Studiengang Informatik

Masterarbeit

Verfolgung deformierbarer Objekte mittels
Multidomänenclusterverfeinerung zur Laufzeit

Arne Humann

Bremen, den 28. Januar 2016

Erstgutachter: Prof. Dr. Gabriel Zachmann
Zweitgutachter: Prof. Dr.-Ing. Udo Frese

Humann, Arne
ahumann@informatik.uni-bremen.de
Verfolgung deformierbarer Objekte mittels Multidomänenclusterverfeinerung zur Laufzeit
Masterarbeit, Studiengang Informatik
Universität Bremen, Januar 2016

Selbständigkeitserklärung

Ich erkläre hiermit, die vorliegende Masterarbeit selbständig und ohne die Hilfe Dritter verfasst zu haben. Es wurden von mir keine anderen als die explizit angegebenen Quellen und Hilfsmittel verwendet. Die wissentliche Verwendung von Textausschnitten, Zitaten oder Inhalten anderer Verfasser wurde stets als solche gekennzeichnet.

Bremen, den 28. Januar 2016

Arne Humann

Danksagung

Ich bedanke ich mich bei Prof. Dr. Gabriel Zachmann dafür, dass er die Arbeit als Erstgutachter übernommen und mich mit hilfreichen Ratschlägen unterstützt hat.

Weiterhin bedanke ich mich bei Prof. Dr.-Ing. Udo Frese dafür, dass er sich als Zweitgutachter zur Beurteilung meiner Arbeit zur Verfügung gestellt hat.

Besonders bedanke mich bei Dr. Daniel Mohr, der auch neben seiner Arbeit Zeit für meine persönliche Betreuung gefunden und meine Arbeit durch diverse inhaltliche Diskussionen maßgeblich vorangetrieben hat.

Zudem gilt mein Dank Annika Huflaender, Ingeborg Humann, Malte Humann und Ramón Voß, die große Teile meiner Arbeit gegengelesen haben.

Zusammenfassung

In dieser Arbeit wird ein neues Verfahren zur Erkennung und Verfolgung von Objekten in Folgen von Bildern entworfen und untersucht. Durch ein mehrstufiges Klassifikationsverfahren werden die Bildpunkte in Objekt und Umgebung unterteilt. Dazu werden Messungen mehrerer Sensoren genutzt, die durch das Verfahren zusammengeführt werden, mit dem Ziel Unsicherheiten einzelner Sensoren zu kompensieren. In einem ersten Schritt wird unabhängig für jeden Sensor eine Schätzung über die Klassenzugehörigkeit jedes Bildpunktes erstellt. Diese Einschätzungen aller Sensoren werden in einen gemeinsamen Raum abgebildet, in dem mittels Clustering eine gemeinsame Klassifikation der Eingabedaten erfolgt. Die Cluster werden mit jeder neuen Messung angepasst, sodass ein Objekt über mehrere aufeinander folgende Messungen verfolgt werden kann. Dieses Verfahren wird auf die Auswirkung einzelner Parameter, den Einfluss des verwendeten Distanzmaßes und die Fähigkeit, mehrere Sensoren zu kombinieren hin untersucht.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Die Bilderkennung ist ein Bereich der Informatik, welcher sich damit beschäftigt, Informationen aus Bildern zu extrahieren. Dies lässt sich in vielen Bereichen anwenden. So können beispielsweise handschriftliche Postleitzahlen auf Briefen von einem Computer gelesen werden [Sze11, S. 5], Spielekonsolen können Personen und deren Bewegungen erkennen und zur Steuerung verwenden [SSK+13] und Autos sind in der Lage, Zusammenstöße mit Fußgängern zu vermeiden [KDF+11]. Diese Beispiele zeigen bereits, was für ein Potential im computergestützten Verständnis von Bildern liegt. Dennoch sind Computer noch weit davon entfernt, Bilder so selbstverständlich verstehen und interpretieren zu können wie Menschen [Sze11, S. 3]. Das große Potential und die vergleichsweise noch in den Anfängen steckende Forschung machen diesen Bereich der Informatik zu einem lohnenswerten Forschungsfeld.

Eine wichtige Grundlage für das Verstehen eines Bildes stellt das Erkennen von Objekten dar. Auf Basis erkannter Objekte lassen sich weitere Schlüsse ziehen, die genutzt werden können, um den Inhalt eines Bildes zu verstehen. Für einige Anwendungsfälle ist es zudem notwendig, ein Objekt nicht nur erkennen, sondern über mehrere Bilder hinweg verfolgen zu können, um dessen Verhalten über die Zeit zu erfassen. Dies gilt besonders, wenn auf Bewegungen reagiert werden soll, da eine Bewegung eines Objektes erst über die Zeit messbar ist. Je mehr Eigenschaften eines Objektes ausgenutzt werden können, um es von der Umgebung zu unterscheiden, desto sicherer lässt sich dieses erkennen. Entsprechend bieten sich auf eine Art von Objekt spezialisierte Erkennungsverfahren an, die spezifische Eigenschaften eines Objektes zur Erkennung nutzen können. Eine solche Spezialisierung auf eine Art von Objekt erleichtert zwar dessen Erkennung, schränkt den Einsatz jedoch gleichzeitig auf eben diese Art von Objekt ein. Erkennungsverfahren, die auf eine Art von Objekt spezialisiert sind, lassen sich also nicht ohne Weiteres auf beliebige andere Arten von Objekten anwenden. Es ist jedoch nicht praktikabel für jede Art von Objekt einen eigenen, darauf spezialisierten Erkenner zu verwenden, da die Anzahl

von möglichen Objekten zu groß ist. Entsprechend sind Verfahren zur Erkennung beliebiger Objekte zum generellen Verständnis von Bildern notwendig.

Allerdings ist bereits die spezialisierte Erkennung von Objekten nicht trivial. Durch den Verzicht auf die Nutzung spezifischer Eigenschaften einer Art von Objekt wird die Erkennung zunehmend erschwert. Um dennoch beliebige Objekte erkennen zu können bietet es sich an, möglichst viele allgemein verfügbare Eigenschaften zu verwenden: Jede Eigenschaft stellt eine weitere Dimension dar, in der sich ein Objekt potentiell von der Umgebung unterscheiden lässt. Dies kann durch die Verwendung mehrerer unterschiedlicher Sensoren realisiert werden. Auf diese Weise können neben den Farb- oder Helligkeitswerten eines Objektes auch weitere Eigenschaften für die Erkennung genutzt werden.

Dieser Bereich wurde in den letzten Jahren stark durch die Einführung der Microsoft Kinect vorangetrieben, da diese einen performanten und kostengünstigen RGB-D-Sensor bietet [CS14]. Neben eines Farbbildes lassen sich mit der Kinect somit auch Tiefeninformationen erheben. Als Tiefensensor wird in der Kinect V2 eine Time-of-Flight-Kamera (TOF-Kamera) eingesetzt [CGMS16]. Durch diese wird gemessen, wie lange ausgesendetes Licht benötigt, um von einer Oberfläche zurück zum Sensor reflektiert zu werden. Für diese Messung von eingehendem Licht wird eine Kamera verwendet. Somit entsteht analog zu einem Foto eine in Bildpunkte unterteilte Messung der Distanz vom Sensor zu einer Oberfläche. Die Aufnahmen dieses Sensors können mit denen der RGB-Kamera abgeglichen werden, sodass für einen Bildpunkt anschließend dessen Farbe und Distanz zur Kinect bekannt sind. Weiterhin lässt sich über die Tiefenmessungen und die Eigenschaften der Kamera für die Bildpunkte deren räumliche Position relativ zum Sensor bestimmen. Somit liefert die Kinect eine sechsdimensionale Aufnahme, die aus drei farblichen Dimensionen in Form eines Farbbildes und der dreidimensionalen räumlichen Position jedes Bildpunktes besteht. Dies kann die Erkennung von Objekten unterstützen, da Objekte gleicher Farbe sich in ihrer Position unterscheiden, beziehungsweise können Objekte ähnlicher Position über die Farbe unterschieden werden.

Mit jeder betrachteten Eigenschaft steigt jedoch auch die Größe des Suchraumes, was wiederum den Aufwand erhöht, das Objekt darin zu identifizieren. Dies ist relevant, wenn die Erkennung eines Objektes einer zeitlichen Einschränkung unterliegt. Oft steht nur eine gewisse Zeit zur Verarbeitung von Eingaben bereit, um rechtzeitig auf diese reagieren zu können. Im Falle

des selbständig Zusammenstöße vermeidenden Autos [KDF+11] muss dieses zum Beispiel die Daten auswerten, bevor es mit einer Person kollidiert. Daher bietet es sich an, in solchen Szenarien mit schnell zu erhebenden Objekteigenschaften und schnell zu berechnenden Verfahren zu arbeiten.

Somit ist die Bilderkennung ein Forschungsfeld, in dem es immer noch Grundlagen zu untersuchen gilt. Ebenfalls ist die Kombination mehrerer Eingabedaten zur Verbesserung der Erkennung ein noch nicht ausgeschöpfter Bereich innerhalb der Bilderkennung. Zudem stellt die Möglichkeit einer Laufzeitoptimierung eine weitere Herausforderung für Bilderkennungsverfahren dar. Aufgrund dieser Facetten ist die Objekterkennung innerhalb der Bilderkennung ein spannendes Feld, in dem noch viel ungenutztes Potential steckt.

1.2 Ziel der Arbeit

In Abschnitt 1.1 wurde die Herausforderung der Erkennung und Verfolgung von generischen Objekten in Bildern aufgezeigt. Diese Arbeit setzt sich daher zum Ziel, einen neuen Ansatz für die Erkennung und Verfolgung von allgemeinen, deformierbaren Objekten in Bildern zu entwickeln und untersuchen. Dieser soll Daten mehrerer unterschiedlicher Sensoren zur Erkennung und Verfolgung von Objekten in aufeinander folgenden Messungen nutzen, um so unterschiedliche Eigenschaften des Objektes berücksichtigen zu können. Als Grundlage der Untersuchung sollen dafür die von der Kinect V2 gelieferten RGB-Bilder und die zugehörigen räumlichen Informationen dienen, wobei das zu entwickelnde Verfahren nicht auf diese limitiert sein soll. Entsprechend werden keine zur Erkennung genutzten Optimierungen durchgeführt, die explizit Eigenschaften der eingegebenen Daten ausnutzen, um das Verfahren diesbezüglich allgemein zu halten. Dabei sollen die Informationen der einzelnen eingegebenen Dimensionen zusammengeführt werden, um eine alle Eingangsräume berücksichtigende Klassifizierung der Daten durchzuführen. Für jeden Bildpunkt und den dazu korrespondierenden Messungen in den verwendeten Eingangsräumen wird so entscheiden, ob diese dem Objekt oder der Umgebung angehören, sodass das Verfahren eine Unterteilung aller Messungen in Objekt und Umgebung liefert. Das zu entwickelnde Verfahren soll dabei in Hinblick auf eine mögliche Laufzeitoptimierung entworfen werden, jedoch soll in dieser Arbeit vorrangig die Funktion des Verfahrens erarbeitet und untersucht werden.

1.3 Aufbau der Arbeit

Im folgenden Kapitel 2 werden für das zu entwickelnde Verfahren relevante Ansätze vorgestellt und auf eine mögliche Verwendung in dieser Arbeit hin betrachtet. In Kapitel 3 werden die einzelnen Komponenten und der Aufbau des Verfahrens besprochen. Im Anschluss daran wird das Verhalten des Verfahrens in Kapitel 4 unter verschiedenen Bedingungen getestet und evaluiert. Den Abschluss bildet Kapitel 5, in dem rückblickend bewertet wird, inwieweit das Ziel der Arbeit umgesetzt werden konnte und das einen Ausblick über mögliche Erweiterungen und Änderungen des Verfahrens gibt.

2 Ähnliche Arbeiten

In diesem Kapitel werden Verfahren aus dem Bereich der Bilderkennung mit Bezug zu dieser Arbeit vorgestellt, welche sich mit der Erkennung von Objekten und der Aufteilung eines Bildes in Vorder- und Hintergrund beschäftigen. Abschnitt 2.1 listet diese auf, fasst sie jeweils kurz zusammen und gibt eine Einschätzung über die Bedeutung für diese Arbeit ab. Dem schließt sich in Abschnitt 2.2 eine detailierte Beschreibung eines der für diese Arbeit ausschlaggebenden Verfahren an. Zuletzt werden in Abschnitt 2.3 die vorgestellten Techniken auf ihre Eignung für den Einsatz in dieser Arbeit hin untersucht.

2.1 Stand der Technik

Seit der Einführung eines günstigen Sensors zur Tiefenmessung durch die Microsoft Kinect hat sich die Forschung zunehmend mit dem Bereich der Tiefenmessung beschäftigt, weswegen gerade in den letzten Jahren vermehrt Fortschritte in der Kombination von Tiefenmessungen und Farbbildern gemacht wurden [CS14]. In diesem Abschnitt werden einige vielversprechende Arbeiten zu diesem Thema vorgestellt, deren Themenschwerpunkte dieser Arbeit ausreichend ähneln, um darin verwendete Verfahren für diese Arbeit adaptieren zu können.

Mit ihren Handtracking-Verfahren widmen sich Oikonomidis et al. [OKA11] der Herausforderung, die Pose einer Hand zu bestimmen. Anhand der Hautfarbe ermittelt das Verfahren, welche Bildpunkte zur Hand gehören und transferiert diese Informationen auf die Tiefendaten, um so die zur Hand gehörenden Tiefendaten von denen der Umgebung zu trennen. Die Hautfarbe wird dabei durch ein vorab angelerntes Verfahren ermittelt, welches seine Repräsentation über die Zeit hinweg aktualisiert, um auch Änderungen zu berücksichtigen. Anschließend wird eine Particle Swarm Optimization (PSO) genutzt, um ein Handmodell an die zur Hand gehörenden Tiefendaten anzupassen durch Minimierung der Abweichung des Modells zu den Messdaten. PSO verwendet dabei mehrere Partikel, hier auf Basis des letzten Frames geschätzte Handposen, und verändert diese iterativ in Hinblick auf das bisher optimalste Ergebnis eines Partikels selbst, so wie der bisher optimalsten Pose aller Partikel. In diesem Fall wird nach einer festgelegten

Anzahl an Iterationen gestoppt und der global am besten abschneidende Partikel als Ergebnis verwendet. Da das Modell über 27 Freiheitsgrade verfügt, ist dies ein zeitintensives Verfahren. Daher werden Teile des Algorithmus nicht auf der CPU sondern auf der GPU berechnet. Weiterhin wird die Bestimmung der sechsdimensionalen Position der Hand von der Bestimmung der restlichen Dimensionen getrennt, da die Position vergleichsweise robust und schnell zu bestimmen ist. Auf diese weise wird die Dimensionalität des Suchraumes für die feineren Dimensionen reduziert und damit beschleunigt. Dies geschieht, da die CPU zwar eine Vielzahl unterschiedlicher Befehlen ausführen kann, aber trotz heutigen Prozessoren mit mehreren Kernen nur eine kleine Menge Daten gleichzeitig verarbeiten kann. Dem gegenüber stehen die auf Grafikkarten verwendeten GPUs: Diese haben einen im Vergleich zur CPU geringen Umfang an unterschiedlichen Befehlen, bieten aber aufgrund ihrer Struktur aus sehr vielen Rechenkernen die Möglichkeit, viele Daten gleichzeitig, also parallel, zu verarbeiten. Können Daten gleichzeitig verarbeitet werden, verkürzt dies die Zeit, die es braucht, um alle Daten zu verarbeiten und reduziert somit die Laufzeit. Auch wenn sich dieses Verfahren für die Erkennung von Objekten anpassen ließe, so bliebe nach wie vor die Notwendigkeit, vorweg ein Modell des gesuchten Objektes erstellen zu müssen.

Knoop et al. [KVD06] befassen sich mit einem anderen Bereich der Bilderkennung, dem Erkennen und Verfolgen von Personen. In ihrer Arbeit wird gezeigt, wie sich Daten verschiedenster Sensoren, die im Bereich der Personenerkennung verwendet werden, zu einer gemeinsamen Repräsentation zusammenführen lassen. Der Algorithmus kann so Daten von TOF-Kameras, Daten die durch Stereokameras erhoben wurden, aber auch Informationen, die durch Aufnahmen von Markierungen gemacht wurden, verarbeiten. An diese zusammengeführten Messungen wird anschließend das Personenmodell mittels Iterative Closest Point (ICP) Verfahren angeglichen, um eine detaillierte Erkennung des Körpers zu erhalten. ICP gleicht dabei die Orientierung der Punktmengen des Personenmodells und der Messdaten iterativ an, in dem es schrittweise die Parameter des Personenmodells ändert und so die Distanz reduziert. Ein solches Angleichen wird auch Registrieren genannt. Auch wenn dieses Verfahren interessante Ansätze für diese Arbeit beinhaltet, so bleibt, wie auch schon bei dem vorher genannten, das Problem des vorab zu erstelleden Modells.

Spinello und Arras [SA11] stellen ein Verfahren vor, welches Farb- und Tiefendaten verwendet, um Personen zu erkennen. Das Verfahren erfasst einzelne Bildbereiche des Farbbildes in Form von Gradientenverteilungen, über welche ein Histogram erstellt wird, ein so genanntes Histogram of Gradients (HOG). Auf Tiefendaten angewandt wird dieses Verfahren in der Arbeit

Histogram of Depth (HOD) genannt. Die Histogramme dienen als Repräsentation der Eingabedaten und werden mittels einer vorweg angelernten State Vector Machine (SVM) in Personen und Hintergrund unterteilt, um somit Personen verfolgen zu können. Eine SVM lernt dabei eine Hyperebene, welche die Daten durch einen möglichst großen Grenzbereich trennt [MR05, S. 231ff]. Das Lernen der Hyperebene ist dabei jedoch nicht trivial und kann viel Rechenzeit benötigen. In ihrer Veröffentlichung [SA12] wird weiterhin gezeigt, wie sich auf dieser Basis nicht nur Personen, sondern auch Objekte erkennen und verfolgen lassen. Ähnlich wie die Voraussetzung eines Modells stellt sich auch in diesem Verfahren die SVM als problematisch heraus; um korrekte Klassifizierungen liefern zu können, muss sie vorab auf Trainingsdaten angelernt werden.

Auch Park et al. [PLW11] befassen sich mit der Erkennung von Objekten. Ihr vorgestelltes Verfahren verfolgt ein Objekt, indem es dessen Eigenschaften in Form von Dominant Orientation Templates (DOT) [HLI+10] lernt. DOT repräsentiert einen Bildausschnitt über die Orientierung der am stärksten ausgeprägten darin enthaltenen Gradienten, da diese eine robuste Eigenschaft darstellen. Diese Repräsentation eines Musters kann genutzt werden, um dieses trotz leichter Änderungen in einem Bild zu lokalisieren. Ist die Position eines solchen Musters bekannt, kann darüber auf die Position des gesuchten Objekts geschlossen werden. Zusätzlich zu den Erfassten Mustern des Farbbildes werden die korrespondierenden Tiefendaten erhoben. Diese dienen dem auf die Lokalisierung durch DOT folgenden ICP-Verfahren als Initialisierung. Durch ICP wird das gelernte Modell auf die Tiefenmessung abgebildet, um so die Position und Rotation des Objekts zu erhalten, welche anschließend nochmals unter Verwendung der sich ergebenden Kontur des Objektes verfeinert werden. Zur Initialisierung des Verfahrens dient ein definiertes Muster, welches neben dem zu lernenden Objekt platziert wird und somit Rückschlüsse über die Position des Objekts zulässt, was die initiale Erhebung von Mustern für das DOT-Verfahren ermöglicht. Im Anschluss wird die Kamera um das Objekt herum bewegt. Durch Verfolgung des Objektes kann das Verfahren dabei nach und nach Muster von allen Seiten des Objekts aufnehmen und dessen Eigenschaften rundum lernen. Dies scheint eine gute Methode zu sein, um Objekte zu lernen, jedoch ist die generelle Anwendung aufgrund des zur Initialisierung notwendigen Musters schwierig.

Ein anderer Ansatz, der sich ebenfalls mit dem Verfolgen und Erkennen beschäftigt, sich aber ebenfalls zum Lernen von Objekten eigent, wird von Krainin et al. [KHRF11] vorgestellt. Mittels eines Roboterarms wird ein Objekt vor eine Kamera gehalten und rotiert. Der Algorithmus ist dabei in der Lage sowohl den Arm als darüber auch das Objekt zu erkennen. Ein zur Laufzeit ge-

neriertes Modell des Objektes sowie ein vorab erstelltes Modell des Roboterarms werden zusammen mittels ICP den Messdaten angeglichen, um so eine Schätzung für den aktuellen Zustand beider zu erhalten. ICP verwendet dabei eine umfangreiche Fehlerfunktion zur Bestimmung der Abweichung von Modell und Messung. Neben den auf Surfeln basierenden Modellen des Roboterarms und des Objekts, fließen in die Fehlerfunktion mittels RANSAC erhobene Features, Farbabweichungen zwischen Surfeln und den korrespondierenden Pixeln sowie der Zustand des Armes und Objektes aus dem vorherigen Bild mit ein. Jeder Bildpunkt wird dabei durch einen Surfel repräsentiert, dabei handelt es sich um eine kreisförmige Oberfläche. Diese wird durch eine Position, ihre Normale und ihren Radius beschrieben, wobei der Radius so gewählt wird, dass die Größe des Surfels den Bereich des entsprechenden Pixels im Bild umfasst. Dies ermöglicht eine einfache Repräsentation der Daten und wird genutzt, um Rückschlüsse über die Sichtbarkeit von Bereichen des Objektes zu ziehen. Die so ermittelte Schätzung des gemessenen Zustands wird über die Zeit mittels eines Kalmanfilters verarbeitet, um Messfehler auszugleichen und somit eine verlässliche Positionsbestimmung zu erhalten. Diese Arbeit ist stark an die mit einem Roboterarm kommenden Möglichkeiten angepasst, was den Transfer der verwendeten Konzepte auf eine allgemeine Situation, ohne Roboterarm, erschwert. Dennoch ist der hier angewandte Kalmanfilter ein gutes Mittel, Messfehler über die Zeit zu minimieren. Allerdings benötigt der Kalmanfilter ein Modell welches die Veränderungen des Objekts ausdrückt, was für ein nicht näher spezifiziertes, deformierbares Objekt nur schwer umzusetzen ist.

Die vorherigen Ansätze befassen sich mit dem Erkennen von Objekten zum maschinellen Lernen und setzen dabei Modelle oder Manipulatoren voraus. Jordt und Koch [JK11] befassen sich hingegen mit dem Erkennen und Verfolgen von deformierbaren Objekten ohne vorherige Modelle oder Roboterarme. Ihr Verfahren verwendet ein Poligonnetz, um das Objekt nachzubilden und Non-Uniform rational B-Splines (NURBS) für dessen Verformung. Eine NURBS ist dabei eine Fläche im dreidimensionalen Raum, welche sich verformen lässt. Werden die Punkte des Poligonnetzes dadurch referenziert, lässt sich die Transformation auf der Fläche auf das Poligonnetz des Objektes übertragen. Auf diese Weise kann das Objekt als Poligonnetz unabhängig von seiner Verformumg erfasst werden und die Verformung separat als NURBS dargestellt werden. Durch diese Trennung lässt sich die Berechnung der Verformung einfacher darstellen und aufs Objekt übertragen. Da das Registrieren eines solchen Modells aufwändig ist, verwendet das Verfahren einen Covariance Matrix Adaption Evolution Stragegy (CMA-ES) Algorithmus. Dieser nutzt über eine Gauß-Verteilung zufällig ausgewählte Stichproben, um auf deren Basis eine Covarianz Matrix zu optimieren, so dass diese die Daten letztendlich gut repräsentiert. Da

das Verfahren nur mit den verhältnismäßig wenigen erhobenen Stichproben arbeitet, benötigt es weniger Zeit als andere Optimierungsverfahren, welche alle Datenpunkte berücksichtigen. Die vom Optimierungsverfahren genutzte Fehlerfunktion basiert dabei sowohl auf dem Abstand von Farb- und Tiefendaten, als auch auf einem Wert, der die Verformung der NURBS repräsentiert. Dies ermöglicht es ihnen, auch texturlose deformierbare Objekte mit 6Hz zu tracken. Das Verfahren ist sehr vielversprechend, nicht zuletzt aufgrund seiner Geschwindigkeit. Jedoch ist es durch sein verwendetes Modell stark an die Eingabe von Farb- und Tiefendaten gekoppelt. Da das Modell jedoch definierend für dieses Verfahren ist, scheint sich das Verfahren nur schwer auf andere beziehungsweise allgemeine Eingabedaten übertragen zu lassen.

Ist das Verfahren von Koo et al. [KLK14], mit welchem sich mehrere Objekte gleichzeitig verfolgen lassen, ist ebenfalls vielversprechend. Dieses setzt auf das Registrieren von sechsdimensionalen Punktwolken, bestehend aus Farbe und räumlicher Position, mittels ICP und Gaussian Mixture Model (GMM). In einem GMM werden mehrere Gauß-Verteilungen gewichtet zu einem Modell zusammengefasst. Wird die Distanz zu einem GMM berechnet, so wird intern die Distanz zu jeder Gauß-Verteilung bestimmt und die gewichtete Summe gebildet. Die Anzahl der Gauß-Verteilungen steuert dabei, wie genau das Modell ist. Die vom Verfahren verwendeten Messungen sind bereits vorverarbeitet, sodass nur noch zu den Objekten gehörende Datenpunkte enthalten sind. Entsprechend der letzten Messung werden die aktuellen Datenpunkte vorläufig den in der Szene befindlichen Objekten zugeteilt, um anschließend für jedes Objekt ein neues GMM zu erstellen. Die einzelnen Gauß-Verteilungen der GMM werden mittels Multi-Frame Tracking (MFT) in einen zeitlichen Zusammenhang mit Verteilungen aus früheren Iterationen gesetzt, um so die Geschwindigkeit der Verteilungen festzustellen. MFT vergleicht dabei die Verteilungen mit denen aus früheren Messungen und baut entsprechend ein Modell über den Verlauf der Verteilung auf. Über die Position und die Geschwindigkeit wird anschließend ein Tempo-Spatial Topological Graph (TSTG) aufgebaut, welcher Verteilungen durch Kanten verbindet, die durch Ähnlichkeit in Position und Geschwindigkeit gewichtet werden. Da dabei ein gewisser Schwellwert an Kantengewicht überschritten werden muss, werden nur zum gleichen Objekt gehörende Verteilungen miteinander verbunden und können so zu einem das Objekt repräsentierenden GMM zusammengefasst werden. Die internen Gauß-Verteilungen der so erzeugten GMM der Objekte werden mittels Hierarchical Clustering (HC) zusammengefasst; dabei nutzt HC eine Gauß-Verteilung um mehrere Gauß-Verteilungen abzubilden. Dies reduziert den zum Vergleich dieser GMM benötigten Aufwand. Auf Basis dieser reduzierten GMM wird anschließend nochmals ein MFT durchgeführt, um zu bestimmen, welches GMM welchem Ob-

jekt entspricht und diese somit über die Zeit hinweg verfolgen zu können. Dieser Ansatz ist sehr vielversprechend, setzt aber voraus, dass bereits nur noch zu den Objekten gehörende Datenpunkte vorhanden sind. Somit setzt das Verfahren voraus, was in dieser Arbeit entwickelt werden soll, eine Trennung des Objektes von der Umgebung.

In eine andere Richtung arbeiten Barnich und Van Droogenbroeck [BVD11], die sich mit der Background Subtraction, der Trennung des Bildes in Vorder- und Hintergrund, befassen. Dabei werden sich bewegende Objekte als Vordergrund eingestuft. Auch wenn das Verfahren lediglich auf Farbbilder setzt, ist dieser Entwurf wegen seiner enormen Geschwindigkeit interessant. So wird in der Arbeit sogar der Einsatz des Verfahrens auf einer Digitalkamera gezeigt. Anstelle eines auf Verteilungen basierenden Modells wird der Hintergrund durch erhobene Stichproben repräsentiert. Zur Klassifizierung eines Bildpunktes wird lediglich geprüft, ob eine gewisse Anzahl an erhobenen Stichproben in einer gewissen Nähe zu diesem Punkt liegen. Dies ermöglicht häufig eine Klassifizierung bereits vor Betrachten aller Stichproben. Wird dieser Schwellwert an ähnlichen Stichproben überschritten, so wird der Bildpunkt als Hintergrund eingestuft. Während der Aktualisierung der Stichproben ersetzen neue Proben nicht immer die älteste Probe, stattdessen wird zufällig über eine Gauß-Verteilung entschieden, welche der Messungen ersetzt wird. Um zu vermeiden, dass zum Vordergrund gehörende Stichproben im Hintergrundmodell eingefügt werden, werden zur Aktualisierung nur Pixel, die als zum Hintergrund gehörend klassifiziert wurden verwendet. Um dennoch das Modell von durch den Vordergrund verdeckten Pixeln aktualisieren zu können, wird die Ähnlichkeit von nahe beieinander gelegenen Punkten ausgenutzt. Wird ein Pixel aktualisiert, fügt er seinen Wert auch in die Modelle der umliegenden Pixel mit ein. So werden auch in Grenzgebieten die Pixel, die durch das Objekt verdeckt sind weiterhin aktuell gehalten. Zudem wird ausgenutzt, nicht in jedem Durchlauf alle Mustermengen aktualisieren zu müssen. Stattdessen wird stets eine zufällige Zeitspannte festgelegt, nach der eine Aktualisierung stattfindet, wodurch gleichzeitig der Einfluss periodischer Effekte berücksichtigt wird. Auch wenn sich dieses Verfahren nur mit der Trennung in Vorder- und Hintergrund beschäftigt, so wäre es dennoch denkbar, einige der Ansätze auch für die Objekterkennung zu nutzen und ein Modell auf Basis von Stichproben zu verwenden.

Ein anderer Ansatz zur Trennung in Vorder- und Hintergrund wird von Camplani und Salgado [CS14] vorgestellt. Anders als das vorherige Verfahren setzt der Algorithmus dabei sowohl auf Farb- als auch auf Tiefendaten und nutzt deren unterschiedliche Natur aus, um robuste Ergebnisse zu erzielen. Da ihr Algorithmus grundlegende Ideen zu dieser Arbeit geliefert hat, wird er im Folgenden ausführlich vorgestellt.

2.2 Verfahren von Camplani und Salgado

Camplani und Salgado beschäftigen sich in ihrer Arbeit „Background foreground segmentation with RGB-D Kinect data: An efficient combination of classifiers“ [CS14] mit der automatisierten Trennung eines Bildes in Vorder- und Hintergrund. Dabei setzt das Verfahren auf die unterschiedlichen Eigenschaften der Farb- und Tiefendaten, welche über einen Kinect-Sensor erfasst werden. Pixelweise wird in jeder der beiden Domänen ein eigener schwacher Klassifikator angewandt. Die Ergebnisse der einzelnen Domänen werden erst anschließend zur einem Gesamtergebnis zusammengefasst.

Das Verfahren arbeitet auf RGB-D-Daten, diese bestehen aus einem RGB-Bild und Tiefenmessungen. Auf dieser Basis ist das Verfahren in der Lage, verschiedene Herausforderungen im Bereich der Vorder- und Hintergrundtrennung zu bewältigen, wie Lichtänderungen, oder nicht zu unterscheidende Farb- oder Tiefenwerte von Vorder- und Hintergrund, Farb- beziehungsweise Tiefencamouflage genannt. Das Verfahren arbeitet pixelweise, derartige Ansätze sind meist simpel aufgebaut und benötigen entsprechend wenig Rechenzeit, was ein großer Vorteil gegenüber komplexeren Verfahren ist. Werden die Pixel zudem unabhängig voneinander bearbeitet, kann dies auch parallel geschehen, was sich bei einer entsprechenden Optimierung immens auf die Laufzeit auswirken kann.

2.2.1 Vorder- und Hintergrundmodell

Die Wahrscheinlichkeit eines Wertes x_s, an einem bestimmten Pixel s zur Klasse des Vordergrundes ω_{fg} zu gehören, wird als Gleichverteilung modelliert, da keine weiteren Annahmen über den Vordergrund gemacht werden. In diese Gleichverteilung werden die Anzahl der verwendeten Bits pro Kanal b und die Anzahl der verwendeten Kanäle pro Pixel C einbezogen, um die Anzahl der möglichen Werte zu bestimmen. Daraus berechnet sich die Wahrscheinlichkeitsverteilung des Vordergrundes wie folgt:

$$p(x_s|\omega_{fg}) = \frac{1}{(2^b)^C}$$

Der Hintergrund hingegen wird durch das von Stauffer und Grimson in ihrer Arbeit „Adaptive background mixture models for real-time tracking“ [SG99] vorgestellte Verfahren realisiert. Das Modell arbeitet pixelweise und repräsentiert den Hintergrund an jedem Pixel durch eine Mixture of Gaussians (MoG). Eine solche MoG ist dabei eine Menge von K Normalverteilungen $\mathcal{N}(\mu,\sigma)$, bestehend aus einem Erwartungswert μ und einer Standardabweichung σ. Jede dieser

Verteilungen fließt mit einem Gewicht v in das Modell ein. Wie viele Verteilungen gewählt werden, hängt von der gewünschten Genauigkeit ab, mit der die MoG den Hintergrund modelliert. Die Standardabweichung, der Erwartungswert und das Gewicht der einzelnen Verteilungen werden nach und nach gelernt. Um die Wahrscheinlichkeit der Zugehörigkeit zum Hintergrund ω_{hg} des Wertes x_s zu ermitteln, wird die Zugehörigkeit zu jeder der Verteilungen im MoG bestimmt und diese entsprechend der Gewichte der Verteilungen summiert:

$$p(x_s|\omega_{bg}) = \sum_{i=1}^{K} v_i \mathcal{N}(x_s, \mu_i, \sigma_i)$$

Nach der Klassifikation eines Pixels der aktuellen Messung $x_{s,t+1}$ wird das Hintergrundmodell aktualisiert $\mathcal{N}_{i,t}(\mu_{i,t}, \sigma_{i,t})$, welches noch auf der letzten Messung basiert. Hierzu werden die Verteilungen des MoG nach dem Verhältnis von ihrem Gewicht zu ihrer Standardabweichung $v_{i,t}/\sigma_{i,t}$ sortiert. Der Faktor dient dabei als ein Qualitätsmaß, da ein höheres Gewicht für eine in der Vergangenheit häufiger aktualisierte und somit auch häufig zutreffende Verteilung und eine geringe Standardabweichung für eine gute Genauigkeit spricht. Um zu überprüfen, welche der Verteilungen durch $x_{s,t+1}$ aktualisiert wird, wird entsprechend ihrer Sortierung für die Verteilungen geprüft ob $x_{s,t+1}$ ihnen näher als das λ-fache ihrer Standardabweichtung liegt. Die erste Verteilung, welche die Bedingung $(x_{s,t+1} - \mu_{i,t}) < \lambda\sigma_{i,t}$ erfüllt, wird entsprechend der Messdaten aktualisiert. Über die Lernrate α wird festgelegt, wie stark diese Änderungen ausfallen:

$$\begin{aligned}
v_{i,t+1} &= v_{i,t}(1-\alpha) + \alpha \\
p &= \alpha \mathcal{N}_{i,t}(x_{s,t+1}, \mu_{i,t}, \sigma_{i,t}) \\
\mu_{i,t+1} &= \mu_{i,t}(1-p) + p x_{s,t+1} \\
\sigma^2_{i,t+1} &= \sigma^2_{i,t}(1-p) + p(x_{i,t+1} - \mu_{i,t+1})^2
\end{aligned}$$

Während die Messdaten nur in die erste zutreffende Verteilung einfließen, wird das Gewicht aller anderen um den Faktor α reduziert:

$$v_{i,t+1} = v_{i,t}(1-\alpha)$$

Durch dieses Vorgehen kann eine zu geringe Standardabweichung einen negativen Einfluss auf das Verfahren haben, da kleine Änderungen kaum noch einfließen. Um die Anpassungsfähigkeit des Algorithmus zu gewährleisten, führen Camplani und Salgado eine minimale Abweichung σ_{min} ein, welche als Untergrenze der Standardabweichung verwendet wird.

