

Taking Mobile Multi-Object Tracking to the Next Level

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der
RWTH Aachen University zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Diplom-Informatiker

Dennis Mitzel

aus Katschar, Kasachstan

Berichter: Prof. Dr. Bastian Leibe

Prof. Dr. Luc Van Gool

Tag der mündlichen Prüfung: 30.09.2013

Diese Dissertation ist auf den Internetseiten der Hochschulbibliothek online verfügbar.

Selected Topics in Computer Vision

herausgegeben von
Prof. Dr. Bastian Leibe
Lehr- und Forschungsgebiet Informatik 8
(Computer Vision)
RWTH Aachen University

Band 1

Dennis Mitzel

Taking Mobile Multi-Object Tracking to the Next Level

Shaker Verlag
Aachen 2014

Bibliographic information published by the Deutsche Nationalbibliothek

The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at <http://dnb.d-nb.de>.

Zugl.: D 82 (Diss. RWTH Aachen University, 2013)

Copyright Shaker Verlag 2014

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the publishers.

Printed in Germany.

ISBN 978-3-8440-2524-8

ISSN 2198-3372

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen

Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9

Internet: www.shaker.de • e-mail: info@shaker.de

Abstract

Recent years have seen considerable progress in automotive safety and autonomous navigation applications, fueled by the remarkable advance of individual Computer Vision components, such as object detection, tracking, stereo and visual odometry. The goal in such applications is to automatically infer semantic understanding from the environment, observed from a moving vehicle equipped with a camera system. The pedestrian detection and tracking components constitute an actively researched part in scene understanding, important for safe navigation, path planning, and collision avoidance.

Classical *tracking-by-detection* approaches require a robust object detector that needs to be executed in every frame. However, the detector is typically the most computationally expensive component, especially if more than one object class needs to be detected. A first goal of this thesis was to develop a vision system based on stereo camera input that is able to detect and track multiple pedestrians in real-time. To this end, we propose a hybrid tracking system that combines a computationally cheap low-level tracker with a more complex high-level tracker. The low-level trackers are either based on level-set segmentation or stereo range data together with a point registration algorithm and are employed in order to follow individual pedestrians over time, starting from an initial object detection. In order to cope with drift and to bridge occlusions that cannot be resolved by low-level trackers, the resulting tracklet outputs are fed to a high-level multi-hypothesis tracker, which performs longer-term data association. With this integration we obtain a real-time tracking framework by reducing object detector applications to fewer frames or even to few small image regions when stereo data is available. Reduction of expensive detector evaluations is especially relevant for the deployment on mobile platforms, where real-time performance is crucial and computational resources are notoriously limited.

To overcome another limitation of a classical *tracking-by-detection* pipeline, employment only for tracking of objects for which a pre-trained object classifier is available, we propose a *tracking-before-detection* system that is able to track known and unknown

objects robustly, based purely on stereo information. With this approach we track all visible objects in the scene by first segmenting the point cloud into individual objects and associating them to trajectories based on a simple registration algorithm. The core of our approach is a compact 3D representation that allows us to robustly track a large variety of objects, while building up models of their 3D shape online. In addition to improving tracking performance, this representation allows us to detect anomalous shapes, such as carried items on a person’s body. Moreover, classical pedestrian tracking approaches ignore important aspects of human behavior, that should be considered for better scene understanding. Humans are not moving independently, but they closely interact with their surroundings, which includes not only other persons, but also further scene objects. Being able to track not only humans but also their objects, such as child strollers, suitcases, walking aids and bicycles, we propose a probabilistic approach for classifying person-object interactions, which associates objects simultaneously to persons and predicts their interaction type.

In order to demonstrate the capabilities of proposed tracking algorithms, we evaluated them on several challenging video sequences, captured in busy and crowded shopping street environments. As our experiments prove we come closer to the goal of better scene understanding, being able to detect and track multiple objects in the scene in real time and to predict their possible interactions.

Zusammenfassung

In den letzten Jahren hat die Entwicklung von Fahrerassistenzsystemen und mobilen Robotern erhebliche Fortschritte gemacht. Dies wurde möglich durch bemerkenswerte Fortschritte von einzelnen Methoden des maschinellen Sehens wie Objekterkennung, Objektverfolgung, Stereotiefenschätzung und Stereokamera-basierte Odometrie. Das Ziel dieser Methoden beim Einsatz in mobilen Robotern ist es, dem Roboter ein Szenenverständnis zu vermitteln. Möglich wird dies durch das automatische Auswerten von Bildern einer auf dem Roboter montierten Kamera. Objekterkennung und Objektverfolgung sind die für das Szenenverständnis wichtigsten Komponenten, da diese sichere Navigation, Pfadplanung und Kollisionsvermeidung ermöglichen und deshalb zu stark erforschten Gebieten des maschinellen Sehens gehören.

Ein klassisches Verfahren zur Objektverfolgung wird durch den sogenannten *Tracking-by-Detection* Ansatz realisiert. Hierbei wird für jedes Videobild ein Objektdetektor ausgewertet und die resultierenden Objektdetektionen dann mit Hilfe der Odometrie frameübergreifend zu Trajektorien verbunden. Der Nachteil dieses klassischen Ansatzes ist der zwingend notwendige Einsatz eines Objektdetektors auf jedem Frame. Da dieser Detektor typischerweise die rechenintensivste Komponente der Tracking-Pipeline ist, wird dadurch der Einsatz vom *Tracking-by-Detection* für echtzeitkritische Anwendungen unmöglich. Aus diesem Grund war das erste Ziel der Arbeit die Entwicklung eines Objektverfolgungsverfahrens, welches ausgehend von Bildern einer Stereokamera Fußgänger in Echtzeit finden und verfolgen kann. Dazu haben wir einen hybriden Objektverfolgungsansatz entwickelt, welcher einen recheneffizienten *Low-Level Tracker* und einen *High-Level Tracker* kombiniert. Der *Low-Level Tracker* basiert entweder auf einer *Level-Set* Segmentierung oder Stereotiefe kombiniert mit dem *ICP* Algorithmus. Diese Tracker sind verantwortlich für die Verfolgung von Fußgängern über die Zeit basierend auf einer initialen Objektdetektion. Da die *Low-Level Tracker* nicht mit Abweichungen von der echten Position des Objektes, oft verursacht durch Verdeckungen, umgehen können wird das Verfolgungssystem durch einen *High-Level Tracker* erweitert. Der *High-*

Level Tracker erzeugt lange Trajektorien und erkennt durch entsprechende Konsistenztests die Divergenz der *Low-Level Tracker*. Durch diese Kombination wird die Auswertung eines Detektors auf wenige Frames oder sogar wenige kleine Bildregionen pro Frame reduziert. Diese drastische Reduktion schafft die Voraussetzung für ein echtzeitfähiges System, das den Einsatz auf mobilen Robotern erst möglich macht.

Im zweiten Teil der Arbeit stellen wir einen neuen *Tracking-before-Detection* Ansatz vor. Dieser erlaubt es uns, nicht nur bekannte Objektkategorien, wie Fußgänger, sondern auch unbekannte, vorher ungesehene Kategorien zu verfolgen. Mit diesem Ansatz überwinden wir auch die starke Einschränkung von typischen *Tracking-by-Detection* Verfahren, dass ein vortrainierter Objektdetektor erforderlich ist und können somit alle sichtbaren Objekte der Szene verfolgen. Dazu verwenden wir die Punktwolken, die mit Hilfe der Stereoschätzung extrahiert werden. Die Punktwolken werden dabei in individuelle Objekte segmentiert und zu Objekttrajektorien verbunden. Dies geschieht mit Hilfe eines Registrierungsverfahrens, welches zwei Punktwolken auf einander registriert. Den Kern des Verfahrens bildet eine neue, kompakte 3D Objektrepräsentation, die uns auf der einen Seite robuste Verfolgung von beliebigen Objekten erlaubt und auf der anderen Seite das Lernen von 3D-Objektformen online ermöglicht. Die gelernten 3D-Objektformen für Fußgänger erlauben uns die Detektion von getragenen Objekten wie Taschen. Basierend auf der Fähigkeit der Verfolgung von allen Objekten einer Szene wurde in Rahmen dieser Arbeit ein weiterer wichtiger Aspekt der Bewegung von Menschen untersucht. Menschen bewegen sich nicht unabhängig, sondern interagieren sehr stark mit ihrer Umgebung. Diese besteht nicht nur aus anderen Menschen, sondern auch aus weiteren unbekannten Objekten wie Kinderwägen, Koffern, Gehhilfen und Fahrrädern. Um diese Interaktionen modellieren zu können, stellen wir einen neuen probabilistischen Ansatz vor, der uns erlaubt Objekte mit Personen zu assoziieren. Gleichzeitig lässt sich die Art der Interaktion vorhersagen, was wiederum für die Verbesserung der Objektverfolgung verwendet werden kann.

Um die Leistungsfähigkeit der vorgestellten Verfahren zu demonstrieren, haben wir die Algorithmen auf mehreren anspruchsvollen Sequenzen aus sehr belebten Einkaufstraßen evaluiert. Unsere Experimente zeigen, dass wir dem Ziel von einem besseren Szenenverständnis deutlich näher gekommen sind. Wir sind in der Lage Objekte in Echtzeit zu finden, zu verfolgen und ihre Interaktionen vorherzusagen.

Acknowledgments

This dissertation is a product of the invaluable support I received in the last few years from a number of great people.

First and foremost I thank my supervisor Prof. Dr. Bastian Leibe for his continuing intensive support in all stages during my PhD time, for great and inspirational ideas, countless fruitful discussions and teaching me what research is all about. My thanks goes also to Prof. Dr. Luc Van Gool for his interest in my work and agreeing to co-examine my thesis.

I would like to express my gratitude to my diploma thesis supervisor Prof. Dr. Daniel Cremers who intrigued my interest in Computer Vision from the first lecture.

I am deeply grateful to all my colleagues Esther Horbert, Tobias Weyand, Patrick Sudowe, Georgios Floros and Wolfgang Mehner making the life in UMIC really fun. Furthermore, Tobias thank you for offering me a place to sleep during the nights close to the deadlines, that become longer and longer the closer the deadline was approaching and especially thank you for the excellent espresso in the morning. Georgios thank you for all the great Greek supplies as the amazing feta and spinach pies or the excellent olive oil from Crete. Esther thank you very much for being supportive during the difficult time in the first year of my PhD. Patrick thank you for all the discussions about cars in context of car detection in video sequences, but also convincing me why Volkswagen Golf GT is the best car ever. Moreover, I would like to thank the team of KHAO-LAK-BEACH being always punctual in delivering the best Vietnamese food during rough deadline time.

Thanks also to my diploma/master theses students who have contributed to this theses with their work: Esther Horbert, Tobias Baumgartner, Philipp Fischer, Stefan Breuers, Wolfgang Mehner, Emmanouil Tzouridis, Seyed Hamidreza Odabai-Fard, Jonathan Meyer.

Furthermore, I would like to thank Tanja, my two years old daughter, for coaching me to get along with only 3-4 hours sleep per night, which helped me to stay awake and concentrated during long nights before the deadlines.

Last but not least, I would like to thank my family and my friends, especially my parents always believing in me and being always there.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Contributions	2
1.3	Structure of the Thesis	3
2	State of the Art	7
2.1	Image-based Tracking	7
2.2	Tracking-by-Detection	10
2.3	Stereo-based Tracking	14
3	Preliminaries	17
3.1	Object Detection	17
3.2	Stereo Estimation	20
3.3	Visual Odometry	23
3.4	Multi-Hypothesis Tracking	26
3.5	Real-Time Tracking-by-Detection	31
3.6	Camera Setup	36
3.7	Discussion	37
I	Hybrid High-Level/Low-Level Tracking	39
4	Hybrid High-Level/Low-Level Tracking	41
4.1	Motivation	41
4.2	Related Work	44
4.3	Integrated Tracking Frameworks	46
4.4	Hybrid Tracking with Level Sets	47
4.5	Hybrid Tracking with ICP	61
4.6	Discussion	73

II	ROI based Object Detection and Tracking	77
5	Robust ROI Extraction and Segmentation	81
5.1	Point Cloud Labeling	82
5.2	ROI Extraction	84
5.3	ROI Segmentation	85
5.4	Experimental Results	87
5.5	Discussion	90
6	Close-Range Human Detection and Tracking	91
6.1	Related Work	92
6.2	Approach	93
6.3	Experimental Evaluation	97
6.4	Extensions	101
6.5	Discussion	103
7	Tracking with Time-Constrained Detection	107
7.1	Related Work	108
7.2	System Overview	109
7.3	Poisson Process Attention Model	110
7.4	Detailed Implementation	111
7.5	Experimental Results	115
7.6	Discussion	117
III	Tracking People and Their Objects	121
8	Tracking Known and Unknown Objects	125
8.1	Related Work	126
8.2	Overview	128
8.3	3D Object Representation	129
8.4	Stereo Depth-Based Tracking-Before-Detection	130
8.5	Carried Item Detection	132
8.6	Experimental Results	135
8.7	Discussion	138
9	Person-Person and Person-Object Interaction	141
9.1	Related Work	143
9.2	Modeling Person-Object Interactions	143
9.3	Learning	145
9.4	Inference and Prediction	146
9.5	Robust 3D Data Association and Tracking	148
9.6	Experimental Results	151

9.7 Discussion	156
10 Conclusion	161
10.1 Contributions	162
10.2 Perspectives	163
Bibliography	167

1.1. Motivation

Computer Vision is a broad and varied research field concerned with the problem of extracting semantic information from the images of a scene. Having its beginning in the early 1970s, it has been a very vivid research area. Scientists around the world put enormous effort into the development of algorithms and methods trying to tackle the problem of extracting relevant information from existing images. As a result, the techniques of this field have applications in a wide range of scenarios, including manufacturing, security, robotics, car industry, communication and many more. For many applications the behavior of humans in urban scenarios is of particular interest. For example, a traffic safety application could analyze a video stream from a camera system, mounted inside a car or on a mobile robot, in order to issue warnings in case of future path intersections or possible collisions. In order to achieve this goal, methods are required that can process video streams automatically and in real-time. Furthermore, in order to understand the behavior of people, it is also important to recognize and track other objects in their surroundings. In practical scenarios, this includes a large variety of objects such as bicycles, child strollers, shopping carts, trolleys, or wheelchairs. In recent years a number of tracking-by-detection approaches have been proposed to address these goals, reaching remarkable performance for robust people detection and tracking in dynamic and complex real-world scenes. However, those approaches have two major limitations. On the one side they are not yet satisfactory for use on autonomous platforms with respect to their requirements for computational power and energy consumption. On the other side they are naturally restricted to tracking objects for which pre-trained detector models (e.g., pedestrians) are available.

In this thesis, we investigate the problem of multi-object tracking in busy inner-city scenarios. Starting with classical tracking-by-detection approaches, we focus on algorithmic means for improving run-time efficiency in order to make them applicable for use on a mobile robot. Based on the lessons learned from this endeavor, we investigate different

means for reducing the dependency on an expensive object detector by introducing a hybrid tracking framework. Such a framework is a combination of a computationally cheap low-level tracker with a high-level tracker. The low-level tracker follows pedestrians over time after an initial detection and thus takes over the role of the computationally expensive object detector. We investigate different choices for the low-level tracker, exploring both appearance-based and depth-based approaches. In both cases the low-level tracker is augmented by a high-level tracker that, using the tracklets output by the low-level tracker, performs longer-term data association bridging drift and occlusions that cannot be resolved by the low-level trackers. In the second part of the thesis, we then spin this idea further. Assuming that a mechanism exists for extracting regions-of-interest from the input video data (in our case from stereo data), we explore how those ROIs can be used for both simplifying and improving the object detection and tracking stages. Finally, we address the problem of tracking both known and unknown scene objects, which is a prerequisite for robust performance in many real-world settings such as mobile robotics and intelligent vehicles. For this, we extend the ROI-based scheme to a true tracking-before-detection approach, which can automatically track a large number of object candidates even before knowing their categories. This paradigm shift has important consequences for the design of the tracking pipeline. In particular, before tracking we first need to decide how an object candidate that we want to track is defined. To this end, we make use of regions-of-interest which are robustly segmented into candidate objects. Each such region is then tracked independently in 3D using a model-based point cloud registration tracker. In order to learn a representation of the objects, we develop an approach that reconstructs 3D shape models of each tracked object, which allow us besides robustly tracking a large variety of objects, an analysis of their shape. In addition, relying on the tracking results of known and unknown objects, we analyze person-object interactions and use this knowledge to make improved predictions for the continuation of observed trajectories. In this sense, we believe that the contributions of this thesis have brought tracking a significant step towards the next level.

1.2. Contributions

In detail we have made the following contributions:

- We show how the classical *tracking-by-detection* framework can be complemented by a cheap and fast low-level tracker, based either on appearance or depth, resulting in a real-time tracking system. We systematically present the required consistency checks and interactions between the components in order to solve the difficulties in street-level mobile tracking tasks with a number of non-trivial challenges.
- We present an integrated system for upper body detection which is purely based on depth information. The system overcomes the drawback of classical full-body detectors, which often fail to detect pedestrians close to the camera, due to strong

occlusions. It is highly optimized for inner city scenarios and yields superior performance on very challenging outdoor data with more than 40fps.

- We present a novel *tracking-before-detection* system, that, relying only on stereo depth information, is able to track a large variability of objects with unknown appearance in very complex street scenarios, while simultaneously building up their 3D shape models. The framework combines: *visual odometry*, *ground plane estimation*, *point cloud classification*, *ROI extraction*, *segmentation* and *tracking* into a robust framework.
- Relying on the output of our *tracking-before-detection* system, including the tracked objects and their 3D shape models, we present a probabilistic framework for classifying person-object interactions. The system associates unknown objects to the persons and can help in stabilizing trajectories and adapting dynamic models for certain object/person constellations.
- We acquired several hours of video material from mobile stereo camera systems and annotated various sequences with bounding boxes around pedestrians, object/person segmentations and corresponding actions between pedestrians and objects. These sequences, including corresponding annotations, stereo depth, visual odometry, and estimated ground planes have been made publicly available, allowing other researchers to build upon our results without having to assemble a complete system on their own.

1.3. Structure of the Thesis

The thesis is structured as follows:

In **Chapter 2** - “State of the Art” we will summarize the general state-of-the-art work related to *tracking-by-detection* approaches. In each following chapter additional related work will be presented which discusses the difference between the proposed approach and already existing approaches in more detail.

Chapter 3 - “Preliminaries” introduces the basic components we rely on in the following chapters. This includes the mobile platforms we constructed and used in order to acquire data sets, a brief introduction to *object detection*, *visual odometry* and *stereo estimation* and a combination of these components into a *multi-hypothesis tracking* system that performs in real-time.

In **Chapter 4** - “Hybrid High-Level/Low-Level Tracking” we present an integrated framework for mobile street-level tracking of multiple persons. In contrast to classic *tracking-by-detection* approaches, we propose a hybrid-tracking approach that employs

efficient low-level trackers in order to follow individual pedestrians over time. The low-level trackers can either be based on a level-set color segmentation or on stereo depth data combined with ICP registration. Both trackers are initialized and periodically updated by a pedestrian detector and are kept robust through a series of consistency checks. In order to cope with drift and to bridge occlusions, the resulting tracklet outputs are fed to a *high-level multi-hypothesis tracker* (Mitzel *et al.*, 2011b), which performs longer-term data association. This design has the advantage of simplifying short-term data association, resulting in higher-quality tracks that can be maintained even in situations where the pedestrian detector does no longer yield good detections. In addition, it only requires the pedestrian detector to be active part of the time, resulting in computational savings. The chapter is based on research originally presented in (Mitzel and Leibe, 2011; Mitzel *et al.*, 2010).

The following three chapters explore different aspects of region-of-interest based tracking, making use of stereo depth as additional cue.

In **Chapter 5** - “Robust ROI Extraction and Segmentation” we describe a robust region-of-interest (ROI) extraction approach based on depth information, optimized for busy shopping street scenarios. The idea behind the *classical* ROI extraction using stereo data is to fix the attention of the detector/tracker only on the few regions which may contain a target object. The ROIs themselves are represented by grouped bins from a 2D histogram collected from projected 3D points. We introduce several extensions to the classical approach in order to cope with the problem of partially occluded objects and propose a segmentation procedure for dividing ROIs which usually contain several objects, into individual objects by using the Quick Shift algorithm (Vedaldi and Soatto, 2008). The chapter describes a major component we build upon in several of the following chapters, as well as in several publications (Baumgartner *et al.*, 2013; Mitzel and Leibe, 2011, 2012a,b; Mitzel *et al.*, 2011a).

In **Chapter 6** - “Close-Range Human Detection and Tracking” we consider the problem of multi-person detection from the perspective of a head mounted stereo camera. Since pedestrians close to the camera cannot be detected by classical full-body detectors due to strong occlusion, we propose a stereo depth-template based detection approach for close-range pedestrians. We perform a sliding window procedure, where we measure the similarity between a learned depth template and the depth image. To reduce the search space of the detector we slide the detector only over few selected ROIs. The ROI selection allows us to further constrain the number of scales to be evaluated, significantly reducing the computational cost. Besides the technical design and evaluation of our proposed detector, a second main contribution of this chapter is its empirical demonstration of the somewhat surprising fact that such a relatively simple and fast approach can reach superior detection performance on very challenging outdoor data. The chapter is based on research originally presented in (Mitzel and Leibe, 2012a; Mitzel *et al.*, 2011b).

In **Chapter 7** - “Tracking with Time-Constrained Detection” we consider the problem of making best use of an object detector with a fixed and very small time budget. This constraint is not unusual and often arises in robotic scenarios, where, e.g., several vision-based components need to share processing power. The question we pose is: Given a fixed time budget that allows for detector-based verification of k small regions-of-interest in the image, what are the best regions to attend to in order to obtain stable tracking performance? We address this problem by applying a statistical Poisson process model in order to rate the urgency by which individual ROIs should be attended to. These ROIs are initially extracted from a 3D depth-based occupancy map of the scene, as described in Chapter 5, and are then tracked over time. This allows us to balance the system resources in order to satisfy the twin goals of detecting newly appearing objects, while maintaining the quality of existing object trajectories. The chapter is based on research originally presented in (Mitzel *et al.*, 2011a,b).

Finally, the following two chapters develop methods that allow us to extend tracking to objects people are interacting with.

In **Chapter 8** - “Tracking Known and Unknown Objects” we aim to take mobile multi-object tracking to the next level. Our approaches presented in the previous chapters work in a *tracking-by-detection* manner, which limits them to object categories for which pre-trained detector models (e.g., for pedestrians) are available. In contrast, in this chapter we propose a *tracking-before-detection* approach that can track both known and unknown object categories in very challenging street scenes. Our approach relies on noisy stereo depth data in order to segment and track objects in 3D. At its core is a novel, compact 3D representation (Generalized Christmas Tree - GCT) that allows us to robustly track a large variety of objects, while building up models of their 3D shape online. In addition to improving tracking performance, this representation allows us to detect anomalous shapes, such as carried items on a person’s body. The chapter is based on our recent research presented in (Mitzel and Leibe, 2012b; Mitzel *et al.*, 2011b).

In **Chapter 9** - “Person-Person and Person-Object Interaction” we investigate, given the ability of tracking both known and unknown objects, whether we can derive any relationships between those objects. Here, we rely on the fact that humans are not moving independently, but they closely interact with their environment, which includes not only other persons, but also different scene objects. Typical everyday scenarios include people moving in groups, pushing child strollers, or pulling luggage items. Thus, we propose a probabilistic approach for classifying such person-object interactions, associating objects to persons, and predicting how the interaction will most likely continue. Our approach relies on stereo depth information in order to track all scene objects in 3D, while simultaneously building up their 3D shape models, presented in the previous chapter. These models and their relative spatial arrangement are then fed into a prob-

abilistic graphical model which jointly infers pairwise interactions and object classes. The inferred interactions can then be used to support tracking by recovering lost object tracks. The chapter fuses the research presented in (Baumgartner *et al.*, 2013; Mitzel and Leibe, 2012b; Mitzel *et al.*, 2011b) and adds some further experiments.

Chapter 10 concludes the thesis and gives an outlook on possible future research directions and extensions.

Note: *The thesis is based on the technical contributions of my respective first author publications (Baumgartner et al., 2013; Mitzel and Leibe, 2011, 2012a,b; Mitzel et al., 2010, 2011a,b). Several images and text passages in the three major parts of this thesis are taken from these articles. However, additional and new content about further extensions has been added in order to provide deeper insights into our approaches. Several publications resulted from diploma or master thesis projects which were supervised by Prof. Bastian Leibe and myself. For these particular parts an additional note will be given in the footnote of the corresponding chapter.*

In this thesis, we will mainly focus on multi-object tracking approaches from moving vehicles, e.g. a mobile robot that is equipped with a pair of synchronized, forward-looking cameras. This task is very challenging, since multiple objects may appear or emerge from occlusions in every frame that need to be detected. Since background modeling (Stauffer and Grimson, 1999) is no longer applicable in a mobile scenario, this is typically done using visual object detectors (Dollar *et al.*, 2009). Consequently, *tracking-by-detection* has become the dominant paradigm for multi-object tracking applications (Andriluka *et al.*, 2008; Ess *et al.*, 2009b; Huang *et al.*, 2008; Leibe *et al.*, 2008a; Okuma *et al.*, 2004; Wu and Nevatia, 2007). In such a framework, a generic person detector is applied to every frame of the input video sequence, and the resulting detections are associated to tracks. This leads to challenging data association problems, since the detections may themselves be noisy, containing false positives and misaligned detection bounding boxes (Dollar *et al.*, 2009). Several approaches have been proposed to address this issue, e.g., by optimizing over a larger temporal window using dynamic programming (Berclaz *et al.*, 2006), multi-hypothesis tracking (Arras *et al.*, 2008), model selection (Leibe *et al.*, 2008a), network flow optimization (Zhang and Nevatia, 2008), hierarchical (Huang *et al.*, 2008) or MCMC data association (Zhao *et al.*, 2008). In the following, we will review the different tracking approaches in more detail starting with the early methods based on background subtraction, proceeding with state-of-the art *tracking-by-detection* approaches and concluding with stereo based detection and tracking methods.

2.1. Image-based Tracking

A considerable number of object tracking approaches has been proposed starting in the seventies. Each of these tracking approaches requires an object detection procedure which is executed either in every frame or only initially when an object enters the tracking area. Early detection methods relied on temporal information in order to find changing image regions, caused by dynamics of the objects, by simply differencing

temporally adjacent frames. The pixels undergoing strong changes are then marked as foreground and are connected to coherent blobs by a simple connected components algorithm. These frame differencing approaches are known as *background subtraction* and have been intensively studied since the seventies starting with a pioneering work from Jain and Nagel (1979). However, simple frame differencing is sensitive to illumination changes and usually yields many false positives. To cope with this problem, Wren *et al.* (1997) propose to model the intensity of each pixel in the static background with a *normal distribution*, learning the mean and the variance from several consecutive frames. Once the model is learned, pixels in the new frames deviating from the distribution are marked as foreground. Stauffer and Grimson (2000) propose to model the pixel values using mixtures of Gaussians, which provide better fits for outdoor scenes, where the repetitive movement of tree leaves, shadows or reflectance are correctly classified as background, but yield false positives using the unimodal model from Wren *et al.* (1997). Elgammal *et al.* (2000) extend the subtraction process, under the assumption that the neighboring pixels are likely to have the same label, such that an individual pixel should also match to nearby pixel values. This extension corresponds to a typical smoothing assumption, reducing the typical pepper/salt noise significantly. The limitation of all *background subtraction* methods is that they are only applicable for image sequences acquired from a static camera. Although, there are approaches e.g., from Irani and Anandan (1998) that attempt to compensate for the camera motion by building up mosaics for the background pixels, they are still limited to a slow camera motion, which make them not applicable for our scenarios.

Considering the connected regions in the image corresponding to moving objects, the next step in the tracking pipeline is to track these blobs by establishing blob correspondence across frames. Grimson *et al.* (1998) and Wren *et al.* (1997) perform blob tracking based on a Kalman Filter (Gelb, 1974) by updating the position of the center of the blobs and their size. Paragios and Deriche (2000) rely on *background subtraction* in order to obtain an initial detection of the moving regions in the image, which are used as initialization for a level set contour. The level set formulation captures the motion detection and the tracking task simultaneously, by forcing the contour to converge towards the moving area, avoiding areas with a high gradient or static objects. Isard and MacCormick (2001) propose to jointly model the foreground and background by using mixtures of Gaussians, similar to (Stauffer and Grimson, 2000). Given a ground plane estimate, the foreground regions, modeled as cylinders, are projected into a 3D world coordinate system and are tracked by employing a particle filter, which models 3D position, shape and velocity of the objects. The limitation of the standard blob tracking approaches is that they require either only one object to be present in the scene or the objects moving to have a certain distance to each other in the image plane. The lack of a mechanism to separate blobs into individual objects makes it difficult to generate unique track labels for different people. To address this problem, Haritaoglu *et al.* (2000) propose to utilize the vertical projection histogram of the contour resulting from background subtraction in order to determine whether the foreground region con-

tains multiple people. To this end, a vertical projection histogram template is learned from single person annotations and each contour from an incoming blob is compared to the learned silhouette from a single person using the Sum of Absolute Differences (SAD) method (Haritaoglu *et al.*, 1998) yielding a separation into individual pedestrian regions. The regions are then approximated by a rectangle and are tracked using a second order motion model. Collins *et al.* (2001) classify the blobs into different classes before tracking by providing class labels for each blob based on a neural networks approach. Building on this, they propose a classifier based on linear discriminant analysis to distinguish between different vehicle types. Similar to (Hager and Belhumeur, 1998; Zheng and Chellappa, 1995) they then apply an image region matching approach for the tracking part, that determines the best match to the current region by normalized cross correlation using intensity values around candidate regions in the new image.

Another well-established routine towards efficient image based tracking is to use a sparse collection of features such as edges and prior knowledge about the model of the target objects. In general, model-based tracking approaches try to obtain more information about the tracked objects by estimating their precise pose and they use this information for predicting future motion more accurately. Gennery (1982) presented one of the first approaches for tracking of solid 3D-objects assuming the model to be known. Using a procedure similar to the Kalman Filter including prediction and updating steps, Gennery (1982) propose a six degree of freedom model for modeling the position and the orientation of the 3D object, which are predicted using previous knowledge and updated by extracting edge elements closest to the predicted line segments of the model. (Koller *et al.*, 1993) propose a vehicle detection and tracking framework based on a 3D car model. By clustering coherently moving image features first, image regions likely to contain target objects are extracted. Assuming that the vehicles are moving on a planar ground surface and given a rough estimate of the plane and camera parameters, edge segments extracted from the image are matched to the 2D model edges obtained from back-projection of the 3D polyhedral model placed on the ground plane. The 3D object is tracked on the ground plane assuming an uniform motion model along a circular arc, using a prediction/update framework. Similarly, Dellaert and Thorpe (1997) propose an approach that tracks vehicles in highway scenarios by predicting how an imaginary cube around a car position in 3D will fit the projection to the image plane. Prior knowledge about the objects to track was also extensively employed in region based level-set tracking approaches (Cremers, 2006; Leventon *et al.*, 2000; Tsai *et al.*, 2001). The approaches perform a local optimization, iterating between a segmentation and a warping step to track an object's contour over time incorporating prior knowledge about the shape. Since both steps only need to be evaluated in a narrow band around the currently tracked contour, they can be implemented very efficiently. However, in general region based approaches suffer from the fact that a fixed model of the target object is required, which makes it hard to apply them for complex outdoor settings where e.g. a huge number of vehicle types/shapes are available. Furthermore, occlusions and clutter in real scenarios will often cause divergence in the described approaches.

A number of approaches have been proposed in the context of template matching based tracking. The general procedure in this category of tracking approaches is to perform a brute force search for a region in the image similar to a predefined template. A template can be fixed (Comaniciu *et al.*, 2003; Schweitzer *et al.*, 2002) or generated and updated online (Grabner *et al.*, 2006; Jepson *et al.*, 2003). Comaniciu *et al.* (2003) represent a target object by a simple weighted color histogram. Instead of a brute force search for the new location of the object in the next frame, they propose a *Mean-Shift* approach (Cheng, 1995) that searches for the mode by trying to find a position in the new frame that maximizes the appearance similarity of the template and the corresponding image location. Grabner *et al.* (2006) propose an on-line AdaBoost (Freund and Schapire, 1995) based approach, that starting from an initial detection learns a template of the object in an AdaBoost framework, extracting features inside the detection bounding box as positive examples and around the bounding box as negative background examples. In the following frames a most likely new position is found by sampling the neighborhood of the previous object position. The template is then updated based on the new features. This allows to adapt the template while tracking the object, coping with appearance changes of the object caused by illumination changes or plane rotations. In general, online updated template based methods perform better due to the ability to compensate for rotations or scale and appearance changes. Because fixed template-based approaches encode the object appearance usually generated from only a single view, they are only suitable for tracking objects that undergo little pose changes.

For further reading, we refer the reader to a survey from (Yilmaz *et al.*, 2006) for a thorough review of image-based tracking approaches.

2.2. Tracking-by-Detection

As mentioned before, measurement extraction approaches employed for early tracking methods were often relying on background subtraction, which makes them not applicable for our scenarios with a moving cameras. Low-level region/template-based tracking approaches are usually sensitive to illumination, appearance changes or occlusions. In addition, they require a high-level tracker for robust multi-object tracking, which performs consistency checks for recovering from failures caused by the low-level trackers, as we will show in Chapter 4. Furthermore, these approaches do not have a discriminative model that classifies objects into different categories of interest, making the data association much harder. However, due to remarkable progress in object detection and classification (Dalal and Triggs, 2005; Dollar *et al.*, 2010; Felzenszwalb *et al.*, 2010b) the most successful approach for tracking in recent years has been *tracking-by-detection*. In this process the output of an object detector e.g., (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b; Leibe *et al.*, 2008b) executed in each frame is integrated into long-term trajectories. In general, a typical *tracking-by-detection* pipeline is divided into two parts: first, a trajectory hypothesis generation process that, using an appearance and dynam-

ical model, links the detection from adjacent frames into trajectory hypotheses; second, an optimization process that selects a set of hypotheses that is most likely representative for the scene.

Many approaches (Choi and Savarese, 2010; Ess *et al.*, 2009b; Leibe *et al.*, 2008a; Okuma *et al.*, 2004; Wu and Nevatia, 2007; Yang and Nevatia, 2012; Zhang and Nevatia, 2008), including our own approaches presented later, employ a simple color histogram extracted from the image content inside a detection bounding box in order to decide, based on some histogram distance measurement, whether two detections should be linked together or not. As similarity measurement (Ess *et al.*, 2009b; Leibe *et al.*, 2008a; Wu and Nevatia, 2007) employ the Bhattacharyya distance. Kuo *et al.* (2010) build an appearance model based on several complementary features such as color histograms, HOG features (Dalal and Triggs, 2005), modeling object shape and covariance matrices (Tuzel *et al.*, 2006), describing the texture of a detection. With these features an appearance representation is discriminatively trained in a *one vs. all* manner for each trajectory using a boosting framework. Obviously, a simple color-based data association procedure is not really robust for a pedestrian tracking scenario due to illumination changes and shadow artifacts or indistinguishable clothing (E.g., pedestrians dress mostly dark in winter and are not distinguishable during the linking process). Therefore, in the data association process the appearance model is usually reinforced by a dynamic model responsible for spatially plausible association. When tracking is performed in 3D world coordinates, a common dynamic model assumption is employed in many approaches, the constant velocity model, which has shown to be sufficient for our tracking scenarios and other state-of-the-art approaches (Choi and Savarese, 2010; Ess *et al.*, 2009b; Gavrilu and Munder, 2007; Leibe *et al.*, 2008a). In 2D an implicit assumption on constant velocity is made, associating detections with a similar 2D position and scale (Wu and Nevatia, 2007; Yang and Nevatia, 2012; Zhang and Nevatia, 2008).

We employ a *hypothesize-and-verify* procedure (cf. Chapter 3.5) similar to (Ess *et al.*, 2009b; Leibe *et al.*, 2008a) as the optimization process in our tracking approaches in order to infer the most likely trajectory set that best represents the observations from past and current frames. The *hypothesize* step generates an overcomplete set of trajectory hypotheses by linking the pedestrian detections in a space-time volume using the Extended Kalman Filter with a constant velocity motion model and a histogram-based appearance model. For obtaining an optimal set of trajectory hypotheses in the *verify* step we apply the MDL (Minimum Description Length) approach, similar to (Leibe *et al.*, 2008a). Another approach for data association often used for offline tracking is based on a *tracklet* generation process followed by a global optimization (Andriluka *et al.*, 2008; Huang *et al.*, 2008; Kuo *et al.*, 2010; Singh *et al.*, 2008; Stauffer, 2003; Yang and Nevatia, 2012). Tracklets are confident snippets of trajectories including detections with a very high affinity. For the final global data association and the connection of tracklets to final trajectories, many approaches (Huang *et al.*, 2008; Singh *et al.*, 2008; Wu and Nevatia, 2007) rely on the Hungarian algorithm proposed by Kuhn (1955). Another popular optimization process employed for the global data association task is

based on the general min-cut/max-flow network paradigm (Berclaz *et al.*, 2011; Izadinia and Shah, 2012; Leal-Taixe *et al.*, 2011; Pirsiavash *et al.*, 2011; Zhang and Nevatia, 2008), where the tracking problem is modeled as an optimization of a flow network. The nodes of the network represent the individual detections and frame-to-frame links represent the affinity between the corresponding detections. In order to obtain the optimal assignment of detections to trajectories Zhang and Nevatia (2008) use a min-flow algorithm from Goldberg (1997) which is repeatedly applied with different amounts of flow (equivalent to number of objects), inferring occlusions and associations iteratively. In order to cope with long term occlusions, besides differentiating the value of flow, the graph is expanded with possible occlusion hypotheses that are linked with observed tracklet pairs if consistent with the appearance and scale. The complexity of the employed optimization procedure (Goldberg, 1997) is polynomial in the number of frames. However, recently Pirsiavash *et al.* (2011) proposed to solve the multi-object tracking problem formulated as a flow network using a greedy algorithm that sequentially instantiates tracks using a shortest path procedure, which results in linear time complexity in the number of frames and the number of objects. Andriyenko and Schindler (2011); Andriyenko *et al.* (2012); Yamaguchi *et al.* (2011) propose to formulate the data association problem as minimization of a continuous energy function. The energy function consists of different terms that model a desired configuration for a pedestrian trajectory by linking detections plausibly, with regard to pedestrian dynamics, collision avoidance and object persistence. Trying to approximate the most realistic trajectory configuration that reflects the real world scenario, the energy functionals usually become highly non-convex. To cope with non-convexity Andriyenko and Schindler (2011) propose to optimize the energy function using a conjugate gradient method which is augmented with trans-dimensional jumps, allowing to jump out of local minima and thus to find a better configuration that minimizes the energy. Yamaguchi *et al.* (2011) however, employ a variant of the simplex algorithm combined with several restarts in order to escape the local minima.

So far, the presented *tracking-by-detection* approaches perform tracking without taking into account the interaction between the individual persons, having weak constraints on the pedestrian motion, assuming a constant velocity model for human dynamics. However, usually in a real world scenario human behavior is influenced by many factors such as the intended goal, other scene objects and obstacles. Based on this fact, several approaches have been proposed to model the human motion by incorporating physical and social constraints of the surroundings (Leal-Taixe *et al.*, 2011; Luber *et al.*, 2010; Pellegrini *et al.*, 2009; Yamaguchi *et al.*, 2011). Originally, social force models were employed for crowd simulations (Helbing and Molnár, 1995; Klügl and Rindsfuser, 2007; Saboia and Goldenstein, 2011) modeling the behavior of pedestrians in evacuation scenarios or modeling plausible dynamics of crowds of virtual pedestrians in computer graphics (Heigeas *et al.*, 2003; Lerner *et al.*, 2007).

Helbing and Molnár (1995) propose to model the dynamics of pedestrians by *social forces*. In particular, the assumption is made that each object in the scene emanates

a repulsive force. These forces represent the fact that humans always want to keep a certain minimum distance to other scene participants and static objects. Consequently, the movement of each pedestrian is constrained, by forces from other scene objects, while moving to the desired destination with a desired speed. The proposed *social-force* model consists of three parts: the first part models the acceleration towards the desired velocity of motion; the second part models the repulsive effect of walls and other people; and the third part models an attraction energy emanated by the motivation of pedestrians to reach a certain goal. Helbing and Molnár (1995) show that a weighted combination of these terms allows a very realistic pedestrian dynamics simulation. Inspired by this social force concept, Luber *et al.* (2010) propose to integrate it into a multi-hypothesis target tracker using only measurements from a laser range finder. To this end, the social force model is combined with the prediction step of a Kalman Filter, pulling the prediction to a position in the scene that is consistent with the individual goal and desired speed and accounts for the influence of the environment and other people. The presented experiments show that using social forces results in a more realistic prediction model of human motion, reducing the data association error especially after occlusions. Parallel to the approach of Luber *et al.* (2010), Pellegrini *et al.* (2009) propose a social force model based approach (Linear Trajectory Avoidance) for motion prediction in the context of visual multi-person tracking. The proposed approach models the dynamics of each pedestrian based on an energy field which is affected by three different terms similar to the work of Helbing and Molnár (1995): interaction cost, desired speed and direction and intermediate goal. In particular, they define an individual energy field for each pedestrian, which is a function over the possible velocity vectors the person can choose. Consequently, in each frame for each pedestrian the decision about the moving direction and velocity is the minimum inside this individual energy field which makes these pedestrians move in the optimal direction. For computing the interaction cost, the point of closest approach (obtained by a simple linear extrapolation of the given trajectory so far) to other pedestrians in the near future is used, rather than just the current positions as in (Helbing and Molnár, 1995). This point of closest approach is then used in order to adapt the speed and walking direction in order to minimize the collision likelihood. In the course of his master thesis project, Fischer (2012) reimplemented both state-of-the-art approaches (Luber *et al.*, 2010; Pellegrini *et al.*, 2009) and integrated the proposed models into the prediction step of our *high-level* tracker presented in Chapter 3. The disadvantage of the approach from Luber *et al.* (2010) is the modeling of energy potentials only at pedestrians current location resulting in a late effect of the forces and consequently very abrupt direction change as soon as the pedestrians are really close to each other. Modeling the driving force from the point of closest approach as proposed by Pellegrini *et al.* (2009) results in more plausible pedestrian trajectories. Since this point of closest approach is usually further away from the current position, the path and velocity adaptation begins earlier yielding much smoother trajectories compared to the results from Luber *et al.* (2010). A further interesting fact we derived during the evaluation of both approaches on our sequences is that social forces can describe and

predict human interaction only for scenarios where the objects are moving independently and not in groups. Pedestrians moving in groups violate the social force rules generating an undesired repulsive effect that yields a wrong prediction. Leal-Taixe *et al.* (2011); Yamaguchi *et al.* (2011) have recognized this problem and first perform group detection before applying social force models for prediction of the future motion. This is also a motivation behind our group detection and interaction type classification approach developed in Chapter 9.

2.3. Stereo-based Tracking

As already mentioned in the previous chapter, the enormous progress in object detection (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b), made the development of robust *tracking-by-detection* approaches possible in the first place (Andriluka *et al.*, 2008; Ess *et al.*, 2009b; Gavrila and Munder, 2007; Huang *et al.*, 2008; Leibe *et al.*, 2008a; Okuma *et al.*, 2004; Wu and Nevatia, 2007). However, classical *tracking-by-detection* methods typically require the approach to execute a computationally expensive object detector in each frame, making it hard to achieve real-time performance at the system level. Many object detection approaches targeted at real-time applications follow a simple strategy in order to alleviate this problem by extracting ROIs based on object motion (Enzweiler *et al.*, 2008), (not applicable for our scenes, since our camera is also moving) or texture content (Shashua *et al.*, 2004) to reduce the detector search space. In our work, we mostly rely on depth as an additional cue in order to constrain object detection to small ROIs, similar to (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007; Geronimo *et al.*, 2010a). These ROIs are extracted in each frame and are evaluated by the detector to feed a tracking-by-detection process.

Bansal *et al.* (2010) extract ROIs by projecting the 3D points from a stereo depth map onto the estimated 2D ground plane. The local maxima of this projection are backprojected to the image, forming the ROIs which are evaluated in each frame by the detector. The detector output is then associated to trajectories using a correlation tracker. Since only a small number of ROIs is processed in each frame, their approach nearly reaches real-time performance. Similar to Bansal *et al.* (2010), Bajracharya *et al.* (2009a) uses range data from stereo in order to generate regions-of-interest. Using shape features that are extracted from 3D points of ROIs, pedestrians are detected. Since the strategy of Bajracharya *et al.* (2009a) is to robustly detect and track pedestrians in open land scenes with few potential ROI candidates, they did not track individual objects, but ROIs (usually consisting of several pedestrians in crowded scenarios). The goal of tracking ROIs is to reduce the number of false positives due to aggregation of the classification information of single frames and to estimate the velocity of the ROIs. The association of ROIs from frame-to-frame is performed by simply matching the color histograms of the individual ROIs, extracted from the corresponding 2D positions in the image. Gavrila and Munder (2007) constrain the search space using ROIs for generating

detection hypotheses. Detection hypotheses are represented by measuring the Chamfer distance between a learned shape contour model and the image input, which are then verified by cross correlation between the two stereo images. For trajectory generation Gavrilu and Munder (2007) employ a simple α - β filter (Benedict and Bordner, 1962) (closely related to the Kalman Filter) that propagates the uncertainty of the bounding box position and the depth, in a predict-update manner. Similar to our trajectory generation process presented in Chapter 3.5 the resulting hypothesis set can contain multiple-tracks with one and the same measurement assigned. In order to obtain an optimal hypothesis set Gavrilu and Munder (2007) use the aforementioned Hungarian algorithm (Kuhn, 1955). Luber *et al.* (2011) propose a 3D pedestrian detection and tracking approach based on RGB-D data from a Microsoft Kinect camera. Pedestrians are detected by combining the popular HOG detector (Dalal and Triggs, 2005) applied to the color image with a depth image based detector, Histograms of Oriented Depths (HOD). Both detectors are executed independently and the resulting output is fused using a weighted mean. The detections are then fed into an online-tracking system proposed by Grabner *et al.* (2006) which learns the model of tracked objects based on a boosting framework. The output of the detector and the online-tracker is used in order to generate observations for a Multi-Hypotheses Tracker (MHT) similar to (Reid, 1979) that generates and maintains long-term trajectories. This approach is similar to our proposed hybrid-tracking approaches presented in Chapter 4. On the one side the online-tracker is a low-level tracker that in case of false negatives generates new observations which are fed to the MHT (high-level tracker). The high-level tracker on the other side drives the low-level tracker to updated object positions and thus reduces the sensitivity to drift, typical for online image-based trackers.

