Visuelle Verfahren für ortsbezogene Dienste

Chadly Marouane

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

vorgelegt von
Chadly Marouane

Tag der Einreichung: 23. Januar 2017

Visuelle Verfahren für ortsbezogene Dienste

Chadly Marouane

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

vorgelegt von
Chadly Marouane

1. Berichterstatter:	Prof. Dr. Claudia Linnhoff-Popien
2. Berichterstatter:	Prof. Dr.-Ing. Klaus Wehrle
Tag der Einreichung:	23. Januar 2017
Tag der Disputation:	03.07.2017

Eidesstattliche Versicherung

(siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Chadly Marouane

Zusammenfassung

Durch die rasante Entwicklung und Verbreitung mobiler Endgeräte, wie z.B. Smartphones, Tablets und Wearables, ist die Anzahl ortsbezogener Dienste in den letzten Jahren enorm angestiegen. Die Positionsbestimmung des Nutzers ist für solche Dienste zwingend notwendig. Funkbasierte Verfahren wie zum Beispiel GPS oder WLAN ermöglichen die Lokalisierung des Nutzers. Trotz der hohen Genauigkeit und der stetigen Weiterentwicklung der Verfahren haben diese Technologien ihre Grenzen und stehen nicht immer zur Verfügung, weshalb andere Lösungen unterstützend oder als Alternative zum Einsatz kommen müssen. Optische Sensoren, speziell Kamerasensoren oder sogenannte *Actioncams*, bieten hierfür eine Alternative an. Die Kombination aus hochauflösenden Kameras und der Leistungsfähigkeit mobiler Endgeräte bietet die Möglichkeit Methoden zur Positionsbestimmung und Aktivitätserkennung zu realisieren. Obwohl bereits heute einige visuelle Verfahren für ortsbezogene Dienste genutzt werden, steckt die Entwicklung auf diesem Gebiet immer noch in den Kinderschuhen. Daher werden in der vorliegenden Arbeit neuartige und erweiterte visuelle Verfahren zur Positionsbestimmung und Aktivitätserkennung vorgestellt, welche rein auf der Kamera mobiler Endgeräte basierend umgesetzt werden. Dafür werden markante visuelle Merkmale aus Bildsequenzen extrahiert und für die Bestimmung der relativen und absoluten Position verwendet. Zusätzlich werden die selben markanten Merkmale zur Bestimmung der aktuellen Aktivität eines Nutzers genutzt.

In der vorliegenden Arbeit wird ein Verfahren zum effizienten Vergleich von Bildinformationen vorgestellt. Dazu wird ein Prozess entwickelt, der aus mehreren Vergleichsstufen besteht. Ähnlich wie bei einem Siebverfahren werden dabei stufenweise falsche Bilder aussortiert. Ziel ist es, ein Verfahren zu entwickeln, das fotografierte Bildsequenzen von Objekten oder ganzen Standorten in kürzester Zeit eindeutig wieder erkennen kann. Anschließend werden drei neue Erweiterungen für ein visuelles Positionierungssystem, namens *MoViPS*, vorgestellt. Diese Erweiterungen sollen das System für einen Echtzeitbetrieb nutzbar machen. Zwei Erweiterungen verkürzen allgemein die Antwortzeit und machen gleichzeitig den gesamten Prozess effizienter und robuster. Die dritte Erweiterung verbessert das bestehende Positionskorrekturverfahren aus *MoViPS*. Im letzten Teil dieser Arbeit wird ein Verfahren der visuellen Odometrie vorgestellt, das auf Basis eines Kamerasensors funktioniert, der aus der Ego-Perspektive einer Person aufzeichnet. Anders als bekannte State-of-the-Art-Verfahren basiert das Verfahren auf einem visuellen Schrittzähler und einem visuellen Kompass. Zusätzlich wird eine visuelle Aktivitätserkennung vorge-

stellt. Mit Hilfe der Eigenschaften markanter Merkmale, die aus einer Bildsequenz extrahiert werden, können Schritte sowie Aktivitäten erkannt werden. Die Beiträge der vorliegenden Arbeit liefern somit wichtige Grundbausteine für die Entwicklung und Erweiterung visueller Positionierungssysteme. Sie bieten zusätzlich visuelle Verfahren für ortsbezogene Dienste an, die die einfache Position und Aktivitäten eines Nutzers bestimmen können.

Abstract

Due to rapid development and wide distribution of mobile devices, such as Smartphones, tablets and wearables, the number of location-based services has increased enormously in recent years. For such services the position of a user is mandatory. Therefore wireless technologies and systems like e.g. GPS or WiFi are used for location determination. Despite high accuracy and continuous development these systems come with limitations like restricted availability. In order to compensate these restrictions, other technologies like optical sensors, especially camera sensors or so-called *Actioncams* are used as alternative or supportive systems. The combination of high-resolution cameras and the performance of mobile devices allow position determination and activity analysis. Although location-based services already make use of visual methods, development in this field is still in its infancy. This dissertation presents novel and advanced visual methods for localization determination and activity analysis based on mobile device cameras. Prominent visual features extracted from image sequences provide the basis for this kind of relative and absolute positioning and actvity determination. This work reveals a method for efficient comparison of image information. For this purpose a process is developed, which consists of several comparison steps to remove false images similary to a sieve filter. The aim is to develop a method that can clearly recognize photographed image sequences of objects or entire locations in a very short time. Subsequently three new extensions for a visual positioning system, called *MoVIPS*, are presented. Two of these extensions in general reduce response time and increase process efficiency and robustness. The third extension improves the existing positions correction method of *MoVIPS*. The last part of this work presents a camera sensor based method of visual odometry which records from first person perspective. Unlike known state-of-the-art methods, this method uses visual step counting and a visual compass. In addition, a visual activity detection is presented.
Using the properties of prominent visual features that are extracted from an image sequence, steps and activities can be detected. The contributions of this work provides important foundations for the development and expansion of visual positioning systems. They also offer a visual method for location-based services that can determine the simple position and activities of a user.

Inhaltsverzeichnis

Danksagung

Diese Dissertation ist während meiner Zeit am Lehrstuhl für Mobile und Verteilte Systeme an der Ludwig-Maximilians-Universität München sowie an der Virality GmbH entstanden. In dieser Zeit habe ich in unterschiedlichster Form Unterstützung und Motivation von verschiedenen Personen erfahren, denen ich an dieser Stelle gerne danken möchte. Mein ganz besonderer Dank gilt Frau Prof. Dr. Claudia Linnhoff-Popien, die mir an ihrem Lehrstuhl sowie in der Virality GmbH ein vielfältiges und inspirierendes Arbeitsumfeld geboten hat. Insbesondere möchte ich mich für das stets entgegengebrachte Vertrauen und die große Unterstützung bei der Verfolgung meiner Interessen und Ziele bedanken. Weiterhin gilt mein Dank Herrn Prof. Dr.-Ing. Klaus Wehrle für die Übernahme der Zweitberichterstattung und Herrn Prof. Dr. Heinrich Hußmann für sein Mitwirken als Vorsitzender der Prüfungskommission. Ein großes Dankeschön geht an alle meine Kollegen am Lehrstuhl sowie an mein Team in der Virality GmbH. Hervorheben möchte ich an dieser Stelle Andre Ebert, Sebastian Feld, Benno Rott, Lorenz Schauer und Matthias Schmidmaier, die mich auf meinem Wege zur Promotion tatkräftig unterstützt haben. Ganz besonderen Dank gilt Marco Maier, der mich für die wissenschaftliche Arbeit begeistern konnte und bei zahlreichen bis spät in die Nacht reichenden Diskussionen zur Seite stand. Mein größter Dank gilt jedoch meiner Familie. Insbesondere möchte ich meiner Mutter danken, die immer für mich da war und mich unentwegt unterstützt hat. Danke dir Mama.

1 Einleitung

Durch die rasante Entwicklung und Verbreitung mobiler Endgeräte, wie z.B. Smartphones, Tablets und Wearables, ist die Anzahl ortsbezogener Dienste enorm angestiegen. Viele Anwendungen ortsbezogener Dienste sind nicht mehr aus unserem Alltag wegzudenken. Sie sind Teil unseres Lebens geworden und beeinflussen unsere zukünftigen Entscheidungen und Verhaltensweisen. Dabei nutzen wir z.B. die Wetter-App am Morgen, um zu schauen ob wir den Regenschirm brauchen, schauen in die Verkehrsanbindungs-App, um den schnellsten Weg zur Arbeit zu finden, und informieren uns in der lokalen Nachrichten-App über aktuelle Themen. Überall begegnen uns Situationen, in denen wir diese Dienste aus Informationsbedarf oder Langeweile immer wieder nutzen. Dabei haben fast alle ortsbezogenen Dienste gemein, dass sie die aktuelle Position eines Nutzers sowie seine Aktivität zur Bereitstellung ihrer Dienste nutzen. Um die aktuelle Position oder gar die Aktivität eines Nutzers zu bestimmen, werden Sensordaten und Schnittstellen über das eigene mobile Endgerät verwendet. Dabei kommen funkbasierte Technologien wie GPS oder WLAN zum Einsatz. Trotzdem mangelt es immer noch an Verfahren zur Ortsbestimmung und Positionierung, die es erlauben, eine durchgehende Positionsbestimmung an allen Orten zu ermöglichen. Mittlerweile gibt es zwar einige Systeme, die eine solche durchgehende Positionierung ermöglichen, jedoch sind sie zu teuer, zu komplex und nicht für den allgemein Gebrauch geeignet.
Motiviert durch die hohe Verbreitung von Kamerasensoren in Smartphones und Tablets sowie die gleichzeitig geringen Kosten, der autarken Funktionsweise und der rasanten Entwicklung im Bereich der Bildverarbeitung, ist der Einsatz visueller Verfahren zur Positionsbestimmung und Aktivitätserkennung erstrebenswert. Besonders durch die technologische Entwicklung und die Einführung einer ganz neuen Geräteklasse, der sogenannten *Wearables*, ist die Chance solche Verfahren weiter zu entwickeln immens gestiegen. So werden Geräte wie z.B. die *Google Glass*[1] oder *Microsoft Holo Lens*[2] bereits mit visuellen Verfahren betrieben, da sie grundsätzlich eine oder mehrere Kamerasensoren einsetzten, um die Lage der Kamera oder anderer Objekte im Raum zu bestimmen (vgl. Abb. 1.1). Dadurch können Anwendungen der *Augmented Reality* realisiert werden, um z.B. virtuelle Objekte realitätsgetreu und in Echtzeit anzeigen zu können [1, 2].
Bereist heute werden zahlreiche Polizisten in den USA und auch in Deutschland mit sogenannten *Bodycams* ausgestattet, um einen vollständigen Polizeieinsatz

[1] www.google.com/glass/start/
[2] www.microsoft.com/microsoft-hololens/en-us

Abbildung 1.1: Beispiel eines gertragenen *Wearables* anhand der *Google Glass*.

überwachen zu können [3]. Dabei lassen sich durch die aufgezeichneten Videosequenzen zahlreiche Informationen ableiten, um Sachverhalte rekonstruieren und besser verstehen zu können. Diese Informationen könnten in naher Zukunft bereits in Echtzeit analysiert und ausgewertet werden. Dadurch könnten Einsätze effizienter und sicherer gestaltet werden.
Allgemein lässt sich mit visuellen Sensoren, insbesondere durch die günstigen Kamerasensoren, eine Reihe von sinnvollen Verfahren zur Bestimmung der Position, der Lage, der Aktivität und des Kontextes realisieren. Daher werden in der vorliegenden Arbeit visuelle Verfahren für ortsbezogene Dienste vorgestellt. Diese Verfahren nutzen die Videosequenz einer mobilen Kamera, die entweder von einer Person getragen oder an einer Person direkt befestigt wird. Anhand dieser Verfahren werden die aktuelle Position, die gegangene Trajektorie sowie die getätigten Aktivitäten einer Person bestimmt.

1.1 Ortsbezogene Dienste

Standortbezogene oder auch *ortsbezogene Dienste* stellen meist mobile Anwendungen dar, die anhand der aktuellen Position einer Person selektive Informationen bereitstellen [4, 5, 6]. Anwendungsgebiete ortsbezogener Dienste werden häufig im Bereich der Navigation, der Ortsauskunft und der Sicherheit verwendet. Dienste der Navigation werden durch Routenplaner oder Fahrzeugnavigation bereitgestellt. Dienste der Ortsauskunft sind dagegen vielfältiger. Diese können z.B. Informationen oder Bewertungen über ein Restaurant, lokale Wetterinformationen oder auch Werbemaßnahmen beinhalten. Im Bereich der Sicherheit werden solche Dienste meist zum Schutz vor unberechtigten Zugriffen eingesetzt. In diesem Fall dient die eigene Position als zweiter Faktor zur Authentifizierung gegenüber einem zugriffsberechtigten System.
Hauptvoraussetzung eines ortsbezogenen Dienstes ist das Wissen über die eige-

ne Position. Diese kann auf unterschiedliche Art und Weise bestimmt werden, indem entweder direkt oder indirekt die unmittelbare Umgebung betrachtet wird. Häufig nutzen solche Dienste ein mobiles Endgerät, das seine Position anhand eines Lokalisierungsmechanismus kontinuierlich bestimmen kann [7]. Ein solcher Lokalisierungsmechanismus wird heutzutage durch eine Vielzahl uns im Alltag bekannter Technologien bereitgestellt, die im Folgenden kurz vorgestellt werden.

Satellitengestützte Positionierung Die bekannteste Variante der satellitengestützten Positionierungssysteme stellt das *Global Positioning System* (GPS) dar, das in den 1970er Jahren durch das US-Verteidigungsministerium entwickelt wurde [8]. Die Position wird dabei anhand von mindestens drei sichtbaren Satelliten im All und einem Empfängergerät auf der Erde bestimmt. Das Empfängergerät misst die Zeitdauer der ausstrahlenden Signale aller sichtbaren Satelliten, um deren Distanzen zum Empfängergerät zu bestimmen. Anhand der Distanzen lässt sich die Position des Empfängergerätes bestimmen. Solche Empfängergeräte werden mittlerweile sehr günstig produziert und in heutigen Smartphones, Tablets oder anderen mobilen Endgeräten verbaut. GPS hat eine Positionsgenauigkeit von bis zu fünf Metern. Mit Hilfe von Korrekturverfahren sind sogar Genauigkeiten bis zu einem Meter möglich.

Zellbasierte Ortungsverfahren Die zellbasierte Ortung basiert auf der Positionierung innerhalb eines Mobilfunknetzes [9]. Ein mobiles Endgerät, dass über ein Mobilfunkmodul verfügt und eingeschaltet wird, wählt sich automatisch in das Mobilfunknetz über eine Basisstation ein. Da ein Mobilfunknetz aus mehreren Basisstationen besteht, kann die Position anhand der Signalstärke einer Basisstation ermittelt werden. Voraussetzung für eine solche Positionierung ist, dass die Position der Basisstation bekannt ist. Da außerhalb von Ballungsräumen die Dichte an Basisstationen sehr gering ist, kann die Positionsgenauigkeit stark schwanken und sehr ungenau sein.

WLAN-basierte Ortung Die WLAN-basierte Ortung nutzt die an einem Ort charakteristischen WLAN-Signale zur Positionsbestimmung [10]. Besonders durch die hohe Dichte an öffentlichen und privaten *WLAN-Access-Points* hat sich diese Form der Positionierung gerade in Großstädten sehr etabliert. Fast jedes mobile Endgerät verfügt bereits über ein WLAN-Modul, das sich zusätzlich zur Positionierung verwenden lässt, indem die WLAN-Signale an einem Ort gemessen werden. Es haben sich besonders zwei Verfahren etabliert, nämlich die Lateration von WLAN-Signalen und die Charakterisierung eines Ortes anhand von WLAN-Fingerprints. Für das erste Verfahren müssen die Positionen der *WLAN-Access-Points* bekannt sein, um eine Lateration überhaupt durchführen zu können. Ein *WLAN-Fingerprint*, der ein Abbild der empfangenen WLAN-Signale an einem Ort darstellt, muss dagegen zumindest einmal im Vorfeld an dem selben Ort aufgezeichnet werden. Gerade

in Großstädten stellt die WLAN-Basierte Ortung eine unterstützende und auch stromsparende Alternative zu GPS dar.

Neben diesen drei bekannten Technologien existieren viele andere Funkbasierte Technologien, die zur Positionierung eingesetzt werden können, wie z.B. der *Near-Field-Communication-* [11] oder *Bluetooth-Standard* [12]. Ebenfalls verwenden einige Verfahren zusätzlich die Sensorik des Smartphones, wie z.B. den Kompass, das Gyroskop und den Beschleunigungssensor. Dadurch soll eine feingranulare Positions- und Lagebestimmung auf kleinstem Raum möglich sein. Auch visuelle Verfahren zur Positions- und Lagebestimmung anhand von Kamerasensoren gewinnen immer mehr an Bedeutung und werden für Industrie und Forschung interessanter. Gerade die Tatsache, dass visuelle Verfahren meist unabhängig von Infrastrukturen funktionieren können, die Kosten für einen Kamerasensor sehr gering sind und die technische Entwicklung zur Verarbeitung von Bildinformation sich rasant gesteigert hat, macht die Auswertung mit visuellen Verfahren so interessant. Daher werden im Rahmen dieser Arbeit visuelle Verfahren vorgestellt, die genau diese Vorteile nutzen. Es wird gezeigt, dass ihre Anwendung im Bereich ortsbezogener Dienste sinnvoll ist.

1.2 Aufbau der Arbeit

Der Aufbau der Arbeit wird im Folgenden vorgestellt. Kapitel 2 behandelt Grundlagen des maschinellen Sehens. Dabei werden zunächst grundlegende Verfahren erläutert, insbesondere das visuelle Feature-Verfahren *Speeded Up Robust Features*, kurz SURF. Dieses Verfahren dient dabei als Grundlage der im Rahmen dieser Arbeit vorgestellten Methoden. Abschließend werden Anwendungsgebiete und Beispiele des maschinellen Sehens durchleuchtet.
Kapitel 3 stellt ein Bildvergleichsverfahren basierend auf SURF vor, das aus mehreren Vergleichsstufen besteht. Die Entwicklung eines eigenen Vergleichsverfahrens ist notwendig, da es für mehrere Verfahren, die im Rahmen dieser Arbeit vorgestellt werden, benötigt wird. Dafür wird zunächst ein Überblick über grundlegende Verfahren des Bildvergleichs auf Basis von Features gegeben. Dann werden Verfahren der Quantisierung und des Kontur-Vergleiches vorgestellt und in den mehrstufigen Vergleichsprozess integriert. Anschließend wird der mehrstufige Vergleichsprozess anhand eines Anwendungsbeispieles evaluiert.
Kapitel 4 stellt drei Erweiterungen für ein bereits bestehendes visuelles Positionierungssystem mit dem Namen *MoVIPS* dar. Für ein besseres Verständnis wird zunächst ein Überblick über visuelle Positionierungssysteme gegeben. Da die Erweiterungen auf *MoVIPS* angewendet werden, wird das visuelle Positionierungssystem selbst vorgestellt und die Problemstellungen näher gebracht. Die drei Erweiterungen beschleunigen die Anfragezeit und verbessern die Positionsgenauigkeit des visuellen Positionierungssystem. Alle drei Verfahren wer-

den einzeln evaluiert und die Ergebnisse vorgestellt.
Kapitel 5 stellt ein visuelles Odometrie-Verfahren vor, dass ausschließlich auf Basis von visuellen Merkmalen (SURF-Features) die Eigenbewegung einer Person misst und draus die relative Position ableiten kann. Dabei verwendet das Verfahren einen visuellen Schrittzähler und einen visuellen Kompass auf dessen Basis die Position bestimmt werden kann. Sowohl der Schrittzähler als auch das gesamte visuelle Odometrie-Verfahren werden anschließend evaluiert. Abschließend werden drei Ansätze einer Aktivitätserkennung basierend auf den selben visuellen Merkmalen vorgestellt. Dabei werden klassische Aktivitäten wie *Laufen*, *Gehen* und *Stehen* sowie weitere Aktivitäten untersucht und auf ihre Erkennungsrate evaluiert.
Zum Schluss wird eine Zusammenfassung gegeben und der Einsatz der vorgestellten visuellen Verfahren für ortsbezogene Dienste diskutiert. Dabei wird auf die Möglichkeit der schnellen und effizienten Positionsbestimmung und auf die gleichzeitige Klassifizierung von Aktivitäten nochmals Bezug genommen. Ebenfalls wird auf die Kombination aus dem visuellen Positionierungssystem und dem Verfahren der visuellen Odometrie im Hinblick auf die Anforderungen der Positionsbestimmung und Aktivitätserkennung ortsbezogener Dienste eingegangen.

1.3 Zugrundeliegende Vorarbeiten

Ein Teil der Konzepte und Ergebnisse, die in dieser Arbeit vorgestellt werden, sind in insgesamt sechs Vorarbeiten veröffentlicht worden. Die folgende Liste gibt einen Überblick, in welche Abschnitte die Vorarbeiten eingeflossen sind:

- In Abschnitt 3.3 wird der mehrstufige Vergleichsprozess vorgestellt. Große Teile des Vergleichsprozesses basieren auf der Vorarbeit von Marouane et al. [13]. Andre Ebert hat bei der Diskussion der Vorarbeit unterstützt. Zu den Eigenanteilen des Autors der vorliegenden Arbeit gehören die Konzeption und Evaluierung des mehrstufigen Vergleichsprozesses.

- Das in Abschnitt 3.4 vorgestellte SURFLogo System basiert ebenfalls auf der Vorarbeit von Marouane et al. [13]. Andre Ebert hat bei der Diskussion der Vorarbeit unterstützt. Zu den Eigenanteilen des Autors der vorliegenden Arbeit gehört die Konzeption, Implementierung und Evaluierung des SURFLogo System.

- Die Erweiterungen des visuellen Positionierungssystem *MoVIPS* aus Abschnitt 4.4 basieren auf der Vorarbeit von Marouane et al. [14]. Martin Werner hat Idee zu dem Konzept zur Erweiterung des Vergleichsverfahren um den Schrittzähler und Kompass entwickelt. Marco Maier und Sebastian Feld haben bei der Diskussion der Vorarbeit unterstützt. Zu

den Eigenanteilen des Autors der vorliegenden Arbeit gehören die Konzeption und Evaluierung der Erweiterung zur Quantisierung der Features und der Erweiterung des Verfahrens der Positionskorrektur um die Rotationsinvarianz.

- Der Ansatz des visuellen Schrittzählers basiered auf den Feature-Koordinaten aus Unterabschnitt 5.4.2.1 basiert auf der Vorarbeit von Marouane et al. [15]. Andre Ebert hat bei der Diskussion und zur Literaturrecherche der Vorarbeit beigetragen. Zu den Eigenanteilen des Autors der vorliegenden Arbeit gehören die Konzeption und Evaluierung des visuellen Schrittzählers basierend auf den Feature-Koordinaten.

- Der zweite Ansatz des visuellen Schrittzählers basierend auf dem Attribut der *Orientation* aus Unterabschnitt 5.4.2.2 basiert auf der Vorarbeit von Marouane et al. [16]. Das Konzept des *OrientationScore* wurde gemeinsam von Maximilian Christl und dem Autor der vorliegenden Arbeit entwickelt. Andre Ebert hat bei der Diskussion und Literaturrecherche der Vorarbeit beigetragen. Claudia Linnhoff-Popien hat bei der Diskussion der Vorarbeit unterstützt. Zu den Eigenanteilen des Autors der vorliegenden Arbeit gehören die allgemeine Konzeption und Evaluierung des visuellen Schrittzählers basierend auf dem Attribut der *Orientation*.

- Der in Unterabschnitt 5.4.4 vorgestellte visuelle Kompass basiert auf den Vorarbeiten von Marouane et al. [17, 18]. Das Konzept des visuellen Kompass wurde in mehreren Schritten in den Vorarbeiten erweitert und wurde mit gleichem Anteil von Robert Gutschale, Alexander Leupold und von dem Autor der vorliegenden Arbeit mitentwickelt. Marco Maier und Andre Ebert haben bei der Diskussion, bei der Ausarbeitung und bei der Literaturrecherche der Vorarbeit unterstützt. Claudia Linnhoff-Popien hat zur Diskussion der Vorarbeit beigetragen.

- Das in Abschnitt 5.4 vorgestellte System der visuellen Odoemtrie basiert auf der Vorarbeit von Marouane et al. [18]. Robert Gutschale hat bei der Zusammenstellung der Daten und bei der Evaluierung des Konzeptes maßgeblich mitgeholfen. Zu den Eigenanteilen des Autors der vorliegenden Arbeit gehören die Entwicklung des Konzeptes und Teile der Evaluierung des visuellen Odometrie-Verfahrens. Claudia Linnhoff-Popien hat zur Diskussion der Vorarbeit beigetragen.

- Die in Unterabschnitt 5.5.1 vorgestellte Aktivitätserkennung basierend auf dem Feature-Attribut der *Scale* und der Schrittfrequenz basieren auf der Arbeit von Marouane et al. [16]. Das Konzept ist in gleichen Teilen von Maximilian Christl und vom Autor der vorliegenden Arbeit entwickelt und evaluiert worden.

Die Kerninhalte in Kapitel 3 wurden vom Autor bereits in [13] publiziert. Sowohl die Idee, das Konzept und die Evaluation sind bereits in der Vorveröffent-

lichung enthalten. Ebenfalls im Paper bereits enthalten sind die Abbildungen 3.4, 3.6, 3.7, 3.8, 3.9, 3.10, 3.11, 3.12a und 3.12b.
Die Kerninhalte in Kapitel 4 wurden vom Autor bereits in [14] publiziert. Sowohl die Idee, das Konzept und die Evaluation sind bereits in der Vorveröffentlichung enthalten. Ebenfalls im Paper bereits enthalten sind die Abbildungen 4.7, 4.8, 4.10a, 4.10b, 4.11a, 4.11b, 4.14b und 4.12a.
Die Kerninhalte in Kapitel 5 wurden vom Autor bereits in [15, 18, 16, 17] publiziert. Sowohl die Idee, das Konzept und die Evaluation sind bereits in den Vorveröffentlichungen enthalten. Ebenfalls in den Vorveröffentlichungen bereits enthalten sind die Abbildungen 5.9a, 5.10, 5.12, 5.13, 5.15 und 5.18. Die Inhalte bzgl. der Aktivitätserkennung mit Klassifikatoren sowie die darauf aufbauende Evaluierung (vgl. Abschnitt 5.5) sind in den Vorveröffentlichungen nicht enthalten.

2 Grundlagen des maschinellen Sehens

Das Gebiet des maschinellen Sehens ist ein stetig wachsendes Feld, welches in einem rasanten Tempo immer komplexere Systeme und Verfahren hervorbringt. Hinter jedem Verfahren und System steht eine meist einfache Kamera, die einzelne oder ganze Sequenzen von Bildern aufzeichnet. Um aus einem einzelnen Bild oder einer Sequenz Informationen ableiten zu können, muss man wissen welche Informationen aus einem Bild verwendet und wie man diese Informationen möglichst vollständig und automatisiert extrahiert. Dafür muss zunächst eine geeignete Bildrepräsentation gefunden werden, um diese Bildinformationen verarbeiten zu können. Außerdem sind Verfahren der Vorverarbeitung, Analyse und Extraktion notwendig, um aus diesen Informationen intelligente und automatisierte Verfahren und Systeme entwickeln zu können. Im Folgenden Kapitel werden einige Grundlagen aus dem Bereich des maschinellen Sehen vorgestellt. Es werden zunächst die unterschiedlichen Bildrepräsentationen erläutert und erklärt. Anschließend werden die wichtigsten Verfahren, die zum größten Teil im Rahmen dieser Arbeit verwendet werden, vorgestellt. Im Abschnitt 2.3 wird das Feature-Analyse und Extraktions-Verfahren visueller Merkmale *SURF* genauer vorgestellt, da dieses im Rahmen der vorliegenden Arbeit eine große Bedeutung für die Entwicklung der eigenen Verfahren hat. Abschließend werden existierende und forschungsnahe Systeme aus dem Bereich des maschinellen Sehens vorgestellt.

2.1 Bildrepräsentation: Darstellung von Bildinformationen

Um optische Objekte, analoge Bilder oder ganze Szenarien maschinell bearbeiten zu können, müssen diese zunächst in ein digitales Bildformat übertragen werden. Ein digitales Bild $I(x, y)$ wird durch eine zweidimensionale Matrix beschrieben, bei der jeder Punkt (x, y), auch Pixel genannt, einer Helligkeitsstufe zugeordnet wird. Eine Bildmatrix besteht aus einer Menge $x * y$ Pixeln, von der jedes Pixel durch eine Spalte x und Zeile y anhand eines Wert bestimmt wird. Dieses einfache Model zur Beschreibung eines Bildes wird zur Datenhaltung von monochromen Bildern oder Graustufen-Bildern verwendet. Farbbilder oder multispektrale Bilder, die aus mehreren Farbkanälen bestehen, werden durch ein erweitertes Bildmodel beschrieben. Dieses sieht vor für jeden

Farbkanal f eine eigene Funktion $I_f(x, y)$ zu verwenden. Im Folgenden werden die unterschiedlichen Typen von Bildrepräsentationen in Form von *Binärbilder*, *Graustufenbilder*, *Farbbilder* und *multispektral Bilder* vorgestellt.

2.1.1 Binärbilder

Binärbilder stellen den einfachsten Typen an Bildrepräsentationen dar. Jedes Pixel (x, y) eines Bildes wird durch die Farbe schwarz oder weiß repräsentiert und jeweils mit einer 1 oder 0 in der Bildmatrix gespeichert. Da ein Binärbild nur aus zwei Zuständen pro Pixel beschrieben wird, kann jedes Pixel durch einen Bit im Speicher gehalten werden. Ein Binärbild hat dadurch eine sehr geringe Datengröße, welche durch einen hohen Verlust an Bildinformationen bedingt wird. Gerade für Verfahren aus dem Bereich des maschinellen Sehens werden Binärbilder ausschließlich für die Darstellung von Konturen oder Formen eines Objektes verwendet, da sie effizienter zu verarbeiten sind [19]. Im Bereich der Texterkennung oder in der Objekterkennung werden Binärbilder zur einfachen Verarbeitung verwendet. Auch beim klassischen Fax werden Dokumente in ein Binärbildformat übertragen.

2.1.2 Graustufenbilder

Graustufenbilder sind eine erweitere Form der Binärbilder und haben anstelle von zwei Zuständen einen größeren Wertebereich pro Pixel. Jeder Wert beschreibt die Helligkeit eines Pixels. Die Größe des Wertebereiches bestimmt die Anzahl an unterschiedlichen möglichen Helligkeitsstufen eines Pixels. Klassischerweise verfügen Graustufenbilder jeweils 8-Bit pro Pixel und können damit 256 unterschiedliche Helligkeitstuffen abbilden [20]. Die unterschiedlichen Helligkeitstuffen liegen zwischen 0 (weiß) und 255 (schwarz). Graustufenbilder mit einem solchen Wertebereich haben eine ausreichende Darstellungsmöglichkeit und genügen den Anforderungen der visuellen Wahrnehmung eines Menschen [21]. Die 8-Bit Repräsentation pro Pixel hat ihren Ursprung unter anderem darin, dass 8-Bit einem Byte entsprechen und diese Größe die kleinste adressierbare Einheit in klassischen Rechnerarchitekturen ist. Anwendungen, die eine höhere und feinere Darstellung der unterschiedlichen Helligkeitsstufen benötigen, nutzen 12- oder 16 Bit pro Pixel. Solche Wertebereiche werden meist im Bereich der Medizin oder Raumfahrtechnik eingesetzt.

2.1.3 Farbbilder

Farbbilder können durch drei Abbildungen von Graustufenbildern beschrieben werden, wobei jeder Wertebereich eines Graustufenbildes einen anderen Farbfrequenzbereich beschreibt. Licht z.B. lässt sich erst durch viele verschiedene Farbfrequenzbereiche beschreiben. Ähnlich verhält es sich beim Farbbild, wobei hierfür nur drei unterschiedliche Farbfrequenzbereiche benötigt werden. Farbbilder verwenden klassischerweise die Farbfrequenzbereiche rot, grün und

blau und haben damit 24 Bit pro Pixel. Daher kommt auch die Bezeichnung der RGB-Bilder aus dem Bereich der Bildverarbeitung. Die drei Frequenzbereiche sind sogenannte Grundfarben und ermöglichen durch Mischen der Grundfarben unterschiedliche Farben für das menschliche Auge darzustellen [22]. Ein Pixel wird durch einen Vektor (R, G, B) beschrieben. Neben diesem Model zur Beschreibung von Farbbildern gibt es andere und für den Menschen intuitiv verständlichere Farbmodelle bei denen die Helligkeitsinformationen getrennt von den Farbinformationen eines Bildes betrachtet werden. Dabei werden die Helligkeitsinformationen durch einen eindimensionalen und die Farbinformationen durch einen zweidimensionalen Vektor beschrieben. Ein Beispiel dafür ist das Hue-Saturation-Lightness Farbmodell (HSL). Dieses erlaubt es, Farben einfacher und verständlicher zu beschreiben. Die *Lightness* beschreibt die eigentliche Helligkeit einer Farbe und die *Hue* ist die wahrzunehmende Farbe. Die *Saturation* ist ein Maß dafür wie viel Weißanteil in einer Farbe steckt. So lässt sich z.B. die Farbe Türkis als Farbe Blau mit einem höheren Weißanteil beschreiben. Eine ähnliche Farbmodellbeschreibung ist das Hue-Saturation-Value Farbmodell (HSV). Dieses beschreibt ähnlich wie das HSL Farbmodell Farben mit dem einzigen Unterschied, dass der Wert *value* das Maximum aus den Farbwerten Rot, Grün und Blau abbildet.

2.1.4 Multispektral Bilder

Das menschliche Auge kann nur einen sehr begrenzten Spektralbereich elektromagnetischer Wellen wahrnehmen. Dieser Bereich wird als Licht zusammengefasst und umfasst alle sichtbaren Farben. Wellenlängen von z.B. ultravioletter Strahlung, Infrarot-Strahlen oder auch Röntgenstrahlen können Menschen mit dem Auge nicht wahrnehmen. Diese Art von Informationen werden in sogenannten multispektralen Bildern abgebildet [23]. Dabei werden die Informationen aus den unterschiedlichen Frequenzbereichen in ein RGB-Farbmodell konvertiert. Beinhalten die Informationen mehr als nur drei Frequenzbereiche, werden hierfür andere komplexere Farbmodelle verwendet, die jedoch eine Reduktion des Informationsgehaltes beinhalten. Multispektrale Bilder werden meist für Satellitenbildaufzeichnungen, Sonarsysteme, Infrarot-Bildern oder auch in der Medizin zur Diagnostik verwendet.

2.2 Methoden der Bildverarbeitung

Um geeignete Informationen aus einem Bild oder aus einer Sequenz von Bilder zu erhalten, benötigt man unterschiedliche Verfahren der Bildverarbeitung. Diese Verfahren können aus einer Kombination mehrerer Teilverfahren bestehen oder simple Operationen auf einem Bild darstellen. Die Methoden bewirken meist eine Reduktion der Bildinformationen und helfen dabei die wichtigen Informationen zur Problemfindung hervorzuheben. Ziel dieser Verfahren ist es komplexe Bildinformationen soweit zu reduzieren, damit ein anderes Verfahren

oder System sie weiter verarbeiten kann. So lassen sich Objekte in einem Bild segmentieren, so dass ein Roboter diese greifen kann; oder spezielle Farbbereiche lassen sich durch solche Verfahren einfacher hervorheben, welche z.B. dabei helfen bösartige Hauttumore ausfindig zu machen. Im Folgenden werden die einzelnen Verfahren vorgestellt.

2.2.1 Filter

Filter werden in der Bildverarbeitung zur Verbesserung oder zur Modifizierung eines Bildes angewendet. Das Hervorheben, Entfernen oder Verfremden von bestimmten Merkmalen und Eigenschaften eines Bildes ist das Ziel einer Filteroperation. Eine Filteroperation wird jeweils auf einen Pixel angewendet, jedoch werden die Nachbarpixel mit einbezogen. Das heißt, dass der Wert des gerade zu betrachtenden Pixels $p(x, y)$ durch die einmalige oder mehrmalige Anwendung bestimmter Operationen auf den Werten der Nachbarpixel bestimmt wird.
Es gibt verschiedene Arten von Filtern, die im Folgenden kurz vorgestellt werden.

2.2.1.1 Glätten - Weichzeichner

Ein Glättungsfilter oder Weichzeichner wird meist eingesetzt um Rauscheffekte im Bild zu reduzieren. Eine solche Filteroperation wendet lineare Filter auf einem Bild an. Dabei wird der Wert für einen Pixel (x, y) durch die gewichtete Summe aller Pixelwerte der lokalen Nachbarschaft berechnet. Die Gewichtung der Summe aller Pixelwerte durch den linearen Filter wird mit Hilfe einer Matrix, auch **Kernel** genannt, bestimmt. Die Größe der Nachbarschaft hat Einfluss auf die Größe des Kernels, welche wiederum die Stärke des Weichzeichners bestimmt. Da während einer Glättung eines Bildes jeder Pixel betrachtet wird, kann man den Weichzeichner auch als sich verschiebendes Fenster (engl. *sliding window*) betrachten.
Um aus einem Eingabebild I ein geglättetes Ausgabebild O zu erhalten, wird mit einem definierten Kernel K jeder Pixel (x, y) wie folgt berechnet [24]:

$$O(x, y) = \sum_{m,n} I(x + m, y + n) * K(m, n)$$

Es gibt unterschiedliche Weichzeichner, die zum Einsatz kommen. Die bekanntesten sind der Median-Filter sowie der der Gauß-Filter. Ein Median-Filter eignet sich gut bei Bildern, die hauptsächlich gesprenkelte Rauschpunkte haben. Dagegen eignet sich ein Gauß-Filter eher zur Hervorhebung von Kanten.

2.2.1.2 Unscharfmaskierung

Unscharfmaskierung (engl. *sharpening*) ist ein Filter, der Ränder und feine Details in einem Bild hervorhebt. Der Filter verwendet dafür die Ableitungen

erster und zweiter Ordnung. Mit Hilfe der Ableitung erster Ordnung werden alle Helligkeitswerte berechnet, sodass man ein sogenanntes Gradienten-Bild erhält. Mit Hilfe der Ableitung zweiter Ordnung lässt sich die Divergenz jedes Gradienten bestimmten. Da ein digitales Bild für jeden Pixel nur einen diskreten Wertbereiche einsetzt, werden für die Unscharfmaskierung die abgeleiteten Bildversionen erster und zweiter Ordnung verwendet.
Die Ableitung erster Ordnung erzeugt in einem Bild dickere Bildkanten und wird meist in Verbindung mit Kanten-Detektoren verwendet. Die Ableitung zweiter Ordnung dient eher zum Hervorheben feiner Details in einem Bild. Die bekanntesten Varianten eines solchen Filters sind der Sobel-Operator und der Laplace-Operator [20].

2.2.1.3 Bildpyramiden

Bildpyramiden werden eingesetzt, sobald man mit der originalen Auflösung eines Bildes nicht alleine auskommt. Hierfür werden unterschiedliche und geringere Bildauflösungen von dem Originalbild erzeugt und beginnend mit der kleinsten Auflösung bis zum Originalbild eine Pyramide sequenziell aufgebaut. Dabei hat ein solcher Aufbau den Vorteil, dass man z.B. bei der Objektsuche erst mit der kleinsten Bildauflösung beginnt und nur dann mit den nächst größeren Auflösungen weiter macht, solange man das Objekt noch nicht gefunden hat. Dies ermöglicht ein sequenzielles Arbeiten mit steigender Genauigkeit. Es gibt zwei Varianten dieser Bildpyramiden, die im Folgenden kurz vorgestellt werden.

Gauß-Pyramide Diese Form der Bildpyramide reduziert die Auflösung des Originalbildes, indem in jeder Stufe l jeweils die Spalten und Zeilen der Bildmatrix I_l reduziert werden. Das heißt, dass nach jeder Stufe das Bild genau um die Hälfte seiner Zeilen und Spalten reduziert wird. Anschließend wird anhand des Gauß-Filters die Nachbarschaft der Stufe $l-1$ betrachtet, um den Wert für jeden Pixel (x, y) aus der Stufe l zu berechnen. Der Name der Gauß-Pyramide leitet sich durch den verwendeten Weichzeichner ab.

Laplace-Pyramide Diese Bildpyramide setzt die vorher beschriebene Gauß-Pyramide ein und kann als dessen inverses Gegenstück betrachtet werden. Jede Stufe $l_{laplace}$ berechnet sich aus der Differenz der Stufen l_{gauss} und $l_{gauss}-1$ der Gauß-Pyramide. Diese Operation wird auch *Differential of Gaussian* (DoG) genannt. Das Ergebnis der Differenzbildung lässt die Bilder jeder Stufe wie sogenannte *"Rahmenbilder"* wirken. Dieser Effekt entsteht dadurch, dass die Bilder der nächst höheren Stufe um die Hälfte kleiner sind und bei der Subtraktion die Ränder überstehen. Dadurch bleiben die Ränder bestehen wohingegen der mittlere Bereich größtenteils weiß bleibt.

2.2.2 Segmentierung

Die Segmentierung stellt eine der wichtigsten Methoden im Bereich des maschinellen Sehens dar. Einzelne Bereiche oder Objekte auf einem Bild oder in einer Bildsequenz zu erkennen und zu isolieren, sind für viele automatisierte Prozesse des maschinellen Sehens entscheidende Verfahren. So werden z.B. auf Überwachungssystemen bewegliche Objekte wie Fahrzeuge oder Menschen vom starren Hintergrund erkannt und hervorgehoben. Auch auf einzelnen Bildern können Objekte vom restlichen Hintergrund des Bildes isoliert werden. Hierfür können Bereiche des Bildes durch Farboperationen selektiert werden, indem jedes Pixel auf eine bestimme Farbe oder ganze Farbbereiche geprüft und anschließend ein- oder ausgeblendet wird. Zur Segmentierung eignen sich auch Klassifikatoren, die spezifische Merkmale oder Bildbereiche von einander unterscheiden können. Des Weiteren gibt es auch die Möglichkeit markante Kanten oder ganze Konturen innerhalb eines Bildes ausfindig zumachen, indem z.B. spezielle Filter verwendet werden. Im Folgenden werden einige dieser Verfahren und Werkzeuge aus dem Bereich der Segmentierung vorgestellt.

2.2.2.1 Background Subtraction

Die *Background Subtraction* oder zu deutsch Hintergrund-Subtraktion ist eine der einfachsten und effizientesten Methoden zur Segmentierung von Objekten in Bildsequenzen. Das Verfahren blendet den Hintergrund vollständig aus, um dadurch Objekte im Vordergrund hervorzuheben [20]. Speziell im Bereich der Videoüberwachung wird diese Methode eingesetzt, um z.B. Fahrzeuge, Menschen oder bewegliche Objekte ausfindig zu machen. Es gibt einfache und komplexe Umsetzungen der Background Subtraction, wobei den meisten Verfahren das sogenannte *Frame Differencing* zu Grunde liegt. Dabei wird im einfachsten Fall das erste Bild einer Bildersequenz, die von einer statischen Kamera aufgezeichnet wird, als sogenanntes Hintergrundbild verwendet. Dieses wird anschließend dafür verwendet, die Differenz aus jedem darauffolgenden Bild aus der Sequenz zu berechnen. Häufig werden die Bilder in Graustufenbilder konvertiert, sodass die Segmentierung durch die Subtraktion eindeutiger ausfällt. Zusätzlich verwenden viele Verfahren in einem zweiten Schritt eine Grenzwertfunktion, bei der jeder Pixel unterhalb des Grenzwertes mit einer 0 bewertet wird und ansonsten eine 1 als Wert erhält. Das Ergebnis ist am Ende ein Binärbild, dass den beweglichen Vordergrund abbildet (vgl. Abb. 2.1).
Erweitere Varianten der Background Subtraction verwenden nicht nur das erste Bild für das Hintergrundmodell, sondern ersetzten nach einiger Zeit oder nach bestimmten Regeln das Hintergrundmodell mit einem anderen Bild.

2.2.2.2 Template Matching

Das *Template Matching* ermöglicht es in einem zuvor unbekannten Bild einen vorgegebenen Bildausschnitt zu suchen. Als Eingabe wird ein abzugleichen-

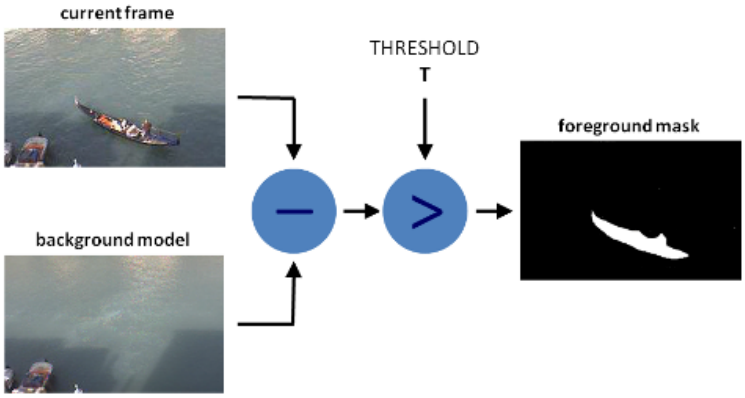


Abbildung 2.1: Background Subtraktion: Differenzbildung aus dem Hintergrundbild (background model) und dem aktuellen Bild (currentframe) einer Videosequenz sowie die Anwendung einer Grenzwertfunktion (THRESHOLD). Das Ergebnis ist ein Binärbild (foreground mask) mit dem erfolgreich segementierten Objekt. [25]

der Bildausschnitt (*Template Image*) benötigt. Dieses sogenannte Template T wird Pixel für Pixel über das zu untersuchende Bild I gelegt und eine Übereinstimmungs-Operation ausgeführt. Das Ergebnis wird als Grauwert in einer Ergebnismatrix R gespeichert. Um die Ergebnismatrix zu erhalten können unterschiedliche Übereinstimmungs-Operation eingesetzt werden. Die Methode der quadrierten Differenz berechnet dabei für jede mögliche Position eines Templates $T(\acute{x}, \acute{y})$ die Differenz mit der aktuellen Position des zu untersuchenden Bildes $I(x + \acute{x}, y + \acute{y})$ und quadriert diese anschließend [19].

$$R(x, y) = \sum_{\acute{x},\acute{y}} (T(\acute{x}, \acute{y}) - I(x + \acute{x}, y + \acute{y}))^2$$

Die Ergebnismatrix R entspricht letztendlich einem Graustufenbild. Der dunkelste Pixel in dem Graustufenbild stellt das Zentrum des vermeintlichen Fundortes des zu suchenden Templates dar. Neben der Methode der quadrierten Differenz, gibt es eine erweiterte Variante, welche die Wert zusätzlich normiert. Die Wahl für die richtige Übereinstimmungs-Operation ist abhängig von der Beschaffenheit des Bildmaterials.

2.2.2.3 Kantendetektoren

Konturen können ebenfalls verwendet werden, um ein Bild zu segmentieren und Objekte hervorzuheben. Um Konturen in einem Bild ausfindig zu machen, empfiehlt es sich die Kanten zu betrachten. Durch das Hervorheben der Kanten, lassen sich Konturen in einem Bild einfacher finden. Kantendetektoren sind damit ein entscheidendes Hilfsmittel zur Auffindung von Konturen. Die wohl bekannteste Methode zur Hervorhebung von Kanten ist der *Canny*

Edge Detector von Canny et al. [26]. Allgemein lassen sich Kanten durch sehr starke Helligkeitsunterschiede in einem Bild ausfindig machen. Ist ein Pixel extrem dunkel und sein benachbartes Pixel extrem hell, liegt womöglich eine lokale Kante vor. Da in der Praxis damit nicht alle Kanten erkannt werden, müssen einige zusätzliche Operationen vorgenommen werden. Der Canny Edge Detektor benötigt hierfür insgesamt vier Operationen. Die erste Operation sieht eine Glättung mittels Gauß-Filter vor. Dabei werden auf dem Eingabebild Rauschpixel eliminiert und eine Anpassung von stark abweichenden Regionen angewendet. Die zweite Operation berechnet die Gradienten der einzelnen Pixel vor. Dafür wird für jedes Pixel (x, y) eine partielle Ableitung in x- und y-Richtung vorgenommen. Durch die berechneten Gradienten lassen sich die Richtung und Stärke von Kanten ermitteln. Die dritte Operation vergleicht für jedes Pixel die Stärke der Gradienten mit den Nachbarpixeln entlang der Gradientenrichtung. Das Pixel mit dem höchsten Stärkegrad, wird als potenzieller Kantenpixel gespeichert. So werden einzig die Maxima entlang einer Kante berücksichtigt. Bei der letzten Operation findet eine sogenannte Hysterese statt. Hierfür werden zwei vordefinierte Grenzwerte (Ober-/Untergrenze) genutzt. Liegen die Pixelwerte über dem oberen Grenzwert, werden diese als Kantenpixel angenommen. Wird ein Kantenpixel gefunden, werden alle darauf folgenden Pixel, die betrachtet werden und über der unteren Schranke liegen, der Kante hinzugefügt.

2.2.2.4 Kontur-Findung

Wie schon im vorherigen Abschnitt erwähnt, lassen sich mit Hilfe von Konturen Bilder segmentieren. Dabei enthält eine Kontur eine Menge von Kanten zusätzlich mit der Information in welcher Beziehung diese Kanten zu einander stehen. Der Unterschied zur vorherigen Kantendetektion besteht darin, dass durch die zusätzliche Information eine Selektion der einzelnen Kanten möglich ist. Nach Anwendung eines Kantendetektors lassen sich in einem Bild Konturen durch eine Sequenz von einzelnen Bildpunkten beschreiben. Eine Kontur lässt sich ähnlich wie ein Polygon beschreiben und wird durch einen äußeren Rand eingegrenzt. Eine solche Sequenz kann jeweils durch Bildkoordinaten beschrieben werden oder durch einen sogenannten *Freeman-Code* [27], welche die Kontur durch eine Aneinanderreihung von diskreten Richtungsvektoren beschreibt (vgl. Abb. 2.2).

2.2.2.5 Morphologische Operationen

Morphologische Operationen sind unter anderem ein wichtiges Werkzeug für einige Segmentierungsverfahren. Die *Erosion* sowie die *Dilatation* stellen zwei Operation dar, welche die Möglichkeit bieten einen Pixelwert abhängig von seiner Umgebung zu modifizieren. Bei beiden Operationen wird ein sogenanntes Strukturelement definiert, welches einen Ankerpunkt besitzt. Ein Strukturelement stellt eine Bildmatrix beliebiger Größe dar. Dieses ist im einfachsten Fall

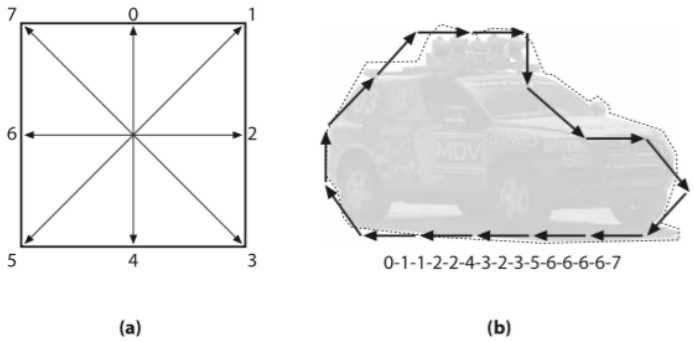


Abbildung 2.2: Freeman-Code: (a) acht diskrete Richtungsvektoren (0-7) - (b) Beschreibung einer Kontur durch die disrekten Richtungsvektoren [24]

ein 3 x 3 großes Pixelbild, welches im Zentrum seinen Ankerpunkt hat. Das Strukturbild wird wie ein Fenster auf das zu bearbeitende Bild gelegt und Pixel für Pixel verschoben. Abhängig von der jeweiligen Operation werden nun alle überlappenden Werte mit Ausnahme des Ankerpunktes verwendet, um das Pixel des zu bearbeitenden Bildes an der Stelle des Ankerpunktes zu verändern. So wird bei der Erosion das Pixel unterhalb des Ankerpunktes durch das Minimum aller Werte innerhalb des Strukturbildes verändert. Dies ermöglicht es lokale Rauschpixel zu eliminieren. Dagegen wird bei der Dilatation das Maximum aller Werte innerhalb des Strukturbildes verwendet. Diese Operation ermöglicht es auseinanderliegende Konturen miteinander zu verbinden. Beide Verfahren werden als Grundlage für die erweiterten Verfahren *Closing* und *Opening* verwendet. Dabei wird beim Closing zunächst die Dilatation und anschließend die Erosion eingesetzt. Somit können Löcher in Strukturen gefüllt werden ohne das die Größe zunimmt. Beim Opening dagegen werden die Operationen umgekehrt eingesetzt. Hier werden zunächst einzelne Rauschflächen entfernt bevor die übriggebliebene Struktur zur alten Größe verstärkt wird. Beide Methoden werden gezielt in der Background Subtaktion oder in der Kontur-Findung eingesetzt um feinere Ergebnisse zu erzielen.

2.2.3 Transformationen

Bildtransformationen stellen eine Sammlung an Werkzeugen zur Konvertierung von Bildinformationen dar. Allgemein wird durch eine Transformation die Repräsentation eines Bildes in eine völlig andere Form gebracht. Da Transformationen im Bereich des maschinellen Sehens, speziell in der Bildbearbeitung, von großer Bedeutung sind, werden einige der wichtigsten Operationen und Werkzeuge, die dafür notwendig sind, in diesem Abschnitt vorgestellt.

2.2.3.1 Hough-Transformation

Eine Hough-Transformation ist eine Methode zur Hervorherbung von Linien, Kreisen oder anderen primitiven Formen [28]. Sie hat im Gegensatz zu den klassischen Kanten-Detektoren den Vorteil, dass sie auch bei verrauschten Bildern relativ robust Kanten und Formen erkennen und sauber hervorheben kann. Sie zählt als Operator zu den Transformationen, weil sie durch das Abbilden der Bildpunkte in einen anderen Modelraum markante Merkmale einfacher sichtbar macht.

Die Transformation bildet ganze Geraden und Kurven als Punkte im sogenannten *Hough-Raum* ab. Das heißt, dass kollineare Geradenstücke auf einen gemeinsamen Punkt im Hough-Raum fallen. Zur Transformation der Punkte aus dem Bildraum in den Hough-Raum wird eine Geradengleichung verwendet. Im einfachsten Fall lässt sich eine Gerade durch folgende Formel abbilden:

$$y_i = ax_i + b$$

Wobei a die Steigung, b der Achsenabschnitt und (x_i, y_i) alle Punkte auf der Geraden sind. Leider lassen sich mit dieser einfachen Geradengleichung Steigungen vertikaler Geraden nicht abbilden, da $m = \infty$ gilt. Daher verwendet man üblicherweise die *Hessische Normalform* zur Abbildung von Geraden, die wie folgt definiert ist:

$$d_i = x_i * cos(\theta) + y_i * cos(\theta_i)$$

Der Abstand d wird für jeden einzelnen Punkt (x_i, y_i) durch die Normale ausgehend vom Ursprung des Koordinatensystems des Bildraumes (x, y) bestimmt. Der Winkel θ liegt zwischen der Normalen und der x-Achse des Bildraumes. Der Hough-Raum ist ein *Dualraum*, wobei eine Achse durch den Abstand d und die andere Achse durch den Winkel θ definiert sind. Der Wertebereich für die Punkte (d_i, θ_i) ist dabei endlich und diskret.

In der Praxis wird das Verfahren wie folgt angewendet. Ein Bild wird zuvor mit einem einfachen Kantendetektor (vgl. Abs. 2.2.2.3) bearbeitet, sodass nur noch ein Binärbild übrig bleibt und die einfache Kontur des Bildes durch weiße Pixel hervorgehoben wird. Dann werden für jeden einzelnen weißen Pixel (x_i, y_i) die Werte d_i und θ_i berechnet und durch eine Akkumulator $AKK[d_i, \theta_i]$ dokumentiert, indem an der Stelle der Wert inkrementiert wird.

Sobald jeder Pixel bearbeitet wurde, lassen sich anhand des Akkumulators Geraden finden, indem man die Extremwerte der Matrix AKK betrachtet. Durch die Hessische Normalform lassen sich anschließend die Geraden im Ursprungsbild einzeichnen.

2.2.3.2 Integralbild

Integralbilder stellen in der Bildverarbeitung ein sehr wichtiges Werkzeug zur schnellen Berechnung von Pixelsummen dar. Das Summieren von Pixelwerten einer Bildregion ist speziell bei der Anwendung von sogenannten *Haar Wavlets*, die in der Objekt-Erkennung und in der Gesichtserkennung verwendet werden [29], relevant.
Für einen Punkt (x, y) eines Integralbildes I_{sum} wird die Summe aller Intensitätswerte der Pixel, die innerhalb des aufspannenden Rechteckes vom Bildursprungspunkt (0,0) bis zum Punkt selbst enthalten sind, berechnet. Dadurch ergibt sich folgende Formel zur Berechnung der Summe aus einem Bild I:

$$I_{sum}(x, y) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(x, y)$$

Jeder Pixel lässt sich aus der Summe der vorherigen Pixel der selben Zeile und aus dem letzten Pixel der vorherigen Zeile sehr effizient zusammenrechnen.
Durch die Transformation eines Bildes in ein Integralbild erhält man den Vorteil, dass die Berechnung von Intensitätswerten eines ganzen Bildes oder eines Ausschnittes davon immer durch vier Punkte berechnen werden kann. So berechnet sich der Intensitätswert eines Rechteckes R mit den vier Kantenpunkten P_1, P_2, P_3 und P_4 wie folgt:

$$I_{sum}(R) = I_{sum}(P_1) + I_{sum}(P_4) - I_{sum}(P_2) - I_{sum}(P_3)$$

Dabei liegt der Ursprung des Koordinatensystems oben links und der Punkt P_1 ist die obere linke und der Punkt P_4 die untere rechte Kante des Rechtecks.

2.2.3.3 Diskrete Fourier-Transformation

Die Fourier-Transformation ist eine sehr hilfreiche Operation in der Bildverarbeitung zur Beschreibung eines Bildes und dessen Eigenschaften. Sie wird neben der Bildanalyse auch als Filter, zur Bildrekonstruktion sowie zur Bildkompression eingesetzt. Die Fourier-Transformation zerlegt ein Bild in seine einzelnen sinus- und cosinus-förmigen Anteile. Das Ergebnis der Transformation ist eine Abbildung des Bildes im Frequenzraum. Jeder Punkt eines Bild entspricht dabei einer bestimmten Frequenz.
Da ausschließlich digitale Bilder in der Bildverarbeitung verwendet werden, wird die *diskrete Fourier-Transformation* (DFT) eingesetzt. Diese hat einen beschränkten Frequenzraum, der jedoch ausreichend für die Beschreibung eines digitalen Bild ist [30].
Zum besseren Verständnis der DFT, sollen im Folgenden einige Bespiele die Einsatzzwecke beschreiben.

Analyse: Sie ermöglicht es ein Bild auf seine Beschaffenheit zu analysieren. So lässt sich z.B. ein Bild mit einer weißen Wand folgendermaßen durch die

DFT beschreiben. Betrachtet man das Bild von links nach rechts sowie von oben nach unten, weisen die Pixel keine große Veränderung bezüglich ihrer Intensität auf. Dies mach sich bei der DFT durch sehr niedrige Frequenzen bemerkbar. Anders sieht das z.B. bei einem Bild mit einem Lattenzaun aus. Hier Ändern sich die Pixelwerte von links nach rechts kontinuierlich. Die DFT bildet ein solches Verhalten durch hohe Frequenzen auf der x-Achse ab, wohingegen auf der y-Achse die niedrigen Frequenzen bestehen bleiben.

Filter: Die DFT kann man auch als Filter verwenden, der unterschiedliche Eigenschaften eines Bildes hervorheben kann. Nimmt man Pixelwerte aus einem Bild heraus, die durch die DFT hohe Frequenzen erzeugen, wirkt es trüber. Anders verhält es sich, wenn man Pixelwerte aus dem Bild entfernt, die sehr niedrige Frequenzen erzeugen. Dies hat den Effekt, dass das Bild schärfer wirkt und Kanten hervorgehoben werden.

