Fakultät Informatik/Mathematik

Hochschule für
Technik und Wirtschaft
Dresden
University of Applied Sciences

Fachbereich Künstliche Intelligenz

AKTUELLE METHODEN DER BACKGROUND SUBTRACTION UND DEREN ANWENDUNG ALS VORVERARBEITUNG EINER GESTÜRZTEN–PERSON–ERKENNUNG

Jan Brose

Studiengang: Angewandte Informatik
Immatrikulationsjahr 2018

MASTER-ARBEIT

zur Erlangung des akademischen Grades

MASTER OF SCIENCE (M. SC.)

Gutachter

M.Sc. Robert Erzgräber

Betreuender Hochschullehrer

Prof. Dr.-Ing. habil. Hans-Joachim Böhme

Eingereicht am: 24. September 2020

In der vorliegenden Arbeit wird bei personenbezogenen Substantiven und Pronomen die Sprachform verwendet, die laut Duden „jemanden“ – eine Person – beschreibt. Aus Respekt wird der Mensch somit stets unabhängig vom biologischen/-sozialen/kulturellen/... Geschlecht als Person angenommen.

Danksagung

Ich möchte allen Danken, die mich auf dem Weg bis hier hin und durch das Masterstudium unterstützt haben. Konkret möchte ich da meinen Eltern, meinem Bruder, Laura, meinen Kommilitonen, meinen Professoren, den wissenschaftlichen Mitarbeitern, sowie der Arbeitsgruppe für Künstliche Intelligenz und Kognitive Robotik der HTW danken.

INHALTSVERZEICHNIS

AKRONYME

CNN Convolutional Neural Network. 17–22, *Glossar:* CNN

fps Bilder pro Sekunde. 11, 14, 20, 23, 24, 39, 47, 48, *Glossar:* fps

GAN Generative Adversarial Network. 17, 22, 24, *Glossar:* GAN

GPE Gestürzte–Personen–Erkennung. 2, 4–6, 8, 12, 21, 24, 31, 39, 40, 49, *Glossar:* GPE

RANSAC Random Sample Consensus. 13, 26, 28–30, 39, 40, *Glossar:* RANSAC

RGB–D RGB–Tiefenbild. 8, 11–18, 20, 24, 27, 31, 39, 40, 47, 48, *Glossar:* RGB–D

RoNisCo Robotic Nightshift Companion. 1, 6–8, 16, *Glossar:* RoNisCo

WL Weaklearner. 11, *Glossar:* WL

GLOSSAR

CNN Bei Convolutional Neural Networks handelt es sich um ein Konzept für neuronale Netze, welche häufig für Bild- und Audiodatenverarbeitung verwendet werden. 17

fps Bilder pro Sekunde (frames per second) gibt an wie viele Bilder in einer Sekunde zu sehen sind, erzeugt oder berechnet werden. 11

GAN Bei Generative Adversarial Networks handelt es sich um ein Konzept für neuronale Netze, welche zum unüberwachten Lernen genutzt werden, dieses besteht aus 2 neuronalen Netzwerken. Einem Generator der Daten erzeugt und einen Diskriminator, der diese klassifiziert. 17

GPE Eine Gestürzte–Personen–Erkennung ist ein Modul, welches erfassen soll, ob sich in einer ihm präsentierten Szene eine gestürzte Person befindet. 2

RANSAC ist die Abkürzung für Random Sample Consensus. Dabei handelt es sich um einen Resampling–Algorithmus zur Schätzung eines Modells innerhalb einer Reihe von Messwerten mit Ausreißern. 13

RGB–D Ein RGB–Tiefenbild (RGB–Depth image) ist ein Bild in welchem jeder Pixelwert die Entfernung dieses Bildpunktes von der Kamera repräsentiert. 8

RoNisCo Ein Assistenzrobotersystem, welches Pflegekräfte bei ihrer Nachtschicht unterstützen soll. 1

WL Ein Weaklearner ist ein Bestandteil eines Adaboost–Algorithmus. Dabei handelt es sich um einen Klassifikator, der für die Aufgabe schlecht performt. Im Adaboost–Algorithmus werden mehrere Weaklearner kombiniert und ihre Ergebnisse gewichtet zu einem zusammengefasst. Die Performance des Adaboost ist besser als die der einzelnen Weaklearner. 11

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

1 EINLEITUNG

Aktuell sind in Deutschland über 3,4 Millionen Menschen auf Pflege angewiesen. Mehr als 800 000 davon werden stationär in Pflegeeinrichtungen betreut.[1] Gleichzeitig wird in der Nachrichtenlandschaft[2] und in der politischen Debatte wiederholt über einen Pflegenotstand, also die Überarbeitung des Personals aufgrund eines hohen Arbeitspensums gesprochen.
Parallel dazu schreitet die Digitalisierung in der Gesellschaft weiter voran. Maschinen, Computer und Roboter werden an vielen Stellen eingesetzt, um Menschen bei ihrer Arbeit zu unterstützen und diese zu erleichtern. So ergibt sich ab einem gewissen Punkt die Frage, ob man auch Assistenzsysteme in der Pflege einsetzen kann. Ein Projekt, welches sich mit dieser Frage befasst, ist das Robotic Nightshift Companion (RoNisCo) Projekt, welches von [Lischke et al., 2017] beschrieben wird.
Dieses Projekt ist Teil der Forschung der Arbeitsgruppe für Künstliche Intelligenz und Kognitive Robotik der HTW–Dresden um Prof. Dr.-Ing. habil. Hans-Joachim Böhme. Dabei werden in Kooperation mit einer lokalen Pflegeeinrichtung Anwendungsmöglichkeiten für Roboter–Assistenzsysteme untersucht, wie dies von [Bahrmann et al., 2020] beschrieben wird. Eine dieser Anwendungsmöglichkeiten ist der bereits erwähnte Nachtwächter.
Dieser soll das Pflegepersonal während der Nachtschicht unterstützen, in welcher sich der Personalmangel besonders dadurch zeigt, das teilweise weniger Pflegekräfte als Stockwerke vorhanden sind. Zusätzlich dazu ist es durch den Grundriss des Gebäudes, welcher in Abbildung 1.1 zu sehen ist, nicht möglich, ein Stockwerk von einem Punkt aus komplett im Blick zu haben. Des Weiteren umfassen die Aufgaben von Pflegekräften nicht nur das Betreuen und Versorgen der Bewohner beziehungsweise Patienten in den entsprechenden Einrichtungen, sondern, wie man in [OAK, 2016] unter dem Punkt Berufsbild der Pflegefachkraft nachlesen kann, auch das Anfertigen von Dokumentationen zu ihrer Arbeit. Gleichzeitig können sie auch nicht auf den Fluren unterwegs sein, während sie sich um einen konkreten Bewohner kümmern. Zu dieser Situation kommt der Faktor, dass in dieser konkreten Einrichtung, ähnlich wie in vielen weiteren Pflegeeinrichtungen in Deutschland[3] auch, viele Bewohner mit verschiedenen Formen von Demenz leben. Symptome dieser Krankheit können eine Orientierungslosigkeit in Zeit und Ort sein, welche beispielsweise dazu führen kann, dass sie Nachts einkaufen gehen wollen und die Einrichtung verlassen. Dies ist in mehrerlei Hinsicht gefährlich für diese Personen. Falls sie auf den Fluren oder im Treppenhaus stürzen, könnten diese Stürze viel zu

[1] **Seite 5**; Pflegestatistik et al., 2018.
[2] Eubel und Heine, 2013; Stalinski, 2017.
[3] Schäufele et al., 2013.

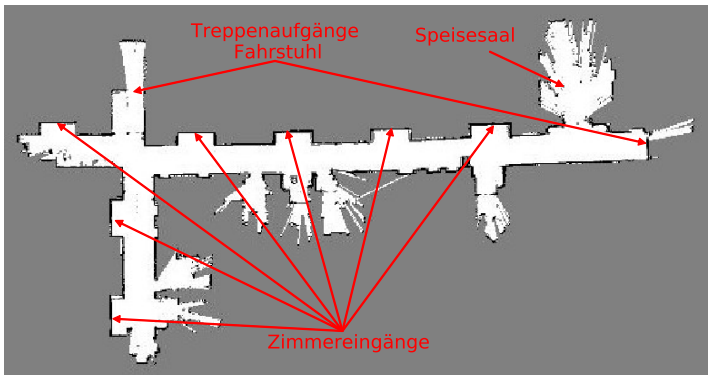

Abbildung 1.1: Grundriss eines Stockwerks in der Pflegeeinrichtung, dargestellt als Karte, die durch den Roboter gelernt wurde

spät bemerkt werden. Andererseits bestehen auch Gefahren für sie, wenn es ihnen gelingt, die Einrichtung zu verlassen. Deshalb sollten beide Szenarien vermieden werden.
Dazu bieten sich wie in [Brose, 2018] dargestellt zwei grundlegende technische Umsetzungen an. Zum einen ein statisches Kamerasystem und zum anderen ein Assistenzroboter, wie er in Abbildung 1.2 gezeigt wird. Wie aus den bisherigen Ausführungen ersichtlich ist, wurde sich für das Assistenzrobotersystem entschieden. Dieses kann die Flure selbstständig überwachen, auf denen aktuell kein Personal sein kann und dieses im Notfall informieren. Dabei bietet es den Vorteil, dass der Roboter eine kritische Situation zusätzlich aus verschiedenen Perspektiven betrachten kann, falls eine zu hohe Unsicherheit besteht. Zusätzlich dazu kann er die Bewohner auch ansprechen und diese somit beruhigen oder zumindest ablenken, bis das Personal vor Ort ist. Dieser Vorteil wird von [Brose, 2018] genauer ausgeführt und behandelt.
Um diese Aufgaben erfüllen zu können, muss das Assistenzsystem einige Fähigkeiten besitzen. Zu diesen gehören sich selbst im Gebäude orientieren, mit dem Pflegepersonal und den Bewohnern kommunizieren und Personen zu erkennen. Letzteres lässt sich in zwei Teile unterteilen. Der eine Teil ist das Erkennen von Personen in natürlichen Köperhaltungen, wie laufend, sitzend und stehend. Während der andere Teil zum Ziel hat, gestürzte Personen erkennen. Dabei besteht die Schwierigkeit darin, dass diese mitunter sehr unnatürliche Körperhaltungen einnehmen, beziehungsweise für klassische Personenerkennungen wichtige Körperteile verdeckt sind. Dies würde die Zuverlässigkeit des Systems an einem sehr kritischen Punkt reduzieren, da gerade diese Personen schnellstmöglich Hilfe benötigen. Aus diesem Grund existiert neben der Personenerkennung für Menschen mit natürlicheren Körperhaltungen eine Gestürzte–Personen–Erkennung (GPE).
Dafür existiert schon ein prototypischer Algorithmus. Allerdings benötigt er eine Vorverar-

Abbildung 1.2: Roboter „Anna Constantia“ – eine mobile Plattform vom Typ Scitos G5, hergestellt von MetraLabs

beitung in Form einer Background Subtraction der Sensordaten, die das Kamera–Modul des Roboters liefert, damit er mit diesen effizient arbeiten kann. Außerdem muss sich diese auch für die durch die Hardware und das Szenario gesetzten Rahmenbedingungen eignen. Die Umsetzung dieser Vorverarbeitung und deren Einbindung in den Prototypen soll das Objekt dieser Arbeit sein.
Dafür wird im weiteren Verlauf der Arbeit auch noch einmal ausführlicher auf den Klassifikator, die limitierenden Faktoren und den aktuellen technischen Stand im Bereich Background Subtraction eingegangen.

2 ZIELSTELLUNG

In dieser Arbeit soll im Rahmen des Roboter–Nachtwächter–Projektes und der in diesem lokalisierten GPE eine Background Subtraction für den Einsatz am Roboter entwickelt werden.
Dazu soll zuerst ein Überblick über den aktuellen technischem Stand in diesem Bereich geschaffen werden. Zudem müssen die Rahmenbedingungen, die durch das Projekt, die Hardware und den Klassifikator gegeben sind, beleuchtet und bedacht werden.
Im Anschluss daran erfolgt eine praktische Umsetzung eines Algorithmus als Ergebnis der Recherche. Ziel dabei ist ein Modul, welches aus den Sensordaten des Roboters die für den Klassifikator relevanten Daten extrahiert. Diese Daten müssen dann an den Klassifikator in einer für ihn nutzbaren Form weitergeleitet werden.
Das Zusammenspiel der beiden Komponenten muss im Anschluss getestet werden.
Im Verlauf der Arbeit erfolgt eine Einordnung in den Gesamtkontext des Projektes allgemein und den Klassifikator speziell.

3 LÖSUNGSWEG

Zur Lösung des Problems der Vorverarbeitung der Daten für die GPE ist es zuerst notwendig, die Rahmenbedingungen genauer zu umreißen. Dafür wird das Roboter–Nachtwächter–Projekt und sein Szenario ausführlicher erläutert. Im Anschluss daran soll die Auswahl des Algorithmus für den Klassifikator genauer erläutert werden. Dabei wird auf die Abhängigkeit von einer Background Subtraction als Vorverarbeitung besonders eingegangen. Darauf folgend wird der aktuelle Stand der Technik in dem Bereich betrachtet. Dazu werden klassische Algorithmen und Deep–Learning–Algorithmen unterschieden. Im Anschluss daran muss aus diesen Methoden eine, welche sich für das Szenario eignet, ausgewählt und umgesetzt werden. Diese soll an der entsprechenden Stelle, welche in Abbildung 3.1 aus dem Kontext der GPE entnommen werden kann, im Klassifikationsprozess integriert werden.

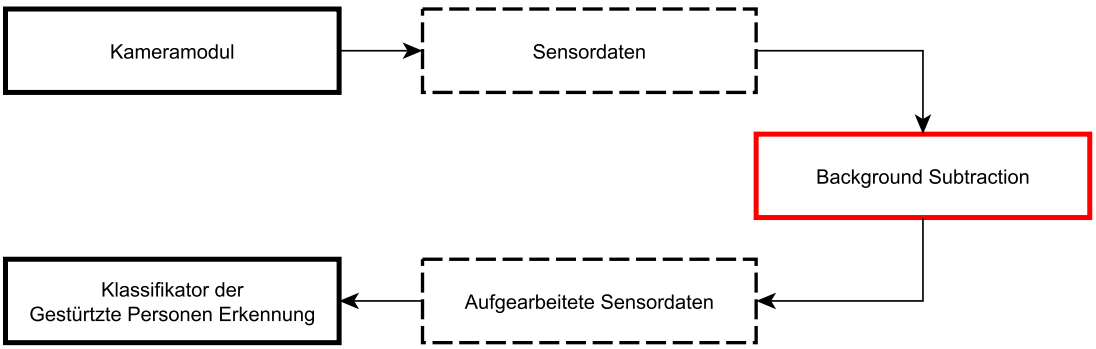

Abbildung 3.1: Kontext der Background Subtraction in der gesamten GPE

4 HAUPTTEIL

In diesem Teil geht es um die Umsetzung des Lösungsweges zur Erreichung des Ziels. Dafür muss das Projekt in Abschnitt 4.1 zuerst in den Kontext des RoNisCo–Projektes eingeordnet werden. Im Anschluss daran wird die GPE mit ihrem Klassifikator thematisiert. Darauffolgend werden im Abschnitt 4.3 die Rahmenbedingungen, die sich aus dem Projekt und der GPE ergeben, definiert. Dem schließt sich eine Literaturrecherche zum Thema Background Subtraction an, wobei zwischen klassischen Ansätzen und Ansätzen mit Deep–Learning unterschieden wird. Die Entscheidung, welcher Ansatz umgesetzt werden soll, erfolgt im Anschluss, gefolgt von der Umsetzung in Abschnitt 4.6.

4.1 EINORDNUNG IN DAS NACHTWÄCHTER–PROJEKT

Bei dem Nachtwächter–Projekt handelt es sich um einen Teil eines Forschungsprojektes, welches die Praktikabilität von Roboterassistenzsystemen bei der Anwendung in Pflegeeinrichtungen ergründen möchte und daraus einen Prototyp entwickeln will. Dabei soll untersucht werden, wie die Unterstützung der Pflegekräfte durch das System aussehen und funktionieren kann. Anwendungsbereiche könnten Unterstützung bei Therapien oder der Unterhaltung der Bewohner sein. Besonders bietet sich auch die Nachtschicht an.
In jenem Szenario ist das Nachtwächter–Projekt, wie es von [Lischke et al., 2017] und [Brose, 2018] beschrieben und auch als RoNisCo bezeichnet wird, angesiedelt. Der bereits in Kapitel 1 erwähnte Personalmangel ist in der Nachtschicht besonders bemerkbar. In der konkreten Einrichtung, mit welcher im Projekt kooperiert wird, führt das dazu, dass weniger Pflegekräfte als Stockwerke mit Bewohnern vor Ort sind. Gleichzeitig lebt in Pflegeeinrichtungen in Deutschland ein sehr großer Anteil an Patienten mit Formen von Demenzerkrankungen, wie in der in [Schäufele et al., 2013] durchgeführten Untersuchung festgestellt wurde. Ein Symptom von Demenzerkrankungen kann eine Orientierungslosigkeit in Raum und Zeit sein. Die kann dazu führen, dass Bewohner nachts ihre Zimmer verlassen und versuchen könnten, die Einrichtung zu verlassen. Sollte es ihnen gelingen, das Gebäude zu verlassen, ist es ihnen nicht möglich, dies auch wieder zu betreten, da die Türen aus Brandschutzgründen von innen zwar öffenbar, von außen aber verschlossen sind. Das ist aber nicht das einzige Risiko für diese Personen. Bei älteren Menschen besteht ein höheres Sturzrisiko. Dieses könnte dazu führen, dass die Personen selbstständig nicht wieder aufstehen können oder sich verletzen. Gleichzeitig erhöht die geringe Anzahl an Pflegekräften die Wahrscheinlichkeit, dass eine Person die Einrichtung verlässt oder ein Sturz zu spät bemerkt wird.