2.2.2 Funktionsweise der schwachen Klassifikatoren

Die Vorder- und Hintergrundmodelle werden für jede Domäne angelegt. Der Farbraumklassifizierer CL_C verwendet jeweils ein Vordergrund- und Hintergrundmodell mit drei Farbkanälen, während der Distanzraumklassifizierer CL_D jeweils nur einen Kanal in seinen Modellen verwendet. Auf Basis der a posterior Wahrscheinlichkeit einer Klasse $P(\omega_i|x_s)$ bestimmen die Klassifikatoren einen Zugehörigkeitswert, welcher später verwendet wird, um zu bestimmen welche Klasse den höchsten Gesamtwert erreicht und als Klasse für den gegebenen Pixel angenommen wird. Die a posterior Wahrscheinlichkeit ergibt sich aus drei Faktoren $P(\omega_i|x_s) = P(\omega_i)p(x_s|\omega_i)/p(x_s)$, wobei $P(\omega_i)$ die generelle Wahrscheinlichkeit für das Auftreten einer Klasse als gleich wahrscheinlich angenommen wird und daher für die Berechnung des Zugehörigkeitswertes ignoriert werden kann. Zudem lässt sich die Wahrscheinlichkeit des Auftretens von x_s wie folgt ausdrücken: $p(x_s) = \sum_{i=1}^{c} p(x|\omega_i)P(\omega_i)$, wobei c die Anzahl der verschiedenen Klassen entspricht. Da $P(\omega_i)$ jedoch als gleich für alle ω_i angenommen wird, ist $p(x_s)$ lediglich ein gleichbleibender Faktor, der für die Bestimmung des Zugehörigkeitswertes ebenfalls entfällt. Somit bleibt für die Bestimmung des Zugehörigkeitswertes lediglich $p(x_s|\omega_i)$, welche sich, wie oben gezeigt, aus den Vorder- und Hintergrundmodellen ergibt.

2.2.3 Zusammenführung

Da jeder Klassifikator für jede Klasse einen Zugehörigkeitswert ermittelt, ergibt sich eine $c \times l$-Matrix (Decision Profie, DP), wobei c der Anzahl der Klassen entspricht und l der Anzahl der Klassifikatoren.

$$DP(x_s) = \begin{pmatrix} d_{1,1} & \cdots & d_{1,j} & \cdots & d_{1,c} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ d_{i,1} & \cdots & d_{i,j} & \cdots & d_{i,c} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ d_{l,1} & \cdots & d_{l,j} & \cdots & d_{l,c} \end{pmatrix}$$

Ein Zeilenvektor dieser Matrix entspricht allen Werten eines Klassifikators, während ein Spaltenvektor $M_j(x_s)$ alle Zugehörigkeitswerte einer Klasse repräsentiert. Durch eine gewichteten Summe der einzelnen Klassifikatorergebnisse $(d_{1,j},...,d_{l,j})$ in $M_j(x_s)$ wird eine Gesamtzugehörigkeit $S_j(x_s)$ zu einer Klasse ermittelt:

$$S_j(x_s) = \sum_{i=1}^{l} W_i d_{i,j}$$

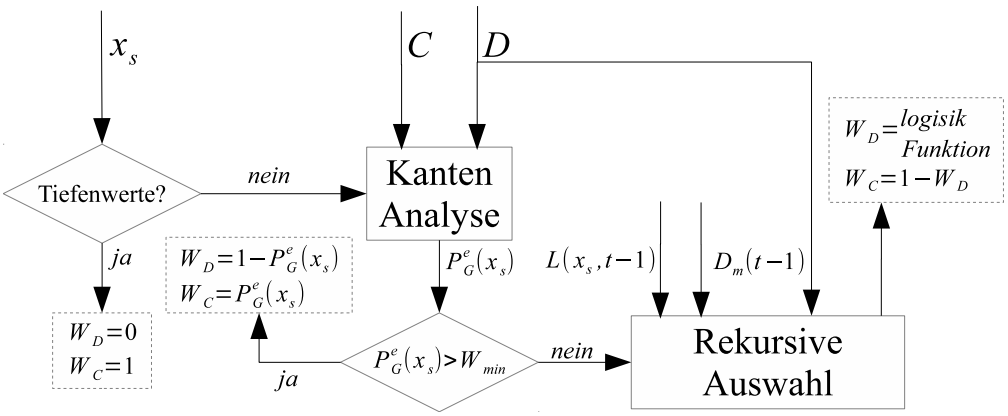


Abbildung 2.1: Diese Abbildung ist eine Übersetzung der Abbildung 2 aus [CS14] und zeigt, wie die Gewichte für den farbraumbasierten und den tiefenraumbasierten Klassifikator (W_C, W_D) bestimmt werden. Der an Pixel s gemessene Wert x_s wird auf das Vorhandensein seiner Tiefenkomponente geprüft. Ist diese gegeben, wird mittels in Farbbild C und Tiefenbild D gefundenen Kanten die Nähe des Pixels zu einer solchen bestimmt $P^e_G(x_s)$. Genügt diese Nähe nicht einem Schwellwert W_{min} so werden, basierend auf den Label des Pixels im vorherigen Durchlauf $L(x_s, t-1)$ und dem zugehörigen Hintergrundmodell $D_m(t-1)$, die Gewichte bestimmt.

Die Klasse mit der größten Gesamtzugehörigkeit wird als Label für den Pixel verwendet. In Abbildung 2.1 wird gezeigt, wie die Gewichte W_i ermittelt werden. Deser Prozess lässt sich in drei Stufen unterteilen:

1. Bedingt durch die Art des Sensors kann es zu fehlenden Tiefendaten an einigen Pixeln kommen. In solchen Fällen werden die Daten des tiefenbasierten Klassifikators ignoriert. Entsprechend werden die Gewichte des farbbasierten Klassifikators $W_C(x_s) = 1$ und des tiefenbasireten Klassifikators $W_D(x_s) = 0$ gesetzt.

2. Sind in beiden Domänen Daten verfügbar, so wird das Gewicht der Klassifikatoren anhand der Nähe des Pixels zu einer Kante im Farb- und Tiefenraum $P^e_G(x_s)$ bestimmt, die so genannte Edge Closeness Probability. Übersteigt diese Kantennähe einen Schwellwert W_{min}, werden die Gewichte $W_C(x_s) = P^e_G(x_s)$ und $W_D(x_s) = 1 - W_C(x_s)$ gewählt.

3. Unterschreitet die Kantennähe den Schwellwert, so werden die Informationen aus dem letzten Bild verwendet, um die Gewichte zu bestimmen.

Um die Kantennähe $P^e_G(x_s)$ zu bestimmen, wird auf alle Eingabebilder ein Kantenerkenner angewandt, welcher zu jedem Bild eine binäre Maske B_i liefert, die angibt, ob ein Pixel zu

einer Kante gehört oder nicht. Anschließend wird die Kantennähe $P_i^e(x_s)$ bestimmt, indem das Maximum der durch eine Gauß-Verteilung G gewichteten Nachbarn H des Pixels im Binärbild B_i gesucht wird: $P_i^e(x_s) = max\{B_i(h \in H) * G\}$.

Auf diese Weise wird die Kantennähe im Farbraum $P_C^e(x_s)$, sowie im Tiefenraum $P_D^e(x_S)$ eines Pixels ermittelt. Die globale Kantennähe ergibt sich durch Multiplikation der beiden $P_G^e(x_s) = P_C^e(x_s)P_D^e(x_s)$. Hinter dieser Art, die Gewichte zu bestimmen steht die Annahme, dass Kanten im Farbraum weniger Rauschen unterworfen sind und der farbraumbasierte Klassifikator entsprechend in der Nähe von Kanten bevorzugt werden sollte. Sind also sich gegenseitig bestätigende Kanten im Farb- und im Tiefenraum vorhanden, so fällt W_C ein höheres Gewicht zu. Fällt die Kante in einem der beiden Bilder schwächer aus, steigt das Gwicht W_D und der Tiefenraumklassifikator gewinnt an Einfluss. Ist in einem der beiden Bilder jedoch keine Kante zu erkennen, wie es bei Farb- oder Tiefencamouflage der Fall wäre, so wird der Schwellwert W_{min} nicht überschritten und die Gewichte werden auf andere Weise gewählt.

In diesem dritten Fall werden die Label des letzten Durchlaufes $L(x_s, t-1)$ in die Wahl der Gewichte einbezogen. Wurde ein Pixel im letzten Durchlauf als Hintergrund L_{bg} gelabelt, so werden trotz der geringen Kantennähe die Gewichte wie oben beschrieben über $P_G^e(x_s)$ bestimmt. Wurde er jedoch den Vordergrund zugeschrieben L_{fg}, werden neue Gewichte berechnet. Diese basieren auf dem Tiefenraum-Abstand des Vordergrundpixels zum Hintergrundmodell des letzten Durchgangs $\delta(x_s, t-1) = |D(x_s, t-1) - \mu_s(t-1)/\sigma_s(t-1)|$, wobei es sich bei $D_(x_s, t-1)$ um die Tiefe des Pixels im letzten Durchlauf handelt und $\mu_s(t-1)$ und $\sigma_s(t-1)$ der Erwartungswert und die Standardabweichung der repräsentativsten Verteilung des Hintergrundmodelles aus dem letzten Durchgang beschreiben. Mit abnehmender Distanz besteht die Möglichkeit von Tiefencamouflage, weshalb dem tiefenbasierten Klassifikator ein entsprechend geringeres Gewicht zugeteilt wird. Die Gewichte bewegen sich dabei zwischen der bereits als Schwellwert verwendeten Untergrenze W_{min} und der Obergrenze W_{max}, was durch eine verallgemeinerte logistische Funktion erreicht wird.

2.2.4 Initialisierung

Aufgrund seiner Natur bedarf das Vordergrundmodell keinerlei Initialisierung. Das Hintergrundmodell wird mittels einer adaptiven Lernrate, welche für die ersten Bilder erhöht ist, initialisiert. Dabei wird zwischen Farb- und Tiefenraum unterschieden. Im Farbraum wird der Messwert als Erwartungswert verwendet und über die benachbarten Pixel die Standardabweichung bestimmt. Für die Initalisierung des Tiefenraumhintergrundmodells wird der Erwartungswert ebenso auf

den Pixelwert gesetzt, die Standardabweichung jedoch nicht über die benachbarten Pixel, sondern über den rechnerischen, sich durch die Distanz zum Sensor ergebenden, Messfehler bestimmt.

2.2.5 Eigenschaften

Camplani und Salgado vergleichen die Ergebnisse ihres Algorithmus mit denen von anderen aktuellen Verfahren. Dabei liegt der Fokus auf den Erkennungsraten unter verschiedenen Herausforderungen. Die verwendeten Testszenarien umfassen Sequenzen mit Schwerpunkten auf Farbcamouflage, Tiefencamouflage, Schatten und bewegte Objekte im Hintergrund, sowie eine Sequenz, die alle diese Fälle beinhaltet. Für jede Sequenz und jedes Verfahren werden dabei folgende Faktoren erfasst: Die Falsch Positiven (FP) Hintergrundpixel, welche fälschlich als Vordergrund klassifiziert werden, die Falsch Negativen (FN) Vordergrundpixel, welche als Hintergrundpixel eingestuft werden, sowie die Gesamtheit der fehlerhaft klassifizierten Pixel in Relation zur Bildgröße. Über die FP und FN wird zudem ein weiterer Faktor bestimmt, welcher diese beiden in ein Verhältnis setzt. Dieser wird sowohl für das gesamte Bild berechnet, als auch nur in kantennahen Bildregionen, um die Hypothese der den Übergang zwischen Objekt und Hintergrund besser modellierenden Farbdaten zu überprüfen. Abschließend wird für jeden Test über die einzelnen erhobenen Faktoren eine Rangfolge der Verfahren erstellt, welche nochmals zu einer gesamt Rangfolge zusammengefasst werden, um zu ermitteln welches Verfahren in welchem Test und welches Verfahren über alle Tests hinweg die zuverlässigste Erkennung ermöglicht. Wie angenommen scheint der farbbasierte Klassifizierer in Kantennähe bessere Ergebnisse zu liefern. Durch die Kombination mit dem tiefenbasierten Klassifizierer ist der Algorithmus zwar nicht in jedem einzelnen Test das am besten abschneidende Verfahren, zeigt aber über die verschiedenen Testfälle hinweg stets gute Ergebnisse, weshalb er in der Gesamtwertung das robusteste Ergebnis liefert.

2.2.6 Zusammenfassung

Mit ihrem Algorithmus lösen Camplani und Salgado einige Probleme. Durch Verwendung von Farb- und Tiefenbild und Ausnutzen derer Eigenschaften erhalten sie robuste Klassifikationen. Durch ihren Ansatz, beide Räume separat voneinander zu verarbeiten und erst die Ergebnisse der schwachen Klassifikatoren zusammen zu führen, können sie die eigentlich nicht direkt vergleichbaren Domänen in einem gemeinsamen Raum vergleichen.

Dabei fällt jedoch die ausschließlich qualitative Natur der Tests auf; Laufzeiten werden nicht genannt. Selbst unter der Annahme, es seien noch Optimierungen möglich, scheint das Verfahren rechenintensiv zu sein. Die Kantenberechnung und die Nutzung und Aktualisierung des verwendeten pixelweisen und MoG-Basierten Modells stellen für sich genommen bereits einen nicht zu unterschätzenden Rechenaufwand dar. So wird im Fazit erwähnt, dass die rechenintensivste Komponente des Algorithmus die Bestimmung der Parameter zur Ermittlung der Hintergrundwahrscheinlichkeit ist. Zudem fällt laut Auswertung die Bestimmung der Gewichte auf Basis von Kanten ebenfalls ins Gewicht.

Ein weiterer wichtiger Faktor, welcher relevant im Hinblick auf diese Arbeit ist, stellt die eigentliche Aufgabe des Algorithmus dar. Er wurde entwickelt, um Vorder- und Hintergrund zu trennen, nicht jedoch um konkrete Objekte in einem Bild zu verfolgen. Auch wenn sich somit nicht der gesamte Ansatz übernehmen lässt, sind dennoch vielversprechende Konzepte im Algorithmus enthalten, welche sich auch im Kontext dieser Arbeit anwenden lassen.

2.3 Bewertung der Techniken

Alle genannten Algorithmen haben eins gemein: Sie alle verwenden ein Modell, um ein Objekt, den Hintergrund, oder Ähnliches abzubilden. Entsprechend gibt es zahlreiche Ansätze, wie diese Modelle im Einzelfall aufgebaut sind und funktionieren. Vielseitig einsetzbar und somit auch viel genutzt sind dabei Modelle, die eine gewisse Unschärfe zulassen und auf Wahrscheinlichkeitsverteilungen basieren. Auch wenn diese Modelle teilweise viel Verwaltungsaufwand bedeuten, liefern sie dennoch eine akzeptable Laufzeit und gute repräsentative Eigenschaften.

Ein ebenfalls viel verwendeter Ansatz ist das Angleichen eines Modells an die Messdaten, um so das Objekt und dessen Abweichungen zum Modell zu ermitteln. Dazu werden diverse Methoden, wie ICP, PSO, oder CMA-ET optimiert und verwendet. Auch wenn diese Verfahren gute Ergebnisse liefern und auf die Anforderungen des einzelnen Algorithmus optimiert sind, sind sie dennoch rechen- und somit auch zeitintensiv. Daher erscheint es ratsam, alternative Ansätze zu verwenden, die möglicherweise bessere Laufzeiteigenschaften aufweisen.

Verfahren, welche nicht auf Registrierung von Modellen setzen, repräsentieren die Daten oft nicht als ein dreidimensionales beziehungsweise sechsdimensionales Modell des Objektes, sondern durch Erfassen verschiedener Eigenschaften, wie HOG/HOD, Kanten, oder auch durch Abweichungen im Farb- und Tiefenraum. Diese werden zumeist unabhängig voneinander auf kleinen Teilen des Bildes erhoben, was die Möglichkeit der Parallelisierung dieser Verfahren of-

fensichtlich macht. Weiterhin sind diese Verfahren zumeist in ihrem Aufwand simpel; erst durch die große Anzahl der zu bestimmenden lokalen Eigenschaften steigt die Laufzeit.

Daher wird in dieser Arbeit ein Ansatz auf Basis simpler, auf Wahrscheinlichkeiten basierender Modelle gewählt, welcher lokale Eigenschaften der Daten verwendet, um daraus Schlüsse über die Zugehörigkeit eines Objektes zu einer Klasse zu ziehen.

3 Aufbau des Verfahrens

Das in dieser Arbeit entworfene Verfahren und die Auswahl der einzelnen Komponenten werden in diesem Kapitel erläutert. Abschnitt 3.1 gibt eine knappe Einführung in die grundlegende Funktionsweise des Verfahrens, damit später erläuterten Komponenten in den Kontext des gesamten Ablaufes eingeordnet werden können. Darauf folgt in Abschnitt 3.2 eine Beschreibung des im Verfahren verwendeten Modells des Objektes und der Umgebung und eine Erläuterung wieso dies gewählt wurde. Abschnitt 3.3 beschäftigt sich damit, wie die Modelle genutzt werden, um die Daten für die Klassifizierung vorzubereiten. Im anschließenden Abschnitt 3.4 wird die Idee des zur endgültigen Klassifizierung genutzten Verfahrens und dessen Funktionsweise erläutert. In Abschnitt 3.5 wird eine Schwachstelle bezüglich der Größenunterschiede von Objekt und Umgebung aufgezeigt und besprochen, wie diese behoben wird. Danach werden in Abschnitt 3.6 einige Details der programmatischen Umsetzung des Verfahrens behandelt. Den Abschluss des Kapitels bildet Abschnitt 3.7, in dem erklärt wird, wie die vorher vorgestellten Verfahren im Ablauf des Algorithmus zusammenspielen, um Objekte zu verfolgen.

3.1 Übersicht über das Verfahren

Die Aufgabe des Verfahrens ist es, ein reales Objekt, wie einen Gegenstand oder eine Person, über eine Reihe von Messungen hinweg zu verfolgen. Dabei sollen Messungen verschiedener Sensoren genutzt werden. Pro Sensor wird dabei von einem Eingangsraum ausgegangen, welcher die Messdaten des Sensors beinhaltet und in Form von Vektoren repräsentiert. Da diese Vektoren die eingehenden Daten beinhalten, werden sie im Folgenden Eingangsvektoren genannt, um sie von anderen Vektoren abzugrenzen. Dabei kann eine Messung aus mehreren Eingangsvektoren bestehen. Wird beispielsweise mit einer Kamera ein Bild aufgenommen, entspricht das gesamte Bild der Messung, jeder Pixel des Bildes entspricht einem Eingangsvektor. Der Eingangsraum umfasst dabei alle Eingangsvektoren des Bildes, seine Dimensionalität entspricht dabei der vom Bild verwendeten Anzahl an Farbkanälen.

Je nach verwendeter Anzahl an Sensoren führt das Verfahren so pro Zeitschritt mehrere Eingangsräume zu einem gemeinsamen Ergebnis zusammen. Dabei ist es wichtig, Eingangsvektoren der verschiedenen Eingangsräume, welche den gleichen Teil der realen Welt abbilden, einander zuordnen zu können. Dies ermöglicht es, solche Gruppe von Eingangsvektoren später als gemeinsame Erfassung der Eigenschaften eines Bereiches der realen Welt zusammenzuführen. Im Falle von mehreren Bildern wird hierzu der Index jedes Pixels beziehungsweise dessen Position im Bild genutzt.

In jedem Zeitschritt ordnet das Verfahren jede Eingangsvektor-Gruppe dem Objekt oder der Umgebung des Objektes zu. Die Umgebung umfasst dabei alle Eingabevektor-Gruppen, die nicht zum Objekt gehören. Wie Abbildung 3.1 zeigt, lässt sich das Verfahren dabei in folgende Abschnitte unterteilen:

1. Separat für jeden Eingangsvektor im Eingangsraum wird durch einfach aufgebaute Klassifikatoren eine Schätzung über die Zugehörigkeit des Eingangsvektors abgegeben. Da diese Klassifikatoren nur eine Vorstufe für die eigentliche Klassifikation erzeugen, werden sie im Folgenden schwache Klassifikatoren genannt. Die durch sie erzeugten Schätzungen einer jeden Eingangsvektor-Gruppe werden zu einem Schätzungsvektor zusammengefasst, welcher im Schätzungsraum liegt.

2. Im Schätzungsraum wird mittels eines Lernverfahrens auf Basis der Schätzungsvektoren eine binäre Klassifizierung der Eingangsvektor-Gruppen ermittelt, welche die Daten in Objekt und Umgebung unterteilt.

3. Auf Basis der Klassifizierung werden die von den schwachen Klassifikatoren verwendeten Modelle jedes Eingangsraumes aktualisiert, um diese als Grundlage für die Klassifizierung der nächsten Messungen zu nutzen.

3.2 Objekt- und Umgebungsmodell

Um entscheiden zu können, wie ähnlich ein Eingangsvektor dem Objekt und der Umgebung ist, wird eine Modell Beider benötigt. Wie in Kapitel 2 erwähnt, gibt es unterschiedliche Ansätze solche Modelle zu realisieren, die meist stark auf das Verfahren abgestimmt werden, um sie effizient einsetzen zu können. Da dieses Verfahren durch seine schwachen Klassifikatoren individuell für jeden Eingangsvektor eine Aussage über dessen Zugehörigkeit treffen soll, wird ein

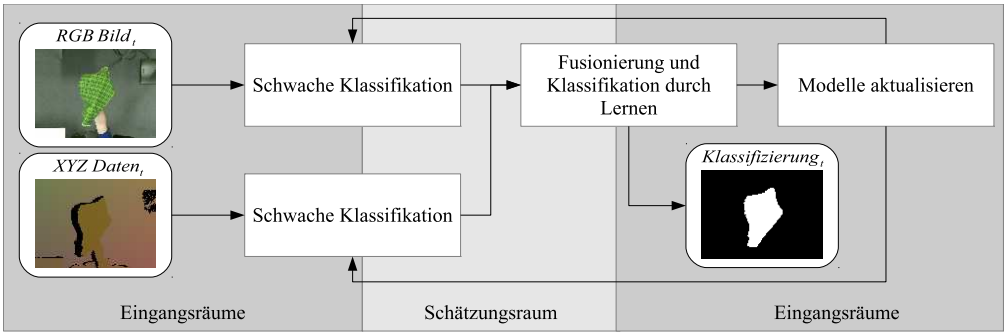


Abbildung 3.1: Dieses Diagramm zeigt die grundlegenden Abschnitte des in dieser Arbeit vorgestellten Verfahrens. Rechtecke mit spitzen Ecken repräsentieren verarbeitende Schritte, Rechtecke mit abgerundeten Ecken stellen Ein- und Ausgaben dar.

Modell benötigt, was dies effizient erlaubt. Die von Camplani und Salgado [CS14] und von Barnich und Van Droogenbroeck [BVD11] vorgestellten Modelle bieten diese Möglichkeit, da sie die Eigenschaften lokal an einem Bildpunkt modellieren und zudem aufgrund ihrer Einfachheit schnelle Laufzeiten erreichen.

Was die Geschwindigkeit betrifft ist das von Barnich und Van Droogenbroeck vorgestellte Hintergrundmodell sehr effizient. Durch Verwendung von Stichproben gemessener Werte zur Repräsentation des Hintergrundes, statt aufwändiger auf Wahrscheinlichkeitsverteilungen basierender Modelle, fällt die Berechnung der Zugehörigkeit und die Aktualisierung einfach und somit schnell aus. Um einen Wert zu klassifizieren wird geprüft, ob dessen Distanz zu einer Stichprobe unter einem Schwellwert liegt. Ist diese Anforderung für eine gewisse Anzahl an Stichproben erfüllt, wird der Wert als zum Hintergrund gehörend eingestuft. Dies erlaubt es im Normalfall vorzeitig zu einem Ergebnis zu gelangen und nicht alle Stichproben prüfen zu müssen. Eine Einschränkung dieses Modells ist allerdings die Eigenschaft, nur eine binäre Klassifizierung zu liefern. Während dies im Falle von [BVD11] der gewünschte Effekt ist, bedeutet es jedoch für das hier entwickelte Verfahren einen Informationsverslust. Denn anders als bei Barnich und Van Droogenbroeck dient das Modell nicht allein der endgültigen Klassifizierung, stattdessen sollen die Unsicherheiten erhalten bleiben und dem nachfolgenden Klassifikator helfen, Tendenzen in den Daten zu erkennen und somit auch unsichere Daten besser klassifizieren zu können.

Daher eignet sich ein Verfahren ähnlich dem von Camplani und Salgado [CS14], welches mehrere Gauß-Verteilungen zur Modellierung des Hintergrundes verwendet. Somit können Unsicherheiten über die Zugehörigkeit eines Wertes zum Modell als Wahrscheinlichkeit ausge-

drückt werden. Durch Verwendung mehrerer Verteilungen mit zugehörigem Gewicht kann ein Grad für die Gesamtzugehörigkeit zu dieser MoG bestimmt werden. Jedoch bedeutet dies für die Berechnung der Distanz zur MoG die Distanz zu jeder darin enthaltenen Verteilung bestimmen zu müssen, was in einem hohen Aufwand resultieren kann. Zudem ist auch die Wahl der Strategie zur Aktualisierung ausschlaggebend, sowohl für die Leistung des Verfahrens, als auch für dessen Laufzeit. Denn neben der Entscheidung, welche Verteilungen wie aktualisiert werden, müssen zudem noch die Gewichte der einzelnen Verteilungen gewählt werden, damit die MoG auch das gewünschte Ergebnis liefert.

Im Falle der von [CS14] gewählten Methode zur Aktualisierung müssen dazu alle Verteilungen in einer MoG sortiert werden, wie in Abschnitt 2.2.1 beschrieben. Auf Basis dieser Sortierung wird die innerhalb der MoG zu aktualisierende Verteilung bestimmt. Mit jeder Messung wird zudem, selbst wenn keine Verteilung als solches aktualisiert wird, immer das Gewicht aller Verteilungen angepasst, um die Alterung der Verteilungen zu realisieren. Für sich genommen ist der zusätzliche Aufwand durch die Aktualisierung überschaubar. Jedoch wird für jeden Bildpunkt eine solche MoG aktualisiert, wodurch diese Mechanik zur Aktualisierung den Gesamtaufwand merklich beeinflusst.

Ein weiterer wichtiger Punkt ist das von Camplani und Salgado verwendete starre Vordergrundmodell. Dabei handelt es sich um eine Gleichverteilung, da keine Annahmen über die Ausprägung des Vordergrundes gemacht werden. Da in dem hier entwickelten Verfahren jedoch explizit Annahmen über die Eigenschaften des Objektes gemacht werden und sich diese dynamisch ändern können, ist eine Gleichverteilung ungeeignet. Somit lässt sich die von Camplani und Salgado gewählte Kombination der Modelle nicht auf diese Arbeit übertragen. Statt des Vordergrundmodells die Methode des Hintergrundmodells zu verwenden bedeutet ebenfalls, den ohnehin hohen Aufwand durch dieses Modell zu verdoppeln und ist damit keine Option. Daher ist auch der von Camplani und Salgado gewählte Ansatz von Vorder- und Hintergrundmodell aufgrund der abweichenden Anforderungen, die das in dieser Arbeit entwickelte Verfahren an die Modelle stellt, nicht geeignet.

Da sowohl für Objekt als auch Umgebung ein dynamisches Modell verwendet werden soll und dieses mit entsprechendem Aufwand verbunden ist, wird ein sehr simples Modell verwendet, welches sich dennoch dynamisch anpassen lässt. In jedem Eingangsraum werden Objekt und Umgebung durch jeweils eine Wahrscheinlichkeitsverteilung repräsentiert, im Folgenden Eingangsraum-Verteilung genannt. Dies erlaubt es den schwachen Klassifikatoren eine Einschätzung über die Zugehörigkeit eines Eingangsvektors zu der jeweiligen Klasse zu erhalten. Um

die Zugehörigkeit zu einer Verteilung zu bestimmen, wird die Mahalanobis-Distanz eingesetzt. Diese ermöglicht es eine Matrix zu verwenden, um die Beziehungen einzelner Dimensionen zueinander in der Distanzberechnung zu berücksichtigen. Auf diese Weise kann das Distanzmaß besser an die Ausprägung der zugrundeliegenden Daten angepasst werden. Die Berechnung der Mahalanobis-Distanz wird in Abschnitt 3.3 genauer erläutert. Auf Basis dieser Zugehörigkeiten lässt sich dann eine Einschätzung über die Klasse des Eingangsvektors erzeugen, die es zudem ermöglicht, auch Unsicherheiten festzustellen.

Sobald die Klassifikation abgeschlossen ist, werden über die einer Klasse zugeordneten Vektoren die Eigenschaften dieser Klasse in jedem Eingangsraum abgebildet. Dabei werden sämtliche Eingangsvektor-Gruppen $P = \{p_1, ..., p_n\}$ betrachtet, wobei n der Anzahl an Eingangsvektoren in jedem Eingangsraum entspricht. Jede dieser Gruppen enthält aus jedem der m Eingangsräume $\mathcal{E} = \{E_1, ..., E_m\}$ einen Eingangsvektor $e_{i,j} \in E_j, 1 \leq i \leq n, 1 \leq j \leq m$, so dass jeder Eingangsvektor genau einer Gruppe zugeordnet ist $p_i = \{e_{i,1}, ..., e_{i,m}\} \in P$. Ziel ist es dabei, eine Eingangsraum-Verteilung $v_{k,j}$ für jede der Klasse $k \in K$ in jedem Eingangsraum E_j zu bestimmen, wobei die Menge aller Klassen $K = \{obj, umg\}$ die Klassen des Objekts und der Umgebung enthält. Da eine Eingangsvektor-Gruppe eindeutig einer Klasse zugeordnet ist, lässt sich die Menge aller Gruppen nach Klassen unterteilen $P = P_1 \cup ... \cup P_{|K|}$, so dass jedes p_i genau einer Teilmenge angehört $P_{k_1} \cap P_{k_2} \neq \emptyset, k_1 \neq k_2$. Da die Klasse jedes Eingangsvektors implizit über die Klasse der Eingangsvektor-Gruppe der er angehört gegeben ist, lässt sich jeder Eingangsraum analog nach Klassen unterteilen $E_j = E_{j,1} \cup ... \cup E_{j,|K|}, E_{j,k_1} \cap E_{j,k_2} \neq \emptyset, k_1 \neq k_2$. Der Erwartungswert $\mu_{k,j}$ einer Eingangsraum-Verteilung $v_{k,j}$ entspricht dem Durchschnitt der einer Klasse angehörenden Eingangsvektoren. Zur Bestimmung des Erwartungswertes $\mu_{k,j}$ einer Eingangsraum-Verteilung $v_{k,j}$ wird die Summe aller Eingangsvektoren des Eingangsraumes gebildet, die der Klasse der Eingangsraum-Verteilung angehören $e_{i,j,k} \in E_{j,k}$ und durch die Anzahl der zu dieser Klasse gehörenden Eingangsvektor-Gruppen $|P_k|$ geteilt:

$$\mu_{k,j} = \frac{\sum\limits_{e_{i,j,k} \in E_{j,k}} e_{i,j,k}}{|P_k|} \tag{3.1}$$

Zur Berechnung der Mahalanobis-Distanz wird zudem eine die Beziehung der Daten beschreibende Matrix benötigt. Um diese zu bestimmen wird der Ansatz von Arnonkijpanich et al. [AHH11] verwendet. Sie erzeugen eine Matrix, welche die Beziehung der Dimensionen der Daten widerspiegelt und dabei symmetrisch und positiv definit ist und deren Determinante gleich 1 ist. Dies erlaubt es ihnen diese Matrix zur Berechnung der Mahalanobis-Distanz einzusetzen.