In Bildverarbeitungstools wird statt der DFT die *schnelle Fourier-Transformation* (FFT) eingesetzt, da diese eine effizientere Berechnung vornimmt [31]. Sie wird z.B. in der schnellen Ausführung zur diskreten Faltung von Bildern verwendet oder für den *Wiener-Filter* zur Rauschunterdrückung in Bildern eingesetzt [32].

2.2.3.4 Histogrammequalisierung

In der Bildverarbeitung wird meist ein Histogramm zur Beschreibung und Verteilung der einzelnen Farb- und Graustufenwerte (Intensitäten) eines Bildes verwendet. Liegen in einem solchen Histogramm die Farb- oder Graustufenwerte eines Bildes sehr dicht beieinander, kann eine Equalisierung des Histogrammes zu einer besseren Verteilung führen. Die Histogrammequalisierung verteilt dabei die zu nah aneinanderliegenden Farb- und Graustufenwerte auf das gesamte Spektrum.

(a) Histogrammequalisierung: Verteilung der Grauwerte ohne Equalisierung

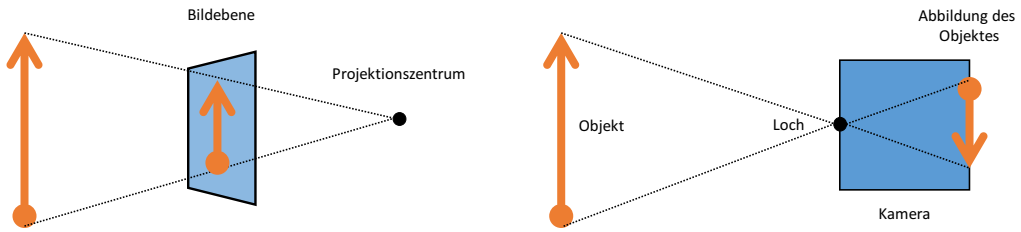
(b) Histogrammequalisierung: Verteilung der Grauwerte nach der Equalisierung

Abbildung 2.3: Histogrammequalisierung: (a) Ohne Equalisierung ist das Bild kaum zu erkennen, die Graustufenwerte liegen zu nah beieinander - (b) Nach der Equalisierung sind die Graustufenwerte besser auf das Spektrum verteilt. Das Bild ist klarer [20].

Abbildung 2.3a zeigt das Originalbild (links) vor Anwendung der Equalisierung und das entsprechende Histogramm (rechts) mit der Verteilung der Grauwerte. Die Werte liegen viel zu nah beieinander, sodass das Bild nicht eindeutig erkennbar ist. Nach der Anwendung der Equalisierung erkennt man in Abbildung 2.3b, dass das Bild (links) nun klarer erkennbar und die Verteilung im Histogramm (rechts) verteilt auf das gesamte Spektrum ist.

2.2.3.5 Perspektivische Projektion

Diese Form der Transformation dient zur Abbildung von 3D-Objekten auf eine 2D-Bildebene. Sie wird häufig als Kameramodell in der 3D-Computergrafik verwendet. Die perspektivische Projektion lässt sich durch das bekannte Beispiel einer Lochkamera einfach erklären (vgl. Abb. 2.4). Die Lochkamera besteht aus einem Kameragehäuse mit einem sehr kleinen Loch durch das das Licht eindringen kann. Das Objekt wird auf der Rückseite des Kameragehäuses auf dem Kopf abgebildet.



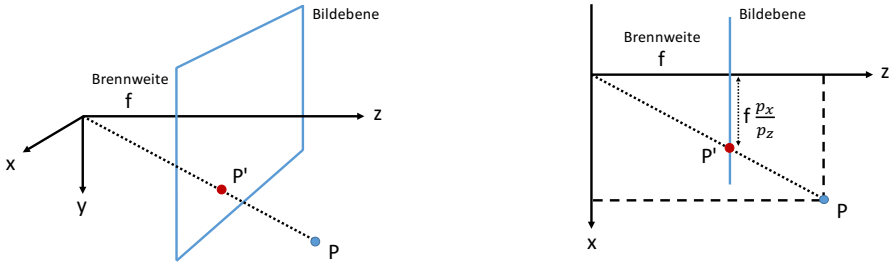
(a) (b)

Abbildung 2.4: (a) Perspektivische Projektion anhand des Beispiels einer Lochkamera. (b) 2D-Bildebene vor dem Projektionszentrum [33].

In der Computergrafik wird die imaginäre Bildebene daher vor dem Projektionszentrum erzeugt, sodass das Bild richtig herum abgebildet wird. Um einen 3D-Punkt P auf der Bildebene nun abbilden zu können, gilt folgende Projektionsvorschrift:

$$proj_{perspektive}\left(\begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}\right) = \begin{pmatrix} f\frac{p_x}{p_z} \\ f\frac{p_y}{p_z} \\ f \end{pmatrix}$$

Dies lässt sich durch die in Abbildung 2.5 aufgezeigte Projekten des Punktes P in P' mit Hilfe des Strahlensatzes ableiten. Mit Hilfe der perspektivischen Projektion $proj_{perspektive}$ und der Brennweite f lässt sich jede 3D-Koordinate auf eine 2D-Bildebene abbilden.



(a) (b)

Abbildung 2.5: (a) Perspektivische Projektion des Punktes P auf die 2D-Bildebene mit dem Punkt P'. (b) Anwendung des Strahlensatzes speziell am x-Wert P_x zur Projektion auf die eindimensionelle Ebene [33].

Durch die Verwendung von homogenen Koordinaten lässt sich die perspektivische Projektion als 4 x 4 Matrix, also einer linearen Abbildung, beschreiben [34]. Dafür wird der Punkt P in eine homogene Koordinate $\bar{P} = (p_x, p_y, p_z, 1)^T$ umgewandelt. Dadurch lässt sich mit folgender Matrix die perspektivische Projektion berechnen:

$$\bar{P'} = \begin{pmatrix} fp_x \\ fp_y \\ fp_z \\ p_z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & f & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}.$$

Durch die Verwendung der homogenen Koordinaten ergibt sich folgender Vorteil. Alle Transformationen, also die Translation, Rotation und Skalierung, lassen sich durch eine einheitliche 4 x 4 Matrix abbilden. Durch die Multiplikation der homogenen Punktkoordinate mit der Transformationsmatrix wird die Transformation auf den Punkt angewendet. Des Weiteren lassen sich komplexe Transformationen aus mehreren einzelnen einfachen Transformationen zusammen bauen. Durch die Multiplikation der einzelnen Matrizen ergibt sich die komplexe Transformation. Auch die Umkehrung einer Transformation wird einfacher, da hierfür die inverse Transformationsmatrix verwendet werden kann.

2.2.4 Visuelle Features

Viele Problemstellungen im Bereich des maschinellen Sehens erfordern eine von der reinen Pixelrepräsentation losgelöste Beschreibung von Bildern. So scheitert bereits die Identifikation oder der Vergleich von Bildern auf Basis der Pixelrepräsentation, sobald die Bilder nicht mehr identisch sind, sondern

sich in der Farbe, Beleuchtung, Skalierung oder Rotation unterscheiden. Daher versucht man Bilder oder Teilbereiche von Bildern durch sogenannte *Features* (Merkmale) zu beschreiben. Features sind eine komplexe und charakteristisch eindeutige Beschreibung eines Punktes oder einer ganzen Teilfläche im Bild. Je nach Verfahren können Features einfache Kanten, Eckpunkte, markante Pixel oder ganze Teilflächen umfassen. Die Verfahren sind zum Teil robust gegen gängige Transformationen oder Kontrastveränderungen und erlauben damit einen sicheren Bildvergleich. Man unterscheidet zwischen der Feature-Analyse und der Feature-Extraktion. Ersteres analysiert ein Bild nach markanten Features, indem unterschiedliche Operationen zur Hervorhebung eingesetzt werden. Durch die Feature-Extraktion wird anschließend die Region um ein gefundenes Feature betrachtet und diese in Form eines diskreten Vektors überführt. Erst durch die diskreten Vektoren können die Bilder anhand von geeigneten Matching-Verfahren eindeutig verglichen werden.
Features und deren Einsatzzweck lassen sich in drei große Kategorien unterteilen [35]. Die erste Kategorie betrachtet die Semantik eines Features. Features, die einen Kantenpunkt repräsentieren, stellen z.B. auf einer Luftaufnahme Straßen dar. Die zweite Kategorie verwendet Features als sogenannte Ankerpunkte. Hier spielt die eigentliche Information, für was das Feature steht, keine Rolle. Ankerpunkte werden speziell in Videosequenzen zum Tracken, zur Bestimmung der Pose einer Kamera oder zur 3D-Rekonstruktion verwendet [36]. Die dritte Kategorie verwendet Features zur robusten Bildrepräsentation. Dies ermöglicht eine fast eindeutige Wiedererkennung von Objekten und ganzen Szenerien ohne eine Aufwendige Segmentierung des Bildes vorzunehmen [37, 38, 39].
Im Folgenden werden die gängigsten Feature-Analyse-Verfahren aufgeführt und klassifiziert. Dann werden Verfahren zur Feature-Extraktion näher vorgestellt. Abschließend werden die dazugehörigen Machting-Verfahren erläutert.

2.2.4.1 Feature-Analyse

Die Feature-Analyse dient zur Auffindung markanter und eindeutiger Merkmale in einem Bild. Dabei gibt es unterschiedliche Arten von markanten Features, die je nach Verfahren ausfindig gemacht werden können. Die wichtigsten zwei Eigenschaften eines Feature sind seine *Robustheit* und *Beständigkeit*. Die Eigenschaften setzen voraus, dass man ein Feature auch bei unterschiedlicher Skalierung, Rotation oder Translation sowie bei unterschiedlichen Lichtintensitäten mit den selben Operationen wieder erkennen kann.
Es gibt zahlreiche Ansätze aus dem Bereich des maschinellen Sehens bei denen Features auf unterschiedliche Art und Weise analysiert und erkannt werden können. Bereits sehr früh wurde in der visuellen Objekterkennung erkannt, dass sich überschneidende Geraden sowie scharfe Kanten als gute Indikatoren für markante Features einsetzen lassen [40, 41].
Die unterschiedlichen Ansätze und Methoden zur Erkennung von Features werden im Folgenden kategorisiert und vorgestellt.

Scharfe Kanten in Konturen: Diese Methoden versuchen explizit scharfe Kanten an Konturen ausfindig zu machen und diese als Features zu verwenden. Um scharfe Kanten zu erkennen, verwenden viele Verfahren zur Analyse einer zu untersuchenden Kontur die Freeman-Code Darstellung (vgl. Kap. 2.2.2.4). Diese Verfahren [42] suchen entweder nach lokalen Maxima in der Sequenz des Freeman-Codes einer Kontur [43, 44], nach Vektoren, die auf einen Richtungswechsel hinweisen, oder nach auffälligen Verhaltensweisen der Vektoren innerhalb des Freeman-Codes [45, 46].

Bildintensitäten: Diese Verfahren suchen nach Regionen im Bild, die von sehr starken Intensitätsschwankungen betroffen sind. Dabei werden die Bilder meist in ein Graustufenbild konvertiert und mit Hilfe von Ableitungsfunktionen der ersten und zweiten Ordnung auf solche Intensitätsschwankungen untersucht. Einige Verfahren verwenden dabei als Ableitungsfunktion die Hesse-Matrix, um speziell markante Regionen im Bild hervorzuheben [47, 48, 49]. Erweiterungen die auf Basis einer Ableitungsfunktion erster Ordnung basieren, wurden durch [50, 51, 52] vorgestellt und führten zur Entwicklung des sogenannten *Harris-Detektor*, der sehr robust markante Eckpunkte hervorheben kann [53].

Farbwerte: Ähnlich wie bei den Bildintensitäten, verwenden diese Verfahren unterschiedliche Farbkanäle eines Bildes, um markante Regionen ausfindig zu machen [54, 55, 56]. Diese Erzeugen aus einem Farbbild eine sogenannte *Saliency Map*, die eine vereinfachte Farbabstraktion des Ursprungsbildes darstellt. Das heißt, das Pixel mit ähnlichen Farben zu einer Fläche zusammengefasst werden und dadurch markante Punktregionen, bei denen es einen sehr starken Farbunterschied gibt, hervorgehoben werden. Einige Verfahren setzten stattdessen auf eine erweiterte Variante des Harris-Detekors für den RGB-Farbraum ein [57].

Skalen- und Rotationsinvarianz: Die zuvor vorgestellten Verfahren haben den Nachteil, dass sie nicht zur Analyse von robusten Features ausgelegt sind. Robuste Features müssen auf unterschiedlichen Skalierungen oder Veränderungen des Blickwinkels eines Bildes zu finden sein. Bereits Anfang der 1980er wurde der Ansatz vorgestellt, Features auf Basis einer lokalen Extremwertsuche in der dreidimensionalen Nachbarschaft zu suchen [58]. Unter einer dreidimensionalen Nachbarschaft versteht man die umliegenden Pixel auf der eigenen sowie auf den direkt benachbarten Skalen eines Bildes. Dabei wird zur Erstellung der Skalen eine Bildpyramide aus dem zu untersuchenden Bild erzeugt (vgl. Kap. 2.2.1.3). Einige dieser Verfahren verwenden einen *Laplacian of Gaussian* (LoG), der einen Gaußfilter und den Laplace-Opertor auf ein Bild anwendet, um markante Kanten hervorzuheben [59, 60]. In [61] wurde gezeigt, dass man mit einem *Differential of Gaussian* (DoG) robustere Features bei

der Extremwertsuche erhält. Features müssen auch gegenüber affinen Transformationen robust sein. Diese entstehen z.B. sobald man ein Bild aus einem anderen Blickwinkel betrachtet. Um Robust gegenüber affinen Transformationen zu sein, verwenden die Verfahren entweder die Kanten [62] oder einzelne Gradientenwerte [63, 64] in der lokalen Umgebung eines Features.

2.2.4.2 Feature-Extraktion und Feature-Deskriptor

Der Schritt der Feature-Extraktion befasst sich mit dem Auslesen und der eindeutigen Beschreibung eines Features. Dies ist notwendig, um Features auf Basis ihrer spezifischen Eigenschaften und Beschaffenheiten miteinander vergleichen zu können. Features können wie eingangs erwähnt einfache Punkte, Kanten, Ecken oder ganze Flächen sein. Die Beschreibung der Lage oder Position anhand von Koordinaten reicht für die Beschreibung eines Features nicht aus. Die Umgebung oder die unmittelbare Nachbarschaft eignen sich eher für die Beschreibung eines Features. Die lokale Umgebung stellt eine ideale Grundlage zur Beschreibung dar, weil sie durch die enthaltenen Intensitätswerte und Pixelstrukturen besser unterscheidbar ist. Hierfür verwendet man Histogramme oder sogenannte Deskriptoren, die im Grunde aus einem n-dimensionalen Vektor bestehen. Lowe et al. verwenden einen Deskriptor, der aus den Intensitätswerten jedes einzelnen Pixels einer lokalen Umgebung besteht [61, 65]. Andere Ansätze betrachten anstatt der reinen Intensität die Kantendichte um das Zentrum eines Features und erhalten dadurch eine Verteilung, welche meist als Histogramm abgebildet wird [66]. Zur Beschreibung der lokalen Umgebung zählen auch die gemeinsamen Eigenschaften aller Elemente innerhalb einer zu beschreibenden Umgebung. So wird bei Verfahren, die eine Bildpyramide verwenden, die Skalenebene als gemeinsame Eigenschaft vermerkt [67]. Auch das gemeinsame Zentrum, die Größe des Features sowie die Form werden als gemeinsame Eigenschaft verwendet. In [68] wird zusätzlich die dominanteste Orientierung einer Umgebung betrachtet. Diese ergibt sich aus der Verteilung aller Intensitätswerte (Gradienten) innerhalb einer Umgebung. Die Orientierung ermöglicht es, ausgehend von ihr, einen rotationsinvarianten Deskriptor zu beschreiben.
Insgesamt können folgende Informationen zur Beschreibung eines Features und seiner lokalen Umgebung verwendet werden:

- **Histogramm:** Verteilung von Intensitäten, Kantendichte, Farbdichte ausgehend vom Zentrum eines Features.
- **Deskriptor:** Vektor zur Beschreibung jedes Pixelwertes (Intensität, Farbe) innerhalb einer lokalen Umgebung.
- **Lage/Position:** (x, y) Koordinate des Feature-Zentrums.
- **Größe:** Größe der Fläche eines Features. Oft wird die lokale Umgebung dazugerechnet.

- **Form:** Beschreibt die äußere Kontur eines Features. Wird durch eine Sequenz von Punktkoordinaten beschrieben.
- **Skalierung:** Ebene der Skale auf der die lokale Umgebung des Features liegt. Voraussetzung ist, dass das Verfahren eine Skalenpyramide einsetzt.
- **Orientierung:** Gibt die dominanteste Orientierung aller Gradientenwerte (Intensitäten) einer lokalen Umgebung an.

2.2.4.3 Matching Verfahren

Erst die Beschreibung eines Features ermöglicht seine Vergleichbarkeit mit anderen Features. Wie im vorherigen Abschnitt beschrieben, können unterschiedliche Eigenschaften dafür verwendet werden. Der Deskriptor eignet sich am besten für einen Vergleich. Die Robustheit und Beständigkeit eines Feature-Deskriptor ist durch seine lokale Umgebung gegeben. Zwei Deskriptoren lassen sich mit Hilfe einer Distanz-Vektor-Funktion einfach miteinander auf Eindeutigkeit oder auf eine gewisse Ähnlichkeit vergleichen. Verfahren, die eine Invarianz bezüglich der Skalierung und Rotation unterstützen, verstärken die Genauigkeit und Eindeutigkeit beim Vergleich.
Folgende zwei Distanzfunktionen werden am häufigsten beim Vergleich von Deskriptoren eingesetzt.

- **Euklidische Distanz:** Es beschreibt die Distanz d zweier Vektoren a und b im euklidischen Raum (n-dimensionaler Raum). Sie wird wie folgt berechnet: $d(a,b) = \sqrt{(a_1 - b_1)^2 + ... + (a_n - b_n)^2}$
- **Kosinus Distanz:** Es ist ein Ähnlichkeitsmaß, das anhand der Berechnung des Kosinus-Winkels $cos(\theta)$ zwischen zwei Vektoren a und b bestimmt wird. Dabei liegt der Winkel zwischen -1 (ungleich) und $+1$ (gleich). Sie wird wie folgt berechnet $cos(\theta) = \frac{a*b}{\|a\|\|b\|}$

Die einfachste Form des Vergleiches von mehreren Features ergibt sich durch eine solche Distanzfunktion. Das Feature-Pärchen mit der geringsten Distanz wird als ähnlich oder gleich betrachtet. Da die Distanz zwischen zwei Pärchen im seltensten Fall 0 (Euklidische Distanz) oder 1 (Cosinus Distanz) ist, benötigt man für die Aussage einer Ähnlichkeit eine obere Schranke, um zu große Distanzen auszuschließen. Die Schwierigkeit ist es einen geeigneten Wert für die obere Schranke zu finden.
Erweiterte Verfahren verwenden daher neben einer oberen Schranke T zusätzlich das Verhältnis zwischen den Distanzen der ähnlichsten n-Deskriptor-Pärchen [61, 69, 68, 70]. Sei f_1 das Referenz-Feature und f_2 und f_3 die zwei nächst ähnlichsten Features, dann wird die eigentliche Distanz wie folgt bestimmt,

$$distance_{ratio} = \frac{d(f_1, f_2)}{d(f_1, f_3)}$$

Durch die Berechnung des Verhältnisses beider Distanzen lässt sich unabhängig von der verwendeten Distanz-Funktion eine obere Schranke T einsetzen. Das heißt, liegt ein Wert unterhalb von $distance_{ratio} < T$, gelten zwei Features als gleich.

2.3 SURF Feature-Analyse und -Extraktion

Der *Speeded Up Robust Features* Algorithmus, abgekürzt SURF, von Bay et al. [68] dient im Rahmen dieser Arbeit als Grundlage für die hier vorgestellten Verfahren. Aus diesem Grund wird dem Feature-Analyse- und -Extraktions-Verfahren ein eigenes Kapitel gewidmet, da seine speziellen Eigenschaften für die vorgestellten Konzepte und Verfahren für ein besseres Verständnis erläutert werden müssen (vgl. Kapitel 3, 4 & 5).

Im Folgenden wird zunächst auf eine kurze Beschreibung des Algorithmus eingegangen. Dann werden die wichtigsten Attribute des Feature-Verfahrens erläutert. Anschließend wird die Analyse, die Extraktion sowie das *Matching* (Vergleich) des Verfahrens beschrieben. Abschließend wird auf eine spezielle Implementierung, die im Rahmen der gesamten Arbeit verwendet wird, eingegangen.

2.3.1 Überblick

Der *Speeded Up Robust Features* Algorithmus, kurz SURF, ist ein robustes Feature-Detektor- und Extraktions-Verfahren von lokalen Merkmalen (Punkte und Flächen) in einem Bild. Das ursprüngliche Verfahren, auf dem der Algorithmus basiert, ist der *Scale-invariant feature transform* Algorithmus, abgekürzt SIFT, von Loewe et al. [71, 65]. SIFT wurde als eines der ersten Feature Verfahren vorgestellt, das invariant gegenüber Rotationen, Skalierungen und Translationen ist. Die Invarianz gegenüber Transformationen hat den Vorteil, dass Merkmale auch dann erkannt werden können, wenn sie aus einer anderen Perspektive oder Distanz betrachtet werden.

Bei der Entwicklung von SURF war das Ziel, ein Verfahren zu entwickeln, das sich performanter und effizienter als das SIFT-Verfahren verhält und gleichzeitig nicht an Qualität und Güte verliert. Um dieses Ziel zu erreichen haben Bay et al. ihr Verfahren durch den Austausch von effizienteren Methoden sowie durch die Reduzierung von Informationen in der Feature-Analyse und Feature-Extraktion beschleunigt. Bei der Feature-Analyse wurden dabei Methoden eingesetzt, die eine schnelle Approximation einer Ableitungsfunktion und Extremwertsuche auf einem Integralbild ausführen (vgl. Abschnitt 2.3.3). Bei der Feature-Extraktion wurde einerseits die zu beschreibende Region eines Merkmales verkleinert und ebenfalls zur Beschreibung der Region die Integralbilddarstellung verwendet (vgl. Abschnitt 2.3.4).

Abbildung 2.6 zeigt ein Beispiel gefundener Features nach Anwendung des SURF-Verfahren. Dabei stellen die Kreise die Region eines Features dar und der schwarze Punkt den zugehörigen Mittelpunkt.

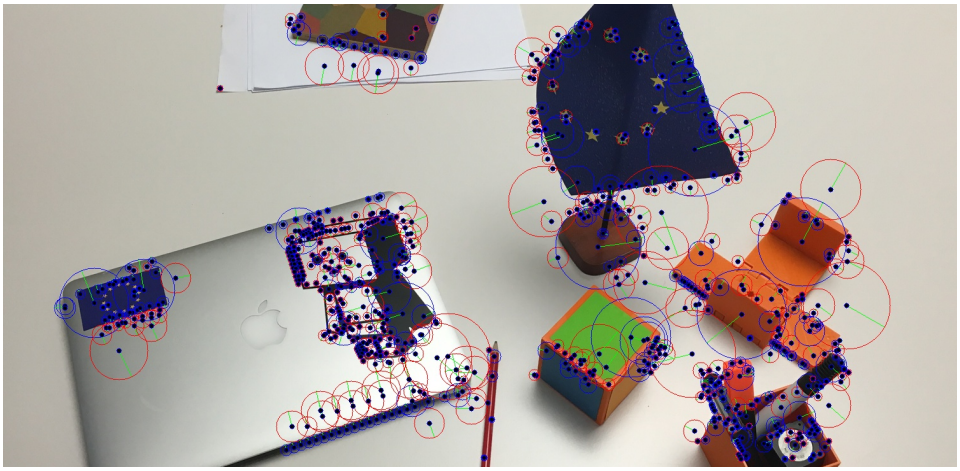


Abbildung 2.6: Anwendung des SURF-Algorithmus auf ein Bild. Die roten und blauen Kreise stellen die Regionen eines Features dar.

2.3.2 Attribute

Die Besonderheit des Feature Verfahrens sind die unterschiedlichen Attribute, die ein Feature fast eindeutig beschreiben. Dazu gehören neben der zweidimensionalen Koordinate auf dem Bild die *Scale*, die *Orientation* und der Deskriptor. Anhand der drei zuletzt genannten Attribute lassen sich Features miteinander robust vergleichen. Diese werden in der eigentlichen Feature-Extraktion ermittelt. Da jedoch diese Attribute im Rahmen dieser Arbeit für die unterschiedlichen Verfahren zum Einsatz kommen, werden diese kurz vorgestellt.

2.3.2.1 Bildkoordinate

Die Bildkoordinate stellt das einfachste zu beschreibende Attribut dar. Es gibt die Position eines Features an und wird als zweidimensionale (x, y)-Koordinate abgebildet. Die Bildkoordinate stellt zusätzlich den Mittelpunkt eines Features dar und dient als einziger Ankerpunkt zur Beschreibung der weiteren drei Attribute.
Allein durch die Bildkoordinate lassen sich die Features auf zwei unterschiedlichen Bildern mit dem selben Feature nicht vergleichen, da sich der Punkt bei unterschiedlichen Aufnahmewinkeln in der Distanz und Perspektive verschiebt und somit eine andere Bildkoordinate erhält.
Das Attribut gibt jedoch Auskunft darüber inwieweit sich ein Feature zum Referenz-Feature in der Position verändert hat. Anhand dieser Eigenschaft

können Bewegungen auf einem Bildpaar oder auf einer Bildsequenz in Richtung der horizontalen (x) und vertikalen (y) Bildachse festgestellt werden. Insbesondere die Bewegungen mehrerer Bildkoordinaten unterschiedlicher Features können Rückschlüsse auf die Transformation eines Bildpaares geben.

2.3.2.2 Scale

Die *Scale* gibt an, mit welcher Auflösung ein Feature in einem Bild gefunden wird. Da bei der Feature-Analyse zur Auffindung markanter SURF Features eine Extremwertsuche auf einer Bildpyramide ausgeführt wird, können die Extremwerte auf unterschiedlichen Ebenen der Pyramide liegen. Jede Ebene der Pyramide gibt dabei die momentane Skalierung des Bildes an. Das heißt, dass mit jeder steigenden Ebene die Skalierung des Bildes größer wird. Der *Scale*-Wert gibt somit die Ebene der Pyramide an. Die *Scale* gibt gleichzeitig darüber Auskunft wie groß die zu betrachtende Region eines Features auf dem aktuellen Bild ist [72].
Anhand dieser Eigenschaft kann das Attribut hergenommen werden, um aus einer Bildsequenz Informationen über die Bewegung abzuleiten. Das heißt Bewegungen in Richtung der Z-Achse eines Bildes, können durch die Veränderung der *Scale* wahrgenommen werden (vgl. Kapitel 5).

2.3.2.3 Orientation

Die *Orientation* gibt die dominanteste Ausrichtung des Helligkeitsverlaufes einer zur betrachtenden Region eines Features an. Dabei wird der Verlauf relativ zur X-Achse und die Ausrichtung in einem Winkelmaß angegeben. Dafür wird ein Kreis ausgehend von dem Mittelpunkt des Features betrachtet, dessen Radiusgröße abhängig von der *Scale* des Features ist.
Die Ermittlung der *Orientation* ist für einen robusten Vergleich von zwei Features von großer Bedeutung. Beim Vergleich von zwei eindeutig gleichen Features, von denen ein Feature um 90 Grad rotiert ist, kann anhand der *Orientation* die Rotation festgestellt werden. Vor jedem Vergleich werden daher alle Features auf die gleiche Ausrichtung der *Orientation* gedreht.
Die *Orientation* kann ähnlich wie die Bildkoordinate zur Ermittlung einer Bewegung hergenommen werden. Sie gibt Auskunft über die Rotation eines Bildes um die Z-Achse.

2.3.2.4 Deskriptor

Der Deskriptor beschreibt die eigentliche Region eines Features. Der Deskriptor ist ein 64-stelliger Vektor, der die Beschaffenheit und die unterschiedlichen Helligkeitswerte innerhalb dieser Region beschreibt. Die Größe der Region ist abhängig von der *Scale* und wird durch ein Quadrat umspannt, welche den Feature-Mittelpunkt als Zentrum hat.
Der Deskriptor wird zum Vergleich von Feature-Paaren verwendet. Anhand der

Berechnung des Abstandes zwischen zwei Vektoren, kann eine Aussage über die Gleichheit getroffen werden. Desto näher die Vektoren zueinander liegen, desto ähnlicher sind sie.

2.3.3 Feature-Analyse

Die Feature-Analyse des SURF-Verfahrens ist wie bei allen klassischen Feature Verfahren darauf ausgelegt markante Bereiche in einem Bild ausfindig zu machen (vgl. Abschnitt 2.2.4.1). Hierfür wird bei SURF eine Extremwertsuche auf einer Bildpyramide ausgeführt. Es setzt zur Extremwertsuche die Hesse-Matrix $H(X, \delta)$ ein, die für jeden Punkt eines Bildes mit dem entsprechenden *Scale*-Wert δ berechnet wird. Der *Scale*-Wert δ beschreibt die aktuelle Ebene des Bildes auf der Bildpyramide [73].

$$H(X, \delta) = \begin{bmatrix} L_{xx}(X, \delta) & L_{xy}(X, \delta) \\ L_{xy}(X, \delta) & L_{yy}(X, \delta) \end{bmatrix}$$

Dabei entsprechen die Funktionen L_{xx}, L_{yy} und L_{xy} jeweils einer Faltung der Gauß-Funktion zweiter Ordnung. Die Funktionen eignen sich sehr gut für eine Extremwertsuche innerhalb einer Bildpyramide mit ihren unterschiedlichen Ebenen.
Da für die Berechnung jeder Ebene aus der Bildpyramide die Gauß-Funktionen angewendet werden muss, ist die Extremwertesuche sehr rechenintensiv und kann sehr zeitaufwendig sein. Daher haben Bay et al. einen schnelleren Ansatz für die Berechnung der Gauß-Funktionen angewendet, die eine Approximation der genannten Funktionen darstellt.
Sie verwenden für die Approximation spezielle *Box-Filter*, die eine sehr schnelle Berechnung auf der Integralbilddarstellung des Bildes ermöglichen. Jede Faltung der Gauß-Funktion zweiter Ordnung wird durch einen 9 x 9 großen Box-Filter abgebildet (Abb. 2.7).

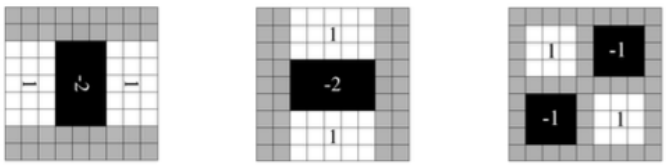


Abbildung 2.7: (Von rechts nach links) Approximation der Gauß-Funktionen zweiter Ordnung L_{xx}, L_{yy} und L_{xy} durch Box-Filter [68].

Jeder Box-Filter wird wie ein Fenster über das zu untersuchende Bild gelegt und Pixel für Pixel verschoben. Dabei liegt der zu betrachtende Pixel im Zentrum des Box-Filters. Es werden jeweils für jeden Bereich die Werte mit den darunterliegenden Intensitäten gewichtet und die Summe aller Werte

dem aktuellen Pixel zu geordnet. Durch die Integralbilddarstellung müssen dafür nicht alle Pixel innerhalb des Box-Filters einzeln betrachtet und gewichtet werden. Dies geschieht, indem man für jede unterschiedliche gewichtete Fläche des Box-Filters das Integral der darunterliegenden Pixel-Fläche mit dem Wert der gewichteten Fläche multipliziert (vgl. Abschnitt 2.2.3.2).
Da die Extremwertsuche mit der klassischen Hesse-Matrix auf jeder Ebene der Bildpyramide ausgeführt wird, muss dies auch bei der Anwendung der Box-Filter beachtet werden. Anstatt, dass man das Bild mit einem Gauß-Filter künstlich skaliert, wird stattdessen der Box-Filter vergrößert und über das selbe Bild gelegt. Hierfür erweitern die Autoren die Größe des Box-Filter jeweils für jede Ebene der Bildpyramide (Beginnend mit einer Größe von 9 x 9, 15 x 15, 21 x 21 usw.). Durch die höhere Pixeldichte ergibt sich auch hierdurch kein Mehraufwand bei der Anwendung der Box-Filter auf das Bild, weil dies durch den Einsatz der Integralbilddarstellung eingegrenzt wird.
Die eigentliche Extremwertsuche erfolgt in der sogenannten 3 x 3 x 3 Nachbarschaft, also auf der eigenen sowie auf der darüber und darunter liegenden Ebene. Da die Übergänge zwischen den einzelnen Ebenen der Bildpyramide durch den Box-Filter zu grob sind, werden bei der Extremwertsuche die Werte nochmals interpoliert nach der Methode von Braun et al. [74].
Die Autoren des SURF-Algorithmus bezeichnen ihr Feature-Analyse-Verfahren als sogenannten *Fast-Hessian*, da es eine approximierte Variante der Hesse-Matrix beinhaltet und dieses für die Feature-Analyse fast gleich gute Ergebnisse liefert.
Als endgültiges Ergebnis liefert die Feature-Analyse eine Sequenz aus mehreren Tupeln, die aus einer Bildkoordinate (x, y) und einer *Scale* δ bestehen. Diese werden anschließend für die darauffolgende Feature-Extraktion verwendet.

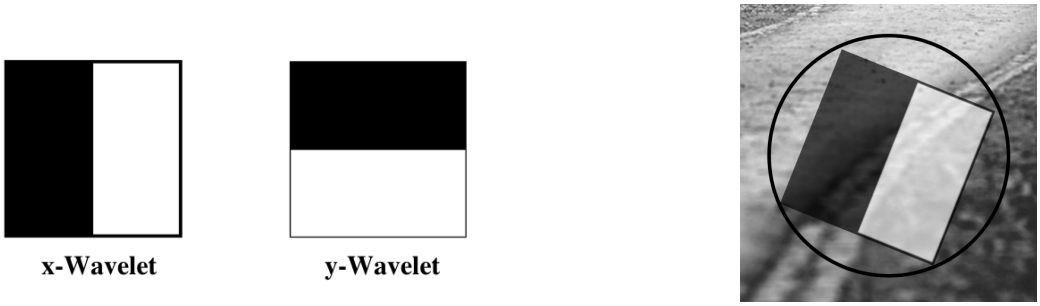
2.3.4 Feature-Extraktion und -Deskriptor

Das Verfahren zur Feature-Extraktion ist sehr stark angelehnt an die Variante von Loewe et al., welche in SIFT bereits umgesetzt ist [65]. Dies hat unteranderem den Hintergrund, dass sich kein anderes Verfahren besser verhält bezüglich der Robustheit und Performance [70] und daher der Einsatz eines zumindest vergleichbaren Feature-Extraktors und -Deskriptors für die Autoren unabdingbar war.
Zur Beschreibung des Deskriptors verwenden sie die Intensitätswerte (Gardienten) eines Features, da sich diese bei einer Veränderung der Skalierung, Rotation oder Translation relativ geringfügig unterscheiden.

Die Feature-Extraktion erfolgt in zwei Schritten, wobei als Eingabe des Verfahrens die Sequenz an Features aus der Feature-Analyse vorausgesetzt wird. Die zwei Schritte bestehen aus der Ermittlung der *Orientation* sowie aus der Extraktion des Deskriptors. Diese werden im Folgenden genauer beschrieben.

2.3.4.1 Ermittlung der Orientation

Die Ermittlung der *Orientation* ist für die Extraktion des Deskriptors und die damit verbundene Rotationsinvarianz sehr wichtig. Die *Orientation* ergibt sich aus der Ausrichtung des Helligkeitsverlaufes eines Features. Zur ihrer Bestimmung sind die Bildkoordinate (x, y) sowie die *Scale* δ eines Features notwendig. Zunächst wird ein Kreis um die Bildkoordinate mit einem Radius von 6δ gezogen, welche die Region des Features zur Ermittlung der *Orientation* eingrenzt. Anschließend werden zwei einfache *Wavelets* verwendet, die eine quadratische Fläche und eine Seitengröße von 4δ haben (vgl. Abb.2.8a). Diese zwei Wavelets dienen zur Bestimmung des Helligkeitsverlaufes entlang der x- und der y-Achse. Das Wavelet wird in das Zentrum des Features gelegt und mit Hilfe der Integralbilddarstellung die Summe der darunterliegenden Intensitätswerte gewichtet und bestimmt. Dies wird in $\frac{\pi}{3}$ Intervallen jeweils für das x- und y-Wavelet durchgeführt und die Summe aus den beiden Wavelet-Operationen dem Intervall zugeordnet (vgl. Abb. 2.8b).



(a) (b)

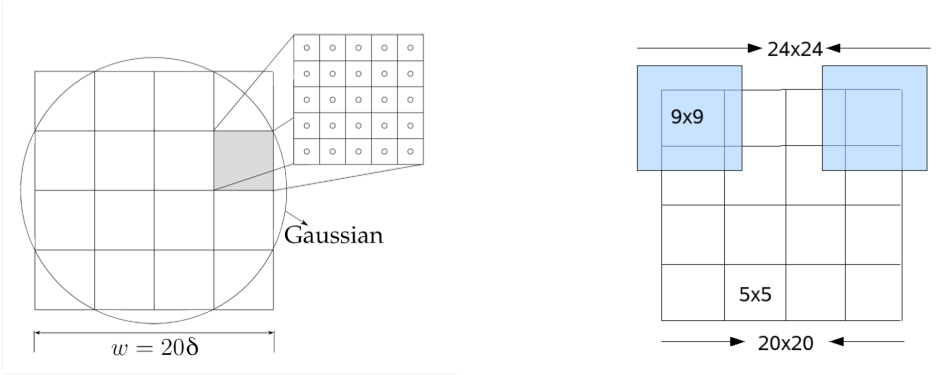
Abbildung 2.8: *Orientation*: (a) x-Wavelet (links) und y-Wavelet (rechts) [68] - (b) Anwendung eines x-Wavelets auf ein Feature.

Die Orientierung ergibt sich aus dem Intervall mit dem größten Wert. Die Auflösung der *Orientation* ist mit einer Quantisierungsstufe von $\frac{\pi}{3}$ gering, jedoch genügt sie, um eine eine hohe Robustheit gegenüber Rotationen eines Features zu erhalten [68].

2.3.4.2 Extraktion des Deskriptors

Die Extraktion des Deskriptors ist der zweite Schritt der allgemeinen Feature-Extraktion. Dabei wird zunächst ein Quadrat um die Bildkoordinate des Features gelegt. Dieses wird entsprechend der *Orientation* rotiert. Die Größe des Quadrates beträgt 20δ (vgl. Abb. 2.9a). Dieses wird wiederum aufgeteilt in 4 x 4 gleich große quadratische Sub-Flächen. Aus jeder Sub-Fläche wird die Verteilung aller Intensitäten betrachtet. Diese werden mit denen in der *Orientation* eingesetzten x- und y-Wavelets ermittelt. Die Wavelets haben jeweils eine Größe von 9δ und ragen damit an den Rändern um 2δ heraus (vgl. Abb. 2.9b). Die

Wavelets werden im folgenden zur Vereinfachung d_x und d_y genannt Jedes Wavevlet wird zusätzlich gewichtet. Dazu wird ein Gaußfilter verwendet, welcher ausgehend vom Zentrum des Features (Bildkoordinate) angewendet wird.



(a) (b)

Abbildung 2.9: *Orientation*: (a) Aufteilung der Feature-Fläche mit der Größe von 20δ in 4 x 4 Sub-Flächen. Der Kreis symbolisiert den Gaußfilter zur Gewichtung der Sub-Flächen - (b) Wavelets (blau) mit der Größe von 9δ werden auf die Sub-Flächen angewendet um die Intensitätsverteilung zu berechnen [75].

Für jede Sub-Fläche wird nun die Summe aus den d_x und d_y berechnet. Neben der Information der allgemeinen Verteilung der Intensität einer Sub-Fläche wird die Polarität der Intensitätsunterschiede betrachtet. Diese ergibt sich aus den Summen der Absolutbeträge $|d_x|$ und $|d_y|$. Insgesamt ergeben sich für jede Sub-Fläche vier Werte $\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|$.
Der Deskriptor wird aus den Werten der 4 x 4 Sub-Flächen zusammengestellt. Damit ergibt sich ein 64-stelliger Vektor, der das Feature beschreibt.

2.3.5 Matching

Für das Matching, das Vergleichsverfahren von Features, verwenden Bay et al. den Deskriptor, die *Scale* und die *Orientation* eines jeden einzelnen Features. Bevor der eigentliche Vergleich stattfindet, wird das Vorzeichen des Laplace-Operators zweier Feature-Paare betrachtet. Dieses erhält man ohne zusätzlichen Aufwand aus dem Prozess der Feature-Analyse während der Extremwertsuche. Das Vorzeichen besagt ob ein helles Feature auf einem dunklen Hintergrund liegt oder umgekehrt. Dadurch lassen sich bereits im Vorfeld nicht gleiche Features ausschließen.

Sobald man alle offensichtlichen Features aussortiert hat, beginnt der eigentliche Vergleich. Hierfür werden die Deskriptoren eines Feature-Paares verwendet, die mit Hilfe der Euklidischen Distanz verglichen werden (vgl. Abschnitt 2.2.4.3). Die Deskriptoren werden dazu vor dem Vergleich abhängig

von ihrer *Orientation* in eine gemeinsame Richtung rotiert, sodass der Distanzvergleich invariant gegenüber Rotationen des Features ist.
Statt sich auf einen diskreten Wert für eine Distanz von zwei gleichen Deskriptoren fest zu setzen, haben die Autoren eine geeignetere Lösung eingesetzt. Sie betrachten nicht nur die kleinste Distanz eines Deskriptor-Paares aus einer Menge von Vergleichs-Deskriptoren und einem Referenz-Deskriptor, sondern zusätzlich die Distanz des zweit kleinsten Deskriptor-Paares, wobei der Referenz-Deskriptor immer derselbe ist.
Dabei gehen die Autoren von der Annahme aus, dass der Abstand zwischen der ersten und zweiten Distanz zweier Deskriptor-Paare immer unterhalb einer gleich definierten Grenze liegen muss, damit zwei Deskriptoren gleich oder ähnlich sind. Um den Abstand zu bestimmten, nehmen sie das Verhältnis zwischen der ersten und zweiten Distanz. Sie haben mit einer empirischen Untersuchung bewiesen, dass ein Verhältnis unterhalb von 0,65 eine Ähnlichkeit von zwei Deskriptoren erfüllt [73].

2.3.6 Unterschiedliche Implementierungen

Nach der Veröffentlichung des SURF Feature-Verfahrens im Jahre 2006, entstanden bereits sehr schnell die ersten Implementierungen. Die Umsetzungen unterscheiden sich dabei in ihrer Performance, Genauigkeit und Lizenzierung. Daher werden die wichtigsten unterschiedlichen Implementierungen kurz vorgestellt und anschließend die Wahl, für die im Rahmen dieser Arbeit gewählten Implementierung, begründet.

2.3.6.1 OpenCV SURF

Diese Variante der SURF Implementierung ist Bestandteil der bekannten und großen Computer Vision Bibliothek *OpenCV* [76]. Die Bibliothek ist größtenteils quelloffen und beinhaltet nur einige Module, die für die kommerzielle Nutzung nicht verwendet werden dürfen. Dazu zählt die Original-Implementierung des SURF-Algorithmus, die unter Patent steht. Trotzdem kann die Variante zu Forschungszwecken für die nicht-kommerzielle Nutzung verwendet werden. Der Basis-Code ist in C++ implementiert. Sowohl der Teil der Feature-Analyse als auch der Feature-Extraktion werden als separate Module angeboten. So lassen sich unterschiedliche Feature-Verfahren in Analyse und Extraktion beliebig variieren.
Für die Feature-Extraktion bietet die Bibliothek eine Implementierung an, die für eine Berechnung auf der Grafik-Prozessor-Einheit (GPU) ausgelegt ist.
Allgemein ist das Verfahren sehr performant und kann daher auf mobilen Endgeräten, wie z.B. einem Smartphone, in Echtzeit ausgeführt werden.

2.3.6.2 JavaSurf

Ursprünglich wurde der SURF-Algorithmus in der Programmiersprache C++ geschrieben, da zahlreiche komplexe Rechenoperationen ausgeführt werden müssen und das Verfahren trotzdem performant bleiben soll. Da heutzutage die meisten komplexen Rechenoperationen in einer Java-Laufzeitumgebung (JRE) fast die selben Rechenzeiten benötigen, stellt eine Umsetzung des SURF-Algorithmus in Java-Code kein Hindernis mehr dar. Besonders die Tatsache, dass viele andere SURF-Implementierungen auf die *OpenCV* Bibliothek zurückgreifen, macht eine einfache Einbindung von SURF-Implementierungen in eigene Verfahren schwieriger. Daher stellt die *JavaSurf* Implementierung eine gute Alternative in Java-Code dar [77]. Sie lässt sich einfach in bestehende Java-Projekte integrieren und kann daher ohne weitere Bibliotheken und anderen Abhängigkeiten verwendet werden.
Nachteilig an dieser Variante sind jedoch einige Veränderungen an der vorgeschlagenen Referenz-Implementierung [78]. Es werden einige Parameter verändert, um schnellere Ausführungen zu erzielen, die jedoch zu weniger robusten Ergebnissen führen.

2.3.6.3 CUDA SURF

CUDA SURF [79] ist eine Implementierung speziell für die *CUDA Plattform* von *NVIDIA* [80]. Die Implementierung ist auf das Berechnen paralleler Operationen ausgelegt. Diese sind insbesondere bei der Extraktion der Deskriptoren sinnvoll, da ein paralleles Auslesen und Verarbeiten möglich ist.
Durch die Parallelisierung ist die Ausführung von SURF extrem schnell und echtzeitfähig. Insbesondere für den Einsatz von hochauflösenden Bildern oder ganzen Bildsequenzen stellt diese Variante eine geeignete Lösung dar. Nachteil der Implementierung ist die Abhängigkeit von der *CUDA Plattform* sowie die Voraussetzung *NVIDIA*-eigene Hardware einsetzen zu müssen. Besonders auf mobilen Endgeräten führt dies zu Schwierigkeiten und macht damit die Nutzung dieser Variante der SURF-Implementierung obsolet.

2.3.6.4 OpenSURF

OpenSURF ist eine weitere Variante der SURF-Implementierung [72]. Sie ist in der Programmiersprache C++ geschrieben und nutzt einige Methoden und Operationen der *OpenCV* Bibliothek. Diese Variante unterscheidet sich teilweise von der Referenz-Implementierung bei der Anwendung von einigen Bildoperationen [78]. Dadurch, dass die Implementierung die Feature-Extraktion etwas anders berechnet, erhält man robustere Deskriptoren als in der Referenz-Implementierung. Dabei liegt der Unterschied ausschließlich beim Auslesen der einzelnen Deskriptor-Werte. In der Referenz-Implementierung werden die Intensitätswerte innerhalb des Quadrates, das die Fläche des Features umspannt, abhängig von der *Orientation* interpoliert. Das Verfahren von *OpenSURF* ro-

tiert das Quadrat vorher, so dass die *Orientation* in Richtung der positiven y-Achse zeigt. Dies hat zwar den Nachteil, dass die Ausführung weniger performant ist, jedoch die Ergebnisse beim Matching besser ausfallen.

2.3.6.5 Zusammenfassung

Alle genannten Verfahren der SURF-Implementierung haben ihre Vor- und Nachteile und sind speziell für die jeweiligen Anwendungsfälle angepasst. Dabei kann die Implementierung von *OpenCV SURF* beim Vergleich von unterschiedlichen Feature-Verfahren schnell eingesetzt und mit anderen Verfahren einfach kombiniert werden.
Die Java-Implementierung eignet sich dagegen für das schnelle *Prototyping* von Lösungen, die auf ein Feature-Verfahren angewiesen sind. Eine Portierung auf mobile Endgeräte ist schnell möglich, auch wenn die Performance im Vergleich zu den anderen Implementierungen schlechter ist. Des Weiteren ist diese Variante im Vergleich zu anderen Implementierungen weniger robust und fehleranfällig.
Die auf der *CUDA Plattform*-basierende Implementierung eignet sich für sehr große und stationäre Systeme, die viele Ressourcen zur Verfügung stellen können. Dabei ist der Einsatz dieser Variante eher für komplexere und hochauflösende Bilder und Bildsequenzen geeignet.
Die *OpenSURF*-Implementierung stellt eine einfache und ähnliche Variante zur der Referenz-Implementierung dar. Sie hat jedoch den Vorteil, dass sie im Gegensatz zur Referenz-Implementierung Open-Source ist. Damit lassen sich Konstrukte im Code besser verstehen und eventuell anpassen. Des Weiteren verfügt diese Variante über eine verbesserte Feature-Extraktion.

Im Rahmen dieser Arbeit stellt die *OpenSURF*-Implementierung die Variante mit den meisten Vorzügen dar und wird deshalb, für die in dieser Arbeit vorgestellten Verfahren, eingesetzt.

2.4 Anwendungsgebiete des maschinellen Sehens

In den letzten dreißig Jahren haben sich die Verfahren im Bereich des maschinellen Sehens immens weiterentwickelt. Besonders Verfahren zur Erkennung von Objekten, die eine Beschreibung, eine Vermessung, Lagebestimmung und Klassifizierung vornehmen, werden häufig eingesetzt, um Entscheidungen für automatisierte Prozesse zu bestimmen. Aber auch das tatsächliche Verstehen und Deuten von Bildinhalten und ganzen Bildszenen hat sich in diesem Bereich sehr stark weiterentwickelt [81].

Allgemein werden Verfahren des maschinellen Sehens sehr erfolgreich im industriellen Umfeld eingesetzt. Besonders zur Kontrolle von Produkten und

ganzen Prozessabläufen werden zahlreiche Methoden der Bildverarbeitung eingesetzt [82]. Diese rasante Entwicklung im industriellen Bereich hat dazu beigetragen, dass ganze Forschungsabteilungen an neuen Lösungen und Erweiterungen visueller Verfahren entwickeln und diese bereits erfolgreich in ihre Produkte integrieren [83, 84, 85].
Die größte Schwierigkeit stellt der Einsatz von solchen Verfahren in natürlichen Umgebungen dar. Viele Verfahren sind für den Einsatz in industriellen Umgebungen entwickelt worden, in denen die Umgebung eigens für das Verfahren angepasst und einer ständigen Kontrolle unterzogen wird. Dies führte gerade im letzten Jahrzehnt zu einem Umdenken bei der Entwicklung solcher Verfahren und ganzer Systeme. Besonders im Bereich der Automobilindustrie werden viele Verfahren für das sichere Fahren in natürlichen Umgebungen wie dem Straßenverkehr weiterentwickelt und mit Hinblick auf das autonome Fahren verbessert [86, 87, 88].
Im Folgenden werden die wichtigsten Anwendungsgebiete des maschinellen Sehens durchleuchtet und wichtige Systeme aus diesen Bereichen vorgestellt.

2.4.1 Visuelles Tracking

Das Verfolgen von einzelnen Punkten oder ganzen Objekten innerhalb eines Bildverlaufes bezeichnet man als visuelles Tracking. Dabei nutzen die Verfahren meist eine Videokamera, um Objekte und ihre Positionen kontinuierlich in einer Bildsequenz zu verfolgen.
Trackingverfahren sind Teil anderer Anwendungsgebiete des maschinellen Sehens. So wird es z.B. in der Objekterkennung zur eindeutigen Identifizierung beweglicher Objekte verwendet. Im Bereich der *Augmented Reality* wird es zur Bestimmung der eignen Lage oder zur stabilen Projektion auf Objekten im Raum eingesetzt. Nicht zuletzt wird es auch im Bereich der visuellen Odometrie zur Eindämmung von Abweichungen bei der Bestimmung der Trajektorie verwendet.

Grundsätzlich wird ein Tracker eingesetzt, um die Betrachtungswinkel der Kamera sowie die Lage mehrerer Punkte oder eines ganzen Objektes im Raum möglichst genau und in Echtzeit zu bestimmen. Die Kamera kann dabei fest an einer Position montiert sein oder sich selber mobil bewegen.
Nach Li et al. kann ein Trackingverfahren in folgende Schritte unterteilt werden [89]:

- **Initialisierung:** Dieser Schritt kann automatisiert oder manuell ausgeführt werden. Dabei wird entweder der Bereich um ein Objekt einmalig selbst definiert oder man setzt Objekterkennungsverfahren ein, die das Objekt automatisch finden (z.B. Gesichtserkennung).

- **Bestimmung des Erscheinungsbildes:** Dieser Schritt lässt sich durch zwei Varianten realisieren. Die erste Variante versucht mit Feature-

Verfahren markante Bereiche des Objektes für eine robuste Beschreibung zu detektieren. Die zweite Variante nutzt ein mathematisches Model zur Beschreibung des Objektes.

- **Bewegungsabschätzung:** Um ein Objekt auch über einen längeren Zeitraum in Echtzeit verfolgen zu können, benötigt man Verfahren zur Abschätzung der Bewegung, wie z.B. einen Kalman-Filter [90], Partikel Filter [91] oder die lineare Regression [92]. Diese haben den Vorteil, dass man durch die Eingrenzung des zu untersuchenden Bereiches weniger Ressourcen benötigt und dadurch eine Echtzeitfähigkeit des Trackings garantiert werden kann.

- **Lokalisierung:** Dieser Schritt beinhaltet die finale Suche und Auswahl nach dem zu verfolgenden Objekt. Dies wird entweder durch einen *Greedy-Algorithmus* oder durch einen Schätzer, wie z.B. einem Bayes-Schätzer, gelöst [93].

2.4.1.1 Beispiele für Tracking-Systeme

Im Folgenden werden einige Beispiele für Tracking-Systeme vorgestellt. Dabei unterscheiden sie sich im Einsatzgebiet und in ihrer Umsetzung.

Überwachungssysteme Der häufigste Einsatz von Tracking-Systemen ist im Bereich der Videoüberwachung von Personen. Dabei kommen meist fest montierte Kameras zum Einsatz, wie man sie in Kauf- und Parkhäusern auffindet. Dadurch, dass die Kamera fest montiert ist, kommen häufig *Background Subtraction* Methoden zum Einsatz [94], die den statischen Hintergrund vom Vordergrund trennen (vgl. Abschnitt 2.2.2.1). Neben der simplen *Background Subtraction* kommen andere Segmentierungs-Methoden zum Einsatz, wie z.B. die Konturfindung oder das Template Matching [95]. Die segmentierten Bereiche im Vordergrund werden anschließend als Objekte wahrgenommen und über die fortlaufende Bildsequenz verfolgt [96]. Jedes zu sehende Objekt wird dabei eigenständig verfolgt. Die Kamera kann dabei aus der Vogel- oder aus einer vertikal erhöhten Perspektive auf die zu untersuchende Szene gerichtet sein (vgl. Abb. 2.10).

Motion Capture Unter dieser Form des Trackings versteht man die Erfassung und Aufzeichnung einer Bewegung eines Objektes. Es wird verwendet um natürliche Bewegungen eines Menschen oder eines Tieres in ein digitales Format zu übertragen, damit es digital reproduziert werden kann. Da besonders menschliche Bewegungsabläufe sehr schwer am Computer nachzubilden sind, hat sich das Verfahren des Motion Capture besonders im Bereich des Animationsfilms sehr etabliert. Dabei werden aufgezeichnete Bewegungsabläufe auf animierte Charaktere oder Avatare abgebildet.

Abbildung 2.10: Aufzeichnung eines visuellen Tracking-Systems: (Von links nach rechts) Personen werden über die drei Bilder hinweg erkannt und verfolgt. Jedes Objekt wir mit einer schwarzen Elipse und einem eindeutigen Index (gelb) gekennzeichnet [96].

Für eine realitätsnahe und vollständige Aufzeichnung von Bewegungsabläufen kommen meist mehrere Kameras zum Einsatz, die aus unterschiedlichen Perspektiven die Bewegung aufzeichnen. Zum Tracken der Person sowie der einzelnen Gelenke werden je nach Feingranularität unterschiedliche Verfahren eingesetzt. Dabei verwenden die Verfahren entweder Marker (künstliche Markierungen) oder natürliche Merkmale (natürliche Muster oder Konturen) um Bewegungen einer Person ableiten zu können [97, 98].

Abbildung 2.11: Beispiel eines Motion Caputres mit Markern. Die weißen Kugeln an der Person werden zur Aufzeichnung der Bewegung verwendet [99].

Das Tracking mit Markern ist für die Aufzeichnung von Bewegungsabläufen die stabilste Variante. Diese werden von einem visuellen Verfahren einfacher erkannt (vgl. Abb. 2.11). Die Verfahren werden sogar soweit eingesetzt, dass einzelne Gesichtsgesten eines Menschen erfasst werden und dadurch animierte Figuren fast menschlich wirken können.

2.4.2 Objekterkennung

Die visuelle Objekterkennung ist ein Verfahren zu Identifizierung und Lokalisierung von Objekten innerhalb eines Bildes. Objekte werden dabei anhand ihrer charakteristischen Eigenschaften identifiziert. Diese charakteristischen Eigenschaften ergeben sich aus der Kontur, der Oberflächeneigenschaft, der Farbverteilung, der Größenrelationen und anhand markanter Merkmale eines Objektes.
Im Prinzip lassen sich die unterschiedlichen Verfahren in drei Klassen unterteilen, nämlich in:

Methoden mit invarianten Eigenschaften Diese Methoden beruhen darauf, dass sie markante Eingenschaften eines Objektes in einem Bild ausfindig machen. Dabei kann das Bild im Vorfeld segmentiert worden sein, sodass die einzelnen Segmente betrachtet werden. Die Eigenschaften sind meist invariant bezüglich Transformationen, speziell einfache Rotationen und Translationen. Solche Eigenschaften können einfache Farbwerte und Farbhäufungen, Konturen oder klassische Features sein. In [100] verwenden Swain et al. ein Farben-Histogramm, um ganze Objekte innerhalb eines Bildes ausfindig zu machen. Wolfson et al. setzten dagegen das *Geometric Hashing* ein, welches die Punkte einer Kontur so einsetzt, dass sie bei einer Rotation, Translation oder bei einer Teilmenge der Punkte ein Objekt wieder erkennen kann [101].

Template Matching Dieses Verfahren stellt eines der bekanntesten Möglichkeiten zur Objekterkennung dar. Das Template Matching nutzt dabei ein einfaches Graustufenbild als Referenz eines Objektes (Template) und sucht damit Pixel für Pixel anhand einer Übereinstimmungsfunktion nach dem gegebenen Objekt innerhalb eines Bildes (vgl. Abschnitt 2.2.2.2). Wichtig für die Robustheit des Verfahrens ist, dass das Template nur das zu suchende Objekt beinhaltet und keine anderen Informationen abbildet. Variationen des Template Matching skalieren das Template unterschiedlich und rotieren es um seine eigene Achse bei der Suche des Objektes innerhalb des Bildes. Dadurch wird zusätzlich zur Translationsinvarianz eine Rotations- und Skaleninvarianz geschaffen. Ein großer Nachteil des Verfahrens ist die fehlende Invarianz gegenüber unterschiedlichen Kontrastverhältnissen. Im Gegensatz zu der zuvor genannten Klasse, nutzt das Template Matching nicht nur einzelne Eigenschaften eines zu suchenden Objektes, sondern verwendet das gesamte Bild zum Vergleich.

