Die approbierte Originalversion dieser Dissertation ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.
http://www.ub.tuwien.ac.at

The approved original version of this thesis is available at the main library of the Vienna University of Technology.
http://www.ub.tuwien.ac.at/eng

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Towards Exploiting Redundancy for 3D Scene Understanding from Videos

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der technischen Wissenschaften

by

Michael Hödlmoser
Registration Number 0827396

to the Faculty of Informatics
at the Vienna University of Technology

Advisors: Priv. Doz. Dr. Martin Kampel
Prof. Dr. Marc Pollefeys
Dr. Branislav Mičušík

The dissertation has been reviewed by:

(Priv. Doz. Dr. Martin Kampel)

(Prof. Dr. Marc Pollefeys)

Vienna, 09.04.2013

(Michael Hödlmoser)

Erklärung zur Verfassung der Arbeit

Michael Hödlmoser
Kreuzgasse 49/20, 1180 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 09.04.2013

(Michael Hödlmoser)

Acknowledgments

Before pursuing a PhD, people told me that finishing a PhD within a reasonable amount of time would be hard work and one definitely needs to have a strong will. Now, at the end of my studies, I know they were right. Completing a PhD is a major project which is intellectually challenging when trying to solve complex problems and emotionally draining when potential publications are simply rejected because good fortune is not with you. Furthermore, one needs to be passionate about a topic and - which in my opinion is the most important thing - one needs to have the right people surrounding and accompanying you on the journey.

I was lucky to have those supportive people surrounding me. First of all, I deeply thank Martin Kampel for giving me the opportunity to carry out this PhD work. Martin gave me unfaltering support whenever I needed it, he always had an open ear for any kind of problems and helped me to meet the right people. One of those people is Branislav Mičušík, one of the most impressive researchers I ever met. He taught me how to write high quality papers, helped me not to lose track of the „big picture“, supported me to find the right problems which were worth to be solved and showed faith in my work even if I lost faith in it myself. One more person I had the pleasure to work with is Marc Pollefeys. I thank Marc, first for giving me twice the opportunity to work with his amazing CVG group in Zurich, and second for serving on my committee. Thanks to all the people at the CVG group for their support and many fruitful discussions. Special thanks to Alex Mansfield for proofreading this thesis and giving me lots of helpful feedback. Thanks also to my co-author Ming-Yu Liu.

My appreciation for sharing their amazingly good mood with me goes to my boss Robert Sablatnig and his CVL team - Florian Kleber, Markus Diem, Stefan Fiel, Rainer Planinc, Fabian Hollaus, Michael Smolle, Peter Lischka, Katharina Pois, Melanie Gau, Elisabeth Wetzinger, Albert Kavelar, Jana Machajdik, Manuel Keglevic - to CVL friends Michael Bleyer, Christoph Rhemann, Angelika Garz, Julian Stöttinger and to the team from CogVis Ltd. I am especially grateful to Andreas Zweng and Sebastian Zambanini for sharing their offices and for having many fruitful discussions with me - at work or while drinking beer afterwards. Thanks for letting me be part of the CVL - such a togetherness is impressive! I also thank Michael Brandstötter who gave me the opportunity to work at CogVis Ltd.

I especially thank my parents, Monika and Klaus, for their boundless support throughout my whole life and for going through all the ups and downs of this PhD together with me. I furthermore want to express my heartfelt gratitude to my brother Stefan for his care, his support and for our outstanding friendship. Special thanks to Gerhard for introducing me to computer science quite a while ago. Thanks to all the rest of my beloved family for being there for me. When doing a PhD, working hard is important but it is also important to enjoy the free time. I am much obliged to have amazing friends, who make my free time enjoyable - thank you!

Most importantly, I am deeply grateful to my beloved Marie-Therese for constantly supporting me, for her endless love and patience, for partying with me when a paper was accepted, but also for comforting me when one was rejected! Thank you guys so much from the bottom of my heart,

Michael

The research presented in this dissertation was partially supported by the Austrian Research Promotion Agency (FFG) under project 830042 (Classification and Pose Recovery in Image Sequences, CAPRI) and CogVis Ltd.

“ Take things as they come.
But try to make things come
as you would like to take them. “

Curt Goetz (1888 - 1960)

Abstract

Humans are living in a 3D world and are able to reason about 3D properties. They can for example estimate occlusion boundaries and spatial arrangements of objects which is necessary for object manipulation and navigation. When capturing various real world scenes on 2D image planes, they may vary in terms of involved objects. These objects can for example be pedestrians, vehicles, chairs, walls, mountains. They may also provide a variety of appearances, namely different dimensions, or colors. Images showing these scenes may also be taken under changing environmental settings which may be changes in terms of lighting, weather condition, or the time of the day. Captured scenes do also differ regarding the events and actions which occur among objects. Due to this endless number of variations, computer vision applications and algorithms are designed to handle a specific task at a specific environmental setting in order to assure a certain accuracy for that task.

This thesis deals with overcoming the specificity of a single algorithm and increasing its robustness by exploiting redundant information for solving 3D computer vision tasks. Redundancy in this context is a combination of different algorithms, or a variety of different cues. The task of 3D scene understanding is divided into three parts, namely calibration, 3D reconstruction and 3D reasoning.

Two auto-calibration methods for traffic surveillance scenarios from videos are presented. The first algorithm estimates extrinsic and intrinsic camera parameters for a whole network of cameras from analyzing pedestrians. The second proposed method aims for calibrating a surveillance camera from pedestrians and zebra-crossings. Redundant information for calibration is gathered by combining multiple instances of a pedestrian over time and by combining static and dynamic objects by means of pedestrians and zebra-crossings.

Man-made indoor environments suffer from flat and textureless surfaces, where conventional, feature-based 3D reconstruction pipelines fail to estimate the 3D scene layout. In order to overcome this problem, the proposed work combines conventional feature matching techniques with 3D information coming from semantic reasoning. It is assumed that an image can be segmented in parts where each segment can be modeled by a planar patch. The patches' 3D surface normal orientations are estimated and a pixel-wise optimization is exploited in order to get the globally best surface normal orientation for each pixel. Redundant information for 3D surface labeling and 3D reconstruction is achieved by combining different segmentation methods when performing semantic reasoning and by combining semantic information with geometric information coming from conventional feature matches.

The task of 3D reasoning covers the description of a variety of different events which are for example human actions or interactions between objects. Nevertheless, 3D pose estimation and 3D tracking are the basis for analyzing these events. Therefore, two pose estimation, object classification and tracking algorithms for vehicles, which are the most important objects to be analyzed in computer vision besides persons, are presented. The proposed methods exploit existing 3D models for pose estimation and classification of vehicles. This first enables overcoming the ill-posed problem of projecting a pixel from the image plane into 3D space and second speeds up the training phase of collecting annotated data. The best pose is then found by obtaining a matching score between 3D model projections and the input frame and by determining a global optimization over subsequent frames. It is also shown that the accuracy increases when having multiple viewpoints. Redundant information for 3D reasoning is obtained by using existing 3D models, by incorporating temporal consistency, and by considering multiple viewpoints.