Grundlage hierfür stellen die sich aus der Distanz zum Erwartungswert der jeweiligen Klasse im Eingangsraum ergebenden Matrizen $(e_{i,j,k} - \mu_{k,j})(e_{i,j,k} - \mu_{k,j})^T$, welche zu einer Matrix $A_{k,j} = \sum_{e_{i,j,k} \in E_{j,k}} (e_{i,j,k} - \mu_{k,j})(e_{i,j,k} - \mu_{k,j})^T$ summiert werden. Um $\Lambda_{k,j}$ mit den genannten Eigenschaften zu erhalten, wird $A_{k,j}$ anschließend invertiert und über die $det(A_{k,j})$ in Bezug zur Anzahl der Dimensionen o_j so skaliert, dass die Determinante $det(\Lambda_{k,j}) = 1$ ist.

$$\Lambda_{k,j} = A_{k,j}^{-1} (det(A_{k,j}))^{\frac{1}{o_j}} \tag{3.2}$$

Auf diese Weise wird pro Eingangsraum und pro Klasse eine Eingangsraum-Verteilung $v_{k,j}$ durch ihren Erwartungswert $\mu_{k,j}$ und eine für die Distanzberechnung verwendbare Matrix $\Lambda_{k,j}$ repräsentiert.

3.3 Schwache Klassifikatoren

Die Verwendung von mehreren Sensoren, welche Daten unterschiedlicher Natur erfassen, bietet folgenden Vorteil: sind die Daten eines Sensors nicht eindeutig, kann diese Unsicherheit durch die Daten eines anderen Sensors aufgefangen werden. Gleichzeitig stellt sich allerdings die Herausforderung diese unterschiedlichen Eingangsräume zu einem gemeinsamen Ergebnis zu vereinen. Ein Problem dabei ist die Findung eines gemeinsamen Distanzmaßes. Dieses wird benötigt, um Ähnlichkeiten von Eingangsdaten zu einem Modell des Objektes oder der Umgebung zu bestimmen. Unterschiedliche Sensoren können auf unterschiedlichen Skalen mit unterschiedlichen Einheiten arbeiten. Da diese Daten nicht immer vergleichbar sind, ist es schwer ein allgemeingültiges Distanzmaß darauf zu definieren, um Ähnlichkeiten korrekt auszudrücken. So ist zum Beispiel nicht eindeutig, wie ein Distanzmaß über Meter und Farbkanäle zu definieren ist. Selbst unter Normierung der Abweichung zum entsprechenden Eingangsraum ist diese Frage nicht eindeutig zu beantworten. Daher werden die Eingangsräume zuerst durch schwache Klassifikatoren unabhängig voneinander bewertet. Da die Eingangsvektoren innerhalb eines Eingangsraumes die gleichen Einheiten verwenden, kann über diese ein Distanzmaß definiert werden. Die schwachen Klassifikatoren nutzen dieses, um für jeden Eingangsvektor in einem Eingangsraum jeweils eine Einschätzung seiner Zugehörigkeit abzugeben. Das Zusammenführen der einzelnen Eingangsräume geschieht erst auf Basis dieser Einschätzungen. Diese liegen alle im gemeinsamen Schätzungsraum, weswegen sich nun ein Distanzmaß über die sich aus den Eingangsräumen ergebenden Schätzungen definieren lässt.

Ein weiterer Vorteil dieses Vorgehens besteht in der Größe des zu verarbeitenden Raumes. Jede Dimension eines Eingangsraums vergrößert den gemeinsamen Gesamtraum exponentiell. Dies erhöht den Aufwand, Relationen zwischen Daten in diesem Raum herzustellen und zu verarbeiten. Werden Eingangsräume, welche jeweils nur einen Teil der Dimensionen des Gesamtraumes aufweisen, einzeln verarbeitet, sinkt der Arbeitsaufwand entsprechend. Somit wird durch die unabhängige Verarbeitung der einzelnen Eingangsräume durch schwache Klassifikatoren nicht nur die Vergleichbarkeit für ein Distanzmaß geschaffen, sondern gleichzeitig auch der Gesamtaufwand reduziert, da jeder Klassifikator den Eingangsraum für die spätere Klassifizierung durch eine Dimension repräsentiert.

Um eine Einschätzung der Zugehörigkeit eines Eingangsvektors $e_{i,j}$ abzugeben, bestimmen die schwachen Klassifikatoren dessen Distanz zum entsprechenden Modell $d(e_{i,j}, v_{k,j})$. Die einzelnen Dimensionen eines Eingangsraumes E_j können dabei unterschiedlich stark ausgeprägt sein. Um dennoch alle Dimensionen gleichermaßen berücksichtigen zu können, wird die Mahalanobis-Distanz als Distanzmaß verwendet. Diese erlaubt es, den Dimensionen unterschiedlich viel Gewicht zukommen zu lassen und das Distanzmaß so an die vorliegenden Daten anzupassen. Die Mahalanobis-Distanz bestimmt sich dabei über den Erwartungswert $\mu_{k,j}$ der Verteilung $v_{k,j}$ und der zugehörigen Matrix $\Lambda_{k,j}$, welche das Verhältnis der einzelnen Dimensionen repräsentiert.

$$d(e_{i,j}, v_{k,j}) = \sqrt{(e_{i,j} - \mu_{k,j})^T \Lambda_{k,j} (e_{i,j} - \mu_{k,j})} \tag{3.3}$$

Auf dieses Weise wird von den schwachen Klassifikatoren für die Eingangsvektoren einer Messung ermittelt, ob diese näher am Objekt oder an der Umgebung liegen. Dies macht die Bestimmung der Distanz zu einer sehr häufig benötigten Funktion, weswegen ihr Aufwand entsprechend gering gehalten werden soll. Da die Distanz nur verwendet wird, um zu überprüfen ob ein Eingangsvektor dem Objekt oder der Umgebung näher liegt, kann an dieser Stelle auf die verhältnismäßig teure Operation des Ziehens der Wurzel verzichtet werden. Deshalb wird im Verfahren die quadrierte Mahalanobis-Distanz $d(e_{i,j}, v_{k,j})$ verwendet. Weil die Distanz nun quadriert ist, fallen kleine Entfernungen nach wie vor klein aus, während große Distanzen verhältnismäßig stark anwachsen. Durch diesen Effekt wird der Fokus auf die nähere Umgebung der jeweiligen Verteilung gerückt, was dort genauere Unterscheidungen erlaubt. Dies ist ein gewünschter Nebeneffekt, da die schwer eindeutig zuzuordnenden Eingangsvektoren in der Nähe des Objektes beziehungsweise der Umgebung liegen und somit schwer von diesen zu trennen sind. Eingangsvektoren, die weit von der jeweiligen Verteilung entfernt liegen, sind hingegen

eindeutig als nicht zur Objekt beziehungsweise zur Umgebung gehörend einzustufen. Auf Basis dieser quadrierten Mahalanobis-Distanz wird die Zugehörigkeit $b_{k,j}(e_{i,j})$ eines Vektors zu den einzelnen Klassen $k \in \{obj, umg\}$ im Eingangsraum E_j bestimmt. Dabei soll $b_{k,j}(e_{i,j})$ für die spätere Vergleichbarkeit in ein festgelegtes Intervall $[0,1]$ fallen, was durch Verwenden einer natürlichen Exponentialfunktion $e^{(-\frac{1}{2}d(e_{i,j},v_{k,j})^2)}$ erreicht wird. Große Distanzen gehen dabei gegen 1, während kleine Distanzen sich 0 nähern. Da die Zugehörigkeit sich jedoch genau entgegengesetzt verhalten soll, so dass eine große Zugehörigkeit aus einer kleinen Distanz und eine geringe Zugehörigkeit einer großen Distanz entspricht, wird die obige Exponentialfunktion von 1 subtrahiert:

$$b_{k,j}(e_{i,j}) = 1 - e^{(-\frac{1}{2}d(e_{i,j},v_{k,j})^2)} \tag{3.4}$$

Auf Basis der Zugehörigkeit berechnet sich die Gesamtzugehörigkeit $b_{ges,j}(e_{i,j}) = b_{obj,j}(e_{i,j}) - b_{umg,j}(e_{i,j})$, welche im Wertebereich $[-1,1]$ liegt. Weist ein Eingangsvektor eine große Zugehörigkeit zum Objekt $b_{obj,j}(e_{i,j}) \to 1$ und eine kleine zur Umgebung $b_{umg,j}(e_{i,j}) \to 0$ auf, so nähert sich $b_{ges,j}(e_{i,j}) \to 1$. Im umgekehrten Falle $b_{obj,j}(e_{i,j}) \to 0, b_{umg,j}(e_{i,j}) \to 1$ nähert sich $b_{ges,j}(e_{i,j}) \to -1$. Weisen beide Zugehörigkeiten jedoch widersprüchliche oder unsichere Informationen auf - durch große oder kleine Zugehörigkeiten zu beiden Klassen - so nähert sich $b_{ges,j} \to 0$, wodurch diese Unsicherheit ausgedrückt wird. Auf Basis dieser Schätzungen über die Zugehörigkeit der Eingangsvektoren führt ein nachfolgender Klassifikator die Daten der unterschiedlichen Sensoren zusammen, um entstandene Unsicherheiten in einzelnen Eingangsräumen aufzulösen.

3.4 Lernen zum finalen Klassifizieren

Im Schätzungsraum S fließen die durch die schwachen Klassifikatoren ermittelten Zugehörigkeiten der einzelnen Eingangsvektoren zusammen. Der alle schwachen Klassifikationen einer Eingangsvektor-Gruppe enthaltende Vektor im Schätzungsraum wird im Folgenden Schätzungsvektor $s_i \in S$ genannt: $s_i = \{b_{ges,1}(e_{i,1}), ..., b_{ges,m}(e_{i,m})\}$.

Um diese Schätzungen der Zugehörigkeit zusammenzuführen gibt es verschiedene Ansätze. So können die Ergebnisse der schwachen Klassifikatoren unter Verwendung einfacher Funktionen, wie etwa einer gewichteten Summe, zu einem gemeinsamen Ergebnis zusammengefasst werden, wie es Camplani und Salgado [CS14] vorschlagen. Das Zusammenfassen selbst ist dabei nur von geringem Aufwand und kann somit schnell durchgeführt werden. Der Aufwand dieser Methode liegt viel mehr darin, die verwendeten Gewichte zu bestimmen. Für einen be-

kannten Fall wie in [CS14] können so die unterschiedlichen Eigenschaften der einzelnen Sensoren berücksichtigt und für eine optimale Zusammenführung ausgenutzt werden. Im Umgang mit allgemeinen Sensoren muss allerdings eine allgemeingültige Zusammenführung der Daten ermöglicht werden, welche entsprechend nicht im Vorfeld auf spezielle Eigenschaften von Sensordaten zugeschnitten werden kann.

Stattdessen muss in solchen Fällen dynamisch entschieden werden, welche Eigenschaften welches Eingangsraumes eine Klasse ausmachen. Hierzu bieten sich Lernverfahren an, die aus gegebenen Daten diese relevanten Eigenschaften erheben, auf deren Basis dann eine Klassifikation stattfinden kann. Für diese Lernverfahren ergeben sich jedoch Einschränkungen aus der Art, wie sie im hier entwickelten Verfahren eingesetzt werden sollen. Da das zu verfolgende Objekt erst mit der Testsequenz bekannt ist, kann nicht im Vorfeld über eine separate Menge von Messungen gelernt werden, wie das Objekt beschaffen ist. Somit wird ein schnelles Lernen des Verfahrens auf Basis der ersten Messungen vorausgesetzt. Bestenfalls kann das Lernverfahren bereits anhand der ersten Eingangsdaten ein Modell erstellen.

Die Entwicklung des Verfahrens in dieser Arbeit in Hinblick auf potentielle Laufzeitoptimierungen bringt weitere Anforderungen an das eingesetzte Lernverfahren mit sich. Das Lernverfahren soll neben der schnellen Initialisierung des Modells ebenfalls stark parallelisierbar sein. Denn gerade im Bereich der Bildverarbeitung liegen meist viele Eingangsvektoren in Form der einzelnen Bildpunkte vor. Mit wachsender Bildgröße wächst die Anzahl der Eingangsvektoren quadratisch, wobei sich der Verarbeitungsaufwand entsprechend erhöht. Kann ein Verfahren die Eingangsvektoren jedoch unabhängig voneinander verarbeiten, lässt sich die Laufzeit stark durch Parallelisierung verbessern. Dabei werden die Eingangsvektoren nicht nacheinander abgearbeitet, sondern es können zur selben Zeit mehrere Eingangsvektoren parallel verarbeitet werden. Sind die einzelnen Arbeitsschritte simpel aufgebaut, muss eine solche Parallelisierung nicht auf die CPU begrenzt sein, sondern kann auf eine Grafikkarte ausgelagert werden. Die Kerne einer Grafikkarte besitzen zwar einen geringeren Funktionsumfang als die einer CPU, jedoch besitzt eine Grafikkarte zahlreiche solcher Kerne, was eine höhere Parallelisierung erlaubt.

Da das Verfahren auch deformierbare Objekte verfolgen können soll, ist es zudem notwendig, das Modell nach jedem Datensatz aktualisieren zu können. Auch dies soll auf eine effiziente Weise passieren und sich gegebenenfalls auch parallelisieren lassen. Die Aktualisierung sollte daher entsprechend einfache Befehle verwenden, die unabhängig voneinander auf den Eingangsvektoren ausgeführt werden können.

Eine Methode, Datensätze in Klassen oder Gruppen einzuteilen, stellen die so genanten Clusteringverfahren dar. Sie werden zu genau diesem Zweck eingesetzt, um Anhäufungen ähnlicher Eingangsvektoren zu finden und diese entsprechend in sogenannte Cluster zu unterteilen, welche später zur Klassifizierung genutzt werden können. Maimon und Rokach [MR05, S. 278ff] unterscheiden zwei wesentliche Arten von Clusteringverfahren: hierarchische und partitionierende Verfahren. Hierarchische Clusteringverfahren teilen die vorliegenden Daten hierarchisch Partitionen zu. Dabei gibt es Ansätze, welche die Daten nach und nach rekursiv unterteilen und solche, die jeden Datenpunkt inital als Partition ansehen und diese rekursiv zusammenfassen. Dabei können einmal erstellte Partitionen jedoch nicht mehr geändert werden. Dem entgegen stehen partitionierende Verfahren. Diese benötigen meist eine vorgegebene Anzahl an Clustern. Sie teilen jeden Datenpunkt einem Cluster zu. Diese Zuteilung wird dabei iterativ optimiert, so dass die Cluster am Ende die Anhäufungen in den Daten abbilden. Gegenüber den hierarchischen Clusteringverfahren bietet dieser Ansatz den Vorteil der weiterhin veränderbaren Cluster, was in diesem Verfahren ausgenutzt werden kann. Die Anzahl der Cluster vorab festzulegen ist dabei unproblematisch, da diese bereits durch die gewünschte Unterteilung in Objekt und Umgebung gegeben ist.

Es wird davon ausgegangen, dass zwischen zwei Messungen nur geringfügige Änderungen in den Daten auftreten. Dies lässt sich im Clusteringverfahren ausnutzen. Denn durch die geringen Änderungen stellt das Ergebnis auf Basis der vorherigen Messungen eine gute Initialisierung für die aktuellen Messungen dar. Daher muss das Clusteringverfahren nicht wieder von vorne beginnen, sondern es genügen wenige Anpassungsschritte, um die Cluster an die neuen Daten anzugleichen. Dies reduziert den Aufwand des Clusteringverfahrens stark, da ein Großteil der zur Optimierung notwendigen Iterationen übersprungen werden kann.

In dieser Arbeit wird daher für die Zusammenführung der Schätzungen das Clusteringverfahren Matrix Neural Gas (MNG) verwendet, wie von Arnonkijpanich et al. in [AHH11] vorgestellt. Dieses ist ein Vertreter der partitionierenden Clusteringverfahren. MNG stellt die Cluster als Verteilungen dar, im Folgenden Prototypen genannt. Auf Basis dieser Verteilungen erfolgt die Klassifizierung. MNG ist eine Erweiterung des Neural Gas Verfahrens und erlaubt die Nutzung der Mahalanobis-Distanz als Distanzmaß. Dies bietet den Vorteil, die unterschiedliche Ausprägung in verschiedenen Dimensionen zu berücksichtigen, wie bereits in Abschnitt 3.2 beschrieben. Die Prototypen von MNG werden, wie die in Abschnitt 3.2 vorgestellten Modelle durch ihren Median $\tilde{\mu}_k$ und eine für die Mahalanobis-Distanzberechung verwendete Matrix $\tilde{\Lambda}_k$ repräsentiert.

MNG durchläuft zur Bestimmung der Prototypen auf Basis der Schätzungsvektoren in jeder Iteration drei wesentliche Schritte:

- Für jeden Schätzungsvektor wird eine Rangliste der Prototypen auf Basis ihrer Distanz zum Schätzungsvektor erstellt. Die Rangliste wird ist dabei nach aufsteigender Distanz sortiert, so dass eine geringe Distanz zu einem niedrigen Rang führt.

- Anschließend werden die Erwartungswerte der Prototypen neu bestimmt. Wie viel Einfluss ein Schätzungsvektor dabei auf einen Prototypen ausübt, wird über den Rang des Prototypen geregelt.

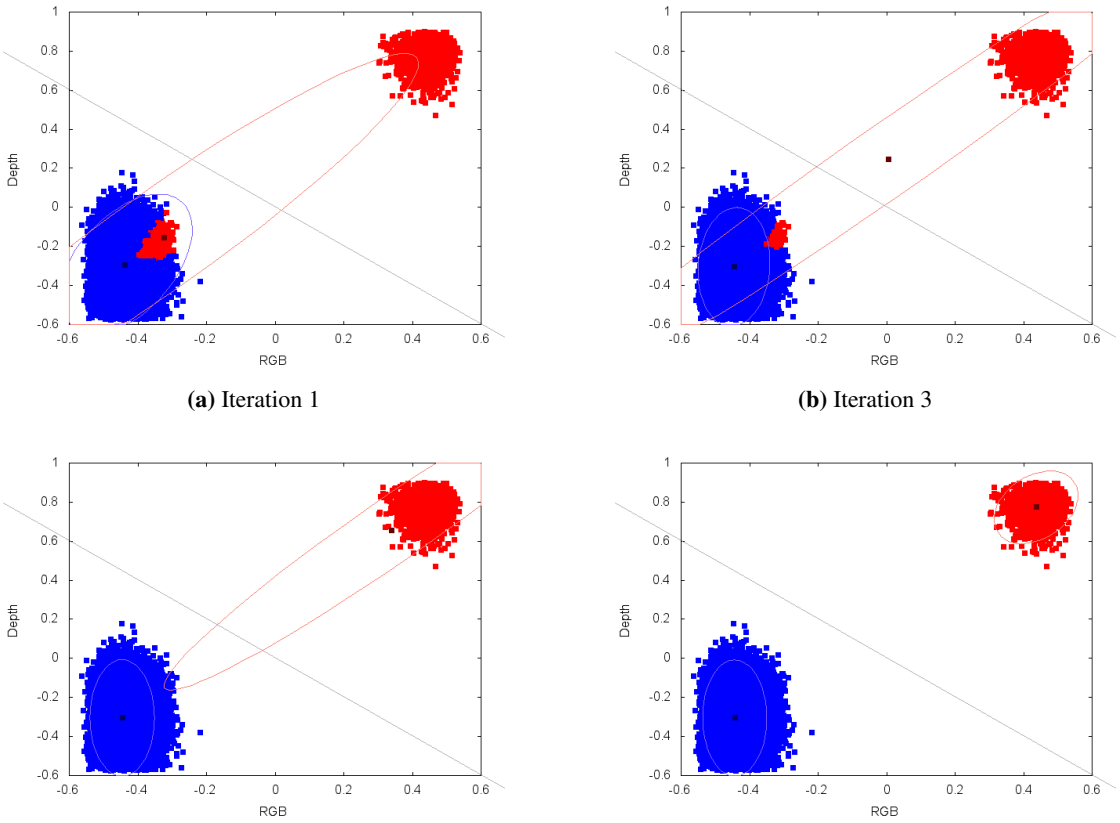
- Im letzten Schritt werden die für die Mahalanobis-Distanz verwendeten Matrizen jedes Prototypen neu bestimmt. Der Einfluss der Schätzungsvektoren auf die Matrizen wird ebenfalls über den Rang der Prototypen geregelt.

Die Bewertung der Prototypen für jeden Schätzungsvektor s_i wird dabei über die quadrierte Mahalanobis-Distanz $d(s_i, \tilde{v}_k)^2$ des Schätzungsvektors zum Prototypen $\tilde{v}_k$ der Klasse k bestimmt. Wurde die Distanz zu jedem Prototypen bestimmt, wird über diese eine Rangliste der Prototypen gebildet. Je näher ein Prototyp dem Schätzungsvektor ist, umso niedriger ist sein Rang $r_{i,k}$. Somit erhält der Prototyp, der dem Schätzungsvektor am nächsten gelegen ist den Rang 0, der am weitesten entfernte den Rang $|K|-1$. Anschließend werden die Prototypen neu berechnet. Das Gewicht, mit dem ein Schätzungsvektor in einen Prototypen einfließt, wird über die Funktion $h(\sigma, r_{i,k})$ bestimmt. Die Funktion $h(\sigma, r_{i,k}) = exp(-\frac{r_{i,k}}{\sigma})$ bildet den Rang auf eine die Nachbarschaft definierende Gauß-Verteilung mit der Standardabweichung σ und Erwartungswert 0 ab, wobei σ mit jeder Iteration kleiner wird. Durch diese Gewichtung werden anfangs auch entferntere Schätzungsvektoren stärker in die Berechnung der Prototypen einbezogen. Mit zunehmenden Iterationen verkleinert sich die durch σ definierte Nachbarschaft und nur noch nähergelegene Schätzungsvektoren haben einen nennenswerten Einfluss auf einen Prototypen. Dies macht das Verfahren robust gegenüber der Initialisierung der Prototypen, da die Initialisierung durch den anfänglich starken Einfluss aller Schätzungsvektoren kaum Einfluss auf das Endergebnis hat. Dieses Verhalten ist in Abbildung 3.2 zu erkennen. Die Erwartungswerte $\tilde{\mu}_k$ der neuen Prototypen bestimmen sich über die gewichtete und normalisierte Summe der einzelnen Schätzungsvektoren:

$$\tilde{\mu}_k = \frac{\sum_{i=1}^{n} h(\sigma, r_{i,k}) s_i}{\sum_{i=1}^{n} h(\sigma, r_{i,k})} \tag{3.5}$$

1: **procedure** MATRIX NEURAL GAS(S_t)
2: $t \leftarrow 0$
3: **while** $t < maxIterations$ **and** $error > errorThreshold$ **do**
4: $\sigma \leftarrow calcSigma(i, maxIterations)$
5: **for all** $k \in \{Object, Surroundings\}$ **do**
6: $\tilde{v}_{k,t-1} \leftarrow \tilde{v}_{k,t}$
7: **end for**
8: **for all** $s_i \in S_t$ **do**
9: **for all** $k \in \{Object, Surroundings\}$ **do**
10: $d_{i,k} \leftarrow d(s_i, \tilde{v}_k)^2$
11: **end for**
12: $\{r_{i,1}, ..., r_{i,k}\} \leftarrow rankByDistance(\{\tilde{v}_{1,t}, ..., \tilde{v}_{k,t}\}, \{d_{i,0}, ..., d_{i,k}\})$
13: **end for**
14: **for all** $k \in \{Object, Surroundings\}$ **do**
15: $\tilde{\mu}_k \leftarrow 0$
16: $w_k \leftarrow 0$
17: **for all** $s_i \in S_t$ **do**
18: $\tilde{\mu}_k \leftarrow \tilde{\mu}_k + h(\sigma, r_{i,k}) s_i$
19: $w_k \leftarrow w_k + h(\sigma, r_{i,k})$
20: **end for**
21: $\tilde{\mu}_k \leftarrow \frac{\tilde{\mu}_k}{w_k}$
22: **end for**
23: **for all** $k \in \{Object, Surroundings\}$ **do**
24: $\tilde{A}_k \leftarrow 0$
25: **for all** $s_i \in S_t$ **do**
26: $\tilde{A}_k \leftarrow \tilde{A}_k + h(\sigma, r_{i,k})(s_i - \tilde{\mu}_k)(s_i - \tilde{\mu}_k)^T$
27: **end for**
28: $\tilde{\Lambda}_k \leftarrow \tilde{A}_k^{-1} det(\tilde{A}_k)^{(\frac{1}{m})}$
29: **end for**
30: $error \leftarrow calcError(\{\tilde{v}_{0,t-1}, ..., \tilde{v}_{k,t-1}\}, \{\tilde{v}_{0,t}, ..., \tilde{v}_{k,t}\})$
31: $t \leftarrow t + 1$
32: **end while**
33: **end procedure**

Algorithmus 3.1: Dieser Pseudocode zeigt den Ablauf von Matrix Neural Gas. Zwischen Zeile 8 und Zeile 13 wird zu jedem Schätzungsvektor s_i eine Rangliste der Prototypen $\tilde{v}_k$ auf Basis deren Distanz zum Schätzungsvektor erstellt. Entsprechend ihres Rang in der Rangliste werden die Prototypen von den Schätzungsvektoren beeinflusst. Dazu werden die Erwartungswerte der Prototypen $\tilde{\mu}_k$ in den Zeilen 14 bis 22 neu bestimmt. Die zugehörigen, für die Mahalanobis-Distanz benötigten Matrizen Λ_k werden in den Zeilen 23 bis 29 ermittelt.



(a) Iteration 1

(b) Iteration 3

(c) Iteration 4

(d) Iteration 5

Abbildung 3.2: Die Grafiken zeigen, wie Matrix Neural Gas die den Schätzungsvektoren zugrundeliegende Struktur lernt. Die Schätzungsvektoren sind dabei rot und blau eingefärbt, um die Zugehörigkeit zum jeweiligen Prototyp zu zeigen. Der rote Prototyp spiegelt das Objekt wieder, repräsentiert durch einen dunkelroten Punkt an der Position seines Erwartungswertes und einer hellroten Fehlerellipse. Der blaue Prototyp spiegelt die Umgebung wieder, sein Erwartungswert ist durch einen dunkelblauen Punkt und seine Fehlerellipse durch eine hellblaue Ellipse gekennzeichnet. Es ist zu erkennen, wie sich die Prototypen über die Iterationen hinweg verhalten: erst zieht es sie zum globalen Schwerpunkt, anschließend gleich sie sich langsam den Anhäufungen an.

Die für die Mahalanobis-Distanzberechnung benötigten Matrizen $\tilde{\Lambda}_k$ werden analog dazu gebildet. Dazu wird für jeden Schätzungsvektor die Differenz zum Erwartungswert in Form eines Differenzvektors berechnet. Dieser wird mit sich selbst transponiert multipliziert. Auf diese Weise entsteht eine symmetrische Matrix, welche die Abweichungen des Schätzungsvektors enthält. Diese Matrizen werden durch $h(\sigma, r_{i,k})$ gewichtet zu einer Matrix $\tilde{A}_k$ aufsummiert, welche dann

verwendet wird, um die Matrix für die Mahalanobis-Distanz zu ermitteln. Dies geschieht nach dem gleichen Verfahren wie bereits in Abschnitt 3.2 beschrieben:

$$\tilde{A}_k = \sum_{i=1}^{n} h(\sigma, r_{i,k})(s_i - \tilde{\mu}_k)(s_i - \tilde{\mu}_k)^T \tag{3.6}$$

$$\tilde{\Lambda}_k = \tilde{A}_k^{-1} det(\tilde{A}_k)^{\frac{1}{m}} \tag{3.7}$$

Auf diese Weise werden die Prototypen nach und nach an die Daten angepasst, bis ein Abbruchkriterium erreicht wird. Dies kann eine gewisse Anzahl an Iterationen, oder eine einen Schwellwert unterschreitende Abweichung zu den Prototypen der vorherigen Iteration sein.

Wie bereits erwähnt, lässt sich die iterative Natur von MNG in diesem Verfahren ausnutzen, um die Laufzeit des Clustering bei folgenden Messungen verringern. Die Steuerung des Gewichts der Schätzungsvektoren über σ dient der Robustheit von MNG gegenüber der Initialisierung. Dies ist eine in der Regel wünschenswerte Eigenschaft, die jedoch in diesem Fall nicht zwingend notwendig ist. Da erwartet wird, dass sich das Objekt sowie die Umgebung innerhalb zweier aufeinanderfolgender Messungen nur leicht verändern, müssen nicht von Grund auf neue Prototypen gebildet werden. Statt dessen bieten die Prototypen der letzten Messung eine gute Initialisierung, welche nur noch geringfügig auf die Änderungen zwischen den Messungen angepasst werden muss. Dies wird durch eine entsprechend kleine Nachbarschaft beziehungsweise ein kleines σ erreicht. Auf diese Weise reichen wenige Iterationen bereits aus, um eine gute Aktualisierung der Prototypen zu erhalten.

Da die grobe Ausprägung des Objektes und der Umgebung im Schätzungsraum bekannt ist, kann diese zur Initialisierung geschätzt werden. Dies bietet das Potential weniger Iterationen zur Initialisierung zu durchlaufen und somit den Aufwand zu reduzieren. Der Erwartungswert des das Objekt widerspiegelnden Prototyps kann in jeder Dimension mit 1 initialisiert werden, da dies jeweils eine hohe Übereinstimmung mit dem Objekt repräsentiert, wie in Abschnitt 3.3 beschrieben. Entsprechend orientieren sich die tatsächlich zum Objekt gehörenden Schätzungsvektoren in Richtung dieses Vektors. Analog dazu kann der Erwartungswert des die Umgebung widerspiegelnden Prototyps in jeder Dimension mit -1 initialisiert werden. Wird allen Dimensionen die gleiche Gewichtung zugewiesen, so werden die Daten initial entlang der durch den Nullpunkt verlaufenden, die Unsicherheit widerspiegelnden Gerade getrennt und bedürfen nur noch geringfügiger Anpassung.

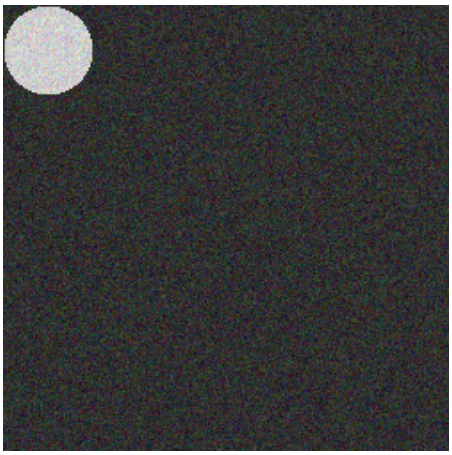
Wie bereits erwähnt, bietet MNG zudem auch Potential zur Parallelisierung. Betrachtet man Algorithms 3.1, welcher das vorab vorgestellte MNG als Pseudocode darstellt, fallen die immer wieder über alle Schätzungsvektoren iterierenden Schleifen auf. Da die Schätzungsvektoren unabhängig voneinander mit den Prototypen in Beziehung gesetzt werden, bevor die Ergebnisse zusammengeführt werden, bietet dies die Möglichkeit zur Parallelisierung. Statt stets über alle Schätzungsvektoren zu iterieren, können diese zeitgleich zueinander parallel verarbeitet werden. Somit bietet sich MNG als ein Lernverfahren an, das durch Parallelisierung und Variieren der Iterationen optimiert und welches auf Basis einer Messung initialisiert werden kann.

3.5 Größenverhältnis von Objekt und Umgebung

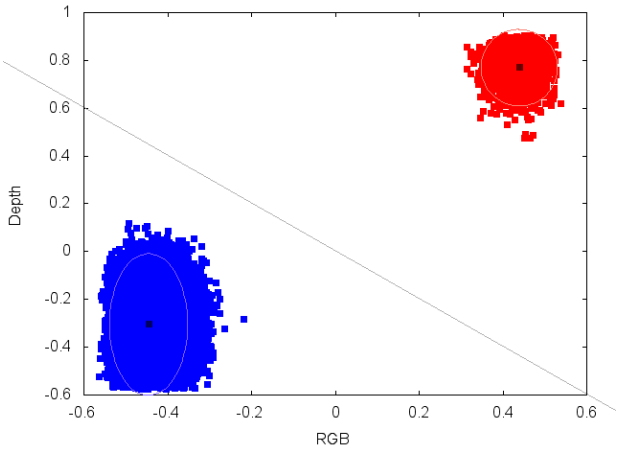
Oftmals macht das zu verfolgende Objekt nur einen Bruchteil der eigentlichen Daten aus. Ist dadurch die Anzahl der zu einer Anhäufung gehörenden Vektoren sehr ungleich verteilt, kann dies das Ergebnis von MNG beeinflussen. Wie bereits in Abschnitt 3.4 erwähnt, verwendet MNG ein Gewicht in Verbindung mit einer Rangfolge, um zu steuern inwieweit ein Schätzungsvektor einen Prototypen beeinflusst. MNG wird dadurch robuster gegenüber seiner Initialisierung.

In den ersten Iterationen werden alle Prototypen stark von allen Schätzungsvektoren beeinflusst. Entsprechend liegt der Erwartungswert nahe dem Durchschnitt aller Schätzungsvektoren. Nach und nach orientieren sich die Prototypen zu den Anhäufungen von Schätzungsvektoren und richten sich lokaler aus. Dieses Verfahren birgt jedoch ein Problem, wenn die vorhandenen Anhäufungen starke Abweichungen in der Anzahl der zugehörigen Schätzungsvektoren aufweisen. Die Prototypen werden dadurch nach wie vor zu Beginn im Zentrum der Schätzungsvektoren zusammengezogen. Die zahlreichen, aber schwach gewichteten Schätzungsvektoren können aufgrund ihrer Anzahl allerdings einen stärkeren Einfluss auf einen Prototypen besitzen, als die wenigen stark gewichteten Schätzungsvektoren der eigentlich zugehörigen Anhäufung. Durch diese Beeinflussung ist es einem Prototypen unter Umständen nicht möglich, sich von der stark gewichteten Masse zu entfernen, um sich einer anderen kleinen Anhäufung anzupassen, Abbildung 3.3 illustriert dies.