Klassifizierung Die Klassifizierung ist ein Verfahren zur Einteilung von unterschiedlichen Objekten in verschiedene Kategorien. Dabei können mehrere unterschiedliche Eigenschafen zur Definition einer Klasse verwendet werden. So lassen sich die Konturpunkte, Farbwerte, Intentsitätswerte sowie die Gesamtheit der Feature usw. zu einem Mustervektor zusammenfassen. Damit ist

es möglich ein Objekt in einem Bild auch bei einem unvollständigen Mustervektor wieder zuerkennen.
Die Klassifizierung wird im Alltagsgebrauch nicht für das Auffinden eines einmaligen Objektes verwendet, sondern eher für das Auffinden einer Klasse von Objekten. Das heißt, man sucht mit Klassifikatoren z.B. nach Menschen, Autos oder Blumen. Dadurch, dass sich ähnliche Objekte auch untereinander unterscheiden können, muss der Mustervektor anders gebildet werden. Im einfachsten Fall bildet man aus dem Mittelwert aller Mustervektoren einer Klasse einen gemeinsamen Mustervektor. Bessere Verfahren, wie die Hauptkomponentenanalyse, transformieren anschließend den gemeinsamen Mustervektor in einen niedriger-dimensionalen Unterraum, in dem der Hauptteil der Datenvariation liegt [102]. Der Mustervektor wird dadurch auf seine markantesten Muster mit der höchsten Varianz zu einander reduziert. Neuere Verfahren verwenden dagegen *neuronale Netze*, die zwar rechenaufwendiger und komplexer sind, jedoch in ihrer Genauigkeit und Trefferquote besser sind [103].

2.4.2.1 Beispiele der Objekterkennung

Im Folgenden werden drei Fallbeispiele der Objekterkennung vorgestellt.

Gesichtserkennung Diese Form der Objekterkennung wird besonders in den letzten 10 Jahren immer häufiger eingesetzt. So wird die Gesichtserkennung speziell für Überwachungssystme an viel frequentierten Orten, wie z.B. U-Bahn oder Stadtzentrum, eingesetzt [104]. Dabei werden die Verfahren zur automatisierten Auffindung von strafrechtlich verfolgten Personen eingesetzt. Ein weiteres Beispiel der Gesichtserkennung ist die Passkontrolle an einigen Flughäfen, bei denen ein Bildscanner das Gesicht identifiziert und eindeutig einer Person zuordnet [105].

Fahrzeugerkennung Die Erkennung von Fahrzeugen wird schon seit vielen Jahren in der Verkehrsüberwachung eingesetzt. Meist werden die Verfahren zur Detektion und Verfolgung von Fahrzeugen auf der Autobahn eingesetzt [106, 107]. Seltener werden die Verfahren zur Analyse von innerstädtischen Verkehrssituationen verwendet [108].
Ein spezieller Anwendungsbereich der Fahrzeugerkennung beschäftigt sich mit der automatischen Erkennung von Nummernschildern, die gerade für Mautsysteme auf kostenpflichtigen Straßen eingesetzt werden [109].
In der Regel wird die Fahrzeugerkennung zur Zählung durchfahrender Fahrzeuge in Parkhäusern oder Straßen eingesetzt sowie zur groben Klassifizierung in PKW, LKW und Motorräder [110]. Neuere Arbeiten, wie z.B. in [111, 112], versuchen die unterschiedlichen Fahrzeugmodelle zu klassifizieren und erreichen damit eine Erkennungsrate von über 90%.

2.4.3 Visuelle Positionierung und Odometrie

Die visuelle Positionierung und die spezielle Form der visuellen Odometrie haben in den letzten Jahren zunehmend das Interesse in der Forschung und der Industrie geweckt. Dies ist mit ein Grund, weshalb diese Methoden im Bereich des maschinellen Sehens einem eigenständigen Anwendungsgebiet zugeordnet werden.
Ein Vorteil dieser Methoden und Verfahren gegenüber herkömmlichen Positionierungssystemen ist, dass auf eine externe Infrastruktur zu großen Teilen verzichtet werden kann. Visuelle Informationen sind überall vorhanden und können durch die weite Verbreitung von mobilen Endgeräten mit Kamerasensoren genutzt werden.
Das Grundprinzip der visuellen Positionierung basiert darauf, dass man anhand eines Anfragebildes seine aktuelle Position bestimmt, indem es mit Bildern, deren Positionen absolut oder relativ bekannt sind, verglichen wird. Zum Vergleich der Bilder werden markante Merkmale verwendet.
Für diese Form der Positionierung gibt es unterschiedliche Variationen, die sich allgemein in drei Kategorien unterteilen lassen. Diese werden im Folgenden kurz vorgestellt.

Positionierung mittels Standortdatenbank Die erste Kategorie von Positionierungssystemen nutzt eine sogenannte Standortdatenbank, die im Vorfeld erzeugt wird und aus mehreren Bildern, die einen Ort oder Raum abbilden, besteht. Zusätzlich beinhaltet sie in der Regel eine Reihe von weiteren Datensätzen, die aus einer Bildbeschreibung des Ortes und einer relativen oder absoluten Koordinate bestehen können. Zur Ermittlung der eigenen Position wird ein Bild des aktuellen Aufenthaltsortes mit der Standortdatenbank verglichen. Zur Beschreibung der Bilder werden üblicherweise visuelle Features verwendet, die rotations- und skaleninvariant sind [113, 114, 39]. Eine detailliertere Beschreibung solcher Verfahren wird in Kapitel 4 nochmals gegeben.

Visuelle Odometrie Odometrie bezeichnet ein Verfahren zur Bestimmung der Lage und Position eines sich bewegenden Systems. Die Lage und Position wird anhand der Informationen des Antriebssystems bestimmt. Die visuelle Odometrie, welche eine speziellere Form der Odometrie ist, umfasst alle Verfahren, die durch Bildinformationen aus einem Kamerasystem Rückschlüsse auf die Eigenbewegung ziehen. Die ersten stabilen System der visuellen Odometrie, wurden erstmals für die Mond- und Mars-Rovermissionen entwickelt und eingesetzt [115].
Grundsätzlich existieren verschiedene Ansätze zur Umsetzung der visuellen Odometrie. Die meisten Verfahren setzten auf eine 3D-Rekonstruktion der Umgebung durch binoculare Kamerasysteme [116]. Neuere Verfahren setzen dagegen monoculare Kamerasysteme zur Gewinnung von Tiefeninformationen ein [117]. Häufiger jedoch werden Verfahren mit einer monocularen Kamera

zur Analyse der Bildbewegung auf der zweidimensionalen Bildfläche eingesetzt [118]. Eine detailliertere Beschreibung solcher Verfahren wird in Kapitel 5 nochmals gegeben.

Simultaneous Localization and Mapping Die dritte Kategorie setzt eine hybride Lösung ein, die aus einer Kombination der beiden vorherigen Kategorien besteht. Die Verfahren lokalisieren sich und erzeugen dabei synchron eine Standortdatenbank, die aus Bildern und ihrer geschätzten Position bestehen [119, 120]. Die Abschätzung der Position erfolgt nicht immer durch visuelle Odometrie, sondern kann auch durch andere Odometrie-Verfahren umgesetzt werden. Durch das kontinuierliche Aktualisieren der Standortdatenbank mit aktuelleren Bilder und neu geschätzten Positionen, wird die eigene Position mit der Zeit genauer bestimmt und korrigiert.

2.4.4 Augmented Reality

Unter *Augmented Reality* oder auch *erweiterte Realität* (kurz AR) versteht man die Erweiterung der Realitätswahrnehmung durch computergestützte Systeme. Die unterschiedlichen Kanäle der Wahrnehmung können dabei visuell, auditorisch, haptisch und olfaktorisch sein. Die visuelle Wahrnehmung stellt einen der wichtigsten Kanäle dar. Es kommen zahlreiche Methoden des maschinellen Sehens zum Einsatz, die für eine robuste Wahrnehmung der erweiterten Realität Sorge tragen.
Folgende Kriterien müssen laut Azuma für ein AR-System gelten, damit es eine erweitere Realität unterstützen kann [121]:

1. Reale und virtuelle Inhalte müssen gemeinsam auf einem Bild zu sehen sein. Das heißt, es können virtuelle Objekte in die reale Welt eingefügt werden oder reale Objekte ausgeblendet werden.

2. Das System muss interaktiv in Echtzeit sein. Die Erweiterung der Realität muss zeitgleich mit der Wahrnehmung der realen Umgebung stattfinden. Es dürfen keine Latenzen auftreten.

3. Virtuelle Objekte müssen im 3D-Raum registriert sein. Sie werden abhängig von der Position der Kamera und ihrer Orientierung in der realen Welt verankert und werden mit jeder neuen Kameratransformation neu gerendert.

2.4.4.1 Beispiele für AR-Systeme

Anwendungsgebiete für AR-Systeme gibt es zahlreiche. So werden sie in der Industrie zur Konstruktion und Reparatur komplexer Systeme eingesetzt. In der Medizin werden sie als Operationsunterstützung verwendet. Bei unterschiedlichen Transportmitteln, wie z.B. dem Flugzeug oder Auto, wird diese

Technologie in Form von sogenannten *Head-up-Displays* (HUD) bereitgestellt, die Informationen visuell und kontaktanalog auf einer durchsichtige Scheibe darstellen. AR-Systeme werden vermehrt im Bereich des Sports und in Live-Berichterstattungen genutzt. Dabei werden Ziellinien beim Rudern oder Werbeflächen bei Fussball-Veranstaltungen in Echtzeit eingeblendet, so dass man den Unterschied zwischen realen und virtuellen Objekten kaum wahrnehmen kann.
Im Folgenden werden zwei unterschiedliche AR-Systeme aus verschiedenen Anwendungsbereichen vorgestellt.

AR im America's Cup Der *America's Cup* stellt Weltweit einen der größten Höhepunkte der Segel-Sportsaison dar. Da die Regeln recht kompliziert zu verstehen sind und der Verlauf auf dem Wasser mit den unterschiedlichen Segelbooten etwas unübersichtlich ist, haben sich die Initiatoren für eine Lösung mit einem AR-System entschieden. Sie blenden über die Live-Bilder virtuelle Linien und Objekte ein und informieren dadurch den Zuschauer über die aktuellen Ereignisse auf dem Wasser.
Allgemein werden dadurch unterschiedliche Vorteile erreicht. Die Boote können durch die Einblendung virtueller Flaggen unterschieden werden. Die zu segelnden Pfade der einzelnen Boote werden auf dem Wasser eingezeichnet. Die Geschwindigkeit der Boote wird durch einen Farbverlauf hervorgehoben.

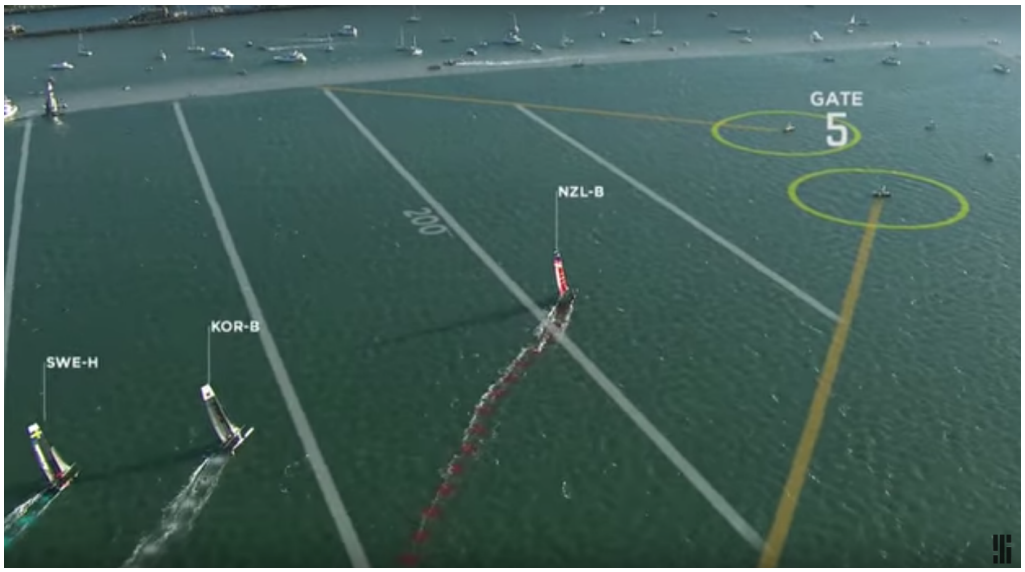


Abbildung 2.12: Einsatz von AR-Technologie zum Americas's Cup. Der Segelverlauf jedes Bootes wird farblich markiert. Einzeichnung von Zonen, Bereichen und Grenzen [122].

Um solch ein AR-System realisieren zu können, müssen neben der Analyse des Kamerabildes zusätzliche Sensoren und weitere Kamerasysteme eingesetzt werden. Dabei werden die Positionen der Boote zusätzlich durch Lagesensoren

oder GPS ermittelt.

IKEA Katalog App Das Einrichtungshaus *IKEA* hat im Jahr 2013 eine Anwendung für das Smarphone entwickeln lassen, dass einzelne Möbel aus dem Katalog des Möbelhauses interaktiv im eigenen Zuhause darstellen kann [123]. Dabei wird das entsprechende Möbelstück mit der Anwendung ausgewählt, und anschließend wird es auf dem Videobild der Kamera angezeigt. Das Möbelstück lässt sich in Echtzeit verschieben und im Raum realistisch abbilden. Das AR-System wird allein durch die Sensoren des Smartphones, also dem Lagesensor und der Kamera, bereitgestellt. Die Anwendung erlaubt es dem Nutzer, sich im Raum mit dem Smartphone frei zu bewegen, sodass man das virtuelle Objekt aus allen Perspektiven betrachten kann (vgl. Abb. 2.13).

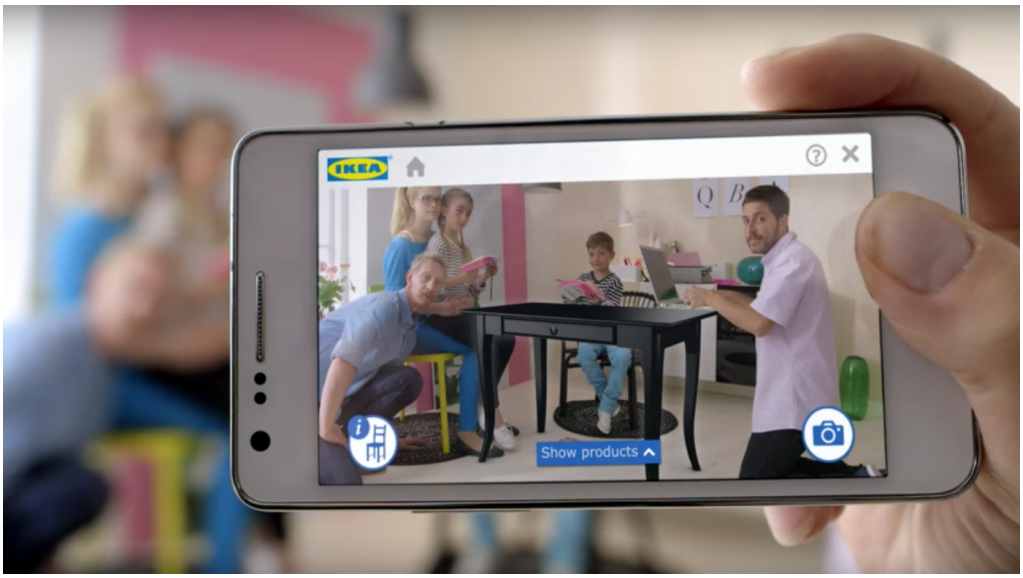


Abbildung 2.13: Anwendungsbeispiel der *IKEA* AR-App: Ein virtueller Tisch wird in die Szenarie, welche durch die Samrtphonekamera aufgezeichnet wird, eingeblendet [123].

2.5 Zusammenfassung

Dieses Kapitel beschäftigte sich mit der Vorstellung der Grundlagen aus dem Bereich des maschinellen Sehens. Zunächst wurden unterschiedliche Bildrepräsentationen vorgestellt und erläutert. Anschließend wurden die wichtigsten Verfahren und Methoden aus dem Bereich der Bildverarbeitung näher gebracht. Hierfür wurden visuelle Filter, Verfahren der Segmentierung, verschiedene Verfahren der Transformation sowie visuelle Features vorgestellt. In Anschnitt 2.3 wurde das visuelle Feature-Verfahren *SURF* genauer beschrieben und vorgestellt, da dieses im Rahmen der vorliegenden Arbeit eine große

Bedeutung für die Entwicklung der eigenen Verfahren hat. Abschließend wurde ein Überblick existierender und forschungsnaher Systeme aus dem Bereich des maschinellen Sehens vorgestellt. Dafür wurden die Anwendungsgebiete des visuellen Tracking, der Objekterkennung, der visuellen Positionierung sowie der Augmented Reality beschrieben und anhand von Beispielen erläutert.

3 Effizientes Vergleichsverfahren von Bildinformationen

Um ein Bild interpretieren zu können, neigen wir als Mensch dazu, die einzelnen Objekte, die darin abgebildet sind, zu identifizieren und in einen Gesamtkontext zu stellen. Dadurch schaffen wir es gesehene Bilder und die darin enthaltenen Kontexte wieder zuerkennen und zu verstehen. Auch dann wenn sich einige Details zu dem aktuellen Bild unterscheiden [124]. So können wir ein menschliches Gesicht anhand von typischen Merkmalen wie einer Nase, einem Mund und den Augen identifizieren. Um ein Gesicht endgültig als solches zu interpretieren, verknüpfen wir die gesehenen Merkmale zu einem für uns verständlichen Bild. Solche Merkmale werden anhand ihrer auffälligen Oberfläche, Farbe, Kontur oder aus einer Kombination aus den genannten Merkmalen beschrieben.
Im Bereich des maschinellen Sehens spielen Bildvergleiche ebenfalls eine entscheidende Rolle. Sie sind wichtige Schlüsselverfahren, um ein Bild analysieren und interpretieren zu können. Besonders bei der Entwicklung von visuellen Verfahren, die speziell in autonomen Systemen eingesetzt werden, sind Bildvergleiche für viele Bedingungen und die damit zu treffenden Einscheidungen sehr wichtig.
In der Automobilindustrie werden bereits heute zahlreiche visuelle Verfahren eingesetzt, um Teilbereiche des Fahrens autonom oder assistiert zu gestalten. So werden z.B. bei Spurassistenten die Fahrbahnmarkierungen durch Vergleichsverfahren analysiert, um beim Überqueren der Fahrbahnline den Fahrer rechtzeitig hinzuweisen [86]. Bildvergleiche werden auch bei der Erkennung von Verkehrsschildern eingesetzt, um dem Fahrer eine einfache Übersicht im meist unüberschaubaren Stadtverkehr zugeben [125].
Auch im Rahmen dieser Arbeit ist der Vergleich von Bildern ein wichtiges Schlüsselverfahren. Insbesondere der Vergleich von visuellen Features, die zur Beschreibung der Bilder verwendet werden, ist für die Entwicklung der vorgestellten Verfahren sehr wichtig. Für das Verfahren in Kapitel 4 und Kapitel 5 ist der Einsatz eines effizienten und genauen Vergleichsverfahren von entscheidender Bedeutung. Das Verfahren muss dabei je nach Einsatz unterschiedliche Anforderungen erfüllen und je nach geforderter Genauigkeit immer noch effizient und robust bleiben. In diesem Kapitel wird daher ein mehrstufiger Vergleichsprozess vorgestellt, der diese Anforderungen erfüllt.

Das Verfahren wird dabei im Rahmen des Systems SURFLogo, welches eine eigens entwickelte verteilte Anwendung aus dem Bereich des *Mobile-Tagging* ist, eingesetzt und evaluiert.
Im Folgenden werden existierende Vergleichsverfahren vorgestellt und klassifiziert. Dadurch wird ein Querschnitt auf die unterschiedlichen Herangehensweisen auf die Thematik des Bildervergleichs basierend auf visuellen Features gegeben. Anschließend wird ein eigener Ansatz eines mehrstufigen Vergleichsverfahrens gegeben, der mit jeder weiteren Stufe an Genauigkeit zunimmt. Dieses mehrstufige Vergleichsverfahren wird in das Konzept von SURFLogo integriert, welches im Anschluss vorgestellt und evaluiert wird. Die Evaluation gibt Aufschlüsse über die Güte des mehrstufigen Vergleichsverfahrens. Abschließend wird eine Zusammenfassung über das gesamte Kapitel geben.

3.1 Eigene Vorveröffentlichungen

Die Kerninhalte dieses Kapitels wurden vom Autor bereits in [13] publiziert. Wie in Kapitel 1.3 dargestellt, stammen die im Paper und im Folgenden präsentierten Inhalte bzgl. der Idee, des Konzepts und der Evaluation vom Autor der vorliegenden Arbeit. Ebenfalls im Paper bereits enthalten sind die Abbildungen 3.4, 3.6, 3.7, 3.8, 3.9, 3.10, 3.11, 3.12a und 3.12b.

3.2 Klassifizierung existierender Verlgleichsverfahren

Wie in Kapitel 2 bereits beschrieben, gibt es unterschiedliche Möglichkeiten und Abstraktionsebenen, auf deren Basis man Bilder miteinander vergleichen kann. Das Feature-Verfahren SURF (vgl. Abschnitt 2.3) hat sich als geeignetes Verfahren zur Bildanalyse herauskristallisiert. Sowohl zur Beschreibung und zum Vergleich von Bildern als auch durch seine performante Ausführbarkeit auf mobilen Endgeräten, wie z.B. Smartphones, ist das Verfahren durch seine Rotations- und Skaleninvarianz sowie seiner Robustheit gegenüber unterschiedlichen Lichtverhältnissen sehr gut geeignet. Daher wird im Rahmen dieser Arbeit das Feature-Verfahren SURF eingesetzt. Der Fokus der unterschiedlichen Klassen von Vergleichsverfahren wird im Folgenden auf Feature-basierten Ansätzen liegen.
Allgemein wird der Vergleich von Bildern als *Image Retrieval* bezeichnet. Es gibt zwei Klassen von Vergleichsverfahren, die im Bereich visueller Features verwendet werden: Die erste Klasse vergleicht direkt auf Basis der Features eines Bildes, während die zweite Klasse über die Menge aller Features durch eine Quantisierung prototypische Merkmale berechnet und diese dafür einsetzt. Im Folgenden werden die zwei Klassen genauer vorgestellt.

3.2.1 Vergleich auf visuellen Features

Der direkte Vergleich auf visuellen Features beinhaltet immer einen Vektorvergleich. Der Vektor ist durch den Deskriptor, welcher die Umgebung eines Feature beschreibt, gegeben. Zu den einfachsten Verfahren, die zu dieser Klasse zählen, gehören die sogenannten Voting-Verfahren. Jedes Feature eines Anfragebildes votiert für das Bild, das über die Menge aller Vergleichsbilder, die größte Ähnlichkeit hat. Hierfür wird die Distanz zwischen den Deskriptoren betrachtet und ein bestimmter Schwellwert für eine Mindestähnlichkeit definiert. Statt der größten Ähnlichkeit kann auch alternativ jedes Feature für ein Bild voten, das innerhalb eines definierten Wertebereichs liegt. Das Bild mit den meisten Votes wird als das ähnlichste Bild ausgewählt. Die Voting-Verfahren zählen zu den *Nearest Neighbor* Verlgeichsverfahren [126, 127]. Abbildung 3.1 veranschaulicht wie entscheidet die Wahl einer geeigneten Schwellwertgröße der Mindestähnlichkeit ist. Wählt man einen zu kleinen Schwellwert, kann im ungünstigsten Fall die Falsch-Negativ-Rate ansteigen. Im umgekehrten Fall kann bei einem zu großen Schwellwert die Falsch-Positv-Rate ansteigen.

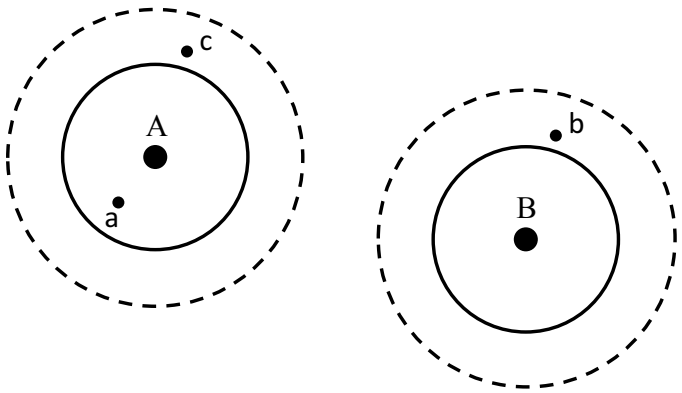


Abbildung 3.1: Nearest Neighbor Verfahren [124] - Die Großbuchstaben A und B sind zwei Features, die mit Features aus einer Menge von Vergleichsbildern (Kleinbuchstaben a, b, c) verglichen werden. Die Kreise stellen jeweils einen Schwellwert dar. Die durchgezogene Line enspricht einem kleinen Schwellwert, die gestrichelte Linie einem großen Schwellwert. Der kleine Schwellwert ist für A ausreichend, aber für B zu klein, sodass das ähnliche Feature b ausserhalb ligt. Bei einem größeren Schwellwert liegen zwar beide ähnlichen Features a und b innerhalb, jedoch wird auch das Feature c als ähnlich zu A betrachtet.

Die Schwierigkeit, einen geeigneten Schwellwert oder Wertebereich für eine Mindestähnlichkeit zu finden, wird durch die Verfahren der *Nearest Neighbor Distance Ratio* besser gelöst [71, 69, 70, 68]. Bei dieser Art von Verfahren wird,

statt ausschließlich das Feature mit der größten Ähnlichkeit zu betrachten, auch das Feature mit der zweitgrößten Ähnlichkeit beim Vergleich zwischen zwei Features verwendet (vgl. Abb. 3.2). Aus dem Verhältnis beider Ähnlichkeiten des jeweiligen Feature-Paares erhält man ein Maß zur Bestimmung der Güte der tatsächlichen Ähnlichkeit eines Features. Dadurch, dass man das Verhältnis als Ähnlichkeitsmaß verwendet, lässt sich ein geeigneter Schwellwert finden, der abhängig von der Verteilung der Ähnlichkeiten jedes einzelnen Features innerhalb des Bildes ist und somit für unterschiedliche Klassen von Bildern einsetzbar ist.

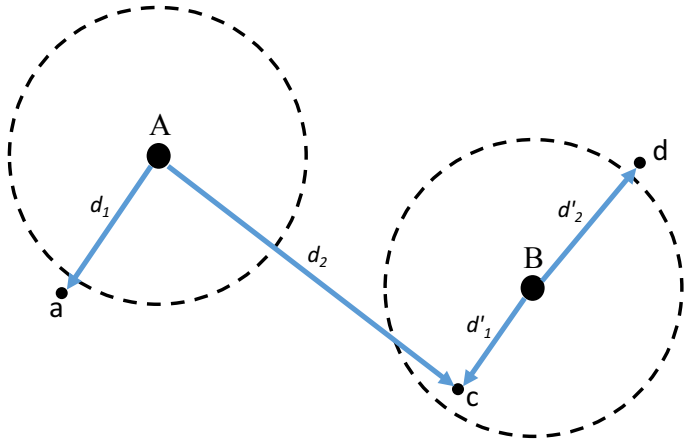


Abbildung 3.2: Nearest Neighbor Distance Ratio [124] - Bei einem festen Schwellert (gestrichelte Linie), wie es beim Nearest Neighbor Verfahren der Fall ist, würde das Feature A kein ähnliches Feature finden und Feature B dagegen eine Ähnlichkeit mit Feature c fälschlicher Weise aufweisen. Nimmt man die Verhältnisse der Distanzen d_1/d_2 und d'_1/d'_2, werden die korrekten Features für A und B gefunden.

In der Regel wird ein Verhältnis verwendet, dass eine ausreichende Distanz zu einander bestimmt. Das Bild, das die meisten ähnlichen Features zu dem Anfragebild aufweist, wird als Gewinner gewählt. Das heißt, dass die Anzahl an ähnlichen Features für ein Vergleichbarkeitsmaß zwischen zwei Bildern entscheidend ist.

Omercevic et al. gehen dabei einen Schritt weiter und betrachten zusätzlich die Abstandsverteilung einer Menge von nächst näheren Nachbarn [128]. Dabei werden die zu vergleichenden Features so gewichtet, dass nicht ausschließlich die Ähnlichkeit, sondern auch die Unterscheidbarkeit zu einer Menge nächster Nachbarn berücksichtigt wird. Das Verfahren betrachtet für ein Feature q eine große Menge an k nächsten Nachbarn v_i mit $(i = 1, ..., k)$. Dabei muss das k so gewählt sein, dass es eine realistische Menge zutreffender

Features bei weitem übertrifft. Im Normalfall liegen die nächsten 10 bis 20 Features in einem solchen realistischen Bereich. Die Autoren empfehlen daher einen Wert mit $k = 100$. Zur Berechnung der Ähnlichkeit zwischen jedem Tupel (q, v_i) verwenden sie das Skalarprodukt, das sie anschließend normalisieren. Die k nächsten Nachbarn werden absteigend ihrer Ähnlichkeit sortiert. Jeder Nachbar wird nachträglich mit einer Wahrscheinlichkeitsfunktion, die abhängig von der Ähnlichkeitsverteilung der Menge ist, gewichtet. Durch die Betrachtung einer größeren Menge und der zusätzlichen Gewichtung, erhält man genauere Ergebnisse, jedoch leidet die Performanz darunter immens. Diese Variante zählt zu den sogenannten *Meaningful Nearest Neighbors* Verfahren.

3.2.2 Vergleich auf prototypischen Merkmalen

Allgemein weisen die zuvor vorgestellten Verfahren, die auf Basis ihrer Features einen Vergleich ausführen, einen Nachteil bezüglich ihrer Laufzeit auf. Bei einem Vergleich von zwei Bildern wird jedes Feature m_i des einen Bildes M mit jedem Feature n_j des anderen Bildes N verglichen. Der eigentliche Vergleich wiederum wird anhand der Deskriptoren beider Features durch eine Abstandsfunktion D bestimmt. Die Summe aller gleichen Features gibt die Ähnlichkeit beider Bilder an.

$$\sum_{i,j} D(m_i, n_j) \text{ , mit } D(m,n) = \begin{cases} 1, & \text{falls } m \approx n \\ 0, & \text{sonst} \end{cases}$$

Wird jedoch ein Referenzbild mit mehr als einem Bild verglichen, ist das Vergleichsverfahren nicht mehr praktikabel, da der Grundvergleich bereits zu aufwendig ist. Besonders, da die Anzahl an Features bei jedem Bild variiert und diese je nach Konfiguration zwischen 100 und 1000 liegen kann. Um diesem Problem entgegenzuwirken, vergleichen die Verfahren der zweiten Klasse auf effizientere Weise, indem sie die Vergleiche zwischen zwei Features durch Quantisierung der Deskriptoren im Voraus berechnen. Das heißt, dass ein Bild nicht mehr anhand aller seiner Features verglichen wird, sondern durch einen Deskriptor, der repräsentativ für alle Features eines Bildes steht.
Zur Umsetzung der Quantisierung werden aus einer großen Menge verschiedener Features *Cluster* gebildet. Jedes Feature aus einem Bild wird einem Clusterzentrum mit dem geringstem Abstand zugeordnet. Demnach wird jedes Bild durch ein Histogramm von Clusterhäufigkeiten repräsentiert. Insgesamt wird dadurch eine effizientere Vergleichbarkeit mit anderen Bildern erreicht. Da viele dieser Konzepte ihren Ursprung im Bereich des *Dokumentenretrieval* haben, werden in Analogie dazu die Clusterzentren als *visuelle Wörter*, die Menge aller Wörter als *Vokabular* und die Bildrepräsentation durch das Histogramm als *Bag-of-Words* bezeichnet. Im Folgenden werden die einzelnen Prozesse der Quantisierung genauer erläutert. Dabei wird zunächst der Prozess zum Erzeugen eines geeigneten Vokabulars beschrieben, das für den nächsten Prozess

zur Bildung einer Bag-of-Words Repräsentation eines Bildes benötigt wird. Zusätzlich werden Verfahren zur Gewichtung der Bag-of-Words Repräsentationen vorgestellt.

3.2.2.1 Vokabular

Um ein Vokabular zu erzeugen, benötigt man eine große Menge unterschiedlicher Features. Diese können aus einer Menge von Vergleichsbildern oder einer großen Bilderdatenbank bestehen. Anschließend wird diese Menge geclustert. Dabei ist das Ziel des Clustering eine möglichst distinktive Teilmenge zu partitionieren, sodass ähnliche Features demselben Cluster zugeordnet werden und sich von Features aus anderen Clustern stark unterscheiden. Die Zentren der Cluster entsprechen den prototypischen Merkmalen, also den visuellen Wörter, und bilden damit das Vokabular für die Bag-of-Words-Repräsentation.
Sivic et al. waren eine der Ersten, die ein solches Verfahren zur Quantisierung von Features einsetzten [129]. Als Bilddatenbank verwendeten sie alle Frames aus zwei kompletten Kinofilmen. Als Cluster-Algorithmus setzten sie eine Variante des *k-Means* ein [130], welcher auf einer Untermenge der Bilddatenbank angewendet wurde. Idealerweise verwendet man die gesamte Menge aller Features aus der Bilddatenbank, jedoch steigt die benötigte Zeit zur Clusterbildung immens an. Besonders bei großen Feature-Mengen und hochdimensionalen Deskriptoren entsteht dieser Effekt. Daher bietet es sich an eine Untermenge zur Clusterbildung zu verwenden. Es muss darauf geachtet werden, dass die Untermenge zufällig ausgewählt ist und die Features möglichst aus repräsentativen Bildern der Gesamtmenge bestehen.
Wegen seiner Einfachheit wird häufig der *k-Means*-Algorithmus zur Clusterbildung der Feature-Menge verwendet. Das Verfahren wählt zu Beginn k zufällige Features aus der zu clusternden Menge aus und definiert sie als vorläufige Clusterzentren. Nun werden die restlichen Features den Clusterzentren mit der geringsten Distanz zugeordnet. Anschließend werden aus den gebildeten Clustern neue Clusterzentren definiert, welche sich aus dem Mittel aller Features eines Clusters berechnen lassen. Die letzten beiden Schritte werden mehrmals iteriert bis sich die Clusterzuordnung jedes Features nicht mehr ändert.
Durch den Einsatz des *k-Means* Algorithmus entstehen jedoch einige Probleme, die es zu lösen gilt. So muss z.B. die Anzahl k der Cluster im Voraus bestimmt werden. Wählt man ein zu kleines k, lassen sich die Features nur schwer unterscheiden. Ein Großteil der Informationen kann durch ein zu klein gewähltes k verloren gehen, so dass unähnliche Features durch das selbe Wort repräsentiert werden. Wird dagegen ein zu großes k gewählt, leidet die Performanz des Vergleichsverfahrens darunter, da die Bag-of-Words-Deskriptoren sehr lang werden und der Prozess zur Zuordnung jedes einzelnen Features zu einem Wort sehr zeitaufwendig wird. Die Anzahl k hängt von zwei Faktoren ab. Nämlich von der Anzahl an Vergleichsbildern und von der Varianz der in den Bildern abgebildeten visuellen Strukturen. Die Anzahl an Vergleichsbildern ist entscheidend für k, da bei sehr vielen Vergleichsbildern und den vielen Featu-

res mehr Wörter benötigt werden, um deren charakteristische Eigenschaften geeignet abbilden zu können. Beim zweiten Faktor muss bei sehr ähnlichen visuellen Strukturen auf den Vergleichsbildern ein größeres k gewählt werden, da die Unterscheidbarkeit sehr schwierig wird. So benötigten z.B. Bilder von Gebäudegängen ein größeres k (mehr Wörter) zur Unterscheidung, da sie meist sehr ähnlich sind. Sind die Bilder dagegen sehr komplex und unterschiedlich in ihrer Struktur, ist ein kleineres k (weniger Wörter) zur Kategorisierung ausreichend.

Ein weiteres Problem beim *k-Means* Algorithmus ergibt sich aus der Wahl der Startpunkte für die Cluster-Konstellation zu Beginn des Clusterings. Da diese zufällig ausgewählt werden, beeinflusst dies die Qualität des Vokabulars. So bietet z.B. der *k-Means++* einen Lösungsansatz für die Auwahl einer geeigneten Wahl von Startpunkten [131]. Das Verfahren hat jedoch den Nachteil sehr zeitaufwendig zu sein. Alternativ bietet es sich an mehrere Durchläufe des einfachen *k-Means* durchzuführen und anhand eines Test-Sets von Vergleichsbildern die beste Cluster-Konstellation zu evaluieren.

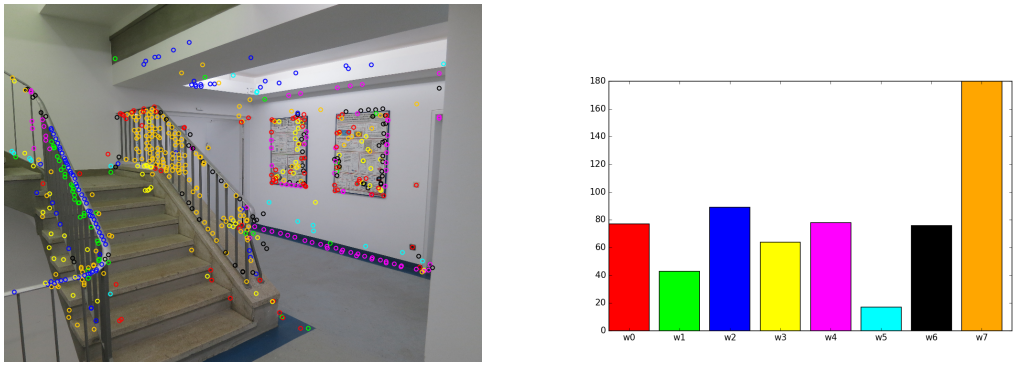
3.2.2.2 Bag-of-Words Repräsentation

Nachdem ein passendes Vokabular erzeugt wurde, lässt sich jedes Bild durch Quantisierung in eine Bag-of-Words Repräsentation berechnen. Die Quantisierung geht dabei wie folgt vor. Jedes Feature eines Bildes wird dem ähnlichsten visuellen Wort aus dem Vokabular zugeordnet. Dadurch erhält man eine Verteilung von Worthäufigkeiten für jedes Bild (vgl. Abb. 3.2.2.2). Die Verteilung lässt sich auch als Vektor darstellen, indem man jedem Wort w im Vokabular einen Index i zuweist, wobei $i \in (0, ..., k)$ gilt und k die Anzahl an Clustern wiedergibt. Ein Bild wird in der Bag-of-Words Repräsentation wie folgt beschrieben:

$$BoW = (b_1, b_2, ..., b_k)$$

Die Stelle b_i gibt die Häufigkeit der Features an, die einem visuellen Wort w_i aus dem Vokabular zugeordnet sind.

Durch die Quantisierung und die daraus resultierende Bag-of-Words Repräsentation wird ein Bild, das durch 100 bis 1000 Features beschrieben wird, auf einen Vektor der Länge k reduziert. Damit ergeben sich enorme Verbesserungen im Hinblick auf die Geschwindigkeit bei den Bildvergleichen, da nur noch ein Vektor pro Bild statt mehrere Features und ihre Deskriptoren betrachtet werden müssen. Ein Nachteil der Quantisierung ist der bereits erwähnte Verlust von Informationen. Dies hat jedoch auch den Effekt, dass Features, die z.B. als Bildrauschen wahrgenommen werden, einem einzelnen Word zugeordnet werden und dadurch sich einfacher identifizieren lassen.



(a) (b)

Abbildung 3.3: Bag-of-Words Repräsentation eines Bildes: (a) Bild-Features, die Wörtern zugeordnet sind; hier durch verschiedene Farben gekennzeichnet - (b) Histogramm der Worthäufigkeiten. Das Vokabular besteht aus acht Wörtern.

3.2.2.3 Wortgewichtung

Durch die Häufigkeit der Zuweisungen eines Features zu einem Wort sowie durch die allgemeine Beobachtung des Vokabulars, lassen sich charakteristische Eigenschaften identifizieren. Diese geben z.B. Rückschlüsse darüber ob die Menge der Features nur einfaches Bildrauschen oder sehr markante und einzigartige Strukturen abbilden. So sind Wörter, die selten vorkommen, relevanter für eine eindeutige Vergleichbarkeit von Bildern als welche, die in allen Bildern häufig vorkommen. Daher ist eine Gewichtung der Wörter ein gutes Werkzeug, um solche Anomalien auszugleichen. Es lassen sich Analog zum Dokumentenretrieval Gewichtungsschema auf die Bag-of-Words Repräsentation einsetzten. Einer der bekanntesten Gewichtungsschema stellt das *term-frequency-inverse-document-frequency* (tf-idf) dar [129, 114]. Diese lässt sich wie folgt auf eine Bag-of-Words Repräsentation $(b_1, ..., b_k)$ ansetzen, wobei k die Clustergröße des Vokabulars angibt:

$$b_i = \frac{n_{i,B}}{n_B} log \frac{N}{1 + n_i}$$

Dabei ist $n_{i,B}$ die Anzahl an Vorkommen des Wortes w_i im Bild B. n_B ist die Summe aller Features im Bild B. N ist die Summe aller Features aller Vergleichsbilder und n_i die Anzahl an Vorkommen des Wortes w_i in allen Vergleichsbildern. Durch die Normierung der Vorkommenshäufigkeit *(term frequency)* $\frac{n_{i,B}}{n_B}$ werden Wörter hoch gewichtet, die häufig in einem Bild vorkommen. Dagegen werden durch die inverse Dokumenthäufigkeit *(inverse document frequency)* $log \frac{N}{1+n_i}$ Wörter hoch gewichtet, die insgesamt über alle Vergleichsbilder nur selten auftreten. Das heißt, dass dadurch Wörter hoch gewichtet werden, die ein Bild gut beschreiben.

3.2.2.4 Bag-of-Words Vergleich

Der Vergleich zweier Bilder anhand ihrer Bag-of-Words Repräsentation ist letztendlich ein Vektorvergleich beider Histogramme, die eine Häufigkeitsverteilung ihrer Features, bezogen auf das Vokabular, beschreiben. Zum Vergleich von zwei Bildern müssen folgende Schritte ausgeführt werden:

- Extraktion aller Features aus einem Bild.
- Erzeugung der Bag-of-Words Repräsentation. Jedes Feature wird einem Wort zugeordnet.
- *(optional)* Gewichtung der Bag-of-Words Repräsentation nach *tf-ifd*.

Die Ähnlichkeit zweier Bilder wird durch Berechnung der Vektordistanz ihrer beiden Bag-of-Words Repräsentationen bestimmt. Zur Berechnung der Distanz eignen sich die Euklidische- sowie die Kosinus-Distanz (vgl Abschnitt 2.2.4.3). Klassischerweise werden bei Vergleichen mit einem Referenzbild Bilder aus einer bestehenden Datenbank verwendet. Um den Vergleichsprozess zu beschleunigen, lassen sich die Bag-of-Repräsentationen für die Bilder aus Datenbank bereits im Vorfeld berechnen.
Insgesamt ist durch die Quantisierung nur noch eine Vergleichsoperation notwendig. Damit ergibt sich eine lineare Komplexität des Vergleichsalgorithmus bezüglich der Anzahl an Vergleichsbildern mit einem Bild. Zwar wird dadurch die Effizienz des Vergleichsverfahrens gesteigert, jedoch steigt damit auch die Fehlerrate durch die Quantisierung gegenüber dem Vergleichsverfahren basierend auf den visuellen Features.

3.3 Effizientes Vergleichsverfahren - Mehrstufiger Vergleichsprozess

Da im Rahmen dieser Arbeit der Vergleich von Bildern eine große Bedeutung hat, ist ein Vergleichsverfahren, dass einerseits schnell und andererseits eindeutig ist sehr wichtig. Die meisten existierende Verfahren aus dem Bereich der visuellen Feature-Verfahren verwenden ausschließlich eine charakteristische Eigenschaft zum Vergleich von Bildern. Dies führt dazu, dass die Qualität eines Bildvergleiches allein von der Beschaffenheit dieser charakteristischen Eigenschaft abhängt. Um diesem Problem entgegenzuwirken wurde ein eigenes Vergleichsverfahren entwickelt, welches insbesondere für die Systeme aus Abschnitt 3.4 sowie aus Kapitel 4 und Kapitel 5 eingesetzt wird. Dieses Verfahren ist effizient und kann je nach eingestellter Konfiguration Bilder eindeutig wiedererkennen. Das Verfahren ist dabei mehrstufig aufgebaut und setzt für jede Stufe ein anderes Vergleichskriterium ein. Die Stufen bauen aufeinander auf und lassen sich je nach Tiefe abstellen. Dadurch lässt sich der Grad der Genauigkeit einstellen sowie die Effizienz je nach Anforderung regeln. Insgesamt

wurden drei unterschiedliche Stufen entwickelt und in das System SURFLogo, welches in Abschnitt 3.4 vorgestellt wird, integriert und evaluiert. Das Vergleichsverfahren nutzt als Verarbeitungsgrundlage und zur Beschreibung eines Bildes visuelle Features, hierfür speziell den SURF-Algorithmus (vgl. Abs. 2.3). Im Folgenden wird der Vorverarbeitungsschritt für das Vergleichsverfahren erläutert. Anschließend wird der eigentliche mehrstufige Vergleichsprozess vorgestellt, indem die einzelnen Stufen der Reihenfolge nach beschrieben werden.

3.3.1 Vorverarbeitung

Das Vergleichsverfahren basiert auf insgesamt drei Stufen. Die erste Stufe nutzt die Bag-of-Words Repräsentation als Grundlage für den Vergleich. Die zweite Stufe vergleicht auf Basis der visuellen SURF-Features. Die letzte Stufe vergleicht auf Basis der Kontur, die durch die Position der Features gegeben ist. Bevor jedoch der eigentliche Vergleich stattfindet, unterliegt jedes Bild einem Vorverarbeitungsschritt. Abbildung 3.4 zeigt die einzelnen Abläufe der Vorverarbeitung für ein Bild.

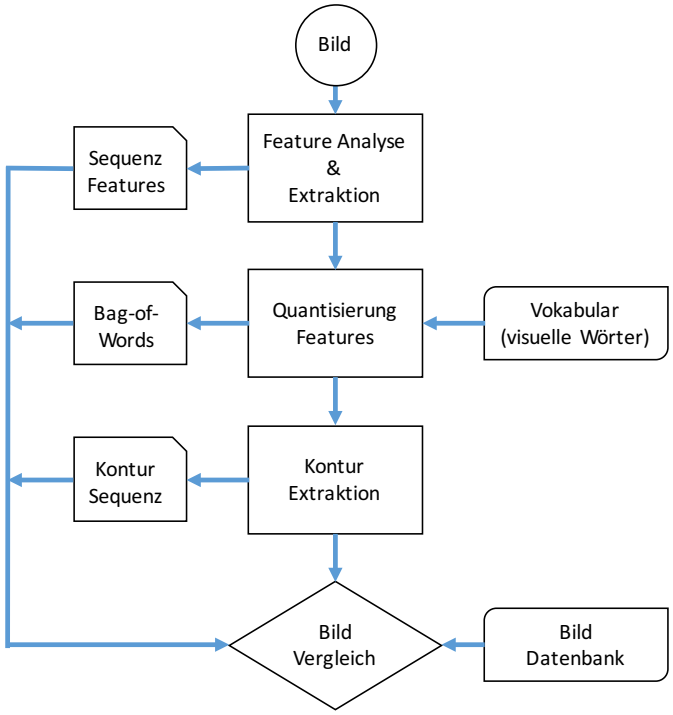


Abbildung 3.4: Vorverarbeitung des Bildes

Das Bild wird zunächst auf Basis des SURF-Algorithmus auf Features untersucht. Die Features $Seq_{Features}$ werden anschließend extrahiert und gespeichert. Zusätzlich werden sie für den nächsten Durchlauf weiter verarbeitet. Hierfür werden sie quantisiert und zu einer Bag-of-Words Repräsentation des Bildes abgebildet. Voraussetzung für die Quantisierung ist die Existenz eines

Vokabulars. Idealerweise beinhaltet das Vokabular eine ausgewogene Menge an Wörtern, das ein breites Spektrum an Features repräsentiert. Die Bag-of-Words Repräsentation $Quant_{BoW}(Seq_{Features})$ wird anschließend abgespeichert. Der letzte Durchlauf verwendet ebenfalls die SURF-Features. Hierfür werden jedoch nur die zweidimensionalen Bildkoordinaten jedes Features der Reihenfolge nach von Links nach Rechts, von Oben nach Unten, als Sequenz $Seq_{Contour}(Seq_{Features})$ extrahiert und gespeichert. Insgesamt erhält man nach dem Vorverarbeitungsschritt eines Bildes folgenden Vektor, der im weiteren Verlauf als *mehrstufiger Vergleichsvektor* bezeichnet wird:

$$\begin{pmatrix} Seq_{Features} \\ Quant_{BoW}(Seq_{Features}) \\ Seq_{Contour}(Seq_{Features}) \end{pmatrix}$$

Anhand des mehrstufigen Vergleichsvektors eines Bildes und einer Bilddatenbank, deren Bilder im Vorfeld den selben Vorverarbeitungsschritt unterliegen, wird der mehrstufige Vergleich durchgeführt. Dabei müssen die beiden letzten Stellen des mehrstufigen Vergleichsvektors nicht zwingend belegt sein. Bei Nichtbelegung werden diese Stufen einfach übersprungen oder ausgelassen.

3.3.2 Stufenweiser Vergleich

Der mehrstufige Vergleichsprozess besteht aus drei Stufen, die nacheinander ausgeführt werden. Jede Stufe kann ausgelassen oder übersprungen werden. Die Ergebnismenge einer Stufe wird auf die darauffolgende Stufe als Eingabemenge weiter gegeben. Die Stufen werden nun der Reihenfolge nach vorgestellt.

Vergleich basierend auf den visuellen Wörtern

Die erste Stufe des mehrstufigen Vergleichsprozesses setzt die Bag-of-Words Repräsentation eines Bildes ein. Voraussetzung für den Vergleich ist, dass der mehrstufige Vergleichsvektor an dieser Stelle belegt ist, so dass jeweils eine Bag-of-Words Repräsentation BoW_B des Referenzbildes B sowie BoW_{D_i} für jedes Bild D_i aus der Bilddatenbank $D = (D_0, D_1, ..., D_i)$ existiert. Zur Ermittlung der Ähnlichkeiten zwischen dem Referenzbild B und jedem Bild D_i aus der Datenbank, wird die Kosinus-Distanz zwischen BoW_B und BoW_{D_i} berechnet. Sie gibt jeweils den Winkel zwischen den zwei Vektoren an, wobei der Wertebereich zwischen 0 (keine Ähnlichkeit) und 1 (identisch) liegen kann. Vor jedem Vergleich werden die Vektoren zusätzlich mit dem Gewichtungsschema *tf-idf* vorverarbeitet, sodass gut beschreibende Wörter eines Bildes hoch gewichtet werden (vgl. Abs. 3.2.2.3). Die Bilder D_i werden absteigend ihrer Kosinus-Distanz sortiert, sodass das ähnlichste Bild an erster Stelle und das unähnlichste Bild an letzter Stelle liegt. Als Ergebnismenge wird die sortier-

te Liste an die nächste Stufe weitergegeben. Als zusätzliches Kriterium kann die Liste reduziert werden, indem Vergleichsbilder grundsätzlich aus der Liste entfernt werden, sobald sie unterhalb einer definierten Schranke liegen. Damit werden im Vorfeld bereist falsch-positive Vergleichsbilder ausgeschlossen.

Vergleich basierend auf den visuellen Features

Die zweite Stufe des mehrstufigen Vergleichsprozesses verwendet die SURF-Features der Bilder. Dafür wird jeweils die Sequenz $Seq_{Features_B}$ des Referenzbildes B mit jeder Sequenz $Seq_{Features_{D_i}}$ der Vergleichsbilder aus der Datenbank D verglichen. Da ein Vergleich mit jedem Bild D_i aus der Datenbank D sehr zeitaufwendig ist, wird die Ergebnismenge, soweit sie vorhanden ist, aus der vorherigen Stufe verwendet und idealerweise auf eine Menge k-nächster Nachbarn beschränkt. Das heißt, idealerweise werden nur die k ähnlichsten Bilder betrachtet, um einen schnelleren Vergleich zu erreichen. Der Vergleich findet auf den Deskriptoren der jeweiligen Feature-Paare statt. Hierfür wird die *Nearest Neighbor Distance Ratio*, die in Abschnitt 3.2.1 breits vorgestellt wurde, eingesetzt. Als Funktion zur Berechnung der Distanz zwischen zwei Vektoren wird die Euklidische Distanz verwendet. Der endgültige Vergleich zwischen einem Referenzbild B und einem Vergleichsbild D_i gibt als Ergebnis die Anzahl an erfolgreichen Treffern ihrer übereinstimmenden Features zurück. Das Vergleichsbild mit der höchsten Trefferzahl stellt das ähnlichste Vergleichsbild zum Referenzbild dar. Als Ergebnismenge dieser Stufe werden die Eingabebilder als sortierte Liste zurückgegeben, wobei das Bild mit der höchsten Trefferzahl an erster Stelle und das mit der niedrigsten an letzter Stelle steht. Die Liste kann zusätzlich reduziert werden, indem falsch-positive Vergleichsbilder herausgenommen werden, sobald ihre Trefferzahl unterhalb einer definierten Schranke liegt. Das macht besonders dann Sinn, sobald die Trefferzahl so gering ist, dass eine Ahnlichkeit nicht garantiert werden kann.

Vergleich basierend auf der Kontur

Die dritte Stufe des mehrstufigen Vergleichsprozesses verwendet die Kontur als Vergleichskriterium. Diese ergibt sich aus der Menge an Features eines Bildes. Hierfür wird ebenfalls als Eingabemenge die Sequenz $Seq_{Features_B}$ des Referenzbildes B und die Sequenz $Seq_{Features_{D_i}}$ der Vergleichsbilder D_i verwendet. Zum Vergleich der Konturen wird die Hausdorff-Distanz eingesetzt [132]. Diese gibt einen diskreten Wert an, inwieweit zwei Konturen übereinstimmen und sich ähneln. Da die Vergleichbarkeit von Konturen selbst durch einfachste Transformationen negativ beeinflusst wird, eignet sich die dritte Stufe des Vergleichsprozesses eher für Bildvergleiche, die keine Translation, Rotation oder Skalierung des Bildes voraussetzen. Je kleiner der Wert der Hausdorff-Distanz ist, desto größer ist die Ähnlichkeit zwischen zwei Bildern. Die Berechnung der Hausdorff-Distanz ist sehr zeitaufwendig, weshalb die zu vergleichende Eingabemenge im Vorfeld stark reduziert wird. Zusätzlich wird eine obere Schranke

für den Wert der Hausdorff-Distanz eingeführt, um falsch-positive Vergleichsbilder auszuschließen.

Zusammenfassung

Ein Vergleich von Bildern wird mit dem mehrstufigen Vergleichsprozess mit jeder weiteren Stufe anhand von strengeren Kriterien genauer. Dabei werden in der ersten Stufe die quantisierten Features eines Referenzbildes und seiner Vergleichsbilder verglichen. Als Ergebnis wird eine sortiere Liste aller Vergleichsbilder zurückgegeben. Diese werden anschließend an die nächste Stufe des Vergleichsprozesses weitergeben. Die Liste kann neben der beschriebenen oberen Schranke auf eine maximale Anzahl beschränkt werden. Das Referenzbild wird anhand der Sequenz seiner Features und der Feature-Sequenzen der Vergleichsbilder entsprechend ihrer Trefferzahl verglichen. Als Resultat wird eine neu sortierte Ergebnisliste zurückgegeben. Die Liste kann auf eine maximale Anzahl beschränkt werden. In der letzten Stufe des Vergleichsprozesses wird das Referenzbild mit der übergebenen Ergebnisliste der vorherigen Stufe anhand ihrer Konturen verglichen. Dabei wird die Hausdorff-Distanz eingesetzt und das Bild, das unterhalb der oberen Schranke liegt und die niedrigste Distanz hat, als Ergebnis zurückgegeben.
Durch die drei Stufen wird ein möglichst hoher Grad an Genauigkeit erreicht. Insbesondere wenn die Werte der einzelnen Schranken innerhalb der jeweiligen Stufen gut gesetzt sind, lassen sich falsch-positive Vergleichsbilder so gut wie ausschließen. Eine ausführliche Evaluierung des mehrstufigen Vergleichsprozesses wird im folgenden Abschnitt anhand des SURFLogo Systems vorgestellt.

3.4 Das SURFLogo System

Das SURFLogo System wurde als eine Alternative zum klassischen *Mobile Tagging* entwickelt. Dabei versteht man unter Mobile Tagging den weiterentwickelten Prozess zur Verknüpfung von realen Objekten mit digitalen Informationen [133]. Die Verknüpfung von realen und digitalen Inhalten stellt ein spannendes, aber auch komplexes Feld aktueller medialer Entwicklungen dar [134, 135, 136].
Beim Mobile Tagging werden Objekte mit Barcodes, sogenannten *Mobile Tags* versehen, die kodierte Informationen enthalten. Ein Mobile Tag lässt sich mit einer Anwendung auf einem Handy, Smartphone oder einem anderem mobilen Gerät mit Kamera einfach auslesen. Sie werden zur Vermarktung und Onlineverlinkung von Angeboten in der Werbung verwendet. Man nutzt sie auch für das Verlinken von Orten auf Onlinekarten *(Google Maps)* [137], von Onlineprofilen in sozialen Netzwerken oder zur Verlinkung von mobilen Anwendungen in den jeweiligen *AppStores* [138]. Besonders bei der Bewerbung mobiler Anwendungen werden sie immer häufiger eingesetzt. Dabei werden QR-Codes als

Mobile Tags verwendet und neben einer zu bewerbenden Anwendung abgebildet, sodass man bei Bedarf die Anwendung direkt auf sein mobiles Endgerät herunterladen kann, indem man den QR-Code abscannt. Trotz dieser praktischen Eigenschaft, wirkt ein QR-Code meist deplatziert und unschön. Insbesondere dann wenn ein QR-Code mehr Platz einnimmt als die eigentliche App-Werbung. So gibt es bereits alternative Lösungen des Mobile Tagging, die auf einen visuellen Tag vollständig verzichten und statt dessen funkbasierte Technologien einsetzen. Die Near Field Communication-Technologie (NFC) wird als Mobile Tag für die Tourismusbranche [139], im Smart-Home Bereich [140] und im Mobile Payment [141] bereits verwendet. Auch Bluetooth hat durch seine Weiterentwicklung auf den Standard 4.0 und durch seine stromsparenden Eigenschaften mehr an Bedeutung gewonnen. Technologien wie *Apples iBeacon* und *Goolge Eddystone* werden im Bereich des Proximity Marketings breits erfolgreich eingesetzt [142, 143]. Einen anderen Weg verfolgt Microsoft mit seinen sogenannten *Custom Tags*. Diese bieten ein Verfahren an mit der Bilder, also eigene Logos, Brandings und Fotos, als Mobile Tag verwendet werden können. Das Verfahrens überzieht über ein Bild eine Matrix aus Punkten (Matrix of dots), die von einer speziellen Foto-Anwendung erkannt und ausgelesen werden kann [144].
Angelehnt an die Lösung von Microsoft wird im Folgenden das Konzept und System von SURFLogo vorgestellt, das ermöglicht, mobile Apps anhand ihres Markenlogos eindeutig zu identifizieren. Dabei wird eine eigens entwickelte mobile App eingesetzt, die ähnlich wie bei einem Barcode-Scanner für Smartphones durch ein visuelles Verfahren App-Logos identifizieren und eindeutig zuordnen kann. Zur eindeutigen Identifizierung eines App-Logos wird der mehrstufige Vergleichsprozess aus Abschnitt 3.3 eingesetzt. Anschließend wird eine Evaluierung des Systems gegeben, wobei der Schwerpunkt auf dem Verfahren des mehrstufigen Vergleichsprozess liegt.

3.4.1 Konzept des SURFLogo Systems

Im Folgenden wird zunächst das Konzept beziehungsweise das Bedienkonzept für das System SURFLogo vorgestellt. Anschließend werden die einzelnen Komponenten des verteilten Systems betrachtet. Abschließend wird der Vergleichsalgorithmus, der den mehrstufigen Vergleichsprozess einsetzt, beschrieben.

