annahme eines Lochkamera-Modells auf einem Rechteck platzierte 3D-Punkte auf die Bildebene projiziert. Das Rechteck wurde parallel zur Bildebene und einer Distanz von 58.0 Einheiten zum Ursprung des Kamerakoordinatensystems aufgestellt. Die Höhe und Breite des Rechtecks betragen jeweils 2600.0 und 2400.0 Einheiten. Der FOE lag im Mittelpunkt des Rechtecks. Abbildung 4.3 (a) zeigt den generierten optischen Fluss. Dabei wurde die Kamera insgesamt 38 mal um eine Einheit in Richtung des Rechtecks bewegt. Abbildung 4.3 (b) zeigt die wahren und die berechneten TTC-Werte.

Weil in der Simulation die Kamera dem Rechteck mit einer konstanten Geschwindigkeit angenähert wurde, sind die TTC-Werte monoton fallend und liegen alle auf einer Geraden. Die Steigung der Geraden hängt von der Geschwindigkeit der Annäherung ab. Wie der Abbildung (b) zu entnehmen ist, stimmten die berechneten und

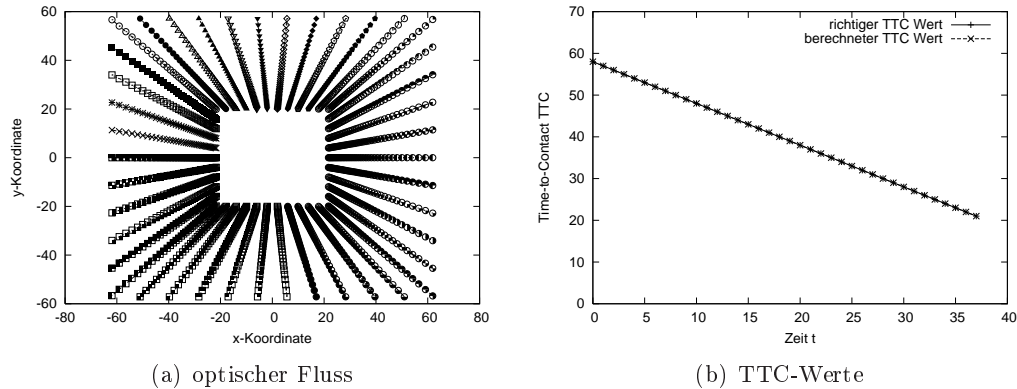


Abbildung 4.3: Simulationsergebnisse mit exakten Daten: (a) zeigt den generierten optischen Fluss; (b) zeigt die richtigen und die berechneten TTC-Werte

die wahren TTC-Werte überein. Dies liegt daran, dass die projizierten Bildpunkte in diesem Experiment exakt ausgemessen wurden. Für alle in Abbildung (a) gezeigten Sequenzen wurden die gleichen TTC-Werte berechnet.

Es ergibt sich allerdings ein anderes Bild sobald man etwas Rauschen zu den gemessenen Bildpunkten hinzufügt. Das Rauschen wurde in der Simulation mittels einer zweidimensionalen Gauß-Verteilung realisiert. Unter der Annahme der statistischen Unabhängigkeit der Streuungen in  $x$ - und  $y$ -Richtung wurde dafür die folgende Formel verwendet:

$$f(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\left(\frac{x - \mu_x}{\sigma_x}\right)^2 - \left(\frac{y - \mu_y}{\sigma_y}\right)^2\right)$$

$\sigma_x$  und  $\sigma_y$  bezeichnen hierbei die Varianz der Streuung in  $x$ - und  $y$ -Richtung. Im Folgenden wird  $\sigma_x$  und  $\sigma_y$  immer der gleiche Wert zugewiesen. Der Mittelwert der Verteilung  $\mu_x$  und  $\mu_y$  wurde immer mit den exakten Positionsdaten des jeweiligen Punktes initialisiert.

Die drei Graphiken in der linken Spalte der Abbildung 4.4 zeigen ein paar ver-rauschte Punktsequenzen. Die Varianz der Streuung wurde beispielhaft auf jeweils 0.1, 0.3 und 0.75 gesetzt. Die rechte Spalte zeigt die entsprechenden TTC-Werte. Schon bei einer Varianz von 0.1 weichen die berechneten TTC-Werte entscheidend von den richtigen ab. Erst gegen Ende der Sequenz lässt sich hier eine leichte An-näherung an die wahren Werte ausmachen. Wie bereits im vorherigen Abschnitt erklärt, haben die Messfehler bei Punktpositionen weiter vom FOE weniger Einfluss auf die Güte der TTC-Werte. Bei höheren Streuungen kann diese Beobachtung al-lerdings nicht mehr gemacht werden. Mehr noch, an keinem Punkt der berechneten



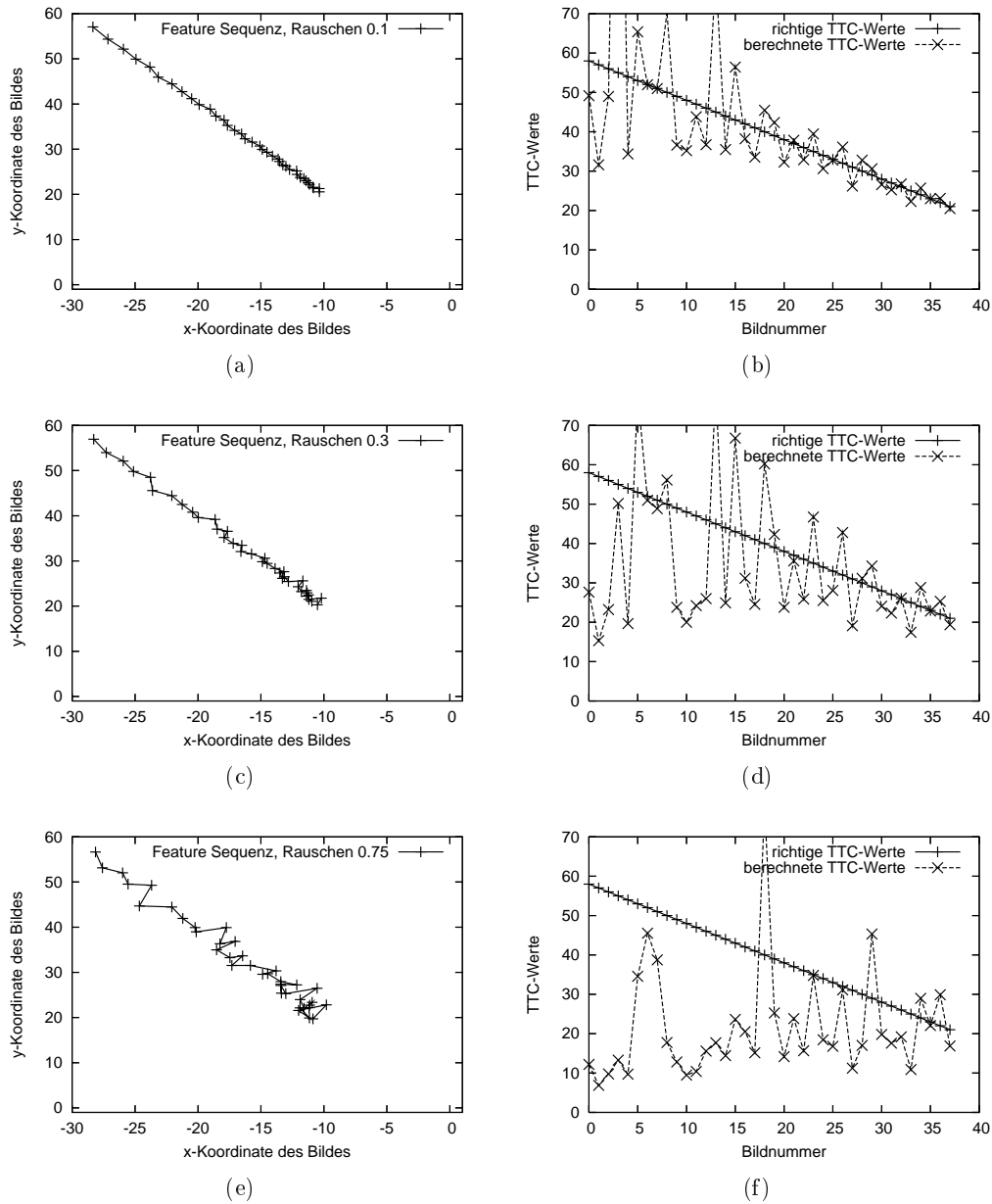


Abbildung 4.4: Linke Spalte zeigt drei in der Simulation verrauschte Punktsequenzen. Dabei wurde jeweils unterschiedliche Varianz der Streuung gewählt: 0.1, 0.3 und 0.75. Die rechte Spalte zeigt die entsprechenden TTC-Werte.

TTC Sequenzen lässt sich die weitere Entwicklung der Werte im Voraus erahnen. Im Unterschied zu der glatten, monoton fallenden Folge der richtigen TTC-Werte, springen die berechneten TTC-Werte in einem Intervall von bis zu vierzig Einheiten.

#### 4.2.2 Ursache für schlechte TTC-Werte

Die erste, nächstliegende Vermutung für die Ursache solch stark schwankender TTC-Werte war, dass die verrauschten Messpunkte nicht wie im Idealfall eine Gerade bilden. Weitere Experimente haben allerdings gezeigt, dass die Abweichungen der Punkten von der Geraden nicht den größten Einfluss auf die Schwankungen der TTC-Werte haben. Zur Testzwecken wurden in der Simulation die verrauschten Bildpunkte auf die jeweilige Gerade projiziert und die TTC-Werte auf den neuen Punktpositionen berechnet. Dies brachte allerdings nicht den erhofften Erfolg. In seltenen Fällen konnten zwar weniger Schwankungen bei den berechneten TTC-Werten beobachtet werden, doch dies war eher dem Zufall zuzurechnen.

Weitere Experimente haben dann allerdings gezeigt, dass die berechneten TTC-Werte nur dann eine glatte Gerade bilden, wenn der euklidische Abstand zwischen zwei aufeinander folgenden Bildpunkten einer Sequenz stetig zunimmt. Abbildung 4.5 verdeutlicht diesen Sachverhalt. Die durchgezogene Linie in der linken Graphik

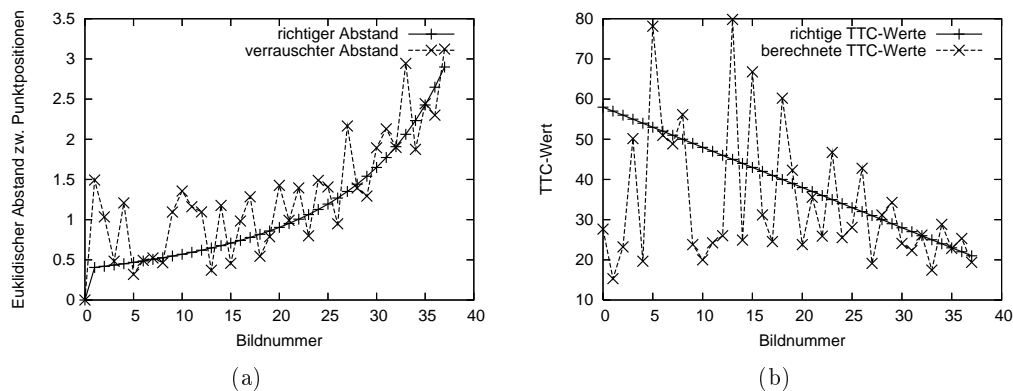


Abbildung 4.5: (a) zeigt die Abstände zwischen zwei Bildpunkten einer exakten und einer verrauschten Sequenz. (b) zeigt die entsprechenden TTC-Werte.

der Abbildung zeigt wie der euklidischen Abstand zwischen zwei aufeinander folgenden Bildpunkten einer idealen Sequenz mit jedem weiteren Bild stetig zunimmt. Die Krümmung des eingezeichneten Verlaufs hängt dabei von der 3D-Position des projizierten Bildpunktes ab. Die gestrichelte Linie in der Graphik zeigt zusätzlich noch den euklidischen Abstand zwischen zwei aufeinander folgenden Bildpunkten einer in der Simulation verrauschten Sequenz. Die exakten Punktpositionen wurden

hierbei mit dem Gaußschen Rauschen der Varianz 0.3 verfälscht. Die dabei resultierenden Abstände bilden keine glatte Kurve mehr. Ähnliche Entwicklung spiegelt sich auch in den berechneten TTC-Werten wieder. Die durchgezogene Linie in der rechten Graphik der Abbildung zeigt die richtigen TTC-Werte. Diese wurde auf den exakten Bildpunkten berechnet, deren euklidischer Abstand kontinuierlich anstieg. Die gestrichelte Linie zeigt dagegen TTC-Werte, welche auf den verrauschten Bildpunkten berechnet wurden.

Dieses Ergebnis erklärt auch warum die erste Vermutung sich als falsch erwiesen hat. Die Projektion der verrauschten Bildpunkte auf die Gerade, korrigierte nicht die Abstände zwischen zwei Bildpunkten. Entsprechend schlecht waren auch die resultierende TTC-Werte.

### 4.2.3 Existierende Ansätze zur TTC Berechnung

In der Praxis werden oft ähnlich schlechte Ergebnisse beobachtet. Souhila et al. [SK07] implementierte basierend auf den TTC-Werten ein System zur Hinderniserkennung und -umgehung. Der dazu benötigte optische Fluss wurde mittels der differentiellen Methode von Horn und Schunck [HS81] berechnet. Die Methode liefert einen dichten optischen Fluss, reagiert aber sehr sensibel auf Beleuchtungsschwankungen. Änderungen der Lichtverhältnisse haben einen starken Einfluss auf die Fehlerrate der Messungen und damit auf die Schätzung des optischen Flusses. Die berechneten TTC-Werte schwankten ähnlich denen in Abbildung 4.4. Unter kontrollierten Laborbedingungen konnte der Roboter sicher zwei Hindernisse umfahren. Die Autoren weisen aber darauf hin, dass das Verfahren sehr instabil ist und schlagen vor die Bildmessungen noch mit weiteren Ultraschall- oder Infrarot-Sensoren zu kombinieren.

Mittlerweile lassen sich die Ansätze zur Berechnung von TTC-Werten in zwei Gruppen einteilen. Die erste Gruppe konzentriert sich auf die stabile und fehlerfreie Berechnung des optischen Flusses. Die Ansätze der zweiten Gruppe versuchen dagegen mit den verrauschten Messdaten zu arbeiten.

Existierende Verfahren zur Berechnung des dichten optischen Flusses lassen sich in differentielle Methoden [PCF05] [Nag87] [Nag88] [Nag90], korrelationsbasierte Methoden [BBH<sup>+</sup>89] [BBHP90] [BT80] und verschiedene Variationsmethoden [BBPW04] [BWS05] [ZPB07] einteilen. Die meisten dieser Ansätze funktionieren nur unter bestimmten Annahmen. So wird zum Beispiel oft vorausgesetzt, dass die Lichtverhältnisse in zwei aufeinander folgenden Bildern unverändert bleiben. Die Verfahren liefern auch schlechte Ergebnisse wenn die Verschiebung des Objektes im Bild oder der Kamera zu groß ist.

## **82 Kapitel 4. Entfernungsschätzung bei geradliniger Kamerabewegung**

Eine Mögliche Lösung für diese Probleme bilden Verfahren, die nur markante und einfache zu zuordnende Merkmalspunkte in Bildern verfolgen. Diese Verfahren liefern weit bessere Ergebnisse bei sich ändernden Lichtverhältnissen. Der dabei produzierte optischer Fluss ist allerdings eher spärlich. In der Literatur oft verwendete Verfahren zur Merkmalsextraktion sind Harris-Punkte-Detektor [HS88], KLT-Detektor [ST94] oder SIFT-Detektor [Low04].

In realen Anwendungen kann die Genauigkeit der Messung oft nicht garantiert werden. Unabhängig davon welches Verfahren man zur Berechnung des optischen Flusses verwendet, können die Messdaten aufgrund weiterer Fehlerquellen trotzdem verfälscht werden. So kann die Abweichung zwischen der gemessenen und der theoretisch exakten Position des Punktes im Kamerabild aufgrund der verwendeten Kameraauflösung stark verfälscht werden. Je niedriger die Auflösung desto ungenauer ist die Messung. Auch die Trajektorie der Kamerabewegung kann aufgrund der Faktoren wie Bodenbeschaffenheit oder Art der Kameraanbringung von der angenommenen oder auch von der gemessenen Trajektorie abweichen. Ist der Boden nur etwas uneben, so stimmt die Annahme der exakten geradlinigen Kamerabewegung nicht mehr. Im Gegenteil, es entstehen Vibrationen, die sich sehr stark auf die Messungen im Kamerabild auswirken. Unabhängig von dem Verfahren sind in diesem Fall die Trajektorien der über mehrere Bilder verfolgten Merkmale keine geraden Linien, sondern eher Kurven deren Entwicklung nur grob vorhergesagt werden kann.

In vielen Anwendungen ist es manchmal sehr schwer oder gar unmöglich solche Fehlerquellen vollständig und nachhaltig zu eliminieren. Es besteht daher großes Interesse an Verfahren, die trotz der verrauschten Bildpunkte robuste und vertrauenswürdige TTC-Werte berechnen.

Coombs et. al präsentierte in [CHHN95] ein Verfahren zur autonomen Roboternavigation welches auf der Auswertung der Kamerabilder basiert. Die Entfernungsschätzung zu den bevorstehenden Hindernissen erfolgt mittels der Berechnung von TTC-Werten. Ausgehend von einer konstanten und geraden Bewegung des Roboters konstruieren Coombs et. al. ein lineares Modell welches die Entwicklung der resultierenden TTC-Werte beschreibt. Die Adaption der Modellparameter erfolgt mittels der Minimierung einer Fehlerfunktion, welche in sich auch die zuvor berechneten TTC-Werte beinhaltet. Als Minimierungsverfahren wird das Square Root Information Filter (SRIF) verwendet.

Die Performanz dieses Verfahrens hängt entscheidend von der Güte und der Anzahl der zuvor berechneten TTC-Werten ab. Schwanken diese, so sind auch die berechneten Modellparameter wertlos. Ein weiteren schwerlastender Nachteil dieses Verfahrens ist seine Nichterweiterbarkeit auf andere Bewegungsarten.

Horn et.al. präsentierte 2007 in [HYM07] ein so genannten 'direktes' Verfahren zur Berechnung von TTC-Werten. Die Bestimmung der Werte basiert nur auf der bis zum aktuellen Zeitpunkt akkumulierten Summe von geeigneten Produkten der Gradienten der Bildhelligkeiten. Die damit berechneten Werte sind zu mindestens zu Anfang der Sequenz relativ stabil. Die Autoren berichten allerdings von zunehmenden Schwankungen gegen Ende der entsprechenden TTC-Sequenz. Als Grund nennen sie die immer größer werdende Verschiebungen von Objekten im Bild.

Einen anderen Ansatz verfolgten dagegen Müller et. al. In [MPNMS09] schlagen sie vor, TTC-Werte ausgehend von den Skalierungsänderungen der bevorstehenden Objekte auszurechnen. Die in den letzten 20 Bildern akkumulierten Änderungen der Objektgrößen dienen als Input für insgesamt drei zuvor definierte Bewegungsmodelle. Die endgültige TTC-Schätzung erfolgt ausgehend davon, welches der drei Modelle am besten zu den gemachten Bildmessungen passt.

Die Bewegungsmodelle basieren dabei nicht auf der eigentlichen Kamerabewegung, sondern beschreiben eher die zeitliche Entwicklung der Kameraposition relativ zum bevorstehenden Objekt. Dabei wird eine mögliche Rotation der Kamera komplett vernachlässigt. Auch bei diesem Ansatz würde sich eine Erweiterung auf eine rotierende Kamerabewegung als sehr schwierig erweisen.

Der folgende Abschnitt stellt einen neuen Ansatz zur Berechnung der TTC-Werte vor. Dabei wird zunächst von einer konstanten und geraden Bewegung der Kamera ausgegangen. In den späteren Kapiteln werden dann Erweiterung auf unterschiedliche Bewegungsarten vorgestellt.

## 4.3 Bewegungsmodell basierte TTC Berechnung

Die Identifikation der Hauptursache für schlechte TTC-Werte lieferte entscheidende Hinweise auf die Entwicklung eines neuen Algorithmuses zu Berechnung von TTC-Werten. Ausgehend von den gewonnenen Einsichten wurden im Rahmen dieser Arbeit Modellgleichungen hergeleitet, mittels derer der theoretisch richtige Verlauf einer Punktsequenz im Bild geschätzt werden kann. Im Folgenden werden die einzelnen Schritte zur Konstruktion der Gleichungen anschaulich erklärt. Anschließend werden zwei Verfahren zur iterativen Schätzung der entsprechenden Modellparameter vorgestellt. Beide Verfahren verwenden als Input die ungenauen Messdaten.

### 4.3.1 Konstruktion der Modellgleichungen

Die Grundidee bei der Entwicklung des Algorithmuses war das Aufstellen eines Systems, welches die ungenauen Positionen der verfolgten Bildpunkte entsprechend der

## 84 Kapitel 4. Entfernungsschätzung bei geradliniger Kamerabewegung

geforderten Abständen zwischen denen korrigiert. Dazu wurden im Rahmen dieser Arbeit Gleichungen hergeleitet, die den richtigen Verlauf einer Punktsequenz im Bild modellieren. Für die Korrektur der Punktpositionen wurden dabei die Modellparameter zunächst basierend auf den verrauschten Messdaten der jeweiligen Sequenz angepasst. Mit den korrigierten Punktpositionen war es dann möglich die üblichen Schwankungen bei den TTC-Werten zu unterdrücken.

Im Folgenden werden die Modellgleichungen hergeleitet. Zum besseren Verständnis der Herleitung ist in der Abbildung 4.6 graphisch der Entstehungsprozess einer Punktfolge gezeigt. Die Vorwärtsbewegung der Kamera wurde hier durch die Bewe-

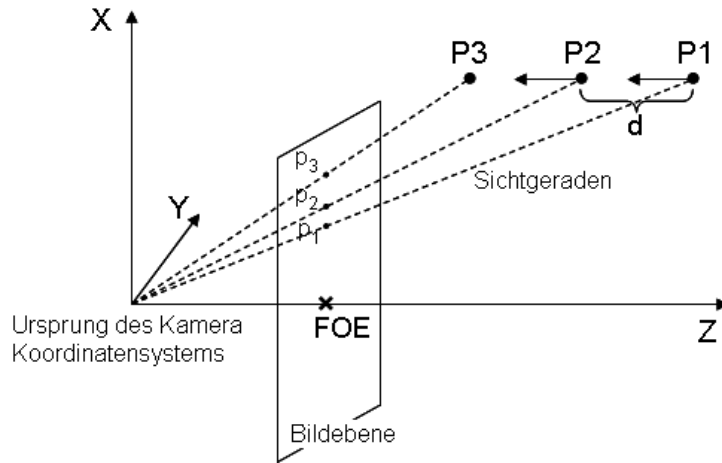


Abbildung 4.6: Entstehungsprozess einer Bildpunkt Folge.

gung des 3D-Punktes  $P$  gegen die Bildebene simuliert. Ähnlich wie in Abbildung 4.1 änderte sich bei der Bewegung des Punktes  $P$  nur die  $Z$ -Koordinate. Zu den Zeitpunkten  $t_1$ ,  $t_2$  und  $t_3$  hat der Punkt  $P$  die folgenden Koordinaten  $P1 = (X, Y, Z)$ ,  $P2 = (X, Y, Z - d)$  und  $P3 = (X, Y, Z - d - d)$ . Der Parameter  $d$  bezeichnet hierbei den zurückgelegten Weg zwischen zwei Zeitpunkten.

Unter der Annahme eines Lochkamera-Modells und der Brennweite  $f = 1$ , kann die Position des projizierten Bildpunktes mit dem 1. Strahlensatz bestimmt werden.

$$\begin{aligned} p_{t1} &= (x_1, y_1) = \left( \frac{X}{Z}, \frac{Y}{Z} \right) \\ p_{t2} &= (x_2, y_2) = \left( \frac{X}{(Z-d)}, \frac{Y}{(Z-d)} \right) \\ p_{t3} &= (x_3, y_3) = \left( \frac{X}{(Z-d-d)}, \frac{Y}{(Z-d-d)} \right) \end{aligned}$$

Betrachtet man nur die  $x$ -Koordinate des projizierten Punktes  $p$ , so kann die  $X$ -Koordinate des 3D-Punktes  $P$  wie folgt ausgedrückt werden (ähnliche Überlegungen

gelten auch für die  $Y$ -Koordinate).

$$X = x_1 \cdot Z \quad (4.2)$$

$$X = x_2 \cdot Z - x_2 \cdot d \quad (4.3)$$

$$X = x_3 \cdot Z - x_3 \cdot d - x_3 \cdot d \quad (4.4)$$

Gleichsetzen der Gleichungen 4.2 und 4.3 und Auflösen nach  $Z$  resultiert in:

$$Z = \frac{x_2 \cdot d}{(x_2 - x_1)} \quad (4.5)$$

Einsetzen der Gleichungen 4.2 und 4.5 in Gleichung 4.4 führt zu:

$$x_1 \left( \frac{x_2 \cdot d}{(x_2 - x_1)} \right) = x_3 \left( \frac{x_2 \cdot d}{(x_2 - x_1)} \right) - x_3 \cdot d - x_3 \cdot d$$

Weiteres Umformen resultiert in:

$$d \left( \frac{x_1 \cdot x_2}{(x_2 - x_1)} \right) = d \left( \frac{x_3 \cdot x_2}{(x_2 - x_1)} - x_3 - x_3 \right)$$

Der in der Praxis nicht bekannt Abstand  $d$  kann nun aus der Gleichung gekürzt werden. Weiteres Umformen führt dann letzten Endes zu der folgenden Gleichung:

$$x_3 = \frac{x_1 \cdot x_2}{2 \cdot x_1 - 1 \cdot x_2}$$

Damit kann die Position des projizierten Punktes  $p$  zum Zeitpunkt  $t_3$  vorhergesagt werden, wenn die Positionen dieses Bildpunktes zu den Zeitpunkten  $t_1$  und  $t_2$  bekannt sind. Mehr noch, mit der vollständigen Induktion lässt sich zeigen, dass die obige Gleichung für beliebiges  $n \in \mathbb{N}_{>2}$  gilt:

$$x(n) = \frac{x_1 \cdot x_2}{((n-1) \cdot x_1 - (n-2) \cdot x_2)} \quad (4.6)$$

Ähnliches gilt auch für die  $y$ -Koordinate des Bildpunktes:

$$y(n) = \frac{y_1 \cdot y_2}{((n-1) \cdot y_1 - (n-2) \cdot y_2)} \quad (4.7)$$

Sind also die ersten beiden Punktpositionen einer Punktfolge bekannt, so können theoretisch die nachfolgenden Positionen mittels der obigen Gleichungen vorhergesagt werden. Mehr noch, mit den vorhergesagten Punktpositionen im Bild können auch die zukünftigen 3D-Positionen des abgebildeten Punktes berechnet werden.

Um zu überprüfen, ob die hergeleiteten Gleichungen die entsprechenden Sequenzen richtig modellieren und ob die vorhergesagten Punktpositionen den richtigen Positionen entsprechen, wurden in der Simulation mehrere Experimente durchgeführt. Abbildung 4.7 zeigt dazu die Ergebnisse eines solchen Experiments. Die linke Graphik zeigt die euklidischen Abstände zwischen zwei aufeinander folgenden Punkten einer, in der Simulation erzeugten Sequenz und einer Folge, die mit den obigen Modellgleichungen konstruiert wurde. Die Parameter der Gleichungen wurden dabei mit den Werten der ersten beiden Punktpositionen der simulierten Sequenz initialisiert. Die durchgezogene Linie mit Kreisen basiert auf den Werten der in der Simulation

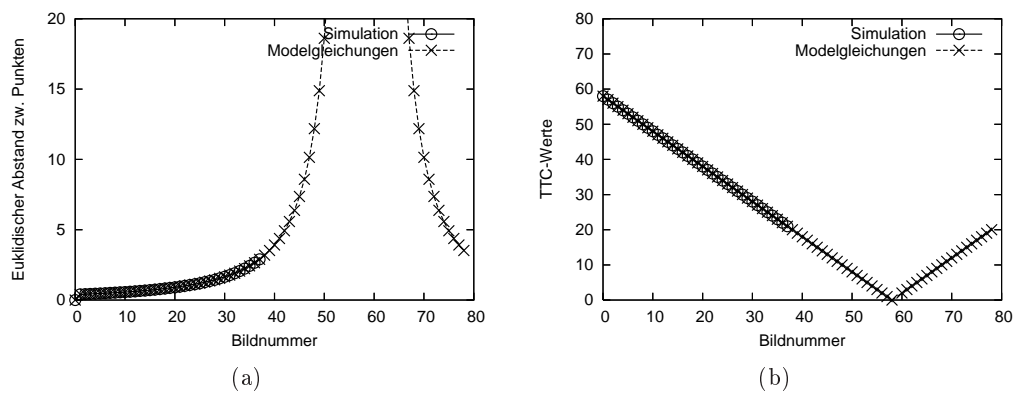


Abbildung 4.7: (a) zeigt die Abstände zwischen den Punkten der in der Simulation erzeugten Punktsequenz und der mittels der Gleichungen 4.6 4.7 konstruierten Folge. (b) zeigt die entsprechenden TTC-Werte.

erzeugten Punktsequenz. Die Positionen der Punkte dieser Sequenz wurden mittels der Projektion des zuvor definierten 3D-Punktes auf die Bildebene bestimmt. Dazu wurden die in der Simulation ebenfalls bekannten extrinsischen und intrinsischen Kameramatrizen verwendet. Wie erwartet, steigen die Abstände stetig an. Die Kurve hört allerdings bei der Bildnummer 38 auf. In der Simulation wurde die Kamera insgesamt 40 mal um eine Streckeneinheit nach vorne verschoben. Die Punktpositionen in den ersten beiden Bildern wurden dabei zur Initialisierung der Modellgleichungen verwendet. Daher wurden diese beim Vergleich der Ergebnisse in dem obigen Experiment nicht weiter berücksichtigt. Der initiale Abstand zum Objekt betrug damit nur 58 Einheiten.

Wie der rechten Graphik der Abbildung zu entnehmen ist, stimmt dieser Wert auch mit dem initialen TTC-Wert überein. Dies liegt daran, dass in dem durchgeführten Experiment die Kamera mit der Geschwindigkeit  $v = \frac{1 \text{ Strecken\_Einheit}}{1 \text{ Zeit\_Einheit}}$  bewegt wurde und zwischen zwei Bildaufnahmen nur eine Streckeneinheit zurückgelegt wurde. Die durchgezogene Linie mit Kreisen fällt in dieser Graphik monoton



runter und hört, wie erwartet bei der Bildnummer 38 auf.

Die gestrichelte Linie mit Kreuzen repräsentiert in beiden Abbildungen Werte, die auf der mit den obigen Modellgleichungen konstruierten Folge basieren. Insgesamt wurden 80 Punkte der Folge berechnet. Wie der Abbildung (a) zu entnehmen ist, stimmen die Abstände zwischen den ersten 38 Punkten mit denen der simulierten Sequenz überein. Die gestrichelte Linie wird allerdings weiter fortgesetzt. Die Abstände zwischen den Punkten steigen weiter an. Der maximale Punkt dieser Kurve wurde bei der Bildnummer 58 erreicht. Dies ist genau der Zeitpunkt, wenn der durch die Folge simulierte 3D-Punkt die XY-Ebene des Kamerakoordinatensystems schneidet. Danach verringern sich die Abstände zwischen den Punkten, und zwar genau so schnell wie sie angestiegen sind. Dies liegt daran, dass der durch die Folge simulierte 3D-Punkt sich nun von der Kamera nach hinten wegbewegt. Der gleiche Sachverhalt spiegelt sich auch in der Abbildung (b) wieder. Die gestrichelte Linie mit Kreuzen fällt zunächst monoton von 58 auf die 0 runter. Ab der Bildnummer 58 ändert sie allerdings ihre Richtung und steigt wieder an. Wie bereits erwähnt, bewegt sich der durch die Folge simulierte 3D-Punkt ab der Bildnummer 58 von der Kamera nach hinten weg. Entsprechend steigt auch der berechnete TTC-Wert.

Die beiden Gleichungen 4.6 und 4.7 modellieren sehr gut die Entwicklung einer Punktfolge im Bild, die durch die Projektion eines bestimmten 3D-Punktes entstehen würde. Unabhängig davon wie weit sich der 3D-Punkt vom Ursprung des Kamerakoordinatensystems befindet oder wie schnell die Kamera bewegt wird, entscheidend sind nur die ersten beiden Positionen der projizierten Punkte. Werden diese exakt ausgemessen, so können auch die zukünftigen Positionen des 3D-Punktes berechnet werden. Doch wie bereits erwähnt gestaltet sich das exakte Ausmessen der Punkte als sehr schwierig. In praktischen Anwendungen kann der erforderliche Grad der Messgenauigkeit oft nicht garantiert werden. Daher wurden im Rahmen dieser Arbeit mehrere Verfahren zur Adaption der Modellparameter basierend auf den fehlerhaften Messdaten der jeweiligen Sequenz entwickelt und getestet.

### 4.3.2 Verfahren zur Online Adaption der Modellparameter

Aufgrund der häufigen Ungenauigkeiten in der Messung der projizierten Bildpunkte können die im letzten Abschnitt definierte Modellgleichungen nicht mit den Positionen der ersten beiden gemessenen Punkten der jeweiligen Sequenz initialisiert werden. Die richtigen Startpositionen der Folge müssen aus den verrauschten Messdaten der jeweiligen Sequenz geschätzt werden.

Um spätere Verwirrung auszuschließen, wurden die beiden Modellgleichungen

4.6 und 4.7 durch Substitution  $a_x = x_1$ ,  $b_x = x_2$ ,  $a_y = y_1$  und  $b_y = y_2$  überführt in:

$$x(n) = \frac{a_x \cdot b_x}{((n-1) \cdot a_x - (n-2) \cdot b_x)} \quad (4.8)$$

$$y(n) = \frac{a_y \cdot b_y}{((n-1) \cdot a_y - (n-2) \cdot b_y)} \quad (4.9)$$

Zur Adaption der Parameter  $a_x$ ,  $b_x$ ,  $a_y$  und  $b_y$  wurden mehrere Verfahren ausprobiert und getestet. Bei der Entwicklung der Verfahren wurde darauf geachtet, dass diese die Voraussetzungen für einen online Einsatz erfüllen. Eine notwendige Bedingung dafür war, eine schnelle und mit der Bildaufnahme zeitnahe Korrektur der Parameter.