As can be seen from the experiments conducted in this thesis, exploiting redundant information improves the accuracy of all three parts of the pipeline and brings computer vision solutions one step closer towards automatically accomplishing a more robust 3D perception than humans achieve.

Kurzfassung

Menschen leben in einer 3D Welt und haben die Möglichkeit, Schlussfolgerungen dreidimensional zu ziehen. So zum Beispiel werden Verdeckungen und räumliche Aufteilungen von Objekten erkannt, wobei diese Fähigkeiten für die Führung von Objekten und für die Navigation notwendig sind. Projektionen von 3D Szenen variieren in Hinblick auf die eingebundenen Objekte, welche zum Beispiel Fußgänger, Autos, Stühle, Wände, oder Berge sein können. Die Objekte können in Hinblick auf ihre Abmessungen und Farben verschieden konzipiert sein. Szenen werden aber auch unter veränderten Aufnahmebedingungen, wie zum Beispiel verschiedener Beleuchtung, Wetterbedingung oder Tageszeit, erfasst. Projizierte Szenen unterscheiden sich zusätzlich in den Ereignissen oder Handlungen, welche durch oder zwischen Objekten auftreten. Aufgrund einer unendlichen Anzahl an Variationen werden Anwendungen und Algorithmen der Bildverarbeitung so entwickelt, dass sie in der Lage sind, eine spezielle Aufgabe unter bestimmten Bedingungen zu bewältigen und dabei eine festgelegte Genauigkeit erreichen.

Diese Arbeit versucht durch die Verwendung von redundanter Information die spezifische Wirksamkeit eines einzelnen Algorithmus aus der 3D Bildverarbeitung zu umgehen und die Robustheit zu erhöhen. Als Redundanz wird in diesem Zusammenhang die Kombination von mehreren Algorithmen oder Merkmalen bezeichnet. Das 3D Verstehen von visuellen Szenen wird in die drei Aufgaben Kamerakalibrierung, 3D Rekonstruktion und 3D Interpretation unterteilt.

Es werden zwei Methoden zur Selbstkalibrierung mittels Videos für Verkehrsüberwachungsszenen präsentiert. Der erste Ansatz schätzt sowohl intrinsische, als auch extrinsische Parameter mehrerer Kameras in einem Netzwerk durch die Analyse von Fußgängern. Die zweite Methode ist in der Lage, eine Verkehrsüberwachungskamera mittels Fußgänger und Zebrastreifen zu kalibrieren. Redundanz zur Kalibrierung wird durch die Kombination von mehreren zeitlichen Instanzen der Fußgänger und mittels Kombination von statischen (Zebrastreifen) und dynamischen Objekten (Fußgänger) gewonnen.

Künstlich geschaffene Innenräume weisen ebene und texturlose Flächen auf, wo merkmalsbasierte Ansätze nicht in der Lage sind, den 3D Szenenaufbau zu rekonstruieren. Zur Lösung dieses Problems präsentiert diese Arbeit die Kombination von 3D Information aus merkmalsbasierten Techniken mit semantischen Interpretationen. Es wird angenommen, dass ein Bild segmentiert werden kann und jedes Segment einer Ebene entspricht. Die 3D Oberflächennormalen der Segmente werden eruiert und die global beste Lösung für die Normale jedes Pixels wird durch eine pixelweise Optimierung erreicht. Redundanz zur 3D Rekonstruktion wird erstens durch mehrere Segmentierungen und zweitens durch die Kombination von merkmalsbasierten Methoden und semantischen Informationen erreicht.

Die 3D Interpretation beinhaltet die Beschreibung einer Vielzahl von Ereignissen, wie zum Beispiel menschliche Handlungen oder Interaktionen zwischen Objekten. Die 3D Posenschätzung und die 3D Verfolgung stellen dabei die Basis dar. Deshalb werden zwei Methoden zur Posenschätzung, Klassifizierung und 3D Verfolgung von Fahrzeugen, die neben Personen die wichtigsten zu analysierenden Objekte in der Bildverarbeitung sind, vorgestellt. Durch Verwendung von existierenden 3D Modellen wird das Problem der 3D Rekonstruktion aus einzelnen Bildern gelöst und die Trainingsphase effizienter gestaltet, da Trainingsdaten nicht manuell annotiert werden. Die beste Pose wird durch Vergleichen der Projektionen der 3D Modelle mit den Eingabebildern und durch eine globale Optimierung über aufeinander folgende Posen eruiert. Es wird auch gezeigt, dass die Genauigkeit unter Berücksichtigung von mehreren Blickrichtungen gesteigert wird. Die Redundanz zur Interpretation der 3D Szene wird durch die Verwendung von 3D Modellen, aber auch durch die zeitliche Optimierung und die gleichzeitige Analyse von mehreren, synchronisierten Blickrichtungen, sichergestellt.

Die Ergebnisse dieser Arbeit verdeutlichen, dass die Ausnutzung von redundanter Information die Genauigkeit aller drei Teile des 3D Verstehens einer visuellen Szene erhöht und somit die Bildverarbeitung näher an eine robustere Wahrnehmung, verglichen mit der menschlichen Wahrnehmungsfähigkeit, gebracht wird.

“

Nothing in the world is worth having or worth doing unless it means effort, pain, difficulty…

I have never in my life envied a human being who led an easy life.

I have envied a great many people who led difficult lives and led them well. “

Theodore Roosevelt (1858 - 1919)

Contents

1 Introduction **1**
1.1 Motivation 5
1.2 Contribution 6
1.3 Thesis Organization 8

2 Related Work **11**
2.1 From Photogrammetry to 3D Computer Vision 11
2.1.1 Calibration 12
2.1.2 Triangulation 12
2.2 3D Computer Vision 13
2.2.1 Calibration 13
2.2.2 3D Reconstruction 17
2.2.3 3D Reasoning 21
2.3 Innovative Aspects and Context of this Thesis 24
2.4 Summary 26

3 Theory and Background **27**
3.1 2D Image Representations 28
3.1.1 Point Features 28
3.1.2 Edge Features 30
3.1.3 Region Features 32
3.2 3D Computer Vision 35
3.2.1 Perspective Projection 35
3.2.2 Camera Auto-Calibration from Vanishing Points 37
3.2.3 Epipolar Geometry 38
3.3 Optimization and Classification Methods 39
3.3.1 Markov Random Field 39
3.3.2 Random Forest 41
3.4 Summary 44

4 Camera Auto-Calibration from Traffic Participants **45**
4.1 Multiple Camera Auto-Calibration Using Pedestrians 46
4.1.1 Vanishing Point Estimation 46

4.1.2 Camera Parameter Estimation 49
4.1.3 Experiments 50
4.2 Single Camera Auto-Calibration Using Pedestrians and Zebra-Crossings 53
4.2.1 Vanishing Point Estimation 54
4.2.2 Camera Parameter Estimation 59
4.2.3 Experiments 60
4.3 Summary 66