Um dem entgegenzuwirken, soll ein Assistenzroboter eingesetzt werden, welcher autonom die Flure in den Stockwerken, in denen aktuell keine Pflegekräfte sind, überwacht. Dieser bietet gegenüber einem statischen Kamerasystem unter anderem folgende Vorteile:

- Der Roboter ist mobiler als fest montierte Kameras.
- Ein Roboter besitzt verschiedenste Sensoren, die er zur Navigation nutzt, die man zusätzlich einbinden kann, wie beispielsweise $360°$ Laserabstandssensoren oder ein Mikrofonearray zur akustischen Ortung.
- Er bietet die Möglichkeit der direkten Roboter–Mensch–Interaktion mit den Bewohnern.
- Es besteht die Möglichkeit, das Pflegepersonal direkt zu unterstützen, zum Beispiel durch Mitführen eines Erste–Hilfe–Koffers. Solche Unterstützungen sind auch gewünscht, wie man bei [Hebesberger et al., 2015] nachlesen kann.
- Ein Roboter ist in verschiedenen Szenarien einsetzbar. So kann derselbe Roboter nachts die Flure überwachen und am Tag bei der Therapie unterstützen oder Bewohner unterhalten.

Aufgrund dieser Vorteile wurde ein Assistenzroboter im RoNisCo ausgewählt.
Um diese Aufgabe übernehmen zu können, muss der Roboter gewisse Fähigkeiten aufweisen. Er muss in der Lage sein, sich selbstständig und zielgerichtet innerhalb eines Stockwerkes zu orientieren und entsprechend zu bewegen. Der Ansatz zur Lösung dieses Problems lässt sich bei [Bahrmann et al., 2016; Bahrmann et al., 2014; Brose, 2018] nachlesen. Zusätzlich muss er sowohl mit dem Personal, welches in anderen Gebäudeteilen unterwegs sein kann, zuverlässig kommunizieren können, falls er oder ein Bewohner Hilfe benötigt. Diese Problematik wird von [Lischke et al., 2017] und [Brose, Bahrmann et al., 2019] genauer betrachtet. Außerdem soll er auch sozial verträglich mit den betroffenen Bewohnern in Kontakt treten können, damit wird sich unter anderem in [Brose, 2018] beschäftigt. Vor allem muss er aber Personen detektieren können.
Bei der Personenerkennung muss man zwischen einer klassischen Personenerkennung, welche dazu gedacht ist, Personen, welche sich im Sichtfeld aufhalten und natürlichere Körperhaltungen einnehmen, zu erkennen, und einer Erkennung für gestürzte Personen unterscheiden. Unter Umständen werden diese teilweise auch von einer klassischen Personenerkennung erkannt, allerdings ist das Risiko für falsch negative Klassifikationen höher. Dieses Risiko sollte aber dringend vermieden werden, da diese Personen schnellstmöglich Hilfe benötigen könnten. Daher wird für solche Szenarien eine gesonderte Erkennung, wie sie in Abschnitt 4.2 beschrieben wird, genutzt. Diese ist darauf ausgelegt, nur gestürzte Personen zu detektieren.

4.2 GESTÜRZTE–PERSONEN–ERKENNUNG (GPE)

Bei einer GPE handelt es sich um eine Sonderform der Personenerkennung, welche sich besonders darauf fokussiert, gestürzte Personen zu erkennen. In [F. Zhao et al., 2018] und [Brose, Erzgräber et al., 2019] wird beschrieben, dass es notwendig wird, Fälle von gestürzten Personen gesondert zu behandeln, da die Zuverlässigkeit von klassischen Personenerkennungen, wie OpenNI SDK und Microsoft Kinect SDK, bei gestürzten Personen geringer ist. Das Problem dabei ist, dass gestürzte Personen selten natürliche Körperhaltungen aufweisen. Zudem können sie von Gehhilfen oder anderen Gegenständen teilweise verdeckt sein. Mitunter sind für die Personenerkennung wichtige Körperteile, wie der Kopf oder Gliedmaßen unter Umständen nicht sichtbar. Die daraus resultierende geringere Zuverlässigkeit der Erkennung ist in der Hinsicht kritisch, da gestürzte Personen ernster verletzt sein können. Aus diesem Grund ist es wichtig Hilfe, schnellstmöglich gewährleisten zu können.
Um diese Problematik zu lösen wurde in [Brose, Erzgräber et al., 2019] der aktuelle Stand der Technik in dem Bereich unter Beachtung der Rahmenbedingungen, die durch das Projekt definiert werden, betrachtet. Diese sind beispielsweise der Einsatz eines mobilen Roboters und die Vergleichbarkeit der Einsatzszenarien.
Dabei haben sich hauptsächlich zwei Ansätze ergeben, die von den Rahmenbedingungen vergleichbar waren. Der eine war [Wengefeld et al., 2016]. Bei diesem wird ein Roboter im häuslichen Umfeld eingesetzt und der Roboter ist von demselben Hersteller wie die für RoNisCo genutzten. Ein anderer Ansatz ist in [Antonello et al., 2017] beschrieben, wo ein Roboter mit vergleichbaren Maßen und demselben Kamerasystem genutzt wird. Beide Ansätze haben gute Ergebnisse geliefert, welche allerdings nicht so gut vergleichbar sind, da sie verschiedene Maße nutzen. Beim Vergleich der beiden Ansätze sind einige Ähnlichkeiten aufgefallen, aus denen ein grundlegender Ablauf für eine GPE abgeleitet werden kann.

4.2.1 ABLAUF DER GPE

Aus beiden Ansätzen ließ sich ein gemeinsamer Ansatz ableiten. Dieser ist in Abbildung 4.1 skizziert. Dabei lässt sich erkennen, dass beide Ansätze mit einem RGB–Tiefenbild (RGB–D) beginnen. Im Anschluss daran wird der Hintergrund entfernt. Danach werden die Bodenebene und alle Punkte oberhalb einer festen Höhe Z entfernt, da gestürzte Personen nur bis zu einer bestimmten Höhe zu erwarten sind. Es folgt ein Filterschritt, in dem alle Punkte, die keine oder sehr wenige Nachbarn haben, entfernt werden. Anschließend wird eine Punktwolke aus den Daten erzeugt. Diese Punktwolke wird mithilfe von Clusterverfahren segmentiert und in Objekte eingeteilt. Im Folgendem werden die Objekte dem Klassifikator zugeführt, welcher entscheidet, ob es sich bei dem Objekt um eine gestürzte Person handelt. Von [Antonello et al., 2017] wird noch vorgeschlagen, die Laserdaten, die der Roboter zur Navigation nutzt, zur Validation des Ergebnisses zu nut-

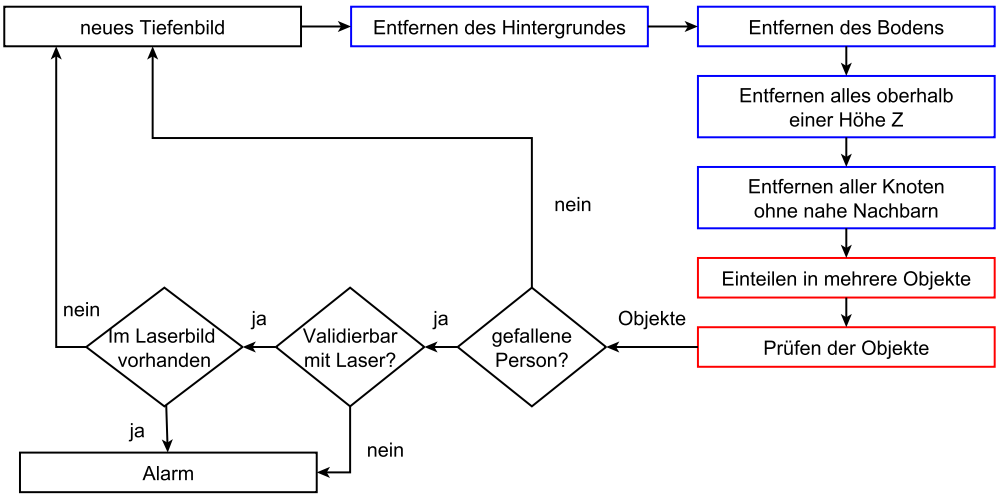

Abbildung 4.1: Vollständiger Ablauf der Klassifikation

zen. Sie schlagen auch vor, bei widersprüchlichen Ergebnissen den Vorteil des Roboters zu nutzen, die Szene erneut aus einem anderen Blickwinkel zu betrachten.
Allerdings unterscheiden sich beide Herangehensweisen. Die Entfernung des Hintergrundes wird beispielsweise auf verschiedene Arten gelöst, ebenso das Unterteilen in Objekte. Im Ansatz von [Wengefeld et al., 2016] wird genauer erklärt, wie Parameter gewählt werden und wie diese berechnet werden. Schlussfolgernd wurde die Entscheidung getroffen, diesen Ansatz zu testen. Aufgrund unserer Rahmenbedingungen war es aber notwendig, den Hintergrund anders zu entfernen und die Vorverarbeitung anzupassen. Es bestand das Ziel, das Erstellen von Daten zum Testen des Klassifikators zu vereinfachen. Dazu wurde der in Abbildung 4.1 gezeigte Ablauf in zwei Schritte unterteilt. Der erste Teil, dessen Elemente blau markiert sind, wird in Abschnitt 4.2.2 beschrieben. Wie die Einteilung in Objekte und die Klassifikation (rot markierte Elemente) erfolgt, wird in Abschnitt 4.2.3 beschrieben.

4.2.2 BISHERIGER ANSATZ FÜR DIE VORVERARBEITUNG

Dieser Teil, welcher auch in [Brose, Erzgräber et al., 2019] beschrieben wird, übernimmt die Aufgaben einer Background Subtraction. Dafür verwendet er Ansätze, welche für statische Kamerasysteme genutzt werden, für das geplante Einsatzszenario allerdings nicht geeignet sind. Diese Aufgabe soll von der in dieser Arbeit neu entwickelten Background Subtraction übernommen werden.
Der bisherige Ansatz zur Lösung der Aufgabe wird in Abbildung 4.2 skizziert. Dabei wird eine Szene zweimal aufgenommen. Die erste Aufnahme enthält die Szene ohne das zu klassifizierende Objekt, während die zweite Szene das Objekt mit beinhaltet. Das Objekt muss das Einzige sein, was sich zwischen den beiden Aufnahmen ändert. Als Nächstes wird eine Boolean Gridmap, mit derselben Auflösung wie die Aufnahmen, erstellt. In dieser Gridmap erhalten alle Zellen den Wert $true$, in welchem die Differenz einen

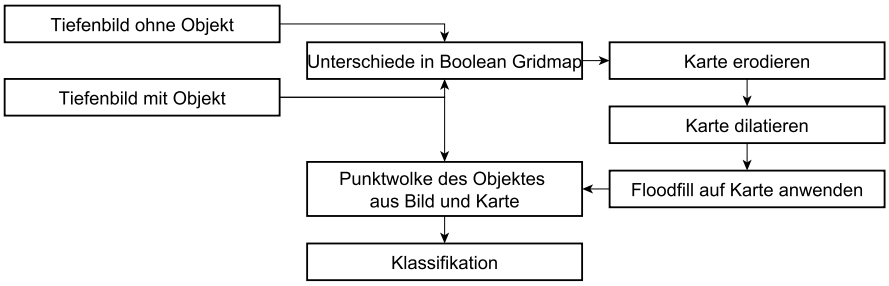

Abbildung 4.2: Ablauf der Vorverarbeitung

Schwellwert überschreitet. Der Schwellwert wurde eingeführt, um beispielsweise minimale Bewegungen des Roboters oder Linseneffekte auszugleichen. Im Anschluss daran werden die Areale auf der Karte, die $true$–Werte enthalten, erodiert. Dies wird genutzt, um nur wenige Zellen umfassende Bereiche zu entfernen. Da davon aber auch Gebiete, welche groß genug sind, betroffen sind und da dadurch kein Informationsverlust entstehen soll, werden diese im nächsten Schritt wieder dilatiert. Im Anschluss daran werden mit einem Floodfill–Algorithmus noch verbliebene Flächen unter einer Grenzanzahl an Zellen entfernt.

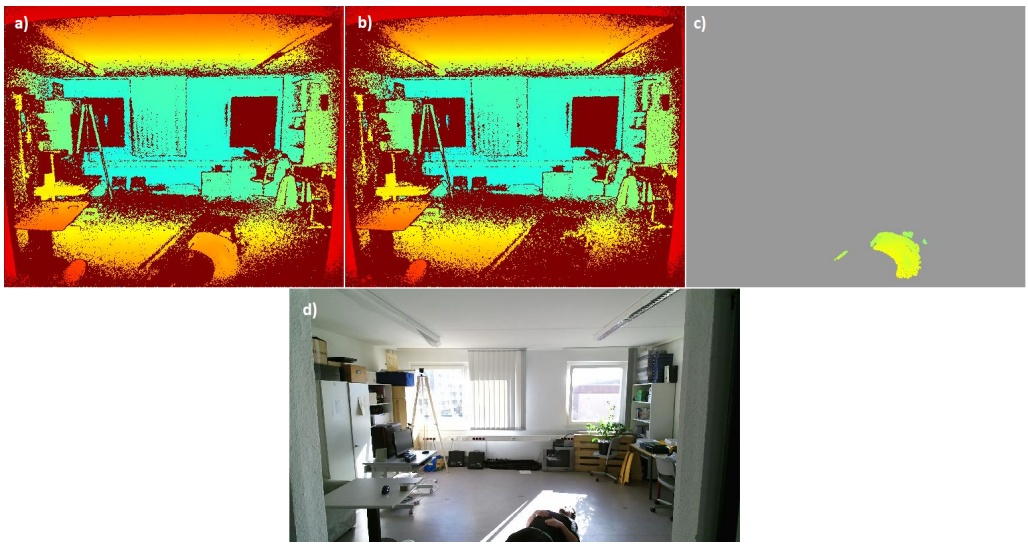

Abbildung 4.3: a) RGB–D–Bild mit Person, b) RGB–D–Bild ohne Person, c) RGB–D–Daten der Person, d) RGB-Bild

Der letzte Schritt vor der Klassifikation erstellt aus der Karte und der Aufnahme der Szene mit dem Objekt eine 3D–Punktwolke des Objektes. Dabei werden die 3D–Koordinaten aller Pixel berechnet, deren Wert kein Fehlerwert ist und die entsprechenden Gridmap–Zellen einen $true$–Wert gespeichert haben. Diese Punktwolke wird zur Weiterverarbeitung an den Klassifikator gegeben.

4.2.3 KLASSIFIKATOR

Wie bereits erwähnt orientiert sich die Umsetzung des Klassifikators an [Wengefeld et al., 2016]. Dazu wird die zuvor erzeugt Punktwolke in Cluster unterteilt. Abweichend von der Umsetzung in der Publikation wird ein Expectation–Maximization–Cluster–Verfahren genutzt, da eine Implementation davon schon an einer anderen Stelle des Projektes genutzt wird. Im Anschluss daran wird, wie von [Wengefeld et al., 2016] beschrieben und in 4.4 zu sehen ist, die Hauptachse des Objektes bestimmt. Dazu werden die beiden

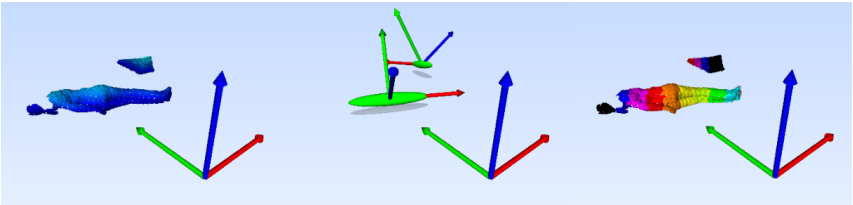

Abbildung 4.4: Suchen der neuen Hauptachse von [Wengefeld et al., 2016]

am weitesten voneinander entfernten Punkte genutzt. Auf Basis dieser neuen Hauptachse wird ein neues Koordinatensystem aufgespannt und die Punktwolke in dieses überführt. Darauffolgend wird das Cluster in Layer einer festen Höhe entlang der Hauptachse eingeteilt. Für die Layer wird einzeln ein Histogramm der Oberflächennormalen berechnet. Diese Histogramme sind die Eingangsdaten der eigentlichen Klassifikation. Dabei wird ein Adaboost–Algorithmus mit 50 Entscheidungsbäumen und Baumtiefe von 4 als Weaklearner (WL) genutzt. Wenn zwei Layer eines Clusters positiv klassifiziert sind, gilt das Cluster als gestürzte Person.
Um die Praktikablität zu testen, wurden mit der in 4.2.2 beschriebenen Methode ungefähr 450 Bilder erstellt. In diesen wurden in einem Laborraum auf dem Boden liegende Personen und andere Objekte aufgenommen. Im Anschluss wurden die Bilder in zwei Gruppen eingeteilt. 50 Bilder wurden für die Validierung zufällig ausgewählt, dabei wurde lediglich Wert darauf gelegt, dass 25 als positiv und 25 als negativ zu klassifizierende dabei sind. Mit den restlichen Daten wurde der Klassifikator trainiert. Bei der Validierung, mit den dem Klassifikator unbekannten Daten, wurde dann ein F–Score von 0,8 erreicht.