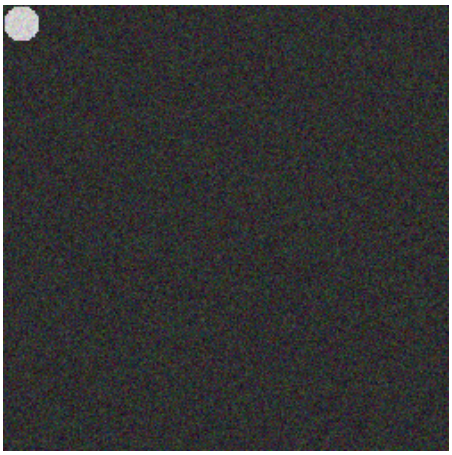
Da das Auftreten eines so starken Ungleichgewichts im vorgesehenen Anwendungsfall realistisch ist, muss das Verfahren entsprechend angepasst werden. Um einen solchen Unterschied feststellen und beheben zu können, muss jedoch bereits bekannt sein, wie viele Schätzungsvektoren einer Ansammlung angehören. Somit muss die Zuordnung der Schätzungsvektoren zum jeweiligen Prototypen bereits bekannt sein. Da dies das eigentliche Ziel des Klassifizierungs-



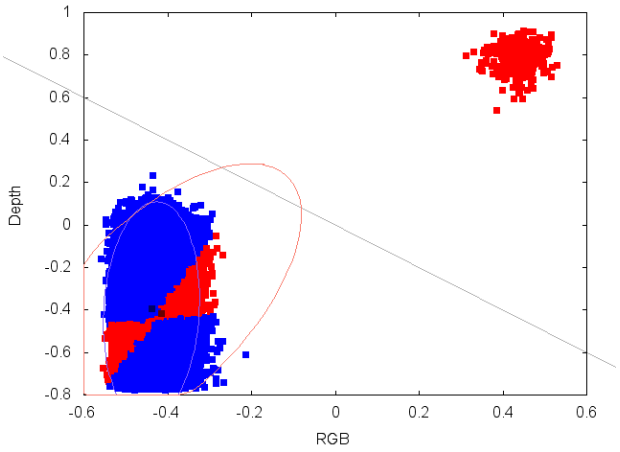
(a) RGB-Bild mit großem Objekt



(b) Korrekte Klassifizierung



(c) RGB-Bild mit kleinem Objekt



(d) Fehlerhafte Klassifizierung

Abbildung 3.3: Die Grafiken zeigen eine korrekte und eine fehlerhafte Klassifizierung durch Matrix Neural Gas. Auf der linken Seite sind die künstlich generierten RGB-Eingangsbilder (a) und (c) abgebildet. Es ist zu erkennen, wie stark der Größenunterschied zwischen dem Objekt (hell) und der Umgebung (dunkel) ist. Nicht abgebildet sind die zu den Farbbildern korrespondierenden räumlichen Daten, welche gleich aufgebaut sind. Auf der rechten Seite ist jeweils der resultierende Schätzungsraum (b) und (d) samt Klassifizierung gezeigt. Die Farbe der Schätzungsvektoren zeigt dabei die Klassifizierung: rot steht für das Objekt, blau für die Umgebung. Die rote und blaue Ellipse sind die um den Erwartungswert eingezeichneten Fehlerellipsen des jeweiligen Prototypen. Es ist zu erkennen, wie sich die unterschiedliche Größe des Objektes im Vergleich zur Umgebung auf die Klassifizierung auswirkt. Während ein verhältnismäßig großes Objekt in (a) zu einer korrekten Klassifizierung der Daten in (b) führt, wirkt sich der in (c) gezeigte starke Unterschied der Anzahl der Schätzungsvektoren von Objekt und Umgebung negativ auf die Prototypen und die Klassifizierung aus, wie in (d) zu sehen ist.

schrittes ist, liegen natürlich zu Beginn der Klassifizierung derartige Informationen noch nicht vor. Deshalb wird für diesen Schritt auf die Ergebnisse der vorherigen Messung zurückgegriffen, um eine Schätzung für die aktuelle Messung zu erhalten. Daraus ergeben sich unterschiedliche Ansätze.

Einer davon ist die Repräsentation der Anhäufung durch gleich viele Schätzungsvektoren. Dafür werden auf Basis der in der letzten Messung erstellten Klassen aus dem aktuellen Datensatz gleich viele Schätzungsvektoren beider Klassen entnommen, um über diese die Klassifizierung zu erstellen. Da sich die Daten zwischen zwei Messungen ändern, werden die so gewählten Vektoren nicht exakt der Verteilung der Klassen entsprechen. Allerdings reicht eine solche Schätzung der Größenverhältnisse aus, um große Unterschiede zu erkennen und zu vermeiden. Dies bringt jedoch weitere zu lösende Probleme mit sich. So bleibt offen, wie viele Schätzungsvektoren gewählt werden, auf welche Weise diese ausgewählt werden und ob Schätzungsvektoren mehrfach gewählt werden können. Vor allem besteht die Möglichkeit, dass relevante Strukturen nicht in der Menge der gewählten Schätzungsvektoren vertreten sind und aufgrund dieser unberücksichtigten Informationen Fehler bei der Klassifizierung entstehen.

Eine Alternative zum Ziehen der Schätzungsvektoren aus den einzelnen Klassen ist es, deren Gewichtung entsprechend ihrer Anzahl anzupassen, mit dem Ziel, beiden Klassen ein gleiches Gewicht zukommen zu lassen. Das Gewicht fällt dabei antiproportional zur Anzahl der Schätzunsvektoren einer Klasse aus. Schätzungsvektoren großer Klassen haben somit jeweils ein geringes Gewicht, während Schätzungsvektoren einer Klasse mit wenig Vektoren ein höheres Gewicht beigemessen wird. Diese Gewichtung wird zusätzlich zum in MNG verwendeten Gewicht zur Steuerung des Einflusses der Nachbarschaft verwendet. Wie auch bei der Ziehung von Vertretern der Klassen wird die Klassifizierung der letzten Messung dafür als Grundlage verwendet. Nach Anzahl der einer Klasse in der letzten Messung zugeordneten Eingangsvektor-Gruppen werden diese, beziehungsweise die sich aus den Gruppen ergebenden Schätzungsvektoren, entsprechend der Klassifizierung der letzten Messung gewichtet. Die durch die Änderung des Objektes in zwei aufeinander folgenden Messungen entstandenen Fehler sind abermals zu vernachlässigen und können durch die überwiegend korrekt gewichteten Schätzungsvektoren und MNG aufgefangen werden.

Ein weiterer Ansatz ist es, das Problem durch Eingrenzung auf die Nachbarschaft des Objektes zu beheben. Der große Unterschied in der Anzahl der Schätzungsvektoren der einzelnen Klassen kommt zustande, da das Objekt klein ist, beziehungsweise nur durch einen geringen Anteil der Eingangsvektoren einer Messung beschrieben wird. Analog dazu ist die Umgebung

durch entsprechend mehr Eingangsvektoren repräsentiert. Wird der betrachtete Bereich auf die Nachbarschaft des Objektes eingegrenzt, so ergibt sich ein ausgeglichenes Verhältnis.

Da die Position eines Objektes erst nach der Klassifizierung bekannt ist, muss eine solche Nachbarschaft, wie die Gewichtung, aufgrund der letzten Messung gewählt werden. Auch hier wird vorausgesetzt, dass ein Objekt seine Position zwischen zwei Messungen nicht schlagartig verändert. Weiterhin führt diese Annahme dazu, dass Eingangsvektoren außerhalb der betrachteten Nachbarschaft, in der sich das Objekt befinden kann, bereits als zur Umgebung gehörend betrachtet werden. Dies erlaubt es, diese Eingangsvektoren bereits ohne umfangreiche Auswertung, nur auf Basis der Nachbarschaft, als zur Umgebung gehörend zu klassifizieren, was wiederum eine starke Reduzierung des Aufwands bedeutet. Durch das ausschließliche Betrachten der Nachbarschaft des Objektes konzentriert sich die Klassifizierung durch MNG zudem stark darauf, die umliegenden Einflüsse um das Objekt herum lokal von diesem abzugrenzen und vermeidet so Störungen durch globale Einflüsse. Dieser Ansatz bietet somit nicht nur die Möglichkeit einer besseren lokalen Klassifizierung, sondern ermöglicht es zudem, nicht in der Nachbarschaft liegende Eingangsvektoren mit konstantem Aufwand klassifizieren zu können.

Um die zu berücksichtigende Nachbarschaft des Objektes zu bestimmen gibt es verschiedene Alternativen. Denkbar wäre ein statisch um das Objekt gezogener rechteckiger oder kreisförmiger Ausschnitt. Dieser ist effizient zu bestimmen, berücksichtigt jedoch nicht die Form des Objektes, wodurch an einigen Stellen mehr Umgebung als notwendig und an anderen Stellen weniger Umgebung als gewünscht berücksichtigt wird. Um dies zu umgehen bietet es sich an, den Bereich des Objektes durch Dilatation zu vergrößern. Dies geschieht auf Basis der Klassifizierung eines Eingangsbildes, in der das Objekt mittels Dilatation vergrößert wird. Auf diese Weise wird die Umgebung um das Objekt herum bei geringem Aufwand gleichmäßig berücksichtigt. Da dieser Ansatz deutliche Vorteile gegenüber der Verwendung von Gewichtung mit sich bringt, arbeitet das Verfahren entsprechend mit einer solchen, um das Objekt herum erstellten Nachbarschaft.

3.6 Umsetzung

Das vorgestellte Verfahren wurde in C++ implementiert, da diese Programmiersprache eine gute Grundlage für spätere Optimierungen bietet und diverse Bibliotheken für diese verfügbar sind. Verwendet wurde zudem die Open Source Bibliothek OpenCV [Ope], welche für die Verwendung in der Bildverarbeitung zugeschnitten und auf Laufzeit optimiert ist. OpenCV wird

dabei vornehmlich seiner Container wegen verwendet, welche dazu dienen Bilder einzulesen und zu verarbeiten und somit auch Matrixberechnungen unterstützen. Diese ermöglichen es zudem, direkt auf den Speicher zuzugreifen und somit die Zugriffe zu optimieren. Da die korrekte Funktionsweise des Verfahrens im Vordergrund der Umsetzung steht, wurden zum Zugriff auf die in den Containern enthaltenden Daten jedoch die dafür vorgesehenen OpenCV-Methoden verwendet.

Die Implementierung wurde in zwei wesentliche Klassen unterteilt. Dabei ist das Verfahren in einer eigenen Klasse namens `MultiDomainTracker` gekapselt. Die einzelnen auszuführenden Schritte lassen sich dabei als Methoden von außen aufrufen und parametrisieren. Dies ermöglicht es, zu Testzwecken Daten an unterschiedlichen Punkten in das Verfahren einzuspielen. Die zweite Klasse dient der Ausführung des Verfahrens und steht stellvertretend für ein Programm, welches das Verfahren benutzt und wurde daher `TestRunner` genannt. Sie dient dazu Ein- und Ausgaben zu steuern, die Eingabedaten zu laden, gegebenenfalls vorzuverarbeiten und in das Verfahren zu speisen.

Als Eingabe dienen die mit einer Kinect V2 aufgenommenen RGB-Bilder und die zugehörigen räumlichen Informationen. Entsprechend ist die Implementierung auf die Verarbeitung von zwei Eingangsräumen zugeschnitten, lässt sich jedoch erweitern. Die Messungen liegen im PNG-Format vor und werden durch `TestRunner` mittels der von OpenCV bereitgestellten Methoden eingelesen. Das PNG-Format wurde aufgrund seiner Komprimierung gewählt; dies wird in Abschnitt 4.1 genauer ausgeführt. Durch Ersetzen der zum Laden verwendeten Methode lassen sich jedoch auch leicht andere Formate einlesen und verwenden.

Die Eingangsdaten werden zur Verarbeitung auf ein Intervall $[0, 1]$ normalisiert, um alle Daten gleich behandeln zu können. Im Falle der räumlichen Informationen können zudem Eingangsvektoren auftreten, für die keine Messdaten vorliegen, beziehungsweise für die die Messung ungültige Daten geliefert hat. Dies ist auf die Art und Weise zurückzuführen, wie die Kinect V2 die Tiefe misst [CGMS16]. Es gibt verschiedene Ansätze, mit diesen fehlenden Daten umzugehen. Für diese Arbeit wurde ein simpler Ansatz gewählt, jedoch ist nicht ausgeschlossen zu einem späteren Zeitpunkt aufwändigere Verfahren einzusetzen. Eingangsvektor-Gruppen mit fehlenden räumlichen Messungen werden in der Verarbeitung ignoriert. Unvollständige Eingangsvektor-Gruppen werden als solche markiert und keiner der beiden Klassen zugeordnet. Dies erlaubt es dem Verfahren dennoch Objekte zu verfolgen und zu klassifizieren.

Um eine Eingabe zu verarbeiten ruft `TestRunner` die Methoden `matrixNeuralGas`, `remaskImage` und `calculateDataSpaceDistributions` von `MultiDomainTracker` der Rei-

henfolge nach auf. Die Methode `matrixNeuralGas` führt die Bestimmung der Prototypen mittels MNG durch. Dabei werden die Schätzungsvektoren einer Eingangsvektor-Gruppe erst zum Zeitpunkt des Zugriffs bestimmt, wodurch dieser Schritt innerhalb der `matrixNeuralGas`-Methode enthalten ist. Somit werden nur die Werte der tatsächlich benötigten Schätzungsvektoren ermittelt. Zudem müssen die Werte auf diese Weise nicht erst bestimmt und später erneut geladen werden, sondern können gleich nach der Berechnung verwendet werden. Wurden die Schätzungsvektoren einmal berechnet, werden sie für weitere Zugriffe zwischengespeichert. Auf die erzeugten Prototypen baut anschließend `remaskImage` auf, welche die Klassifizierung der Eingabe vornimmt und die Nachbarschaft des Objektes unter Verwendung der von OpenCV bereitgestellten Methode zur Dilatation bestimmt. Die Klassifizierung wird von `calculateDataSpaceDistributions` verwendet, um abschließend die Eingangsraum-Modelle zu aktualisieren.

Die zur Initialisierung benötigte Klassifizierung der ersten Eingabe liegt ebenfalls im PNG-Format vor und wird zu Beginn mit den anderen Eingangsdaten geladen. Sie dient der Bestimmung der initialen Eingangsraum-Modelle und der Nachbarschaft. Dazu wird analog zur späteren Verarbeitung die von OpenCV bereitgestellte Methode zur Dilatation sowie die in `MultiDomainTracker` enthaltene Methode `calculateDataSpaceDistributions` verwendet.

Um das akkurate Messen der Ausführungszeit zu ermöglichen wurde zudem die externe Klasse `NanoTimer` [Zac] verwendet.

3.7 Ablauf

In diesem Abschnitt wird beschrieben, wie die vorab vorgestellten Verfahren zusammenspielen, um die Verfolgung eines Objekts über eine Reihe von Messungen zu ermöglichen. Das Erkennungsverfahren ist dabei in vier wesentliche Bereiche unterteilt: Erzeugen der benötigten Schätzungsvektoren, Ermitteln der zur Klassifizierung genutzten Prototypen durch MNG, Klassifizierung auf Basis der Prototypen und Aktualisierung der Eingangsraum-Verteilungen. Diese vier Abschnitte werden bei jeder neuen Messung in der genannten Reihenfolge durchlaufen, Abbildung 3.4 zeigt eine detaillierte Übersicht über den Ablauf des Verfahrens.

Für jede Messung wird dabei auf die Ergebnisse der vorherigen Klassifikation aufgesetzt. Lediglich für die erste Messung muss eine Initialisierung durchgeführt werden. Hierzu benötigt das Verfahren einmalig eine Vorgabe, welche Eingangsvektoren zum Objekt und welche zur Umgebung gehören. Die Wege, eine solche Initialisierung zu erstellen sind zahlreich und umfas-

sen unter anderem einfache Verfahren wie die Initialisierung auf Grund von speziellen Farben, oder durch vorgegebene Bereiche der Tiefe. Aber auch aufwändigere Verfahren zur Objekterkennung, welche einmalig zu Beginn angewendet werden um die Daten anfänglich einzuteilen, sind vorstellbar. Für die Funktion des Verfahrens ist die Art und Weise über welche die initiale Klassifizierung erstellt wird unerheblich. Um Fehler durch Automatisierung der Initialisierung zu vermeiden, wurde in dieser Arbeit daher auf eine manuelle Vorgabe der Klassifikation der Eingangsvektoren gesetzt. Auf Basis der Klassifizierung wird das Objekt ausgeweitet, um die zu betrachtende Nachbarschaft des Objektes zu bestimmen. Aus der vorgegebenen Klassifizierung werden zudem die initialen Eingangsraum-Verteilungen des Objekts und der Umgebung erstellt, wie in Abschnitt 3.2 beschrieben.

Sind die Eingangsraum-Verteilungen bestimmt, kann darüber der Schätzungsraum als Grundlage für MNG gebildet werden. Die verwendete Implementierung von MNG arbeitet wie in Abschnitt 3.4 aufgezeigt und berechnet so die zur aktuellen Messung gehörenden neuen Prototypen, welche zur Klassifizierung genutzt werden können. Allerdings fließen die in Abschnitt 3.5 beschrieben Änderungen mit ein. Somit verwendet MNG nicht alle Schätzungsvektoren, sondern nur solche, die in der Nachbarschaft des Objekts liegen.

Auf die Berechnung der Prototypen folgt die Aktualisierung der Klassifizierungen. Hier wird über alle Eingangsvektoren iteriert und im Schätzungsraum die Distanz zu den einzelnen Prototypen bestimmt. Der am nächsten gelegene Prototyp gibt die Klasse des Eingangsvektors an. Eingangsvektoren außerhalb der Nachbarschaft des Objekts werden ohne vorherige Distanzberechnung als Umgebung klassifiziert. Sind alle Eingangsvektoren klassifiziert, wird, wie in der Initialisierung, die Nachbarschaft des Objektes bestimmt, um diese im nächsten Schritt und zur Verarbeitung der folgenden Messung verwenden zu können.

Ebenfalls analog zur Initialisierung werden daraufhin die Eingangsraum-Verteilungen neu gebildet. Dieser Schritt erfolgt am Ende, denn erst nach der Klassifizierung ist klar, welche Eingangsvektoren in die Neuberechnung der Eingangsraum-Verteilungen einbezogen werden müssen. Während MNG nur die in der Nachbarschaft des Objektes liegenden Daten betrachtet, werden hier alle Bereiche der Daten einbezogen. Da auch außerhalb der Nachbarschaft liegende Eingangsvektoren zur Umgebung gehören, müssen sie mit einbezogen werden, um ein korrektes Modell zu erhalten. Dies ermöglicht es, auch Schätzungen für Werte der nächsten Messung abzugeben, welche nicht in der betrachteten Nachbarschaft enthalten waren.

Nach diesem letzten Schritt ist die Eingabe vollständig bearbeitet und die nächste Messung kann verarbeitet werden. Neben der eigentlichen Klassifizierung der Daten stehen nun die ak-

tualisierten Schätzungsraum-Prototypen zu Verfügung, die Nachbarschaft wurde aktualisiert und die Eingangsraum-Verteilungen wurden erneuert, so dass sie für die nächste Messung genutzt werden können. Die einzelnen vorgestellten Schritte sind dabei modular. Sie können durch andere Verfahren ausgetauscht werden, lediglich die an den nächsten Schritt übergebenen Daten müssen kompatibel sein. So lassen sich die schwachen Klassifikatoren durch andere Verfahren austauschen, MNG lässt sich durch ein anderes Lernverfahren ersetzen und auch das zugrundeliegende Modell der Eingangsraum-Verteilungen kann durch ein anderes ausgetauscht werden.

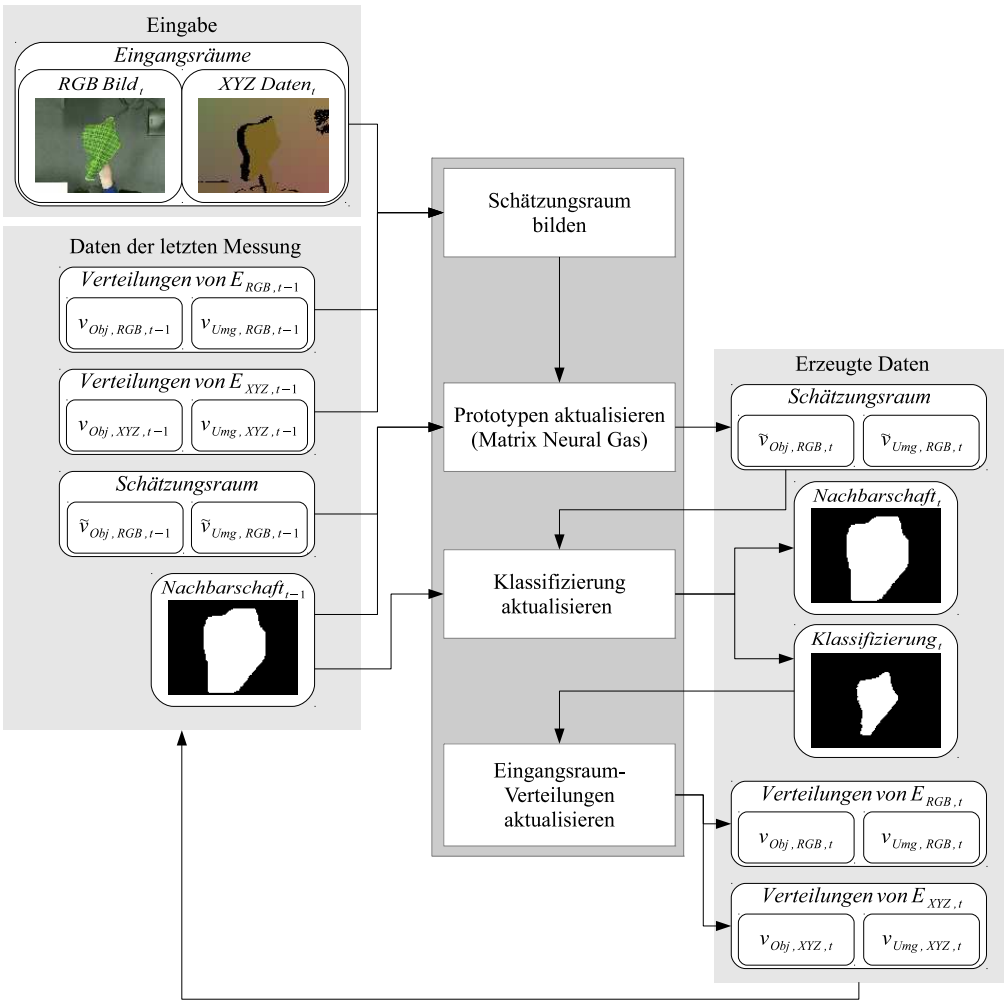


Abbildung 3.4: Die Grafik zeigt den Ablauf des gesamten Verfahrens anhand der von der Kinect gelieferten Messungen. Die rechteckigen Kästen in der Mitte umfassen dabei die einzelnen Arbeitsschritte, welche für jede Messung nacheinander von oben nach unten abgearbeitet werden. Ein- und Ausgaben sind durch abgerundete Rechtecke dargestellt und stellenweise zur besseren Übersicht zusammengefasst. Die Pfeile geben den Fluss der Daten an, und zeigen so, welche Daten in welchen Arbeitsschritt einfließen und aus welchem hervorgehen.

4 Evaluation

In dieses Kapitel wird das Verhalten des Verfahrens unter verschiedenen Einflüssen getestet und analysiert. In Abschnitt 4.1 wird auf die technische Umsetzung der Aufnahme der verwendeten Testdaten eingegangen und welche Vorverarbeitung diese durchlaufen haben. Darauf folgt in Abschnitt 4.2 eine Übersicht über die erhobenen Testsequenzen und wodurch sich diese auszeichnen. Abschnitt 4.3 beschreibt die für alle Tests angewendeten Schreibweisen und Einstufungen. Anschließend wird in Abschnitt 4.4 überprüft, welche Konfiguration des Verfahrens die besten Ergebnisse erbringt. Daraufhin werden in Abschnitt 4.5 die Auswirkungen des gewählten Distanzmaßes untersucht. In Abschnitt 4.6 wird überprüft, wie sich das Verfahren unter verschiedenen Eingaben verhält und ob die Kombination mehrerer Eingaben einen Mehrwert erzeugt. Abschnitt 4.7 beschreibt häufige Fehlerquellen des Verfahrens und wie diese entstehen. Danach wird in Abschnitt 4.8 die Laufzeit des Verfahrens untersucht. Abschließend werden die Testergebnisse in Abschnitt 4.9 rückblickend zusammengefasst.

4.1 Verwendete Daten

Die im Folgenden vorgestellten Testsequenzen wurden mit der Kinect V2 aufgenommen. Diese liefert dabei pro Messung je ein RGB-Bild und die zu jedem Bildpunkt im RGB-Bild gehörenden räumlichen Informationen. Um die zueinander korrespondierenden Bildpunkte der RGB- und der ToF-Kamerabilder zu bestimmen wird das von Microsoft bereitgestellte Kinect for Windows SDK 2.0 [Mic] verwendet. Durch die unterschiedliche Perspektive der beiden Kameras entstehen dennoch Abweichungen. So kommt es gerade in Randgebieten von Objekten zu fehlerhaften Kombinationen aus Farbe und Raum-Koordinate. In solchen Fällen ist selbst manuell nicht entscheidbar, welche Punkte zum Objekt und welche zur Umgebung gehören, da beide Eingangsräume für diese Bildpunkte jeweils widersprüchliche, aber für ihren Raum korrekte Objekte zeigen.

Zudem wird das Framework verwendet, um auf Basis der Tiefenmessungen dreidimensionale räumliche Koordinaten für jeden Bildpunkt zu bestimmen. Die so erhaltenen Daten sind relativ

zur TOF-Kamera. Die Tiefenwerte, im Folgenden als Z-Koordinate betrachtet, steigen mit dem Abstand zur Kamera, wobei der Nullpunkt in der Kamera liegt. Sowohl die horizontale Position, im Folgenden als X-Koordinate betrachtet, als auch die vertikale Position, die im Folgenden als Y-Koordinate betrachtet wird, werden relativ zur Bildmitte angegeben. Die Bildmitte entspricht der Koordinate X=0, Y=0 und nimmt nach rechts unten zu. So erhalten Bildpunkte links der Mitte negative X-Koordinaten, rechts der Mitte liegen positive X-Koordinaten. Die Y-Koordinaten verhalten sich analog, oberhalb der Bildmitte liegen negative Y-Koordinaten, unterhalb positive Y-Koordinaten.

Die aufgenommenen Testsequenzen werden im PNG-Format gespeichert, da dieses für Bilder gedachte Format die Daten verlustfrei komprimiert. Die RGB-Aufnahmen verwenden je Kanal 256 Farbwerte. Das Speichern der unkomprimierten XYZ-Aufnahmen stellt sich als zeitkritisch heraus. Die Kinect V2 ist in der Lage 30 Bilder pro Sekunde [CGMS16] aufzunehmen. Da jede Messung eine beträchtliche Menge an Daten enthält, war es der während der Aufnahmen verwendeten Festplatte nicht möglich, die Daten zwischen zwei Messungen zu speichern. Um die Daten dennoch schnell genug speichern zu können, werden diese ebenfalls komprimiert. Da es sich bei den räumlichen Informationen ebenfalls um Aufnahmen mit drei Dimensionen handelt, wurde analog zu den RGB-Bildern das PNG-Format dazu verwendet. Dieses reduziert die Menge an Daten, die auf die Festplatte geschrieben werden muss, um ein Vielfaches. Vereinzelt benötigt die Festplatte dennoch mehr als die zur Verfügung stehende Zeit. Da dies jedoch nur selten der Fall ist und der zeitliche Abstand der Bilder gering genug ist, dass das Verfahren mit diesen arbeiten kann, stellt dies für diese Arbeit kein Problem dar. Dabei wird jeder der drei XYZ-Kanäle in einen der drei für Farbe gedachten Kanäle geschrieben, so dass X im R-Farbkanal, Y im G-Farbkanal und Z im B-Farbkanal enthalten ist. Dafür werden die XYZ-Kanäle analog zu den RGB-Werten auf einen Bereich von 256 Werten abgebildet. Um die Laufzeit der Testsequenzen zu reduzieren, wurden die RGB- und XYZ-Bilder vor der Verwendung von der durch die Kinect V2 gelieferten Auflösung von 1920x1080 Bildpunkten [CGMS16] um den Faktor fünf auf eine Größe von 384x216 Bildpunkte herunter skaliert.

Neben den aufgenommenen Sequenzen wurden auch künstlich Daten generiert. Deren RGB-Bilder und räumliche Informationen liegen ebenfalls, wie beschrieben, im PNG-Format vor. Sie sind quadratisch aufgebaut und verwenden eine Auflösung von 250x250 Bildpunkten. Die Verteilung der farblichen und räumlichen Werte des Objektes und der Umgebung wird über Gauß-Verteilungen realisiert. Das Objekt ist dabei ein Kreis, welcher sich über das Bild bewegt.

4.2 Sequenzen

Getestet wurden verschiedene Sequenzen, teils künstlich erzeugt, teils real aufgenommen. Da die künstlich erzeugten Testsequenzen simpel aufgebaut sind, dienen sie der Kontrolle des Verfahrens. Ihr einfacher Aufbau ermöglicht es, das Verhalten des Verfahrens klar darzustellen und nachzuvollziehen. Entsprechend spiegeln diese künstlichen Testsequenzen nicht die mögliche Vielfalt von realen Daten wieder. Da das Verfahren jedoch in realen Daten Objekte verfolgen können soll, wurden entsprechend viele reale Sequenzen in die Tests mit aufgenommen. Die Sequenzen sollen die Möglichkeit bieten, Objekt und Umgebung voneinander zu unterscheiden, bilden dabei jedoch verschiedene Herausforderungen für das Verfahren ab, weswegen diese Trennung nicht immer trivial ist. Die Herausforderungen umfassen unter anderem die Verfolgung mehrfarbiger Objekte, wechselnde Lichtverhältnisse, so wie Farb- und Tiefencamouflage.

Eine Auflistung der Sequenzen ist in Tabelle 4.1 zu sehen. Jede Sequenz ist durch vier daraus entnommene Farbbilder repräsentiert, um zu zeigen wie die Testsequenz aufgebaut ist. Bei den ersten fünf Sequenzen handelt es sich um künstlich generierte Daten. Das Objekt, in diesem Fall der zu helle Kreis, bewegt sich dabei stets von links oben nach rechts unten. Die Sequenzen 00 bis 02 verwenden verschieden starke Abgrenzungen vom Objekt zur Umgebung, sowohl farblich aus auch räumlich. Während sich in Sequenz 00 Objekt und Umgebung stark voneinander abheben, sind sie in Sequenz 02 nur noch schwer zu unterscheiden. Sequenzen 03 und 04 verwenden den gleichen Wertebereich für Objekt und Umgebung wie Sequenz 00. Jedoch enthalten diese jeweils in der farblichen beziehungsweise räumlichen Eingabe einen Bereich, der von seinen Werten her dem Objekt gleicht. Dies simuliert jeweils Farb- beziehungsweise Tiefencamouflage. Dabei wird das Verfahren vor eine maximale Unsicherheit gestellt, da die Werte der beiden Eingangsräume stark gegensätzliche Schätzungen liefern. Ob ein solches Phänomen dem Objekt oder der Umgebung zugeschrieben werden soll, ist nicht eindeutig zu klären. Somit zeigen diese Sequenzen, wie das Verfahren mit einem maximalen Widerspruch der Eingangsräume verfährt.

Auf die generierten Sequenzen folgen die realen, mit der Kinect V2 aufgenommenen Sequenzen. Verwendet werden dabei meist deformierbare Objekte sowie markante Farben. Die markanten Farben bieten die Möglichkeit, gezielt zu testende Störfaktoren einzubringen und deren Auswirkung besser nachvollziehen zu können. Die Sequenzen 05 bis 09 zeigen dabei typische Fälle, bei denen eine Person mit einem Objekt interagiert, in dem es dieses durch den Raum bewegt. Durch die physische Verbindung von Person und Objekt, ist das Objekt nur durch die Farbe von der Person zu trennen. Um das Objekt verfolgen zu können, muss das Verfahren diese Unsicherheit in der räumlichen Eingabe entsprechend durch die farbliche Eingabe auflösen.

Daher wird erwartet, dass das Verfahren diese Unsicherheit, wie beschrieben, über die farbliche Messung auflöst und Objekt und Umgebung somit trennen kann.