5 3D Reconstruction from Semantic and Geometric Information 67
5.1 Semantic 3D Surface Labeling of Complex Indoor Scenes 68
5.1.1 Proposed Framework 70
5.1.2 Experiments 73
5.2 3D Reconstruction of Complex Indoor Scenes 77
5.2.1 3D Point Cloud Densification Pipeline 78
5.2.2 Experiments 82
5.3 Summary 88

6 3D Reasoning Using Existing 3D Models 89
6.1 Vehicle Classification and Pose Estimation Using Chamfer Matching 91
6.1.1 Single Frame Vehicle Matching 93
6.1.2 Temporal Model Alignment 95
6.1.3 Experiments 99
6.2 Vehicle Classification and Pose Estimation Using Random Forests 104
6.2.1 Discrete Vehicle Pose Definition 105
6.2.2 Random Forests for Pose Estimation 106
6.2.3 Temporal Model Alignment 109
6.2.4 Experiments 109
6.3 Vehicle Classification and Pose Estimation from Two Views 116
6.3.1 Moving Object Detection 117
6.3.2 Training, Classification and Pose Estimation 118
6.3.3 Experiments 120
6.3.4 Quantitative Experiments 121
6.3.5 Qualitative Experiments 121
6.4 Summary 124

7 Conclusion and Future Work 125

Bibliography 129

Nomenclature 145

Notations 147

CHAPTER 1

Introduction

The human visual system is an important feature to navigate through everyday life. It mainly consists of two parts, namely first seeing and second interpreting the perceived scene, where the first part is also known as cognitive vision and the second part is also known as visual perception (i.e. to think and find causal relationships between objects) [Les and Les, 2008, Alexandrov and Gorsky, 1991]. One of the greatest unanswered questions in research is how the human brain is able to understand and interpret surrounding scenes so fast and precisely. The perception system can be split in three parts, namely *detection*, *recognition* and *reasoning* [Les and Les, 2008, Alexandrov and Gorsky, 1991]. All these parts are shown in Figure 1.1 and are described in the following.

Detection As illustrated in Figure 1.1a, people are able to detect a variety of physical objects (e.g. vehicles, chairs, humans, desks, animals, walls, ceilings,...) in 2D images or videos. This ability is even given when the object in question is placed in an unknown environment. For example, when people search for their own, well-known key, they would also find and identify it in an environmental setting which they never saw before. When a person stands in a room where he or she has never been before, it would not be any problem to detect ceilings, walls, the floor and all the objects within the room [Les and Les, 2008, Alexandrov and Gorsky, 1991].

Recognition Apart from detection, humans experience the ability to identify and recognize specific objects, i.e. to differentiate among different objects of various classes or group similar ones when specific grouping and differentiation rules are given (see Figure 1.1b). It is e.g. possible to categorize a vehicle and determine the type of the car (e.g. hatchback, limousine, truck,...). When seeing a specific but novel vehicle type in a commercial on television, people would immediately know the brand of the car if similar vehicles from this brand are known from somewhere else. The grouping rule in this case is made up by the brand. Products of the same brand provide a lot of similar visual properties which are distinctive from the human's point of view. Recognizing, grouping and differentiating may be performed on a variety of different levels of granularity depending on the differentiation / grouping rule which needs to be set beforehand [Les and Les, 2008].

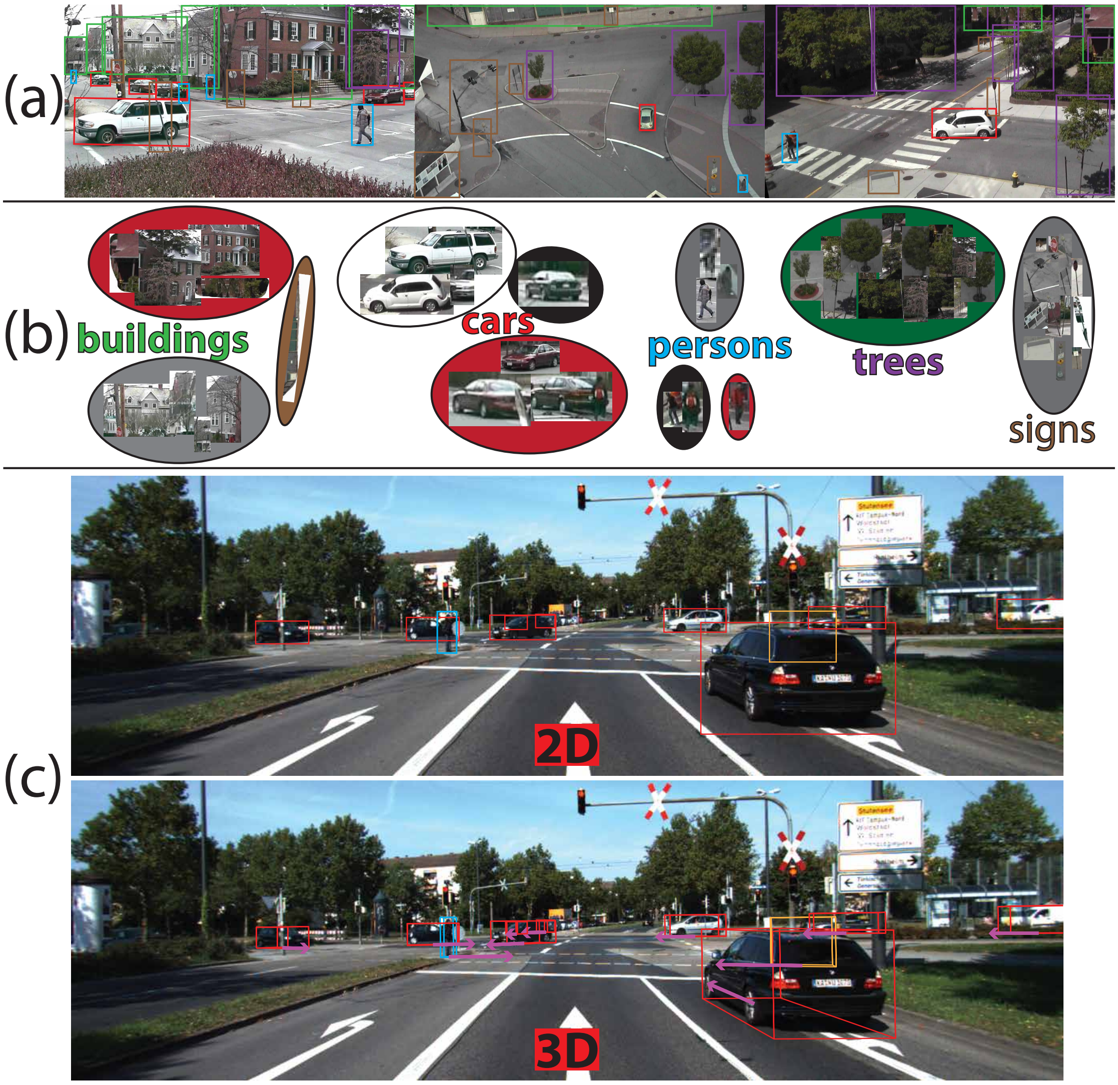


Figure 1.1: A human's visual system is able to (a) detect and localize a number of different physical objects, (b) recognize objects and differentiate among different classes when grouping/differentiation rules are given or set (note that in this example objects are grouped based on color) and (c) infer 3D geometric relationships from 2D images.