4.3 RAHMENBEDINGUNGEN

Durch die Einordnung in den Kontext des Projektes ergeben sich einige zu beachtende technische Rahmenbedingungen.
Die Eingangsdaten für die Background Subtraction sind die RGB–D–Bilder, welche das Kameramodul des Roboters liefert. Dies geschieht mit 10 Bildern pro Sekunde (fps). Daraus ergibt sich, dass die gesamte Klassifikation eines Bildes in weniger als 100 ms erfolgen sollte, damit alle Bilder genutzt werden können.
Gleichzeitig findet parallel dazu eine Personenerkennung statt, welche auch Rechenleistung benötigt. Zusätzlich soll der Roboter im Praxiseinsatz während der Nachtschicht das

Personal unterstützen. Deshalb ist es notwendig, dass er mindestens 8 Stunden Einsatzzeit bietet, ohne geladen werden zu müssen. Zu seiner Einsatzbereitschaft gehören aber nicht nur die Personenerkennung und die GPE, sondern auch die Navigation, der Antrieb und die Kommunikation mit dem Personal und den Bewohnern. Deshalb sollten rechenintensive Algorithmen vermieden werden, sofern es möglich ist.
Durch das klare Szenario im Einsatz für den Nachtwächter auf den Fluren der Einrichtung ergibt sich, dass die Szenen meist sehr wenige bis keine Vordergrundobjekte enthalten. Dies hängt damit zusammen, dass sich auf den Fluren wenige Möbel befinden. Gleichzeitig sind auch in der Nacht wenige Menschen auf den Fluren unterwegs. Folglich besteht der Hintergrund meist nur aus den Wänden, dem Fußboden und der Decke. Außerdem ergeben sich aus dem Szenario noch einige Probleme für die Background Subtraction. Der Algorithmus sollte für RGB–D–Bilder anwendbar sein. Weitere Punkte, die von [Greff et al., 2012] erwähnt werden und zu beachten wären, sind:

- „Moved Objects“ beziehungsweise „Dynamic Background“ beziehen sich darauf, dass sich der Hindergrund verändern kann. Aufgrund der Nutzung eines mobilen Roboters muss davon ausgegangen werden, dass sich der Hintergrund in jedem Bild ändert.

- „Time of Day“ bezeichnet das Problem, welches durch unterschiedliche Beleuchtung zu unterschiedlichen Tageszeiten auftreten kann. In der Nachtschicht sollte das Problem der Sonneneinstrahlung eine geringere Rolle spielen, allerdings kann die Beleuchtung auch durch Licht aus geöffneten Türen, dem An- oder Ausschalten der Beleuchtung oder Mondlicht durch Fenster verändert werden.

- „Bootstrapping“ bezeichnet die Fähigkeit ein Hintergrundmodell trotz der Präsenz von Vordergrundobjekten zu lernen. Das tritt in der Praxis dadurch auf, dass nach jeder Bewegung des Roboters ein neues Hintergrundmodell benötigt wird. In der Anwendung können sich auf der Aufnahme, aber Personen und gestürzte Personen befinden.

- „Uncertainty“ beschreibt das Handhaben von Pixeln, welche keinen Wert beziehungsweise einen Fehlerwert haben. Ein Fehlerwert wird bei RGB–D–Daten beispielsweise dort eingetragen, wo die Entfernung größer als die Reichweite der Kamera ist.

Es ist auch wichtig, die Limitierungen des Kameramoduls zu berücksichtigen. Hierbei handelt es sich um ein Microsoft Kinect V2 Modul, welches ursprünglich für den Einsatz bei einer Spielekonsole entwickelt wurde. Dementsprechend sind Reichweite und Blickfeld begrenzt. Die Distanz der zu erfassenden Objekte muss bei mindestens 50 cm liegen. Zusätzlich sollte das Objekt nicht weiter als 5 bis 8 m entfernt sein, da mit zunehmender Entfernung die Präzision sinkt. Das Sichtfeld beträgt laut [Greff et al., 2012] ungefähr $58°$. Ein weiterer sehr wichtiger Punkt ist, dass Personen nicht als Hintergrundobjekte klassifiziert werden dürfen. Wie bereits in den Kapiteln 1, 4.1 und 4.2 erwähnt wird, benötigen

diese Personen dringend Hilfe. Daher müssen falsch als Hintergrund klassifizierte Personen dringend vermieden werden, damit sie korrekt klassifiziert werden können und die Wahrscheinlichkeit, sie zu übersehen, möglichst minimal ist.

4.4 STAND DER TECHNIK

Bevor neue Ansätze betrachtet werden, bietet sich ein Blick auf die Umsetzung der Background Subtraction in den beiden Ansätzen für den Klassifikator an. Im Ansatz von [Wengefeld et al., 2016] werden für die Background Subtraction 3D Karten der Einsatzumgebung genutzt. Diese 3D Karten werden bei der Einrichtung des Systems in Abwesenheit von Personen erstellt. In der Anwendung wird mit dieser Karte die Background Subtraction für die RGB–D–Bilder durchgeführt. Diese Möglichkeit ist aktuell nicht umsetzbar, da aktuell keinen 3D Kartierungsalgorithmus zur Verfügung steht. Somit können wir auch keine 3D Karte der Einsatzumgebung erstellen und als Eingangsdaten verwenden.
Von [Antonello et al., 2017] wird hingegen ein Ansatz genutzt, welcher keine weiteren Daten außer den RGB–D–Bildern benötigt. Dafür wird zuerst der gesamte Bereich oberhalb einer festen Höhe von 0,7 Metern entfernt. Im Anschluss wird unter Nutzung eines Random Sample Consensus (RANSAC)–Algorithmus die Bodenebene gesucht und entfernt. Die verbleibenden Punkte werden gefiltert, sodass Punkte mit zu wenig Nachbarn entfernt werden. Dann wird die entstandene Punktwolke in Voxel übersegmentiert, sie wird also in mehr Voxel eingeteilt als Objekte in der Punktwolke enthalten sind. Die entstandenen Voxel werden einzeln klassifiziert, ob sie Teil einer Person sind. Auf die positiv klassifizierten Voxel wird anschließend das Clusterverfahren aus dem Klassifikator angewendet, diese Schritte werden in Abbildung 4.5 veranschaulicht.

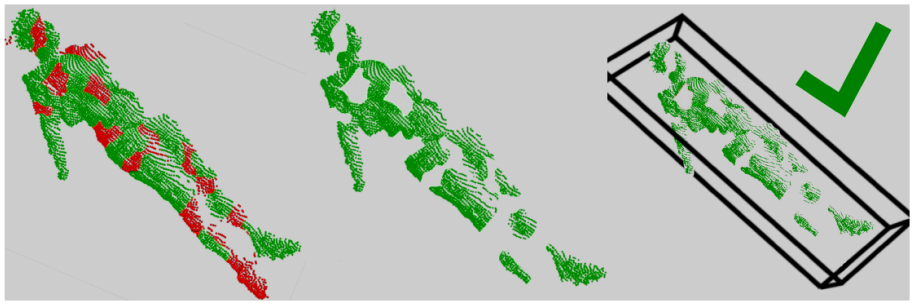

Abbildung 4.5: Voxel und Klassifikation von [Antonello et al., 2017]

Da beide Ansätze in dieser Form nicht direkt umsetzbar sind, wird im Folgenden ein Überblick über den aktuellen Stand der Technik erstellt. Dazu werden in Abschnitt 4.4.1 die Bewertungskriterien für die Modelle zusammengefasst. Basierend auf diesen Kriterien werden in Abschnitt 4.4.2 klassische Algorithmen und in Abschnitt 4.4.3 Deep–Learning–Algorithmen bewertet. Dem folgt eine Gegenüberstellung der besten Ansätze aus beiden Kategorien.

4.4.1 BEWERTUNGSKRITERIEN

Durch die Rahmenbedingungen des Projektes gibt es viele Faktoren, welche bei der Bewertung der Ansätze eine Rolle spielen. Um eine bessere Nachvollziehbarkeit zu erreichen, welche Faktoren die wichtigsten sind, sollten sie konkret benannt werden:

1. Der Ansatz muss für sich bewegende Kameras funktionsfähig sein.
2. Die Wahrscheinlichkeit, dass gestürzte Personen als Hintergrundobjekte betrachtet werden, muss so klein wie möglich sein. Idealerweise soll sie 0 sein.
3. Der Algorithmus muss mit RGB–D–Daten arbeiten können.
4. Der Ansatz sollte so einfach wie möglich sein und mit mindestens 10 fps laufen.
5. Dazu sollte keine zusätzliche oder spezielle Hardware, wie High End Grafikkarten, benötigt werden.

4.4.2 KLASSISCHE ANSÄTZE

Als klassische Ansätze werden Ansätze bezeichnet, welche kein Deep–Learning nutzen und teilweise schon länger zum Einsatz kommen. Eine Übersicht über diese Ansätze bietet [Greff et al., 2012]. Dabei werden „First Frame Subtraction“, „Single Gaussian“, „Codebook Model“ und „Minimum Background“ als grundlegende Ansätze genannt. Zusätzlich werden noch Ansätze betrachtet, welche keiner dieser Kategorien klar zuordenbar sind.

FIRST FRAME SUBTRACTION

Die First Frame Subtraction, wie sie in [Greff et al., 2012] beschrieben wird, nutzt für die Background Subtraction das erste Bild. Von diesem werden alle weiteren Bilder abgezogen. Bei der Überschreitung eines Grenzwertes für den Betrag der Differenz der Werte eines Pixels wird dieses als Vordergrund markiert. Dieses Verfahren lässt sich auch problemlos auf RGB–D–Bilder anwenden. Außerdem hat es eine große Ähnlichkeit mit der in Abschnitt 4.2.2 beschriebenen Umsetzung. Das Verfahren ist entsprechend des ersten Kriteriums für unser Szenario nicht geeignet, da es sehr anfällig für sich ändernde Hintergründe ist und diese dauerhaft auftreten, wenn der Roboter fährt oder sich dreht.

SINGLE GAUSSIAN UND GAUSSIAN MIXTURE MODELS

Von [Greff et al., 2012] wird das Single Gaussian Verfahren so beschrieben, dass für jedes Pixel eine Gauß–Verteilung vorliegt. Während eines Trainings werden dabei Mittelwert und Varianz der Verteilung für jedes Pixel bestimmt. Im weiteren Verlauf wird jedes Pixel, welches um ein konstantes n–faches der Varianz von seinem Mittelwert abweicht, als Vordergrund betrachtet. Im Folgenden werden vier Beispiele für konkrete Umsetzungen betrachtet, um die Eignung für unser Szenario besser einschätzen zu können. Bei

dem Ersten handelt es sich um [Wren et al., 1997]. Aus diesem geht allerdings hervor, dass auch dieser Ansatz für sich stark ändernde Hintergründe sehr anfällig ist. Damit ist dieser Ansatz für ein Szenario mit einem mobilen Roboter ungeeignet.
Es gibt auch Gaussian Mixture Models, wie sie beispielsweise von [Song et al., 2014], [Langmann et al., 2010] und [Nguyen et al., 2014] beschrieben werden. Von [Song et al., 2014] wird die Nutzung von RGB–D–Bildern und Farbbildern vorgeschlagen. Für diese wird dann ein Modell wie oben beschrieben aufgebaut. Damit sollen Camouflage–Effekte, also dass Objekte gleicher Farbe nicht unterschieden werden können, ausgeglichen werden. Gleichzeitig soll die Schwachstelle der Uncertainty bei Pixeln, für die keine Entfernung ermittelt werden konnte, in RGB–D–Bildern gelöst werden.
Ein weiterer Ansatz, der eine Kombination von RGB–D–Bildern und Farbbildern vorschlägt, ist von [Nguyen et al., 2014]. Dabei wird vorgeschlagen, zwei Modelle aufzubauen, eines für die Farbdaten und eines für die RGB–D–Daten. Das Ergebnis ist eine Kombination der Ergebnisse der beiden Modelle, wobei immer das sicherere Ergebnis genutzt wird. Allerdings haben diese Modelle das Problem, dass sie hauptsächlich für statische Szenarien mit sich weniger stark ändernden Hintergründen konzipiert sind. Konkreter benannt wird dies von [Langmann et al., 2010], welcher ausführt, dass sich Gauß–Modelle nur für meist statische Hintergründe eignen. Daher sind sie für unser Szenario eher ungeeignet, da sie dem ersten Kriterium widersprechen.

CODEBOOK MODEL

Für das Codebook Modell fasst [Greff et al., 2012] den Ansatz von [Kim et al., 2005] so zusammen, dass das Hintergrundmodell über einen längeren Zeitraum aufgebaut wird. Dabei werden die Stichprobenwerte jedes Pixels in Codewörter aggregiert. Alle Codewörter bilden das namensgegebende Codebook. Allerdings erwähnt [Greff et al., 2012], dass dieses Modell aufwendiger als die Anderen ist. Der Ansatz ermöglicht es, ein Modell für den Hintergrund trotz des Vorhandenseins dynamischer Vordergrundobjekte aufzubauen. Damit kann das Bootstrapping Problem oder das für dynamische Hintergründe gelöst werden. Von [Kim et al., 2005] wird erwähnt, dass das Modell für Farbbilder konzipiert und mit Anpassungen für Graustufenbilder nutzbar ist. Durch [Ilyas et al., 2009] wurde das Modell modifiziert, um die Genauigkeit zu erhöhen. Es gibt auch Ansätze, welche RGB–D–Daten nutzen, wie durch [Murgia et al., 2014] beschrieben wird. Ziel ist dabei ebenfalls die Genauigkeit zu erhöhen.
Die von [Ilyas et al., 2009] und [Murgia et al., 2014] erwähnten Szenarien beziehen sich nur auf statische Kamerasysteme, wie Überwachungskameras. In seinem Fazit stellt [Greff et al., 2012] fest, dass der grundlegende Ansatz ähnlich gute Ergebnisse liefert wie der Minimum Background. Welcher dabei allerdings schneller und einfacher ist.

MINIMUM BACKGROUND

Dieser Ansatz ist laut [Greff et al., 2012] eines der ersten Modelle, welches komplett für RGB–D–Daten entwickelt wurde. Dabei wird während einer Trainingsphase ein minimaler Tiefenwert für jedes Pixel gespeichert. Ist ein Messpunkt näher an der Kamera als der gespeicherte Wert, wird der Pixel dem Vordergrund zugeordnet. Ein Ansatz zur Nutzung eines solchen Algorithmus zur Erkennung des Sturzrisikos bei älteren Personen wird von [Stone & Skubic, 2011] vorgestellt. In ihrem Versuchsaufbau wird mit statischen Kamerasystemen gearbeitet, was die Anwendbarkeit für RoNisCo schlechter abschätzbar macht. Ein weiterer Ansatz, der diesem Oberbegriff zugeordnet werden kann, ist [Fengliang Xu & Kikuo Fujimura, 2003]. Dabei wird versucht, Menschen unter Nutzung von Graustufen–Bildern und RGB–D–Bildern zu detektieren. Bei ihrer Background Subtraction werden zuerst Wände, Decke und Boden entfernt. Im nächsten Schritt suchen sie interessante Punkte, indem sie die Kanten der Objekte durch Wechsel in der Textur detektieren. Von den Punkten von Interesse spannen sie Regionen auf und fügen diese zu Objekten zusammen, welche sie im weiteren Verlauf analysieren.
Bei ihrer Auswertung kommen [Greff et al., 2012] zu dem Ergebnis, dass Minimum Background Ansätze und Codebook Ansätze die besten Ergebnisse der von ihnen betrachteten liefern. Aufgrund seiner geringeren Komplexität ist dieser Ansatz deutlich schneller als das Codebook Modell.
Auffällig ist, dass die Ansätze von [Stone & Skubic, 2011] und [Fengliang Xu & Kikuo Fujimura, 2003] eine gewisse Ähnlichkeit zum Ansatz von [Antonello et al., 2017] haben. Dort werden in Teilen des Bildes feste Grenzen definiert, bevor die Bodenebene als sicheres Hintergrundobjekt entfernt wird. Des Weiteren wird das Einteilen in mehrere Objekte genutzt, wobei darauf geachtet wird, ob die kleineren Regionen zu einer Person gehören könnten.

WEITERE ANSÄTZE

Es gibt noch weitere Ansätze, wie Wallflower von [Toyama et al., 1999], welche sich nicht eindeutig einer der vier von [Greff et al., 2012] genannten Kategorien zuordnen lassen. [Toyama et al., 1999] nehmen Teile von mehreren der zuvor erwähnten Kategorien, um die Problemtypen bei der Background Subtraction abzufangen. So betrachten sie das Bild beispielsweise auf Pixelebene ähnlich wie bei den Single Gaussian Modellen, betrachten allerdings auch ganze Frames, wie in der First Frame Subtraction. Sehr gut an der Veröffentlichung ist, dass sie ihren Ansatz mehreren anderen konkreten Implementationen der grundlegenden Ansätze gegenüberstellen. Wobei bei der Gegenüberstellung keine Vertreter der Ansätze Codebook Modell und Minimum Background enthalten sind, da beide jünger sind. Sie stellen fest, dass ihr Algorithmus über alle klassischen Probleme betrachtet, die anderen Systeme übertrifft. Allerdings wird aus den Daten ersichtlich, dass der Algorithmus für die in den Rahmenbedingungen (siehe 4.3) erwähnten Proble-

me am anfälligsten ist. Zusätzlich ist der Algorithmus nicht für RGB–D–Daten entwickelt worden und die Erfolgschancen einer solchen Anpassung schwer abschätzbar. Außerdem ist der Ansatz entwickelt worden, um die Background Subtraction für statische Überwachungssysteme zu verbessern. Wir nutzen einen mobilen Roboter, daher ist dieser Ansatz entsprechend des ersten Kriteriums für uns weniger geeignet.
Ein weiterer Ansatz ist eine Active frame subtraction wie sie von [Hashiyama et al., 2003] konzipiert wurde. Diese ist speziell für bewegte Kameras konzipiert und versucht mit Gyrosensoren die Bewegung der Kamera zwischen zwei Bildern zu rekonstruieren. In dem konkreten Ansatz haben sie eine sehr gute Laufzeit von 0,02 Sekunden pro Bild. Aus einem Graustufenbild extrahieren sie ein Konturenbild. Auf dieses wenden sie die Bewegung der Kamera an. Im Anschluss daran haben sie ein Template Matching, um Fußgänger zu finden. Ein Template Matching ist für Fußgänger zuverlässig, da deren Silhouetten eine ähnliche Grundform haben. Gestürzte Personen hingegen haben keine so einheitliche Grundform, da diese sehr unnatürliche Körperhaltungen haben können. Deshalb bietet sich dieses Verfahren entsprechend des zweiten Bewertungskriteriums für unseren Ansatz weniger an.