#### 4.3.2.1 Fehlerminimierendes Verfahren

Das erste Verfahren basiert auf der Minimierung der Fehlerfunktionen  $F_x(a_x, b_x)$  und  $F_y(a_y, b_y)$ . Diese sind definiert als der quadratische Abstand zwischen den gemessenen und den vorhergesagten Positionen der Bildpunkte der bis dahin aufgenommenen Folge.

$$F_x(a_x, b_x) = \sum_{n=3}^N F_x^n(a_x, b_x) = \sum_{n=3}^N \left( x_n - \frac{a_x \cdot b_x}{(n-1) a_x - (n-2) b_x} \right)^2 \quad (4.10)$$

$$F_y(a_y, b_y) = \sum_{n=3}^N F_y^n(a_y, b_y) = \sum_{n=3}^N \left( y_n - \frac{a_y \cdot b_y}{(n-1) a_y - (n-2) b_y} \right)^2 \quad (4.11)$$

$N$  bezeichnet hier die aktuelle Länge der betrachteten Sequenz.  $x_n$  und  $y_n$ ,  $n \in \{3, \dots, N\}$  sind die gemessenen, verrauschten Bildkoordinaten der Punkte.

Die erste Idee zur Minimierung der Funktionen war die Verwendung des klassischen Gradientenabstiegsverfahrens. Es hat sich allerdings herausgestellt, dass dieses Verfahren sich nicht für den online Einsatz eignet. Mit jedem neu aufgenommenen Bild und damit mit jeder Fortsetzung der aktuellen Folge ändern sich die Fehlerfunktionen und der Minimierungsvorgang muss neu gestartet werden. Zur Beschleunigung der Berechnung wurde die neugestartete Minimierung mit den optimalen Parametern der letzten Minimierung initialisiert. Doch auch dies beschleunigte den Optimierungsprozess nicht in dem erforderlichen Masse. Die Zahl der benötigten Iterationsschritten variierte stark. Der erforderliche Berechnungsaufwand für alle aktuellen Punktfolgen konnte nicht vorab geschätzt werden. Dies war der Hauptgrund warum die Verwendung des klassischen Gradientenabstiegsverfahrens für den online Ansatz nicht in Frage kam.

Stattdessen wurde das stochastische Gradientenabstiegsverfahren [Spa05] verwendet. Dieses Verfahren wird beispielsweise im Bereich des Maschinellen Lernens zur Adaption der Modellparameter eingesetzt. Im Gegensatz zum klassischen Gradientenabstiegsverfahren, wird hier der Gradient auf der Grundlage des zuletzt beobachteten Trainingselements approximiert. Die aktuelle Messung der neuen Punktposition im Bild bildet ein solches Trainingselement. Die Parameter  $a_x$  und  $b_x$  der Fehlerfunktion 4.10 werden in jeden Iterationsschritt nach der folgenden Vorschrift aktualisiert. Mit den Parametern  $a_y$  und  $b_y$  wurde entsprechend verfahren.

$$a_x^n = a_x^{n-1} - \delta \left. \frac{\partial F_x^n(a_x, b_x)}{\partial a_x} \right|_{(a_x^{n-1}, b_x^{n-1})}$$

$$b_x^n = b_x^{n-1} - \delta \left. \frac{\partial F_x^n(a_x, b_x)}{\partial b_x} \right|_{(a_x^{n-1}, b_x^{n-1})}$$

$a_x^{n-1}$  und  $b_x^{n-1}$  sind die im letzten Iterationsschritt berechneten Approximationen der Parameter  $a_x$  und  $b_x$ . Die Funktion  $F_x^n(a_x, b_x)$  ist entsprechend der Gleichung 4.10 definiert. Um das Zusammenwirken zwischen den einzelnen Iterationsschritten und neu gemessenen Punktpositionen im Bild zu verdeutlichen, wurde hier für den Iterationsindex ebenfalls die Variable  $n$  verwendet. Der Iterationsindex startet bei  $n = 3$ . Die Parameter  $a_x^2$  und  $b_x^2$  wurden mit  $x$ -Koordinaten der ersten beiden gemessenen Bildpunkten  $p_1 = (x_1, y_1)$  und  $p_2 = (x_2, y_2)$  der aktuellen Folgen initialisiert,  $a_x^2 = x_1$  und  $b_x^2 = x_2$ .

Die beiden Terme  $\left. \frac{\partial F_x^n(a_x, b_x)}{\partial a_x} \right|_{(a_x^n, b_x^n)}$  und  $\left. \frac{\partial F_x^n(a_x, b_x)}{\partial b_x} \right|_{(a_x^n, b_x^n)}$  bilden eine Approximation des Gradienten basierend auf dem letzten Trainingselement, der zuletzt gemessenen Position des Punktes der jeweiligen Sequenz im Bild. Wie den Gleichungen 4.10 und 4.11 zu entnehmen ist, sind die Terme  $F_x^n(a_x, b_x)$  und  $F_y^n(a_y, b_y)$  als quadratische Abweichungen zwischen der zuletzt gemessenen und vorhergesagten Koordinaten der Bildpunkte definiert.

Der Parameter  $\delta$  wird in der Literatur oft als Schrittweite oder Lernrate bezeichnet. Dieser kontrolliert wie stark während der Optimierung die Werte des Definitionsbereiches in jedem Schritt geändert werden. Abhängig von der Form der Fehlerfunktion kann eine unglückliche Wahl dieses Parameters zu einem schlechten Ergebnis führen. Besitzt die Fehleroberfläche tiefe Täler mit geringer Ausdehnung, so führt eine zu groß gewählte Schrittweite dazu, dass diese Minima während der Optimierung übersprungen werden. Bei flachen Plateaus ist dagegen eine zu klein gewählte Schrittweite sehr unvorteilhaft. An diesen Stellen der Fehleroberfläche ist der Betrag des Gradienten sehr klein. Multiplikation des Gradienten mit der Schrittweite führt zu einer sehr geringen Änderung in dem Definitionsbereich. Dies resultiert

dann oft in einer fast vollständigen Stagnation des Optimierungsverfahrens.

Im Rahmen dieses Ansatzes wird die Schrittweite dynamisch in jedem Iterationsschritt angepasst,  $\delta = \frac{0,1}{n^2}$ , wobei  $n$  der Iterationsindex ist. Mit dieser Wahl konnten in der Simulation beste Ergebnisse erzielt werden. Der relativ kleine Wert des Parameters sorgt dafür, dass an den Modellparametern  $a_x$ ,  $b_x$ ,  $a_y$  und  $b_y$  keine großen Änderungen vorgenommen werden. Untersuchungen haben ergeben, dass die Initialisierung der Parameter mit den Koordinaten der ersten beiden Punkten der Sequenz dafür sorgt, dass man schon recht nah an dem gesuchten Minimum liegt. Es ist allerdings an dieser Stelle anzumerken, dass die richtige Wahl der Schrittweite auch dadurch erschwert wird, dass die hier definierte Fehlerfunktion sich im Grunde mit jeder neu hinzukommenden Messung der jeweiligen Punktsequenz ändert.

Die beiden Fehlerfunktionen wurden unter den folgenden Nebenbedingungen minimiert:

$$|a_x| < |b_x| \quad \text{und} \quad |a_y| < |b_y|$$

Diese Bedingungen basieren auf der Tatsache, dass nach der Verschiebung des Ursprungs des Bildkoordinatensystems in den FOE die Punkte einer Sequenz bei einer geraden Bewegung der Kamera von dem FOE divergieren. Bei der Aktualisierung der Modellparameter wurde diese Bedingung, meistens erst bei höheren Iterationsschritten und trotz der kleinen Schrittweite, manchmal verletzt. In diesem Fall wurden die Parameter dann bei den alten Werten belassen.

Die im nächsten Abschnitt präsentierten Ergebnisse dieses Verfahrens zeugen davon, dass die Güte der berechneten TTC-Werte nicht immer vorhergesagt werden kann. Es wurde regelmäßig beobachtet, dass bei einigen Punktfolgen die berechneten TTC-Werte nach einer anfänglichen Annäherung an die richtigen Werte, gegen Ende der Sequenz stark anfangen zu schwanken. Weil sich dieses Problem aufgrund der im nächsten Abschnitt folgenden Diskussion kaum beheben lässt, wurde im Rahmen dieser Arbeit ein alternatives Verfahren zur Schätzung der Parameter entwickelt.

### 4.3.2.2 Iteratives Verfahren

Dieses Verfahren basiert auf der iterativen Schätzung der Parameter  $a_x$ ,  $b_x$ ,  $a_y$  und  $b_y$ . Die Grundidee dabei ist, ausgehend von dem aktuell gemessenen Bildpunkt  $p_n = (x_n, y_n)$  und dem Iterationsindex  $n$  die theoretisch richtigen Parameter auszurechnen. Die dazu benötigten Vorschriften ergeben sich durch einige Umformungen der beiden Modellgleichungen 4.8 und 4.9.

Unter der Annahme, dass die Parameter  $a_x$  und  $a_y$  bekannt sind, können mittels

der folgenden Formeln die entsprechenden Parameter  $b_x$  und  $b_y$  berechnet werden.

$$b_x = \frac{(n-1)a_x \cdot x_n}{a_x + (n-2)x_n} \quad \text{und} \quad b_y = \frac{(n-1)a_y \cdot y_n}{a_y + (n-2)y_n} \quad (4.12)$$

Ähnlich Formeln lassen sich herleiten, wenn die Parameter  $b_x$  und  $b_y$  als bekannt angenommen werden:

$$a_x = \frac{-x_n \cdot (n-2)b_x}{b_x - x_n(n-1)} \quad \text{und} \quad a_y = \frac{-y_n \cdot (n-2)b_y}{b_y - y_n(n-1)} \quad (4.13)$$

Um dem Rechnung zu tragen, dass die gemessenen Punkte üblicherweise verwechselt sind und damit auch die Initialisierung der Modellparameter fehlerbehaftet ist, wurde auf der Grundlage der obigen Gleichungen folgender Algorithmus entwickelt:

- Mit den Parametern aus dem letzten Iterationsschritt berechne eine Vorhersage für die Position des Bildpunktes im Bild  $n$ .
- Berechne den Mittelpunkt  $m(n) = (m_x^n, m_y^n)$  zwischen der vorhergesagten und der gemessenen Position des Bildpunktes im Bild  $n$ .
- Mittels der obigen Formeln aktualisiere die Parameter  $a_x$  und  $a_y$  oder die Parameter  $b_x$  und  $b_y$ . Verwende dazu nicht die zuletzt gemessene Position des Bildpunktes, sondern den im letzten Schritt berechneten Mittelpunkt  $m(n)$ .

Abhängig davon welche Parameter und in welcher Reihenfolge diese aktualisiert werden, wurde bei diesem Verfahren anfangs zwischen drei verschiedenen Aktualisierungsmodi unterschieden:

**Iterativ A:** Während des gesamten Ablaufs werden nur die Parameter  $a_x$  und  $a_y$  aktualisiert. Die Parameter  $b_x$  und  $b_y$  bleiben nach der Initialisierung unverändert.

**Iterativ B:** Während des gesamten Ablaufs werden nur die Parameter  $b_x$  und  $b_y$  aktualisiert. Die Parameter  $a_x$  und  $a_y$  bleiben nach der Initialisierung unverändert.

**Iterativ AB:** In jedem Iterationsschritt werden abwechselnd entweder die Parameter  $a_x$  und  $a_y$  oder die Parameter  $b_x$  und  $b_y$  aktualisiert.

Ausführliche Experimente in der Simulation haben allerdings gezeigt, dass keines dieser drei Modi den anderen überlegen ist. Im Durchschnitt war das Ergebnis des 'Iterativ A'-Ansatzes nur etwas besser als die anderen Aktualisierungsverfahren. Ähnliche Schlussfolgerungen wurden auch in der Diplomarbeit von Engel [Eng08]

dokumentiert. Neben den drei beschriebenen Modi evaluierte er noch zusätzlich andere Aktualisierungsvorschriften. Diese lieferten allerdings weniger gute Resultate.

Basierend auf diesen Einsichten und den bereits in [Eng08] dokumentierten Ergebnissen wurde im Rahmen dieser Arbeit auf die erneute schriftliche Ausführung der Resultate verzichtet. Stattdessen wird im nächsten Abschnitt untersucht welche Faktoren die Leistungsfähigkeit des 'Iterativ A'-Ansatzes beeinflussen.

### 4.4 Ergebnisse im Rahmen einer Simulationsumgebung

Im Rahmen der hier durchgeführten Experimente wurde neben der Leistungsfähigkeit beider Adaptionsverfahren auch untersucht, welches sich besser für die praktischen Anwendungen eignet. Den Schwerpunkt der Untersuchungen bildete dabei die Ursachenfindung für manchmal beobachtete Schwankungen in den resultierenden TTC-Werten. Gleichzeitig wurde auch überprüft wie stabil die Adaptionsansätze gegenüber Messfehlern in den Punktpositionen sind.

Dabei wurden in der Simulation zur Berechnung der TTC-Werte unter Verwendung eines der beiden Adaptionsansätze folgende Schritte durchgeführt:

- Initialisierung der Parameter  $a_x$ ,  $b_x$ ,  $a_y$  und  $b_y$  mit den Koordinaten der ersten beiden Punkten der entsprechenden Sequenz.
- Aktualisierung der Parameter mit jedem neu gemessenen Punkt  $p_n$  im Bild  $n$  unter der Verwendung eines der beiden Verfahren.
- Schätzung der theoretisch richtigen Position des Punktes im Bild  $n - 1$  mit den aktualisierten Parametern.
- Schätzung der theoretisch richtigen Position des Punktes im Bild  $n$  mit den aktualisierten Parametern.
- Berechnung des TTC-Wertes ausgehend von dem Verschiebungsvektor der beiden geschätzten Punkte.

Mit den exakten Daten lieferten beide Verfahren ideale Ergebnisse. Unterschiede zeigten sich sobald man etwas Rauschen zu den gemessenen Bildpunkten hinzugefügt hat. In den folgenden Unterabschnitten werden die Evaluierungsergebnisse der beiden Verfahren vorgestellt.

#### Analyse des fehlerminimierenden Verfahrens

Im Rahmen der Analyse des ersten Ansatzes wurden mehrere Experimente durchgeführt. Dabei wurden zur Simulation realer Bedingungen die Positionen der proj-

zierten 3D-Punkte in der Bildebene absichtlich verfälscht.

Abbildung 4.8 zeigt beispielhaft solch eine verrauschte Punktfolge (Gaußsches Rauschen, Varianz 0.75) und die entsprechenden TTC-Werte. Die verrauschte Punkt-

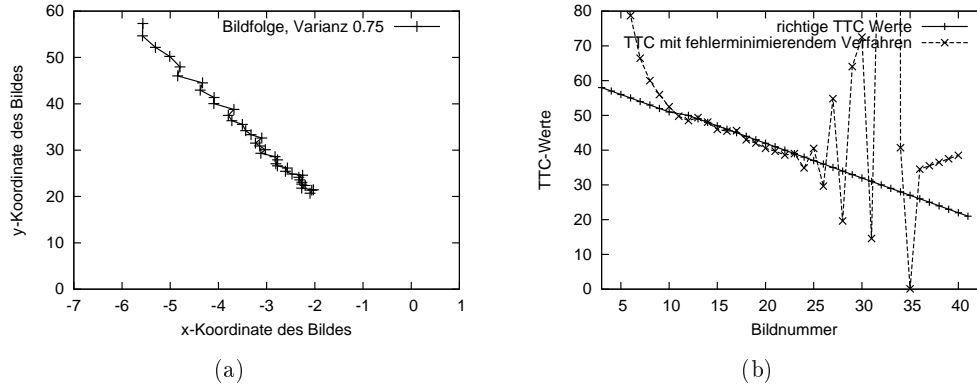


Abbildung 4.8: (a) zeigt eine Punktfolge, die mit Gaussrauschen und Varianz von 0.1 modifiziert wurde. (b) zeigt die richtigen TTC-Werte und solche, die mit dem fehlerminimierenden Verfahren berechnet wurden.

folge ist in Abbildung (a) dargestellt. Das Ergebnis dieses Ansatzes ist in Abbildung (b) zu sehen. Zum Vergleich zeigt hier die nach unten abfallende Gerade die richtigen TTC-Werte. In dem Versuch wurde das Objekt 60 Einheiten vor dem Ursprung des Kamerakoordinatensystems platziert und insgesamt 40 mal nach vorne verschoben. Abbildung (b) zeigt Werte, die erst mit dem dritten Bild berechnet wurden. Die Positionen der Punkte in den ersten beiden Bildern wurden zur Initialisierung der Modellparameter verwendet.

Wie der Abbildung (b) zu entnehmen ist, nähern sich die berechneten TTC-Werte nach einer kurzen Adaptionsphase den richtigen Werten an. Vom 14 bis zum 18 Bild stimmen diese bis auf ein paar Nachkommastellen mit den richtigen Werten überein. Doch ab dem 20 Bild fangen die Werte an zu oszillieren, bis sie dann auf den letzten paar Bildern kontinuierlich ansteigen. Ähnlicher Kurvenverlauf wurde auch bei anderen Punktfolgen beobachtet. Um den Grund für solch ein charakteristisches Verhalten ausfindig zu machen wurden einige weitere Untersuchungen angestellt.

Abbildung 4.9 zeigt die in Gleichung 4.10 definierte Fehlerfunktion  $F_x(a_x, b_x)$  für die oben gezeigte Punktfolge. Wie bereits erwähnt bezeichnet der Parameter  $N$  in dieser Fehlerfunktion die aktuelle Länge der betrachteten Sequenz und wurde daher mit  $N = 40$  initialisiert.

Abbildung 4.9 (a) zeigt die Oberfläche der Fehlerfunktion für  $(a_x, b_x)$  Werte aus dem folgenden Bereich gewählt wurden,  $(a_x, b_x) \in [-10.0 \dots 10.0] \times [-10.0 \dots 10.0]$ . Die senkrechte Achse der Abbildung ist auf den Bereich  $[0 \dots 1000]$  skaliert. Die

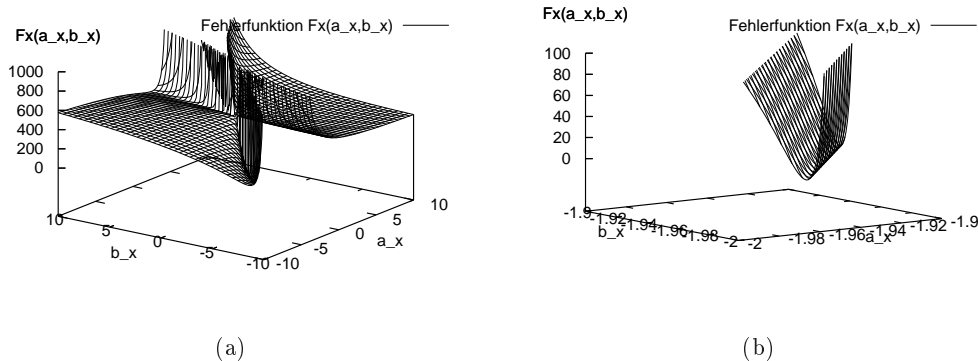


Abbildung 4.9: (a) und (b) zeigen unterschiedlich skaliert die Oberfläche der Fehlerfunktion  $F_x(a_x, b_x)$  für die Punktfolge in Abbildung 4.8.

Fehleroberfläche hat größtenteils Werte bei etwa 600. In der Mitte gibt es aber auch Bereiche, wo der Funktionswert sehr stark und plötzlich abfällt und dann wieder ansteigt. Diese Bereiche sind in so fern interessant, dass der Funktionswert hier fast die Nullgrenze erreicht.

In Abbildung 4.9 (b) wurde der Definitionsbereich auf einen dieser Bereiche  $[-2 \dots -1.9] \times [-2 \dots -1.9]$  verkleinert. Man beachte, dass die senkrechte Achse in dieser Abbildung auf  $[0 \dots 100]$  skaliert ist. Mit der Initialisierung der Parameter mit den Koordinaten der ersten beiden Punkte der Sequenz, hier  $a_x = -1.9239$  und  $b_x = -1.9737$ , befindet man sich schon recht nah an dem gewünschten Minimum. Allerdings kann man ausgehend von der Skalierung des Definitionsbereiches darauf schließen, dass die hier dargestellte Schlucht der Fehlerfunktion sehr schmal ist. Dies erschwert die Minimierung. Die Schrittweite muss exakt gewählt werden. Eine zu große Schrittweite würde zu sehr starken Sprüngen im Wertebereich der Fehlerfunktion führen und damit auch zu starken Schwankungen in den TTC-Werten.

Wie bereits erwähnt wird beim stochastischen Gradientenabstiegsverfahren der Gradient der Fehlerfunktion  $F_x(a_x, b_x)$  in jedem Iterationsschritt auf der Grundlage des zuletzt beobachteten Trainingselements approximiert. Abbildung 4.10 zeigt dazu die Fehlerfunktionen  $F_x^n(a_x, b_x)$  für  $n \in \{10, 20, 25, 30, 35, 39\}$ .

Entsprechend der Gleichung 4.10 sind diese Fehlerfunktionen definiert als der quadratische Abstand zwischen dem gemessenen und dem vorhergesagten Punkt im  $n$ -ten Bild. Der Definitionsbereich wurde auch hier auf das Intervall  $[-2 \dots -1.9] \times [-2 \dots -1.9]$  begrenzt.

Wie der Abbildung 4.10 zu entnehmen ist, können die Fehleroberflächen bei kleinerem  $n$  fast durch eine flache Ebene approximiert werden. Die Berechnung des Gradienten auf solchen Fehleroberflächen und die darauf basierende iterative



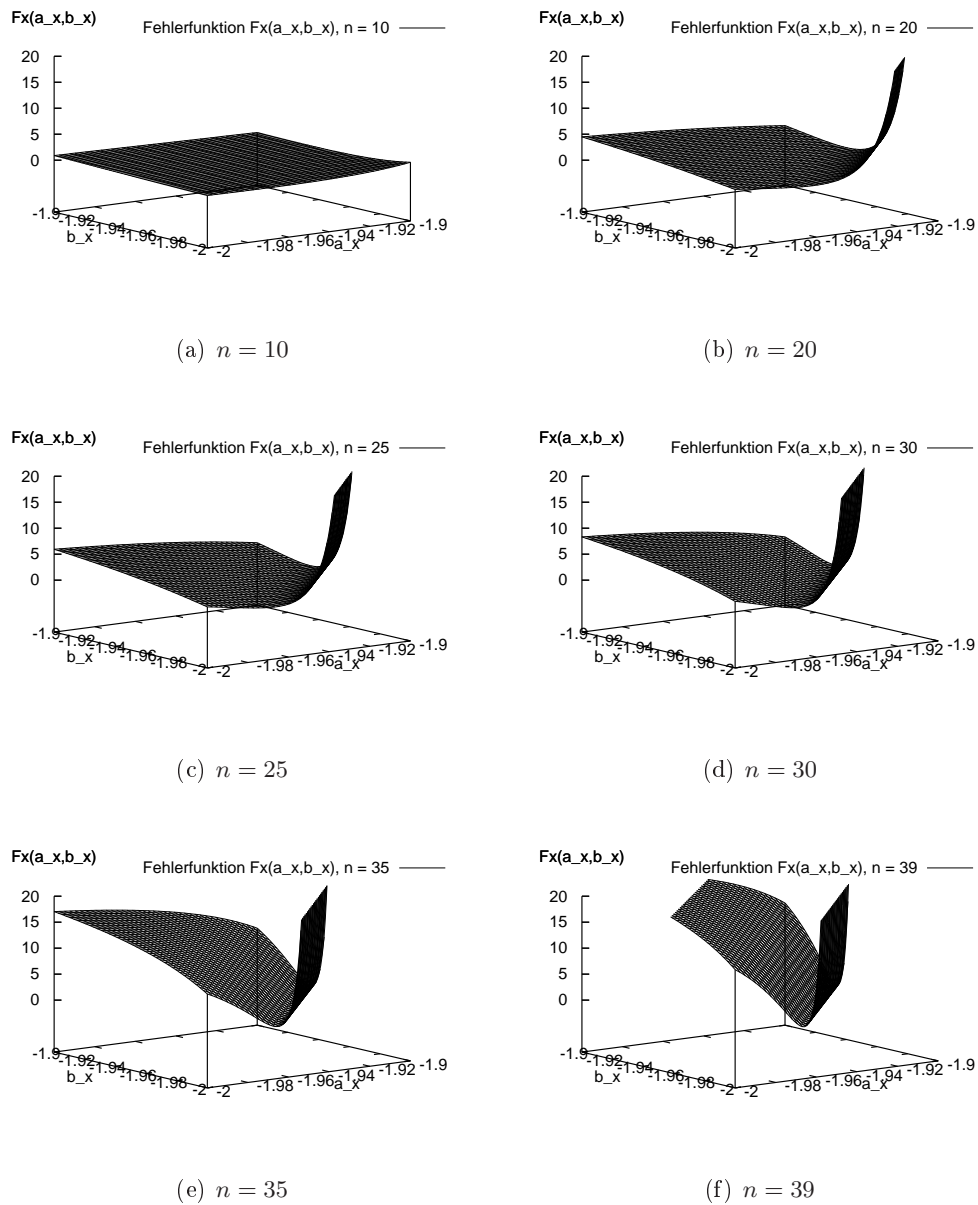


Abbildung 4.10: Fehlerfunktionen  $F_x^n(a_x, b_x)$  für  $n \in \{10, 20, 25, 30, 35, 39\}$ .

Schätzung der Parameter führt unweigerlich zu dem gewünschten Ergebnis. Dies erklärt die anfängliche Annäherung der berechneten TTC-Werte zu den richtigen Werten in Abbildung 4.8 (b).

Die Tatsache, dass in diesem Experiment die berechneten TTC-Werte zwischen dem 10-ten und dem 23-ten Bild sehr nah an den richtigen Werten liegen, lässt sich ebenfalls anhand der Fehleroberflächen begründen. Wie in Abbildung 4.10 zu sehen ist, ändern sich die linken Teile der Fehlerfunktionen für  $n \in [10, \dots, 25]$  kaum. Die einzige beobachtbare Tendenz ist eine leichte Änderung der Steigung in diesem Bereich. Im Gegensatz dazu krümmt sich der rechte Teil der Fehlerfunktionen sehr stark. Es ist also anzunehmen, dass bei diesem Experiment die zu adaptierenden Parameter während des Optimierungsvorgangs sich von links dem Minimum näherten.

Mit steigendem  $n$  wird allerdings die bereits angesprochene Schlucht immer steiler. Wenn die Schrittweite hier nicht richtig angepasst wird, resultiert das in starken Sprüngen in den Funktionswerten. Die in Abbildung 4.8 (b) ab dem 23-ten Bild beobachtete Schwankungen in den TTC-Werten deuten auf eben diesen Sachverhalt hin.

Die später konstant ansteigenden TTC-Werte des ersten Ansatzes lassen sich nun dadurch erklären, dass für  $n = 35$  die Schlucht schon recht schmal ist und die in diesem Iterationsschritt verwendete Schrittweite zu groß war. Dies führte zu einer Schätzung der Parameter die außerhalb der Minimumnachbarschaft lagen. Mehr noch die Fehleroberfläche im Bereich der neu geschätzten Parameter muss sehr flach mit einer Steigung nah der Null sein. Die kontinuierlich ansteigenden TTC-Werte lassen darauf schließen, dass kaum noch Änderungen an den Parametern vorgenommen wurden.

Die richtige Wahl der Schrittweite in jedem Iterationsschritt ist damit entscheidend für die gute Performanz des Verfahrens. Die im Verfahren verwendete dynamische Anpassung der Schrittweite  $\delta$  an die Iterationsnummer reicht nicht aus. Die Änderungen in den Fehleroberflächen hängen nicht von der Anzahl der bis dahin aufgenommenen Bildern ab, sondern von der eigentlichen Entfernung des 3D-Punktes  $P$  zum Ursprung des Kamerakoordinatensystems. Ist der Punkt weit weg, so krümmt sich auch die Fehleroberfläche erst mit einem höheren  $n$ . Anders sieht es aus, wenn der Punkt  $P$  schon nah an der Kamera ist. In diesem Fall weisen die Fehleroberflächen schon beim niedrigen  $n$  eine sehr schmale Schlucht auf. Aufgrund dieses Zusammenhangs gestaltet sich die Anpassung der Schrittweite als sehr schwierig, denn die dazu benötigte Information ist nicht vorhanden.

Die letzte in dieser Diskussion zu klärende Frage ist, warum sich die Fehleroberflächen mit steigendem  $n$  immer weiter krümmen. Welche Faktoren sind für diese

systematische Änderungen der Fehleroberflächen verantwortlich? Um diese Fragen zu beantworten ist es wichtig sich noch einmal vor Augen zu führen wie die Fehlerfunktion aussieht und was mittels der geschätzten Parameter berechnet wird.

Die im letzten Abschnitt vorgestellten Modellgleichungen beschreiben die Entwicklung einer Punktfolge im Bild, die durch die Projektion eines 3D-Punktes bei einer vorwärts Bewegung der Kamera entstehen würde. Die Parameter der Modellgleichungen werden dabei mit den Koordinaten der ersten beiden Punkte der jeweiligen Sequenz initialisiert. Werden die Positionen der Punkte exakt ausgemessen, so stimmen auch die berechneten Vorhersagen mit den nachfolgenden Punktpositionen überein. Abbildung 4.11 zeigt wie mittels der Modellgleichungen vorhergesagten Punktsequenzen aussehen, wenn die ersten beiden Punktpositionen verrauscht werden. Zum Vergleich wurden bei der mittleren Sequenz die Parameter mit den exakten Punktpositionen initialisiert. Insgesamt wurden 40 Punkte in jeder Sequenz berechnet.

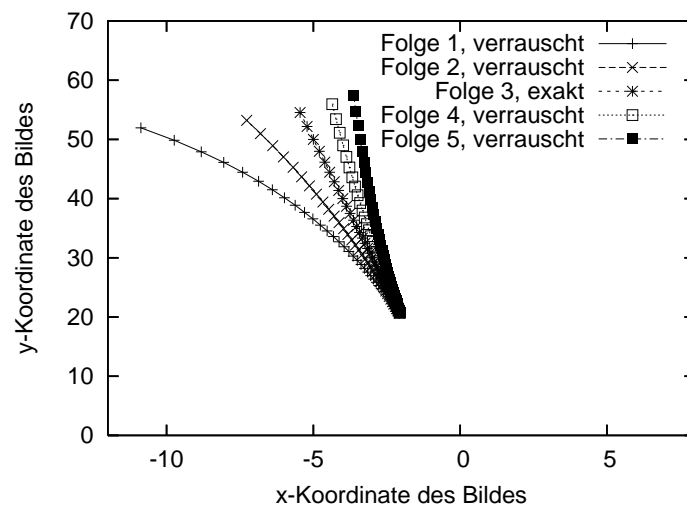


Abbildung 4.11: Fünf unterschiedliche Punktfolgen, welche mit den Modellgleichungen 4.6 und 4.7 konstruiert wurden. Bei allen, außer bei der mittleren Punktfolge wurden die Modellparameter etwas verrauscht.

Es ist leicht zu sehen, dass am Anfang der Sequenzen die euklidische Distanz zwischen den Punkten unterschiedlicher Sequenzen klein ist. Aus diesem Grund sind die Fehlerfunktionen für kleinere  $n$  relativ flach. Die Abstände zwischen dem gemessenen und den vorhergesagten Punkten unterschiedlicher Sequenzen würden hier kaum variieren. Anders sieht es beim größeren  $n$  aus. Die gezeigten Folgen divergieren immer weiter auseinander. Daher verformen sich mit steigendem  $n$  die Fehlerfunktionen immer mehr. Die immer schmalere Schlucht entsteht dadurch, dass mit steigendem  $n$  die Richtigkeit der Punktvorhersage immer sensitiver

zu den gewählten Parametern wird.

Aufgrund der obigen Diskussion wurde auf die weitere Performanzanalyse dieses Verfahrens verzichtet. Die Bestimmung der richtigen Schrittweite  $\delta$  erwies sich als sehr schwierig, denn wie bereits angesprochen hängt diese von dem Abstand des projizierten 3D-Punktes zur Kamera ab. Bei den Experimenten in der Laborumgebung wurde immer wieder die charakteristische Oszillation der TTC-Werte beobachtet. Das zweite Verfahren lieferte hier weit aus stabilere Resultate.

### Analyse des iterativen Verfahrens

Das iterative Verfahren wurde unter den gleichen Bedingungen getestet wie das erste. Die Ergebnisse dieses Verfahrens waren allerdings gegenüber Rauscheinflüssen viel stabiler. Die charakteristischen Schwankungen des fehlerminimierenden Verfahrens gegen Ende der TTC-Sequenz wurden hier nicht beobachtet.

Während der Durchführung der Experimente wurde allerdings festgestellt, dass die Genauigkeit der berechneten TTC-Werte sehr stark davon abhängt, ob die Adaptionsphase für die Modellparameter der jeweiligen Sequenz schon abgeschlossen ist oder nicht. Wie bereits erwähnt, werden die Modellparameter mit den Koordinaten der ersten beiden Punkte der jeweiligen Sequenz initialisiert. Zur Korrektur der Parameter benötigt das Verfahren ein paar weitere Messungen der nachfolgenden Punktpositionen. Die Anzahl der dafür erforderlichen Messungen hängt von der Entfernung des entsprechenden 3D-Punktes ab. Abbildung 4.12 visualisiert diesen Zusammenhang.