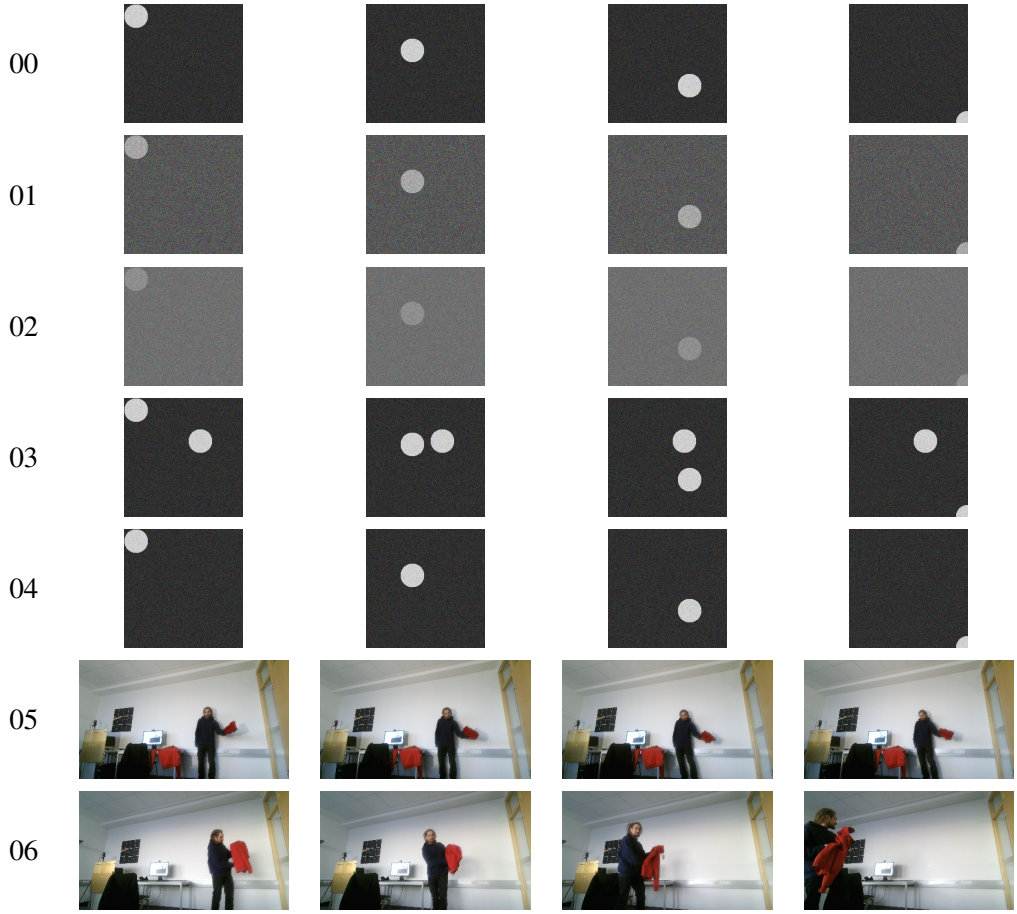
Die Sequenzen 10 bis 16 enthalten Folgen mit Tiefencamouflage, in denen sich das gesamte Objekt in der räumlichen Eingabe nicht mehr von der Umgebung unterscheiden lässt. Wie schon in den Sequenzen zuvor muss das Verfahren diese Unsicherheit über die Farbe auflösen. Anders als im vorherigen Fall betrifft die Überschneidung von Objekt und Umgebung jedoch nicht nur einen kleinen Bereich, sondern die gesamte Kontur des Objektes. Dabei wird vom Verfahren erwartet, das Objekt zuverlässig zu verfolgen, sofern eine Trennung über die Farbe möglich ist. Gleichfalls wird erwartet, dass eine mehrfarbige oder ähnlich farbige Umgebung das Verfahren vor eine Herausforderung stellt. Entsprechend werden Komplikationen währen der Verfolgung in die Sequenzen 10, 14, 15 und 16 erwartet.

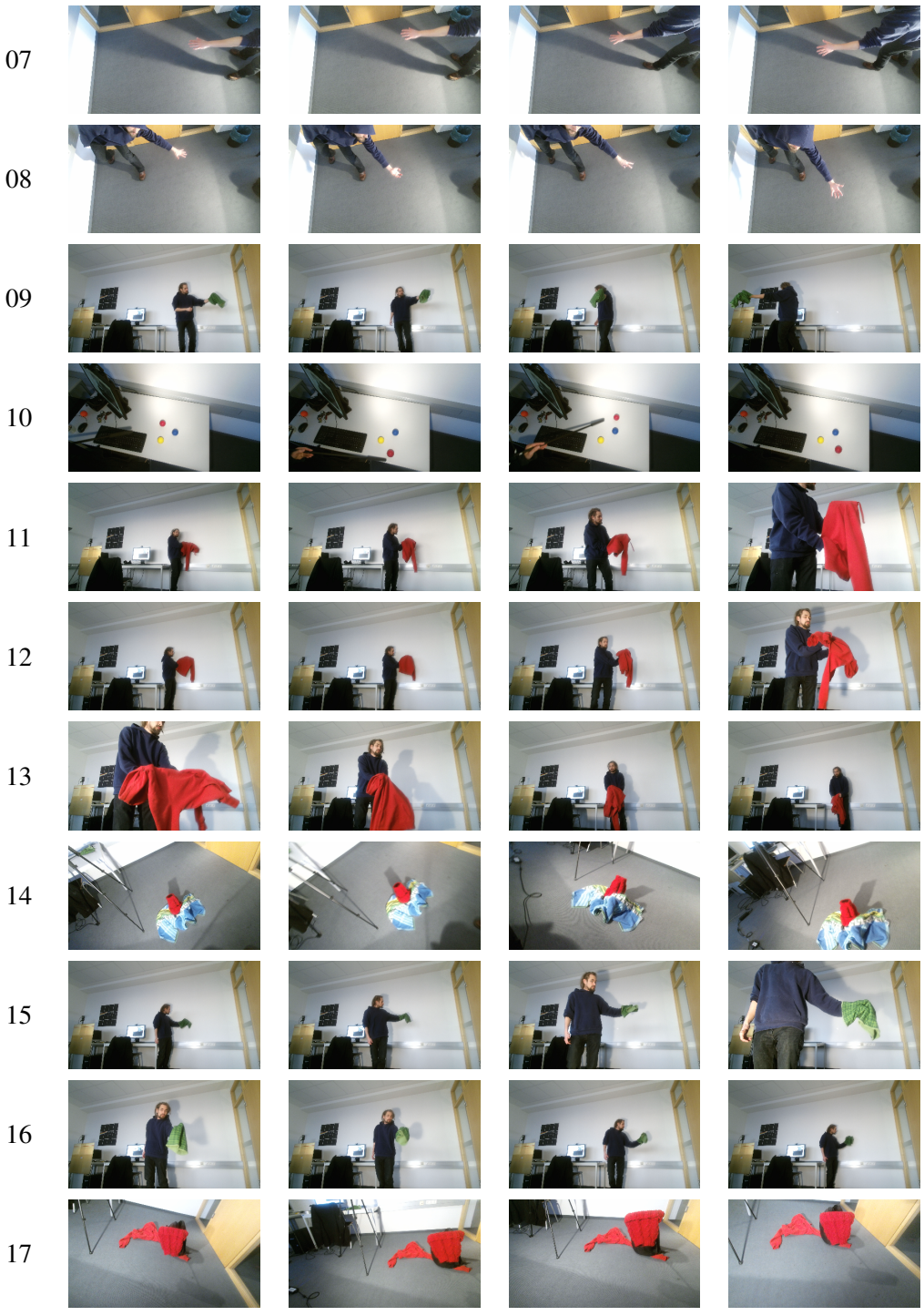
Die Sequenzen 17 bis 24 enthalten Farbcamouflage. Dies sind Bilderfolgen, bei denen sich das Objekt farblich nicht mehr von der Umgebung unterscheidet und ausschließlich auf Basis der räumlichen Informationen getrennt werden muss. Dabei wird erwartet, dass das Verfahren das Objekt in den Sequenzen 17 und 21 zuverlässig aufgrund der räumlichen Unterschiede von der Umgebung unterscheiden kann. In den anderen Fällen reicht die Verwendung räumlicher Informationen allein nicht aus, da sich Objekt und Person räumlich sehr nahe sind. Entsprechend muss hier eine Kombination aus Farbe und räumlichen Informationen verwendet werden, um das Objekt über räumliche Unterschiede vom Großteil der Umgebung zu trennen und über farbliche Unterschiede von der Person. Die Kombination aus Farbe und räumlicher Position sollte es dem Verfahren jedoch ermöglichen, auch diese Unsicherheit aufzulösen.

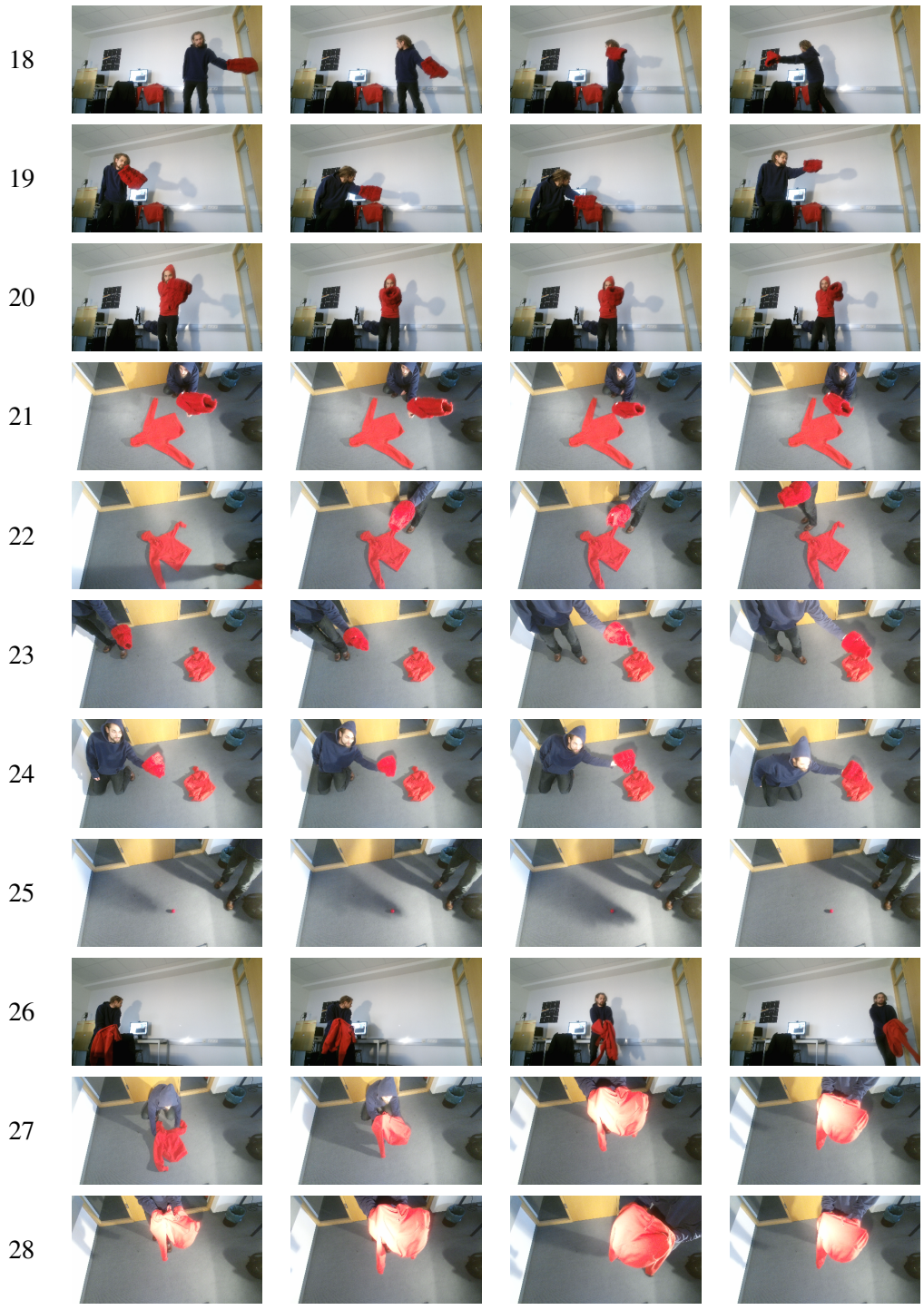
In den Sequenzen 25 bis 28 wird das Verfahren vor die Herausforderung von unterschiedlichen Lichtverhältnissen gestellt, dabei sind Teile des Objektes im Schatten (Sequenz 26) oder starkem Licht ausgesetzt (Sequenz 28). In den anderen beiden Sequenzen verändert sich die Beleuchtung des Objektes. So wird in Sequenz 25 langsam ein zunehmend dunklerer Schatten auf das Objekt geworfen, während sich das Objekt in Sequenz 27 schnell unter eine starke Lichtquelle bewegt. Es wird dabei vom Verfahren erwartet, dass es mit den Änderungen umgehen kann. Sind diese durch die Beleuchtung erzeugten Kontraste bereits im ersten Bild enthalten, wie in Sequenz 28, so fließen sie bereits in das Modell mit ein und sollten entsprechend keine Probleme bei der Verfolgung verursachen. Da das Modell stets aktualisiert wird, sollten wechselnde Lichtverhältnisse das Verfahren ebenfalls nicht behindern. Schlagartige Änderungen der Lichtverhältnisse können das Verfahren jedoch beeinflussen, da das Modell in solchen Fällen nicht langsam angepasst werden kann.

Die Sequenzen 29 und 30 verwenden Objekte unterschiedlicher Farbe, in Sequenz 29 soll lediglich das grüne Tuch, nicht aber die rote Mütze verfolgt werden. Auch wenn diese sich räumlich sehr nah sind, sollte das Verfahren dazu in der Lage sein, sie über die Farbe zu trennen. In Sequenz 30 hingegen gilt es beide Objekte zu verfolgen. Dies stellt für das Verfahren eine Herausforderung dar, da es ein mehrfarbiges Objekt zu verfolgen glaubt und somit der Farbraum weniger eindeutig ist. Die räumlichen Informationen sind zudem ebenfalls nicht eindeutig, da die Objekte abermals physisch mit einer Person verbunden sind und sich zudem voneinander wegbewegen. Das Ergebnis dieser Sequenz ist daher schwer abzuschätzen. Es wird jedoch erwartet, dass das Verfahren nicht in der Lage ist, diese Herausforderung zu bewältigen.

Die abschließenden vier Sequenzen 31 bis 34 liefern Fälle, in denen sich das Objekt stark von der Umgebung abhebt, zumeist durch die Farbe, jedoch auch durch Tiefe. Daher wird für diese Sequenzen erwartet, dass das Verfahren die Objekte erkennt und verfolgt.




07
08
09
10
11
12
13
14
15
16
17


18
19
20
21
22
23
24
25
26
27
28

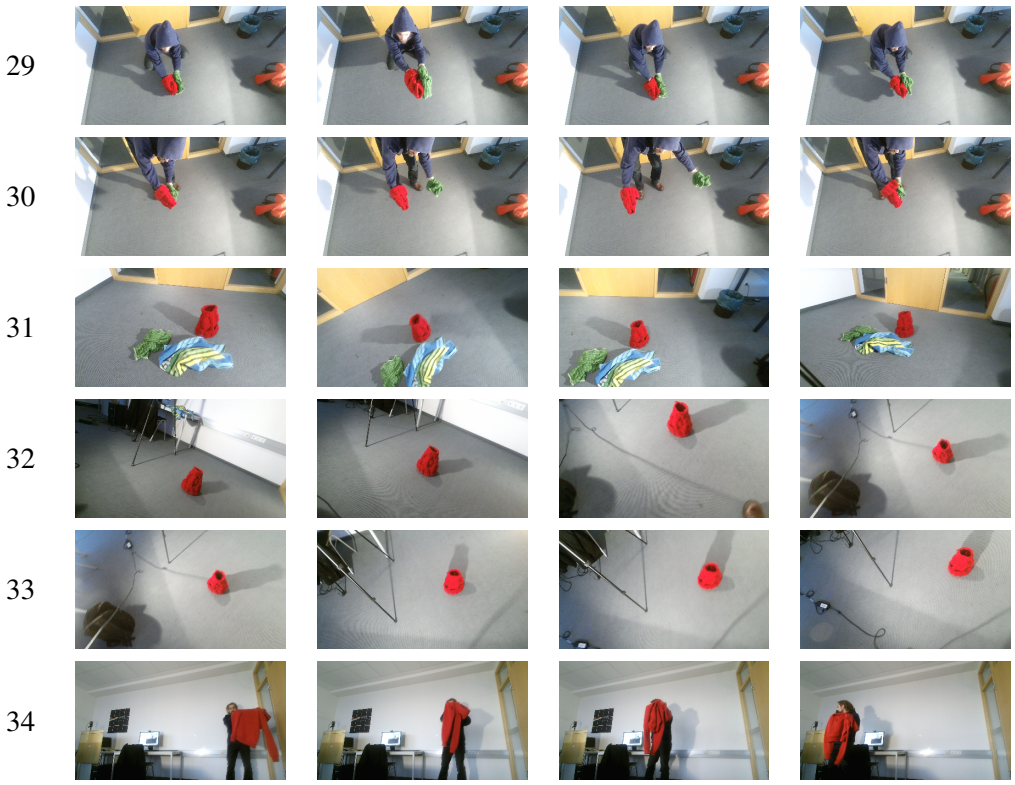


Tabelle 4.1: In dieser Tabelle ist eine Auflistung aller Testsequenzen zu sehen. Sequenzen 00-04 umfassen dabei generierte Daten, während die darauffolgenden Sequenzen mit der Kinect V2 aufgenommen wurden. Eine Sequenz umfasst dabei eine Zeile, in der von links nach rechts der Verlauf der Sequenz zu sehen ist.

4.3 Allgemeiner Aufbau der Tests

Eine Möglichkeit zu prüfen, wie gut das Verfahren arbeitet, ist es die erstellte Klassifikationen mit manuell oder anderweitig erstellten, den tatsächlichen Gegebenheiten entsprechenden Klassifikationen zu vergleichen. Auf diese Weise kann über die Masse an Klassifizierungen ermittelt werden, wie viele Eingangsvektoren korrekt oder fehlerhaft zugeordnet wurden. Wenn dieses Verfahren fehlerhafte Klassifikationen erzeugt, handelt es sich dabei zumeist nicht um einzelne abweichende Eingangsvektoren. Das Verfahren tendiert durch fehlerhafte Klassifikationen dazu, ganze physische Gegenstände in die Klasse des Objektes mit aufzunehmen. Somit ist die genaue Anzahl der korrekt oder fehlerhaft klassifizierten Eingangsvektoren meist irrelevant, da das Verfahren zumeist entweder das Objekt erkennt oder stark von diesem abweicht. Deshalb werden

(a) Gut **(b)** Ausreichend **(c)** Fehlerhaft

Abbildung 4.1: Die drei Abbildungen zeigen jeweils ein Beispiel für die Einteilung in Gut, Ausreichend und Fehlerhaft. In den Testsequenzen sollte jeweils das rote Objekt verfolgt werden: die Mütze (a), der Pullover (b) und der rote Ball (c).

im Folgenden die Auswirkungen der einzelnen Variablen des Verfahrens qualitativ untersucht und die Güte der Erkennung in die folgenden drei Kategorien eingeteilt:

- Gut: Das Objekt konnte korrekt über die gesamte Sequenz hinweg verfolgt werden.
- Ausreichend: Das Objekt konnte mit einigen Abweichungen über die Sequenz hinweg verfolgt werden. Abweichungen können dabei vereinzelt auftretende, fehlerhaft zugeordnete Eingangsvektoren sein, von denen sich das Verfahren jedoch wieder erholt.
- Fehlerhaft: Das Objekt konnte nicht über die ganze Sequenz hinweg verfolgt werden.

Abbildung 4.1 zeigt jeweils ein Beispiel dieser Kategorien. Im Folgenden wird die Anzahl der in einer Kategorie abgeschlossenen Testsequenzen eines Testdurchlaufes als Tupel (Gut, Ausreichend, Fehlerhaft) geschrieben. Sofern nicht anders erwähnt, verwenden alle Testdurchläufe sowohl RGB-Bilder als auch räumliche Informationen, wie in Abschnitt 4.1 beschrieben, als Eingabe. Ebenfalls wird in allen Testdurchläufen die Mahalanobis-Distanz als Distanzmaß auf den Eingangsraum-Modellen und im Schätzungsraum verwendet, sofern nichts Anderes erwähnt ist. Eine Auflistung der Resultate der Testsequenzen in jedem Testdurchlauf entsprechend dieser Kategorien, ist den Tests nachgestellt in Tabelle 4.9 auf Seite 70 aufgeführt.

4.4 Tests zur Parameterbelegung

Verschiedene Ausprägungen der relevanten Parameter des Verfahrens können gravierenden Einfluss auf das Ergebnis haben. Deshalb wird anfänglich untersucht, welche Parameterbelegung

die besten Ergebnisse hervorbringt. Da das Verfahren diverse Parameter verwendet, für die verschiedene Konfigurationen möglich sind, können nicht alle denkbaren Kombinationen überprüft werden. Daher werden die Auswirkungen einzelner Parameteränderungen auf das Verfahren unabhängig voneinander untersucht. In jeder Testreihe werden jeweils nur Änderungen von einzelnen Parametern vorgenommen, während alle anderen Parameter unverändert bleiben. Zur Ermittlung der am besten für den Einsatz geeigneten Parameterbelegung werden folgende Testreihen vorgenommen:

- Die Auswirkung der Verfahren zur Kompensation von gravierenden Größenunterschieden
- Die Auswirkung der Größe der während des MNG verwendeten Nachbarschaft
- Die Auswirkung der Anzahl der initialen Iterationen von MNG auf die Klassifizierung
- Die Auswirkung der Anzahl der nach der Initialisierung zu durchlaufenden Iterationen von MNG
- Die Auswirkung der als Basis verwendeten Anzahl an Iterationen, sowohl während der Initialisierung als auch in nachfolgenden Anwendungen

4.4.1 Testreihe: Kompensation von gravierenden Größenunterschieden während MNG

Wie in Abschnitt 3.5 erwähnt können starke Unterschiede in der Anzahl der zu einer Klasse gehörenden Schätzungsvektoren zu Fehlern führen. In dieser Testreihe werden die zur Vermeidung dieser Fehler vorgeschlagenen Verfahren getestet.

4.4.1.1 Testaufbau

In Tabelle 4.2 ist aufgeführt, welche Tests in diesem Zusammenhang mit welcher Variablenbelegung durchlaufen werden. Die für diese Testreihe relevanten Variablen sind dabei die in Abschnitt 3.5 vorgeschlagene Gewichtung zur Kompensation von starken Größenunterschieden, so wie die Verwendung einer Nachbarschaft. Andere Variablen bleiben während dieser Testreihe unverändert, sodass Änderungen ausschließlich durch die untersuchten Variablen entstehen. Untersucht werden dabei die vier Fälle, welche sich aus der Kombination der zwei Variablen ergeben. Um einen Vergleichswert zu erhalten und entscheiden zu können, ob die Anwendung der

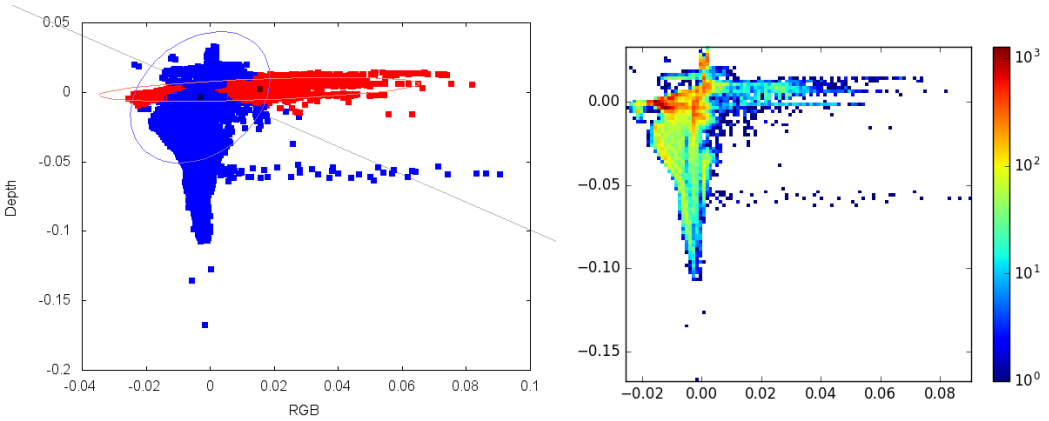
Bezeichnung	Gewichtung der Klassen	Nachbarschaft	Initiale Iterationen	Anpassende Iterationen
Testdurchlauf 1.1	Nein	Nein	10/10	1/10
Testdurchlauf 1.2	Ja	Nein	10/10	1/10
Testdurchlauf 1.3	Nein	10px	10/10	1/10
Testdurchlauf 1.4	Ja	10px	10/10	1/10

Tabelle 4.2: Die Tabelle zeigt die Testkonfigurationen, die verwendet wurden, um die verschiedenen Verfahren zur Kompensation der Größenunterschiede während MNG zu untersuchen.

Verfahren das Ergebnis verbessert, wird zuerst ein Testdurchlauf ohne Gewichtung und Nachbarschaft durchgeführt. Danach werden die beiden Variablen je einzeln geprüft, wobei für die Nachbarschaft eine Größe von 10 Pixeln um das Objekt herum verwendet wird. Zudem wird die gemeinsame Verwendung beider Verfahren überprüft. Dabei wird erwartet, dass das Verfahren unter Anwendung jeweils einer der beiden Methoden besser abschneidet als ohne diese, da entsprechende Fehler entstehen. Zudem wird erwartet, dass die Nachbarschaft ein besseres Ergebnis gegenüber der Gewichtung erzielt, da diese weniger stark in das eigentliche Verfahren eingreift und somit weniger unvorhergesehene Nebeneffekte generiert. Weiterhin wird erwartet, dass die Kombination der beiden Verfahren sich ebenso wie die Verwendung der Nachbarschaft verhält, da die Gewichte der beiden Klassen unter Verwendung der Nachbarschaft etwa gleich ausfallen sollten.

4.4.1.2 Auswertung

Der Verzicht auf den Einsatz der in Abschnitt 3.5 vorgeschlagenen Methoden führt, wie erwartet, zu den beschriebenen Fehlern. Entsprechend ist es dem Verfahren unter der in Testdurchlauf 1.1 verwendeten Konfiguration nur in wenigen Fällen möglich, das Objekt zu verfolgen, was das Ergebnis von (2, 0, 33) zeigt. Nur in den gut getrennten generierten Sequenzen 00 und 04 wird das Objekt verfolgt. In den anderen generierten Sequenzen entstehen durch das große Ungleichgewicht Fehler. Ebenso sind alle anderen Sequenzen diesem Phänomen unterworfen. Die Prototypen werden nah zueinander in den zur Umgebung gehörenden Bereich der Daten gezogen. Durch die große Anzahl an Schätzungsvektoren ist es dem Objekt-Prototypen danach



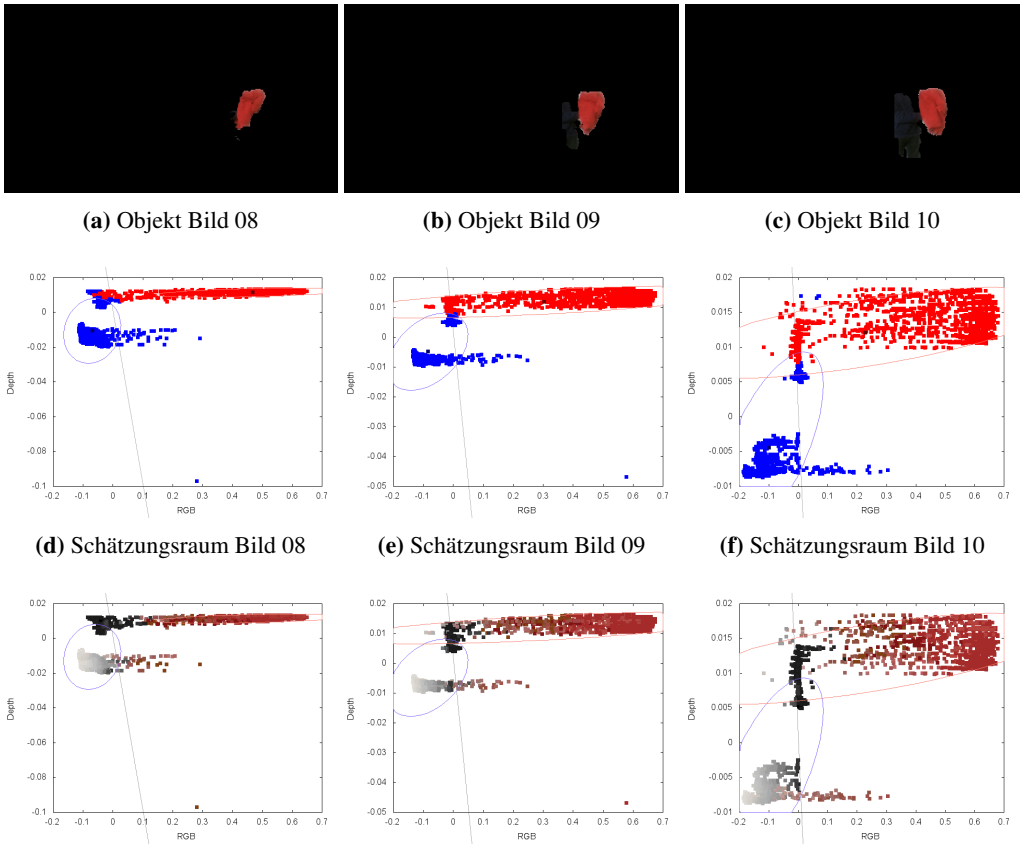
(a) Schätzungsraum **(b)** Schätzungsraum Verteilung

Abbildung 4.2: Diese Grafiken zeigen den Anfang von Testsequenz 15 aus Testdurchlauf 1.2. Zu sehen ist der Schätzungsraum unter verschiedenen Gesichtspunkten. Abbildung (a) zeigt den Schätzungsraum eingefärbt nach Prototyp-Zugehörigkeit. Abbildung (b) zeigt den gleichen Schätzungsraum eingefärbt nach Anzahl der in einem Bereich befindlichen Schätzungsvektoren. In beiden Diagrammen liegt die farbliche Dimension auf der X-Achse und die räumliche Dimension auf der Y-Achse.

nicht möglich, sich von der Umgebung zu lösen. Einzig in den Testsequenzen 00 und 04 ist das Objekt stark genug ausgeprägt, um dem Verfahren die Verfolgung zu ermöglichen.

Testdurchlauf 1.2 zeigt ähnliche Probleme wie Testdurchlauf 1.1 und liefert entsprechende Ergebnisse: (4, 0, 31). Zwar werden die Prototypen stärker von den zum Objekt gehörenden Schätzungsvektoren angezogen als von den zur Umgebung gehörenden, dennoch ist die Konzentration der Umgebungsvektoren häufig so stark, dass beide Prototypen dennoch zur Umgebung gezogen werden. Abbildung 4.2 verdeutlicht dies: In (a) ist zu erkennen, wie der Erwartungswert beider Prototypen zur starken Ansammlung an Schätzungsvektoren gezogen wurde. Wie groß der Unterschied in der Anzahl der Vektoren ist, lässt sich in der zugehörigen Abbildung der Verteilung (b) erkennen.

Testdurchlauf 1.3 liefert bessere Ergebnisse: (9, 4, 22). Die dabei entstandenen Fehler sind meist auf nicht eindeutig zuzuordnende Schätzungsvektoren, im Folgenden unsichere Schätzungsvektoren genannt, zurück zu führen, die dem Objekt zugeordnet werden, tatsächlich jedoch zur Umgebung gehören. Dies geschieht vor allem, wenn das Objekt durch eine Person gehalten wird, wie in Abbildung 4.3 gezeigt. Dieses Phänomen wird in Abschnitt 4.7 genauer beschrieben.



(a) Objekt Bild 08 **(b)** Objekt Bild 09 **(c)** Objekt Bild 10

(d) Schätzungsraum Bild 08 **(e)** Schätzungsraum Bild 09 **(f)** Schätzungsraum Bild 10

(g) Schätzungsraum Farbe Bild 08 **(h)** Schätzungsraum Farbe Bild 09 **(i)** Schätzungsraum Farbe Bild 10

Abbildung 4.3: Diese Grafiken zeigen eine Sequenz von drei Bildern aus dem Testdurchlauf 1.3, Testsequenz 06 von links nach rechts. In der obersten Reihe sind Bereiche des RGB-Bildes zu sehen, welche als Objekt klassifiziert wurden. In der mittleren Reihe ist der jeweilige Schätzungsraum abgebildet, eingefärbt nach Prototyp-Zugehörigkeit. In der letzten Reihe ist ebenfalls der Schätzungsraum abgebildet, wobei die Schätzungsvektoren entsprechend ihrer Farbe im RGB-Bild eingefärbt sind. Es ist zu erkennen, wie der Objekt-Prototyp sich über die Zeit zur Umgebung orientiert und wie zunehmend Bereiche der Person als Objekt klassifiziert werden.

In Testdurchlauf 1.4 tritt zudem noch ein weiteres Phänomen auf, wodurch die Ergebnisse schlechter ausfallen: (5, 5, 25). Durch die Gewichtung soll ein Gleichgewicht geschaffen werden, um die zu starke Anziehung eines Prototypen durch eine große Menge an Schätzungsvektoren zu vermeiden. Dies führt zu einem anderen Verhältnis der Prototypen. Da die zu einer kleinen Anhäufung gehörenden Schätzungsvektoren entsprechend mehr Gewicht erhalten, wird der zur großen Anhäufung gehörende Prototyp stärker von diesen beeinflusst als vorher. Durch Verwendung der Nachbarschaft sind diese Unterschiede nur gering, zeigen jedoch Auswirkungen. Da in der Regel der kleinere Prototyp des Objekts eine höhere Dichte an Punkten aufweist beziehungsweise eine geringere Standardabweichung, wird der Prototyp der Umgebung anfangs leicht, bei zunehmender Nähe aber immer stärker zum Objekt gezogen. Dies führt teilweise soweit, dass der Erwartungswert des Umgebungs-Prototyps sogar in den eigentlich zum Objekt gehörenden Schätzungsvektoren liegt und das Objekt somit verdrängt.

Die Gewichtung allein zur Kompensation der großen Unterschiede liefert somit nicht das erwartete Ergebnis. Auch in Verbindung mit der Nachbarschaft führt der Einsatz nicht zur Verbesserung der Ergebnisse. Entsprechend wird im Folgenden ausschließlich die Nachbarschaft verwendet, um Fehler durch starke Unterschiede in der Anzahl von Schätzungsvektoren einer Klasse zu vermeiden.

4.4.2 Testreihe: Größe der Nachbarschaft

Die Nachbarschaft des Objektes gibt einen Bereich um das Objekt herum an, der für die Klassifizierung betrachtet wird, wie in Abschnitt 3.5 beschrieben. Sie dient dazu, die Anzahl der zu betrachtenden Schätzungsvektoren und damit auch Störeinflüsse zu reduzieren. In dieser Testreihe wird das Verhalten des Verfahrens im Verhältnis zur Größe der Nachbarschaft betrachtet.