2. *State of the Art*

In this thesis we focus on the development of tracking approaches for mobile platforms equipped with a stereo camera. Stereo cameras allow us to extract an additional cue, the depth, which we will show to be very useful over the course of the thesis. We started our experiments with already published datasets captured with a stereo rig mounted on a child stroller in urban scenarios, courtesy of Ess *et al.* (2009a). In addition, we captured more challenging datasets, especially required for the evaluation of our novel *tracking-before-detection* approach for tracking of unknown objects (cf. Chapter 8).

In this chapter, we will present and discuss a subset of approaches we have chosen as components (*stereo*, *visual odometry*, *object detection* and *multi-hypothesis tracker*), which we constantly employed while developing our tracking frameworks. In the following Sec. 3.1, we will introduce a subset of pedestrian detection approaches we used. Then, in Sec. 3.2 we will discuss stereo estimation and explain the use of depth information in our tracking frameworks. In Sec. 3.3 we will shortly present the visual odometry which is used as a necessary means in our framework, allowing us to reason about object trajectories in world coordinates. In Sec. 3.4 we will introduce the Multi-Hypothesis Tracking approach by Ess *et al.* (2009b) which served as basis for our own extended reimplementation. In Sec. 3.5 a real time *tracking-by-detection* approach will be presented combining all of the aforementioned components in a unified framework. Finally, in Sec. 3.6 we will describe the datasets we have captured and used for systematical evaluation of our frameworks.

3.1. Object Detection

The ability to reliably detect pedestrians in real-world images made the progress in *tracking-by-detection* possible in the first place. Pedestrian detectors proposed in recent years have reached remarkable detection performance in street scenes (Benenson *et al.*, 2012; Dalal and Triggs, 2005; Dollar *et al.*, 2010; Felzenszwalb *et al.*, 2010b). Even though these approaches rely on conceptually simple Histograms-of-Oriented-Gradients



Figure 3.1.: Example results of the pedestrian detectors on the BAHNHOF sequence. First row corresponds to the output of the detector proposed by (Felzenszwalb *et al.*, 2010b). Second row (Sudowe and Leibe, 2011) and the last row shows the output from our upper body detector presented in Chapter 6.

(HOG) features, they still reach the best performance for fully observed pedestrians as shown in (Dollar *et al.*, 2009).

The HOG descriptors were originally presented by Dalal and Triggs (2005) in context of a sliding window pedestrian detector. The central idea behind HOG features is that the local object appearance and shape of an object can be robustly described by the distribution of gradients and their magnitude. The distribution is described by histograms that group the gradients with respect to their orientation weighted by the magnitude into bins. These histograms are computed for each squared cell which decompose the image in a dense uniformly spaced grid. In order to reach better illumination invariance Dalal and Triggs (2005) proposes to *contrast-normalize* the cells using a block-wise pattern before concatenating them to a descriptor. To this end, the histograms of the cells embedded in a block (usually consisting of 2×2 cells) are accumulated and all cells are normalized by this accumulated value (cf. Fig. 3.2). The extracted features are then

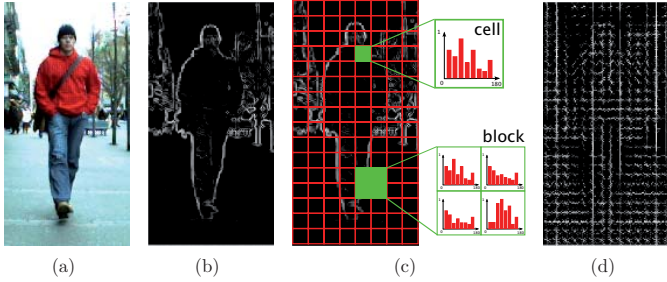


Figure 3.2.: HOG features extraction pipeline. (a) Original image - cropped from first frame of the BAHNHOF sequence. (b) Gradient image. (c) Decomposition of the image into small squared cells. Cells are represented by gradients binned into histograms. Cell histograms are contrast-normalized by the intensity across a block (usually 2×2 cells). (d) Resulting HOG feature.

used in order to train a discriminative, linear SVM classifier. The SVM performs a binary decision for a given image window, whether this window contains an object or not. The decision is repeated during the test procedure for all rectangular windows, at each possible position, by sliding the window over entire image. In order to detect objects at different scales the image is rescaled for several scale strides and the decision process is repeated. The final detections are obtained after non-maximum suppression, which is necessary due to the multi-scale approach. This introduces several additional detections on a person for a number of neighboring scales.

A disadvantage of using HOG features is the high computational effort, making approaches relying on HOG features quite expensive to evaluate and thus limits their use in mobile platforms. The high computational costs are related to the multi-scaling required in order to detect pedestrians at different scales. For each rescaled image the feature extraction procedure needs to be repeated. Generally, the image is not only downscaled, but also upscaled in order to detect pedestrians which are far away from the camera.

Many approaches have been proposed in recent years to speed up detection tasks, including detection cascades (Felzenszwalb *et al.*, 2010a; Viola and Jones, 2004), other and more efficient feature representations than HOG (Benenson *et al.*, 2012; Dollar *et al.*, 2010), and alternatives to the sliding-window search strategy based on ROI extraction relying on, e.g., stereo range data (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007), motion (Enzweiler *et al.*, 2008) and scene geometry (Geronimo *et al.*, 2010a). In order to cope with the problem of partial occlusions, several frameworks have been proposed combining object part detector outputs in a *mixture of experts* manner (Enzweiler *et al.*, 2010; Wojek *et al.*, 2011).

In this thesis, we employed four different publicly available human detectors (Benenson *et al.*, 2012; Felzenszwalb *et al.*, 2010b; Prisacariu and Reid, 2009; Sudowe and Leibe, 2011) and our proposed upper body detector (Mitzel and Leibe, 2012a), which will be explained in detail in Chapter 6. The first two detectors (Prisacariu and Reid, 2009; Sudowe and Leibe, 2011) that we used are efficient reimplementations of the popular HOG feature based approach proposed by Dalal and Triggs (2005). Both reimplementations use the GPU in order to speed up the computationally expensive components of the pipeline such as the feature computation. The reimplementation proposed by Sudowe and Leibe (2011) includes several other extensions, such as limiting the computational efforts to small corridors in the image plane that were estimated by exploiting the given scene geometry, ground plane and camera parameters.

Another detector employed in our recent work is a part-based detector from Felzenszwalb *et al.* (2010b). This uses HOG feature based classifiers in order to detect the body parts and to combine them in a probabilistic model for the final detection.

During the integration of our tracking framework within the European Project EUROPA, we employed the detector from (Benenson *et al.*, 2012), that is based on an efficient detector from (Dollar *et al.*, 2010), but proposes several extension in the detection pipeline for reducing the number of image scales by approximating the image features. By using a Stixel representation (Badino *et al.*, 2009) without dense depth computation, the detector evaluation area can further be reduced to small image regions. These extensions led to a pedestrian detector that runs with 100 fps on a GPU.

3.2. Stereo Estimation

The association of detected objects is usually performed in 3D world coordinates, which enables us to employ physically plausible motion models for the individual objects. The motion model becomes particularly important when the object detector fails, in which case it can continue to predict the most likely current object position.

There are several options in order to convert the detector output (bounding box) into 3D world coordinates. An often used approach in single camera setups is to compute the intersection of a ray through the bounding box footpoint and an estimated ground plane (Ess *et al.*, 2009b; Gavrilu and Munder, 2007; Leibe *et al.*, 2008a). This approach works reasonable well if the bounding box footpoint and the ground plane can be estimated precisely. However, due to imprecise scale selection during the non-maximum suppression process of the object detector and the approximative fit of the ground plane, the 3D positions are usually very noisy, as can be seen in Fig 3.3(c). In several cases detections jumps more than 0.5m from one frame to the next, meaning that they move at a speed of approximately 25km/h, given the framerate of 14 fps. Although both pedestrians are moving at around 4km/h, in several frames the projected positions are oscillating around the same point on the ground plane. However, given the stereo data, the obtained 3D positions are much smoother and more accurate, as shown in Fig. 3.3(d).

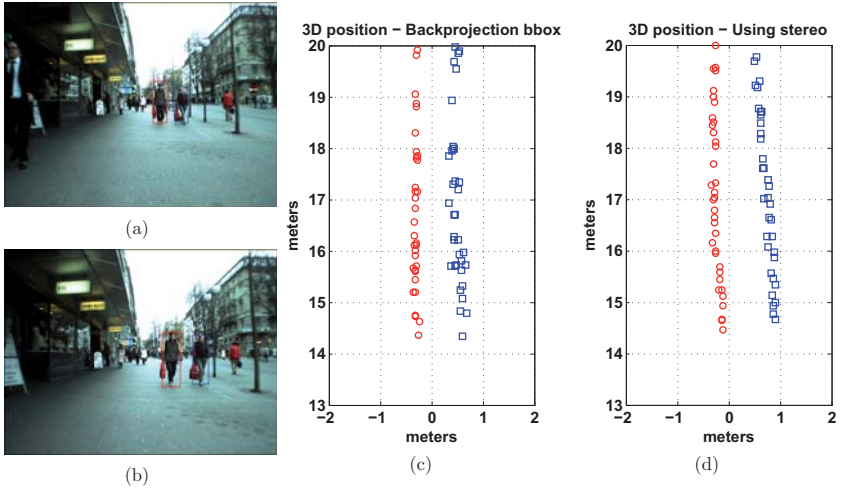


Figure 3.3.: (a,b) Detector output using HOG based implementation by (Sudowe and Leibe, 2011). (c) 3D positions on the ground plane obtained by intersecting the bounding box foot point with the ground plane. (d) 3D positions on the ground plane by employing stereo information.

Laser scanners are often employed in robotic scenarios, since they yield precise and reliable distance measurements. Fusing camera and laser information allows for a more precise retrieval of the 3D position of detected pedestrians. Generally, this is achieved by using laser distances from a front laser on a robot which are back-projected into the image plane. The z-values of the detected objects are obtained by taking the minimum of the laser points falling inside the detection bounding box, as shown in some example images in Fig. 3.4. With such a fusion the association step of a classical *tracking-by-detection* approach can be improved significantly, especially for distant objects. Furthermore, in context of a tracking system this fusion has one more advantage, the tracking results can in turn be employed by the local planning system of a robot. In particular, the tracker output can be used for annotation of the laser points as being objects of interest.

Using laser scanners for distance measurements eliminates most issues arising with stereo data (e.g., failures of algorithms in homogeneous and low-textured image data or reflections in shopping windows), but it introduces different problems. Many current laser range sensors yield measurements for few horizontal planes with a relatively low radial resolution. This means that thin structures of target objects such as pedestrian legs will often not be hit and the resulting distance inside the detection bounding box will correspond to a different object at a further distance. We can cope with this problem

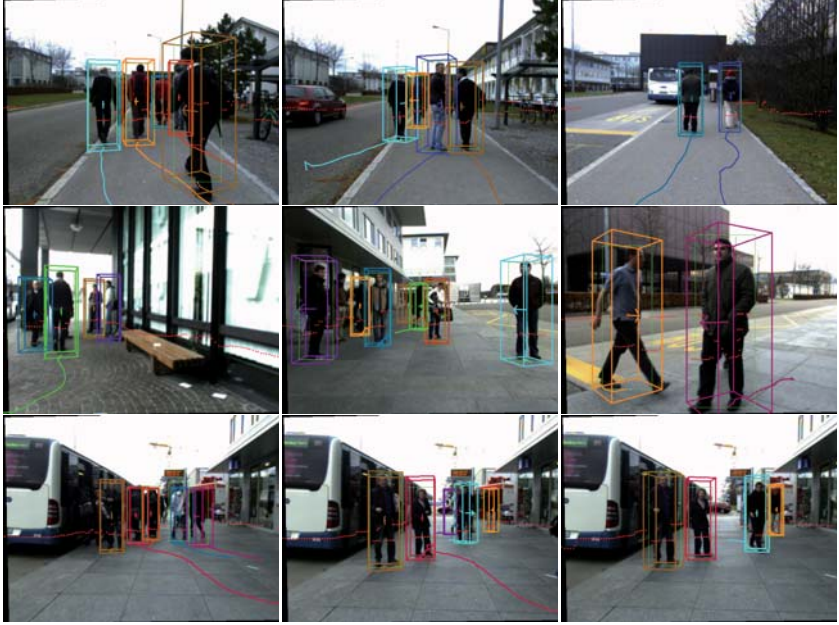


Figure 3.4.: Tracking images showing the back-projected laser information (red dots) falling inside the detection bounding boxes. These distance measurements from the laser were used for a precise 3D position estimation of pedestrians.

by assuming some correlation between the projected bounding box footpoint and the retrieved laser distance.

Basic Stereo Estimation Pipeline. For a given pair of images from a stereo setup, the goal is to estimate the depth value for each pixel. For this, stereo algorithms usually perform a search for corresponding points or patches (small image regions) based on similarity in intensity or in orientation. The correspondence search is usually reduced to a 1D search by rectifying the two images. The rectification sets the epipolar lines of the images parallel to the rows, such that the y-coordinates of the corresponding points become the same. The output of the stereo algorithm is a disparity image, where each pixel states the distance d of x-position in the left image to the x-position of the corresponding point in the right image. Then given the focal length f and the camera baseline B we obtain the depth by

$$z = \frac{fB}{d}. \quad (3.1)$$



Figure 3.5.: (a) A rectified stereo pair, where the red line shows that the corresponding points have the same y-coordinate. (b) Corresponding disparity image obtained with the approach from (Geiger *et al.*, 2010).

In this thesis we relied on two different stereo approaches (Felzenszwalb and Huttenlocher, 2006; Geiger *et al.*, 2010). The approach by Felzenszwalb and Huttenlocher (2006) is a global MRF-based approach and has the advantage that due to enforced smoothness in neighborhood, the resulting stereo estimation is dense and accurate. The MRF-formulation for stereo, however, in general yields an NP-hard optimization problem due to a large label set which needs to be approximated by graph cut (Boykov *et al.*, 2001) or belief propagation based techniques (Weiss and Freeman, 2001). However, both approximation approaches limit the application in real time scenarios due to space and time complexities (20-30 sec. on a 640×480 image).

A second method we used for the experiments was recently introduced by Geiger *et al.* (2010) proposing a new approach that produces accurate stereo depth maps of comparable quality to global approaches, which allows performance close to real time. The approach first extracts a set of support points that can robustly be matched in both images. By performing triangulation on this set of points one obtains a strong prior on possible disparities and reduces the matching ambiguities of the remaining points around the support points. This allows for efficient exploitation of the disparity search space and yields accurate, dense reconstructions.

3.3. Visual Odometry

Our proposed tracking approaches in this thesis are based on a moving camera setup. That means that in order to be able to estimate objects trajectories in global world coordinates, we need to estimate the camera position in each frame. Often the task of the camera position estimation is performed based on wheel speed sensors or inertial measurement units (IMUs). However, the estimation of odometry based on wheel speed sensors is very imprecise. Especially when the robot is moving on slippery terrain and

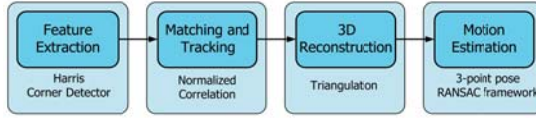


Figure 3.6.: Overview of the visual odometry pipeline presented by Nistér and Stewénius (2007).

spinning of the wheels will cause wrong motion estimation. The IMUs on the other side yield very precise localization estimation but are still very expensive compared to a stereo camera setup. One of the popular visual odometry approaches has been proposed by Nistér and Stewénius (2007), which we will introduce in more detail in the following section.

3.3.1. Basic Visual Odometry Pipeline

As illustrated in Fig. 3.6, the visual odometry estimation approach from (Nistér and Stewénius, 2007) is based on local feature extraction and tracking. Camera pose estimates can then be estimated from feature tracks using a RANSAC-based hypothesize-and-test framework.

Feature extraction. For each frame, the images are at first corrected regarding the radial distortion and are rectified in order to allow for efficient feature matching along the same scanline. Harris corners (Harris and Stephens, 1988) are then extracted from each of the images. The choice of this type of features is justified by their response stability and repeatability with regard to several image transformations as reported in (Mikolajczyk and Schmid, 2004). In order to increase the stability the images are divided into a 10 by 10 grid and a maximum of 30 features in each of the 100 buckets is extracted. The benefits of *bucketing* are that it allows to regulate computational complexity by using a smaller number of features which is an important prerequisite for real time applications. Furthermore, it guarantees that image features are well distributed containing features not only on independently moving objects, but also on the static background which is essential for robust ego-motion estimation.

Feature Matching and Tracking. For each of the extracted feature, a descriptor is created in form of an 10 x 10 pixel patch centered on the feature. The feature descriptors are then matched using normalized cross correlation. Here different descriptors can also be used e.g. SIFT-descriptors introduced by Lowe (2004). The choice of descriptors is usually constrained by the application itself meaning that for a real-time system usually simple patch based descriptors are preferred. Although other features can yield better results, they typically have higher computational complexity. The extracted features are matched between the left and the right image of the current frame, as well as the corresponding left and right image of the previous frame. Two features form a valid pair

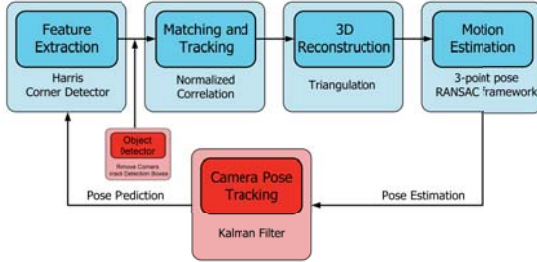


Figure 3.7.: Overview of the visual odometry pipeline presented by Ess *et al.* (2009b). Red boxes represent the extensions to the original visual odometry pipeline proposed by Nistér and Stewénus (2007).

only if they mutually match with each other. The matches from the current frame are then triangulated to form the corresponding points in 3D space.

Motion Estimation. Nistér and Stewénus (2007) propose to compute the pose of the stereo head from triplets of reconstructed landmarks (3-point pose). In particular, a RANSAC hypothesize-and-test framework evaluates each of the hypotheses for the camera pose. The landmarks are grouped into triplets and each of these triples gives a hypothesis for the camera pose. These hypotheses are then tested for all the other landmarks and are evaluated using the back-projection error as a scoring function and the best one is selected.

3.3.2. Extensions for Dynamic Scenes

In this thesis we relied on two different visual odometry approaches (Ess *et al.*, 2009b; Kitt *et al.*, 2010). The work by Ess *et al.* (2009b) extend the original work for visual odometry by Nistér and Stewénus (2007) for dynamic scenes, which in the original form is aimed for static scenarios. The difficulty in dynamic scenarios captured e.g., in shopping streets with many moving objects, is that the feature points are also extracted on the moving objects which will cause drift from the correct camera path. In order to compensate for this Ess *et al.* (2009b) propose to reject features that are localized on pixels corresponding to moving objects by discarding all the Harris corners which lie inside the bounding boxes extracted from one of the pedestrian detector described in Sec. 3.1. Furthermore, in the final stage of the original visual odometry pipeline, Ess *et al.* (2009b) employ a Kalman Filter with a constant-velocity motion model to track the pose of the stereo system, as suggested by (Ess *et al.*, 2009b; Geiger *et al.*, 2011). The constant-velocity Kalman tracker extrapolates the camera path in cases where the majority of the scene pixels are occupied by moving objects and thus limits the effect

of outliers. Fig. 3.7 shows again the extensions proposed by Ess *et al.* (2009b) that are interpreted into Nistér and Stewénus (2007) method. For our experiments we used the reimplementation by Floros and Leibe (2012) of the approach from Ess *et al.* (2009b).

The second stereo camera based ego-motion estimation approach we used in this thesis has been proposed by Kitt *et al.* (2010). In the first step it starts similar to Nistér and Stewénus (2007) approach with extraction of Harris corners (Harris and Stephens, 1988) from each of the images. The image is also divided in buckets and 30 features are kept in each bucket. As descriptors Kitt *et al.* (2010) use fixed-size blocks of image derivatives centered on extracted corners and avoid the usage of more complex features for efficiency reasons. Using the matched features the ego motion is estimated based on trifocal tensor which relates features between three images of the scene relying on the intrinsic as well as the extrinsic calibration parameters of the stereo rig. The motion estimated between two consecutive stereo image pairs is integrated into a Kalman Filter that can cope with non-linearities in the measurements similar to Floros and Leibe (2012). The possible measurements for the Kalman Filter are generated in a RANSAC framework where the possible motion is estimated by sampling 3 feature correspondences. After the Kalman Filter converges, inliers are computed using the Euclidean re-projection error. Features that are above a certain error threshold are considered as outliers and are rejected. These rejected features are usually points lying on independently moving objects that cause drift from the real motion and are not considered for the final ego-motion estimation, which only uses the inliers of the best sample. The combination of Kalman Filter with RANSAC based outlier rejection results in a robust ego-motion estimation even in very dynamic scenes.

3.4. Multi-Hypothesis Tracking

Multi-object tracking from a moving platform is a challenging problem with a number of applications including robotics and automotive safety. In one sentence, multi-object tracking can be summarized as detecting objects in individual frames and then linking these detections across frames. Usually such a tracking approach is more robust in contrast to a pure detector, since it can compensate for the occasional detection failure and ignores false-positive detections popping up in some frames. In this thesis we use an extended reimplementation of the hypothesize-and-select framework proposed by (Ess *et al.*, 2009b; Schindler *et al.*, 2010), which itself is based on the original work presented by Leibe *et al.* (2008a). The decision for this approach is based on its successful employment for pedestrian tracking in busy shopping street scenarios from a moving platform, yielding remarkable performance. Note that we use only the tracking part from (Ess *et al.*, 2009b; Schindler *et al.*, 2010) and do not employ the proposed graphical model which derives a consistent environment model based on underlying object detections, depth and the ground plane estimation. In the following we will refer to this tracking framework as *MDL-Tracker*.

As described by Bar-Shalom and Li (1995) a typical tracking system needs to perform three different data association types:

1. **Detection-to-detection association.**
2. **Detection-to-track association.**
3. **Trajectory-to-trajectory association.**

In the remaining part of this section we will present the *MDL-tracker* in more detail and always point out which part of the tracker is responsible for which data association type.

As input the *MDL-Tracker* requires object detection bounding boxes coming, e.g., from HOG-based object detectors presented in Sec. 3.1; camera localization at each frame from visual odometry; and ground plane estimation. Formally a detection is defined as:

$$o_{i,t_i} = [x_i, C_i^{3D}, t_i, h_i], \quad (3.2)$$

with x_i the objects's 3D ground plane position; t_i the time frame it was obtained at; C_i^{3D} the covariance matrix representing the positional uncertainty (the further the location is from the camera, the higher is the localization uncertainty); and h_i its appearance represented by, e.g., an RGB color histogram. The 3D positions of the objects are obtained by projecting the foot point of the detector bounding box using the current camera position (from visual odometry) and the ground plane estimate, (Fig. 3.8(a)). The observation uncertainty C^{3D} for each 3D point is computed using error back-propagation, assuming a covariance $C_{2D} = [\sigma_x \ 0; \ 0 \ \sigma_y]$ on pixel measurements:

$$C^{3D} = (F^{c1} C_{2D}^{-1} F^{c1} + F^{c2} C_{2D}^{-1} F^{c2})^{-1}, \quad (3.3)$$

where F^{c1} and F^{c2} are the Jacobians of the projection matrices of the individual cameras. The uncertainty increases with the distance between the 3D point and the camera position. The basic units of a tracker are trajectory hypotheses for possible object tracks. Such a trajectory hypothesis is defined as

$$H_n = [I_n, M_n, A_n], \quad (3.4)$$

with I_n the supporting detections also called **I**nliers, M_n its **M**otion model and A_n its **A**pppearance model. The employed motion model M_n and appearance model A_n allow us to formally combine each trajectory hypothesis with a detection:

$$p(o_i|H_n) = p(o_i|M_n) \cdot p(o_i|A_n) \quad (3.5)$$

This means that in each frame we can measure how detections o_i fit trajectory candidates H_n .

3.4.1. Trajectory estimation

As a hypothesis appearance model the *MDL-Tracker* proposes an $(8 \times 8 \times 8)$ -bin color histogram in RGB space which is computed for each detection over an ellipse fitted inside the detection box. The pixel values are weighted with a Gaussian kernel, favoring pixels closer to the center. The similarity between a candidate detection and a trajectory hypothesis is defined as the Bhattacharyya coefficient between their histograms:

$$p(o_i|A_n) = \sum_{r,g,b} \sqrt{h_i(r,g,b)A_n(r,g,b)}, \quad (3.6)$$

with r, g, b being indices of the histogram dimensions. When a new detection is added to the trajectory hypothesis, the appearance model of the hypothesis is adapted as $A_n = (1 - \alpha)h_i + \alpha A_n$.

For describing the motion of a pedestrian the *MDL-tracker* proposes to use an Extended-Kalman Filter (EKF) (Gelb, 1974) assuming a constant-velocity motion model such that the state vector is defined as:

$$\mathbf{s}_t = [x_t, y_t, \theta_t, v_t], \quad (3.7)$$

where (x_t, y_t) is the position of the detection in 3D world space on the ground plane, θ_t the orientation of a pedestrian, and v_t the speed. The derived motion model which is the prediction function for the EKF is defined as follows:

$$f_{t+1}(x_t, y_t, \theta_t, v_t) = \begin{bmatrix} x_t + \delta_t \cdot v_t \cdot \cos(\theta_t) \\ y_t + \delta_t \cdot v_t \cdot \sin(\theta_t) \\ \theta_t + n_\theta \\ v_t + n_v \end{bmatrix}, \quad (3.8)$$

n_θ and n_v are the noise components for orientation and speed, respectively. Then given a current state position s_{t_n} of a trajectory hypothesis H_n , the score of an object o_i at a position x_i under the motion model of the trajectory is determined as:

$$p(o_i|M_n) = e^{-\frac{1}{2}(x_i-s_{t_n})(C_h+C_i^{3D})^{-1}(x_i-s_{t_n})}, \quad (3.9)$$

with C_h representing the system uncertainty of the Kalman Filter and C_i^{3D} corresponds to the observation uncertainty which is especially important in order allow robust association for objects far away from the camera, because their localization is inaccurate as we described in Sec. 3.2.

Trajectory Generation. In order to generate trajectory hypotheses, *MDL-tracker* runs bi-directional EKFs. Starting a EKF from each new detection, new trajectory hypotheses are created by associating new detections backwards in time with past detections in previous frames guided by the motion model. This process is visualized in Fig. 3.8(b) and corresponds to the first association type *Detection-to-detection association*, we mentioned earlier. Additionally, when new pedestrian detections become

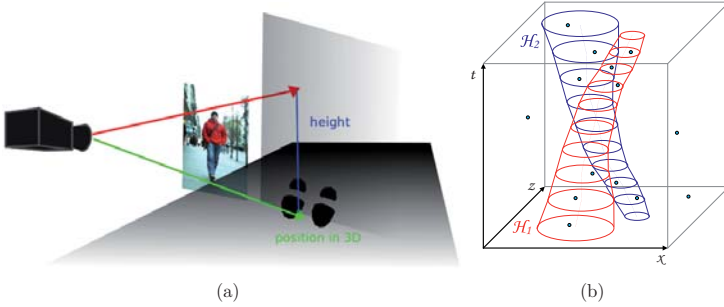


Figure 3.8.: (a) Computation of the 3D position and the height given the detection box and the scene geometry. (b) An example of trajectory generation by employing EKF. The blue hypothesis is one which was started in the new frame (with a high uncertainty represented by the size of the ellipse) and propagated backwards through previous frames. The red one is a hypothesis which was extended in the current frame.

available, the already existing trajectories (trajectories selected in the previous frame and assumed to be correct) are extended by predicting a new position and assigning a new observation closest to the prediction. The extension procedure corresponds to the second association type *Detection-to-track association* of the association scheme. Existing trajectories that could not be extended due to the lack of detections are extended using the so-far tracked uncertainty of the Kalman Filter. More formally the trajectory generation process can be explained by computing the association probability under the motion and appearance model $p(o_i|H_n) = p(o_i|M_n) \cdot p(o_i|A_n)$ for all detections in each frame and then assigning the one with the highest likelihood to the trajectory hypothesis H_n . In order to avoid unlikely associations caused by missing detections the *MDL-tracker* excludes associations that are below a certain threshold.

Trajectory Selection. During the independent generation process of the trajectories, a new detection can be used for extending and also for generating new trajectories. This can cause each detection to end up in several competing trajectory hypotheses. This leads to hypotheses that are not independent due to the constraint that two pedestrians cannot occupy the same spot on the ground plane and only one trajectory hypothesis can be selected. The goal of the trajectory selection process is to find an ‘optimal’ set of trajectories which is as small as possible but at the same time explains as many of the observations collected so far. To this end, for each trajectory H_n a score S_n is assigned that takes into account the score of the inlier detections and its likelihood under the motion and appearance model:

$$S = \sum_i p(o_{i,t_i}|A_n^{t_i}) \cdot p(o_{i,t_i}|M_n^{t_i}) \cdot p(o_{i,t_i}|I^{t_i}), \quad (3.10)$$

with $p(o_{i,t_i}|I^{t_i})$ representing the score of the detection coming from the employed detector. Finding the best subset of $\{H_n\}$ corresponds to a model selection problem and can be represented by a solution of a quadratic binary problem that yields a binary labeling representing a trajectory hypothesis with 1 if it belongs to the final set and 0 if not:

$$f = \sum_{1 \leq i \leq N} \sum_{1 \leq j \leq N} \mathbf{m}_i^T C_{i,j} \mathbf{m}_j \rightarrow \max, \quad (3.11)$$

$$\text{s.t. } \mathbf{m} \in \{0, 1\}^N, \quad (3.12)$$

with $\mathbf{m}$ the boolean vector labeling the selected hypotheses with $m_i = 1$. The diagonal elements c_{ii} of the matrix C are the scores of each H_n determined with Eq. 3.10, which is additionally reduced by a *model factor* ϵ expressing a preference for a solution with a small set of selected hypotheses. This means that the *model factor* defines a minimum score that a hypothesis must reach before being selected. The off-diagonal elements define the interaction cost between trajectory hypotheses consisting of two components. One is the penalty to the overlap of two trajectory footprints on the ground plane, modeling the physical assumption that two pedestrian cannot occupy the same spot on the ground plane at the same time. The second term is the correction of over-counting of detections assigned to both trajectories. Formally it is defined as follows:

$$c_{ii} = -\epsilon_1 + \sum_{o_{i,t_i} \in H_i} ((1 - \epsilon_2) + \epsilon_2 \cdot S(o_{i,t_i})) \quad (3.13)$$

$$c_{ij} = -\frac{1}{2}\epsilon_3 O(H_i, H_j) - \frac{1}{2} \sum_{o_{i,t_i} \in H_i \cap H_j} ((1 - \epsilon_2) + \epsilon_2 \cdot S(o_{i,t_i})), \quad (3.14)$$

with $O(H_i, H_j)$ representing the physical overlap of two trajectory hypotheses on the ground plane using a circular object dimension, with a 0.35 meters radius for a pedestrian. ϵ_2 defines a minimum support for each inlier detection and ϵ_3 weighs the influence of the physical overlap. Setting ϵ_3 to a high value when using the tracker in highly crowded scenarios can result in one of two pedestrian walking close together always being rejected. Thus, trajectory hypotheses interact through penalties if they compete for the same detections or if their space-time footprints overlap. Finding the maximum of Eq. 3.12 is NP-hard, but there are several methods that can find strong local maxima. The *MDL-tracker* here relies on an extended version of the multi-branch method by Schindler *et al.* (2006). For more details on the optimization we refer to the original publication.

Assigning Person Identities. As the model selection process selects the hypotheses independently, we require a method for assigning consistent person IDs to the chosen trajectories. This process correspond to the last association type *Trajectory-to-trajectory association* of the association scheme. In the case an extended trajectory is selected in the final hypothesis set, the assignment of IDs is simple. In the other case, we match the selected trajectory against the list of active tracks based on the number of occupied

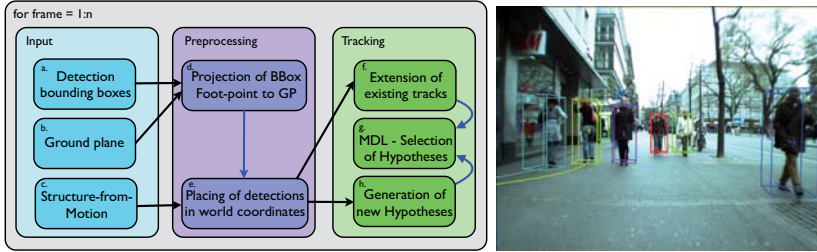


Figure 3.9.: Overview of the tracking pipeline.

observations. If the overlap is larger than 90%, the ID of the active track is assigned, otherwise a new track ID is created and assigned.

3.5. Real-Time Tracking-by-Detection

The development of powerful object detectors (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b; Leibe *et al.*, 2008b) pushed the progress for robust multi-person tracking approaches in challenging crowded scenarios (Andriluka *et al.*, 2008; Ess *et al.*, 2009b; Gavrilu and Munder, 2007; Huang *et al.*, 2008; Leibe *et al.*, 2008a; Okuma *et al.*, 2004; Wu and Nevatia, 2007). However, these approaches including the *MDL-tracker* presented in Chapter 3.4 are typically computationally very expensive and do not achieve real-time performance at the system level. For fulfilling the real-time constraint of robotic applications, many approaches follow a simple strategy of extracting ROIs based on motion (Enzweiler *et al.*, 2008), texture content (Shashua *et al.*, 2004) and stereo depth (Geronimo *et al.*, 2010a) in order to reduce the image area which should be scanned by the detector. Recently, two new ROI based approaches have been proposed, addressing the problem of real-time pedestrian detection from a moving vehicle. First, (Bansal *et al.*, 2010) use a structure classifier to extract candidate ROIs on which the pedestrian detector is applied, resulting in a frame rate of 10Hz. However, the system fails to handle crowded scenes in an urban environment and thus exhibits poor performance in such situations. Secondly, (Bajracharya *et al.*, 2009b) presented a stereo-based pedestrian detection and tracking system that runs at 5Hz. Although their framework works at acceptable error rates in semi-urban datasets, the lack of any occlusion handling mechanism causes this system to under-perform in cluttered urban datasets.

In our approach (Mitzel *et al.*, 2011b), we combine an extended version of the robust multi-hypothesis tracking framework by Ess *et al.* (2009b) described in Chapter 3.4 with a fast GPU-based object detector by Sudowe and Leibe (2011) described in Chapter 3.1 and an optimized visual odometry pipeline by Floros and Leibe (2012) introduced in Chapter 3.3 to a real-time vision-based multi-person tracking system working in crowded

	CPU-HOG	GPU-HOG	
		cudaHOG	groundHOG
Desktop	7-10s	44ms	18ms
Laptop	7-10s	139ms	38ms

Table 3.1.: Run-time performance of **cudaHOG** and **groundHOG** detector (Sudowe and Leibe, 2011) evaluated on a desktop (Core2Quad Q9550, GTX280) and a laptop (Core2Quad 1.86 GHz GTX 285M).

urban environments. In the following we will refer to this tracker as *High-Level tracker*. The *High-Level tracker* was also successfully employed in the European Project EU-ROPA mentioned in Chapter 3.2 providing tracking information of dynamic objects to the local planning system. In brief, the approach works as follows (cf. Fig. 3.9). Using the current camera position from visual odometry and the ground plane estimate, the detector output is transformed to 3D world coordinates (Fig. 3.8(a)). These detections are connected to generate an over-complete set of competing trajectory hypotheses, by starting new trajectories in each frame from the new detections and extending the existing trajectories (see Fig. 3.8(b)). For obtaining the final hypothesis set which best explains the current scene, we apply model selection in every frame similar to the *MDL-tracker*.

3.5.1. Extensions

On top of the *MDL-Tracker* we introduce several extensions in order to fulfill real-time constraints which we will discuss in the following. During the trajectory generation process of the *MDL-Tracker* many redundant trajectory hypotheses are generated since each new detection is used in order to generate a new hypothesis going backwards in time, but also existing trajectories are extended trying to associate these new detections. In most cases the hypothesis set contains two trajectories for one and the same object, where one is then selected during the model selection procedure. However, the MDL selection process is the most expensive component of the tracker and the processing time increases significantly with each additional trajectory hypothesis. Instead in the *High-Level tracker* we first perform the extension step and we exclude each detection that is associated to one of the trajectories for the new trajectory generation process. With this procedure we risk the extension of the existing trajectory to a wrong person, which cannot be recovered anymore. However, in order to minimize this risk we do not exclude the detection if the association probability determined with Eq. 3.5 is too low, and also start the generation of a new trajectory starting from this detection.

Another extension we proposed in Mitzel *et al.* (2011b) is the approximation of the unobserved state variables (θ_t, v_t) of the Kalman Filter based on the prediction and the current observation achieving higher robustness:

$$z_t(x_t, y_t, \theta_t, v_t) = \begin{bmatrix} x_t^o \\ y_t^o \\ \text{atan} \left(\frac{y_t^o - (y_t^p - \delta_t \cdot v_t^p \cdot \sin(\theta_t^p))}{x_t^o - (x_t^p - \delta_t \cdot v_t^p \cdot \cos(\theta_t^p))} \right) \\ \frac{1}{\delta_t} \cdot \sqrt{(x_t^o - (x_t^p - \delta_t \cdot v_t^p \cdot \cos(\theta_t^p)))^2 + (y_t^o - (y_t^p - \delta_t \cdot v_t^p \cdot \sin(\theta_t^p)))^2} \end{bmatrix}, \quad (3.15)$$

where the indices o and p indicate the observed and predicted data. During experiments we observed that when using this approximation, the predicted positions of the objects came closer to the real measurements, reducing the system covariance of the Kalman Filter significantly and thus allowing a better association. This can especially be observed, when setting the number of frames higher. When running the Kalman Filter backwards in time for generating a new hypothesis (around 250 frames) the obtained trajectories were much longer most of the time than without the change. This again indicates a more robust association. The effect of the proposed extension when assessing the performance of our tracker quantitatively was minimal which is due to evaluation criteria proposed by Ess *et al.* (2009b) we used at that time. The quality is measured in the image plane by the projection of the 3D positions of the tracked objects to the image and by then generating a bounding box which is compared to an annotated bounding box. As expected, such an evaluation criterion cannot account for a more precise 3D position estimation. We expect to have a much higher performance gain when using a 3D position based evaluation criteria which assumes the 3D position of the persons to be annotated (which is not trivial for the existing sequences).

3.5.2. Evaluation

In order to evaluate our framework, we applied it on the SHOPPING sequence captured in the city of Aachen and on three further challenging sequences BAHNHOF, SUNNY DAY and JELMOLI from the Zurich Mobile Corpus, described in Chapter 3.6. For validating the performance of our approach quantitatively, we employ the evaluation criteria proposed by (Ess *et al.*, 2009b). The quality is measured by the intersection-over-union between tracked objects and given annotations where matches with an overlap greater than 0.5 are treated as correct. In Fig. 3.10 we show the performance in terms of recall vs. false positives per image (fppi) for all four sequences. As can be seen, our approach achieves state-of-the-art performance on all sequences. In particular, it clearly outperforms the approaches by (Bajracharya *et al.*, 2009b; Bansal *et al.*, 2010) for the BAHNHOF and SUNNY DAY sequences. For the JELMOLI sequence we reach a better performance as the approach proposed by (Bajracharya *et al.*, 2009b), but slightly worse

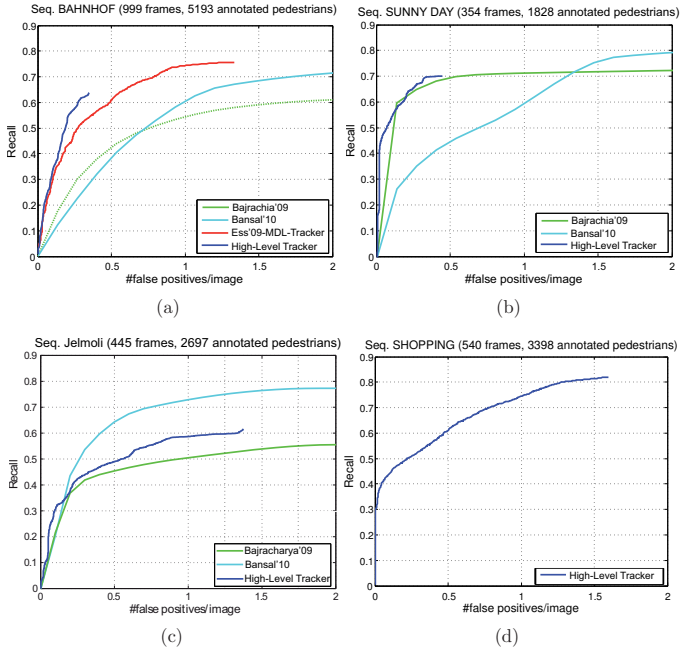


Figure 3.10.: (a) Quantitative tracking performance of our approach compared to the baselines from (Bajracharya *et al.*, 2009b; Bansal *et al.*, 2010; Ess *et al.*, 2009b) on the BAHNHOF sequence. (b) On the SUNNY DAY sequence. (c) On the JELMOLI sequence. (d) On our SHOPPING sequence.

performance than reported by (Bansal *et al.*, 2010). However, Bansal *et al.* (2010) did not use the original annotation set provided by (Ess *et al.*, 2009b) but created their own, leaving out some hard test cases. In addition, we report the high-level tracker’s performance on the SHOPPING sequence as a baseline for our later approaches to compare against.

Computational Performance. The detector is the most computationally expensive component of our framework. By outsourcing the expensive parts such as feature computation to a GPU and by utilizing the scene geometry, the *groundHOG* detector proposed by Sudowe and Leibe (2011) which we introduced Sec. 3.1 performs with more than 25fps (cf. Tab. 3.1) on a gaming laptop. The visual odometry module is optimized for multi-core processors and runs with 12Hz (cf. Chapter 3.3). The tracking component



Figure 3.11.: Results from our tracking framework on the BAHNHOF, SUNNY DAY, JELMOLI and SHOPPING sequences.

is the cheapest part running with more than 50fps. This means in total the full tracking pipeline runs at more than 10Hz on a laptop with Core2Quad 1.86 GHz, 8GB RAM and NVidia GTX 285M graphics card. In a different component configuration which we

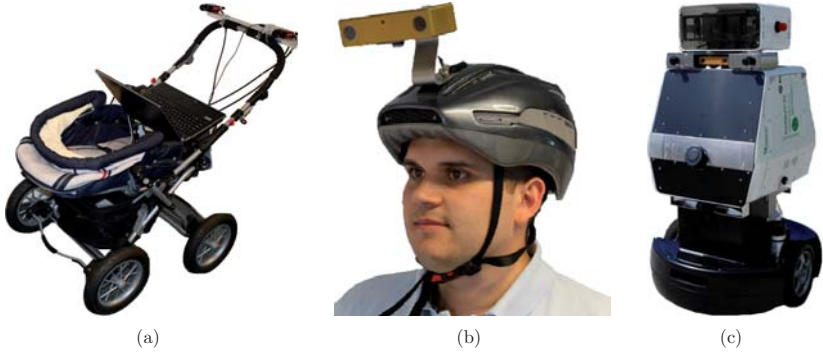


Figure 3.12.: Camera setups employed for recordings of the data sets.

employed during the EUROPA project, the full tracking pipeline ran with more than 18Hz. This configuration was composed of a pedestrian detector from Benenson *et al.* (2012) that runs with 100Hz and we used robot odometry instead of visual odometry.

Qualitative Performance. Fig. 3.11 presents some results from our real time tracker showing that our system can track most of the visible pedestrians correctly in the presence of short time occlusions.

3.6. Camera Setup

For historical reasons, we first used three popular datasets BAHNHOF, SUNNY DAY and JELMOLI from the Zurich Mobile Pedestrian corpus generously provided by the authors of (Ess *et al.*, 2009b) in order to assess the performance of our tracking frameworks. All three sequences were captured with a stereo rig at 13-14fps, with a resolution of 640×480 pixels. The BAHNHOF sequence (999 frames, with 5193 annotated pedestrians of ≥ 60 pixels height) was taken on a crowded sidewalk on a cloudy day. The SUNNY DAY sequence (999 frames, 354 of which are annotated with 1867 annotations) was captured on a sunny day and contains strong illumination changes. The JELMOLI sequence was acquired in a busy square, where pedestrians were moving in arbitrary directions (in contrast to the other two sequences). For all sequences stereo depth maps, structure-from-motion localization, and ground plane estimates are readily available.

Setup 1. In order to assess the performance of our *tracking-before-detection* system presented in Chapter 8 we required a more complex dataset from a shopping scene. To capture a new dataset, we built a similar child stroller setup as used by Ess *et al.* (2009b) (cf. Fig. 3.12(a)). Here we used a stereo rig with *Stingray* cameras with a resolution of 640×480 pixels at 15 fps. In order to obtain synchronized images from two cameras we

used an external trigger box controlled via an USB interface, generously provided by Ess *et al.* (2009b). The stereo rig was mounted on a customary child stroller at a height of 1.2 m and a baseline of 39 cm. The employed lenses had a focal length of 4.5 mm and a field-of-view of about 65°, with very low radial distortion artifacts. This setup was used for capturing image sequences in the center of Aachen which is a very crowded area due to many shops. By pushing the child stroller we basically simulated a directed walk of a robot.