4.4.3 DEEP–LEARNING

In diesem Teil soll es einen Überblick über Deep–Learning Algorithmen geben. Dazu eignet sich als Leitlinie die Veröffentlichung von [Bouwmans et al., 2019]. Diese bietet einen sehr ausführlichen Überblick über aktuelle Umsetzungen des Problems, mit einer anschaulichen Gegenüberstellung der Ansätze. Zusätzlich wird eine Einführung in Deep–Learning vor dem Hintergrund der Bildverarbeitung gegeben.
Grundlegend ist das Vorgehen ähnlich, mithilfe eines neuronalen Netzes wird ein Hintergrundmodell erzeugt. Dieses wird vom aktuellen Bild abgezogen. Das Ergebnis ist der Vordergrund. Für die Erzeugung des Hintergrundes werden verschiedene neuronale Netze genutzt. Bei ihrer Ausarbeitung haben [Bouwmans et al., 2019] sieben Kategorien definiert, in welche sie die Netzwerke einteilen. Diese sind:

- „Basic Convolutional Neural Networks (CNN)“
- „Multi–scale and Cascaded CNNs“
- „Fully CNNs“
- „Deep CNNs“
- „Structured CNNs“
- „3D CNNs“
- „Generative Adversarial Network (GAN)s“

In diese Kategorien haben sie 32 Ansätze eingeordnet.

BASIC CNNS

Die Ansätze, die dieser Kategorie zugeordnet wurden, sind dem Namen entsprechend Basic CNNs. Diese zeichen sich dadurch aus, dass sich ihre Architektur stark an älteren CNNs, wie LeNet-5 von [LeCun et al., 1998] orientiert. Sie besitzen also nur Convolutional Layer, Pooling Layer und einen Fully–connected Layer als letzten Layer, die Anordnung der Layer kann man in der Konzeptskizze von [Braham & Van Droogenbroeck, 2016] in Abbildung 4.6 sehen.

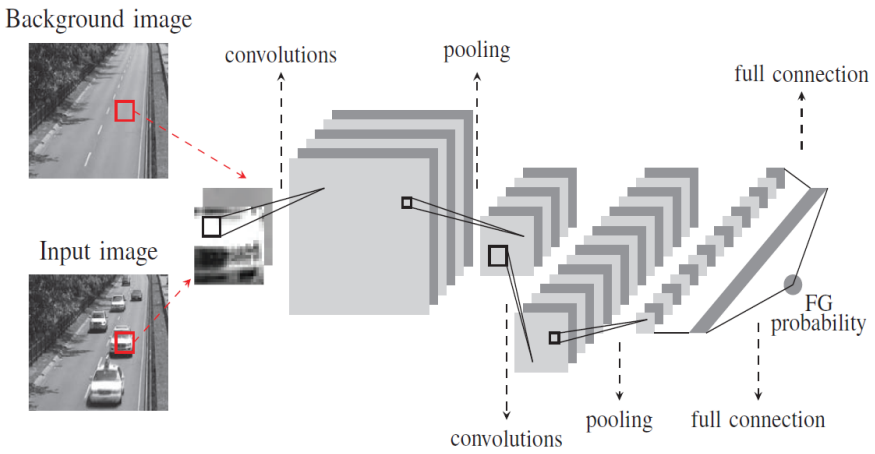

Abbildung 4.6: Konzept eines Basic CNN von [Braham & Van Droogenbroeck, 2016]

Dieser Kategorie wurden fünf Ansätze zugeordnet. Den Tabellen auf den Seiten 12 und 13 in [Bouwmans et al., 2019] lässt sich entnehmen, dass drei grundlegend schon ausgeschlossen werden können, da diese szenenspezifisch und damit nicht für eine bewegte Kamera geeignet sind, da sie bei einem Szenenwechsel erneut trainiert werden müssen, dies widerspricht dem ersten Bewertungskriterium.
Ein viertes Modell wird von [C. Zhao et al., 2018] beschrieben. Allerdings nutzt dieses mit der Random Permutation of Temporal Pixels, welches für jedes Pixel eine Verteilung über seine vorherigen Zustände ist, ein sehr spezielles Feature. Zusätzlich testen sie den Ansatz nur für den CDnet2014 Datensatz von [Y. Wang et al., 2014], welcher viele Daten für viele Probleme der Background Subtraction liefert, allerdings nur mit statischen Kamerasystemen. Daher ist nicht klar, ob dieser Ansatz auch für bewegte Kameras, wie in unserem Szenario, nutzbar ist. Die schlechteren Scores des Systems bei den Tests für Kamera–Zittern und dynamische Hintergründe lassen allerdings vermuten, dass sich dieser Ansatz weniger eignet.
Das letzte Konzept, welches in diese Kategorie eingeordnet wird, ist das von [X. Wang et al., 2018]. Es wirkt sehr vielversprechend, da die präsentierten Ergebnisse erahnen lassen, dass der Ansatz viele der Probleme, wie die Kamerabewegungen, in unserem Szenario lösen könnte. So ist er speziell für RGB–D–Bilder entwickelt. Auch ließ er eine gewisse Robustheit in Bezug auf eine bewegte Kamera erkennen. Allerdings wird

mit einer Titan X (Pascal) GPU mit 32 GB Videospeicher sehr leistungsfähige Hardware benötigt, welches dem fünften Bewertungskriterium widerspricht. Zusätzlich wird keine Aussage über die Laufzeit getroffen. Deshalb ist dieser Ansatz für einen Einsatz auf unserem Roboter eher ungeeignet.

MULTI–SCALE UND CASCADED CNNS

Als Cascaded CNNs werden Netzwerke bezeichnet, welche aus mehreren Netzwerken zusammen gesetzt werden. Der Ansatz von [Ang Lim & Yalim Keles, 2018] ist beispielsweise in Abbildung 4.7 zu sehen.

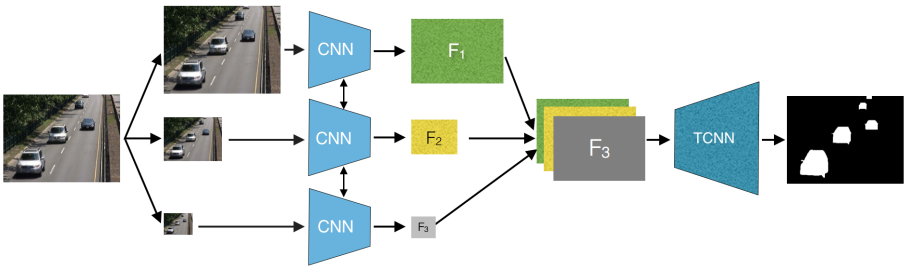

Abbildung 4.7: Architektur Triplet Convolutional Neural Network von [Ang Lim & Yalim Keles, 2018]

Multi–scale CNN zeichnen sich dadurch aus, dass sie in der Lage sind, eigenständig Multi–scale Feature zu extrahieren und zu lernen. Damit erreichen sie eine geringere Abhängigkeit von Eingangs- beziehungsweise Trainingsdaten dieser Art. So soll die Anfälligkeit gegenüber Kamerabewegungen reduziert werden.
Zu den Multi–scale und Cascaded CNNs zählen [Bouwmans et al., 2019] sieben Ansätze. Vier von diesen können direkt ausgeschlossen werden, da diese szenenspezifisch sind.
Die drei weiteren referenzierten Veröffentlichungen haben alle denselben Autor und wurden innerhalb eines Jahres veröffentlicht. Allerdings waren zwei davon zu diesem Zeitpunkt noch Preprints, von denen nur noch eines auffindbar war. Die Ansätze werden in [Ang Lim & Yalim Keles, 2018] und [L. A. Lim & Keles, 2019] beschrieben. Mit diesen erzielen sie sehr gute Ergebnisse für Datensätze wie dem bereits erwähntem CDnet2014 von [Y. Wang et al., 2014]. Allerdings wurden diese Datensätze nur mit statischen Kamerasystemen aufgenommen. Dabei enthalten sie zwar Szenen mit sehr dynamischen Hintergründen und Kamera–Zittern, allerdings keine Aufnahmen, die mit welchen von fahrenden Robotern mit $360°$ drehbarem Kameramodul vergleichbar sind. Zusätzlich wurde sich in der Veröffentlichung hauptsächlich auf die Moving Object Detection konzentriert. Gestürzte Personen können allerdings auch komplett bewegungslos sein, was dazu führen würde, dass sie fälschlich als Hintergrund gelten. Wie man dem zweiten Bewertungskriterium entnehmen kann, ist es für unser Szenario sehr wichtig dies zu vermeiden.

FULLY CNNS

Fully CNN zeichnen sich dadurch aus, dass sie keinen Fully–connected Layer wie die Basic CNNs haben. Für diese Kategorie haben [Bouwmans et al., 2019] sechs Ansätze aufgelistet. Einer davon ist nur szenenspezifisch und damit ungeeignet. [Bouwmans et al., 2019] konnte für drei weitere Ansätze die benötigten Eingangsdaten nur als RGB–Daten schätzen, da dies in den Veröffentlichungen nicht klar kommuniziert wird. Diese Ansätze werden nicht weiter betrachtet. Ein weiteres benötigt drei konkrete Vordergrundmasken als Eingangsdaten, was für unser Szenario nicht umsetzbar ist. Da wir ein ständig wechselndes Umfeld haben und es vermuten lässt, dass dieser Ansatz nur szenenspezifisch ist.
Das führt dazu, dass von den sechs Ansätzen nur noch der von [Zeng & Zhu, 2018] zu betrachten ist. Der Ansatz ist in ihrem Setting echtzeitfähig und läuft mit fast 27 fps. Allerdings nutzen die Autoren eine NVIDIA GTX 1060 GPU, welche leistungsfähiger als die Hardware im Roboter ist. Des Weiteren nutzen sie Infrarotbilder in ihrem Ansatz, was die Abschätzung schwieriger macht, wie gut der Algorithmus für RGB–D–Bilder geeignet ist. Zusätzlich dazu haben sie ihren Ansatz nur mit fünf Bildern der Thermalkategorie des bereits erwähnten Datensatz von [Y. Wang et al., 2014] getestet. So dass auch die Nutzbarkeit mit einer bewegten Kamera nicht abzuschätzen ist. Folglich ist es nicht abschätzbar, ob der Ansatz in unserem Szenario zuverlässige Ergebnisse liefert. Aufgrund der zusätzlich benötigten leistungsfähigen GPU ist er aber entsprechend des fünften Bewertungskriteriums eher ungeeignet.

DEEP CNNS

Unter Deep CNNs werden Netzwerke zusammen gefasst, bei denen die Ergebnisse des CNNs weiter verarbeitet werden und daraus das Ergebnis ermittelt wird. Die Weiterverarbeitung kann auf verschiedene Weisen erfolgen, beispielsweise durch ein klassisches Postprocessing, aber auch unter Nutzung weiterer neuronaler Netzwerke.
Zu den Deep CNNs zählen [Bouwmans et al., 2019] vier Ansätze. Einer ist als szenenspezifisch gekennzeichnet und damit für unser Szenario nicht geeignet.
Der zweite Ansatz ist von [X. Zhao et al., 2017]. Für ihre Experimente nutzen sie, wie viele der zuvor genannten Ansätze auch, den CDnet2014 Datensatz von [Y. Wang et al., 2014]. Beim Betrachten ihrer Ergebnisse fällt auf, dass der Ansatz für Szenen mit dynamischem Hintergrund anfälliger ist, was in unserem Szenario ein Problem ist, welches immer auftritt, wenn sich der Roboter bewegt. Der Ansatz läuft ihren Angaben zufolge mit 5 fps, was halb so schnell ist wie von uns angestrebt. Diese Geschwindigkeit wurde unter Nutzung einer NVIDIA GTX Titan X GPU erreicht, welche man eher im High End Segment einsortieren würde. Daher ist zu erwarten, dass der Ansatz in unserem Szenario langsamer ist und damit die Bildrate noch deutlicher unter den angestrebten 10 fps liegt. Entsprechend des vierten und fünften Bewertungskriteriums ist dieser Ansatz für unser Szenario somit ungeeignet.

Ein weiterer erwähnter Ansatz wird von [Li et al., 2017] beschrieben. Dieser ist auf Überwachungsszenarien spezialisiert. Dabei können sie diesen auf einen Kontext trainieren und für die Überwachung von statischen Szenen nutzen. Als Beispiel dafür haben sie die Detection von Fußgängern oder Fahrzeugen in Aufnahmen von fest installierten Kameras genutzt. Die Tests für den Algorithmus wurden in der Veröffentlichung mit Datensätzen für Fußgänger Detektion und Fahrzeug Detektion durchgeführt. Dabei hat der Algorithmus gute Ergebnisse erzielt. Allerdings ist er für unser Szenario nicht geeignet, da der Ansatz für die Erfüllung des zweiten Bewertungskriteriums in der Lage sein muss, regungslose gestürzte Personen als Vordergrund zu kennzeichnen.
Der letzte Ansatz in dieser Kategorie ist von [Chen et al., 2017]. Dieser Ansatz wurde für eine Moving Object Detektion entwickelt. Als Daten nutzen sie den CDnet2014 Datensatz von [Y. Wang et al., 2014] und den LASIESTA Datensatz von [Cuevas et al., 2016]. Der zweite von den beiden ist speziell für die Moving Object Detektion erstellt. Besonders ist auch, dass er Daten hat, die mit bewegten Kameras aufgenommen wurden. Insgesamt wurden von [Chen et al., 2017] drei aufeinander aufbauende Ansätze entwickelt und getestet. Diese weisen auf ihren Validierungsdaten sehr gute Ergebnisse auf und haben nur eine geringe Erhöhung der Ungenauigkeit durch Kamerabewegungen. Der grundlegende Ansatz, welcher gute Ergebnisse liefert, hat eine Durchschnittslaufzeit von 30 ms auf ihrem System mit einer GTX Titan X GPU. Diese Laufzeit ist sehr klein, so dass die Chance bestehen würde, dass der Algorithmus auch auf unserem Roboter laufen könnte. Die zwei erweiterten Ansätze benötigen auf der leistungsfähigen Hardware, die sie nutzen, schon ungefähr 70 ms und 133 ms und sind für unsere Rahmenbedingungen ungeeignet. Zudem handelt es sich um eine Moving Object Detektion. Nicht alle gestürzten Personen bewegen sich, sie würden somit als Hintergrund klassifiziert werden und von dem Klassifikator der GPE gar nicht betrachtet werden. Daraus resultiert ein Risiko bewegungslose gestürzte Personen zu übersehen, welche dringend Hilfe benötigen. Damit widerspricht der Ansatz dem zweiten Bewertungskriterium und ist für unser Szenario nicht geeignet.

STRUCTURED CNNS

Der Ansatz des Structured CNN zeichnet sich dadurch aus, dass die Foreground Extraction getrennt vom CNN durchgeführt wird. Zusätzlich wird ein Hintergrundmodell verwaltet und aktualisiert, welches im CNN auch zu den Eingangsdaten gehört. Ein Konzept für Netzwerke dieser Kategorie ist in Abbildung 4.8 zu sehen.

Als Structured CNN wird von [Bouwmans et al., 2019] nur der Ansatz von [K. Lim et al., 2017] gezählt. Bei diesem werden wieder Daten aus dem CDnet 2014 Datensatz von [Y. Wang et al., 2014] genutzt. Ihr Ziel ist dabei eine Change Detection. Damit eignet sich dieser Ansatz für unser Szenario nicht, da es eine starke Veränderungen der Szene gibt, sobald der Roboter fährt oder sein Kameramodul dreht. Zum anderen besteht dabei dasselbe Problem wie bei der Moving Object Detektion, dass regungslose gestürzte Per-

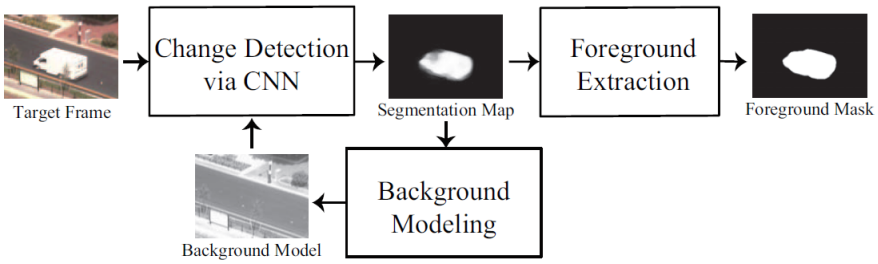

Abbildung 4.8: Konzept eines Structured CNN von [K. Lim et al., 2017]

sonen nicht detektiert werden können. Es bestehen also Widersprüche zum ersten und zweiten Bewertungskriterium.

3D CNNS

Bei 3D CNNs werden in den Convolutional Layern 3D Convolutions, anstelle der sonst in CNN eher genutzten 2D Convolutions, ausgeführt. Zu dieser Kategorie zählt [Bouwmans et al., 2019] die Ansätze von [Sakkos et al., n. d.], [Yu et al., 2018] und [Akilan, 2018].
In [Sakkos et al., n. d.] wird eine Background Subtraction für eine Moving Object Detection entwickelt. Diese liefert gute Ergebnisse für RGB–Bilder. Allerdings besteht, wie schon mehrfach erwähnt, das Problem, dass bei Ansätzen, die dafür konzipiert sind, regungslose gestürzte Personen als Hintergrund gelten. Daher ist dieser aufgrund des Widerspruches zum zweiten Bewertungskriterium für uns nicht geeignet.
Von [Yu et al., 2018] wird ein Ansatz zur Detektion relevanter Bewegung in Heimüberwachungsvideos umgesetzt. Dieser ist sehr schnell, auf einer Grafikkarte kann er ein 15 Sekunden Video in 4-8 ms analysieren, auf einer CPU werden $0,17 - 0,39$ s benötigt. Allerdings wird bei diesem Ansatz immer das komplette Video betrachtet, also in ihrem Szenario 15 Sekunden Aufnahmen. Für unser Szenario ist dieser Ansatz daher aus mehreren Gründen ungeeignet. Er ist nur für statische Überwachungskameras und nur zur Detektion von Bewegungen, was, wie bereits mehrfach erwähnt, problematisch ist. Zusätzlich könnte das Problem mit den 15 Sekunden Videos für Probleme sorgen. Wenn die Person zu Beginn des Videos detektiert wird, kann der Roboter aufgrund seiner Höchstgeschwindigkeit von 1 m/s zum Zeitpunkt der Analyse ungefähr 15 Meter von der Person entfernt sein.
Der letzte Ansatz in dieser Kategorie ist von [Akilan, 2018]. Dieser Ansatz ist für eine Moving Object Detektion, was, wie bereits beschrieben, gegen das zweite Bewertungskriterium verstößt und ihn somit nicht nutzbar macht.