In diesem Experiment wurde die Kamera 100 Einheiten vor dem virtuellen Rechteck, bestehend aus insgesamt 40 3D-Punkten, platziert. Die Positionen der Bildpunkte der entsprechenden Sequenzen wurden mit dem Gaußschen Rauschen und der Varianz 0.8 verfälscht. Die durchgezogene Linie in der Abbildung zeigt die richtigen TTC-Werte. Die gestrichelte Linie ohne Kreuze oder Kreise stellt den Durchschnitt der insgesamt 40 TTC-Werte dar, die zu dem jeweiligen Zeitpunkt mit dem einfachen Verfahren berechnet wurden. Der Durchschnitt wurde hier berechnet um aussagekräftige Werte zu erzielen. Aufgrund des hinzugefügten Rauschens zu den Bildpunkten der Sequenz können die berechneten TTC-Werte einmal gut, einmal weniger gut ausfallen. In der Abbildung ist klar ersichtlich, dass die gestrichelte Linie mit der fortschreitenden Annäherung der Kamera an das Rechteck sich den richtigen TTC-Werten nähern. Auch die am Anfang üblichen Zick-Zack Schwankungen nehmen kontinuierlich ab. Der Grund dafür ist, dass mit fortschreitender Annäherung der Kamera die Distanz zwischen zwei aufeinander folgenden Punkten einer Sequenz immer größer wird. Das hinzugefügte Rauschen zu den Bildpunkten

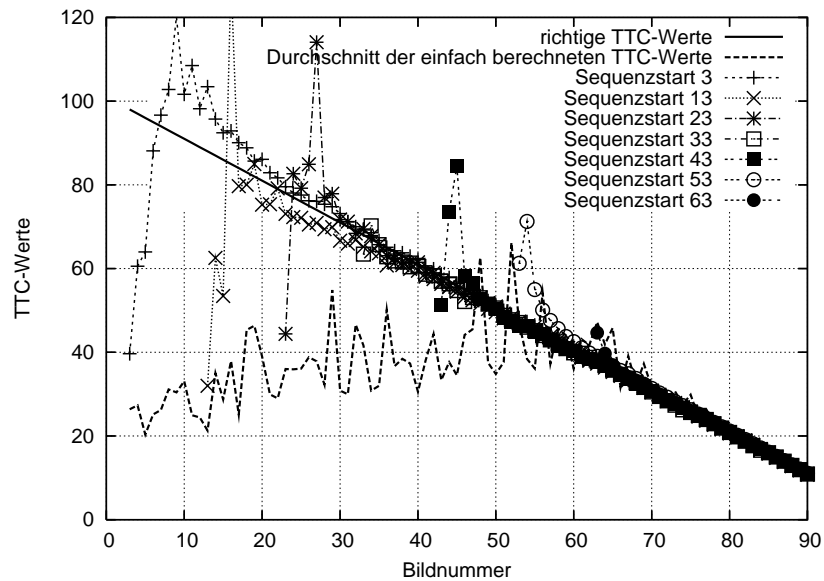


Abbildung 4.12: Einfluss der initialen Entfernung des 3D Punktes auf die Länge der Adaptionphase.

hat damit immer weniger Einfluss auf die Schätzung der richtigen Länge des Verschiebungsvektors. Die Schätzungen der TTC-Werte mit dem einfachen Verfahren werden damit immer genauer. Doch dieses Ergebnis ist in so fern nicht zufriedenstellend, als schon am Anfang des Experimentes die berechneten TTC-Werte im ungefähr dem gleichen Wertebereich lagen. Es ist daher nicht möglich festzustellen, ab welchem Zeitpunkt man sich auf die Richtigkeit der Werte verlassen kann.

Die restlichen Kurven in Abbildung 4.12 zeigen durchschnittliche TTC-Werte, die mit dem iterativen Verfahren vom Typ 'iterativeA' berechnet wurden. Bei den beiden anderen Typen ergaben sich ähnliche Kurven. Um den Einfluss der Adaptionphase zu demonstrieren, wurden die Berechnungen der TTC-Werte zu unterschiedlichen Zeitpunkten gestartet. Das Ergebnis der ersten Berechnung ist durch die punktierte Linie mit Kreuzen dargestellt, die in der Abbildung mit der Bildnummer 3 beginnt. Die ersten beiden Bilder wurden zur Initialisierung des Verfahrens verwendet. Die zweite Berechnung beginnt erst mit der Bildnummer 13, die dritte beginnt mit der Bildnummer 23 usw. Es ist deutlich zu sehen, dass bei allen Berechnungen nach der Adaptionphase die geschätzten Werte kaum noch von den richtigen TTC-Werten abweichen. Im Vergleich zu den einfach berechneten TTC-Werten ist dies eine enorme Verbesserung.

Die Länge der Adaptionphase hängt von zwei Faktoren ab. Zum einen ist es die wirkliche Entfernung des Objektes zum Ursprung des Kamerakoordinatensystems, zum anderen ist es die Genauigkeit der Messungen der Punktpositionen im Bild.

Wie der Abbildung 4.12 zu entnehmen ist, verringert sich die Länge der Adaptionsphase je näher die Kamera zum Objekt bewegt wird. Während bei der ersten Berechnung für die Adaptionsphase etwa 14 Bilder benötigt wurden, waren bei der letzten Berechnung nur 2 Bilder notwendig. Der Grund dafür ist, dass mit der Annäherung der Kamera an das Objekt die Verschiebungsvektoren des so entstehenden optischen Flusses immer größer werden. Das Rauschen, das zu den Bildpunkten hinzugefügt wird, hat damit immer weniger Einfluss auf die Schätzung der richtigen Länge des Verschiebungsvektors. Die Adaptionsphase kann allerdings verlängert werden je mehr Rauschen man zu den Bildpunkten hinzufügt. Die Abbildung 4.13 demonstriert diesen Zusammenhang.

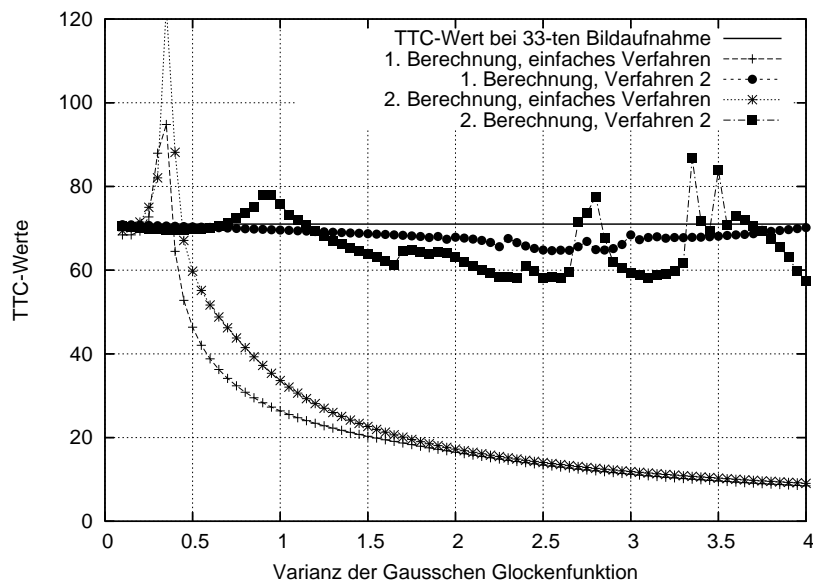


Abbildung 4.13: Wirkung des Rauscheinflusses auf die Länge der Adaptionsphase.

Ähnlich wie im obigen Experiment wurden hier zwei Berechnungen parallel gestartet. Hier wurde ebenfalls das Verfahren 'iterativeA' verwendet. Die erste Adaptionsphase startete mit dem dritten Bild, die zweite mit dem 23-ten. Die Abbildung 4.13 vergleicht die TTC-Werte beider Berechnungen im 33-ten Bild. Der wahre TTC-Wert liegt hier bei 71. Dies ist in der Abbildung durch die gerade Linie dargestellt. Die Kurve mit gefüllten Kreisen stellt die durchschnittlichen TTC-Werte der ersten Berechnung abhängig von dem Rauscheinfluss dar. Ähnlich dem obigen Experiment wurde auch hier der Durchschnitt der jeweils 40 Bildsequenzen berechnet. Die Kurve mit gefüllten Rechtecken stellt die Ergebnisse der zweiten Berechnung dar. Die beiden stark nach unten abfallende gestrichelte Linien mit Kreuzen bildet TTC-Werte, die mit dem Standardverfahren berechnet wurden.

Es ist deutlich zu sehen, dass mit steigendem Rauscheinfluss die TTC-Werte des

einfachen Verfahrens immer weiter von dem richtigen Wert abweichen. Die mit dem zweiten Verfahren berechneten Werte bleiben dagegen relativ nah an der geraden Linie. Vergleicht man die TTC-Werte der ersten und der zweiten Berechnung, so stellt sich heraus, dass bei der ersten Berechnung die Werte weitaus stabiler sind und weniger schwanken im Vergleich zu den Ergebnissen der zweiten Berechnung. Bis zu einem Rauschfaktor von 0.7 stimmen die Ergebnisse der beiden Berechnung fast überein. Ab diesem Wert fängt die Kurve mit den Rechtecken langsam an von dem richtigen Wert zu divergieren. Mit zunehmendem Rauschen reichen die 10 Bilder bei der zweiten Berechnung nicht mehr aus um die Modellparameter richtig zu adaptieren. Die Werte der ersten Berechnung sind nur deshalb besser, weil hier für die Adaption insgesamt 30 Bilder zur Verfügung stehen. Aus der Abbildung 4.13 ist damit klar ersichtlich, dass je größer der Rauscheinfluss bei den Messdaten ist, desto mehr Daten braucht das Verfahren zur Adaption der Modellparameter. Nicht desto trotz ist die Schätzung der durchschnittlichen TTC-Werte mit diesem Verfahren deutlich besser als mit dem üblichen Standardverfahren.

## 4.5 Diskussion und Zusammenfassung

In diesem Kapitel wurde ein neues Verfahren zur Berechnung der TTC-Werte vorgestellt. Es basiert auf den Gesetzen der projektiven Geometrie. Zur genaueren TTC-Berechnung werden hier die theoretisch richtigen Verläufe der 2D-Projektionen von 3D-Punkten während der Kamerabewegung herangezogen. Dabei erfolgt die Schätzung der theoretisch richtigen 2D-Verläufe auf der Grundlage der gemessenen Punktpositionen im Bild. Der Vorteil hierbei ist, dass die TTC-Berechnung nicht direkt auf den oft verrauschten Bildmessungen erfolgt. Der zwischengeschaltete Schritt der Schätzung der 2D-Projektionen sorgt dafür, dass die fehlerhaft gemessene Punktpositionen im Bild weniger Einfluss auf die berechneten TTC-Werte haben.

Das Grundgerüst des Verfahrens bilden Modellgleichungen, mittels derer die Position des entsprechenden Merkmals im kommenden Bild geschätzt werden kann. Zur Adaption der Modellparameter wurden hier zwei Ansätze vorgestellt. Der erste Ansatz basiert auf der Minimierung einer Fehlerfunktion. Um eine online stattfindende Parameteradaption zu gewährleisten wurde hier zur Optimumssuche das stochastische Gradientenabstiegsverfahren eingesetzt. Dabei wird zur Minimierung in jedem Schritt nur die zuletzt erfolgte Messung der entsprechenden Punktposition im Bild herangezogen.

Der zweite Adoptionsansatz verfolgt eine andere Strategie. Die neuen Parameter werden hier basierend auf einer hergeleiteten Vorschrift in jedem Schritt neu berechnet. Dazu wird, ähnlich dem ersten Ansatz nur die zuletzt durchgeführte

## **102 Kapitel 4. Entfernungsschätzung bei geradliniger Kamerabewegung**

Punktmessung herangezogen. Der Vorteil dieses iterativen Ansatzes ist, dass es keine frei wählbare Parameter enthält.

In den anschließenden Experimenten unter kontrollierten Bedingungen wurde festgestellt, dass für die praktische Anwendungen sich der iterative Ansatz besser eignet als das erste fehlerminimierende Verfahren. Die Leistungsfähigkeit des letzteren Ansatzes war sehr stark von der gewählten Schrittweite abhängig. Experimente haben wiederum gezeigt, dass die richtige Wahl dieses Parameters nur durch die anfängliche Entfernung des abgebildeten 3D-Punktes bestimmt werden konnte. Ausgehend von dieser Einsicht war es kaum möglich ein Verfahren zur automatischen Bestimmung der Schrittweite zu entwickeln.

Beim iterativen Verfahren dagegen wurde während der durchgeführten Experimente viel weniger Schwankungen in den resultierenden TTC-Werten beobachtet. Nach dem Abschluss der anfänglichen Adaptionsphase blieben die berechneten TTC-Werte beim gleichen Rauscheinfluss relativ stabil. Die dem fehlerminimierenden Verfahren charakteristische Schwankungen gegen Ende der Sequenz wurden hier nicht beobachtet. Der zweite Ansatz erwies sich ebenfalls als sehr stabil gegenüber erhöhtem Rauscheinfluss in den Positionen der projizierten Punkte in der Bildebene.

Die Herleitung der Modellgleichungen sowie der fehlerminimierende Ansatz zur Adaption der Modellparameter wurden zum ersten mal auf der 18-ten 'International Conference on Computer Graphics and Vision' im Jahr 2008 vorgestellt [SNP08]. Der iterative Ansatz zur Parameteradaption wurde in einem weiteren Artikel auf dem 21-ten 'Fachgespräch Autonome Mobile Systeme 09' präsentiert [SP09a]. Gleichzeitig wurden in diesem Artikel auch Erweiterungen diskutiert, um die sehr einschränkende Annahme einer konstanten Geradeausbewegung der Kamera aufzulösen. Das folgende Kapitel beschreibt die Details dieser Erweiterungen und analysiert die Performanz dieses Verfahrens in praktischen Anwendungen.



# Entfernungsschätzung bei einer 2D Kamerabewegung

---

## Kapitelinhalt

---

<b>5.1</b>	<b>Erweiterung des Ansatzes auf verschiedene Bewegungsarten</b>	<b>104</b>
<b>5.2</b>	<b>Versuchsumgebung und technische Ausstattung . . . . .</b>	<b>111</b>
<b>5.3</b>	<b>Einfluss der Position der Merkmale im Bild . . . . .</b>	<b>113</b>
<b>5.4</b>	<b>Einfluss der Messungenauigkeit der Merkmale . . . . .</b>	<b>117</b>
<b>5.5</b>	<b>Ergebnisse im Rahmen einer Laborumgebung . . . . .</b>	<b>127</b>
<b>5.6</b>	<b>Diskussion und Zusammenfassung . . . . .</b>	<b>131</b>

---

Das im letzten Kapitel vorgestellte Verfahren zur Schätzung der TTC-Werte liefert in der Simulation sehr gute Ergebnisse bei einer konstanten und geraden Bewegung der Kamera. Allerdings ist die Restriktion auf nur diese Art der Bewegung nicht hinreichend für einen Einsatz in realen Anwendungen. In der Praxis werden die TTC-Werte oft für die Navigation oder Steuerung von mobilen Robotern und Fahrzeugen verwendet. Dabei kommen neben den Rotationsfahrten auch Geradeausfahrten mit variierender Geschwindigkeit vor. In solchen Situationen würde das hier vorgestellte Verfahren scheitern, weil die wichtige Annahme einer konstanten, geraden Bewegung der Kamera nicht erfüllt wäre.

Um diese sehr einschränkende Annahme aufzulösen, werden im Rahmen dieses Kapitels Erweiterungen des Ansatzes vorgestellt. Die Parameter der Modellgleichungen werden abhängig von der Bewegungsart der Kamera aktualisiert. In den folgenden Abschnitten werden drei Bewegungsarten betrachtet: Geradeausfahrt mit variierender Geschwindigkeit, einfache Rotation und Kurvenfahrt.

Gleichzeitig werden in diesem Kapitel auch Ergebnisse einer weiteren Experimentenreihe vorgestellt. Diese wurde im Rahmen einer Laborumgebung durchgeführt und diente dazu den praktischen Einsatz des Verfahrens zu testen. Gleichzeitig wurde dabei auch untersucht, welche Merkmale sich besser für die Berechnung von TTC-Werten eignen und ob die Position der Merkmale im Bild ebenfalls einen Einfluss auf die Güte der berechneten TTC-Werte hat.

Das Kapitel schließt ab mit einer Zusammenfassung und einer ausführlichen Diskussion über die dabei erzielten Ergebnisse.

## 5.1 Erweiterung des Ansatzes auf verschiedene Bewegungsarten

Insgesamt wurden drei zusätzliche Bewegungsarten untersucht: Geradeausfahrt mit variierender Geschwindigkeit, Punktrotation und Kurvenfahrt. Die entsprechenden Aktualisierungsmethoden integrieren in sich nahtlos den bereits in Simulation getesteten iterativen Ansatz zur Adaption der Modellparameter und stellen damit eine logische Erweiterung dieses Ansatzes dar.

Für die Anwendbarkeit der Methoden werden noch zusätzliche Informationen über die Kalibrierung und die Bewegungsart der Kamera benötigt. Letzteres kann üblicherweise über die aktuellen Odometriedaten des autonom agierenden Roboters bestimmt werden. Die Kalibrierung der Kamera muss allerdings vor dem Start der Anwendung erfolgen.

### 5.1.1 Gerade Bewegung mit variierender Geschwindigkeit

Damit das Verfahren auch bei geraden Bewegungen mit variabler Geschwindigkeit vernünftige Ergebnisse liefert, müssen die Modellparameter  $a_x$ ,  $b_x$ ,  $a_y$  und  $b_y$  entsprechend der Geschwindigkeitsänderung aktualisiert werden. Die grundlegende Idee dabei ist, die Parameter so anzupassen, als wäre die Kamera von Anfang an mit der geänderten Geschwindigkeit bewegt worden. Die Abbildung 5.1 veranschaulicht diese Idee graphisch.

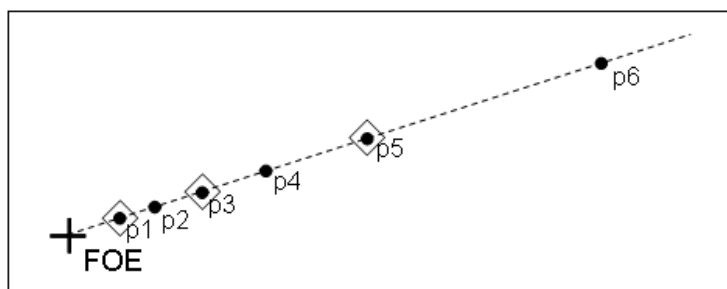


Abbildung 5.1: Entwicklung der Feature Positionen bei variierender Geschwindigkeit der Kamerabewegung.

Hierbei wird davon ausgegangen, dass die Merkmalspositionen exakt ausgemessen werden. Die schwarzen runden Punkte repräsentieren die Positionen eines Merkmals in mehreren aufeinander folgenden Bildern. Weil die Kamera in Richtung des

Objektes bewegt wird, divergiert der Merkmal immer weiter von dem FOE. Hier wird angenommen, dass die Geschwindigkeit der Kamerabewegung zwischen den ersten fünf Bildern konstant ist. Weil die Bilder immer in äquidistanten Zeitabständen aufgenommen werden, kann davon ausgegangen werden, dass auch die zurückgelegte Strecke zwischen zwei Bildern gleich ist. Zwischen dem fünften und dem sechsten Bild wurde die Geschwindigkeit verdoppelt. Die Position des betrachteten Merkmals im sechsten Bild liegt daher viel weiter von dem FOE, als es der Fall wäre, wenn keine Geschwindigkeitsänderung stattgefunden hätte. Die umrandeten Merkmalspositionen in der Abbildung 5.1 repräsentieren die Positionen, die der Merkmal hätte, wenn die Kamera von Anfang an mit der geänderten Geschwindigkeit bewegt worden wäre. Mit der doppelten Geschwindigkeit werden weniger Bilder aufgenommen, bis der Merkmal die letzte Position erreicht. Damit müssen nicht nur die Parameter  $b_x$  und  $b_y$  aktualisiert werden, sondern auch der Iterationsindex  $n$ . Ähnliche Situation ergibt sich, wenn die Geschwindigkeit bzw. die zurückgelegte Strecke zwischen den letzten beiden Bildern verringert wird. Wie der Abbildung 5.1 zu entnehmen ist, müssen die Parameter  $a_x$  und  $a_y$  nicht verändert werden. Unabhängig von der Geschwindigkeit ist die Position des Merkmals im ersten Bild in allen Fällen gleich.

Die Aktualisierung des Iterationsindex hängt von dem Verhältnis der zuletzt gefahrenen Strecke zu der gefahrenen Strecke zwischen den vorletzten beiden Bildern. Die folgende Formel beschreibt die Aktualisierungsvorschrift.

$$n_{neu} = \left( (n_{alt} - 1) \cdot \frac{d_{alt}}{d_{neu}} \right) + 1 \tag{5.1}$$

$n_{alt}$  und  $n_{neu}$  bezeichnen hier jeweils den alten und den neuen Iterationsindex. Der Parameter  $d_{alt}$  ist die zurückgelegte Strecke zwischen den beiden Zeitpunkten der Bildaufnahmen  $t-2$  und  $t-1$ , wobei  $t$  hier den Zeitpunkt der letzten Bildaufnahme bezeichnet. Bezogen auf das Beispiel in Abbildung 5.1 ist dies die Strecke zwischen dem vierten und dem fünften Bild. Der Parameter  $d_{neu}$  bezeichnet die Strecke, die schon mit der geänderten Geschwindigkeit zurückgelegt wurde. In dem obigen Beispiel ist dies die Strecke zwischen dem fünften und dem sechsten Bild.

Damit die bereits vorgestellte Aktualisierungsvorschrift des iterativen Verfahrens aus dem letzten Kapitel wie gewohnt angewendet werden kann, wird der Iterationsindex immer für den vorletzten Zeitpunkt aktualisiert. In dem betrachteten Beispiel ist  $n_{alt} = 5$  zu wählen. Weil die Geschwindigkeit zwischen den letzten beiden Bildern verdoppelt wurde, ist der Term  $\frac{d_{alt}}{d_{neu}} = \frac{1}{2}$ . Damit ist der neue Iterationsindex  $n_{neu} = 3$ . Aus der Formel 5.1 ist klar ersichtlich, dass der neu berechnete Iterationsindex  $n_{neu}$  keine natürliche Zahl sein muss. Abhängig von der Geschwindigkeitsänderung und damit vom Verhältnis der beiden zurückgelegten Strecken kann der

Iterationsindex  $n_{neu}$  auch rationale Zahlen annehmen.

Ausgehend vom neuen Iterationsindex werden die Modellparameter  $b_x$  und  $b_y$  wie folgt aktualisiert:

$$b_x^{neu} = \frac{(n_{neu} - 1) a_x \cdot m_x}{a_x + (n_{neu} - 2) m_x}, \quad b_y^{neu} = \frac{(n_{neu} - 1) a_y \cdot m_y}{a_y + (n_{neu} - 2) m_y}$$

Die beiden Formeln entsprechen den Gleichungen 4.12, die bereits im Kapitel 4 bei der Beschreibung des iterativen Ansatzes zur Adaptierung der Modellparameter hergeleitet wurden. Im Unterschied dazu wurde hier der Iterationsindex  $n$  durch den neuen Iterationsindex  $n_{neu}$  ersetzt. Auch die in Gleichungen 4.12 verwendeten  $(x_n, y_n)$  Punktkoordinaten, wurden hier entsprechend dem iterativen Adaptionsansatz durch den, im letzten Aktualisierungsschritt ausgerechneten Mittelpunkt  $m = (m_x, m_y)$  zwischen der vorhergesagten und gemessenen Merkmalsposition ausgetauscht. Bezogen auf das obige Beispiel wäre dies die Merkmalsposition  $p_5$ .

Mit dem neuen Parametersatz  $(a_x, a_y, b_x^{neu}, b_y^{neu}, n_{neu})$  ist es nun möglich eine Vorhersage über die nächste Merkmalsposition zu treffen. In dem betrachteten Beispiel können dann die entsprechenden Modellparameter, zusammen mit der bereits gemessenen Merkmalsposition  $p_6$  mittels der Aktualisierungsvorschrift des iterativen Ansatzes aktualisiert werden.

### 5.1.2 Punktrotation in der zur Grundfläche parallelen Ebene

Obwohl bei der einfachen Rotation oder Punktrotation keine Vorwärtsbewegung der Kamera stattfindet, verändert sich dennoch der TTC Abstand des 3D-Punktes  $P$  zu der rotierenden Bildebene. Die Abbildung 5.2 veranschaulicht diesen Zusammenhang. Die geänderte Orientierung der Kamera ist hier durch die beiden Pfeile angedeutet. Aufgrund der Rotation ändert sich auch die Position und die Ausrichtung der Bildebene. Der zum 3D-Punkt  $P$  berechnete TTC-Abstand vor und nach der Rotation ist in der Abbildung durch die orthogonalen Geraden zu den beiden Bildebenen dargestellt. Wie ebenfalls deutlich zu sehen ist, ändert sich auch die Position des projizierten 3D-Punktes in der Bildebene. Folglich müssen auch in diesem Fall die Modellparameter  $a_x$ ,  $a_y$ ,  $b_x$  und  $b_y$  aktualisiert werden.

Zur Aktualisierung der Parameter werden Informationen über den Rotationswinkel und über die intrinsische Kalibrierung der Kamera benötigt. Aufgrund der Herleitung der Modellgleichungen sind die Modellparameter die theoretischen Positionen der ersten beiden Punkte einer Sequenz. Diese Positionen sind im Bildkoordinatensystem gegeben, wobei der Ursprung des Koordinatensystems ins FOE verschoben wurde. Vor der Multiplikation mit der Rotationsmatrix müssen diese zunächst ins Kamerakoordinatensystem umgerechnet werden.

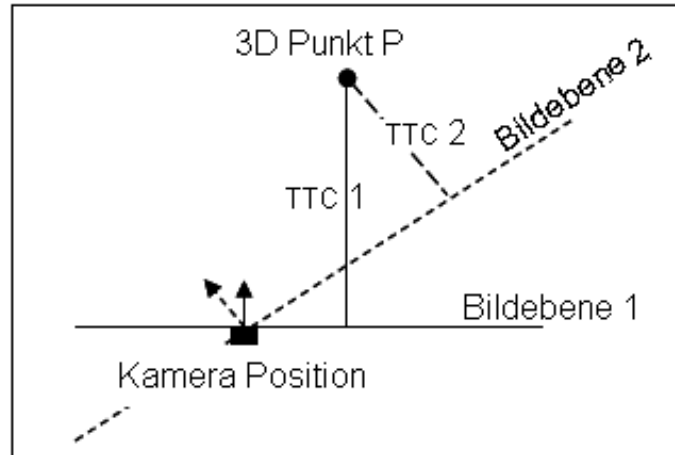


Abbildung 5.2: Graphische Darstellung der Veränderung des TTC Abstandes zwischen dem 3D-Punkt  $P$  und der rotierenden Bildebene der Kamera.

Dies erfolgt mittels der Inversen der intrinsischen Kalibrierungsmatrix:

$$\mathbf{X}^C = M_{int}^{-1} \cdot K \mathbf{x}$$

wobei  $\mathbf{x} = (x, y, 1)$  hier die homogenen Koordinaten des betrachteten Bildpunktes bezeichnet. Die Multiplikation dieses Punktes mit der Matrix  $K$  liefert neue Koordinaten des Punktes und zwar im Bildkoordinatensystem, dessen Ursprung in der oberen linken Ecke des Bildes liegt.  $M_{int}^{-1}$  ist die Inverse der intrinsischen Kalibrierungsmatrix, und  $\mathbf{X}^C$  ist der Vektor des entsprechenden Punktes im Kamerakoodinatenystem.

Die Rotation dieses Vektors liefert einen neuen Vektor  $\mathbf{X}'^C$

$$\mathbf{X}'^C = R \cdot \mathbf{X}^C$$

wobei  $R$  hier die Rotationsmatrix bezeichnet. Die Umrechnung des neuen Vektors in das Bildkoordinatensystem, dessen Ursprung im FOE liegt, liefert

$$\mathbf{x}' = K^{-1} M_{int} \cdot \mathbf{X}'^C$$

Damit ergibt sich die folgende Aktualisierungsvorschrift für die Modellparameter  $a_x$  und  $a_y$ :

$$a_{neu} = T \cdot a_{alt}$$

mit  $T = K^{-1} M_{int} R M_{int}^{-1} K$  und  $a_{alt} = (a_x^{alt}, a_y^{alt}, 1)$ . Die Aktualisierungsvorschrift für die Modellparameter  $b_x$  und  $b_y$  ist analog definiert.

Mit dem neuen Parametersatz kann der neue TTC-Wert für den 3D-Punkt  $P$

wie gewohnt ausgerechnet werden. Der Iterationsindex muss in diesem Fall nicht verändert werden.

Die hier vorgestellte Aktualisierungsvorschrift funktioniert allerdings nur unter der Annahme, dass das Kamerazentrum genau auf der Rotationsachse des Roboters liegt. Bedauerlicherweise ist dies bei vielen Roboterkonfigurationen nicht der Fall. Der nächste Abschnitt zeigt wie solche Situationen als Spezialfälle der Kurvenfahrt interpretiert werden können.

### 5.1.3 Kurvenartige Bewegung

Die Kurvenfahrt ist definiert als eine Zusammensetzung der rotierenden und der translatorischen Bewegung. Dies bedeutet, dass bei der Aufnahme zweier nacheinander folgender Bilder sich sowohl die Position als auch die Orientierung der Kamera geändert hat. Die Abbildung 5.3 repräsentiert diese Situation graphisch. Der Halb-

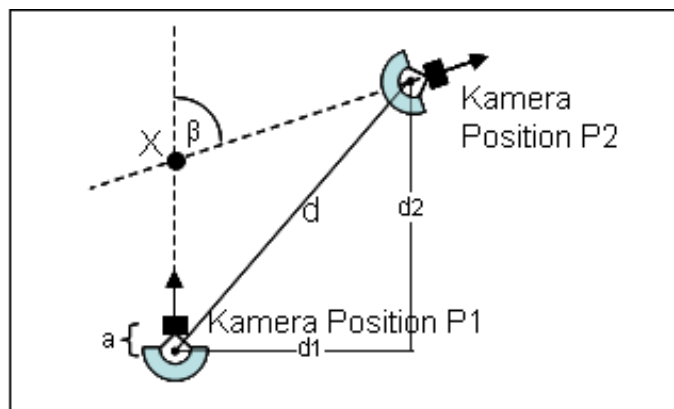


Abbildung 5.3: Graphische Darstellung einer Kurvenfahrt, bei der der translatorische Anteil der Bewegung stärker ausgeprägt ist als der rotierende Anteil.

bogen in der Abbildung repräsentiert den Roboter, der seine Rotationsachse in der Mitte des Bogens hat. Der etwas nach vorne versetzte schwarze Rechteck zusammen mit dem Pfeil gibt die Position und die Orientierung der Kamera an. Die beiden Positionen  $P1$  und  $P2$  bezeichnen die Kamerapositionen zu den beiden Zeitpunkten der Bildaufnahme. Im Unterschied zu der Annahme aus dem vorigen Abschnitt liegt die Kamera hier nicht auf der Rotationsachse des Roboters. Die entsprechende Verschiebung ist in der Abbildung durch den Vektor  $a = (a_1, a_2, a_3)$  bezüglich des Roboterkoordinatensystems gekennzeichnet. Unter der Annahme, dass die Kamera in Fahrtrichtung des Roboters gerichtet ist, gilt ohne Beschränkung der Allgemeinheit  $a_2 = 0$  und  $a_3 = 0$ .

Für das hier vorgestellte Aktualisierungsverfahren muss der relative Abstand zwischen den beiden Kamerapositionen sowie der Rotationswinkel der Kamera be-

kannt sein. In den praktischen Anwendungen lässt sich üblicherweise aus den Odometriedaten des Roboters die Information über die Position des Roboters und den Rotationswinkel gewinnen. In der Abbildung 5.3 ist die relative Verschiebung zwischen den beiden Roboterpositionen mit dem Vektor  $d = (d_1, d_2)$  bezeichnet. Zusammen mit dem bereits angesprochenen Vektor  $a$  lässt sich damit auch die relative Verschiebung der Kamera zu den beiden Zeitpunkten der Bildaufnahme berechnen.

Für die Berechnung der neuen Modellparameter wird die Kurvenfahrt durch eine Abfolge von drei einfachen Schritten simuliert. Dabei werden die Modellparameter und der Iterationsindex so aktualisiert, als wäre die Kamera zwischen den beiden Bildaufnahmen entlang einer Ecke bewegt worden. In Abbildung 5.3 sind die folgenden drei Schritte graphisch dargestellt.

- Translation der Kamera von der Position  $P1$  zur Position  $X$ .
- Rotation der Kamera um den Winkel  $\beta$ .
- Translation der Kamera von der Position  $X$  zur Position  $P2$ .

Zur Durchführung dieser Schritte werden die bereits vorgestellten Aktualisierungsmethoden für Geradeausfahrt mit variierender Geschwindigkeit sowie für einfache Rotation eingesetzt. Die Längen der beiden Strecken  $\overline{P1 X}$  und  $\overline{X P2}$  lassen sich mittels der Geometrie wie folgt ausrechnen:

$$\overline{P1 X} = d_2 - \frac{d_1}{\tan\beta} - a_1, \quad \overline{X P2} = \frac{d_1}{\cos\beta} + a_1$$

wobei  $d_1$  und  $d_2$  hier die  $x$ - und  $y$ -Koordinaten des bekannten Translationsvektors  $d$  bezeichnen. Die einzelnen Schritte zur Neuberechnung der Modellparameter sind im Folgenden zusammengefasst:

**Schritt 1:** Aktualisiere den Iterationsindex und die Modellparameter  $a_x, b_x, a_y$  und  $b_y$  so, als ob die Kamera auch schon vorher zwischen zwei Bildaufnahmen die Strecke  $\overline{P1 X}$  bewegt wurde.

**Schritt 2:** Erhöhe den neuen Iterationsindex um 1 und berechne mit den neuen Modellparametern die theoretische Position des entsprechenden Merkmals im Bild, so als ob die Kamera wirklich zur Position  $X$  bewegt worden wäre.

**Schritt 3:** Aktualisiere die Modellparameter entsprechend dem Rotationswinkel  $\beta$ .

**Schritt 4:** Aktualisiere erneut den Iterationsindex und die Modellparameter  $a_x, b_x, a_y$  und  $b_y$  so, als ob die Kamera auch schon vorher zwischen zwei Bildaufnahmen die Strecke  $\overline{X P2}$  bewegt wurde.