With this setup we captured additional own sequences in busy shopping streets in Aachen. In contrast to the Zurich Mobile Pedestrian corpus, our sequences contain more complex scenarios with many unknown objects, such as child strollers, wheelchairs, suitcases, and walking aids. We annotated one of those sequences, noted in the following as SHOPPING in a similar fashion as Ess *et al.* (2009b), resulting in 3398 pedestrians annotations over 540 frames. Furthermore, we annotated 325 video subsequences consisting of 15,000 frames with person-object annotations and corresponding interactions like group, pushing, pulling, etc.. This sequence will be referred to as ACTIONS in the following.

Setup 2. With the second camera setup the goal was to obtain the images from a human view point. To this end, we employed a head mounted camera setup (Bumblebee2), where the camera was mounted on a cycle helmet as shown in Fig. 3.12(b). We used a Bumblebee2 stereo camera that delivered the images with a resolution of 640×480 at 15 fps. The lenses had the focal length of 3.8 mm and a field of view of around 65°.

Using this setup we captured a sequence TORSO in highly crowded central shopping streets during a sunny day in summer. We have annotated over 2,543 frames with over 19,461 pedestrian bounding boxes in order to assess the performance of our proposed upper body detector in Chapter 6.

Setup 3. The third platform used, was the EUROPA robot shown in Fig. 3.12(c), which was built during the European Project EUROPA¹. The stereo camera setup mounted on the robot was very similar to the aforementioned child stroller setup. We used the same cameras and the same lenses with low radial distortion. The camera height was also similar with 1.1m and a baseline of 40cm.

3.7. Discussion

In this chapter, we have presented a robust system for multi-person tracking in urban environments. The system builds on the framework presented by (Ess *et al.*, 2009b) extending and optimizing it towards real-time performance. In particular, we used an

¹EUROPA - European Robotic Pedestrian Assistant is a project funded by the European Commission within FP7 ((FP7/2007-2013) under grant agreement no 231888). The goal of the EUROPA project was to develop the foundations for service robots designed to autonomously navigate in urban environments outdoors as well as in shopping malls and shops to provide various services to users including guidance, delivery, and transportation.

optimized sliding window object detector proposed by Sudowe and Leibe (2011) employing geometrical constraints for reducing the search space. Furthermore, we employed an optimized and parallelized visual odometry method by Floros and Leibe (2012) which robustly estimates the camera pose using feedback from the object detector. Finally, we have extended a multi-hypothesis tracker with regard to speed, without losing accuracy in tracking performance. As our experimental results show, our system runs in real time, reaching state-of-the-art performance.

Classical *tracking-by-detection* approaches always require a robust object detector that needs to be executed in each frame. However, the detector is typically the most computationally expensive component, especially if more than one object class needs to be detected. We have shown in this chapter that outsourcing the computationally expensive components of the pipeline to the GPU can help in reaching the real time constraints, however, the power consumption of a GPU is very high, around 150W on a gaming laptop. This fact makes the application of GPU based object detectors for a robotic scenario difficult, due to limited power supply from the batteries, which reduces the robot's operational range and battery lifetime significantly. For this reason we will investigate in the following chapters how the usage of the object detector can be reduced by using low-level trackers.

I

Hybrid High-Level/Low-Level Tracking

Hybrid High-Level/Low-Level Tracking

4.1. Motivation

In this chapter, we present our approach to cope with the problem of a computationally expensive object detection component in a *tracking-by-detection* pipeline, which we motivated in the Introduction chapter. As mentioned before, a common *tracking-by-detection* pipeline requires a generic person detector which is applied to every frame of the input video sequence. The resulting detections of the past and current frames are associated into physically plausible trajectories based on some dynamic and appearance models. This leads to challenging data association problems, since the detections themselves may be noisy, containing false positives and misaligned detection bounding boxes. Intuitively, this complex data association seems to be at least to some degree an overkill. Once we have detected a person in one frame, we know its appearance and position and should be able to use this information in order to disambiguate future data associations. This has been attempted by using person-specific color descriptors (e.g. (Andriluka *et al.*, 2008; Ess *et al.*, 2009b; Leibe *et al.*, 2008a)) or online-trained classifiers (Grabner *et al.*, 2006). However, the difficulty here is that no precise segmentation is given – the detector bounding boxes contain many background pixels and the persons’ limbs may undergo considerable articulations, causing the classifiers to drift. We propose to address these problems by complementing the detection-based tracking framework with a robust low-level tracker, which is either based on a level-set (LS) segmentation (noted in the following as LS-tracker) or on an ICP registration of the 3D point clouds (noted in the following as ICP-tracker). In this integration, the *High-Level tracker*, presented in Sec. 3.5, only initializes new tracklets from object detections, while the frame-to-frame target following and data association is taken over by the computationally cheap low-level trackers. The resulting tracked target locations are then transmitted back to the high-level tracker, where they are integrated into 3D trajectories using physically plausible motion models. Such a combination allows to reduce the detector execution significantly, resulting in higher tracking frame rates and thus allowing to use a full track-

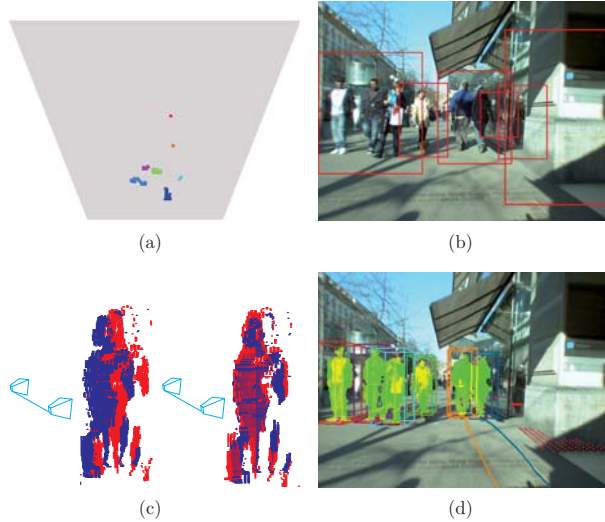


Figure 4.1.: Pipeline of the Hybrid-ICP tracker. (a) Given stereo range data, we first extract and track ROI candidates from the depth map (top down view). (b) Extracted ROIs in 3D are backprojected to the image. (c) Model (red points) and data (blue points) point clouds are aligned using ICP. (d) Results after high-level tracker association using the ICP tracker. Green overlay inside the box corresponds to the depth information that was used for computing the ROIs. Red points plotted on the ground plane represent the ROI foot points.

ing system on a mobile platform. To the full framework LS-Tracker combined with the high-level tracker we will refer in the following as Hybrid-LS tracker and the ICP-tracker combined with high-level tracker will be noted as Hybrid-ICP tracker.

The proposed combination of a LS-tracker with a high-level tracker is made possible by the great progress LS segmentation and tracking approaches have made in recent years (Cremers *et al.*, 2007). Approaches are now available that can obtain robust tracking performance over long and challenging sequences (Bibby and Reid, 2008). In addition, LS-trackers can be efficiently implemented using narrow-band techniques, since they need to process only a small part of the image around the tracked contour. However, the targeted integration is far from trivial. The LS tracking framework has originally been developed for following individual targets over time and has mostly been evaluated for tasks where a manual initialization is given (Bibby and Reid, 2008; Cremers *et al.*, 2007). Here, we need to automatically initialize a large number of tracklets from potentially

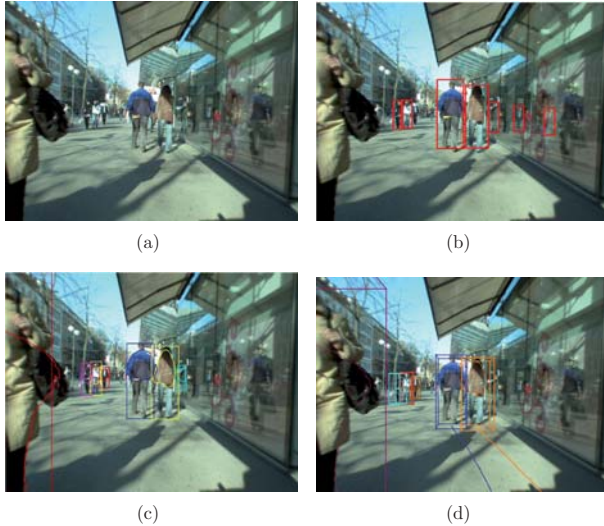


Figure 4.2.: Pipeline of the Hybrid-LS tracker. (a) Input image. (b) Raw object detections. (c) Level set tracking output, including the warped bounding box and the corresponding contour. (d) Resulting long-term 3D trajectories yield by the high-level tracker after associating the output of the LS-tracker.

inaccurate detections. In addition, we need to deal with overlaps and partial occlusions between multiple followed persons, as well as with tracker drift from changing lighting conditions and poor image contrast. Finally, we need to account for cases where a person gets fully occluded for a certain time and comes into view again a few frames later. In order to cope with those challenges, we propose several consistency checks that enforce a plausible trajectory evolution and in case of failures activate the reinitialization of tracklets with the execution of an object detector. Building upon remarkable progress in stereo estimation methods (Geiger *et al.*, 2010), which yield robust dense results in real-time, we investigated how using an ICP-tracker framework can further reduce redundant detector evaluations in a spatial, as well as in a temporal context. In contrast to the LS-tracking framework where the detector is applied on the entire image, having the depth information we could focus on extracted ROIs. The detector is then applied only to relevant parts of the image likely to contain target objects, thus reducing the computation time significantly. In addition, the propagation of ROIs in time allows us to limit detector evaluations only to newly appearing ROIs. Similar to the LS-tracking framework, the cheap ICP-tracker, which solves short-term data association for already

detected persons, takes over the role of the detector in supplying the high-level tracker with precise object location measurements.

Through the combination of low-level trackers with a high-level tracker and a careful interplay of the system components, we reduce the number of detector evaluations to a minimum, such that the whole tracking system runs at more than 10 fps, while reaching state-of-the-art tracking accuracy.

In detail, this chapter makes the following contributions:

- We demonstrate how LS-trackers or ICP-trackers can be integrated into a *tracking-by-detection* framework for robust multi-person tracking.
- Our approach is based on the idea to track each individual pedestrian by an automatically initialized LS- or ICP-tracker. We develop robust methods for performing this initialization from object detections and show how additional geometric constraints and consistency checks can be integrated into the low-level trackers.
- The resulting tracking positions are transmitted to the high-level tracker, which integrates the position evidence into a robust multi-hypothesis trajectory estimation approach making use of physical motion models. This high-level tracker is responsible for initializing new tracks, correcting the low-level tracker's predictions when drift occurs, and tracking person identities through occlusions.
- We experimentally demonstrate that this proposed integration achieves robust multi-person tracking performance in challenging mobile scenarios. In particular, as our approaches do not depend on continuous pedestrian detection, it can also continue tracking persons that are only partially visible.
- An interesting property of our integration is that it does not require the object detector to be executed for every video frame. This is especially relevant for the deployment on mobile platforms, where real-time performance is crucial and computational resources are notoriously limited. We experimentally investigate at what intervals object detections are still required for robust system-level performance.

The following section discusses related work. After that, Sec. 4.3 presents our proposed end-to-end tracking frameworks based on LS- and ICP-trackers. Sec. 4.4 introduces the basic algorithmic components for LS tracking and the integration with the high-level tracker, followed by experimental results. Sec. 4.5 then describes algorithmic components for the ICP-tracker and the interplay with the high-level tracker, concluded by experimental results. Sec. 4.6 concludes the chapter with a discussion about the hybrid tracking method.

4.2. Related Work

Various strategies have been developed for solving the challenging data association problems in multi-object tracking. However, most of them regard only a single-layer tracker

(Ess *et al.*, 2009b; Grabner *et al.*, 2006; Leibe *et al.*, 2008a; Okuma *et al.*, 2004; Wu and Nevatia, 2007), which sometimes makes the problem unnecessarily hard. Most directly related to our hybrid-approaches are the multi-layer models of (Kaucic *et al.*, 2005; Tao *et al.*, 2002), which also initialize a number of low-level trackers to follow individual objects and later integrate their results in a high-level tracker. However, their frameworks are based on aerial scenarios, where adaptive background modeling is still feasible. (Huang *et al.*, 2008) also propose a hierarchical data association framework that links detection responses to form tracklets at an image level, before fusing the tracklets and integrating scene constraints at higher levels. Their approach is however targeted at a surveillance application with a static camera. Ge and Collins (2008) integrate multiple short and low-confidence tracklet hypotheses into consistent tracks using MCMC. In contrast, our approach creates long and highly confident tracklets for individual persons based on low-level trackers and integrates them into an EKF-based multiple-hypothesis tracker. To our knowledge, ours is the first approach that integrates segmentation-based LS-trackers (Bibby and Reid, 2008; Cremers *et al.*, 2007) or ICP-trackers with a *tracking-by-detection* framework for street-level mobile tracking.

For real-time applications, there is a strong interest in reducing the computation time of the object detector, especially for automotive scenarios. Using the proposed LS-tracker we could skip the evaluation of the several video frames and still continue robust tracking due to the known appearance model of the pedestrians tracked so far. Another common strategy in automotive scenarios is to constrain the application of the detector only to those image regions which are likely to include the target objects. With this, we mean that the detector is not evaluated on the entire image plane but only on some promising image areas that need to be extracted first. Different approaches for extracting the ROIs were proposed based on motion (Enzweiler *et al.*, 2008), texture content (Shashua *et al.*, 2004), or (as already mentioned) stereo depth (Bajracharya *et al.*, 2009a; Gavrila and Munder, 2007). In contrast to (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007), in our ICP-based approach we do not evaluate each ROI by the detector in every frame, but propagate the ROIs over time and apply the detector only to newly appearing ROIs.

The idea to use depth information for tracking pedestrians has also been applied in several approaches before. Arras *et al.* (2008) present a pedestrian tracking approach based on single-plane scanner data by detecting and tracking legs separately with Kalman Filters, forming a multi-hypothesis set. The high-level tracks which consist of two legs are extracted from the multi-hypothesis set tackling the problem of occlusion and self-occlusion. However, in our approach we have to deal with dense stereo data, which comes with much higher measurement uncertainties than laser data. Most directly related to our ICP-based approach, Feldman *et al.* (2011) also propose an ICP based method for multi-object tracking in sports scenarios based on laser range data. This approach requires 4 calibrated laser scanners around the sports field to allow the extraction of the complete shape for each of the players. Additionally, a fixed elliptic shape model is assumed for detection and ICP tracking. However, this approach is not suitable for a

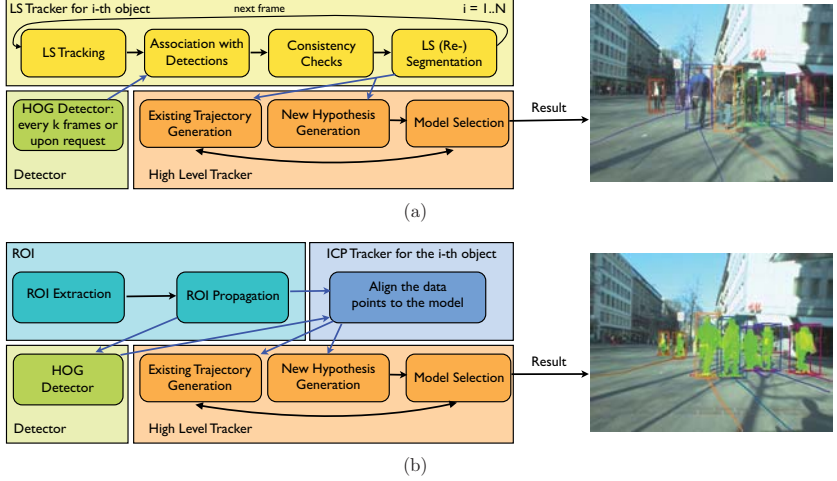


Figure 4.3.: System-level views of our proposed end-to-end tracking frameworks. (a) Overview of the different interactions between the components of our LS-tracker based tracking system. (b) Overview of the different interactions between the components of our ICP-tracker based tracking system. Blue arrows indicate the interaction between the individual components. Black arrows represent the interaction within the components.

mobile scenario, where only partial depth information from a single, front-facing stereo camera rig is available.

4.3. Integrated Tracking Frameworks

Fig. 4.3 shows a system-level overview of our proposed integrated tracking frameworks. Both systems are initialized by detections from a pedestrian detector. For each detected person, an independent low level-tracker is initialized, which follow this person’s motion either in the image space when using the LS-tracker, or in 3D when employing the ICP-tracker. In both frameworks the low-level trackers are kept robust through a series of consistency checks and they transmit the tracked person’s bounding box to the high-level tracker after every frame. The high-level tracker in turn converts the bounding boxes to ground plane coordinates and integrates them into physically plausible trajectories using the model selection framework, as described in Sec. 3.4.

During regular operation of the Hybrid-LS tracker, the object detector only needs to be activated in regular intervals in order to prevent existing tracklets from degenerating

and to start new ones for newly appearing pedestrians. The positive detections are then associated to already existing LS tracklets and for the remaining detection the LS-tracker segments the torso of the person inside a detection box. In the next frame, the segmented contour of the torso is tracked and the resulting warp is applied to the associated detection box in order to obtain the new object position. This new position, augmented by the object ID, is then fed into the high-level tracker that transforms it into 3D world coordinates and checks if it is still consistent with the appearance and the motion model in 3D. In addition, the LS-tracklets can request new detector evaluations when they become uncertain. Overall, this results in considerable computational savings, as we will show in Sec. 4.4.3.

In the Hybrid-ICP tracker, we avoid the execution of an object detector in regular intervals by tracking the ROIs (cf. Fig. 4.3, ROI Propagation component) using constant-velocity Kalman Filters. ROIs that could not be associated are assumed to have newly appeared in the scene and are evaluated by the detector. Successful detections are passed to the ICP tracker, which computes a 3D model for each detection. This model is represented by the 3D points which are within a pedestrian-sized cylinder placed on the ground plane, centered at the foot point of the detection. For already existing trajectories, the ICP tracker extracts the 3D points that are located within a cylinder at the modes of the 2D grid map within the trajectory’s Kalman Filter prediction covariance. It then aligns the model points from the previous frame to the newly extracted data points using ICP, resulting in a precise estimate of the new object location. From this new location, a new virtual detection is generated by projecting the position back to the image and transmitting it to the high-level tracker. Similar to the Hybrid-LS tracker the detections are passed to the high-level tracker and are used for generating new tracks and for extending the existing tracks based on an EKF using a pedestrian specific motion model.

4.4. Hybrid Tracking with Level Sets

In the following, we describe the two different low-level trackers and their integration in detail.

4.4.1. Level-Set Tracking¹

In order to follow detected pedestrians starting from an initial detection we rely on the level-set based segmentation and tracking approach proposed by Bibby and Reid (2008). Level-set is a contour based segmentation approach in which the contour C is

¹The Level-Set Tracker was implemented and evaluated by E. Horbert in context of her Diploma thesis, supervised by D. Mitzel and B. Leibe. The achieved results were published in our joint publication (Mitzel *et al.*, 2010) which will be presented in the following section.

represented implicitly as the zero level set of an embedding function $\Phi : \Omega \rightarrow \mathbb{R}$ (cf. Fig. 4.4(b)):

$$C = \{x \in \Omega | \Phi(x) = 0\}. \quad (4.1)$$

When presenting contour based segmentation, the *Snakes* approach proposed by Kass *et al.* (1988) has to be mentioned since it is one of the groundbreaking publications in image processing. The Snakes approach is the first variational approach for image segmentation which proposes to minimize the following functional:

$$E(C) = E_{ext}(C) + E_{int}(C) \quad (4.2)$$

with an external energy component:

$$E_{ext}(C) = - \int |\nabla I(C(s))|^2 ds \quad (4.3)$$

and an internal one:

$$E_{int}(C) = \int_0^1 \left\{ \frac{\alpha}{2} |C_s(c)|^2 + \frac{\beta}{2} |C_{ss}(s)|^2 \right\}, \quad (4.4)$$

with I the input image, $C : [0, 1] \rightarrow \Omega$ denotes a parametric curve and C_s , C_{ss} are the first and second derivatives of the curve with respect to s . In words, the external energy measures how well any given curve C coincides with the maxima of the image gradients. The internal energy is a regularizer which induces smoothness on the computed curves. However, compared to modern segmentation approaches, the Snakes approach is only of limited practical use. Due to many strong gradients in the image the contour often evolves to local maxima. As a consequence, the initial contour must be initialized sufficiently close to the final desired solution. Furthermore, the evolution of parametric curves is a numerically challenging problem as one needs to avoid self-intersections and instabilities. However, the level-set approach, does not require a specific choice of parametrization and secondly the topology of the contour is not fixed. Due to non-convexity usually we also only obtain a local optimum. However, for our approach it is not an essential drawback, since we use a re-initialization procedure, which we will introduce later on, that requests a detector evaluation once the contour is not consistent with tracking.

Before presenting the implementation details of the LS-tracker we will first shortly introduce the approach by Bibby and Reid (2008) we used as basis for following initial pedestrian detections over time. The approach is color based and in an initial step builds a foreground model using pixels contained in the bounding box obtained from a starting detection. Pixels from an inflated bounding box are used to build the background model. These two initial distributions are then used in order to perform the segmentation. The segmentation is an iterative process that moves the contour excluding some pixels from foreground to background or vice versa and in turn rebuilds the model. This procedure

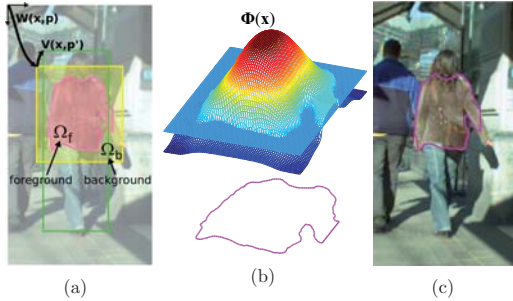


Figure 4.4.: Level-set segmentation. The contour separates object Ω_f from background Ω_b in a reference frame given by the warp $W(\mathbf{x}, \mathbf{p})$, which is related to the person’s bounding box by the displacement $V(\mathbf{x}, \mathbf{p}')$. This contour is the zero level-set of the embedding function Φ .

is iterated until the contour converges. Before performing the same segmentation step in the next frame, Bibby and Reid (2008) propose to perform frame-to-frame registration first. To this end, a rigid registration is computed that only optimizes the pose while keeping the contour itself constant. The goal is to find a warp into the next frame such that when placing the contour in the next frame it best fits the learned color models. After the registration step a segmentation step is performed, which usually converges after few iterations.

Segmentation. More formally, the segmentation consists of three parts that penalize the deviation from the foreground and background model, two regularizers which penalize the deviation of the embedding function from a signed distance function (Li *et al.*, 2005), and one that induces smoothness of the contour by penalizing the length of the contour. An initial segmentation is performed by iteratively optimizing the following energy functional with the following gradient flow (Bibby and Reid, 2008):

$$\begin{aligned} \frac{\partial P(\Phi, \mathbf{p} | \Omega)}{\partial \Phi} = & \frac{\delta_\epsilon(\Phi)(P_f - P_b)}{P(\mathbf{x} | \Phi, \mathbf{p}, \mathbf{y})} - \frac{1}{\sigma^2} \left[\nabla^2 \Phi - \text{div} \left(\frac{\nabla \Phi}{|\nabla \Phi|} \right) \right] + \\ & \lambda \delta_\epsilon(\Phi) \text{div} \left(\frac{\nabla \Phi}{|\nabla \Phi|} \right) \end{aligned} \quad (4.5)$$

where $P(\mathbf{x}_i | \Phi, \mathbf{p}, \mathbf{y}_i) = H_\epsilon(\Phi(\mathbf{x}_i))P_f + (1 - H_\epsilon(\Phi(\mathbf{x}_i)))P_b$, ∇^2 is the Laplacian, H_ϵ is a smoothed Heaviside step function and δ_ϵ its derivative, a smoothed Dirac delta function.

P_f and P_b are the pixel-wise posteriors of the foreground and background models given the pixel color. These models are represented with RGB color histograms and initialized from the detection bounding box. The segmentation is performed in several iterations

and these models are rebuilt in every iteration, after every move of the contour. The second term enforces the smoothness of the height field Φ shown in Fig. 4.4(b). Note that in the original formulation proposed by Bibby and Reid (2008) the third term was not included, however, in our experiments we observed that only when using this term we could obtain a smooth contour well aligned with object borders. The difference of the third term to the second term is that the third term enforces the smoothness of the contour itself indicated by $\delta_\epsilon(\Phi)$ which is one only at the zero (contour) level and zero everywhere else. The second term enforces the smoothness of the entire level set function Φ .

Tracking. The tracking part aims at warping the next image frame such that it best fits the current level-set contour with regard to the pixel-wise posteriors P_f and P_b of the foreground and background models. It is also possible to skip the tracking part and place the level set function at the current position and simply perform the segmentation using the color models. However, by registering the level set function first, fewer iterations of the following, computationally more expensive segmentation step are required. The warp $W(\mathbf{x}, \mathbf{p})$ is a transformation of the reference frame with parameters $\mathbf{p}$, which model the type of the transformation. We restrict the transformation in our case to translation and scaling, but any transformation can be used here which forms a group (Baker and Matthews, 2004). Translation and scale transformation is sufficient for our application since we are working with sequences that are captured at a high frame rate at around 15fps. This means that the frame-to-frame shape difference of the objects of interest can be sufficiently modeled by a translation with scale transformation. For determining the location of the contour in the next image we optimize the following functional (Bibby and Reid, 2008) which yields the incremental transformation $\Delta \mathbf{p}$ applied to the embedded function:

$$\Delta \mathbf{p} = \left[\sum_{i=1}^N \frac{1}{2P(\mathbf{x}_i|\Phi, \mathbf{p}, \mathbf{y}_i)} \left[\frac{P_f}{H_\epsilon(\Phi(\mathbf{x}_i))} - \frac{P_b}{(1 - H_\epsilon(\Phi(\mathbf{x}_i)))} \right] \mathbf{J}^T \mathbf{J} \right]^{-1} \times \sum_{i=1}^N \frac{(P_f - P_b) \mathbf{J}^T}{P(\mathbf{x}_i|\Phi, \mathbf{p}, \mathbf{y}_i)} \quad (4.6)$$

with $\mathbf{J} = \delta_\epsilon(\Phi(\mathbf{x}_i)) \nabla \Phi(\mathbf{x}_i) \frac{\partial W}{\partial \Delta \mathbf{p}}$, where $\frac{\partial W}{\partial \Delta \mathbf{p}}$ is the Jacobian of the warp.

After performing the warp of the embedded function only two segmentation iterations are sufficient in order to take account of the shape changes.

Extensions using Depth Information. Bibby and Reid (2008) only use color information in their proposed formulation in order to model the foreground and background. However, during experiments we observed that the LS-tracker often diverges caused by image areas with low color contrast, when lighting changes occur, or when other people or background structures contain similar colors. To cope with this problem we propose to extend the original approach by including stereo depth information.

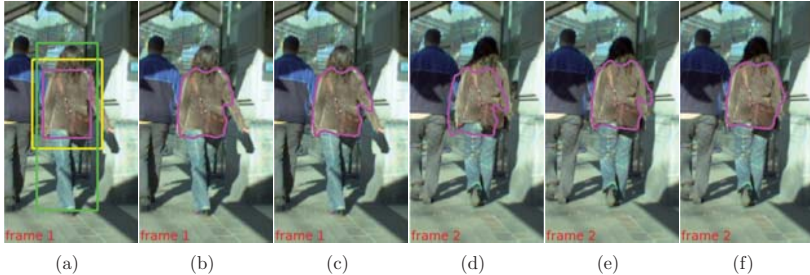


Figure 4.5.: Initialization of the LS tracker: (a) detection box (green), initial object frame (yellow), and initialization of the level-set (magenta) (b,c) evolved level-set after 40 and 150 iterations (d) level-set transferred to next frame (e) after warping (f) after shape adaptation (5 iterations).

We used depth information only in the segmentation component. We stored the median depth of the foreground area after convergence of the segmentation. For each depth value in the next image surrounded by the contour we then assigned a probability according to a Gaussian distribution around the median depth. With the Gaussian distribution we model the fact that in our case the objects are moving and thus the depth changes slightly from frame to frame. The two pixel-wise posterior probabilities for color and depth are individually normalized as in Bibby and Reid (2008) and then merged with a weighting factor α (set to 0.1 in all of our experiments).

$$P_i = (1 - \alpha)P_{i,color} + \alpha P_{i,depth}, \quad i \in \{f, b\}, \quad (4.7)$$

4.4.2. Combined Tracker

We now present the stages of our combined tracking framework. The difficulty of the street-level mobile tracking task brings with it a number of non-trivial challenges, which we address by consistency checks and carefully modeled interactions between the components of the tracking framework.

Object Detection. For pedestrian detection, we apply the widely used HOG detector (Dalal and Triggs, 2005) in the efficient *fastHOG* GPU implementation by (Prisacariu and Reid, 2009). Detections that are inconsistent with the scene geometry are filtered out by enforcing a ground plane corridor.

Level-Set Initialization. Upon initialization, the LS-tracker tries to segment the torso of the person inside a detection box. To this end, a new level-set embedding function is initialized with a rectangular box (see Fig. 4.5), and the level-set segmentation is iterated for 150 steps. In the next frame, the contour is tracked and the resulting warp

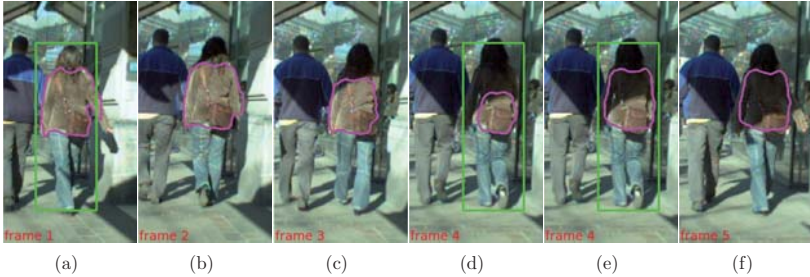


Figure 4.6.: Adaptation to lighting changes: (a-c) tracked shape becomes too small due to lighting changes (d,e) level-set re-initialization is triggered (f) tracking can continue.

is applied to the object frame and the associated detection box in order to obtain the new object position. Afterwards, the level-set shape is adapted for 5 iterations. We track only the person’s torso, since this body part deforms only very little, requiring fewer shape adaptation iterations than tracking the full body. This speeds up level-set tracking and increases the robustness, since it limits the amount of “bleeding” that can occur to similar-colored background pixels. To infer the person’s full extent, we maintain the transformation $V(\mathbf{x}, \mathbf{p}')$ from the warped reference frame to the original bounding box.

Multi-Region Handling and Overlap Detection. When tracking several persons, each of the tracked contours is represented by its own level-set. Even if there are overlaps, the level-sets will not interact directly (as, e.g., in (Brox and Weickert, 2004)). Instead, we use the stereo depth in order to resolve overlaps. All tracked persons are sorted according to their distance from the camera and the closest person is updated first. All pixels belonging to the resulting segmentation are masked out, such that they cannot be used by the remaining persons.

This leaves us with some persons that are only partially visible, which is in fact the same case as a person leaving the image frame. We developed a method for dealing with partial visibility without losing shape information. As can be seen in eq. (4.6), only a narrow band of pixels around the contour, which is determined by $\delta_\epsilon(\Phi)$, is taken into account for tracking. If pixels are masked out or are outside the image frame, we set δ_ϵ to zero for those pixels, which will result in tracking only the visible part of the contour. Thus, if an object becomes completely visible again, the shape will still fit. Objects are discarded if only a small part of the area inside the contour (50% for person-person occlusions, 20% for occlusions by image borders) remains visible.

Level-Set Re-initialization. Lighting changes or similar colors near the object can cause the contour to shrink during tracking (see Fig. 4.6) or to bleed out during shape adaptation. By periodically updating a tracklet bounding box with new detector bound-

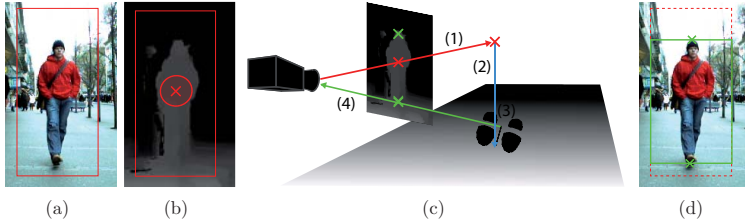


Figure 4.7.: Depth-based bounding box correction: (a) original bounding box (b) depth map (c) correction procedure (d) corrected bounding box (see text for details).

ing boxes, it is possible to identify degenerating shapes based on their size in relation to the bounding box. This is done by first performing the level-set tracking step for adapting the contour to the new image and then matching the tracked location to new detector boxes. If the box overlap (measured by intersection-over-union) is above 0.5, the detection box is used to update the relationship $V(\mathbf{x}, \mathbf{p}')$ between box and warp. The level-set contour itself is only updated if its area gets too small or too large with respect to the updated box, or if 20% of its content lie outside the box. Thus, the tracklet integrity is maintained and an ID change is avoided (cf. Fig. 4.6).

Consistency Checks. For robust operation, it is necessary to check the consistency of the tracking results. An object could be occluded, leave the image frame or be lost for other reasons. This may not even have any effect on the convergence of the LS-tracker, which might get stuck on some local image structure, resulting in a wrong track. We therefore perform the following checks in order to identify corrupted tracklets. (1) If the object is occluded and only background colors remain, the shape will typically shrink massively within a few frames. If such a case is detected, the tracklet is terminated. (2) We keep track of the median depth inside the tracked contour and react if this value changes too fast. We distinguish two cases here: If the median depth decreases too fast, this indicates an occlusion by another object; if the depth increases too fast, the object was probably lost. We terminate the tracklets in both cases. (3) Finally, objects whose median depth does not fit their ground-plane distance are also discarded. Typically, a failed consistency check indicates a tracking failure and will result in a request for the detector to be activated in the next frame. An exception are cases where an occlusion is “explained” by the high-level tracker (see below), or when the object is close to the image boundary and is about to leave the image.

Depth-based Bounding Box Correction. Level-set (re-)initialization and high-level 3D trajectory integration require accurately aligned bounding boxes. In general, the HOG detector however yields detections with a certain border area. Similarly, the boxes provided by the LS-tracker may drift due to articulations and shape changes of the

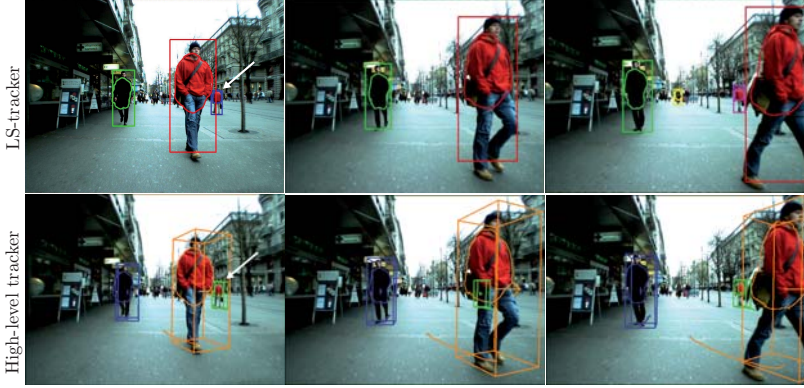


Figure 4.8.: Example for the occlusion handling process: (top row) contours tracked by the LS-tracker; (bottom row) output of the high-level tracker. When the distant person is temporarily occluded, its LS tracklet is terminated. As soon as the occlusion is over, a new tracklet is started. The high-level tracker connects both tracklets through the occlusion and maintains the person’s identity.

level-set contour and need to be corrected. We therefore apply the following correction procedure both to new detections and after each level-set tracking step. Starting from the original bounding box (Fig. 4.7(a)), we first compute the median depth around the bounding box center (Fig. 4.7(b)). We then determine the corresponding 3D point using the camera calibration from SfM and project it onto the ground plane (Fig. 4.7(c), steps(1)+(2)). We add a fixed offset in the viewing direction in order to determine the person’s central foot point, and finally project the resulting 3D point back to the image (Fig. 4.7(c), steps (3)+(4)). This determines the bottom line of the corrected bounding box. The top line is found by searching for the highest point inside the bounding box that is within 0.5m of the median depth (Fig. 4.7(d)). As a final step, we verify that the resulting bounding box aspect ratio is in the range $[\frac{1}{3}, \frac{2}{3}]$. Bounding boxes falling outside this range are rejected as likely false positives.

Requesting New Detections. New detections are requested in the following cases: (1) if a tracklet has not received an updated detection in the last k frames; (2) if a tracking failure cannot be explained by an occlusion or by the tracked person leaving the image; (3) if no request has been issued for k frames (e.g., since no object is currently tracked). A tracklet will not request new detections if it is close to the image boundary, as the chance for finding a detection there would be small. If a tracklet receives no updated detection despite its request, it will repeat the request, but will continue to be tracked as long as it passes the consistency and depth correction checks.



Figure 4.9.: Examples demonstrating our approach’s capability to continue tracking persons close to the camera and/or the image borders, where object detection is no longer applicable.

Integration with High-Level Tracker. The high-level tracker’s task is to integrate the tracklet bounding boxes into physically plausible 3D trajectories. This is done by first creating an *observation* at each tracked person’s 3D foot point and then associating this observation to trajectory hypotheses. The overall procedure is similar to the general *tracking-by-detection* framework described in Sec. 3.4. However, we make the following changes in order to account for the additional information provided by the LS-tracker.

Since we already know the tracklet identity of each observation from the LS-tracker, we can use this information in order to simplify data association. Thus, we first try to extend each existing trajectory hypothesis by searching for an observation matching the trajectory’s currently followed tracklet ID in a gating area around the Kalman Filter prediction. If such an observation can be found, it will directly be associated with the trajectory. Note that in this case, only the motion model is considered; the appearance is assumed to be correct due to the association performed by the LS-tracker. In case no observation with the correct tracklet ID can be found, we try to find the best-matching observation under the trajectory’s motion and appearance model (again within a gating area determined by the Kalman Filter uncertainty). If such a new observation can be found, the trajectory takes on the new tracklet ID, thus connecting the two tracklets. This latter case can occur if the LS-tracker diverges and fails the consistency checks (in which case the tracklet will be terminated), if the tracked bounding box is rejected by the depth correction (in which case the tracklet may persist for up to k frames and can be recovered), or if the tracked object is occluded or leaves the image.

In addition to the above, each observation is used to start a new trajectory hypothesis, which searches backwards in time in order to find a potentially better explanation for the observed data. This makes it possible to automatically create tracks for newly appearing persons and to correct earlier tracking errors. The final set of accepted tracks is then obtained by performing model selection, as described in Section 3.4.

Tracking through Occlusions. As motivated above, a main advantage of the image-based low-level tracker, compared to a pure *tracking-by-detection* approach, is that it simplifies data association, thus making it easier to integrate observed pedestrian loca-

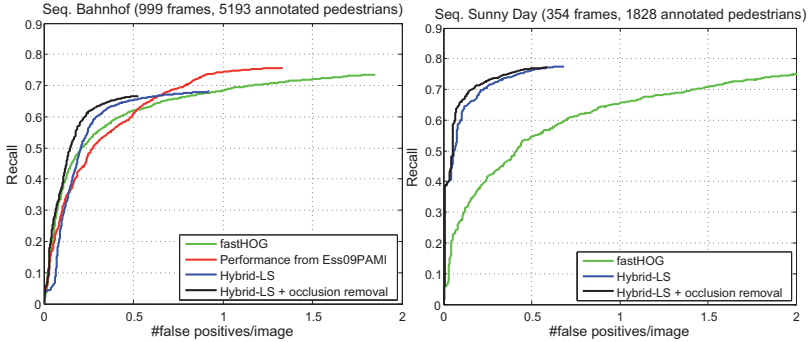


Figure 4.10.: Quantitative tracking performance of our approach compared to different baselines.

tions into valid tracks. The image-based tracklet generation will however fail when the tracked person gets occluded, which often occurs in practice. This is a limitation of any image-based tracking approach. While strategies can be devised to cope with short-term occlusions at the image-level, they would make this component unnecessarily complex. In our approach, we instead address this issue by explicit occlusion handling on the high-level tracker’s side. In order to bridge short-time occlusions, we keep potentially occluded trajectories alive for up to 15 frames and extrapolate their position on the ground plane using the Kalman Filter. Since the latter’s positional uncertainty grows with the absence of observations, the corresponding person can likely be associated to the predicted trajectory again when reappearing from the occlusion.

In addition, the high-level tracker can predict person-person occlusions and reinitialize the image-based tracker when those are over. For this, we backproject the predicted 3D bounding box of each tracked person into the image and compute the bounding box overlap using the intersection-over-union criterion. If the overlap is larger than 0.5, then an occlusion is likely to occur. This information is stored together with the occluded trajectory and is transmitted to the corresponding LS tracklet, which will typically be terminated 1-2 frames later when the consistency check fails. When the corresponding object is predicted to become visible again a few frames later, the object detector is fired in order to recover the person with as little delay as possible. This “safe termination” and subsequent new tracklet generation strategy proved to be robust in our experiments. It is similar in spirit to the *track-suspend-fail* strategy proposed in (Kaucic *et al.*, 2005), but our approach extends the idea through the integration of the robust multi-hypothesis tracking framework.

Fig. 4.8 shows an example where this occlusion handling process is used in practice. Cued by the occlusion prediction and the failed depth consistency check, the LS tracklet

	BAHNHOF	SUNNY DAY
# persons	67	28
mostly hit	54	23
part. tracked	4	1
mostly missed	9	4
false alarms	9	6
ID sw.	6	3
ID sw. (long occ.)	12	3

Table 4.1.: Track-level evaluation according to the criteria by (Wu and Nevatia, 2007).

is terminated in order to avoid degeneracies (which would be likely in this case due to the similar color distributions). On the high-level tracker’s side, the trajectory is however extrapolated through the occlusion. As soon as the occluded person becomes visible again, the object detector is fired again in order to initialize a new LS tracklet, which is correctly associated to the trajectory, maintaining the person’s identity.

4.4.3. Experimental Results

Datasets. We evaluated our approach on two challenging sequences BAHNHOF and SUNNY DAY from the Zurich Mobile Pedestrian corpus, presented in Sec. 3.6. Both sequences come with stereo depth maps extracted with an MRF-based approach from (Felzenszwalb and Huttenlocher, 2006), structure-from-motion localization from (Ess *et al.*, 2009b), and ground plane estimates. Similar to (Ess *et al.*, 2009b), we upscale all images to twice their original resolution in order to detect also pedestrians at larger distances. Using the upscaled images, *fastHOG* performed at 2-3fps (10fps for original images). In contrast to (Ess *et al.*, 2009b; Leibe *et al.*, 2008a), we however only use the left camera stream for detection and tracking, thus reducing the necessary processing effort. All system parameters were kept the same throughout both sequences.

Tracking Performance. Figure 4.9 shows qualitative results of our approach, demonstrating its capability to continue tracking persons that appear close to the camera or that are partially occluded by the image boundaries. This is a fundamental advantage our tracking framework can offer over pure *tracking-by-detection* approaches.

In order to assess our approach’s performance quantitatively, we adopt the evaluation criteria from (Ess *et al.*, 2009b) and measure the intersection-over-union of tracked person bounding boxes and annotations in every frame. We accept detections having an overlap greater than 0.5 as correct and report the results in terms of *recall* vs. *false positives per image* (fppi). Fig. 4.10 shows the resulting performance curves when we set the maximum re-initialization interval to $k = 5$ frames (in blue), together with the baseline of *fastHOG* (in green). As can be seen, our approach achieves good performance, reaching 65% and 76% recall at 0.5 fppi for BAHNHOF and SUNNY DAY, respectively.

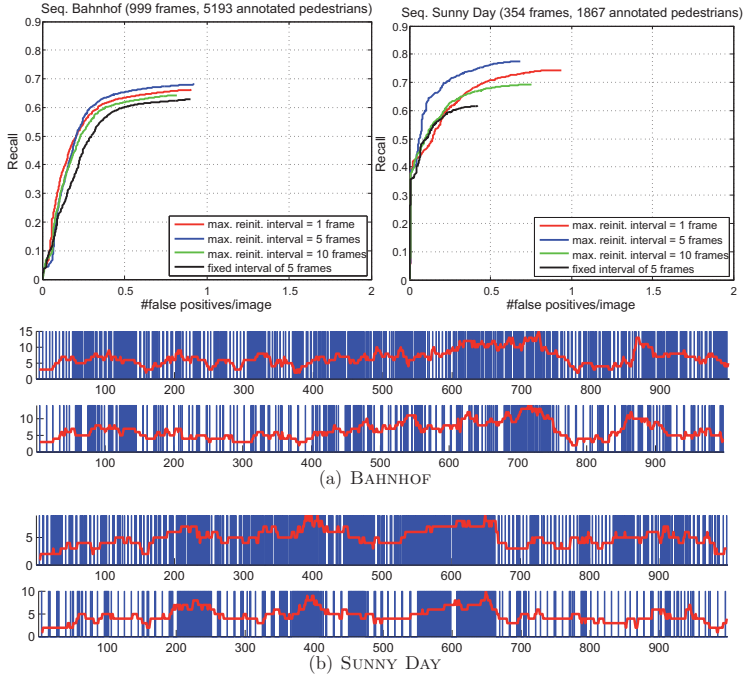


Figure 4.11.: (top) Tracking performance for the two test sequences when varying the maximum re-initialization interval; (bottom) Frequency of detector activations for both sequences for an interval of 5 (first) and 10 (second) frames. The red curve shows the number of tracked pedestrians.