3.4.1.1 Bedienkonzept

Die Grundidee von SURFLogo fußt darauf die Bewerbung von Apps durch Mobile Tags speziell auf Werbeplakaten oder Onlinewerbung vollständig zu ersetzten, indem anstelle des Mobile Tags das App-Logo selbst dafür verwendet wird (vgl. Abb. 3.4.1.1).
Das App-Logo wird von einer speziellen Smartphone-Anwendung *(SURFLogoApp)* mit Hilfe der Smartphone-Kamera erkannt und eindeutig identifiziert,

(a) (b)

Abbildung 3.5: App-Werbung der Münchner Verkehrsgesellschaft: (a) Plakatwerbung mit Abbildung des App-Logos und eines QR-Codes zum Herunterladen der App - (b) App-Logo der Münchner Verkehrsgesellschaft [145].

um anschließend die App zum Download bereitzustellen. Damit ein Nutzer sowie die SURFLogoApp das App-Logo schnell erkennen kann, wird es leicht modifiziert und mit einem schwarzen quadratischen Doppelrahmen versehen. Ein solches App-Logo wird im Folgenden einfach *SURFLogo* genannt.
Das Bedienkonzept für einen Anwender lässt sich in drei einfache Schritte unterteilen (vgl. Abb. 3.6):

- Der Anwender erkennt ein SURFLogo anhand des quadratischen Doppelrahmen und startet die SURFLogoApp auf seinem Smartphone.
- Nun scannt der Anwender das SURFLogo mit der SURFLogoApp ab.
- Die SURFLogoApp identifiziert die App anhand des Logos und bietet dem Anwender einen Link zum Download an.

3.4.1.2 Komponenten

Das SURFLogo Sytems ist eine Verteilte Anwendung, die aus den drei Komponenten *Datenbank*, *Anfrage-Server* und der mobilen App *SURFLogoApp* besteht.

Datenbank Die Datenbank beinhaltet die App-Logos aller registrierten Apps, die als SURFLogo erkannt werden sollen. Dafür werden Bildinformationen und Metadaten des Logos in der Datenbank hinterlegt. Da der Vergleichsalgorithmus auf dem mehrstufigen Vergleichsprozess basiert, bestehen die Bildinformationen aus SURF-Features. Hierfür werden pro App-Logo jeweils eine Sequenz all seiner Features $Seq_{Feature}$, eine Bag-of-Words Repräsentation der

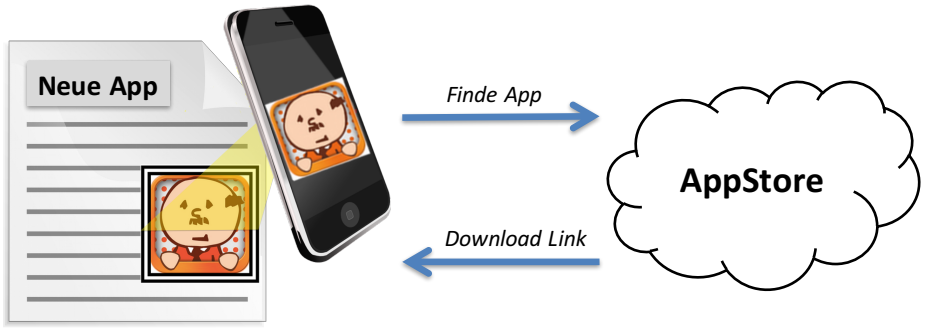


Abbildung 3.6: Bedienkonzept des SURFLogo Systems - eine neue App wird durch ein SURFLogo beworben und durch die SURFLogoApp gescannt; das Logo wird identifiziert und die App aus dem AppStore zum Herunterladen bereitgestellt [13].

quantisierten Features $Seq_{Contour}(Seq_{Features})$ und die sich zusammensetzende Kontur aus den Feature-Bildkoordinaten $Seq_{Contour}(Seq_{Features})$ verwendet. Zusätzlich wird ein Hyperlink URL_{App} hinterlegt, der auf die registrierte App im AppStore verweist. Damit hat jede registrierte App ein SURFLogo, das durch den Vektor $SURFLogo_i$, der in der Datenbank $Database_{SURFLogo}$ hinterlegt ist, repräsentiert wird.

$$SURFLogo_i = \begin{pmatrix} Seq_{Feature} \\ Quant_{BoW}(Seq_{Features}) \\ Seq_{Contour}(Seq_{Features}) \\ URL_{App} \end{pmatrix}, i \in Database_{SURFLogo}$$

Die Datenbank beinhaltet zusätzlich ein Vokabular, auf dessen Basis die Sequenz aus Features zu einer Bag-of-Words Repräsentation quantisiert werden. Das Vokabular wird für neue zu registrierende Apps und für den Vergleichsalgorithmus benötigt.

Anfrage-Server Der Anfrage-Server dient zur Verarbeitung von SURFLogo-Anfragen. Der Inhalt einer Anfrage besteht aus einer Sequenz aus Features ($Seq_{Feature}$), die ein App-Logo beschreiben. Die Feature-Sequenz wird in Kombination mit dem Vergleichsalgorithmus und der Datenbank dafür eingesetzt, ein entsprechendes SURFLogo eindeutig zu identifizieren. Bei erfolgreicher Suche wird ein Hyperlink (URL_{App}) des SURFLogo-Eintrages aus der Datenbank als Antwort zurückgegeben, ansonsten ist die Antwort leer.

Mobile App - SURFLogoApp Die mobile SURFLogoApp dient zum Scannen und zum Identifizieren von SURFLogos. Damit ein SURFLogo auch als solches von der Kamera des mobilen Endgerätes erkannt werden kann, wird

um das App-Logo ein schwarzer quadratischer Doppelrahmen gelegt (vgl. Abb. 3.7). Anhand des Rahmens lässt sich mit einem visuellen Verfahren das SURF-Logo innerhalb eines Kamerabildes automatisiert erkennen und extrahieren.

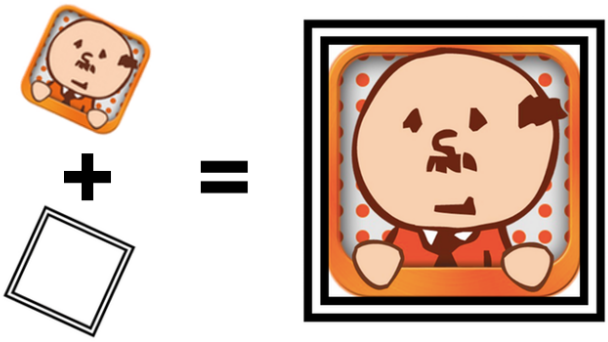


Abbildung 3.7: Beispiel für ein SURFLogo: ein App-Logo sowie ein schwarzer Doppelrahmen ergeben ein SURFLogo, das automatisch von der SURFLogoApp erkannt wird [13].

Das visuelle Verfahren zum Erkennen von SURFLogos funktioniert wie folgt: (1) Das Kamerabild wird nach einem schwarzen quadratischen Doppelrahmen durchsucht. (2) Bei Erfolg wird das Bild entsprechend der Ausrichtung des Rahmens rotiert und das App-Logo aus dem Doppelrahmen entfernt und gespeichert. (3) Anschließend werden aus dem gespeicherten Bild alle Features mit dem SURF-Algorithmus extrahiert und zu einer Sequenz zusammengefasst ($Seq_{Feature}$).
Die extrahierte Sequenz von Features $Seq_{Feature}$ wird an den Anfrage-Server gesendet. Bei einer erfolgreichen Antwort erhält die SURFLogoApp einen Hyperlink (URL_{App}), der für das Herunterladen der App benötigt wird. Wird kein entsprechendes SURFLogo gefunden, wird eine leere Ergebnismenge zurückgegeben.

3.4.1.3 Vergleichsalgorithmus

Der Algorithmus dient zur eindeutigen Identifizierung und Zuordnung von SURFLogos. Dabei wird der mehrstufige Vergleichsprozess aus Abschnitt 3.3 eingesetzt, um ein abfotografiertes SURFLogo eindeutig zu identifizieren und falsch-positive Ergebnisse auszuschließen.
Der Algorithmus setzt alle drei Stufen des Vergleichsprozesses ein. In der ersten Stufe wird auf Basis der Bag-of-Word Repräsentation verglichen. Hierfür werden alle quantisierten Features der SURFLogos aus der Datenbank mit dem zu vergleichenden quantisierten Feature des Anfrage-Logos verglichen. Es werden die k ähnlichsten SURFLogos an die nächste Stufe weitergegeben. Es kann ein Wert für k zwischen 2 und 25 gewählt werden. Ein idealer Wert wird in Abschnitt 3.4.2 evaluiert.

In der nächsten Stufe werden die k ähnlichsten SURFLogos auf Basis ihrer SURF-Features verglichen und die Menge absteigend nach der Anzahl gleicher Features neu sortiert. Durch den Vergleich auf den SURF-Features werden mögliche falsch-positive Werte herausgefiltert. Zusätzlich wird eine untere Schranke verwendet, die SURFLogos bei einer Unterschreitung einer Mindestanzahl ähnlicher Features aussortiert. Ein Wert für die untere Schranke wird ebenfalls in Abschnitt 3.4.2 evaluiert.

In der letzten Stufe werden die SURFLogos anhand aller ihrer Feature-Bildkoordinaten, also ihrer Kontur, verglichen, indem die Ähnlichkeit mittels Hausdorff-Distanz bestimmt wird. Das Vergleichspaar mit der niedrigsten Distanz gewinnt. Liegt die niedrigste Distanz über einer definierten oberen Schranke, wird das Vergleichsverfahren ohne erfolgreiches Ergebnis abgebrochen. Da der Vergleich von Konturen mittels der Hausdorff-Distanz sehr zeitaufwendig ist, werden nur die zwei ähnlichsten SURFLogos aus der vorherigen Stufe betrachtet. In Abschnitt 3.4.2 wird der Wert für eine obere Schranke evaluiert.

Insgesamt wird mit jeder Stufe die Ergebnismenge soweit reduziert, dass man am Ende des gesamten Prozesses ein eindeutiges Ergebnis erhält. Durch die Schranken wird zusätzlich erreicht, dass falsch-positive Ergebnisse im Vorfeld herausgefiltert werden.

3.4.2 Evaluierung

Die Evaluierung gibt einen Einblick auf die allgemeinen Performanz des SURFLogo Systems. Dazu wird der Prozess zur Erzeugung des Vokabulars, das entscheidend für die Qualität der Quantisierung von Features ist, unter verschiedenen Bedingungen evaluiert. Des Weiteren wird der mehrstufige Vergleichsprozess unter Anwendung des SURFLogo Systems in allen seinen drei Stufen ausgewertet.

Zur Evaluierung wird ein Datensatz verschiedener SURFLogos sowie ein Testset, das von Studienteilnehmern durch Anwendung der SURFLogoApp erzeugt wurde, eingesetzt.

3.4.2.1 Konfiguration und Setup

Für die gesamte Evaluierung wird ein Datenset bestehend aus 5541 SURFLogos verwendet, die aus unterschiedlichen App-Logos bestehen. Dabei wurden die Logos aus dem *Goolge PlayStore* zufällig ausgewählt und heruntergeladen.

Insgesamt werden 2.118.240 SURF-Features aus allen App-Logos extrahiert und in der Datenbank nach einer Vorverarbeitung als SURFLogo abgelegt. Für das Testset wurden 759 gelabelte SURFLogos von sechs unterschiedlichen Nutzern mit einer modifizierten SURFLogoApp erzeugt.

Teilmenge	10%	20%	100%
Dauer	4h 1m	15h 39m	4d 12h 2m

Tabelle 3.1: Zeitaufwand für das Clustern mit einer Clustergröße von 512 - 3 Cluster-Mengen mit jeweils 10%, 20% und 100% der Features aus der Datenbank.

3.4.2.2 Erzeugen des Vokabulars

Die Quantisierung der Features ist abhängig von mehreren Variablen, die für eine erfolgreiche Suche im mehrstufigen Vergleichsprozess notwendig sind. So sind die Menge und Selektion der Features zur Erzeugung des Vokabulars sowie die Anzahl der Wörter eines Vokabulars entscheidet für die Qualität der Quantisierung. Da allgemein durch die Quantisierung ein Informationsverlust besteht, muss bereits bei der Erzeugung des Vokabulars sowie bei der Auswahl und Menge der Features folgendes beachtet werden: (1) Die Menge an Features muss groß sein; idealerweise beinhaltet sie alle Features aus der Datenbank. (2) Die Menge repräsentiert ein breites Spektrum unterschiedlicher Features.
Für die Erzeugung des Vokabulars muss die Menge an Features geclustert werden. Dies geschieht mit Hilfe des *k-Mean* Algorithmus (vgl. Abs. 3.2.2.1). Da das Clustering, insbesondere mit dem *k-Mean* Algorithmus, bei einer hohen Anzahl von SURF-Features mit seinen 64-stelligen Deskriptoren sehr zeitintensiv ist, werden für das Erzeugen des Vokabulars unterschiedlich große Mengen an Features evaluiert. Dabei wird ausgewertet, ob eine kleinere Teilmenge für ein ausgewogenes und qualitativ hochwertiges Vokabular bereits ausreicht.
Für die Auswertung werden drei unterschiedliche Cluster-Mengen ausgewählt, die jeweils aus 10%, 20% und 100% aller Features aus der Datenbank bestehen. Tabelle 3.4.2.2 veranschaulicht, dass die Größe der Cluster-Menge in Abhängigkeit mit der Dauer des Cluster-Prozesses zusammenhängt. Besonders der zeitliche Unterschied zwischen einer Cluster-Menge von 10% und 100% ist mit über vier Tagen gravierend hoch.
Zusätzlich zeigt Abbildung 3.8 jeweils die Erfolgsquote für die drei unterschiedlichen Vokabulare, die aus den drei Cluster-Mengen (10%, 20% und 100%) erzeugt worden sind. Dabei wurde die Erfolgsquote für jedes Vokabular mit einem Testset an SURFLogos mit dem Vergleichsprozess der ersten Stufe ausgewertet. Man erkennt deutlich, dass die Ergebnisse trotz größerer Cluster-Menge nicht besser werden.
Neben der richtigen Menge an Features zur Erzeugung des Vokabulars, ist auch die Anzahl an Wörtern eines Vokabular entscheidend für die Güte des Verfahrens. Die Größe des Vokabulars ist äquivalent mit der Anzahl der Cluster. Ist die Anzahl der Cluster zu niedrig gewählt, ist der Informationsverlust beim Quantisieren größer und die Qualität des Ergebnisses nimmt ab. Wird die Anzahl zu hoch gewählt, nimmt der durch die Quantisierung gewonnene Zeitgewinn deutlich ab. Abbildung 3.9 zeigt, dass mit einer steigenden Anzahl

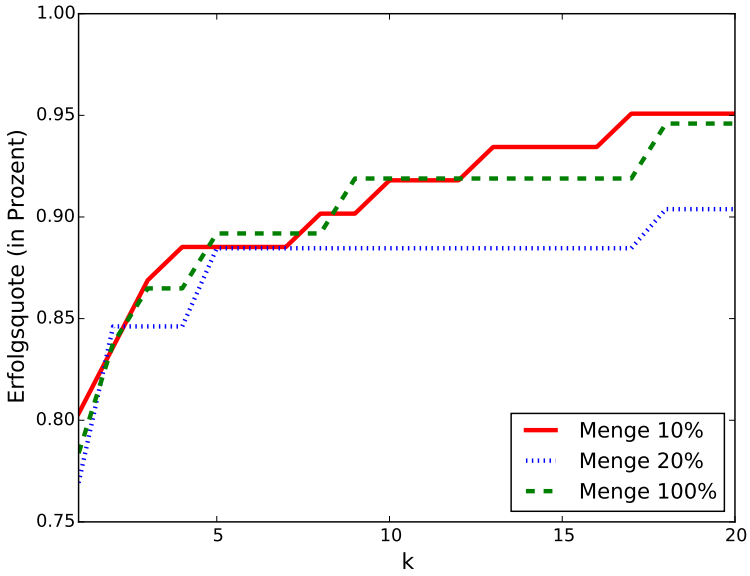


Abbildung 3.8: Erfolgsquote verschiedener Vokabulare mit steigendem k ähnlichster Nachbarn - Das Vokabular ist aus den Cluster-Mengen mit 10%, 20% und 100% aller Features aus der Datenbank erzeugt worden [13].

an Clustern die Qualität der Ergebnisse bereits bei einem niedrigen k ähnlicher Nachbarn relativ hoch ist. Für das Erzeugen des Vokabulars werden 10% aller Features aus der Datenbank verwendet.
Die Erfolgsquote unterscheidet sich zwischen den Clustergrößen 32 und 512 bereits bei einem k von eins um knapp 30%. Dieser Unterschied nimmt zwar mit steigendem k ab, stagniert jedoch bereits bei einem k von 18.
Insgesamt zeigen die Ergebnisse, dass ein Vokabular mit einer kleineren Teilmenge aller Features aus der Datenbank erzeugt werden kann und dabei die Qualität nicht abnimmt. Des Weiteren ist die Wahl einer größeren Anzahl an Vokabeln bei vielen App-Logos ebenfalls entscheidend für die Qualität. Daher wird für die weitere Evaluierung das Vokabular mit 10% aller Features aus der Datenbank und einer Größe des Vokabular mit 512 Wörtern erzeugt.

3.4.2.3 Auswertung des mehrstufigen Vergleichsprozesses

Der mehrstufige Vergleichsprozess teilt sich in drei Stufen auf. Dabei wird jede Stufe einzeln ausgewertet. Zusätzlich werden für jede Stufe die Werte für die einzelnen Schranken sowie eine ideale Größe der Eingabemenge evaluiert.

Stufe 1: Bag-of-Words-Vergleich In dieser Stufe werden die SURFLogos auf Basis ihrer quantisierten Features, also ihrer Bag-of-Words Repräsentation verglichen. Da durch die Quantisierung ein Informationsverlust besteht, sind

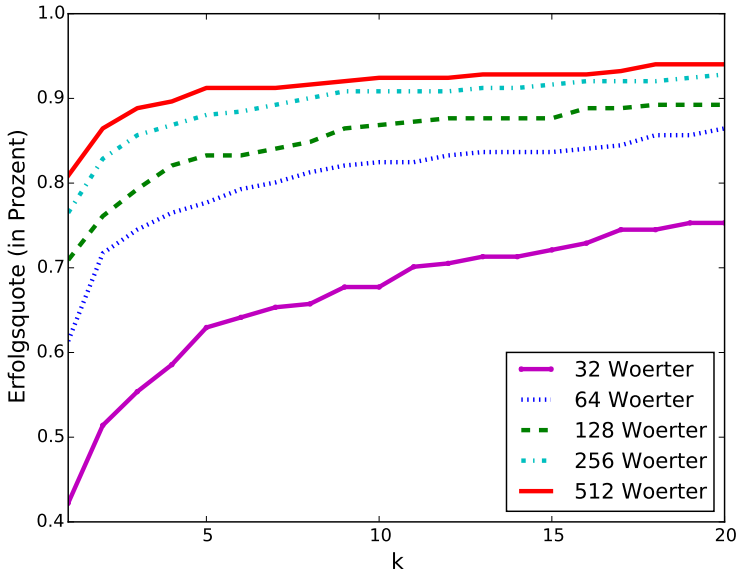


Abbildung 3.9: Erfolgsquote verschiedener Vokabeluare mit unterschiedlicher Anzahl an Wörtern (32, 64, 128, 256, 512) - k entpsricht der Anzahl an ähnlihcher Nachbarn. Das Vokabular wurde mit 10% aller Features aus der Datenbank erzeugt [13].

die Vergleiche auch etwas ungenauer. Das heißt, dass das Ergebnis zwar gefunden wird, jedoch nicht unbedingt als ähnlichstes SURFLogo betrachtet wird, sondern unter den k ähnlichsten Nachbarn liegt. Daher ist es wichtig eine geeignete Größe für k nächste Nachbarn zu finden, die als Ergebnismenge an die nächste Stufe weitergegeben werden.
Entscheidend ist auch eine geeignete Menge zu finden, so dass das richtige Ergebnis noch enthalten ist und trotzdem die Menge klein genug bleibt. In Abbildung 3.9 wurde bereits gezeigt, dass mit ansteigendem k der Ergebnismenge die Erfolgsquote besser wird. Bereits ab einem k von 18 stagniert die Erfolgsquote bei knapp 94%.
Zur Sicherheit wird das endgültige k um einen Puffer von zwei erweitert, sodass die Ergebnismenge dieser Stufe immer eine maximale Größe von 20 hat. Um falsch-positive Ergebnisse auszuschließen, wird eine untere Schranke eingeführt. Da für die Ähnlichkeiten zweier Bag-of-Words Repräsentationen die Kosinus-Distanz verwendet wird, liegt der Wertebereich immer zwischen 0 (unähnlich) und 1 (identisch). Die Evaluierung hat gezeigt, dass alle falsch-positiven Ergebnisse immer unterhalb eines Wertes von 0,5 liegen. Daher wurde dieser Wert als untere Schranke für die Ergebnismenge verwendet.

Stufe 2: SURF-Feature-Vergleich In der nächsten Stufe werden die 20 ähnlichsten SURFLogos auf Basis ihrer SURF-Features verglichen und entsprechend ihrer Ähnlichkeit neu sortiert. Der Vergleich auf Basis ihrer SURF-

Features ist wesentlich genauer, sodass in knapp 93% aller Vergleichsdurchläufe das richte Ergebnis an erster oder zweiter Stelle der neu sortierten Menge liegt. In 91% aller Fälle liegt das richtige Ergebnis sogar an erster Stelle. Um falsch-positive Ergebnisse auszuschließen, wird ein geeigneter Wert für eine untere Schranke eingeführt. In dieser Stufe basiert die Ähnlichkeit zweier SURFLogos auf der Anzahl gleicher Features. Daher wird ein Wert evaluiert, der möglichst viele falsch-positive Ergebnisse aussortiert, aber gleichzeitig ein richtiges Ergebnis beibehält.
Abbildung 3.10 zeigt den Verlauf für unterschiedliche Schrankenwerte von 0 bis 30. Dabei wird deutlich, dass ein Wert zwischen 10 und 20 geeignet ist, um möglichst viele falsch-positive SURFLogos auszuschließen. Insbesondere ein Wert von 10 schließt möglichst viele falsch-positive und äußerst wenige richtige SURFLogos (falsch-negativ) aus.

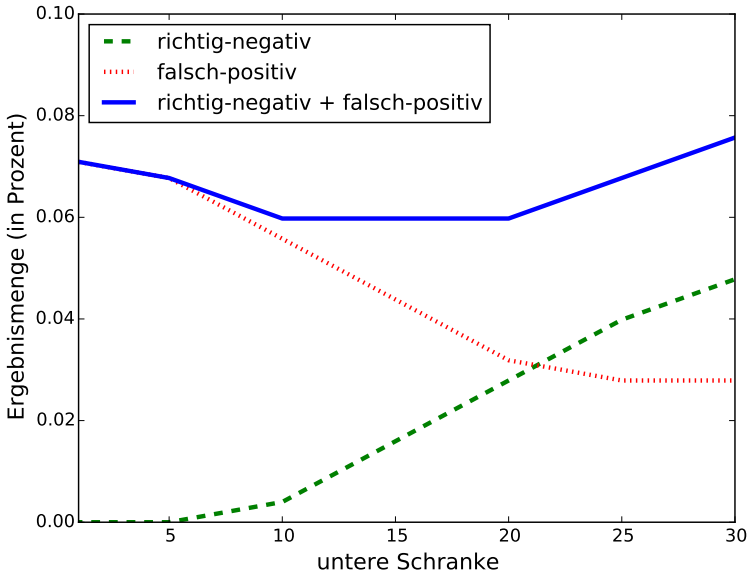


Abbildung 3.10: Evaluierung einer unteren Schranke für das Vergleichverfahren in Stufe 2 - Anwendung unterschiedlicher Schrankenwerte (0, 5, 10, 15, 20, 25, 30). Die Summe aus den falsch-positiven und falsch-negativen Werten hat ihr Minimum zwischen 10 und 20 [13].

Liegt demnach der Wert der Ähnlichkeit aller SURFLogos aus der Ergebnismenge unter 10, wird kein Ergebnis zurückgegeben und der Prozess abgebrochen. Insgesamt zeigt sich, dass nach den zwei Vergleichsstufen das richtige Ergebnis entweder gar nicht gefunden wird oder in 99% aller Fälle unter den ersten beiden ähnlichsten SURFLogos liegt.
Somit wird die Ergebnismenge für die nächste Stufe auf die ersten beiden ähnlichsten SURFLogos beschränkt. Das eine Prozent aller Fälle, bei dem das

Ergebnis weiter hinten liegt, wird vernachlässigt, da dieses keine große Auswirkung auf das Gesamtergebnis hat.

Stufe 3: Kontur Vergleich Die letzte und dritte Stufe des mehrstufigen Vergleichsprozesses nutzt die Bildkoordinaten der gemeinsamen Features und berechnet die Ähnlichkeit zwischen zwei SURFLogos mit der Hausdorff-Distanz. Die Bildkoordinaten der Features bilden die Kontur eines SURFLogos ab. Anhand des Kontur-Vergleiches wird garantiert, dass das SURFLogo mit der geringsten Distanz auch dem richtigen Ergebnis entspricht. Da nur die Bildkoordinaten der gemeinsamen Features zum Vergleich verwendet werden, wird garantiert, dass keine Features, die durch Bildrauschen oder andere Störfaktoren entstehen, für den Vergleich der Konturen verwendet werden. Damit Vergleichspaare mit einer geringen Anzahl an gemeinsamen Feature-Punkten kein besseres Ergebnis bei der Distanzberechnung erhalten, werden die Distanzen normiert, indem die Distanz durch die gesamte Summe gemeinsamer Features geteilt wird.

Um falsch-positive Ergebnisse auszuschließen, wird zusätzlich eine obere Schranke für die Hausdorff-Distanz eingeführt. Abbildung 3.11 zeigt, dass die Distanzen der richtigen Ergebnisse deutlich unter einer Distanz von 75 liegen. SURFLogos, deren Distanzen sehr nah beieinander liegen, sind meist App-Logos die sich nur durch geringe Details unterscheiden. Solche kleinen Unterschiede ergeben sich bei Logos einer App, die es z.B. als kostenlose und kostenpflichtige Version gibt (vgl. Abb. 3.12a).

Aufgrund dieser hinzugewonnen Erkenntnis wird die dritte Stufe um folgende Bedingung erweitert:

- Alle Distanzen die über einer Schranke von 75 liegen, treffen als richtiges Ergebnis nicht zu.

- Unterscheiden sich die Distanzen zweier Vergleichspaare um weniger als eins, werden beide SURFLogos als richtiges Ergebnis gewertet.

Als Ergebnis wird das Bild mit der geringsten Distanz zurück gegeben. Insgesamt werden damit 93% aller SURFLogos richtig erkannt und gefunden. In den übrigen 7% liefert das Verfahren kein Ergebnis zurück. Damit ergeben sich keine falsch-positiven Resultate, die von der letzten Stufe zurück gegeben werden.

Ohne die Bedingung, dass bei Vergleichspaaren mit sehr geringen Distanzunterschieden beide Ergebnisse zurück gegeben werden, verschlechtert sich das Ergebnis nur sehr gering. Insgesamt werden dadurch nur noch 92% aller SURFLogos erfolgreich erkannt.

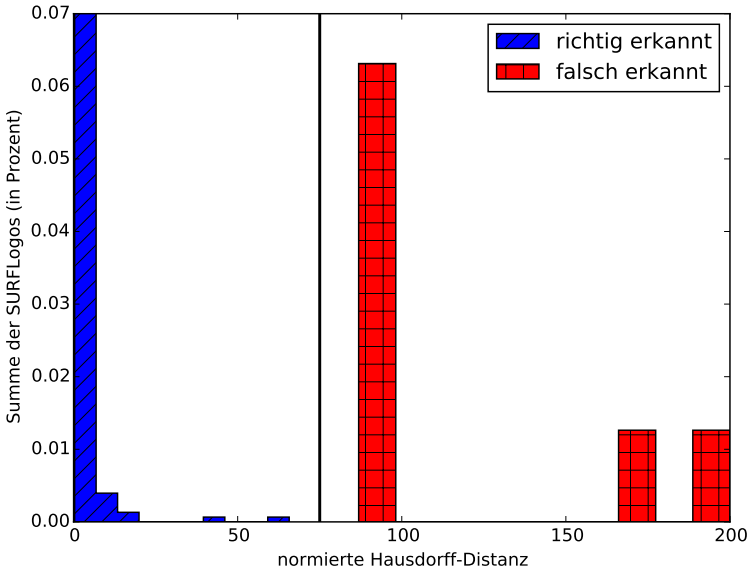
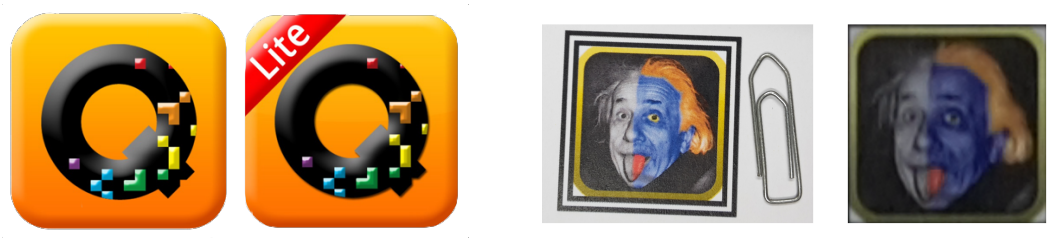


Abbildung 3.11: Evaluierung einer oberen Schranke für das Vergleichverfahren in Stufe 3 - die Distanzen der richtig erkannten SURFLogos liegen unterhalb von 75 (schwarze vertikale Linie). Abgesehen von einigen Ausnahmen haben die meisten richtig erkannten SURFLogos eine Distanz zwischen 0 und 5 [13].

3.5 Zusammenfassung

Dieses Kapitel beschäftigte sich mit der Vorstellung und Evaluierung eines effizienten Vergleichsverfahren, das auf Basis von visuellen Features Bilder schnell und eindeutig vergleichen kann. Dafür wurde ein mehrstufiges Vergleichsverfahren entwickelt, das mit jeder Stufe eine Vergleichsmenge genauer analysiert und aussortiert. Um das Verfahren zu evaluieren, wurde es in ein bestehendes System, das alle Stufen des Vergleichsprozesses verwendet, integriert. Damit konnten die Ergebnisse für alle einzelnen Stufen genau ausgewertet werden. Die Parameter des Verfahrens wurden speziell an das SURFLogo System angepasst. Die Ergebnisse der ersten Stufe haben gezeigt, dass ein Vokabular mit 512 Wörtern zur Quantisierung der Features eines Bildes eine geeignete Größe darstellt. 82% aller SURFLogos werden in dieser Stufe bereits als richtiges Ergebnis zurückgegeben. Da die Trefferquote jedoch nicht ausreichend ist, wird die Menge der 20 ähnlichsten Kandidaten aus dem Vergleichsprozess betrachtet, wodurch die Trefferquote auf 94% ansteigt. In der nächsten Stufe wird der Vergleich auf Basis der Features durchgeführt. Die Ergebnisse haben gezeigt, dass durch eine geeignete Schranke die Trefferquote bei 93% liegt und das richtige Ergebnis meist an erster oder zweiter Stelle liegt. Um letztendlich keine falsch-positiven Ergebnisse zu erhalten, wurde die dritte Stufe eingesetzt, welche auf Basis der Bildkoordinaten aller gemeinsamen Features das Bild ver-



(a) (b)

Abbildung 3.12: (a) Beispiel eines App-Logos, das es als kostenlose und kostenpflichtige Version gibt - (b) Beispiel eines SURFLogos: links die gedruckte Version, rechts die gescannte Version mit einer schlechteren Auflösung

gleicht. Damit werden die eigentlichen Konturen des SURFLogos miteinander verglichen. Um falsch-positive Ergebnis endgültig auszuschließen, wurde eine obere Schranke eingeführt. Insgesamt hat der mehrstufige Vergleichsprozess eine Erfolgsquote von 93%. In keinem der Fälle wurde ein falsch-positives Ergebnis zurückgegeben. Trotz dieser guten Ergebnisse zeigt das Verfahren auch, dass in 7% aller Fälle keine SURFLogos erkannt werden konnten. Dies wird zum Teil durch die schlechte Qualität der abfotografierten SURFLogos bedingt. Auch Interferenzen, die durch Bildrauschen oder durch zaghafte Bewegungen entstehen, tragen dazu bei, dass SURFLogos nicht gefunden werden können. Ein Nachteil des Verfahrens ergibt sich zusätzlich aus seiner Farbenblindheit, die durch den SURF-Algorithmus bedingt wird. Der Algorithmus konvertiert das Eingabebild vor der Feature-Analyse in ein Graustufenbild. Dadurch können SURFLogos, die zwar eine identische Kontur aber unterschiedliche Farben haben, nicht voneinander unterschieden werden.
Der mehrstufige Vergleichsprozess wird in erweiterter Form in den Verfahren aus Kapitel 4 und Kapitel 5 eingesetzt.

4 Visuelle Verfahren zur Positionsbestimmung

Die Bedeutung ortsbezogener Dienste hat in den letzten vergangenen Jahren immer mehr zugenommen. Insbesondere Dienste, die auf die Position des Nutzers zurückgreifen, sind durch die rasante Verbreitung mobiler Endgeräte, die leistungsstärker und mit entsprechender Sensorik ausgestattet sind, nahezu überall verfügbar. Beispiele für solche Dienste stellen die klassische Navigation, Point-of-Interest Suche und Touristenführer dar. Eine Positionierung mittels mobilem Endgerät erfolgt entweder durch das integrierte GPS-Modul oder durch eine WLAN-basierte Ortung, welche durch den Empfang unterschiedlicher Signale mehrerer WLAN-Access-Points und einer öffentlichen Datenbank die Position eines WLAN-fähigen Endgerätes berechnen kann [146, 147]. Besonders in Großstädten ist die Dichte von WLAN-Access-Points so hoch, dass eine auf wenige Meter genaue Standortbestimmung möglich ist. Innerhalb von Gebäuden ist die Positionsbestimmung anhand von GPS oder WLAN-basierter Ortung eingeschränkter und bedingt nutzbar. GPS ist ein satellitengestütztes System, welches auf den Empfang von Satellitensignalen angewiesen ist. Dies macht es schwierig eine genaue Bestimmung der Position innerhalb von Gebäuden zu ermöglichen. Dagegen hat die WLAN-basierte Ortung innerhalb von Gebäuden bessere Voraussetzungen. Viele Gebäude besitzen bereits eine Infrastruktur an WLAN-Access-Points, die zusätzlich zur Positionsbestimmung verwendet werden kann. Öffentliche Datenbanken, welche z.B. von *Google*[1] oder *Apple*[2] für ihre eigenen Kartendienste eingesetzt werden, verfügen nicht über alle Gebäudedaten, da diese zu aufwendig zu betreiben sind oder bewusst von Gebäudebetreibern unter Verschluss gehalten werden. Daher werden in einigen Fällen eigene Lokalisierungsdienste mit Hilfe von proprietären Systemen aufwendig selber betrieben, um ortsbezogene Dienste an eigenen Standorten anbieten zu können [148, 149]. Erschwert wird die Umsetzung von eigenen WLAN-basierten Ortungsdiensten durch die Firmenpolitik mancher Hardwarehersteller, indem diese das Auslesen der WLAN-Signale betriebssystemseitig vollständig unterbinden [150].
Eine Alternative zu den WLAN-basierten Technologien, stellen die visuellen Positionierungssysteme dar, bei denen Verfahren des maschinellen Sehens und der Bildverarbeitung zum Einsatz kommen. Durch die geringen Kosten einfacher Kamerasensoren und der hohen Verbreitung in mobilen Endgeräten, ist

[1] `https://www.google.com/intl/de_de/maps/about`
[2] `http://www.apple.com/de/ios/maps/`

der Einsatz visueller Verfahren zur Positionierung realisierbar geworden. Besonders die Tatsache, dass keine externe Infrastruktur wie bei GPS, WLAN oder anderen funkbasierten Technologien notwendig ist, macht die Umsetzung solcher Systeme vergleichbar einfacher und kostengünstiger. Ein Beispiel für ein solches System stellt das am Lehrstuhl für Mobile und Verteilte Systeme der Ludwig-Maximilians-Universität München im Jahr 2011 entwickelte *Mobile Visual Indoor Positioning Systems* (MoVIPS) dar [39]. Es extrahiert visuelle Features aus einem Bild einer Smartphone-Kamera und vergleicht sie mit einer Standortdatenbank, die georeferenzierte Ortsbilder beinhaltet, um die Position des Bildes zu ermitteln. MoVIPS kann die Position eines Standortbildes bis zu zwei Meter genau bestimmen. Zur Berechnung der Position wird jedoch ein sehr langsamer Bildvergleichsprozess eingesetzt, der bei einer steigenden Anzahl von gleichzeitigen Positionsanfragen nicht skaliert. Aus diesem Grund werden in diesem Kapitel drei Erweiterungen der eingesetzten visuellen Verfahren vorgestellt, die das System MoVIPS an entsprechender Stelle in seiner Effizienz und Genauigkeit verbessern. Folgende Erweiterungen werden dafür vorgestellt:

- Die erste Erweiterung setzt ein Schrittzählerverfahren basierend auf dem Beschleunigungssensor und dem integrierten Kompass des Smartphones ein, um eine gefilterte Suche auf der Ortsdatenbank auszuführen. Dadurch wird die Anzahl der zu vergleichenden Bilder möglichst klein gehalten.

- Die zweite Erweiterung verwendet eine reduzierte Variante des mehrstufigen Vergleichsprozesses zur Beschleunigung der Suchzeit auf der Ortsdatenbank (vgl. Kapitel 3). Dazu wird zusätzlich der mehrstufige Vergleichsprozess durch ein neues *Reranking*-Verfahren erweitert, welches den Vergleich auf Basis der Features nochmals beschleunigt.

- Die dritte Erweiterung macht das ursprünglich in MoVIPS entwickelte Verfahren zur Positionskorrektur rotationsinvariant. Dadurch funktioniert eine Korrektur der Position auch dann noch, wenn das Eingabebild zum Referenzbild in einem anderen Winkel erstellt wird.

Das übrige Kapitel ist wie folgt gegliedert. Zunächst wird ein Einblick in die unterschiedlichen visuellen Positionierungssysteme gegeben, um ein besseres Verständnis für diesen Bereich zu bekommen. Dann wird das visuelle Positionierungssystem MoVIPS vorgestellt, insbesondere die Verfahren, die durch die Erweiterungen verbessert werden. Anschließend werden die einzelnen Erweiterungen im Detail erklärt und evaluiert. Abschließend wird eine Zusammenfassung des vollständigen Kapitels gegeben.

4.1 Eigene Vorveröffentlichungen

Die Kerninhalte dieses Kapitels wurden vom Autor bereits in [14] publiziert. Wie in Kapitel 1.3 dargestellt, stammen die im Paper und im Folgenden präsentierten Inhalte bzgl. der Idee, des Konzepts und der Evaluation vom Autor der vorliegenden Arbeit. Ebenfalls im Paper bereits enthalten sind die Abbildungen 4.7, 4.8, 4.10a, 4.10b, 4.11a, 4.11b, 4.14b und 4.12a.

4.2 Visuelle Positionierungssysteme

Visuelle Positionierungssysteme stellen eine Alternative zu den klassischen funkbasierten Technologien und Systemen dar. Sie sind insbesondere für den Einsatz in Gebäuden, Flughäfen, Bahnhöfen und anderen Innenbereichen gut geeignet. Größtenteils besitzen alle visuellen Verfahren zur Bestimmung der Position den Vorteil, dass sie unabhängig von externen Infrastrukturen funktionieren. Visuelle Informationen lassen sich so gut wie in allen natürlichen Umgebungen wiederfinden und können durch einfache Kamerasensoren, wie sie z.B. in aktuellen Smartphones verbaut sind, analysiert werden. Anhand markanter Eigenschaften eines Bildes können selbst ähnliche Räume durch solche Verfahren eindeutig unterschieden werden. Die eigentlichen Herausforderungen ergeben sich eher durch die hohe Komplexität und die benötige Rechenzeit solcher Verfahren. Insbesondere die Architektur eines solchen Systems muss wohl überlegt sein. So können bestimmte Verfahren wegen der geringen Leistung auf einem mobilen Endgerät nicht ausgeführt werden und müssen deshalb auf entsprechende Server ausgelagert werden. Auch die Wahl der Algorithmen zur Analyse und zum Vergleich der Bilder entscheidet darüber, welche verteilten Komponenten zusätzlich bereitgestellt werden müssen oder unter welchen lokalen Bedingungen das System allgemein eingesetzt werden kann. Einige Verfahren setzen hierfür z.B. eine im Vorfeld erzeugte Datenbank bestehend aus Standortbildern voraus. Andere erzeugen diese erst zur Laufzeit und nutzen bis dahin andere Sensordaten zur Positionsbestimmung. Die Beschaffenheit und Struktur einer Umgebung hat ebenfalls Einfluss auf die Wahl des Verfahrens zur Extraktion markanter Merkmale, welche zur Charakterisierung eines Bildes notwendig sind. Je nach Verfahren können einzelne Features, Kanten, Farben oder ganze Objekte aus einem Bild extrahiert werden und als Vergleichsgrundlage verwendet werden.
Alle Systeme haben gemein, dass sie die selben Prozessschritte durchlaufen. Dazu gehören:

- die Standortaufzeichnung
- die Analyse des Standortbildes
- der Bildvergleich zur Positionsbestimmung

Die einzelnen Prozessschritte werden der Reihenfolge nach ausgeführt und werden im Detail je nach visuellem Positionierungssystem anders umgesetzt. Im Folgenden werden die einzelnen Prozessschritte genauer vorgestellt und anhand von Beispielen implementierter Systeme näher gebracht.

4.2.1 Standortaufzeichnung

Jede Ortsbestimmung eines visuellen Verfahrens beginnt mit dem Aufzeichnen des eigenen Standortes. Dabei wird meist ein Kamerasensor eines mobilen Endgerätes verwendet. Das aufgezeichnete Bild enthält idealerweise die charakteristischen Eigenschaften eines Ortes.
Neben der Bildinformationen benötigen einige Systeme zusätzliche Daten des aktuellen Ortes. Dafür werden zusätzlich funkbasierte Technologien eingesetzt, die empfangene WLAN- oder Bluetooth-Signale des Endgerätes zur Charakterisierung des Standortes verwenden. Des Weiteren werden Daten des Beschleunigungssensors oder des Kompass, soweit diese vom mobilen Endgerät bereitgestellt werden, zur Bestimmung der Lage und Ausrichtung des Standortbildes verwendet.
Abhängig von der Komplexität des Verfahrens zur Auswertung eines Bildes und zu dessen Positionsbestimmung wird dieses entweder auf dem mobilen Endgerät selbst ausgeführt oder an eine Serverkomponente weiter gegeben. Dies hängt wiederum von mehreren Faktoren des jeweiligen visuellen Positionierungssystem ab, die im folgenden kurz erläutert werden.

Leistung des Endgerätes Die folgenden Schritte müssen neben der Aufzeichnung eines Bildes vom einem mobilen Endgerät ausgeführt werden. Einerseits wird das Bild auf seine charakteristischen Merkmale analysiert und anderseits diese mit einer Datenbank verglichen, um den Standort zu bestimmen. Eine Analyse eines Standortbildes auf dem Endgerät auszuführen, hängt von der Komplexität des Algorithmus ab. Um möglichst gute charakteristische Eigenschaften aus einem Bild zu extrahieren, benötigt es mehrere Bildverarbeitungsoperationen, die sehr rechenaufwendig sein können und daher idealerweise parallelisiert werden müssen. Der Vergleich der Standortbilder mit einer Datenbank benötigt für den Vergleichsalgorithmus und für die Datenhaltung ebenfalls ausreichend Leistung und Speicher, um schnelle Positionsergebnisse liefern können.
Gerade bei mobilen Endgeräten, wie z.B. Smartphones, steht eine solche Leistung nicht immer zur Verfügung, um alle eben genannten Schritte lokal auszuführen. Daher lagern einige Systeme Bestandteile der Analyse und des Vergleiches auf eine Serverkomponente aus.
Hile et al. verwenden in ihrem vorgestellten Indoor-Positionierungssystem eine einfache Handykamera zum Erzeugen von Standortbildern [151]. Da die Leistung des Handys zur Analyse und zum Vergleich von Standortbildern zu gering ist, setzten sie eine Serverkomponente ein.

Netzzugriff Sobald Bestandteile des Verfahrens auf eine Serverkomponente auslagert werden, muss eine Netzkommunikation möglich sein. Hierfür eignet sich das integrierte WLAN- oder Mobilfunkmodul des mobilen Endgerätes, welches so gut wie in jedem Smartphone, Tablet oder Laptop verbaut ist. Der Zugang über das Mobilfunkmodul hängt selbstverständlich von der Qualität und Netzabdeckung des Mobilfunkbetreibers ab. Neben der Bereitstellung des Netzzugriff ist die Datenmenge entscheidend, die jeweils für eine Anfrage benötigt wird. Dabei kommt es auf das jeweilige Verfahren an wie groß die Datenmenge ist, da entweder mehrere Standortbilder, ein einzelnes Standortbild oder nur die charakteristischen Eigenschaften als komprimierte Daten pro Anfrage über das Netz geschickt werden. Auch das Anfrage-Intervall kann das zur Verfügung gestellte Datenvolumen, insbesondere bei Mobilfunkverträgen, vollständig ausschöpfen. Das Anfrage-Intervall beeinflusst unter anderem die Genauigkeit der Positionsbestimmung. Daher muss ein ideales Verhältnis zwischen dem Anfrage-Intervall, der genutzten Datenmenge und der Positionsgenauigkeit gefunden werden.
Girod et al. [152] verwenden ebenfalls eine Serverkomponente, jedoch verarbeiten sie das Standortbild lokal auf dem Endgerät. Sie extrahieren die charakteristischen Bildeigenschaften und reduzieren damit die Datenmenge.

Datenbank Die Datenbank stellt die Grundlage zum Vergleich aller Standortbilder dar. Sie beinhaltet für jeden Standort, der durch das visuelle Positionierungssystem abgedeckt wird, einen Eintrag. Ein solcher Eintrag besteht meist aus einem oder mehreren Bildern des Standortes. Zusätzlich werden im Vorfeld extrahierte Bildeigenschaften, eine relative oder absolute Koordinate sowie weitere Metadaten dieser Bilder für einen Eintrag verwendet. Je nach Abdeckung des Positionierungssystem benötigt eine solche Datenbank sehr viel Speicher. Gleichzeitig steigt mit der Größe der Datenbank abhängig vom Vergleichsverfahren die Abarbeitungszeit immens an.
Auf mobilen Endgeräten ist der Speicher sowie die Leistung begrenzt, weshalb eine Auslagerung der Datenbank auf eine Serverkomponente sinnvoll ist.
Kawaji et al. verwenden für ihr visuelles Positionierungssystem pro Standort ein 360-Grad Bild, das mit einer omnidirektionalen Kamera erzeugt wird [153]. Eine Datenbank bestehend aus mehrere solcher Bilder wäre für den Speicher eines mobilen Endgerätes einfach zu groß.

4.2.2 Analyse des Standortbildes

Die einfachste Möglichkeit ein Standortbild mit einem anderen zu vergleichen, ist sie Pixel für Pixel mit einander zu vergleichen. Da die Bilder jedoch aus unterschiedlichen Perspektiven und Distanzen erzeugt werden, ist ein solcher Vergleich nicht zielführend. Daher werden charakteristische Bildeigenschaften verwendet, um Standortbilder miteinander vergleichen zu können, da sie robuster gegenüber Rotationen und Translationen des Bildes sind.

Nachdem ein Bild durch den Prozess der Standortaufzeichnung erzeugt wird, wird es auf seine charakteristischen Eigenschaften analysiert und entsprechend extrahiert, sodass diese für den Vergleichsprozess verwendet werden können.
Je nach visuellem Positionierungssystem werden unterschiedliche visuelle, charakteristische Eigenschaften verwendet. Die Wahl, welche Eigenschaften verwendet werden, hängt davon ab, ob die Analyse und Extraktion des Standortbildes bereits auf dem mobilen Endgerät ausgeführt oder auf einer Serverkomponente ausgelagert wird.
Es gibt unterschiedliche Verfahren zur Analyse und Extraktion charakteristischer Bildeigenschaften. Daher werden im Folgenden einige Verfahren mit entsprechenden Beispielen vorgestellt.

Kanten und geometrische Formen Einige Verfahren verwenden Kantendetektoren, um einfache Kanten, Ecken oder ganze geometrische Formen hervorzuheben (vgl. Abs. 2.2.2.3). Dabei werden nicht die Kanten und Ecken selbst verwendet, sondern ihre Konstellation sowie ihre Verhältnisse zueinander. Diese prägen die charakteristischen Eigenschaften eines Bildes. Abbildung 4.2.2 zeigt ein Beispiel für ein Standortbild und dessen charakteristischen Eigenschaften.

(a) (b)

Abbildung 4.1: Anwendung eines Kantendetektors auf ein Standortbild: (a) Standortbild vor der Anwendung des Kantendetektors - (b) Standortbild nach Anwendung des Kantendetektors: Türrahmen und Fensterrahmen werden deutlich hervorgehoben.

In [151] verwenden Hile et al. ebenfalls einen Kantendetektor zur Analyse und Extraktion charakteristischer Eigenschaften eines Standortbildes. Diese setzen ihr System grundsätzlich innerhalb von Gebäuden, speziell auf Fluren, ein. Sie nutzen die Tatsache aus, dass man durch die Anzahl an Türen und Eckwänden innerhalb eines Ganges Eindeutigkeiten erhalten kann. Diese ergeben sich durch die Verhältnisse und Konstellationen ihrer Kanten und Ecken. Sie

verwenden ausschließlich Verhältnisse und Konstellationen von Schnittkanten, um ein Standortbild zu charakterisieren.

Visuelle Features - Markante Merkmale Ein Nachteil, der durch den Einsatz eines Kantendetektors entsteht, wirkt sich in Umgebungen oder Standorten aus, die durch keine markanten Kanten auffallen. Dies hat Zufolge, dass ein solcher Standort nicht mehr eindeutig gefunden werden kann. Daher greift die Mehrzahl visueller Positionierungssysteme auf visuelle Features zur Charakterisierung eines Standortes zurück (vgl. Abs. 2.2.4). Die Feature-Verfahren haben vor allem den Vorteil, dass sie neben einfacher Kanten auch Kontrastverläufe und andere markante Bereiche in einem Bild berücksichtigen. Durch den Deskriptor, der die Umgebung eines Features beschreibt, kann jedes für sich selbst betrachtet werden und das Nichterkennen eines Features wirkt sich dadurch nicht zu sehr auf die Vergleichbarkeit von zwei Standortbildern aus. Neben dieser Eigenschaft sind die einzelnen Features meist rotations- und skaleninvariant. Das heißt, auch wenn das Standortbild aus einer anderen Perspektive oder Distanz zum Referenzbild erstellt wird, bleibt die Vergleichbarkeit trotzdem erhalten. Abbildung 4.2 zeigt ein Beispiel für die Anwendung eines Feature-Verfahrens auf ein Standortbild.

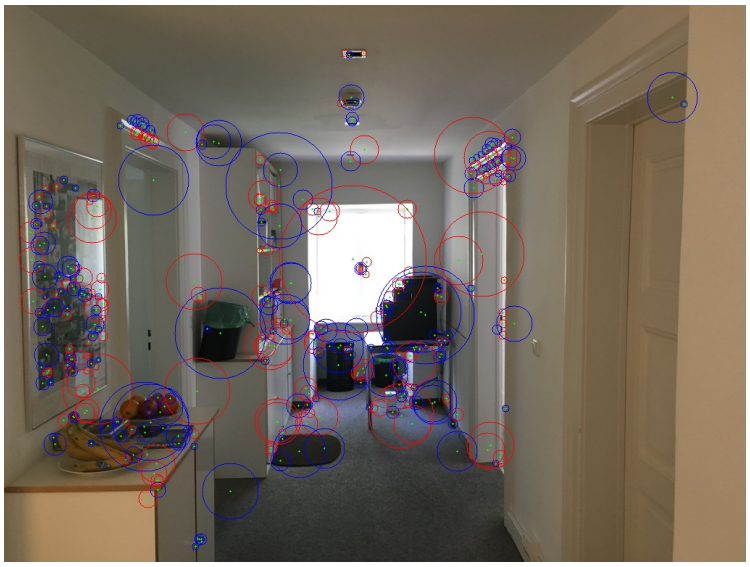


Abbildung 4.2: Anwendung eines Feature-Verfahrens auf ein Standortbild: Features werden auf dem Standortbild durch die Kreise hervorgehoben.

Die Autoren in [153] verwenden für ihr visuelles Posionierungssystem und zur Charatersierung der Bildeigenschaften das *PCA-SIFT* Feature-Verfahren [67]. Dieses ist eine komprimierte Variante des klassischen *SIFT* Feature-Verfahren [65] und beschleunigt durch seinen komprimierten Deskriptor die Vergleichsoperation von Features. Auch MoVIPS, das in Abschnitt 4.3 ge-

nauer vorgestellt wird, verwendet ein Feature-Verfahren basierend auf dem *SURF*-Algorithmus [68].

Objekterkennung Einen anderen Ansatz zur Bestimmung der Position stellen Verfahren dar, die eine Objekterkennung auf einem Standortbild einsetzen. Hierfür wird das Bild zur Positionsbestimmung auf einer abstrakteren Ebene betrachtet. Statt einzelne Kanten oder Features in einem Bild für einen Vergleich herzunehmen, werden ganze Objekte dafür verwendet. Eine Umgebung oder ein Standort wird durch seine enthaltenen und abgebildeten Objekte beschrieben. So kann z.B. ein Büroraum durch einen Schreibtisch, einen Bürostuhl und einen Monitor beschrieben werden [154]. Enthält nun ein Standortbild diese drei Objekte, wird der Büroraum als Standort identifiziert. Abbildung 4.3 zeigt ein Beispiel für einen Büroraum, der alle drei gesuchten Objekte enthält.

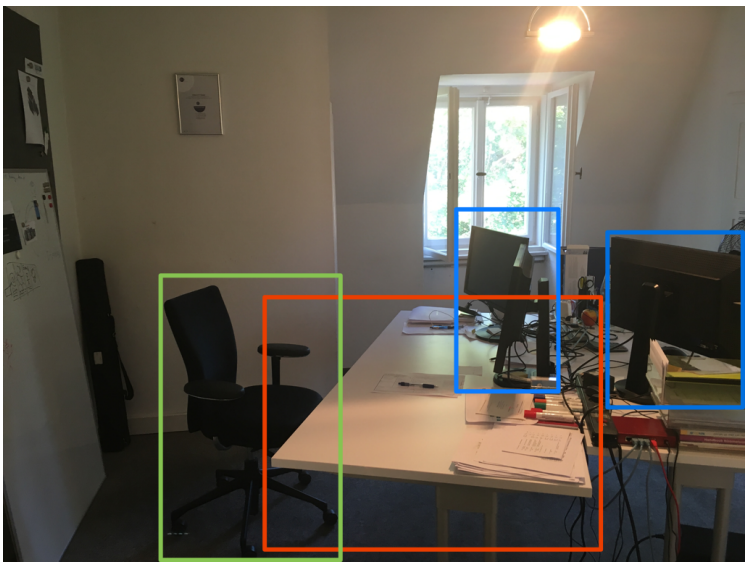


Abbildung 4.3: Standortbild eines klassischen Büroraums: Die Objekterkennung erkennt die drei Objekte Schreibtisch (rot), Bürostuhl (grün) und die Monitore (blau).

Die Objekterkennung selbst kann durch das Erkennen von Features, Kanten und Konturen umgesetzt werden. Der eigentliche Vergleich findet jedoch durch die Charakterisierung eines Raumes anhand seiner enthaltenen Objekte statt.

In [155] zeigen Collet et al., dass eine Erkennung mehrerer Objekte sowie die Bestimmung ihrer Pose innerhalb eines Bildes möglich ist. Torralba et al. zeigen sogar, dass die Objekterkennung für die Wiedererkennung von einzelnen Räumen oder ganzen öffentlichen Plätzen verwendbar ist [156].

4.2.3 Bildvergleich zur Positionsbestimmung

Der Bildvergleich stellt den wichtigsten und aufwendigsten Teilprozess im Bereich der visuellen Positionierung dar. Ziel des Bildvergleichs ist es, das richtige Standortbild aus einer gelabelten Menge zu finden. Dabei kommen unterschiedliche Verfahren zum Einsatz, die unter anderem von den vorherigen Prozessen, wie der Standortaufzeichnung und der Analyse des Standortbildes, abhängen. Die gelabelte Menge wird in einer Datenbank gehalten und kann neben dem eigentlichen Standortbild folgende zusätzliche Attribute pro Eintrag beinhalten:

- **Koordinaten** - die Position kann sowohl als relative als auch absolute Koordinate angegeben werden. Diese beziehen sich auf einen Lageplan oder auf geographische Koordinaten.
- **Charakteristische Bildeigenschaften** - zum Vergleich der Standortbilder werden eindeutige Merkmale benötigt. Diese können aus Konstellationen und Verhältnissen von Kanten, aus mehreren Features oder aus einer Zuordnung von Objekten zu einem Ort bestehen.
- **Fingerabdruck** - empfangene WLAN- und Bluetooth-Signale, die an einem Standort zu einen bestimmten Zeitpunkt aufgezeichnet worden sind, können zur groben Eingrenzung einer Position verwendet werden [157].
- **Blickrichtung** - die Blickrichtung, aus der ein Bild für einen Ort erzeugt wird, kann für die Wahl des richtigen Standortbildes entscheidend sein. Sie kann als Winkel oder auch Himmelsrichtung angeben werden.
- **Kontextinformation** - Anhand von Sensordaten können aktuelle Umweltbedingungen einer Umgebung erfasst und aufgezeichnet werden. Auch durch Aktivitäten, die durch Daten eines Beschleunigungssensors erfasst werden, kann ein Ort definiert werden. All diese Handlungszusammenhänge zu bestimmten Zeitpunkten können als Kontextinformationen eines Orts betrachtet werden. Diese wiederum können zur (relativen) Bestimmung eines Orts verwendet werden [158].
- **Semantik** - Die Bedeutung eines Ortes kann ebenfalls zur Lagebestimmung verwendet werden. Dabei kann ein Standort durch eine Beschreibung, eine Adresse oder eine Raumnummer definiert sein.

Der eigentliche Vergleich findet tatsächlich auf den charakteristischen Eigenschaften eines Bildes statt. Jedoch ist der Vergleichsprozess auf Basis der charakteristischen Bildeigenschaften bei einer hohen Anzahl an zu vergleichenden Standortbildern sehr zeitaufwendig. Daher setzen viele visuelle Positionierungssysteme beim Vergleichsprozess zunächst andere Attribute ein, die einen schnelleren Vergleich ermöglichen. Dadurch lassen sich bereits

im Vorfeld viele Einträge herausfiltern, sodass nur noch eine kleinere Menge vergleicht werden muss.

Die einfachste Variante zum Herausfiltern von Standortbildern lässt sich durch die Koordinaten und die Blickrichtung realisieren. Dabei kann angenommen werden, dass wenn einmal die eigene Position bekannt ist, eine zeitnahe Anfrage mit hoher Wahrscheinlichkeit in der Nähe der alten bekannten Position stattfindet. Mit dieser Annahme kann der Vergleich der Menge aller Standortbilder durch einen Radius begrenzt werden. Dieser kann abhängig von der zeitlichen Differenz, die zwischen der alten bekannten Position und der aktuellen Anfrage liegt, variabel gesetzt werden. Zusätzlich kann, falls beim Erzeugen des Standortbildes die Blickrichtung bekannt ist, diese zum Herausfiltern ebenfalls verwendet werden. Hierfür werden ausschließlich Standortbilder zum Vergleich herangezogen, die die selbe Blickrichtung aufweisen.