**Schritt 5:** Mit den neuen Parametern und dem wieder um eins erhöhten Iterationsindex berechne eine Vorhersage für die Position des Features und zusammen mit der Messung im aktuellen Bild aktualisiere die Modellparameter entsprechend dem iterativen Adaptionsverfahren aus Kapitel 4.

In der in Abbildung 5.3 dargestellten Situation ist die Länge der berechneten Strecke  $\overline{P1 X}$  positiv. In manchen Fällen kann der Wert allerdings auch ein negatives Vorzeichen haben. In solchen Situationen muss die Kamera theoretisch rückwärts bewegt werden, um die Position  $X$  zu erreichen. Diese Fälle treten auf, wenn bei der Kurvenfahrt zwischen zwei Bildern die Rotation viel stärker ausgeprägt ist als der entsprechende Translationsanteil. Die Abbildung 5.4 gibt dazu ein Beispiel. Zur bes-

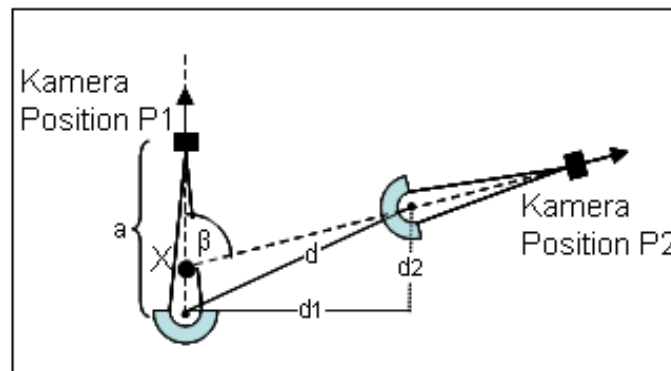


Abbildung 5.4: Graphische Darstellung einer Kurvenfahrt. In dieser Situation ist der rotierende Anteil der Bewegung stärker ausgeprägt als der translatorische Anteil.

seren Anschauung wurde die Verschiebung  $a$  der Kamera von der Rotationsachse des Roboters in diesem Beispiel um einiges vergrößert. Wie der Abbildung zu entnehmen ist, muss die Kamera von der Position  $P1$  rückwärts zur Position  $X$  verschoben werden. Für die Aktualisierung der Parameter bedeutet dies, dass im zweiten Schritt der neue Iterationsindex nicht um eins erhöht, sondern stattdessen um eins verringert wird. Die damit berechnete neue Position des Merkmals im Bild entspricht der Position, die der Merkmal hätte, wenn die Kamera zu einem früheren Zeitpunkt an der Position  $X$  gewesen wäre. Die restlichen Schritte des Aktualisierungsverfahrens bleiben gleich.

Besteht die Kurvenfahrt aus reiner Rotation, so gilt für die Strecken  $\overline{P1 X} = a_1$  und  $\overline{X P2} = a_1$ . Damit können auch die Fällen abgedeckt werden, wenn das Kamerazentrum nicht auf der Rotationsachse des Roboters liegt und der Roboter eine rein rotierende Bewegung durchführt.



## 5.2 Versuchsumgebung und technische Ausstattung

Um aussagekräftige Resultate über die Performanz der Verfahren zu erzielen, wurden diese unter kontrollierten Bedingungen getestet. Der Laborraum wurde so präpariert, dass zu jedem Zeitpunkt der Bildaufnahme die verbleibende Entfernung des Roboters zu den Hindernissen bekannt war. Damit war es möglich die Genauigkeit der berechneten TTC-Werte zu untersuchen.

### Umgebung

Für die Experimente wurde eine Fläche von  $28,549m^2$  im Laborraum präpariert. Auf dem  $3,63m \times 7,865m$  großem Rechteck wurden drei Hindernisse mit unterschiedlichem Abstand zur anfänglichen Position des Roboters aufgestellt. Abbildung 5.5 zeigt das Kamerabild, welches von der Startposition der Roboters aufgenommen wurde und den schematischen und maßstabgetreuen Aufbau der Versuchsumgebung. Hierbei wurden bewusst großflächige und rechtwinklige Hindernisse gewählt.

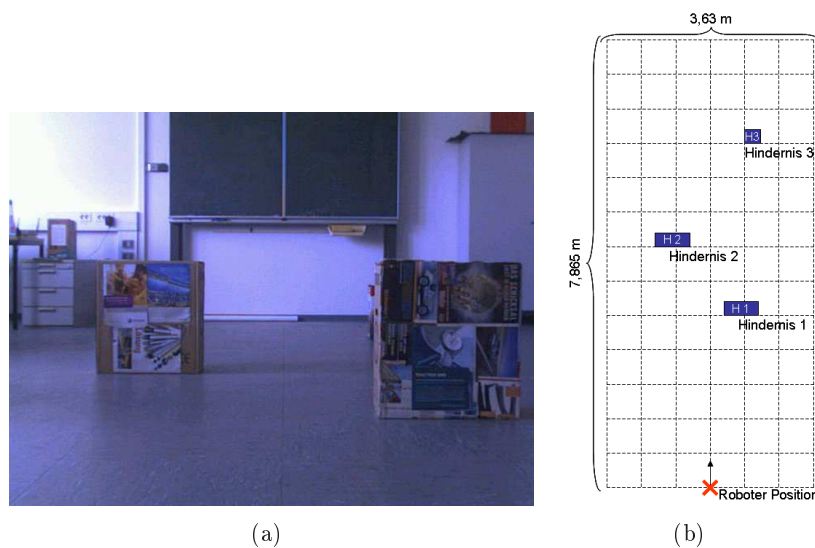


Abbildung 5.5: (a) zeigt das Kamerabild welches von der Startposition des Roboters aufgenommen wurde. (b) zeigt graphisch den maßstabgetreuen Aufbau der Versuchsumgebung.

Diese wurden parallel zur Bildebene der Kamera aufgestellt. Ein solcher Aufbau vereinfachte sehr die Auswertung der Experimente. Auf diese Weise konnten allen Punkten eines bestimmten Bildbereichs ein bestimmter TTC-Wert zugeordnet werden.

Die verwendeten Kartons wurden mit verschiedenen Zeitungsausschnitten beklebt. Damit konnten genügend Merkmale auf diesen extrahiert und verfolgt werden.

### Technische Ausstattung

Bei den durchgeführten Experimenten kam der ER1 Roboter der Firma Evolution Robotics zum Einsatz. Dabei handelt es sich um einen fahrenden Roboter, der sich auf zwei Antriebsrädern und einem Stützrad bewegt. Den Körper bildet ein System, welches basierend auf dem Steck-Prinzip (ähnlich der Fischertechnik) sich beliebig konfigurieren lässt. Der Roboter besitzt zusätzlich zwei Motoren für den Antrieb, ein Notebook zur Steuerung sowie einen 12V Akku. Abbildung 5.6 zeigt unterschiedliche Ansichten des Roboters.

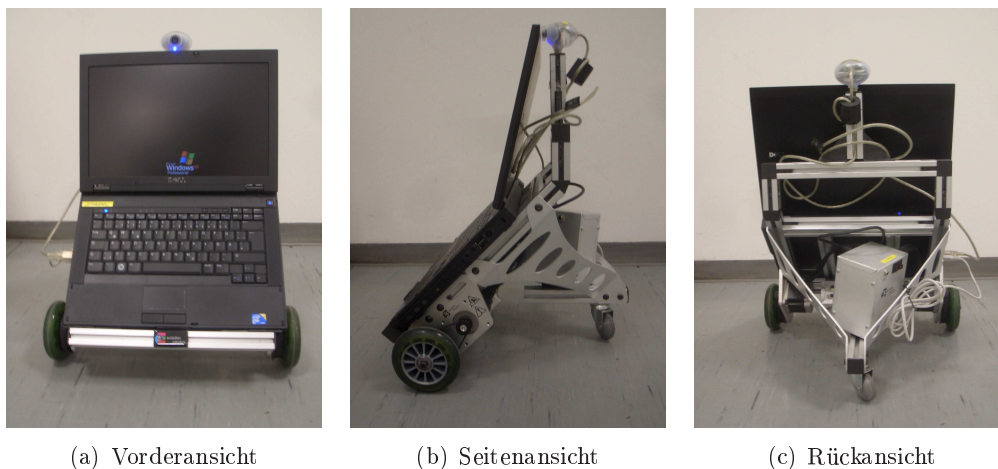


Abbildung 5.6: Unterschiedliche Ansichten des eingesetzten ER1 Roboters der Firma Evolution Robotics.

Für die Steuerung wurde ein Notebook mit folgenden Spezifikationen eingesetzt:

- Intel(R) Pentium(R) M Prozessor, 1.60GHz und 512Mb Arbeitsspeicher.

Die Kamera wurde im Rahmen der Experimente in Fahrtrichtung des Roboters gerichtet. Dabei wurde die folgende Kamera verwendet.

- iRez Kitter USB Webcam, Auflösung  $320 \times 240$

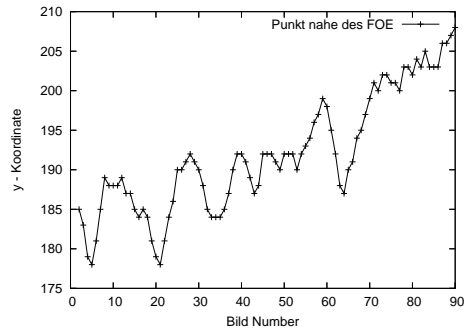
Vor der Inbetriebnahme wurde die Kamera mittels des 'ERSP Camera Calibration Tool' intrinsisch kalibriert. Dabei wurden insgesamt 7 von der Kameraposition unabhängige Kameraparameter bestimmt, mittels derer es möglich war das Kamerabild zu entzerren. Gerade Linien wurden im Bild wieder als gerade Linien dargestellt.

Aufgrund der Bodenbeschaffenheit im Labor und der eher instabilen Befestigung der Kamera, wurden während der Fahrt und besonders beim Start Schwankungen in Kamerabildern beobachtet. Um den Ausmaß der Schwankungen zu demonstrieren wurde während der Annäherung des Roboters an ein Stativ eine Videosequenz

aufgenommen. Der Stativ wurde dabei vor dem Roboter in einer Entfernung von  $7,75m$  platziert. Der höchste Punkt des Stativs im Bild entsprach in etwa der Position des FOE. In der aufgenommenen Videosequenz wurde dieser Punkt in jedem Bild manuell markiert. Abbildung 5.7 (a) zeigt das am Anfang der Sequenz aufgenommene Kamerabild und Abbildung 5.7 (b) zeigt die Entwicklung der  $y$ -Koordinate des angesprochenen Punktes während der gesamten Sequenz.



(a)



(b)

Abbildung 5.7: (a) zeigt das am Anfang der Sequenz aufgenommene Kamerabild; (b) zeigt die Entwicklung der  $y$ -Koordinate eines Punktes nahe dem FOE während der gesamten Sequenz.

Es ist deutlich zu sehen, dass fast während der gesamten Sequenz sehr starke Schwankungen in der Entwicklung der  $y$ -Koordinate des betrachteten Punktes vorhanden sind. Erst gegen Ende lässt sich eine Tendenz nach oben erkennen. In der  $x$ -Koordinate wurden ebenfalls Schwankungen beobachtet. Diese waren bei dem betrachteten Punkt allerdings nicht so stark. Wie bereits erwähnt, wurden die Positionen des Punktes in der Sequenz manuell markiert. Damit sind die gezeigten Schwankungen allein auf Bodenunebenheiten und instabile Befestigung der Kamera zurückzuführen.

Bei den im Folgenden präsentierten Experimenten wurde, wenn nicht anderes vermerkt, die Position des FOE vor dem Start der Roboterbewegung manuell kalibriert. Dies hatte zur Folge, dass beim Rauscheinfluss nicht explizit zwischen den Messfehlern bei den Positionen der Merkmale und den äußeren Einflüssen der Umgebung unterschieden wurde. Es wurde damit in den meisten Fällen der kumulierte Rauscheinfluss betrachtet.

### 5.3 Einfluss der Position der Merkmale im Bild

Bei den ersten durchgeführten Experimenten wurde der Roboter mit einer konstanten Geschwindigkeit vorwärts bewegt. Dabei wurde als erstes festgestellt, dass der

Anfang der Merkmalssequenz und die Richtung in die sich diese entwickelt einen starken Einfluss auf die Güte der entsprechenden TTC-Berechnung haben.

Abbildung 5.8 (a) zeigt ein vom Roboter während der Fahrt aufgenommenes Bild. Der rot umrandete Rechteck markiert den Bereich aus dem einige SIFT-Merkmale für die TTC-Berechnung ausgewählt wurden. Abbildung 5.8 (b) zeigt die Entwicklung der entsprechenden TTC-Werte. Dabei entspricht die gerade durch-

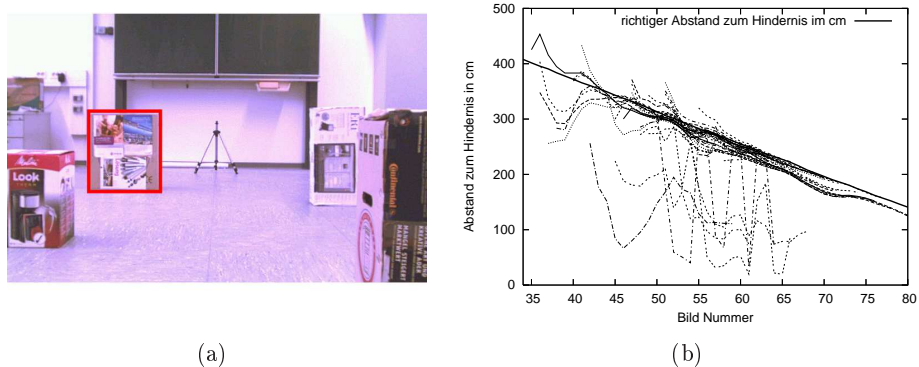


Abbildung 5.8: (a) Der rot umrandete Rechteck markiert den Bereich aus dem einige Merkmale für die TTC-Berechnung ausgewählt wurden. (b) Graphische Darstellung der bereits in Distanz umgerechneten TTC-Werte.

gezogene Linie mit der negativen Steigung der wahren Distanz zum betrachteten Hindernis. Die anderen Kurven entsprechen den in Distanz umgerechneten TTC-Werten, welche aus den jeweiligen Merkmalssequenzen berechnet wurden. Es ist deutlich zu sehen, dass einige der berechneten Kurven sich schon nach kurzer Zeit den richtigen Werten annähern, während andere weit von den richtigen Werten entfernt, starken Schwankungen unterliegen. Genauere Untersuchungen haben ergeben, dass das Problem in der unglücklichen Initialisierung der Modellparameter  $a_x$ ,  $a_y$ ,  $b_x$  und  $b_y$  liegt. Davon betroffen waren hauptsächlich Punkte, die sich entlang der  $x$ - oder der  $y$ - Achse bewegten. Messfehler in den Positionen der Merkmale hatten hier einen stärkeren Wirkungsgrad. Tabelle 5.1 gibt hierzu einen Überblick über die initialen Werte der Parameter  $a_x$ ,  $a_y$ ,  $b_x$  und  $b_y$  einiger guter und einiger schlechter Sequenzen. Die Güte einer Sequenz charakterisiert sich über die Genauigkeit der entsprechenden TTC-Werte. Die in der Tabelle dargestellten Werte entsprechen den ersten beiden Positionen  $p_1$  und  $p_2$  eines Merkmals in der entsprechenden Sequenz. Es gilt  $p_1 = (a_x, a_y)$  und  $p_2 = (b_x, b_y)$ .

In dem hier untersuchten Beispiel liegt der Verdacht nahe, dass die Ursache für die instabile TTC-Werte bei schlechten Sequenzen in dem zu geringen Betrag der  $a_y$  und  $b_y$  Parameter liegt. Diese Sequenzen bewegten sich entlang der  $x$ -Achse. Weitere Untersuchungen haben allerdings ergeben, dass das Problem viel mehr auf den

schlechte Sequenzen				gute Sequenzen			
$a_x$	$b_x$	$a_y$	$b_y$	$a_x$	$b_x$	$a_y$	$b_y$
-169.07	-171.25	5.22	4.62	-206.53	-209.37	31.16	33.01
-190.42	-193.56	-3.67	-4.76	-191.83	-195.64	12.08	12.82
-200.18	-203.26	-0.66	-3.86	-186.89	-189.94	15.82	15.72
-169.28	-172.43	0.16	-3.05	-166.87	-169.69	10.95	10.89
-177.28	-181.04	-0.52	-0.79	-173.95	-177.10	29.83	29.36
-169.01	-172.28	0.19	0.20	-259.23	-263.90	-15.61	-16.65

Tabelle 5.1: Initiale Parameter  $a_x$ ,  $a_y$ ,  $b_x$  und  $b_y$  einiger ausgewählter guter und schlechter Sequenzen, die bei dem in Abbildung 5.8 gezeigten Versuch entstanden sind.

prozentualen Unterschied zwischen den Parametern zurückzuführen ist. Die bereits hergeleiteten Formeln zur Vorhersage der nächsten Merkmalsposition

$$x(n) = \frac{a_x \cdot b_x}{((n-1) \cdot a_x - (n-2) \cdot b_x)}, \quad y(n) = \frac{a_y \cdot b_y}{((n-1) \cdot a_y - (n-2) \cdot b_y)}$$

liefern nur dann richtige Ergebnisse, wenn die folgenden Bedingungen erfüllt sind:

$$|(n-1) \cdot a_x| > |(n-2) \cdot b_x|, \quad |(n-1) \cdot a_y| > |(n-2) \cdot b_y|$$

Mit diesen Bedingungen ist sichergestellt, dass die Nenner in den obigen Gleichungen positiv sind. Experimentell wurde ermittelt, dass im anderen Fall die ausgerechneten Vorhersagen nicht zuverlässig sind. Demnach muss für die initialen Parameter  $a_y$  und  $b_y$  der Sequenzen folgendes gelten:

$$\frac{(n-1)}{(n-2)} \cdot a_y > b_y \quad \Rightarrow \quad 2 \cdot a_y > b_y$$

wobei hier  $n = 3$  ist. Entsprechendes gilt auch für die Parameter  $a_x$  und  $b_x$ . Der prozentuale Unterschied zwischen den Parametern muss damit kleiner als 50% sein. Ist dies nicht der Fall, so schafft das Verfahren kaum diesen initialen Fehler zu korrigieren.

Experimentell wurde herausgefunden, dass bei guten Sequenzen die Unterschiede zwischen beiden Parameterpaaren  $(a_x, b_x)$  bzw.  $(a_y, b_y)$  bei etwa 2% liegen. Damit war es möglich schlecht initialisierte Sequenzen von vorneherein zu identifizieren. Dazu wurde eine Schwelle auf 10% gesetzt. War der prozentuale Unterschied in einem der Parameterpaare größer, so wurde die Sequenz als 'schlecht initialisiert' markiert. Variationen dieses Schwellenwertes hatten allerdings kaum einen Einfluss auf die Performanz des Verfahrens. In dem untersuchten Beispiel erfüllen die ersten

vier der insgesamt sechs betrachteten schlechten Sequenzen diese Bedingung nicht. In den beiden anderen Fällen liegt der Betrag der beiden Parameter  $a_y$  und  $b_y$  unter 1.0. In solchen Fällen wurden die Sequenzen ebenfalls als 'schlecht initialisiert' markiert.

Weitere Experimente haben ergeben, dass bei den meisten als 'schlecht' markierten Sequenzen immer nur ein Parameterpaar für das schlechte Ergebnis verantwortlich war. Das jeweils andere Parameterpaar hatte dagegen sehr gute Initialisierungswerte. Wie der Tabelle 5.1 zu entnehmen ist, erfüllen auch in dem untersuchten Beispiel die Parameter  $a_x$  und  $b_x$  der schlechten Sequenzen die oben formulierte Bedingung.

Berücksichtigt man nun bei der Berechnung der TTC-Werte nur das gut initialisierte Parameterpaar, so lassen damit sehr gute Ergebnisse erzielen. Bei dem betrachteten Beispiel wurde bei den schlechten Sequenzen für die Berechnung der TTC-Werte nur die  $x$ -Koordinate der Merkmalspositionen herangezogen. Abbildung 5.9 zeigt die alten und die neuen in Distanz umgerechneten TTC-Werte für Sequenzen, die vorher als 'schlecht initialisiert' markiert wurden. Eine Steigerung der

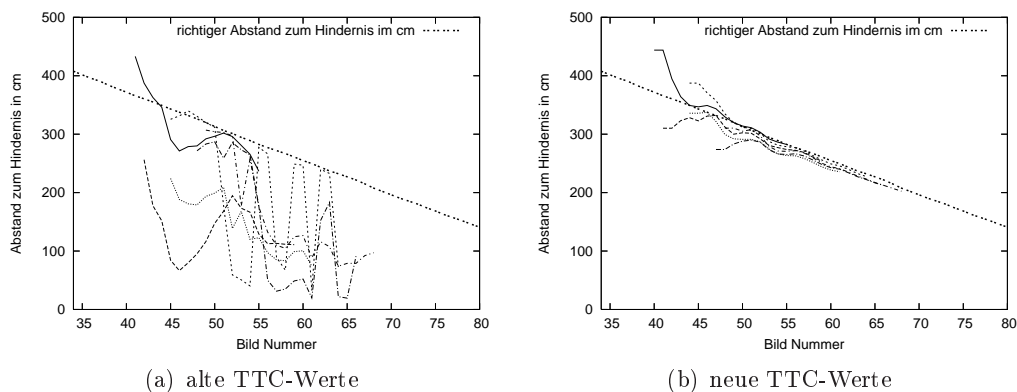


Abbildung 5.9: In Distanz umgerechneten TTC-Werte: (a) bei der Berechnung wurden beide Parameterpaare  $(a_x, b_x)$  und  $(a_y, b_y)$  berücksichtigt; (b) bei der Berechnung wurde nur das gut initialisierte Parameterpaar berücksichtigt.

TTC-Genauigkeit ist bei allen Sequenzen deutlich zu erkennen. Schon nach wenigen Schritten nähern sich bei allen Sequenzen die berechneten TTC-Werte den wahren Werten an.

Mit der Überprüfung der Güte der Initialisierungsparameter der Sequenzen und die damit verbundene Berücksichtigung oder Nichtberücksichtigung der entsprechenden Koordinaten bei der Berechnung der TTC-Werte, konnte eine entscheidende Verbesserung der Performanz des Verfahrens erzielt werden.

## 5.4 Einfluss der Messungengenauigkeit der Merkmale

Die Genauigkeit mit der die Merkmale im Bild gemessen werden hat ebenfalls einen primären Einfluss auf die Robustheit der Schätzung der TTC-Werte. Diese Beobachtungen wurden auch bei den Experimenten in der Simulationsumgebung bestätigt. Hier wurden Messfehler unter Verwendung des Gaußschen Rauschens modelliert. In den praktischen Anwendungen ist der Fehler bei den Positionsmessungen der Merkmale von der Wahl derjenigen abhängig. Für unterschiedliche Anwendungen wurden in den letzten Jahren eine breite Palette an verschiedenen Merkmalen und entsprechenden Verfolgungsverfahren entwickelt und getestet. Eine gute Übersicht darüber findet sich in [MS05]. Im Rahmen dieser Arbeit wurde untersucht, welches der folgenden zwei Merkmale sich besser für das Verfahren der TTC-Berechnung eignet:

- Scale Invariant Feature Transform, SIFT-Merkmale.
- Kanade-Lucas-Tomasi Features, KLT-Merkmale.

Eine kurze Einführung in die Eigenschaften und die damit verbundenen Vorteile der SIFT-Merkmale hat bereits im Kapitel 3 dieser Arbeit im Abschnitt 'Einfache Methode zur Objektverfolgung' stattgefunden. Es sei an dieser Stelle noch die Existenz des SURF-Merkmals (Speeded Up Robust Features) [BTG06] angemerkt. Diese Merkmale liefern qualitativ ähnlich gute Ergebnisse wie SIFT, sind allerdings etwas leichter in dem Berechnungsaufwand. Im Rahmen dieser Arbeit wurden diese Merkmale nicht untersucht, denn wie sich später zeigen wird, eignen sich die KLT-Merkmale ebenfalls für die Verwendung in dem Verfahren. Der Berechnungsaufwand der KLT-Merkmale liegt deutlich unter dem für SURF- und SIFT-Merkmal, die Qualität der berechneten TTC-Werte steht allerdings im nichts nach.

Die Extraktion und Verfahren zur robusten Verfolgung von KLT-Merkmalen gehen auf die Arbeiten von Bruce D. Lucas, Takeo Kanade und Carlo Tomasi zurück [ST94], [TK91]. Die einzelnen Merkmale werden aufgrund der Charakteristika der lokalen Nachbarschaft des Merkmals ausgewählt. Dabei werden zunächst die Eigenwerte der zuvor berechneten  $2 \times 2$  Gradientenmatrix bestimmt. Liegen beide Eigenwerte über einem bestimmten Grenzwert, so deutet dies auf eine Ecke im Bild hin. Merkmale, die eine Ecke repräsentieren eignen sich besonders gut zur Korrespondenzfindung im nachfolgenden Bild. Die Verfolgung ausgewählter Merkmale erfolgt dabei mittels der Newton-Raphson Methode [DH02]. Beim Einsatz von 'multiresolution tracking' können auch relativ große Verschiebungen zwischen den Bildern bewältigt werden.

### Versuchsdurchführung

Um den Einfluss beider Merkmale auf die TTC-Berechnung miteinander vergleichen zu können, wurde zunächst sichergestellt, dass die entsprechenden Berechnungen auf der jeweils gleichen Bildsequenz durchgeführt wurden. Dazu wurden zunächst, mit der auf dem Roboter angebrachten Kamera während der unterschiedlichen Roboterbewegungen mehrere Videosequenzen aufgenommen. Die Odometriedaten des Roboters, sowie die genauen Zeitpunkte der Bildaufnahmen wurden dabei für die spätere Verwendung in einer separaten Datei gespeichert. Die eigentliche Berechnung der TTC-Werte erfolgte dann offline.

Zur Aufnahme der Sequenzen wurde das Labor entsprechend der Beschreibung im Abschnitt 'Versuchsaufbau' präpariert. Für die Evaluierung wurden Merkmale auf dem zweiten Hindernis auf der linken Seite ausgewählt. Dieses Objekt ist in der Abbildung 5.8 (a) rot umrandet. Um jederzeit den richtigen Abstand des Roboters zum Hindernis berechnen zu können, wurden zur Vereinfachung die Bewegungen des Roboters auf eine Geradeausfahrt beschränkt. Dieses Experiment wurde mehrere Male wiederholt. Die Geschwindigkeit des Roboters wurde dabei wie folgt variiert:  $2\text{ cm/s}$ ,  $4\text{ cm/s}$ ,  $6\text{ cm/s}$  und  $8\text{ cm/s}$ .

Bei der offline Berechnung der TTC-Werte wurde der FOE zunächst konstant in der Bildmitte angenommen. Damit war sichergestellt, dass der so hinzugefügte Fehler unabhängig von den aktuell verwendeten Merkmalen war. Die so erzielten Ergebnisse erlaubten einen besseren Vergleich, denn damit war der 'variable' Fehleranteil allein auf die Messfehler bei den Merkmalspositionen zurückzuführen. Die Ergebnisse dieser Berechnungen, sowie die zur Bewertung herangezogenen Bewertungskriterien werden im Folgenden ausführlich beschrieben.

### Bewertungskriterien

Um qualitative Aussagen über die Güte der berechneten TTC-Werte treffen zu können, wurden zwei unterschiedliche Bewertungskriterien entwickelt. Beide Kriterien zielen darauf ab, die Güte der berechneten TTC-Werte mathematisch zu erfassen. Anhand der in Abbildung 5.10 beispielhaft gezeigten Ergebnisse der TTC-Berechnung werden diese im Folgenden erklärt:

**Kriterium 1:** Histogramm der Steigungen der berechneten TTC-Sequenzen. In Abbildung 5.10 können Sequenzen beobachtet werden, die sich zwar den richtigen TTC-Werten annähern, doch im Gesamtverlauf eine falsche Richtung aufweisen. Die 'Richtung' der Sequenz wird hier durch die Steigung der Geraden definiert, die die Werte der jeweiligen TTC-Sequenz bilden.



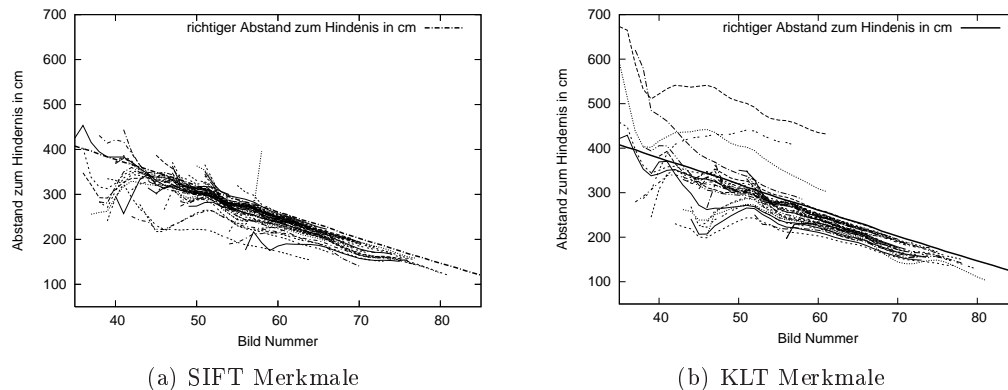


Abbildung 5.10: In Distanz umgerechnete TTC-Werte verschiedener Merkmalssequenzen mit dem gleichen Abstand zum Hindernis. Als Merkmal wurden (a) SIFT-Merkmale und (b) KLT-Merkmale verwendet.

Zur Berechnung dieser Geraden wurde zunächst die 'Methode der kleinsten Quadrate' [MT77] verwendet. Mit diesem Verfahren wird eine Gerade bestimmt, so dass der akkumulierte Abstand aller Punkte der TTC-Sequenz zu dieser Gerade minimal ist. Dieses Verfahren hat sich hier allerdings nicht bewährt. Aufgrund der Adaptionsphase und der damit stark schwankenden TTC-Werten am Anfang jeder Sequenz spiegelten die berechneten Geraden die eigentliche Richtung der Sequenz nicht richtig wieder. Bessere Ergebnisse konnten stattdessen mit dem RANSAC-Algorithmus (Random Sample Consensus) [FB87] erzielt werden. Dies ist ein iteratives Verfahren und verfolgt eine andere Strategie. Zuerst werden zufällig zwei Punkte der entsprechenden TTC-Sequenz ausgewählt und eine Gerade durch diese bestimmt. Im nächsten Schritt wird überprüft, wie viele der restlichen Punkte derselben Sequenz einen Abstand zu dieser Gerade haben, der unter einem bestimmten Schwellenwert liegt. Die Menge dieser Punkte wird als 'consensus set' bezeichnet. Nach mehrmaliger Wiederholung der beiden Schritte wird letzten Endes die Gerade ausgewählt, die den größten 'consensus set' hat. Damit ist der RANSAC-Algorithmus sehr stabil gegenüber Ausreißern, die wie der Abbildung 5.10 zu entnehmen ist hauptsächlich am Anfang jeder TTC-Sequenz vorhanden sind.

**Kriterium 2:** Histogramm der Abweichungen der berechneten TTC-Werten von den richtigen Werten.

Zur Berechnung des Histogramms wird zunächst ein schmaler Schlauch um die Gerade gelegt, die die richtigen TTC-Werte repräsentiert. In der Abbildung 5.10 ist dies die steil nach unten abfallende Gerade. Im nächsten Schritt wird bestimmt, bei wie vielen Sequenzen mehr als die Hälfte der jeweiligen TTC-Werte in diesem Schlauch

liegen. Das vollständige Histogramm wird erzeugt, indem die Breite des Schlauchs immer weiter vergrößert wird. In Abbildung 5.10 (a) würden ab einer Breite des Schlauchs von etwa 50 cm kaum weitere Sequenzen hinzukommen. In Abbildung (b) dagegen würden auch ab einer Breite von etwa 100 cm noch weitere Sequenzen hinzukommen. Damit erlaubt dieses Histogramm einen schnellen Überblick über die Anzahl der Ausreißer. In diesem Fall sind es TTC-Sequenzen, die stark von den richtigen TTC-Werten abweichen. Eine andere Idee für die Erzeugung des Histogramms war, die Breite des Schlauchs festzuhalten und den Schlauch entsprechend nach oben und nach unten zu verschieben. Es hat sich allerdings herausgestellt, dass abhängig von der Steigung des Schlauchs einige TTC-Sequenzen nicht erfasst wurden, weil aufgrund ihrer Richtung zu keinem Zeitpunkt mehr als die Hälfte der entsprechenden TTC-Werte in diesem Schlauch lagen.

### Diskussion der Ergebnisse

Die Evaluierung der Ergebnisse führte zu der Schlussfolgerung, dass keines der untersuchten Merkmale dem anderen in der Genauigkeit der berechneten TTC-Werte überlegen war. Die folgenden Abbildungen 5.11, 5.12, 5.13 und 5.14 zeigen die auf den resultierenden TTC-Sequenzen berechneten Histogramme bei unterschiedlichen Geschwindigkeiten des Roboters. Abbildungen (a) und (b) zeigen die Ergebnisse der beiden Bewertungskriterien für die KLT-Merkmale, und Abbildungen (c) und (d) zeigen die entsprechenden Graphiken für SIFT-Merkmale.

Unabhängig von der Geschwindigkeit des Roboters lassen sich in den gezeigten Graphiken kaum Unterschiede feststellen. Sowohl bei KLT als auch bei SIFT-Merkmalen weist das Histogramm der Steigungen eine gewisse Streuung um einen leicht erkennbaren Pik herum. Weil bei den Berechnungen der entsprechenden Geraden mittels RANSAC auch ein Zufallsfaktor vorhanden ist, ist dieser Pik nicht immer deutlich ausgeprägt. Beim mehrmaligen Starten des RANSAC-Algorithmuses auf den gleichen TTC-Sequenzen wurden allerdings nur kleinere Änderungen der groben Struktur des Steigungshistogramms beobachtet.