It should be mentioned, however, that the bounding box evaluation criterion penalizes a tracker compared to a single-frame detector for two reasons: (1) The tracker typically needs several frames before it reports a track with sufficient confidence (losing recall) and (2) it predicts a person’s location through occlusions, even though those locations are in most cases not annotated in the test set (losing precision). In order not to penalize the tracker’s property of predicting a person’s location through occlusions, we additionally report the performance curve when filtering out tracked bounding boxes which are more than 50% occluded by other boxes (in black). Note that this is not a change in the approach itself, but just a change in the way the obtained results are reported, and it only compensates for an issue that caused the previously reported results to be unfairly penalized. The effect is a small but consistent reduction in the measured false-positive

rate in the high-precision region of the performance curve. We will therefore adopt this reporting scheme for all following bounding box-level evaluations.

For comparison, we also provide the performance curve reported by (Ess *et al.*, 2009b) on BAHNHOF, which is also based on HOG detections (shown in red, no such curve is available for SUNNY DAY). This approach integrates detections from both camera streams and thus obtains a higher recall. Its performance should be compared to our blue curve, since no occlusion removal was performed in (Ess *et al.*, 2009b). Still, it can be seen that our approach achieves better performance in the high-precision range, despite only using a single camera stream. This is a result of the better data association provided by the image-level tracklets.

Tab. 4.1 also reports a track-level evaluation according to the criteria by (Wu and Nevatia, 2007), showing that most pedestrians are correctly tracked and only few ID switches occur. Fig. 4.12 shows results of our combined tracker for both test sequences and visualizes the obtained level-set contours. Our system is able to track most of the visible pedestrians correctly in a very busy environment with many occlusions.

Efficiency Considerations. One of our goals was to reduce the dependence on the costly pedestrian detector. Even though efficient GPU implementations are now available for HOG (e.g. (Prisacariu and Reid, 2009; Sudowe and Leibe, 2011)), the excessive power consumption of GPUs is a major restriction for their use in mobile robotics applications. In contrast, the level-set tracking approach employed here can be very efficiently implemented on regular CPUs. (Bibby and Reid, 2008) report a framerate of 85Hz for tracking a single target of size 180×180 pixels in their implementation. In our application, we track targets at a lower resolution of 80×100 pixels and therefore expect even faster performance once our code is fully optimized.

An important consideration in this respect is how often the pedestrian detector needs to be activated for robust tracking performance. Our approach lets the LS-tracker request detections whenever required, but enforces a maximum re-initialization interval of k frames per track. A detector activation means that the detector is executed on the entire image. After activation requested from some existing track all other tracks are re-initialized as well. Fig. 4.11 shows the effective frequency of detector activations when setting this interval to $k \in \{1, 5, 10\}$, together with the resulting tracking performance. A setting of $k = 5$ provides the best tracking quality with a detector activation on average every 1.66 frames. By increasing the maximum interval to 10 frames, the detector activation rate falls to every 2.71 frames at a small loss in recall that is still comparable to (Ess *et al.*, 2009b) at 0.5 fppi. Considering that (Ess *et al.*, 2009b) performed detection in both camera streams, our approach thus requires 5.42 times less detector activations. Finally, we show the performance when activating the detector at a fixed interval of 5 frames, without additional requests. This results in a small drop in recall, but still yields good overall performance.



Figure 4.12.: Example tracking results of our LS-tracking approach on BAHNHOF and SUNNY DAY test sequences.

4.4.4. Summary

So far, we have presented a first instantiation of a hybrid tracking framework for mobile street-level multi-person tracking. Our approach combines the advantages of a fast segmentation-based LS-tracker for following individual persons with the robustness of a high-level multi-hypothesis tracking framework for performing longer-term data association. As our experiments have shown, the approach reaches state-of-the-art performance, while requiring fewer detector evaluations than conventional *tracking-by-detection* approaches. The obtained results open several interesting research perspectives. The requested detector activations for tracklet re-initialization could be restricted to the tracklet’s immediate neighborhood, thus resulting in further speedups. In addition, the obtained level-set segmentation could be a possible starting point for articulated tracking which can be explored in future research.

Still, the presented hybrid tracker realization based on level sets has a number of limitations. The LS tracking requires a precise initialization and will fail if the initial detection bounding box is not accurate enough, caused by selection of a wrong scale in the non-maximum procedure of the detector. Furthermore, as a local optimization approach, LS tracking can easily diverge in image areas with low color contrast or when lighting changes occur (cf. Fig. 4.6). This problem is not specific for LS segmentation but for any segmentation-based tracking approach based purely on color information, having difficulties in low-contrast street scenes. Another issue is that pedestrians are objects that undergo strong articulation while moving. Since LS segmentation cannot easily deal with strongly articulated shapes we applied the the LS trackers only on persons’ torsos; the legs were not captured. An extended LS approach taking into account shape priors to capture the entire body shape has been proposed by Horbert *et al.* (2011). However, this approach is also purely based on color information and will diverge in low-contrast scenes. Generally, we would like a tracking approach that scales well to a large variety of different object categories, with both rigid and articulated shapes, and that can perform an automatic, class-independent segmentation initialization.

In the following, we therefore present a second hybrid tracker instantiation that addresses some of those problems by building upon a depth based low-level tracker.

4.5. Hybrid Tracking with ICP

In this section, we will describe another instantiation of a hybrid tracker, which combines a depth based ICP low-level tracker with a high-level tracker. Similar to the LS-tracker, initiated by a detector response, the ICP-tracker follows individual pedestrians over time. However, the ICP-tracker relies just on the raw depth information in order to obtain the frame-by-frame displacement of the initial pedestrian position. We exploit the depth information not only for tracking but also in order to extract ROIs which reduce the search space for the object detector both spatially and temporally. The ROIs

are extracted as described in Chapter 5, by projecting the 3D points from a depth map to a 2D ground plane. The local maxima of the map are backprojected to the image building the ROIs which are then evaluated by a detector (spatial reduction of the search space). Instead of evaluating all ROIs by the detector in every frame, the ROIs are tracked over time using cheap, Kalman Filter based data association and only newly appearing ROIs are checked by the detector (temporal reduction of the search space). For each ROI with a positive detector response an ICP-tracker is started which registers the successive 3D point clouds from the corresponding ROIs, yielding the displacement of the point cloud from one frame to another. As the name indicates, the ICP-tracker relies on the ICP (Iterative Closest Point) algorithm (Besl and McKay, 1992; Chen and Medioni, 1991), which is a popular method for aligning two 3D models based on their geometry. Given two points clouds $\mathcal{M}_k = \{\mathbf{m}_i\}_{i=1}^{N_{\mathcal{M}}}$ and $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{N_{\mathcal{P}}}$ the goal of the method is to iteratively find the rotation and the translation which minimizes the alignment error between the point clouds:

$$\mathcal{E} = \sum_i \|\mathbf{p}_i^{NN} - \mathbf{R}\mathbf{m}_i - \mathbf{t}\|^2 \quad (4.8)$$

where the point $\mathbf{p}_i^{NN} \in \mathcal{P}$ is the closest point to $\mathbf{m}_i \in \mathcal{M}_k$ according to some distance function d :

$$\mathbf{p}_i^{NN} = \underset{\mathbf{p} \in \mathcal{P}}{\operatorname{argmin}} \{d(\mathbf{p}, \mathbf{m}_i)\} \quad (4.9)$$

In each iteration, first the closest points are computed and then the rotation and translation that minimizes Eq. 8.1 is applied to the points $\mathcal{M}_k$, resulting in a new point cloud $\mathcal{M}_{k+1}$. The iteration is repeated for the new point cloud $\mathcal{M}_{k+1}$. Finally, the rotation and the translation of each iteration step are accumulated, resulting in a final transformation. The individual steps of the method are summarized in pseudo code in Algorithm 1 which is based on the method presented by Arun *et al.* (1987) for estimating the translation and rotation components. This method is non-iterative and finds the least-squares solution of $\mathbf{R}$ and $\mathbf{t}$ for the case when the correspondences between the points in the point clouds are known. Since in our case we do not know the correspondences and rely on the assumption that the corresponding points are points with the smallest distance to each other, we iteratively apply the method from Arun *et al.* (1987) and recompute in each iteration the correspondences after moving the template points by the estimated relative rotation and translation. In the following we will only use a simplified version of the general ICP point cloud alignment described in Algorithm 1 where we drop the rotation estimation, however in Chapter 8 and 9 we will use the described elaborate version, also taking the rotation into account.

Considering our goal to track pedestrians, we can ignore the rotation estimation, since pedestrians are assumed to move upright and since the depth information is only

Algorithm 1 ICP (Iterative Closest Point) algorithm.

```

1: function ICP( $R, t$ )( $m, p, nrIter$ )
2: // ICP (Iterative Closest Point) algorithm is a popular method for registration of
   two point clouds based on their geometry. The method iteratively aligns the point
   clouds revising the rotation and the translation which minimizes the alignment error
   between those point clouds.
3:  $R = \text{eye}(3)$ ;  $t = [0 \ 0 \ 0]$ ;
4: for  $i \in [1:nrIter]$  do
5:   // establish correspondences and compute centroids
6:   for all points  $p_i$  in  $p$  do
7:     // transform point according to  $R, t$ 
8:      $query = R * p_i + t$ 
9:     // search nearest neighbor in  $m$ 
10:     $result = \text{nearest}(query, m)$ 
11:    // set model/template point add the point to centroid
12:     $pm_i = result$ ;  $\mu_{Model} += pm_i$ 
13:     $pt_i = query$ ;  $\mu_{Temp} += pt_i$ 
14:   end for
15:   // subtract mean
16:    $\mu_{Model} = \mu_{Model}/\text{size}(pm)$ 
17:    $\mu_{Temp} = \mu_{Temp}/\text{size}(pt)$ 
18:    $qm = pm - \text{ones}(\text{size}(pm), 1) * \mu_{Model}$ 
19:    $qt = pt - \text{ones}(\text{size}(pt), 1) * \mu_{Temp}$ 
20:   // compute relative rotation matrix  $R$  and translation vector  $t$ 
21:    $H = \text{trans}(qt) * qm$ 
22:    $[U \ S \ V] = \text{svd}(H)$ 
23:    $\text{interR} = V * \text{trans}(U)$ 
24:    $\text{interT} = \text{trans}(\mu_{Model}) - R * \text{trans}(\mu_{Temp})$ 
25:   // update  $R$  and  $t$ 
26:    $R = \text{interR} * R$ 
27:    $t = \text{interR} * t + \text{interT}$ 
28: end for
29: return  $R, t$ 
```

available from one viewpoint. This constrains the ICP algorithm to approximate only the translation between two point clouds as follows:

$$\mathcal{E} = \sum_i \|\mathbf{p}_i^{NN} - \mathbf{m}_i - \mathbf{t}\|^2 \quad (4.10)$$

In each iteration step the distance between the centroids of all corresponding (closest) points $\mathbf{p}_i^{NN} \in \mathcal{P}$ and $\mathbf{m}_i \in \mathcal{M}_k$ is used in order to update $\mathcal{M}_k$. This can also be derived

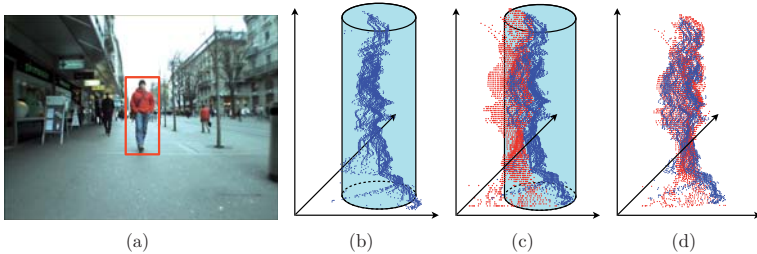


Figure 4.13.: (a) Detection from the color image. (b) Building up the initial model, which is represented as 3D points sampled from the cylinder placed on the ground plane position of the detection. (c) The model points (red) and the data points (blue) before the ICP alignment. (d) The result of the ICP algorithm showing a correct alignment of the data and model points.

from Algorithm 1 when replacing the rotation matrix by the identity matrix. We found that after 3 iterations the points $\mathcal{M}_k$ are already well aligned with the points $\mathcal{P}$, as illustrated in the example in Fig. 4.13.

The ICP-tracker consists of two steps, which are iteratively repeated.

1. For each detection we generate a 3D model. The 3D model is represented by the 3D points which are sampled from a cylinder placed on the 3D position of the detection (see Fig. 4.13b). The cylinder size roughly approximates a person size with a radius of $0.35m$ and a height of $2.0m$.
2. The model points are used for computing the new position of the person in the next frame by aligning them to the data points, applying the presented ICP algorithm. The new position is then backprojected to the image generating a new detection bounding box, which is passed to the high-level tracker. This new detection bounding box is treated as the initial detection bounding box in the next frame and we continue with step 1.

Based on the raw depth data the ICP-tracker cannot recognize occlusions or a person leaving the scene. Because of this it can easily get stuck on some background area. However, this divergence can be detected by the high-level tracker which associates the low-level tracker output, the 2D bounding boxes to the trajectories. In case of an occlusion, the ICP-tracker output will typically violate the motion or appearance model of the current trajectories and will not be associated to any of them. Thus, in each frame the ICP-trackers whose detection could not be assigned to any of the existing trajectories are terminated by the high-level tracker.

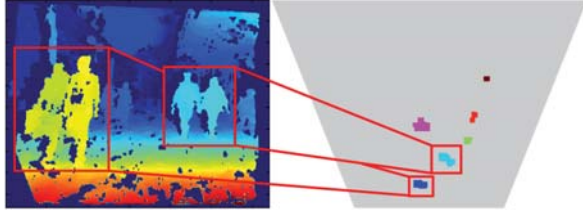


Figure 4.14.: Extraction of the ROIs (right) by projecting the 3D points from the depth map (left) to a ground plane.

4.5.1. Detailed Implementation

This section describes the realization of our framework’s individual components in more detail.

Depth based ROI Generation. For extracting the ROIs we rely on the *classical* ROI extraction procedure. This approach will be introduced in detail in Chapter 5.2. Briefly summarized, the 3D points of the scene are projected onto the ground plane to form a 2D histogram. The bins of the histogram are weighted by the distance to the camera and are thresholded in order to avoid noisy regions. The weighting is necessary since the objects that are further away consist of fewer points and would therefore be rejected due to the thresholding process. The final ROIs in 3D are the connected components on the grid map, as seen in Fig. 4.14. The position of each ROI is specified by the center of mass of the corresponding points. Additionally, we keep the width of the ROI and also the histogram modes. The modes are required for the low-level tracker in order to distinguish between two or more pedestrians walking close together, which will produce a single connected ROI.

For each ROI in 3D we set a rectangle at the center of mass of the ROI with the width of the ROI and a height of 2 meters positioned parallel to the camera. This rectangle is projected to the image in order to obtain the corresponding image region that is evaluated by the detector.

Object Detection. As a pedestrian detector we employ a GPU based HOG detector from Sudowe and Leibe (2011). With this implementation we can achieve the same detection performance as reported in the original paper Dalal and Triggs (2005) and it requires only 40ms for a 640×480 image (this detector run-time was at the time we ran the experiments. In the meantime the detector was improved reaching higher computational performance).

In contrast to pure *tracking-by-detection* approaches (Ess *et al.*, 2009b), we run the detector only on few small ROIs and only for some scales. If a new ROI shows up in a frame, the detector is run on this ROI, evaluating only 5 scales instead of 27 as in the original approach. We already know the rough scale by dividing the height of the

back-projected ROI by the detector window height. As the height of the pedestrian in the ROI is not known a priori, we consider two additional scales above and below the computed scale in order to assure the detection of the pedestrian.

The evaluation of an ROI by the detector requires on average only 2-3ms. In addition, our ROI-based system is also able to detect pedestrians that are smaller than 128 pixels (in a classical HOG-based approach the detection window is usually 64×128 pixels). Therefore, without upscaling the image the smallest pedestrian height that can be detected in the image is 128 pixels). In particular, for pedestrians which are far away from the camera the backprojected height h of the bounding box in the image will be smaller than 128 pixels, resulting in a scale of $s = \frac{128}{h}$. If the scale s is greater than 1, the ROI will be upscaled and the corresponding person can be found by sliding over the upscaled ROI with the standard HOG detection window. For achieving equivalent performance with the sliding window detector, the image would need to be upscaled by a factor of two to a size of 1280×960 . Processing such an image with our GPU detector would require more than 180ms, making it not applicable for real-time systems.

ROI Propagation. In contrast to existing ROI-based detection and tracking approaches (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010), we propagate the detector information for each ROI over time, in order to further reduce the detector execution and to find pedestrians entering the scene. In particular, in each frame, the newly extracted regions-of-interest are associated with the ROIs from the last frame. To this end, we apply a Kalman Filter with a constant-velocity model starting for each newly extracted ROI that is not associated to previous ROI tracks. The system uncertainty is propagated from frame to frame for each ROI and is employed for finding the new predicted position. Only for the newly extracted ROIs that could not be associated, we run the detector and if a detection is found a new ICP-tracker is started for this detection. With this process, we restrict the detector execution only to new ROIs which might emerge due to pedestrians entering the scene or objects becoming visible due to motion of the observer. As a consequence, in a semi-crowded scene the detector is usually kept idle for a significant number of frames (in our experiments with the BAHNHOF sequence, largest gap without detector execution adds up to 15 frames). In contrast to the LS-tracking approach we can now avoid the periodical detector executions every k^{th} -frame, which were originally needed in order to find new pedestrian entering the scene.

Occlusion Handling. When dealing with stereo range data, we have to deal with the well-known shadowing problem, which occurs for the parts of the image which are only visible from one camera. This problem is visualized in Fig. 4.15(b). Due to the lack of depth information in the area behind the person close to the camera, it is not possible to find a region of interest for the person marked with the yellow arrow. In the example in Fig. 4.15(b), it takes three further frames until the shadow disappears and the marked person is visible in the stereo range data. In order to be able to associate already existing tracks after an occlusion as fast as possible and to avoid losing the

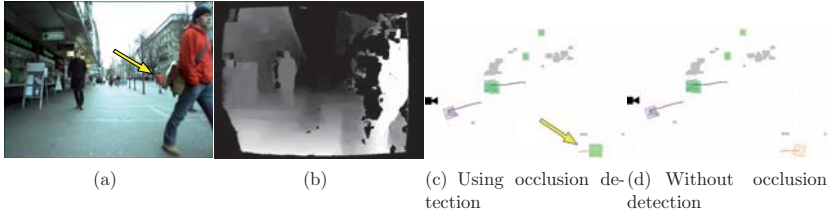


Figure 4.15.: Example showing the shadowing problem in stereo range data and our strategy to resolve it using explicit occlusion detection. (a,b): No regular ROI is generated for the person marked with a yellow arrow due to shadowing artifacts in the depth map. (c,d): The green areas are the ROIs which were verified by the detector in the current frame. Without occlusion detection, the person is missed. In contrast, the occlusion detection module prompts the detector to directly verify the area where the person reappears, although there is still no depth information available.

trajectory (after 15 frames without observation, the trajectory is removed), we propose to predict possible person-person occlusions and create a ROI when those are over. For that, we use a similar strategy as in the LS-tracker framework where we project the 3D prediction of the EKF of each tracked person into the image and compute the bounding box overlap using the intersection-over-union criterion. If the overlap is above 0.5, then the occlusion is likely to occur and we mark the person as occluded. In each further frame, we check for all occluded persons if they become visible again within 15 frames. If a person reappears, we create a ROI at this position and run the detector to evaluate this region and to revise the tracker, giving a new observation for the EKF.

In Fig. 4.15(c),(d) we show the ROIs that are evaluated if the person, marked with the yellow arrow, reappears after occlusion. As can be seen in Fig. 4.15(c), the result of occlusion detection directs the detector to evaluate the area where the person reappears, although there is no ROI extracted from depth. In contrast, Fig. 4.15(d) shows the result without using the described method, where the person remains undetected.

Reinitialization. Because of false negatives of the detector, some of the pedestrians could be missed if newly appearing ROIs are evaluated only once. As a consequence, some pedestrian trajectories could be partially or entirely missed. For robustness, we therefore propose to periodically re-trigger the object detector for ROIs which did not have a positive detector response since they appeared in the scene. In particular, during the ROI association process we also propagate the frame at which a ROI emerged into the scene and was consequently evaluated by the detector. For ROIs with negative initial detector response we execute the detector once again after five and ten frames. If the detector outputs a detection we continue with the ICP-tracker; if the detector response is negative we continue propagating the ROI, but do not run any further detector evaluations on it. When skipping this reinitialization step the tracking performance drops

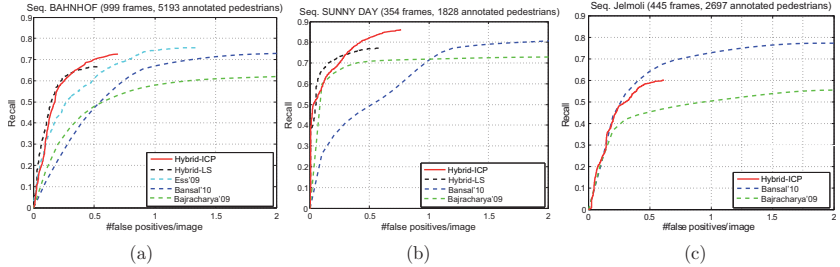


Figure 4.16.: Quantitative tracking performance of our approach compared to different baselines on the BAHNHOF, JELMOLI and SUNNY DAY sequences from (Ess *et al.*, 2009b). The results show that our approach can reach state-of-the-art performance. For all three test sets the same parameter sets were used.

significantly, e.g. the recall at 0.5 fpi decreases from 0.7 to 0.63 for the BAHNHOF sequence. This effect is not surprising, since many pedestrians in the BAHNHOF sequence initially appear at a far range from the camera. This means that the initial ROI patch in the image is significantly smaller than the HOG template window of 64×128 . In order to apply HOG detector for that image patch it needs to be upscaled, which blurs the edges of the shape of the pedestrian and consequently reduces the expressiveness of the gradients. Thus it is likely that the HOG detector response will be negative for the initial ROI. When now we start the ROI propagation and do not perform re-triggering, the pedestrian which generated the initial ROI will not be tracked over the entire lifetime.

Consistency Checks. In order to further increase robustness, we propose consistency checks similar to the LS-tracker which allow us to detect drift of the ICP-tracker. The consistency checks are performed by the high-level tracker while associating the resulting detections from the ICP-trackers. If a detection from the ICP-tracker is physically inconsistent in motion and does not fit the appearance with at least one of the existing trajectories, the high-level tracker terminates the particular ICP-tracker. A detection is inconsistent in motion if it is not inside of predicted uncertainty range of the Kalman Filter for any of the existing trajectory. It violates the appearance model if not at least for one existing trajectory the Bhattacharyya distance of the color histograms of the detection and the trajectory is above 0.85. For each trajectory which could not be extended due to deviation of the ICP-tracker, the high-level tracker generates a ROI at the current position of the trajectory. The generated ROI is treated as a newly emerged ROI, which is consequently evaluated by the detector and new ROIs in the next frame are tried to be associated with it. This process assures that if the ICP-tracker fails, the trajectory can still robustly be extended. Obviously, only those tracks are considered here that are not labeled as occluded.

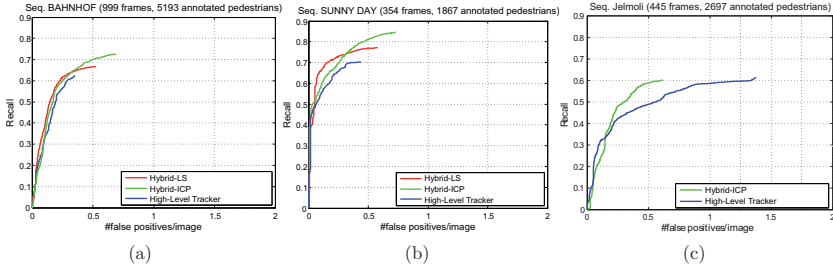


Figure 4.17.: Quantitative tracking performance comparison of the proposed low-level tracking approaches to our classical tracking-by-detection framework presented in Chapter 3.5.

4.5.2. Experimental Results

We experimentally evaluate our approach on three challenging video sequences from the Zurich Mobile Pedestrian Corpus (cf. Chapter 3.6): BAHNHOF, JELMOLI and SUNNY DAY. For the depth estimation we used the fast and robust algorithm presented by Geiger *et al.* (2010) described in Chapter 3.2.

Quantitative Performance. For assessing the performance of our tracking system, we applied the evaluation criteria from Ess *et al.* (2009b). To this end the tracked bounding boxes are compared to manually annotated ground truth bounding boxes in each frame. A bounding box is assumed to be correct if the intersection-over-union overlap with a ground truth bounding box is greater than 0.5. Fig. 4.16 presents the performance curves in terms of recall vs. false positives per image (fppi) for three sequences. As can be seen, our approach achieves state-of-the-art performance. For comparison, we also provide the curves reported by Ess *et al.* (2009b) (only BAHNHOF), our level-set based approach Mitzel *et al.* (2010) (only BAHNHOF and SUNNY DAY) and the approaches by (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010). Note that Bansal *et al.* (2010) did not use the original annotation files provided by Ess *et al.* (2009b) but created their own, leaving out some hard test cases.

In addition, we show in Fig. 4.17 a comparison of the tracking performance of the proposed hybrid tracking approaches to the classical *tracking-by-detection* method we presented in Chapter 3.5. On all tested sequences, the Hybrid-LS and Hybrid-ICP perform significantly better. In case of the Hybrid-ICP tracker this can be explained by the reduced number of false positives from the detector induced by the restriction of the search space of the detector to ROIs. The Hybrid-LS-tracker compensates significantly better for detector failures, since it can rely on the learned appearance model of the tracked objects. It can therefore bridge longer periods of detection failures and continue to robustly track objects based only on local appearance information. In comparison,

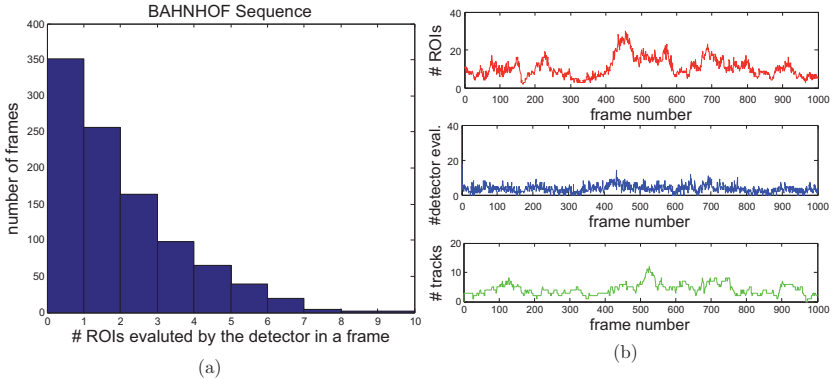


Figure 4.18.: (a) Histogram of the number of ROI evaluations by the detector per frame for the BAHNHOF sequence. (b) number of extracted ROIs in each frame (red), number of ROIs that were evaluated by the detector (blue), and number of final tracks estimated for each frame (green).

the classical *tracking-by-detection* approach extrapolates the position of the object using Kalman Filters without any observation information. The extrapolation, however, might deviate from the real position of the object, such that in case of re-detection association will not be possible anymore. This might be one of the reasons for a better tracking performance of the Hybrid-LS-tracker.

Computational Performance. Most of the computation time in standard *tracking-by-detection* approaches is required for the object detector, especially if more than one object class/view needs to be detected. In order to evaluate the effect of our proposed ICP-tracker on reducing this computation, we performed the following timing experiments (using a machine with Intel Core2 Quad CPU Q9550 @ 2.83GHz processor, 8GB RAM, and an NVidia GeForce GTX 280 graphics card).

Running the ICP-tracker with all available model and data points is very time consuming and not really necessary, as shown in Tab. 4.2. Here, we illustrate the effect of using only a fixed number of randomly sampled points from the model and data for the ICP-tracker. With only 100 points, we can already reach state-of-the-art tracking accuracy (recall of 0.7 @ 0.5 fpi on the BAHNHOF sequence) with a speed of 10.86 fps.

As a baseline, we compare the run-time of our approach to a pure *tracking-by-detection* system. To this end, we run the GPU-accelerated object detector by Sudowe and Leibe (2011) (detector run-time was 40ms for a 640×480 image at the time those experiments were made) over the entire image in each frame and use the high-level tracker for data association. The frame rate is significantly higher (14.59 fps) compared to the proposed hybrid tracker (10.86 fps), but the recall decreases considerably (from 0.70 to 0.60 @ 0.5

baseline	speed in fps	recall@0.5 fppi
pure t-by-d 640×480	14.59	0.60
pure t-by-d 1280×960	4.62	0.69
no ICP, eval. all ROIs	6.54	0.70
ICP, number of sampled points	speed in fps	recall@0.5 fppi
ICP, 20 points	11.62	0.67
ICP, 50 points	11.36	0.69
ICP, 100 points	10.86	0.70
ICP, 500 points	7.23	0.69
ICP, all points	1.09	0.70

Table 4.2.: Comparison of our proposed ICP-tracker results to several baselines on the BAHNHOF sequence. We report the effect of the number of (randomly sampled) ICP points on the final tracker run-time and accuracy.

fppi), since only pedestrians larger than 128 pixels (the height of the sliding window of the HOG detector) could be detected. When upscaling the image to twice its original resolution, the missing pedestrians can be detected (recall of 0.70 @ 0.5 fppi); however, the frame rate drops considerably to 4.6 fps.

For comparison, we also report an experiment where we evaluate the effect of the ROI propagation scheme. Instead of running the ICP-tracker starting from an initial detection, the detector is run in each frame for all ROIs. As expected, we reach high recall in this test (0.70 @ 0.5 fppi), but the frame rate drops significantly to 6.54 fps, since we redundantly run the detector for each ROI in each frame instead of propagating the information whether the ROI contains a person or not. In Fig.4.18 we illustrate the triggering rate of the detector for the BAHNHOF sequence. Note that in two third of the frames, the detector only runs for at most two small ROIs. On average, the detector is executed for 2.38 ROIs per frame.

Fig.4.18(b,c) presents the relation between the number of detector evaluations, number of all ROIs and number of valid tracks. As expected, in the part of the scene with many tracks also the number of detector evaluations increases. Due to occlusion and clutter in crowded parts of the scene, the ICP-tracker diverges, causing the consistency check to fail. Thus, the high-level tracker generates additional ROIs that are evaluated in order to achieve robust tracking performance and not to lose tracks.

Qualitative Evaluation. Finally, Fig.4.19 shows some qualitative results achieved on the BAHNHOF and SUNNY DAY sequences. It can be seen that our system is able to track most of the visible persons correctly keeping correct person identities. In addition, to the tracking bounding box, we visualize the depth information that was used for computing the ROIs (in green inside the box). The ROI foot points are plotted as red points on the ground plane.



Figure 4.19.: Example results on the test sequences SUNNY DAY and BAHNHOF. The green overlay inside the box corresponds to the depth information used for computing the ROIs. Red points on the ground plane represent the ROI foot points.

4.5.3. Summary

We have presented a second instantiation of our proposed hybrid tracking framework for mobile multi-person tracking based on an ICP-tracker. Our approach reduces the use of a computationally expensive detector significantly by a combination of ROI propagation, low-level ICP tracking and a high-level tracker. As our experimental evaluations show, we can reach state-of-the-art tracking performance with this combination at a run-time that is suitable for real-time applications. For the application of this Hybrid-ICP tracker to more complex tracking scenarios also including other object classes such as cars, where several different viewpoint detectors need to be evaluated for each ROI, we expect the run-time benefit to be even larger.

Still, the presented hybrid tracker realization based on ICP has a number of limitations. For historical reasons we use in the Hybrid-ICP tracker a quite simple ROI generation procedure. This outputs blobs comprising usually of several pedestrians together. However, in Chapter 5 we propose a ROI segmentation approach that separates the ROIs in individual objects. Using the separated points clouds will improve the ICP registration since only points from the corresponding person will be selected for the registration from the next frame. Furthermore, the ICP alignment itself is a very simple algorithm, taking into account only translation. But an improved version of this step will be presented in Chapter 8, where we also take the rotation into account reaching a more robust localization and orientation of the tracked person.

The entire tracking approach depends on stereo depth to be computed before the actual tracking pipeline can be applied. This introduces failure modes of the ICP-trackers when stereo depth cannot be estimated reliably enough. In addition, stereo computation is still quite expensive (implementation from Geiger *et al.* (2010) requires 100ms for a 640×480 image on a desktop CPU - Intel Core2 Quad CPU Q9550 @ 2.83GHz processor) such that the entire tracking system can be at most as fast as the stereo computation component. However, for indoor application instead of using stereo setup an interesting alternative would be the RGB-D data from Microsoft Kinect/Asus Xtion sensors, where depth data comes at no additional cost.

4.6. Discussion

The experiments on both frameworks, Hybrid-LS and Hybrid-ICP trackers, corroborate our claim that a combination of a cheap low-level tracker with a high-level tracker is reasonable and allows reliable and fast tracking. Since it requires significantly less detector executions than the standard *tracking-by-detection* pipelines, the hybrid-tracking approach also becomes attractive for mobile applications.

So far, both hybrid tracking instances were developed independently considering pure image-based or depth-based low-level trackers. The next step would be to combine advantages of both trackers in one framework. As described before, the Hybrid-LS

requires periodical detector applications every k frames in order to detect pedestrians entering the scene. Using the ROI tracking idea in the Hybrid-LS tracker, we could, however, avoid this reinitialization step, since the newly appearing pedestrians could be represented as ROIs that could not be associated to existing ROI tracks. Thus, one promising combination would be to include the ROI extraction and tracking process into the LS-tracking framework. With this integration, we could avoid periodical detector applications and also constrain the detector evaluations to small ROIs.

Failures of consistency checks in both approaches cause additional detector evaluations. With this concept it is, however, not simply predictable how much computational resources are required during the entire tracking process, since the effort is strongly coupled to the scenario, e.g., the crowdedness of a scene. This is usually not an issue if only the tracking framework is run on a single laptop on a mobile setup, but it can become a problem if several components (e.g., visual odometry, stereo estimation and tracker) need to share a single computational device. We will address this problem using an ROI-based tracker in Chapter 7 with a time-constrained detector. With this framework we can reach a similar tracking performance as with both low-level based tracking frameworks, but due to the fixed and predictable requirements of the computational resources it could directly be employed on a mobile platform.

As demonstrated in this chapter, stereo based region-of-interest extraction makes it possible to reduce the computational effort of an object detector significantly. However, the approach used here still has some limitations, that need to be addressed for more robust performance. ROIs should be represented in a more general way, one that allows to detect splitting and merging during the tracking process. Such splitting or merging of an ROI can be caused due to slight distance changes between two objects in the scene. Tracking the simple representation we used in this chapter, where each ROI was represented by a center of mass of relevant points, fails in case of merging or splitting, since the center of mass and region dimensions abruptly. Such abrupt changes cannot be compensated by the Kalman Filter and therefore cause ROI tracks to be lost, which in turn causes redundant evaluation by the detector of the unassociated ROIs. Furthermore, the approach in the original form is sensitive to drift caused by misassociation during the ICP registration step. As described before, in order to obtain the new position of the tracked object we employ ICP, registering the actual points of so far tracked objects to new points in the new frame. These new points are sampled from the closest mode of the ground plane projection. In a deserted or a semi-urban scene with few pedestrians walking separately and not in groups, this is a valid assumption and will lead to correct data association. However, in urban and crowded environments with many pedestrians moving in groups, such a simple data association procedure often fails and causes drift when the closest mode corresponds to another close object in the group. This misassociation will surely be detected with the proposed consistency check performed by the high-level tracker, but will still cause another detector evaluation. In the following chapters, we propose extensions in order to address this issue. The detection of ROI merging or splitting becomes unnecessary when the ROIs are split in individual objects at the

outset. Then the association of the ROIs becomes less ambiguous. For this reason, we will propose an ROI segmentation procedure in Chapter 5 that separates the ROI into individual object regions based on mode search.

II

ROI based Object Detection and Tracking

Motivation. The ability to reliably detect and track pedestrians in real-world images is required for a variety of automotive and robotics applications. Recently proposed *tracking-by-detection* approaches (Andriluka *et al.*, 2008; Ess *et al.*, 2009b; Gavrila and Munder, 2007; Leibe *et al.*, 2008a; Wu and Nevatia, 2007) yield remarkable performance for tracking, but all rely on a detector’s output as the name *tracking-by-detection* already indicates. The state-of-the-art detectors (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b) however, are computationally very expensive and are consequently the bottleneck of tracking due to the feature design of HOG, which cannot be approximated in the multi-scale approach and which therefore needs to be extracted for each scale of the image independently. With our hybrid-tracking approaches presented in the first part of the thesis, we have shown that by relying on cheap low-level trackers we can avoid the execution of the detector in each frame, saving computational resources. Depth information allowed us to further reduce the computational effort of the detector by fixing the attention of the detector only to promising ROIs likely to contain target objects. However, there are still two problems which need to be addressed in order to allow the application of our pedestrian tracking approaches in a mobile setup. Due to the complexity of a typical urban scene where multiple objects can appear or emerge from occlusions, low-level trackers can easily diverge which will trigger an additional detector execution. For this reason it is not possible to specify and fix computational resources required for the detector since it is strongly dependent on the scene situation, which makes it difficult to use such a system on a mobile platform, where the components need to share resources. With our ROI based tracking approach we could reduce the computational effort of the detector by reducing the search space, but we still rely on the computationally expensive HOG-based pedestrian detector. With this combination of ROI and a GPU-based HOG implementation by (Sudowe and Leibe, 2011) we are able to reach real-time performance for the entire system. However, the power consumption of a GPU is enormous, hindering the application on a mobile vehicle with limited battery power capacity. For this reason, the next part of the thesis will investigate how far we can go in the direction of *mobile* tracking. Here, we will rely on depth information, because we have already shown that with our depth based hybrid tracking approach, we can reduce the computational time of the detector significantly by extracting promising ROIs. This strategy has been already successfully applied for several other mobile tracking approaches e.g., (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007).

In the next chapter, *Robust ROI Extraction and Segmentation*, we will introduce the standard ROI extraction procedure that was already proposed by (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007) and show how it can be augmented by a segmentation process that divides the ROIs, usually enfolding several objects, into individual objects. In the Chapter 6, we will introduce our depth-template based upper-body detector that relies on robust ROI extraction to yield good detection performance. The detector is optimized for a typical scene of interest in a crowded urban scenario and requires only the depth information as input. With the proposed method we can

solve the two aforementioned problems: we can detect pedestrians that are partially occluded by the image borders due to small distances to the camera, and we can get rid of the expensive GPU-based object detector. The proposed detector runs with more than 40fps on a single CPU, allowing a direct application on a mobile vehicle. In the Chapter 7, we present a method that makes it possible to restrict the detector to a fixed time budget in the context of the number of ROI verifications per frame. The task we tried to solve is, assuming a system where the object detector has a limited time budget to attend only to a small and fixed number of ROIs per frame, how should the selection of ROIs that are attended by the detector be organized to achieve the major goal of stable and robust tracking performance. The resulting proposed approach combines a cheap low-level tracking system that follows the ROIs over time with a Poisson Process-based selection model that models the urgency with which the ROI should be attended to. By fixing the detector evaluation, we are able to approximate the required computational resources and thus enable the application of our tracking system on a mobile platform with a shared computational unit.

Robust ROI Extraction and Segmentation

As mentioned before, the state-of-the-art object detection approaches such as, e.g., (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b) yield highly accurate human detection results for fully observed pedestrians. However, these approaches are quite expensive to evaluate, which limits their deployment for use on autonomous platforms. To deal with this problem, several approaches have been proposed to restrict the evaluation of the classifier to only few ROIs that are extracted from the stereo range data (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007). In our approaches (Mitzel and Leibe, 2011, 2012b), presented in Chapter 8 and Chapter 4, we follow a similar strategy of ROI extraction based on stereo information, in order to reduce the search space for the detector. This results in less computational effort and also a lower number of possible false positives since only image regions are evaluated that are likely to contain target objects.

Consequently, in this chapter, we will focus on robust region extraction in busy street scenarios. Using input from a stereo camera setup we want to jointly estimate the ground surface and regions in the image containing objects of interest. The ROIs are represented by connected bins from a 2D histogram, collected from 3D points projected to the ground plane. In contrast to general ROI extraction, we perform some preprocessing in order to exclude 3D points that come from walls and other elevated structures by removing regions that continuously extend beyond a certain height (so-called **fixed** objects). This step is necessary because the ROIs will form the input for a *tracking-before-detection* framework, presented in Chapter 8, that tracks all ROI segments. Usually, objects with a small distance on the ground plane are aggregated to a single cohesive ROI. Since we are interested in tracking individual objects, we segment ground plane projections into individual object segments using a mode search algorithm. As a byproduct of the ROI estimation, which extracts all points above the ground plane, we obtain a rough segmentation of the points on the ground plane itself. We use those remaining points to obtain a more accurate ground plane estimate by fitting a plane to them using RANSAC (Fischler and Bolles, 1981). We repeat this process in every video frame for updating

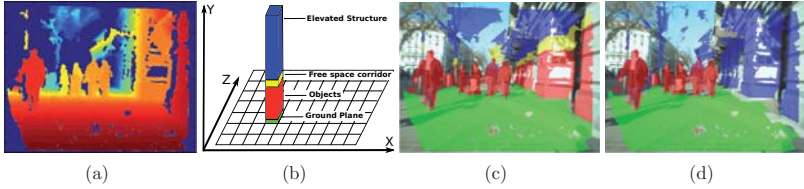


Figure 5.1.: Visualization of our elevated structures classification. (a) Depth map. (b) Height histogram. (c) Labeling of the 3D points according to the different height bands (**green** - ground plane band, **red** - object band, **yellow** - free space band, **blue** - fixed object band). (d) Final classification result of points into three classes.

the ground plane, in order to compensate for camera tilts caused by surface irregularities or head movements.

The main contributions found in this chapter are: (1) Dense, depth-based classification of 3D points in different classes **objects**, **ground plane**, **fixed objects**. (2) Segmentation of ROIs which usually contain several objects, into individual *sub*-ROIs. (3) A method for ROI extraction that accumulates the point density distribution over several layers allowing a better and more precise segmentation of occluded objects.

The chapter is structured as follows. Sec. 5.1 describes the pipeline for the classification of the 3D points into the three categories: **object**, **ground plane** and **fixed** structures (buildings, trees, etc.) by analyzing the height distribution. In Sec. 5.2, we present the classical pipeline for ROI extraction and an extension which makes it more robust for busy scenarios. Then, in Sec. 5.3 we show how the ROIs can be segmented into subregions representing individual objects. Following that, Sec. 5.4 presents some qualitative and quantitative results, before the chapter is concluded in Sec. 5.5.

5.1. Point Cloud Labeling

Several approaches in robotics have been proposed for static obstacle detection (Badino *et al.*, 2007; Ess *et al.*, 2009a), where point clouds from 3D sensors are fused over a time window and the resulting 3D points are then projected to the ground plane forming an occupancy map. From the occupancy map, the free space where the robot can move, can be computed using dynamic programming. We apply a similar approach for fusing the point clouds in a reference frame using SfM. These fused 3D points are then segmented into three different classes: **objects**, **ground plane** and **fixed** objects. Given this segmentation, we can exclude the 3D points of the fixed objects before ROI extraction, which will reduce the number of typical false pedestrian detections in the reflections of shopping windows on the one side, and avoid failures caused by tracking irrelevant scene structure on the other side.

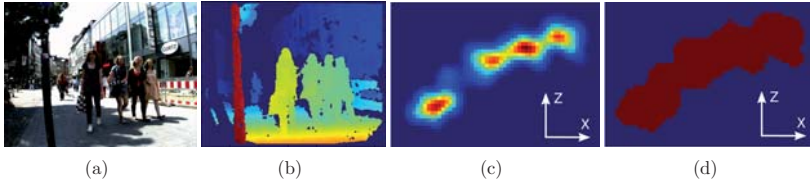


Figure 5.2.: Visualization of our stereo ROI extraction procedure. (a) Original image. (b) Depth map. (c) Projection onto ground plane of the four women in front. (d) The four women are merged into a single connected component in the *classical* ROI extraction process.

Before the classification step, we first need to obtain an accurate ground plane estimate. To this end, we compute the occupancy map from the fused point clouds, where we project all 3D points within a 2m height corridor to a rough estimate of the location of the ground plane based on the camera height of the recording vehicle. The 3D points inside bins with a high density are excluded and the majority of the remaining points corresponds to points on the real ground plane, which we estimate by fitting a plane to them using RANSAC (Fischler and Bolles, 1981). However, correctness of the resulting ground plane estimation strongly depends on an unobstructed visible surface of the ground, which is unfortunately often occluded by people walking close to the camera. One solution for this problem could be to treat the ground plane estimations in each frame as noisy measurements and using a Kalman Filter to smooth out the outliers.