3D Reasoning Going even further, humans are also able to localize the detected objects in 3D, to estimate geometric 3D relationships among different objects, to analyze an object interacting with other objects or with its environment and to combine semantic information with the object seen in the image (see Figure 1.1c). Extracting metric information from a 2D image can be established in a fraction of a second and the observed scene may vary between a single microscopic laboratory slide and a whole city seen from a helicopter's point of view. When going from images to videos, the human brain is for example also able to predict the metric movement of a vehicle from one frame to a subsequent one with the slightest effort in order to have the object moving smoothly [Les and Les, 2008, Marr, 1983].

The holy grail of computer vision is to obtain even more reliable and consistent 3D information out of 2D images than humans are able to do by combining computational power and intelligent image analysis software. Operating in 3D space provides the advantage of enabling 3D reasoning. This is for example necessary for the detection of occlusions and occluded objects, for the extraction of metric information, for navigational tasks which humans are experiencing.

The term *3D scene understanding* is used for describing a computer vision framework modeling a human's visual perception system. Such a system consists of three parts, which are described in the following and where the workflow can be seen in Figure 1.2.

Calibration The first step for gathering a 3D representation of the scene is known as camera calibration. Camera calibration describes the task of determining the interrelationship between image and camera plane, which is described by so called intrinsic parameters, as well as the relationship between camera plane and 3D world, also known as the extrinsic parameters. This information is necessary to map between 2D image coordinates and 3D world coordinates. Figure 1.2b illustrates a sample camera position (denoted by $\mathbf{C} = (x, y, z)$) in 3D space, where the 3D reconstruction is represented by the road markings seen in the input image (Figure 1.2a).

3D Reconstruction After knowing the camera position and the relationship between image plane and world coordinate system, the next step is to reconstruct the surrounding scene. Going from 2D to 3D is an ill-posed problem which can be solved (i) by exploiting the principle of triangulation [Hartley and Zisserman, 2003] from corresponding points between two or more camera views, or (ii) by using prior information such as 3D models. Figure 1.2c presents the 3D reconstruction of the scene seen in the input image.

3D Reasoning The last step in a computer vision framework, which models the human perception system in 3D, is known as 3D reasoning. Having a specific set of rules defined for an event of interest, the framework must be able to interpret the scene and to describe the events observed in the scene. Figure 1.2d shows the 3D reconstruction of the environment seen in the input image in combination with vehicles (represented by red boxes), pedestrians (represented by blue boxes), cyclists (represented by yellow boxes) and their moving directions, which are indicated by the magenta arrows.

For this research, each of the three tasks is evaluated independently. Nevertheless, they are all evaluated on image sequences (videos) coming from static surveillance cameras, where the reasons for using this input data are explained in the following.

- **Using videos as input** allows performing e.g. object tracking, object interaction detection, or action detection. As can be seen from these samples, many 3D reasoning tasks are performed over time.
- **Using videos as input** allows exploiting temporal cues in order to achieve both consistency and redundancy over time.
- **Using static cameras** allows obtaining a calibration setting which is valid for a longer time compared to moving ones.

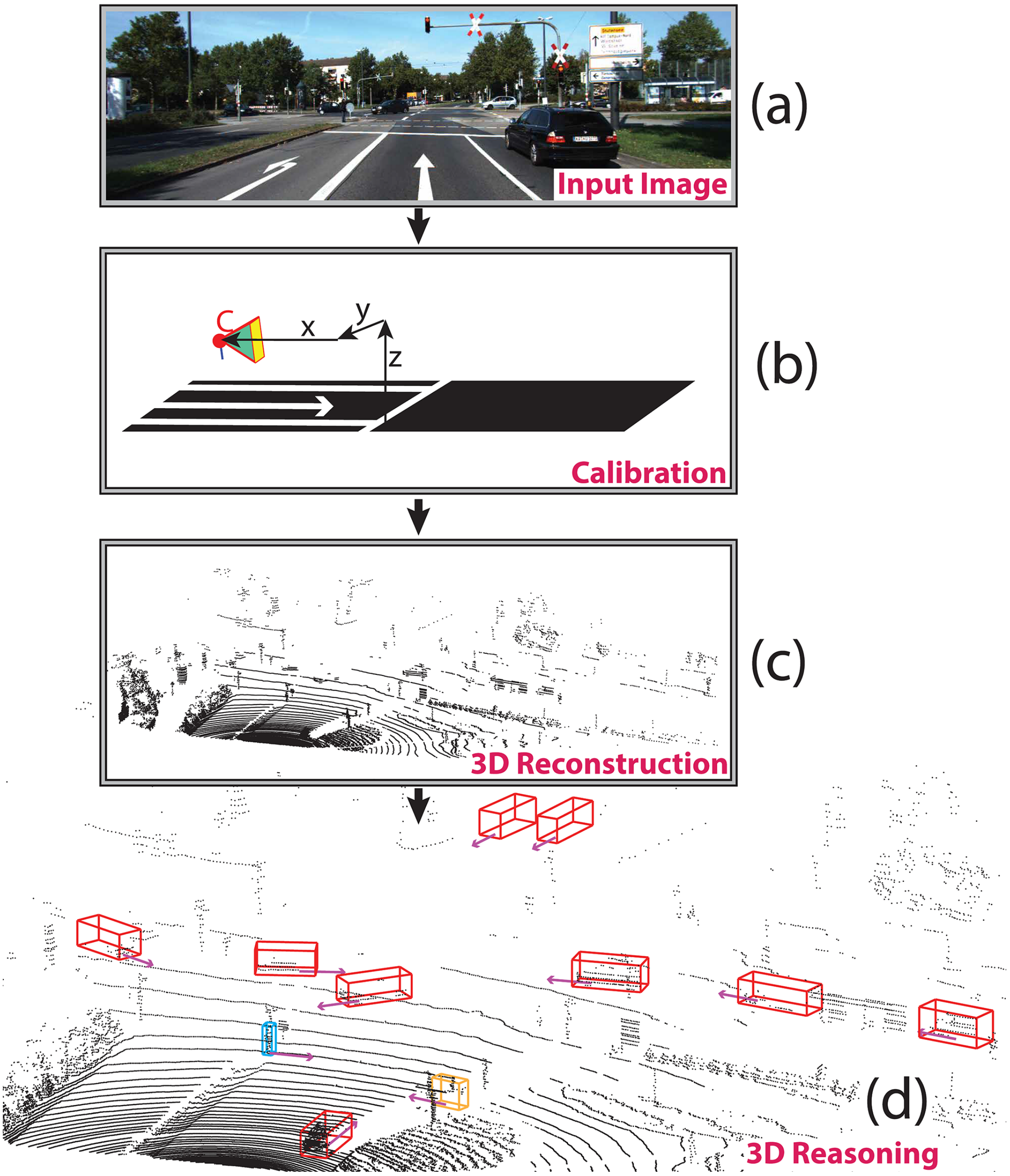


Figure 1.2: (a) Having an input image, a visual computing 3D scene understanding framework consists of three parts, namely (b) camera calibration, (c) 3D reconstruction and (d) 3D reasoning. These algorithmic steps are used in a bottom-up style where each part combines the 3D information resulting from its previous step with the 2D information given from a single input image or from a whole input image sequence.