Wie den Abbildungen zu entnehmen ist, nimmt die Ausprägung des erwähnten Piks bei höheren Geschwindigkeiten deutlich ab. Dies ist dadurch zu erklären, dass bei höheren Geschwindigkeiten sowohl die Anzahl als auch die Länge der entsprechenden TTC-Sequenzen immer stärker abnimmt. Aufgrund der zur Verfügung stehenden Laborgröße, wurde vom Roboter bei allen Experimenten die gleiche Strecke zurückgelegt. Bei erhöhter Geschwindigkeit wurden auf dieser Strecke zum einen weniger Bilder aufgenommen, zum anderen war die Verschiebung zwischen zwei aufeinander folgenden Bildern größer als dies bei kleineren Geschwindigkeiten der Fall

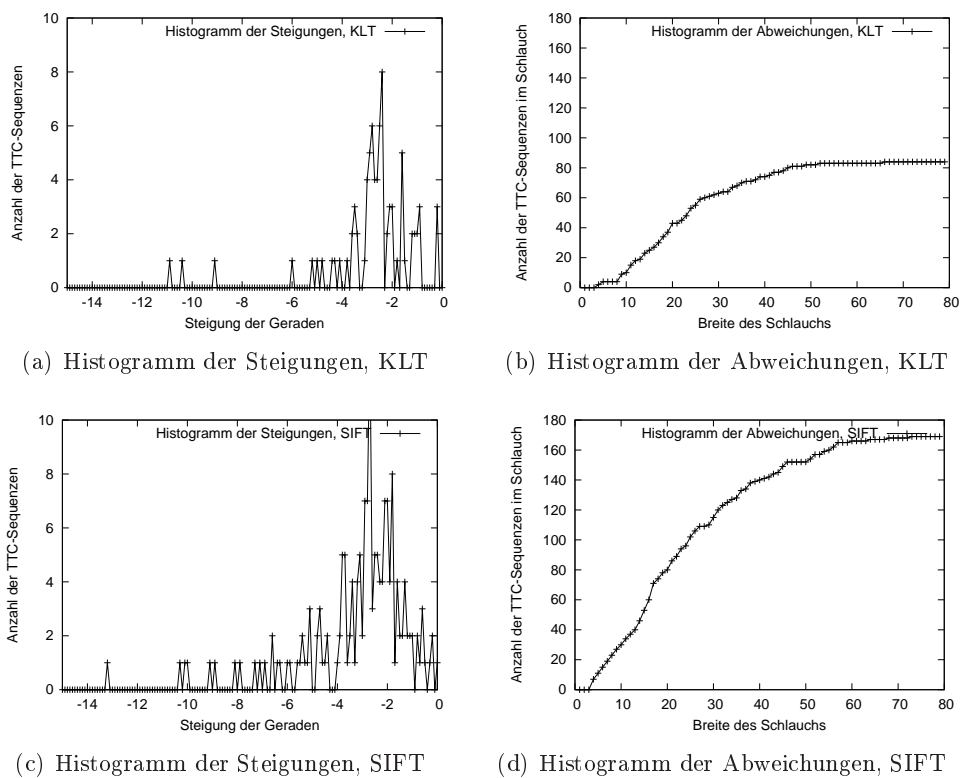
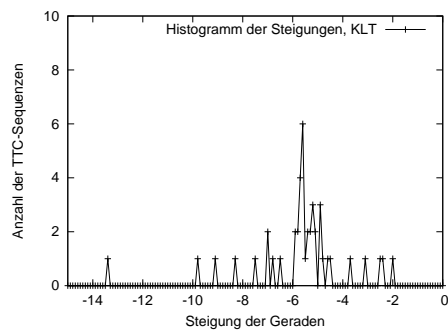
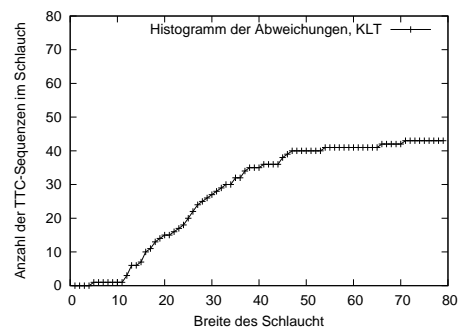


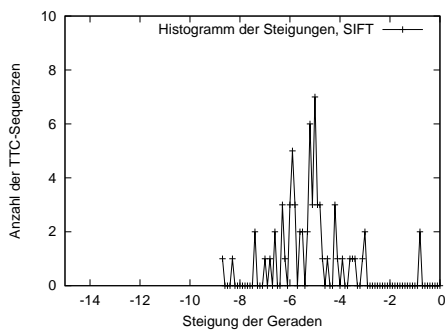
Abbildung 5.11: Resultierende Histogramme der beiden Kriterien bei einer Robotergergeschwindigkeit von  $2\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale; (c) und (d) sind Histogramme für SIFT-Merkmale.



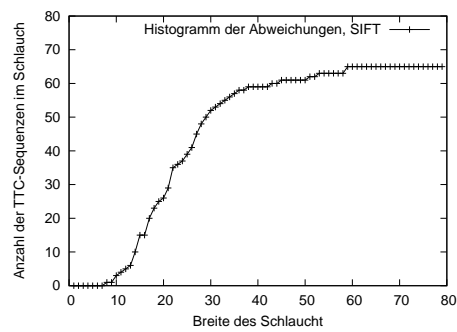
(a) Histogramm der Steigungen, KLT



(b) Histogramm der Abweichungen, KLT



(c) Histogramm der Steigungen, SIFT



(d) Histogramm der Abweichungen, SIFT

Abbildung 5.12: Resultierende Histogramme der beiden Kriterien bei einer Roboter-  
 geschwindigkeit von  $4\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale;  
 (c) und (d) sind Histogramme für SIFT-Merkmale.

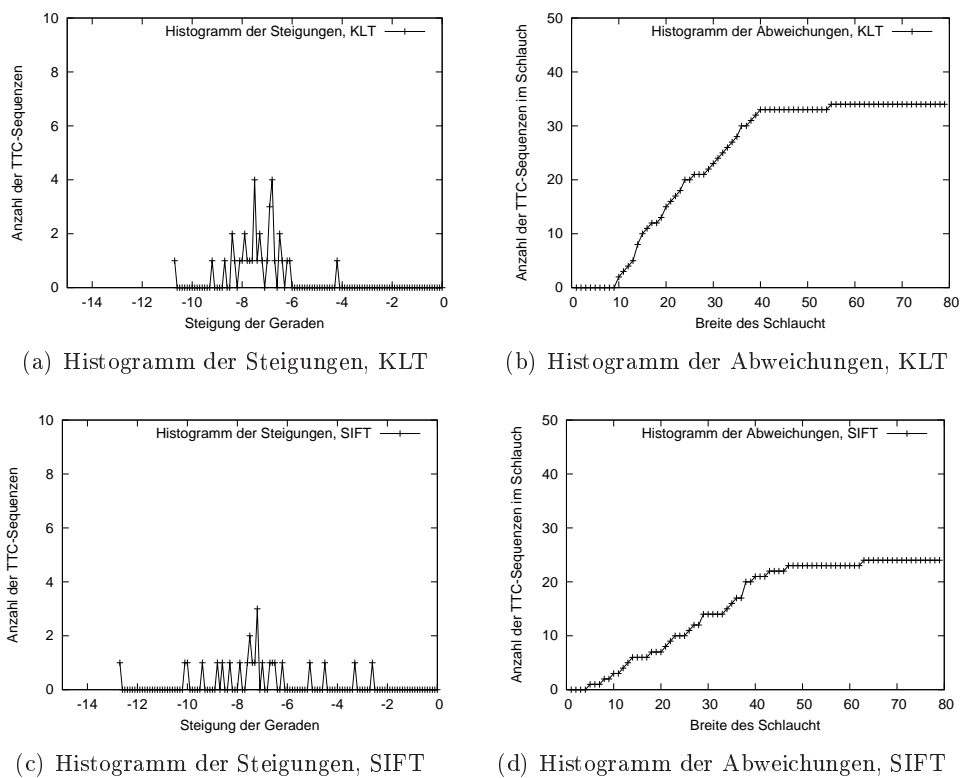
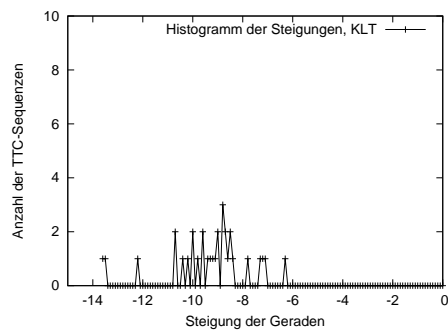
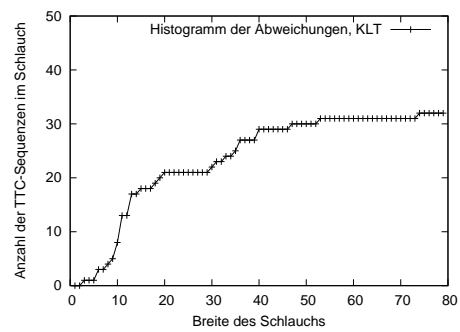


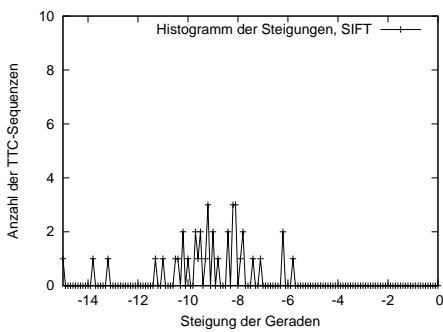
Abbildung 5.13: Resultierende Histogramme der beiden Kriterien bei einer Robotergergeschwindigkeit von  $6\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale; (c) und (d) sind Histogramme für SIFT-Merkmale.



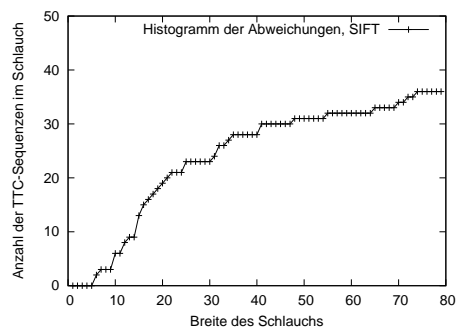
(a) Histogramm der Steigungen, KLT



(b) Histogramm der Abweichungen, KLT



(c) Histogramm der Steigungen, SIFT



(d) Histogramm der Abweichungen, SIFT

Abbildung 5.14: Resultierende Histogramme der beiden Kriterien bei einer Roboter-  
 geschwindigkeit von  $8\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale;  
 (c) und (d) sind Histogramme für SIFT-Merkmale.

war. Der größere Unterschied zwischen den aufgenommenen Bildern sorgte dafür, dass immer weniger Merkmale von Bild zu Bild robust verfolgt werden konnten.

Die immer kleiner werdende Anzahl der betrachteten TTC-Sequenzen lässt sich auch an den Ergebnissen des zweiten Kriteriums ablesen. Der maximal erreichte Wert in diesen Histogrammen nimmt bei höheren Geschwindigkeiten ab. Während bei SIFT-Merkmalen und einer Geschwindigkeit von  $2\text{ cm/s}$  dieser bei etwa 170 liegt, sind es bei einer Geschwindigkeit von  $8\text{ cm/s}$  nur noch 37 Sequenzen, die erfolgreich konstruiert wurden. Ähnliche Tendenz lässt sich auch bei KLT-Merkmalen beobachten.

Ebenfalls leicht zu erkennen ist die Verschiebung des erwähnten Piks in den Steigungshistogrammen. Mit höheren Geschwindigkeiten wandert dieser immer weiter nach links. Der Grund dafür ist, dass die berechneten TTC-Werte bei höheren Geschwindigkeiten schneller abnehmen und damit auch der Betrag der Steigung der entsprechenden Gerade höher ist.

Die eindeutige Verschiebung des Piks spricht dafür, dass bei allen Experimenten unabhängig von der Art der verwendeten Merkmale alle berechneten TTC-Sequenzen sich ungefähr in die richtige Richtung entwickelten. Weil bei der Berechnung der entsprechenden Geraden die Sequenzen als Ganzes betrachtet wurden, kann daraus auch geschlossen werden, dass die TTC-Sequenzen relativ glatt waren und keinen starken Schwankungen unterlagen. Im anderen Fall würden die Steigungen der berechneten Geraden stark streuen.

Die Histogramme des zweiten Kriteriums zeigen anschaulich wie stark die berechneten TTC-Sequenzen von den richtigen Werten abweichen. Während der durchgeführten Experimente wurden oft TTC-Sequenzen beobachtet, die über einen längeren Zeitraum einen immer gleichen Abstand von der richtigen TTC-Gerade hatten. Dabei wurde keine Korrelation zwischen der Steigung der Sequenz und dem entsprechenden Abstand festgestellt.

Wie den Abbildungen 5.11, 5.12, 5.13 und 5.14 zu entnehmen ist, gab es bei dieser Experimentenreihe nur wenige Sequenzen, die im Schlauch der Breite  $< 10$  lagen. Dies ist dadurch zu erklären, dass bei der Berechnung des Histogramms eine Sequenz nur dann als 'im Schlauch liegend' markiert wurde, wenn mehr als die Hälfte der entsprechenden TTC-Werte im Schlauch lagen. Dabei ist zu beachten, dass abhängig von der Geschwindigkeit des Roboters die durchschnittliche Länge einer TTC-Sequenz bei etwa 25 Werten liegt und die Länge der anfänglichen Adaptionsphase bei etwa 13 – 17 Werten. Sequenzen, die nach den üblichen starken Schwankungen in der Adaptionsphase anfangen sich den richtigen Werten anzunähern, fallen entsprechend der obigen Bedingung aus dem engeren Schlauch heraus.

Wie den Abbildungen des zweiten Kriteriums ebenfalls zu entnehmen ist, liegt

der Abschnitt mit der größten Steigung des Histogramms im Bereich 10 – 25 der Schlauchbreite. Diese Tatsache weist darauf hin, dass die meisten Sequenzen innerhalb dieses Intervalls von der richtigen TTC-Geraden abweichen. Allerdings wurde bei den Experimenten auch festgestellt, dass die meisten berechneten TTC-Sequenzen sich während des ganzen Prozesses immer weiter der richtigen Gerade annähern. Zwar sind dabei keine Schwankungen zu beobachten, doch resultiert diese stetige Annäherung oft in einem höheren Betrag der Steigung der entsprechenden Gerade. Dies erklärt auch die leichte Streuung in den Histogrammen des ersten Kriteriums.

Die Abweichung der berechneten TTC-Sequenzen von den richtigen Werten lässt sich auf die unebene Fahrbahn zurückführen. Wie bereits im Abschnitt 'Versuchsaufbau' erklärt, wurden während der Roboterfahrt regelmäßig leichte Kameraschwankungen beobachtet. Der vorher manuell bestimmte FOE stimmte damit nicht immer mit der aktuell richtigen Position desjenigen überein. Neben den Fehlern in den Messungen der Merkmalspositionen, haben diese Störungen ebenfalls einen Einfluss auf die Berechnung der TTC-Werte.

Die in diesen Experimenten erzielten Ergebnisse zeigen allerdings, dass das angewendete Verfahren sehr gut mit diesen Störungen umgehen kann. Nach dem Abschluss der Adaptionsphase zeigten alle TTC-Sequenzen einen glatten Verlauf und näherten sich dabei den richtigen Werten an. Die Anwendung des einfachen Verfahrens zur Berechnung der TTC-Werte konnte bei diesen Experimenten, unter solch starken Rauscheinflüssen in keinsten Weise mit dem hier entwickelten Verfahren konkurrieren. Zu keinem Zeitpunkt der Bildaufnahme wäre es bei dem Standardverfahren möglich gewesen irgendeine Aussage darüber zu treffen, wie weit der Roboter sich nun vor dem Objekt befindet.

Unabhängig von der Güte der berechneten TTC-Sequenzen zeigen die obigen Abbildungen 5.11, 5.12, 5.13 und 5.14 ebenfalls, dass die Art der verwendeten Merkmale kaum Einfluss auf die erzielten Ergebnisse hat. Die Histogramme beider Kriterien sehen bei KLT- und SIFT-Merkmalen sehr ähnlich aus. Daher wurde in den nachfolgenden Experimenten auf die Verwendung der SIFT-Merkmale verzichtet. Im Unterschied zu den KLT-Merkmalen ist der Berechnungsaufwand für die SIFT-Merkmale sehr hoch und steht in keinem Verhältnis zu der mit diesem Merkmal erzielten Güte der TTC-Sequenzen.

Der nachfolgende Abschnitt beschreibt Ergebnisse, die bei einer variierenden Roboterbewegung beobachtet wurden.



## 5.5 Ergebnisse im Rahmen einer Laborumgebung

Um die Praxistauglichkeit der in diesem Kapitel vorgestellten Erweiterungen des Ansatzes für die variierende Roboterbewegung zu überprüfen, wurde im Rahmen dieser Arbeit eine weitere Experimentenreihe durchgeführt. Dabei wurde untersucht, wie gut der Roboter den bevorstehenden Hindernissen ausweichen kann.

Anders als im vorigen Abschnitt wurde hier die Güte der erzielten Ergebnisse anhand der Dauer der kollisionsfreien Roboterbewegung festgemacht. Die Evaluierung der Performanz anhand der Abweichungen der berechneten TTC-Werte von den richtigen Werten erwies sich in dieser Situation als nicht durchführbar. Anders als bei einer Geradeausfahrt des Roboters konnte bei einer rotierenden Bewegung die zur Evaluierung erforderliche Genauigkeit der Odometriedaten des Roboters nicht vorausgesetzt werden. Damit war es nicht möglich den genauen Winkel der Kamerabildebene zum bevorstehenden Hindernis zu ermitteln. Auch aufgrund des online Betriebs konnte nicht von vorn herein festgelegt werden welche Merkmale zur Berechnung der TTC-Werte vom Verfahren ausgewählt werden.

Zusätzlich ist auch zu beachten, dass die Verfahren zur Adaption der Parameter mehrere Bildaufnahmen brauchen. Bei einer rotierenden Bewegung können allerdings nicht beliebig viele Bilder zur Verfügung gestellt werden. Im Gegenteil die Anzahl der brauchbaren Bilder ist sehr begrenzt. Bei einer geringen Rotationsgeschwindigkeit können zwar mehr Bilder aufgenommen werden, bis ein betrachteter Punkt aus dem Sichtbereich der Kamera verschwindet, doch reicht diese Anzahl an Bildern oft nicht aus um aussagekräftige Resultate zu erzielen.

Es stellt sich auch die Frage in wie fern eine erneute Evaluierung der Verfahren basierend auf den TTC Abweichungen notwendig ist. Betrachtet man noch einmal die hier vorgestellten Erweiterungen, so stellt man fest dass alle drei Ansätze in ihrem Kern das Adaptionsverfahren für die Geradeausfahrt mit konstanter Geschwindigkeit beinhalten. Die einzigen zusätzlichen Fehlerquellen bei diesen Ansätzen sind die Odometriedaten des Roboters und die eventuell fehlerhafte Kalibrierung der Kamera am Anfang der Experimentenreihe. Damit lässt sich die Praxistauglichkeit dieser Ansätze auch anhand der Dauer der kollisionsfreien Roboterbewegung feststellen.

Die folgenden beiden Abschnitte geben einen kurzen Überblick unter welchen Bedingungen die Experimente durchgeführt und welche Ergebnisse dabei erzielt wurden.

### 5.5.1 Versuchsdurchführung

Zur Durchführung der Experimente wurde ein Teil des Flurs so präpariert, dass dieser einen schmalen geschlossenen Raum bildete. Die Form des Raumes wurde dabei

mit Bedacht ausgewählt. Der relativ kleine Abstand zwischen den zwei parallelen Wänden sorgte dafür, dass der Roboter kaum in der Lage war lange und gerade Strecken zu fahren. Während einer Fahrt in dem länglichen Flur musste er immer wieder den bevorstehenden Wänden ausweichen. Dabei führte er neben reinen Rotationsbewegungen auch Kurvenfahrten aus.

Die Abbildung 5.15 zeigt den maßstabgetreuen Grundriss des Flurs und eine entsprechende Fotoaufnahme. Die weißen Wände des Flurs wurden mit verschiede-

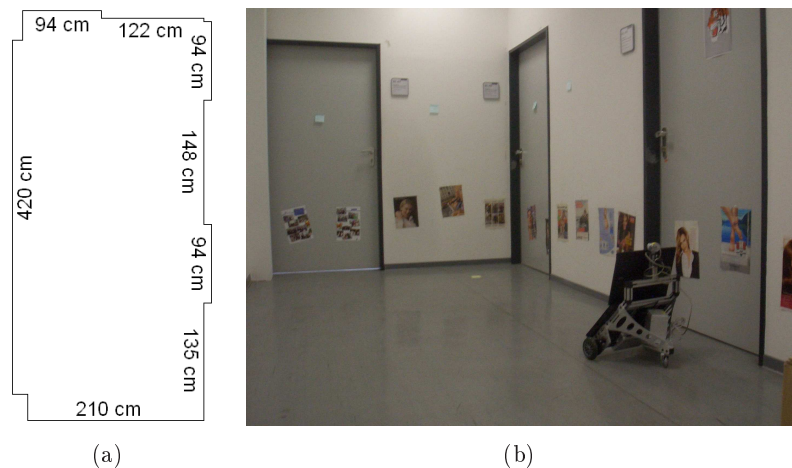


Abbildung 5.15: Für die Experimente verwendeter Flur: (a) maßstabgetreuer Grundriss; (b) Fotoaufnahme des Flurs

nen Bildausschnitten einer Zeitschrift beklebt. Dabei entspricht die Höhe in der die Bilder aufgebracht wurden in etwa der Höhe der Kamera auf dem Roboter. Diese Maßnahme war insofern notwendig, weil auf einem homogenen Hintergrund keine KLT-Merkmale extrahiert werden können. Ähnlich vielen anderen Verfahren der Merkmalsextraktion, ist auch hier eine der grundlegenden Bedingungen ein Bild auf dem zu mindestens etwas Struktur vorhanden ist.

Bei dieser Experimentenreihe wurde das gleiche Robotermodell verwendet, welches auch schon zu Anfang dieses Kapitels beschrieben wurden. Die Ausstattung des Roboters stimmte ebenfalls mit der obigen Beschreibung überein. Die Startposition und die anfängliche Orientierung des Roboters wurde bei jedem Versuch willkürlich gewählt. Die durchschnittliche Geschwindigkeit des Roboters betrug etwa  $4\text{cm/s}$ .

### 5.5.2 Ergebnisse der durchgeführten Experimente

Insgesamt wurden in dieser Experimentenreihe mehrere Versuche durchgeführt. Bei jedem Versuch wurde gemessen wie lange der Roboter sich kollisionsfrei im Flur bewegen kann. Gleichzeitig wurde auch gezählt wie oft der Roboter den bevorste-

henden Wänden ausgewichen ist. Die Tabelle 5.2 fasst die dabei erzielten Ergebnisse zusammen. Bei allen durchgeführten Versuchen war der Roboter in der Lage sich

	Dauer der Navigation	vermiedene Kollisionen
Versuch 1	über 20 min.	16
Versuch 2	über 20 min.	17
Versuch 3	über 20 min.	16

Tabelle 5.2: Ergebnisse der Roboternavigation im geschlossenen Raum

länger als 20 Minuten kollisionsfrei im Flur zu bewegen. Nach etwa 20 Minuten wurden die Versuche aufgrund der zu langen Dauer manuell abgebrochen. In dieser Zeit ist der Roboter im Durchschnitt etwa 16 mal den bevorstehenden Wänden ausgewichen. Ein dieser Arbeit beigefügtes fünf minütiges Video zeigt beispielhaft eine solche Roboterfahrt.

Die Abbildung 5.16 zeigt zusätzlich einige Ausschnitte der vom Roboter gefahrenen Trajektorien. Diese Graphiken wurden basierend auf den Odometriedaten des

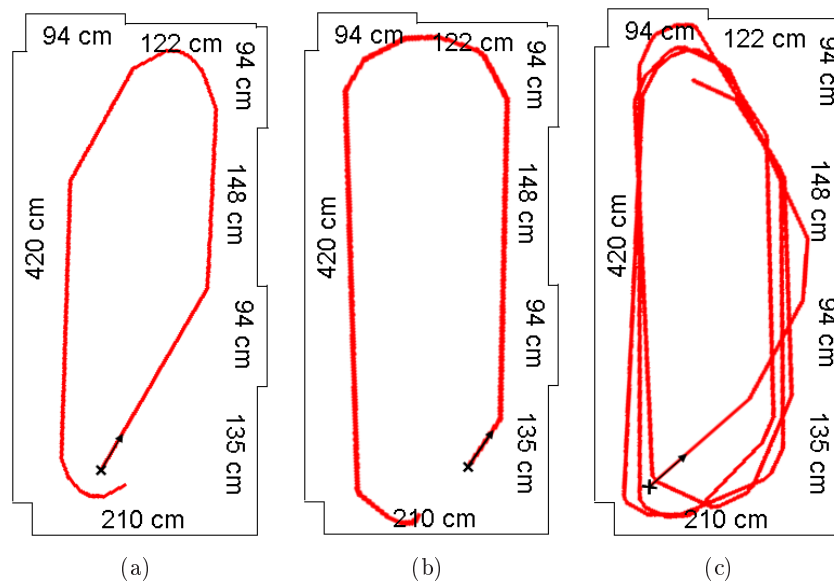


Abbildung 5.16: Graphische Darstellung einiger vom Roboter gefahrenen Trajektorien. Als Grundlage dienten für diese Graphiken die entsprechenden Odometriedaten des Roboters.

Roboters erstellt. Dazu wurden die Daten während der gesamten Roboterbewegung in einer separaten Datei mitprotokolliert. Der schwarze Kreuz mit dem Pfeil gibt die anfängliche Startposition und die Orientierung des Roboters an. Es ist deutlich zu sehen, dass der Roboter an einigen Stellen innerhalb kurzer Zeitintervalle mehrere Rotationen durchführt, die letzten Endes in einer Kurvenfahrt resultieren.

Die Ergebnisse der durchgeführten Experimente zeugen von der praktischen Anwendbarkeit des entwickelten Ansatzes. Mittels der Bildinformation der Kamera und seinen Odometriedaten war der Roboter in der Lage robust über einen längeren Zeitraum den bevorstehenden Hindernissen auszuweichen. Vergleicht man dieses Ergebnis mit den in der Arbeit von Souhila [SK07] berichteten Befunden, so ist eine deutliche Performanzsteigerung zu erkennen. Souhila et. al. berichten von einer Kurvenfahrt, während der der Roboter erfolgreich zwei Hindernissen ausgewichen ist. Die Autoren weisen allerdings mehrmals daraufhin, dass dieses Ergebnis nicht mit einer üblicherweise gewünschten Stetigkeit erzielbar war. Aufgrund der zu stark schwankenden TTC-Werte war der Roboter nicht in der Lage den Hindernissen robust auszuweichen. Mit der Verwendung des hier vorgestellten Verfahrens hätten diese Probleme umgangen werden können.

### 5.5.3 Laufzeit des Verfahrens

Mit der Verwendung des iterativen Ansatzes zur Adaption der Modellparameter ist das hier vorgestellte Verfahren zur Berechnung von TTC-Werten echtzeitfähig. Anders als bei dem fehlerminimierenden Adoptionsansatz aus Kapitel 4 beinhaltet die Aktualisierungsvorschrift bei dem iterativen Ansatz nur eine begrenzte Anzahl an rudimentalen Rechenschritten. Folgend den Gleichungen 4.12 aus Kapitel 4 sind hierbei zur Adaption eines Parametersatzes insgesamt 14 Rechenoperationen notwendig. Zusätzlich kommt noch die Berechnung des dabei verwendeten Mittelpunktes zwischen der gemessenen und der vorhergesagten Merkmalsposition hinzu. Doch auch die stellt keinen nennenswerten Aufwand dar.

Anders sieht es aus bei der Extraktion der KLT-Merkmale im Bild. Im Rahmen dieser Arbeit wurde eine Implementation von Stan Birchfield [Bir] verwendet. Wie er selbst schreibt liegt der Schwerpunkt dieser Implementierung auf dem einfachen Verständnis der durchgeführten Operationenfolge sowie der leichten Handhabung des Codes. Abhängig von der Struktur und dem Informationsgehalt des verwendeten Bildes können damit allerdings im Durchschnitt nur ein bis zwei Bilder pro Sekunde verarbeitet werden. Im Rahmen der oben beschriebenen Experimente stellte dies in so fern kein Problem dar, als dass die Geschwindigkeit des Roboters manuell auf nur  $4\text{ cm/s}$  eingestellt wurde.

Bei Anwendungen wo eine niedrigere Verarbeitungszeit benötigt wird, kann ähnlich wie bei SIFT-Merkmalen eine GPU basierte Implementierung Abhilfe schaffen. Eine frei verfügbare Version einer solchen Implementierung gibt es von Sudipta Sinha [Sin10]. Folgend seiner Evaluierungsergebnissen können damit bis zu  $40\text{ fps}$  bei einer Bildgröße von  $1024 \times 768$  erreicht werden. Voraussetzung für diese Zahl ist

eine maximal zu extrahierende Anzahl an KLT-Merkmalen von 100. Weitere Einsichten in die Leistungsfähigkeit dieser GPU Implementierung können hier [Sin10] entnommen werden.

Schlussfolgernd lässt sich sagen, dass unter Verwendung der richtigen Hardware und der entsprechenden Software das im Rahmen dieser Arbeit vorgestellte Verfahren zur TTC-Berechnung durchaus echtzeitfähig ist.

## 5.6 Diskussion und Zusammenfassung

Die in diesem Kapitel vorgestellten Ergebnisse bestätigen deutlich die praktische Anwendbarkeit des hier entwickelten Ansatzes zur robusten Berechnung der TTC-Werte. Ausgehend von den Bildern nur einer Kamera ist es damit möglich den Roboter kollisionsfrei und ohne zusätzliche Umgebungsinformation in der Umwelt navigieren zu lassen.

Die Ergebnisse haben gezeigt, dass die anfänglichen Positionen der betrachteten Merkmalssequenzen im Kamerabild durchaus Einfluss auf die Güte der berechneten TTC-Werte haben. Hatten die Sequenzen einen streng horizontalen oder vertikalen Verlauf, so waren die resultierenden TTC-Werte sehr instabil. Einen Ausweg bildete in diesem Fall die Vernachlässigung eines Parameterpaares und ausschließliche Benutzung des Anderen. Anhand eines Beispiels wurde hier gezeigt, dass auf diese Weise die Abweichungen der berechneten TTC-Werte von den richtigen Werten drastisch vermindert werden konnten.

Innerhalb dieser Experimente wurde auch untersucht in wie fern die Güte der berechneten TTC-Werte davon abhängt, welchen Algorithmus man zur Extraktion und Verfolgung von Merkmalen im Bild verwendet. Hier wurde der Einfluss der KLT- und der SIFT-Merkmale miteinander verglichen. Dabei wurde festgestellt, dass obwohl die SIFT-Merkmale in fast allen Vergleichsstudien die vordersten Plätze besetzen, im Rahmen der TTC-Berechnung konnten damit keine Vorteile gegenüber der Verwendung der KLT-Merkmale erzielt werden. Die auf gleichen Bildaufnahmen berechneten TTC-Sequenzen zeigten kaum Unterschiede. Dies spiegelte sich auch in den Histogrammen der zwei zum Vergleich herangezogenen Kriterien. Beim optischen Vergleich dieser konnten keine markanten Eigenschaften entdeckt werden, die für die Verwendung des einen oder des anderen Merkmals charakteristisch wären. Aus diesem Grund wurde bei den weiteren Experimenten nur auf die KLT-Merkmale zurückgegriffen. Der Berechnungsaufwand dieser Merkmale ist viel geringer als bei SIFT-Merkmalen.

Gleichzeitig wurde in diesem Kapitel die Performanz der drei Erweiterungen des Ansatzes auf variierende Roboterbewegungen untersucht. Als Gütemaß wurde

dafür zum einen die Anzahl der nicht stattfindenden Kollisionen, zum anderen die Dauer der kollisionsfreien Navigation des Roboters herangezogen. Die Ergebnisse zeugten davon, dass das Verfahren beim Einsatz in geschlossenen Räumen eine hohe Performanz erreicht.

Vergleicht man noch einmal die in diesem Kapitel präsentierten TTC-Sequenzen mit den Sequenzen, die mittels des Standardverfahrens (Kapitel 4) berechnet wurden, so stellt man sofort eine deutliche Verbesserung fest. Nach dem Abschluss der Adaptionphase zeigten fast alle Sequenzen einen glatten Verlauf und näherten sich immer weiter den richtigen TTC-Werten an. Während bei den Ergebnissen des Standardverfahrens kaum Vorhersagen über den weiteren Verlauf der TTC-Sequenz möglich waren, konnte mit dem hier entwickelten Verfahren eine sichere Navigation des Roboters gewährleistet werden.

Es sei an dieser Stelle angemerkt, dass die gute Performanz dieses Verfahrens nicht auf der Nachbearbeitung der mit dem Standardverfahren berechneten TTC-Werten basiert. Anders als die meisten anderen Ansätze, nutzt dieses Verfahren die Gesetze der projektiven Geometrie und erreicht erst dadurch seine überdurchschnittlich Performanz. Die für das Verfahren zusätzlich benötigte Information aus den vorangehenden Bildern ist nicht zu kritisieren, denn diese Bildinformation ist vorhanden und Akquisition dieser erfordert keinen zusätzlichen Aufwand.

Wie bereits erwähnt wurden die wichtigsten Punkte dieses Kapitels zum ersten Mal auf dem 21-ten 'Fachgespräch Autonome Mobile Systeme 09' vorgestellt.

# Durchführung und Ergebnisse einer Fallstudie

---

## Kapitelinhalt

---

<b>6.1</b>	<b>Implementierung und Durchführung . . . . .</b>	<b>134</b>
<b>6.2</b>	<b>Ergebnisse . . . . .</b>	<b>139</b>
<b>6.3</b>	<b>Diskussion und Zusammenfassung . . . . .</b>	<b>151</b>

---

Um die praktische Anwendbarkeit der in vorigen Kapiteln vorgestellten Ansätze zu untersuchen, wurden diese noch zusätzlich im Rahmen einer Fallstudie getestet.