Eine weitere Möglichkeit die Vergleichsmenge im Voraus zu filtern, ist der Einsatz von WLAN-Fingerabdrücken. Hierfür können zur groben Positionierung die WLAN-Signale, die am aktuellen Standort empfangen werden, verwendet werden. Dabei gibt es zwei Möglichkeiten: (1) Es wird ein separates System eingesetzt, das auf Basis der WLAN-Signale die aktuelle Position ermittelt. (2) Die WLAN-Signale werden mit der Anfrage an das visuelle Positionierungssystem mitgeschickt und werden dann erst ausgewertet.
Die erste Lösung setzt den Einsatz eines weiteren Systems voraus. Damit sind zusätzliche Kosten verbunden, die durch eine Anschaffung und Wartung des Systems entstehen. Zwar gibt es bereits Dienstleister, die ein solches System inklusive Wartung bereitstellen, jedoch bleiben die Kosten erhalten und die Positionsgenauigkeit ist innerhalb von Gebäuden zu ungenau. Dagegen setzt die zweite Lösung voraus, dass die WLAN-Fingerabdrücke selbst aufgezeichnet werden und das Verfahren zur Ermittelung einer groben Position anhand der WLAN-Signale selbst implementiert wird.

Insgesamt gibt es verschiedene Varianten die Vergleichsmenge im Voraus zu filtern. Jedoch steht am Ende immer ein Vergleich auf Basis der charakteristischen Bildeigenschaften an. Dabei gibt es, insbesondere wenn visuelle Features für charakteristische Bildeigenschaften eingesetzt werden, Möglichkeiten den Vergleich durchaus zu beschleunigen. So setzen viele Verfahren auf den Einsatz quantisierter Features, um eine Bag-of-Words Repräsentation eines Standortbildes zu erhalten (vgl. Abs. 3.2.2.1). Clum et al. erweitern den Vergleich auf Basis der Bag-of-Words Repräsentation und setzen eine Variation des min-Hash-Algorithmus an, um die Suche bei sehr großen Datenmengen nochmals zu beschleunigen [159].

4.3 MoVIPS - Mobile Visual Indoor Positioning System

Das *Mobile Visual Indoor Positioning System*, kurz MoVIPS, ist am Lehrstuhl für Mobile und Verteilte Systeme der Ludwig-Maximillians-Universität München von Werner et al. entwickelt worden [39]. Es ist ein visuelles Positionierungssystem, das eine Position mit einer handelsüblichen Smartphonekamera bestimmen kann. MoVIPS nutzt zusätzlich WLAN-Fingerabdrücke, um die grobe Position des Smartphones zu ermitteln und dadurch die Suchanfrage auf einer Standortbilddatenbank zu beschleunigen. Die Standortbilddatenbank wird in einer sogenannten Offline-Phase erstellt, das heißt bevor das System in Betrieb geht. Dabei wird für jeden Standort ein Bild erzeugt und die exakte Position sowie die Blickrichtung auf einem Lageplan festgehalten. Zur Positionsbestimmung wird in der Online-Phase, also wenn das System betriebsbereit ist, anhand eines Ähnlichkeitsvergleiches ein Kamerabild mit der Standortbilddatenbank verglichen. Für den Ähnlichkeitsvergleich werden SURF-Features, die zur Beschreibung eines Standortbildes verwendet werden, extrahiert und eingesetzt (vgl. Abs. 2.3). Diese werden für die Standortbilder in der Datenbank im Vorfeld und für das aktuelle Kamerabild unmittelbar vor dem Vergleich extrahiert. Das Standortbild mit den meisten Übereinstimmungen wird als vorläufiger Kandidat ausgewählt. Anschließend wird die Position des ausgewählten Standortbildes einer zusätzlichen Korrektur unterzogen, sodass die geschätzte Position annähernd dem aktuellen Aufenthaltsort der Kamera entspricht. MoVIPS erreicht dadurch eine Genauigkeit von bis zu zwei Metern in der Positionsbestimmung.
Da dieses Kapitel Ansätze zur Erweiterung von MoVIPS vorstellt, wird im Folgenden das Verfahren zur Bildanalyse, zum Vergleich der Standortbilder und zur Korrektur der Position genauer vorgestellt. Anschließend werden die Schwachpunkte der einzeln Verfahren näher erläutert.

4.3.1 Bildanalyse

Jedes Standortbild wird mit dem Verfahren des SURF-Algorithmus auf visuelle Features analysiert. Dabei werden markante Merkmale aus dem Bild extrahiert. In MoVIPS werden ausschließlich der 64-stellige Deskriptor, der die Umgebung eines Features beschreibt, eine Bildkoordinate, die das Zentrum eines Features bestimmt, und eine Orientierung, die die Richtung des dominantesten Kontrastabfalls eines Features angibt, verwendet. Abbildung 4.4 zeigt die drei Attribute eines jeden Features, die durch MoVIPS aus dem Verfahren der Bildanalyse als Menge zurückgegeben wird. Jedes Standortbild wird durch eine Menge von SURF-Features aus jeweils drei Attributen beschrieben. Die Bildanalyse findet sowohl bei der Erstellung der Datenbank sowie bei einer Anfrage eines Standortbildes statt.
In der ursprünglichen Implementierung von MoVIPS wird das aktuelle Kame-

Abbildung 4.4: Abbildung eines SURF-Features auf einem Standortbild - Lokale Umgebung eines Features (blaues Viereck), Bildkoordinate (roter Punkt), Orientierung (grüne Linie).

rabild nicht auf dem Smartphone auf seine SURF-Features analysiert, sondern auf einer Serverkomponente, an die das Standortbild weitergeleitet wird.

4.3.2 Vergleichsverfahren

Das Vergleichsverfahren in MoVIPS basiert auf einem Ähnlichkeitsvergleich zwischen dem aktuellen Kamerabild und jedem einzelnen Standortbild aus der Datenbank. Die Ähnlichkeit zweier Bilder wird durch ihre Anzahl gemeinsamer Features bestimmt. Dabei wird die Euklidische Distanz zwischen zwei Features betrachtet. Ein Feature gilt somit als ähnlich, wenn es die kleinste Euklidische Distanz zu allen übrigen Features aufweist und der Abstand zu einem Feature mit der zweitkleinsten Euklidischen Distanz aus der Menge aller Features eines Bildes größer als 35% ist (vgl. Abs. 2.3.5). Jedes Kamerabild wird mit jedem Standortbild verglichen und die Anzahl an gemeinsamen Features bestimmt. Dieser Schritt wird ebenfalls in umgekehrter Reihenfolge mit jedem Standortbild und dem Kamerabild ausgeführt. Somit wird für ein Bildpaar insgesamt zweimal die Anzahl gemeinsamer Features bestimmt. Das Auswerten in umgekehrter Reihenfolge ist notwendig, da eine unterschiedliche Anzahl gemeinsamer Features bestimmt wird. Die Summe aus der Anzahl gemeinsamer Features aus beiden Richtungen gilt als endgültiges Ähnlichkeitsmaß zwischen zwei Bildern. Das Standortbild mit der größten Anzahl wird als aktueller Standort bestimmt und die hinterlege Position aus der Datenbank herangezogen. Die Position entspricht einer 2D-Koordinate auf einem Lageplan. Abbildung 4.5 zeigt ein Beispiel für den Vergleich von zwei Standortbildern.

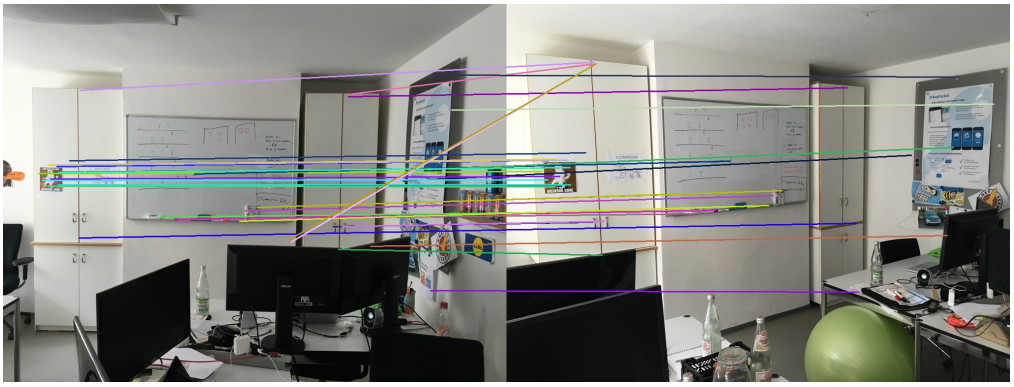


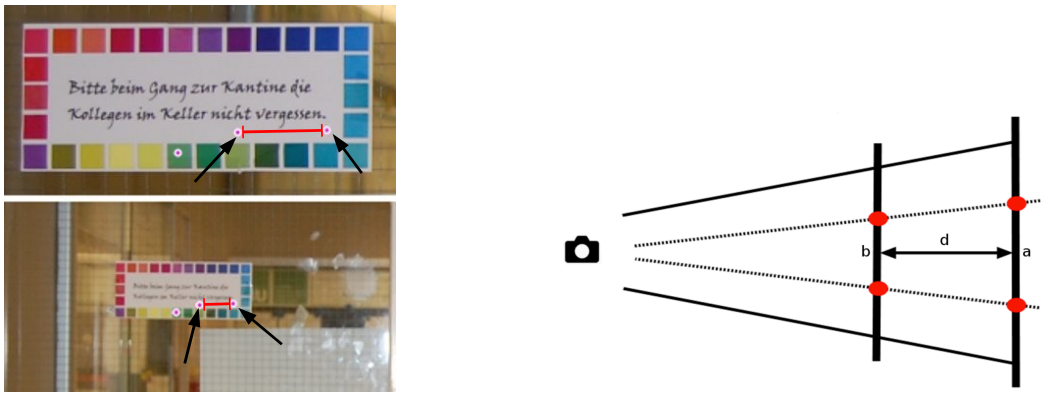
Abbildung 4.5: Vergleich von zwei Standortbildern auf Basis ihrer SURF-Features - ähnliche Features werden durch farbliche Linien verbunden.

4.3.3 Positionskorrektur

MoVIPS beinhaltet ein Verfahren zur zusätzlichen Positionskorrektur. Ein Kamerabild kann mit einem Standortbild aus der Datenbank übereinstimmen, jedoch muss es nicht aus derselben Distanz erzeugt worden sein. Die Motive auf dem Standortbild und dem Kamerabild stimmen überein, sind jedoch unterschiedlich skaliert. Das heißt, dass die aktuelle Position des Kamerabildes nicht genau mit der Position des Standortbildes übereinstimmt. Um diese Ungenauigkeit auszugleichen, wurde ein Verfahren zur Positionskorrektur entwickelt, welches mittels Strahlensatz die aktuelle Position korrigieren kann. Dafür muss jedoch vorausgesetzt werden, dass der Abstand zum Motiv eines jeden Standortbildes aus der Datenbank bekannt ist. In MoVIPS wird aus Gründen des höheren Aufwandes beim Erzeugen der Standortbilder für die Datenbank die Abstandsangabe zum Motiv weggelassen. Stattdessen wird zur Schätzung des tatsächlichen Abstandes des Motivs eine Annäherung durch eine lineare Regression wie folgt durchgeführt:

$$d = \alpha \frac{a}{b}$$

Dafür wird der Koeffizient α eingeführt, der mit Hilfe von mehreren gleichen Standortbildern mit unterschiedlichen bekannten Abständen zum Motiv empirisch ermittelt wird. Der tatsächliche Abstand d zu einem Motiv wird durch das Verhältnis a/b und dem Koeffizienten α bestimmt (vgl. Abb. 4.6b). Das Verhältnis a/b wird durch Paare korrespondierender Features in beiden Bildern bestimmt. Abbildung 4.6a zeigt ein solches Beispiel von einem Paar korrespondierender Features.
Da der Vergleich von SURF-Features auch falsch-positive Ergebnisse aufweist, kann es bei der Bestimmung der Verhältnisse zu verfälschten Ergebnissen kommen. Damit die korrigierte Position durch solche Ausreißer nicht stark beeinflusst wird, wird der mittlere Abstand über alle Paare korrespondierender



(a) (b)

Abbildung 4.6: Positionskorrektur anhand von Paaren korrespondierender Features [39] - (a) Zwei Bilder mit demselben Motiv mit unterschiedlichen Abständen. Korrespondierende Features haben jeweils eine andere Länge (rote Linie). (b) Anwendung des Strahlensatzes auf ein Paar korrespondierender Punkte auf zwei unterschiedlichen Ebenen (Bildern).

Features berechnet. Es hat sich gezeigt, dass der Durchschnitt aus dem harmonischen Mittel und dem Mittelwert für ein ausgeglichenes Ergebnis sorgt.

4.3.4 Problemstellungen von MoVIPS

Insgesamt kann MoVIPS die Position eines Standortbildes bis zu zwei Meter genau innerhalb von Gebäuden mittels einer handelsüblichen Smartphonekamera bestimmen. Diese hohe Genauigkeit wird jedoch nur erreicht solange eine hohe Abdeckung an Standortbildern für den Einsatzbereich existiert. Für den Einsatz des Systems sind ausschließlich eine mobile Anwendung, eine Serverkomponente und ein Netzzugang notwendig.
Trotz dieser guten Ergebnisse, birgt das System noch einige Schwächen. Besonders der hohe Zeitaufwand, der durch das Vergleichen eines Kamerabildes mit den Standortbildern aus der Datenbank entsteht, erzeugt sehr lange Anfragezeiten, sobald eine hohe Abdeckung durch Standortbilder besteht. Zwar wird in MoVIPS durch WLAN-Fingerabdrücke der Suchradius nach Standortbildern auf der Datenbank begrenzt, trotzdem kann die Anzahl an zu vergleichenden Standortbildern immer noch hoch sein. Des Weiteren besteht eine Abhängigkeit zu einem WLAN-basierten Positionierungssystem, dass zusätzliche Kosten durch Anschaffung, Betrieb und Wartung mit sich bringt.
Eine weitere Schwäche besteht bei der eigentlichen Anfrage der Position durch das Kamerabild. Dieses wird zwar durch die Smartphonekamera erzeugt, jedoch sofort an die Serverkomponente weitergeleitet, wodurch große Datenmengen über das Netz verschickt werden müssen. Das heißt, dass eine hohe Datenrate verfügbar sein muss und dadurch höhere Kosten entstehen können.

Insbesondere wenn der Einsatz von ganzen Bildsequenzen für das Vergleichsverfahren verwendet wird, ist eine Ausführung der Bildanalyse auf dem Smartphone notwendig, um die Datenmenge zu reduzieren.
Auch das Verfahren der Positionskorrektur hat seine Schwächen, da es nicht rotationsinvariant ist. Ein Kamerabild, das aus einem anderen Winkel erzeugt wird als das Standortbild und trotzdem den selben Abstand zum Motiv hat, erhält durch die Rotation der Features ein falsches Ergebnis der Abstandskorrektur. In schmalen Gängen, wie es in MoVIPS größtenteils der Fall war, ist eine solche Situation seltener anzutreffen, aber in größeren Räumlichkeiten, wie z.B. in einer Aula oder Mensa, führt dies zu Problemen.
Es lassen sich einige Schwächen des Systems aufzeigen, die zu einer Ungenauigkeit der Positionierung führen können. Daher werden in Abschnitt 4.4 einige Erweiterungen und Ansätze zur Verbesserung des Systems vorgestellt.

4.4 Erweiterungen des mobilen visuellen Positionierungssystem MoVIPS

Zahlreiche Entwicklungen und Errungenschaften aus dem Bereich der Positionierung und des maschinellen Sehens haben dazu geführt Verbesserungen und Erweiterungen für das *Mobile Visual Indoor Positioning System* zu entwickeln. Diese verbessern das System in seiner Effizienz und Genauigkeit. Es werden drei Ansätze vorgestellt, die die genannten Schwächen aus Abschnitt 4.3.4 weitestgehend beheben.
Neben den drei Ansätzen, die im Folgenden vorgestellt werden, wurde die Implementierung von MoVIPS im Rahmen dieser Arbeit überarbeitet. Dabei wurde die mobile Anwendung sowie die Serverkomponente des verteilten Systems neu entwickelt und durch weitere Module erweitert. Die mobile Anwendung wurde durch eine native Implementierung des SURF-Algorithmus vervollständigt, sodass die Bildanalyse bereits auf dem Endgerät stattfinden kann. Die Serverkomponente wurde vollständig in Java-Code umgeschrieben. Zusätzlich wurde die Kommunikation zwischen den einzelnen Entitäten des verteilten Systems durch wohldefinierte Schnittstellen vereinfacht.

4.4.1 Vergleichsverfahren: Schrittzähler und Kompass

Die erste Erweiterung beschleunigt das Vergleichsverfahren aus MoVIPS, indem es Sensordaten des Smartphones verwendet, um die Anzahl zur vergleichender Standortbilder aus der Datenbank vorfiltern zu können. Hierfür werden der Beschleunigungssensor und der integrierte Kompass des Smartphones eingesetzt. Anhand des Beschleunigungssensor können Schritte erkannt und gezählt werden [160]. Mit Hilfe des Kompass wird die Blickrichtung auf das fotografierende Motiv erfasst. Die Sensordaten ermöglichen eine Schätzung der

aktuellen Position, sodass ein Vorfiltern der Datenbank bei einer Standortabfrage umgesetzt werden kann.
Dafür müssen genau zwei Fälle in Betracht gezogen werden, nämlich:

1. Die Abschätzung der initialen Position.
2. Die darauffolgenden Positionsschätzungen, wobei die letzte Position bekannt sein muss.

Im ersten Fall lässt sich nur die Blickrichtung auf das Motiv des erzeugten Kamerabildes verwenden. Im zweiten Fall kann sowohl die Blickrichtung als auch der geschätzte Abstand zur vorherigen bekannten Position verwendet werden.
Im Folgenden wird das Verfahren unter Einsatz der ermittelten Blickrichtung sowie unter Einsatz der Bewegungsrichtung und Schrittzahl erläutert. Abschließend wird die Erweiterung des Vergleichsverfahrens anhand von MoVIPS evaluiert.

4.4.1.1 Blickrichtung

Beim Erzeugen eines Kamerabildes zur Bestimmung der Position, wird das Smartphone in Richtung des zu fotografierenden Motivs gehalten. Die Blickrichtung der Kamera kann durch einen digitalen Kompass, der in jedem handelsüblichen Smartphone verbaut ist, ermittelt werden. Die Blickrichtung wird in Grad angeben und folgendermaßen ausgerichtet: Ein Blick in Richtung des magnetischen Nordpol hat einen Winkel von 0 Grad und wächst mit einer Drehung im Uhrzeigersinn entsprechend an. Ein digitaler Kompass hat jedoch den Nachteil, dass er sehr empfindlich gegenüber magnetischen Störquellen und elektromagnetischer Strahlung ist, welche besonders in Gebäuden auftreten kann. Aus diesem Grund werden die Rohdaten des Kompasses zusätzlich mit Hilfe des integrierten Gyroskop geglättet. Trotz dieser Maßnahme können die Kompassdaten immer noch fehlerbehaftet und ungenau sein. Daher wird statt den diskreten Winkel einer Blickrichtung zu verwenden, ein bestimmter Toleranzbereich um den Winkel betrachtet.
Mit Hilfe des Toleranzbereich um den Winkel kann eine Vorfilterung auf der Standortdatenbank relativ einfach umgesetzt werden. Voraussetzung dafür ist, dass jeder Eintrag aus der Standortdatenbank eine Angabe zur Blickrichtung des abgebildeten Motivs beinhaltet (Aufnahmewinkel).
Unter der Voraussetzung einer Gleichverteilung der Aufnahmewinkel aller Einträge aus der Standortdatenbank, kann bei einem Toleranzbereich von ± 90 Grad die Hälfte aller Vergleiche im Vorfeld ausgeschlossen werden.

4.4.1.2 Bewegungsrichtung und Schrittzahl

Um die Bewegungsrichtung und die Anzahl der Schritte eines Nutzers zu schätzten, wird ebenfalls der integrierte Kompass und Beschleunigungssensor

des Smartphones verwendet. Der Beschleunigungssensor dient zum Erkennen und Zählen von ausgeführten Schritten. Sobald ein Schritt ausgeführt wird, bestimmt der Kompass die aktuelle Blickrichtung des Nutzers. Somit wird neben der Anzahl der Schritte für jeden einzelnen Schritt die Blickrichtung zusätzlich festgehalten. Die gelabelten Schritte werden anschließend auf ein Histogramm abgebildet, dass in 10 Grad Abschnitten unterteilt ist. Abbildung 4.7 zeigt ein solches Schritt-Histogramm, welches insgesamt aus 30 Schritten besteht.

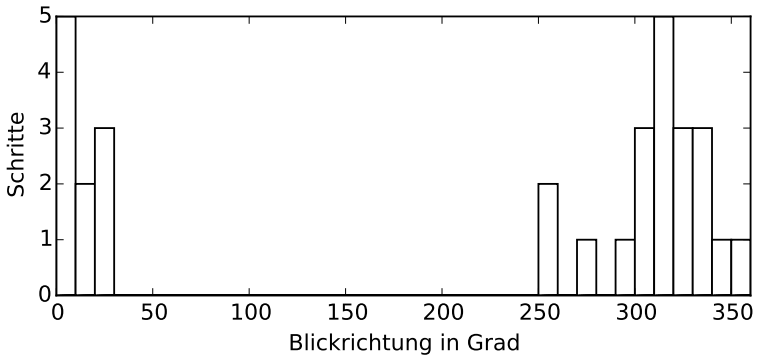


Abbildung 4.7: Schritt-Histogramm - 30 Schritte gruppiert anhand ihrer Bewegungsrichtung in 10 Grad Intervalle [14].

Das Schritt-Histogramm wird bei jeder Positionsanfrage mitverwendet. Es werden nur Standortbilder verglichen, die innerhalb der Histogrammabschnitte ausgehend von der letzten bekannten Position des Nutzers liegen (vgl. Abb. 4.8).

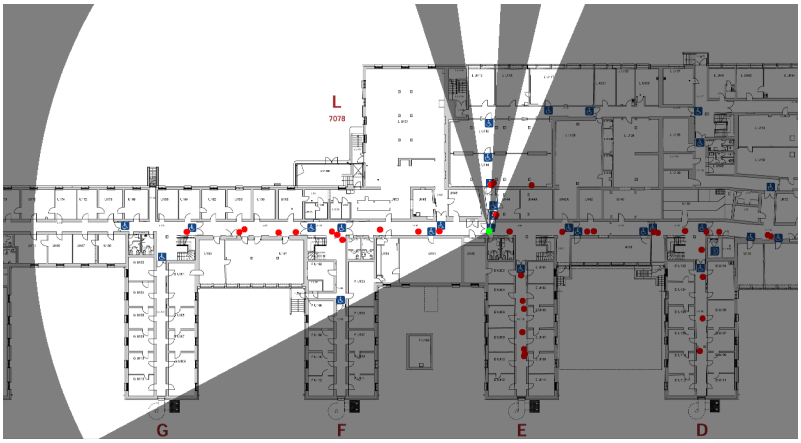


Abbildung 4.8: Abbildung des Schritt-Histogramms auf einen Gebäudeplan - die hell hervorgehobenen Bereiche beinhalten die zuvergleichenden Standortbilder aus der Datenbank (rote Punkte). Ausgehend von der letzten bekannten Position (grüner Punkt) werden die Bereiche eingezeichnet [14].

4.4.1.3 Evaluierung des Verfahrens

Zur Evaluierung des Verfahrens wurde eine Standortdatenbank bestehend aus 75 Einträgen erstellt. Hierfür wurden Standortbilder im Institutsgebäude der Informatik an der Ludwig-Maximilians-Universität München gemacht. Die Kameraaufnahmen wurden innerhalb des Ganges in einem Abstand von 1,5 bis 2 Meter erstellt, wobei immer zwei Bilder pro Position erzeugt wurden, die jeweils den Gang um 180 Grad rotiert abbilden. Auf Aufnahmen, die ausschließlich die Wände abbilden, wurde verzichtet. An kreuzenden Gängen wurden ebenfalls Standortbilder gemacht, wobei hierfür immer vier Bilder erzeugt wurden, die jeweils um 90 Grad rotiert den Gang abbilden.
Für die Testszenarien wurden insgesamt 20 Positionierungsdurchläufe erstellt. Jeder Durchlauf ist wie folgt aufgebaut:

- Es erfolgt eine erste Positionsbestimmung mittels Kamerabild
- Ein Nutzer läuft mit dem Smartphone 5 bis 15 Meter innerhalb des Gebäudes (Geh-Phase)
- Es erfolgt eine zweite Positionsbestimmung mittels Kamerabild

In 10 Positionierungsdurchläufen erfolgt die erste und zweite Positionsbestimmung jeweils an der selben Stelle. Zusätzlich wird in der Hälfte aller Durchläufe das Smartphone zwischen der ersten und zweiten Positionsbestimmung in der Hosentasche getragen und in der anderen Hälfte in der Hand gehalten.
Für die Positionierungsdurchläufe wird eine modifizierte MoVIPS Version eingesetzt. Hierfür wurde einerseits eine Server-seitige Erweiterung zur Vorfilterung der Datenbank hinzugefügt und anderseits die mobile Anwendung erweitert, sodass die aktuelle Blickrichtung einer Kameraaufnahme bei jeder Anfrage mitgeschickt wird. Das Verfahren der Positionskorrektur aus Abschnitt 4.3.3 wurde für die Evaluierung weggelassen.

Im Folgenden werden die Ergebnisse der Evaluierung verglichen. Dabei wird die ursprüngliche Vergleichsmethode *(brute force)* und die erweiterte Methode *(Blickrichtung und Schritt-Histogramm)* miteinander verglichen. Zusätzlich werden die Verfahren nochmal einzeln miteinander verglichen.

Durchschnittliche Vergleichsdauer Abbildung 4.9 zeigt die durchschnittliche Vergleichsdauer für die unterschiedlichen Methoden. Die Vergleichsdauer sinkt um 35% allein durch den Einsatz der Blickrichtung. Durch den Einsatz des Schritt-Histogramms sinkt die Vergleichsdauer um 53% im Vergleich zur ursprünglichen Methode. Die Kombination aus Blickrichtung und Schritt-Histogramm beschleunigt das Vergleichsverfahren um insgesamt 84%. Die Vergleichsdauer sinkt umso stärker, je höher die Filterbedingungen sind, da die Zeit linear von der Anzahl der Standortbilder aus der Datenbank abhängt.

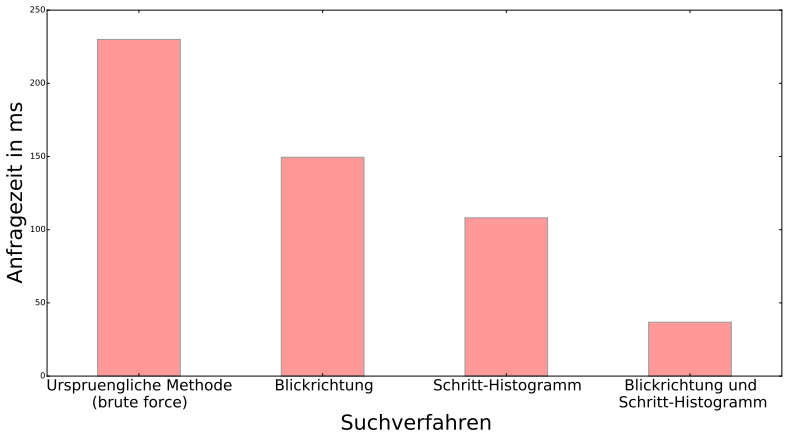
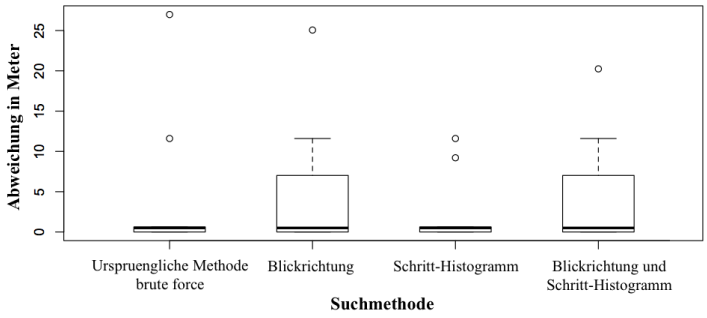


Abbildung 4.9: Vergleich der durchschnittlichen Verlgeichsdauer - (von links nach rechts) ursprüngliche Methode aus MoVIPS, Verfahren mit Blickrichtung, Verfahren mit Schritt-Histogramm und Verfahren mit Blickrichtung und Schritt-Histogramm [14].

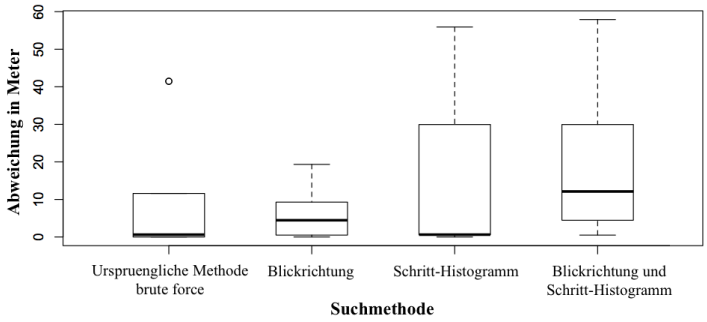
Vergleicht man die Positionsgenauigkeit im Vergleich zum ursprünglichen Verfahren, sind die Ergebnisse für die einzelnen Methoden *Blickrichtung* und *Schritt-Histogramm* mit einer Positionsgenauigkeit von jeweils 80% und 76% weitaus schlechter. Dies wird im Folgenden im Detail betrachtet.

Positionsgenauigkeit Die 20 Positionierungsdurchläufe sind in zwei Varianten aufgeteilt, bei denen das Smartphone während der Geh-Phase entweder in der Hand oder in der Hosentasche liegt. Die Ergebnisse der Positionsgenauigkeit werden für die zwei Varianten einzeln betrachtet. Die Evaluation hat gezeigt, dass die Position des Smartphones für die Genauigkeit der Positionsbestimmung einen signifikanten Einfluss hat. Hält man das Smartphone während der Geh-Phase von 5 bis 15 Metern in der Hand, sodass das Display immer auf das Gesicht des Nutzers gerichtet ist, ist die Positionsgenauigkeit weitaus höher als wenn es in der Hosentasche liegt. Abbildung 4.10a und Abbildung 4.10b zeigen die einzelnen Ergebnisse für die Positionsgenauigkeit jeweils für das Smartphone in der Hand und in der Hosentasche. Betrachtet man die Ergebnisse aus dem Verfahren mit dem *Schritt-Histogramm*, ist die Genauigkeit sogar höher als bei der ursprünglichen Methode. Die Verfahren mit der *Blickrichtung* und der Kombination aus beidem erzielen bessere Ergebnisse mit dem Smartphone in der Hand als wenn es in der Hosentasche liegt. Sobald das Smartphone in der Hosentasche liegt, sind die Ergebnisse weitaus schlechter. Die Kombination aus *Blickrichtung* und *Schritt-Histogramm* versagt sogar vollständig und erzielt kein genaues Ergebnis.
Eine Erklärung für diese Ungenauigkeit ist, dass das Smartphone in der Hosentasche während der Geh-Phase bei jedem erkannten Schritt die Blickrichtung nicht richtig bestimmen kann. Dies ist besonders für das Verfahren mit dem

Schritt-Histogramm nicht förderlich. Auch bei dem Verfahren zur Erkennung der Blickrichtung können Fehler entstehen. Da die Sensoren des Kompasses sehr anfällig sind, wird das Gyroskop unterstützend eingesetzt, um die Kompassdaten zu glätten. Dieser Prozess benötigt jedoch eine gewisse Zeit zur Kalibrierung. Wenn das Smartphone aus der Hosentasche gezogen wird, um ein Kamerabild zu erzeugen, kann wegen der fehlenden Kalibrierungszeit die tatsächliche Blickrichtung nicht bestimmt werden.



(a) Smartphone wird in der Hand gehalten.



(b) Smartphone liegt in der Hosentasche.

Abbildung 4.10: Boxplotdarstellung der Positionsabweichung in Meter für alle 20 Positionierungsdurchläufe. [14]

4.4.2 Vergleichsverfahren: Quantisierung und Reranking

Die zweite Erweiterung versucht ebenfalls das Vergleichsverfahren aus MoVIPS zu beschleunigen. Hierfür wird das ursprüngliche Vergleichsverfahren vollständig durch den mehrstufigen Vergleichsprozess aus Abschnitt 3.3 ersetzt. Es werden ausschließlich die erste und zweite Stufe des mehrstufigen Vergleichsprozesses verwendet. Das heißt, dass sowohl ein Vergleich auf Basis der quantisierten Features als auch ein Vergleich auf Basis der SURF-Features stattfindet. Die zweite Stufe des mehrstufigen Vergleichsprozesses wurde zusätzlich um ein

verbessertes *Reranking-Verfahren* erweitert, sodass der Vergleich auf Basis der Features ebenfalls beschleunigt wird.
Im Folgenden wird die Integration und Anpassung des Vergleichsverfahren beschrieben. Dann wird das verbesserte *Reranking-Verfahren* im Detail erläutert. Zuletzt wird die Erweiterung des Vergleichsverfahren anhand von MoVIPS evaluiert.

4.4.2.1 Vergleich

Das ursprüngliche Vergleichsverfahren aus MoVIPS vergleicht Standortbilder ausschließlich auf Basis ihrer SURF-Features. Da der Vergleichsprozess bei einer hohen Anzahl von Standortbildern sehr zeitaufwendig ist, findet zunächst ein Vergleich auf Basis der quantisierten Features statt. Dieser erfolgt durch den mehrstufigen Vergleichsprozess, der in Abschnitt 3.3 ausführlich beschrieben wird. Es werden ausschließlich die ersten beiden Stufen verwendet. Da in der ersten Stufe die SURF-Features quantisiert werden, ist ein Vokabular mit einer geeigneten Anzahl an Wörtern notwendig. Das Vokabular wird im Vorfeld aus einer Menge unterschiedlicher Standortbilder erzeugt. Die Standortbilder aus der Datenbank werden ebenfalls im Vorfeld vorverarbeitet. Dabei werden aus jedem Standortbild SURF-Features extrahiert und in der Datenbank gespeichert. Die extrahierten SURF-Features werden anschließend für jedes Standortbild quantisiert und zum Erzeugen einer Bag-of-Words Repräsentation verwendet (vgl. Abs. 3.2.2.1). Auch die Bag-of-Words Repräsentation wird für jedes Standortbild in der Datenbank gespeichert.

Bei einer Positionsnanfrage wird die Kameraaufnahme mit allen Standortbildern aus der Datenbank verglichen, indem aus ihr alle SURF-Features extrahiert und ebenfalls zu einer Bag-of-Words Repräsentation quantisiert werden. Anschließend wird die Bag-of-Words Repräsentation zum Ähnlichkeitsvergleich mit den Standortbildern herangezogen. Zuvor werden jedoch die Bag-of-Words Repräsentationen mit dem Gewichtungsschema *tf-idf* vorverarbeitet. Die Ähnlichkeit wird durch die Kosinus-Distanz eines Bag-of-Word Paares bestimmt. Die Standortbilder werden absteigend ihrer Ähnlichkeit sortiert. Diese sortierte Liste aus Standortbildern wird anschließend an die zweite Stufe weitergeben, die alle Standortbilder auf Basis ihrer SURF-Features vergleicht. Damit nicht alle Standortbilder und alle SURF-Features miteinander verglichen werden müssen, wird ein *Reranking-Verfahren* eingeführt, das die Anzahl der zu vergleichenden Features stark reduziert. Das Standortbild mit den meisten gemeinsamen Features wird als Standort zurückgegeben.

4.4.2.2 Reranking

Die Grundidee des *Reranking-Verfahrens* ist es, die Effizienz des Bag-of-Words Ansatzes mit der Präzision des direkten Feature-Vergleichs zu verbinden. Während zur groben Einstufung eines Standortbildes in der ersten Stufe keine hohe

Genauigkeit erforderlich ist, muss für die endgültige Entscheidung in der zweiten Stufe eine hohe Präzision gegeben sein. Damit nicht die komplette Menge aller Features eines Standortbildes betrachtet werden muss, wird daher nur eine Untermenge an Features für den Vergleich hergenommen. Um eine ausgewogene Untermenge für den Vergleich zu erhalten, wird eine sogenannte *stoplist* und ein Parameter q eingeführt. Die *stoplist* enthält die Indizes visueller Wörter eines Vokabulars, die nach einem bestimmten Kriterium ausgewählt werden. Alle Features, die einem visuellen Wort aus der *stoplist* zugeordnet sind, werden für den Vergleichsprozess aus der zweiten Stufe ausgeschlossen. Die *stoplist* wird nach folgendem Algorithmus zusammengestellt:

1. Sortiere jedes Wort w_i aus dem Vokabular absteigend nach der Anzahl aller zugeordneten Features N_{w_i}
2. Berechne die Summe N aus allen Features in der Datenbank
3. Füge den Index i eines Wortes der *stoplist* hinzu, solange folgende Bedingung gilt:

$$\sum_{i \in stoplist} N_{w_i} > q \cdot N$$

Durch das *Reranking-Verfahren* werden häufig vorkommende Features für den Vergleich ausgeschlossen, sodass möglichst viele markante und selten vorkommende Features übrig bleiben. Dadurch wird einerseits die Anzahl an Vergleichen reduziert und anderseits werden falsch-positive Ergebnisse von vornherein ausgeschlossen.

4.4.2.3 Evaluierung des Verfahrens

Für die Datenbank wurde ein eigener Datensatz erzeugt, dass aus insgesamt 192 Standortbildern besteht. Die Standortbilder wurden ebenfalls im Institutsgebäude der Informatik an der Ludwig-Maximilians-Universität München erstellt. Es wurden für jede Position 8 Kameraaufnahmen gemacht, die jeweils um 45 Grad verschoben sind. Die Motive der Standortbilder zeigen Gänge sowie Eingangsbereiche des Gebäudes. Es wurden aus allen Standortbildern insgesamt 443763 Features extrahiert.
Für die Evaluierung des Verfahrens wurde ein Testdatensatz mit nochmals 30 Kameraaufnahmen von unterschiedlichen Positionen im Gebäude erstellt. Dabei wurde darauf geachtet die Aufnahmen aus verschiedenen Kamerawinkeln und unter unterschiedlichen Lichtverhältnissen zu erzeugen.
Da das Extrahieren der SURF-Features ebenfalls Zeit in Anspruch nimmt und die Evaluierung ausschließlich das Vergleichsverfahren untersucht, werden die SURF-Features im Vorfeld aus dem Testdatensatz extrahiert und in die Datenbank gespeichert, sodass dies keinen Einfluss auf die Ergebnisse hat. Als

Grundlage wird MoVIPS eingesetzt und für die Evaluierung angepasst, sodass der Testdatensatz mit unterschiedlichen Parametern automatisiert evaluiert werden kann. Außerdem wird das Vergleichsverfahren entsprechend ausgetauscht und angepasst. Die Positionskorrektur wird für eine bessere Vergleichbarkeit der Positionsgenauigkeit herausgenommen.
Im Folgenden werden die Ergebnisse aus der Evaluierung für die einzelnen Stufen des mehrstufigen Vergleichsverfahrens, das in MoVIPS integriert wurde, einzeln betrachtet und vorgestellt.

Stufe 1: Bag-of-Word Für das Erzeugen des Vokabulars werden unterschiedliche Cluster-Größen mit $c = [32,64,128,256,512,1024,2056,4096,8192]$ verwendet. Dabei entspricht c der Anzahl an Wörter eines Vokabulars voc_c. Für jedes c wird das Clustering zur Erzeugung des Vokabulars dreimal ausgeführt, sodass immer das Vokabular mit der besten Positionsgenauigkeit verwendet wird. Hierfür wird das gewichtete Ergebnis mit der *tf-idf* verwendet.

	PF (m)	*t* (ms)	*PF tf-idf* (m)	*t* (ms)
Feature-Verfahren	8.14	63605	-	-
voc_{32}	15.12	23	12.23	20
voc_{64}	14.64	49	12.22	39
voc_{128}	15.59	98	14.82	81
voc_{256}	11.04	164	10.17	180
voc_{512}	11.01	380	10.05	352
voc_{1024}	11.19	786	10.57	761
voc_{2048}	12.30	1566	11.31	1561
voc_{4096}	13.04	3199	11.40	3158
voc_{8192}	12.63	6277	11.71	6199

Tabelle 4.1: Durchschnittliche Vergleichsdauer (t) und Positionsfehler (PF) für den Vergleich auf Basis der unterschiedlichen Größen des Vokabulars im Vergleich zum Feature-Verfahren. Der Positionsfehler wird jeweils als ungewichtet und gewichtet in Metern (m) angeben. Die Vergleichsdauer wird in Millisekunden (ms) angeben.

Tabelle 4.1 zeigt deutlich den positiven Effekt, der durch das Gewichten der Bag-of-Word Repräsentation entsteht. Es reduziert den Positionsfehler um 10% ohne merklich höheren Zeitaufwand. Trotzdem wird deutlich, dass der Vergleich auf Basis der Bag-of-Words Repräsentation immer noch einen höheren Positionsfehler im Gegensatz zu dem Vergleich auf Basis der Features hat. Das Ergebnis wird erst mit der nächsten Stufe des mehrstufigen Vergleichsprozesses verbessert.

Stufe 2: Reranking-Verfahren Die zweite Stufe des mehrstufigen Vergleichsprozesses reduziert den Positionsfehler, der durch die Quantisierung

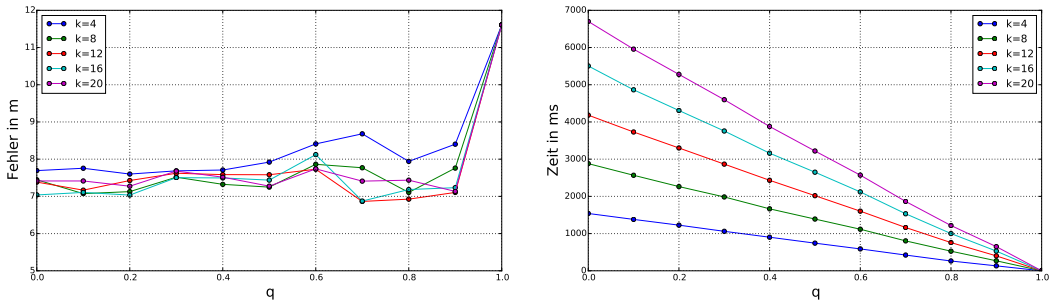
entsteht. Hierfür wird das *Reranking-Verfahren* eingeführt, dass nur eine Untermenge an Features für den Vergleich verwendet. Zusätzlich wird die Menge an zu vergleichenden Standortbildern begrenzt. Da bereits durch die Quantisierung der vorherigen Stufe falsche Standortbilder nach hinten sortiert werden, ist ein Vergleich dieser unnötig. Daher werden die ersten k Ergebnisse mit $k = [4, 8, 12, 16, 20]$, die aus der vorherigen Stufe zurückgegeben wurden, betrachtet. Für den Parameter q, der das Verhältnis der zu vergleichenden Features bestimmt, wird ein Wertebereich zwischen 0,0 und 0,9 verwendet. Es werden ebenfalls unterschiedliche Größen für das Vokabular voc_c mit c = [32,64,128,256,512,1024,2056,4096,8192] eingesetzt.

Tabelle 4.2 stellt exemplarisch für $k = 8$ den Positionsfehler nach Anwendung des *Reranking*-Verfahrens für die unterschiedlichen q dar. Die Ergebnisse zeigen sehr deutlich, dass der durchschnittliche Positionsfehler über alle Vokabelgrößen und unterschiedlichen Werte für q unter dem ursprünglichen Vergleichsverfahren liegt. Abbildung 4.11a zeigt zusätzlich den mittleren Positionsfehler aller Vokabelgrößen für unterschiedliche k und q. Selbst für ein $k = 4$ sind die Ergebnisse durch den Einsatz des *Reranking*-Verfahrens signifikant besser. Erst ab einem $k = 8$ werden die Ergebnisse für ein größeres k nicht besser. Betrachtet man den Effekt, der durch den Parameter q entsteht, wird deutlich, dass selbst für ein q = 0,9, also 10% der markantesten Features, ein gutes Ergebnis erzielt werden kann.

k = 8	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	∅
voc_{32}	6.52	6.52	6.52	6.35	7.36	7.36	7.01	6.67	6.12	8.01	6.85
voc_{64}	5.73	5.04	5.92	5.05	6.12	5.71	7.38	7.53	7.39	5.94	6.18
voc_{128}	6.61	6.61	6.18	6.37	6.37	6.56	5.45	6.92	6.35	10.41	6.78
voc_{256}	7.90	6.97	6.97	8.31	6.29	7.37	7.80	7.68	7.28	7.22	7.38
voc_{512}	7.46	7.86	7.80	7.81	7.20	7.53	8.11	7.84	7.58	9.30	7.85
voc_{1024}	8.50	6.91	7.18	9.46	7.79	6.53	8.15	7.38	8.24	9.66	7.98
voc_{2048}	8.99	8.49	8.53	8.79	8.94	8.95	10.64	8.97	6.89	6.21	8.54
voc_{4096}	6.76	6.41	6.41	7.63	7.63	7.46	8.91	9.03	7.05	6.17	7.34
voc_{8192}	8.54	8.85	8.64	7.88	8.20	7.77	7.29	7.90	7.06	6.92	7.90
∅	7.45	7.07	7.13	7.52	7.32	7.25	7.86	7.77	7.11	7.76	7.42

Tabelle 4.2: *Reranking*-Verfahren - Positionsfehler in Metern für $k = 8$ und unterschiedlichen q zwischen 0 und 0,9 unter Verwendung verschiedener Größen des Vokabulars.

Die Zeit, die für das *Reranking*-Verfahren verbraucht wird, ist einerseits von der Größe des Vokabulars und anderseits von den Parametern k und q abhängig. Abbildung 4.11b zeigt, dass der zeitliche Aufwand linear ansteigt sobald der Parameter q kleiner und der Parameter k größer wird. Dies ergibt sich aus der Anzahl an Vergleichsoperationen, die von der zu betrachtenden Menge an Standortbildern und der Anzahl an Features abhängt.



(a) (b)

Abbildung 4.11: *Reranking*-Verfahren durchschnittlicher Positionsfehler und zeitlicher Aufwand für unterschiedliche Werte für k und q - (a) durchschnittlicher Positionsfehler in Metern, (b) zeitlicher Auwand in Millisekunden. [14]

4.4.3 Positionskorrektur: Berücksichtigung der Rotation

Die dritte Erweiterung verbessert das Verfahren der Positionskorrektur in MoVIPS um die Eigenschaft der Rotationsinvarianz. Das ursprüngliche Verfahren betrachtet für ein Vergleichspaar alle gemeinsamen Features aus der Kameraaufnahme (Anfragebild) und dem Standortbild (Referenzbild) und misst paarweise die Distanzen zwischen jeweils zwei korrespondierenden Punktpaaren a und b, woraus das Verhältnis a/b berechnet wird. Dieses Verhältnis wird mit einer Konstanten α multipliziert, welche das Sichtfeld der Kamera beschreibt, um die korrigierte Position zu berechnen. Eine detaillierte Beschreibung des Verfahrens ist in Abschnitt 4.3.3 gegeben. Das ursprüngliche Verfahren hat jedoch einen Nachteil. Sobald der Blickwinkel auf das abgebildete Motiv des Anfragebildes und des Referenzbildes unterschiedlich ist, entstehen größere Fehler in der Positionskorrektur. Durch die Rotation werden die Features und ihre Bildkoordinaten auf einem zweidimensionalen Bild so verschoben, dass Verbindungslinien zwischen zwei Feature-Paaren verzerrt oder entzerrt werden. Ein solcher Effekt wird auch als perspektivische Transformation bezeichnet. Die Erweiterung setzt an dieser perspektivischen Transformation an und versucht die Rotation zwischen dem Anfragebild und Referenzbild zu bestimmen, um den Fehler der Positionskorrektur auszugleichen.

4.4.3.1 Perspektivische Transformation

Das Verfahren der Positionskorrektur wurde soweit erweitert, dass anhand der perspektivischen Transformation zweier Bildebenen der eigentliche Rotationswinkel bestimmt werden kann. Dies erfolgt durch eine Abschätzung der Homographie zweier Bilder [161]. Als Ergebnis erhält man eine perspektivische Transformationsmatrix H, mit dessen Hilfe eine genauere Position berechnet werden kann.

$$\sum_{i \in Matches} \begin{bmatrix} x_i^{'} \\ y_i^{'} \\ 1 \end{bmatrix} = H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

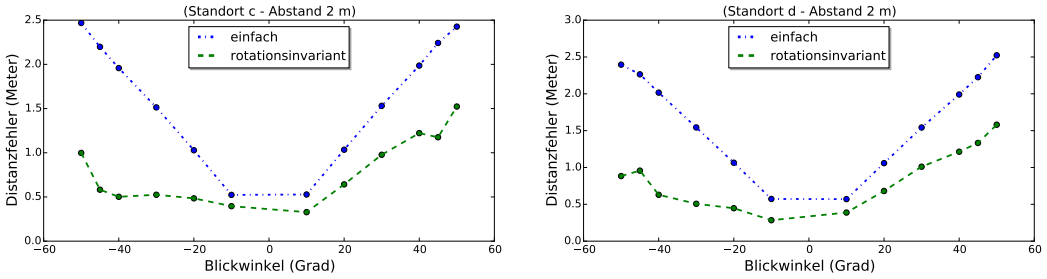
Zur Berechnung der Matrix H werden alle gemeinsamen Features ($Matches$) verwendet. Dafür werden aus allen *Matches* die Bildkoordinaten des Anfragebildes (x_i, y_i) sowie des Referenzbildes $(x_i^{'}, y_i^{'})$ eingesetzt, um eine initiale Homographie mittels eines Annäherungsverfahrens zu ermitteln. Das Annäherugnsverfahren benötigt mindestens vier gemeinsame Features, da sonst eine sinnvolle Berechnung der Homographie nicht möglich ist. Aus der daraus berechneten perspektivischen Transformationsmatrix H kann mittels einer Dekomposition die Rotationsmatrix ermittelt werden. Diese kann wiederum verwendet werden, um die drei Euler-Winkel *Roll (roll)*, *Nick (pitch)* und *Gier (yaw)* zu bestimmen. Aus der Kombination aus dem *Roll*-Winkel und dem Abstand, den man bereits mit dem ursprünglichen Verfahren erhält, lässt sich eine genauere Position berechnen.

4.4.3.2 Evaluierung des Verfahrens

Für die Evaluierung der Erweiterung wurde ein eigener Testdatensatz erzeugt, der aus insgesamt vier unterschiedlichen Standortmotiven besteht. Jeder Standort wurde mehrmals fotografiert, wobei der Abstand von 0 bis 8 Metern mit einem Intervall von einem Meter zunahm. Dadurch wurden insgesamt pro Standort 9 Bilder mit unterschiedlichem Abstand erzeugt. Zusätzlich wurde jedes Bild künstlich rotiert, sodass die Winkel zwischen $\pm 50 Grad$ abgedeckt werden. Insgesamt wurden damit 480 unterschiedliche Bilder erzeugt.
Um die Ergebnisse aus der erweiterten Variante der Positionskorrektur vergleichen zu können, wird sie mit der ursprünglichen Positionskorrektur aus MoVIPS sowie mit einer hybriden Variante für jedes Standortbild evaluiert.
Im Folgenden werden die Ergebnisse der rotationsinvarianten und der ursprünglichen Positionskorrektur miteinander verglichen. Anschließend wird eine hybride Variante der Positionskorrektur im Detail vorgestellt.

Rotationsinvariante und ursprüngliche Positionskorrektur Die Ergebnisse der Evaluierung haben gezeigt, dass die rotationsinvariante Positionskorrektur ein weitaus genaueres Ergebnis als die ursprüngliche Variante erzielt. Der mittlere Fehler mit der ursprünglichen Variante der Positionskorrektur liegt bei 2,93 Metern mit einer Standardabweichung von 1,99 Metern. Dagegen hat die rotationsinvariante Positionskorrektur einen mittleren Fehler von 2,50 Metern mit einer höheren Standartabweichung von 3,56 Metern. Der Unterschied zwischen beiden Varianten liegt nur bei 0,43 Metern. Schaut man sich jedoch die Ergebnisse im Detail an, hat die rotationsinvariante Version durchweg eine geringere Distanzfehlerquote. Abbildung 4.12 zeigt die Ergebnisse für unterschiedliche Bilder mit jeweils einem Abstand von 2 Metern für unterschiedliche

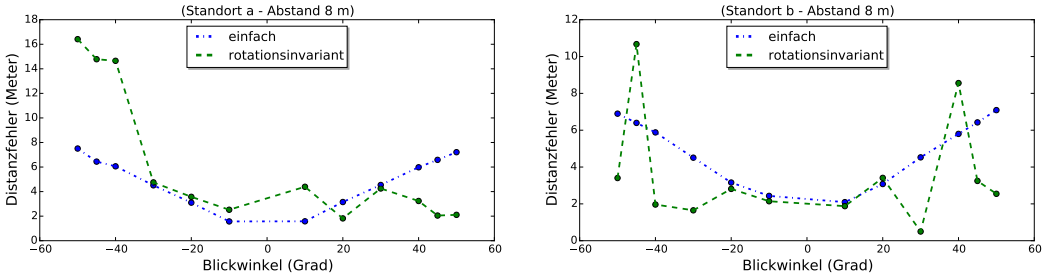
Blickwinkel zu einem Standortmotiv. Die höhere Standardabweichung entsteht durch wenige Ausreißer, die das Gesamtergebnis allgemein verschlechtern.



(a) (b)

Abbildung 4.12: Vergleich der ürsprünglichen und rotationsinvarianten Positionskorrektur - (a) Distanzfehler für Standort *c* mit einem Abstand von 2 Metern - (b) Distanzfehler für Standort *d* mit einem Abstand von 2 Metern. [14]

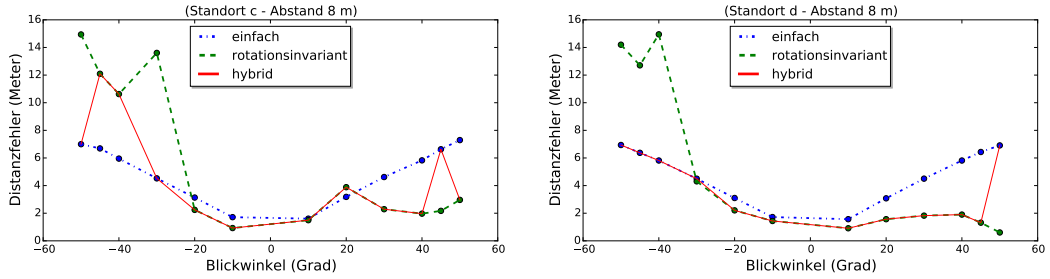
In Abbildung 4.13 wird deutlich, dass die Fehlerrate für die rotationsinvariante Positionskorrektur mit einem höheren Abstand zum Motiv und einem größeren Blickwinkel extrem ansteigt. Sie verhält sich weitaus schlechter als die ursprüngliche Methode aus MoVIPS. Die falsche Berechnung der Position entsteht durch eine fehlerhafte Abschätzung der Homographie, welche durch das Annäherungsverfahren verursacht wird. Das Annäherungsverfahren ist abhängig von der Anzahl gemeinsamer Features. Desto mehr gemeinsamer Features für das Annäherungsverfahren verwendet werden, desto genauer ist die Abschätzung. Mit zunehmenden Abstand sinkt jedoch die Anzahl gemeinsamer Features.



(a) (b)

Abbildung 4.13: Vergleich der ürsprünglichen und rotationsinvarianten Positionskorrektur - (a) Distanzfehler für Standort *a* mit einem Abstand von 8 Metern - (b) Distanzfehler für Standort *b* mit einem Abstand von 8 Metern. [14]

Hybrider Ansatz einer Positionskorrektur Während der Evaluierung haben einige Ergebnisse ein auffälliges Verhalten bezüglich des hohen Distanzfehlers bei zu großen Abständen zum Motiv eines Standortes aufgewiesen. Die Daten haben gezeigt, dass in fast allen Fällen, bei denen die Homographie falsch bestimmt wurde, ein viel zu hoher Euler-Winkel *Roll* bestimmt wurde. Dabei weisen die falsch berechneten Winkel einen Wert von über 90 Grad auf. Da solche Winkel sehr unwahrscheinlich sind und so gut wie nie vorkommen, können diese durch eine obere Schranke eindeutig identifiziert und ausgeschlossen werden.
Daher wurde ein hybrider Ansatz entwickelt, der bei einem Überschreiten des Winkels einer obereren Schranke von der rotationsinvarianten Methode auf die ursprüngliche Methode von MoVIPS wechselt. Im Rahmen der Evaluierung hat sich ergeben, dass ein Winkel von 70 Grad für eine solche Schranke geeignet ist. Insgesamt verbessern sich die Ergebnisse durch den hybriden Ansatz soweit, dass die Distanzfehlerrate auf 2,21 Meter und die Standartabweichung auf 2,44 Meter sinkt. Trotz des hybriden Ansatzes ist die Standartabweichung immer noch etwas höher als bei der ursprünglichen Methode aus MoVIPS. Abbildung 4.14 zeigt zwei unterschiedlichen Beispiele mit den Ergebnisse der Distanzfehlerrate des hybriden Ansatzes im Vergleich zu der rotationsinvarianten und ursprünglichen Methode. Dabei fällt auf, dass der hybride Ansatz besonders bei hohen Rotationswinkeln nicht immer auf das richtige Verfahren wechselt und dadurch der Distanzfehler für einige Blickwinkel immer noch zu groß bleibt. Trotz dieser wenigen Ausreißer ist das Ergebnis der Distanzfehlerrate für den hybriden Ansatz insgesamt besser als bei den anderen Varianten.



(a) (b)

Abbildung 4.14: Hybrider Ansatz der Positionskorrektur - (a) Distanzfehler für Standort *c* mit einem Abstand von 8 Metern - (b) Distanzfehler für Standort *d* mit einem Abstand von 8 Metern. [14]

4.5 Zusammenfassung

Dieses Kapitel beschäftigte sich mit der Vorstellung von drei essentiellen Erweiterungen und Verbesserungen für das visuelle Indoor-Positionierungssystem

MoVIPS. Speziell das Vergleichsverfahren sowie die Positionskorrektur aus MoVIPS werden durch die Erweiterungen bezüglich ihrer Genauigkeit und Effizienz weiterentwickelt und verbessert.
Die erste Erweiterung beschleunigt das Vergleichsverfahren um 84%, indem es die Standortdatenbank anhand von Odometriedaten vorfiltert. Die Odometriedaten werden anhand des Beschleunigungssensors, des Gyroskops und des Kompasses eines Smartphones ermittelt. Durch die Vorfilterung wird die Anzahl der zu vergleichenden Bilder möglichst klein gehalten.
Die zweite Erweiterung beschleunigt ebenfalls den Vergleichsprozess, indem es das ursprüngliche Verfahren durch das mehrstufige Vergleichsverfahren aus Abschnitt 3.3 ersetzt. Dabei wurde das mehrstufige Vergleichsverfahren angepasst und auf die ersten beiden Stufen beschränkt. Durch die Kombination quantisierter Features für den Vergleich in der ersten Stufe sowie durch ein neu entwickeltes *Reranking*-Verfahren in der zweiten Stufe wird das Vergleichsverfahren bei gleichbleibender Genauigkeit enorm beschleunigt.
Die dritte Erweiterung verbessert das ursprüngliche Verfahren der Positionskorrektur aus MoVIPS. Es erweitert das Verfahren um die Fähigkeit der Rotationsinvarianz. Dadurch funktioniert eine Korrektur der Position auch dann noch, wenn das Eingabebild zum Referenzbild in einem anderen Winkel und Distanz erstellt wurde. Dafür wurden zwei Varianten entwickelt, die sowohl aus einer reinen rotionsinvarianten als auch einem hybriden Lösungsansatz bestehen. Durch die Erweiterung wurde gezeigt, dass die Genauigkeit des Verfahrens der Positionskorrektur um 33% steigt.
Insgesamt können die drei Erweiterungen als sinnvoll betrachtet und als geeignete Erweiterungen für das visuelle Indoor-Positionierungssystem MoVIPS eingesetzt werden.
Trotzdem gibt es für MoVIPS noch zahlreiche andere Erweiterungsmöglichkeiten, um es effizienter und besser zu machen. So ist z.B. die Positionsgenauigkeit nicht nur abhängig vom Vergleichsverfahren und der Positionskorrektur, sondern auch von der Größe der Standortdatenbank. Eine solche Datenbank für einen größeren Gebäudekomplex zu erzeugen und aktuell zu halten, kann für die Integration von MoVIPS eine Hürde darstellen. Um die Kosten einer Standortdatenbank überschaubar zu halten, bietet es sich daher an Verfahren aus dem Bereich der Koppelnavigation und der Odometrie einzusetzen, um die Position bereits vor einer Standortabfrage abzuschätzen. Hierfür können neben der klassischen Verwendung des Beschleunigungssensor und des Kompass zu Bestimmung der Eigenbewegung und Position auch die Bildinformationen der Kamera verwendet werden. Dadurch kann die Anzahl an Anfragen durch Standortbilder merklich reduziert werden und eine hohe Abdeckung an Standortbildern in der Datenbank muss nicht mehr vollständig geben sein.