GANS

Die letzte Kategorie von [Bouwmans et al., 2019] sind GANs. In diese Kategorie haben sie vier Ansätze eingeordnet. Allerdings haben sie zu zwei Ansätzen davon sehr wenige

Informationen gefunden, daher werden nur die zwei Ansätze von [Bakkay et al., 2018] und [Bahri et al., 2018] betrachtet.
Der Ansatz von [Bakkay et al., 2018] läuft mit 10 fps auf einem Intel i7 CPU, was unsere Bedingungen erfüllt. Gleichzeitig weist es für die klassischen Probleme wie Kamera–Zittern und dynamischen Hintergrund sehr gute Ergebnisse auf. Die Robustheit in Bezug auf eine bewegte Kamera oder unbewegte beziehungsweise wenig bewegte Vordergrundobjekte ist allerdings nicht ersichtlich. Die Nutzung des CDnet2014 Datensatz und der geplante zukünftige Einsatz in einem Projekt für sichere Straßenquerungen lässt allerdings eher auf ein statisches Einsatzszenario und einen Fokus auf eine Moving Object Detection vermuten. Daher ist er für unser Szenario entsprechend des ersten und zweiten Bewertungskriteriums eher ungeeignet.
Bei dem Ansatz von [Bahri et al., 2018] ist das Ziel eine Moving Object Detektion zu entwickeln. Dabei besteht, wie bereits mehrfach zuvor erwähnt, das Problem, dass in unserem Szenario gestürzte Personen Vordergrund Objekte sein sollen. Damit besteht ein Widerspruch zum zweiten Bewertungskriterium. Aus diesem Grund ist dieser Ansatz ungeeignet für unser Szenario.

4.4.4 GEGENÜBERSTELLUNG/ZUSAMMENFASSUNG

Zur besseren Veranschaulichung der Ergebnisse der Recherche werden alle Ansätze in Tabellenform gegenüber gestellt. Die klassischen Ansätze werden in der Tabelle A.1 und die Deep–Learning Ansätze in der Tabelle A.2 im Anhang aufgelistet. Dabei wird die Eignung, beziehungsweise Probleme, der Ansätze in Bezug auf die Bewertungskriterien aus Abschnitt 4.4.1 gezeigt. Zur besseren Visualisierung wird eine Farbkodierierung genutzt. Rot markierte Zellen sind kritische Probleme, die gegen das entsprechende Kriterium klar verstoßen. Ein Beispiel dafür ist, dass Zellen, welche für die Nutzbarkeit bewegter Kameras ein „nein“ enthalten, rot markiert sind, da dies ein kritisches Problem ist, welches den Ansatz für uns nicht nutzbar macht.
Orange markierte Zellen markieren Probleme, welche eventuell lösbar sind, aber auch zu kritischen Problemen führen könnten. Beispiele dafür können die Nutzung von RGB–Bildern sein oder die unklare Nutzbarkeit für bewegte Kameras.
Zellen, die Kriterien betreffen, welche bei der Betrachtung der Absätze in den beiden vorherigen Abschnitten für diesen Ansatz nicht erwähnt wurden, werden mit einem „-“ markiert. Das unerwähnt lassen dieser Kriterien kann darauf zurückzuführen sein, dass in der ursprünglichen Veröffentlichung dazu keine Informationen enthalten waren oder dass der Ansatz aufgrund eines kritischen Verstoßes gegen ein anderes Kriterium nicht weiter betrachtet wurde.
Beim Betrachten der Tabellen fällt auf, dass sehr viele der Ansätze Probleme mit bewegten Kameras haben. Eine Ursache dafür ist, dass sehr viele Ansätze nur für Überwachungsszenarien mit statischen Kameras bestimmt sind oder sie lediglich den CDnet 2014 Datensatz von [Y. Wang et al., 2014] nutzen. Auffällig ist auch, dass sehr viele der

Deep–Learning Ansätze explizit für die Moving Object Detection gedacht sind und damit nur die bewegten Personen in unseren Aufnahmen erkennen können. Es gibt nur wenige Ansätze, die explizit für RGB–D–Bilder konzipiert worden sind, wobei eine Anpassung dafür laut [Bouwmans et al., 2019] für viele der Deep–Learning Ansätze möglich wäre. Zur Laufzeit der Ansätze ist festzustellen, dass es besonders bei den Deep–Learning Ansätzen einige gibt, welche auf unserer Hardware wahrscheinlich nicht echtzeitlauffähig sind.
Bei einer genaueren Betrachtung der Tabellen fällt auf, dass bei den Minimum Background Ansätzen die wenigsten kritischen Probleme zu finden sind. Der widerspruchsloseste in dieser Kategorie ist der von [Antonello et al., 2017].

4.5 ENTSCHEIDUNG FÜR DIE UMSETZUNG

Auf Basis der Ergebnisse der vorherigen Recherche ist eine Entscheidung zu treffen. Dabei ist es wichtig einen Ansatz zu wählen, der möglichst wenige Reibungspunkte mit unseren festgelegten Kriterien hat. Diese waren die Eignung für den Einsatz auf einem mobilen Roboter, eine geringe Wahrscheinlichkeit dafür, eine gestürzte Person als Hintergrund zu klassifizieren, die Fähigkeit RGB–D–Daten zu nutzen und eine Lauffähigkeit mit mindestens 10 fps, ohne dafür besondere Hardware zu benötigen.
Eine der Feststellungen in der Gegenüberstellung war, dass Ansätze der Kategorie Minimum Background die wenigsten kritischen Punkte haben. Die Ansätze sind für RGB–D–Daten ausgelegt. Zusätzlich haben sie eine gewisse Ähnlichkeit mit der Vorverarbeitung des Ansatzes von [Antonello et al., 2017] für die GPE. Daraus lässt sich schließen, dass solche Ansätze für unser Szenario mit einem mobilen Roboter anwendbar sind. Zusätzlich ist eine Entfernung von Wänden, Decke und Boden, wie sie von [Fengliang Xu & Kikuo Fujimura, 2003] vorgeschlagen wird, in unserem Szenario sehr günstig. In vielen Fällen wird dies der einzige Hintergrund sein, da auf den Gängen andere Objekte eher selten sind. Damit sollte es nicht möglich sein, eine Person als Hintergrundobjekt zu klassifizieren, da sie sich in der Form sehr von Ebenen wie Wänden unterscheidet. Zusätzlich wird in keinem der Ansätze besondere Hardware, wie High End Grafikkarten, erwähnt und [Antonello et al., 2017] zeigt, dass Ansätze dieser Art echtzeitfähig sein können.
Ein zweiter Ansatz, der vielversprechend wirkt, sind die GANs. Allerdings wurden die dabei betrachteten Ansätze eher als Moving Object Detection konzipiert und sind deshalb nicht sehr geeignet für Szenen, in denen die gestürzten Personen ruhig liegen.
Aus diesen Gründen wird für die Umsetzung der Ansatz des Minimum Background gewählt.

4.6 UMSETZUNG

Basierend auf der Entscheidung für die Umsetzung eines Minimum Background Ansatzes soll es in diesem Teil um die Implementation und Konzeptionierung eines Ansatzes gehen. Dazu werden die Ansätze von [Antonello et al., 2017] und [Fengliang Xu & Kikuo Fujimura, 2003] zur Orientierung genutzt. Daraus ergeben sich für uns die fünf in Abbildung 4.9 gezeigten Arbeitsschritte. Dabei können die letzten beiden Schritte als Filter–Schritt zusammengefasst werden.

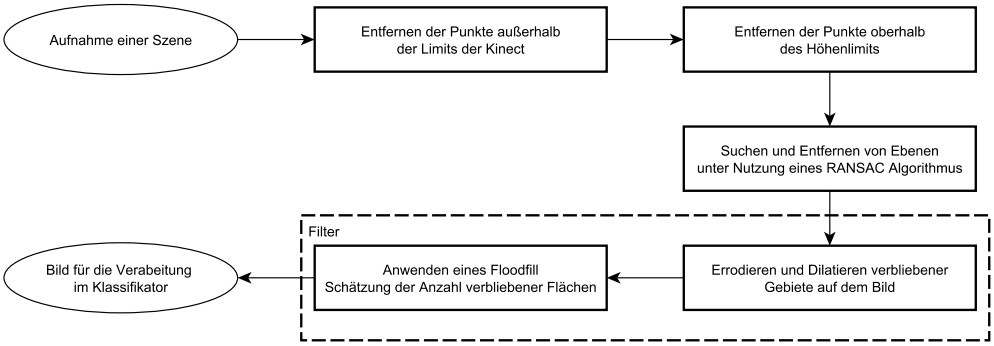

Abbildung 4.9: Skizze Implementation der Backgroundsubtraction

Zum Datentransport wird im Projekt ein Containersystem genutzt, in welchem die Daten mit einem Zeitstempel gespeichert werden. Dies bietet den Vorteil, dass für die Daten bekannt ist, wie aktuell sie sind und somit auch welches die aktuellsten Daten sind. Dieses System, wie es auch im Klassifikator genutzt wird, soll zum Transport der Bild–Daten in der Background Subtraction benutzt werden. Dies hat den Vorteil, dass einzelne Arbeitsschritte in sogenannten Syncprozessoren abgearbeitet werden können. Ein Syncprozessor überwacht einen oder mehrere Container, und wenn in diesen Container neue Daten geschrieben werden, reagiert er und führt mit den Daten eine Aktion durch. In der Umsetzung soll jeder der fünf in Abbildung 4.9 gezeigten Schritte in einem eigenen Syncprozessor umgesetzt werden. Dies bietet den Vorteil, dass einzelne Schritte im Nachhinein leichter ausgetauscht oder angepasst werden können, falls sich effizientere Algorithmen oder Implementationen ergeben. Es muss dadurch nicht so sehr darauf geachtet werden, dass die komplette Background Subtraction in weniger als 100 ms erfolgt, sondern nur die Berechnung der einzelnen Schritte weniger als 100 ms benötigt. Dies wird dadurch möglich, dass die Syncprozessoren parallel arbeiten können und so beispielsweise der erste Prozessor schon ein neues Bild betrachten kann, während die anderen noch ältere Bilder bearbeiten. Ein weiterer Vorteil dieses Vorgehens ist ein übersichtlicherer Gesamtablauf. Diesen Vorteilen steht der Nachteil einer erhöhten Datenmenge gegenüber, der aber durch die Vorteile aufgewogen wird.
Der erste Schritt aus Abbildung 4.9 wird im Abschnitt 4.6.1 genauer beschrieben.
Im zweiten Schritt werden Daten aussortiert, welche oberhalb von 0,7 Meter liegen und

damit nicht im Suchbereich für gestürzte Personen auf Fluren in der Nacht sind. Die Umsetzung dieses Schrittes wird in Abschnitt 4.6.2 beschrieben.
Im nachfolgenden Schritt werden mithilfe eines RANSAC–Algorithmus Ebenen in den Daten gesucht und anschließend entfernt. Dadurch sollen Wände, die Bodenebene und mögliche Objekte in den Gängen, welche von ihrer Form her keine gestürzte Person sein können, entfernt werden. Auf die Implementation dieses Schrittes wird in Abschnitt 4.6.3 eingegangen.
Im Anschluss daran sollen in den letzten beiden Schritten die verbliebenen Daten so gefiltert werden, dass Punkte mit wenigen Nachbarn beziehungsweise kleine Gebiete entfernt werden. Diese Schritte werden in Abschnitt 4.6.4 beschrieben.
Nachdem die Arbeitsschritte feststehen, wird ein Konzept zur Visualisierung der Ergebnisse dieser Schritte benötigt. Darauf wird in Abschnitt 4.6.5 eingegangen.

4.6.1 LIMITS DER KINECT

Die Limits des im Roboter genutzten Kameramoduls liegen, wie bereits in den Rahmenbedingungen (Abschnitt 4.3) erwähnt, bei 0,5 Metern als Minimaldistanz. Für die Maximaldistanz gibt es verschiedene Angaben. Diese bewegen sich in einem Raum von 5 Metern, wie es in [Greff et al., 2012] angegeben wird, bis hin zu 8 Metern, welche laut [Ashley, 2014] die Maximaldistanz ist, für welche die Sensoren Daten liefern. Um einen Kompromiss aus diesen Angaben, dem Szenario und eigenen Beobachtungen zu finden, beschränken wir die Tiefe auf 6,5 Meter, welches die Mitte zwischen der größten und kleinsten Angabe ist. Daraus folgend empfiehlt es sich, Daten, welche außerhalb dieses Bereiches liegen, nicht zu berücksichtigen.
Die Umsetzung dieses Schrittes ist daher recht simpel. Es wird für jedes Pixel betrachtet, welchen Distanzwert es hat und wenn dieser außerhalb dieser Limits liegt, wird er verworfen, indem er mit einem Dummy–Wert überschrieben wird, mit welchem er in folgenden Schritten identifiziert werden kann.
Dieser Schritt ist nicht so aufwendig, dass dafür ein eigener Syncprozessor genutzt werden muss. Dadurch ist es später einfacher, den Algorithmus für ein Kameramodul mit anderen Limits anzupassen, da dazu nur dieser Syncprozessor ersetzt werden muss und die restlichen Schritte davon unberührt bleiben.
In Abbildung 4.10 kann man ein Beispiel für die Arbeit des Moduls sehen. Dazu sind links die Rohdaten und rechts das Ergebnis dieses Schrittes abgebildet. Dabei fällt auf, dass nur einzelne Pixel betroffen sind.

4.6.2 HÖHENLIMIT

Da gestürzte Personen nicht in größeren Höhen innerhalb eines Bildes zu erwarten sind, bietet es sich an, ein Höhenlimit einzuführen. Dieser Schritt wird auch von [Antonello et al., 2017] vorgeschlagen, welche einen Grenzwert von 0,7 Metern nutzen. Da oberhalb dieses Wertes keine gestürzten Personen zu erwarten sind, wird dieser Grenzwert auch

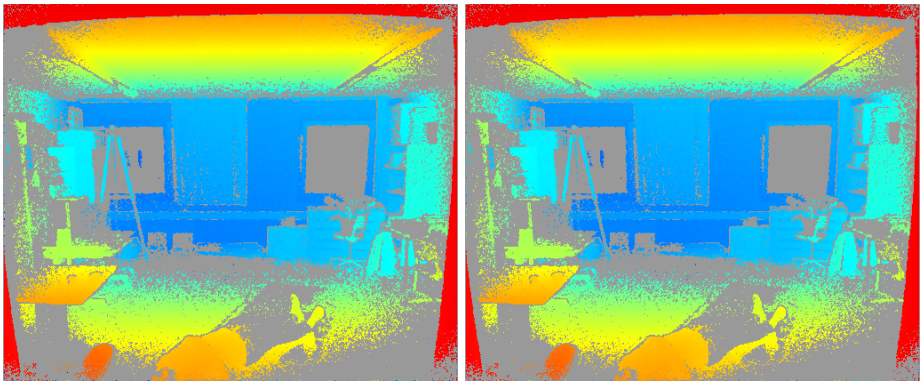

Abbildung 4.10: Beispiel für die Kincet Limits

für unsere Umsetzung genutzt. Im Vergleich zum vorherigen Schritt ist in diesem die Umsetzung aufwendiger.
Mithilfe des implementierten Kameramodells für unsere RGB–D–Daten, welches für jedes Pixel einen Richtungsnormalenvektor gespeichert hat, ist es möglich, für jedes Pixel einen Punkt im 3D Raum zu errechnen. Die Punkte werden durch die Skalarmultiplikation des Richtungsnormalenvektoren mit den Distanzwerten der gültigen Pixel berechnet.
Allerdings ist dieses Koordinatensystem, in dem die Punkte liegen, noch nicht geeignet, ein Höhenlimit umzusetzen, da der Koordinatenursprung im Zentrum der Kamera liegt und durch die Neigung des Kameramoduls um die X–Achse rotiert ist. Dieses Problem ist in Abbildung 4.11 skizziert.

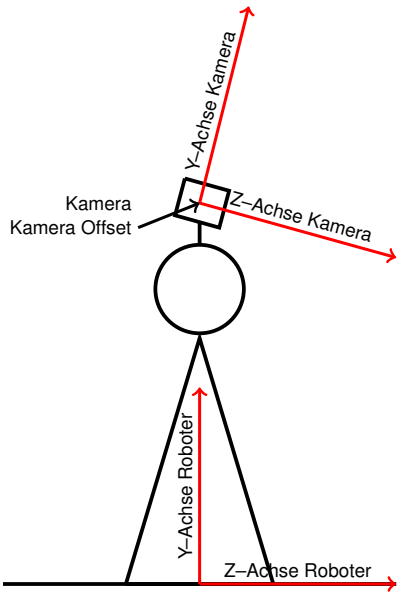

Abbildung 4.11: Skizze Koordinatensysteme von Roboter und Kamera

Zusätzlich liegt auch noch eine Rotation um die Y-Achse vor, welche in diesem Koordinatensystem die Höhe abbildet. Diese kann allerdings vernachlässigt werden, da diese keinen Einfluss auf die Höhe eines Punktes hat.

Daraus ergibt sich, dass die Punkte entsprechend dem Neigungswinkel des Kameramoduls um die X-Achse rotiert werden müssen. Im Anschluss daran muss noch der Offset des Kameramoduls, also dessen Position am Roboter, berücksichtigt werden.
Nach diesen beiden Berechnungen können die Punkte, bei welchen ein Höhenlimit von 0.7 Metern an der Y-Achse überschritten wird, aussortiert werden.
Ein Beispiel wie sich dieser Schritt auf die Ergebnisse auswirkt, ist in Abbildung 4.12 zu sehen. Dazu sind links die Eingangsdaten dieses Schrittes und rechts seine Ergebnisse abgebildet. Man erkennt, dass in diesem Schritt für gewöhnlich sehr viele Punkte aussortiert werden.