Ziel der Fallstudie war die Simulation eines Lagerhaus Szenarios. Ein autonom agierender Roboter hat hier die Aufgabe eine schwere Palette oder ein Paket zu einer vorher spezifizierten Zielposition zu transportieren. Die Daten zu seiner groben Fahrtrichtung bekommt er aus den Bildinformationen einer Weitsichtkamera, die an der Decke des Lagerhauses montiert ist. In ihrem Sichtbereich befindet sich sowohl der Roboter als auch die gewünschte Zielposition. Um eine stabile und zugleich auch eine einfache und schnelle Installation des gesamten Systems zu gewährleisten, wurde im Rahmen dieses Szenarios explizit auf die Verwendung eines Stereo-Kamerasystems verzichtet. Die Bildinformation der Deckenkamera liefert damit keine 3D-Daten zur lokalen Umgebung des Roboters, sondern dient nur zur Bestimmung der gewünschten groben Fahrtrichtung des Roboters. Die Entfernungsschätzung zu den bevorstehenden Hindernissen erfolgt hier mittels der Bildauswertung der Kamera, die direkt auf dem Roboter montiert ist.

Aufgrund eines solch flexiblen Systemaufbaus entfällt in diesem Szenario die Notwendigkeit einer ressourcenaufwendigen Verwaltung einer globalen Karte. Die zur sicheren Navigation des Roboters erforderliche Information wird hier online berechnet und basiert ausschließlich auf den jüngsten Bilddaten der beiden Kameras. Damit kann sichergestellt werden, dass das mittels der bereits vorgestellten Verfahren konstruierte System robust auf mögliche Verschiebungen von Hindernissen reagiert. Auch weitere Störfaktoren wie wechselnde Lichtverhältnisse oder teilweise

bzw. vollständige Verdeckung des zu navigierenden Objektes sollten keinen großen Einfluss auf die Performanz des Gesamtsystems haben.

Das vorliegende Kapitel beschreibt zunächst die programmtechnischen Aspekte der Implementierung und den Aufbau der Versuchsumgebung. Anschließend werden in den nächsten Abschnitten die erzielten Ergebnisse vorgestellt. Der Schwerpunkt der durchgeführten Experimente lag dabei nicht auf der erneuten Evaluierung der einzelnen Verfahren, sondern viel mehr auf der Beurteilung derer gemeinsamen Einsatzfähigkeit.

## 6.1 Implementierung und Durchführung

Nach der Beschreibung der programmtechnischen Aspekte der Implementierung, wird in diesem Abschnitt anschließend der Aufbau der entsprechenden Versuchsumgebung erklärt. Die in der Fallstudie verwendete Hardware sowie die genaue Position und Orientierung der eingesetzten Komponenten werden hier ausführlich beschrieben.

### 6.1.1 Programmtechnische Aspekte der Implementierung

Die programmtechnische Implementierung der in der vorliegenden Arbeit vorgestellten Verfahren erfolgte in der Programmiersprache C++ unter Zuhilfenahme der integrierten Entwicklungsumgebung Microsoft Visual Studio 2003 Professional Edition. Zusätzlich wurden dabei folgende C++ Bibliotheken verwendet:

**C++-Standardbibliothek** Standardisierte C++ Programmierbibliothek zur allgemeinen Verwendung. Sie stellt verschiedene generische Container, Funktionen zu deren Manipulierung, Funktionsobjekte und Datenströme für den Dateizugriff, Unterstützung von Sprachmitteln sowie einfache Funktionen zur Verfügung.

**OpenCV 2.1** Eine freie C++ Programmbibliothek mit Algorithmen für die Bildverarbeitung und maschinelles Sehen. Die Bibliothek umfasst unter anderem Algorithmen für Gesichtsdetektion, 3D-Funktionalität, Haar-Klassifikatoren, verschiedene sehr schnelle Filter (Sobel, Canny, Gauß) und Funktionen für die Kamerakalibrierung.

**ERSP 3.0** Von Evolution Robotics zur Verfügung gestellte Bibliothek zur Ansteuerung der Roboterhardware (ER1 Personal Robot System).

Die ERSP Bibliothek stellt zusätzlich eine Software-Architektur zur Verfügung, mittels derer die einzelnen Programmstücke zu Blöcken (sog. Behavior) zusammenge-



fasst werden können. Ein Behavior beinhaltet üblicherweise eine vollständige Implementierung eines bestimmten Algorithmuses. Eingabe und die berechnete Ausgabe des Algorithmuses bilden gleichzeitig auch die Ein- und Ausgabe des entsprechenden Behaviors. Bei der Implementierung der Verfahren wurde durchgehend der C++ Variablentyp 'double' verwendet.

Mittels des sog. Behavior Composers, einer User Interface Oberfläche kann der Programmfluss (sog. Behavior Netzwerk) bestehend aus einzelnen Behaviors modelliert werden. Abbildungen 6.1, 6.2 und 6.3 zeigen die Screenshots der im Rahmen dieser Arbeit konstruierten Behavior Netzwerke.

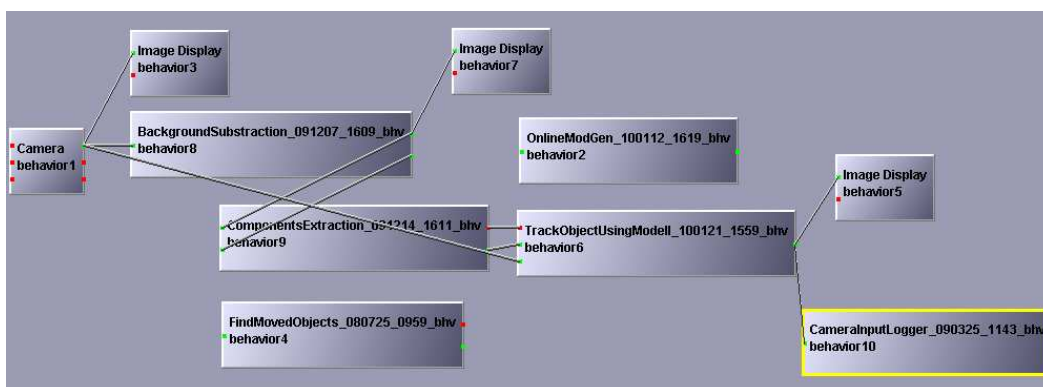


Abbildung 6.1: Behavior Netzwerk zur online Generierung eines Objektmodells. Nach dem Umschalten der Komponenten wurde dieses Netzwerk auch zum Verfolgen eines Objektes mittels eines bereits generierten Modells benutzt.

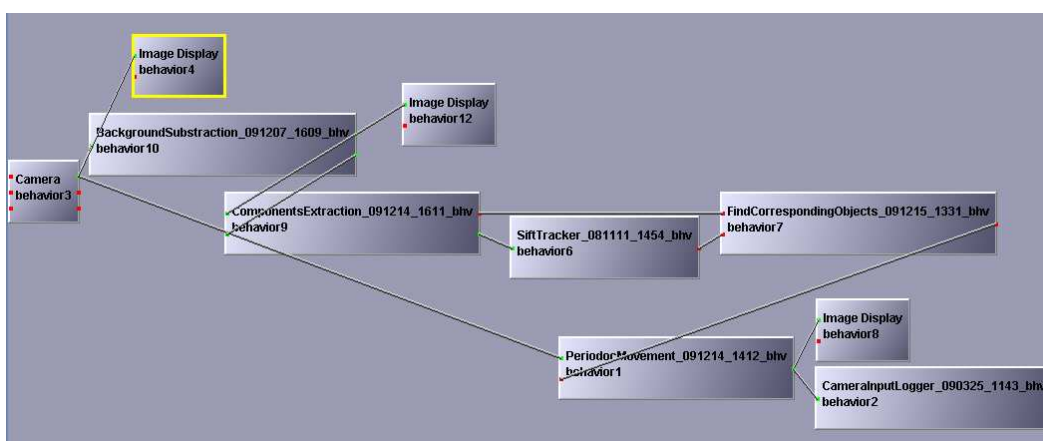


Abbildung 6.2: Behavior Netzwerk zur Erkennung einer periodischen Bewegung.

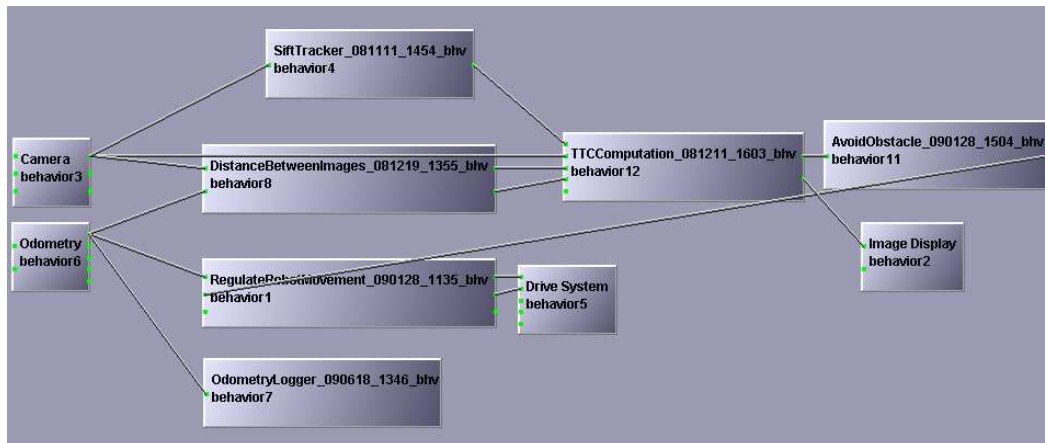


Abbildung 6.3: Behavior Netzwerk zur Berechnung von Time-To-Contact Werten zu den bevorstehenden Hindernisse.

Die Rechtecke repräsentieren hierbei die Behavior. Die Interaktionen der einzelnen Komponenten sind durch Linien gekennzeichnet. Die Ausgaben des einen Programmstücks bilden die Eingabe für das nächste Programmstück. Der gesamte Programmfluss läuft von links nach rechts in einer unendlichen Schleife.

Das in Abbildung 6.1 dargestellte Behavior Netzwerk erfüllt zum einen den Zweck der online Modellgenerierung und zum anderen der Verfolgung des Objektes mittels eines bereits konstruierten Modells. Online Modellgenerierung ist hier durch das Behavior 'OnlineModGen\_100112\_1619\_bhv' repräsentiert. In der Abbildung ist dieses nicht an die Komponente der Vordergrund-Hintergrund Separierung 'BackgroundSubstraction\_091207\_1609\_bhv' angeschlossen, weil das abgebildete Netzwerk sich in dem Modus der Objektverfolgung befindet. Das Behavior 'TrackObjectUsingModell\_100121\_1559\_bhv' benutzt das bereits konstruierte Modell zur Objektverfolgung und leitet seine Ausgabe in Form eines Bildes an das Behavior 'Image Display'. In diesem Bild ist das verfolgte Objekt deutlich markiert. Das bereits erwähnte Behavior 'BackgroundSubstraction\_091207\_1609\_bhv' leistet hier zusammen mit dem Behavior 'ComponentsExtraction\_694214\_1611\_bhv' die Vorarbeit, indem es mittels des Verfahrens zur Vordergrund-Hintergrund Separierung aus Kapitel 2 alle bewegten Objekte der Szene extrahiert und zu den einzelnen Komponenten zusammenfasst.

Das Behavior Netzwerk in Abbildung 6.2 dient der Erkennung einer periodischen Objektbewegung. Das Verfahren aus Kapitel 3 zur Erkennung einer solchen Bewegung verbirgt sich hinter dem Behavior 'PeriodicMovement\_091214\_1412\_bhv'. Die beiden Behaviors 'BackgroundSubstraction\_091207\_1609\_bhv' und 'Component-

sExtraction\_694214\_1611\_bhv' leisten auch hier die Vorarbeit der Extraktion der bewegten Objekte der Szene. Diese Information wird dann an die beiden Behavior 'SiftTracker\_081111\_1454\_bhv' und 'FindCorrespondingObjects\_091215\_1331\_bhv' weitergeleitet, welche die zur Erkennung einer periodischen Bewegung notwendigen Trajektorien der Objekte berechnen.

Das Behavior Netzwerk zur Kollisionsvermeidung ist in Abbildung 6.3 dargestellt. Die Verfahren zur Berechnung von Time-To-Contact Werten aus Kapitel 5 verbergen sich hinter dem Behavior 'TTCComputation\_081211\_1603\_bhv'. Dieses leitet seine Ausgabe an das Behavior 'AvoidObstacle\_090128\_1504\_bhv' weiter, wo über das 'Drive System' des Roboters dafür gesorgt wird, dass dieser dem bevorstehenden Hindernis ausweicht. Wie der Abbildung ebenfalls zu entnehmen ist, wird hier nicht nur die Kamerainformation verwendet sondern auch die Odometriedaten des Roboters. Die entsprechenden Behaviors tragen die Namen 'Camera' und 'Odometry'.

Der zur Benutzung der einzelnen Komponenten relevante Programmcode ist auf der beiliegenden CD verfügbar und kann als Basis zum weiteren Ausbau des Systems verwendet werden.

### 6.1.2 Aufbau der Versuchsumgebung

Zur Durchführung der Experimente wurde ähnlich wie in Kapitel 5 ein abgegrenzter Flurbereich verwendet. Die weißen Wände des Flurs wurden mit verschiedenen Ausschnitten unterschiedlicher Zeitschriftenartikel beklebt. Diese Maßnahme war in sofern notwendig, als dass im Gegensatz zu strukturbeinhaltenden Bildern auf weißen Flächen keine markante Punkte extrahiert werden können. Dies ist allerdings, wie in Kapitel 5 dargestellt, eine wichtige Voraussetzung für die kollisionsfreie Navigation des Roboters.

Abbildung 6.4 zeigt zwei Fotoaufnahmen des entsprechenden Versuchsaufbaus. Das linke Bild zeigt eine Weitsichtaufnahme des Flurs, das rechte dagegen eine Nahaufnahme des Roboters und der verwendeten Weitsichtkamera. Bei diesen Experimenten wurde ebenfalls der ER1 Roboter von Evolution Robotics verwendet. Die unterschiedlichen Ansichten dieses Roboters, sowie eine kurze Beschreibung seines Aufbaus befinden sich in Kapitel 5. Der entscheidende Vorteil bei der Verwendung dieses Roboters lag in der dafür mitgelieferten Software. Mittels eines Steuerungsmoduls ließ sich dieser leicht in die gewünschte Position navigieren oder auf die Durchführung eines bestimmten Bewegungsmusters vorprogrammieren.

Im rechten Bild ist ebenfalls deutlich zu sehen die bei diesen Experimenten verwendete Weitsichtkamera. Mittels eines Stativs wurde die Kamera in einer Höhe



Abbildung 6.4: Für die Experimente verwendeter Flur: (a) Weitaufnahme des Flurs; (b) Nahaufnahme des Roboters und der verwendeten UEye-Kamera auf dem Stativ.

von 1,25 m montiert und entspricht im Rahmen des hierbei simulierten Szenarios einer Deckenkamera. Abbildung 6.5 zeigt eine Nahaufnahme dieser Kamera sowie ein mittels dieser Kamera, aus dieser Perspektive aufgenommenes Bild.

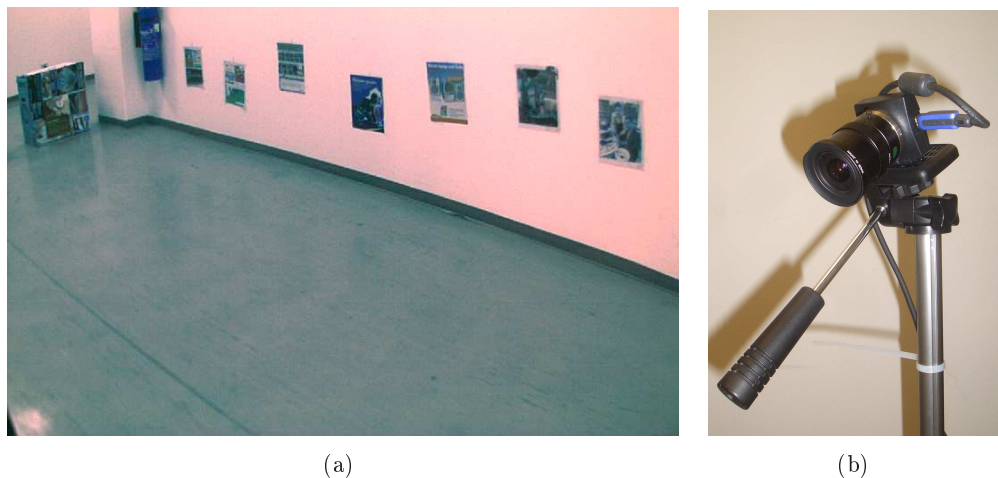


Abbildung 6.5: Für die Experimente verwendete Weitsichtkamera: (a) Sichtbereich der Weitsichtkamera; (b) als Weitsichtkamera verwendete UEye-Kamera.

Die technischen Koordinaten der Kamera sind die folgenden:

- UEye UI-1220-C Farbkamera, Auflösung  $752 \times 480$

In der Abbildung 6.5 links ist deutlich zu sehen, dass aufgrund der gewählten Position und Höhe die Kamera einen größeren Bereich des verwendeten Flurs überschaut.

### 6.1.3 Durchführung der Fallstudie

Die Durchführung der Experimente erfolgte in mehreren Schritten. Zur anfänglichen Detektion des Roboters im Kamerabild wurde dieser zunächst auf die Durchführung einer einfachen periodischen Bewegung vorprogrammiert. Im Rahmen dieses Szenarios kam eine Vor- und Rückwärtsbewegung zum Einsatz. Der hauptsächliche Grund dafür war, dass während dieser Bewegung die anfängliche Ansicht des Roboters im Bild der Weitsichtkamera sich kaum änderte. Auf diese Weise war es möglich die entsprechende Trajektorie des Roboters mittels des SIFT-basierten Verfolgungsansatzes zu konstruieren. Wie bereits in Kapitel 3 erklärt, würde sich hierfür eine Kreisbewegung oder eine quadratische Bewegung des Roboters aufgrund der gewählten Perspektive der Weitsichtkamera wenig eignen.

Nach der erfolgreichen Detektion des Roboters wurde das Modul der 'Online Modellgenerierung' gestartet. Zur Vereinfachung wurde der Roboter während dieses Stadiums manuell gesteuert. Um ein vollständiges Modell des Roboters zu erhalten, wurde hierbei keineswegs auf eine gleichmäßige Bewegung des Roboters geachtet. Vielmehr glich die Bewegung einer eher chaotischen, ziellosen Fahrt in dem zur Verfügung gestellten Raum. Das dabei erzeugte Modell des Roboters diente anschließend zur dessen Verfolgung im Bild der Weitsichtkamera.

Zeitgleich zur Konstruktion der Robotertrajektorie im Bild der Weitsichtkamera wurde lokal auf dem Roboter das Modul der 'Entfernungsschätzung' gestartet. Der Roboter wurde zu diesem Zeitpunkt nicht mehr manuell gesteuert. War kein Hindernis in Sicht, so bewegte sich dieser Vorwärts mit einer Geschwindigkeit von  $4\text{ cm/s}$ . Im Falle einer bevorstehenden Wand führte der Roboter eine rotierende Bewegung durch. Damit die vom Roboter tatsächlich gefahrene Trajektorie auch für die spätere Verarbeitung zur Verfügung stand, wurden die Odometriedaten des Roboters während seiner Bewegung in einer separaten Datei gespeichert.

Diese und sonstige während dieser Experimentenreihe erzielten Ergebnisse werden in dem folgenden Abschnitt ausführlich geschildert.

## 6.2 Ergebnisse

Zur besseren Übersichtlichkeit der während der Fallstudie erzielten Ergebnisse, wurde das vorliegende Abschnitt in drei Unterabschnitte aufgeteilt. Der erste Teil beschäftigt sich mit der anfänglichen Detektion des Roboters im Bild der Weitsichtkamera. Hier werden Bilder und Graphiken gezeigt, die die Konstruktion der Roboter Trajektorie sowie die Detektion einer periodischen Bewegung dokumentieren. Der zweite Teil präsentiert die Ergebnisse des Moduls der 'Online Modellgenerierung'.

Dieser Abschnitt beinhaltet auch ein Protokoll darüber welche Ansichten zu welchem Zeitpunkt zu dem entsprechenden Robotermodell zusammengefasst werden. Im dritten Teil wird untersucht wie stabil sich der Roboter mittels des vorher konstruierten Modells im Bild der Weitsichtkamera verfolgen lässt.

### 6.2.1 Bewegungsbasierte Roboterdetektion

Wie bereits in Kapitel 3 erklärt wurde die zur bewegungsbasierten Roboterdetektion notwendige Trajektorie des Roboters mittels eines SIFT-basierten Verfolgungsansatzes konstruiert. Um Rechenkapazität zu sparen wurde die Berechnung der SIFT Merkmale auf einzelne Bildausschnitte begrenzt, die die bewegten Objekte der Szene repräsentieren. Die Extraktion solcher Bildbereiche erfolgte mittels des in Kapitel 2 vorgestellten Ansatzes.

Abbildung 6.6 zeigt exemplarisch die während der Roboterbewegung berechneten SIFT-Merkmale zur unterschiedlichen Zeitpunkten. Der rote Schwarm, der dem

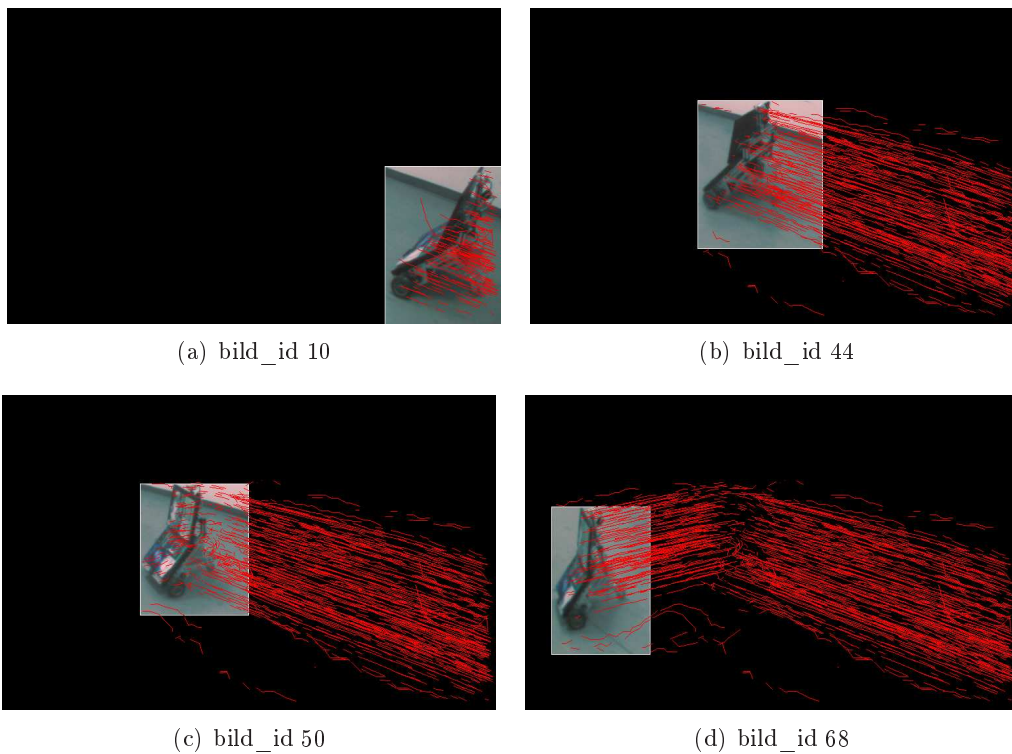


Abbildung 6.6: Die Positionen der während der Roboterbewegung berechneten SIFT-Merkmale sind hier durch kurze rote Fäden dargestellt. Ein Pfad verbindet korrespondierende SIFT-Merkmale in aufeinander folgenden Bildern.

Anschein nach den Roboter verfolgt, besteht hier aus einzelnen kurzen Fäden, die korrespondierende SIFT-Merkmale in zwei aufeinander folgenden Bildern miteinan-

der verbinden. Kann ein Merkmal über mehrere Bilder stabil verfolgt werden, so verlängert sich auch der entsprechende Faden.

Wie der Abbildung 6.6 ebenfalls zu entnehmen ist, kann die Anzahl der verfolgten Merkmale vorher nicht festgelegt werden. Abhängig von der Qualität des Bildes, dem aktuell vorhandenen Rauschanteil und den Eigenschaften des betrachteten Objektes können im Laufe der Bewegung neue Merkmale hinzukommen, während andere verschwinden. Besonders deutlich wird dies zu dem Zeitpunkt, als der Roboter eine Drehung durchführt. An dieser Stelle nimmt die Dichte des Schwarms deutlich ab. Der Grund dafür liegt in der Tatsache, dass während der Drehung viele Merkmale im neuen Bild nicht wiedergefunden werden können.

Würde der Roboter an dieser Stelle eine stärkere Rotation durchführen, so ist davon auszugehen, dass der erwähnte Schwarm an dieser Stelle einen vollständigen Riss erfahren würde. Ergebnisse ähnlicher Experimente bestätigen diese Vermutung. Dieses Problem kann allerdings umgangen werden, wenn man die Kamera etwas höher platziert. Denn aus der so erhaltenen Perspektive würde sich die Ansicht des Roboters bei einer rotierenden Bewegung im Kamerabild nicht so stark ändern.

Aufgrund des bereits beschriebenen Versuchsaufbaus und den obigen Überlegungen wurde im Rahmen dieser Fallstudie eine einfache Vor- und Rückwärtsbewegung für das periodische Bewegungsmuster ausgewählt. Folgend dem in Kapitel 3 beschriebenen Verfahren zur Erkennung dieses Musters, zeigt die Abbildung 6.7 die dabei resultierende periodische Kurve. Ähnlich den Ergebnissen in Kapitel 3 sind

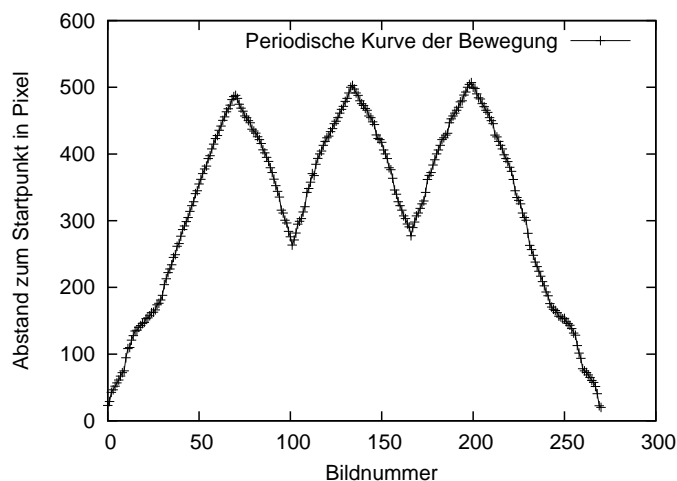


Abbildung 6.7: Periodische Kurve, die bei der Vor- und Rückwärtsbewegung des Roboters entstanden ist.

auch hier die zur Erkennung notwendigen Maxima und Minima deutlich zu erkennen.

Abbildung 6.8 zeigt zusätzlich noch einige Aufnahmen der entsprechenden Bild-

sequenz. In den linken Bildern dieser Abbildung sind die dazu korrespondierenden Ergebnisse des hier verwendeten Vordergrund - Hintergrund Separationsverfahrens, zusammen mit der Berechnung der achsenparallelen Rechtecke um die bewegenden Objekte der Szenen zusehen. Die rechten Bilder der Abbildung zeigen dagegen die originalen Bildaufnahmen der Weitsichtkamera. Das rote Quadrat hier markiert den Mittelpunkt des Rechtecks um das bewegende Objekte, in diesem Fall den Roboter. Wie bereits erwähnt dient der Mittelpunkt des Rechtecks zur Positionsbestimmung des interessierenden Objektes im Kamerabild. Im unteren rechten Bild findet sich noch zusätzlich ein roter Kreis um das rote Quadrat. Diese Markierung dient dem Zweck aufzuzeigen, dass eben dieses Objekt als das Objekt erkannt wurde, welches sich periodisch bewegt.

Interessant war bei diesem Experiment zu beobachten, dass bei einer längeren periodischen Bewegung die Konturen des Objektes langsam in den Hintergrund verschwanden. Als Konsequenz daraus wurde auch die Größe des Rechtecks verzerrt und damit auch die Position des Objektes im Kamerabild verschoben. Dies resultierte dann in einer nicht mehr so glatten periodischen Kurve, wie diese in Abbildung 6.7 zu sehen ist.

Der Grund für dieses Systemverhalten war der, dass das hier eingesetzte Vordergrund - Hintergrund Separationsverfahren bei einer längeren periodischen Bewegung des Objektes dieses nicht mehr als ein Vordergrundobjekt der Szene akzeptierte. Die Objektbewegungen wurden immer mehr dem 'dynamischen' Hintergrund der Szene zugeordnet, ähnlich dem in Kapitel 2 gezeigten Experiment mit dem im Hintergrund schwankenden Baum. Dieses Problem ließ sich allerdings umgehen, indem der Roboter an eine andere Stelle bewegt wurde.

Nachdem auf diese Weise die erste Roboteransicht gewonnen wurde, startete das Modul 'Online Modellgenerierung'. Die dabei erzielten Ergebnisse sind im folgenden Abschnitt dargestellt.

### 6.2.2 Online Generierung eines Robotermodells

Wie bereits in Kapitel 3 erklärt, hängt die Qualität des mit diesem Moduls konstruierten Objektmodells von mehreren Faktoren ab. Eines der wichtigsten Kriterien ist hierbei die Bewegung des Objektes. Rotiert dieses zu schnell, so werden in der Datenbank üblicherweise zunächst mehrere Modelle mit jeweils unterschiedlichen Ansichten des entsprechenden Objektes gespeichert. Erst wenn eine neue Ansicht mehreren Modellen gleichzeitig zugeordnet werden kann, werden diese zu einem Modell zusammengefasst. Aus diesem Grund kann es durchaus vorkommen, dass zur Generierung eines vollständigen Modells eine länger andauernde Objekt Bewegung



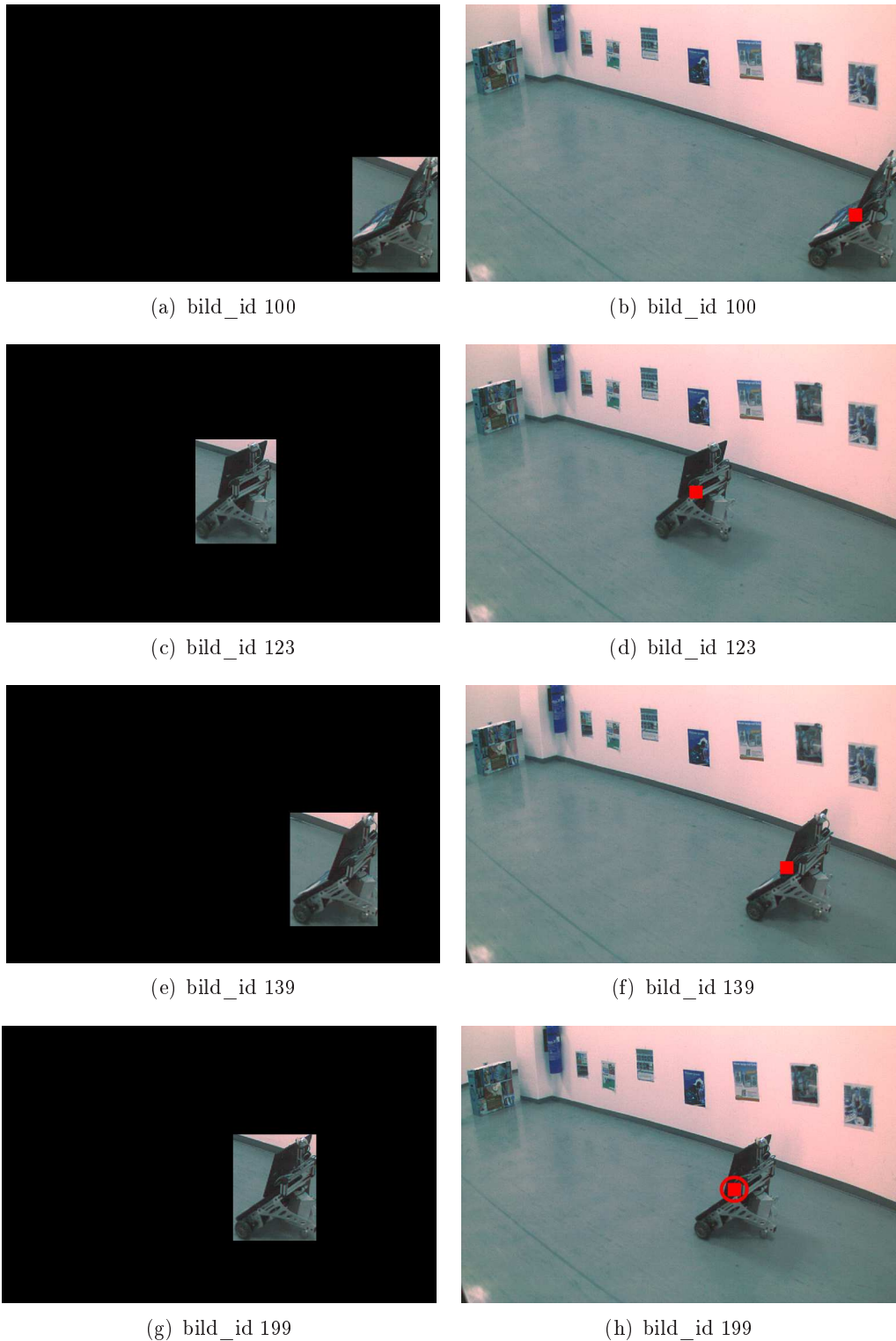
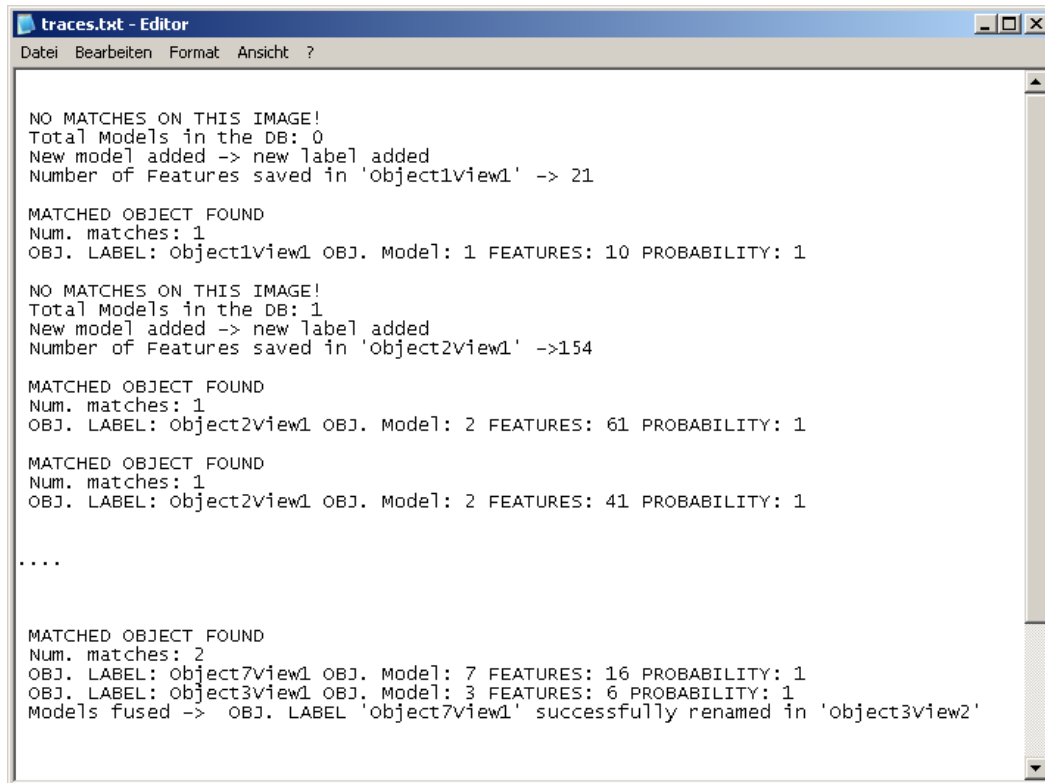


Abbildung 6.8: Linke Bilder: Ergebnisse des in Kapitel 2 vorgestellten Vordergrund - Hintergrund Separationsverfahrens zusammen mit der Berechnung des achsenparallelen Rechtecks um den bewegenden Roboter; rechte Bilder: Detektion des Roboters aufgrund der von ihm durchgeführten periodischen Bewegung.

notwendig ist.