- **Using surveillance videos** allows exploiting humans and vehicles for 3D scene understanding as these objects are the most frequent ones to be analyzed in surveillance scenarios [Leotta and Mundy, 2011].

1.1 Motivation

Real world scenarios consist of a vast variation of involved objects (e.g. objects providing different dimensions, pedestrians wearing different clothes, cars providing different colors,...), but they can also provide different environmental settings (e.g. lighting, shadows, rain, snow, day, night,...), and do also offer an endless number of different events and actions which can occur (e.g. one object is sitting on top of another one, persons are fighting, persons are shaking hands, pedestrians are carrying bags,...). Due to these variations, computer vision applications are always designed to handle a specific task at a specific environmental setting. In recent years, computational power has been massively increased which gives the advantage that tasks, which computers were not able to solve earlier, can be solved nowadays [Zach, 2007, Hennessy and Patterson, 2011]. The proposed methods exploit this increase in computational power. Novel algorithmic approaches, which bring us one step closer to general 3D scene understanding, are presented.

In this thesis, the term ***redundancy*** is used as a synonym for combining different algorithms, or a number of different cues which are designed to tackle a specific task. Redundant information is then exploited in order to make the final result robust against the described environmental setup changes. When having multiple cues or algorithms, getting a globally correct result can be shifted to a later stage. The tremendous increase of computational power enables gathering multiple hypotheses for a variety of applications and lets the algorithm decide afterwards which hypothesis may be the best one in order to get a final result. As described in the previous paragraph, a 3D scene understanding framework in computer vision consists of three parts, namely camera calibration, 3D reconstruction and 3D reasoning. This work exploits redundant information gathered from different sources for all the three parts of the pipeline and returns a final result at the end of each stage. Particular goals of this work can be summarized as follows.

1. Development of a camera auto-calibration method which exploits 3D features and 3D principles of objects shown in the image. By using redundant data, coming from multiple objects or multiple instances of an object, a practical and efficient calibration framework with no (or little) user interaction is established.

2. Development of a 3D labeling and 3D reconstruction algorithm. The goal is to obtain a 3D labeling and 3D reconstruction framework for scenes where no (or less) texture is available. Texture is needed for conventional reconstruction methods which are based on localizing and matching discriminative features. Having less or no texture means that these features get less discriminative and incorrect matches may occur among different viewpoints. This problem is tackled by combining redundant information obtained from multiple cues.

3. Development of a 3D reasoning framework. One specific task in the domain of 3D reasoning, namely vehicle pose estimation and 3D tracking is performed. Pose estimation and 3D tracking is chosen as this is the initial step towards full 3D reasoning of a scene, as e.g. stated in [Geiger et al., 2012, Geiger et al., 2011]. As a robust pose classifier must be trained on a huge number of labeled and clustered training images, these images are generated synthetically from 3D Computer-Aided Design (CAD) models in order to speed up the training phase of manually collecting and labeling real world images. Apart from speeding up the training phase, using 3D models also enables to overcome the ill-posed problem of going from 2D images to the 3D world and enables reasoning about hidden parts of the object in a scene as the dimensions and other features (e.g. color, shape,. . .) of the corresponding 3D model from an object seen in 2D images are known in advance. This research also answers the question whether or not aggregating information from multiple cameras increases the pose estimation results.

1.2 Contribution

There is an endless number of different influences which change the representation of a 3D scene on a 2D image plane in computer vision. Each of the developed methods described in literature provides advantages as well as drawbacks when it performs a specific task, if it is applied on specific image regions, or with a specific environmental setting. The main contribution of this research lies in showing that by combining different methods and algorithms related to 3D scene understanding, which in the rest of this work is referred to as ***redundancy***, the final outcome of the pipeline can be improved. In detail, these contributions are described as follows, where major parts are presented in scientific publications and presentations within the last three and a half years.

1. **Exploiting Redundancy for Calibration:** Camera calibration can be established using vanishing points or other features obtained from a single image or from multiple ones. This work exploits the prior knowledge that a person, who is observed for a sequence of frames, does not change its height and uses this information for gathering intrinsic as well as extrinsic parameters automatically. Calibrating a whole camera network from pedestrians only and calibrating a single traffic surveillance camera from a combination of pedestrians and zebra-crossings is shown. It is demonstrated in the experiments section that an improved performance is achieved compared to state-of-the-art methods.

   The redundant information exploited for calibrating a camera is obtained by combining (i) multiple instances of the same pedestrian over time and (ii) static (zebra-crossings) and dynamic (pedestrian) objects.

   The proposed work dealing with camera calibration was mainly presented in two publications. [Hödlmoser and Kampel, 2010] describe the calibration from pedestrians only, [Hödlmoser et al., 2011b] illustrate the method using a combination of pedestrians and zebra-crossings.

2. **Exploiting Redundancy for 3D Surface Normal Labeling and 3D Reconstruction:** 3D Reconstruction is usually done by generating a sparse point cloud obtained by triangulation followed by a densification [Furukawa and Ponce, 2007]. In case of challenging indoor environments, this is not possible anymore because there may be incorrect matches between corresponding camera views due to similar features obtained from flat and textureless surfaces (e.g. walls, floors) [Häne et al., 2012]. To compensate this lack of feature matches, the semantic information available in 2D images is exploited to estimate both a corresponding 3D position and a 3D surface normal for each pixel. A semantic classifier is therefore applied on a single segmented image in order to get a likelihood for a segment providing one of the surface normals within a discrete set of them. A segment is assumed to be modeled by a planar patch. To improve the accuracy of the labeling step, the presented method combines multiple segmentation algorithms in order to come up with a stronger classifier for both 3D surface normal labeling and 3D reconstruction in challenging environments. The final surface normal label and the corresponding 3D point for each pixel are then obtained by evaluating the spatial smoothness between neighboring pixels and solving it in an optimization framework. As can be seen in the experiments section, using redundant information allows obtaining more accurate results compared to state-of-the-art methods in terms of both labeling and reconstruction.

   The redundancy for 3D labeling and 3D reconstruction is therefore gathered by combining multiple cues coming (i) from a variety of different shaped segments, obtained from various segmentation methods and (ii) from combining conventional feature based matches with semantic patch-based 2D information.