The classification pipeline is visualized in Fig. 5.1. The incoming point clouds are fused over a time period of 5-10 frames and the accumulated 3D points are projected to a 2D ground plane, forming a 2D histogram. For each bin of the 2D histogram, we compute a height histogram from the corresponding points. The height histogram has four height bands: the **ground plane** band, **object** band, the **free corridor** band and the **elevated structures** band, Fig. 5.1(b). The free space corridor has a significant effect on classification performance. The simplest way to classify elevated structures would be to assign the points in bins with a high point density in the **elevated structures** band to the label **fixed** object. However, in the shopping street scenarios this will often cause mislabeling of the objects due to overhanging building parts. With a free space corridor we make the assumption that there is always a free space (2m-2.3m) between the heads of the pedestrians and the overhangs from buildings or trees. This means that, if for a bin in the 2D histogram the point density in the free space corridor is low (number of fused frames), the points in the object band are labeled as objects and the points inside the free space bin and the elevated bin are labeled as **fixed** objects. Even though we call the third band **free space** band suggesting that this band should be free of points we make still the assumption that due to noisy depth and structure-from-motion input some points fall inside. For bins with a high density (number of points higher than the

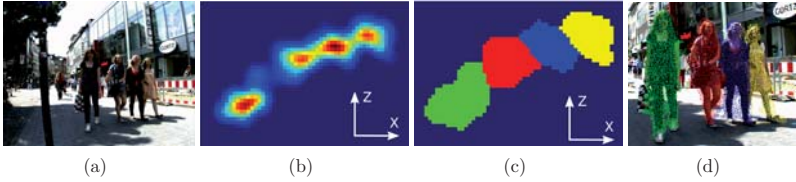


Figure 5.3.: Visualization of our stereo ROI segmentation procedure. (a) Original image. (b) Histogram of the ground plane projection. (c) Segmentation of the ROI into individual object clusters using the Quick Shift algorithm (Vedaldi and Soatto, 2008). (d) Corresponding point cloud segmentation.

number of fused frames) in the free space corridor band, all points are labeled as belonging to the class **fixed** object. This results in a final point classification as shown in Fig. 5.1(d). As can be seen, we obtain an accurate labeling for all three classes. The 3D points labeled as **fixed** object are excluded from further processing. Bansal *et al.* (2010) propose a similar labeling approach in order to restrict the detector’s search space to regions of the image labeled as objects. Instead of dividing the height histogram in four bins they use an uniformly histogram with k bins. Each pixel is then associated with the corresponding height histogram. Then the structural labels of each pixel is represented as the likelihood of the associated uniform height histogram. The likelihood densities are determined by kernel density estimation on the feature space extracted from a small annotated data set. Major advantage of our approach is that it can robustly cope with the problem of overhanging structures as discussed above. With the approach proposed by Bansal *et al.* (2010) for each pixel a fixed label is assigned based on the extracted height histogram. Since in case of overhanging structure the height histogram contains not only points of the **object** class, but also the **building** class, all pixels that fall inside this height histogram will still be labeled with one and the same label, either building or person. Therefore, part of the points will be always mis-labeled. Another advantage of our approach is that it is more general by simply exploiting the **free space** assumption, whereas the approach presented by Bansal *et al.* (2010) needs to be retrained in case of usage of different scene structure, requiring another annotated set of images.

5.2. ROI Extraction

Following the approaches from (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010) we project all 3D points labeled as **objects** onto the ground plane. We collect the points in a 2D histogram and weight them according to a quadratic distance to the camera in order to compensate for the fact that objects that are further away consist of fewer points and would therefore be rejected in the further processing. The histogram bins (Fig. 5.2(c))

are first smoothed with a Gaussian Filter ($\sigma = 2.0$ in x-direction and $\sigma = 3.0$ in z-direction) and are then thresholded in order to remove irrelevant areas caused by noise. The remaining bins after thresholding are grouped into connected components using an 8-neighborhood. The resulting connected components are shown in Fig. 5.2(d). For each ROI in 3D we set a rectangle at the center of mass of the ROI with the width of the ROI and a height of the highest point within the corresponding bins, oriented parallel to the camera. By projecting this rectangle to the image, we obtain the corresponding image region that can then be evaluated by the detector. We applied this ROI extraction mechanism successfully in our following approaches (Mitzel and Leibe, 2011, 2012a; Mitzel *et al.*, 2011a) described in Chapters 4, 6, 7, reducing not only the computational effort for the detector through the restricted search space, but also lowering the number of false positives. In the following, we will denote the ROI extraction approach described in this section as *classical* ROI extraction. Note that the *classical* ROI extraction does not include the ROI labeling described in the previous chapter.

5.3. ROI Segmentation

As can be seen in Fig. 5.2(d), people walking close to each other are often fused into one connected component in the ground projection. In order to track them individually we can either apply a pedestrian detector on the corresponding image region in order to obtain accurate positions for each object (as done in our approach (Mitzel and Leibe, 2011), presented in Chapter 4) or segment the connected components further into distinctive regions using a smoothed version of the original histogram (cf. Fig. 5.2(c)). For segmentation, we employ the Quick Shift algorithm (Vedaldi and Soatto, 2008), which is a fast variant of Mean-Shift (Fukunaga and Hostetler, 1975). Quick Shift finds the modes of a density $P(x)$ by shifting each point x_i to the nearest neighboring point y_i with a higher density value. It is formally defined as follows:

$$y_i = \underset{j: P_j > P_i}{\operatorname{argmin}} d(x_i, x_j), \quad P_i = \frac{1}{N} \sum_{j=1}^N \theta(d(x_i, x_j)) \quad (5.1)$$

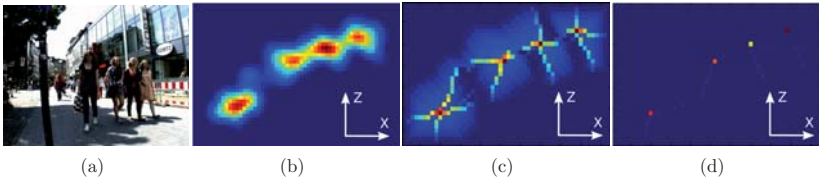
where P_i is the density estimate for point x_i with kernel function θ , and $d(x_i, x_j)$ is the distance between points x_i and x_j . In our case, we operate on a regular grid and use a fixed window (the 8-neighborhood around a pixel) as the kernel, which lets each pixel simply move to the neighbor with the highest value. We start Quick Shift for each point in the smoothed ground projection histogram and move it to its respective neighbor with the highest density until a mode is reached. The points on the way to the mode are automatically associated to this mode. As a result, we obtain a segmentation of the ROIs into individual objects, as shown in Fig. 5.3(c). Fig. 5.3(d) shows the corresponding segmented point cloud. An optimized version of the algorithm is summarized in pseudo code in Algorithm 2.

Algorithm 2 ROI Segmentation using Quick Shift (Vedaldi and Soatto, 2008).

```

1: function ROISEGMENTATION(matrix SegmentedRois)(matrix hist, int w, int h)
2: At first we compute a so-called shift matrix for the input histogram matrix where we store the shift
   in the direction of the maximum for each position.
3: matrix shiftMatrix(w,h,0);
4: for (int i = 0; i < w; i++) do
5:   for (int j = 0; j < h; j++) do
6:     maxOfMatrix(i,j) = index of highest element in 8-neighborhood of hist(i,j);
7:   end for
8: end for
9: for (int i = 0; i < w * h; i++) do
10:  modeLabel = shiftMatrix(i);
11:  Find mode by traversing the path until a maximum is reached
12:  while shiftMatrix(i)  $\neq$  shiftMatrix(shiftMatrix(i)) do
13:    shiftMatrix(i) = shiftMatrix(shiftMatrix(i));
14:    modeLabel = shiftMatrix(shiftMatrix(i));
15:  end while
16:  Assign to each bin on the path to the mode the label from the mode
17:  nextPos = shiftMatrix(shiftMatrix(i));
18:  while (shiftMatrix(nextPos)  $\neq$  modeLabel) do
19:    nextPos = shiftMatrix(shiftMatrix(i));
20:    shiftMatrix(shiftMatrix(i)) = modeLabel;
21:  end while
22: end for
23: return shiftMatrix;
24: The images below visualize the cost when using a simple brute force approach where we simply
   follow each path of the shift matrix independently and do not store the bins we have already
   attended (c). As can be seen in (c) several paths with a high attendance rate emerge. Instead of
   traversing this path several times by simply storing the label of the mode for each bin on the path
   to the mode, we enforce that these bins are attended only once. This results in significantly less
   effort, as shown in (d).

```



There is a major issue with the *classical* ROI extraction procedure as described in Sec. 5.2, which we address in our approach (Baumgartner *et al.*, 2013) presented in Chapter 9. The *classical* ROI extraction approach ignores the fact that target objects we are interested in for tracking need to be connected to the ground plane. As shown in Fig. 5.4(a), only the torso of the woman pushing the stroller is visible, which means that only these points will contribute to the ground projection histogram bins, resulting in

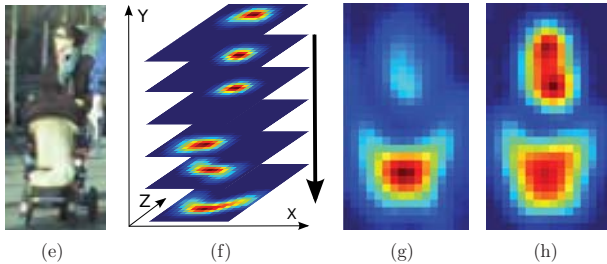


Figure 5.4.: Visualization of the proposed *accumulated* ROI extraction procedure: (a) image cutout, woman with a stroller, (b) visualization of height layers, (c) ground projection using the approach from (Bajracharya *et al.*, 2009a), (d) ground projection using our approach.

a very low bin value, as shown in Fig. 5.4(c), which will be rejected in the thresholding process. Instead, we propose a new procedure that splits the projection process over different height levels, as shown in Fig. 5.4(b). Starting with the highest level, we project all points above onto it. In the next steps, the points between two layers are projected to the lower layer and for each bin that is empty but was occupied in the layer above, we propagate the value from this upper layer. This process can be formally defined as follows:

$$\text{bin}(x, z, l - 1) = \begin{cases} \text{bin}(x, z, l - 1) & \text{if } \text{bin}(x, z, l - 1) > 0 \\ \text{bin}(x, z, l) & \text{if } \text{bin}(x, z, l - 1) == 0 \end{cases} , \quad (5.2)$$

where (x, z) correspond to the bin index x , and z in the histogram grid and l to the layer number. With this process we propagate the histogram value for frontally occluded parts up to the ground plane. Then, by summing over all layers, we obtain two distinctive modes for both objects, as shown in Fig. 5.4(d) and hence compensate for frontal occlusions. In the following we will refer to the accumulated ROI extraction as *accumulated* ROI extraction and for all approaches presented in the following section we will explicitly mention if the point cloud was pre-filtered first using point cloud labeling presented in Chapter 5.1.

5.4. Experimental Results

To evaluate our segmentation approach, we report the performance on our training dataset, described in more detail in Chapter 3.6. The dataset consists of 7,885 frames and was captured with the stereo camera Bumblebee2, in a controlled environment, from a static setup, in order to simplify the annotation process. For each frame we

	classical	accumulated
precision	0.9690	0.9849
recall	0.9174	0.9452

Table 5.1.: Precision-Recall for presented ROI segmentation approaches.

Figure 5.5.: Failure cases in segmentation using the *classical* ROI extraction approach. (a) Exemplary result of segmentation using the *classical* ROI generation approach, clearly showing over-segmentation. (b) the same as (a), but using the proposed *accumulated* ROI extraction process. (c) *Classical* ROI extraction approach, clearly showing under-segmentation. (d) the same as (c), but with *accumulated* ROI extraction, showing the correct segmentation of pedestrian and object.

manually annotated the number of individual target objects. Due to the smoothing of the 2D histogram after the projection of the 3D points, it is sufficient to just compare the number of segmented objects to the number of annotated ones, since in all observed cases if the number was correct, then the segmented objects coincided. In case the number of segmented objects is lower than the number of annotations, this corresponds to under-segmentation, as shown in 5.5(c). If the number is higher this means over-segmentation, as shown in 5.5(a). In Tab. 5.1 we present the precision and recall numbers of segmentation when using the *classical* ROI generation process vs. the proposed one, where we propagate the bins of occluded areas (*accumulated*). As can be seen, we already obtain good results with the *classical* approach, however, using the *accumulated* approach, we further increase precision and recall (by more than two percentage points) by correctly segmenting difficult cases as shown in Fig. 5.5. In Fig. 5.5(b) we show that using the *accumulated* approach, we resolve the problem of over-segmentation by closing the hole in the depth of the child-stroller (Fig. 5.5(a)) that caused two modes in the distribution. In Fig. 5.5(d) using the *accumulated* ROI extraction approach we obtain a correct number of objects in contrast to the *classical* ROI extraction approach (Fig. 5.5(c)). However, this example also nicely shows the issue of the segmentation using only depth information. As can be seen in Fig. 5.5(d) some parts of the child-stroller are labeled as pedestrian. This is caused by the limited number of disparities, which means that the objects that are further away from the camera are mapped to



Figure 5.6.: Exemplary segmentation results, showing that the proposed approach is able to correctly segment the blobs of objects into individuals.

the same disparity plane and thus we lose the volumetric information that we require for our segmentation approach. The disparity distribution is dependent on the baseline and resolution of the stereo camera system. Using the Bumblebee2 cameras described in Chapter 3.6 limits the operational distance range of our approach for up to 10 meters, which is sufficient for robotics applications, but may be problematic in car scenarios, where a larger distance range is required due to a higher speed of the vehicles. Fig. 5.6 shows some segmentation results, dividing blobs containing several persons correctly into individuals.

5.5. Discussion

In this chapter, we have presented a process for extracting regions likely to contain target objects. The ROI procedure forms an important basic component for several approaches we describe in Chapters 4, 8, 9. Given the stereo range data, we compute ROIs by collapsing all 3D points to a ground plane and extracting image regions having a high density of points. These regions-of-interest can be used to guide the detector to only a few promising image areas, reducing computational effort and the number of false positives, as we show in Chapter 4. We proposed several extension steps in order to make the ROI extraction procedure more robust and applicable for our scenarios. At first, the raw stereo data is preprocessed by classifying each 3D point as either belonging to a ground plane, an object or a **fixed** object (elevated objects as buildings, trees, lanterns). Obviously, the resulting ROIs from points labeled as **objects** are more likely to contain target objects and by excluding points from elevated objects we avoid typical false positives caused by reflections on building facades. Secondly, we propose a segmentation procedure in order to divide the ROI blobs that usually consist of several objects into individual objects using the Quick Shift algorithm. This step will allow us to develop a *tracking-before-detection* pipeline which will be described in Chapter 8. Based on the assumption that all target objects must be connected to a ground plane we proposed a better ROI extraction processes that reduces the number of over-segmentations and under-segmentations.

Close-Range Human Detection and Tracking

In this chapter, we address the problem of stereo based object detection from the perspective of a moving human observer wearing a head-mounted stereo camera system. On the one hand, from this viewpoint many pedestrians in a crowded scenario are only partially visible due to occlusions at the image boundaries. In such situations, standard full-body object detectors such as (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b) are not well-suited since they cannot deal with the large degree of occlusion. On the other hand, we can take advantage of the elevated viewpoint of a head-mounted camera, which typically leaves the head-shoulder region of close-by pedestrians well visible. The development of such an upper body detector was motivated by a real problem experienced during the EUROPA Project. During presentation of the robotic integration platform (cf. Fig. 3.12(b)) in shopping zones, pedestrians were often coming close to the robot or were passing by the robot from the sides. Consequently, the lower part of the body was not visible in the camera images and thus these pedestrians were not visible for a full-body-object detector such as (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b). This is a crucial issue, since especially pedestrians close to the robot must be detected robustly and efficiently in order to avoid dangerous collisions. The key objective of our approach, published in Mitzel and Leibe (2012a), is to detect exactly these pedestrians robustly and efficiently, for which a standard full-body object detector consistently fails.

Taking inspiration from a recently proposed human upper body detector for Kinect RGB-Depth data by (Choi *et al.*, 2011a), we propose an improved stereo depth-template based approach which can quickly and reliably detect close-by pedestrians. Based on the *classical* ROI extraction procedure, presented in Chapter 5.2, we extract ROIs in order to reduce the search space of the detector. Our approach learns a continuous normalized depth template from annotations of human upper bodies and slides this template over the extracted depth ROIs at several scales in order to compute a normalized similarity distance score. The output of this process are distance matrices whose entries represent the distance between the template and the overlayed segment of the ROI for each scale. After performing non-minimum suppression (NMS) in the distance matrices we obtain

several detections (from different scales) for a person that are pruned to a set of final detections by a second, template-based NMS stage. We systematically evaluate this approach and characterize the effects of its parameters. In addition, we show how it can be integrated into a mobile multi-person tracking framework.

Besides the technical design and evaluation of our proposed detector, a second main contribution is its empirical demonstration of the somewhat surprising fact that such a relatively simple and fast approach can reach superior detection performance on very challenging outdoor data. This is even more surprising since our approach is based entirely on stereo range data, which is considerably more noisy than the Kinect RGB-D data often used for indoor scenarios. We argue that in a scenario of a shopping street, where the majority of the extracted ROIs contain a pedestrian, it might be overkill to use a computationally expensive and complex detector such as Felzenszwalb *et al.* (2010b), since with such a simple approach we can already detect most of the pedestrians in the scene with a low false positive rate.

The chapter is structured as follows. The next section discusses related work. After that, Sec. 6.2 presents an overview of our detection framework. In Sec. 6.3 we show a detailed experimental evaluation of the full design space of the detector. Sec. 6.4 introduces possible extensions to the proposed detector that can further decrease the computational time.

6.1. Related Work

The ability to reliably detect pedestrians in real-world images is required for a variety of automotive and robotics applications. Nowadays, state-of-the-art object detectors such as (Dalal and Triggs, 2005; Felzenszwalb *et al.*, 2010b) yield remarkably accurate detections for fully observed pedestrians. However, highly occluded scenarios still present major problems. Furthermore, these approaches are computationally expensive, which limits their applicability on autonomous platforms.

For detecting pedestrians that are undergoing partial occlusions, Wojek *et al.* (2011) propose a framework where a full object detector and object part detectors are combined in a mixture of experts based on their expected visibility. Enzweiler *et al.* (2010) learn local body part detectors combined in a mixture-of-experts framework supported by stereo and flow cues. Wang *et al.* (2009) performed occlusion handling in a modified SVM framework by combining HOG and LBP features. However, all of these methods suffer from high computational cost of either features or classifiers.

To deal with this problem, several approaches have been proposed to restrict the execution of the detector to only few ROIs that are extracted based on, e.g., stereo range data (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrilu and Munder, 2007), motion (Enzweiler *et al.*, 2008) and scene geometry (Geronimo *et al.*, 2010a; Hoiem *et al.*, 2006; Sudowe and Leibe, 2011). We follow a similar strategy of ROI extraction based on stereo information in order to reduce the search space for the detector. As a result,

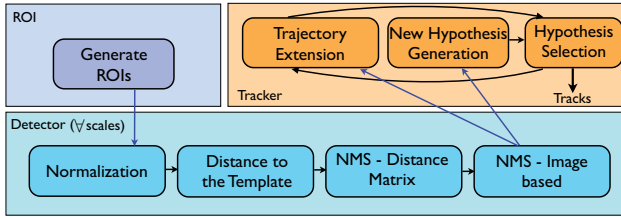


Figure 6.1.: Overview of the different modules of the proposed approach. Blue arrows indicate the interaction between the individual modules. Black arrows represent the interaction within the modules.

we reduce the computational effort, but also the number of possible false positives since only image regions are evaluated that are likely to contain target objects.

Several existing approaches incorporate the stereo information in order to improve detection accuracy. Gavrilu and Munder (2007) employ depth cues for first extracting the ROIs. After generating detection hypotheses by measuring the Chamfer distance between a learned shape contour model and the image input, the hypotheses are verified by cross correlation between the two stereo images. Walk *et al.* (2010) propose a disparity statistics feature which is combined with motion features and HOG, resulting in a large gain in performance. Spinello and Arras (2011) propose a full-body pedestrian detector using dense depth data from an RGB-D Kinect camera. Based on the idea of HOG, the approach introduces Histograms of Oriented Depths as a new feature that follows the same computation procedure as HOG. The approach by Ikemura and Fujiyoshi (2010) proposes a histogram-based procedure for detection of humans in the *time-of-flight* (TOF) data, by learning a histogram-based template that is used in the detection process. The approach by Choi *et al.* (2011a) proposes an ensemble of detectors on multiple perceptual cues, where the output of all detectors is fused using an RJ-MCMC tracker. One of their detectors is a manually created binary template that is compared to the depth image of a Kinect RGB-D sensor to measure the distance to the observed shape of a human. We take inspiration from this approach, but refine and improve it in several respects. In particular, we propose a continuous normalized-depth template, which is learned from annotated upper bodies of pedestrians in noisy stereo data. As we show experimentally, our detector performs significantly better than the binary one, as it represents the distribution of the depth values of the objects more precisely.

6.2. Approach

The key objective of our approach is to be able to detect pedestrians in a close range to the camera, where a standard full body detector will usually fail. In addition, by

focusing only on promising ROIs which are likely to contain a target object, we want to significantly reduce the computational cost and the number of false positives of our detector.

In Fig. 6.1 we illustrate a compact overview of our proposed detection and tracking framework. For each new frame, given the stereo pair and the corresponding depth map, we apply the *classical* ROI extraction procedure as already presented in more details in Chapter 5.2. For each extracted 3D ROI, we generate the corresponding ROI in the image plane by backprojecting it from the ground plane to the image. The 2D ROIs are passed to the detector which slides the learned upper body template over the ROIs and computes a distance matrix consisting of the Euclidean distances between the template and each overlaid, normalized depth image segment. Using a minimum filter on the distance matrix, we obtain possible bounding box hypotheses for the upper bodies. These hypotheses are further pruned to a final detection set by using a template-based intersection-over-union (IoU) NMS stage, where the detection with the lowest distance is chosen first and all other detections within a certain overlap are removed iteratively.

6.2.1. Learning

We first learn a depth template from 600 annotations of upper bodies. To this end, the annotated depth regions are first normalized by the median depth and scaled to a fixed size of 150×150 pixels. Before summing up the annotated bounding box contents to a final template, a binary mask of the same size is generated for each annotation, where the invalid pixels (pixels that failed the left-right check of the stereo algorithm) are set to zero and valid pixels to one. Finally, all the annotations and masks are summed up to obtain the template illustrated in Fig. 6.3(a) by dividing the summed annotations by the summed masks in a point-wise manner.

6.2.2. ROI Extraction

In contrast to classical sliding window object detectors, such as HOG (Dalal and Triggs, 2005), we reduce the search space for the detector to few small ROIs and few scales in the image. In order to achieve this, we exploit the prior information about the location of the objects on the ground plane based on stereo information, similar to (Bansal *et al.*, 2010).

The ROI extraction corresponds to the approach *classical* ROI we presented in Chapter 5.2. From the 3D ROIs, we generate the corresponding 2D ROIs, which will be scanned by the detector in image space. This is done by backprojecting the ROIs from 3D to the image plane. The width of the corresponding bounding box is derived from the width of the ROI, and the height is obtained from the highest point that is enclosed by the ROI. Before extracting the ROIs, we first perform point cloud labeling presented in Chapter 5.1, in order to filter out points from **elevated** structures and to fix the attention of the detector only to the ROI generated from points labeled as **objects**.

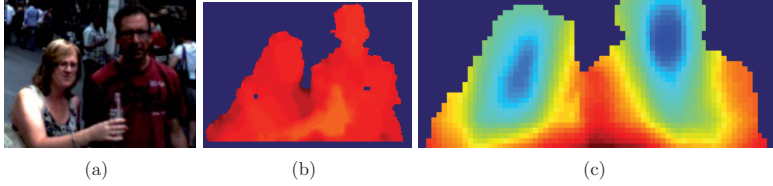


Figure 6.2.: (a) Original ROI from the left image. (b) Input depth from the ROI. (c) The resulting distance matrix for the initial scale.

6.2.3. Depth Template Detector

The pipeline for the detector consists of the following steps. At first, for each ROI in the image plane, we discard the pixels which are not in the depth range of the ROIs in 3D by setting them to zero, as illustrated in Fig. 6.2(b). Then starting from an initial template size that is one third of the ROI height, we slide the template over the ROI. At each position, the segment of the ROI that is overlaid with the template is normalized with its median depth and then the distance between the template and the segment is computed. As a final result we obtain a distance matrix that contains for each position of the template the corresponding distance to the segment in the depth image (cf. Fig. 6.2(c)).

Distance Measure. For the distance measure between the template and the ROI we explore two options: the *Hamming Distance* on binarized template and depth as proposed by (Choi *et al.*, 2011a),

$$\sum_{(i,j) \in W} |T_{bin}(i, j) - I_{bin}(x + i, y + j)| \quad (6.1)$$

and the *Euclidean Distance* on continuous depth-normalized templates:

$$\sum_{(i,j) \in W} \min(d_{max}, (T(i, j) - I(x + i, y + j))^2), \quad (6.2)$$

where in both cases T denotes the template and I the ROI.

In the case that two pedestrians are walking closely behind each other, the resulting ROI will cover both of them. The person in the background will, however, be occluded by the person in front. By overlaying the template over the person behind, some of its pixels will fall in the occluded region. The resulting distance between the template and the overlaid region will increase, leading to a rejection of the detection. To avoid such false negatives, we use a truncated distance that clips the contribution of each pixel to a fixed maximum value.

Multi-Scale Handling. Most of the pedestrians in our scenarios are walking close to each other in groups of two or more people. This often leads to ROIs that represent

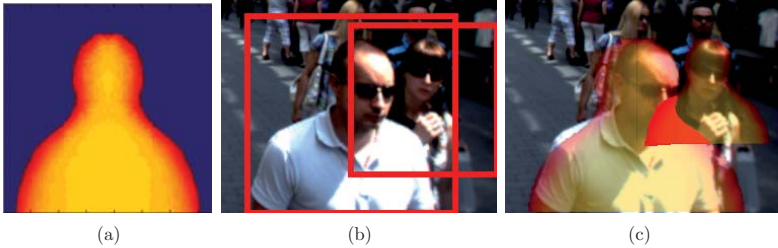


Figure 6.3.: (a) Depth template learned based on 600 upper body annotations. (b) Illustration of the bounding box based intersection over union. (c) Template based intersection over union.

the persons not individually but cover entire groups of pedestrians. As the pedestrian heights may vary, we need to rescale the template, because the initial scale estimate based on the height of the 2D ROI might not be representative for all pedestrians in the group. Note, here we only need to down-sample the template starting from the initial scale, since the height of the ROI in the image plane is based on the highest 3D point (from the tallest person) that fall inside the 3D ROI, as already mentioned before. Each downsampled version of the template is slid over the ROI again, generating an additional distance matrix.

Non-Minimum Suppression. For each scale of the template, we obtain a distance matrix on which we perform NMS by applying a minimum filter of size 3×3 . As a result we obtain few positions in the distance matrix which are the minima in their neighborhood. The multi-scale approach introduces several additional detections on a person for a number of neighboring scales, as the scale stride is usually small. To reduce this set to only one representative detection for each pedestrian, we additionally perform an image based NMS, where we iteratively select the detected bounding box with the lowest distance and remove all bounding boxes that have a certain overlap with this box. A standard approach for this is to compare the bounding box intersection-over-union (IoU) score. However, such a strategy will cause some correct detections to be rejected, as shown in Fig. 6.3 (b), (e.g., for persons walking behind each other, where the overlap between the boxes is high). To cope with this problem, we propose a more precise intersection criterion which is based on the intersection between the templates (intersected pixels that are non-zero) of the individual detections. In Fig. 6.3(b,c) we show the comparison between the bounding box based IoU and the template based one. In our experiments we used a threshold of 0.2 in IoU for pruning the detections. In the classical bounding box based approach, this leads to rejection of one of the detections, whereas in the template based approach both detections will be kept. Further

quantitative experiments (see Sec. 6.3) underline that this criterion is better suited for our scenario.

6.3. Experimental Evaluation

To assess the performance of our upper body detector we report a detailed experimental evaluation of the full design space of the detector that gives clear guidelines how to apply it for optimal performance and fast evaluation.

Dataset. The evaluation was performed on our TORSO dataset presented in Chapter 3.6 that contains very challenging sequences captured from a head-mounted camera setup (cf. Fig. 3.12(c)). The images were captured at 15 fps in highly crowded central shopping streets. The evaluation set consists of 2,543 frames, in which we have annotated 19,461 pedestrians. For all frames we have computed visual odometry using the approach proposed by Geiger *et al.* (2011). For stereo depth estimation we used the fast and robust algorithm presented in (Geiger *et al.*, 2010).

In all the following experiments we applied the evaluation criteria from Everingham *et al.* (2010). In each frame the detection boxes are compared to manually annotated ground-truth boxes. In case that the intersection-over-union overlap of a detection box with the ground truth annotations is above 0.5, the detection is assumed to be correct. To investigate the performance of our detector for different distance ranges, we adapted the annotation files by setting the annotations that are beyond the range to *don't cares* (which result in neither false positives nor missing detections).

Overall Performance. In Fig. 6.4 we present a detailed evaluation in terms of recall vs. false positives per image (fpfi) over several parameters in our detector pipeline. First, we compare our proposed continuous template to a binary version similar to the one used in (Choi *et al.*, 2011a) for three different distance ranges of 5, 7, and 10 meters. To generate a binary depth-shape template (which was generated manually in (Choi *et al.*, 2011a)), we used our continuous template by simply converting all pixels above zero to one. As shown in Fig. 6.4(a), our detector performs significantly better (3-5% higher recall) for the close range of up to 7 meters, which shows that our continuous template represents the depth distribution on the object better. Note that the binary template approach also benefits from the innovations of our ROI extraction, template learning, and template-based NMS and would probably perform worse in the original form as presented in (Choi *et al.*, 2011a).

Comparison with DPM. We also run the DPM detector from (Felzenszwalb *et al.*, 2010b) on our sequences. The results plotted in Fig. 6.4(b) underline the complexity of the dataset, as many persons are occluded by the image boundaries and cannot be detected with a full-body detector. As expected, the DPM detector performs better for the range up to 7 meters than for the range up to 5 meters, since starting from 5

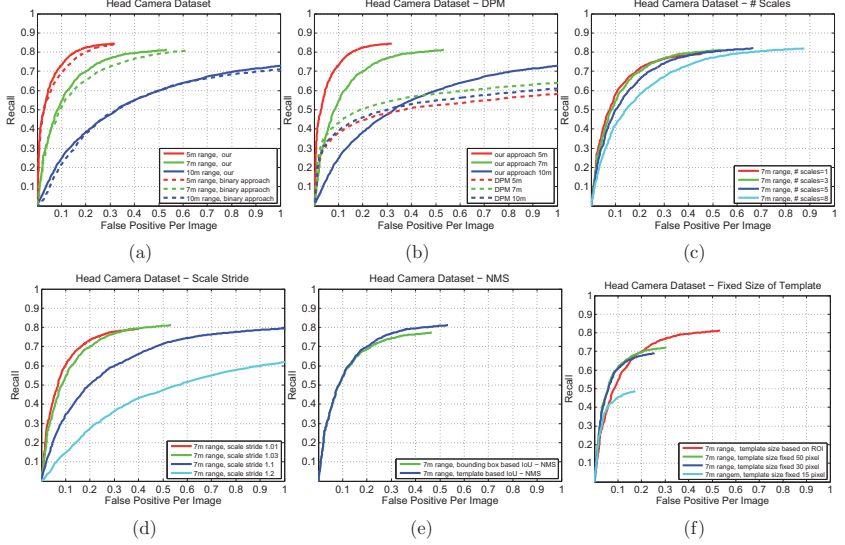


Figure 6.4.: Quantitative detection performance in recall vs. fppi of our approach performed on the full design space of the detector. If not explicitly mentioned otherwise, the number of evaluated scales was set to three and the scale stride was set to 1.03. (a) Performance of our detector for different distance ranges in comparison to a baseline approach (Choi *et al.*, 2011a). (b) Comparison of our detector to the DPM detector (Felzenszwalb *et al.*, 2010b) for different distance ranges. (c) Analysis of the number of scales. (d) Analysis of employing different values for the scale stride. In this experiment we used 5 scales. (e) Evaluation of different IoU criteria, namely template based and bounding box based, as discussed in Sec. 6.2.3. (f) Effect of setting the size of the template to a fixed value. The number of evaluated scales was set to one.

meters the pedestrians become fully visible. Still, the results clearly show our approach’s superior performance for this test scenario.

Number of Scales. In the next experiment, we explore how many scale evaluations are necessary in order to reach the best possible performance. The results of this test parameter are plotted in Fig. 6.4(c), showing that we reach the best performance with 1-3 downscales of the template (the scale stride in this experiment was 1.03). Essentially, the result indicates that the initial scale estimation based on the height of the 2D ROI box is accurate enough and further downscaled versions of the template introduce more false positives.



Figure 6.5.: Detection results, showing that our detector detects most of the present pedestrians close to the camera. The maximum range for this test was set to seven meters.

Scale Stride. The above observation is corroborated by our next experiment, in which we vary the scale stride using 5 scales. As shown in Fig.6.4(d), increasing the scale stride from a base setting of 1.01 introduces more false positives without significantly increasing recall. In all further experiments, we use a scale stride of 1.03, which is a good compromise between detection accuracy and processing speed.

NMS. Next, we explore how the different NMS approaches described in Sec. 6.2.3 affect the overall performance of the detector. As expected, the template-based NMS performs better (Fig.6.4(e)), since it takes into account the expected pedestrian shape. We obtain a 3% higher recall at 0.4 fppi. Note for all the experiments we used an overlap threshold of 0.2.

Fixed Size Template. As can be seen in the example results shown in Fig. 6.5, many pedestrians appear close to the camera, which results in large 2D ROI boxes. In a further experiment, we therefore investigate how the resolution of the template affects



Figure 6.6.: Tracking results obtained for the *High-level* the tracker framework presented in Chapter 3.5.

the performance of the detector and whether we can reduce the initial resolution of the template in order to decrease computational cost. The results in Fig. 6.4(f) show that by fixing the size of the template to a fixed small value of 30×30 pixels and down-sampling the ROI to the corresponding size, we still obtain a reasonable performance of about 70% recall at 0.3 fppi.

Computational Performance. Our current system, including ROI candidate generation, object detection, and tracking runs without any parallelization with more than 25 fps on a single CPU of an Intel Core2 Quad Q9550 @ 2.83GHz, 8GB RAM. (For this experiment we used 3 scales, scale stride 1.03 and distance range of 7 meters). There is still considerable optimization potential, since the ROI comparisons can easily be parallelized.

Qualitative Evaluation. Finally, Fig. 6.5 shows some qualitative results achieved on our sequences. It can be seen that our system is able to detect nearly all pedestrians correctly with only few false positives. In addition, our approach correctly tracks detected

persons over time keeping correct person identities (cf. Fig. 6.6). For tracking we used the *High-level* tracker presented in Chapter 3.5, where the height of detection bounding boxes was adapted to a full-body representation by multiplying it with three.

6.4. Extensions

In this section we will describe two further extensions that allow us to decrease the computational effort further and thus to reach frame rates over 45Hz with a similar detection performance. A typical sliding window pipeline is computationally inefficient, because the classifier function has to be evaluated over a large set of candidate sub-windows. One possible solution could be to increase the position stride, which will reduce the number of classifier evaluations, but can cause imprecise object localization and increase the number of false negatives. Instead of increasing the position stride, we fix the evaluation of the template only to local maxima of the contour, extracted from the corresponding ROIs as shown in Fig. 6.8(c). The contour is represented by a maximum y non-zero value at each x position which is extracted from the corresponding depth image of an ROI (cf. Fig. 6.8(b)). In order to reduce the number of local maxima in the depth image that are caused by noise, the contour is smoothed with a Gaussian before maxima extraction. As can be seen in Fig. 6.8(c), the extracted local maxima do not correspond to the exact location of the middle of the human head. For this reason the depth template is evaluated for each scale not only at the local maxima but also in an eight-neighborhood pattern. In order to analyze the effect of this extension on detection performance quantitatively we applied the detector on two sequences *BAHNHOF* and *SUNNY DAY* from the Zurich Mobile pedestrian corpus. Since the original annotations were not subdivided into different distance ranges we run the detector for all ROIs for up to 30 meters. As shown in Fig. 6.9, the resulting performance is the same or even slightly better (cf. 6.9(a)) as when using a sliding window approach, caused by reduction of the number of false positives. For the sake of completeness the plots contain further detection results using the HOG based detection approaches *fastHOG* (Prisacariu and Reid, 2009), *cudaHOG* and *groundHOG* (Sudowe and Leibe, 2011), corroborating that even with such a simple detector we obtain state-of-the-art detection performance. Besides reaching a similar performance with the local maxima extension, we could reduce the computational time from 40ms to 28ms (= 35fps on a single CPU of an Intel Core2 Quad Q9550 @ 2.83GHz, 8GB RAM) for the entire system including ROI candidate generation, object detection, and tracking, but excluding stereo computation.

When considering the extracted contour shown in 6.8(c), the question arises whether it is really necessary to compute the distance between the depth template and the overlaid ROI area for each pixel. Or would it be already sufficient to compute the distance between the corresponding contours at the local maxima locations? The results presented in Fig. 6.9 indicate that using the distance between the contours only is sufficient. The *local max + contour* curve is slightly better or equals to the sliding window based depth



Figure 6.7.: Detection results on BAHNHOF and SUNNY DAY sequences. Shown results were generated using the proposed extension when using the evaluation of local maxima and comparing only the contour and not the entire depth template.

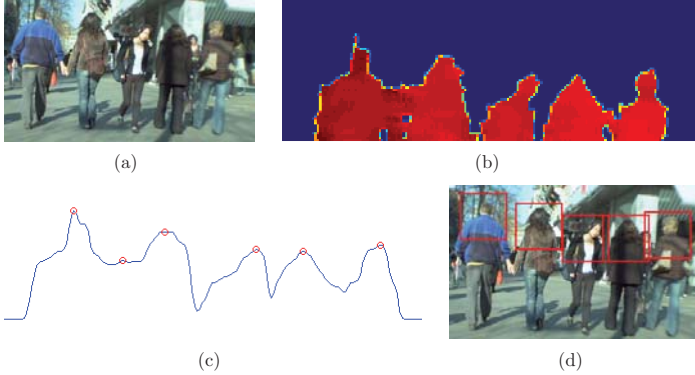


Figure 6.8.: Overview of the contour based approach for upper body detection using local maxima. (a) Original image. (b) Corresponding depth image crop. (c) Resulting contour with the extracted local maxima. (d) Resulting detections.

detector, but with this extension we could boost the computational performance from 25fps to 45fps for the entire detection and tracking system.

6.5. Discussion

In this chapter, we have presented a stereo depth-template based detection approach for detecting pedestrians in the close range. The combination of the ROI extraction with the detector allows us to restrict the detector evaluation only to few scales and to few small segments in the image, which reduces the number of false positives significantly. We have shown that even though this approach is simple and fast to apply, we reach superior performance on noisy stereo data for very challenging scenarios from crowded shopping streets. Additionally, we have shown how this detector can be integrated into an ROI-based multi-object tracking framework. Our simple approach for pedestrian detection works well in typical shopping street scenarios with ROIs that are likely to contain pedestrians. However, we expect that in different scenarios with, e.g., much vegetation, small bushes, or cars, there will be some ROIs having a similar omega shape as we learned from pedestrians, causing more false positives than in the shopping street scenes. The number of false positives increases as well when increasing the detection range as we have shown in our experiments. When increasing the operative range of the detector for up to 10 meters the performance drops significantly. This is also what we have expected, because for objects further away from the camera the depth information becomes imprecise due to the short baseline of 12cm of the employed Bumblebee2 cam-

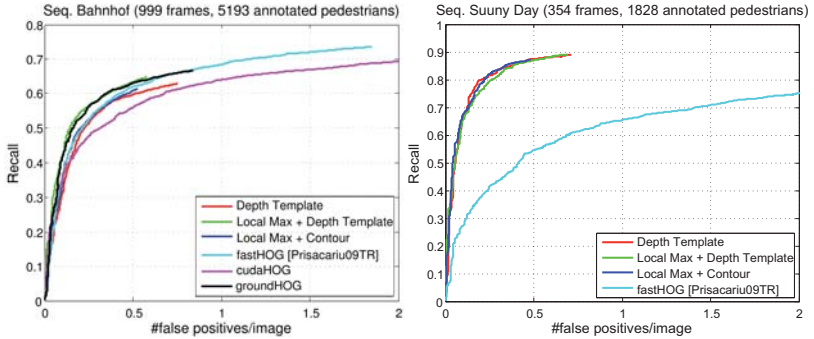


Figure 6.9.: Resulting detection performance by including different extensions of the upper body detector on BAHNHOF and SUNNY DAY sequences. For *cudaHOG*, *groundHOG* (Sudowe and Leibe, 2011) and *fastHOG* (Prisacariu and Reid, 2009) we plot the performance for upscaled images to twice of its original resolution (640×480).

era. One solution could be to use different pedestrian detectors that are selected based on the distance to the camera. For distances up to 7 meters, our proposed upper-body detector could be applied; for distances beyond 7 meters we could run the HOG pedestrian detector, since pedestrians at this distance will be fully visible in the image and not occluded by the image borders. Furthermore, when restricting the HOG detector to run for pedestrians at far range, we will automatically reduce the search space of the detector to a small region in the image occupied by the extracted ROI. With such a combination, we could still achieve real-time performance.

Another interesting extension of the proposed detector approach would be the application of ROI segmentation presented in the Chapter before, which will further reduce the computational effort. Before scanning the extracted ROIs with the learned depth-template, we could divide the ROIs into individual objects. With this we could avoid the multi-scaling that we introduced in order to ensure detection of pedestrians of different sizes comprised by an ROI. Additionally required computational resources for the ROI segmentation could be neglected since the segmentation takes less than 2ms for each ROI.

During the experiments with the proposed detector we made another important observation. The depth image from a person with a front/back orientation with respect to the camera differs significantly from a person with a left/right orientation. Especially in the latter case, the derived distance to the template is often above the rejection threshold, which leads to several false negatives. One possible solution in order to cope with this problem could be to learn orientation dependent (front/back, left/right) depth templates, which can then be applied in parallel during the ROI scanning procedure.

The resulting output of both detectors can then be merged and passed to the proposed non-minimum suppression. Since both detectors can be applied in parallel, the resulting computational performance should stay similar to the one we reached with a single depth-template evaluation.

Tracking with Time-Constrained Detection

In this chapter, we consider a problem that we were facing during the integration in the European Project EUROPA, mentioned in the Chapter 3. Several vision components such as visual odometry, object detector and tracker had to be running on a single laptop, which means the resources had to be distributed between the mentioned components. Since the object detector is the most expensive component of our framework, we tried to investigate the case of an object detector with a fixed time budget *in the context of a tracking system*. We assume that the detector can only process a small number of ROIs in each frame, but we balance the ROI selection over time, such that at each time instant, only those ROI candidates are considered for which attention is most urgently required in order to produce stable tracking results. The question we pose is: given a detector with a budget to attend to k ROIs in each frame and a cheap low-level tracking system to follow ROI candidates over time, which are the ones that should be selected? To address this question, we propose the following approach. We first create ROI candidates from a depth map of the scene and from already existing object trajectories. These candidates are associated and tracked over time using local depth and appearance information. We then model the selection process of k ROIs to be verified by the detector using a statistical Poisson process model. Briefly stated, this model associates each tracked ROI candidate with a low probability of causing an important event. For regions in the background, this event means that the region now contains a person, despite of this having previously been verified as not being the case. For regions on tracked person trajectories, the event indicates a tracking failure that causes the low-level tracker to drift. In both cases, the occurrence of an event has the consequence that the region should be attended to and be verified by the object detector. Since we cannot predict where those events will happen, we model their probability of occurrence using a Poisson process. The result of this process indicates the *urgency* by which the detector should attend to a region in order to limit the probability of the event influencing the tracking results. In our approach, the urgency of a region is additionally moderated by its *utility* for maintaining tracking performance, which gives preference to regions close to the camera.

In order to separate the different effects of foreground and background regions (i.e., regions stemming from already existing trajectories and regions in which no person has been found yet), we propose to apply a two-tiered model. For foreground regions, the Poisson process accumulates the uncertainty of the individual tracking steps, while it assumes a fixed event occurrence rate for the background regions. In addition, we extend the model with a special treatment for trajectories that are predicted to emerge from an occlusion. As such an event requires immediate attention in order to apply potential corrections, we always give preference to such regions. Once the selected ROIs have been verified by the detector, its output is converted to 3D world coordinates using the camera position from SfM, together with an estimate of the ground plane. We then integrate the 3D measurements in a multi-hypothesis tracking approach similar to the one presented in Chapter 3.4. As our experimental results will demonstrate, our approach reaches state-of-the-art performance with high tracking quality, even with a significantly reduced time budget for the detector. We experimentally investigate the time budget required for robust system-level performance and show that employing the stochastic Poisson process model optimizes ROI selection, such that only three detector evaluations per frame are sufficient for obtaining a highly robust tracking system. The approach developed in this chapter was published in Mitzel *et al.* (2011a).

In summary, we make the following contributions:

- We demonstrate how ROI selection can be optimized in general by employing a Poisson process model and how this model can be adapted for a *tracking-by-detection* approach.
- In order to satisfy the conflicting goals of detecting new objects while stabilizing already existing tracks, we propose a two-tiered realization of the Poisson process model that takes into account a track’s accumulated uncertainty.
- We experimentally show that the proposed framework achieves robust multi-person tracking performance even with few ROI detector evaluations, making it possible to reduce detector evaluation to a minimum.

7.1. Related Work

The task of selecting ROIs is closely related to visual attention. Saliency based visual attention systems (Frintrop *et al.*, 2005; Itti *et al.*, 1998) typically extract image areas which differ from surrounding distractors by their unique color and intensity. However, those approaches are not applicable for our task, where not only one individual object sticks out of the background, but where many people with potentially similar appearance need to be detected and reliably tracked.

Poisson process models were already applied for active scene exploration with pan-tilt-zoom cameras (Sommerlade and Reid, 2008a,b, 2010). Those approaches model the chance of pedestrians appearing in a certain area in order to decide whether to zoom

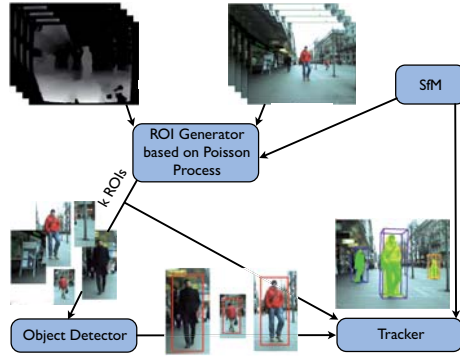


Figure 7.1.: Overview over the different components of our tracking system and their connections.

inside this area and risk missing some occurrence of newly appearing persons in other areas. Our scenario is however different in that the Poisson process models are attached to moving objects, which periodically need to be re-attended, rather than to fixed areas.