4.4.2.1 Testaufbau

Es werden drei unterschiedliche Größen untersucht. Dazu wird ausschließlich der Wert der Nachbarschaftsgröße geändert, wie Tabelle 4.3 zu entnehmen. Bei den untersuchten Nachbarschaftsgrößen handelt es sich um 5, 10 und 20 Pixel. Es wird erwartet, dass die Vergrößerung der Nachbarschaft mehr Schätzungsvektoren, die nicht zum Objekt gehören, in die Klassifizierung einbringt. Der Umgebungs-Prototyp wird entsprechend eine geringere Abweichung aufweisen, wodurch unsichere Schätzungsvektoren eher dem Objekt-Prototypen zugeordnet werden. Umgekehrt gilt dies für die Verringerung der Nachbarschaft. Sinkt die Anzahl der zur Umgebung gehörenden Schätzungsvektoren, wird mit einer verhältnismäßig größeren Abweichung des Pro-

Bezeichnung	Gewichtung der Klassen	Nachbarschaft	Initiale Iterationen	Anpassende Iterationen
Testdurchlauf 2.1	Nein	5px	10/10	1/10
Testdurchlauf 2.2	Nein	10px	10/10	1/10
Testdurchlauf 2.3	Nein	20px	10/10	1/10

Tabelle 4.3: Die Tabelle zeigt die für die Tests zur Größe der Nachbarschaft verwendeten Konfigurationen.

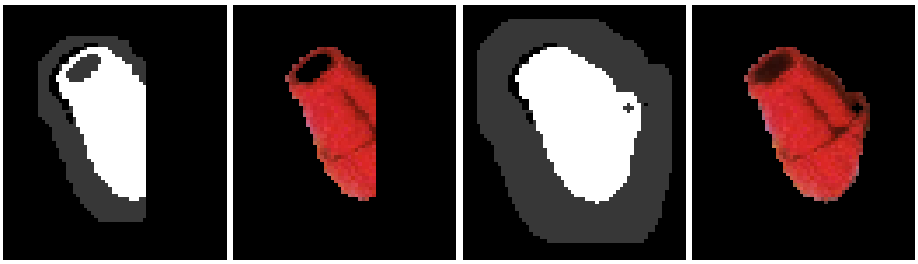
totypen gerechnet. Dies macht die Zuordnung von unsicheren Schätzungsvektoren zur Umgebung wahrscheinlicher. Daher wird erwartet, dass eine mittlere Größe für die Nachbarschaft die robusteste Erkennung liefert.

4.4.2.2 Auswertung

Die große Nachbarschaft von 20 Pixeln zeigt die erwarteten Probleme, daher schneidet Testdurchlauf 2.3 am schlechtesten ab: (0, 8, 27). Der Umgebungs-Prototyp wird stärker zum Zentrum der zur Umgebung gehörenden Schätzungsvektoren gezogen und somit werden mehr unsichere Schätzungsvektoren dem Objekt zugeordnet. Entsprechend werden zunehmend zur Umgebung gehörende Schätzungsvektoren dem Objekt hinzugefügt. In einigen Fällen wird die eigentliche Umgebung dabei fast gänzlich durch das Objekt verdrängt.

Mit einer Nachbarschaftsgröße von 10 Pixeln liefert das Verfahren in Testdurchlauf 2.2 bessere Ergebnisse: (9, 4, 22). Hier wird der Objekt-Prototyp nach wie vor durch den Einfluss von unsicheren Schätzungsvektoren weiter ausgedehnt und geht somit nach und nach in die Umgebung über.

Durch die kleine Nachbarschaft verringert Testdurchlauf 2.1 diesen Effekt, kann ihn jedoch auch nicht gänzlich vermeiden. Entsprechend fallen die Testergebnisse aus: (12, 2, 21). Die besseren Ergebnisse folgen aus der unschärferen und somit weiter ausgedehnten Umgebung, wodurch nicht eindeutig zuzuordnende Schätzungsvektoren eher an die Umgebung fallen. Auch hier ist es dem Verfahren jedoch beim Großteil der Sequenzen nicht möglich, das Objekt dauerhaft zu erkennen und zu verfolgen.



(a) (b) (c) (d)

Abbildung 4.4: Diese Abbildungen zeigen die gleichen Ausschnitte aus dem Ergebnis der Klassifikation von Testsequenz 14 in Testdurchlauf 2.1 ((a) und (b)) und 2.2 ((c) und (d)) zum gleichen Zeitpunkt. Abbildungen (a) und (c) zeigen dabei die Unterteilung der Nachbarschaft in Objekt (weiß) und Umgebung (grau). Messungen außerhalb der Nachbarschaft oder ohne räumliche Messungen sind schwarz dargestellt. Abbildungen (b) und (d) zeigen jeweils das hervorgehende Objekt im Farbbild. Es lässt sich erkennen, dass das Objekt in Testdurchlauf 2.1 abgeschnitten ist, da ein Teil des Objektes außerhalb der Nachbarschaft liegt.

Ein wichtiger Gesichtspunkt wurde hierbei jedoch außer Acht gelassen. Bewegt sich ein Objekt zwischen zwei Aufnahmen außerhalb der betrachteten Nachbarschaft, werden Teile des Objektes nicht mehr von MNG berücksichtigt und als Umgebung klassifiziert. Abbildung 4.4 zeigt dies beispielhaft an den Testdurchläufen 2.1 und 2.2. Für die Ergebnisse dieser Testreihe wurde diese Eigenschaft außer Acht gelassen und nur die Klassifikation innerhalb der Nachbarschaft bewertet. Testdurchlauf 2.1 liefert mit der kleinsten Nachbarschaft somit die besten Ergebnisse, tatsächlich liegen allerdings häufig Teile des Objekts außerhalb der Nachbarschaft, da diese zu klein gewählt ist. Bei einer Größe von 10 Pixeln tritt dieses Problem nur noch stellenweise auf. Die Nachbarschaftsgröße von 20 Pixeln beinhaltete zwar stets das gesamte Objekt, lieferte aber schlechtere Erkennungsraten. Daher wird als Kompromiss eine Größe von 10 Pixeln verwendet.

4.4.3 Testreihe: Anzahl der initialen Iterationen von MNG

Durch die Anzahl der für die erste Messung durchlaufenen Iterationen kann der Einfluss der Initialisierung auf MNG verändert werden. Da jedoch Vorwissen über den Schätzungsraum in die Initialisierung einfließt, besteht die Möglichkeit einer bereits zutreffenden Initialisierung. Entsprechend wird in dieser Testreihe untersucht, wie stark die Initialisierung durch MNG angepasst werden muss, um ein Objekt verfolgen zu können.

Bezeichnung	Gewichtung der Klassen	Nachbarschaft	Initiale Iterationen	Anpassende Iterationen
Testdurchlauf 3.1	Nein	10px	10/10	1/10
Testdurchlauf 3.2	Nein	10px	5/10	1/10
Testdurchlauf 3.3	Nein	10px	1/10	1/10

Tabelle 4.4: Die Tabelle zeigt die für die Tests zur Anzahl der initialen Iterationen verwendeten Konfigurationen.

4.4.3.1 Testaufbau

Für die Testdurchläufe werden die in Tabelle 4.4 aufgeführten Konfigurationen verwendet. Es wird dabei geprüft, wie sich das Verfahren bei zehn, fünf und einer initialen Iteration verhält. Da die Gewichtung in jeder Iteration auch von der Gesamtzahl von theoretisch durchlaufenen Iterationen abhängt, wird jeweils die gleiche Gesamtzahl von zehn Iterationen verwendet. Da die Einflüsse auf die Prototypen mit zunehmenden Iterationen lokaler ausfallen, wird erwartet, dass sich die Prototypen in Testdurchläufen mit mehr Iterationen genauer an die Struktur der Schätzungsvektoren anpassen. Gleichzeitig wird jedoch auch erwartet, dass eine solche genaue Anpassung aufgrund der Initialisierung nicht notwendig ist und sich, wenn überhaupt, nur anfänglich minimale Abweichungen ergeben werden.

4.4.3.2 Auswertung

Die Testdurchläufe unterscheiden sich, wie erwartet, nur anfänglich voneinander. Insgesamt liefern alle drei Testdurchläufe ähnliche Ergebnisse. So erreicht Testdurchlauf 3.1 mit zehn von zehn initialen Iterationen eine Erkennung von (9, 4, 22), Testdurchläufe 3.2 und 3.3 unterscheiden sich jedoch kaum, mit (9, 3, 23). Mit Ausnahme von sieben Testsequenzen gleichen sich alle Sequenzen in Testdurchlauf 3.2 denen von Testdurchlauf 3.1 bereits nach spätestens vier Messungen an. In Testdurchlauf 3.3 erreichen die Sequenzen eine solche Übereinstimmung erst nach bis zu acht Messungen, mit Ausnahme von zehn Sequenzen. Die Testsequenzen, die sich nicht innerhalb der ersten Messungen exakt angleichen, liefern dennoch vergleichbare Ergebnisse. Zusammenfassend betrachtet liefert die genauere Anpassung der Prototypen durch mehr Iterationen minimal bessere Ergebnisse. Bei weniger initialen Iterationen ist das Verfahren anfäl-

liger gegenüber unsicheren Schätzungsvektoren in den ersten Messungen. Sollte die Laufzeit der Initialisierung relevant für den Einsatz des Verfahrens sein, kann daher die Anzahl der initialen Iterationen auf Kosten der Erkennungsrate, entsprechend reduziert werden.

4.4.4 Testreihe: Anzahl der anpassenden Iterationen von MNG

Die Anzahl der Iterationen, welche zur Anpassung der Prototypen auf neue Messungen verwendet wird, kann Auswirkungen auf die Prototypen und somit auf die Klassifizierung der Daten haben. Da MNG ein verhältnismäßig rechen- und somit laufzeitintensiver Vorgang des Verfahrens ist, bedeutet jede zusätzliche Iteration zur Anpassung eine deutlich erhöhte Gesamtlaufzeit für jede Messung. Das Ergebnis der vorherigen Messung wird als gute Initialisierung der Prototypen angesehen, vorausgesetzt Objekt und Umgebung haben sich in zwei aufeinander folgenden Messungen nicht stark verändert. Werden mehr Iterationen zur Anpassung durchlaufen, sinkt der Einfluss dieser Initialisierung und die Prototypen passen sich den Daten genauer an. Daher soll in dieser Testreihe untersucht werden, ob eine langwierige Anpassung notwendig ist, oder ob eine geringfügige Anpassung der Prototypen bereits ausreicht, um ein Objekt zu verfolgen.

4.4.4.1 Testaufbau

Zur Untersuchung des Einflusses der zur Anpassung genutzten Iterationen auf die Erkennung, werden die drei in Tabelle 4.5 aufgeführten Konfigurationen getestet. Es werden eine, fünf und zehn Iterationen zur Anpassung durchgeführt. Als Basis dient eine Gesamtzahl von zehn Iterationen, dabei werden jeweils nur die letzten Iterationen durchlaufen. Es wird erwartet, dass sich die Prototypen mit zunehmender Anzahl an Iterationen genauer an den Schätzungsraum anpassen. Jedoch wird weiterhin erwartet, dass die Initialisierung auf der letzten Messung in Verbindung mit wenigen anpassenden Iterationen bereits zu Verfolgung eines Objektes genügt.

4.4.4.2 Auswertung

Anders als erwartet führt die größere Anzahl an Iterationen nicht zu einer genaueren Klassifizierung. Testdurchlauf 4.1 durchläuft am wenigsten Iterationen, erbringt jedoch das beste Ergebnis: (9, 4, 22). Testdurchlauf 4.2 und 4.3 schneiden mit (8, 2, 25) und (9, 1, 25) schlechter ab. Da die Prototypen in den ersten Iterationen global beeinflusst werden, haben hohe Konzentrationen von Schätzungsvektoren entsprechend einen stärkeren Einfluss auf die Prototypen. Dies lässt sich exemplarisch in Abbildung 4.5 erkennen, welche Auszüge aus Testsequenz 05 zeigt. Dort

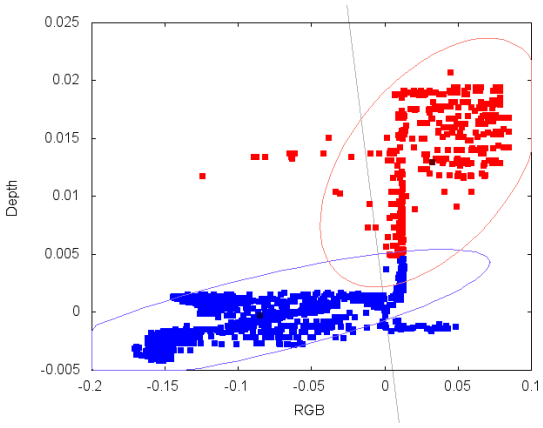
Bezeichnung	Gewichtung der Klassen	Nachbarschaft	Initiale Iterationen	Anpassende Iterationen
Testdurchlauf 4.1	Nein	10px	10/10	1/10
Testdurchlauf 4.2	Nein	10px	10/10	5/10
Testdurchlauf 4.3	Nein	10px	10/10	10/10

Tabelle 4.5: Die Tabelle zeigt die für die Tests zur Anzahl der zur Anpassung durchgeführten Iterationen verwendeten Konfigurationen.

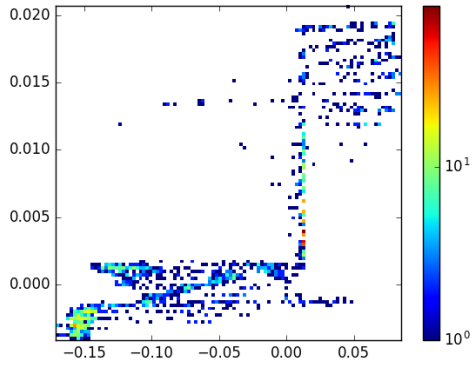
ist von oben nach unten drei mal der Schätzungsraum der verschiedenen Testdurchläufe 4.1 bis 4.3 zum gleichen Zeitpunkt abgebildet. Einmal in der Unterteilung nach Klassen (links) und einmal als Häufigkeitsverteilung (rechts). Es ist zu erkennen, wie die Prototypen mit zunehmender Unabhängigkeit von der Initialisierung stärker durch die hohen Konzentrationen der Schätzungsvektoren angezogen werden. Die eigentlich zum Objekt gehörenden Schätzungsvektoren entsprechen der Anhäufung oben rechts. Sowohl die Anhäufung unten links, als die unsicheren Schätzungsvektoren, welche die Anhäufungen verbinden, sind nicht Teil des Objektes. Es ist zu erkennen, dass die Prototypen durch ihre stärkere Unabhängigkeit vermehrt durch Schätzungsvektoren außerhalb ihrer Klasse angezogen werden und somit dazu tendieren, sich von ihren eigentlichen Klassen weg, in Richtung starker Anhäufungen zu bewegen. Dies führt speziell beim Objekt, zu vermehrt fälschlicherweise aufgenommenen Schätzungsvektoren, was wiederum weitere Fehler nach sich zieht. Die lokale Anpassung der Initialisierung liefert daher bessere Ergebnisse, da sie Abweichungen durch globale Einflüsse vermeidet. Daher liefert eine geringe Anzahl der zur Anpassung verwendeten Iterationen in diesem Fall eine besseres Erkennung.

4.4.5 Testreihe: Anzahl der Basis an Iterationen von MNG

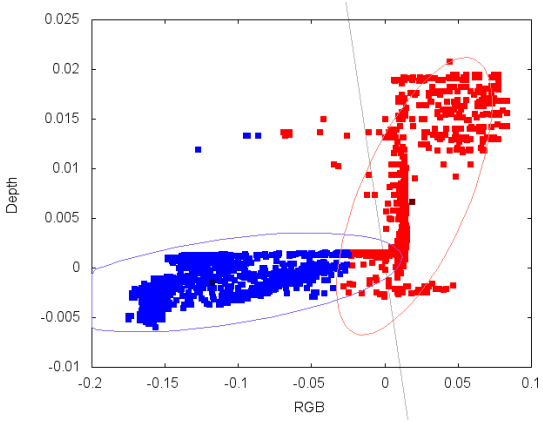
Die Anzahl der Basis an Iterationen von MNG hat Einfluss auf die Gewichtung der Schätzungsvektoren innerhalb von MNG. Während über die Anzahl der durchlaufenen Iterationen gesteuert werden kann, wie lokal Anpassungen ausfallen, steuert die zugrundegelegte Basis an Iterationen, wie stark sich die Gewichte mit jeder Iteration ändern. Diese regeln somit, wie weich oder auch grob die Schritte der einzelnen Iterationen verlaufen. Eine große Anzahl führt zu einem weichen Übergang zwischen den Gewichten und somit zu einer langsameren, aber präziseren Bestimmung der Prototypen. Eine geringere Anzahl führt zu stärkeren Sprüngen in den Gewichten



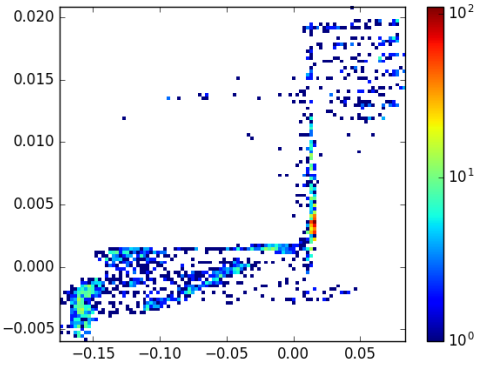
(a) Testdurchlauf 4.1: Schätzungsraum nach Klassen



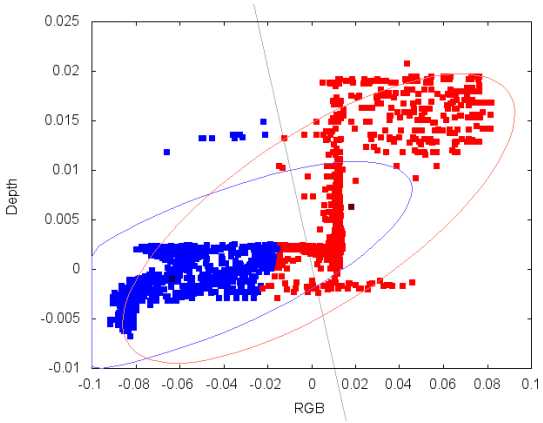
(b) Testdurchlauf 4.1: Schätzungsraum nach Anzahl



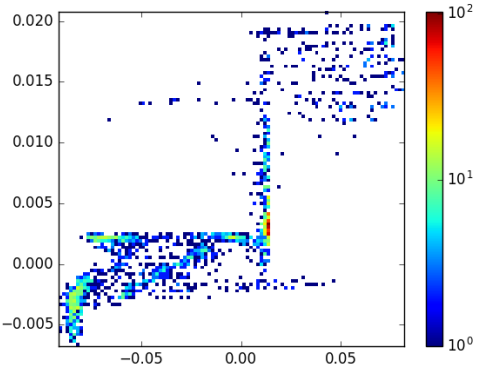
(c) Testdurchlauf 4.2: Schätzungsraum nach Klassen



(d) Testdurchlauf 4.2: Schätzungsraum nach Anzahl



(e) Testdurchlauf 4.3: Schätzungsraum nach Klassen



(f) Testdurchlauf 4.3: Schätzungsraum nach Anzahl

Abbildung 4.5: Die Abbildungen zeigen jeweils den Schätzungsraum von Testsequenz 05 zum gleichen Zeitpunkt in den Testdurchläufen 4.1 bis 4.3. Jede Zeile ist dabei einem Testdurchlauf zugeordnet. Links befindet sich jeweils der nach Prototyp-Zugehörigkeit eingefärbte Schätzungsraum, rechts der nach Anzahl der Schätzungsvektoren im jeweiligen Bereich eingefärbte Schätzungsraum. Über die Grafiken hinweg ist zu erkennen, wie sich die Prototypen im Vergleich zu den Ansammlungen an Schätzungsvektoren über die verschiedenen Testdurchläufe verhalten.

Bezeichnung	Gewichtung der Klassen	Nachbarschaft	Initiale Iterationen	Anpassende Iterationen
Testdurchlauf 5.1	Nein	10px	5/5	1/5
Testdurchlauf 5.2	Nein	10px	10/10	1/10
Testdurchlauf 5.3	Nein	10px	20/20	1/20

Tabelle 4.6: Die Tabelle zeigt die für die Tests zur Basis an Iterationen verwendeten Konfigurationen.

über die Iterationen hinweg und somit auch zu sprunghaften Änderungen der Prototypen. Dadurch können feine Strukturen unter Umständen unberücksichtigt bleiben. In dieser Testreihe wird daher die Auswirkung von verschiedenen Anzahlen an zugrundegelegten Iterationen auf die Erkennung des Objektes untersucht.

4.4.5.1 Testaufbau

In dieser Testreihe werden drei unterschiedliche Ausprägungen der Iterationen getestet, die in Tabelle 4.6 aufgeführt sind. Dabei werden Testdurchläufe mit fünf, zehn und 20 Iterationen durchgeführt. Zur Initialisierung werden stets alle Iterationen durchlaufen, zur Anpassung wird jedoch jeweils nur die letzte Iteration verwendet. Es wird erwartet, dass eine größere Anzahl an Iterationen bessere Ergebnisse liefert, da die Initialisierung präziser wird und die Anpassung lokaler ausfällt. Entsprechend wird von Testlauf 5.1 die geringste Anzahl an korrekt klassifizierten Testsequenzen erwartet.

4.4.5.2 Auswertung

Anders als erwartet liefern alle drei Testdurchläufe gleiche Erkennungsraten: (9, 4, 22). Unterschiede lassen sich erst bei der detailierten Betrachtung erkennen. Bis auf fünf Testsequenzen liefern Testdurchlauf 5.2 und 5.3 die gleichen Resultate. Die fünf verbleibenden Testsequenzen liefern jeweils ab der 3., 4., 5., 11. und 18. Messung ebenfalls die gleichen Ergebnisse.

Auch ein Großteil der Testsequenzen in Testdurchlauf 5.1 liefert bereits von Anfang an mit den anderen Testdurchläufen übereinstimmende Ergebnisse, lediglich neun Testsequenzen unterscheiden sich. Von diesen neun Testsequenzen liefern sieben ebenfalls nach wenigen Messungen

die gleichen Ergebnisse, wie die anderen Testdurchläufe. Die verbleibenden zwei Testsequenzen liefern zwar keine exakt mit den anderen Testdurchläufen übereinstimmenden Ergebnisse, weichen jedoch nur geringfügig von diesen ab.

Wie beschrieben, führt die geringere Zahl an zugrundeliegenden Iterationen zu stärkeren Sprüngen in der Gewichtung. Dies führt schon während der Initalisierung zur fehlerhaften Zuordnung von Schätzungsvektoren. In dieser Testreihe haben sich dadurch keine Unterschiede in den Erkennungsraten ergeben, da die zu Fehlern führenden Unsicherheiten über einen längeren Zeitraum bestanden und somit lediglich in späteren Iterationen zu Fehlern geführt haben. Ab einer gewissen Anzahl von Iterationen bringt eine größere Anzahl keine relevanten Änderungen mit sich. Entsprechend werden im Folgenden weiterhin zehn Iterationen verwendet.

4.4.6 Gewählte Konfiguration

Es wurden verschiedene Parameterbelegungen überprüft, um eine allgemein einsetzbare Konfiguration zu ermitteln. Dabei wurde überprüft, ob eine zusätzliche Gewichtung der Schätzungsvektoren oder die Verwendung einer Nachbarschaft das Verhalten von MNG positiv beeinflusst. Es stellte sich heraus, dass die Nachbarschaft dazu geeignet ist, weswegen diese verwendet wird. Eine kleinere Nachbarschaft erbringt zwar innerhalb dieser eine bessere Erkennung, kann jedoch nur bei geringer Bewegung des Objektes eingesetzt werden. Als Kompromiss zwischen Erkennungsrate und möglicher Bewegung des Objektes wird daher eine Nachbarschaftsgröße von 10 Pixeln verwendet. Für den MNG-Schritt werden als Basis zehn Iterationen genutzt, da diese eine ausreichend feine Anpassung ermöglichen. Zur Initialisierung werden alle dieser zehn Iterationen durchlaufen, um eine bestmögliche Initialisierung sicher zu stellen. Die Anpassung der Prototypen geschieht jedoch nur über eine Iteration. Dies ermöglicht es, das Objekt ohne Störungen durch globale Effekte lokal anzupassen. Diese Konfiguration wird für die folgenden Tests verwendet.

4.5 Tests des verwendeten Distanzmaßes

Die Wahl des Distanzmaßes spielt eine entscheidende Rolle, da dieses zum einen die Qualität der schwachen Klassifikatoren als auch das Ergebnis von MNG beeinflusst. Anders als der Euklidische Abstand ist die Mahalanobis-Distanz in der Lage, sich an die zugrundeliegenden Daten anzupassen. Daher wird erwartet, dass die Verwendung der Mahalanobis-Distanz bessere Ergebnisse als der Einsatz des Euklidischen Abstands erbringt. Zudem wird eine *e*-Funktion

verwendet, um den Fokus auf die nähere Umgebung der Verteilungen zu setzen. Gleichzeitig dient sie dazu den Wertebereich auf ein vorgesehenes Intervall abzubilden und starke Schwankungen somit zu vermeiden. Die Auswirkungen dieser Mechanismen sollen in dieser Testreihe untersucht werden.

4.5.1 Testaufbau

Zur Untersuchung des Einflusses des genutzten Distanzmaßes werden verschiedene Kombinationen betrachtet, wie in Tabelle 4.7 aufgeführt. Es wird der Einsatz der Mahalanobis-Distanz zur Bestimmung der Schätzungsvektoren durch die schwachen Klassifikatoren und während MNG unterschieden. Zudem wird die Anwendung der *e*-Funktion als eine weitere Variable behandelt.

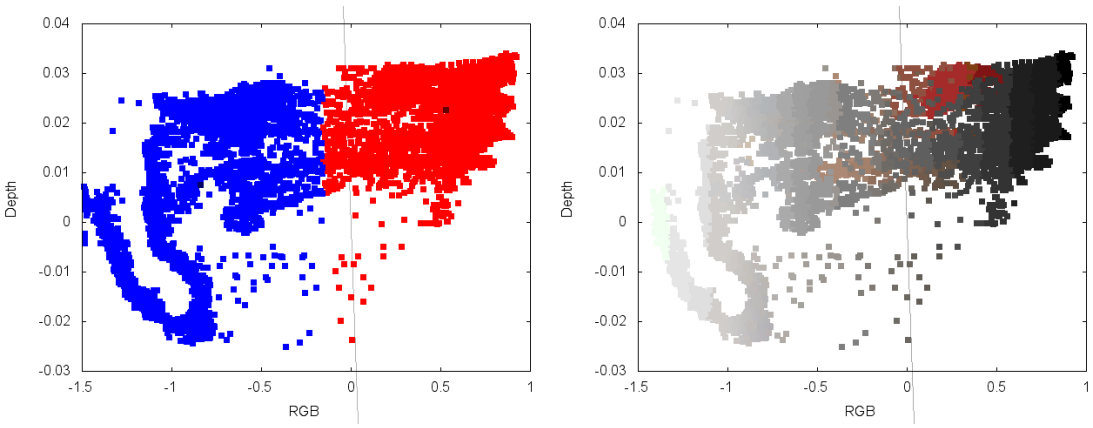
In den Testdurchläufen 6.1 und 6.2 verwenden sowohl die schwachen Klassifikatoren, als auch MNG die Mahalanobis-Distanz. Testdurchlauf 6.1 nutzt dabei die *e*-Funktion, während in Testdurchlauf 6.2 auf deren Einsatz verzichtet wird. Die Testdurchläufe 6.3 und 6.4 verhalten sich analog dazu, allerdings verwenden die schwachen Klassifikatoren und MNG in diesen Testdurchläufen jeweils den Euklidischen Abstand. Um den Einfluss des Distanzmaßes auf MNG zu untersuchen wird anschließend in Testdurchlauf 6.5 MNG mit dem Euklidischen Abstand ausgeführt, während die schwachen Klassifikatoren mit der Mahalanobis-Distanz arbeiten. Testdurchlauf 6.6 bildet den umgekehrten Fall ab, wobei MNG die Mahalanobis-Distanz verwendet und die schwachen Klassifikatoren mit dem Euklidischen Abstand arbeiten. Testdurchläufe 6.5 und 6.6 verwenden dabei die *e*-Funktion. Erwartet wird, dass die Mahalanobis-Distanz eine bessere Verfolgung des Objektes ermöglicht, da das Distanzmaß besser auf die Daten abgestimmt ist. Ebenfalls wird erwartet, dass der Einsatz der *e*-Funktion dabei hilft, die Klassen zu trennen und somit die Klassifizierung verbessert.

4.5.2 Auswertung

Alle Testdurchläufe, in denen die schwachen Klassifikatoren mit dem Euklidischen Abstand arbeiten, weisen den gleichen Fehler auf. Durch die Verwendung des Euklidischen Abstands wird das Ergebnis des schwachen Klassifikators der farblichen Dimension unbrauchbar. Da alle Dimensionen des Farbraumes gleich gewichtet werden, ist die Distanz zwischen Werten mit ähnlicher Ausprägung der drei Dimensionen sehr gering. Der schwache Klassifikator liefert somit nach Helligkeit geordnete Werte, wodurch sich die eigentlich zum Objekt und zur Umgebung gehörenden Messungen nicht mehr trennen lassen, siehe Abbildung 4.6. Dies betrifft Testdurchläufe 6.3, 6.4 und 6.6.

Bezeichnung	Eingangsraum-Modelle	MNG	*e*-Funktion
Testdurchlauf 6.1	Mahalanobis-Distanz	Mahalanobis-Distanz	Ja
Testdurchlauf 6.2	Mahalanobis-Distanz	Mahalanobis-Distanz	Nein
Testdurchlauf 6.3	Euklidischer Abstand	Euklidischer Abstand	Ja
Testdurchlauf 6.4	Euklidischer Abstand	Euklidischer Abstand	Nein
Testdurchlauf 6.5	Mahalanobis-Distanz	Euklidischer Abstand	Ja
Testdurchlauf 6.6	Euklidischer Abstand	Mahalanobis-Distanz	Ja

Tabelle 4.7: Die Tabelle zeigt die für die Tests des verwendeten Distanzmaßes genutzten Konfigurationen.



(a) Schätzungsraum Klassifikation

(b) Schätzungsraum in RGB-Farbe

Abbildung 4.6: Diese Abbildungen zeigen jeweils den Schätzungsraum von Testsequenz 06 in Testdurchlauf 6.3. Abbildung (a) zeigt den Schätzungsraum nach Zuordnung zu den Prototypen eingefärbt. Abbildung (b) zeigt den gleichen Schätzungsraum, jedoch sind die Schätzungsvektoren nach ihrer Repräsentation im Farbraum eingefärbt. Es ist zu erkennen, dass sie ihrer Helligkeit nach aufgereiht sind.

In den Testdurchläufen 6.3 und 6.4 wirkt sich zudem die Verwendung des Euklidischen Abstands durch MNG negativ auf die Erkennung aus. In den aufgenommenen Testsequenzen ist die räumliche Dimension im Schätzungsraum wesentlich geringer ausgedehnt als die farbliche Dimension. Durch Verwendung des Euklidischen Abstands hat die räumliche Dimension somit kaum Einfluss auf die Prototypen. So teilen diese beiden Testdurchläufe die Daten meist nach Helligkeit, da die farbliche Dimension nach Helligkeit sortiert ist. Somit gelingt die Trennung nur in den ohnehin nach Helligkeit trennbaren Sequenzen, welche neben vier der generierten Testsequenzen noch weitere vier reale Sequenzen umfasst. Beide Testdurchläufe erreichen so folgende Erkennung: (8, 0, 27).

Testdurchlauf 6.6 verwendet im MNG-Schritt die Mahalanobis-Distanz. Die räumlichen Informationen allein reichen jedoch zumeist nicht zur Verfolgung des Objektes aus. Durch die Sortierung der farblichen Dimension wird die Erkennung weiter gestört. Daher kann das Objekt in Testdurchlauf 6.6 nur in vier generierten und zwei weiteren Testsequenzen verfolgt werden. Die Ergebnisse fallen gegenüber denen von Testdurchlauf 6.3 und 6.4 schlechter aus: (6, 0, 29).

Im Gegensatz dazu stehen die Ergebnisse von Testdurchlauf 6.5. Da die schwachen Klassifikatoren die Mahalanobis-Distanz verwenden, ist eine bessere Trennung der erzeugten Schätzungen in Objekt und Umgebung möglich. Durch Verwendung des Euklidischen Abstands durch MNG entsteht jedoch das bereits in Testdurchlauf 6.3 und 6.4 beschriebene Problem. Die räumliche Dimension ist wesentlich schwächer ausgedehnt als die farbliche Dimension, wodurch die Messungen vornehmlich entsprechend der farblichen Dimension klassifiziert werden. In vielen der Testsequenzen ist eine Verfolgung des Objektes über die Farbe möglich, wodurch sich für Testdurchlauf 6.5 eine Erkennung von (16, 3, 16) ergibt. Da jedoch tatsächlich die räumliche Dimension ignoriert wird und der Erfolg nur dem Aufbau der Testsequenzen zuzuschreiben ist, kann dieses Ergebnis nicht als Regelfall angesehen werden.