   The content dealing with this contribution emerged from the methods described in [Hödlmoser and Mičušík, 2013] which describes labeling the image, and the workflow outlined in [Hödlmoser et al., 2013a] which illustrates the 3D reconstruction from these labeled images. Additionally, requirements for running a 3D reconstruction pipeline on an embedded device, which is not directly related to this research, are published in [Hödlmoser et al., 2011a].

3. **Exploiting Redundancy for 3D Reasoning:** The last part of a scene understanding framework is a broad field covering a vast variation of events. This work picks out a specific task and deals with 3D pose estimation and tracking of objects as this is the basis for all 3D reasoning algorithms, as stated in [Geiger et al., 2011, Geiger et al., 2012]. More specifically, vehicles are used since they are the second most important objects to be analyzed in computer vision besides persons [Leotta and Mundy, 2011]. The advantage of analyzing vehicles over persons is that the shape and texture of a vehicle is more discriminative than the shape of a person [Leotta, 2010]. This advantage even increases if the silhouettes are seen from a larger distance (5-30 meters) which is usually the case in surveillance scenarios. The best matching pose for a given vehicle is determined in an image sequence. It is therefore proposed to exploit redundancy, rank all possible poses for each frame and let an optimization framework afterwards decide which pose to be the best for each frame by also considering subsequent poses. In order to get multiple pose hypotheses for each frame, multiple images showing a vehicle from different poses must

be collected. To overcome the problem of collecting and sorting real world images based on the vehicle's pose, existing 3D models are exploited. It is also shown that aggregating information from multiple views increases the accuracy of determining the pose and the class of the vehicle. It is shown that the accuracy increases compared to the result coming from analyzing each camera separately.

The redundancy for 3D reasoning is obtained by (i) gathering synthetic images from 3D models rendered with different poses, (ii) combining information about the pose and class over time in an image sequence and (iii) combining results from different viewpoints simultaneously.

The core algorithms of the parts dealing with 3D reasoning are presented in three publications. [Hödlmoser et al., 2011c] present a vehicle pose estimation algorithm using edge features, [Hödlmoser et al., 2012] describe pose estimation using a classifier trained on multiple vehicle models, and [Hödlmoser et al., 2013b] tackle simultaneous pose estimation from multiple viewpoints.

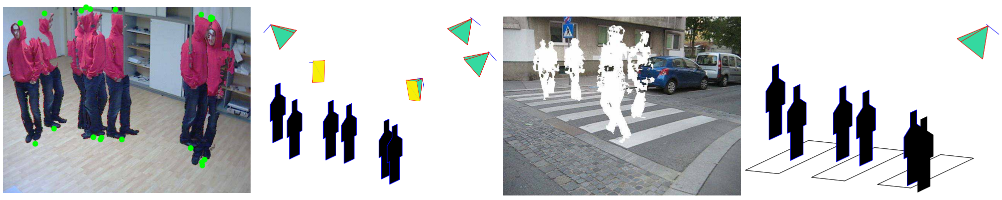
1.3 Thesis Organization

The contributions are directly related to the research fields calibration, 3D reconstruction and 3D reasoning and are therefore divided into three technical chapters (4-6). Each of the three chapters holds an outline of the proposed method and an experimental evaluation of the proposed approach, where outcomes of the pipeline are compared to results obtained by state-of-the-art algorithms. Before that, Chapter 2 gives an overview on related work, Chapter 3 describes the basic principles needed to explain the implementations outlined in the following chapters. The rest of the work is therefore organized as follows.

Chapter 2 presents an overview on state-of-the-art methods related to the three steps needed in computer vision to extract 3D information out of 2D images, namely calibration, 3D reconstruction and 3D reasoning and shows how they are all related to this research and how the content of existing methods differ from the proposed framework.

Chapter 3 outlines the fundamental principles which are used in Chapters 4-6. It shows how image and video data are described and represented in computer vision applications. Different image description features and representative implementations are outlined. Next, principles about the interrelationship between image plane, camera plane and 3D world coordinate system are described and fundamentals of 3D computer vision are explained. Computer vision algorithms, which find the final solution from multiple hypotheses, are based on a variety of optimization and classification methods. The goal of this evaluation is to get the most likely, globally optimized and best fitting result from all these hypotheses. Therefore, Chapter 3 describes an optimization method and a classification algorithm, which are used in the accompanying implementations for this work, namely Random Forests (RFs) and Markov Random Fields (MRFs).

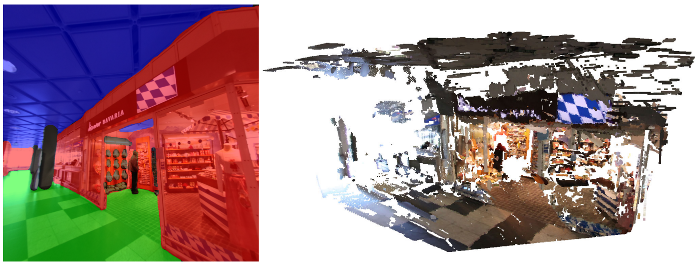
Chapter 4 describes two novel auto-calibration methods for finding a camera's parameters. Auto-calibration can be established after the extraction of three vanishing points which are or-



(a) (b)

Figure 1.3: Calibration using (a) pedestrians only and (b) pedestrians and a zebra-crossing.

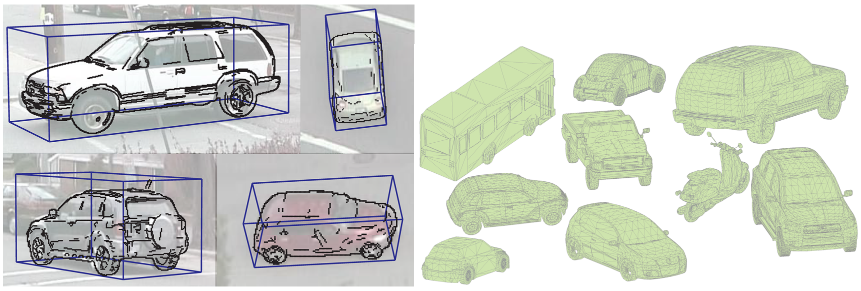
thogonal to each other. The first approach proposes to obtain these vanishing points from a walking pedestrian. Having at least two images of a walking pedestrian from different time instances, the vanishing points can be extracted by using top and bottom points of the person. Multiple surveillance cameras within a network are calibrated using this information. Figure 1.3a illustrates the outcome of the approach. From an input image, showing multiple instances of a pedestrian with extracted top and bottom points (left), camera positions, which are observing the same scene with pedestrians walking on the ground plane, are determined in 3D space (right). The second approach describes the calibration of a single static traffic surveillance camera, where vanishing points are extracted from both static and dynamic objects. The advantage of static objects is that the results are more robust against outliers, the advantage of dynamic objects is that the method is more flexible. Pedestrians (dynamic objects) are walking on or near zebra-crossings (static objects) in a traffic scenario. This knowledge is then exploited for calibration. It is shown that the results can be improved by combining the two object types over using pedestrians only. The approach is shown in Figure 1.3b. Multiple instances of a pedestrian are extracted (drawn as white silhouettes in a single background image which contains a zebra-crossing (left)) and the corresponding 3D locations of pedestrians, zebra-crossing, and camera are determined in 3D space (right).