7.2. System Overview

Fig. 7.1 shows an overview of our proposed tracking system. The system consists of three major components: ROI candidate generation, object detection, and tracking, which will be explained in more detail in the following.

For extracting the ROI candidates, we rely on depth information, which we assume to be available nowadays in real-time through dedicated sensors (e.g., Microsoft’s Kinect) or hardware processing solutions (e.g., (Rabe *et al.*, 2010)). In addition, we use visual odometry to estimate the camera vehicle’s ego-motion, and we estimate the scene ground plane in each frame. For both tasks, there are also real-time approaches available (Labayrade and Aubert, 2003; Mei *et al.*, 2009) as well as approach by Floros and Leibe (2012) we presented in Chapter 3.3.

Given a color image and a corresponding depth map, we extract ROI candidates using the classical approach, described in Chapter 5.2. For each new ROI candidate, a Poisson process is initialized modeling the urgency of verification of the ROI by the detector. The candidate regions of past frames are associated with the newly extracted ROI candidates in the current frame and the urgency is propagated to the new regions. Depending on the time budget defined for the detector, a certain number k of ROIs with the highest urgency is verified by the detector. The detector output then enables the multi-hypothesis tracker to initialize new trajectories or to extend the existing ones based

on an Extended Kalman Filter (EKF). Overall, this results in a robust tracking system running at more than 15 Hz and allowing to define a time budget for the computationally expensive object detector. The time budget is represented by the number k of ROIs which are verified in each frame.

7.3. Poisson Process Attention Model

A Poisson process is a stochastic process in which events occur continuously and independently of each other. Mathematically, the process is described by a collection of random variables $\{N(t) : t \geq 0\}$ where $N(t)$ is the number of events that have occurred up to time t . Given a rate parameter λ , if the interval times are independent and obey exponential distributions (Poisson distributions) $Exp(\lambda)$, then a Poisson process is formally defined as

$$P\{\text{interval time} > t\} = \exp(-\lambda t)$$

From this, one can directly derive the chance of an event to occur. This chance increases with each time step since the last event occurrence at t_0 and can be defined as:

$$p(T < (t - t_0)) = 1 - \exp(-\lambda(t - t_0)) \quad (7.1)$$

where T is the waiting time until the next event. As described above, we fix the time budget for the detector, such that it is only allowed to evaluate a certain number of ROI candidates. In order to resolve the question in which order the ROI candidates should be verified by the detector, we will in the following model the urgency for verifying the region of interest using a Poisson process.

Having a busy street scene scenario, where pedestrians show up regularly, we consider the occurrence of a person within an ROI as a random event. We model the waiting time T until the next occurrence within ROI $\mathbf{r}$ by an exponential distribution with the occurrence rate λ . The chance of an occurrence having taken place always increases with each frame where the ROI $\mathbf{r}$ is not verified by the detector and is computed by Eq. 7.1. As we are interested in tracking all persons in the scene, this chance corresponds to the urgency by which the ROI should be attended to in order to detect newly appearing persons.

After generating new ROI candidates, we associate the ROIs from the last frame with the new ones and propagate the time when the previous ROI was evaluated to the new ROI, thus increasing the urgency for verifying this part of the image. More details on the association procedure for ROIs can be found in Section 7.4. For the remaining new ROIs that could not be associated to ROIs from the previous frame, we start a new Poisson process. The urgency for the new ROIs is set to the lowest value by setting $t_0 = t$. Consequently, the detector always attends the ROI candidates with the highest urgency.

So far, the described ROI association process is a so-called background process, where we try to find newly appearing persons in the background and start new tracks for

them. In addition, we run a second Poisson process (foreground process) for each already existing trajectory, which incorporates the track consistency based on the appearance model. This step assures that already found trajectories do not get lost due to low fixed-time budgets for the detector. To this end, we run a non-homogeneous Poisson process, whose rate function can change over time, representing the appearance model's consistency of the track:

$$\lambda(t) = w_{\text{tr}} \sum_{i=t_0}^t (1 - bhatta(t_i)), \quad (7.2)$$

where w_{tr} is a weighting factor, t_0 the time since the last detector verification, and $bhatta$ is the Bhattacharyya coefficient between the new region's color histogram and the last associated detection. When a region was evaluated by the detector, the urgency is reset to zero as $t_0 = t$ in Eq.7.1.

In addition to the Poisson process, we introduce a further *utility* factor, which weights the ROI candidates with respect to their distance to the camera as follows:

$$utility(\mathbf{r}) = 1 - \exp(-w_d/d_{cam}), \quad (7.3)$$

where w_d is a weighting parameter and d_{cam} represents the distance to the camera. The *utility* factor is necessary in order to give the detector a preference for attending close-by regions, which are important for tasks like collision avoidance or pedestrian safety. This is a valid assumption as it was already shown in several saliency applications (Jansen *et al.*, 2009; Lang *et al.*, 2012) that objects closer to the viewer are usually examined first.

In order to select the k ROIs for verification, both measures, *urgency* and *utility*, are combined for ROI $\mathbf{r}$ as:

$$w(\mathbf{r}) = 1 - \exp(-\lambda(t - t_l) - w_d/d_{cam}) \quad (7.4)$$

where t_l represents the frame where the ROI was last evaluated. The detector is then triggered only for the k ROIs with the highest weight $w(\mathbf{r})$.

7.4. Detailed Implementation

Depth based ROI Generation. The idea behind the ROI extraction using stereo data is to fix the attention of the detector only on the few regions which may contain a wanted object. This allows us to run the computationally expensive detector only on small image regions regarding only few scales, rather than sliding over the whole image and all possible scales, which is computationally very expensive.

The results of ROI generation are shown in Fig. 7.2. ROI candidates were extracted based on the *classical* ROI extraction process from Chapter 5.2. Each ROI candidate is represented by its center of mass and its width. Note that these ROIs are regions

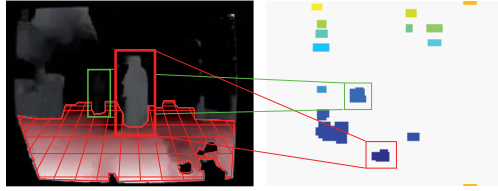


Figure 7.2.: An example of the stereo based ROI generation. The left part is the depth map. The right image part is the stereo range data projected onto the ground plane.

in 3D world coordinates. In order to obtain the corresponding image region, we take a rectangle with the width of the ROI, and a height of 2 meters centered on the ROI's center of mass and parallel to the camera and project it into the image.

Object Detection. For pedestrian detection, we use the popular HOG detector proposed by Dalal and Triggs (2005) in an efficient GPU implementation by Sudowe and Leibe (2011). Without any further constraints, this implementation processes 640×480 images at 22Hz ¹ and 1280×960 images at 5Hz , while achieving the same detection performance as the original HOG detector from Dalal and Triggs (2005). When applied to full images, the GPU's power consumption however presents a serious limitation to its use for autonomous systems, restricting the vision system's battery life.

The advantage of our approach is that only small regions of interest need to be evaluated, rather than sliding the detector over the entire image for all 27 scales. Thus, in each frame we call the detector for evaluating k ROIs with 5 scales per ROI. The scales are determined as follows. The base scale is the result of dividing the height of the ROI in image coordinates by 128 (the height of the sliding window of the HOG detector). To ensure the detection of pedestrians inside the ROI, we also consider two scales above the base scale and two scales below that are computed in multiplicative scale steps of 1.05. To summarize, we run the detector on small ROIs, which means computing the features only for those small regions and considering only five instead of 27 scales per region. This gives us an enormous speed-up – on average, only 2 ms computation time are required per ROI, instead of 44 ms if we slide over the entire image.

ROI-based Tracking. The tracking model we employ in this chapter is the one presented in Chapter 3.4. Besides SfM and the ground plane, it requires as input the bounding box detections of the corresponding target objects, in each frame. However, since we set a fixed time budget for the detector and verify only k regions of interest, some of the measurements that are required for extending the existing trajectories will be missed. To cope with this problem, we additionally use the ROI candidates for gen-

¹Timings reflect the state of the cudaHOG detector at the time of development and are measured on a machine with Intel Core2 Quad CPU Q9550 @ 2.83GHz processor, 8GB RAM, and an Nvidia GeForce GTX 280 graphics card.

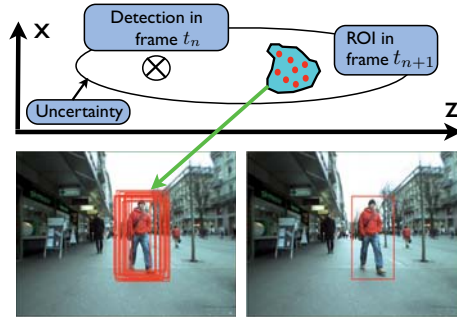


Figure 7.3.: Visualization of how we create new observations for the EKF based on the detection from the last frame and the extracted ROI candidates. For each detection from the last frame, we sample points from ROI candidates which are inside a certain uncertainty region around the last detection. The final observation is computed by the mean of sampled points which are weighted corresponding to the Bhattacharyya distance between the last detection and the back-projection of the individual sampled position.

erating measurements (observations) required for the EKF updating step. This is done in the following way (cf. Fig. 7.3). For frame t_{n+1} , we sample for each detection in frame t_n a number of $M = 20$ points randomly from the regions of interest in frame t_{n+1} that are within a certain uncertainty region around the detection. These points are back-projected to the image, generating possible detection bounding boxes. Next, we compute the appearance similarity based on RGB color histograms of the generated bounding boxes with the detection in frame t_n , employing the Bhattacharyya distance. The final detection is the mean of all sampled points weighted by the appearance similarity. Alternatively, the ICP tracker presented in Chapter 4 could be employed instead of the proposed sampling approach in order to generate detections for the tracker.

In some cases, due to noisy depth information, the new sampled detection is not correctly aligned to the pedestrian, causing the tracker to drift. By using the non-homogeneous Poisson process for each existing track, as described in Section 7.3, the drifting is detected through the appearance change. Thus, the urgency for a detector verification increases rapidly, resulting in the detector to be triggered for this area and the track to be revised. This is a crucial step in our tracking system, since we require at least three successive measurements for starting (high-level) a trajectory, but once an ROI is evaluated, the urgency for that ROI is set to zero and as a consequence it will not be verified by the detector in the next frames.

ROI Propagation. In each frame, the newly extracted regions of interest need to be associated with the ROIs from the last frame. To this end, we define a certain covariance that depends on the maximum velocity of a pedestrian. We assume that a pedestrian

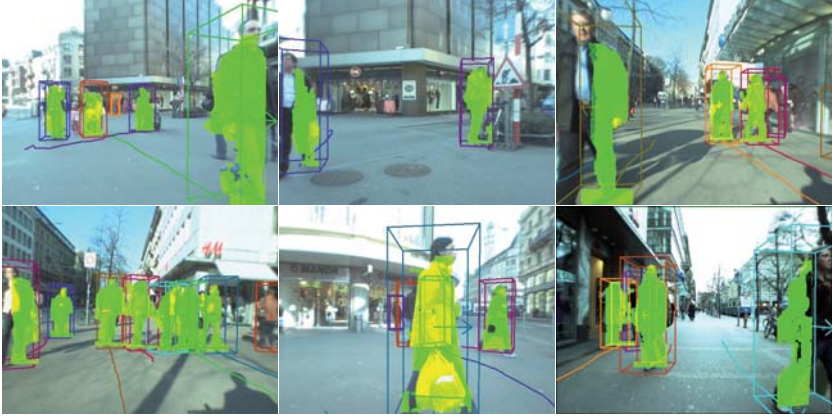


Figure 7.4.: Demonstration of our approach’s capability to continue tracking close to the camera and/or the image borders.

moves with at most $1.38m/s$ (corresponding to $5km/h$). Thus in the z -direction the covariance is set to $1.38^2/(frame\ rate)$ and in x -direction to $0.4^2/(frame\ rate)$. Then given a new ROI and the covariance matrix

$$\Sigma = \begin{pmatrix} cov_x & 0 \\ 0 & cov_z \end{pmatrix}$$

with the aforementioned values, we evaluate the following equation for each ROI in the last frame:

$$w = \exp(-0.5 * (\mathbf{x}_{old} - \mathbf{x}_{new})^T \Sigma (\mathbf{x}_{old} - \mathbf{x}_{new})) \quad (7.5)$$

where $\mathbf{x}_{new}$ is the center position of the new ROI and $\mathbf{x}_{old}$ of the old one. For $w > 0.95$, the ROIs are linked and the frame number of the last verification by the detector is updated for the new ROI with the number of the associated one. In case more than one ROI is inside the uncertainty region, we associate the new ROI with the closest one.

For historical reasons, we relied for that approach on the *classical* ROI extraction procedure (without point cloud labeling) that output ROIs that usually contain several objects grouped together. However, while groups of objects are moving the resulting ROI could split into individual objects or merge again from frame to frame, causing wrong associations during the ROI propagation. A simple extensions to overcome this problem could be to segment the ROIs in each frame into individual objects as we proposed in Chapter 5.3.

Occlusion Handling. For correct association of a person reappearing after a person-to-person occlusion, it is helpful to detect imminent occlusions first. Similar to the

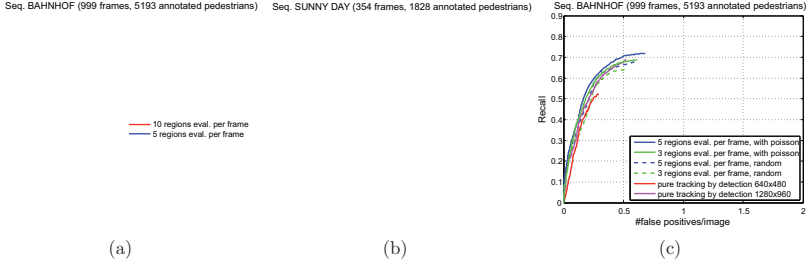


Figure 7.5.: Quantitative tracking performance of our approach compared to different baselines on the BAHNHOF and SUNNY DAY sequences from (Ess *et al.*, 2009b).

approach already described for the LS-tracker, we project the 3D prediction of the EKF of each tracked person into the image, computing the bounding box overlap. For persons with an overlap above 0.5, the occlusion is likely to occur and the person farther from the camera is marked as occluded. For the next 15 frames, we check whether the person is likely to reappear by performing the same bounding box check on its extrapolated EKF prediction. When a reappearance is likely, we create a virtual ROI at the predicted location with the urgency $u_{\max}$, forcing the detector to evaluate this region. This virtual ROI generation is necessary, since it is likely that no valid depth data will be available for the reappearing person due to stereo shadowing from the previously occluding person.

7.5. Experimental Results

In order to evaluate our approach, we applied it to two challenging sequences, BAHNHOF and SUNNY DAY, from the Zurich Mobile Pedestrian corpus, presented in Chapter 3.6.

Tracking Performance. We use the evaluation criteria from (Ess *et al.*, 2009b). Tracking quality is measured by the intersection-over-union of tracked person bounding boxes and ground truth annotations in every frame. Fig. 7.5(a),(b) presents the performance curves in terms of recall vs. false positives per image (fppi) for different numbers of detection verifications k per frame for both sequences. As can be seen, our approach achieves good performance even when verifying only three ROIs per frame. For comparison, we also provide the curves reported by (Ess *et al.*, 2009b) (only BAHNHOF), our Hybrid-LS approach presented in Chapter 4, and (Bajracharya *et al.*, 2009b; Bansal *et al.*, 2010) (both sequences). In both cases, our approach achieves higher recall at 0.5 fppi, showing the advantage of depth-based track propagation. At higher precision levels, the performance is only slightly worse than (Mitzel *et al.*, 2010), even when only using three detector verifications per frame. The weighting parameters of the Poisson process w_{tr} and w_d were set to 0.7 and 10.

7. Tracking with Time-Constrained Detection

# ROIs eval. per frame	(all ($\varnothing$ 12.5))	10	5	3	1	pure t-by-det 640×480	pure t-by-det 1280×960
Runtime for 1000 frames (sec.)	101.4	92.4	78.5	64.5	49.6	69.05	220.02
fps	9.86	10.82	12.74	15.50	20.16	14.48	4.54

Table 7.1.: Runtime for the overall system for 1000 frames on the BAHNHOF sequence. With *t-by-det* as abbr. for *tracking-by-detection*.

Furthermore, we evaluated whether modeling the ROI selection with a Poisson process really pays off. To this end, we randomly sampled k ROI candidates in every frame, instead of using the Poisson model, and evaluated them by the detector. As can be seen in Fig. 7.5(c), the Poisson process model indeed results in better performance (3.5% at 0.5 fppi for five evaluations per frame and 4.8% for three evaluations per frame). The utility factor brings 0.5 – 1%. For a practical application, the benefit is however larger than this number suggests, since the utility factor helps the system focus on tracking close-by persons, which are important for collision avoidance.

In addition, Fig. 7.5(c) compares our approach’s performance to the one of a pure *tracking-by-detection* system, where the detector slides over the entire image (640×480) and all 27 scales in each frame. The poor performance of the latter case can be explained by the fact that we use the same detector setup as in (Dalal and Triggs, 2005) with a 640×128 pixel detection window that constrains the smallest possible detection to this size. However, the annotations contain pedestrians that are much smaller than 128 pixels. In contrast, our approach scales ROIs to the appropriate detection size based on the measured distance to the ROI, permitting upscaling of ROIs by up to a factor of 2. This allows us to also detect pedestrians that are farther away from the camera without additional cost. For achieving equivalent performance with a sliding-window detector, we would need to process a 1280×960 image, for which the GPU detector alone requires 180ms per frame (without any tracking), which is far away from the 15fps (Tab. 7.1).

Computational Performance. The main single computational cost item in a pure *tracking-by-detection* approach is still the computationally expensive detector. With our approach, we can reduce the detector computation time significantly by attending only to a fixed number of small regions. Evaluating a full 640 × 480 image with our detector implementation requires 44ms, compared to 2ms for a single region of interest. Overall, our system, including ROI candidate generation, object detection, and tracking runs at more than 15 frames per second (Table 7.1) on a machine with an Intel Core2 Quad Q9550 @ 2.83GHz, 8GB RAM, and an NVidia GTX 280 graphics card. This does not include stereo computation. However, dedicated hardware solutions (Rabe *et al.*, 2010) and depth sensors (e.g. Kinect) have in the meantime become available that could take over this job. An algorithmic solution for stereo estimation proposed by Geiger *et al.* (2010) which we used for our other approaches requires 100-120ms for a 640 × 480

image. Combining this approach with our tracking framework will clearly slow down the entire pipeline, but one possible solution could be to apply the stereo estimation on a downsampled image that will reduce the computational effort for a 320×240 image to less than 55ms. We assume that even with such a reduced resolution for the stereo maps, the ROI extraction and tracking components should not be affected negatively. The odometry data is assumed to be given as well, which is not a restriction since odometry/SLAM components are standard in mobile robotic systems. The experiments with the EUROPA robot (cf. Chapter 3) corroborated that robot odometry is also precise enough for the tracking purpose.

Qualitative Evaluation. Similar to our previous approach presented in Chapter 4, we can continue tracking pedestrians that are close to the camera or that are partially occluded by the image boundaries (cf. Fig. 7.4). This is an advantage compared to pure *tracking-by-detection* approaches (Ess *et al.*, 2009b), which cannot continue such tracks robustly due to missing detections. This problem could also be overcome in a pure *tracking-by-detection* framework by employing a detector with partial occlusion handling, such as the one from Enzweiler *et al.* (2010) or our proposed upper body detector (cf. Chapter 6) that we designed most of all to deal with the occlusions at image borders caused by close distance to the camera.

Fig. 7.6 presents results of our tracker on both test sequences BAHNHOF and SUNNY DAY, verifying $k = 5$ regions in each frame. In addition to the tracker bounding boxes, we visualize the depth information that was used for computing the ROIs. As can be seen, our system is able to track most of the visible pedestrians correctly in a very busy environment with many occlusions.

7.6. Discussion

We have presented a robust system for mobile street-level multi-person tracking. The core of our system is formed by a stochastic Poisson process that models an optimal ROI candidate selection given a fixed time budget for the object detector. As our experiments have shown, the approach runs at more than 15 frames per second and reaches state-of-the-art performance, while requiring the verification of only 3-5 ROIs in every frame. Our results open several interesting research perspectives. By integrating the depth information over time, one can distinguish between moving and static objects and consequently employ Poisson processes with higher rates for moving objects, since those objects are likely to be pedestrians. Considering a multi-class tracking problem from the view of a moving vehicle where several different object detectors need to be evaluated the selection problem becomes even more complicated. Usually classes like pedestrians or bicyclists are favored due to high vulnerability in case of collision. In such a scenario the Poisson models should be class specific such that for favored object classes the occurrence rate is higher in order to enforce a more frequent verification by the detector for stable tracking results. Furthermore, the proposed Poisson model



Figure 7.6.: Example results on the test sequences BAHNHOF and SUNNY DAY.

can also be integrated into our *tracking-before-detection* framework that will next be presented in Chapter 8. The tracked ROI segments could be augmented by a Poisson

process and evaluated by different classifiers depending on the urgency and stability of the track.

In the original implementation of our approach we still relied on the computationally expensive GPU based HOG detector which limits the application of the tracking framework on a mobile platform. However, for the future development, the HOG based detector can be exchanged by our upper body detector that we presented in Chapter 6. As we have shown with a quantitative evaluation on BAHNHOF and SUNNY DAY (cf. Fig. 6.9) the results of the upper body detector and the HOG reimplementation from Sudowe and Leibe (2011) are comparable, which should consequently lead to similar tracking results.

III

Tracking People and Their Objects

Motivation. Tracking-by-detection approaches, as used in the previous chapters, are naturally restricted to tracking objects for which pre-trained detector models are available.

In order to understand the behavior of people, it is also important to recognize and track other objects in their surroundings. In practical scenarios, this includes a large variety of objects such as bicycles, child strollers, shopping carts, trolleys, or wheelchairs (see Fig. 8.1 for some examples). In addition, people’s motion is often affected by accessories such as backpacks, shopping bags, walking aids, or other personal items. While some of those object categories are sufficiently frequent and distinctive that the effort pays off to learn generic detectors for them in an offline fashion (Felzenszwalb *et al.*, 2010b), this is typically not the case in general. There is an almost endless variety of baggage items that one would need to learn models for, and this variety may change from one place to the next and depend on short-lived fashion trends. We therefore need to develop methods that can detect and track also novel object types and learn models for them online.

This problem is not trivial to solve, because it necessitates a solution for a more fundamental issue, namely the question “what is an object”? When interpreting text, it is easy to detect that there is an unknown word, but in order to do the equivalent in vision, one needs to segment out the object first, i.e., to determine its precise extent in space and time from the input video stream. This requires either familiarity with the unknown object’s appearance (which we explicitly assume not to be known a-priori) or an interpretation of enough of the object’s surroundings to determine that there is an image region that still needs to be explained.

Assuming, that in addition to the known object categories we can track also unknown scene entities such as child strollers, suitcases or wheelchairs, another task arises that needs to be solved for better scene understanding, namely the association of the unknown objects to persons that cause the motion of the unknown objects by pushing or pulling them. Knowing that a person is pushing a child stroller can directly affect the underlying motion model of the Kalman Filter, since the child stroller might enforce a non-holonomic constraint on the agent motion due to its construction. A person pulling suitcases may affect the motion of other scene participants that will avoid it in a larger radius in order to avoid a collision. When observing a group of people, it is apparent that the individuals affect each other’s motion when trying to adapt their motion to the speed and the walking direction of the group. These examples show that being able to associate unknown objects and known objects and to infer the type of interaction we could not only further improve tracking but also obtain a much better semantic scene understanding that in turn can help the robots interact more socially with people. This means that if we can predict the interaction between objects and humans we could enforce the robot not to plan to drive between objects that interact with each other, even if there is enough space in-between those.

In the next Chapter we will present an approach that can track known and unknown object categories in a *tracking-before-detection* manner. We combine our proposed ROI

segmentation approach with a simple ICP tracker that can track the objects over time. On top of it we propose a novel object representation model, *Generalized Christmas Tree (GCT)*, which is built up for each tracked object and essentially captures the volumetric information about the object modeling also the uncertainty of each part of the model. Using this model, we can not only improve tracking, but also use it in order to detect items that are carried by pedestrians. In the following Chapter 9, we use the tracks of known and unknown objects in order to associate them with each other and to infer a possible interaction type. Using a graphical model that requires as input the relative position and velocity between a pair of objects and the corresponding GCTs as appearance model, this model can automatically infer whether a person is pushing or pulling a corresponding object or whether the pair of objects are two persons that walk together in a group.

Tracking Known and Unknown Objects

In this chapter, we describe our 3D tracking approach (Mitzel and Leibe, 2012b) that addresses the problem of tracking, besides known object types such as pedestrians, also unknown scene objects: child strollers, walking aids, suit cases, wheelchairs, etc. We refer to those objects as unknown, because usually it does not pay off to design a detector for them since there is an almost endless appearance variability due to quickly changing fashions. Through this high variety a learned generic detector will not yield the detection performance required for robust tracking. Other reasons why unknown objects were neglected so far in the context of tracking are on the one side the importance of human detection for traffic safety, since hitting a suitcase is not that critical as hitting a human and on the other side the low frequency of occurrence of unknown objects in a typical urban scenario. However, for a better understanding of the behavior of people, a robust tracking system must recognize and track other objects in pedestrian environments, since these objects strongly affect people’s motion.

In contrast to the existing approaches for mobile multi-object tracking in street scenes (Andriluka *et al.*, 2010; Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Ess *et al.*, 2009b; Kuo *et al.*, 2010; Leibe *et al.*, 2008a; Mitzel *et al.*, 2010; Wojek *et al.*, 2011), we propose to adopt a rigorous *tracking-before-detection* strategy. We make use of noisy stereo depth estimates to extract regions-of-interest (ROIs) in the input images and robustly segment them into candidate objects. Each such region is then tracked independently in 3D using a model-based iterative-closest-point (ICP) (Besl and McKay, 1992) tracker. For each tracked object, we then pass the corresponding image ROI to an object detector (Felzenszwalb *et al.*, 2010b) for classification into pedestrians and other objects. As a result, we can not only track a large number of pedestrians and unknown objects, but we additionally save computation, since the person classifier needs to be applied only once for an entire track.

At the core of our approach is a novel, compact 3D representation that allows us to integrate and refine both volumetric and surface information about tracked objects over time. This representation makes it possible to robustly track a large variety of

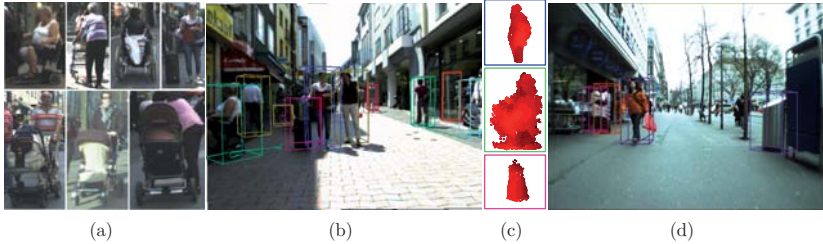


Figure 8.1.: In addition to pedestrians, real-world scenarios contain a large variety of other dynamic objects (a), for which there are usually no generic detectors available. We propose an approach that can reliably track those objects, together with the accompanying pedestrians, from a mobile platform (b). Our approach builds up 3D shape models for all tracked objects (c). By comparing those models to a statistical model of typical pedestrian shapes, it additionally allows us to create hypotheses about the presence of carried items (d, shown in red).

objects despite high levels of noise in the stereo input. As a side result, our approach builds up an integrated shape model of the tracked objects that can be used for further analysis. In a final step, we use the online-learned models in order to analyze the shape of tracked persons in more detail. By comparing their observed shapes with a learned statistical shape template of unencumbered pedestrians, our approach can generate plausible hypotheses about the presence of carried items on their bodies. To our knowledge, ours is the first approach to render such detailed shape analysis possible for a vision system on-board a mobile platform.

8.1. Related Work

The first challenge in any tracking system is to reliably segment objects of interest from their background. For mobile pedestrian tracking, this is usually done using the *tracking-by-detection* paradigm (Andriluka *et al.*, 2010; Ess *et al.*, 2009b; Kuo *et al.*, 2010; Leibe *et al.*, 2008a; Luber *et al.*, 2011; Wojek *et al.*, 2011), where the initial segmentation task is performed by an object detector and tracks are then extended by additional detections. In some approaches, the detector is only used for initializing new tracks, which are then propagated by low-level trackers (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Mitzel *et al.*, 2010). Common to all of those approaches is that only objects are tracked which have first been verified by the detector.

The contrary paradigm of *tracking-before-detection* was first used for tracking tasks in radar data (Prengaman *et al.*, 1982; Salmond and Birch, 2001), where measurements were probabilistically integrated over time before passing them to a classifier in order to

compensate for low signal-to-noise ratios. Similarly, tracking-before-detection has been used for driver assistance systems based on stereo input (e.g., Danescu *et al.* (2011)). The difficulty of obtaining robust and accurate stereo measurements has however often restricted such approaches to structured driving environments, where objects of interest are usually well separated from each other and their backgrounds, whereas detection-based approaches are usually used for pedestrian tracking (Geronimo *et al.*, 2010b). The development of highly accurate dense 3D laser range sensors (LIDAR) has made it possible to use tracking-before-detection approaches also for more challenging autonomous driving scenarios (Kaestner *et al.*, 2012; Petrovskaya and Thrun, 2009). However, the excessive cost of current LIDAR sensors (list prices 30k-75k US\$) restricts more widespread use. In contrast, the approach proposed in this chapter is able to operate in very challenging pedestrian zone scenarios, while being robust enough to work on cheap stereo vision input.

Stereo depth-based ROI extraction is often used in order to speed up pedestrian detection and reduce the number of false positive detections in mobile scenarios (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Gavrila and Munder, 2007; Geronimo *et al.*, 2010b). This is usually based on the assumption that objects of interest occur on a ground surface, and various road shape models have been proposed to remove the ground points (Geronimo *et al.*, 2010b; Keller *et al.*, 2009). Our ROI extraction approach is similar to the ones from (Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010), but it is especially optimized for robust region extraction in busy scenarios, as we describe in Chapter 5.

After extracting ROIs for candidate objects, we continue tracking them in 3D using ICP (Besl and McKay, 1992). Several approaches have been proposed in the past for ICP-based multi-person tracking (Feldman *et al.*, 2012; Mitzel and Leibe, 2011). Our approach differs from them in that it builds up a compact 3D model for each tracked object capturing both surface and volumetric properties. This allows our approach to integrate and refine the noisy stereo measurements over time, resulting in more robust tracking performance. Recently, methods have also been proposed for learning object categories from tracked LIDAR point clouds (Teichman and Thrun, 2011; Teichman *et al.*, 2011). Our proposed 3D representation makes such algorithms applicable to noisy stereo data, while simplifying the learning task by providing valuable integrated shape information in addition to appearance and trajectory data. The approach by Bjorkman and Kragic (2010) represents unknown object shapes (extracted with the help of fixation cues of a robotic stereo vision system) by 3D Gaussians. Our 3D representation preserves more detail about surface shape in addition to observed shape variances, which helps us cope with articulated objects.

For detecting carried objects in static surveillance camera footage, Damen and Hogg (2008) have proposed an approach that learns view-specific temporal templates of an unencumbered pedestrian's occupied area and compares this template to the segmented area of tracked objects obtained by background modeling. We take inspiration from their approach, but generalize it to the case of a moving camera, where segmentation and model construction is a much harder problem. Haritaoglu *et al.* (2000) exploits

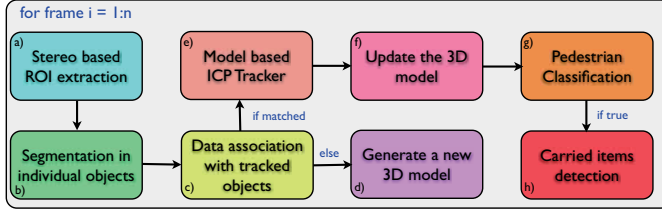


Figure 8.2.: Overview of the different steps of our approach.

the fact that the silhouette of humans is symmetric in order to detect carried items. From the detected foreground region the pixels are classified around the major axis into belonging to the symmetric part or not. Non-symmetric pixels are grouped together into regions which are labeled as carried item if they exceed a certain size. However, the approach works only if a person is carrying a single item. In case a person has a bag in each hand the approach will not detect any carried items, since the extracted silhouette will be symmetric.

The rest of the chapter is structured as follows. Sec. 8.2 gives an overview of our processing pipeline. Sec. 8.3 then introduces our compact 3D object representation, and Sec. 8.4 describes how it is used for tracking-before-detection. Sec. 8.5 explains our approach for carried item detection. Finally, Sec. 8.6 presents experimental results.

8.2. Overview

Fig. 8.2 shows a system-level overview of our proposed 3D tracking framework. Given a new stereo image pair and its corresponding depth map, we first generate ROIs and segment them into individual object areas ((a)-(b)), as described in Chapter 5. We then perform data association (c) with a nearest-neighbor filter in order to associate new candidate ROIs with existing objects based on ROI bounding box overlap in the ground projection. For each unmatched object, we build up a compact 3D model (d) from the 3D points within its ROI and start a new track. In the following frames, we then use this 3D model in order to precisely register the tracked object surface to the ROI's 3D point cloud using model-based ICP (e). After each successful alignment, we update the model with the new 3D points, building up an uncertainty estimate for each surface point (f). When the tracked object comes into detection range, we apply a pedestrian detector to its ROI in order to classify the tracked object as person or non-person (g). Note that this has to be done only once for each track. Finally, we perform detailed shape analysis for tracked persons close to the camera (h). By comparing their accumulated 3D models with a learned model of unencumbered pedestrian shapes, we create hypotheses for carried items (e.g., backpacks or shopping bags) on their bodies.

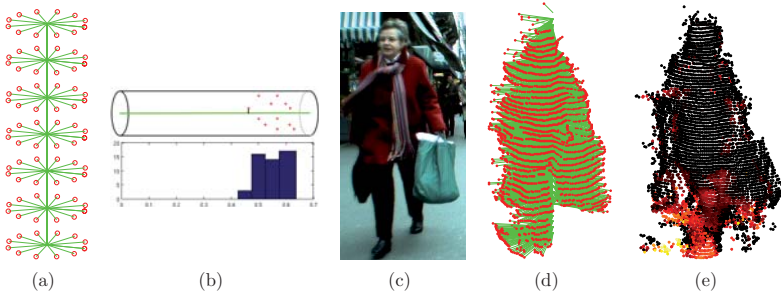


Figure 8.3.: We represent arbitrary 3D objects by a novel *Generalized Christmas Tree* (GCT) model, a cylindrical shape model composed of a vertical center axis and several layers of equally spaced horizontal rays (a). Along each ray, we accumulate observed 3D points in a distance histogram (b). This representation naturally adapts to the shape of tracked objects (c), allowing us to use median points (d) and variances (e) of each ray for robust model-based tracking. (For better visualization, only a subset of the height layers are shown).

8.3. 3D Object Representation

Our goal is to capture the approximate 3D shape of arbitrary objects in order to facilitate accurate 3D tracking and more detailed shape analysis. A main difficulty when trying to do this from stereo depth input is that the raw depth estimates are very noisy and depth resolution diminishes with distance. (For example, in our data, a pedestrian at 9m distance may cover only 1-2 disparity levels). To work with such data, we therefore need to integrate shape information over time to smooth out noise. Since our aim is to develop an approach that scales well to a large number of objects, on the other hand, it is important that the representation is compact, capturing the essence of a novel object shape in a form that is memory efficient and easy to update.

We propose a novel, compact shape representation that fulfills those goals, called a *Generalized Christmas Tree* (GCT). As visualized in Fig. 8.3, this model consists of a vertical center axis and several height layers of equally sampled horizontal rays emanating from this axis. Each ray stores the distance distribution of observed 3D surface points within a certain cylindrical cross-section. This way, the GCT captures both a robust estimate for the most likely object surface, represented by the median points for each ray, and the variance along each ray caused by noise and articulations.

To construct a GCT from an ROI, we cast radial rays from the center of the ROI over a fixed number of discrete height levels (see Fig. 8.3(a,d)). For each ray, we consider all 3D points that fall inside a cylindrical volume and select the closest point to the ray as their representative. This point is projected onto the ray and only the distance

from the center to the projected point is stored. When the model is updated in the following frames, additional points are associated to the ray. In order to allow tracking over long sequences with a fixed memory footprint, we store the distances in a histogram (Fig. 8.3(b)).

This is clearly an approximation, as not all object shapes can be accurately captured by a GCT. As only angular distances to the farthest surface are stored, some shape details, such as the space between a person’s arms and its body, are lost in the representation. Still, the approximation is reasonable for a large variety of objects encountered in street scenes. In addition, the representation is very compact, requiring only the storage of a fixed number of histogram bins for each ray, which is far less information than the alternative of storing a full point cloud for each frame in which the object is observed.

As the stereo depth image contains only information about the front surface of a tracked object, each GCT will typically only capture a partial shape. Still, by aligning the GCT with the object trajectory, we are able to grow and complete the object model whenever an object turns around. Thus, the GCT model can also capture volumetric information. When initializing a new GCT, a main uncertainty factor is the distance from the observed object surface to its center. Since the center location is not a-priori known for novel objects, we first make an initial guess, assuming an object radius of 50cm. Instead of directly building up a distance histogram for each ray, we first keep a list of the raw associated distances for several frames. When enough evidence has been accumulated, we update the representation by reconstructing a point cloud from the stored distances, re-estimating the object center to fit the point cloud dimensions, and resampling all reconstructed points into the rays to fit the shifted object center.

8.4. Stereo Depth-Based Tracking-Before-Detection

In the following, we present the detailed stages of our tracking pipeline. As first, we generate a set of segmented ROIs for potential objects in the scene from the stereo depth image using the *classical* ROI extraction procedure, described in Chapter 5.2. Before ROI extraction the 3D point clouds were filtered, based on our point cloud labeling approach described in Chapter 5.1, to contain only points labeled as *objects*.

Data Association. In order to associate ROIs with existing tracks, we match them to each track’s ROI from the previous frame. We assume ROIs to match if the intersection-over-union of their ground projection footprints is over 50%. We associate each track with the closest matching unassociated ROI and start new tracks for all ROIs that cannot be associated.

ICP Tracking and Model Update. After associating each track with a new ROI, we align the accumulated GCT object models with the ROIs’ 3D points in the new frame. For this registration step, we adapt the ICP algorithm (Besl and McKay, 1992). Briefly

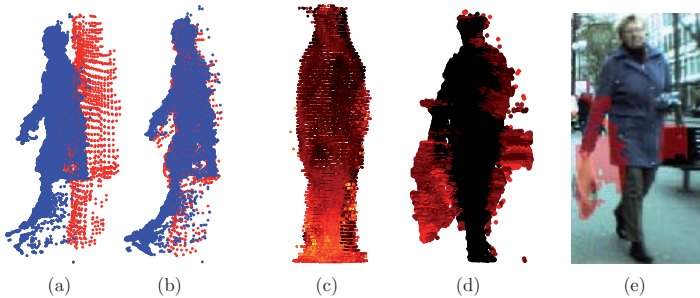


Figure 8.4.: (a) Visualization of the ICP registration procedure for aligning model points (red) of a tracked object to new 3D points from the current frame (blue). (b) Result of the ICP alignment. (c) Visualization of the learned pedestrian shape model. Points are drawn at the median distances; the colors represent the variances. (d) Result of comparing a tracked pedestrian model to the learned shape model. The brighter the color, the larger is the difference in point distributions. (e) Result after Graph-Cut segmentation for carried-item detection (overlayed in red).

stated, ICP iteratively computes the rotation $\mathbf{R}$ and translation $\mathbf{t}$ for aligning two point clouds $\mathcal{M} = \{\mathbf{m}_i\}_{i=1}^{|\mathcal{M}|}$ and $\mathcal{D} = \{\mathbf{d}_i\}_{i=1}^{|\mathcal{D}|}$ based on correspondences between closest points

$$\mathcal{E}(\mathbf{R}, \mathbf{t}) = \sum_i^{|\mathcal{M}|} \sum_j^{|\mathcal{D}|} w_{i,j} \|\mathbf{d}_i - (\mathbf{R}\mathbf{m}_j + \mathbf{t})\|^2 \quad (8.1)$$

where $w_{i,j}$ is 1 if $\mathbf{d}_i$ is closest point to $\mathbf{m}_j$ and 0 otherwise. We restrict $\mathbf{R}$ to 1D rotations around the ground plane normal and use the Euclidean distance as point-to-point distance measure. The individual steps of the general ICP algorithm are summarized in pseudo code in Algorithm 1, described in Chapter 4.5.

Fig. 8.4(a-b) shows the resulting registration procedure. For each GCT model, we first generate the 3D points at the position of the previous frame by taking the median of the distances accumulated so far, corresponding to $\mathbf{m}_j$ in Eq. 8.1 (red points in Fig. 8.4(a)). The blue points are the 3D points from the overlapping ROI, which correspond to $\mathbf{d}_i$ in Eq. 8.1. In order to robustly register articulated objects, we weight each GCT point by the standard deviation of its distance distribution, adapting $w_{i,j}$ accordingly. After performing the ICP registration, we obtain the translation and rotation for moving the GCT model to the new position in the current frame. Next, we update the registered GCT model with the new 3D points. For this, we again match 3D points to the model's rays, choose the closest point as representative for each matched ray, and add it to the stored distance distribution.

Object Classification. Whenever a newly generated track comes into detection range, its ROI is passed to a pedestrian detector for person/non-person classification. For this, we crop a small region around the back-projected segmented 3D points and only evaluate the detector in this region. As long as the object is then continuously tracked, no further classification is performed. In our experiments, this meant that our approach only needed to evaluate on average 0.7 small ROIs per frame with the detector (we used the DPM detector from Felzenszwalb *et al.* (2010b)). As our results in Sec. 9.6 will show, this procedure still achieves competitive pedestrian tracking performance.

Safe Tracklet Termination. The proposed approach generates long tracklets, but obviously cannot track through occlusions, as it depends on the registration of point clouds. In order to cope with occlusions and temporary segmentation failures, we therefore opt for a *safe termination* strategy (Kaucic *et al.*, 2005). When no ROI can be associated to a tracked object for t_{term} frames, the corresponding tracklet is terminated (we set $t_{term} = 3$). As soon as the object emerges from the occlusion again, a new tracklet will be generated, which can be connected to the same object by a higher-level tracker, similar as presented in Chapter 4. In order not to confuse the later evaluation, we do not use such an additional tracking stage in this Chapter, but only report the raw tracklet results.

8.5. Carried Item Detection

Through our tracking process, we accumulate and refine a 3D model of each tracked person, which we can use in order to analyze its shape and volume in more detail. In the following, we propose a procedure to detect carried items, such as backpacks and hand- or shopping bags, from this information. The basic idea behind our approach is to compare the online model to a learned statistical shape template of unencumbered pedestrians in order to detect deviations that cannot be explained by the variation in GCT volume during a gait cycle.

Pedestrian Model. In order to learn a statistical pedestrian shape model that matches the sensing characteristics of our noisy stereo input, we collected a training set of 12 GCT models of pedestrians moving in different directions over a duration of 15-20 frames from a separate training sequence. Half of the pedestrians were not carrying any bags or other items; the other half was carrying some item that we manually segmented out from the stereo depth images. We found this variability to be necessary in order to account for the different hand and arm positions people adopt when carrying items.

As the different training models capture different views of the tracked pedestrians, we used their trajectory directions in order to align the models to a consistent orientation. After this registration, we merged the distance distributions stored in the corresponding rays. The resulting full model is shown in Fig. 8.4(c). As can be expected, the variance of the distances in the bottom part is higher due to strong leg articulation.

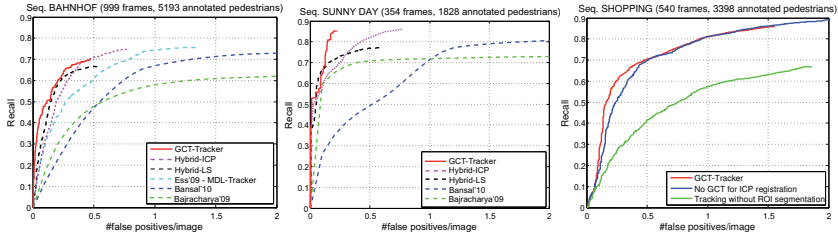


Figure 8.5.: Quantitative pedestrian tracking performance of our approach compared to our Hybrid-LS, Hybrid-ICP trackers and other baselines on the BAHNHOF and SUNNY DAY sequences from (Ess *et al.*, 2009b) and on our own sequence SHOPPING. The results show that the proposed GCT-Tracker outperforms our previous approaches as well as the current state-of-the-art.

Model Alignment. In order to compare distance distributions between the corresponding model and object rays, we first require an accurate model alignment. We therefore first register the tracked object GCT to the learned model GCT based on the observed walking direction. Before comparing the two models, however, an additional fine registration step is required. This is necessary, because the object center was only selected based on a rough guess when the track was initialized, potentially resulting in an assignment of surface points to different rays between the two models. We therefore first reconstruct a point cloud from the tracked object GCT and align this point cloud to the learned model by an additional ICP registration. We then shift the center of the object GCT and adapt the stored distance distributions accordingly. After this registration, the distance distributions are comparable.

We are interested in determining which parts of the tracked object surface cannot be explained by the natural statistical variations in body proportions, articulations, and shapes of pedestrians that are not carrying items. In order to do this, we compare the stored distance distributions of corresponding model rays using the Bhattacharyya distance $bhattach(P, Q) = 1 - \sqrt{\sum_i p_i q_i}$. Fig.8.4(d) shows a model of a tracked object with carried items after comparison to the learned model. As can be seen, the bags are clearly separated from the the body.