5 Visuelle Odometrie zur Bestimmung der Eigenbewegung und Aktivität

Bereits im vorangegangen Kapitel wurde die Bedeutung der Positionsbestimmung für ortsbezogene Dienste erläutert. Speziell Positionsverfahren, die visuelle Daten aus optischen Sensoren wie einer Kamera verwenden, sind in den letzten Jahren sehr populär geworden. Optische Sensoren sind günstig zu produzieren und bereits in vielen mobilen Geräten, wie z.B. Samrtphones und Tablets, verbaut. Das ist mit ein Grund, weshalb sie für Aufgaben der Lokalisierung und Navigation in kleinen Robotern, Drohnen *(Micro Aerial Vehicle)* oder auch im Kontext mit Menschen eingesetzt werden. Visuelle Verfahren, speziell aus dem Bereich der visuellen Odometrie, haben den Vorteil ohne jegliche künstliche Infrastruktur zu funktionieren. Eine visuelle Positionierung, wie sie in Kapitel 4 durch das visuelle Positionierungssystem MoVIPS bereits vorgestellt wurde, setzt Bildvergleiche zwischen aktuellen Kameraaufnahmen und einer Menge von Standortbildern ein. Hierfür ist immer eine Datenbank bestehend aus vielen unterschiedlichen Standortbildern notwendig, um eine Positionierung für einen größeren Bereich zu ermöglichen. Die Positionsgenauigkeit ist dadurch immer abhängig von der Aktualität der Datenbank sowie von der Anzahl an unterschiedlichen Standortbildern. Nachteilig ist auch, dass eine Serverkomponente zur Verfügung gestellt werden muss, um eine Standortdatenbank bereitzustellen oder Standort-Anfragen auszuwerten. Dies setzt zusätzlich voraus, dass ein Netzzugang bereitgestellt werden muss.

Dagegen benötigen Verfahren aus dem Bereich der visuellen Odometrie keinerlei Datenbank oder Vorwissen über einen zu lokalisierenden Bereich, um eine erfolgreiche Positionierung durchzuführen zu können. Anhand von visuellen Features können Verfahren der visuellen Odometrie die Eigenbewegung der Kamera bestimmen, um daraus anschließend die eigene Position zu berechnen. Auch die Ausführung des Verfahrens kann lokal auf einem mobilen Endgerät stattfinden, sodass eine Serverkomponente und ein Netzzugang wie in MoVIPS nicht mehr notwendig sind. Damit eine absolute Position ermittelt werden kann, muss zuvor eine bekannte Startposition vor Beginn jeder Messung angegeben werden. Dadurch kann ausgehend von der bekannten Startposition eine Trajektorie mit absoluten Koordinaten geschätzt und rekonstruiert werden.

Erste Ansätze der visuellen Odometrie haben ihren Ursprung im Bereich der

Raumfahrttechnik. Speziell für die Mond- und Mars-Rover wurden solche visuellen Verfahren entwickelt [162, 115]. Verfahren wie die satellitengestützte Positionierung und Navigation sind wegen fehlender Satelliten auf Mond oder Mars nicht möglich. Ebenfalls ist eine Positionierung mittels klassischer Odometrie basierend auf dem Antriebssystem zu ungenau, da die Beschaffenheit der Mond- und Marsoberfläche zu sandig und staubig ist.
Mittlerweile werden solche Verfahren auch unter irdischen Bedingungen verwendet, um bewegliche Objekte oder einzelne Personen zu positionieren und zu navigieren [163]. Durch die Einführung einer neuen Geräteklasse, der sogenannten *Wearables*, wie z.B. der *Google Glass*[1] oder *Microsoft Holo Lens*[2], sind sinnvolle Anwendungsbereiche der visuellen Odometrie entstanden. Neben der Positionierung und Navigation werden Verfahren der visuellen Odoemtrie auch zur Posebestimmung einer Kamera verwendet, um z.B. augmentierte virtuelle Objekte realitätsgetreu in einem Kamerabild in Echtzeit anzeigen zu können [1, 2].
In diesem Kapitel wird ein visuelles Odometrie-Verfahren vorgestellt, das auf Basis einer am Oberkörper befestigten Kamera die Trajektorie einer Person rekonstruieren kann und gleichzeitig die aktuelle Aktivität eines Nutzers zu jedem Zeitpunkt bestimmt.
Das visuelle Odometrie-Verfahren nutzt dafür einen visuellen Schrittzähler und einen visuellen Kompass, der auf Basis von Videosequenzen funktioniert. Dabei werden die einzelnen Bilder auf SURF-Features analysiert und alle Features eines Bildpaares miteinander verglichen (vgl. Abs. 2.3). Anhand der gemeinsamen Features und ihrer Attribute werden Muster zur Erkennung von einzelnen Schritten sowie Bewegungsrichtungen bestimmt. Die Aktivitätserkennung nutzt ebenfalls die Attribute gemeinsamer Features, um einzelne Aktivitäten wie *Laufen*, *Gehen* und *Stehen* zu identifizieren. Neben den einfachen Aktivitäten werden zusätzlich weitere untersucht und evaluiert.
Das Kapitel ist wie folgt aufgebaut. Zunächst werden die Grundlagen aus dem Bereich der visuellen Odometrie vorgestellt. Zur besseren Einordnung werden dann Verfahren der Koppelnavigation und Aktivitätserkennung beschrieben, da diese für die in diesem Kapitel beschriebenen Ansätze eine hohe Relevanz haben. Dann wird ein eigener Ansatz zur visuellen Odometrie sowie ein eigener Ansatz einer visuellen Aktivitätserkennung vorgestellt und beschrieben. Beide Ansätze werden anschließend evaluiert und diskutiert. Zuletzt wird eine Zusammenfassung über das gesamte Kapitel geben.

5.1 Eigene Vorveröffentlichungen

Die Kerninhalte dieses Kapitels wurden vom Autor bereits in [15, 18, 16, 17] publiziert. Wie in Kapitel 1.3 dargestellt, stammen die im Paper und im Fol-

[1] www.google.com/glass/start/
[2] www.microsoft.com/microsoft-hololens/en-us

genden präsentierten Inhalte bzgl. der Idee, des Konzepts und der Evaluation vom Autor der vorliegenden Arbeit. Ebenfalls in den Vorveröffentlichungen bereits enthalten sind die Abbildungen 5.9a, 5.10, 5.12, 5.13, 5.15 und 5.18. Die Inhalte bzgl. der Aktivitätserkennung mit Klassifikatoren sowie die darauf aufbauende Evaluierung (vgl. Abschnitt 5.5) sind in den Vorveröffentlichungen nicht enthalten.

5.2 Grundlagen der visuellen Odometrie

Allgemein setzen Verfahren der visuellen Odometrie Bildinformationen einer oder mehrerer Kameras ein. Ziel der visuellen Odometrie ist es, die Eigenbewegung allein aus einer Sequenz von aufeinanderfolgenden Kamerabildern abzuschätzen. Erste Arbeiten der visuellen Odometrie sind durch die Mond- und Mars-Rover Missionen motiviert [162, 115]. Weil Verfahren der klassischen Odometrie, die die Eigenbewegung anhand des Antriebssystems bestimmen, auf der sandigen und steinigen Oberfläche fehleranfällig waren, wurden visuelle Verfahren eingesetzt.
Eine der ersten Pionierarbeiten der visuellen Odometrie ist durch die Forschung von Hans Moravec bereits in den frühen 1980er Jahren vorangetrieben worden [164]. Dabei hat er eine Kamera auf einem Fahrzeug fixiert, das auf Gleisen vor und zurück geschoben werden konnte, um einzelne Kameraaufnahmen zu erzeugen und anhand dieser Bilder die Eigenbewegung des Fahrzeuges abzuschätzen. Der Begriff der visuellen Odometrie wurde jedoch erst durch die herausragende Arbeit von Nister et al. geprägt, die sich von der klassischen Odometrie inspirieren lassen haben [117]. Sie haben ein Verfahren entwickelt, das anhand einer Stereo-Kamera und den dadurch enthaltenen Bildinformationen die Eigenbewegung exakt bestimmen kann. Heutzutage wird die visuelle Odometrie vermehrt in autonomen Fahrzeugen und in unbemannten Drohnen eingesetzt [165, 166].
Im Folgenden werden die essentiellen Methoden der visuellen Bewegungsabschätzung vorgestellt. Anschließend werden Beispiele einiger Verfahren und Systeme, die die visuelle Odometrie umsetzen, präsentiert.

5.2.1 Methoden der Bewegungsabschätzung

Es gibt zahlreiche und unterschiedliche Methoden, die eine Eigenbewegung anhand von Bildsequenzen abschätzen können. Insgesamt lassen sich diese Methoden in zwei Klassen unterteilen, nämlich in die *Feature-basierten* und in die *direkten* Methoden. Klassischerweise basieren die meisten Verfahren auf der Auswertung visueller Features [167, 168, 169, 170]. Dabei wird die relative Bewegung von korrespondierenden Features auf zwei aufeinanderfolgenden Kameraaufnahmen zur Bewegungsabschätzung herangezogen. Erst in jüngster Zeit werden vermehrt auch direkte Methoden zur Bewegungsabschätzung eingesetzt. Diese Methoden analysieren die Veränderung von Intensitätswerten

ganzer Bereiche aufeinanderfolgender Bilder.
Im Folgenden werden zunächst die Feature-basierten und anschließend die direkten Methoden vorgestellt.

5.2.1.1 Feature-basierte Methoden

Die Feature-basierten Methoden teilen das Problem zur Abschätzung der Eigenbewegung in drei separate Schritte auf [163]:

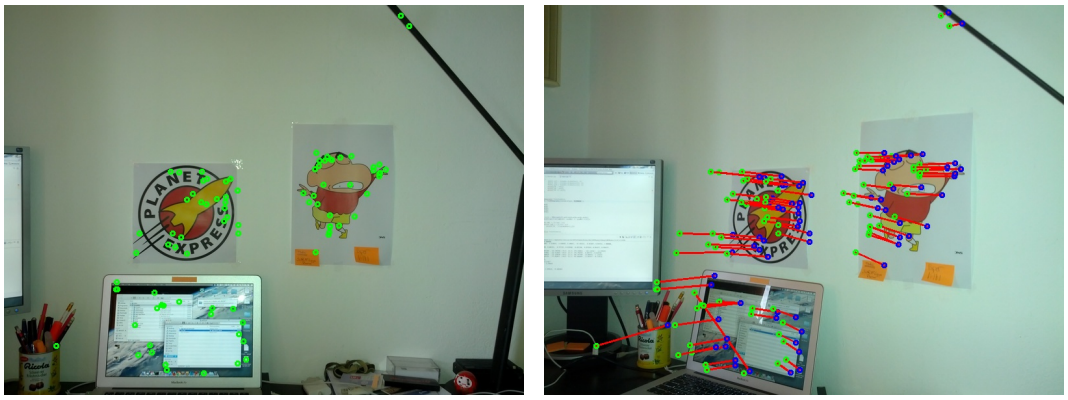
1. Zunächst wird eine Videosequenz analysiert. Dafür werden aus jedem einzelnen Bild der Sequenz Features extrahiert und mit den Features der anderen Bilder verglichen. Features, die in mindestens zwei Bildern auftauchen, werden für den nächsten Schritt verwendet.

2. Im zweiten Schritt dienen diese Features als Grundlage zur Bestimmung der groben Eigenbewegung. Die relative Bewegung zwischen zwei korrespondierenden Feature-Koordinaten wird zur Berechnung der aktuellen Position verwendet.

3. Im dritten Schritt findet eine nachträgliche Optimierung der Eigenbewegungsschätzung statt. Hierfür werden die Ergebnisse aus mehr als zwei Bildern einer Videosequenz verglichen und dadurch das Ergebnis verbessert.

Für den ersten Schritt ist die richtige Wahl des Feature-Verfahrens entscheidend, da das Auswirkungen auf die Qualität der Bewegungsabschätzung hat. Feature-Verfahren, die robust gegenüber Transformationen sind, erbringen gute Ergebnisse, jedoch sind sie meist nicht effizient genug und können für den Einsatz unter Echtzeitbedingungen ungeeignet sein.
Für den zweiten Schritt ist die Anzahl an Kameras entscheidend. Die grobe Abschätzung der Eigenbewegung kann durch unterschiedliche Art und Weisen erfolgen. Diese kann bei mindestens zwei Kameras so umgesetzt werden, dass die Schätzung der Eigenbewegung anhand von korrespondierenden 3D-Feature-Koordinaten zweier Bildern bestimmt werden kann. Wird nur eine Kamera eingesetzt, gibt es zwei Möglichkeiten. Die erste Variante schätzt die Eigenbewegung anhand von korrespondierenden 2D-Feature-Koordinaten zwischen zwei Bildern ab. Dagegen verwendet die zweite Variante ein etwas komplexeres Verfahren. Hierfür werden durch die vorangegangene Schätzung der Eigenbewegung die 3D-Feature-Koordinaten des hervorgegangenen Bildes trianguliert und aus den 2D-Feature-Koordinaten des darauffolgenden Bildes die Eigenbewegung bestimmt. Die Schätzung der Eigenbewegung auf Basis von 3D-Feature-Koordinaten führt immer zu einem genaueren Ergebnis. Die Triangulation der 3D-Feature-Koordinaten anhand einer Sequenz von Eigenbewegungen ist sehr rechenintensiv und daher nicht für alle Systeme geeignet.

Der letzte Schritt kann prinzipiell weggelassen werden und ist für die eigentliche Bestimmung der Eigenbewegung nicht notwendig. Trotzdem sollte von der Optimierung Gebrauch gemacht werden, da Ausreißer, die durch Rauschen oder Ungenauigkeiten des Feature-Verfahrens entstehen, das Gesamtergebnis enorm verfälschen können. Insbesondere beim Einsatz einer monokularen Kamera ist eine nachträgliche Korrektur notwendig, da durch die fehlenden Tiefeninformationen zusätzliche Ungenauigkeiten entstehen können [171].
Abbildung 5.1 zeigt korrespondierende Features, die aus zwei aufeinanderfolgenden Bildern einer Videosequenz extrahiert wurden. Anhand der relativen Verschiebung jedes einzelnen Features kann eine Bewegung der Kamera abgeleitet werden.



(a) (b)

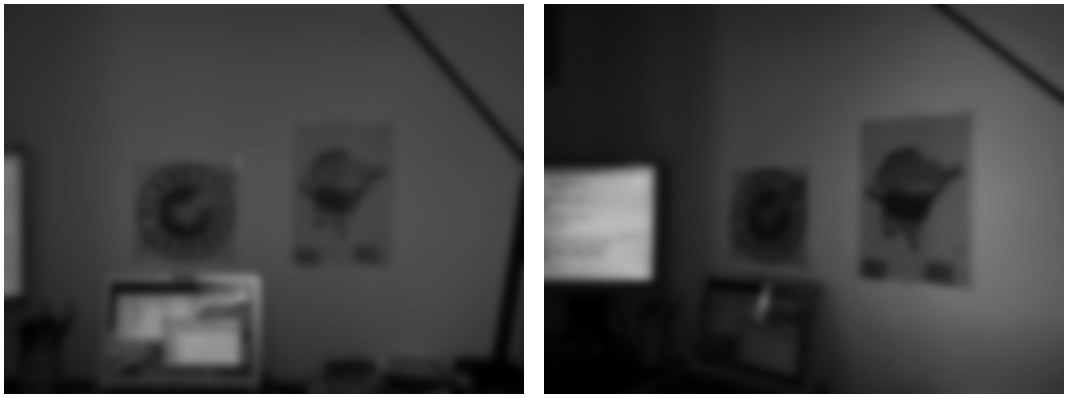
Abbildung 5.1: Feature-basierte Bewgegunsabschätzung anhand von zwei aufeinanderfolgenden Frames - (a) Bild vor der Transformation. Koorespondierende Features sind als grüne Punkte eingezeichnet. (b) Bild nach einer 45 Grad Rotation, koorespondierende Features sind als blaue Punkte eingezeichnet. Die alte Position und die Bewegung der korrespondierenden Features ist anhand der grünen Punkte und der roten Linien verdeutlicht.

Durch die Abarbeitung aller drei Schritte kann eine erfolgreiche Abschätzung der Eigenbewegung allein durch die relative Bewegung visueller Features innerhalb einer Videosequenz erfolgen.

5.2.1.2 Direkte Methoden

Direkte Methoden verzichten vollständig auf den Einsatz von Feature-Verfahren. Stattdessen verwenden diese Methoden Intensitätswerte eines Bildes, um Strukturen oder Bewegungen zu bestimmen. Irani et al. definieren *direkte Methoden* als Verfahren zur Schätzung von Strukturen, Oberflächen und Bewegungen, die durch das Bestimmen der Intensitätswerte jedes einzelnen Pixel anhand der direkten Betrachtung des gesamten Bildes erfolgt [172].

Direkte Methoden nutzen alle Informationen eines Bildes aus, auch wenn die Intensitätsunterschiede sehr gering sind. Feature-basierte Methoden haben im Gegensatz zu direkten Methoden den Nachteil, dass sie in Bildsequenzen, die durch wenig markante Texturen [173] oder durch Unschärfe [174] geprägt sind, wenig gemeinsame Features finden und dadurch eine Ungenauigkeit in der Bewegungsabschätzung entsteht. Ein weiterer Vorteil ergibt sich daraus, dass die Analyse und Extraktion der Features entfällt, da ausschließlich die Intensitätswerte eines Bildes direkt betrachtet werden.
Abbildung 5.2 zeigt beispielhaft, welche Eigenschaften des Bildes von den direkten Methoden verwendet werden. Anhand der Veränderung der Intensitätsunterschiede jedes einzelnen Pixels kann einerseits die relative Eigenbewegung abgeschätzt werden und anderseits das Bild grob segmentiert werden, sodass einzelne Objekte deutlich herausstechen. Aus einer längeren Videosequenz mehrerer aufeinanderfolgender Bilder können bereits grobe Tiefeninformationen über einzelne Segmente im Bild abgeleitet werden.



(a) (b)

Abbildung 5.2: Direkte Methode der Bewgegunsabschätzung anhand von zwei aufeinanderfolgenden Frames - (a) Graustufenbild vor der Transformation. (b) Graustufenbild nach einer 45 Grad Rotation. Die Rotation wird durch die Verschiebung der Intensitätswerte jedes einzelnen Pixel nach rechts bestimmt.

5.2.2 Beispiele der visuellen Odometrie

Die visuelle Odometrie und ihre Verfahren werden erfolgreich in verschiedenen Bereichen eingesetzt. So wird sie in der Raumfahrt zu Automatisierung von Fahrzeugen und Flugkörpern, in der Robotik zur autonomen Erkundung der Umwelt, in unbemannten Drohnen zur Positionierung, in Fahrer-Assistenz-Systemen zur Unterstützung des Fahrens und im Consumermarkt für kleine Soft- und Hardwarelösungen eingesetzt [171].

Diese unterschiedlichen Bereiche werden im Folgenden anhand von ausgewählten Beispielen näher vorgestellt.

Raumfahrt Die visuelle Odomtrie wird bereits sehr erfolgreich in der Weltraumforschung eingesetzt. Besonders für die Mond- und Mars-Missionen wird sie zur Eigenbewegungsabschätzung der Rover eingesetzt [115]. Die bekannteste Anwendung der visuellen Odometrie wurde von der NASA für die Mars-Mission entwickelt. Dabei werden seit dem Jahr 2004 Verfahren der visuellen Odometrie eingesetzt, um die Eigenbewegung der zwei eingesetzten *Mars Exploration Rover Spirit* und *Opportunity* zu bestimmen [175, 176]. Neben der Schätzung der Eigenbewegung auf dem Mars, werden die Verfahren ebenfalls für den automatisierten und reibungslosen Landeprozess von Flugkörpern eingesetzt [177]. Dabei wird anhand von binokularen Kamerasystemen die Oberfläche eines Planeten während des Landeprozesses analysiert, sodass eine dreidimensionale Oberflächenstruktur bestimmt werden kann. Dieses Verfahren ermöglicht es auf unbekanntem Terrain geeignete Landeflächen für ein großes Flugobjekt automatisiert und in Echtzeit ausfindig zu machen.

Robotik und unbemannte Drohnen Besonders im Bereich der Robotik und im Bereich der unbemannten Drohen sind die Verfahren der visuellen Odometrie essentiell wichtig für eine autonome und erfolgreiche Positionierung und Navigation. *Micro Air Vehicle* sind unbenannte Flugdrohnen, die sich durch ihre geringe Größe, Gewicht und Fluggeschwindigkeit auszeichnen. Diese Drohnen setzen eine oder mehrere kleine einfache Kameras ein, um daraus ihre Eigenbewegung und Position zu bestimmen. In der Arbeit von Forster et al. wird ein visuelles Odometrie-Verfahren vorgestellt, das die Trajektorie eines Quadrocopters mit einer monokularen Kamera und einer kleinen Recheneinheit exakt bestimmen kann [178]. Dabei findet die vollständige Auswertung der Trajektorie in Echtzeit statt. Auch im Bereich von autonomen Unterwasserfahrzeugen wird die visuelle Odometrie eingesetzt. Unter Wasser ist eine Positionierung mittels GPS viel zu ungenau. Daher eignen sich Kamerasysteme als kostengünstigere Alternative zur Positionierung. Die Verfahren nutzen zur Bestimmung der Eigenbewegung den markanten Meeresgrund, der für visuelle Feature-Verfahren hervoragend geeignet ist [179].

Fahrer-Assistenz-Systeme Die visuelle Odometrie ist für die Automobilindustrie ebenfalls von großer Bedeutung. Speziell für ihre Fahrer-Assistenz-Systeme und neuerdings auch für die Umsetzung autonom fahrender Fahrzeuge ist die Vielzahl der Einsatzmöglichkeiten von visuellen Verfahren immens gestiegen. Im Bereich der Fahrer-Assistenz-Systeme werden Verfahren der visuellen Bewegungsabschätzung verwendet, um Gefahren rechtzeitig zu erkennen und das Fahrzeug abzubremsen. Daimler hat mit seinem *6D-Vision System* ein Verfahren entwickelt, das neben der Eigenbewegung des Fahrzeuges andere bewegliche Objekte im Straßenverkehr erkennt und daraus mögliche kritische

Situationen identifiziert [180]. Dabei analysiert eine binokulare Kamera das Straßenbild auf bewegliche Features, die anschließend auf ihre Objektzugehörigkeit, Geschwindigkeit und Richtung klassifiziert werden. Besonders die niedrigen Preise von Kamerasensoren machen den Einsatz visueller Verfahren, speziell die visuelle Odometrie, für die Automobilindustrie so interessant.

Consumermarkt Auch im Consumermarkt finden sich vermehrt Anwendugen der visuellen Odometrie. Einfache Umsetzungen der Verfahren werden für kleine Softwarelösungen, insbesondere im mobilen Anwendungsbereich, eingesetzt. Ein Beispiel für die Umsetzung solcher Verfahren ist die *Dacuda Scanner Mouse* [181]. Diese ist sowohl eine optische Computer-Maus als auch ein Bild-Scanner, der eine Oberfläche durch einfaches darüber wischen erfasst und zu einem Gesamtbild zusammensetzt. Um die einzelnen Bildsegmente zusammenzufassen, wird ein Verfahren der Eigenbewegungsabschätzung der Maus eingesetzt. Die Technologie wurde von dem Unternehmen weiterentwickelt, sodass eine Portierung dieser Technologie auf mobile Endgeräte mit eingebauter Kamera möglich ist. Ein konkretes Anwendungsbeispiel ist eine mobile Anwendung, die aus Videosequenzen dreidimensionale Objekte rekonstruieren kann.

5.3 Odometrie, Koppelnavigation und Aktivitätserkennung

Neben der visuellen Odometrie gibt es ähnliche Disziplinen, die sich mit der relativen Positionsbestimmung und Eigenbewegungsabschätzung eines sich bewegenden Objektes beschäftigen. Dazu zählen die klassische Odometrie sowie die Koppelnavigation. Anders als die visuelle Odometrie werden statt der visuellen Kamerasensoren Bewegungssensoren oder Antriebssysteme mit unterschiedlichen Messmethoden eingesetzt. Eine erweiterte Variante stellt das *Simultaneous Localisation and Mapping* dar, welches neben der Bewegungsabschätzung gleichzeitig einen Lageplan der bereits gesichteten Umgebung rekonstruiert.
Neben der Bewegungsabschätzung und relativen Positionierung gibt es einen weiteren Bereich, der untersucht wird. Dazu zählt die Aktivitätserkennung, die sich klassischerweise aus den Bewegungsmustern eines sich bewegenden Objektes ebenfalls ableiten lässt.
Die Verfahren aus Abschnitt 5.4 und Abschnitt 5.5 sind zum Teil durch die Methoden der Koppelnavigation und der Aktivitätserkennung inspiriert. Im Folgenden werden diese Bereiche vorgestellt.

5.3.1 Odometrie

Die Odometrie, auch Hodometrie genannt, ist die Beobachtung der Räder eines Antriebssystems zur Abschätzung einer gefahrenen Wegstrecke [182]. Klas-

sischerweise wird diese Form der Odometrie an Fahrzeugen mit Radantrieb eingesetzt, wie z.B. ein Kraftfahrzeug. Im einfachsten Fall werden die Radumdrehungen zur Messung der Wegstrecke gezählt. Systeme, wie z.B. humanoide Roboter mit zwei Beinen, zählen dagegen die Anzahl an Schritten. Ein Gerät, das die Anzahl an Radumdrehungen oder Schritten zählt, wird auch *Odometer* genannt. Die Odometrie ist zwar relativ einfach umzusetzen, wird jedoch wegen ihrer hohen Fehlerrate, die z.B. durch Radschlupf entsteht, nur unterstützend zu anderen Messsystemen eingesetzt.
In der Fahrzeugtechnik wird die Odometrie neben der Radumdrehung durch den Lenkradwinkel, der Gierrate und gemessener Beschleunigungsdaten anderer Sensoren bestimmt. Zusätzlich setzen einige Verfahren zur weiteren Reduktion von Fehlern Kalman-Filter ein [182]. In der Robotik hat das Zählen der Schritte eines humanoiden Roboters eine kleinere Bedeutung für die Messung der Wegstrecke als für statistische Auswertungen des System. Hierfür werden eher visuelle Kamerasysteme oder Laser-Scanner zur Bestimmung der eigenen Lage und Abschätzung der gelaufenen Distanz verwendet [183, 184].

5.3.2 Koppelnavigation

Unter dem Begriff der Koppelnavigation, auch *Dead Reckoning* genannt, versteht man ein Verfahren zur eigenen Bewegungsabschätzung und Positionierung anhand von Beschleunigungs- und/oder Orientierungsdaten. Die Koppelnavigation wird in der Literatur abhängig von der Domäne, in der sie eingesetzt wird, anders verstanden und definiert. In der Luftfahrt wird unter dem Begriff der Koppelnavigation das Navigieren anhand von markanten Wegpunkten (Landmarken) während einer Flugstrecke von einem Ort zum anderen verstanden [185]. Dabei sind Landmarken aus dem Flugzeug ersichtliche bodengebundene Objekte, wie z.B. Straßen, Leuchttürme und Schienen. In der Schifffahrt wird die Koppelnavigation bereits seit dem 13. Jahrhundert zur Positionierung und Navigation verwendet [186]. Anhand eines Log, einer Sanduhr und einem Kompass wird die Position auf See ermittelt. Der Log wird zur Bestimmung der aktuellen Geschwindigkeit und der Kompass zur Orientierung verwendet.
Durch die fortschreitende Entwicklung und hohen Verbreitung mobiler Endgeräte mit integrierter Sensorik, wurde die Koppelnavigation besonders für den Forschungsbereich der Mobilen und Verteilten Systeme interessant. Viele mobile Endgeräte, wie z.B. Smartphones oder Tablets, besitzen heutzutage einen Beschleunigungssensor, ein Gyroskop und einen Kompass. Dadurch kann insbesondere durch den Beschleunigungssensor und den Kompass die Eigenbewegung und Position eines sich bewegenden Objektes bestimmt werden. Diese Form der Koppelnavigation wird größtenteils für die Positionierung innerhalb von Gebäuden oder Orten, an denen eine Lokalisierung mittels GPS zu ungenau ist, eingesetzt. Zur Bewegungsabschätzung und Positionierung von Personen wird das sogenannte *Pedestrian Dead Reckoning* verwendet. Gehende Personen haben ein zerstreutes und komplexeres Fortbewegungsmuster, das

von der Koppelnavigation anders interpretiert und übersetzt werden muss. Das *Pedestrian Dead Reckoning* versucht das Problem zu lösen, indem es die Beschleunigungssensordaten zur Erkennung von einzelnen Schritten verwendet [187, 160]. Jeder ausgeführte Schritt wird gezählt und dabei gleichzeitig die jeweilige Bewegungsrichtung anhand von Kompass und Gyroskop bestimmt. Um eine genaue Positionierung einer Person durchführen zu können, muss die Schrittlänge bekannt sein. Hierfür empfehlen Pratama et al. eine Formel, die aus der Größe einer Person und einem konstanten Wert die Schrittlänge berechnet [188]. Kim et al. verwenden für unterschiedliche Schrittgeschwindigkeiten unterschiedliche Schrittlängen [189]. Um Fehler zu vermeiden, die sich über die Zeit akkumulieren, werden zusätzlich Kalman- oder Partikel-Filter eingesetzt.

5.3.3 Simultaneous Localisation and Mapping

Das *Simultaneous Localisation and Mapping*, abgekürzt *SLAM*, ist eine besondere Art der Bewegungsabschätzung und Positionierung. Neben der eigentlichen Positionierung wird aus dem bereits erkundeten Pfad eine Karte oder ein Lageplan erzeugt [190]. Grundsätzlich werden *SLAM*-Verfahren in der Robotik eingesetzt und für Systeme im Innen- sowie Außenbereich, für Unterwasser und in der Luft verwendet. Zur Bestimmung der Eigenbewegung und Positionierung kommen unterschiedliche Verfahren zum Einsatz, die aus dem Bereich der Odometrie oder Koppelnavigation stammen können [191]. Die Besonderheit liegt darin, dass die Trajektorie während des Erstellens gleichzeitig zur Orientierung der eigenen Position verwendet wird. Diese wird im laufenden Prozess immer wieder korrigiert und aktualisiert. Ein wichtiges Konzept stellt das sogenannte *Loop Closure* dar. Orte die vorher schon einmal erkundet worden sind, können zur Korrektur der geschätzten Trajektorie und Position verwendet werden. Hierfür eignet sich die spezielle Form der visuellen *SLAM*-Verfahren *(VSLAM)*. Diese verwenden visuelle Odometrie-Verfahren für die Positionierung und können einen bereits gesehenen Ort anhand der visuellen Aufzeichnungen einfacher verifizieren [119]. Zusätzlich kann aus den aufgezeichneten Bildern und der rekonstruierten Trajektorie ein visueller Rundgang erstellt werden.

5.3.4 Aktivitätserkennung

Die Aktivitätserkennung hat besonders durch die Entwicklung der *Pedestrian Dead Reckoning* an Bedeutung gewonnen. Durch das markante Muster der Ausschläge, die durch den Beschleunigungssensor eines Smartphones bei einer Geh- oder Laufbewegung einer Person entstehen, können Schritte relativ einfach erkannt und gezählt werden. So können bereits durch die Schrittfrequenz Aktivitäten wie Rennen und Gehen voneinander unterschieden werden [192]. Brajdic et al. untersuchten hierfür das *Time Domain*, das *Frequency Domain* und das *Symbolic String Domain* Verfahren, um aus einer zeitlichen Abfolge

von Schritten unterschiedliche Aktivitäten zu erkennen [193]. Ihre Ergebnisse haben gezeigt, dass sich das *Time Domain* Verfahren durch eine Extremwertsuche über den zeitlichen Verlauf hinweg zur Unterscheidung von Aktivitäten wie Laufen, Gehen und Springen sehr gut eignet. Fritz-Walter et al. betrachten ausschließlich die markanten Ausschläge, die durch den Beschleunigungssensor entstehen, um eine Stillstands-Aktivität neben einer Geh- und Laufaktivität sauber unterscheiden zu können [194]. Hierfür klassifizieren sie die Beschleunigungssensordaten und verwenden für jede Aktivität unterschiedliche Schwellwerte.
Park et al. setzen zur Erkennung von Aktivitäten Verfahren des maschinellen Lernens ein [195]. Dafür werden die Sensordaten des Beschleunigungssensor für jede Aktivität klassifiziert. Sie haben gezeigt, dass die Qualität der Aktivitätserkennung von den Trainingsdatensätzen sehr stark abhängt. Die Klassifikation verschlechtert sich, sobald die Sensordaten zur Auswertung von einem anderen System als vom dem konditionierten System stammen. Zudem ist für eine sehr gute Erkennung der Aktivitäten ein individueller Trainingsdatensatz pro Person notwendig.
Einen interessanten Ansatz einer visuellen Aktivitätserkennung verfolgen Gutierrez-Gomez et al. in ihrer Arbeit [196]. Sie verwenden für ihr Verfahren eine omnidirektionale Helmkamera, um anhand von Oszillationen der Videosequenz Schritte und Aktivitäten zu erkennen. In Kombination mit einem *SLAM*-Verfahren können sie robust eine gegangene Trajektorie anhand der Helmkamera rekonstruieren.

5.4 Verarbeitung von visuellen Merkmalen zur Bestimmung der Eigenbewegung und Position

In diesem Abschnitt wird das neu entwickelte Verfahren der visuellen Odometrie, speziell für Fußgänger, vorgestellt. Dafür werden visuelle Merkmale zur Bestimmung der Eigenbewegung verwendet, um eine genaue Positionsbestimmung einer Person durchzuführen. Als visuelle Merkmale werden SURF-Features eines Bildes verwendet, um auffällige und markante Bewegungsmuster zu erkennen. Besonders die Veränderungen der Attribute eines Features innerhalb einer Bildreihe, wie z.B. bei einer Videosequenz, können solche Bewegungsmuster abbilden. Anders als bei den klassischen Verfahren der visuellen Odometrie wird ein visueller Schrittzähler und ein visueller Kompass eingesetzt. Dabei unterscheidet sich diese Methode dadurch, dass die Features nicht zur Rekonstruktion von 3D-Koordinaten verwendet werden, um daraus eine Bewegung abzuleiten und zu messen, sondern zur Bestimmung von Schritten und Richtungen. Das heißt, eine Trajektorie wird durch die gegangenen Schritte und die Bewegungsrichtung bestimmt. Als Eingabe verwenden der Schrittzähler und der Kompass eine vorverarbeitete Videosequenz. Die Video-

sequenz wird mit Hilfe einer Kamera, die aus der Ego-Perspektive einer Person aufzeichnet, erstellt. Abbildung 5.3 stellt einen Prozessablauf des visuellen Odometrie-Verfahrens dar.

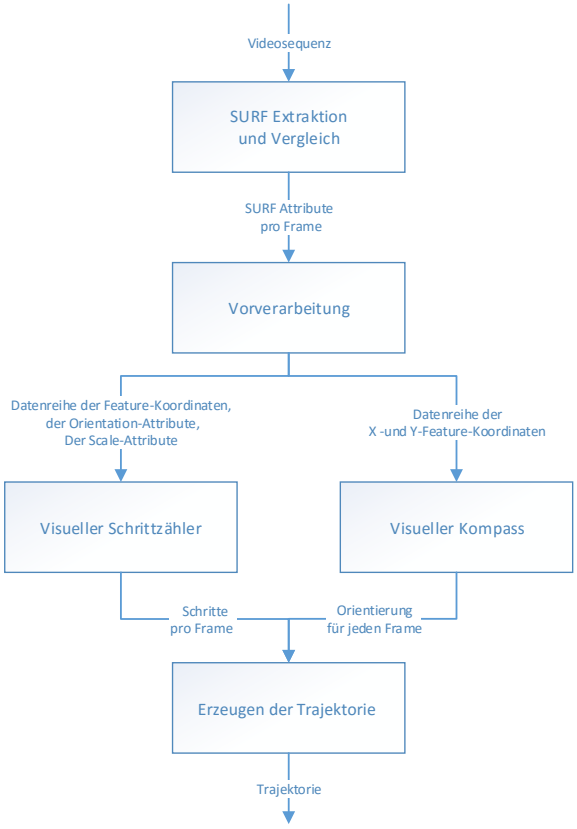


Abbildung 5.3: Teilprozese des visuellen Odometrie-Verfahrens [18].

Im ersten Schritt werden SURF-Features aus einer Videosequenz extrahiert und miteinander verglichen. Attribute der SURF-Feature-Paare werden an den Folgeprozess weitergegeben. Diese Attribute werden vorverarbeitet und anschließend an den visuellen Schrittzähler und den visuellen Kompass weitergegeben. Die gezählten Schritte sowie die Bewegungsrichtung zu jedem Schritt werden an den letzten Teilprozess weitergeben, um eine Trajektorie zu erzeugen.
Im Folgenden werden die wichtigsten Prozesse im Detail vorgestellt und diskutiert. Es wird der Prozess zur Extraktion der SURF-Features und deren Vergleich erläutert. Die einzelnen Attribute und ihre Eigenschaften werden ebenfalls beschrieben. Anschließend werden unterschiedliche Verfahren zur Vorverarbeitung der SURF-Attribute vorgestellt. Dann werden drei Varianten einer Schritt-Erkennung sowie der visuelle Kompass beschrieben. Anschließend wird der Prozess zur Bildung einer Trajektorie erläutert. Zuletzt wird eine ausführliche Evaluierung des Schrittzähler-Verfahrens sowie des visuellen Kompass

gegeben.

5.4.1 SURF-Extraktion und Vergleich

Der erste Teilprozess erhält als Eingabe eine Videosequenz, die von einer sogenannten *Body-Mount Kamera* aufgezeichnet wird. Die *Body-Mount Kamera* wird auf Brusthöhe getragen und am Oberkörper einer Person befestigt. Jedes Bild (Frame) der Videosequenz wird mit dem SURF-Algorithmus auf Features analysiert (vgl. Abs. 2.3). Zur Feature-Extraktion der Bilder wird der SURF-Algorithmus eingesetzt, da er einerseits robust und in effizienter Zeit ausführbar ist und anderseits das Feature-Verfahren neben dem Deskriptor zur Beschreibung des Features weitere wichtige Attribute mit sich bringt. Diese Attribute sind für die spätere Analyse und Auswertung von Bewegungs- und Schrittmustern sehr wichtig.
Die Videosequenz wird iterativ ausgewertet, sodass Features eines Frames mit den Features des darauffolgenden Frames verglichen werden. Für jedes Frame-Paar erhält man somit eine Menge gemeinsamer Feature-Paare, die im folgenden als *Match* bezeichnet werden. Für die nächsten Teilprozesse der Vorverarbeitung werden die SURF-Attribute *Orientation*, *Scale* und die Feature-Koordinaten (X, Y) verwendet (vgl. Abs. 2.3.2). Die *Orientation* beschreibt die dominanteste Ausrichtung des Helligkeitsverlaufes der zur betrachtenden Umgebung eines Features. Die *Scale* beschreibt die Auflösung eines Features. Die Attribute (X, Y) beschreiben die Bildkoordinate des Feature-Zentrums.
Für eine zusätzliche Robustheit des Vergleichsverfahrens werden im Folgenden zwei optionale Erweiterungen vorgestellt und diskutiert. Diese reduzieren die Wahrscheinlichkeit, dass zwei ungleiche Features als ähnlich oder gleich betrachtet werden.

Fenster-Vergleich Ein zu vergleichendes Feature, das in allen darauffolgenden Frames ein *Match* findet, wird mit hoher Wahrscheinlichkeit kein falschpositives Ergebnis sein. Daher bietet es sich an, ein Feature über einen längeren Frame-Verlauf zu vergleichen, um die Robustheit des Vergleichsverfahren zu erhöhen. Dafür findet innerhalb eines Fensters, bestehend aus n Frames, ein Feature-Vergleich mit jedem Frame statt. Ist n größer als 1, findet ein Feature-Vergleich mit den nächsten $n-1$ Frames statt. Ein Feature gilt nur als gleich, solange es in den darauffolgenden $n-1$ Frames zu finden ist.
Die Anzahl an Vergleichen steigt, desto Größer das n ist. Das führt dazu, dass der Vergleichsprozess allgemein zeitaufwendiger wird. Entscheidend ist die richtige Wahl der Fenstergröße n. Ein zu großes n setzt voraus, dass ein Feature über einen längeren Frame-Verlauf zu sehen sein muss. Dies ist nicht immer gegeben, besonders wenn ein zu beobachtendes Feature aus dem Betrachtungsfeld des Bildes verschwindet. Dadurch kann die Anzahl an gemeinsamer Features drastisch sinken, sodass eine anschließende Bestimmung der Eigenbewegung nicht mehr möglich ist.

Auslassen von Frames Dieser Ansatz versucht, statt ein Feature über einen längeren Frame-Verlauf zu verfolgen, einzelne m Frames auszulassen, um den selben Effekt zu erzielen. Klassischerweise haben Videosequenzen eine Frame-Rate von 25 Bilder pro Sekunde. Eine markante Veränderung der SURF-Attribute fällt jedoch innerhalb einer Sekunde, also innerhalb von 25 Bildern, sehr gering aus. Das heißt, dass die Bildfolgen sehr ähnlich zueinander und daher redundant für die Auswertung sind. Es empfiehlt sich daher allgemein einige Frames dazwischen auszulassen. Anders als bei der Variante des Fenster-Vergleiches, findet dadurch immer nur ein Feature-Vergleich statt, da m Frames zwischen den zu vergleichenden Frames ausgelassen werden. Dies hat den Vorteil, dass weniger Frames mit dem SURF-Algorithmus verarbeitet werden müssen und in der selben Zeit weniger Vergleiche statt finden.

5.4.2 Vorverarbeitung der SURF-Attribute

Die Vorverarbeitung der Attribute trägt dazu bei, dass die Auswertung der Bewegungsmuster einer Videosequenz einfacher zu charakterisieren sind. Besonders der visuelle Schrittzähler benötigt hierfür vorverarbeitete Datensätze. Es werden die Attribute von zwei gleichen Features *(Match)* aus jeweils einem Frame-Paar betrachtet. Für jeden *Match* werden die Differenzen ihrer Attribute berechnet. Dies erfolgt je nach Attribut unterschiedlich.
Im Folgenden werden die Vorverarbeitugnsschritte für die einzelnen Attribute vorgestellt. Dabei wird auf ihre besonderen Eigenschaften, speziell für die visuelle Odometrie, eingegangen.

5.4.2.1 Feature-Koordinate

In [15] wurde bereits gezeigt, dass die Bildkoordinaten von gleichen Features zur Auswertung einer Bewegung innerhalb einer Videosequenz, die aus der Ego-Perspektive einer Person aufgezeichnet wird, verwendet werden können. Dabei lässt sich ein sich wiederholendes charakteristisches Muster nachweisen, das während des Gehens einer Person auftritt. Neben diesem charakteristischen Muster lassen sich auch einfache Bewegungen der Kamera nachweisen. Eine allgemeine Bewegung der Features entlang der horizontalen Achse des Bildes lässt z.B. auf eine Drehung schließen. Bewegt sich die Mehrheit der Features aus dem Bildzentrum heraus, lässt sich daraus eine Vorwärtsbewegung ableiten und umgekehrt eine Rückwärtsbewegung.
Insgesamt können die Feature-Koordinaten sowohl für den visuellen Schrittzähler als auch für den visuellen Kompass verwendet werden. Hierfür werden die Bildkoordinaten aller gemeinsamen Features *(Match)* eines Frame-Paares jeweils für Schrittzähler und Kompass wie folgt verarbeitet.

Schrittzähler Es wird für alle *Matches* eines Frame-Paares die Euklidische Norm aus ihren Vektorkoordinaten, also ihren Bildkoordinaten, berechnet. Dadurch erhält man für jeden Match seine relative Bewegung innerhalb dieses

Frame-Paares. Die relative Bewegung eines jeden *Matches* wird anschließend positiv oder negativ gewichtet abhängig von seiner vertikalen Verschiebung auf dem Bild. Die Summe aus der gewichteten relativen Bewegung jedes *Matches* wird für das Frame-Paar berechnet und durch die Anzahl aller gemeinsamen Matches geteilt. Somit erhält man für jedes Frame-Paar einen Wert, der eine Aussage über die relative Bewegung zulässt.

Kompass Die Bildkoordinaten eines Features eigenen sich ebenfalls für die Auswertung der Bewegungsrichtung. Hierfür werden die X und Y Koordinaten eines *Matches* einzeln betrachtet und vorverarbeitet. Dabei wird für jedes Frame-Paar der Mittelwert der horizontalen und der vertikalen Bewegung aller *Matches* anhand der relativen Bewegung ΔX und ΔY einzeln berechnet. Dadurch ergibt sich für jedes Frame-Paar F ein Mittelwert ΔX_F und ΔY_F.

$$\Delta X_F = \frac{\sum_{M \in Matches} \Delta X_M}{\#Matches} \quad , \quad \Delta Y_F = \frac{\sum_{M \in Matches} \Delta Y_M}{\#Matches}$$

5.4.2.2 Orientation

Die *Orientation* beschreibt die dominanteste Ausrichtung des Helligkeitsverlaufes der zur betrachtenden Umgebung eines Features. Anhand der Differenz der *Orientation* eines Matches lässt sich ebenfalls eine Bewegung ableiten. Dieses Verhalten lässt sich anhand des Beispieles eines Weinglases einfach erklären. Ein Weinglas hat einen Glanzpunkt, der durch eine oder mehrere Lichtquellen erzeugt wird. Geht man nun auf das Glas zu, scheint es als ob der Glanzpunkt sich durch die Eigenbewegung hin und her bewegt. Dies wird durch eine leichte Pendelbewegung beim Gehen verursacht, die durch den abwechselnden Einsatz des linken und rechten Fußes entsteht. Dieses Verhalten lässt sich bei der *Orientation* eines SURF-Features ebenfalls nachweisen. Aus diesem Grund wird diese Eigenschaft speziell für den Einsatz des visuellen Schrittzählers verwendet.
Da die *Orientation* in einem Winkelmaß angegeben wird, kann ihre Differenz nicht durch eine einfache Subtraktion der *Orientation*-Attribute erfolgen ($Orientation_{diff}$). Eine Drehung von 359 Grad zu 1 Grad soll nicht als 358 Grad Drehung betrachtet werden. Außerdem kann eine Rotation von über 180 Grad zwischen einem Frame-Paar ausgeschlossen werden. Daher wird die eigentliche Differenz der *Orientation* eines *Match* M mit der $\Delta Orientation$ bestimmt und wie folgt berechnet:

$$\Delta Orientation_M = \begin{cases} 0 - (Orientation_{diff} + \pi) & \text{falls } Orientation_{diff} < -\pi \\ 0 - (Orientation_{diff} - \pi) & \text{falls } Orientation_{diff} > \pi \\ Orientation_{diff} & \text{sonst} \end{cases}$$

Aus der Summe der $\Delta Orientation_{M_i}$ aller *Matches* M_i, die durch die Anzahl

aller *Matches* geteilt wird, erhält man die durchschnittliche $\Delta Orientation_F$ für ein Frame-Paar F.

$$\Delta Orientation_F = \frac{\sum_{M \in Matches} \Delta Orientation_M}{\#Matches}$$

Die $\Delta Orientation_F$ ist jedoch anfällig für Ausreißer. Sobald ein falsch-positiver *Match* eine zu hohe $\Delta Orientation$ aufweist, kann dieser das Gesamtergebnis der $\Delta Orientation_F$ verfälschen. Um diesen Effekt auszugleichen, werden die einzelnen Werte gewichtet und vereinheitlicht. Hierfür wird der *Orientation-Score* eingeführt, der jeden Match mit einem $+1$ oder einem -1 bewertet. Somit wird der $OrientationScore_M$ für ein *Match* M wie folgt bestimmt:

$$OrientationScore_M = \begin{cases} +1 & \text{falls } \Delta Orientation_M > 0 \\ -1 & \text{falls } \Delta Orientation_M < 0 \\ 0 & \text{sonst} \end{cases}$$

Die Summe aus dem *OrientationScore* aller *Matches* eines Frame-Paares geteilt durch die Anzahl aller *Matches* ergibt den $OrientationScore_F$. Dieser liegt immer innerhalb eines Intervalls von $[-1, +1]$. Ein Wert von -1 oder $+1$ weist auf eine Rotation aller *Matches* in dieselbe Richtung hin, wohingegen ein Wert von 0 bedeutet, dass gleich viele *Matches* in die entgegengesetzten Richtungen rotieren.

$$\Delta OrientationScore_F = \frac{\sum_{M \in Matches} \Delta OrientationScore_M}{\#Matches}$$

Als Resultat erhält man letztendlich eine Datenreihe, die durch den $OrientationScore_F$ für jedes Frame-Paar F ein Bewegungsmuster innerhalb einer Videosequenz beschreibt.

5.4.2.3 Scale

Die *Scale* gibt an mit welcher Auflösung ein Feature in einem Bild gefunden wird. Dadurch kann die Größe der Umgebung eines Features innerhalb eines Bildes bestimmt werden. Vergleicht man nun die *Scale*-Werte eines *Match*, lässt sich daraus ableiten, ob man sich einem Feature genähert oder sich von ihm entfernt hat. Daher ist auch dieses Attribut für die Bestimmung der Bewegung einer Person gut geeignet. Es wird für jeden *Match* eines Frame-Paares die Differenz aus den *Scale*-Werten berechnet und das arithmetische Mittel daraus bestimmt, welches das $\Delta Scale_F$ für jedes Frame-Paar F bestimmt.

$$\Delta Scale_F = \frac{\sum_{M \in Matches} \Delta Scale_M}{\#Matches}$$

5.4.3 Schritterkennung anhand von Feature-Attributen

Die Datenreihen aus dem Vorverarbeitungschritt werden anschließend für die Schritterkennung verwendet. Es werden speziell die Datenreihen bestehend aus den Feature-Koordinaten und aus der *Orientation* eingesetzt. Es hat sich gezeigt, dass sich diese beiden Datenreihen für eine Auswertung von Schritten am besten eignen.
Daher wurden zwei Ansätze eines Schritterkennungsverfahren entwickelt, die sich grundsätzlich durch den Einsatz der ausgewählten SURF-Feature-Attribute und der Wahl der Methoden unterscheiden. Das erste Verfahren verwendet die relative Bewegung der Feature-Koordinaten und leitet daraus einzelne Schritte ab. Das zweite Verfahren verwendet die relative Veränderung der *Orientation* (speziell die $OrientationScore_F$), um einzelne Schritte zu erkennen. Das erste Verfahren basierend auf den Feature-Koordinaten wird nur kurz vorgestellt, da es im Rahmen dieser Arbeit nur in Grundzügen konzeptioniert wurde. Der Fokus liegt auf dem zweiten Verfahren basierend auf den *Orientation*-Attributen.

5.4.3.1 Relative Bewegung der Feature-Koordinaten

Das Ausführen einer Geh-Bewegung führt dazu, dass sich Feature-Koordinaten innerhalb einer Videosequenz durch das Auftreten und Absetzten der Füße entlang der vertikalen und horizontalen Achse bewegen.
Die Datenreihe der Feature-Koordinaten aus dem Vorverarbeitungsschritt wird zur Auswertung der Schritte verwendet (vgl. Abs. 5.4.2.1). Jeder Wert dieser Datenreihe bildet die durchschnittliche Bewegung eines Frame-Paares ab. Die Datenreihe wird zunächst um Rauscheffekte geglättet, die durch die Ungenauigkeit des Verfahrens oder durch falsch-positive *Matches* entstehen, zu reduzieren. Hierfür wird ein Polynomfilter, eine Variante des *Savitzky-Golay-Filter*, verwendet. Dieser glättet jeden Wert einer Datenreihe anhand einer vorgegeben Formel, die abhängig von einer gewählten Fenstergröße und einer Polynomordnung ist. Dieser Filter hat den Vorteil, dass die Höhen und Breiten von Peaks der ursprünglichen Datenreihe besser erhalten bleiben, während andere Glättungs-Verfahren diese charakteristischen Werte herausnehmen. Abbildung 5.4 zeigt beispielhaft eine geglättete Datenreihe, die das Schrittmuster einer gehenden Person abbildet. Als Grundlage dient eine Videosequenz mit einer Auflösung von 640 x 360 Pixel und einer Frame-Rate von 25 FPS, die aus der Ego-Perspektive einer Person aufzeichnet. Innerhalb dieser Videosequenz geht die Person insgesamt 30 Schritte.
Die ausgeprägten und sich wiederholenden Peaks lassen vermuten, dass diese durch die Schritte verursacht werden. Eine weitere Untersuchung, die weitere Videosequenzen mit solchen Schrittmustern analysiert hat, hat ergeben, dass diese Peaks tatsächlich durch Schritte ausgelöst werden. Ein Schritt besteht demzufolge aus zwei entgegengesetzten Extremwerten. Daher werden die Schritte mit Hilfe eine Extremwertsuche innerhalb eines gleitenden Fensters

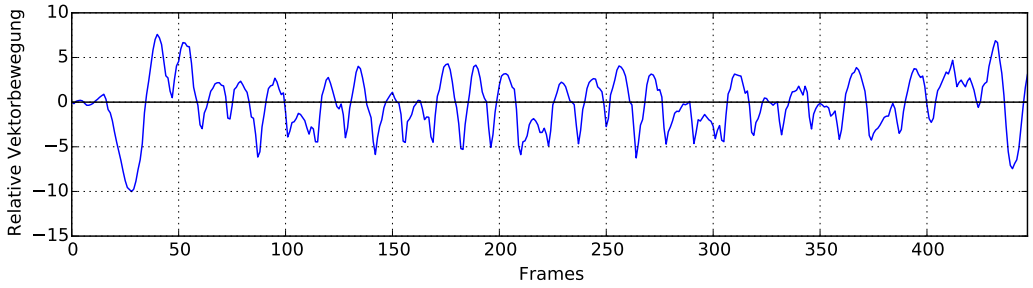


Abbildung 5.4: Schritterkennung anhand der relativen Vektorbewgung von SURF Feature-Koordinaten. Die Datenreihe bildet 30 getätigte Schritten ab.

mit fester Größe identifiziert. Da andere Bewegungen zu ähnlichen Peaks führen können, wird eine obere und untere Schranke für alle Extremwerte eingeführt, um ausschließlich Schritte zu berücksichtigen. Als weitere Einschränkung gilt, dass immer ein Minimum auf ein Maximum oder umgekehrt folgen muss. Das hat den Grund, dass bei jedem Schritt eine Auf- und Abbewegung entsteht, die eine solche Reihenfolge an Extremwerten erzeugt.
Die Anzahl an Extremwerten, die über bzw. unter einer dieser Schranken liegen, ergibt die Summe aller gegangenen Schritten, wobei die Anzahl nochmal halbiert werden muss, da ein Schritt immer aus einem Maximum und Minimum besteht.
Das Verfahren hat sich jedoch, trotz dieser anfänglich guten Ergebnisse, anfällig gegenüber einigen Störfaktoren erwiesen. Abhängig von der Art und Weise wie eine Person einen Schritt ausführt, werden die Höhen und Breiten der einzelnen Peaks beeinflusst. Das macht es schwierig geeignete Schranken für die Extremwerte zu definieren. Das führt dazu, dass entweder zu wenig oder zu viele Schritte erkannt werden. Ein weiterer Störfaktor ergibt sich durch Objekte im Bild, die sich unabhängig von der gehenden Personen bewegen. Beweglichen Objekte tragen zu einem Verfälschen des Schrittmusters bei, da sie selber Features erzeugen und sich im Gegensatz zu statischen Features selbst bei einer Stillstand-Phase bewegen.

5.4.3.2 Relative Veränderung der Orientation

Anders als die Feature-Koordinaten verhält sich die *Orientation* eines Features nicht so störanfällig bei Eigenbewegungen sobald andere bewegliche Objekte im Bild auftauchen. Die *Orientation* beschreibt die dominanteste Ausrichtung eines Helligkeitsverlaufes eines Features und wird in einem Winkelmaß angegeben. Objekte, die sich ausschließlich auf der horizontalen oder vertikalen Achse durch eine Translation bewegen, führen zu keinen großen Veränderungen der *Orientation*. Dagegen führen pendelnde Bewegungen, wie sie z.B. durch das Gehen oder Laufen erzeugt werden, zu Rotationen der *Orientation*. Das heißt,

dass die eigene Bewegung mehr Einfluss auf die Veränderung der *Orientation* hat als andere bewegliche Objekte, wie z.B. vorbeifahrende Fahrzeuge oder passierende Personen.

Das hier vorgestellte Verfahren zur Schritterkennung basiert demnach auf der Betrachtung der Veränderung der *Orientation*-Werte. Schritte werden anhand von markanten Ausschlägen, die durch die Veränderung der *Orientation* entstehen, erkannt.

Als Grundlage dient die Datenreihe bestehend aus den Werten der $OrientationScore_F$ des Vorverarbeitungsschrittes. Diese wird zunächst geglättet, um Rauscheffekte zu reduzieren. Dafür kann ein *Savitzky-Golay-Filter* oder ein *Butterworth Tiefpassfilter* verwendet werden. Vorherige Versuche haben gezeigt, dass sich beide Filter bei einer Glättung der Datenreihe sehr ähnlich verhalten und gleich gute Ergebnisse liefern. Dies ist dadurch bedingt, dass sich der *Savitzky-Golay-Filter* durch seine variablen Glättungsfaktoren und Fenstergrößen bei entsprechender Konfiguration ähnlich einem Tiefpassfilter verhält.

Abbildung 5.6 zeigt beispielhaft eine geglättete Datenreihe, die das Schrittmuster einer gehenden Person abbildet. Als Grundlage dient die selbe Videosequenz, die auch in Abschnitt 5.4.3.1 verwendet wird.

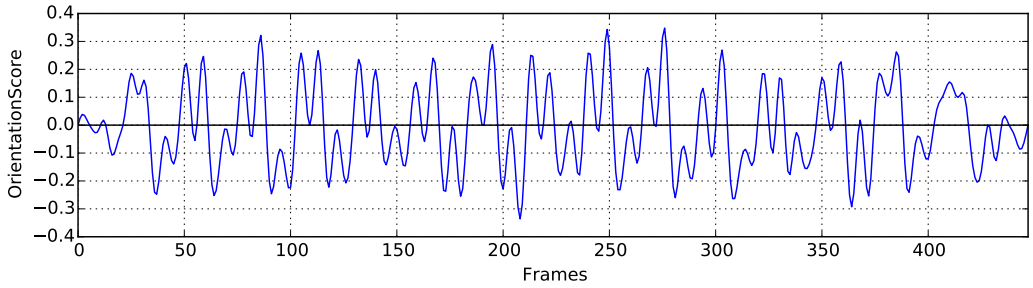


Abbildung 5.5: Schritterkennung anhand des $OrientationScore_F$. Die Datenreihe bildet 30 getätigte Schritte ab.

Man erkennt ein eindeutiges Muster, das sich ungefähr 15 Mal wiederholt und damit genau der Hälfte der ausgeführten 30 Schritte entspricht. Dieses Muster wird durch eine leichte Pendelbewegung beim Gehen verursacht, die durch den abwechselnden Einsatz des linken und rechten Fußes entstehen. Je nach Ausrichtung der Kamera entspricht ein Maximum einem Schritt mit dem rechten Fuß und ein Minimum einem Schritt mit dem linken Fuß.