(a) (b)

Figure 1.4: (a) 3D surface orientation labeling. (b) Reconstruction using the labeled image.

Chapter 5 presents a novel 3D labeling and 3D reconstruction framework. In man-made environments, conventional feature-based 3D reconstruction pipelines suffer from incorrect or missing matches and therefore provide incomplete 3D models. The chapter describes how to improve the reconstruction results by incorporating semantic 2D information, where an image is segmented and each segment is assumed to be described by a planar patch. The surface normal and a corresponding 3D point for each pixel is then estimated by solving an optimization framework to enforce spatial smoothness between neighboring pixels. It is also shown in the experiments of this chapter, that the accuracy of these results can be even further increased by estimating the surface normal of this patch and combining multiple segmentation methods. Figure 1.4a shows the result of the proposed approach for labeling the 2D image into *ground plane* (green), *ceiling*(blue), *vertical*(red). Image regions which are excluded from labeling are marked black. This image is then used to obtain a 3D reconstruction of the scene (see Figure 1.4b).



(a) (b)

Figure 1.5: (a) 3D vehicle classification and pose estimation using (b) projections of existing 3D models.

Chapter 6 explains a novel method for solving specific task within this field, namely pose estimation and classification of vehicles. This task is chosen since it is used as a basis for 3D reasoning frameworks [Geiger et al., 2011, Geiger et al., 2012, Leotta and Mundy, 2011]. Two different approaches for vehicle tracking are outlined in this chapter. Both rank all possible poses for each frame using multiple projections of different 3D models. The first method does this ranking based on comparing the edges within the image with the edges resulting from rendering the models. The second approach explains another ranking strategy, where the score is obtained by training an RF on all the 3D model projections. Both methods then find the best matching pose for each frame by evaluating the poses for each frame in combination with previous and subsequent poses in the video. This chapter also shows that estimating a vehicle's pose and class from two views simultaneously increases the accuracy over analyzing each viewpoint separately. Figure 1.5a shows sample vehicle classification and pose estimation results using projections of existing 3D models which are shown in Figure 1.5b.

Chapter 7 gives a summary and an outlook to possible ideas or research directions for future work. These ideas help to improve the proposed framework in further developments.

CHAPTER 2

Related Work

3D computer vision methods evolved from the early photogrammetric inventions before and also after the introduction of the principles of photography. The following review presents a historical review ranging from the early beginnings of photography in combination with photogrammetry to modern 3D computer vision and its analogy to the human visual perception system.

2.1 From Photogrammetry to 3D Computer Vision

The main goal of photogrammetry is to obtain a precise mapping and to achieve accurate measurements [Clarke and Fryer, 1998]. Photography (greek for *'drawing with light'* was first introduced in 1839 by Sir John Herschel, who was an English mathematician and astronomer [Schaaf, 1992]. This invention brought the advantage of obtaining a correct perspective projection of a 3D scene which was not perfectly the case with paintings.

Gathering 3D information from 2D photographs was first used to create topographic maps and to obtain terrain models. In 1726, Kappeler compiled a topographic map of a Swiss mountain range using perspective drawings. It is not documented how he generated these drawings. Therefore, the term *photogrammetry* was officially introduced in literature by [Laussedat, 1899] which let Aime Laussedat become the *father of photogrammetry* [Sturm, 2011]. At the beginning, the approach needed perspective drawings in order to extract 3D information. At the introduction of photography, these drawings were then replaced by photographs. This invention was first tested in 1849 and enabled measuring heights and distances as well as planar point triangulations. In the following decades, there were several inventions which enabled using multi-camera systems having well defined properties (e.g. fixed distances and orientations among cameras). In [Tissandier, 1886] for example the authors describe a multi-camera system consisting of 7 fixed mounted cameras attached to a balloon which was used to obtain images from an aerial point of view. [Scheimpflug, 1904] presents the principles of how the lens and the back of a camera must be tilted in order to focus a plane which is not parallel to the image plane.

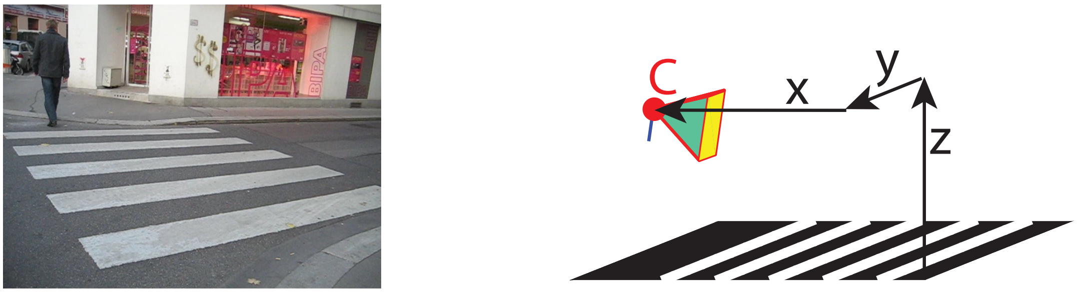
In order to work with cameras from an arbitrary point of view having arbitrary intrinsic parameters, the next step is to obtain these properties from 2D images in combination with calculating the 3D structure of the projected scene. Two steps are required to get these properties namely calibration, consisting of finding a camera's intrinsic parameters its pose in 3D space and its relative orientation to a camera providing a different viewpoint, and triangulation which is also known as estimating the 3D structure of the scene [Sturm, 2011].

2.1.1 Calibration

In photogrammetry, the problem of calibration is divided into three parts which are (i) finding the intrinsic parameters, (ii) estimating the camera pose in 3D space and (iii) calculating the relative pose between two cameras. Camera self-calibration for both computer vision and photogrammetry is defined as estimating these parameters from only the information available in the images [Hartley and Zisserman, 2003]. In 1892, Meydenbauer explained that finding a camera's intrinsic parameters is possible for a rotating one [Meydenbauer, 1892]. Finsterwalder, a German mathematician, described that the task of camera calibration from a rigid object, where the camera is moving with general motion, is also solvable [Finsterwalder, 1899]. Both methods assume to have the principal point available and only determine the focal length. They additionally assume that the camera is rotated around either its vertical or horizontal axis. Additionally to obtaining the focal length, the method proposed by [Sutor, 1939] also recovers the principal point from rotating cameras. Mathematical models for estimating a camera's intrinsic parameters were published by [Brandenberger, 1948] and [Brown, 1956].