Carried Item Segmentation. The final carried item segmentation is based on Conditional Random Fields (CRFs), which have been extensively used for image segmentation. We use two labels $L = \{\text{carried item, no carried item}\}$ and the following energy function, defined over unary potentials $\psi_i(y_i)$ and pairwise potentials $\psi_{ij}(y_i, y_j)$:

$$\mathbf{E}(\mathbf{y}) = \sum_{i \in V} \psi_i(y_i) + \sum_{(i,j) \in E} \psi_{ij}(y_i, y_j) \quad (8.2)$$



Figure 8.6.: Carried item detection results (in red). The bottom row shows some failure cases which are caused by depth smoothing, excessive body proportions, and unusually wide clothing.

We define the unary potentials on the pixel level as $\psi_i(y_i) = -\log(p(y_i|r_i))$ where $p(y_i|r_i) = \text{bhatta}(r_{i,\text{object}}, r_{i,\text{model}})s(r_{i,\text{object}})$. Here, $\text{bhatta}(\cdot)$ is the Bhattacharyya distance between the distance histograms of the online tracked and learned model rays. $s(\cdot)$ is a sigmoid weighting function applied to the component of the ray distance $r_{i,\text{object}}$ that is orthogonal to the walking direction. The sigmoid function can be understood as a prior that carried items are usually not inside the leg area. Thus, the sigmoid keeps the weights of points that are further away from the object center and are likely to belong to the carried items, but suppresses the noise that comes from strong articulations in the leg area. The unary potentials are set for the pixel positions of the back-projected rays. For the pairwise potentials, we use a contrast-sensitive Potts model (Rother *et al.*, 2004) based on the image colors as $\psi_{ij}(y_i, y_j) = \theta_{i,j} \exp(-\beta \|x_i - x_j\|^2) \delta(y_i \neq y_j)$. We solve the CRF using standard Graph-Cuts with α -expansion (Boykov *et al.*, 2001).

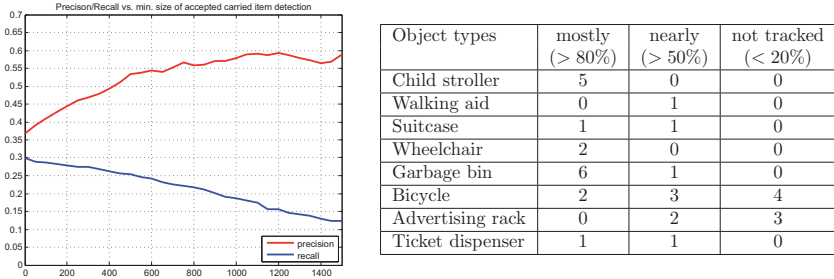


Figure 8.7.: (left) Per-pixel evaluation of the carried item segmentation accuracy. The carried items were annotated in every 4th frame of the BAHNHOF sequence, resulting in a total of 282 annotated items over 250 frames. (right) Performance evaluation for tracking of unknown objects.

8.6. Experimental Results

Datasets. To evaluate our approach, we report tracking performance on aforementioned popular datasets: BAHNHOF and SUNNY DAY from (Ess *et al.*, 2009b). Furthermore, we report quantitative and qualitative performance for several additional own sequences, which we captured in busy shopping streets using a similar child stroller setup. Our sequences contain more complex scenarios with many unknown objects, such as child strollers, wheelchairs, suitcases, and walking aids. We annotated one of those sequences (SHOPPING) in a similar fashion as the above, resulting in 3398 pedestrians annotations over 540 frames. For stereo range estimation we used the robust publicly available approach proposed by (Geiger *et al.*, 2010). Pedestrian classification was done using the DPM detector from (Felzenszwalb *et al.*, 2010b).

Pedestrian Tracking Performance. We first verify that our approach achieves competitive tracking performance for pedestrians. For this, we use the evaluation criteria presented by (Ess *et al.*, 2009b). In every frame we measure the intersection-over-union of tracked person bounding boxes and annotations. Detections with an overlap larger than 0.5 are accepted as correct. The results are presented in terms of recall vs. false positives per image (fppi). Figs. 8.5(a) and (b) present the achieved performance for BAHNHOF and SUNNY DAY. It can be seen that our approach achieves state-of-the-art performance on BAHNHOF and significantly outperforms previously published results on SUNNY DAY. Note that, in contrast to *tracking-by-detection* approaches where the detector needs to be executed for the entire image in each frame, we only evaluate the detector for 0.7 ROIs per frame on average.

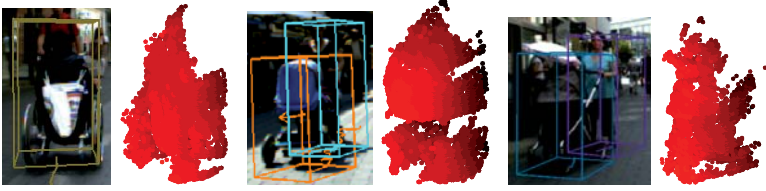


Figure 8.8.: Tracking results and accumulated GCT models for several unknown objects.

In addition, Fig. 8.5(c) shows the tracking performance on the SHOPPING sequence (red curve). As can be seen, even in such a complex and highly crowded scenario the achieved performance is comparable to the ones observed for the BAHNHOF and SUNNY DAY sequences. We additionally compare our approach to two baselines. The green curve in Fig. 8.5(c) illustrates the performance of our tracking system without the ROI segmentation and association step in each frame. Instead, we only perform an initial segmentation for each object in the first frame where it is visible and then sample 3D points for the ICP registration step in an uncertainty region around the projected position in all further frames (similar to our approach presented in Chapter 4.5). This works reasonably well for individual objects. However, it causes drifting of the tracks for pedestrians moving in a group. The blue curve in Fig. 8.5(c) shows the performance of our system when using standard ICP based on the original 3D points from the previous frame (instead of model-based ICP based on the accumulated GCTs) for the registration with the new points of the associated ROI. As can be seen, using the GCT model for tracking results in more robust performance, which can be explained by a more accurate object representation that compensates for noisy and fragmentary stereo data.

Fig 8.9 shows some qualitative pedestrian and unknown object tracking results on the SHOPPING sequence. It can be seen that our system is able to track most of the visible persons and objects correctly.

Object Tracking Performance. For assessing the performance of our approach for tracking unknown objects, we have processed 6060 frames of video material, which contained the unknown objects listed in Tab. 8.7. As can be seen, important dynamic objects such as child strollers and wheelchairs are correctly tracked most of the time. The performance for some other object types, such as stationed bicycles, is still lower due to the fact that the bicycles are often segmented into two or more areas. For each tracked object, we accumulate a GCT model capturing its detailed shape. Some of the resulting shape models are shown in Fig. 8.8, and additional tracking results are shown in Fig. 8.10. As can be seen, our approach succeeds in capturing the shape essentials, providing sufficient detail to support future recognition approaches.

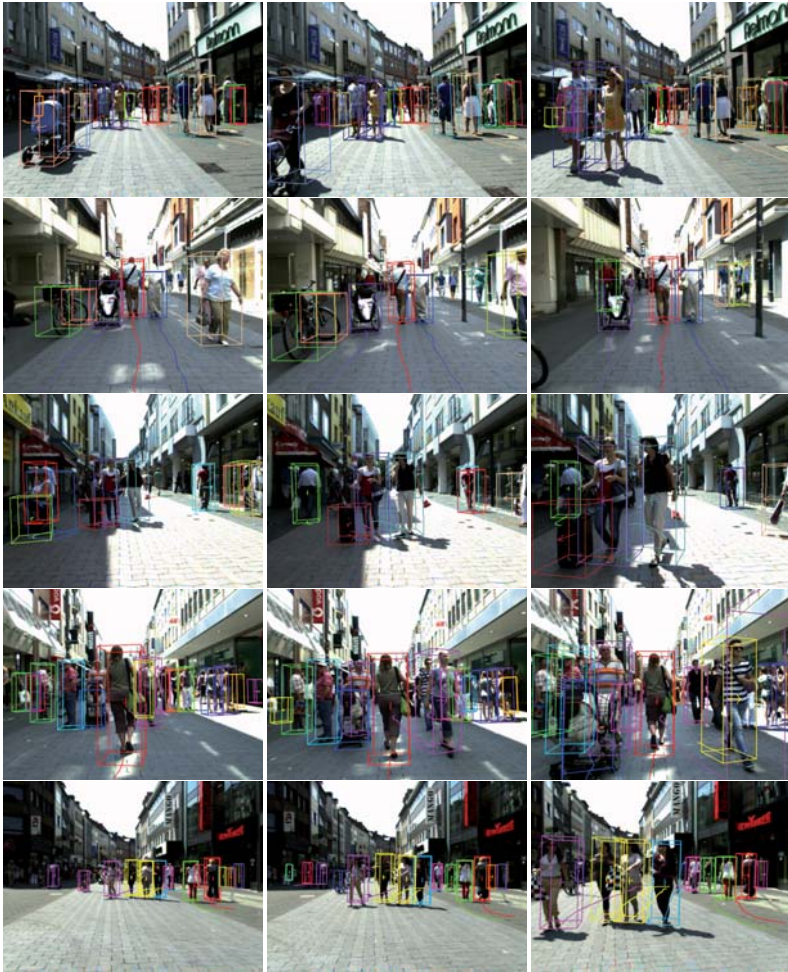


Figure 8.9.: Example tracking results on the test sequence SHOPPING, showing that our framework is able to track all visible objects correctly in a very busy environment with many occlusions, keeping the correct object ID.

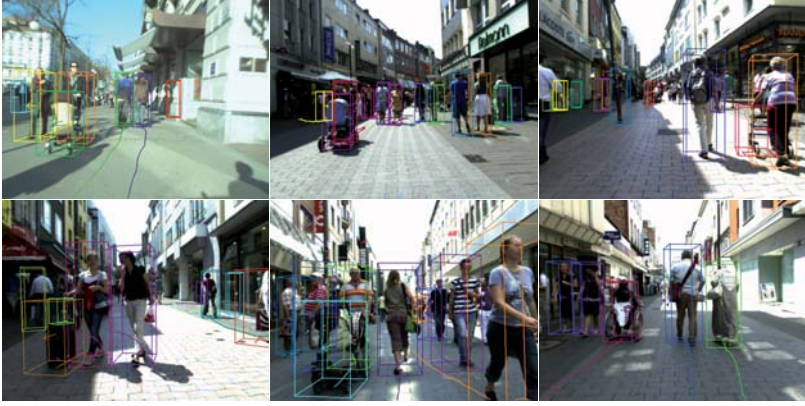


Figure 8.10.: Results for tracking of pedestrians and unknown objects.

Carried Item Detection Performance. For evaluating the performance of the carried item detection we labeled all carried items (in total 282) in every fourth frame of the BAHNHOF sequence. We compared our hypothesized segmentation results pixel-wise with the labeled data, resulting in the precision and recall plots in Fig. 8.7(left). Removing small noisy detections results in higher precision at the cost of some recall. Overall, our approach achieves a carried item segmentation precision of about 60%. Given the very noisy nature of our stereo input data and the small size of the items in question, we think this is a very encouraging result. Some example results are shown in Fig. 8.6.

8.7. Discussion

In this chapter, we have presented our 3D tracking approach for tracking both known and unknown objects in busy scenes based on stereo range data. The core of our approach is a novel 3D representation for objects which is built online by accumulating object surface information over the tracking process based on ICP registration. Apart from achieving state-of-the-art tracking performance, the adaptive 3D model representation allows us also to analyze the shape of tracked persons in more detail. By comparing the observed shape with a learned probabilistic shape template of pedestrians, we can hypothesize the parts of the shape that are likely to be carried items. Additionally, we can not only track a large number of pedestrians and unknown objects over time, but we save computation, since the person classifier needs to be applied only once for each object track.

The experiments corroborate our claim that the proposed *tracking-before-detection* strategy is well suited for the problem of tracking known and unknown object types. Even in such complex inner city scenarios the objects are correctly separated from each other and accumulated in long and robust tracklets. Further extensions that we will present in the next chapter allow a more robust data association, yielding longer trajectories and better 3D object representations avoiding the *safe termination* and resulting fragmentation of the tracks.

The carried item detection needs further investigation, since it suffers strongly from the misalignment to the reference direction based on the noisy tracking direction and the accumulated error of the ICP registration. However, we expect that it can already be improved by using a more precise statistical shape template of unencumbered pedestrian, generated, e.g., from synthetic data. This will at least avoid the registration errors accumulated during the training process and thus yield an exactly merged model representation.

Person-Person and Person-Object Interaction

Despite the progress we made with tracking all scene participants presented in the previous chapter, we are still treating the pedestrians as independently moving entities, ignoring the fact that humans are not moving independently, but they closely interact with their environment, such as other persons or scene objects. In the context of tracking, people behave differently when they are walking in groups trying to keep the same speed and walking direction or when pushing a child-stroller performing non-holonomic motion due to mechanical constraints of the stroller. For that reason, being able to estimate the interaction information has several advantages. It can improve prediction for trajectory continuation based on the one participant of the interaction in case of detection/tracking failures for another participant or adaptation of dynamic models for certain object/person constellations, such as the aforementioned example with a child-stroller. Thus, in this chapter, we present an approach that, given tracks from persons and other scene objects as presented in Chapter 8, jointly infers both the object class and the interaction type from observed appearances and dynamics. The core component of our approach is a probabilistic graphical model that relates object appearance and spatial arrangement consistently over time. This model can determine which persons and objects belong together and in what way they interact. Based on the recognized interaction, it can then predict how the interaction will most likely continue and how one object's trajectory will be affected by another object's observed motion.

Realizing such an approach for a mobile platform cannot be done in a standard *tracking-by-detection* framework based on pre-trained object detectors (Andriluka *et al.*, 2010; Bajracharya *et al.*, 2009a; Bansal *et al.*, 2010; Ess *et al.*, 2009b; Leibe *et al.*, 2008a; Wojek *et al.*, 2011), since object class inference will only become possible after an object configuration has already been tracked for several frames. We therefore build up this approach on the basis of our *tracking-before-detection* approach from Chapter 8, based on low-level stereo region segmentation and multi-hypothesis data association.

The benefit of our approach is that it enables us to track a large variability of objects with potentially unknown appearance, while achieving increased robustness to classifi-

Figure 9.1.: Our proposed approach models pairwise interactions between persons and objects in a probabilistic graphical model, taking into account object shape, relative arrangement, and temporal consistency. Thus, it can infer which objects belong to which persons and predict how the interactions will continue. Recognized interactions are visualized by colored lines linking the foot points of interacting objects (Legend: pull side right, pull side left, pull right, pull left, push, group).

cation failures. For an example, consider the scene shown in Fig. 9.1. Our approach fails to recognize the child in the bottom right corner of the left figure as a *person* (visualized by a cylinder). In a *tracking-by-detection* approach, this would cause a tracking failure. Instead, our approach treats the child as an *unknown moving object* (visualized by a box) and it can still recognize that this object forms a group with the child’s mother (shown by the green connecting line), thus affecting the mother’s trajectory.

In detail, we make the following contributions:

- We propose a probabilistic graphical model for recognizing pairwise person-object interactions taking into account object shape, relative arrangement, and temporal consistency. This model can jointly infer object classes and interaction patterns more robustly than could be done from individual observations. In particular, it can resolve which object belongs to which person, arriving at improved scene understanding.
- This scene interpretation allows our approach to make improved predictions for the continuation of each tracked object’s trajectory with increased robustness to occlusions and detection failures.
- In order to make this approach feasible on noisy stereo depth data, we propose several detailed extensions to our approach presented in Chapter 8, spanning the entire tracking pipeline. This includes novel methods for improved region candidate extraction, data association, and multi-hypothesis discrimination.
- We introduce a novel benchmark dataset for person-object interaction consisting of 325 video sequences with a total of almost 15,000 frames and use it to quantitatively evaluate our approach’s performance.

The chapter is structured as follows. The following section discusses related work. After that, Sec. 9.2 presents the proposed graphical model for object and interaction

classification. Sec. 9.3 discusses how model parameters are learned, and Sec. 9.4 shows how the model is used for inference and prediction. Sec. 9.5 integrates the model into a tracking pipeline for robust scene interpretation. Finally, Sec. 9.6 presents experimental results.

9.1. Related Work

Incorporating social walking models into modeling the dynamics of individual pedestrians (Pellegrini *et al.*, 2009; Yamaguchi *et al.*, 2011) and groups (Amer and Todorovic, 2011; Amer *et al.*, 2012; Choi and Savarese, 2012; Lau *et al.*, 2009; Pellegrini *et al.*, 2010) has been shown to yield significant improvement for tracking in crowded scenes. Choi *et al.* (2009, 2011b) proposed a spatial feature that captures the distribution of persons around each anchor person including the position and the orientation. From this distribution a possible collective activity of a certain scene can be derived. In the follow-up publication Choi and Savarese (2012) have shown that tracking results can be improved by simultaneously tracking multiple people and estimating their collective activities. However, those approaches consider only other pedestrians as possible scene objects and ignore the impact of large variety of other objects such as bicycles, child strollers, shopping carts, or wheelchairs often present in street scenes. A main reason for this is the lack of reliable classifiers spanning the large variety of scene object classes.

There are several approaches that model person-object interactions in static surveillance camera footage using background modeling. For example, Smith *et al.* (2006) propose to detect abandoned luggage items by analyzing the size and velocity of tracked foreground blobs. Damen and Hogg (2008) propose a more elaborate approach for carried item detection that compares the segmented object area to learned temporal templates of pedestrian shapes. Such approaches are limited by the requirement of background modeling, which makes them not applicable for our scenarios with a moving camera.

9.2. Modeling Person-Object Interactions¹

We model all person-object interactions in the scene in a pairwise manner. This has two important implications: On the one hand, we assume each observable interaction to have exactly two actors. On the other hand, our model becomes easy to handle and learn, and inference can be performed in an efficient way. We try to robustly explain what is happening in the scene under the basic assumption that persons' actions will be the dominant cause of observable object motion, meaning that a non-autonomous object can only move because of the action of a person interacting with it. Having analyzed

¹The graphical model was implemented and evaluated by Tobias Baumgartner in the context of his master thesis, supervised by D. Mitzel and B. Leibe. The achieved results were published in our joint publication (Baumgartner *et al.*, 2013) and will be presented in the following section.

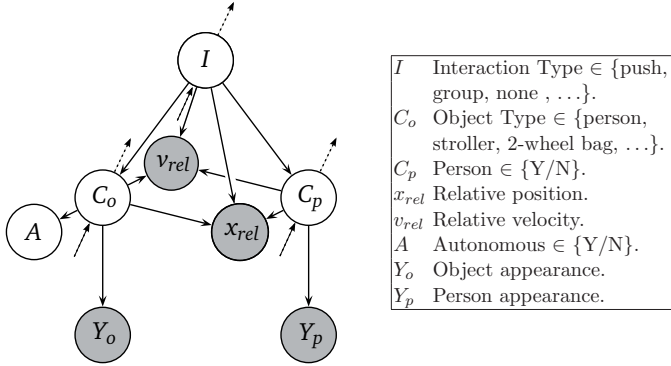


Figure 9.2.: (left) Bayesian Network for object person interaction, dashed lines indicate inference from preceding and to subsequent frames. (right) table of variables in Bayesian Network.

a scene and interpreted all interactions, our model can then use this information in a generative way in order to predict future motion and support tracking.

Looking at a scene of various given objects, their past trajectories and current positions, we derive a number of individual and pairwise features to infer the type of interaction. Firstly, we model the appearance of objects and persons and try to assign them to one of the following classes: *stroller*, *2-wheel bag*, *4-wheel bag*, *walking aid*, *person*, *autonomous* (covering objects that do not require an external person to move, such as electric wheelchairs), and *noise*. For each person-object and person-person pair, we can determine their relative positions in the scene, as well as their relative velocities derived from their trajectories. Together with the object appearances, we use those as features in order to infer the interaction type. In this approach, we consider 6 different interaction classes, as shown in Fig. 9.1(right), plus the additional class *none*. In our setting of pairwise interaction, the action *group* is defined—and later on evaluated—as true if and only if two persons belong to the same group of people. An intuitive notion of group transitivity will then allow us to robustly identify all persons belonging to the same group.

We always consider interactions between a dominant actor (a person) and an object being acted upon (which can also be a second person in the case of groups). Since we do not know the object class a priori, we also look at pairs where the *active* entity is a non-person. All such interactions shall be detected as *none*. Hence, for each action we determine a probability for the actor to be a person. In a similar fashion, we model whether or not an object in an interaction acts in an *autonomous* way (as another pedestrian or a wheelchair would do).

Fig. 9.2 illustrates our proposed model and an overview of the used random variables. Using this model, the likelihood of an observed interaction can be decomposed as:

$$\begin{aligned} p(I, C_o, C_p, v_{rel}, x_{rel}, A, Y_o, Y_p) = \\ p(I) \cdot p(C_o|I) \cdot p(C_p|I) \cdot p(x_{rel}|I, C_o, C_p) \cdot \\ p(v_{rel}|I, C_o, C_p) \cdot p(Y_o|C_o) \cdot p(Y_p|C_p) \cdot p(A|C_o) \end{aligned}$$

Except for $p(Y_*|C_*)$ all of these factors are multinomial distributions learned from the training data, as described in Sec. 9.6. The likelihood for a certain observed appearance will be computed using two different classifiers.

At runtime we will then observe the appearances of our actors Y_o and Y_p , as well as their relative positions and velocities, x_{rel} and v_{rel} , respectively (cf. colored nodes in Fig. 9.2). To infer an interaction between these two, as well as the object type and person classification, we perform exact Belief Propagation using the junction tree algorithm (Pearl, 1986).

The object-type classifier assumes a correct tracking and the input of a 3D point cloud that only contains points belonging to the person to be classified. Later in Sec. 9.5, we show how to construct these stable inputs from noisy data.

9.3. Learning

Relative Position and Velocity. We define all relative positions in a log-polar coordinate system. Fig. 9.3 shows the learned relative positions in our model for 10 bins for each angle and log distances. The intuition for these grids is that a person is located in the center of the *spider web* facing downwards. An object on her left will hence be represented by the bin to the right of the middle point. Unsurprisingly, these probability distributions correctly reflect spatial arrangements. For example, one would always expect a stroller that is pushed by a person to be located in front of her (cf. Fig. 9.3(top left)).

Object Classifiers. We use and evaluate two different methods for object classification. The first one is based on a simple height measure for an object. From our training data we learn a multinomial height distribution for the different object classes and use this to predict the class given the observed height. Since we always assume noisy 3D data, the height is smoothed over subsequent frames before classification.

For the second classifier, we use a more complex object shape model based on our volumetric GCT representation, presented in Chapter 8. We determine a volume distribution for each learned object, as described in Sec. 9.5, and classify by computing per-class posteriors based on the observed volumes. Fig. 9.4 shows the learned models for person, stroller, 2- and 4-wheel bag.

Given the volume histogram $\mathbf{x}$ for a GCT (e.g. Fig. 9.6), we evaluate the class posterior $p(C_j|\mathbf{x})$ for class C_j . We assume uniform priors $p(C_k)$. Also, we make a naive

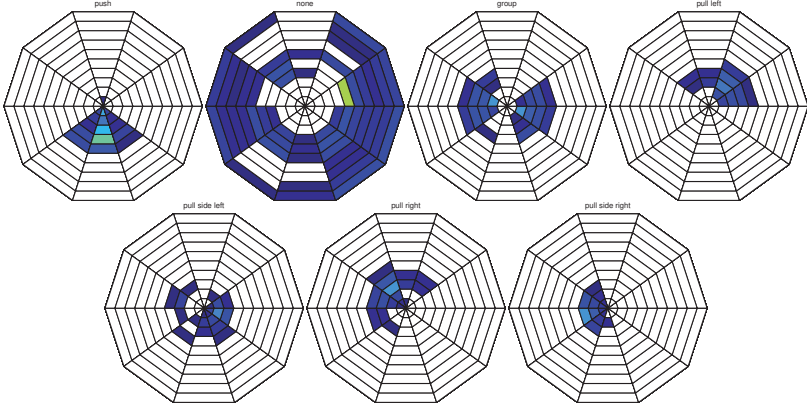


Figure 9.3.: Learned conditional distributions for relative positions in a log-polar grid.

Bayes assumption and regard the volume distribution in the different height bins as independent, which leaves us with:

$$p(C_j|\mathbf{x}) = \frac{p(\mathbf{x}|C_j) \cdot p(C_j)}{\sum_k p(\mathbf{x}|C_k) \cdot p(C_k)} = \frac{\prod_i p(x_i|C_j)}{\sum_k \prod_i p(x_i|C_k)} \quad (9.1)$$

9.4. Inference and Prediction

Inference. In a scene with n entities (persons/dynamic or static objects) there are $n \cdot (n-1)$ pairwise interactions. Despite the complex nature of predicting all interactions in a scene, exact inference is feasible for our model due to its constraining setup. Using a simply pairwise model does not guarantee *scene consistency* though. This means that an object o might for example be detected to interact with two persons $p_{1/2}$ in a scene, being interpreted as a stroller in the first case and as a suitcase in the second. It cannot be both at the same time. Our approach to incorporating evidence from other interactions in the same scene is to marginalize object types over all pairwise assignments and thus interconnecting all C_o and C_p that belong to the same entity.

Another clue we use for our prediction is evidence from past frames. The rationale is that an object that has been detected as a person in one frame is likely to be a person again in the next one (likely, but not certain due to tracking uncertainties). Again we set priors on the corresponding distributions from one frame to another (cf. dashed lines in Fig. 9.2).

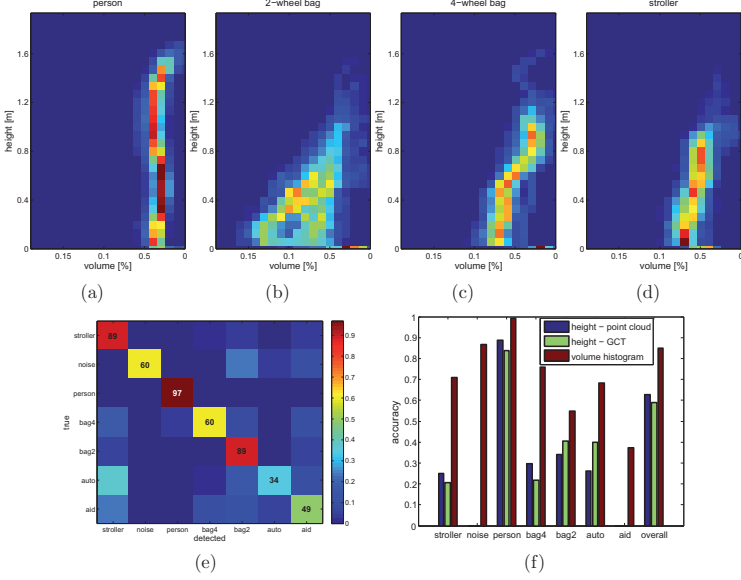


Figure 9.4.: (a-d) Learned GCT histogram classifiers for person, 2/4-wheel bag and stroller. (e,f) performance of classifier.

All in all, $p(C_o|I)$ for entity a interacting with entity b at time t in iteration i evolves to:

$$p(C_{o,t,i}^{a,b} | I_{ab}, C_{t,i-1}^a, C_{t-1,i}^a) = p(C_{o,t,i}^{a,b} | I_{ab}) \cdot p(C_{t,i-1}^a) \cdot p(C_{t-1,i}^a)$$

$$p(C_{t,i-1}^a) = \prod_{j=1; j \neq a} p(C_{o,t,i-1}^{a,j} | I_{aj}) \cdot (C_{p,t,i-1}^{a,j} | I_{aj})$$

$p(C_{t-1}^a)$ is computed analogously. Here a is the unique index of one entity in the scene and we initialize with a uniform distributions for $p(C_0^a)$. We include priors over interactions I_j of a with all other entities j from the last iteration $i - 1$, as well as the last frame $t - 1$. The maximum number of iterations is indexed by i_M . In different interactions, a can occur either as object or person, meaning we compute its C_o or C_p respectively. The other way round, we can then also use both of those random variables to produce a more stable prior in the next iteration or frame for predicting a .

Prediction. Having acquired a certain level of semantic scene understanding, we can now use our Bayesian network to also support other tasks. For example, tracking can be facilitated in a setting where objects are occluded or lost. Knowing that a person

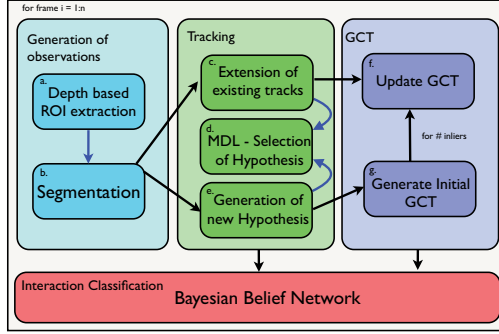


Figure 9.5.: Overview of observation generation for the proposed Bayesian Network.

pushed a stroller s in the past frames raises the suspicion he will do so again in the current frame. Suppose we lost track of this stroller. We can plug this information into our model and infer a probability distribution of the expected location of the lost object. Furthermore, we can infer the relative position of s to all other entities $j \in J$ for the set of all entities J that it interacted with in the past frame. The more interactions were observed before, the more certain we can be when inferring the new position x_s :

$$p(x_s|J) \sim \prod_{j \in J} \mathcal{L} [x_j; p(x_{rel(s,j)}|I_{sj}, C_p^j, C_o^s)], \quad (9.2)$$

where $\mathcal{L}[\mathbf{x}; \mathbf{p}]$ is the probability distribution of positions according to $\mathbf{p}$ (i.e. a log polar grid as in Fig. 9.3) around the center point $\mathbf{x}$.

9.5. Robust 3D Data Association and Tracking

Overview. Fig. 9.5 shows an overview of our tracking system that we use for generating observations (the positions, velocities and 3D object shapes) for the proposed graphical model. Given a stereo pair of images and depth maps, we first classify the 3D points into **object**, **ground plane** and **fixed object**, as described in Chapter 5.1. From the points classified as **objects** we generate *accumulated* ROIs, as presented in Chapter 5.3 (a) and segment them into individual object areas (b), more details are given in Chapter 5. The center of mass projected onto the plane of the individual object and the 3D points embedded by the segmented area form the input for our multi-hypothesis tracker. In each frame, the newly extracted objects are linked to trajectory hypotheses on the ground plane by starting new trajectories backwards in time (e) and extending already existing tracks with new observation (c). In order to capture the approximate shape of 3D objects

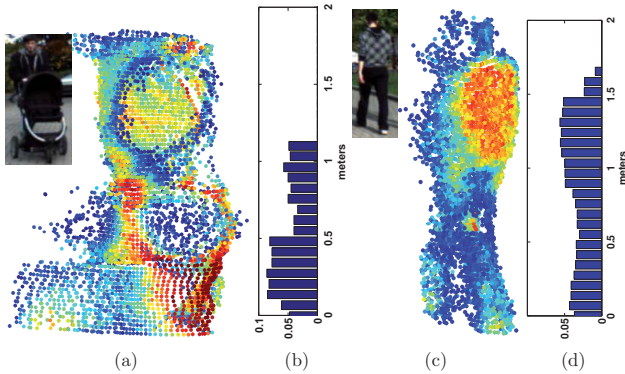


Figure 9.6.: Visualization of the accumulated GCTs for a stroller (a) and a human (c) and the corresponding volumetric features (b,d). The color of GCT points corresponds to the significance of the ray represented by the number of accumulated distances.

which we use for object classification we employ the GCT representation presented in the previous chapter. For each newly generated hypothesis from the tracker, we generate a GCT starting from the first inlier and update it by propagating the GCT sequentially over all inliers of the hypothesis (f,g). In case of extending an existing trajectory(f), the GCT is updated by registering it to the point cloud of the new observation and accumulating the new distance information. With the process so far we obtain an over-complete set of trajectory hypotheses which we prune to a final set as described in Chapter. 3.4. Finally, positions, velocities and features based on GCTs are passed to the graphical model for classifying person object interaction.

GCT Features. From the GCTs we generate for each trajectory hypothesis a volumetric feature (Fig. 9.6) which we use in the proposed model in order to classify the objects in different classes. Thus, for each valid trajectory we compute a volumetric histogram over height as follows:

$$|V_i| = \sum_{r_j \in V_i: \text{support}(r_j) > \theta} \text{med}(r_j), \quad (9.3)$$

where V_i is the bin, $\text{med}(r_j)$ is the median distance of the ray r_j and $\text{support}(r_j) > \theta$ means that we consider only rays that have accumulated at least θ distances already, where θ is interlinked to the lifetime of the GCT. By using the support function we reject rays that were originated from noisy outliers.

Measuring Overlap. In addition, we exploit GCTs in the model selection procedure, where we model the interaction between the trajectories by considering the intersection

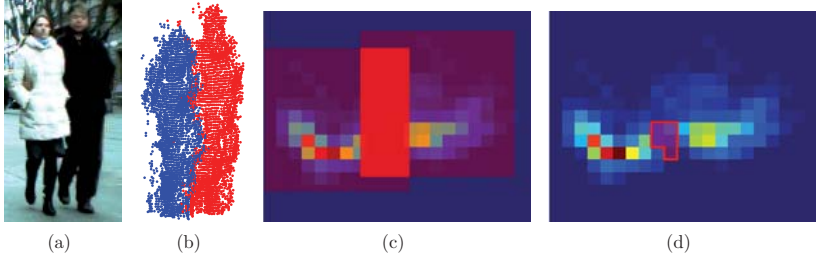


Figure 9.7.: Visualization of the proposed tracks interaction: (a) image cutout, two persons walking close together, (b) visualization of corresponding GCTs, (c) ground projections of GCT rays and the intersection between GCTs foot prints (d) standard approach for interaction computation assuming a fixed size footprint.

between the footprints of individual tracks. A common assumption, used in *tracking-by-detection* approaches e.g. (Ess *et al.*, 2009b), is that two objects cannot occupy the same spot in 3D space at the same time. Modeling object footprints by a fixed rectangular (or circular) shape leads to high interaction costs for close-by objects due to high overlap, as shown in Fig. 9.7(c), which can cause the rejection of one of the trajectory hypotheses. Instead, we propose an adaptive approach to compute the intersection of two objects based on their GCTs. For that, the reconstructed points of GCTs of both objects are projected to the ground plane forming a 2D histogram, Fig. 9.7(d). The projected ray points are weighted by the number of distances of the corresponding ray and thus represent the significance of a ray and the ground projection bin. As shown in Fig. 9.7(d), the bin intersection between the objects is significantly smaller than in the standard approach and using the weighting results in a very low intersection value. The final intersection score is obtained by computing the Bhattacharyya distance between the two normalized histograms. This extension makes tracking more robust in our scenarios, since objects such as strollers or suitcases are usually situated close to a person.

Tracker. As our tracking core, we employ an extended version (cf. Chapter 3.5) of the robust multi-hypothesis tracking framework presented in Chapter 3.4. As input, the tracker obtains the camera position from SfM, the ground plane and the segmented ROIs, instead of standard detection bounding boxes. From the 3D points of the segmented regions, we generate the footpoint position of the object by simply taking the center of mass of the point cloud and projecting it onto the ground plane. Furthermore, the 3D points are back-projected to the image in order to obtain a color histogram for each object required for the trajectory hypothesis generation process in order to associate the detections. The footpoint positions of the objects are linked to an over-complete set of trajectory hypotheses in a two-fold trajectory generation process and are then pruned

Action	train		test	
	# Seq.	# Fra.	# Seq.	# Fra.
push	68	3496	47	1456
group	48	2485	62	2394
pull right	9	408	27	1535
pull side right	10	563	6	297
pull left	11	516	27	1329
pull side left	7	417	3	119
sum	153	7885	172	7130

Table 9.1.: Statistics on number of actions on training and test sets.

to a final hypotheses set that is most consistent with the scene using model selection. For more details we refer to Chapter 3.4.

9.6. Experimental Results

Datasets. In order to train and test the proposed graphical model, we captured a dataset ACTIONS with a Bumblebee2 stereo camera containing 325 sequences with over 15015 frames. For training, we manually segmented the ROI areas of individual objects and generated tracks (including the GCTs) by employing the proposed tracker. For each tracked object, we annotated an action and a reference object it is interacting with. The training dataset was captured in a controlled environment from a static setup in order to simplify the annotation process. For the test dataset, we acquired the images in crowded and challenging shopping streets from a moving platform with different object appearances and dynamics. In Tab. 9.1 we present detailed numbers of actions on training and test sets. In total, we have annotated 153 sequences (7885 frames) and used 172 sequences (7130 frames) in order to assess the performance of our model. For the stereo estimation we used the robust approach from (Geiger *et al.*, 2010).

Tracking Performance. The person object interaction classification strongly depends on the output of the tracker since it requires positions, velocities and GCTs of the individual objects. For that reason, we first verify that our tracking approach is sufficiently robust and allows tracking in complex mobile scenarios. To this end, we experimentally evaluated our approach on three sequences, BAHNHOF, SUNNY DAY and our own sequence SHOPPING, which we captured in busy shopping street in Aachen using a similar child stroller setup as for BAHNHOF and SUNNY DAY sequences. We apply the evaluation criteria from (Ess *et al.*, 2009b) where the tracked bounding boxes are compared to manually annotated bounding boxes in each frame. Since our approach tracks all objects in the scene, but in this dataset only the pedestrians are annotated, we classify each segmented ROI using the pedestrian classifier from (Felzenszwalb *et al.*, 2010b) before

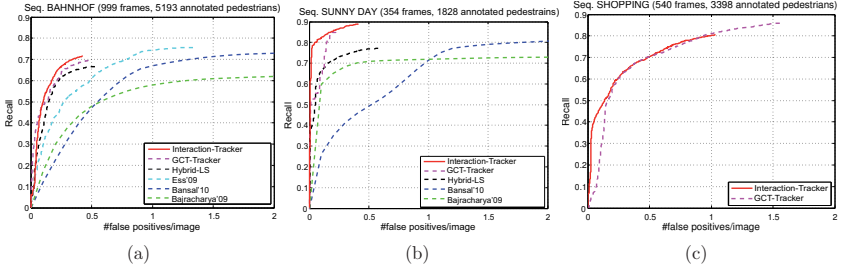


Figure 9.8.: Pedestrian tracking performance on (a) BAHNHOF, (b) SUNNY DAY and (c) SHOPPING.

passing it to the tracker. Fig. 9.8 presents the performance curves in terms of recall vs. false positives per image. As can be seen, our approach surpasses state-of-the-art performance, including all our previous approaches.

Furthermore, in Fig. 9.10 (a,b) we show the overall results of the interaction detection system, in Fig. 9.10(a) on the annotated segmentations and in Fig. 9.10(b) on tracking results on data from our new data set. Observe that apart from the tracking method used (annotated vs. automated) nothing changes between those experiments. The just slightly worse results in mAP for the experiments on the entire system confirms, that the tracking performance for our new data set is also just a little worse than the annotations. We can conclude that the tracking component is very robust and generalizes also to the new data set.

In Fig. 9.9 we present exemplary *tracking-before-detection* results for the SHOPPING, BAHNHOF and SUNNY DAY sequences. Those are pure tracking result without application of a pedestrian detector. As can be seen, the tracker is able to track most of the visible objects correctly in a very busy environment with many occlusions. Furthermore, the results corroborate that the preprocessing steps such as point cloud classification and ROI segmentation, perform remarkable well, since no tracks are initiated on buildings or trees and all objects are tracked separately.

Interaction Classification. We evaluate our action detection framework on our annotated training data, as well as on real life scenes described above. In order to assess the difficulty of the classification task, we first evaluate several simple baseline classifiers. These simple baselines follow two easy rules. First, if two objects are close together, they must interact in some way. Secondly, if both of these objects are persons then we just detected a *group*, else the baseline(*action*) detects *action*. In all other cases there is no interaction at all. Algorithm 3 summarizes this compactly.

Furthermore, we compare the final action detector with a detector based on a classifier that only takes into account the height of a tracked object, as described in Sec. 9.2.



Figure 9.9.: Result images showing pure *tracking-before-detection* results. (First row) SUNNY DAY sequence. (Second row) BAHNHOF sequence. (Third row) SHOPPING sequence. (Fourth row) JELMOLI sequence. Note that no pedestrian classifier was applied here. The results further corroborate that the point cloud classifier presented in Chapter 5 classifies the points from elevated structures correctly as no tracks are extracted on the buildings.

Also, we try our detector without any classifier, i.e., assuming a uniform distribution over object classes. The results for the crossvalidation on the training data are shown in Fig. 9.10(a). At the same time, we compare the system without inference between subsequent frames to the integrated approach (cf. dashed vs. solid lines). We only show

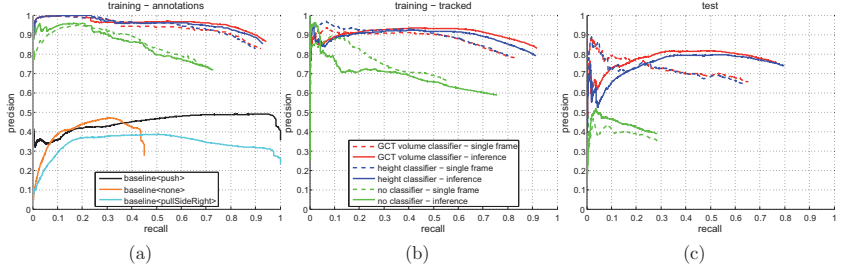


Figure 9.10.: Interaction classification, full pairwise evaluation. (a) using manual point cloud segmentation annotations, (b) using tracked point cloud data. (c) on dynamic scenes acquired in an inner city.

Algorithm 3 Baseline algorithm for given *action*.

```

1: function BASELINE(action)( $x_1, x_2$ )
2:   for  $x_1, x_2 \in \text{Obj}$  do
3:     if  $d(x_1, x_2) > \text{thresh}$  then
4:       return none
5:     else
6:       if  $x_1 = \text{person} \wedge x_2 = \text{person}$  then
7:         return group
8:       else if  $x_1 \neq \text{person} \wedge x_2 \neq \text{person}$  then
9:         return none
10:      else
11:        return action
12:      end if
13:    end if
14:  end for
    
```

three baselines (*push*, *none*, and *pull side right*), since these dominate the other baselines. Clearly, the performance of our approach is above the other presented approaches. We reach a *mean average precision* (mAP) of 0.907 vs. an mAP of 0.893 for the runner-up, the full system using a height classifier. In general, the timely inference is better than performing inference in each frame separately: Single-frame mAP for our system is 0.869 (cf. Fig. 9.10(a)). Just for the system without classifier (cf. Fig. 9.10(a)) we get a better performance if we do not take into account evidence from past frames. The reason here is that we would only propagate mainly false detections and have a better chance of detecting an interaction correctly if we take no priors into account.

Next, we perform the same experiment on our training data, but this time with actual results from our tracking pipeline instead of tracking results based on annotated object

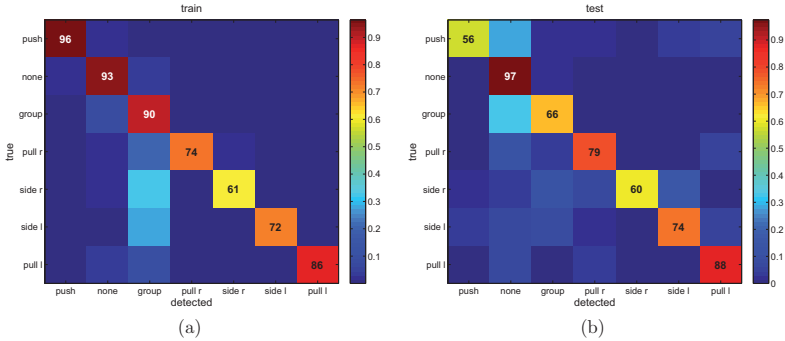


Figure 9.11.: Confusion matrices of our action detection for training (a) and testing (b).

segmentations. This is shown in Fig. 9.10(b). Because of the competitive performance of our tracking system, we do not lose much against the results in our experiments before. The mAP reduces from 0.907 to still 0.838 for our detector.

Finally, we evaluate the performance of our action detector for the test set of challenging scenes with a dynamic camera. We reach an mAP of 0.624 with the full combination of our tracker, object classifier based on GCTs, interaction model and frame inference (cf. Fig. 9.10(c)).

Taking a deeper look into the failures of our action detector (cf. Fig. 9.11) reveals that we perform consistently well on the action *none*, which means we have just few false positives. Transitioning from training to test data, we lose most accuracy in the actions *group* and *push*. All other action detection accuracies stay high.

Object Classification. In Fig. 9.4(e,f) we show the classification performance of our new classifier in comparison with a simpler height-based classifier. For the simple height classifier, we used the distance to the plane from the highest point from the segmented ROI. We also compare to a third classifier based on accumulated height information from the GCTs. As can be seen our proposed classifier performs significantly better for all object types. Both height-based classifier yield very similar performance.

Activity Prediction. As mentioned in Sec. 9.4, we can use our model to also perform a predictive task. In our evaluation we compare this prediction against a linear extrapolation by a *Kalman filter*. We measure success in this test as the closest prediction to the actual path. When we lose track of an object, the Kalman filter will predict future positions based on its underlying motion model. Our inference-based prediction observes the positions of all other entities in the scene and uses the interaction distribution it learned so far to infer the most likely position of the lost object.

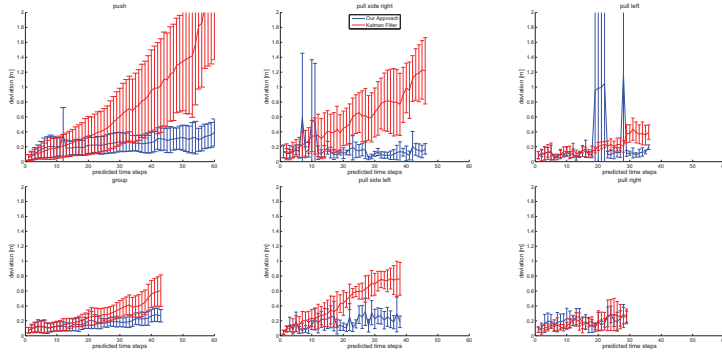


Figure 9.12.: Error Bars of position prediction.