Die eigentliche Schritterkennung erfolgt ebenfalls durch eine Extremwertsuche innerhalb eines gleitenden Fensters, wobei jeder erkannte Extremwert eindeutig einem Fuß zugeordnet wird. Da jedoch nicht jeder Extremwert einem tatsächlichen Schritt entspricht, werden einige optionale Korrekturverfahren hinzugefügt, welche im Folgenden erläutert werden.

Plausibilitätsregel Die Plausibilitätsregel stellt die einfachste und sinnvollste Prüfung über alle gefunden Extremwerte dar. Sie prüft ob ein Maximum immer größer Null und ein Minimum kleiner Null ist. Dies kann vorausgesetzt werden, da ein Schrittwechsel vom linken zum rechten Fuß oder umgekehrt immer einen Übergang durch den Nullpunkt erfordert. Trifft dieser Sachverhalt nicht zu, wird der Extremwert als potenzieller Schritt ignoriert. Dieses Korrekturverfahren sollte obligatorisch eingesetzt werden.

Statische Schranke Eine definierte obere und untere Schranke verhindert, dass durch andere Bewegungen erzeugte Extremwerte als Schritt erkannt werden. Hierfür wird ein geeigneter Wert benötigt, der sich für unterschiedliche Arten und Weisen, wie eine Person geht, verwenden lässt. In Abbildung 5.6 sind die obere und untere Schranke im Einsatz. Es werden nur Extremwerte außerhalb beider Schranken berücksichtigt.

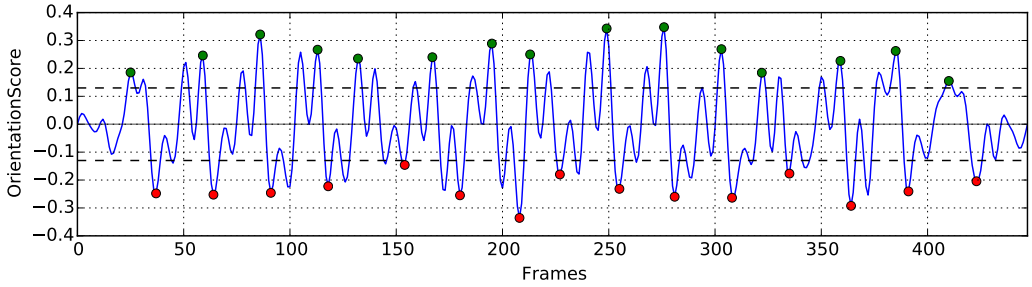


Abbildung 5.6: Schritterkennung unter Einsatz einer oberen und einer unteren Schranke (gestrichelte Linie). Grüne Punkte stellen ein Maximum und rote Punkte ein Minimum dar.

Es werden insgesamt 30 Extremwerte gefunden, die die Einschränkung der statischen Schranke erfüllen. In diesem Fall werden die 30 tatsächlich getätigten Schritte durch die Extremwertsuche mit Hilfe der statischen Schranke identifiziert. Dieses Korrekturverfahren sollte obligatorisch eingesetzt werden.

Dynamische Schranke Die dynamische Schranke stellt eine Erweiterung der statischen Schranke dar. Unterschiedliche Personen weisen unterschiedliche Schrittmuster auf, die sich ebenfalls auf die Höhen und Breiten der Peaks auswirken. Zwar wird bereits durch den $OrientationScore_F$ dieser Sachverhalt reduziert, jedoch kann es immer noch zu solchen Schwankungen kommen. Daher wird eine obere und untere dynamische Schranke zusätzlich zu den statischen Schranken eingeführt. Die dynamische Schranke (obere und untere) wird für jeden Extremwert, der die Bedingung einer der statischen Schranken erfüllt, neu berechnet. Dafür werden die letzten n Schritt-Paare betrachtet, wobei für jedes Schritt-Paar (z.B. rechter Fuß folgt auf linken Fuß) der $OrientationScore_F$ herangezogen wird. Für jedes Schritt-Paar wird der Absolutbetrag aus

der Differenz des $OrientationScore_F$ berechnet. Der Median der Absolutbeträge aller betrachten Schritt-Paare multipliziert mit einer empirisch ermittelten Konstante K ergeben die dynamische Schranke.

$$median(\sum_{i=m-n}^{m} \mid \Delta SchrittPaar_i \mid) * K$$

Dieses Korrekturverfahren kann als optional eingesetzt werden. Abbildung 5.7 zeigt den Einsatz der dynamischen Schranke für eine Videosequenz, in der erst Gelaufen, dann kurz Gestanden und anschließend Gegangen wird.

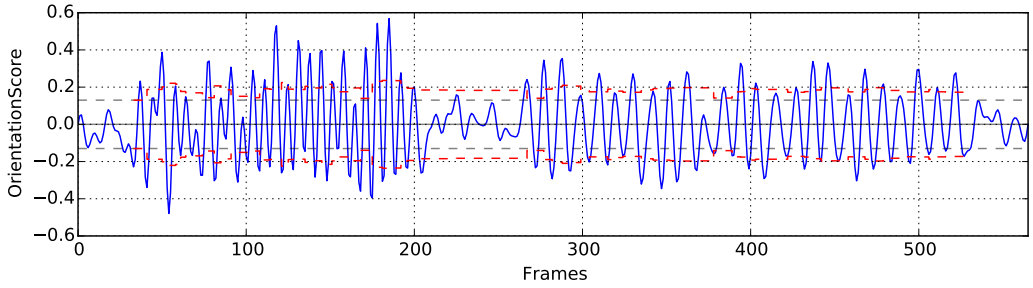
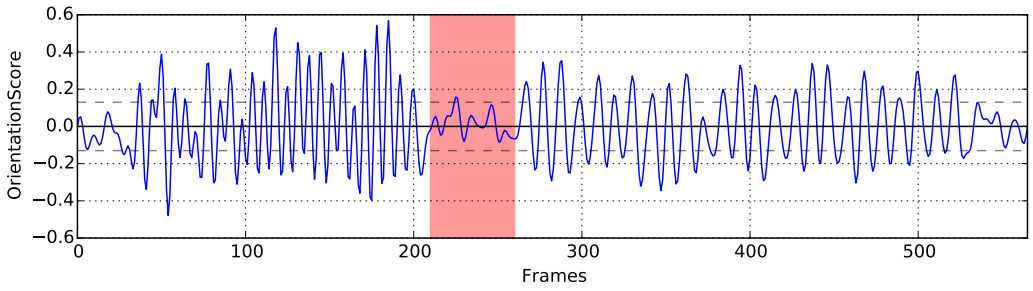


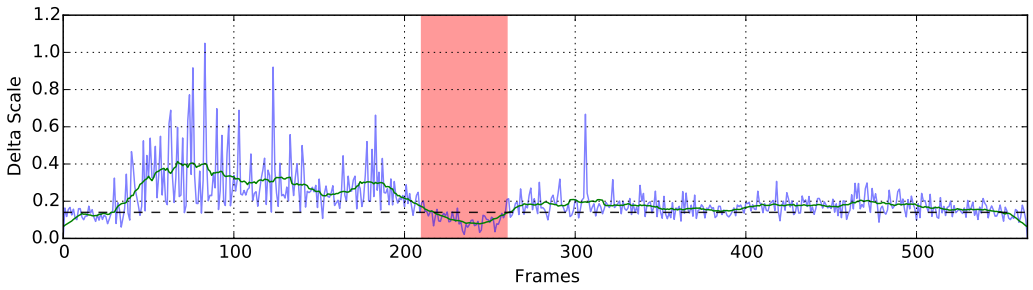
Abbildung 5.7: Anwendung der dynamischen Schranke (rote gestrichelte Linie). Zwichen Frame 210 und 260 findet eine Stillstands-Phase statt, die von der statischen Schranke (schwarze gestrichelte Linie) mindestens als ein Schritt erkannt wird.

Stillstandserkennung Dieses Korrekturverfahren setzt ebenfalls auf eine zusätzliche Schranke. Diese wird jedoch nur eingesetzt, sobald der beschriebene *Scale*-Wert aus Abschnitt 5.4.2.3 einen bestimmten Grenzwert unterschreitet. Die Differenz aus der *Scale* eines *Match* gibt an ob man sich zu einem Feature genähert oder von ihm entfernt hat ($\Delta Scale_F$). Diese Eigenschaft kann dafür verwendet werden, um Stillstands-Phasen einer Person festzustellen. Zur Erkennung von Stillstands-Phasen wird die Datenreihe aus dem Vorverarbeitungsschritt aus Abschnitt 5.4.2.3 verwendet. Diese Datenreihe wird geglättet, indem hierfür ein gleitender Mittelwert mit einer festgelegten Fenstergröße verwendet wird. Liegt $\Delta Scale_F$ unterhalb eines empirisch evaluierten Grenzwertes, wird eine Stillstands-Phase vermutet und ein potenzieller Schritt zu diesem Zeitpunkt mit der zusätzlichen Schranke geprüft. Abbildung 5.8a zeigt die Schritterkennung anhand des $OrientationScore_F$. Die Videosequenz besteht aus einer Lauf-, Steh- und einer abschließenden Geh-Phase. Dagegen bildet Abbildung 5.8b die $\Delta Scale_F$ für jedes Frame-Paar F aus der selben Videosequenz ab. Sowohl die geglättete (grüner Graph) als auch die nicht geglättete Datenreihe (blauer Graph) werden darin abgebildet. Der rote Bereich stellt die Stillstands-Phase in beiden Abbildungen dar. Die geglättete $\Delta Scale_F$

liegt innerhalb der Stillstands-Phase unterhalb des Grenzwertes. Auch dieses Korrekturverfahren kann optional eingesetzt werden.



(a)



(b)

Abbildung 5.8: Schritterkennung anhand der $OrientationScore_F$ und der $\Delta Scale_F$. Zwichen Frame 210 und 260 findet eine Stillstands-Phase statt (roter Bereich): (a) geglättete Datenreihe mit $OrientationScore_F$ - (b) Datenreihe mit $\Delta Scale_F$. Der Grenzwert zwischen Stillstands-Phase und Aktivität ist als gestrichelte Linie gekennzeichnet.

5.4.4 Visueller Kompass

Im Folgenden wird eine Variante eines visuellen Kompasses vorgestellt, der die aktuelle Bewegungsrichtung einer Person feststellt und deren Veränderung über die Zeit beobachtet. Dieser Kompass bestimmt auf Basis der Feature-Koordinaten des Vorverarbeitungsschrittes aus Abschnitt 5.4.2.1 die aktuelle Bewegungsrichtung einer Person. Es wird speziell die horizontale Bewegungsänderung der Feature-Koordinaten verwendet. Der Kompass basiert auf der Annahme, dass das Koordinatensystem ausgehend von der *Body-Mount Kamera*, die am Oberkörper einer Person verankert ist, ausgerichtet wird.
Sobald eine Person geradeaus geht, bleiben die Kamera und damit der Oberkörper immer nach vorne ausgerichtet. Geht die selbe Person eine Kurve, ändert die Kamera die Blickrichtung ebenfalls in die selbe Richtung wie der Oberkörper. Sowohl das Geradeaus-Gehen als auch der Kurvengang sind Bewegungen,

die parallel zum Boden verlaufen. Das heißt, dass die Hauptbewegung der Feature-Koordinaten zwischen zwei Frames entlang der x-Achse verläuft. Geht eine Person geradeaus, entstehen nur minimale Veränderungen der Feature-Koordinaten auf der x-Achse. Dagegen führt eine Drehung der Person während einer Stillstands- oder Geh-Phase zu einer größeren und länger andauernden Veränderung der Feature-Koordinaten entlang der x-Achse.
Dies veranlasst grundsätzlich die horizontale Bewegung der Feature-Koordinaten ΔX_F für den visuellen Kompass zu verwenden, um eine aktuelle Bewegungsrichtung zu berechnen.
Zur Auswertung wird die Datenreihe des Vorverarbeitungsschrittes bestehend aus den ΔX_F Werten zur Bestimmung der Richtung verwendet. Die Datenreihe wird geglättet, indem für jeden Wert ΔX_F eines Frame-Paares F ein gleitender Median mit einer festen Fenstergröße berechnet wird. Dieser berechnete Median-Wert wird anschließend mit einer Konstanten $1/K$ multipliziert. Dabei dient K zur Umrechnung der Drehrichtung und ist abhängig vom Sichtfeld *(Field of View)* und der Auflösung der Kamera. Die Konstante gibt an wie viele Pixel einer Drehung einem Grad entsprechen. Sind Sichtfeld und Auflösung der Kamera bekannt, kann die Konstante K wie folgt berechnet werden:

$$K = \frac{Aufloesung_X}{Sichtfeld}$$

Für eine handelsübliche *GoPro Hero 3+* errechnet sich bei einer Auflösung von 640 x 360 Pixeln und einem Sichtfeld von 64,4 Grad eine Konstante K mit einem Wert von 5,59.
Der visuelle Kompass benötigt möglichst viele *Matches*, die von einer Drehung beeinflusst werden, um ein robustes Ergebnis des Drehwinkels zu erhalten. Durch die hohe Anzahl an *Matches* wird verhindert, dass falsch-positive oder nicht zur Eigenbewegung zugehörige *Matches* das Ergebnis verfälschen.
In einigen speziellen Situationen kann es beim visuellen Kompass beim Geradeaus-Gehen zu einem Drift des Drehwinkels kommen. Eine solche Situation entsteht, sobald über einen längeren Zeitraum auf einer der beiden Bildhälften überdurchschnittlich viele *Matches* gefunden werden. Dies führt dazu, dass ein Ungleichgewicht bei der Berechnung des Drehwinkels entsteht. Daher wird in Abschnitt 5.5.3.2 eine Erweiterung des visuellen Kompass vorgeschlagen, die durch eine Aktivitätserkennung ausschließlich Kurven und Drehungen erkennt und dadurch nur zu diesen Phasen den visuellen Kompass aktiviert.

5.4.5 Erzeugen der Trajektorie

Der letzte Teilprozess erzeugt aus der Kombination der Ergebnisse des visuellen Schrittzählers und des visuellen Kompasses eine Trajektorie. Dabei bestehen die Ergebnisse des visuellen Schrittzählers aus einer Sequenz *Schritt*, die zu jedem entdeckten Schritt den Zeitpunkt in Form der Frame-Nummer enthält. Für den visuellen Kompass wird ebenfalls eine Sequenz *Kompass* als Ergeb-

nis bereitgestellt, die für jede Frame-Nummer die Bewegungsrichtung in Grad enthält. Zusätzlich zu den beiden Sequenzen wird die Schrittlänge der Person, die die Trajektorie gegangen und aufgezeichnet hat, benötigt. Hierfür wird die Formel von Pratama et al. verwendet, die aus der Größe einer Person und einem konstanten Wert die Schrittlänge $L_{Schritt}$ berechnet [188].
Das Erzeugen der Trajektorie erfolgt durch das Iterieren der Schritt-Sequenz *Schritt*. Für jeden Schritt $Schritt_i$ wird anhand der Frame-Nummer die zugehörige Bewegungsrichtung aus der Kompass-Sequenz $Kompass_{Schritt_i}$ ermittelt. Die Koordinaten (x, y) werden für jeden Schritt $Schritt_i$ mit Hilfe der Sinus- und Kosinus-Funktion durch folgende Formeln bestimmt:

$$x_{Schritt_i} = x_{Schritt_{i-1}} + L_{Schritt} * sin(Kompass_{Schritt_i})$$
$$y_{Schritt_i} = y_{Schritt_{i-1}} + L_{Schritt} * cos(Kompass_{Schritt_i})$$

Damit die Trajektorie absolut angegeben werden kann, müssen die Startkoordinaten der Trajektorie bekannt sein und angeben werden.

5.4.6 Evaluierung des Schrittzählers und Kompasses

Im folgenden werden der Schrittzähler sowie der Kompass evaluiert. Dafür wird im ersten Teil der Evaluierung der visuelle Schrittzähler alleine betrachtet und auf seine Genauigkeit geprüft. Es werden dabei unterschiedliche Parameter, wie die Auflösung und Frame-Rate der Kamera, mitberücksichtigt. Der visuelle Schrittzähler wird anschließend mit anderen handelsüblichen Schrittzählern verglichen.
Im zweiten Teil der Evaluierung werden Schrittzähler und Kompass gemeinsam evaluiert. Dafür werden drei Szenarien mit unterschiedlich vorgegebenen Trajektorien verwendet, die von mehreren Personen einzeln aufgezeichnet werden.

5.4.6.1 Genauigkeit des Schrittzählers

In diesem Abschnitt wird der visuelle Schrittzähler basierend auf dem *OrientationScore* evaluiert. Dafür werden erstmals die Bedingungen der Evaluierung erläutert. Dann werden zunächst die Parameter für eine statische Schranke des *OrientationScore* sowie eine geeignete Auflösung und Frame-Rate evaluiert. Anschließend wird das Schritterkennungsverfahren anhand von 250-Schritte-Videosequenzen auf seine Genauigkeit geprüft. Abschließend wird ein Vergleich mit handelsüblichen Schrittzählern gegeben.

Setup Für die Evaluierung wurden unterschiedliche Videosequenzen im Außen- und Innenbereich von unterschiedlichen Personen aufgezeichnet. Jede Videosequenz enthält immer drei unterschiedliche Aktivitäten (Gehen, Laufen

und Stehen). Zur Evaluierung einer geeigneten statischen Schranke sowie einer geeigneten Auflösung und Frame-Rate wurden dafür 100-Schritte-Videos aufgezeichnet. Ein 100-Schritte-Video enthält eine Sequenz von 50 Schritten Gehen, eine längere Stillstands-Phase und eine anschließende Sequenz von 50 Schritten Laufen. Insgesamt wurden neun Videosequenzen von vier unterschiedlichen Nutzern aufgezeichnet.
Für die Evaluierung des Schritterkennungsverfahren und den Vergleich mit anderen Schrittzählern wurden 250-Schritte Videosequenzen aufgezeichnet. Jedes Video enthält eine Sequenz aus 100 Schritten Gehen, einer längeren Stillstands-Phase, einer Sequenz aus 100 Schritten Laufen, gefolgt von einer weiteren längeren Stillstands-Phase und einer Sequenz aus 50 Schritten Gehen. Insgesamt wurden zwölf 250-Schritte Videos von sechs unterschiedlichen Personen aufgezeichnet.
Alle Videosequenzen wurden mit einer *GoPro Hero3+* Kamera mit einem sogenannten *Body-Mount* (Brustgurt zur Befestigung der Kamera) aufgezeichnet (vgl. Abb. 5.9a). Die Kamera ist auf die Geh-Richtung ausgerichtet und zeichnet Videos mit einer Auflösung von 1920 x 1080 Pixeln und einer Frame-Rate von 25 Hz (Frames pro Sekunde) auf. Für den Vergleich mit anderen Schrittzählern wurde ein einfacher Schrittzähler *Haptime Pedometer*, ein *Apple iPhone 6* und ein *Apple iPod Nano (6. Generation)* verwendet (vgl. Abb. 5.9b).

(a) (b)

Abbildung 5.9: Body-Mount Kamera und Schrittzähler: (a) *GoPro Hero3+* Kamera mit *Body-Mount*-Befestigung - (b) Schrittzähler (von links nach rechts): *Haptime Pedometer*, *Apple iPod Nano (6. Generation)* und *Apple iPhone 6*.

Statische Schranke des OrientationScore Eine statische Schranke des *OrientationScore* ist notwendig, um zwischen einem Schritt und anderen Aktivitäten oder Rauschen zu unterscheiden. Da die Entscheidung einer geeigneten

Auflösung und Frame-Rate nicht getroffen und evaluiert wurde, werden für unterschiedliche Varianten jeweils eigene Schrankenwerte evaluiert. Dafür werden beginnend mit einem Wert von 0,01 bis 0,5 in 0,02 Schritten unterschiedliche Werte für eine Schranke eingesetzt. Es werden alle neun 100-Schritte Videosequenzen verwendet. Für jede Auflösung und Frame-Rate wird der beste Wert für eine statische Schranke bestimmt. Auf eine Frame-Rate von 25 FPS wurde verzichtet, da diese sich nur sehr geringfügig im Vergleich zu einer Frame-Rate von 20 FPS unterscheidet. Tabelle 5.1 und 5.2 zeigen, dass ein Wert für eine statische Schranke mit steigender Auflösung sinkt. Je höher die Auflösungen, desto mehr Features und *Matches* werden gefunden. Dies führt jedoch dazu, dass mehr falsch-positive *Matches* gefunden werden, die zu einer allgemeinen Abflachung des *OrientationScore* führen. Für unterschiedliche Frame-Raten unterscheidet sich der optimale Wert einer statischen Schranke nur wenig mit Ausnahme bei einer Frame-Rate von 5 FPS.

Auflösung	45p	90p	180p	360p	720p	1080p
Schranke	0.21	0.20	0.15	0.13	0.10	0.10
Abweichung in %	+ 0 556	− 0 750	+ 0 917	+ 0 361	+ 0 417	+ 0 028

Tabelle 5.1: Optimale Werte einer statischen Schranke für den OrientationScore mit unterschiedlichen Auflösungen unter Verwendung aller Frame-Raten.

Frame-Rate (FPS)	5	10	15	20
Schranke	0.08	0.17	0.16	0.15
Abweichung in %	− 0 574	+ 0 537	− 0 593	+ 0 389

Tabelle 5.2: Optimale Werte einer statischen Schranke für den OrientationScore mit unterschiedlichen Frame-Raten unter Verwendung aller Auflösungen.

Auflösung und Frame-Rate Eine Videosequenz mit einer geringen Auflösung und niedrigeren Frame-Raten wird schneller verarbeitet, weil in ihr weniger Features gefunden und verglichen werden. Jedoch kann dadurch die Qualität der Videosequenz sinken. Für die Evaluierung werden unterschiedliche Auflösungen und Frame-Raten verwendet. Die Auflösungen haben dabei ein Seitenverhältnis von 16:9 und eine vertikale Auflösung von jeweils 45, 90, 180, 360, 720 und 1080 Pixeln. Die Frame-Raten liegen bei 5, 10, 15 und 20 FPS. Auf eine Frame-Rate von 25 FPS wurde verzichtet, da diese sich fast identisch zu einer Frame-Rate von 20 FPS verhält.
Für die Evaluierung werden wieder alle neun 100-Schritte Videosequenzen verwendet. Zur Bestimmung einer geeigneten Frame-Rate und Auflösung werden alle Variation mit einander evaluiert.

-	5 FPS (σ)	10 FPS (σ)	15 FPS (σ)	20 FPS (σ)
45p	69.889 (13.152)	99.444 (18.816)	104.111 (22.684)	120.667 (18.886)
90p	90.333 (7.454)	101.889 (4.148)	99.667 (8.756)	104.333 (6.037)
180p	98.222 (8.942)	105.444 (6.669)	102.889 (6.707)	99.889 (7.062)
360p	104.444 (7.395)	100.778 (3.224)	99.889 (2.885)	98.000 (4.028)
720p	101.333 (5.185)	102.000 (2.625)	100.444 (2.872)	98.111 (5.705)
1080p	103.000 (5.944)	101.444 (3.500)	100.0 (2.749)	95.111 (8.812)

Tabelle 5.3: Mittelwert und Standardabeichung σ aller erkannten Schritte für jede Frame-Rate und Auflösung.

Tabelle 5.3 zeigt deutlich, dass eine Frame-Rate von 5 FPS und eine Auflösung von 45p die schlechtesten Ergebnisse erzeugt. Dagegen ist eine Auflösung von 1080p und eine Frame-Rate von 15 FPS ideal. Vergleicht man die Standardabweichung der Schrittzählung, hat man mit einer Auflösung von 720p und einer Frame-Rate von 10 FPS die geringste Streuung. Eine Auflösung von 45p ist am ungenauesten. Eine Frame-Rate von 5 FPS und 20 FPS ergeben im Mittel ebenfalls ungenaue Werte. Eine Auflösung von 360p, 720p und 1080p mit einer Frame-Rate von 10 FPS und 15 FPS erzielen im Mittel die besten Ergebnisse. Neben der Genauigkeit wird auch die Performance des visuellen Schrittzählers evaluiert. Tabelle 5.4 zeigt, dass mit steigender Frame-Rate und höherer Auflösung die Laufzeit hauptsächlich bedingt durch die Extraktion und den Vergleich der Features ansteigt.

-	5 FPS	10 FPS	15 FPS	20 FPS	∅
45p	0.360	0.708	1.056	1.707	0.958
90p	1.667	3.329	5.003	8.185	4.546
180p	7.882	15.248	23.305	36.939	20.844
360p	34.867	67.000	99.450	159.028	90.086
720p	183.921	363.595	546.081	892.239	496.459
1080p	618.551	1206.902	1818.60	5826.558	2367.653
∅	141.208	276.130	415.583	1154.109	-

Tabelle 5.4: Laufzeit des Schritterkennungsverfahren für eine 100-Schritt Videosequenz mit einer Videolänge von 63 Sekunden. Alle Werte werden in Sekunden angegeben.

Setzt man die Laufzeit des Schritterkennungsverfahren als weiteres Kriterium bei der Wahl der richtigen Auflösung und Frame-Rate ein, erhält man das stabilste Ergebnis mit einer Auflösung von 360p und einer Frame-Rate von 10 FPS. Mit dieser Kombination werden in 67 Sekunden die Schritte fast in Echtzeit erkannt. Die tatsächliche Laufzeit der Videosequenz beträgt 63 Sekunden. Für die weitere Evaluierung werden Frame-Raten von 5 FPS und 20 FPS sowie die Auflösung von 45p ausgeschlossen.

-	90p	180p	360p	720p	1080p
10 FPS	247.666	243.75	249.833	247.083	248.583
15 FPS	240.416	233.416	239.333	225.833	229.583

Tabelle 5.5: Mittelwert aller gezählten Schritte pro Auflösung und Frame-Rate unter Einsatz der Plausibilitätsregel und der statischen Schranke.

Schritterkennungsverfahren Zur Schritterkennung wird das vorgestellte Verfahren aus Abschnitt 5.4.3.2 mit dem *OrientationScore* verwendet. Zusätzlich werden die obligatorischen Korrekturverfahren bestehend aus der *Plausibilitätsregel* und der *statischen Schranke* eingesetzt. Für die statische Schranke werden die evaluierten Werte aus Tabelle 5.1 und 5.2 verwendet.

Zur Evaluierung des Schritterkennungsverfahrens werden die 12 Videosequenzen bestehend aus jeweils 250 Schritten verwendet. Tabelle 5.5 fasst die Ergebnisse aus den 250-Schritte Videosequenzen zusammen. Eine Frame-Rate von 10 FPS sowie eine Auflösung von 360p erzielen insgesamt die besten Ergebnisse.

Vergleich mit anderen Schrittzählern Nach der separaten Evaluierung des visuellen Schrittzählers, wird er mit anderen handelsüblichen Schrittzählern verglichen. Für dieses Experiment werden ein *Haptime Pedometer*, ein *Apple iPhone 6* und ein *Apple iPod Nano (6. Generation)* verwendet. Das iPhone 6 und der iPod Nano werden beide in die Hosentasche einer Versuchsperson gelegt. Die Haptime wird an den Arm angebracht. Während der Aufzeichnung der 250-Schritte-Videosequenzen wurden die eben genannten Schrittzähler an die sechs Versuchspersonen angebracht. Für den Vergleich mit den anderen Schrittzählern wurde der visuelle Schrittzähler mit einer Auflösung von 360p und einer Frame-Rate von 10 eingesetzt. Tabelle 5.6 zeigt die Ergebnisse aller Schrittzähler pro Versuchsperson. Das iPhone 6 und das iPod Nano zählen die 250 Schritte sehr genau, der Fitness-Tracker Haptime dagegen fällt negativ auf und erzeugt einen mittleren Fehler von 5,83 Schritten. Der visuelle Schrittzähler (VSZ) verhält sich zum iPhone 6 und dem iPod Nano relativ ähnlich in der Genauigkeit. Die Standabweichung fällt für den visuellen Schrittzähler etwas schlechter aus und hat eine größere Verteilung im Gegensatz zu dem iPhone 6 und dem iPod Nano.

5.4.6.2 Genauigkeit der Trajektorie

Um den visuellen Schrittzähler in Kombination mit dem Kompass zu evaluieren, werden drei Szenarien mit unterschiedlichen Pfaden betrachtet. Die Videosequenzen wurden innerhalb und außerhalb des Universitätsgebäudes der Ludwig-Maximilians-Universität an der Oettingenstraße 67 in München, Deutschland aufgezeichnet. Die drei Szenarien sowie das Grundsetup werden im folgenden vorgestellt.

Video	Haptime	iPod Nano	iPhone 6	VSZ
user 01 a	246	248	250	242
user 01 b	252	248	247	249
user 02 a	254	252	254	253
user 02 b	251	252	252	253
user 03 a	247	251	251	254
user 03 b	245	251	252	254
user 04 a	252	247	251	252
user 04 b	242	251	249	241
user 05 a	246	253	255	252
user 05 b	214	247	250	242
user 06 a	245	250	248	254
user 06 b	245	244	249	252
Mittelwert	244.167	249.500	250.333	249.833
Standardabweichung	10.262	2.566	2.593	4,896

Tabelle 5.6: Vergleich des visuellen Schrittzählers mit handelsüblichen Schrittzählern. Visueller Schrittzähler (VSZ).

Setup Die Videosequenzen aller Szenarien wurden durch fünf Versuchspersonen aufgezeichnet. Die Versuchspersonen haben unterschiedliche Körpergrößen. Insgesamt wurden 29 Videosequenzen erstellt, wobei die ersten beiden Szenarien durch jeweils 10 Videos und das letzte Szenario aus 9 Videosequenzen abgebildet werden. Zur Vergleichbarkeit der Schritte wurden diese während des Versuches mit Hilfe eines mechanischen Schrittzählers durch den Versuchsleiter mitgezählt. Die Videos wurden mit einer an den Versuchspersonen befestigten *Body-Mount* Kamera *(GoPro Hero3+)* und einer Auflösung von 640 x 360 Pixeln und einer Frame-Rate von 10 FPS aufgezeichnet.

Szenario 1 Der Pfad befindet sich innerhalb des Universitätsgebäudes (vgl. Abb. 5.10). Er hat eine Länge von 148,5 Metern und führt größtenteils durch Flure und Gänge des Gebäudes. In der Mitte des Pfades liegen die Kurven 2 bis 6, die sich innerhalb der Cafeteria des Gebäudes befinden. Die Cafeteria ist ein großer Raum mit zahlreichen Objekten, wie Tische, Stühle und Lebensmittelautomaten. In einigen Durchläufen gab es kleine Variationen im Bereich der Cafeteria, welche mit einer gepunkteten Linie in Abbildung 5.10 zwischen Kurve 5 und 6 dargestellt werden.
Abbildung 5.11a und 5.11b zeigen ein Beispiel einer gut und einer schlecht rekonstruierten Trajektorie aus allen Versuchsdurchläufen. In fast allen Versuchsdurchläufen entsteht durch den visuellen Kompass immer der selbe identische Drift nach links, der sich bis zur Kurve 1 hinzieht. Insgesamt unterscheiden sich die Ergebnisse in ihrer Genauigkeit nur durch die ungenauen Messungen an den Kurven 5 bis 7. Speziell Kurve 7 erzeugt bei allen Trajektorien der Versuchspersonen große Messfehler. Selbst bei der besten rekonstruierten Tra-

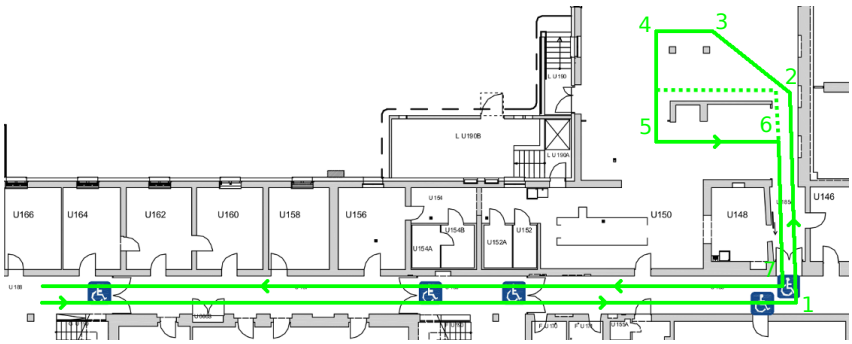
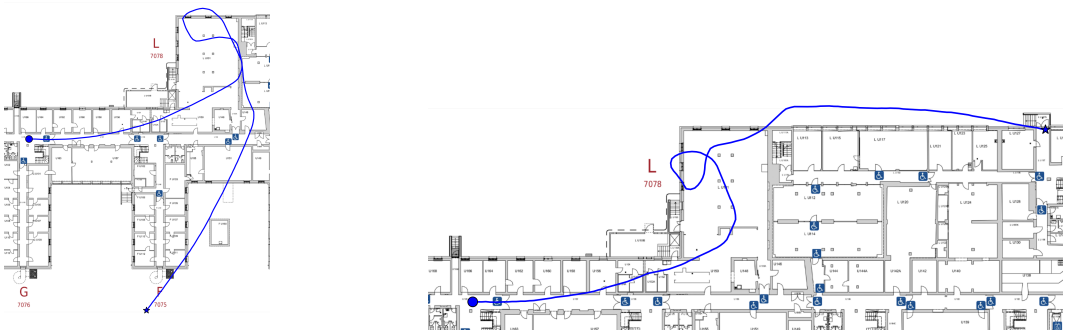


Abbildung 5.10: Szenario 1: In einigen Durchläufen gab es kleine Variationen im Bereich der Cafeteria, welche mit einer gepunkteten Linie zwischen Kurve 5 und 6 dargestellt werden. Die Geh-Richtung wird anhand der Pfeile verdeutlicht, die Kurven sind nummeriert. Die Länge des Pfades beträgt 148,5 Meter. [18]

jektorie wird durch den visuellen Kompass ein falscher Winkel von 45 Grad statt 90 Grad gemessen.



(a) (b)

Abbildung 5.11: Trajektorien des Szenario 1: Der Kreis stellt den Startpunkt und der Stern den Endpunkt dar. (a) Gut rekonstruierte Trajektorie - (b) Schlecht rekonstruierte Trajektorie.

Die gemessene Distanz des Schrittzählers erzielt dagegen bessere Ergebnisse. Insgesamt wird ein durchschnittlicher Fehler von 8,86 Metern mit einer Standardabweichung von 21,76 Meter erreicht. Das heißt, dass im Mittel 9 Schritte (5%) zu viel gezählt werden.
Abbildung 5.12 stellt die Ergebnisse nochmals grafisch dar. Für kurze Distanzen sind die Messfehler relativ gering. Bei einer Distanz von 5 Metern liegt er bei unter 0,77 Meter (Standardabweichung 0,17 Meter). Diese nehmen bei einer Distanz von 25 Metern zu und erreichen einen Messfehler von 5,52 Meter (Standardabweichung 1,19 Meter). Bei einer Distanz von 50 Metern ist der Messfehler schon deutlich höher und liegt bei 11,31 Metern (Standardabweichung 3,72 Meter). Der visuelle Kompass weist ebenfalls bei kurzen Distanzen

geringe Messfehler auf. Bei 5 Meter erreicht der visuelle Kompass einen Messfehler von 2,25 Grad und bei 50 Metern bereits 21,56 Grad. Allgemein weisen die Ergebnisse eine hohe Abweichung auf und werden mit steigender Distanz zunehmend schlechter. Besonders in diesem Szenario ist die Aussagekraft des Kompass nicht so hoch.

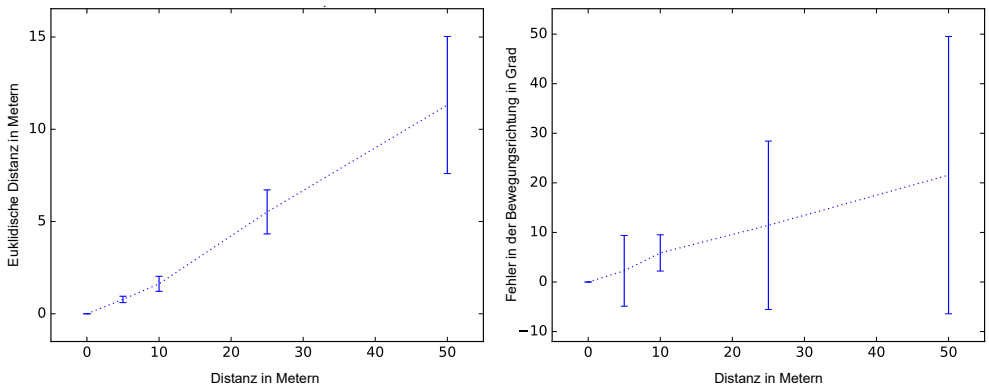


Abbildung 5.12: Szenario 1: Messfehler über alle Trajektorien der Versuchspersonen in Metern (links) und in Grad (rechts). [18]

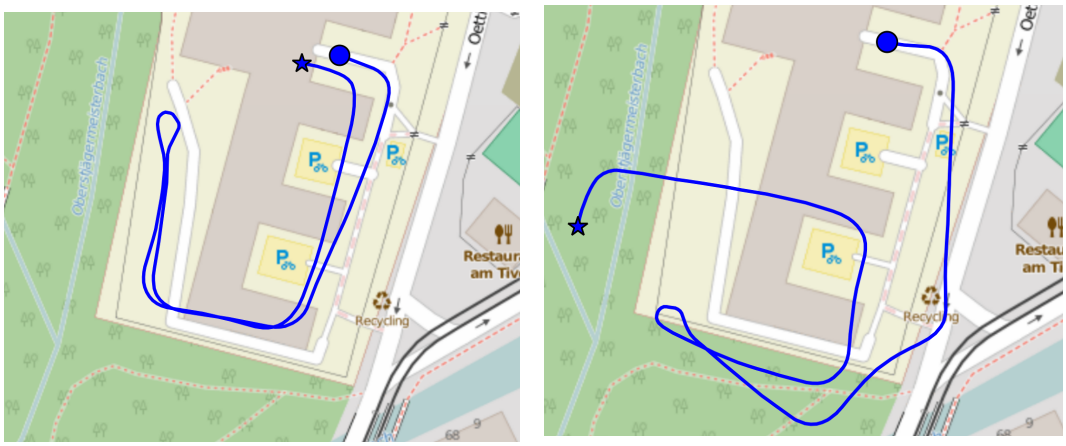
Zusammengefasst zeigen die Ergebnisse, dass der Pfad in Teilen recht gut rekonstruiert werden kann. Der visuelle Kompass hat dagegen eine sehr hohe Abweichung und verschlechtert damit die Rekonstruktion der Trajektorie insgesamt. Ein Problem stellen die weißen Wände in den Gängen dar. Sobald auf einer Seite mehr Textur vorhanden ist, führt dies zu einem Drift des visuellen Kompasses. In den Kurven verhält es sich ähnlich zu den weißen Wänden, sobald in einer Kurve mehr Texturen auf einer Seite zu sehen sind, führt dies in vielen Fällen zu einer falschen Berechnung des Winkels.

Szenario 2 Der Pfad in Szenario 2 ist in Abbildung 5.13 abgebildet. Dieser verläuft um das Gebäude herum und hat eine Länge von 387,5 Metern. Der erste Abschnitt zwischen Kurve 1 und Kurve 2 verläuft durch den Parkplatz des Gebäudes. Der restliche Abschnitt besteht aus einer Parkanlage, wobei das Gebäude beim Hinweg auf der rechten Seite und beim Rückweg auf der linken Seite liegt. In den Videosequenzen tauchen teilweise andere Personen auf, die das Sichtfeld der Kamera betreten.
Abbildungen 5.14a und 5.14b veranschaulichen beispielhaft eine gut und ein schlecht rekonstruierte Trajektorie aller Versuchsdurchläufe. Abgesehen von den Drifts, besonders in Kurve 2, sind die Ergebnisse relativ genau. Gerade Strecken werden akkurat erkannt und von dem Verfahren gut rekonstruiert. In der zweite Hälfte des Pfades, also beim Rückweg, erzeugt der visuelle Kompass in fast allen Trajektorien sehr wenige Drifts.
Betrachtet man die gemessene Distanz des Schrittzählers, erzeugt er einen durchschnittlichen Fehler von 3,8 Metern mit einer Standardabweichung von

Abbildung 5.13: Szenario 2: Ein Rundgang um das Unverisitätsgebäude mit einer Länge von 387,5 Metern. [18]

13,89 Metern. Das heißt, dass im Mittel 4 Schritte (0,84%) zu viel gezählt werden.
Abbildung 5.15 stellt die Ergebnisse nochmals grafisch dar. Der Schrittzähler hat bei Distanzen bis zu 10 Metern einen sehr geringen Messfehler. Bei einer Distanz von 5 Metern ergibt sich ein Messfehler von 0,89 Metern (Standardabweichung 0,22 Meter) und bei einer Distanz von 10 Metern sind es 2,26 Meter (Standardabweichung 0,47 Meter). Bei einer Distanz von 25 Metern steigt der Messfehler auf 7,95 Meter (Standardabweichung 2,41 Meter). Bei einer Distanz von 100 Metern liegt der Messfehler bereits bei 32,43 Metern (Standardabweichung 4,26 Meter). Der visuelle Kompass weist größten Teils nur positive Werte für den Messfehler auf. Das heißt, dass grundsätzlich ein Drift nach links den Messfehler verursacht hat. Insgesamt liegt der Messfehler zwischen 5,06 Grad und 9,67 Grad, wobei die Standardabweichung bei einer Distanz bis zu 75 Metern zwischen 7,06 und 10,28 Grad liegt. Erst bei einer Distanz von 100 Metern steigt die Standardabweichung stark an und liegt bei 20,36 Grad.
Zusammengefasst lassen sich die Ergebnisse aus Szenario 2 positiv bewerten. Die größte Fehlerquelle scheint wieder der visuelle Kompass zu sein, der in Kurven die tatsächliche Drehung nicht wirklich erfassen kann. Zusätzlich tauchten bewegliche Objekte (z.B. Fahrradfahrer, fahrende Autos) während der Aufzeichnungen auf, die als Störfaktoren den visuellen Kompass negativ beeinflusst haben. Insgesamt sind die Ergebnisse im Vergleich zum Szenario 1 deutlich besser.

Szenario 3 Der Pfad aus Szenario drei entspricht ebenfalls einem Rundgang um das Universitätsgebäude (vgl. Abb. 5.16). Der Pfad hat insgesamt eine Länge von 370,5 Metern. Ein Teil des Pfades verläuft an einer befahrenen Straße. Die andere Hälfte verläuft im Innenhof des Universitätsgebäudes. Auf den Videosequenzen sind überdurchschnittlich viele Passanten oder fahrende Fahrzeuge zu sehen.
Abbildungen 5.17a und 5.17b veranschaulichen beispielhaft eine gut und eine



(a) (b)

Abbildung 5.14: Trajektorien des Szenarios 2: Der Kreist stellt den Startpunkt und der Stern den Endpunkt dar. (a) Gut rekonstruierte Trajektorie - (b) Schlecht rekonstruierte Trajektorie.

schlecht rekonstruierte Trajektorie aller Versuchsdurchläufe.

In allen Trajektorien der Versuchspersonen wird Kurve 5 sehr schlecht erkannt und erzeugt die größten Messfehler. Allgemein erzeugen die Kurven in fast allen Trajektorien einen zu hohen Messfehler. Auch ein leichter Drift in geraden Streckenabschnitten entsteht durch fahrende Autos und vorbeigehenden Passanten.

Der Schrittzähler erzeugt einen Messfehler von 3,6 Metern (Standardabweichung 13,02 Metern). Das heißt, dass im Mittel 4 Schritte (0,9%) zu viel gezählt werden.

Abbildung 5.18 veranschaulicht nochmals die Ergebnisse. Die Ergebnisse verhalten sich fast identisch zu Szenario 2 bei Distanzen bis zu 50 Metern. Erst bei Distanzen ab 75 Metern steigt der Messfehler erheblich an und liegt bei 44,24 Metern (Standardabweichung 9,93 Meter). Der visuelle Kompass weist bei kleinen Distanzen bis zu 25 Metern einen geringen Messfehler auf. $-0{,}36$ Grad bei einer Distanz von 5 Meter (Standardabweichung 4,55 Grad), 5,57 Grad für eine Distanz von 10 Metern (Standardabweichung 5,41 Grad) und 3,31 Grad bei einer Distanz von 25 Metern (Standardabweichung 12,89 Grad). Bei 100 Metern steigt der Messfehler erheblich an und liegt bereits bei 44,09 Grad (Standardabweichung 32,28 Grad)

Insgesamt fallen die Ergebnisse schlechter aus als bei Szenario 2, welches ebenfalls wie dieses Szenario außerhalb des Gebäudes aufgezeichnet wurde. Die schlechten Ergebnisse sind nicht überraschend, da in diesem Szenario bewusst mehr Hindernisse eingeführt wurden. Speziell der Umstand, dass vermehrt Passanten, Fahrradfahrer und fahrende Fahrzeuge im Sichtfeld der Kamera waren, sollten das Verfahren auf seine Robustheit gegenüber solcher Störfaktoren evaluieren. Hinzu kam, dass die Kurven ebenfalls schwieriger als in den

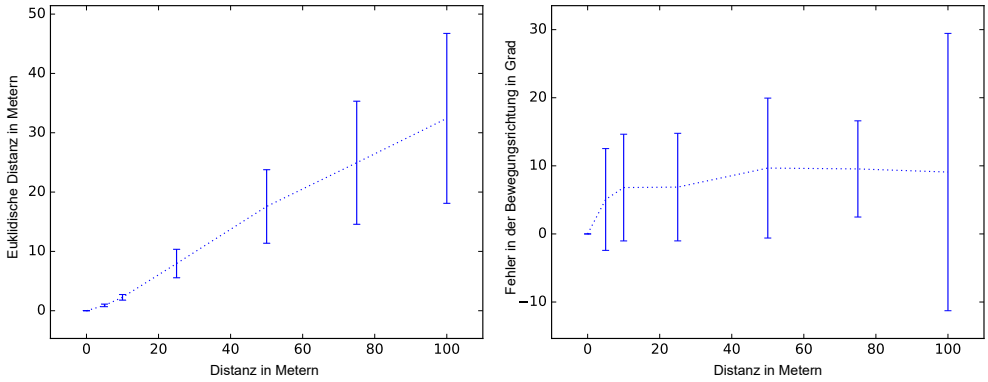


Abbildung 5.15: Szenario 2: Messfehler über alle Trajektorien der Versuchspersonen in Metern (links) und in Grad (rechts). [18]

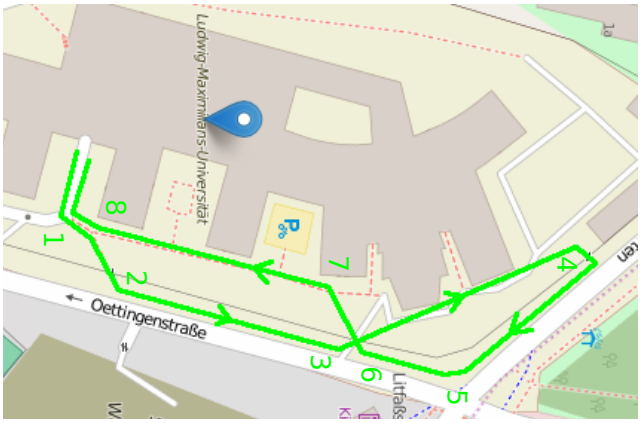
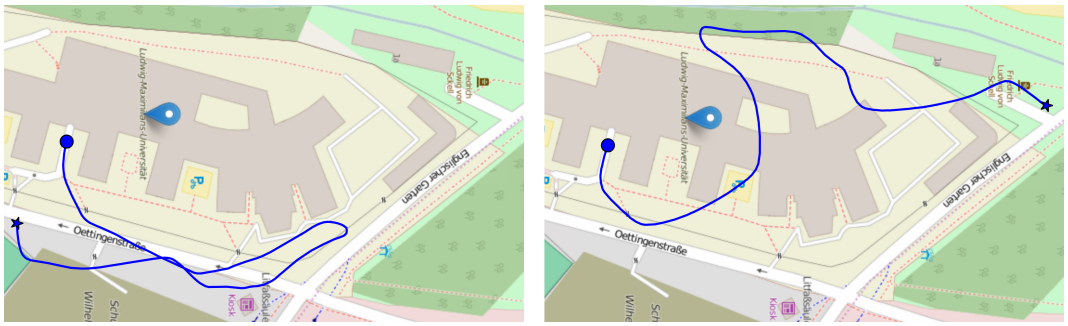


Abbildung 5.16: Szenario 3: Ein Rundgang um das Unverisitätsgebäude an der Straße entlang und durch den Innenhof. Die Länge des Pfades beträgt 370,5 Metern. [18]

anderen Szenarien zu erfassen waren. Kurve 4 bestand aus einer Drehung von 180 Grad. Kurve 5 war ebenfalls schwieriger zu erfassen, da der Verlauf der Kurve sehr lang und die Winkelveränderung dadurch sehr gering war. Die Ergebnisse zeigen, dass der Schrittzähler sehr genau ist und die Trajektorie grundsätzlich durch den visuellen Kompass verfälscht wird.

5.5 Verarbeitung von visuellen Merkmalen zur Aktivitätserkennung

Bereits in Abschnitt 5.4.3.2 wird für das Schrittkorrekturverfahren der *Stillstandserkennung* die $\Delta Scale_F$ zur Bestimmung einer solchen Stillstands-Phase eingesetzt. Der Einsatz der *Stillstandserkennung* hat dazu beigetragen, die $\Delta Scale_F$ und andere Attribute eines SURF-Features zu verwenden, um Aus-



(a) (b)

Abbildung 5.17: Trajektorien des Szenario 3: Der Kreist stellt den Startpunkt und der Stern den Endpunkt dar. (a) Gut rekonstruierte Trajektorie - (b) Schlecht rekonstruierte Trajektorie.

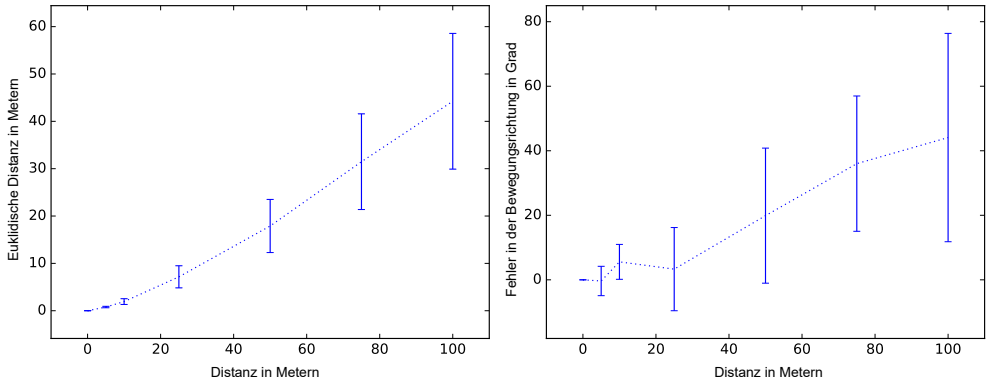


Abbildung 5.18: Szenario 3: Messfehler über alle Trajektorien der Versuchspersonen in Metern (links) und in Grad (rechts). [18]

sagen über die Aktivitäten einer Person zu treffen. Insbesondere Aktivitäten wie das Gehen, Laufen oder Stehen werden betrachtet und anhand der relativen Veränderung der Feature-Attribute festgestellt. Dafür kann bereits anhand der Schrittfrequenz aus dem visuellen Schrittzähler eine Aussage über die aktuelle Aktivität einer Person getroffen werden. Neben den klassischen Aktivitäten wie *Laufen*, *Gehen* und *Stehen* werden andere Aktivitäten wie *Treppensteigen*, *Kurven gehen* oder *auf der Stelle drehen* ebenfalls untersucht.
Im Folgenden werden drei Ansätze der Aktivitätserkennung anhand von Feature-Attributen vorgestellt. Der erste Ansatz kategorisiert anhand der $\Delta Scale_F$ einfache Aktivitäten wie Gehen, Stehen und Laufen. Der zweite Ansatz nutzt die Schrittfrequenz des visuellen Schrittzählers, um die selben Aktivitäten zu bestimmen. Der dritte Ansatz betrachtet alle Attribute eines Features und ermittelt anhand von Klassifikatoren, die durch Verfahren des maschinellen Lernens erzeugt werden, Aktivitäten eines Nutzers. Alle drei Ansätze werden anschließend evaluiert und gegenüber gestellt und diskutiert.

5.5.1 Scale-Attribut und Schrittfrequenz

In Abschnitt 5.4.3.2 wurde bereits gezeigt, dass durch die Einführung eines Grenzwertes die $\Delta Scale_F$ verwendet werden kann, um Stillstands-Phasen von anderen Aktivitäten zu unterschieden. Führt man einen weiteren Grenzwert ein, der zwischen den Aktivitäten *Gehen* und *Laufen* unterscheidet, lassen sich insgesamt drei Aktivitäten unterscheiden. Abbildung 5.19 zeigt beispielhaft eine Videosequenz, in der die drei Aktivitäten *Laufen*, *Stehen* und *Gehen* in der genannten Reihenfolge, durch den geglätteten $\Delta Scale_F$ abgebildet werden.

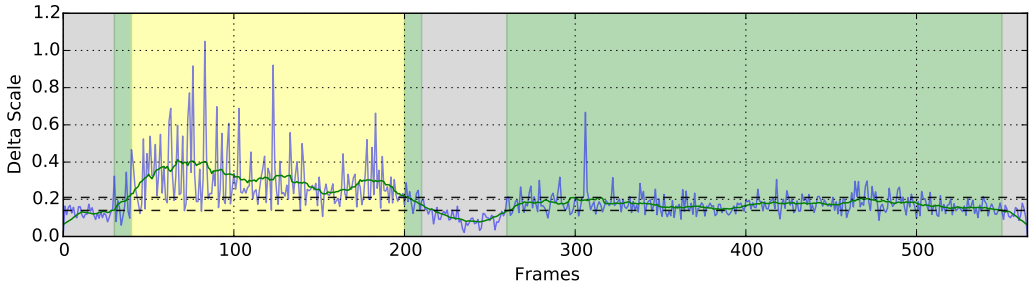


Abbildung 5.19: Aktivitätserkennung anhand der $\Delta Scale_F$ und zwei Grenzwerten (gestrichelte Linie): Aktivitäten in der genannten Reihenfolge - *Laufen* (grün), *Stehen* (grau) und *Gehen* (grün).

Es werden alle drei Aktivitäten durch den Grenzwert genau getrennt und in der selben Reihenfolge erkannt. Die Übergänge in den Aktivitäten, besonders vor der Laufaktivität, werden korrekt erkannt. Der Proband in der Videosequenz steht tatsächlich einige Frames still und benötigt ebenfalls ein paar Frames, um auf eine entsprechende Laufgeschwindigkeit zu kommen. Den selben Übergang erkennt man auch kurz nach der Laufaktivität. Damit die $\Delta Scale_F$ zur Aktivitätserkennung verwendet werden kann, wird sie anhand eines gleitenden Mittelwertes geglättet. Dafür wird eine Fenstergröße von 20 Frames eingesetzt. Für die ersten Versuche, die $\Delta Scale_F$ zur Aktivitätserkennung zu verwenden, wurden Videosequenzen mit einer Auflösung von 360p und einer Frame-Rate von 10 FPS eingesetzt. Diese ersten guten Ergebnisse haben dazu geführt, die $\Delta Scale_F$ als eigenen Ansatz einer Aktivitätserkennung zu verwenden und anschließend mit den anderen beiden Ansätzen zu vergleichen.

Neben dem Ansatz, die $\Delta Scale_F$ für die Aktivitätserkennung zu verwenden, wird ein weiterer Ansatz vorgestellt, der die Schrittfrequenz aus dem visuellen Schrittzähler verwendet. Die Schrittfrequenz wird auf Basis der *Schritte* pro *Minute* berechnet. Dafür wird der Ansatz von Quervain et al. eingesetzt [197]. Diese schlagen vor die drei Aktivitäten *Laufen*, *Gehen* und *Stehen* in einzelne Wertebereiche anhand ihrer Schrittfrequenz aufzuteilen (vgl. Tabelle 5.7).
Der Wert für die Schrittfrequenz wird für jedes Frame-Paar wie folgt ermit-

keine Bewegung bzw. Stehen	Schrittfrequenz ≤ 50
Gehen	$50 <$ Schrittfrequenz ≤ 130
Laufen	$130 <$ Schrittfrequenz

Tabelle 5.7: Unterteilung der Schrittfrequenz in drei Aktivitäten. Die Schrittfrequenz wird aus den getätigten Schritten pro Minute berechnet. [197].

telt. Ein Fenster von zwei Sekunden wird um das aktuelle Frame-Paar gelegt und die enthaltenen Schritte gezählt. Anschließend wird der Wert für jedes zwei Sekunden Fenster auf eine Minute hochgerechnet und durch einen gleitenden Mittelwert nochmals geglättet. Erste Ergebnisse haben gezeigt, dass der Ansatz die Schrittfrequenz zu Aktivitätserkennung zu verwenden, gut funktioniert. Daher wird auch dieser Ansatz mit den anderen beiden Varianten der Aktivitätserkennung verglichen.

5.5.2 Klassifizierung der Feature-Attribute

Der dritte und letzte Ansatz der Aktivitätserkennung geht einen Schritt weiter und verwendet alle Attribute eines Features. Hierfür werden Verfahren des maschinellen Lernens verwendet, die auf Basis von Trainingsdaten geeignete Klassifikatoren entwickeln. Diese werden verwendet, um aus einer unbekannten Videosequenz Aktivitäten zu erkennen.
Damit eine solche Aktivitätserkennung reibungslos funktioniert, müssen einige Vorarbeiten geleistet werden. Zunächst müssen Aktivitäten definiert werden, die erkannt werden sollen. Anschließend benötigt man für jede definierte Aktivität ausreichend Trainingsdaten. Dann müssen die Daten in der Trainingsphase verarbeitet werden, um aus ihnen robuste Klassifikatoren zu erhalten. Erst dann kann mit Hilfe der Klassifikatoren in der Erkennungsphase eine Videosequenz auf ihre Aktivitäten klassifiziert werden.

5.5.2.1 Auswahl der Aktivitäten

Nicht jede Aktivität kann durch einen Klassifikator zu 100 Prozent erkannt werden. Jedoch wurde bereits gezeigt, dass sich klassische Aktivitäten, wie das *Laufen*, *Gehen* und *Stehen*, bereits einfach anhand der $\Delta Scale_F$ unterscheiden lassen (vgl. 5.5.1). Diese Eigenschaften lassen darauf schließen, dass sie durch einen Klassifikator ebenfalls unterscheidbar sind. Andere Aktivitäten wie das *Treppensteigen*, *Kurven gehen* oder *auf der Stelle drehen* stellen besonders für das visuelle Odometrie-Verfahren interessante Aktivitäten dar. Der visuelle Kompass würde sich anhand der Kurvenerkennung entscheidend verbessern lassen, indem der Kompass nur aktiviert wird solange eine solche Aktivität erkannt wird.
Daher werden folgende Aktivitäten für diesen Ansatz der Aktivitätserkennung verwendet und untersucht:

- Stehen bzw. keine Bewegung
- Gehen
- Laufen
- Kurven gehen (rechts/links)
- Treppensteigen (auf/ab)
- auf der Stelle drehen (rechts/links)

Für alle Aktivitäten müssen längere Videosequenzen von unterschiedlichen Nutzern aufgezeichnet werden. Entscheidend für die Aufzeichnung der Daten ist, dass keine anderen Störfaktoren während der Ausführung einer Aktivität vorkommen, da sonst Verunreinigungen in den Daten bestehen. Die Videosequenzen werden für die Trainingsphase, in der die Klassifikatoren bestimmt werden, vollständig gelabelt.