Die beiden Testdurchläufe 6.1 und 6.2 verwenden beide die Mahalanobis-Distanz, sowohl zur Berechnung der Schätzungen durch die schwachen Klassifikatoren, als auch im MNG-Schritt. Die Verwendung der e-Funktion in Testdurchlauf 6.1 legt den Fokus, wie erwartet, auf den Nahbereich der Verteilungen im Eingangsraum und liefert so deutlicher voneinander getrennte Anhäufungen im Schätzungsraum. Dies hilft die einzelnen Prototypen im Falle von vielen unsicheren Schätzungsvektoren besser voneinander abzugrenzen. Testdurchlauf 6.1 erreicht dadurch ein Ergebnis von (9, 4, 22). Testdurchlauf 6.2 liefert ähnliche Ergebnisse: (10, 2, 23). Von den vier Testsequenzen mit einigen Fehlern in Testdurchlauf 6.1 können in Testdurchlauf 6.2 somit in nur zweien das Objekt verfolgt werden. Eine Testsequenz liefert in Testdurchlauf 6.2 jedoch

ein besseres Ergebnis als in Testdurchlauf 6.1. In diesem Fall führt genau die Verwendung der e-Funktion zu einem Problem. Dadurch, dass einige der eigentlich zur Umgebung gehörenden Schätzungsvektoren dem Objekt zugeordnet werden und dieses ohnehin nur einen sehr kleinen Bereich im Farbraum einnimmt, werden die zur Umgebung zugeordneten Schätzungsvektoren stärker zum Objekt gezogen als es ohne die e-Funktion der Fall ist. Dabei handelt es sich jedoch um einen Spezialfall, der nur einmal auftritt. Für den Großteil der Testsequenzen bietet die Anwendung der e-Funktion eine bessere Trennung der Daten und somit eine bessere Klassifikation.

Zusammenfassend betrachtet zeigen diese Tests, dass der Einsatz des Euklidischen Abstands sowohl im Zuge der schwachen Klassifikatoren, als auch durch MNG Fehler erzeugt. Die durch die schwachen Klassifikatoren erzeugten Schätzungen sind nicht in Objekt und Umgebung zu trennen. Die Mahalanobis-Distanz ermöglicht es hingegen, markante Eigenschaften auszunutzen und auf diese Weise in Objekt und Umgebung teilbare Schätzungen abzugeben. Verwendet MNG den Euklidischen Abstand als Distanzmaß, so haben schwach ausgeprägte Dimensionen unter Umständen keinen Einfluss auf die Klassifizierung. Auch hier ermöglicht die Verwendung der Mahalanobis-Distanz eine Anpassung an die Daten. So ist der genutzte Wertebereich in einer Dimension nicht ausschlaggebend für deren Einfluss auf die Klassifizierung.

4.6 Tests zur Kombination von Eingaben

Überschneiden sich die Eigenschaften von Objekt und Umgebung in einem Eingangsraum, entstehen Unsicherheiten, wodurch die Trennung in Objekt und Umgebung erschwert wird. Daher wurde das Verfahren dahingehend entwickelt, mehrere Eingangsräume zu verwenden, um auftretende Unsicherheiten in den Eingangsräumen durch andere Eingangsräume kompensieren zu können. In dieser Testreihe wird deshalb untersucht, wie sich das Verfahren unter Verwendung einzelner und mehrerer Eingangsräume verhält.

4.6.1 Testaufbau

Die Konfiguration der einzelnen Testdurchläufe ist in Tabelle 4.8 aufgeführt. Die von der Kinect V2 gelieferten Messungen werden in den Testdurchläufen 7.1 bis 7.3 einzeln verarbeitet, um einen Vergleich zur Kombination mehrer Messungen zu erhalten. In Testdurchlauf 7.4 und 7.5 wird die Kombination eines Farbbildes mit den zugehörigen räumlichen Informationen untersucht. Die HSV-Bilder werden dabei über die von OpenCV bereitgestellte Funktion zum Konvertieren von BGR- zu HSV-Bildern aus den bereits in den Testsequenzen vorliegen-

Testdurchlauf	7.1	7.2	7.3	7.4	7.5
Eingangsräume	RGB	HSV	XYZ	RGB, XYZ	HSV, XYZ

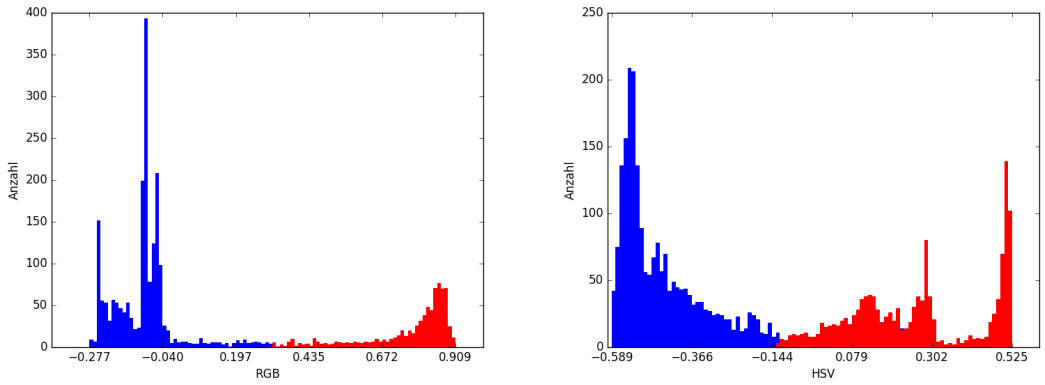
Tabelle 4.8: Die Tabelle zeigt die für die Tests zur Kombination mehrerer Eingangsräume verwendeten Konfigurationen.

den RGB-Aufnahmen erzeugt. Es wird dabei erwartet, dass das Verfahren unter Verwendung der einzelnen Eingangsräume schlechter abschneidet, mit Ausnahme des RGB-Raumes, da sich bereits in Testdurchlauf 6.5 eine Affinität der Testsequenzen für die Unterteilung auf RGB-Basis gezeigt hat. Von den beide Eingangsräume kombinierenden Testdurchläufen wird entsprechend ein besseres Ergebnis erwartet.

4.6.2 Auswertung

Testdurchlauf 7.1 basiert nur auf RGB-Bildern und liefert, wie erwartet, eine hohe Erkennungsrate: (20, 2, 13). Das Verfahren scheitert allerdings daran, Bereiche der Umgebung, welche die gleiche Farbe wie das Objekt aufweisen, vom Objekt zu trennen. Somit liefert Testdurchlauf 7.1 zwar die höchste Erkennungsrate, scheitert aber genau an den Problemen, die durch die Kombination mehrerer Eingangsräume gelöst werden sollen.

Testdurchlauf 7.2, welcher ebenfalls nur Farbbilder als Eingabe verwendet, schließt deutlich schlechter ab. Die Teilung der Daten ist in der HSV-Repräsentation nicht so eindeutig wie in der RGB-Repräsentation. Die vom schwachen Klassifikator verwendete Distanz der Eingangsvektoren zur Modellverteilung erzeugt durch die Repräsentation der Daten durch Farbton, Sättigung und Dunkelstufe weniger eindeutige Werte, wie in Abbildung 4.7 zu sehen ist. Diese Unsicherheiten resultieren unter anderem aus der Farbe des Objektes. Da in den Testsequenzen häufig rote Objekte verwendet werden und dieser Farbton in der HSV-Repräsentation auf die beiden Enden des Spektrums abgebildet wird, liegt der Erwartungswert entsprechend in der Mitte des Spektrums. Somit ist die Entfernung der zum Objekt gehörenden Schätzungsvektoren zum Erwartungswert in dieser Dimension größer als die Entfernung zu den Schätzungsvektoren der Umgebung. Weiterhin sind die Objekte meist über ihren Farbton definiert, während Sättigung und Dunkelstufe stärker über das Spektrum verteilt sind. Im RGB-Raum äußert sich dies durch eine gleichmäßige Änderung in allen Kanälen, die sich über das Modell abbilden und trennen lässt, während im HSV-Raum zwei Kanäle schwer zu trennende Werte enthalten. Somit führt die



(a) **(b)**

Abbildung 4.7: Diese Diagramme zeigen die Verteilung des Schätzungsraums der gleichen Aufnahme aus Testsequenz 14 während Testdurchlauf 7.1 (a) und 7.2 (b). Die Balken sind anteilig nach ihrer Klassifizierung als Umgebung (blau) oder als Objekt (rot) eingefärbt und geben an, wie viele Schätzungsvektoren im auf der X-Achse aufgetragenen Wertebereich des Balkens auftreten. In (a) wird entsprechend ein RGB-Bild als Eingabe verwendet, welches zwei deutliche Anhäufungen von Schätzungsvektoren an den Rändern des Wertebereiches erzeugt. Der auf einem HSV-Bild basierende Schätzungsraum (b) enthält ebenfalls starke Ausprägungen an den äußeren Rändern des verwendeten Wertebereichs, jedoch treten verhältnismäßig mehr unsichere Schätzungsvektoren im mittleren Wertebereich auf.

ausschließliche Verwendung von HSV-Bildern als Eingabe in Testdurchlauf 7.2 zu entsprechend niedrigeren Erkennungsraten: (6, 1, 28).

Die niedrigsten Erkennungsraten liefert jedoch Testdurchlauf 7.3, der ausschließlich mit räumlichen Informationen arbeitet. Wie im Falle von Testdurchlauf 7.2 entstehen auch hier viele nicht eindeutig zuzuordnende Schätzungsvektoren, die zwischen den Prototypen liegen. Solche Unsicherheiten entstehen durch Messungen, die räumlich zwischen den Eingangsraum-Verteilungen liegen. Deren Ursprung liegt meist in der physischen Verbindung von Objekt und Umgebung, da das Objekt auf einer Oberfläche liegt oder durch eine Person oder einen Gegenstand gehalten und bewegt wird. In solchen Fällen entsteht ein fließender Übergang zwischen Objekt und Umgebung. Entsprechend ist das Verfahren unter alleiniger Verwendung der räumlichen Informationen selten in der Lage, das Objekt eindeutig von der Umgebung zu trennen und zu verfolgen. Dies spiegelt sich in den Testergebnissen von Testdurchlauf 7.3 wider: (4, 1, 30).

Die beiden Testdurchläufe 7.4 und 7.5 verwenden je zwei Eingangsräume, bestehend aus einem Farbbild und räumlichen Informationen. Dabei unterliegt Testdurchlauf 7.5 auch den in

bereits Testdurchlauf 7.2 aufgetretenen Unsicherheiten, die sich aus der Verwendung des HSV-Bildes ergeben. Anders als erwartet werden in Testdurchlauf 7.5 die Unsicherheiten des HSV-Bildes und der räumlichen Informationen nicht durch die jeweils andere Dimension aufgelöst. Stattdessen verstärken diese sich gegenseitig, wodurch Testdurchlauf 7.5 schlechtere Ergebnisse liefert als erwartet: (6, 0, 29).

Die Ergebnisse von Testdurchlauf 7.4 fallen mit (9, 4, 22) besser aus. Dies ist auf die Verwendung der gut trennbaren RGB-Bilder zurückzuführen. Durch Kombination mit den RGB-Messungen können die räumlichen Unsicherheiten stellenweise aufgelöst werden.

Die Tests zeigen, dass das Verfahren nicht in der Lage ist Unsicherheiten aufzulösen, wenn ausschließlich unsichere Eingangsräume kombiniert werden. Allerdings lassen sich unsichere Eingangsräume mit besser trennbaren kombinieren, um Unsicherheiten aufzulösen. Im Vergleich zu der ausschließlichen Verwendung von gut trennbaren Eingangsräumen stellt dies eine Verschlechterung der Ergebnisse dar. Da jedoch keinerlei Vorwissen über das Objekt vorausgesetzt wird, kann im Vorfeld nicht entschieden werden, welche Eingangsräume zu welchen Ergebnissen führen. Somit kann das Verfahren genutzt werden, um einen Ausgleich über mehrere Eingangsräume hinweg zu erzeugen.

4.7 Analyse der Fehlerquellen

Entgegen der Erwartungen ist die Anzahl der vollständig korrekt klassifizierten Testsequenzen über alle Testdurchläufe hinweg gering. Eine Übersicht über alle Ergebnisse aller Kombinationen von Testdurchläufen und Testsequenzen ist in Tabelle 4.9 aufgeführt. Hervor stechen vor allem die Testdurchläufe, deren Klassifikation vornehmlich auf Basis von RGB-Daten stattfindet. Dies ist auf die Eigenschaften der Testsequenzen zurück zu führen, die sich farblich in Objekt und Umgebung trennen lassen. In Kombination mit räumlichen Informationen erbringt das Verfahren jedoch schlechtere Ergebnisse. Daher wird untersucht, warum die Kombination mehrerer Eingangsräume nicht mindestens genau so gut abschneidet, wie ein einzelner Eingangsraum für sich genommen.

Zu diesem Zweck werden häufig fehlerhafte Testsequenzen untersucht. Von diesen wird erwartet, entsprechend starke Ausprägungen der zu Fehlern führenden Phänomene aufzuweisen. Als solche werden Testsequenzen eingestuft, die in weniger als fünf Testdurchläufen eine wenigstens ausreichende Erkennung erbrachten. Dies umfasst 20 Testsequenzen.

	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	
1.1	+	+	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
1.2	+	+	-	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1.3	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
1.4	+	-	o	+	+	-	-	-	-	-	-	-	-	-	o	-	-	-	-	-	o	-	-	-	-	+	o	-	-	-	-	o	-	-	+	-
2.1	+	+	+	+	+	-	-	-	-	-	+	-	+	-	+	-	o	-	-	-	-	+	-	-	-	+	o	-	-	-	-	-	-	-	+	-
2.2	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
2.3	+	+	-	+	+	-	-	-	-	-	-	-	-	-	+	-	-	-	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-	+	+	-
3.1	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
3.2	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	-	-	-	-	o	-	o	-	+	-
3.3	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	-	-	-	-	o	-	o	-	+	-
4.1	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
4.2	+	+	-	+	+	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-	-	-	-	-	-	+	o	-	-	-	o	-	-	-	+	-
4.3	+	+	-	+	+	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	-	-	-	-	o	-	-	-	+	-
5.1	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
5.2	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
5.3	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
6.1	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
6.2	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	+	-	-	-	-	+	-	-	-	+	o	-	-	-	-	-	-	-	+	-
6.3	+	+	-	+	+	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	+	+	-	-
6.4	+	+	-	+	+	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	+	+	-	-
6.5	+	+	-	+	+	-	-	+	-	o	+	+	+	o	+	+	+	+	-	-	-	-	-	-	-	+	-	-	-	-	-	+	+	+	o	-
6.6	+	+	-	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-	+	-	-
7.1	+	+	-	+	+	-	+	+	+	o	+	+	+	o	+	+	+	-	+	-	-	-	-	-	-	+	+	-	-	-	-	+	+	+	+	-
7.2	+	o	-	+	+	-	-	-	-	-	-	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	-
7.3	+	-	-	+	+	o	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-
7.4	+	+	-	+	+	-	-	-	-	o	-	-	+	-	+	-	-	-	-	-	-	+	-	-	-	+	o	-	-	-	o	-	o	-	+	-
7.5	+	+	-	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	+	-	-

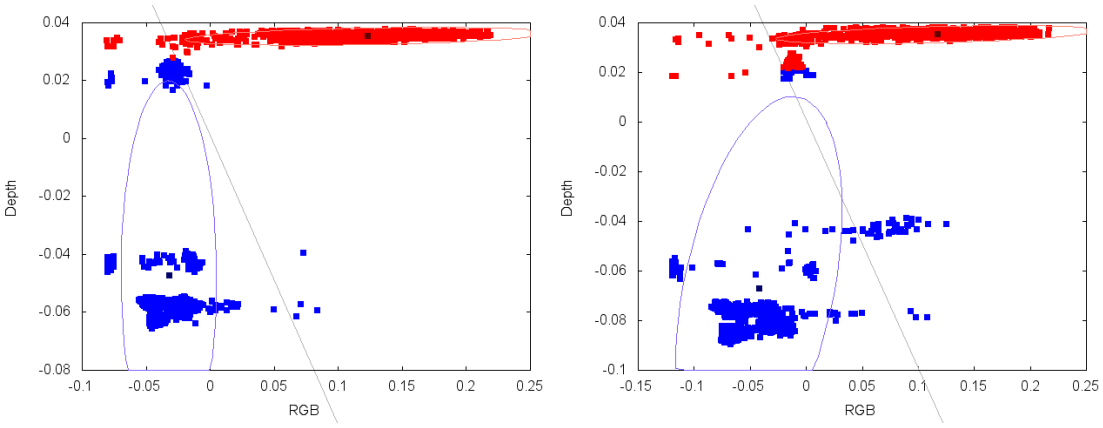
Tabelle 4.9: Die Tabelle zeigt die Resultate aller Testsequenzen in allen Testdurchläufen. Ein '+' repräsentiert das fehlerfreie Verfolgen des Objektes (Gut), 'o' steht für das Verfolgen des Objektes mit einigen Fehlern (Ausreichend), '-' steht für das Scheitern der Verfolgung des Objektes (Fehlerhaft). Testdurchläufe 1.3, 2.2, 3.1, 4.1, 5.2, 6.1 und 7.4 verwenden dabei die gleiche Konfiguration und sind nur der Vollständigkeit halber mehrfach aufgeführt.

Dabei hat die Kategorie, nach welcher die Testsequenzen in Abschnitt 4.2 ausgewählt wurden, nur wenig Einfluss auf deren Fehleranfälligkeit. Abgesehen von den Sequenzen 31 bis 34, in denen ein stark hervorstechendes Objekt gewählt wurde, sind in allen Kategorien diverse fehlerhafte Testsequenzen enthalten.

Die Fehler enstehen vorrangig durch fälschlicherweise als zum Objekt gehörend eingestufte Schätzungsvektoren, die tatsächlich zur Umgebung gehören. Dies führt zu einer Annäherung des Objekt-Prototypen an die Umgebung, was wiederum zur Verstärkung dieses Fehlers führt. Die Ursache solcher fehlerhaft klassifizierter Schätzungsvektoren liegt vornehmlich in den nicht eindeutig zu klassifizierenden, unsicheren Schätzungsvektoren. Diese entstehen, wenn alle schwachen Klassifikatoren gegensätzliche oder gleichermaßen unsichere Schätzungen liefern. In solchen Fällen sammeln sich Schätzungsvektoren zwischen den eigentlichen Clustern des Objektes und der Umgebung.

Diese können auf verschiedene Weisen entstehen, durch ähnliche Ausprägungen in der Farbe oder durch räumliche Nähe. Je nach Ursprung der Unsicherheit bilden sich die Anhäufungen solcher unsicheren Schätzungsvektoren in unterschiedlichen Bereichen des Schätzungsraumes. Sind diese in der räumlichen Dimension dem Objekt sehr ähnlich, in der farblichen Dimension jedoch gut zu trennen, sammeln sich diese unsicheren Schätzungsvektoren entsprechend in der räumlichen Dimension auf Höhe des Objektes, in der farblichen Dimension jedoch auf Höhe der Umgebung. In der in dieser Arbeit verwendeten zweidimensionalen Darstellung des Schätzungsraumes entspricht dies der oberen linken Ecke. Eine andere Ausprägung ist die Ähnlichkeit zum Objekt in der farblichen Dimension, bei gleichzeitig guter Trennung in der räumlichen Dimension. Solche unsicheren Schätzungsvektoren sammeln sich entsprechend in der farblichen Dimension auf Höhe des Objektes und in der räumlichen Dimension auf Höhe der Umgebung. In den genutzten zweidimensionalen Darstellungen des Schätzungsraum entspricht dies der unteren rechten Ecke. Teilweise entstehen jedoch auch Unsicherheiten in beiden Eingangsräumen, wodurch sich die Schätzungsvektoren mittig zwischen Objekt und Umgebung und entsprechend auch in den verwendeten Diagrammen mittig ansiedeln. Diese verbinden dabei oft Objekt und Umgebung, was die korrekte Zuteilung erschwert.

Die Ursachen für solche Unsicherheiten können, wie bereits beschrieben, verschieden sein. Zumeist liefert der Arm einer Person, die das Objekt hält, oder ein Gegenstand, an dem das zu verfolgende Objekt gehalten wird oder auf dem es aufliegt, diese Art von Unsicherheit. Da diese Gegenstände physisch mit dem Objekt verbunden sind, führen sie entsprechend zu Unsicherheiten in der räumlichen Dimension. Dies ist somit auch zumeist der Grund für die Verschlech-



(a) Schätzungsraum Aufname 23

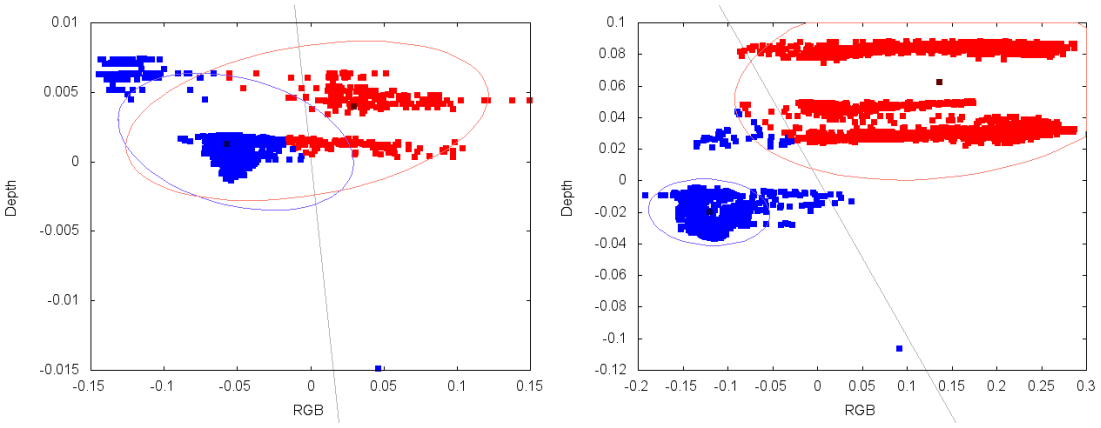
(b) Schätzungsraum Aufname 24

Abbildung 4.8: Diese Grafiken zeigen den Schätzungsraum zweier aufeinander folgender Messungen aus Testsequenz 19, eingefärbt nach Zugehörigkeit zu den Prototypen. Zwischen den Prototypen liegt eine kleine Anhäufung unsicherer Schätzungsvektoren, die durch Änderung zwischen den Aufnahmen in (a) der Umgebung, in (b) jedoch dem Objekt zugeordnet wird.

terung der Erkennungsraten, bei Verwendung der farblichen und räumlichen Informationen gegenüber der Verwendung der ausschließlich farblichen Informationen. Da das Objekt nahezu in jeder Testsequenz auf diese Weise mit einem anderen Gegenstand verbunden ist, fallen viele der Testsequenzen entsprechend fehlerhaft aus.

Das Auftreten solcher Anhäufungen von unsicheren Schätzungsvektoren ist in jeder Testsequenz anders und kann zudem auch über die Testdurchläufe hinweg variieren. Dennoch lassen sich einige häufiger auftretende Ausprägungen erkennen. Am häufigsten ist die Ausprägung einer weiteren, kleinen Anhäufung, die in der farblichen Dimension gut vom Objekt getrennt ist, in der räumlichen Dimension dem Objekt jedoch sehr nahe liegt. Durch die Initialisierung ist diese für gewöhnlich der Umgebung zugeordnet. Der Umgebungs-Prototyp ist dadurch in der räumlichen Dimension weit ausgedehnt und schließt die kleine Anhäufung mit ein. Treten jedoch Änderungen in der räumlichen Dimension ein, passt sich die Umgebung meist stärker an, als das Objekt. Abbildung 4.8 zeigt dies beispielhaft anhand des Schätzungsraums zweier aufeinander folgender Messungen aus Testsequenz 09. Durch eine solche Anpassung liegt die Anhäufung unsicherer Schätzungsvektoren danach häufig näher am Objekt als an der Umgebung und fließt somit ins Objekt ein.

Stellenweise sind die durch die Unsicherheiten enstehenden Anhäufungen ähnlich stark ausgeprägt, wie jene, die durch das Objekt oder die Umgebung entstehen. Da es im Schätzungs-



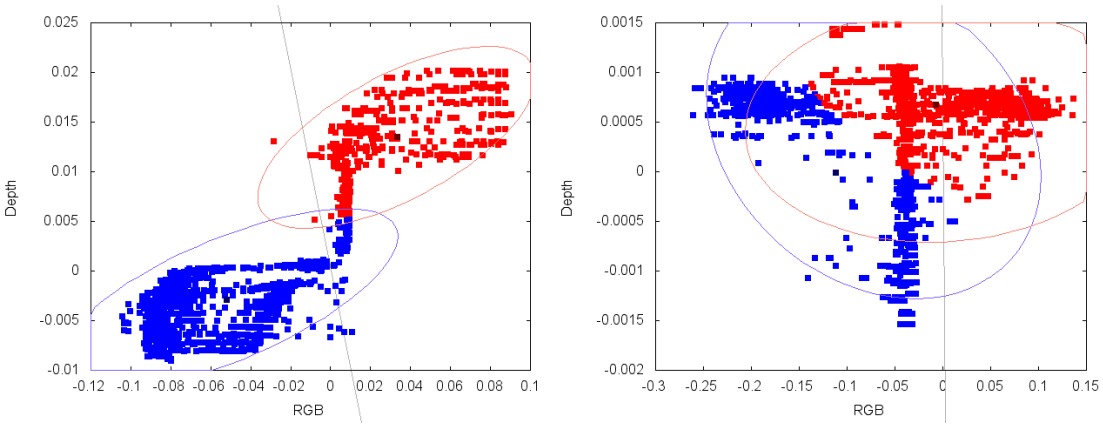
(a) Testsequenz 08 **(b)** Testsequenz 20

Abbildung 4.9: Die Diagramme zeigen jeweils den Schätzungsraum aus Testsequenz 08 (a) und 20 (b), eingefärbt nach Zugehörigkeit zu den Prototypen. Zu erkennen sind mehrere getrennte Anhäufungen, die in Objekt und Umgebung aufgeteilt werden.

raum dadurch mehr als zwei große Anhäufungen gibt, führt dies meist zu Fehlern. MNG muss die Schätzungsvektoren aufgrund der Vorgabe von zwei Prototypen entsprechend auf diese zwei Prototypen aufteilen, auch wenn tatsächlich mehr Prototypen für eine korrekte Teilung benötigt würden. Somit werden Anhäufungen teilweise oder ganz einem, möglicherweise falschen, Prototypen zugeordnet. Abbildung 4.9 zeigt zwei solcher Fälle, bei denen sich durch den das Objekt haltenden Arm (a) und die das Objekt haltende Person (b) eine oder mehr zusätzliche Anhäufungen ergeben, die zu fehlerhaften Klassifikationen führen.

Anhäufungen unsicherer Schätzungsvektoren gehen häufig auch fließend in das Objekt oder die Umgebung über. In solchen Fällen werden sie durch MNG dem jeweiligen Prototypen zugeordnet. Da diese Unsicherheiten jedoch meist durch Übergänge von der Umgebung zum Objekt entstehen, ist eine absolute Zuteilung einer solchen Anhäufung oft falsch. Teils verbindet eine Anhäufung von unsicheren Schätzungsvektoren so auch beide Anhäufungen des Objekts und der Umgebung. Dies führt häufig zu einer einzelnen, nicht mehr separierbaren, Anhäufung. In solchen Fällen unterteilt MNG diese Anhäufung in zwei Klassen, wie es in Abbildung 4.10 zu sehen ist. Aufgrund der fließenden Übergänge ergeben sich in solchen Situationen häufig Fehler.

Weiterhin kann es zu fehlerhaften Zuweisungen von unsicheren Schätzungsvektoren kommen, wenn vorher unbekannte beziehungsweise nicht im betrachteten Abschnitt liegende Gegenstän-



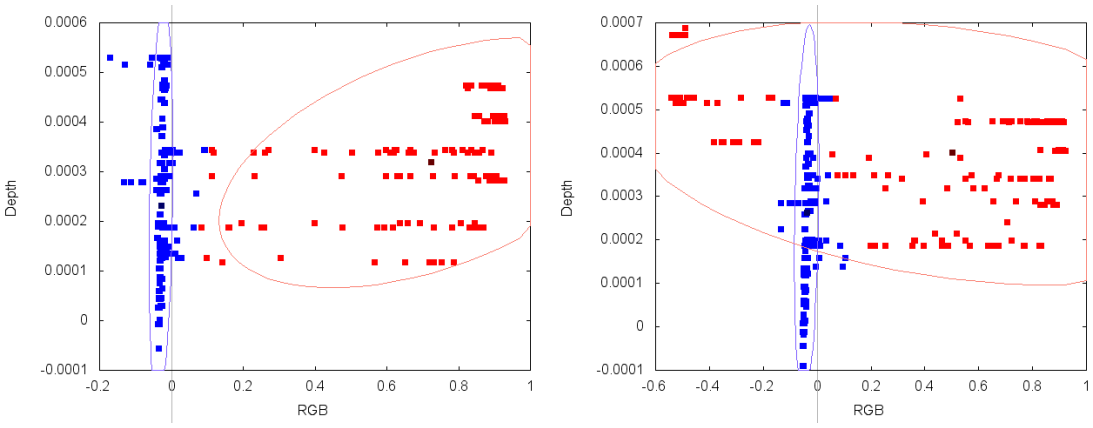
(a) Testsequenz 05 **(b)** Testsequenz 29

Abbildung 4.10: Die Diagramme zeigen jeweils den Schätzungsraum aus Testsequenz 05 (a) und 29 (b), eingefärbt nach Zugehörigkeit zu den Prototypen. Zu erkennen sind die mittig zwischen den Prototypen des Objektes und der Umgebung angeordneten unsicheren Schätzungsvektoren und wie diese zugeteilt werden.

de in die Nachbarschaft des Objektes eintreten. Weichen die Werte der neuen Schätzungsvektoren stark von den bisher betrachteten ab, kann dies zu Fehlern führen, da diese nicht während der Erstellung der für die Mahalanobis-Distanz verwendeten Matrix berücksichtigt wurden. Ein solcher Fall ist in Abbildung 4.11 zu sehen. Dieser neu in den betrachteten Bereich eingetretene Gegenstand weist einen starken farblichen Unterschied zum Objekt auf. Dennoch wird er aufgrund der Ausdehnung des Prototypen dem Objekt zugeordnet.

Zudem können durch neue Schätzungsvektoren auch stets die vorab beschriebenen Fehler auftreten. Da diese jedoch spontan entstehen, können sie nicht durch eine Initialisierung kompensiert werden. Somit sind neu in den betrachteten Bereich eintretende Gegenstände häufig eine Fehlerquelle.

Die Ursache dieser Fehler liegt im verwendeten Eingangsraum-Modell und der Unfähigkeit der verwendeten Verfahren, diesen Fehler zu kompensieren. Die Klassen werden in den jeweiligen Eingangsräumen durch eine Verteilung abgebildet. Dafür werden die gesamten Messdaten berücksichtigt, um ein Modell zu erhalten, das es erlaubt für jeden Eingangsvektor eine Schätzung über dessen Zugehörigkeit zu bestimmen. Da alle Messungen, die nicht dem Objekt zugeordnet werden, der Umgebung angehören, ist diese für gewöhnlich sehr inhomogen. Somit können innerhalb der Umgebung bereits große Distanzunterschiede entstehen. Weiterhin kann durch das Auftreten dieser großen Distanzen die Distanz zum Objekt unbedeutend werden.



(a) Schätzungsraum Messung 28

(b) Schätzungsraum Messung 29

Abbildung 4.11: Die Diagramme zeigen den nach Klassifikation eingefärbten Schätzungsraum zweier aufeinander folgender Messungen aus Testsequenz 10. Mit Messung 29 ist ein neuer Gegenstand in den betrachteten Bereich eingetreten, der dem Objekt zugeordnet wird.

Dies spiegelt sich in den Schätzungen der schwachen Klassifikatoren wider. Da MNG Verteilungen verwendet, um die Klassen zu repräsentieren und dafür die Distanzen betrachtet, ist MNG ebenfalls nicht in der Lage, diesen Umstand aufzulösen und die Schätzungsvektoren korrekt zu trennen.

4.8 Laufzeitanalyse

Das Verfahren wurde im Ausblick auf eine spätere Optimierung der Laufzeit hin entwickelt und macht die zur Verfolgung von Objekten benötigte Laufzeit zu einer relevanten Eigenschaft des Verfahrens. Daher wurde die Laufzeit des Verfahrens unter Verwendung der vorab ermittelten Konfiguration gemessen. Die Möglichkeiten der Optimierung wurden dabei nicht ausgeschöpft, da die grundsätzliche Funktionalität des Verfahrens während der Entwicklung vorrangig war. Dennoch lassen die Testergebnisse eine Einschätzung über die Größenordnung der benötigten Laufzeit zu. Ausgeführt wurden diese Tests auf einem handelsüblichen PC mit einer Intel i5-2500K CPU mit einer Taktfrequenz von 3.3GHz und 8GB RAM. Das Verfahren ist nicht parallelisiert, somit wurde in den Tests nur ein Kern zur Ausführung verwendet.

Gemessen wurden die Ausführungszeiten der anpassenden und zur Verfolgung genutzten Schritte, um ein Bild davon zu erhalten, wie sich das Verfahren bei der Verfolgung eines Objek-

tes verhält. Je verarbeiteter Messung wurden drei Zeiten erhoben: Die von MNG benötigte Zeit zur Anpassung der Prototypen, die zur Klassifikation der Eingangsvektor-Gruppen auf Basis der Prototypen benötigte Zeit sowie die zur Aktualisierung der Eingangsraum-Modelle benötigte Zeit. Aus der Summe dieser drei Messungen ergibt sich die Gesamtlaufzeit des Verfahrens zur Verfolgung eines Objektes. Nicht berücksichtigt wurden entsprechende zusätzlich zur eigentlichen Verarbeitung anfallende Zeiten, wie das Einlesen der Bilder, die Ausgabe von Daten oder das Schreiben von Ergebnissen auf die Festplatte.

Die Testergebnisse sind in Abbildung 4.12 als Balkendiagramme aufgeführt. Die Diagramme wurden über die durchschnittlichen Laufzeiten der Testsequenzen gebildet. In den oberen Diagrammen sind die Testergebnisse unter Betrachtung aller Testsequenzen abgebildet. Es ist zu erkennen, dass die durchschnittliche Ausführungszeit zur Verarbeitung einer Messung über alle Testsequenzen hinweg 145ms beträgt, jedoch starke Abweichungen aufweist. Ebenfalls sind starke Abweichungen in der zur Klassifikation benötigten Zeit zu erkennen, welche im Durchschnitt jedoch 55ms beträgt. Dies lässt sich unter anderem auf die fehlerhaften Testdurchläufe zurückführen. Im Fehlerfall umfasst die Klasse des Objektes meist mehr Messungen als es eigentlich der Fall sein sollte, wodurch die betrachtete Nachbarschaft größer ausfällt. Dies führt zu einer erhöhten Laufzeit von MNG und der Klassifikation. Nahezu konstant ist jedoch die zur Aktualisierung der Eingangsraum-Modelle benötigte Zeit mit 84ms. Da in diesem Schritt immer alle Eingangsvektoren betrachtet werden, hat die Größe des Objektes keinen Einfluss auf die Laufzeit. Die Gesamtlaufzeit beträgt im Durchschnitt 284ms, kann aber ebenfalls stark abweichen.