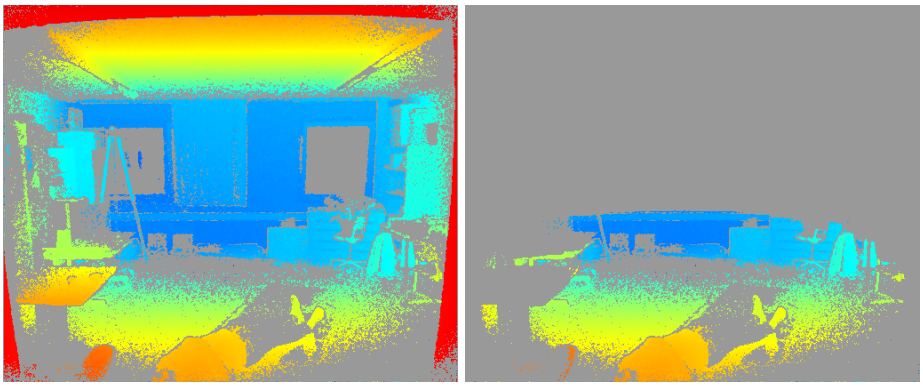

Abbildung 4.12: Beispiel für das Höhenlimit

4.6.3 RANSAC

In diesem Schritt sollen die Bodenebene und Objekte, welche große Ebenen enthalten und somit nicht Teil einer Person sein können, entfernt werden. Dazu bietet sich ein RANSAC–Algorithmus an. Dieser wird klassisch dafür genutzt, ein Modell für Daten bestmöglich anzupassen. Dem Modell entsprechend werden dafür ein oder mehrere Punkte zufällig aus den Daten ausgewählt. Mit diesen Punkten wird das Modell dann angepasst. Im nächsten Schritt wird die Unterstützung des Modells berechnet, also wie viele Punkte zu dem Modell passen. Dabei kann auch eine gewisse Fehlertoleranz beachtet werden, also Punkte mit einer Abweichung, die kleiner als ein Grenzwert ist, werden als passend gewertet. Diese 3 Schritte werden dann n-mal wiederholt und das Modell mit der höchsten Unterstützung gespeichert.
Für unseren Anwendungsfall bietet sich die Hessesche Normalform für Ebenen als Modell an, wie es in Abbildung 4.13 skizziert ist. Da mit dieser der Fehlerwert, also Abstand eines Punktes zur Ebene, sehr einfach und einheitlich berechnet werden kann.
Für die Konstruktion des Modells werden drei Punkte benötigt. Allerdings sollten diese drei nicht vollkommen zufällig gewählt werden. Zum einen muss sichergestellt werden, dass aus den drei Punkten eine Hessesche Normalform berechnet werden kann. Dazu müssen es drei verschiedene Punkte sein und sie dürfen nicht auf einer Geraden liegen. Zusätzlich sollte nur ein Punkt komplett zufällig gewählt sein, um möglichst sinnvolle

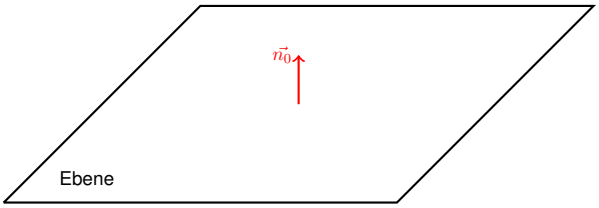

Abbildung 4.13: Skizze Hessesche Normalform

Ebenen konstruieren zu können. Sind alle drei Punkte komplett zufällig, kann es passieren, dass diese in mehreren zu entfernenden Ebenen im Raum verteilt sind und eine höhere Wahrscheinlichkeit für eher schlechte Modelle besteht. Daher wird nur ein Punkt vollkommen zufällig gewählt. Die beiden anderen werden unter Berücksichtigung der anderen Bedingungen so gewählt, dass sie in einem bestimmten Radius um den ersten Punkt liegen, sofern dies möglich ist. In unserer Umsetzung ist dieser Radius mit 0,25 gewählt, da dieser die Wahrscheinlichkeit erhöht, auch Ebenen im kleineren Rahmen zu finden, wie beispielsweise Schränke.
Die Tiefenbilder besitzen eine Auflösung von 512×424 Pixeln. Daraus ergeben sich im Maximalfall 217 088 Punkte. Dies ist eine sehr große Zahl und würde wahrscheinlich dazu führen, dass die Laufzeitziele nicht erreichbar sind. Aus diesem Grund wird von [Kurban et al., 2015] vorgeschlagen die Auflösung des Bildes zu reduzieren. In unserem Fall wird dies dadurch erreicht, dass wir in der unteren rechte Ecke des Bildes starten und aus 3×3 Pixelfeldern nur den mittleren Punkt zur Punktwolke hinzufügen, der Schritt ist in Abbildung 4.14 skizziert.

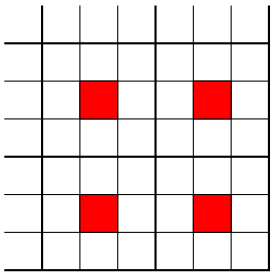

Abbildung 4.14: Skizze Reduzierung der Punkte

Als Nächstes ist es wichtig die Implementation des Algorithmus zu betrachten. Von [Kurban et al., 2015] wird dafür eine Implementation aus der Point Cloud Library von [Rusu & Cousins, 2011] genutzt. Diese ist allerdings in C++ implementiert und bei der von uns angestrebten Java Implementation schwieriger zu integrieren. Daher war es notwendig, eine Alternative zu finden, welche mit OpenIMAJ von [Hare et al., 2011] gefunden wurde. Diese bietet ein Grundgerüst für eine RANSAC–Implementation, welche durch eigene Modelle und Fehlermodelle ergänzt wird und so sehr gut auf das Szenario angepasst werden kann.

Nachdem feststeht, wie die Implementation erfolgen soll, wird es notwendig, den letzten Parameter des RANSAC festlegen. Dabei handelt es sich um die Anzahl der Wiederholungen der zufälligen Auswahlschritte. Dieser wurde von [Kurban et al., 2015] auf 50 gesetzt, für unsere Implementation haben wir uns hingegen für den Wert von 100 entschieden, da dieser beim Testen zuverlässigere Ergebnisse geliefert hat.
Mit feststehenden Rahmenbedingungen für den RANSAC–Algorithmus kann der Aufbau des Syncprozessors für diesen Schritt betrachtet werden. Für diesen wird im ersten Schritt eine Punktwolke, wie beschrieben, erzeugt. Darauf wird anschließend der RANSAC–Algorithmus angewendet und die gefundenen Ebenen entfernt, solange die Unterstützung des gefundenen Modells oberhalb eines Grenzwertes liegt. Der Grenzwert wird in unserer Implementation relativ bestimmt, er hängt also von der Größe der Punktwolke ab. Er wird so lange angewendet bis die Unterstützung größer als $\frac{1}{10}$ der Punktwolke ist. Um eine sehr häufige Wiederholung zu verhindern und eine gewisse verbleibende Anzahl an Punkten garantieren zu können, wird eine Mindestunterstützung festgelegt, bei welcher die Wiederholung abgebrochen wird. Dieser Wert ist auf 50 definiert.
Ein Beispiel für die Arbeit des RANSAC–Algorithmus kann man in Abbildung 4.15 sehen. Dazu sind links die Eingangsdaten dieses Schrittes und rechts die Ergebnisse abgebildet. Dabei erkennt man, dass sehr viele Punkte innerhalb der Ebene aus der Punktwolke entfernt werden, aber weiterhin vereinzelte Punkte der Ebene, welche nicht ausreichend von dem mit dem RANSAC berechneten Modell unterstützt werden, erhalten bleiben.

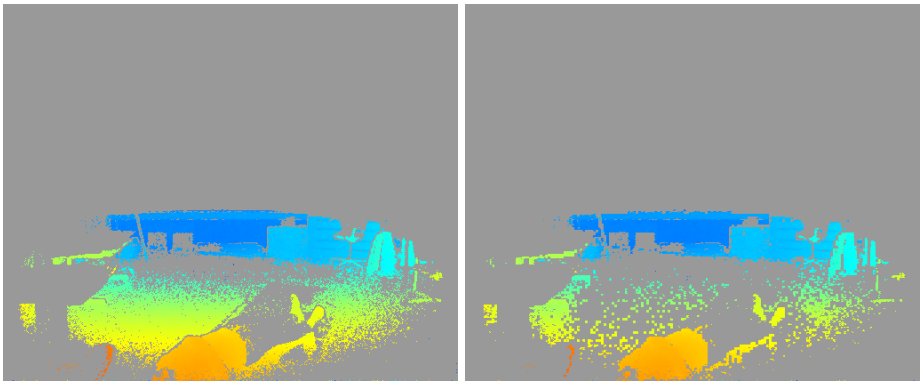

Abbildung 4.15: Beispiel für den RANSAC

4.6.4 FILTER

Der Filter ist teilweise von der bisherigen Lösung mit Differenzbildern, wie sie auch in [Brose, Erzgräber et al., 2019] beschrieben wird, übernommen. Aus Gründen der Laufzeit der einzelnen Schritte wird dieser Schritt auf 2 Syncprozessoren aufgeteilt.
Im ersten Syncprozessor wird aus den Ergebnissen des vorherigen Schrittes eine Boolean–Gridmap erstellt. Eine Zelle in dieser hat den Wert $true$, wenn das Pixel zum Vordergrund zählt, und $false$, wenn sie zum Hintergrund zählt. Im Anschluss daran werden die positiven Flächen auf der Karte sechsmal erodiert. Dabei werden alle Zellen mit dem Wert $true$

auf $false$ gesetzt, welche nicht mindestens fünf Nachbarn in einer Moore–Nachbarschaft haben, die auch den Wert $true$ haben. Die Ergebniskarte wird dann zweimal dilatiert. Dazu wird für alle Zellen mit $false$ Wert geprüft, ob sie mindestens fünf Nachbarn mit $true$ als Wert haben. Gilt das, wird der Wert der Zelle auf $true$ gesetzt. Im Anschluss daran wird für alle Zellen mit $true$ als Wert geprüft, ob es einen gültigen Wert im RGB–D–Bild gibt, falls nicht wird die Zelle auf $false$ gesetzt.
Im zweiten Syncprozessor wird auf die Karte aus dem ersten Schritt noch ein Floodfill–Algorithmus angewendet, um Flächen aus Zellen mit $true$ Werten auf der Karte zu finden und Flächen unter einer bestimmten Größe zu entfernen. Im Anschluss wird auf Basis eines Floodfills die Zahl der verbliebenen Flächen ermittelt. Diese kann im Klassifakator als Schätzwert für die Anzahl an Clustern genutzt werden.
In Abbildung 4.16 ist ein Beispiel für die Arbeit des Filters abgebildet. Dazu sind links die Eingangsdaten für den Filter und rechts die Ergebnisse, welche auch dem Ergebnis der gesamten Background Subtraction entsprechen, abgebildet. In diesem Beispiel sieht man wie die vielen einzelnen Punkte erfolgreich entfernt werden, ohne dass die größeren Flächen davon betroffen sind, so erkennt man im vorderen Bereich die erfolgreich extrahirte Person, während im hinteren Bereich kleinere Objekte mit unregelmäßigen Oberflächen erhalten sind.

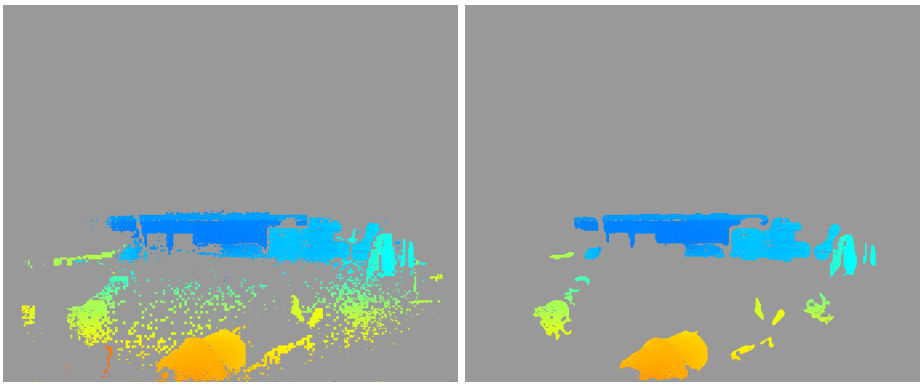

Abbildung 4.16: Beispiel für den Filter

4.6.5 VISUALISIERUNG DER ERGEBNISSE

Bei der Visualisierung gibt es zwei Punkte, die zu betrachten sind. Der Erste wäre, die Ergebnisse der Background Subtraction und ihren einzelnen Schritte erkennbar zu machen. Das ist im Rahmen dieser Arbeit wichtig und nützlich, im Kontext der gesamten GPE allerdings eher ein Debugging Tool. Der Zweite ist im Kontext der GPE wichtiger. Dabei sollen im Anschluss an die Klassifikation die einzelnen Cluster auf dem Bild markiert werden können. Dies ist erstrebenswert, da es hilfreich seien kann, wenn der Bildbereich, in welchem die gestürzte Person ist, hervorgehoben werden kann. So wird es für das Personal einfacher, die gestürzte Person auf dem Bild zu erkennen.
Beide Punkte können durch einen Ansatz gelöst werden. Dieser besteht darin eine Da-

tenstruktur einzuführen, welche eine Verbindung zwischen den 3D Koordinaten und den Pixel–Koordinaten herstellt. Mit dieser Datenstruktur kann die Visualisierung der Daten der Punktwolke für die einzelnen Schritte wieder als Tiefenbild rekonstruiert werden. Dieses kann dann zur Visualisierung der Ergebnisse genutzt werden.
Für die Lokalisierung der Cluster in den Bildern eignet sich diese Methode auch, da damit für jedes Cluster die Grenzen ermittelt werden können und somit der zu markierende Bereich.

5 ERGEBNISSE

In diesem Kapitel werden die Ergebnisse der Umsetzung der Background Subtraction betrachtet und bewertet. Im Abschnitt 5.1 wird die Laufzeit der einzelnen Schritte der Umsetzung betrachtet. Daran schließt sich die Betrachtung von Metriken zum Vergleich der Umsetzung mit anderen Ansätzen an. Dem schließt sich in Abschnitt 5.3 ein Vergleich mit dem idealen Ergebnis an. Als Nächstes wird im Abschnitt 5.4 der neue Ansatz und dem bisherigen Ansatz verglichen. Zum Abschluss wird ein Fazit mit den Stärken und Schwächen der Umsetzung in Abschnitt 5.5 gezogen.

5.1 LAUFZEIT

Damit die Umsetzung die Rahmenbedingungen für die Laufzeit erfüllt, welche in Abschnitt 4.3 festgelegt wurden, müssen die einzelnen Schritte in weniger als 100 ms abgeschlossen werden. Um dieses zu überwachen, wurde für jeden Schritt die Zeit gespeichert, während ein Datensatz mit 935 Bildern betrachtet wurde. Die Ergebnisse sind in Abbildung 5.1 zu sehen.

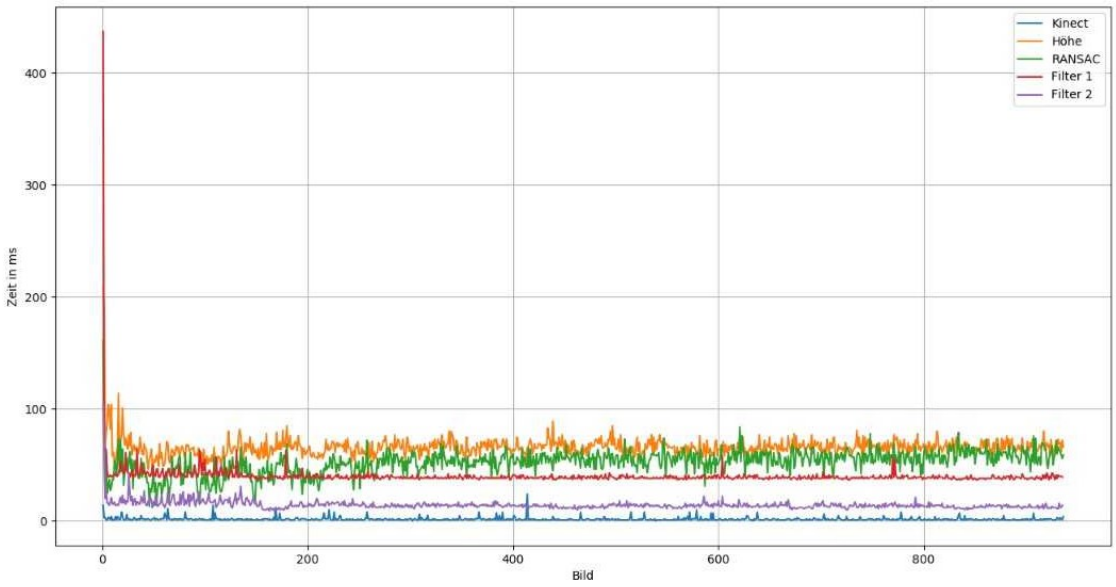

Abbildung 5.1: Laufzeiten der einzelnen Schritte

Da es anhand eines Diagramms nicht so einfach ist Durchschnittswerte und Extremwerte abzulesen, werden diese in Tabelle 5.1 zusammengefasst.