Abbildung 6.9 zeigt einen Ausschnitt aus dem Protokoll, welches die einzelnen Schritte der Modellgenerierung dokumentiert. In dem hier dargestellten Fall ist die



```
traces.txt - Editor
Datei Bearbeiten Format Ansicht ?

NO MATCHES ON THIS IMAGE!
Total Models in the DB: 0
New model added -> new label added
Number of Features saved in 'object1view1' -> 21

MATCHED OBJECT FOUND
Num. matches: 1
OBJ. LABEL: Object1view1 OBJ. Model: 1 FEATURES: 10 PROBABILITY: 1

NO MATCHES ON THIS IMAGE!
Total Models in the DB: 1
New model added -> new label added
Number of Features saved in 'Object2view1' ->154

MATCHED OBJECT FOUND
Num. matches: 1
OBJ. LABEL: Object2view1 OBJ. Model: 2 FEATURES: 61 PROBABILITY: 1

MATCHED OBJECT FOUND
Num. matches: 1
OBJ. LABEL: object2view1 OBJ. Model: 2 FEATURES: 41 PROBABILITY: 1

....

MATCHED OBJECT FOUND
Num. matches: 2
OBJ. LABEL: Object7view1 OBJ. Model: 7 FEATURES: 16 PROBABILITY: 1
OBJ. LABEL: object3view1 OBJ. Model: 3 FEATURES: 6 PROBABILITY: 1
Models fused -> OBJ. LABEL 'object7view1' successfully renamed in 'object3view2'
```

Abbildung 6.9: Ausschnitt aus dem Protokoll, welches die einzelnen Schritte der Modellgenerierung dokumentiert.

verwendete Datenbank zunächst leer. Daher wird als erstes ein neues Modell erzeugt und diesem die neue Objektansicht zugeordnet. Diese bekommt den Namen 'Object1View1'. Die darauf folgende Objektansicht kann der bereits gespeicherten erfolgreich zugeordnet werden. Die Meldung 'MATCHED OBJECT FOUND' mit den entsprechenden Erklärungen bestätigt dies.

Aufgrund der Objektbewegung kann allerdings bereits die nächste Objektansicht dem Modell in der Datenbank nicht mehr zugeordnet werden. Daher wird ein weiteres Modell mit der aktuellen Objektansicht in der Datenbank abgelegt. Die neu gespeicherte Objektansicht bekommt den Namen 'Object2View1'. Die nächsten beiden Einträge in dem Protokoll zeugen davon, dass die neu gewonnenen Objektansichten der zuletzt gespeicherten Ansicht zugeordnet werden können.

Der letzte Eintrag in dem Ausschnitt des Protokolls ist ein Beispiel dafür, wie

zwei Modelle zu einem zusammengefasst werden. Die aktuelle Objektansicht kann hier den Modellen 7 und 3 gleichzeitig zugeordnet werden. Aus diesem Grund werden die im Modell 7 gespeicherten Ansichten dem Modell 3 zugeordnet. Die Ansicht 'Object7View1' des Modells 7 wird umbenannt in 'Object3View2'.

Nachdem der Roboter bei diesem Experiment ausreichend lange durch den Flur bewegt wurde, wurden dessen unterschiedliche Ansichten letzten Endes nur einem Modell in der Datenbank zugeordnet. Die Abbildung 6.10 zeigt die Ansichten dieses Modells.

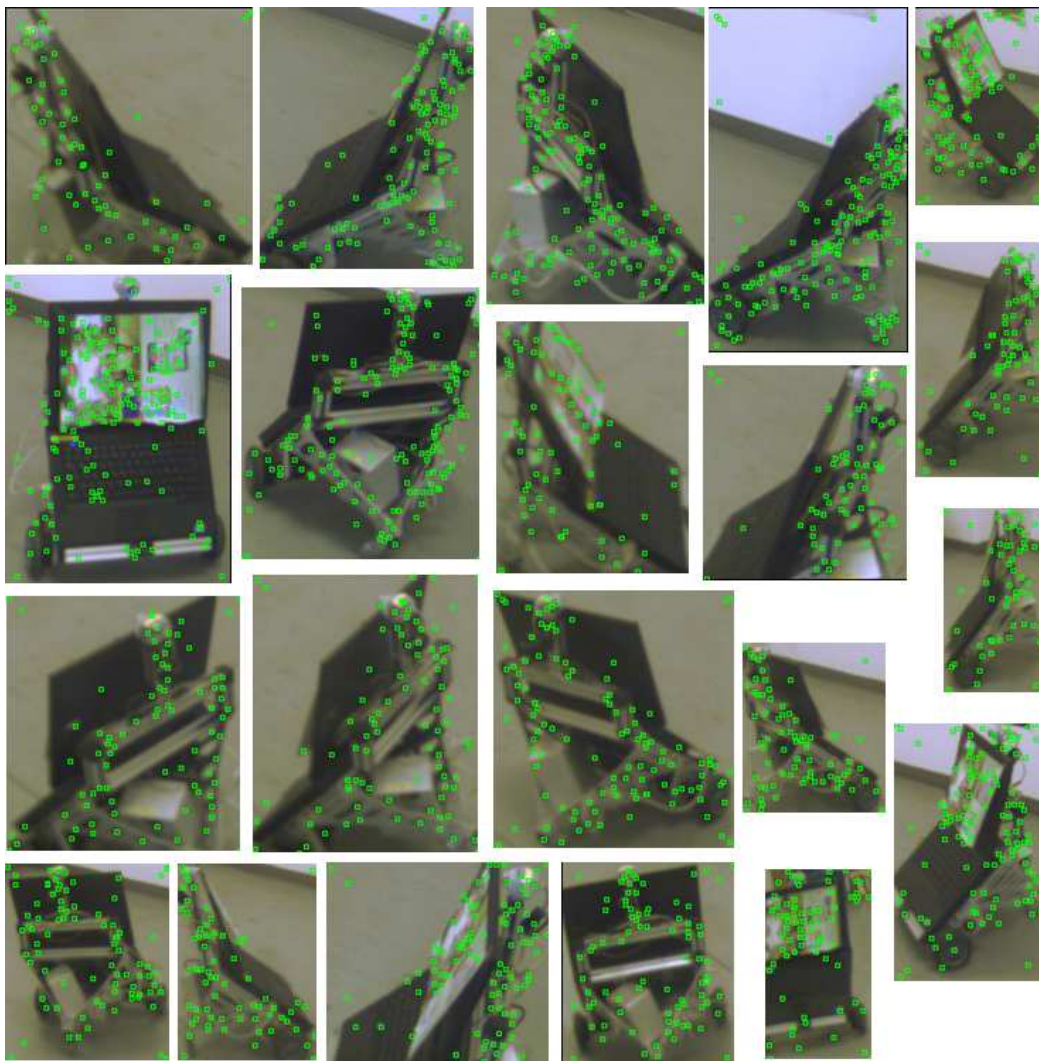


Abbildung 6.10: Unterschiedliche Ansichten des online generierten Robotermodells. Die grünen Punkte entsprechen den Positionen der hierbei extrahierten SIFT-Merkmale.

Im Unterschied zu dem im Kapitel 3 gezeigten Modell fällt hier sofort auf, dass dieses Modell aus viel mehr Ansichten besteht, als das zuvor gezeigte. Auch die Größe der unterschiedlichen Ansichten variiert hier sehr viel stärker. Ebenfalls auffällig ist die Tatsache, dass bei diesem Modell die vordere Seite des Roboters nur durch eine Ansicht repräsentiert ist. Alle anderen Ansichten zeigen entweder den hinteren oder den seitlichen Teil des Roboters.

Einen entscheidenden Hinweis auf solch starke Abweichung zu dem im Kapitel 3 gezeigten Modell liefert ein genauer der bei diesen Experimenten verwendeten Robotern. Obwohl beide Male der gleiche ER1 Roboter von Evolution Robotics zum Einsatz kam, wurde er doch an unterschiedlichen Stellen mit verschiedenen Zeitschriften beklebt. Im dem hier gezeigten Experiment wurde nur die vordere Seite des Roboters äußerlich modifiziert. In dem im Kapitel 3 gezeigten Experiment wurde zusätzlich noch der seitliche und der hintere Teil des Roboters mit Zeitschriften beklebt. Damit war es möglich auch aus diesen Perspektiven genügend markante SIFT-Merkmale zu extrahieren, so dass eine stabile Zuordnung zu leichten Abweichungen dieser Ansichten erfolgen konnte. In den in Abbildung 6.10 gezeigten Ansichten wurde zwar auch genügend Merkmale auf dem seitlichen und hinteren Teil des Roboters extrahiert, doch ist die Qualität dieser Merkmale nicht zu vergleichen mit der Qualität der Merkmale, die auf einem Zeitschriftenartikel gefunden werden.

Auf Zeitschriftenartikeln finden sich sehr viele Stellen, die eine eindeutige Struktur vorweisen können. Der nackte hintere Teil des Roboters liefert hier dagegen relativ wenig Spielraum. Auf den grauen Stäben, die das Gerüst des Roboters bilden, lassen sich kaum Merkmale mit eindeutiger Struktur finden. Die meisten aus dieser Perspektive extrahierten Merkmale sind daher qualitativ eher niederwertig und die relativ hohe Anzahl dieser lässt sich durch das vorhandene Bildrauschen erklären. Kompensiert wird diese Tatsache durch die entsprechend hohe Anzahl unterschiedlicher Ansichten im Modell des Roboters, die hauptsächlich seinen seitlichen oder hinteren Teil zeigen.

Die Ergebnisse dieses Experimentes liefern wichtige Einsichten darüber, welche Objekteigenschaften erforderlich sind, damit ein möglichst kompaktes Modell von diesem erzeugt werden kann. Besitzt eine Objektansicht kaum Struktur so verläuft auch die Konstruktion des Modells schwieriger. Viel mehr Ansichten müssen der Datenbank hinzugefügt werden, um ein vollständiges Modell zu erhalten.

Es ist an dieser Stelle anzumerken, dass das hier vorgestellte Verfahren komplett versagen würde, wenn die verschiedenen Ansichten des betrachteten Objektes hauptsächlich aus homogenen und gleichmäßig gefärbten Flächen bestehen würden. In so einem Fall wären die wichtigen Voraussetzungen zur Extraktion markanter Merkmale nicht erfüllt. Eine mögliche Lösung für dieses Problem wäre die Verwal-

tung der unterschiedlichen Objektansichten in Form von Farbhistogrammen. Zum Vergleich zweier Farbhistogramme wären die folgenden zwei Verfahren denkbar:

1. Berechnung des Bhattacharyya Abstandes [Bha43] zwischen den möglichen Kandidaten.
2. Interpretation der Farbhistogramme als eine Zeitreihe und Vergleich dieser mittels des Dynamic-Time-Warping Verfahrens (DTW) [RJ93].

Nichtsdestotrotz war es mit dem hier erstellten Robotermodell möglich, diesen während der gesamten Bildsequenz stabil zu verfolgen. Der nächste Abschnitt zeigt die hierbei erzielten Ergebnisse.

### 6.2.3 Konstruktion der Robotertrajektorie

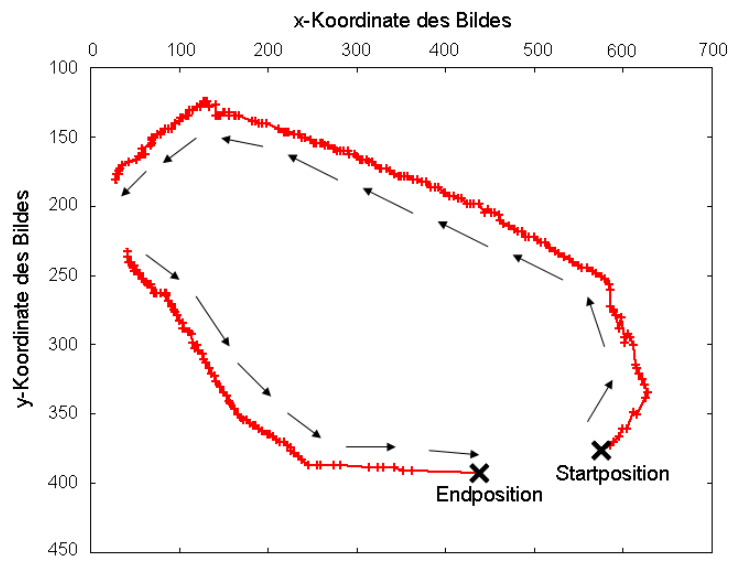
Die anschließende Verfolgung des Roboters im Bild der Weitsichtkamera erfolgte unter Zuhilfenahme des bereits im vorigen Abschnitt gezeigten Robotermodells. Dazu wurden zunächst mittels des in Kapitel 2 vorgestellten Vordergrund-Hintergrund Separationsverfahrens alle bewegten Objekte der Szene detektiert und nur die entsprechenden Bildausschnitte zum Vergleich mit den in der Datenbank gespeicherten Daten herangezogen. Der Vergleich der Bildausschnitte mit den verschiedenen Ansichten in der Datenbank basierte dabei auf den daraus extrahierten SIFT-Merkmalen. Ähnlich wie auch im Kapitel 3 kam hier die ERSP-Bibliothek zum Einsatz.

Abbildung 6.11 zeigt beispielhaft eine auf diese Weise konstruierte Robotertrajektorie im Bild der eingesetzten Weitsichtkamera. Zur besseren Darstellung zeigt die Abbildung (a) hier noch einmal den Sichtbereich der Kamera. Abbildung (b) zeigt die entsprechende Trajektorie des Roboters im Kamerabild. Ein Punkt in dieser Graphik entspricht dem Mittelpunkt des den Roboter umrandeten Rechtecks zu dem jeweiligen Zeitpunkt. Die Start- und Endposition sind durch schwarze Kreuze gekennzeichnet. Die Pfeile geben die Richtung der Roboterbewegung an. Auffallend ist hierbei die kurzzeitige Unterbrechung der roten Kurve in der linken oberen Ecke. Dieser Zustand lässt sich allerdings durch das Verschwinden des Roboters aus dem Sichtbereich der Kamera erklären. Vergleicht man noch einmal diese Graphik mit der in Abbildung (a) gezeigten Bildaufnahme, so stellt man fest, dass eben dieser Ausschnitt des Flurs außerhalb des überwachten Bereiches liegt.

An diesem Beispiel lässt sich sehr gut der entscheidende Vorteil dieses Verfolgungsverfahrens beobachten. Unter Verwendung des vorher konstruierten Modells ist es möglich die Trajektorie des entsprechenden Objektes auch dann fortzusetzen, wenn dieser zeitweise aus dem Sichtbereich der verwendeten Kamera verschwindet. Es ist allerdings an dieser Stelle auch anzumerken, dass die dazu erforderliche Detektion des Objektes stark von dem in der Datenbank gespeicherten Modell abhängt.



(a)



(b)

Abbildung 6.11: (a) Sichtbereich der Weitsicht-Kamera; (b) Trajektorie des Roboters im Kamerabild

Existieren nur vollständige Ansichten des Objektes, so kann die Detektion desjenigen versagen, wenn dieses nur teilweise sichtbar ist. Natürlich ist in dieser Situation auch entscheidend wieviele verwendbare SIFT-Merkmale aus dem entsprechenden Bildausschnitt extrahiert werden konnten. In dem oben gezeigten Beispiel trat dieser Fall gegen Ende der Roboterbewegung auf. In dem unteren Teil der Abbildung (b) ist deutlich zu sehen, dass die Punktdichte im letzten Abschnitt der Trajektorie stark abnimmt. An dieser Stelle war nur der obere Teil des Roboters sichtbar.

Abbildung 6.12 zeigt einige Aufnahmen der während dieses Experimentes aufgenommenen Bildsequenz. Die hierbei zu sehende Beschriftungen des Roboters sind Namen der jeweiligen Ansicht in der Datenbank, zu der die aktuellen Bilddaten am besten zugeordnet werden konnten. Daran ist deutlich zu erkennen, dass auch eine plötzliche Rotation des Objektes zu keinem Abbruch der Trajektorie führt. Lediglich im Bild (d) findet sich hier keine Beschriftung. Zu diesem Zeitpunkt ist der Roboter nur teilweise sichtbar und ist gerade dabei den Sichtbereich der Kamera wieder zu betreten.

Die Route des Roboters war bei diesem Experiment vorher nicht festgelegt. Vielmehr resultierte diese aus den vorprogrammierten Verhaltensregeln. Im Falle eines positiven Ergebnisses des Modul 'Entfernungsschätzung', führte der Roboter eine Linksrotation durch um dem Hindernis auszuweichen. Sonst war er auf eine einfache Geradeausfahrt vorprogrammiert.

Die Ergebnisse des Moduls 'Entfernungsschätzung' wurden bereits im Kapitel 5 ausführlich diskutiert. Aus diesem Grund wurde hier auf eine erneute Darstellung dieser verzichtet.

#### 6.2.4 Randbedingungen

Während der Durchführung der Fallstudie kristallisierten sich einige Randbedingungen heraus, deren Einhalten für einen erfolgreichen Einsatz des Systems notwendig ist.

- Das Verfahren zu Vordergrund-Hintergrund Separation detektiert erfolgreich bewegte Objekte, wenn deren Abbild im Kamerabild nicht kleiner ist als die im Vorfeld gewählte Größe für ein Bildbereich (vgl. Kapitel 2). Die Größe des Abbildes im Kamerabild hängt dabei zum einen von der realen Größe des Objektes selbst, zum anderen von seiner Entfernung zur Kamera ab. Die Größe des Bildbereiches (vgl. Kapitel 2) sollte daher entsprechend dem Einsatzgebiet gewählt werden. Zu beachten ist hierbei sowohl die maximal mögliche Entfernung von Objekten zum Kamerabild als auch deren minimal mögliche Größe.

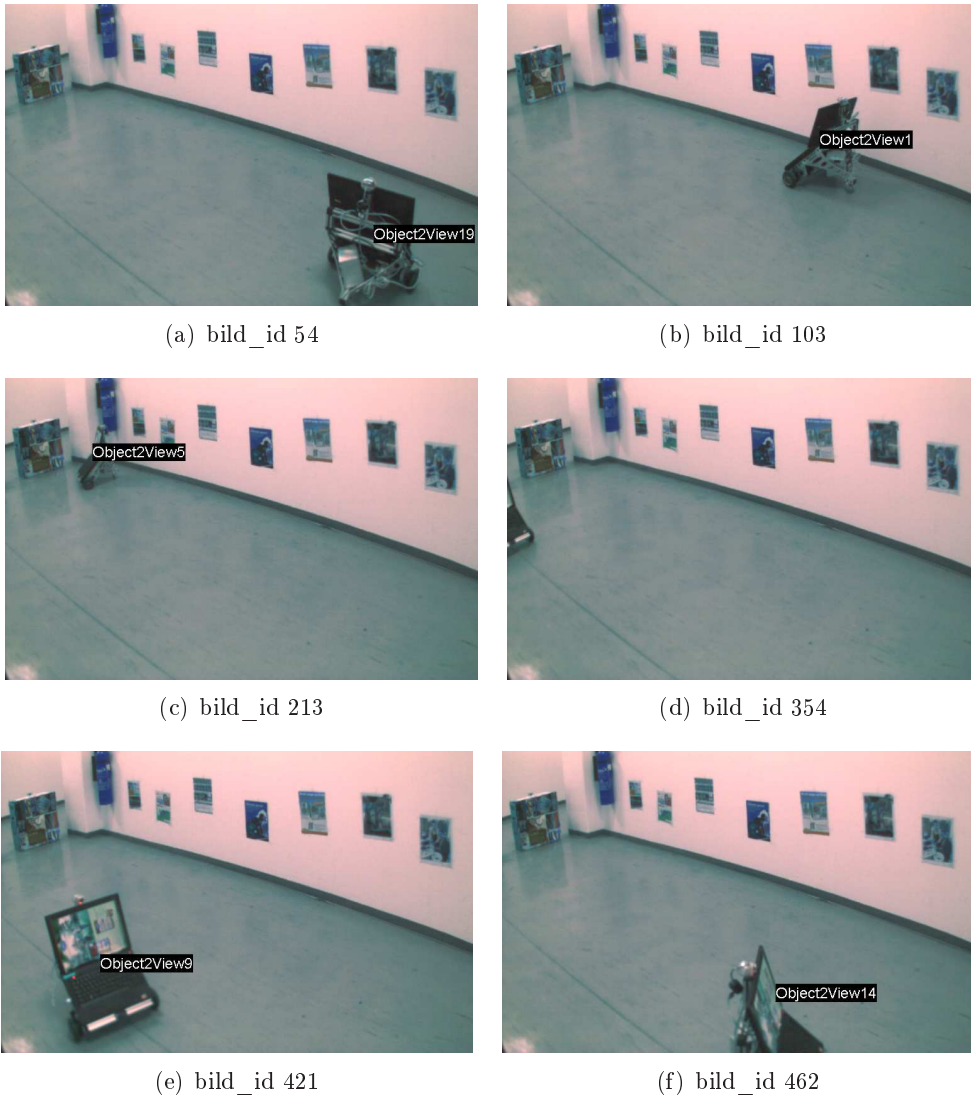


Abbildung 6.12: Konstruktion der Robotertrajektorie im Bild der Weitsichtkamera. Die Beschriftungen in den einzelnen Bildern entsprechen den Namen der Objektansichten in der Datenbank, zu denen die aktuellen Bilddaten am besten zugeordnet werden konnten.



- Eine erfolgreiche Objektwiedererkennung mittels SIFT-Merkmale (vgl. Kapitel 3) ist nur möglich wenn folgende zwei Bedingungen erfüllt sind: 1. Das Abbild des Objektes im Kamerabild weist soviel Struktur auf, dass genügend SIFT-Merkmale extrahiert werden können; 2. Das Objekt ist im Kamerabild deutlich zu erkennen. Als Orientierung sollte für das zweite Kriterium eine Bildgröße von  $40 \times 40$  Pixel dienen.
- Zur Anwendung des Verfahrens zur Berechnung von Time-To-Contact Werten (vgl. Kapitel 4 und 5) ist es notwendig, dass sowohl auf dem bevorstehenden Objekt als auch in der Umgebung genügend markante Merkmale extrahiert und verfolgt werden können. Zu viele homogene Flächen in unmittelbaren Umgebung resultieren in einem sehr sparsen Modell der Umgebung. Dies kann unter Umständen dazu führen, dass Roboter gegen ein bevorstehendes Hindernis stößt, obwohl das Verfahren zur Berechnung von Time-To-Contact Werten fehlerfrei funktioniert.

### 6.3 Diskussion und Zusammenfassung

Als gesamtes Ergebnis dieser Fallstudie lässt sich dokumentieren, dass die im Rahmen dieser Arbeit entwickelten Ansätze eine solide Basis für ein System bilden, welches in dem bereits oben beschriebenen Lagerhausszenario seine Anwendung finden könnte. Die Extraktion der bewegten Objekten in der Szene, Detektion des interessierenden Objektes aufgrund seines Bewegungsmusters und online Generierung des entsprechenden Modells sorgen dafür, dass der notwendige Installationsaufwand des Systems auf ein Minimum begrenzt wäre.

Gleichzeitig bildet das Modul 'Entfernungsschätzung', welches lokal auf dem Roboter ausgeführt wird, zusammen mit der im Bild der Weitsichtkamera konstruierten Objekttrajektorie eine vernünftige Grundlage für eine stabile Navigation des Roboters zu einem vorher festgelegten Zielpunkt. Im Falle einer Abweichung von der optimalen Route würde das System dem Roboter mitteilen, wie dieser seine Bewegungsrichtung ändern sollte. Der Roboter würde dann unter Einbezug der ihm zur Verfügung stehenden Information des Moduls 'Entfernungsschätzung' die für ihn zu diesem Zeitpunkt optimale Entscheidung treffen. Diese Art der Datenfusion unter Berücksichtigung vorher festgelegter Prioritäten wurde bereits im Rahmen des am Lehrstuhl 'Intelligente Systeme' durchgeführten Praxisprojektes realisiert. Aus diesem Grund kann an dieser Stelle mit Zuversicht behauptet werden, dass ein solches System schon bereits in naher Zukunft seine praktische Anwendung finden könnte.



# Zusammenfassung und Ausblick

---

Ziel der vorliegenden Dissertation war die Entwicklung eines robusten, flexiblen und zugleich auch leicht zu montierenden Systems zur sicheren Navigation eines Roboters in einer zunächst unbekanntem Umgebung. Eines der grundlegenden Bedingungen war dabei die ausnahmslose Beschränkung der Robotersensorik auf Videokameras. Sowohl die Berechnung der Robotertrajektorie zum Zielpunkt, als auch die Schätzung der Entfernung zu den bevorstehenden Hindernisobjekten hatte ausschließlich auf der Auswertung der gesammelten Bildinformation zu erfolgen.

Das resultierende System sollte seinen Ansatz im Rahmen eines Lagerhaussszenarios finden. Eine an der Decke montierte Weitsichtkamera überschaute einen größeren Bereich der Szene und hat sowohl den Roboter als auch die gewünschte Zielposition in ihrem Sichtbereich. Die Bildinformation dieser Kamera dient zur groben Berechnung der Robotertrajektorie zum Zielpunkt. Die Detektion und auch die Umgehung der auf dieser Route befindlichen Hindernissen erfolgt in diesem Szenario mittels einer direkt auf dem Roboter montierten Kamera.

Zur Realisierung dieses Systems wurde im Rahmen der vorliegenden Arbeit in den drei Teilgebieten der Objektdetektion, Objektwiedererkennung und Entfernungsschätzung bereits bestehende Ansätze erweitern und neue Verfahren konzipiert.

Die initiale Erkennung des zu navigierenden Objektes erfolgte hier in zwei Schritten. Zunächst wurden alle bewegenden Objekte der in der Deckenkamera abgebildeten Szene mittels eines Vordergrund- Hintergrund Separationsverfahrens extrahiert. Die anschließende Selektion des interessierenden Objektes basierte auf der Erkennung eines vorher spezifizierten Bewegungsmusters, welches während der Initialisierungsphase von dem Objekt ausgeführt wurde.

Dazu wurde im Rahmen dieser Arbeit ein neues Vordergrund- Hintergrund Separationsverfahren entwickelt. Die Konstruktion des Hintergrundmodells erfolgt hier auf der Basis der Grauwertstruktur der abgebildeten Szene. Die Struktur wird mittels der diskreten Kosinustransformation ermittelt und in Form von ausgewählten Koeffizienten zur weiteren Verarbeitung im Modell gespeichert. Die Koeffizientenauswahl erfolgt hier im Sinne der Rauschunterdrückung. Koeffizienten, die die feineren Details der Szene repräsentieren, werden bei der Modellbildung nicht berücksichtigt.

Zur Modellierung eines dynamischen Hintergrunds der Szene werden folgend dem Ansatz von Staufer und Grimson mehrere unterschiedliche Koeffizientenvektoren zur Repräsentation eines Bildbereichs verwendet. Das Hinzufügen neuer Vektoren oder das Entfernen bereits vorhandener aus dem Modell erfolgt hier online und hängt entscheidend von der Dynamik der Szene ab.

Zur Detektion von Vordergrundobjekten wird bei diesem Verfahren sowohl die örtliche als auch zeitliche Information einer Objektbewegung ausgenutzt. Ein Bildbereich wird hier nur dann als 'zum Vordergrund gehörend' markiert, wenn mindestens einer seiner benachbarten Bildbereiche im vorigen Bild als 'vom Hintergrund abweichend' markiert worden war. Das auf diese Weise generierte Vordergrundbild erwies sich als sehr stabil gegenüber äußeren Rauscheinflüssen.

Die Performanz dieses Verfahrens wurde im Rahmen einer umfangreichen Experimentenreihe mit den Ergebnissen dreier weiteren in der Literatur oft verwendeten Verfahren ([LHE03], [KB01] und [CXS<sup>+</sup>07]) verglichen. Dabei wurde unter Zuhilfenahme ausgewählter Bildsequenzen untersucht wie stabil die Verfahren auf plötzliche und sukzessive Lichtänderungen reagieren, wie sie mit zeitweisen Verschiebungen von Hintergrundobjekten umgehen, wie robust sie leichte periodische Bewegungen im Hintergrund der Szene handhaben und wie schnell sich das Hintergrundmodell der jeweiligen Verfahren an die geänderten Bedingungen anpasst.

Die hierbei erzielten Ergebnisse zeugten davon, dass das im Rahmen dieser Arbeit entwickelte Verfahren in den meisten Situationen den anderen Ansätzen vorzuziehen ist. In allen Testreihen lieferte dieses Verfahren bessere oder mit den anderen Ansätzen vergleichbare Ergebnisse. Das Verfahren war auch in der Lage Objekte dort zu erkennen, wo andere Ansätze komplett versagten. Üblicherweise waren dies Bildbereiche mit wenig Kontrast zwischen dem Objekt und dem Hintergrund der Szene.

Der hohe Grad an Stabilität und Robustheit dieses Verfahrens gegenüber äußeren Rauscheinflüssen wird allerdings durch die Tatsache kompensiert, dass die Konturen der extrahierten Objekte nicht akkurat sind. Im manchen Anwendungen kann dies negative Auswirkungen auf die Performanz des Gesamtsystems haben. Mögliche Verbesserung dieses Mankos wäre die Kombination dieses Verfahrens mit einem bereits bestehenden Ansatz, der die Objektkonturen genau extrahiert. Das hier entwickelte Verfahren würde dann zur Vorselektion der relevanten Bildbereiche dienen.

Wie bereits erwähnt, basierte die Selektion des zu navigierenden Objektes auf der Erkennung eines vorher spezifizierten Bewegungsmusters. Dazu wurde ein Verfahren entwickelt, welches die Erkennung von periodischen Bewegungen erlaubt. Die Einschränkung auf periodische Bewegungsmuster erfolgte dabei aus zwei Gründen.

Zum einen ist es die Einfachheit der Bewegung. Jedes autonom agierende Objekt ist in der Lage diese durchführen. Zum anderen lässt sich eine geordnete und vorhersagbare Bewegung einfacher und damit auch stabiler erkennen, als dies beispielsweise bei einer komplizierten Bewegungskonformation der Fall wäre. Das hier entwickelte Verfahren war in der Lage die Periodizität sowohl einer einfachen Vor- und Rückwärtsbewegung als auch einer vom Roboter durchgeführten Kreisbewegung zu erkennen.

Ein weiterführender Schritt in diese Richtung wäre die Erarbeitung eines Ansatzes für die Erkennung von schwierigeren Bewegungsabläufen. Solche Verfahren wären interessant bei der Feststellung von Bewegungsanomalien zum Beispiel in dem Bereich der Kameraüberwachung in Kaufhäusern oder öffentlichen Parkplätzen.

Die dazu benötigten Objekttrajektorien wurden hier mittels eines einfachen Ansatzes zur Objektverfolgung ermittelt. Es basiert auf der Erkennung von korrespondierenden SIFT-Merkmalen in zwei aufeinander folgenden Bildern einer Kameraaufnahme. Dieses Verfahren zur Objektverfolgung liefert stabile Ergebnisse, wenn sich das äußere Erscheinungsbild des Objektes von Bild zu Bild wenig ändert. Im Rahmen der hier betrachteten Anwendung hatte diese Annahme keine nennenswerten Konsequenzen. Dies liegt unter anderem daran, dass für die initiale Erkennung des interessierenden Objektes keine zeitlichen Begrenzungen gesetzt waren. Im Falle einer fehlgeschlagenen Erkennung bewegte sich das Objekt solange weiter, bis dieses von dem System als das gesuchte Objekt detektiert wurde.