Da die fehlerhaften Testsequenzen die zur Verfolgung benötigte Laufzeit verfälschen, zeigen die unteren Diagramme in Abbildung 4.12 die von fehlerhaften Testsequenzen bereinigten Werte. Entsprechend geringer fallen die Laufzeiten und Abweichungen aus. Im Durchschnitt werden 87ms für den MNG-Schritt und 29ms für die Klassifizierung einer Messung benötigt. Die Aktualisierung der Eingangsraum-Modelle benötigt weiterhin fast konstant 83ms. Analog verringert sich die Gesamtlaufzeit und deren Abweichung. So beträgt die Gesamtlaufzeit unter ausschließlicher Betrachtung der erfolgreichen Testsequenzen 199ms.

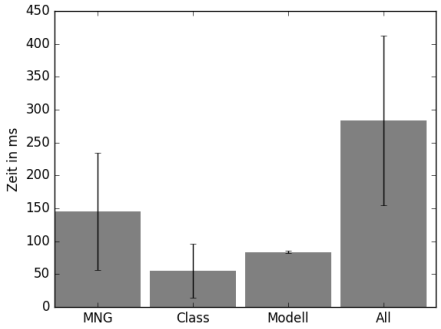
Da die Laufzeiten mit der Anzahl der zu verarbeitenden Daten schwankt, sind rechts in Abbildung 4.12 die zur Anzahl der Eingangs- beziehungsweise Schätzungsvektoren normalisierten Zeiten aufgeführt. Die Laufzeiten von MNG- und Klassifizierungsschritt sind jeweils zur Anzahl der in der Nachbarschaft enthaltenen Schätzungsvektoren normalisiert. Die zur Aktualisierung der Eingangsraum-Modelle benötigte Zeit ist zur Anzahl der Eingangsvektor-Gruppen

normalisiert. Trotz der Normalisierung sind weiterhin Unterschiede in den Ausführungszeiten der Ergebnisse aller Testsequenzen und denen der erfolgreichen Testsequenzen zu erkennen. Die Laufzeit des normalisierten MNG-Schrittes steigt bei ausschließlicher Betrachtung erfolgreicher Testsequenzen von 21μs auf 25μs. Ein Großteil des Aufwandes von MNG hängt von der Anzahl der betrachteten Schätzungsvektoren ab. Allerdings müssen einige Operationen unabhängig von der Anzahl der Schätzungsvektoren immer durchgeführt werden. Somit teilt sich die Laufzeit von MNG in einen konstanten und einen variablen Teil. Da im Falle einer erfolgreichen Testsequenz weniger Schätzungsvektoren betrachtet werden, steigt der Aufwand des konstanten Anteils pro Schätzungsvektor entsprechend an. Dies führt zu der erhöhten normalisierten Laufzeit gegenüber der Verwendung aller Testsequenzen. Gleich fällt hingegen die Laufzeit der Klassifizierung aus, pro Schätzungsvektor werden 7μs benötigt. Anders als im Falle von MNG ist der konstante Anteil am Aufwand der Klassifizierung zu gering, um die Werte merklich zu beeinflussen. Die Aktualisierung der Eingangsraum-Modelle benötigt ebenfalls eine konstante Zeit. Dies ist auf die sich nicht ändernde Anzahl an verarbeiteten Eingangsvektoren zurückzuführen.

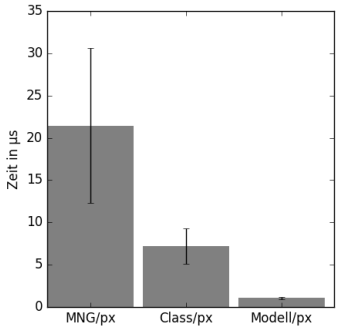
Die betrachteten Ausführungszeiten zeigen, dass die Laufzeit des Verfahrens zum Großteil vom MNG-Schritt abhängt, jedoch auch die Aktualisierung der Eingangsraum-Modelle einen erheblichen Teil der Zeit benötigt. Zudem ist das Verfahren mit den im Erfolgsfall durchschnittlich benötigten 199ms noch weit entfernt von einer Echtzeitverarbeitung der mit 30 Bildern pro Sekunde aufgenommenen Messungen der Kinect V2. Echtzeit bedeutet in diesem Fall, eine Messung zwischen der Aufnahme zweier Messungen verarbeiten zu können. Es ist unwahrscheinlich, dass die weitere Optimierung des Quellcodes die Laufzeit weit genug reduziert, um die Verarbeitung der Kinect V2 Messungen in Echtzeit zu erlauben. Allerdings wurde das Verfahren in Hinblick auf eine mögliche Parallelisierung entworfen. Eine optimierte Umsetzung für den Einsatz des Verfahrens auf einer Grafikkarte könnte die benötigte Beschleunigung erbringen.

4.9 Zusammenfassung der Testergebnisse

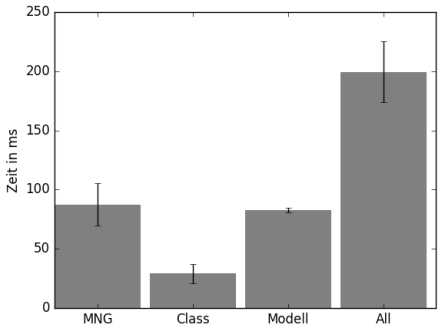
In diesem Kapitel wurde das im Kapitel 3 entwickelte Verfahren auf sein Verhalten bezüglich unterschiedlicher Konfigurationen hin getestet. Dabei wurde eine allgemein einsetzbare Parameterbelegung ermittelt und die in Kapitel 3 getroffenen Entscheidungen verifiziert.



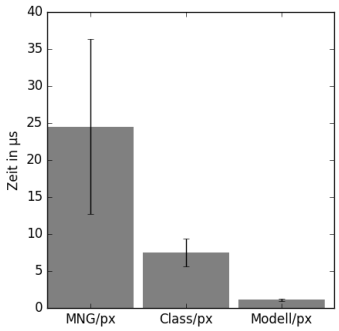
(a) Durchschnitt über alle



(b) Zeit pro Pixel über alle



(c) Durchschnitt über Erfolge



(d) Zeit pro Pixel über Erfolge

Abbildung 4.12: Diese Diagramme zeigen die durchschnittliche Ausführungszeit der Testsequenzen aus Testdurchlauf 7.4 (links) und die zur Anzahl der verarbeiteten Daten normaliserten Zeiten (rechts). Diagramme (a) und (b) zeigen die Werte aller Testsequenzen, während (c) und (d) lediglich die erfolgreichen Sequenzen einbeziehen. Die Balken zeigen die Zeiten der verschiedenen Abschnitte des Verfahrens. 'MNG' entspricht der von MNG benötigten Zeit, 'Class' die zur Neuklassifizierung der Daten, 'Modell' ist die zur Aktualisierung der Eingangsraum-Modelle benötigte Zeit und 'All' entspricht der Gesamtlaufzeit und somit der Summe der drei anderen. 'MNG/px' und 'Class/px' sind die jeweils zur Anzahl der betrachteten Schätzungsvektoren in der Nachbarschaft des Objektes normalisierten Zeiten, während 'Modell/px' die zur Anzahl der gesamten Eingangsvektoren normalisierte Zeit zeigt.

Zur Findung einer geeigneten Konfiguration wurden unterschiedliche Parameterbelegungen überprüft: Es wurde untersucht, welche Auswirkung der Einsatz der vorgeschlagenen Gewichtung und der Nachbarschaft auf die Erkennungsraten hat. Zudem wurde das Verhalten des Verfahrens unter verschiedenen Größen der Nachbarschaft überprüft. Ebenfalls untersucht wurden die Auswirkung der von MNG durchgeführten Iterationen zur Initialisierung sowie der zur Anpassung der Prototypen durchgeführten Iterationen.

Dabei erbrachte die Nutzung der Gewichtung der einzelnen Klassen auf Basis der zugehörigen Vektoren nicht das erwartete Ergebnis und führte vermehrt zu fehlerhaften Klassifikationen. Die Verwendung einer Nachbarschaft mit einer mittleren Größe hingegen konnte die Erkennungsraten positiv beeinflussen. Die Untersuchung der Iterationen von MNG zeigte, dass eine Initialisierung durch viele Iterationen minimal bessere Ergebnisse lieferte als eine auf wenigen Iterationen basierende Initialisierung. Somit kann, falls notwendig, die Initialisierung auf Kosten der Robustheit beschleunigt werden. Bei der Anpassung der Prototypen lieferten weniger Iterationen ein besseres Ergebnis, da diese die Initialisierung durch die Ergebnisse der vorherigen Messung nur lokal anpassen und so weniger stark auf Störungen reagieren. Weiterhin genügt eine mittlere Anzahl an Basisiterationen von MNG, um das Verfahren zu initialisieren und ein Objekt zu verfolgen.

Nach der Bestimmung einer geeigneten Konfiguration wurde das verwendete Distanzmaß untersucht. Dazu wurde betrachtet, ob die Mahalanobis-Distanz durch die dynamische Anpassung an die zugrundeliegenden Messungen dem Verfahren den erwarteten Vorteil gegenüber dem Einsatz des Euklidischen Abstands brachte. Dabei erbrachten die schwachen Klassifikatoren deutlich bessere Schätzungen unter Verwendung der Mahalanobis-Distanz. Zudem führte die Verwendung des Euklidischen Abstands durch MNG dazu, dass einzelne Eingangsräume für die Klassifikation unberücksichtigt bleiben. Somit ist die Mahalanobis-Distanz in beiden Fällen dem Euklidischen Abstand vorzuziehen.

Abschließend wurde betrachtet, wie sich die Kombination verschiedener Eingabedaten auf die Ergebnisse des Verfahrens auswirkt. Es stellte sich heraus, dass die Verwendung eines Eingangsraumes, der gut an die Gegebenheiten angepasst ist, bessere Ergebnisse liefern kann als die Kombination mehrerer Eingangsräume durch das Verfahren. Da sich jedoch ohne Vorwissen nicht entscheiden lässt, welcher Eingangsraum die Verfolgung eines Objektes erlaubt, ist dies keine allgemein verwendbare Option. Das Verfahren hingegen konnte mehrere Eingangsräume kombinieren, wodurch sich Fehler ausgleichen ließen. Allerdings bleiben die Erkennungsraten dabei hinter denen eines einzelnen, gut angepassten Eingangsraumes zurück.

Da die Erkennungsraten über alle Testdurchläufe hinweg schlechter als erwartet waren, wurde das Verfahren genauer auf sein Fehlerverhalten hin begutachtet und es konnten einige häufig auftretende Fehlerfälle erkannt werden. Es zeigte sich eine Anfälligkeit des Verfahrens gegenüber falschen Klassifikationen. Die fehlerhafte Zuweisungen von Schätzungsvektoren kann zu einer Änderung der Prototypen führen, welche wiederum weitere fehlerhafte Zuweisungen mit sich führt. Somit kann sich ein Fehler selbst verstärken und die Verfolgung eines Objektes stören. Zudem konnte ein Ursprung für unsichere Schätzungsvektoren ermittelt werden. Aufgrund des verwendeten Eingangsraum-Modells können innerhalb einer Klasse größere Distanzen entstehen als zwischen den Klassen. Dies wird durch die schwachen Klassifikatoren in den Schätzungsraum abgebildet. Weil MNG nicht in der Lage ist diesen Fehler zu kompensieren, entstehen so fehlerhafte Klassifikationen. Da die verwendeten räumlichen Informationen der Testsequenzen diesen Fehler begünstigen, konnte nur in wenigen Fällen das jeweilige Objekt verfolgt werden.

Zudem wurden die Ausführungszeiten des Verfahrens erhoben und besprochen. Dabei wurde ausschließlich die zur Verfolgung eines Objektes benötigte Zeit betrachtet. Als ausschlaggebend für die Gesamtlaufzeit des Verfahrens wurden sowohl MNG als auch die Aktualisierung der Eingangsraum-Modelle ermittelt. Die Laufzeit von MNG hängt dabei von der Größe der betrachteten Nachbarschaft ab und variiert entsprechend stark. Hingegen ist die zur Aktualisierung der Eingangsraum-Modelle benötigte Laufzeit abhängig von der Menge der Eingangsvektor-Gruppen, weswegen diese kaum variiert. Die Ausführungszeiten des Verfahrens liegen um ein Vielfaches höher, als es für die Objekterkennung in Echtzeit auf Basis der Kinect V2 nötig ist. Jedoch wird davon ausgegangen, dass sich das Verfahren durch eine Übersetzung auf eine GPU entsprechend beschleunigen lässt.

5 Zusammenfassung und Ausblick

Dieses Kapitel schließt die Arbeit ab. In Abschnitt 5.1 wird ein zusammenfassender Rückblick über die Arbeit gegeben. Anschließend wird in Abschnitt 5.2 ein Blick auf mögliche zukünftige Verbesserungen und Erweiterungen des Verfahrens geworfen, die nicht mehr im Rahmen dieser Arbeit umgesetzt werden konnten.

5.1 Zusammenfassung

Ziel der Arbeit war es, einen neuen Ansatz zur Erkennung und Verfolgung von allgemeinen, deformierbaren Objekten in aufeinander folgenden Bildern zu entwickeln und zu untersuchen. Dieser sollte die eingegebenen Daten in Objekt und Umgebung unterteilen. Dazu wurden Messungen mehrerer unterschiedlicher Sensoren kombiniert, um die Eigenschaften des Objektes in mehreren Dimensionen erfassen zu können. Stellvertretend für eine Vielzahl an möglichen Eingaben wurden die Daten der Kinect V2 verwendet, um das Verfahren zu untersuchen. Zudem wurde das Verfahren in Hinblick auf eine mögliche Laufzeitoptimierung entworfen.

Um das Ziel der Arbeit umzusetzen, wurde ein mehrstufiger Ansatz gewählt. Für jeden Eingangsraum wird innerhalb dessen eine Schätzung über die Zugehörigkeit jedes Eingangsvektors erstellt. Die Einschätzungen aller Eingangsräume werden dann zur finalen Klassifikation genutzt. Dies reduziert die zu betrachtenden Dimensionen bei der Zusammenführung der Eingangsräume und ermöglicht den Vergleich von Daten mit unterschiedlichen Einheiten. Dazu werden als Modell des Objektes und der Umgebung im jeweiligen Eingangsraum einfache Verteilungen verwendet. Der einfache Aufbau der Modelle erlaubt eine effiziente Nutzung, die sich in einer geringen Laufzeit niederschlägt. Die Zugehörigkeit eines Eingangsvektors wird auf Basis seiner Distanz zu den beiden Verteilungen bestimmt. Dazu wird die Mahalanobis-Distanz eingesetzt, da diese eine Anpassung an die zugrundelegenden Ausprägungen der Daten erlaubt. Die so erzeugten Schätzungen werden in einem gemeinsamen Schätzungsraum in Form von Schätzungsvektoren zusammengefasst. Mittels MNG werden diese in Objekt und Umgebung unterteilt. Objekt und Umgebung werden dabei durch jeweils einen Prototypen im Schätzungs-

raum dargestellt. Zur Verfolgung eines Objektes werden die Prototypen der vorherigen Messung durch MNG angepasst. Da große Unterschiede in der Anzahl der zu einer Klasse gehörenden Schätzungsvektoren zu Fehlern führen, wird der Einsatz von MNG auf die Nachbarschaft des Objektes begrenzt. Durch Verwendung der Nachbarschaft wird zudem die Anzahl zu betrachtender Schätzungsvektoren und somit die Laufzeit des Verfahrens reduziert. Die finale Klassifikation geschieht auf Basis der von MNG erzeugten Prototypen.

Um zu überprüfen, ob die für das Verfahren verwendeten Mechanismen das erwartete Ergebnis liefern, wurden anschließend entsprechende Tests durchgeführt. Dabei wurde das Verfahren für jeden Testdurchlauf auf 35 Testsequenzen angewandt. Diese setzen sich aus fünf künstlich generierten und 30 mit der Kinect V2 aufgenommenen Testsequenzen zusammen. Als erstes wurden Tests zur Ermittlung einer allgemein einsetzbaren Konfiguration des Verfahrens durchgeführt, um diese in den darauf folgenden Tests einsetzen zu können. Es zeigte sich, dass die Eingrenzung der von MNG verarbeiteten Schätzungsvektoren auf die Nachbarschaft des Objektes die andernfalls auftretenden Fehler vermeiden kann. Eine Nachbarschaft von mittlerer Größe erlaubt dabei größere Bewegungen des Objektes zwischen zwei Messungen, liefert aber gleichzeitig eine ausreichend zutreffende Klassifizierung. In den Tests reichte eine geringe Anzahl an MNG-Iterationen während der Initialisierung aus, um Objekte zu verfolgen Jedoch lieferte eine ausführliche Initialisierung bessere Ergebnisse. Zudem genügte in den Testfällen eine geringe Anzahl an MNG-Iterationen zur Verfolgung des Objektes, da diese weniger stark von globalen Einflüssen betroffen waren. Jedoch führte die Verwendung einer geringeren Basis an Iterationen während der Tests zu schlechteren Erkennungsraten, da die Anpassung der Prototypen in größeren und somit gröberen Sprüngen geschieht. Dennoch zeigte sich oberhalb einer mittleren Anzahl an Iterationen keine Änderung mehr.

Nach der Bestimmung der Konfiguration wurde überprüft, ob die Mahalanobis-Distanz den erwarteten Mehrwert gegenüber der Verwendung des Euklidischen Abstands erbringt. Die Tests zeigten, dass die Mahalanobis-Distanz aufgrund ihrer Anpassungsfähigkeit dem Euklidischen Abstand vorzuziehen ist. Die Schätzungen der schwachen Klassifikatoren waren unter Verwendung des Euklidischen Abstands nicht durch MNG in Objekt und Umgebung zu trennen. Die Verwendung der Mahalanobis-Distanz führte hingegen zu unterscheidbaren Schätzungen. Da die Schätzungen der einzelnen Eingangsräume in unterschiedlichen Wertebereichen liegen können, lieferte die Verwendung der Mahalanobis-Distanzmaß während MNG ebenfalls bessere Ergebnisse. Durch Einsatz des Euklidischen Abstands flossen schwach ausgeprägte Dimensionen kaum für die Klassifizierung ein.

In einem letzten Test wurde das Verhalten des Verfahrens unter Verwendung einzelner und mehrerer Eingangsräume untersucht, um zu überprüfen, ob das Verfahren Unsicherheiten durch Kombination mehrerer Eingangsräume ausgleichen kann. Um mehr als die zwei gelieferten Eingangsräume aus den Kinect-Daten zu generieren, wurden auf Basis der RGB-Bilder zusätzlich HSV-Bilder erzeugt. Es zeigte sich, dass Unsicherheiten in einem Eingangsraum durch einen anderen kompensiert werden können. Jedoch fiel in den Tests das Ergebnis der Kombination zweier Eingangsräume schlechter aus, als die ausschließlich auf dem besseren der beiden durchgeführte Klassifikation. Da jedoch im Vorfeld keine Annahmen über das Objekt gemacht werden können und es somit nicht zu entscheiden ist, welche Eingabedaten gute und welche schlechte Ergebnisse liefern, kann das Verfahren genutzt werden, um einen Ausgleich über mehrere Eingangsräume herzustellen.

In einer anschließenden Analyse wurde nach dem Ursprung für die Fehlklassifikationen durch das Verfahren gesucht. Dazu wurden die in vielen Testdurchläufen zu Fehlern führenden Testsequenzen genauer betrachtet. Dabei konnte ein häufiges Auftreten von gegensätzlichen Einschätzungen durch die schwachen Klassifikatoren festgestellt werden, was zu einer hohen Unsicherheit führte. Entsprechend wurden Schätzungsvektoren in vielen Fällen dem falschen Prototypen zugeordnet. Oft orientierte sich das Objekt durch fehlerhafte Zuordnungen näher zur Umgebung, was weitere fehlerhafte Zuordnungen begünstigte. Somit zeigte sich, dass das Verfahren sehr anfällig für diese Art von sich selbst verstärkendem Fehler ist. Die Ursache für die zu solchen Fehlern führenden Unsicherheiten wurde in den Eingangsraum-Modellen gefunden. Durch Verwendung von nur einer Verteilung zur Repräsentation der inhomogenen Umgebung können Distanzen innerhalb der Umgebung größer ausfallen als die Distanzen zwischen Objekt und Umgebung. Die schwachen Klassifikatoren bilden diese Distanzen durch entsprechende Schätzungen direkt in den Schätzungsraum ab. Da MNG ebenfalls nicht dazu geeignet ist, diese Fehler zu kompensieren, führt dies zu fehlerhaften Klassifikationen.

Ein weiteres Problem des Verfahrens konnte ebenfalls erfasst werden. Durch den Eintritt neuer Gegenstände in die betrachtete Nachbarschaft des Objektes können ebenfalls fehlerhafte Zuordnungen von Schätzungsvektoren entstehen. Die durch einen solchen Gegenstand erzeugten Schätzungen können außerhalb des bisher betrachteten Wertebereichs liegen. Da die Mahalanobis-Distanz jedoch nur den bisher bekannten Wertebereich zur Anpassung nutzt, werden solche neuen Schätzungsvektoren häufig falsch zugeordnet.

Zuletzt wurden die Ausführungszeiten der Implementierung des Verfahrens betrachtet. Es zeigte sich, dass die Implementierung die Messungen nicht ausreichend schnell klassifizieren

kann, um die Daten der Kinect V2 in Echtzeit zu verarbeiten. Allerdings sind die Laufzeiten niedrig genug, als dass eine Parallelisierung und Portierung des Verfahrens auf eine GPU das Verfahren ausreichend beschleunigen könnte.

Zusammenfassend ist das in dieser Arbeit entwickelte Verfahren also in der Lage, mehrere Eingangsräume zu kombinieren, um über diese eine gemeinsame Klassifikation zu erstellen und ein Objekt somit zu verfolgen. Unsicherheiten einzelner Eingangsräume können dabei durch andere Eingangsräume zu einem gewissen Teil kompensiert werden. Die Erkennung ist dabei jedoch schlechter als es auf Basis des besten an der Erkennung beteiligten Eingangsraumes alleinemöglich wäre. Da es jedoch nicht möglich ist, diesen vorab zu bestimmen, bietet das Verfahren eine Methode, Unsicherheiten auszugleichen. Weiterhin ist die benötigte Ausführungszeit zwar zu hoch für eine Verfolgung von Objekten über die Kinect V2 in Echtzeit. Das Verfahren ist allerdings schnell genug, als dass weitere Optimierungen dies ermöglichen könnte.

5.2 Ausblick

Das Verfahren besitzt einige Schwachstellen, welche die Erkennung und Verfolgung eines Objektes beeinflussen. Dennoch ist es grundsätzlich in der Lage, Objekte zu erkennen. Es sind verschiedene Ansätze zur Behebung dieser Schwachstellen vorstellbar.

Ein Problem sind die von den schwachen Klassifikatoren nicht ausreichend getrennten Objekt- und Umgebungs-Schätzungsvektoren, die MNG ebenfalls nicht zu trennen vermag. Da das Problem jedoch vornehmlich durch die Art und Weise der schwachen Klassifikatoren ausgelöst wird, liegt es nahe, bereits in diesem Schritt einzugreifen. Eine Lösung könnte der Einsatz eines anderen Modells zur Repräsentation des Objektes und der Umgebung sein. Da die Umgebung alles beinhaltet, was nicht dem Objekt entspricht, ist sie in der Regel sehr inhomogen. Durch die Verwendung nur einer Verteilung zur Repräsentation der Umgebung können Distanzen innerhalb der Umgebung so größer ausfallen, als die Distanz zwischen Objekt und Umgebung. Daher wäre es denkbar, ein auf mehreren Verteilungen basierendes Modell, ähnlich einer MoG, zu verwenden, um die verschiedenen Eigenschaften der die Umgebung bildenden Gegenstände genauer abbilden zu können. Werden zur Distanzbestimmung die nächst gelegenen Verteilungen innerhalb dieses Modells bevorzugt betrachtet, würde die Umgebung besser widergespiegelt. Allerdings bedeutet dies einen erheblichen Mehraufwand für die Distanzbestimmung. Zudem müsste eine Methode zur Aktualisierung der Verteilungen gefunden werden, so dass die Verteilungen die Umgebung repräsentieren und sich effizient aktualisieren lassen. Eine Alternative

könnten auch Modelle sein, welche keine Verteilungen nutzen, sondern die Umgebung und das Objekt auf andere Weise repräsentieren. Angelehnt an [BVD11] könnten diese auf Stichproben basieren. In einem solchen Fall müsste ebenfalls erarbeitet werden, nach welchem Schema Stichproben ausgewählt und aktualisiert werden. Zudem müsste eine effiziente Methode definiert werden, die es erlaubt, die Zugehörigkeit zu bestimmen und Unsicherheiten auszudrücken.

Ebenfalls denkbar wäre es, die Funktionsweise der schwachen Klassifikatoren grundlegend zu ändern, sodass diese auf andere Weise eine Einschätzung der Zugehörigkeit zur Umgebung oder zum Objekt liefern. Somit könnten sie dazu in der Lage sein, die durch die Modelle entstehenden Probleme zu kompensieren.

Eine weitere Möglichkeit wäre es, die Zusammenführung zu ändern und somit MNG anzupassen oder zu ersetzen. In einigen Fällen könnte das Problem durch Einführen weiterer Prototypen behoben werden. Dabei müsste sichergestellt bleiben, dass der das Objekt repräsentierende Prototyp auch stets von den anderen zu unterscheiden ist, um das Objekt weiterhin verfolgen zu können. Die Anzahl der benötigten Prototypen kann sich jedoch von Messung zu Messung ändern, weshalb diese dynamisch ermittelt werden müsste. Dies legt nahe, ein anderes Verfahren als MNG zu verwenden, welches die Anzahl der Prototypen beziehungsweise Cluster entsprechend ermitteln und anpassen kann.

Ein anderer Ansatz, welcher auch das Problem neu entstandener Schätzungsvektoren mit starker Abweichung des bisher berücksichtigten Wertebereichs lösen könnte, wäre es MNG durch ein anderes Verfahren zu ersetzen, welches den Schätzungsraum nicht auf Basis von Verteilungen, sondern auf andere Weise trennt. Dies könnte angelehnt an SVM [MR05, S. 231ff] sein und lediglich den Bereich betrachten, in dem die Schätzungsvektoren der unterschiedlichen Klassen aufeinandertreffen. Ein solches Verfahren müsste weiterhin in der Lage sein, das zur Trennung genutzte Kriterium über die Zeit den sich ändernden Bedingungen anzupassen.

Neben der Verbesserung der Erkennung und Verfolgung sind zudem noch Ansätze zur Beschleunigung des Verfahrens denkbar. Wie in Abschnitt 3.6 erwähnt wird die von OpenCV vorgesehene Methode zum Zugriff auf die eingesetzten Container verwendet. Ebenso denkbar wäre es, direkt auf den Speicher zuzugreifen, was es potentiell ermöglicht, die Zugriffe besser an das Verfahren anzupassen und zu beschleunigen.

Die Aktualisierung der Eingangsraum-Modelle bietet einen weiteren Ansatzpunkt, um das Verfahren zu beschleunigen. Diese erfasst alle Eingangsvektoren gleichermaßen. Durch deren große Anzahl wächst entsprechend der Aufwand. Da die Umgebung zumeist einen großen Teil der Daten ausmacht, ist sie somit ausschlaggebend. Aufgrund der vielen Daten ist jedoch damit

zu rechnen, dass sie viel Redundanz enthält. Würde diese geringer abgetastet, könnte der zur Aktualisierung benötigte Aufwand reduziert werden, ohne die Eingangsraum-Modelle der Umgebung stark zu verändern. Wie in [BVD11] vorgeschlagen wäre dabei eine zufällige Auswahl der berücksichtigten Eingangsvektoren anstelle einer regelmäßigen Abtastung eine Möglichkeit, um auch regelmäßige Strukturen zu erfassen.

Die Parallelisierung des Verfahrens bietet eine weitere Methode, die Laufzeit zu reduzieren. Wie bereits in Abschnitt 3.4 erwähnt ist das Verfahren dafür geeignet, parallelisiert zu werden. Unter Verwendung einer Grafikkarte könnte sich so die Laufzeit des Verfahrens drastisch reduzieren lassen. Auf diese Weise ließe sich das Verfahren auch in zeitkritischen Umgebungen einsetzen, um Objekte zu erkennen und darauf zu reagieren.

Somit steckt trotz der niedrigen Erkennungsraten Potential in diesem Verfahren. Durch die vorgeschlagenen Anpassungen könnten die durch das Eingangsraum-Modell erzeugten Fehler kompensiert und die Laufzeit reduziert werden. Dann wäre ein vielseitiger Einsatz des Verfahrens vorstellbar, da das Erkennen von Objekten eine grundlegende Komponente in der Bilderkennung ist.

A DVD

Auf der beiliegenden DVD ist der Quellcode der Implementierung des in Kapitel 3 entwickelten Verfahrens enthalten. Details wie dieser zu Verwenden ist sind in der `readme.txt` aufgeführt. Weiterhin befinden sich die für die Evaluation in Kapitel 4 genutzten Testsequenzen auf dem Datenträger. Diese liegen in ihrer skalierten Form vor, wie sie zur Durchführung der Tests verwendet wurden. Aus Platzgründen liegen nur beispielhaft die Testergebnisse von Testdurchlauf 7.4 bei, da dieser das Verfahren am besten repräsentiert. Zudem befindet sich eine digitale Version dieser Arbeit auf der DVD.

Literaturverzeichnis

[AHH11] Banchar Arnonkijpanich, Alexander Hasenfuss und Barbara Hammer. *Local matrix adaptation in topographic neural maps*. *Neurocomputing*, 74(4):522 – 539, 2011.

[BVD11] O. Barnich und M. Van Droogenbroeck. *ViBe: A Universal Background Subtraction Algorithm for Video Sequences*. *Image Processing, IEEE Transactions on*, 20(6):1709–1724, Juni 2011.

[CGMS16] Andrea Corti, Silvio Giancola, Giacomo Mainetti und Remo Sala. *A metrological characterization of the Kinect V2 time-of-flight camera*. *Robotics and Autonomous Systems*, 75, Part B:584 – 594, 2016.

[CS14] Massimo Camplani und Luis Salgado. *Background foreground segmentation with RGB-D Kinect data: An efficient combination of classifiers*. *Journal of Visual Communication and Image Representation*, 25(1):122 – 136, 2014. Visual Understanding and Applications with RGB-D Cameras.

[HLI+10] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua und N. Navab. *Dominant orientation templates for real-time detection of texture-less objects*. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, Seiten 2257–2264, Juni 2010.

[JK11] Andreas Jordt und Reinhard Koch. *Fast Tracking of Deformable Objects in Depth and Colour Video*. In *BMVC*, Seiten 1–11, 2011.

[KDF+11] C.G. Keller, Thao Dang, H. Fritz, A. Joos, C. Rabe und D.M. Gavrila. *Active Pedestrian Safety by Automatic Braking and Evasive Steering*. *Intelligent Transportation Systems, IEEE Transactions on*, 12(4):1292–1304, Dezember 2011.

[KHRF11] Michael Krainin, Peter Henry, Xiaofeng Ren und Dieter Fox. *Manipulator and object tracking for in-hand 3D object modeling*. *The International Journal of Robotics Research*, 30(11):1311–1327, 2011.

[KLK14] Seongyong Koo, Dongheui Lee und Dong-Soo Kwon. *Incremental object learning and robust tracking of multiple objects from RGB-D point set data*. *Journal of Visual Communication and Image Representation*, 25(1):108 – 121, 2014. Visual Understanding and Applications with RGB-D Cameras.

[KVD06] S. Knoop, S. Vacek und R. Dillmann. *Sensor fusion for 3D human body tracking with an articulated 3D body model*. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, Seiten 1686–1691, Mai 2006.

[Mic] Microsoft. *Kinect tools and resources*. URL: `https://dev.windows.com/en-us/kinect/tools` [Stand: 27.01.2016].

[MR05] Oded Maimon und Lior Rokach. *Data mining and knowledge discovery handbook*, Band 2. Springer, 2005.

[OKA11] Iason Oikonomidis, Nikolaos Kyriazis und Antonis A Argyros. *Efficient model-based 3D tracking of hand articulations using Kinect.* In *BMVC*, Seiten 1–11, 2011.

[Ope] *OpenCV*. URL: `http://opencv.org/` [Stand: 27.01.2016].

[PLW11] Youngmin Park, V. Lepetit und Woontack Woo. *Texture-less object tracking with online training using an RGB-D camera*. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, Seiten 121–126, Oktober 2011.

[SA11] L. Spinello und K.O. Arras. *People detection in RGB-D data*. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, Seiten 3838–3843, September 2011.

[SA12] L. Spinello und K.O. Arras. *Leveraging RGB-D Data: Adaptive fusion and domain adaptation for object detection*. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, Seiten 4469–4474, Mai 2012.

[SG99] Chris Stauffer und W. E L Grimson. *Adaptive background mixture models for real-time tracking*. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, Band 2, Seiten 246–252, 1999.

[SSK+13] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook und Richard Moore. *Real-time Human Pose Recognition in Parts from Single Depth Images*. *Commun. ACM*, 56(1):116–124, Januar 2013.

[Sze11] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer London, 2011.

[Zac] Gabriel Zachmann. *NanoTimer*. URL: `http://cgvr.cs.uni-bremen.de/research/colldet/data/CollDetDoc/classcol_1_1_nano_timer.html` [Stand: 27.01.2016].