	Minimale Laufzeit	Maximale Laufzeit	Durchschnitte Laufzeit
Kinect	0 ms	24 ms	1,71 ms
Höhe	46 ms	208 ms	65,88 ms
RANSAC	13 ms	161 ms	52,42 ms
Filter 1	37 ms	437 ms	40,1 ms
Filter 2	8 ms	104 ms	13,32 ms

Tabelle 5.1: Tabelle mit Laufzeitgrenzwerten

Wenn man die Ergebnisse betrachtet, fällt auf, dass bis auf sehr wenige Ausnahmen das Laufzeitziel von weniger als 100 ms pro Schritt erreicht wird. Die durchschnittliche Laufzeit liegt in jedem Schritt deutlich unter der anvisierten Zeit von 100 ms. Lediglich bei Filter 1 kommt es im ersten Durchlauf zu einer sehr starken Abweichung von diesem Zielwert. Dies hängt allerdings damit zusammen, dass in diesem Schritt der Speicherplatz für die Gridmaps allokiert wird. Dabei handelt es sich um einen einmaligen Vorgang zu Beginn, weshalb dies für die gesamte Betrachtung nicht von Bedeutung sein sollte. Erkennbar ist auch, dass die Kinect Limits eine sehr geringe Laufzeit haben, so dass sie nicht notwendigerweise einen eigenen Syncprozessor benötigt hätten. Die Entscheidung dafür wurde getroffen, da so der Code besser zu warten ist. So ist es beispielsweise möglich, das Kameramodul auszutauschen, und es müssten nur die Limits angepasst werden, die restliche Implementation würde unberührt bleiben.

5.2 METRIKEN UND VERGLEICH MIT ANDEREN ANSÄTZEN

Um die Umsetzung besser bewerten und mit anderen vergleichen zu können, werden Metriken benötigt. Eine weitverbreitete Metrik ist der F_1–Score. Dieser wird von einigen der im Abschnitt 4.4 betrachteten Ansätzen genutzt. Dafür wird für jedes Pixel betrachtet, ob dieses richtig als Vordergrund (TP), falsch als Vordergrund (FP), falsch als Hintergrund (FN) oder richtig als Hintergrund (TN) klassifiziert wurde. Daraus kann dann der F_1–Score wie in Formel 5.1 berechnet werden.

$$F_1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{5.1}$$

Diese Metrik hat allerdings auf die Anwendung für diesen Ansatz mehrere Probleme, die dafür sorgen, dass sie nicht so gut geeignet ist. Zum einen würden nicht mehr die Bilder als Ganzes betrachtet, sondern die einzelnen Pixel. Das Problem dabei ist, dass der Kontext innerhalb des Bildes verloren geht. Dieser ist allerdings für die geplante Anwendung sehr wichtig, da die Background Subtraction als Vorverarbeitung für einen anderen Klassifikator gedacht ist. So ist die Aussage, wie viele Pixel korrekt klassifiziert wurden, nicht so aussagekräftig für die Funktionsweise, da durch den Score nicht deutlich wird, wie gut diese Ergebnisse für die Weiterverarbeitung geeignet sind. So kann beispielswei-

se ein Ergebnis, in welchem 20 % der Pixel der Person falsch klassifiziert sind, für den Klassifikator nutzbarer sein als eines, in welchem nur 10 % der Pixel falsch klassifiziert sind, diese aber beispielsweise die Person unterteilen, so dass sie in zwei Cluster eingeteilt werden würde, welche nicht gut klassifizierbar sind. Außerdem kann ein Ergebnis, in welchem ein Hintergrundobjekt nicht korrekt entfernt werden konnte, ein gutes Ergebnis sein, wenn die Person trotzdem in einem Stück enthalten ist. Dieses Hintergrundobjekt würde in diesem Fall als eigenes Cluster betrachtet und als keine Person klassifiziert werden. Ein anderes Problem dabei ist, dass man zum Prüfen der Ergebnisse mit dieser Metrik die perfekten Ergebnisse benötigt. Diese müssten in unserem Fall für den Datensatz erst erzeugt werden. Dies ist für 935 Bilder im Datensatz allerdings sehr aufwendig. Ein weiterer Punkt ist, dass nur Ergebnisse mit einer Metrik verglichen werden können, wenn die Ansätze mit den selben Daten getestet wurden. Dies ist für sehr viele der in Abschnitt 4.4 beschriebenen Verfahren nicht möglich, da diese in großen Teilen andere Eingangsdaten nutzen, beziehungsweise die Datensätze, die sie nutzen, für unser Szenario wenig aussagekräftig sind.
Grundlegend ist es sehr schwierig, eine Metrik für unser Szenario zu finden, welche feste Grenzwerte hat, da das Ziel ist Daten zu erzeugen, mit welchen der Klassifikator arbeiten kann. Dafür muss alles was eine Person sein könnte, als Vordergrund klassifiziert werden, und alles, was keine Person sein kann, als Hintergrund. So wäre lediglich denkbar, einen Score zu nutzen, welcher nur das Gebiet um die tatsächliche Person betrachtet. Dabei ergibt sich erneut das Problem mit dem Aufwand. Dazu müssten die perfekten Ergebnisse aus 935 Bildern händisch erstellt werden und zusätzlich die Bereiche um die Person für jedes Bild bestimmt werden. Dies würde den Aufwand erneut deutlich erhöhen, ohne dabei eine gut geeignete Metrik zu haben.
Um aber dennoch einen Vergleich zu perfekten Ergebnissen zu haben wäre es möglich, einige Bilder anzupassen und sie dem Ergebnis gegenüber zu stellen, wie dies in Abschnitt 5.3 vorgenommen wird.
Ähnliches gilt für den Vergleich mit dem bisherigen Ansatz, welcher im Abschnitt 5.4 getätigt wird. Dabei ist zu bedenken, dass der bisherige Ansatz nur mit einer Referenzaufnahme funktioniert. Würde man die Kamera in dem Szenario im Vergleich zur Referenzaufnahme drehen oder den Roboter bewegen, wäre das Ergebnis des bisherigen Ansatzes unbrauchbar.

5.3 VERGLEICH MIT IDEALEM ERGEBNIS

In diesem Abschnitt wird betrachtet, wie gut sich die Umsetzung im Bezug zu einer idealen Lösung verhält. Dafür wurden drei Szenen ausgewählt. Alle Szenen sind durch das Platzieren von Personen auf dem Boden in dem Szenario, welches in Abbildung 5.2 als RGB–Bild zu sehen ist, erzeugt.

Abbildung 5.2: RGB Bild des Szenarios, in dem die Szenen erstellt wurden

Für den Vergleich werden im Folgenden die Rohdaten dem idealen Ergebnis und dem erreichten Ergebnis gegenübergestellt. Dafür sind in den Abbildungen 5.3, 5.4 und 5.5 links die Rohdaten, in der Mitte die idealen Ergebnisse und rechts die mit der Background Subtraction erreichten Ergebnisse abgebildet. Die idealen Ergebnisse wurden dafür händisch erstellt. Dazu wurde das Bild der Rohdaten genutzt. In diesem wurden dann alle Stellen, außer der Person, als Hintergrund markiert. Dieses Verfahren ist ziemlich aufwendig.

Abbildung 5.3: Vergleich mit idealem Ergebnis Szene 1

Beim Betrachten der Szene in Abbildung 5.3 fällt auf, dass nahezu die komplette Person erhalten ist. Allerdings sind auch noch sehr große Gebiete erhalten, welche nicht zur Person gehören beziehungsweise keine Person sind. Bei diesen Regionen handelt es sich hauptsächlich um Areale, welche unter dem Fenster lokalisiert sind. In diesem Bereich befinden sich in der Szenerie sehr viele kleine Objekte, welche nachts auf Fluren in Pflegeeinrichtungen in dieser Menge eher nicht zu erwarten sind. Zusätzlich würde der Klassifikator im Anschluss daran diese als keine Person klassifizieren. Gleichzeitig ist die Gesamtzahl der Punkte stark reduziert worden und große Flächen, wie die Bodenebene, wurden entfernt. Daher ist dieses Ergebnis positiv zu bewerten.
In der zweiten Szene, welche in Abbildung 5.4 zu sehen ist, ist das Ergebnis ähnlich

Abbildung 5.4: Vergleich mit idealem Ergebnis Szene 2

gut wie bei der Ersten. Dabei fällt aber auch auf, dass hier Teile der Bodenebene nicht entfernt werden konnten. Dennoch wurde der Hauptteil der Bodenebene entfernt, insbesondere im Umkreis der Person. Somit würde wieder ein Ergebnis geliefert werden, mit welchem der Klassifikator arbeiten kann. In dieser Szene ist zu beachten, dass die Gardinen geöffnet sind und damit sehr starke Beleuchtungseffekte auftreten.

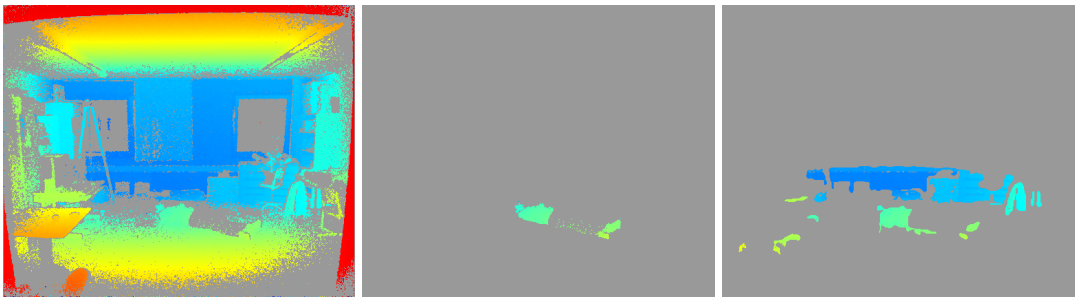

Abbildung 5.5: Vergleich mit idealem Ergebnis Szene 3

In der dritten Szene befindet sich die Person weiter entfernt von der Kamera, das Ergebnis ist aber ähnlich gut wie bei den beiden vorherigen. Die Person wurde nicht entfernt und ist in großen Teilen erhalten. Allerdings konnten hier die kleinen Objekte nicht gut entfernt werden. Dennoch ist es ein Ergebnis, mit welchem der Klassifikator gut arbeiten könnte.
Daraus kann man schließen, dass die Ergebnisse für unser Szenario gut geeignet sind, da sie für den Klassifikator brauchbare Daten liefern und die Punktanzahl stark reduziert wurde. Zusätzlich wird nahezu die komplette Person erhalten.

5.4 VERGLEICH MIT BISHERIGER UMSETZUNG

In diesem Abschnitt werden die Ergebnisse aus dem bisherigen Ansatz mit denen der Background Subtraction verglichen. Dafür werden dieselben drei Szenen betrachtet wie beim Vergleich mit dem idealen Ergebnis in Abschnitt 5.3. In den Abbildungen 5.6, 5.7 und 5.8 sind dafür links erneut die Rohdaten, in der Mitte die Ergebnisse mit dem bisherigen Ansatz und rechts die Ergebnisse der Background Subtraction zu sehen.

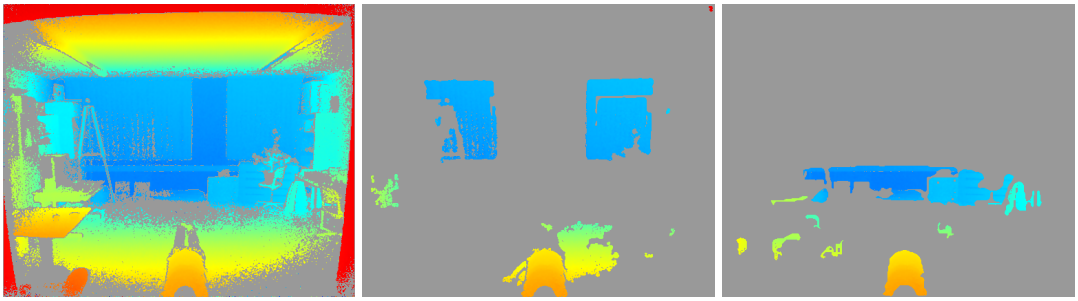

Abbildung 5.6: Vergleich mit bisherigem Ergebnis Szene 1

In Abbildung 5.6 erkennt man, dass der bisherige Ansatz mit Referenzbildern mit kleineren Objekten nahezu kein Probleme hat. Allerdings ist er sehr anfällig für Änderungen im Vergleich zur Referenzaufnahme. Dies erkennt man beispielsweise an den Gardinen im Hintergrund. Eine andere Region, in dem dies sehr auffällt, ist rechts von der Person. Durch geschlossene Gardinen ist die Beleuchtung anders als in der Referenzaufnahme, so wurde ein sehr großes Bodenareal falsch klassifiziert. Gegen diese Probleme ist der neue Ansatz nicht anfällig und benötigt keine Referenzaufnahme mehr.

Abbildung 5.7: Vergleich mit bisherigem Ergebnis Szene 2

In Abbildung 5.7 ist ein nahezu perfektes Ergebnis im Vergleich zum idealen Ergebnis erreicht worden. Lediglich die Gardinen haben eine leicht andere Position als in der Referenzaufnahme. In diesem Fall liefert das bisherige Verfahren ein besseres Ergebnis.

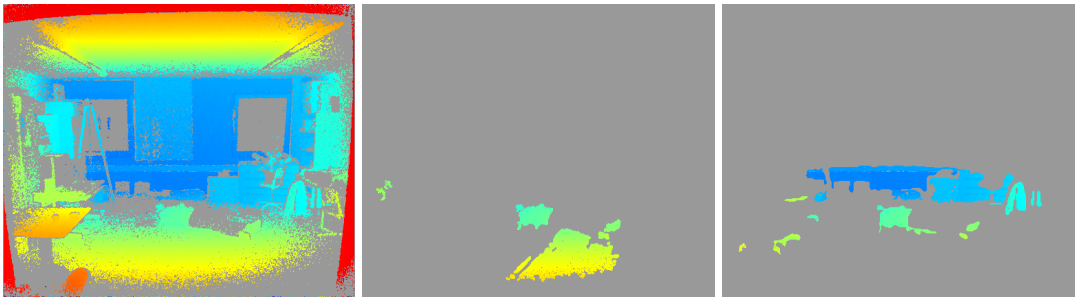

Abbildung 5.8: Vergleich mit bisherigem Ergebnis Szene 3

In der dritten Szene in Abbildung 5.8 erkennt man erneut im hinteren Bereich die Stärken der bisherigen Umsetzung. Im vorderen Bereich erkennt man allerdings auch die Anfäl-

ligkeit für leichte Veränderungen in der Szene durch Beleuchtung oder Bewegung von Objekten im Vergleich zur Referenzaufnahme.
Diese drei Szenen zeigen, dass der neue Ansatz dem Bisherigen in Bezug auf die Robustheit gegenüber Kamerabewegungen und Veränderungen der Szene deutlich überlegen ist. In Bezug auf kleinere Objekte ist er in einem statischen Szenario anfälliger. Im Gegensatz zu dem bisherigen Ansatz ist der Neue auch für mobile Kameras nutzbar und daher auch in Bezug auf unser Szenario deutlich besser geeignet.

5.5 FAZIT

In diesem Abschnitt soll ein Fazit über die Umsetzung gezogen werden. Dabei ist es hilfreich einen Überblick über die Fähigkeiten und positiven Eigenschaften der entwickelten Background Subtraction zu geben. Diese Fähigkeiten sind:

- Die Verarbeitung von RGB–D–Daten in brauchbare Daten für den Klassifikator der GPE.
- Die Entfernung eines Großteils des Hintergrundes.
- Die gestürzten Personen bleiben nahezu komplett erhalten.
- Die Unabhängigkeit von Referenzaufnahmen.
- Die Anwendbarkeit für bewegte Kameras.
- Die Verarbeitung der Daten mit einer Bildrate von 10 fps.
- Die Hardware–Anforderungen entsprechen den Möglichkeiten des Roboters.
- Eine gut wartbare Implementation, die den einfachen Austausch einzelner Schritte ermöglicht.

Es sollten auch die Schwächen und negativeren Eigenschaften betrachtet werden. Diese wären:

- Hintergrundobjekte mit unebener Oberfläche können nicht entfernt werden.
- Gebiete mit sehr vielen Hintergrundobjekten werden nicht zuverlässig entfernt.
- Die gesamte Analyse eines Bildes ist erst nach bis zu 500 ms abgeschlossen.
- Der RANSAC–Algorithmus findet nicht immer gute Ebenen zum Entfernen.
- Die Schwächen des Kameramoduls können sich auf die Ergebnisse auswirken.

Beim Vergleich der positiven mit den negativen Punkten fällt auf, dass die Positiven überwiegen. Die ersten beiden negativen Punkte sind zwar nicht optimal, allerdings ist der

Klassifikator der GPE in der Lage, solche Objekte zuverlässig als Nicht–Person zu klassifizieren. Das Problem im dritten Punkt ist dadurch entschärft, dass gleichzeitig mit den verarbeiteten Daten das Infrarotbild der Szene weitergegeben wird. Somit ist sichergestellt, dass die Person sicher auf dem korrekten Bild lokalisiert werden kann und dieses für die Alarmierung des Personals genutzt werden kann.
Lediglich die letzten beiden Punkte sind nicht so einfach zu entschärfen. Beim Ersten kann es passieren, dass der RANSAC keine guten Ergebnisse bei der Suche nach Ebenen findet und sie entsprechend nicht entfernen kann. Dieses Problem tritt selten auf, ein Beispiel für das Auftreten ist in Abbildung 5.9 zu sehen. Für dieses kann der Klassifikator der GPE mithilfe des Cluster–Verfahrens aber auch ein Ergebnis liefern. Er muss dafür allerdings mehr Cluster prüfen.