Die so gewonnene erste Ansicht des zu navigierenden Objektes bildete zugleich auch die Basis für ein weiteres im Rahmen dieser Arbeit entwickeltes Verfahren. Dieses Verfahren verfolgt das Ziel der robusten Objektverfolgung und dient im Rahmen des betrachteten Szenarios dem Zweck der Roboterverfolgung und damit auch zur Korrektur seiner Trajektorie zum Zielpunkt. Den Kern des Verfahrens bildet ein Modul, welches bereits im Online-Betrieb ein Modell von dem interessierenden Objekt erstellt und dieses im weiteren Verlauf des Systems immer weiter adaptiert. Die eigentliche Objektverfolgung findet dann mittels eines Vergleichs zwischen den aktuellen Bilddaten der Kamera und denen in der Datenbank gespeicherten Objektmodellen.

Die unterschiedlichen 'multiview appearance' Modelle werden in der Datenbank als eine Ansammlung von unterschiedlichen Ansichten des entsprechenden Objektes gespeichert. Eine Ansicht eines Objektes wird in Form der daraus extrahierten SIFT-Merkmale repräsentiert. Das entsprechende Modell wird nur dann um weitere Ansichten erweitert, falls die neu gewonnenen Ansichten nicht ausreichend durch die bereits im Modell gespeicherten Ansichten abgedeckt werden. Die Anzahl der im Modell gespeicherten Ansichten hängt damit entscheidend von der Art der Bewegung

ab, die das Objekt durchführt während das System aktiv ist. Führt das Objekt während des gesamten Betriebs keine Rotation durch, so ist meistens nur eine Ansicht dieses Objektes ausreichend. Die im Rahmen der Arbeit durchgeführten Experimente haben gezeigt, dass bei einer rotierenden Objektbewegung das so erstellte Modell durchaus mit einem mittels eines Rotationstellers konstruierten Modell vergleichbar ist. Ein positiver Effekt des hier vorgestellten Verfahrens ist allerdings, dass die Skalierung der Objektansichten ideal an die Distanz der Kamera zum Objekt angepasst ist.

Der entscheidende Vorteil dieses Verfahrens ist auch, dass das auf diese Weise konstruierte Objektmodell optimal an die jeweilige Anwendungsumgebung angepasst ist. Abhängig von der Position und Orientierung der Kamera werden in der Datenbank nur solche Objektansichten gespeichert, die auch von dieser Kamera aufgenommen werden können. Die Verwendung einer Deckenkamera würde beispielsweise in einem Modell resultieren, welches ausschließlich Ansichten aus der Vogelperspektive beinhaltet.

Ein weiterer Vorteil gegenüber anderen Verfahren ist die hierbei nicht benötigte Trainingsphase. Üblicherweise findet diese im Offline-Betrieb statt und setzt voraus, dass alle für die Verfolgung relevanten Objekte vorab bekannt sind. In vielen Anwendungen kann eine solche Annahme allerdings oft nicht erfüllt werden. In solchen Fällen würde das eben vorgestellte Verfahren die einzige Alternative bieten.

Den letzten wichtigen Beitrag dieser Arbeit bildet das Modul 'Entfernungsschätzung'. Ausgehend von der Bildinformation der auf dem Roboter montierten Kamera und seiner Bewegung wurde ein Verfahren entwickelt, welches eine robuste Berechnung von Time-To-Contact (TTC) Werten erlaubt. Die bessere Performanz dieses Verfahrens gegenüber anderen Ansätzen resultiert aus der Tatsache, dass hier die Berechnung der Werte nicht ausschließlich auf den Messdaten der extrahierten Bildpunkte basiert. Im Rahmen dieser Arbeit wurden Modellgleichungen hergeleitet, die die Positionsänderungen der relevanten Punkte in Bildern der Kamera während der Roboterbewegung beschreiben. Als Grundlage wurden hierzu nur die bekannte Roboterbewegung sowie die allgemein gültigen Gesetze der projektiven Geometrie herangezogen.

Bei einer richtigen Wahl der entsprechenden Modellparameter war es damit möglich die theoretisch richtige Positionen der Bildpunkte vorherzusagen. Für die Adaption der Parameter wurden zwei unterschiedliche Ansätze hergeleitet, die jeweils auf den historischen Daten der bis zum aktuellen Zeitpunkt gemessenen Punktpositionen basieren. Ausgehend von den in der Simulation erzielten Ergebnissen wurde nur eines dieser Verfahren während des praktischen Ansatzes im Rahmen einer Laborumgebung getestet. Die dabei erzielten Ergebnisse zeugen von der bereits erwähnten

Stabilität und dem hohen Grad der Genauigkeit in den berechneten TTC-Werte. Ausgestattet mit einer einfachen Web-Kamera war der Roboter unter Verwendung dieses Verfahrens in der Lage über einen längeren Zeitraum zuverlässig den bevorstehenden Hindernissen auszuweichen. Anders als in anderen Arbeiten berichtet, war hier die Verwendung von weiteren Robotersensoren nicht notwendig.

Zum Ende der Arbeit wurden alle hier vorgestellten Verfahren im Rahmen einer Fallstudie auf die Möglichkeit eines praktischen Ansatzes untersucht. Der dazu verwendete Experimentenaufbau entsprach dem bereits oben beschriebenen Szenario. Ergebnisse dieser Fallstudie haben gezeigt, dass unter Verwendung der im Rahmen dieser Arbeit entwickelten Verfahren ein erfolgreicher Einsatz in einem realen Lagerhaus mit autonom agierenden Robotern durchaus schon bald Realität werden könnte. Ein weiterer logischer Schritt dieser Arbeit wäre die Realisierung dieses Vorsatzes.





# Abbildungsverzeichnis

2.1	(a) anhand von Trainingsbildern erstelltes Hintergrundmodell der Szene; (b) aktuelles Kamerabild; (c) Ergebnisbild, welches nur die Vordergrund Objekte enthält. . . . .	8
2.2	Graphische Darstellung der zweidimensionalen Basisfunktionen der diskreten Kosinustransformation. . . . .	11
2.3	3D Visualisierung der (a) Originalwerte im 8x8 Block und (b) der entsprechenden Koeffizienten der diskreten Kosinustransformation. . . . .	12
2.4	Koeffiziente, der durch Kreuze markierten Basisfunktionen, werden in einem Vektor als das Hintergrundmodell des jeweiligen Bildausschnitts gespeichert. . . . .	14
2.5	(a) Originalaufnahme eines Büroraumes; (b) Visualisierung des entsprechenden Hintergrundmodells mittels der inversen Kosinustransformation und nur unter Berücksichtigung der ausgewählten Koeffizienten. . . . .	15
2.6	(a) Originalaufnahme einer Straßenszene; (b) Resultierendes Vordergrundbild, ohne Berücksichtigung der zweiten Bedingung. Während der Aufnahmen waren in der Szene keine bewegenden Objekte vorhanden. . . . .	18
2.7	Graphische Darstellung des Zusammenwirkens zwischen dem Vordergrund- und dem Hintergrundmodell des Bildbereichs $k$ . Die Zusammenlegung beider Modelle zum 'neuen' Hintergrundmodell des Bereichs ist situationsabhängig. . . . .	20
2.8	Graphische Darstellung eines sich langsam bewegenden Objektes. Ohne Berücksichtigung der zweiten Bedingung würden bei einer Wahl von $N = 3$ Teile dieses Objektes bereits im vierten Bild in den Hintergrund der Szene eingebaut. . . . .	21
2.9	Einige Aufnahmen aus dem Video der Parkplatzüberwachung. Anhand dieses Videos wurde untersucht wie gut das hier vorgestellte Verfahren das Hintergrundmodell der Szene bei zeitweisen Verschiebungen von Objekten aktualisiert. . . . .	25
2.10	Einige Bilder aus den Aufnahmen eines Aufenthaltsbereich in einem Bürogebäude. Anhand dieses Videos [LiT] wurde untersucht wie stabil die Verfahren gegenüber plötzlichen Lichtänderungen sind. . . . .	25

2.11	Einige Bilder aus den Aufnahmen eines Wohnzimmers. Anhand dieses Videos [KT99] wurde die Robustheit der Verfahren gegenüber sukzessiven Lichtänderungen untersucht. . . . .	26
2.12	Einige Aufnahmen aus dem Video mit einem schwankenden Baum. Anhand dieses Videos [KT99] wurde getestet, wie gut die Verfahren mit einem dynamischen Hintergrund einer Szene umgehen können. . . . .	26
2.13	Einige Aufnahmen aus dem Bereich der Bahnhofüberwachung. Anhand dieses Videos [PET05] wurde untersucht wie gut die Verfahren Personen vom Hintergrund der Szene extrahieren können, auch wenn nur wenig Kontrast vorhanden ist. . . . .	27
2.14	Ergebnisse des im Rahmen der Arbeit entwickelten Ansatzes im Bereich der Parkplatzüberwachung. Die Vordergrundobjekte bilden hier alle durch weiße Rechtecke markierte Bereiche. . . . .	28
2.15	Vergleich der Ergebnisse kurz nachdem das Licht ausgeschaltet wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	30
2.16	Detektion von Personen im abgedunkelten Raum, kurz nachdem das Licht ausgeschaltet wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	32
2.17	Detektion von Personen im abgedunkelten Raum nachdem die Adaptionsphase bei allen Verfahren abgeschlossen wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	32
2.18	Vergleich der Ergebnisse kurz nachdem das Licht wieder eingeschaltet wurde: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	33
2.19	Erneute Detektion von Personen in dem nun wieder beleuchteten Raum: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	34
2.20	Vergleich der Detektionsergebnisse bei sukzessiven Lichtänderungen: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	35
2.21	Performanz der Verfahren bei einem dynamischen Hintergrund der Szene: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	36

2.22	Graphische Darstellung der Größenänderung der Hintergrundmodelle einiger ausgewählter Bereiche. 'Position $n \times m$ ' bezeichnet die Position des entsprechenden Bereiches im Bild. Alle ausgewählten Bereiche repräsentieren den dynamischen Teil des Hintergrunds. . . . .	37
2.23	Performanz der Verfahren beim kaum vorhandenen Kontrast zwischen den Personen und dem Hintergrund der Szene im oberen Bereich des Bildes: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	38
2.24	Performanz der Verfahren bei der Selektion von Personenschatten: (a) das originale Kamerabild; (b)-(e) resultierende Vordergrundbilder der entsprechenden Verfahren. . . . .	39
3.1	Ergebnis der Detektion des bewegenden Objekts: (a) der rote Rechteck wurde mittels des Moduls 'Komponenten Extraktion' berechnet; (b) Vordergrundbild des im Kapitel 2 vorgestellten Vordergrund-Hintergrund Separationsverfahrens. . . . .	52
3.2	Eingabe und Ergebnis des SIFT-Verfahrens: (a) als Eingabe dienen ausschließlich Bildausschnitte, die die bewegenden Objekte der Szene repräsentieren; (b) die grünen Punkte kennzeichnen hier die Positionen der extrahierten SIFT-Merkmale. . . . .	53
3.3	Beispiel für eine, während einer periodischen Bewegung resultierenden Kurve. Die ursprünglichen Daten der Trajektorie wurde hierfür in ein entsprechendes Format transformiert. . . . .	55
3.4	Bei den Experimenten oft beobachtete Situation: Die Anzahl und die Art, der auf einem Objekt extrahierten SIFT-Merkmale variieren von Bild zu Bild. . . . .	55
3.5	Grafische Repräsentation des prinzipiellen Vorgehens bei der online Modell Generierung. . . . .	59
3.6	Einige Aufnahmen einer Bildsequenz während der Roboterbewegung: 1. Reihe: Originalaufnahmen der Kamera zu unterschiedlichen Zeitpunkten; 2. Reihe: Vordergrundbilder des im Kapitel 2 vorgestellten Separationsverfahrens; 3. Reihe: Ergebnisse des Moduls 'Komponenten Extraktion'. . . . .	62
3.7	(a) zeigt im Kamerabild aufgenommene Objekttrajektorie einer Vor- und Rückwärtsbewegung, die insgesamt drei mal durchgeführt wurde; (b) zeigt die entsprechende periodische Kurve, die ausgehend von den Daten im Bild (a) berechnet wurde. . . . .	64

3.8	(a) zeigt im Kamerabild aufgenommene Objekttrajektorie einer kreisförmigen Bewegung, die insgesamt drei mal durchgeführt wurde; (b) zeigt die entsprechende periodische Kurve, die ausgehend von den Daten im Bild (a) berechnet wurde. . . . .	65
3.9	(a) zeigt im Kamerabild aufgenommene Objekttrajektorie einer rechteckigen Bewegung, die insgesamt drei mal durchgeführt wurde; (b) zeigt die entsprechende periodische Kurve, die ausgehend von den Daten im Bild (a) berechnet wurde. . . . .	66
3.10	Unterschiedliche Ansichten eines Modells des Roboters, die online während der Roboterbewegung konstruiert wurden. . . . .	67
4.1	Graphische Darstellung zur Entstehung des optischen Flusses. . . . .	73
4.2	Optischer Fluss: die durch Pfeile gekennzeichneten Verschiebungsvektoren bilden den Optischen Fluss. Der hier dargestellte Fluss entsteht bei einer geraden Bewegung der Kamera in Richtung des FOEs. . . . .	74
4.3	Simulationsergebnisse mit exakten Daten: (a) zeigt den generierten optischen Fluss; (b) zeigt die richtigen und die berechneten TTC-Werte	78
4.4	Linke Spalte zeigt drei in der Simulation verrauschte Punktsequenzen. Dabei wurde jeweils unterschiedliche Varianz der Streuung gewählt: 0.1, 0.3 und 0.75. Die rechte Spalte zeigt die entsprechenden TTC-Werte. . . . .	79
4.5	(a) zeigt die Abstände zwischen zwei Bildpunkten einer exakten und einer verrauschten Sequenz. (b) zeigt die entsprechenden TTC-Werte.	80
4.6	Entstehungsprozess einer Bildpunkt Folge. . . . .	84
4.7	(a) zeigt die Abstände zwischen den Punkten der in der Simulation erzeugten Punktsequenz und der mittels der Gleichungen 4.6 4.7 konstruierten Folge. (b) zeigt die entsprechenden TTC-Werte. . . . .	86
4.8	(a) zeigt eine Punktfolge, die mit Gaussrauschen und Varianz von 0.1 modifiziert wurde. (b) zeigt die richtigen TTC-Werte und solche, die mit dem fehlerminimierenden Verfahren berechnet wurden. . . . .	93
4.9	(a) und (b) zeigen unterschiedlich skaliert die Oberfläche der Fehlerfunktion $F_x(a_x, b_x)$ für die Punktfolge in Abbildung 4.8. . . . .	94
4.10	Fehlerfunktionen $F_x^n(a_x, b_x)$ für $n \in \{10, 20, 25, 30, 35, 39\}$ . . . . .	95
4.11	Fünf unterschiedliche Punktfolgen, welche mit den Modellgleichungen 4.6 und 4.7 konstruiert wurden. Bei allen, außer bei der mittleren Punktfolge wurden die Modellparameter etwas verrauscht. . . . .	97
4.12	Einfluss der initialen Entfernung des 3D Punktes auf die Länge der Adaptionsphase. . . . .	99

---

4.13	Wirkung des Rauscheinflusses auf die Länge der Adaptionphase. . .	100
5.1	Entwicklung der Feature Positionen bei variierender Geschwindigkeit der Kamerabewegung. . . . .	104
5.2	Graphische Darstellung der Veränderung des TTC Abstandes zwischen dem 3D-Punkt $P$ und der rotierenden Bildebene der Kamera. .	107
5.3	Graphische Darstellung einer Kurvenfahrt, bei der der translatorische Anteil der Bewegung stärker ausgeprägt ist als der rotierende Anteil.	108
5.4	Graphische Darstellung einer Kurvenfahrt. In dieser Situation ist der rotierende Anteil der Bewegung stärker ausgeprägt als der translatorische Anteil. . . . .	110
5.5	(a) zeigt das Kamerabild welches von der Startposition des Roboters aufgenommen wurde. (b) zeigt graphisch den maßstabgetreuen Aufbau der Versuchsumgebung. . . . .	111
5.6	Unterschiedliche Ansichten des eingesetzten ER1 Roboters der Firma Evolution Robotics. . . . .	112
5.7	(a) zeigt das am Anfang der Sequenz aufgenommene Kamerabild; (b) zeigt die Entwicklung der $y$ -Koordinate eines Punktes nahe dem FOE während der gesamten Sequenz. . . . .	113
5.8	(a) Der rot umrandete Rechteck markiert den Bereich aus dem einige Merkmale für die TTC-Berechnung ausgewählt wurden. (b) Graphische Darstellung der bereits in Distanz umgerechneten TTC-Werte. .	114
5.9	In Distanz umgerechneten TTC-Werte: (a) bei der Berechnung wurden beide Parameterpaare $(a_x, b_x)$ und $(a_y, b_y)$ berücksichtigt; (b) bei der Berechnung wurde nur das gut initialisierte Parameterpaar berücksichtigt. . . . .	116
5.10	In Distanz umgerechnete TTC-Werte verschiedener Merkmalssequenzen mit dem gleichen Abstand zum Hindernis. Als Merkmal wurden (a) SIFT-Merkmale und (b) KLT-Merkmale verwendet. . . . .	119
5.11	Resultierende Histogramme der beiden Kriterien bei einer Robotergeschwindigkeit von $2\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale; (c) und (d) sind Histogramme für SIFT-Merkmale. . . .	121
5.12	Resultierende Histogramme der beiden Kriterien bei einer Robotergeschwindigkeit von $4\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale; (c) und (d) sind Histogramme für SIFT-Merkmale. . . .	122
5.13	Resultierende Histogramme der beiden Kriterien bei einer Robotergeschwindigkeit von $6\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT-Merkmale; (c) und (d) sind Histogramme für SIFT-Merkmale. . . .	123

5.14	Resultierende Histogramme der beiden Kriterien bei einer Roboter- geschwindigkeit von $8\text{ cm/s}$ ; (a) und (b) sind Histogramme für KLT- Merkmale; (c) und (d) sind Histogramme für SIFT-Merkmale. . . . .	124
5.15	Für die Experimente verwendeter Flur: (a) maßstabgetreuer Grund- riss; (b) Fotoaufnahme des Flurs . . . . .	128
5.16	Graphische Darstellung einiger vom Roboter gefahrenen Trajektori- en. Als Grundlage dienten für diese Graphiken die entsprechenden Odometriedaten des Roboters. . . . .	129
6.1	Behavior Netzwerk zur online Generierung eines Objektmodells. Nach dem Umschalten der Komponenten wurde dieses Netzwerk auch zum Verfolgen eines Objektes mittels eines bereits generierten Modells be- nutzt. . . . .	135
6.2	Behavior Netzwerk zur Erkennung einer periodischen Bewegung. . .	135
6.3	Behavior Netzwerk zur Berechnung von Time-To-Contact Werten zu den bevorstehenden Hindernisse. . . . .	136
6.4	Für die Experimente verwendeter Flur: (a) Weitaufnahme des Flurs; (b) Nahaufnahme des Roboters und der verwendeten UEye-Kamera auf dem Stativ. . . . .	138
6.5	Für die Experimente verwendete Weitsichtkamera: (a) Sichtbereich der Weitsichtkamera; (b) als Weitsichtkamera verwendete UEye-Kamera.	138
6.6	Die Positionen der während der Roboterbewegung berechneten SIFT- Merkmale sind hier durch kurze rote Fäden dargestellt. Ein Pfad ver- bindet korrespondierende SIFT-Merkmale in aufeinander folgenden Bildern. . . . .	140
6.7	Periodische Kurve, die bei der Vor- und Rückwärtsbewegung des Ro- boters entstanden ist. . . . .	141
6.8	Linke Bilder: Ergebnisse des in Kapitel 2 vorgestellten Vordergrund- - Hintergrund Separationsverfahrens zusammen mit der Berechnung des achsenparallelen Rechtecks um den bewegenden Roboter; rechte Bilder: Detektion des Roboters aufgrund der von ihm durchgeführten periodischen Bewegung. . . . .	143
6.9	Ausschnitt aus dem Protokoll, welches die einzelnen Schritte der Mo- dellgenerierung dokumentiert. . . . .	144
6.10	Unterschiedliche Ansichten des online generierten Robotermodells. Die grünen Punkte entsprechen den Positionen der hierbei extrahier- ten SIFT-Merkmale. . . . .	145

---

6.11 (a) Sichtbereich der Weitsicht-Kamera; (b) Trajektorie des Roboters im Kamerabild . . . . .	148
6.12 Konstruktion der Robotertrajektorie im Bild der Weitsichtkamera. Die Beschriftungen in den einzelnen Bildern entsprechen den Namen der Objektansichten in der Datenbank, zu denen die aktuellen Bild- daten am besten zugeordnet werden konnten. . . . .	150





# Tabellenverzeichnis

2.1	Laufzeit Performanz des hier entwickelten Vordergrund-Hintergrund Separationsverfahrens in Abhängigkeit von der verwendeten Hardware. Als Evaluierungsgrundlage wurde eine Sequenz mit 1000 Bildern mit $640 \times 480$ - Auflösung verwendet. . . . .	40
5.1	Initiale Parameter $a_x$ , $a_y$ , $b_x$ und $b_y$ einiger ausgewählter guter und schlechter Sequenzen, die bei dem in Abbildung 5.8 gezeigten Versuch entstanden sind. . . . .	115
5.2	Ergebnisse der Roboternavigation im geschlossenen Raum . . . . .	129



# Literaturverzeichnis

- [Alo93] Y. Aloimonos. Active vision revisited. In *Active Perception*, pages 1–18, 1993. 2
- [Avi04] S. Avidan. Support vector tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(8):1064–1072, 2004. 49
- [AWB87] Y. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. 1(4):333–356, 1987. 2
- [BB92] D. H. Ballard and C. M. Brown. Principles of animate vision. *CVGIP: Image Underst.*, 56(1):3–21, 1992. 2
- [BBH<sup>+</sup>89] P.J. Burt, J.R. Bergen, R. Hingorani, R.J. Kolczynski, W.A. Lee, A. Leung, J. Lubin, and J. Shvaytser. Object tracking with a moving camera; an application of dynamic motion analysis. In *IEEE Workshop on Visual Motion*, 1989. 81
- [BBHP90] J.R. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. Computing two motions from three frames. In *Proc. 3rd Int. Conf. on Computer Vision*, pages 27–32, 1990. 81
- [BBPW04] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *European Conference on Computer Vision (ECCV). Prague, Czech Republic : Springer(LNCS)*, pages 25–36, 2004. 81
- [BC92] R. Bajcsy and M. Campos. Active and exploratory perception. *CVGIP: Image Underst.*, 56(1):31–40, 1992. 2
- [Bha43] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943. 147
- [Bir] Stan Birchfield. KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker. <http://www.ces.clemson.edu/~stb/klt/>. 130
- [Bir98] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:232, 1998. 48

- [BJ96] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *Computer Vision ECCV 96*, Lecture Notes in Computer Science, pages 329–342, Heidelberg, 1996. Springer Berlin. 49
- [BT80] S.T. Barnard and W.B. Thompson. Disparity analysis of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2(4):333–340, 1980. 81
- [BTG06] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *Proceedings of the 9th European Conference on Computer Vision*, 2006. 117
- [BWS05] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/kanade meets horn/schunck: combining local and global optic flow methods. *International Journal of Computer Vision*, pages 211–231, 2005. 81
- [BY92] A. Blake and A. Yuille. *Active Vision*. MIT Press, 1992. 2
- [Cam95] T. Camus. Calculating time-to-contact using real-time quantized optical flow. *Max-Planck-Institut fuer biologische Kybernetik, Arbeitsgruppe Buelthoff, Technical Report*, 14, 1995. 74
- [Can86] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679–714, 1986. 10, 23
- [CHHN95] D. Coombs, M. Herman, T. Hong, and M. Nashman. Real-time obstacle avoidance using central flow divergence and peripheral flow. *Computer Vision, IEEE International Conference on Computer Vision*, 0:276, 1995. 82
- [CRM03] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Trans. Pattern Analysis Machine Intell.*, 25:564–575, 2003. 48
- [CXS<sup>+</sup>07] Z. Chaohui, D. Xiaohui, Xu. Shuoyu, S. Zheng, and L. Min. An improved moving object detection algorithm based on frame difference and edge detection. In *ICIG '07: Proceedings of the Fourth International Conference on Image and Graphics*, pages 519–523, Washington, DC, USA, 2007. IEEE Computer Society. 10, 23, 29, 154
- [D.82] Marr. D. *Vision. A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman and Company, 1982. 2

- [DH02] P. Deuffhard and A. Hohmann. *Numerische Mathematik I. Eine algorithmisch orientierte Einführung. 3. überarbeitete und erweiterte Auflage.* de Gruyter: Berlin, New York, 2002. 117
- [Eng08] M. Engel. Entwicklung und Evaluierung eines Time-to-Contact Berechnungsverfahrens für Rotationsfahrten., 2008. 91, 92
- [FA95] C. Fermüller and Y. Aloimonos. Vision and action. 13(10):725–744, December 1995. 2
- [FB87] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. pages 726–740, 1987. 119
- [Fre74] H. Freeman. Computer processing of line-drawing images. *ACM Comput. Surv.*, 6(1):57–97, 1974. 52
- [HDC07] M. J. Hossain, M. A. A. Dewan, and O. Chae. Moving object detection for real time video surveillance: An edge based approach. *IEICE Transactions*, 90-B(12):3654–3664, 2007. 10
- [HHD99] T. Horprasert, D. Harwood, and L. Davis. A statistical approach for real-time robust background subtraction and shadow detection. *In: Proc. Seventh IEEE ICCV Frame-rate Workshop Corfu, GK, September*, pages 1–19, 1999. 22
- [HMS<sup>+</sup>07] S. Heymann, K. Müller, A. Smolic, B. Fröhlich, and T. Wiegand. Sift implementation and optimization for general-purpose gpu. pages 317–322, 2007. 69
- [HS81] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981. 81
- [HS88] C. Harris and M. Stephens. A combined corner and edge detection. *In Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988. 82
- [HYM07] B.K.P. Horn, F. Yajun, and I. Masaki. Time to contact relative to a planar surface. *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 68–74, 2007. 83
- [KB01] P. Kaewtrakulpong and R. Bowden. An improved adaptive background mixture model for realtime tracking with shadow detection. 2001. 22, 154

- [KO09] C. Kim and N. E. O'Connor. Using the discrete hadamard transform to detect moving objects in surveillance video. In *VISSAPP (2)*, pages 512–518, 2009. 10
- [KT99] Barry Brumitt Brian Meyers Kentaro Toyama, John Krumm. Test images used for wallflower paper. <http://research.microsoft.com/en-us/um/people/jckrumm/WallFlower/TestImages.htm>, 1999. 26, 160
- [Lee76] D. N. Lee. A theory of visual control of braking based on information about time-to-collision. *Perception*, 5:437–459, 1976. 74
- [LHE03] D. S. Lee, J. Hull, and B. Erol. A bayesian framework for gaussian mixture background modeling. In *Proceedings of IEEE Int. Conf. on Image Processing*, 3:973–976, 2003. 154
- [LHGT03] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Foreground object detection from videos containing complex background. In *In MULTIMEDIA 03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10. ACM Press, 2003. 23
- [LiT] Bitmap files of test sequences to evaluate methods for modelling complex backgrounds. [http://perception.i2r.a-star.edu.sg/bk\\_model/bk\\_index.html](http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html). 25, 159
- [Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision.*, pages 91–110, 2004. 47, 52, 82
- [Mal99] Stéphane Mallat. *A Wavelet Tour of Signal Processing, 2nd edition*. Academic Press, Inc., 1999. 13
- [MPNMS09] D. Müller, J. Pauli, C. Nunn, and S. Mueller-Schneiders. Time to contact estimation using interest points. *Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems*, 2009. 83
- [MS05] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005. 53, 117
- [MT77] F. Mosteller and J. W. Turkey. *Data Analysis and Regression: A Second Course in Statistics*. Addison Wesley; 1 edition, 1977. 119

- [Nag87] H.-H. Nagel. On the estimation of optic flow: Relations between different approaches and some new results. *Artificial Intelligence*, 33:299–324, 1987. 81
- [Nag88] H.-H. Nagel. Optical flow from spatiotemporal filters. *Int. Journal of Computer Vision*, pages 279–302, 1988. 81
- [Nag90] H.-H. Nagel. Extending the 'oriented smoothness constraint' into the temporal domain and the estimation of derivatives of optical flow. *In Proc. 1st European Conf. on Computer Vision*, pages 139–148, 1990. 81
- [OAVE93] B. A. Olshausen, C. H. Anderson, and D. C. Van Essen. A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *Journal of Neuroscience*, 13:4700–4719, 1993. 2
- [PCF05] N. Paragios, Y. Chen, and O. Faugeras. *Mathematical Models in Computer Vision: The Handbook, Chapter: Optical Flow Estimation*. Springer Verlag, 2005. 81
- [PET05] PETS: Performance Evaluation of Tracking and Surveillance. <http://www.cvg.rdg.ac.uk/slides/pets.html>, 2005. 8, 26, 27, 160
- [RE95] P. L. Rosin and T. Ellis. Image difference threshold strategies and shadow detection. In *BMVC '95: Proceedings of the 1995 British conference on Machine vision (Vol. 1)*, pages 347–356, Surrey, UK, UK, 1995. BMVA Press. 8
- [RJ93] L. R. Rabiner and B. Juang. *Fundamentals of speech recognition, Chapter 4*. Prentice-Hall, Inc., 1993. 147
- [RY90] R. Rao, K. and P. Yip. Discrete cosine transform: Algorithms, advantages, applications. *Academic Press, Boston*, 1990. 10
- [Sch93] W. X. Schneider. Space-based visual attention models and object selection: Constraints, problems, and possible solutions. *Psychological Research, Psychologische Forschung*, 56:35–43, 1993. 2
- [SCSP09] M. Sagrebin, D. L. Caparros, D. Stroh, and J. Pauli. Robust object tracking by simultaneous generation of an object model. In *VISSAPP (2)*, pages 392–397, 2009. 70

- [SFPG07] S. N. Sinha, J. M. Frahm, M. Pollefeys, and Y. Genc. Feature tracking and matching in video using programmable graphics hardware. *Machine Vision and Applications*, 2007. 69
- [SG00] C. Stauffer and W. Grimson. Learning patterns of activity using real time tracking. *IEEE Trans. Patt. Analy. Mach. Intell.*, 22(8):747–767, 2000. 9, 17, 22
- [Sin10] Sudipta Sinha. GPU-KLT: A GPU-based Implementation of the Kanade-Lucas-Tomasi Feature Tracker. [http://www.cs.unc.edu/~ssinha/Research/GPU\\_KLT/](http://www.cs.unc.edu/~ssinha/Research/GPU_KLT/), 2010. 130, 131
- [SK07] K. Souhila and A. Karim. Optical flow based robot obstacle avoidance. *International Journal of Advanced Robotic Systems*, 4(1):13–16, 2007. 81, 130
- [SKS08] Elhabian Shireen, Y., El-Sayed Khaled, M., and Ahmed Sumaya, H. Moving object detection in spatial domain using background removal techniques, state-of-art. *Recent Patents on Computer Science*, 1:32–54, 2008. 8
- [SNP08] M. Sagrebin, A. Noglik, and J. Pauli. Robust Time-to-Contact Estimation for Real Time Applications. In *Proceedings of International Conference on Computer Graphics*, pages 128–133, 2008. 102
- [SP09a] M. Sagrebin and J. Pauli. Improved time-to-contact estimation by using information from image sequences. In *21. Fachgespräch Autonome Mobile Systeme (AMS)*, 2009. 102
- [SP09b] M. Sagrebin and J. Pauli. Real-time moving object detection for video surveillance. In *In Proceedings of the Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 31–36, 2009. 43
- [Spa05] J. C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons, Inc., 2005. 89
- [SPH08] M. Sagrebin, J. Pauli, and J. Herwig. Behavior based robot localisation using stereo vision. In *Rob Vis*, pages 427–439, 2008. 70
- [ST94] J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994. 48, 82, 117



- [TK91] C. Tomasi and T. Kanade. Detection and tracking of point features. *Carnegie Mellon University Technical Report CMU-CS-91-132*, 1991. 117
- [Wu07] Changchang Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). <http://cs.unc.edu/~ccwu/siftgpu>, 2007. 69
- [YJS06] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4), 2006. 48, 63
- [ZCZX08] Q. Zhang, Y. Chen, Y. Zhang, and Y. Xu. Sift implementation and optimization for multi-core systems. *In Proceedings of the IEEE International Symposium on In Parallel and Distributed Processing.*, pages 1–8, 2008. 69
- [ZPB07] C. Zach, T. Pock, and H. Bischof. A duality based approach for real-time tv-l1 optical flow. *In: Pattern Recognition Proc. DAGM. Heidelberg, Germany*, pages 214–223, 2007. 81
- [ZT98] D. Ziou and S. Tabbone. Edge detection techniques: an overview. *International Journal on Pattern Recognition and Image Analysis*, 8:537–559, 1998. 10



# Versicherung an Eides Statt

Ich, Maria Sagrebin-Mitzel, Matrikelnummer 596002, wohnhaft in 53757 St. Augustin, versichere an Eides Statt durch meine Unterschrift, dass ich die vorstehende Arbeit selbständig und ohne fremde Hilfe angefertigt und alle Stellen, die ich wörtlich oder annähernd wörtlich aus Veröffentlichungen übernommen habe, als solche kenntlich gemacht habe, mich auch keiner anderen als der angegebenen Literatur oder sonstiger Hilfsmittel bedient habe.

Ich versichere an Eides Statt, dass ich die vorgenannten Angaben nach bestem Wissen und Gewissen gemacht habe und dass die Angaben der Wahrheit entsprechen und ich nichts verschwiegen habe.

Die Strafbarkeit einer falschen eidesstattlichen Versicherung ist mir bekannt, namentlich die Strafandrohung gemäß §156 StGB bis zu drei Jahren Freiheitsstrafe oder Geldstrafe bei vorsätzlicher Begehung der Tat bzw. gemäß §163 StGB bis zu einem Jahr Freiheitsstrafe oder Geldstrafe bei fahrlässiger Begehung.