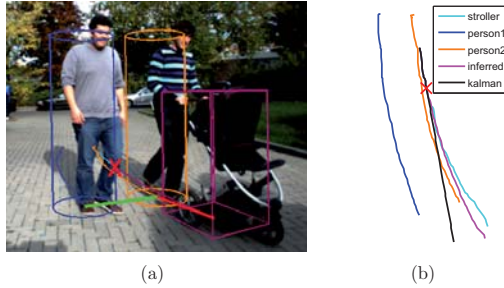


Figure 9.13.: (a) Tracked observation, lost stroller at the red X. (b) Prediction results for Kalman Filter (black line) and our approach (cyan line).

Fig. 9.13 illustrates a typical setup. We run these tests on our training data. Tracking is assumed to be lost after 15 frames and all remaining frames are predicted by the Kalman filter and our model. The results of this experiment are shown in Fig. 9.12. We plot the mean prediction distance including uncertainty vs. number of frames looked ahead. With an increasing number of frames, the Kalman filter diverges significantly more than our approach.

9.7. Discussion

We have presented a framework that can track both known (pedestrians) and unknown objects and simultaneously infer which objects belong together. Furthermore, the pro-



Figure 9.14.: Result images showing tracked persons and their associated objects with correct action inference. Interactions are visualized by linking the foot-points of the interacting objects by a colored line. (Legend: **pull side right**, **pull side left**, **pull right**, **pull left**, **push**, **group**).

posed model can be used to infer object types and the interaction patterns occurring between associated objects. The interaction component is built on top of tracking which means that the tracking component is decoupled and can be exchanged by another approach. The only input required to the model are the relative position and velocity of two entities and some representation of their appearance. With the decomposition of interaction classification into pairwise interactions we ensured the scalability of the framework which has quadratic runtime, and would grow exponentially if using a fully connected model.

The action classification has two advantages. On the one side, it can help to improve predictions for the continuation of each trajectory in case of detection/tracking failures. As we have shown when simulating a tracking loss of one participant of the inferred interaction, by using inference-based prediction based on the motion of another scene participant we could infer the most likely position of the lost object more precisely than when extrapolating its position with a Kalman Filter. On the other side, action classification can be used for adaptation of dynamic models for certain object/person constellations. As our results show the interaction classification works reasonable well and can improve the tracking by inferring the interaction classification back into the tracking loop. One future direction worth investigating, could be to extend the model such that no object category or interaction type needs to be specified and the model can infer from underlying observations possible object categories and interactions by clustering using some non-parametric approach, e.g., a Hierarchical Dirichlet Process (Teh *et al.*, 2006).

One can argue if the proposed set of interactions is useful. Considering scene understanding in an airport or a crowded train station, there are various reasons why it could be interesting to know which person is carrying or was last carrying some specific bag. A stable tracking of that person through the airport might lead to a frame that displays an identifiable view. On the way the tracking could be lost due to viewpoint changes in different cameras or occlusions, which we can tackle by supporting the tracking of once associated persons/objects by that interaction. Assume the bag is occluded for 10 frames. Since we know that the person was carrying it before and we know where the bag should be situated relative to the person, we can continue the bags tracks and find it back later or notify the authorities to have a look at this very video sequence if the bag does not show up next to the person again. In the scope of robot (service robot at an airport or a train station) perception the use-case would be an improvement of path planning. A common strategy here is to incorporate the predictions of the tracks of pedestrians into the motion planning of the robot which results in a more robust and smooth path of the robot. Other scene entities, such as strollers or suitcases, were usually neglected so far or were treated as static objects. With our approach we can provide the information about other scene entities with the corresponding motion model which will result in a more robust path planning. Another advantage is that the robot can become more social given the interaction classification by not planning to drive between entities that interact with each other, even if there is enough space in-between. In case

other interactions are present in the scene and are frequent enough they can easily be learned with our open-ended framework.

In this thesis we have investigated the topic of multi-object tracking from the perspective of a moving observer. The initial part of the work was devoted to overcoming a typical issue of a classical *tracking-by-detection* pipeline, the requirement of executing a computationally expensive object detector in each frame. The second part of the work was concerned with the development of a system that is able to track both known and unknown objects in the scene in a *tracking-before-detection* framework and being able to associate objects involved in an interaction and to predict the corresponding type of interaction.

The hybrid-tracking systems we developed combine a complex high-level tracker with cheap low-level trackers that take the brunt of the work of following individual persons over time. This strategy is based on the idea that once a person is detected in a frame, its appearance and location will change only slightly in the following frames, making simple low-level tracking feasible. Using a cheap appearance-based level-set tracker as the low-level tracker, the detector needs to be activated only every k frames. Given stereo range data the execution of the detector is further reduced to few ROIs in the image and the tracking is performed using the cheap low-level ICP tracker. The high-level tracker associates the resulting tracklets to plausible trajectories, taking care of long-term data association and maintaining person identities in case of occlusions. Through this combination fewer detector evaluations are required than in conventional *tracking-by-detection* approaches, which makes the proposed hybrid-trackers suitable for real time scenarios. Based on the experience with the cheap ICP-based tracking framework, a *tracking-before-detection* framework has been developed that tracks pedestrians and unknown objects purely based on stereo information. This capability of tracking people together with objects they are interacting with allowed the development of an interaction classification framework that classifies the objects and interaction types, and is able to make predictions how this interaction will continue in the next video frames. The developed approach treats those three problems – person-object assignment, object/interaction classification, and prediction – together in a probabilistic graphical model.

Quantitative evaluation results on several challenging sequences captured in crowded inner city scenarios corroborate the capability of the approach to track pedestrians in real time, to track all unknown objects and to associate them to pedestrians and to predict their interaction types.

10.1. Contributions

In detail, the contributions we have made are as follows.

In Chapter 4.4 we proposed a hybrid-tracking system that combines a high-level tracker with a cheap level-set based low-level tracker. The level-set tracker generates tracklets by following individual persons over time using a foreground/background color model. The high-level tracker links the resulting tracklets to long-term plausible trajectories and performs several consistency checks in order to detect failures of the level-set tracker and to enforce reinitialization by executing the object detector. With this combination the detector needs to be activated only every k frames. Tests on challenging shopping street scenarios corroborated that this combination with the proposed consistency checks delivers state-of-the-art tracking performance, but with significantly fewer detector evaluations. In Chapter 4.5 we continued on the strategy of a hybrid-tracker, but instead of the level-set tracker we proposed an ICP-based tracker, which tracks pedestrians over time starting with an initial detection from a detector and updating the position of the detected object by registration of the corresponding point clouds over time. The stereo information allows us to constrain the detector to run only on some small ROIs that are extracted from a 3D depth-based occupancy map. Furthermore, the ROIs are tracked over time and only newly appearing ROIs are evaluated by the detector. Again, the resulting framework delivers stable tracking performance for crowded scenarios, but with a significantly higher frame rate than the classical *tracking-by-detection* approaches. In both hybrid-tracking approaches the high-level tracker performs consistency checks which are required in order to detect drift of the low-level trackers. When a consistency check fails, the high-level tracker requests a detector evaluation either for the entire image as in the level-set approach or for the corresponding ROI in the ICP-based approach. However, such a procedure makes the direct application on a robotic platform difficult, since it is not predictable how much resources are required for the detector. Therefore, in Chapter 7 we proposed a framework that copes with this problem by fixing the budget of the detector to evaluate only a fixed number of ROIs in each frame. Using a Poisson process coupled with each tracked ROI, we balance the ROI selection over time, such that at each time instant, only those ROI candidates are considered for which attention is most urgently required in order to produce stable tracking results. As our experiments show, we reach state-of-the-art tracking performance already with three ROI evaluations per frame. The entire framework runs with more than 15fps and the fixed computation budget allows for a direct application to a mobile system.

In Chapter 6 we showed that with a relatively simple and fast depth-template based approach we can reach remarkable detection performance on very challenging outdoor data. The developed approach performs distance measurements between a learned depth template and each extracted ROI in a sliding window manner. This procedure yields several positive detections on each target object, which are then pruned to the final detection set with a twofold non-minimum suppression method. The presented experiments on stereo sequences showed that our approach achieves superior performance on typical shopping street scenarios. Furthermore, the proposed approach is not only significantly faster (40fps) than state-of-the-art full-body detectors, but also copes with the crucial problem of full-body detectors, detecting pedestrians close to the camera where the lower part of the body is not visible.

In Chapter 8 we introduced a *tracking-before-detection* framework which can be considered as a significant step to a better scene understanding. The proposed ROI segmentation approach allows to segment the ROI into individual objects which are tracked over time using the ICP tracker. During the tracking procedure the proposed 3D shape model is built which improves the tracking performance of the simple ICP tracker on the one side and allows to detect anomalous shapes, such as carried items on a person's body on the other side. The tests on several challenging video sequences of busy pedestrian zones affirm the capability of the framework to track known and unknown object categories with remarkable robustness. In Chapter 9, relying on tracks with corresponding 3D shapes from known and unknown objects yielded by the *tracking-before-detection* approach, we presented an interaction classification framework which associates pedestrians to their objects and infers the corresponding interaction type. The framework is formulated using a graphical model, which allowed to infer predictions for the continuation of each trajectory in case of detection/tracking failures. Furthermore, it can be used for an adaptation of dynamic models for certain object/person constellations.

10.2. Perspectives

One of the most fascinating abilities of humans comes from the visual system that enables us to recognize and understand the complex visual world remarkably fast and accurately. With the methods proposed in this thesis, we have come a tiny step closer to this capability. Being able to detect and track people and objects in real time and predict people's interaction with their world, we are still very far away from the goal of interpreting the entire scene within a blink, as it is performed by humans. There are still a number of challenges and interesting future directions, which we will discuss in the following.

Ground Plane Modeling. In all the proposed approaches in this thesis, we used the simplified assumption of a flat ground surface that is simply approximated by a plane. The plane is estimated from 3D points that are not embedded into ROIs as described in Chapter 5. The majority of these points is usually located close to the camera, therefore

the flat plane approximation is valid for objects close to the camera, but is often violated for farther-away objects. Sequences we have used in order to assess the performance of our methods were captured in city zones, which are usually flat. Thus, the flat ground plane assumption is a good approximation in this case, allowing to generate correct tracks also for objects in far range. However, during tests with the EUROPA robot we encountered the problem several times that the position estimation of pedestrians and their back-projection in the image was not consistent due to the approximation of the ground surface by a single plane. One solution could be to use a better and more precise representation for the ground surface as presented by Wedel *et al.* (2009), where the road surface is modeled as a general parametric B-spline curve.

Moreover, depending on the location of the moving observer it could be beneficial to estimate multiple ground planes. In case the cameras are mounted in a vehicle that drives on the road, a single ground plane estimate is not sufficient, since pedestrians are usually moving on sidewalks that are located higher than the car's road.

Multi-Low-Level Tracker. In Chapter 4, we have presented hybrid tracking systems based on different low-level trackers (LS-tracker and ICP-tracker) which output tracklets that were linked to long-term trajectories using the high-level tracker. The divergence of the low-level trackers in both approaches is detected by several consistency checks and triggers the execution of a computationally expensive object detector. Obviously, the goal is to reduce unnecessary detector evaluation caused by drift of the low-level trackers to a minimum. Considering the situation when both low-level trackers fail, it is evident that their output is complementary. On the one side, the LS-tracker is sensitive to lighting changes caused by shadowing effects, however, the ICP-tracker can track through lighting changes without divergence. Pedestrians with similar appearance, located close together in the image plane due to perspective transformation will cause divergence of the LS-tracker. However, the ICP-tracker will track these pedestrians successfully in this case, due to large spatial distance in 3D world coordinates. On the other side, using the ICP-tracker we have to deal with the well-known shadowing problem, which occurs for the parts of the image which are visible only from one camera. This might cause a delayed initialization of a track since, due to the missing depth information in the shadows, no ROI can be extracted. However, the LS-tracker can track from the initial detection and is not affected by the shadow effects. Overall, one interesting extension would be to employ both low-level trackers in parallel and let the high-level tracker decide during the hypothesis selection process which combination of the evidence is most likely consistent with the scene.

Articulated Tracking. The level-set based hybrid-tracking system presented in Chapter 4 could be a good possible starting point for articulated tracking, where instead of only considering the pedestrians as a point in 3D or a contour, the current body pose is estimated. Surely, it might be beneficial to consider the evolution of the shape over time before inferring the body pose, since the shape information from a single frame is ambiguous. Another interesting approach for body pose estimation could be based

on GCTs. To this end, one can learn from annotations, which rays of the GCT fixed to a reference orientation are selected/activated for the corresponding body pose. The distance of each selected ray will then infer the correct pose of the individual body part. When incorporating the evolution of distances of rays clustered into individual parts over time, one can enforce a smooth transition of each body part from one state to the next, avoiding abrupt non-natural transitions.

ROI Segmentation. Even though we can segment the ROIs into individual object regions for close range pedestrians or objects successfully (cf. Chapter 5.3), the approach fails for objects located farther away from the camera. The volumetric information from farther objects, required for our segmentation approach, becomes unreliable due to the discrete disparity values of stereo approaches and the limited baseline of the stereo camera setup. While the current system relies only on depth information, it might be reasonable to not only consider the density of points during the mode search process, but to also incorporate color information from the individual 3D points inside the ROI. It is then important to couple the segmentation process in 3D with the corresponding result in 2D. Since when incorporating the color information into the segmentation process, the resulting ROI segments in 3D can yield disjoint areas in the image plane.

Another problem is Applying the presented ROI segmentation to non-cylindrical objects such as cars or benches usually results in over-segmentation of these objects. To generalize the tracking of unknown objects to other scenarios, it should be considered how the segmentation process could be adapted in order to segment non-cylindrical objects correctly.

Object Classification. The novel GCT representation introduced in Chapter 8 opens up a number of interesting possibilities for future research. One of the promising directions could be object classification. A database of GCTs built up from annotations for different object classes could be the first step. Each object class in the database will be represented by different GCTs captured from different views of the object. Then, during the tracking process, as soon as the track reaches a certain confidence, by aligning and measuring the similarity of the GCT built up so far to the examples in the database, one can infer the possible object class. Knowing the object class can then be used in order to adapt the motion model for the prediction step in the tracking procedure.

Action Classification. Another interesting direction of research using the GCT representation is human activity recognition. It can be used for example in order to automatically extract semantic information from a video sequence or in surveillance systems in order to detect abnormal behavior of humans and to raise an alarm in possibly dangerous situations. Bobick and Davis (2001) propose motion history images extracted from foreground regions yielded by background subtraction in order to encode motion. In particular, a motion history image (MHI) represents a short span of the video sequence where the foreground images are simply overlayed. For the classification *Hu Moments* (Hu, 1962) are computed for each MHI and are matched to a view dependent database of labeled MHIs using the Bhattacharyya distance. In a similar way, we could use the

temporal information accumulated in the individual rays of the GCT in order to reason about possible activities performed by the tracked objects.

More General GCTs. The GCTs are generated by placing the center of the GCT on the center of the segmented ROI and casting radial rays over a number of discrete height levels (cf. Chapter 8). From the 3D points that fall inside a cylinder along the ray, only the distance from the closest point to the ray to the center axis is stored. However, in the current implementation, we use a fixed length of 0.7m for the cast rays, which works well for our scenarios, because the expected objects (such as pedestrians, child strollers, suitcases) resulting from the ROI segmentation have cylindrical shape with a radius below 0.7m. Assuming that the ROI segmentation yields correct segmentations for non-cylindrical objects such as cars as well, the length of cast rays should be adapted depending on the size of the individual ROI. In general, we can set it to infinite since the construction and updating of the individual GCT occurs only using the associated 3D points from the segment of an ROI.

From Interaction Classification to Better Tracking. In Chapter 9 we have shown an example demonstrating that the learned interaction classification model can also perform a predictive task. In case of segmentation/detection failure, the position of the corresponding object can be inferred from the position of the object it is interacting with. In several cases, it showed a more precise position prediction, especially in the long term, compared to path extrapolation by using a Kalman filter. The next step would be to feed the semantic information about the interaction into the tracking process by, e.g., adapting the prediction step of the Kalman Filter according to the interaction type.

Unsupervised Environment Understanding. The probabilistic approach for classification of object interactions presented in Chapter 9 is limited to predefined object classes and possible interaction types. One direction that could be investigated in the future would be first to drop all assumptions about possible object categories and model them using latent variables. First experiments already showed promising results where the inferred clusters actually correspond to the represented object categories. The next step would be to do the same for the possible interaction types, which should not be predefined but inferred from the possible object class clusters and the relative spatial arrangements of the objects. Such a general model can then infer the possible object categories and their possible interactions in a completely unsupervised way. Employing such an approach on a robot, which can be placed in an arbitrary unknown environment, could allow it to learn the interactions and then use this information for a better motion prediction of the individual objects and thus improve path planning and scene understanding.

Bibliography

- Amer, M. R. and Todorovic, S. (2011). A Chains Model for Localizing Participants of Group Activities in Videos. In *International Conference on Computer Vision*.
- Amer, M. R., Xie, D., Zhao, M., Todorovic, S., and Zhu, S.-C. (2012). Cost-Sensitive Top-Down/Bottom-Up Inference for Multiscale Activity Recognition. In *European Conference on Computer Vision*.
- Andriluka, M., Roth, S., and Schiele, B. (2008). People Tracking-by-Detection and People Detection-by-Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Andriluka, M., Roth, S., and Schiele, B. (2010). Monocular 3D Pose Estimation and Tracking by Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Andriyenko, A. and Schindler, K. (2011). Multi-Target Tracking by Continuous Energy Minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Andriyenko, A., Schindler, K., and Roth, S. (2012). Discrete-Continuous Optimization for Multi-Target Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Arras, K., Grzonka, S., Luber, M., and Burgard, W. (2008). Efficient People Tracking in Laser Range Data Using a Multi-Hypothesis Leg-Tracker with Adaptive Occlusion Probabilities. In *IEEE International Conference on Robotics and Automation*.
- Arun, K. S., Huang, T. S., and Blostein, S. D. (1987). Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**(5).

- Badino, H., Franke, U., and Mester, R. (2007). Free Space Computation using Stochastic Occupancy Grids and Dynamic Programming. In *ICCV Workshop on Dynamical Vision*.
- Badino, H., Franke, U., and Pfeiffer, D. (2009). The Stixel World - A Compact Medium Level Representation of the 3D-World. In *DAGM Annual Pattern Recognition Symposium*.
- Bajracharya, M., Moghaddam, B., Howard, A., Brennan, S., and Matthies, L. (2009a). A Fast Stereo-based System for Detecting and Tracking Pedestrians from a Moving Vehicle. *International Journal of Robotics Research*, **28**(11-12).
- Bajracharya, M., Moghaddam, B., Howard, A., Brennan, S., and Matthies, L. (2009b). Results from a Real-time Stereo-based Pedestrian Detection System on a Moving Vehicle. In *IEEE International Conference on Robotics and Automation*.
- Baker, S. and Matthews, I. (2004). Lucas-Kanade 20 Years On: A Unifying Framework. *International Journal of Computer Vision*, **56**(3).
- Bansal, M., Jung, S. H., Matei, B., Eledath, J., and Sawhney, H. S. (2010). A Real-time Pedestrian Detection System based on Structure and Appearance Classification. In *IEEE International Conference on Robotics and Automation*.
- Bar-Shalom, Y. and Li, X.-R. (1995). *Multitarget-multisensor tracking: Principles and techniques*. YBS Publishing.
- Baumgartner, T., Mitzel, D., and Leibe, B. (2013). Tracking People and Their Objects. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Benedict, T. R. and Bordner, G. W. (1962). Synthesis of an optimal set of radar track-while-scan smoothing equations. In *IRE Transactions on Automatic Control*.
- Benenson, R., Mathias, M., Timofte, R., and Gool, L. V. (2012). Pedestrian Detection at 100 Frames per Second. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Berclaz, J., Fleuret, F., and Fua, P. (2006). Robust People Tracking with Global Trajectory Optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Berclaz, J., Turetken, E., Fleuret, F., and Fua, P. (2011). Multiple Object Tracking using K-Shortest Paths Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33**(9).
- Besl, P. J. and McKay, H. D. (1992). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(2).

- Bibby, C. and Reid, I. (2008). Robust Real-Time Visual Tracking using Pixel-Wise Posteriors. In *European Conference on Computer Vision*.
- Bjorkman, M. and Kragic, D. (2010). Active 3D Segmentation and Detection of Unknown Objects. In *Intelligent Robots and Systems*.
- Bobick, A. F. and Davis, J. W. (2001). The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(3).
- Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(11).
- Breitenstein, M., Reichlin, F., Leibe, B., Koller-Meier, E., and Van Gool, L. (2009). Robust Tracking-by-Detection Using a Detector Confidence Particle Filter. In *International Conference on Computer Vision*.
- Brox, T. and Weickert, J. (2004). Level Set Based Image Segmentation with Multiple Regions. In *DAGM Annual Pattern Recognition Symposium*.
- Chen, Y. and Medioni, G. (1991). Object Modeling by Registration of Multiple Range Images. In *IEEE International Conference on Robotics and Automation*.
- Cheng, Y. (1995). Mean Shift, Mode Seeking, and Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(8).
- Choi, W. and Savarese, S. (2010). Multiple Target Tracking in World Coordinate with Single, Minimally Calibrated Camera. In *European Conference on Computer Vision*.
- Choi, W. and Savarese, S. (2012). A Unified Framework for Multi-Target Tracking and Collective Activity Recognition. In *European Conference on Computer Vision*.
- Choi, W., Shahid, K., and Savarese, S. (2009). What are they doing? : Collective Activity Classification Using Spatio-Temporal Relationship Among People. In *Proc. of 9th International Workshop on Visual Surveillance (VSW09) in conjunction with ICCV*.
- Choi, W., Pantofaru, C., and Savarese, S. (2011a). Detecting and Tracking People using an RGB-D Camera via Multiple Detector Fusion. In *ICCV Workshop, 1st IEEE Workshop on Challenges and Opportunities in Robot Perception*.
- Choi, W., Shahid, K., and Savarese, S. (2011b). Learning Context for Collective Activity Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

- Collins, R. T., Lipton, A. J., Fujiyoshi, H., and Kanade, T. (2001). Algorithms for Cooperative Multisensor Surveillance. *Proceedings of the IEEE*, **89**(10).
- Comaniciu, D., Ramesh, V., and Meer, P. (2003). Kernel-Based Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(5).
- Cremers, D. (2006). Dynamical Statistical Priors for Level Set Based Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**(8).
- Cremers, D., Rousson, M., and Deriche, R. (2007). A Review of Statistical Approaches to Level Set Segmentation Integrating Color, Texture, Motion and Shape. *International Journal of Computer Vision*, **72**(2).
- Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Damen, D. and Hogg, D. (2008). Detecting Carried Objects in Short Video Sequences. In *European Conference on Computer Vision*.
- Danescu, R., Oniga, F., and Nedeveschi, S. (2011). Modeling and Tracking the Driving Environment With a Particle-Based Occupancy Grid. *IEEE Transactions on Intelligent Transportation Systems*, **12**(4).
- Dellaert, F. and Thorpe, C. (1997). Robust Car Tracking using Kalman filtering and Bayesian Templates. In *Conference on Intelligent Transportation Systems*.
- Dollar, P., Wojek, C., Schiele, B., and Perona, P. (2009). Pedestrian Detection: A Benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Dollar, P., Belongie, S., and Perona, P. (2010). The Fastest Pedestrian Detector in the West. In *British Machine Vision Conference*.
- Elgammal, A., Harwood, D., and Davis, L. (2000). Non-parametric Model for Background Subtraction. In *European Conference on Computer Vision*.
- Enzweiler, M., Kanter, P., and Gavrila, D. (2008). Monocular Pedestrian Recognition Using Motion Parallax. In *IEEE Intelligent Vehicles Symposium*.
- Enzweiler, M., Eigenstetter, A., Schiele, B., and Gavrila, D. M. (2010). Multi-Cue Pedestrian Classification with Partial Occlusion Handling. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Ess, A., Schindler, K., Leibe, B., and Van Gool, L. (2009a). Improved Multi-Person Tracking with Active Occlusion Handling. In *IEEE International Conference on Robotics and Automation*.

- Ess, A., Leibe, B., Schindler, K., and Van Gool, L. (2009b). Robust Multi-Person Tracking from a Mobile Platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **31**(10).
- Everingham, M., Gool, L. V., Williams, C., Winn, J., and Zisserman, A. (2010). The PASCAL Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, **88**(14).
- Feldman, A., Hybinette, M., Balch, T., and Cavallaro, R. (2011). The Multi-ICP Tracker: An Online Algorithm for Tracking Multiple Interacting Targets. submitted, www.cc.gatech.edu/~tucker/Papers.
- Feldman, A., Hybinette, M., Balch, T., and Cavallaro, R. (2012). The Multi-ICP Tracker: An Online Algorithm for Tracking Multiple Interacting Targets. *J. Field Robotics*.
- Felzenszwalb, P. and Huttenlocher, D. (2006). Efficient Belief Propagation for Early Vision. *International Journal of Computer Vision*, **70**(1).
- Felzenszwalb, P., Girshick, R., and McAllester, D. (2010a). Cascade Object Detection with Deformable Part Models. *IEEE Conference on Computer Vision and Pattern Recognition*.
- Felzenszwalb, P., Girshick, B., McAllester, D., and Ramanan, D. (2010b). Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(9).
- Fischer, P. (2012). Towards Realistic Multi-Person Tracking by Integration of Social Forces. In *Master thesis - supervised by D. Mitzel and B. Leibe*.
- Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, **24**(6).
- Floros, G. and Leibe, B. (2012). Joint 2D-3D temporally consistent semantic segmentation of street scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Freund, Y. and Schapire, R. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*.
- Frintrop, S., Backer, G., and Rome, E. (2005). Goal-directed Search with a Top-down Modulated Computational Attention System. In *DAGM Annual Pattern Recognition Symposium*.

- Fukunaga, K. and Hostetler, L. (1975). The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition. *IEEE Transactions on Information Theory*, **21**(1).
- Gavrila, D. and Munder, S. (2007). Multi-Cue Pedestrian Detection and Tracking from a Moving Vehicle. *International Journal of Computer Vision*, **73**(1).
- Ge, W. and Collins, R. (2008). Multi-target Data Association by Tracklets with Unsupervised Parameter Estimation. In *British Machine Vision Conference*.
- Geiger, A., Roser, M., and Urtasun, R. (2010). Efficient Large-Scale Stereo Matching. In *Asian Conference on Computer Vision*.
- Geiger, A., Ziegler, J., and Stiller, C. (2011). StereoScan: Dense 3d Reconstruction in Real-time. In *IEEE Intelligent Vehicles Symposium*.
- Gelb, A. (1974). Applied Optimal Estimation. In *MIT Press*.
- Gennery, D. B. (1982). Tracking of known three-dimensional objects. *Association for the Advancement of Artificial Intelligence*.
- Geronimo, D., Sappa, A., Ponsa, D., and Lopez, A. (2010a). 2D-3D-based On-Board Pedestrian Detection System. *Computer Vision and Image Understanding (CVIU)*, **114**.
- Geronimo, D., Lopez, A., Sappa, A., and Graf, T. (2010b). Survey of Pedestrian Detection for Advanced Driver Assistance Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(7).
- Goldberg, A. (1997). An efficient implementation of a scaling minimum-cost flow algorithm. *Journal of Algorithms*, **1**(22).
- Grabner, H., Grabner, M., and Bischof, H. (2006). Real-Time Tracking via On-line Boosting. In *British Machine Vision Conference*.
- Grimson, W. E. L., Stauffer, C., Romano, R., and Lee, L. (1998). Using Adaptive Tracking to Classify and Monitor Activities in a Site. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Hager, G. D. and Belhumeur, P. N. (1998). Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(10).
- Haritaoglu, I., Harwood, D., and Davis, L. S. (1998). Ghost: A Human Body Part Labeling System Using Silhouettes. In *International Conference on Pattern Recognition*.

- Haritaoglu, I., Harwood, D., and Davis, L. S. (2000). W4: Real-Time Surveillance of People and Their Activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(8).
- Harris, C. and Stephens, M. (1988). A Combined Corner and Edge Detector. In *In Proc. of Fourth Alvey Vision Conference*.
- Heigeas, L., Luciani, A., Thollot, J., and Castagné, N. (2003). A Physically-Based Particle Model of Emergent Crowd Behaviors. In *Graphicon*.
- Helbing, D. and Molnár, P. (1995). Social force model for pedestrian dynamics. *Phys. Rev. E*, **51**(5).
- Hoiem, D., Efros, A., and Hebert, M. (2006). Putting Objects Into Perspective. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Horbert, E., Rematas, K., and Leibe, B. (2011). Level-set person segmentation and tracking with multi-region appearance models and top-down shape information. In *International Conference on Computer Vision*.
- Hu, M. K. (1962). Visual Pattern Recognition by Moment Invariants. *IRE Transactions on Information Theory*, **8**.
- Huang, C., Wu, B., and Nevatia, R. (2008). Robust Object Tracking by Hierarchical Association of Detection Responses. In *European Conference on Computer Vision*.
- Ikemura, S. and Fujiyoshi, H. (2010). Real-Time Human Detection using Relational Depth Similarity Features. In *Asian Conference on Computer Vision*.
- Irani, M. and Anandan, P. (1998). Video Indexing Based on Mosaic Representations. *IEEE Trans. Mach. Intell.*, **20**(6).
- Isard, M. and MacCormick, J. (2001). BraMBLe: A Bayesian Multiple-Blob Tracker. In *International Conference on Computer Vision*.
- Itti, L., Koch, C., and Niebur, E. (1998). A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(11).
- Izadinia, H. and Shah, M. (2012). Multiple People Multiple Parts Tracker. In *European Conference on Computer Vision*.
- Jain, R. and Nagel, H. (1979). On the Analysis of Accumulative Difference Pictures from Image Sequences of Real World Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **1**(2).

- Jansen, L., Onat, S., and Koenig, P. (2009). Influence of disparity on fixation and saccades in free viewing of natural scenes. *Journal of Vision*, **9**(1).
- Jepson, A. D., Fleet, J. D., and El-Maraghi, T. F. (2003). Robust Online Appearance Models for Visual Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(10).
- Kaestner, R., Maye, J., and Siegwart, R. (2012). Generative Object Detection and Tracking in 3D Range Data. In *IEEE International Conference on Robotics and Automation*.
- Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active Contour Models. *International Journal of Computer Vision*, **1**(4).
- Kaucic, R., Perera, A., Brooksby, G., Kaufhold, J., and Hoogs, A. (2005). A Unified Framework for Tracking through Occlusions and Across Sensor Gaps. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Keller, C., Llorca, D. F., and Gavrila, D. M. (2009). Dense Stereo-based ROI Generation for Pedestrian Detection. In *DAGM Annual Pattern Recognition Symposium*.
- Kitt, B., Geiger, A., and Lategahn, H. (2010). Visual Odometry based on Stereo Image Sequences with RANSAC-based Outlier Rejection Scheme. In *Intelligent Vehicles Symposium (IV)*.
- Klügl, F. and Rindsfuser, G. (2007). Large-Scale Agent-Based Pedestrian Simulation. In *MATES '07: Proceedings of the 5th German conference on Multiagent System Technologies*.
- Koller, D., Danilidis, K., and Nagel, H. (1993). Model-based object tracking in monocular image sequences of road traffic scenes. *International Journal of Computer Vision*, **10**(3).
- Kuhn, H. W. (1955). The Hungarian Method for the Assignment Problem. *Naval Research Logistic Quarterly*, **2**(1).
- Kuo, C.-H., Huang, C., and Nevatia, R. (2010). Multi-Target Tracking by On-Line Learned Discriminative Appearance Models. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Labayrade, R. and Aubert, D. (2003). A Single Framework for Vehicle Roll, Pitch, Yaw Estimation and Obstacles Detection by Stereovision. In *IEEE Intelligent Vehicles Symposium*.

- Lang, C., Nguyen, T. V., Katti, H., Yadati, K., Kankanhalli, M. S., and Yan, S. (2012). Depth Matters: Influence of Depth Cues on Visual Saliency. In *European Conference on Computer Vision*.
- Lau, B., Arras, K., and Burgard, W. (2009). Tracking Groups of People with a Multi-Hypothesis Tracker. In *IEEE International Conference on Robotics and Automation*.
- Leal-Taixe, L., Pons-Moll, G., and Rosenhahn, B. (2011). Everybody needs somebody: modeling social and grouping behavior on a linear programming multiple people tracker. *ICCV Workshop ICCVW. 1st Workshop on Modeling, Simulation and Visual Analysis of Large Crowds*.
- Leibe, B., Schindler, K., and Van Gool, L. (2008a). Coupled Object Detection and Tracking from Static Cameras and Moving Vehicles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**(10).
- Leibe, B., Leonardis, A., and Schiele, B. (2008b). Robust Object Detection with Interleaved Categorization and Segmentation. *International Journal of Computer Vision*, **77**(1-3).
- Lerner, A., Chrysanthou, Y., and Lischinski, D. (2007). Crowds by Example. *Computer Graphics Forum*, **26**(3).
- Leventon, M. E., Grimson, W. E. L., and Faugeras, O. D. (2000). Statistical Shape Influence in Geodesic Active Contours. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Li, C., Xu, C., Gui, C., and Fox, M. (2005). Level Set Evolution without Re-initialization: A New Variational Formulation. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, **60**(2).
- Luber, M., Stork, J. A., Tipaldi, G. D., and Arras, K. O. (2010). People tracking with human motion predictions from social forces. In *IEEE International Conference on Robotics and Automation*.
- Luber, M., Spinello, L., and Arras, K. (2011). People Tracking in RGB-D Data With Online-Boosted Target Models. In *Intelligent Robots and Systems*.
- Mei, C., Sibley, G., Cummins, M., Newman, P., and Reid, I. (2009). A constant time efficient stereo slam system. In *British Machine Vision Conference*.
- Mikolajczyk, K. and Schmid, C. (2004). Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, **60**(1).

- Mitzel, D. and Leibe, B. (2011). Real-Time Multi-Person Tracking with Detector Assisted Structure Propagation. In *ICCV Workshop, 1st IEEE Workshop on Challenges and Opportunities in Robot Perception*.
- Mitzel, D. and Leibe, B. (2012a). Close-Range Human Detection for Head-Mounted Cameras. In *British Machine Vision Conference*.
- Mitzel, D. and Leibe, B. (2012b). Taking Mobile Multi-Object Tracking to the Next Level: People, Unknown Objects, and Carried Items. In *European Conference on Computer Vision*.
- Mitzel, D., Horbert, E., Ess, A., and Leibe, B. (2010). Multi-person tracking with sparse detection and continuous segmentation. In *European Conference on Computer Vision*.
- Mitzel, D., Sudowe, P., and Leibe, B. (2011a). Real-Time Multi-Person Tracking with Time-Constrained Detection. In *British Machine Vision Conference*.
- Mitzel, D., Floros, G., Sudowe, P., van der Zander, B., and Leibe, B. (2011b). Real Time Vision based Multi-Person Tracking for Mobile Robotics and Intelligent Vehicles. In *International Conference on Intelligent Robotics and Applications*.
- Nistér, D. and Stewénus, H. (2007). A minimal solution to the generalised 3-point pose problem. *Journal of Mathematical Imaging and Vision*.
- Okuma, K., Taleghani, A., de Freitas, N., Little, J., and Lowe, D. (2004). A Boosted Particle Filter: Multitarget Detection and Tracking. In *European Conference on Computer Vision*.
- Paragios, N. and Deriche, R. (2000). Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(3).
- Pearl, J. (1986). Fusion, Propagation, and Structuring in Belief Networks. *Artificial Intelligence*.
- Pellegrini, S., Ess, A., Schindler, K., and Van Gool, L. (2009). You'll Never Walk Alone: Modeling Social Behavior for Multi-Target Tracking. In *International Conference on Computer Vision*.
- Pellegrini, S., Ess, A., and Van Gool, L. (2010). Improving Data Association by Joint Modeling of Pedestrian Trajectories and Groupings. In *European Conference on Computer Vision*.
- Petrovskaya, A. and Thrun, S. (2009). Model Based Vehicle Detection and Tracking for Autonomous Urban Driving. *Autonomous Robots Journal*, **26**(2-3).

- Pirsiavash, H., Ramanan, D., and Fowlkes, C. C. (2011). Globally-optimal greedy algorithms for tracking a variable number of objects. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Pregaman, R., Thurber, R., and Bath, W. (1982). Retrospective Detection Algorithm for Extraction of Weak Targets in Clutter and Interference Environments. In *IEEE Int. Radar Conf.*
- Prisacariu, V. and Reid, I. (2009). fastHOG – a Real-Time GPU Implementation of HOG. Technical Report 2310/09, Dept. of Engineering Science, University of Oxford.
- Rabe, C., Müller, T., Wedel, A., and Franke, U. (2010). Dense, robust, and accurate motion field estimation from stereo image sequences in real-time. In *European Conference on Computer Vision*.
- Reid, D. (1979). An Algorithm for Tracking Multiple Targets. *IEEE Transactions on Automatic Control*, **24**(6).
- Rother, C., Kolmogorov, V., and Blake, A. (2004). Grabcut: Interactive Foreground Extraction Using Iterated Graph Cuts. In *Special Interest Group on Graphics and Interactive Techniques*.
- Saboia, P. and Goldenstein, S. (2011). Crowd Simulation: Improving Pedestrians’ Dynamics by the Application of Lattice-Gas Concepts to the Social Force Model. In *SIBGRAPI - Conference on Graphics, Patterns and Images*.
- Salmond, D. and Birch, H. (2001). A Particle Filter for Track-Before-Detect. In *American Control Conf.*
- Schindler, K., U, J., and Wang, H. (2006). Perspective n-view Multibody Structure-and-Motion through Model Selection. In *European Conference on Computer Vision*.
- Schindler, K., Ess, A., Leibe, B., and Van Gool, L. (2010). Automatic Detection and Tracking of Pedestrians from a Moving Stereo Rig. *Journal of Photogrammetry and Remote Sensing*, **65**(6).
- Schweitzer, H., Bell, J. W., and Wu, F. (2002). Very Fast Template Matching. In *European Conference on Computer Vision*.
- Shashua, A., Gdalyahu, Y., and Hayon, G. (2004). Pedestrian Detection for Driving Assistance Systems: Single-Frame Classification and System Level Performance. In *IEEE Intelligent Vehicles Symposium*.
- Singh, V., Wu, B., and Nevatia, R. (2008). Pedestrian Tracking by Associating Tracklets using Detection Residuals. In *IEEE Workshop on Motion and Video Computing*.

- Smith, K., Quelhas, P., and Gatica-Perez, D. (2006). Detecting Abandoned Luggage Items in a Public Space. In *PETS*.
- Sommerlade, E. and Reid, I. (2008a). Information theoretic Active Scene Exploration. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Sommerlade, E. and Reid, I. (2008b). Information-theoretic Decision Making for Exploration of Dynamic Scenes. In *International Workshop on Attention in Cognitive Systems*.
- Sommerlade, E. and Reid, I. (2010). Probabilistic surveillance with multiple active cameras. In *IEEE International Conference on Robotics and Automation*.
- Spinello, L. and Arras, K. O. (2011). People Detection in RGB-D Data. In *Intelligent Robots and Systems*.
- Stauffer, C. (2003). Estimating tracking sources and sinks. In *Proc. 2nd IEEE workshop on event mining*.
- Stauffer, C. and Grimson, W. (1999). Adaptive Background Mixture Models for Realtime Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Stauffer, C. and Grimson, W. E. L. (2000). Learning Patterns of Activity Using Real-Time Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(8).
- Sudowe, P. and Leibe, B. (2011). Efficient Use of Geometric Constraints for Sliding-Window Object Detection in Video. In *International Conference on Computer Vision Systems*.
- Tao, H., Sawhney, H., and Kumar, R. (2002). Object Tracking with Bayesian Estimation of Dynamic Layer Representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(1).
- Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, **101**(476).
- Teichman, A. and Thrun, S. (2011). Tracking-Based Semi-Supervised Learning. In *Robotics: Science and Systems*.
- Teichman, A., Levinson, J., and Thrun, S. (2011). Towards 3D Object Recognition via Classification of Arbitrary Object Tracks. In *IEEE International Conference on Robotics and Automation*.
- Tsai, A., Yezzi, A., and Willsky, A. (2001). Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification. *Transactions on Image Processing*, **10**(8).

- Tuzel, O., Porikli, F., and Meer, P. (2006). Region Covariance: A Fast Descriptor for Detection And Classification. In *European Conference on Computer Vision*.
- Vedaldi, A. and Soatto, S. (2008). Quick Shift and Kernel Methods for Mode Seeking. In *European Conference on Computer Vision*.
- Viola, P. and Jones, M. (2004). Robust real-time face detection. *International Journal of Computer Vision*, **57**(2).
- Walk, S., Schindler, K., and Schiele, B. (2010). Disparity Statistics for Pedestrian Detection: Combining Appearance, Motion and Stereo. In *European Conference on Computer Vision*.
- Wang, X., Han, T. X., and Yan, S. (2009). An HOG-LBP Human Detector with Partial Occlusion Handling. In *International Conference on Computer Vision*.
- Wedel, A., Rabe, C., Badino, H., Loose, H., Franke, U., and Cremers, D. (2009). B-Spline Modeling of Road Surfaces with and Application to Freespace Estimation. *IEEE Transactions on Intelligent Transportation Systems*, **10**(4).
- Weiss, Y. and Freeman, W. T. (2001). On the Optimality of Solutions of the Max-Product Belief-Propagation Algorithm in Arbitrary Graphs. *IEEE Transactions on Information Theory*, **47**(2).
- Wojek, C., Walk, S., Roth, S., and Schiele, B. (2011). Monocular 3D Scene Understanding with Explicit Occlusion Reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Wren, C., Azarbayejani, A., Darrell, T., and Pentland, A. (1997). Pfunder: Real-Time Tracking of the Human Body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**(7).
- Wu, B. and Nevatia, R. (2007). Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet Part Detectors. *International Journal of Computer Vision*, **75**(2).
- Yamaguchi, K., Berg, A., Ortiz, L., and Berg, T. (2011). Who are You with and where You are Going? In *IEEE Conference on Computer Vision and Pattern Recognition*.
- Yang, B. and Nevatia, R. (2012). Online Learned Discriminative Part-Based Appearance Models for Multi-human Tracking. In *European Conference on Computer Vision*.
- Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *ACM Comput. Surv.*, **38**(4).

- Zhang, L. and Nevatia, R. (2008). Global Data Association for Multi-Object Tracking Using Network Flows. In *European Conference on Computer Vision*.
- Zhao, T., Nevatia, R., and Wu, B. (2008). Segmentation and Tracking of Multiple Humans in Crowded Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**(7).
- Zheng, Q. and Chellappa, R. (1995). Automatic feature point extraction and tracking in image sequences for arbitrary camera motion. *International Journal of Computer Vision*, **15**(1-2).

Curriculum Vitae



Dennis Mitzel

E-Mail	mitzel@vision.rwth-aachen.de
Date of Birth	15.06.1983
Place of Birth	Katschar, Kazakhstan
Citizenship	German

Education

Mar. 2009 – Sep. 2013	Doctoral Student at RWTH Aachen University, Computer Vision Group. Supervisor: Prof. Dr. Bastian Leibe.
Apr. 2004 – Feb. 2009	Computer Science Studies at Rheinische Friedrich-Wilhelms-Universität Bonn. Degree: Dipl. Inform. Supervisor: Prof. Dr. Daniel Cremers.
1999 – 2003	Secondary School Heinrich Boell Gymnasium, Troisdorf. Degree: Hochschulreife
1996 – 1999	Secondary School at Korczak Realschule, Troisdorf. Degree: Hochoberschulreife
1989 – 1996	Lyzeum in Kazakhstan.

Professions

Mar. 2008 – Mar. 2009	Software Engineer, Fraunhofer Gesellschaft, IAIS, Net-Media.
Oct. 2007 – Mar. 2008	Student Assistant, Institute for Computer Science, University of Bonn.
Jun. 2007 – Sep. 2007	Intern, Arvato Systems North America Inc. in Toronto.
Apr. 2006 – Aug. 2006	Tutor, Institute for Computer Science, University of Bonn.
Oct. 2005 – Feb. 2006	Tutor, Institute for Computer Science, University of Bonn.
Jun. 2003 – Mar. 2004	Civil Service, Retirement Home, Hermann-Josef-Lascheid, Spich.

Publications

T. Baumgartner, D. Mitzel, B. Leibe: *Tracking People and Their Objects*, Computer Vision and Pattern Recognition Conference (CVPR), 2013

D. Mitzel, B. Leibe: *Close-Range Human Detection for Head-Mounted Cameras*, British Machine Vision Conference (BMVC), 2012.

D. Mitzel, B. Leibe: *Taking Mobile Multi-Object Tracking to the Next Level: People, Unknown Objects, and Carried Items*, European Conference on Computer Vision (ECCV), 2012.

D. Mitzel, B. Leibe: *Real-Time Multi-Person Tracking with Detector Assisted Structure Propagation*, Workshop on Challenges and Opportunities in Robot Perception (CORP), 2011

D. Mitzel, G. Floros, P. Sudowe, B. van der Zander, B. Leibe: *Real Time Vision based Multi-Person Tracking for Mobile Robotics and Intelligent Vehicles*, International Conference on Intelligent Robotics and Applications (ICIRA), 2011

D. Mitzel, P. Sudowe, B. Leibe: *Real-Time Multi-Person Tracking with Time-Constrained Detection*, British Machine Vision Conference (BMVC), 2011

D. Mitzel, E. Horbert, B. Leibe: *Multi-Person Tracking with Sparse Detection and Continuous Segmentation*, European Conference on Computer Vision (ECCV), 2010

E. Horbert, D. Mitzel, B. Leibe: *Geometrically Constrained Level-Set Tracking for Automotive Applications*, DAGM Annual Pattern Recognition Symposium, 2010

D. Mitzel, T. Pock, T. Schoenemann, D. Cremers: *Video Super Resolution Using Duality Based TV-L1 Optical Flow*, DAGM Annual Pattern Recognition Symposium, 2009