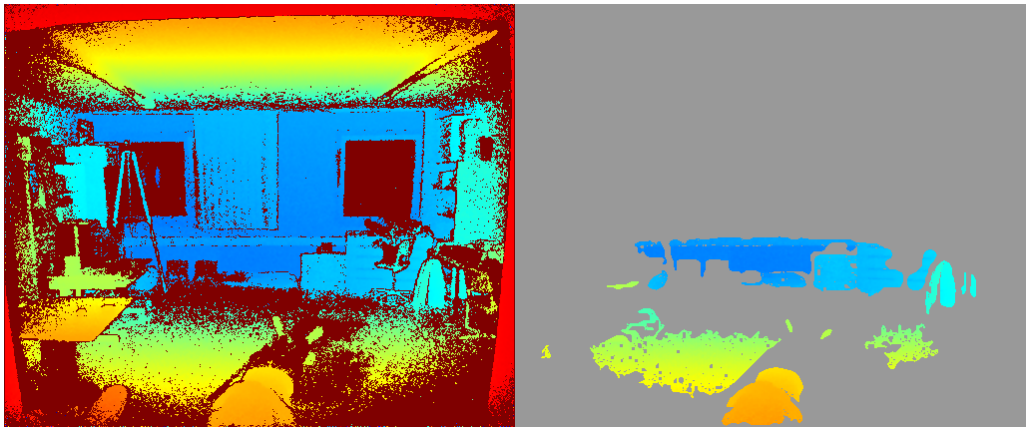

Abbildung 5.9: Beispiel nicht gutes Ergebnis

Der zweite Punkt ist, dass auch schon bei vorherigen Schritten gewisse Probleme und Anfälligkeiten des Kameramoduls aufgefallen sind. Die Probleme sind eine Anfälligkeit für sehr starke Beleuchtungseffekte, wie beispielsweise die Sonne, die durch ein Fenster auf glatte Oberflächen fällt. Ein Beispiel für dieses Problem, welches auch in den Infrarotbildern auftritt, ist in Abbildung A.1 im Anhang zu sehen. Dabei kann man erkennen, dass bei sehr starker Beleuchtung im RGB–D–Bild in den betroffenen Gebieten keine zuverlässigen Ergebnisse ermittelt werden konnten.
Das zweite Problem des Kamaramoduls ist, dass für manche Oberflächen keine zuverlässigen beziehungsweise keine RGB–D–Daten ermittelt werden können. Die beiden Szenarien in Abbildung A.2 im Anhang lassen vermuten, dass es sich dabei um schwarze Oberflächen handelt. Allerdings kann es nicht nur an der Farbe liegen, da für andere schwarze Objekte gültige Werte ermittelt werden konnten. Dieses Problem kann sehr kritisch sein, falls eine gestürzte Person nur Kleidung mit solchen Eigenschaften trägt, könnte sie möglicherweise in den RGB–D–Daten nicht sichtbar sein.

6 AUSBLICK

In diesem Kapitel soll ein Ausblick auf die Zukunft der Arbeit gegeben werden. Eine der ersten Aufgaben wäre die Background Subtraction und den Klassifikator gemeinsam zu testen. Diese Tests sollten nicht nur unter Laborbedingungen, sondern auch in der geplanten Einsatzumgebung stattfinden. In dem Zusammenhang wird es ferner auch wichtig, eine Schnittstelle zur Weitergabe der Ergebnisse an die Kommunikationszentrale beziehungsweise Steuerungszentrale zu schaffen, um die Ergebnisse mit gestürzten Personen an das Pflegepersonal weiterleiten zu können.
Zudem bieten sich Überlegungen an, die zur Reduzierung der Ungenauigkeit und Anfälligkeit gegenüber der im Abschnitt 5.5 genannten Probleme führen können. So wäre es sinnvoll zu testen, welche Faktoren für die erwähnten Probleme mit dem Kameramodul verantwortlich sind und ob diese abgestellt werden können. Zusätzlich ist zu prüfen, wie stark diese Effekte im gewählten Einsatzszenario auftreten. Starke Sonneneinstrahlung ist nachts auf dem Flur einer Pflegeeinrichtung eher nicht zu erwarten, aber es ist wichtig herauszufinden, ob die Beleuchtung vor Ort ähnliche Effekte verursacht.
Zusätzlich bieten sich Überlegungen an, wie die Genauigkeit weiter erhöht werden kann. Eine Möglichkeit wäre die Nutzung von 3D Karten, wie es beispielsweise von [Wengefeld et al., 2016] beschrieben wird. Aktuell werden im Projekt allerdings noch keine 3D Karten genutzt, daher musste diese Möglichkeit vorerst verworfen werden. Dahingehend wäre es notwendig 3D Kartierungsalgorithmen zu testen, zu validieren und gegebenenfalls zu implementieren. Die erstellten 3D Karten können dann genutzt werden, um statische Objekte und Wände zu entfernen und die Genauigkeit zu erhöhen. Anstelle von 3D Karten könnte man auch versuchen, auf Basis der 2D Karten und der Roboterposition Wände und statische Hindernisse zu entfernen.

LITERATUR

Akilan, T. (2018). A foreground inference network for video surveillance using multi-view receptive field. *arXiv preprint arXiv:1801.06593*.

Ang Lim, L. & Yalim Keles, H. (2018). Foreground Segmentation Using a Triplet Convolutional Neural Network for Multiscale Feature Encoding. *arXiv*, arXiv–1801.

Antonello, M., Carraro, M., Pierobon, M. & Menegatti, E. (2017). Fast and robust detection of fallen people from a mobile robot. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4159–4166. https://doi.org/10.1109/IROS.2017.8206276

Ashley, J. (2014). Quick Reference: Kinect 1 vs Kinect 2. *The Imaginative Universal*.

Bahri, F., Shakeri, M. & Ray, N. (2018). Online illumination invariant moving object detection by generative neural network. *arXiv preprint arXiv:1808.01066*.

Bahrmann, F., Hellbach, S. & Böhme, H.-J. A Fuzzy-based Adaptive Environment Model for Indoor Robot Localization. In: *Telehealth and Assistive Technology / 847: Intelligent Systems and Robotics*. 2016, Januar.

Bahrmann, F., Hellbach, S., Keil, S. & Böhme, H.-J. (2014). Understanding Dynamic Environments with Fuzzy Perception. *International Conference on Neural Information Processing*, 553–562.

Bahrmann, F., Vogt, S., Wasic, C., Graessel, E. & Boehme, H.-J. (2020). Towards an All-Day Assignment of a Mobile Service Robot for Elderly Care Homes. *American Journal of Nursing*, *9*(5), 329–337.

Bakkay, M. C., Rashwan, H. A., Salmane, H., Khoudour, L., Puigtt, D. & Ruichek, Y. (2018). BSCGAN: deep background subtraction with conditional generative adversarial networks. *2018 25th IEEE International Conference on Image Processing (ICIP)*, 4018–4022.

Bouwmans, T., Javed, S., Sultana, M. & Jung, S. K. (2019). Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Networks*.

Braham, M. & Van Droogenbroeck, M. (2016). Deep background subtraction with scene-specific convolutional neural networks. *2016 international conference on systems, signals and image processing (IWSSIP)*, 1–4.

Brose, J. (2018). *Ein Roboter–Nachtwächter zur Unterstützung von Pflegekräften*.

Brose, J., Bahrmann, F. & Böhme, H.-J. (2019). Towards the RoNiSCo Mobile ApplicationTowards the RoNiSCo Mobile Application. *Bilateral Student Workshop CTU Prague*, 36.

Brose, J., Erzgräber, R., Bahrmann, F. & Böhme, H.-J. (2019). Development of a Fallen People Detector. *Bilateral Student Workshop CTU Prague*, 41.

Chen, Y., Wang, J., Zhu, B., Tang, M. & Lu, H. (2017). Pixel-wise deep sequence learning for moving object detection. *IEEE Transactions on Circuits and Systems for Video Technology*.

Cuevas, C., Yáñez, E. M. & Garcia, N. (2016). Labeled dataset for integral evaluation of moving object detection algorithms: LASIESTA. *Computer Vision and Image Understanding*, *152*, 103–117.

Eubel, C. & Heine, H. (2013). Pflegenotstand in Deutschland: Auf Personalsuche in Asien. *Online: http:// www.tagesspiegel.de/ politik/ pflegenotstand-in--deutschland-auf-personalsuche-in-asien/ 8011390.html*.

Fengliang Xu & Kikuo Fujimura. (2003). Human detection using depth and gray images. *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, 2003.*, 115–121.

Greff, K., Brandão, A., Krauß, S., Stricker, D. & Clua, E. (2012). A comparison between background subtraction algorithms using a consumer depth camera. *VISAPP (1)*, 431–436.

Hare, J. S., Samangooei, S. & Dupplaw, D. P. (2011). OpenIMAJ and ImageTerrier: Java libraries and tools for scalable multimedia analysis and indexing of images. *Proceedings of the 19th ACM international conference on Multimedia*, 691–694. https://doi.org/10.1145/2072298.2072421

Hashiyama, T., Mochizuki, D., Yano, Y. & Okuma, S. (2003). Active frame subtraction for pedestrian detection from images of moving camera. *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483)*, *1*, 480–485 vol.1.

Hebesberger, D., Körtner, T., Pripfl, J., Gisinger, C., Hanheide, M. et al. (2015). What do staff in eldercare want a robot for? An assessment of potential tasks and user requirements for a long-term deployment.

Ilyas, A., Scuturici, M. & Miguet, S. (2009). Real Time Foreground-Background Segmentation Using a Modified Codebook Model. *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, 454–459.

Kim, K., Chalidabhongse, T. H., Harwood, D. & Davis, L. (2005). Real-time foreground–background segmentation using codebook model. *Real-time imaging*, *11*(3), 172–185.

Kurban, R., Skuka, F. & Bozpolat, H. (2015). Plane segmentation of kinect point clouds using RANSAC. *The 7th international conference on information technology*, 545–551.

Langmann, B., Ghobadi, S. E., Hartmann, K. & Loffeld, O. (2010). Multi-modal background subtraction using gaussian mixture models. *ISPRS Symposium on Photogrammetry Computer Vision and Image Analysis*, 61–66.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Li, X., Ye, M., Liu, Y. & Zhu, C. (2017). Adaptive deep convolutional neural networks for scene-specific object detection. *IEEE Transactions on Circuits and Systems for Video Technology*.

Lim, K., Jang, W.-D. & Kim, C.-S. (2017). Background subtraction using encoder-decoder structured convolutional neural network. *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–6.

Lim, L. A. & Keles, H. Y. (2019). Learning multi-scale features for foreground segmentation. *Pattern Analysis and Applications*, 1–12.

Lischke, F., Bahrmann, F., Hellbach, S. & Böhme, H.-J. (2017). RoNiSCo: Robotic Night Shift Companion. *Machine Learning Reports*, (03), 26–35.

Murgia, J., Meurie, C. & Ruichek, Y. (2014). An Improved Colorimetric Invariants and RGB-Depth-Based Codebook Model for Background Subtraction Using Kinect. In A. Gelbukh, F. C. Espinoza & S. N. Galicia-Haro (Hrsg.), *Human-Inspired Computing and Its Applications* (S. 380–392). Springer International Publishing.

Nguyen, V.-T., Vu, H. & Tran, T.-H. (2014). An efficient combination of RGB and depth for background subtraction. *The National Foundation for Science and Technology Development (NAFOSTED) Conference on Information and Computer Science*, 49–63.

OAK. (2016). Pflegefachkraft Ausbildung, Beruf, Prüfung. *OAK - Online Akademie GmbH & Co. KG*.

Pflegestatistik, D. et al. (2018). *Pflege im Rahmen der Pflegeversicherung Deutschlandergebnisse* (Techn. Ber.).

Rusu, R. B. & Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). *IEEE International Conference on Robotics and Automation (ICRA)*.

Sakkos, D., Liu, H., Han, J. & Shao, L. (n. d.). End-to-end video background subtraction with 3D convolutional networks. *Illumination*, *20*, 6.

Schäufele, M., Köhler, L., Hendlmeier, I., Hoell, A. & Weyerer, S. (2013). Prävalenz von Demenzen und ärztliche Versorgung in deutschen Pflegeheimen: eine bundesweite repräsentative Studie. *Psychiatrische Praxis*, *40*(04), 200–206.

Song, Y., Noh, S., Yu, J., Park, C. & Lee, B. (2014). Background subtraction based on Gaussian mixture models using color and depth information. *The 2014 International Conference on Control, Automation and Information Sciences (ICCAIS 2014)*, 132–135.

Stalinski, S. (2017). Pflegenotstand in Deutschland Überlastet ausgebrannt - und weg. *Online: https://www.tagesschau.de/inland/pflege-notstand-101.html*.

Stone, E. E. & Skubic, M. (2011). Evaluation of an inexpensive depth camera for passive in-home fall risk assessment. *2011 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*, 71–77.

Toyama, K., Krumm, J., Brumitt, B. & Meyers, B. (1999). Wallflower: principles and practice of background maintenance. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, *1*, 255–261 vol.1.

Wang, X., Liu, L., Li, G., Dong, X., Zhao, P. & Feng, X. (2018). Background Subtraction on Depth Videos with Convolutional Neural Networks. *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–7.

Wang, Y., Jodoin, P.-M., Porikli, F., Konrad, J., Benezeth, Y. & Ishwar, P. (2014). CDnet 2014: An expanded change detection benchmark dataset. *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 387–394.

Wengefeld, T., Lewandowski, B. & Groß, H.-M. (2016). Detektion gestürzter Personen in häuslicher Einsatzumgebung. *Proceedings Innteract Conference 2016*, 122–129.

Wren, C. R., Azarbayejani, A., Darrell, T. & Pentland, A. P. (1997). Pfinder: real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7), 780–785.

Yu, R., Wang, H. & Davis, L. S. (2018). ReMotENet: Efficient relevant motion event detection for large-scale home surveillance videos. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1642–1651.

Zeng, D. & Zhu, M. (2018). Multiscale fully convolutional network for foreground object detection in infrared videos. *IEEE Geoscience and Remote Sensing Letters*, *15*(4), 617–621.

Zhao, C., Cham, T., Ren, X., Cai, J. & Zhu, H. (2018). Background Subtraction Based on Deep Pixel Distribution Learning. *2018 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6.

Zhao, F., Cao, Z., Xiao, Y., Mao, J. & Yuan, J. (2018). Real-Time Detection of Fall From Bed Using a Single Depth Camera. *IEEE Transactions on Automation Science and Engineering*, 1–15. https://doi.org/10.1109/TASE.2018.2861382

Zhao, X., Chen, Y., Tang, M. & Wang, J. (2017). Joint background reconstruction and foreground segmentation via a two-stage convolutional neural network. *2017 IEEE International Conference on Multimedia and Expo (ICME)*, 343–348.

ANHANG

Abbildung A.1: Anfälligkeit des Kameramoduls für bestimmte Oberflächeneigenschaften

Abbildung A.2: Anfälligkeit des Kameramoduls für starke Beleuchtung

	Nutzbar für bewegte Kamera	gestürzte Personen immer als Vordergrund	RGB–D–Daten nutzbar	mit mindestens 10 fps lauffähig	keine spezielle Hardware
First Frame Subtraction					
[Brose, Erzgräber et al., 2019]	nein	ja	ja	ja	ja
Single Gaussian					
[Wren et al., 1997]	nein	-	RGB–Bilder	-	-
Gaussian Mixture Models					
[Song et al., 2014]	nein	-	RGB–D–Bilder und RGB Bilder	-	-
[Langmann et al., 2010]	nein	-	-	-	-
[Nguyen et al., 2014]	nein	-	RGB–D–Bilder und RGB Bilder	-	-
Codebook Models					
[Kim et al., 2005]	nein	-	RGB Bilder oder Graustufenbilder	-	-
[Ilyas et al., 2009]	nein	-	-	-	-
[Murgia et al., 2014]	nein	-	-	-	-
Minimum Background					
[Stone & Skubic, 2011]	nein	-	ja	-	-
[Fengliang Xu & Kikuo Fujimura, 2003]	ja	-	RGB–D–Bilder und Graustufenbilder	-	-
[Antonello et al., 2017]	ja	ja	ja	-	-
weitere Ansätze					
[Toyama et al., 1999]	nein	-	nein Graustufenbild	-	-
[Hashiyama et al., 2003]	ja	nein		ja	-

Tabelle A.1: Klassische Ansätze

	Nutzbar für bewegte Kamera	gestürzte Personen immer als Vordergrund	RGB–D–Daten nutzbar	mit mindestens 10 fps lauffähig	keine spezielle Hardware
Basic CNNs					
[C. Zhao et al., 2018]	wahrscheinlich nein	-	nein	-	-
[X. Wang et al., 2018]	ja	-	ja	-	nein
Multi–scale and Cascaded CNNs					
[Ang Lim & Yalim Keles, 2018]	wahrscheinlich nein	nur bewegte	RGB–Bilder	-	-
[L. A. Lim & Keles, 2019]	wahrscheinlich nein	nur bewegte	RGB–Bilder	-	-
Fully CNNs					
[Zeng & Zhu, 2018]	wahrscheinlich nein	-	Infrarotbilder	-	nein
Deep CNNs					
[X. Zhao et al., 2017]	wahrscheinlich nein	-	-	-	nein
[Li et al., 2017]	wahrscheinlich nein	nur bewegte	-	-	-
[Chen et al., 2017]	ja	nur bewegte	-	nur Ansatz 1	-
Structured CNNs					
[K. Lim et al., 2017]	-	nur bewegte	-	-	-
3D CNNs					
[Sakkos et al., n. d.]	-	nur bewegte	RGB–Bilder	-	-
[Yu et al., 2018]	wahrscheinlich nein	nur bewegte	-	nur alle 15 Sekunden ein Video	-
[Akilan, 2018]	-	nur bewegte	-	-	-
GANs					
[Bakkay et al., 2018]	wahrscheinlich nein	wahrscheinlich nur bewegte	-	ja	ja
[Bahri et al., 2018]	-	nur bewegte	-	-	-

Tabelle A.2: Deep–Learning Ansätze

THESEN

- Eine Background Subtraction ist eine effiziente Methode, um interessante Areale für eine GPE zu finden.
- Eine Background Subtraction kann für ein Szenario mit mobilem Roboter genutzt werden.
- Eine Background Subtraction kann ohne Referenzaufnahme arbeiten.
- Eine Background Subtraction kann auch unbewegte Personen als Vordergrund betrachten.

SELBSTSTÄNDIGKEITSERKLÄRUNG

Ich versichere hiermit, dass ich die Master-Arbeit mit dem Titel

„Aktuelle Methoden der Background Subtraction und deren Anwendung als Vorverarbeitung einer Gestürzten–Person–Erkennung“

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Dresden, 24. September 2020

Jan Brose