5.5.2.2 Entwicklung eines Klassifikators

Ein Klassifikator beschreibt eine Reihe von Merkmalen diskreter Werte, die innerhalb eines Zeitfensters vorhanden sind. Dafür müssen markante Merkmale gefunden werden. Es werden aus den Videosequenzen der Trainingsdaten SURF-Features extrahiert und aus dem Mittel aller *Matches* eines Frame-Paares F die Attribute ΔX_F, ΔY_F, $\Delta Orientation_F$, $OrientatonScore_F$ und $\Delta Scale_F$ berechnet.
Da sich eine Aktivität nicht auf ein Frame-Paar reduzieren lässt, müssen alle Feature-Attribute innerhalb eines Zeitfensters betrachtet werden. Damit man eine Aussage über jedes Feature innerhalb eines Zeitfensters treffen kann, werden die Werte pro Feature zu einem diskreten Wert zusammengefasst. Hierfür werden die Mittelwerte, die Medianwerte, die Standardabweichung und die Frequenz eines Features betrachtet. Zudem wird die Korrelation zu anderen Features ebenfalls berechnet.
Folgende Merkmale innerhalb eines Zeitfenster werden für die Berechnung der einzelnen Klassifikatoren eingesetzt:

- **MEAN:** Mittelwert.
- **ABS_MEAN:** Verwendung des Absolutbetrags zur Berechnung des Mittelwertes.
- **MEDIAN:** Median.
- **ABS_MEDIAN:** Verwendung des Absolutbetrags zur Berechnung des Medians.

- **FRQ:** Bestimmen der Hauptfrequenz durch Anwendung der Fourier-Transformation.
- **STD:** Standardabweichung des Mittelwertes.
- **CORR:** Korrelationen zwischen ΔX_F und ΔY_F sowie jeweils zwischen der $\Delta Orientation_F$.

5.5.3 Evaluierung der Aktivitätserkennung

Im folgenden wird die Aktivitätserkennung der drei vorgestellten Ansätze evaluiert. Dafür wird zunächst der Ansatz der Klassifikatoren alleine betrachtet und anschließend mit den Verfahren der $\Delta Scale_F$ und der Schrittfrequenz verglichen.

5.5.3.1 Setup

Für die Evaluierung der Aktivitätserkennung anhand der Klassifikatoren wurden längere Trainingsdaten für die einzelnen Aktivitäten aufgezeichnet. Insgesamt wurden pro Aktivität Videosequenzen mit einer Länge von 3 Minuten aufgezeichnet. Diese wurden jeweils von drei unterschiedlichen Versuchspersonen ausgeführt, sodass insgesamt pro Aktivität 9 Minuten zur Verfügung stehen. Diese Trainingsdaten wurden für die Klassifikatoren verwendet. Das Zeitfenster einer vollständigen Aktivität kann innerhalb von einer bis fünf Sekunden liegen. Daher werden für die Evaluierung unterschiedliche Größen verwendet. Zur Berechnung der Klassifikatoren in der Trainingsphase und für die anschließende Erkennungsphase der Aktivitäten werden folgende Verfahren eingesetzt: die *Support Vector Machine*, der *Random Forest-Klassifikator*, ein *Entscheidungsbaum* und eine *Nächste-Nachbarn-Klassifikation*.
Für den Vergleich mit den anderen Ansätzen der Aktivitätserkennung werden ausschließlich die Aktivitäten *Laufen*, *Gehen* und *Stehen* untersucht. Als Testdaten werden die selben zwölf 250-Schritte Videosequenzen, die bereits zur Evaluierung der Schrittzähler verwendet wurden, eingesetzt (vgl. Abs. 5.4.6). Jede dieser Videosequenzen beinhaltet die drei zu untersuchenden Aktivitäten. Die Videosequenzen haben eine Auflösung von 640 x 360 Pixel (360p) und eine Frame-Rate von 10 FPS.

5.5.3.2 Evaluierung der Klassifikatoren

Insgesamt hat sich gezeigt, dass der Klassifikator für die Aktivität *Treppensteigen* die Ergebnisse negativ beeinflusst. Das Klassifizieren von Treppen-auf-oder-absteigen scheint insgesamt mit allen anderen Aktivitäten Überschneidungen zu haben und verfälscht dadurch die Ergebnisse sehr stark.
Lässt man die Aktivität *Treppensteigen* weg, ergeben sich insbesondere für die klassischen Aktivitäten wie *Laufen*, *Gehen* und *Stehen* gute Ergebnisse. Mit Ausnahme von einigen Ausreißern liegt die Erkennungsrate bei 92%. Die

Ausreißer entstehen besonders bei Übergängen von einer Aktivität wie *Stehen* zu *Laufen* oder umgekehrt. Hier erkennt das Verfahren meist die Aktivität *Kurve gehen* oder *auf der Stelle drehen*. Die Aktivitäten *Kurve gehen* sowie *auf der Stelle drehen* werden größtenteils wiedererkannt, jedoch hängt das immer von der jeweiligen Kurve oder Drehung ab. Gerade das Passieren von engen Kurven wird meist als Aktivität *auf der Stelle drehen* erkannt. Bei einem zu groß gewählten Fenster verhält es sich für die Aktivität *auf der Stelle drehen* ähnlich, da sehr kurzläufige Drehungen auf der Stelle allgemein als ein Kurven gehen erkannt werden. Fasst man beide Aktivitäten zusammen, lässt sich eine allgemeine Kurvenerkennung realisieren, die für das Verfahren des visuellen Kompass eingesetzt werden kann.
Eine ideale Fenstergröße ist gerade bei sehr kurzen Aktivitäten entscheidend. Als geeignete Größe hat sich ein Fenster von zwei Sekunden ergeben. Die Methoden zur Klassifizierung haben relativ ähnliche Werte erzielt, sobald man die Aktivität *Treppensteigen* herausnimmt. Die Erkennungsrate liegt für die *Nächste-Nachbarn-Klassifikation* bei 92,6%, für den *Random Forest-Klassifikator* bei 92,4%, für die *Support Vector Machine* bei 90,7% und für den Klassifikator mit dem *Entscheidungsbaum* bei 87,3%.
Abbildung 5.20 zeigt ein Beispiel für einen Videoausschnitt, indem eine Person geht, kurz stehen bleibt, anschließend eine Kurve geht und dann gefolgt von einem kurzen Moment der Ruhe wieder weiter geht. Der erste Plot, der ΔX_F und ΔY_F abbildet, zeigt für alle Zustände der Aktivitäten charakteristische Verhaltensweisen. Auch die $\Delta Orientation_F$ und der $OrientationScore_F$ zeigen wie erwartet bei der Aktivität *Gehen* das charakteristische Schrittmuster. Bei der Stillstands-Phase sowie in der Kurve sind die Ausschläge nicht mehr klar zu unterscheiden. Plot drei mit der $\Delta Scale_F$ verhält sich ebenfalls wie erwartet sehr unauffällig, da insbesondere bei den Aktivitäten *Stehen* und *in die Kurve gehen* keine schnellen Vorwärts-Bewegungen stattfinden, die zu einem größeren Differenzwert der *Scale*-Attribute führen.
Zusammenfassend hat die Evaluierung ergeben, dass die klassischen Aktivitäten sehr gut erkannt werden können. Die Aktivität *Treppensteigen* lässt sich fast gar nicht klassifizieren. Aktivitäten wie *Kurve gehen* und *auf der Stelle drehen* lassen sich ebenfalls klassifizieren. Es empfiehlt sich jedoch, beide Aktivitäten zusammenzulegen und in Kombination mit dem visuellen Kompass als unterstützenden Mechanismus zu nutzten. Mit Ausnahme des *Entscheidungsbaumes* haben alle anderen gewählten Methoden der Klassifizierung mit einer Fenstergröße von zwei Sekunden eine hohe Erkennungsrate. Das schlechtere Abschneiden des *Entscheidungsbaumes* wird durch die geringe Anzahl an Trainingsdaten zusätzlich bedingt.

5.5.3.3 Vergleich der Aktivitätserkennungsansätze

Um alle drei Ansätze der Aktivitätserkennung miteinander vergleichen zu können, werden die selben Videosequenzen, die bereits zur Evaluierung des Schrittzählers verwendet werden, eingesetzt. Die Videosequenzen beinhalten alle die

Abbildung 5.20: Klassifizierung von Aktivitäten anhand der Feature-Attribute. Von oben beginnend stellt Plot 1 die ΔX_F und ΔY_F, Plot 2 die $\Delta Orientation_F$ und den $OrientationScore_F$ und Plot 3 die $\Delta Scale_F$ dar. Plot 4 stellt die erkannten Aktivitäten in jeweils zwei Sekunden Fenstern dar. Es werden Gehen (grün), Stehen (Grau) und in die Kurve gehen (Gelb) erkannt.

drei klassischen Aktivitäten *Laufen*, *Gehen* und *Stehen*. Insgesamt werden zwölf 250-Schritte Videos von sechs unterschiedlichen Personen für den Vergleich verwendet. Für die Klassifizierung werden die Klassifikatoren mit Hilfe der bereits verwendeten Methoden und einer Fenstergröße von zwei Sekunden pro Aktivität erzeugt.

Da die ersten beiden Ansätze eine Aktivität jeweils pro Frame berechnen, werden die Werte auf eine Fenstergröße von zwei Sekunden interpoliert, um sie mit den Verfahren der Klassifikatoren vergleichbar zu machen. Dafür wird der Mittelwert aus allen Werten, die innerhalb dieser Fenstergröße bestimmt werden, berechnet. Jede getätigte Aktivität aus den 250-Schritte Videosequenzen ist durch ein zwei Sekunden Fenster ebenfalls gelabelt. Insgesamt sind 1076 Fenster mit einer der drei Aktivitäten gelabelt.

KNN	SVM	RFC	DTC	Schrittfrequenz	$\Delta Scale_F$
92,9%	93,8%	92,8%	89,9%	86,8%	59,4%

Tabelle 5.8: Vergleich der drei Ansätze der Aktivitätserkennung (Erkennungsrate aller Aktivitäten in Prozent). *Nächste-Nachbarn-Klassifikation* (KNN), *Support Vector Machine* (SVM), *Random Forest-Klassifikator* (RFC) und Klassifikator mit *Entscheidungsbaum* (DTC).

Tabelle 5.8 fasst die Ergebnisse der Evaluierung zusammen. Diese zeigen, dass der dritte Ansatz mit den Klassifikatoren weitaus bessere Ergebnisse erzielt als die beiden anderen Ansätze. Dagegen weist der erste Ansatz, der die $\Delta Scale_F$ zur Erkennung der Aktivitätserkennung verwendet, mit 59,4% schlechte Ergebnisse auf. Der zweite Ansatz, der die Schrittfrequenz zur Aktivitätserkennung verwendet, hat zwar im Vergleich eine bessere Erkennungsrate mit 86,8%, jedoch übertrifft sie nie die Erkennungsrate einer der Methoden des dritten Ansatzes.

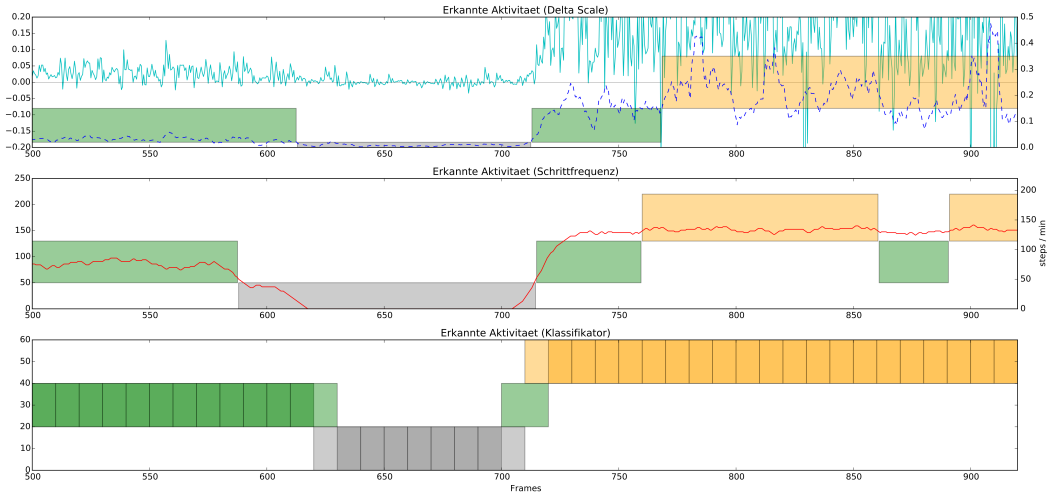


Abbildung 5.21: Beispielgraph der Aktivitätserkennung aller drei Ansätze. Videosequenz besteht aus einer Geh-, Steh- und abschließenden Lauf-Phase. Von oben beginnend zeigt der erste Plot die Aktivitätserkennung anhand der $\Delta Scale_F$, der zweite Plot anhand der Schrittfrequenz und der dritte Plot anhand der Klassifikatoren (SVM). Aktivität *Gehen* (grün), *Laufen* (gelb) und *Stehen* (grau).

Abbildung 5.21 zeigt beispielhaft einen Ausschnitt der Aktivitätserkennung aller drei Ansätze. Dabei besteht die Videosequenz aus einer Geh-, Steh- und abschließenden Lauf-Phase. Der erste Plot von oben beginnend stellt die Aktivitätserkennung anhand der $\Delta Scale_F$ dar. Der Ansatz erkennt zwar vollständig alle Aktivitäten richtig, jedoch werden sie verzögert oder zu lang erkannt. Der

zweite Plot stellt die Aktivitätserkennung anhand der Schrittfrequenz dar. Die Lauf-Phase wird fehlerhaft erkannt, da die Schrittfrequenz zum Teil schwankt. Allgemein ist zu verzeichnen, dass die Schrittfrequenz einige Aktivitäten länger erkennt als sie tatsächlich andauern. Der dritte Plot stellt die Aktivitätserkennung anhand der Klassifikatoren dar. In dem Fall wurden die Klassifikatoren mit der *Support Vector Machine* generiert, da diese sich im Gegensatz zu den anderen Klassifikatoren bei einer geringeren Anzahl an unterschiedlichen Aktivitäten am besten verhält. Die Erkennung verläuft fehlerfrei und ganz genau. Die Steh-Phase, die von den anderen beiden Ansätzen zu lange oder verzögert erkannt wird, kann der Klassifikator exakt bestimmen. Selbst die Übergänge von der Geh- zur Steh- sowie von der Steh- zur Lauf-Phase werden exakt erkannt.

5.6 Zusammenfassung

Dieses Kapitel beschäftigte sich mit der Entwicklung eines Verfahrens der visuellen Odometrie und einer Aktivitätserkennung basierend auf den Attributen visueller Features. Dafür wurden die Grundlagen der visuellen Odometrie vorgestellt sowie ein Einblick in benachbarte Domänen aus diesem Bereich gegeben.
Das visuelle Odometrie-Verfahren besteht aus einem visuellen Schrittzähler und einem visuellen Kompass. Es wurden zwei Ansätze eines visuellen Schrittzählers vorgestellt, die auf Basis der Veränderung der Attribute eines SURF-Features markante Muster eines Schrittes erkennen. Dabei verwendet der erste Ansatz die relative Veränderung der Bildkoordinaten eines SURF-Features und der zweite Ansatz die relative Veränderung des *Orientation*-Attributes eines SURF-Features. Es wurde der zweite Ansatz für das Verfahren der visuellen Odometrie verwendet, da dieser sich robuster gegenüber Störfaktoren verhält. Außerdem wurde ein visueller Kompass vorgestellt, der auf Basis der relativen Veränderung der Bildkoordinaten entlang der horizontalen Bildebene eine Bewegungsrichtung bestimmen kann. Sowohl der Schrittzähler als auch der visuelle Kompass wurden anschließend evaluiert. Es wurde gezeigt, dass der visuelle Schrittzähler im Vergleich zu anderen herkömmlichen Schrittzählern eine ähnlich gute Genauigkeit aufweisen kann und selbst bei längeren Strecken wenig Fehler erzeugt. Der visuelle Kompass wurde mit dem Schrittzähler gemeinsam evaluiert, indem vorgegebene Pfade durchlaufen und rekonstruiert werden mussten. Der Schrittzähler hat sich wie erwartet konstant gut verhalten, wohingegen der Kompass zu größeren Messfehlern bei bereits kleineren Störfaktoren führte. Trotzdem sind die Ergebnisse insgesamt für das visuelle Odometrie-Verfahren überdurchschnittlich gut.
Um den visuellen Kompass robuster gegenüber kleineren Störfaktoren zu machen, wird eine Integration der Aktivitätserkennung empfohlen. Dadurch wird der visuelle Kompass nur aktiviert, sobald eine Aktivität wie *Kurven gehen* erkannt wird.

Anschließend wurden drei Ansätze einer Aktivitätserkennung vorgestellt, die auf Basis der Attribute eines SURF-Features verschiedene Aktivitäten erkennen können. Dabei verwendet der erste Ansatz zwei Grenzwerte für das $\Delta Scale$-Attribut, um die drei klassischen Aktivitäten *Laufen*, *Gehen* und *Stehen* zu unterscheiden. Der zweite Ansatz verwendet die Schrittfrequenz des visuellen Schrittzählers und teilt anhand der Frequenz die Aktivitäten in drei Klassen ein. Der dritte Ansatz unterscheidet die Aktivitäten anhand von Klassifikatoren, die durch Verfahren des maschinellen Lernens bestimmt werden. Es wurden zusätzlich zu den klassischen Aktivitäten weitere betrachtet. Die Evaluierung hat gezeigt, dass die Verfahren der *Support Vector Machine*, der *Nächste-Nachbarn-Klassifikation* und des *Random Forest-Klassifikator* als geeignete Verfahren zur Entwicklung von Klassifikatoren verwendet werden können. Eine Aktivität wird innerhalb eines zwei Sekunden Fensters festgestellt. Zudem wurden die Aktivitäten *Kurven gehen* und *auf der Stelle drehen* als weitere sinnvolle Aktivitäten hinzugefügt. Der Vergleich aller drei Ansätze hat gezeigt, dass sich eine Aktivitätserkennung anhand von Klassifikatoren am besten dafür eignet und eine Erkennungsrate von bis zu 93,8% erzielt.
Die aufgeführten Verfahren der visuellen Odometrie und der visuellen Aktivitätserkennung zeigen, dass sich der Einsatz solcher Verfahren zur Positionsbestimmung und Kontexterkennung lohnen kann. Trotzdem muss z.B. das Verfahren der visuellen Odometrie verbessert werden. Insbesondere der visuelle Kompass muss erweitert werden, um Kurven robuster zu erkennen und die Veränderungswinkel genauer berechnen zu können. Auch das Verfahren der visuellen Aktivitätserkennung muss um weitere Aktivitäten, wie z.B. Fahradfahren und Springen, erweitert werden. Aber auch die Verfahren des maschinellen Lernens haben gezeigt, dass sie nicht alle Aktivitäten robust unterscheiden können. Hierfür ist der Einsatz eines neuronalen Netzwerks vielversprechender und für zukünftige Arbeiten zielführender.

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit stand die Entwicklung visueller Verfahren für ortsbezogene Dienste im Mittelpunkt. Trotz der Möglichkeiten zur Ortsbestimmung und Positionierung mangelt es immer noch an Verfahren, die es erlauben, eine genaue Positionierung durchgehend an allen Orten zu ermöglichen. Motiviert durch die hohe Verbreitung von Kamerasensoren in Smartphones, Tablets und Wearables, die geringen Produktionskosten, die autarke Funktionsweise und die rasante Entwicklung im Bereich der Bildverarbeitung wurde auf die Entwicklung visueller Verfahren gesetzt. Diese Verfahren sollen an Orten eingesetzt werden, bei denen eine Positionierung mit bestehenden Verfahren schwierig oder unmöglich ist.

Um die Herausforderungen der Positionsbestimmung für ortsbezogene Dienste zu lösen, wurde in der vorliegenden Arbeit ein effizientes und mehrstufiges Bildvergleichsverfahren sowie zwei darauf aufbauende visuelle Positionierungsverfahren vorgestellt. Alle Verfahren verwenden zur Bildrepräsentation markante Merkmale (Features) eines Bildes, die anhand des SURF-Algorithmus extrahiert werden.

Als erster Ansatz wurde ein mehrstufiger Vergleichsprozess vorgestellt, der ein fotografiertes Bild aus einer großen Bilddatenbank in kürzester Zeit effizient und eindeutig findet. Dabei wird ähnlich wie bei einem Siebverfahren die Suchmethode feiner und die Anzahl an möglichen Vergleichskandidaten weniger. Ziel war es, ein Verfahren zu entwickeln, das fotografierte Bilder von Objekten oder ganzen Standorten in kürzester Zeit eindeutig wiedererkennt. Insgesamt besteht der mehrstufige Vergleichsprozess aus drei Stufen. Die erste Stufe vergleicht ein Bild anhand seiner quantisierten Features, die zweite Stufe vergleicht ein Bild anhand seiner SURF-Features und die dritte Stufe vergleicht anhand der Kontur-Punkte, die durch die SURF-Features abgebildet werden. Mit jeder weiteren Stufe wird der Vergleichsprozess allgemein zeitaufwendiger, aber dafür genauer und weniger fehleranfällig. Das Verfahren wurde in das SURFLogo System, das als alternativer Ansatz zum klassischen *Mobile Tagging* entwickelt wurde, integriert und damit evaluiert. Die Ergebnisse der Evaluierung haben gezeigt, dass der mehrstufige Vergleichsprozess ein fotografiertes Bild aus einer großen Bilddatenbank schnell und eindeutig wiedererkennen kann. Das Verfahren hat auch gezeigt, dass unter Verwendung aller Stufen des Vergleichsprozesses eine Eindeutigkeit erreicht werden kann, sodass falsch-positive Ergebnisse ausgeschlossen werden können. Der mehrstufige Ver-

gleichsprozess hat sich als sehr gutes Vergleichswerkzeug erwiesen, sodass es in den zwei Verfahren der Positionsbestimmung aus der vorliegenden Arbeit in Teilen eingesetzt wurde.

Als zweiter Ansatz wurden drei Erweiterungen für das bereits bestehende Positionierungssystem *MoVIPS* entwickelt. *MoVIPS* liefert die aktuelle Position einer Person, indem es ein fotografiertes Standortbild mit einer Bilderdatenbank, bestehend aus abgebildeten Standorten, vergleicht und eine hinterlegte Position zurückgibt. Zwei der Erweiterungen sollten den Vergleich auf der Bilderdatenbank beschleunigen. Dabei setzt die erste Erweiterung die Kompass-, Gyroskop- und Beschleunigungssensordaten eines Smartphones ein, um die Suche auf der Datenbank vorzufiltern. Die zweite Erweiterung integriert den mehrstufigen Vergleichsprozess und erweitert die zweite Stufe um ein *Reranking*-Verfahren, welches die Suche nochmals beschleunigt. Die dritte Erweiterung verbessert das ursprüngliche Positionskorrekturverfahren aus *MoVIPS*, indem es die Eigenschaft der Rotationsinvarianz integriert. Alle drei Erweiterungen haben gezeigt, dass das Positionierungssystem *MoVIPS* durch ihre Integration effizienter und genauer funktioniert. Durch die Erweiterungen wird das Positionierungssystem *MoVIPS* für einen Echtzeitbetrieb nutzbar gemacht. Der Nutzen für ortsbezogene Dienste ergibt sich daraus, dass eine Positionierung innerhalb von Gebäuden durch ein visuelles Verfahren ermöglicht wird.

Als dritter Ansatz wurde ein visuelles Odometrie-Verfahren vorgestellt, das anhand von visuellen Features einer Videosequenz, die aus der Ego-Perspektive einer Person aufgezeichnet wird, das Bewegungsmuster analysiert und daraus eine gegangene Trajektorie rekonstruiert. Anhand der Trajektorie kann anschließend die relative Position einer Person bestimmt werden. Das visuelle Odometrie-Verfahren besteht aus einem visuellen Schrittzähler und einem visuellen Kompass. Der Schrittzähler nutzt dabei die Eigenschaften der Feature-Attribute, um Schrittmuster zu erkennen. Der Kompass verwendet dagegen die relative Veränderung der Bildkoordinaten eines Features entlang der horizontalen Bildachse, um eine Bewegungsrichtung abzuleiten. Das visuelle Odometrie-Verfahren wurde evaluiert und hat gezeigt, dass eine längere Trajektorie sinnvoll rekonstruiert werden kann. Der Schrittzähler wurde mit bestehenden Systemen verglichen und erzielte ähnlich gute Ergebnisse. Der visuelle Kompass hat noch einige Schwierigkeiten und leidet an einer gewissen Fehleranfälligkeit in Situationen, in denen Störfaktoren, wie z.B. durchlaufende Passenten oder fahrende Fahrzeuge, auftreten. Neben dem Verfahren der visuellen Odometrie wurde zusätzlich ein Verfahren zur Aktivitätserkennung entwickelt. Dafür wurden drei Varianten vorgestellt. Die erste Variante nutzt Wertebereiche der Feature-Attribute ($\Delta Scale_F$), um klassische Aktivitäten wie *Laufen*, *Gehen* und *Stehen* zu unterscheiden. Die zweite Varianten nutzt die Schrittfrequenz des visuellen Schrittzählers und teilt anhand der Frequenz die Aktivitäten in drei Klassen ein. Die dritte Variante nutzt Klassifikatoren um Aktivitäten zu bestimmen. Neben den klassischen Aktivitäten wurden

weitere untersucht. Dabei sind die Aktivitäten wie *Kurven gehen* und *auf der Stelle drehen* als sinnvoll erachtet worden, da diese für die Verbesserung des visuellen Kompasses genutzt werden können. Alle drei Varianten wurden evaluiert, wobei sich die dritte Variante mit einer Erkennungsrate von 93% als beste erwiesen hat.

Die vorgestellten Verfahren, insbesondere die letzten beiden Ansätze, eigenen sich sehr gut für die Positionsbestimmung und können damit für ortsbezogene Dienste zusätzlich zu bestehenden Systemen verwendet werden. Der mehrstufigen Vergleichsprozess hat sich bereits im Positionierungssystem *MoVIPS* bemerkbar gemacht und das System für größere Standortdatenbanken nutzbar gemacht.

Trotzdem sollte der mehrstufige Vergleichsprozess erweitert und durch andere Stufen ergänzt werden. So ist das Verfahren farbenblind und kann ausschließlich anhand von markanten Mustern und Konturen ein Bild vergleichen. Daher wäre z.B. der Einsatz einer Vorabsortierung anhand eines Farbhistogramms denkbar und sinnvoll.

Die Erweiterungen aus dem zweiten Ansatz machen *MoVIPS* effizienter und genauer, trotzdem gibt es noch zahlreiche weitere Erweiterungsmöglichkeiten, die das System noch effizienter und besser machen können. So könnte die erste Erweiterung durch das visuelle Odometrie-Verfahren aus Abschnitt 5.4 vollständig ersetzt werden. Dadurch würde das ganze Verfahren ausschließlich auf den SURF-Features basieren. Auch die dritte Erweiterung lässt sich noch verbessern, indem das Positionskorrekturverfahren vollständig auf Berechnung der perspektivischen Transformation umgestellt wird. Dadurch würde die Korrektur der Position nochmals genauer berechnet werden.

Auch das visuelle Odometrie-Verfahren ist noch in einem frühen Stadium der Entwicklung und sollte daher erweitert werden. Der Schrittzähler hat bereits gezeigt, dass er sehr genau funktioniert, jedoch hat der visuelle Kompass immer noch große Schwierigkeiten mit Störfaktoren umzugehen. Daher ist eine zukünftige Verbesserung des visuellen Kompasses notwendig. Eine Erweiterung durch die Aktivitätserkennung könnte das Problem in Teilen lösen, indem es z.B. das *Gehen von Kurven* vorab erkennt und dadurch den visuellen Kompass aktiviert. Damit könnten leichte Drifts, die gerade in monotonen langen Gängen entstehen, verhindert werden. Auch die Aktivitätserkennung lässt sich noch auf weitere sinnvolle Aktivitäten hin untersuchen, sodass vollständige Kontexte allein durch die Attribute der visuellen Features erkannt werden können.

Insgesamt lassen sich die vorgestellten Verfahren sinnvoll kombinieren und einsetzen. Sie bieten die Möglichkeit, allein anhand von visuellen Informationen eine Lage- und Positionsbestimmung sowie eine Aktivitätserkennung zur Verfügung zu stellen.

Es bleibt weiterhin interessant, wie sich visuelle Verfahren zukünftig bemerkbar machen und eingesetzt werden. Schon heute gibt es zahlreiche Beispiele für den erfolgreichen Einsatz visueller Verfahren in unserem Alltag. So wer-

den zahlreiche Kamerasensoren und visuelle Verfahren in moderne Automobile[1] eingesetzt, um Teilbereiche des Fahrens autonom und sicher zu gestalten. Auch der Erfolg von *Head-Mounted Displays*, wie z.B. der *Oculus Rift*[2] und der *HTC Vive*[3], gibt den visuellen Verfahren eine immer größer werdende Bedeutung. Die Displays werden in erster Linie zum Abbilden virtueller Welten eingesetzt, jedoch wechselt der Fokus immer mehr auf das Erweitern unserer Realität durch virtuelle Objekte. Hierfür benötigt man robuste Verfahren, die die Eigenbewegung, die Position und den aktuellen Ort sowie den vollständigen Kontext ermitteln können, um möglichst realitätsgetreu virtuelle Objekte in unsere Realität zu integrieren. Die vorgestellten visuellen Verfahren können dazu beitragen, diese Systeme zu unterstützten und zu erweitern. Jedoch werden auch diese visuellen Verfahren an ihre Grenzen kommen und nicht allein ausreichen um diese Anforderungen vollständig zu erfüllen. Trotz dieser Möglichkeiten, die visuelle Sensoren und visuelle Verfahren mit sich bringen, ist ein robustes Verfahren und System erst durch den Einsatz unterschiedlicher Sensoren geben. Daher kann nur durch eine Informationsfusion, welche Daten aus unterschiedlichen Sensoren vereint, eine präzise Aussage über die Position, Aktivität und den Kontext einer Person geben werden.

[1] http://www.bmw.de/de/topics/faszination-bmw/connecteddrive/hochautomatisiertes-fahren.html
[2] https://www3.oculus.com/en-us/rift/
[3] http://www.vive.com/de/

Literaturverzeichnis

[1] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pp. 225–234, IEEE, 2007.

[2] G. Klein and D. Murray, "Parallel tracking and mapping on a camera phone," in *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pp. 83–86, IEEE, 2009.

[3] S. Z. Online, "Bayerische polizei filmt einsätze mit bodycams." http://www.sueddeutsche.de/bayern/pilotversuch-bayerische-polizei-filmt-einsaetze-mit-bodycams-1.2754559, 2015. letzter Abruf 2.8.2016.

[4] J. Schiller and A. Voisard, *Location-based services*. Elsevier, 2004.

[5] A. Küpper, *Location-based services: fundamentals and operation*. John Wiley & Sons, 2005.

[6] I. A. Junglas and R. T. Watson, "Location-based services," *Communications of the ACM*, vol. 51, no. 3, pp. 65–69, 2008.

[7] E. Kovacs and O. Schramm, "Geolocation of mobile devices," Apr. 1 2003. US Patent 6,542,819.

[8] A. El-Rabbany, *Introduction to GPS: The Global Positioning System*. Artech House mobile communications series, Artech House, 2002.

[9] N. Deblauwe, *GSM-based Positioning: Techniques and Applications*. ASP, 2008.

[10] R. Chen, *Ubiquitous Positioning and Mobile Location-Based Services in Smart Phones*. Premier reference source, Information Science Reference, 2012.

[11] B. Ozdenizci, K. Ok, V. Coskun, and M. N. Aydin, "Development of an indoor navigation system using nfc technology," in *2011 Fourth International Conference on Information and Computing*, pp. 11–14, IEEE, 2011.

[12] J. Yang, Z. Wang, and X. Zhang, "An ibeacon-based indoor positioning systems for hospitals," *International Journal of Smart Home*, vol. 9, no. 7, pp. 161–168, 2015.

[13] C. Marouane and A. Ebert, "Surflogo - mobile tagging with app icons," in *7th EAI International Conference on Mobile Computing, Applications and Services (MobiCASE 2015)*, EAI, 2015.

[14] C. Marouane, M. Maier, S. Feld, and M. Werner, "Visual positioning systems - an extension to movips," in *5th International Conference on Indoor Positioning and Indoor Navigation (IPIN 2014)*, pp. 95–104, 2014.

[15] C. Marouane and A. Ebert, "Enabling pedometers on basis of visual feature point conversion," in *1st GI Expert Talk on Localization*, pp. 33–35, 2015.

[16] C. Marouane, A. Ebert, C. Linnhoff-Popien, and M. Christl, "Step and activity detection based on the orientation and scale attributes of the surf algorithm," in *7th International Conference on Indoor Positioning and Indoor Navigation (IPIN 2016)*, pp. 1–8, 2016.

[17] C. Marouane, M. Maier, A. Leupold, and C. Linnhoff-Popien, "Visual odometry using motion vectors from visual feature points," in *7th International Conference on Indoor Positioning and Indoor Navigation (IPIN 2016)*, pp. 1–8, 2016.

[18] C. Marouane, R. Gutschale, and C. Linnhoff-Popien, "Visual odometry for pedestrians based on orientation attributes of surf," in *SAI Intelligent Systems Conference 2016*, pp. 727–736, 2016.

[19] K. D. Tönnies, *Einführuing in die Bildverarbeitung*. Pearson Studium, 2005.

[20] D. G. R. Bradski and A. Kaehler, *Learning Opencv, 1st Edition*. O'Reilly Media, Inc., first ed., 2008.

[21] S. E. Umbaugh, *Digital Image Processing and Analysis: Human and Computer Vision Applications with CVIPtools, Second Edition*. Boca Raton, FL, USA: CRC Press, Inc., 2nd ed., 2010.

[22] N. Welsch and C. C. Liebmann, *Farben: Natur Technik Kunst*, ch. Theorien des Farbensehens, pp. 227–232. Heidelberg: Spektrum Akademischer Verlag, 2012.

[23] A. Nischwitz, M. Fischer, and P. Haberäcker, *Computergrafik und Bildverarbeitung: Alles für Studium und Praxis - Bildverarbeitungswerkzeuge, Beispiel-Software und interaktive Vorlesungen online verfügbar
 (German Edition)*. Vieweg+Teubner Verlag.

[24] G. B. Garcia, O. D. Suarez, J. L. E. Aranda, J. S. Tercero, and I. S. Gracia, *Learning Image Processing with OpenCV*. Packt Publishing - ebooks Account, 2015.

[25] O. Development Team, "Dokumentation des opencv projekts." `http://docs.opencv.org`, Oktober 2014.

[26] L. Ding and A. Goshtasby, "On the canny edge detector," *Pattern Recognition*, vol. 34, pp. 721–725, 2001.

[27] H. Freeman, "Computer processing of line-drawing images," *ACM Comput. Surv.*, vol. 6, no. 1, pp. 57–97, 1974.

[28] P. V. C. Hough, "Machine Analysis Of Bubble Chamber Pictures," in *Proceedings, 2nd International Conference on High-Energy Accelerators and Instrumentation, HEACC 1959*, vol. C590914, pp. 554–558, 1959.

[29] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[30] D. H. Ballard and C. M. Brown, *Computer Vision*. Prentice Hall, 1982.

[31] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[32] K. Kroschel, G. Rigoll, and B. Schuller, "Wiener-filter," in *Statistische Informationstechnik: Signal - und Mustererkennung, Parameter- und Signalschätzung (German Edition)*, pp. 279–317, Springer, 2011.

[33] T. Thormählen, "Grafikprogrammierung 1 - kameras: Perspektivische projektion." `http://www.mathematik.uni-marburg.de/~thormae/lectures/graphics1/graphics_6_1_ger_web.html#11`, Dezember 2014.

[34] O. Schreer, *Stereoanalyse und Bildsynthese*. Springer Berlin Heidelberg, 2005.

[35] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," *Foundations and Trends® in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2008.

[36] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pp. 593–600, IEEE, 1994.

[37] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M. E. Munich, "The vSLAM algorithm for robust localization and mapping," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 24—-29, IEEE, 2005.

[38] H. Kawaji, K. Hatada, T. Yamasaki, and K. Aizawa, "Image-based indoor positioning system: fast image matching using omnidirectional panoramic images," in *Proceedings of the 1st ACM international workshop on Multimodal pervasive video analysis*, pp. 1—-4, 2010.

[39] M. Werner, M. Kessel, and C. Marouane, "Indoor positioning using smartphone camera," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN 2011)*, pp. 1–6, IEEE, 2011.

[40] D. L. Milgram, "Computer methods for creating photomosaics," *IEEE Trans. Comput.*, vol. 24, no. 11, pp. 1113–1119, 1975.

[41] H. Freeman, "A review of relevant problems in the processing of line-drawing data," *Automatic Interpretation and Classification of Images*, pp. 155–174, 1969.

[42] W. Rutkowski and A. Rosenfeld, "A comparison of corner-detection techniques for chain-coded curves, university of maryland," tech. rep., TR-623, College Park, Maryland, 1977.

[43] D. Langridge, "Curve encoding and the detection of discontinuities," *Computer Graphics and Image Processing*, vol. 20, no. 1, pp. 58–71, 1982.

[44] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 12, pp. 1376–1381, 1998.

[45] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1st ed., 1992.

[46] J. Cooper, S. Venkatesh, and L. Kitchen, "Early jump-out corner detectors," in *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pp. 688–689, IEEE, 1991.

[47] P. R. Beaudet, "Rotationally invariant image operators," in *Proceedings of the 4th International Joint Conference on Pattern Recognition*, (Kyoto, Japan), pp. 579–583, 1978.

[48] L. Dreschler and H.-H. Nagel, "Volumetric model and 3d trajectory of a moving car derived from monocular tv frame sequences of a street scene," *Computer Graphics and Image Processing*, vol. 20, no. 3, pp. 199–228, 1982.

[49] O. A. Zuniga and R. M. Haralick, "Corner detection using the facet model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 30–37, 1983.

[50] H. P. Morevec, "Towards automatic visual obstacle avoidance," in *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'77, (San Francisco, CA, USA), pp. 584–584, Morgan Kaufmann Publishers Inc., 1977.

[51] H. P. Moravec, "Visual mapping by a robot rover," in *Proceedings of the 6th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'79, (San Francisco, CA, USA), pp. 598–600, Morgan Kaufmann Publishers Inc., 1979.

[52] J. Cooper, S. Venkatesh, and L. Kitchen, "The dissimilarity corner detector," in *ICAR 1991: Proceedings of the 5th International Conference on Advanced Robotics: Robots in Unstructured Environments*, pp. 1377–1382, IEEE, 1991.

[53] C. Harris and M. Stephens, "A combined corner and edge detector.," in *Alvey vision conference*, vol. 15, p. 50, Citeseer, 1988.

[54] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 11, pp. 1254–1259, 1998.

[55] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature reviews neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[56] R. P. Würtz and T. Lourens, "Corner detection in color images through a multiscale combination of end-stopped cortical cells," *Image and vision computing*, vol. 18, no. 6, pp. 531–541, 2000.

[57] V. Gouet, P. Montesinos, R. Deriche, and D. Pelé, "Evaluation de détecteurs de points d'intérêt pour la couleur," in *12eme Congres Francophone AFRIF–AFIA de Reconnaissance des Formes et Intelligence Artificielle*, pp. 257–266, 2000.

[58] J. L. Crowley and A. C. Parker, "A representation for shape based on peaks and ridges in the difference of low-pass transform," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 2, pp. 156–170, 1984.

[59] T. Lindeberg, "Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention," *International Journal of Computer Vision*, vol. 11, no. 3, pp. 283–318, 1993.

[60] T. Lindeberg, "Feature detection with automatic scale selection," *International journal of computer vision*, vol. 30, no. 2, pp. 79–116, 1998.

[61] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2, pp. 1150–1157, Ieee, 1999.

[62] T. Tuytelaars and L. Van Gool, "Content-based image retrieval based on local affinely invariant regions," in *Visual Information and Information Systems*, pp. 493–500, Springer, 1999.

[63] T. Tuytelaars and L. V. Gool, "Wide baseline stereo matching based on local, affinely invariant regions," in *In Proc. BMVC*, pp. 412–425, 2000.

[64] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.

[65] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[66] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," in *In NIPS*, pp. 831–837, 2000.

[67] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, (Washington, DC, USA), pp. 506–513, IEEE Computer Society, 2004.

[68] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer vision–ECCV 2006*, pp. 404–417, Springer, 2006.

[69] A. Baumberg, "Reliable feature matching across widely separated views," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 1, pp. 774–781, IEEE, 2000.

[70] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1615–1630, 2005.

[71] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, pp. 1150–1157 vol.2, 1999.

[72] C. Evans, "Notes on the opensurf library," Tech. Rep. CSTR-09-001, University of Bristol, January 2009.

[73] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[74] M. Brown and D. Lowe, "Invariant features from interest point groups," in *Proc. BMVC*, pp. 1–10, 2002. doi:10.5244/C.16.23.

[75] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *Computer Vision–ECCV 2008*, pp. 102–115, Springer, 2008.

[76] G. Bradski, "Open source computer vision library," *Dr. Dobb's Journal of Software Tools*, 2000.

[77] D. Fantacci and A. Martini, "Java implementation on speeded up robust features surf." `http://processingsurf.altervista.org/`, March 2010.

[78] H. Bay and L. V. Gool, "http://www.vision.ee.ethz.ch/ surf/download.html." `http://www.vision.ee.ethz.ch/~surf/download.html`, 2008.

[79] A. Schulz, F. Jung, S. Hartte, D. Trick, C. Wojek, J. Schindler, Konrad Ackermann, and M. Goesele, "Cuda surf - a real-time implementation for surf." `https://www.mpi-inf.mpg.de/departments/computer-vision-and-multimodal-computing/research/object-recognition-and-scene-understanding/cuda-surf-a-real-time-implementation-for-surf/`, 2008.

[80] NVIDIA Corporation, "NVIDIA CUDA C programming guide," 2010. Version 3.2.

[81] A. Schierwagen, "Bildverstehen in der ki: Konzepte und probleme," in *Vom Realismus der Bilder. Interdisziplinäre Forschungen zur Semantik bildhafter Darstellungsformen.*, Scriptum Verlag, Magdeburg, 1999.

[82] T. Brosnan and D.-W. Sun, "Improving quality inspection of food products by computer vision—-a review," *Journal of Food Engineering*, vol. 61, no. 1, pp. 3–16, 2004.

[83] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 1–54, 2015.

[84] t3n, "Samsung smart tv: Bedienung per sprache und gesten [ces 2012]." http://t3n.de/news/samsung-smart-tv-bedienung-358579/, 2012. letzter Abruf: 2.8.2016.

[85] S. PlayStation, "Eyetoy: Play." `http://de.playstation.com/ps2/games/detail/item30943/EyeToy-Play/`, 2003. letzter Abruf: 2.8.2016.

[86] H. Marwitz, "Stand und perspektiven der assistenz- und führungssysteme moderner lkw bei DaimlerChrysler," *ATZ - Automobiltechnische Zeitschrift*, vol. 104, no. 5, pp. 476–481, 2002.

[87] S. Kammel, "Autonomes fahren," in *Handbuch Fahrerassistenzsysteme*, pp. 657–663, Springer Science + Business Media, 2009.

[88] P. Pfeffer and M. Harrer, "Überblick – fahrerassistenzsystemfunktionen," in *Lenkungshandbuch*, pp. 457–470, Springer Science + Business Media, 2011.

[89] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, "A survey of appearance models in visual object tracking," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, pp. 58:1–58:48, 2013.

[90] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[91] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 174–188, 2002.

[92] L. Ellis, N. Dowson, J. Matas, and R. Bowden, "Linear regression and adaptive appearance models for fast simultaneous modelling and tracking," *International journal of computer vision*, vol. 95, no. 2, pp. 154–179, 2011.

[93] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Second Edition*. The MIT Press, 2001.

[94] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1151–1163, 2002.

[95] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," in *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pp. 601–608, IEEE, 1998.

[96] T. Zhao and R. Nevatia, "Tracking multiple humans in crowded environment," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2, pp. II–406–II–413 Vol.2, 2004.

[97] D. M. Gavrila, "The visual analysis of human movement: A survey," *Computer vision and image understanding*, vol. 73, no. 1, pp. 82–98, 1999.

[98] S. Corazza, L. Muendermann, A. Chaudhari, T. Demattio, C. Cobelli, and T. P. Andriacchi, "A markerless motion capture system to study musculoskeletal biomechanics: visual hull and simulated annealing approach," *Annals of biomedical engineering*, vol. 34, no. 6, pp. 1019–1029, 2006.

[99] J. C. Chan, H. Leung, J. K. Tang, and T. Komura, "A virtual reality dance training system using motion capture technology," *Learning Technologies, IEEE Transactions on*, vol. 4, no. 2, pp. 187–195, 2011.

[100] M. J. Swain and D. H. Ballard, "Color indexing," *Int J Comput Vision*, vol. 7, no. 1, pp. 11–32, 1991.

[101] H. J. Wolfson and I. Rigoutsos, "Geometric hashing: An overview," *Computing in Science & Engineering*, no. 4, pp. 10–21, 1997.

[102] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[103] F. Camastra and A. Vinciarelli, "Automatic face recognition," in *Advanced Information and Knowledge Processing*, pp. 421–448, Springer Science and Business Media, 2015.

[104] C. Knop, "Biometrische gesichtserkennung - mehr sicherheit durch das digitale abtasten." http://www.faz.net/aktuell/wirtschaft/biometrische-gesichtserkennung-mehr-sicherheit-durch-das-digitale-abtasten-1385644.html, 2006. letzter Abruf: 2.8.2016.

[105] F. B. Brandenburg, "Biometrische gesichtserkennung an berliner flughäfen." http://www.berlin-airport.de/de/presse/pressemitteilungen/2003/2003-01-29-gesichtserkennung/, 2003. letzter Abruf: 2.8.2016.

[106] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37–47, 2002.

[107] N. Uke and R. Thool, "Moving vehicle detection for measuring traffic count using opencv," *Journal of Automation and Control Engineering*, vol. 1, no. 4, pp. 349–352, 2013.

[108] S. Messelodi, C. M. Modena, and M. Zanin, "A computer vision system for the detection and classification of vehicles at urban road intersections," *Pattern analysis and applications*, vol. 8, no. 1-2, pp. 17–31, 2005.

[109] A. Chaturvedi and N. Sethi, "Automatic license plate recognition system using surf features and rbf neural network," *International Journal of Computer Applications*, vol. 70, no. 27, pp. 37–41, 2013.

[110] N. Buch, J. Orwell, and S. A. Velastin, "Detection and classification of vehicles for urban traffic scenes," in *Visual Information Engineering, 2008. VIE 2008. 5th International Conference on*, pp. 182–187, 2008.

[111] D. M. Jang and M. Turk, "Car-rec: A real time car recognition system," in *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pp. 599–605, IEEE, 2011.

[112] C. Marouane, L. Schauer, and P. Bauer, "Classification of vehicle types in car parks using computer vision techniques," in *12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp. 1–9, 2015.

[113] G. Schroth, R. Huitl, D. Chen, M. Abu-Alqumsan, A. Al-Nuaimi, and E. Steinbach, "Mobile visual location recognition," *Signal Processing Magazine, IEEE*, vol. 28, no. 4, pp. 77–89, 2011.

[114] H. Kang, A. A. Efros, M. Hebert, and T. Kanade, "Image matching in large scale indoor environment," in *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pp. 33–40, IEEE, 2009.

[115] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the mars exploration rovers," *Journal of Field Robotics*, vol. 24, no. 3, pp. 169–186, 2007.

[116] C.-Y. Chen, J.-H. Zhang, and B. R. Chang, "Visual odometry with improved adaptive feature tracking," in *Proceedings of the 29th International Conference on Image and Vision Computing New Zealand*, pp. 7–12, ACM, 2014.

[117] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, pp. I–652, IEEE, 2004.

[118] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad, "An overview to visual odometry and visual SLAM: Applications to mobile robotics," *Intell Ind Syst*, vol. 1, no. 4, pp. 289–311, 2015.

[119] N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M. E. Munich, "The vslam algorithm for robust localization and mapping," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 24–29, IEEE, 2005.

[120] J. Shimamura, M. Morimoto, and H. Koike, "Robust vslam for dynamic scenes.," in *MVA*, pp. 344–347, 2011.

[121] R. T. Azuma, "A survey of augmented reality," *Presence: Teleoperators and virtual environments*, vol. 6, no. 4, pp. 355–385, 1997.

[122] S. Honey and K. Milnes, "The augmented reality america's cup." http://spectrum.ieee.org/consumer-electronics/audiovideo/the-augmented-reality-americas-cup, 2013. letzter Abruf: 2.8.2016.

[123] IKEA, "2014 ikea catalogue comes to life with augmented reality." `http://www.ikea.com/ca/en/about_ikea/newsitem/2014catalogue/`, 2013. letzter Abruf: 2.8.2016.

[124] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[125] Bosch, "mydriveassist - verpasse mit deiner unterstützung kein tempolimit mehr." http://iphone.bosch.com/mydriveassist/, 2015. letzter Abruf: 2.8.2016.

[126] F. Schaffalitzky and A. Zisserman, "Automated scene matching in movies," in *Image and Video Retrieval*, pp. 186–197, Springer, 2002.

[127] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1, pp. 525–531, IEEE, 2001.

[128] D. Omercevic, O. Drbohlav, and A. Leonardis, "High-dimensional feature matching: employing the concept of meaningful nearest neighbors," in *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, IEEE, 2007.

[129] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 1470–1477, IEEE, 2003.

[130] J. Macqueen, "Some methods for classification and analysis of multivariate observations," in *In 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

[131] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

[132] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the hausdorff distance," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 9, pp. 850–863, 1993.

[133] M. Hegen, *Mobile Tagging: Potenziale von QR-Codes im Mobile Business*. Diplomica Verlag, 2010.

[134] H. Al-Khalifa, "Utilizing qr code and mobile phones for blinds and visually impaired people," in *Computers Helping People with Special Needs* (K. Miesenberger, J. Klaus, W. Zagler, and A. Karshmer, eds.), vol. 5105 of *Lecture Notes in Computer Science*, pp. 1065–1069, Springer Berlin Heidelberg, 2008.

[135] M. Canadi, W. Höpken, and M. Fuchs, "Application of qr codes in online travel distribution," in *Information and Communication Technologies in Tourism 2010* (U. Gretzel, R. Law, and M. Fuchs, eds.), pp. 137–148, Springer Vienna, 2010.

[136] A. Walsh, "Blurring the boundaries between our physical and electronic libraries," *The Electronic Library*, vol. 29, no. 4, pp. 429–437, 2011.

[137] E. Costa-Montenegro, F. J. González-Castaño, D. Conde-Lagoa, A. B. Barragáns-Martínez, P. S. Rodríguez-Hernández, and F. Gil-Castiñeira, "Qr-maps: An efficient tool for indoor user location based on qr-codes and google maps," in *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, pp. 928–932, IEEE, 2011.

[138] R. Ashford, "Qr codes and academic libraries reaching mobile users," *College & Research Libraries News*, vol. 71, no. 10, pp. 526–530, 2010.

[139] G. Madlmayr and J. Scharinger, "Neue dimension von mobilen tourismusanwendungen durch near field communication-technologie," in *mTourism* (R. Egger and M. Jooss, eds.), pp. 75–88, Gabler, 2010.

[140] M. Darianian and M. Michael, "Smart home mobile rfid-based internet-of-things systems and services," in *Advanced Computer Theory and Engineering, 2008. ICACTE '08. International Conference on*, pp. 116–120, 2008.

[141] G. W.-H. Tan, K.-B. Ooi, S.-C. Chong, and T.-S. Hew, "{NFC} mobile credit card: The next frontier of mobile payment?," *Telematics and Informatics*, vol. 31, no. 2, pp. 292 – 307, 2014.

[142] M. Gast, *Building Applications with IBeacon: Proximity and Location Services with Bluetooth Low Energy*. O'Reilly Media, 2014.

[143] G. Developers, "Google beacons." https://developers.google.com/beacons/, 2015. letzter Abruf: 2.8.2016.

[144] Microsoft, "Mircosoft tag - creating custom tags." http://tag.microsoft.com/what-is-tag/custom-tags.aspx, 2011. letzter Abruf: 2.8.2016.

[145] M. V. mbH (MVG), "Mvg fahrinfo münchen." https://www.mvg.de/services/mobile-services/fahrinfo.html, 2016. letzter Abruf: 2.8.2016.

[146] Mozilla, "Mozilla location service." https://location.services.mozilla.com/, 2016. letzter Abruf: 2.8.2016.

[147] OpenWlanMap.org, "Was ist openwifi.su?." http://openwlanmap.org/, 2016. letzter Abruf: 2.8.2016.

[148] F. IIS, "awiloc - die lokalisierungstechnologie für städte und gebäude." http://www.iis.fraunhofer.de/de/ff/lok/tech/feldstaerke/rssi.html, 2016. letzter Abruf: 2.8.2016.

[149] C. M. AB, "Cps - combain positioning service." http://combain.com/, 2016. letzter Abruf: 2.8.2016.

[150] A. D. Forums, "Is it possible to get wifi signal strength in ios 9." `https://forums.developer.apple.com/message/70567#70567`, 2016. letzter Abruf: 2.8.2016.

[151] H. Hile and G. Borriello, "Information overlay for camera phones in indoor environments," in *International Symposium on Location-and Context-Awareness*, pp. 68–84, Springer, 2007.

[152] B. Girod, V. Chandrasekhar, D. M. Chen, N.-M. Cheung, R. Grzeszczuk, Y. Reznik, G. Takacs, S. S. Tsai, and R. Vedantham, "Mobile visual search," *IEEE signal processing magazine*, vol. 28, no. 4, pp. 61–76, 2011.

[153] H. Kawaji, K. Hatada, T. Yamasaki, and K. Aizawa, "Image-based indoor positioning system: fast image matching using omnidirectional panoramic images," in *Proceedings of the 1st ACM international workshop on Multimodal pervasive video analysis*, pp. 1–4, ACM, 2010.

[154] C. Galleguillos and S. Belongie, "Context based object categorization: A critical survey," *Computer Vision and Image Understanding*, vol. 114, no. 6, pp. 712–722, 2010.

[155] A. Collet Romea, M. Martinez Torres, and S. Srinivasa, "The moped framework: Object recognition and pose estimation for manipulation," *International Journal of Robotics Research*, vol. 30, no. 10, pp. 1284 – 1306, 2011.

[156] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin, "Context-based vision system for place and object recognition," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 273–280, IEEE, 2003.

[157] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 6, pp. 1067–1080, 2007.

[158] M. Maier and F. Dorfmeister, "Fine-grained activity recognition of pedestrians travelling by subway," in *5th International Conference on Mobile Computing, Applications and Services (MobiCASE 2013)*, 2013.

[159] O. Chum, J. Philbin, A. Zisserman, *et al.*, "Near duplicate image detection: min-hash and tf-idf weighting.," in *BMVC*, vol. 810, pp. 812–815, 2008.

[160] J. A. B. Link, P. Smith, N. Viol, and K. Wehrle, "Footpath: Accurate mapbased indoor navigation using smartphones," in *International Conference on Indoor Positioning and Indoor Navigation*, pp. 1–8, 2011.

[161] A. Agarwal, C. Jawahar, and P. Narayanan, "A survey of planar homography estimation techniques," *Centre for Visual Information Technology, Tech. Rep. IIIT/TR/2005/12*, 2005.

[162] Y. Cheng, M. Maimone, and L. Matthies, "Visual odometry on the mars exploration rovers," in *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, vol. 1, pp. 903–910 Vol. 1, 2005.

[163] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *Robotics Automation Magazine, IEEE*, vol. 18, no. 4, pp. 80–92, 2011.

[164] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover.," tech. rep., DTIC Document, 1980.

[165] J. Markoff, "Google cars drive themselves, in traffic," *The New York Times*, vol. 10, no. A1, p. 9, 2010.

[166] J. Kelly, S. Saripalli, and G. S. Sukhatme, "Combined visual and inertial navigation for an unmanned aerial vehicle," in *Field and Service Robotics*, pp. 255–264, Springer, 2008.

[167] P. Corke, D. Strelow, and S. Singh, "Omnidirectional visual odometry for a planetary rover," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 4, pp. 4007–4012, IEEE, 2004.

[168] M. Lhuillier, "Automatic scene structure and camera motion using a catadioptric system," *Computer Vision and Image Understanding*, vol. 109, no. 2, pp. 186 – 203, 2008.

[169] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, vol. 23, no. 1, pp. 3–20, 2006.

[170] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *2008*

IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2531–2538, IEEE, 2008.

[171] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[172] M. Irani and P. Anandan, "About direct methods," in *International Workshop on Vision Algorithms*, pp. 267–277, Springer, 1999.

[173] S. Lovegrove, A. J. Davison, and J. Ibanez-Guzmán, "Accurate visual odometry from a rear parking camera," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pp. 788–793, IEEE, 2011.

[174] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*, pp. 2320–2327, IEEE, 2011.

[175] Y. Cheng, M. Maimone, and L. Matthies, "Visual odometry on the mars exploration rovers," in *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, pp. 903–910, IEEE, 2005.

[176] Y. Cheng, M. W. Maimone, and L. Matthies, "Visual odometry on the mars exploration rovers-a tool to ensure accurate driving and science imaging," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 54–62, 2006.

[177] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *J. Field Robot.*, vol. 27, no. 5, pp. 587–608, 2010.

[178] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15–22, IEEE, 2014.

[179] M. Dunbabin, J. Roberts, K. Usher, G. Winstanley, and P. Corke, "A hybrid auv design for shallow water reef navigation," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2105–2110, IEEE, 2005.

[180] D. AG, "6d-vision." http:// www.6d-vision.com/, 2011. letzter Abruf: 2.8.2016.

[181] D. AG, "Dacuda scanner mouse." http://www.dacuda.com/, 2012. letzter Abruf: 2.8.2016.

[182] H. Winner, S. Hakuli, and G. Wolf, *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*. ATZ/MTZ-Fachbuch, Vieweg+Teubner Verlag, 2011.

[183] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "A floor and obstacle height map for 3d navigation of a humanoid robot," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 1066–1071, IEEE, 2005.

[184] A. Hornung, K. M. Wurm, and M. Bennewitz, "Humanoid robot localization in complex indoor environments," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 1690–1695, IEEE, 2010.

[185] H. Mensen, *Handbuch der Luftfahrt (VDI-Buch) (German Edition)*. Springer Vieweg, 2013.

[186] P. Wagenführ, *Navigation in der nordeuropäischen Schifffahrt des Spätmittelalters: Instrumente und Methoden*. Diplomica Verlag, 2015.

[187] D. Gusenbauer, C. Isert, and J. Krösche, "Self-contained indoor positioning on off-the-shelf mobile devices," in *Indoor positioning and indoor navigation (IPIN), 2010 international conference on*, pp. 1–9, IEEE, 2010.

[188] A. R. Pratama, Widyawan, and R. Hidayat, "Smartphone-based pedestrian dead reckoning as an indoor positioning system," in *System Engineering and Technology (ICSET), 2012 International Conference on*, pp. 1–6, Sept 2012.

[189] J. W. Kim, H. J. Jang, D.-H. Hwang, and C. Park, "A Step, Stride and Heading Determination for the Pedestrian Navigation System," *Journal of Global Positioning Systems*, vol. 3, no. 1&2, pp. 273–279, 2004.

[190] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[191] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[192] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. Cardoso, "Preprocessing techniques for context recognition from accelerometer data," *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 645–662, 2010.

[193] A. Brajdic and R. Harle, "Walk detection and step counting on unconstrained smartphones," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 225–234, ACM, 2013.

[194] Z. Fitz-Walter and D. Tjondronegoro, "Simple classification of walking activities using commodity smart phones," in *Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*, pp. 409–412, ACM, 2009.

[195] J.-g. Park, A. Patel, D. Curtis, S. Teller, and J. Ledlie, "Online pose classification and walking speed estimation using handheld devices," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pp. 113–122, ACM, 2012.

[196] D. Gutiérrez-Gómez and J. J. Guerrero, "Scaled monocular slam for walking people," in *Proceedings of the 2013 International Symposium on Wearable Computers*, pp. 9–12, ACM, 2013.

[197] I. A. Kramers-de Quervain, E. Stüssi, and A. Stacoff, "Ganganalyse beim gehen und laufen," *Schweizerische Zeitschrift für Sportmedizin und Sporttraumatologie*, vol. 56, no. 2, pp. 35–42, 2008.