2.1.2 Triangulation

[Finsterwalder, 1899] also provided a solution for reconstructing the 3D scene from two projective images using four coplanar and two other points. [Kruppa, 1913] outlines that the relative pose between two images taken from two different points of views can be estimated when five corresponding 2D points and the intrinsic parameters are known. These developed *Kruppa Equations* are still used in today's computer vision applications. The principles of epipolar geometry, which is in detail also described in Section 3.2.3, was first introduced by [Hauck, 1883]. The epipolar geometry describes the relationship between two camera's centers, the corresponding image planes, their epipoles, corresponding points and the triangulated 3D point [Hartley and Zisserman, 2003]. Horst von Sanden then described how to calculate the epipoles needed for triangulation in [von Sanden, 1908]. [Thompson, 1968] presents the first approach to describe the relative orientation between two camera views by an equation. Based on that contribution, [Longuet-Higgins, 1987] showed that the principles of the epipolar geometry can be described by a single matrix. In computer vision, the matrix is called essential matrix. Following [Longuet-Higgins, 1987], it can be estimated by using eight corresponding points between the two views. The invention of this 3x3 matrix enabled efficient solving the problem of finding the 3D structure from 2D images for computer vision applications. [Faugeras, 1992] and [Hartley, 1992] propose two methods for 3D reconstruction and camera calibration from two uncalibrated images.



(a) (b)

Figure 2.1: Camera calibration is the determination of a camera's intrinsics (mapping between image and camera plane) and extrinsics (mapping between camera plane and 3D world). (a) Input image. (b) Camera localization in 3D space.

2.2 3D Computer Vision

Different to the field of photogrammetry, the main objectives of computer vision are reconstruction and recognition [Ma et al., 2003]. The research field of computer graphics studies the problem of how 3D objects and their properties can be modeled on 2D images. This modeling is established by exploiting the knowledge of physics. A light ray coming from a source is reflected by the surface of an object, received by the human eye or the lens of a camera and projected onto the retina or a 2D image plane [Sonka et al., 2008]. Doing the inverse task is tackled by computer vision applications. The 3D reconstruction and the properties (e.g. shape, color,...) of a scene and all associated objects, which are projected onto a 2D image plane, should be recovered. Performing 3D scene understanding in computer vision can be divided into three steps which are calibration, 3D reconstruction and 3D reasoning.

2.2.1 Calibration

Similar to photogrammetry, camera calibration in computer vision is known as the determination of the interrelationships between the 3D world coordinate system and the one described by a camera (extrinsic parameters) and between the camera model and the image coordinate system (intrinsic parameters) [Hartley and Zisserman, 2003]. The outcome of the calibration process is visually demonstrated in Figure 2.1. The input image, shown in Figure 2.1a is used to obtain a corresponding 3D pose of the camera's center point $\mathbf{C} = (x, y, z)$ within the scene, which is shown in Figure 2.1b. When talking about camera calibration in this research, it is always meant to be a calibration using the conventional pinhole camera model. Calculating these intrinsic and extrinsic parameters is a preprocessing task for 3D computer vision applications which range from the area of surveillance networks (e.g. security scenarios, or ambient assisted living applications [Zhang et al., 2012]) to autonomous robots and ubiquitous network robotic devices [Tsai et al., 2011]. Having the 3D information can also be helpful for the reconstruction of a scene, as can be seen in [Leibe et al., 2007a] and [Hödlmoser et al., 2013a]. Knowing the 3D information of a scene allows video and image metrology, as described by [Criminisi et al., 2000] or [Guo and Chellappa, 2010]. It also enables classification and pose recovery of vehicles using 3D CAD models, as proposed by [Buch et al., 2009], or [Yoneyama et al., 2005].

According to [Hartley and Zisserman, 2003], camera calibration methods can mainly be divided into two approaches, namely conventional camera calibration using a known calibration object and camera auto-calibration which is also known as camera self-calibration. When the dimensions of an object are known, the 3D information of a scene can precisely be extracted by establishing correspondences between different views showing the same calibration object. As described in Section 2.1.1, in case of auto-calibration, no calibration object is needed in order to calibrate a camera from uncalibrated images.

The disadvantage of conventional camera calibration methods is the extra time needed to obtain a tailored calibration object and to perform the calibration procedure which can then only be done in an offline fashion. As auto-calibration methods do not use known 3D points, the main disadvantage of these methods is that the camera parameters are less accurate than parameters obtained by using conventional methods, as can be seen in [Lv et al., 2006].

2.2.1.1 Conventional Camera Calibration

Conventional camera calibration is performed offline by using a pre-defined and well-known calibration pattern. This method has also been used for photogrammetry [Faig, 1975, Brown, 1971]. In practice, the generation of an adequate calibration pattern is not an easy task because its size should be tailored to meet the requirements for obtaining a certain accuracy in the camera calibration process (i.e. the projection of the calibration pattern must fill most parts of the 2D image). Producing such a tailored calibration pattern and performing an offline calibration procedure takes some extra time compared to executing camera auto-calibration.

[Tsai, 1987] introduced the first computer vision method for camera calibration from known points. They first transform the 3D points in a camera coordinate system. Then, camera coordinates are transformed to image coordinates, lens distortion is corrected and depending on the sensor size and the image resolution, the final image coordinates are estimated. [Heikkilä and Silvén, 1997] propose to estimate a camera's parameters by first using a linear closed-form solution followed by a non-linear least-squares estimation. The method can be used for both calibration grids in 2D and 3D. [Zhang and Zhang, 2000] present a camera calibration method based on [Heikkilä and Silvén, 1997] but using a planar calibration pattern which in practice is a conventional chessboard pattern. This calibration pattern must be seen from multiple views in order to estimate the projective transformation between corresponding 3D points. This method also first exploits a closed-form solution followed by a non-linear refinement to get the camera parameters. A similar method is presented in [Sturm and Maybank, 1999] which is able to calibrate multiple cameras having different intrinsic parameters from a calibration pattern that is seen from at least two different views. All the described calibration techniques are combined in a single Matlab framework called Camera Calibration Toolbox for Matlab (CCTfM) [1]. Calibrating a whole network of non-overlapping cameras with the help of a mirror is presented in [Kumar et al., 2008]. The problem that the calibration pattern needs to be seen from all the devices in a network can be overcome by using a planar mirror. The calculated mirrored camera poses then

---

[1] `http://www.vision.caltech.edu/bouguetj/calib_doc/`, last retrieved on 21.03.2013.

also describe the real camera poses. From an application point of view, a human pose estimation procedure supporting multiple cameras is presented by [Kurillo et al., 2009]. A framework for conventional camera calibration for teleimmersion purposes using corresponding points of interest is proposed. Intrinsic parameters are obtained by using a conventional method based on a defined chessboard pattern. For finding the cameras' extrinsic parameters, two LED markers having a fixed distance and defining a virtual calibration object are used. After the marker is detected on the image plane, the fundamental matrix between two cameras is determined and the relative rotation is calculated. Another camera calibration technique is presented by [Martynov et al., 2011] which exploits an inverted workflow of conventional camera calibration methods. The calibration is done by iteratively adjusting projected markers to be used as input for the refinement and exploiting planar homographies in order to speed up the process.

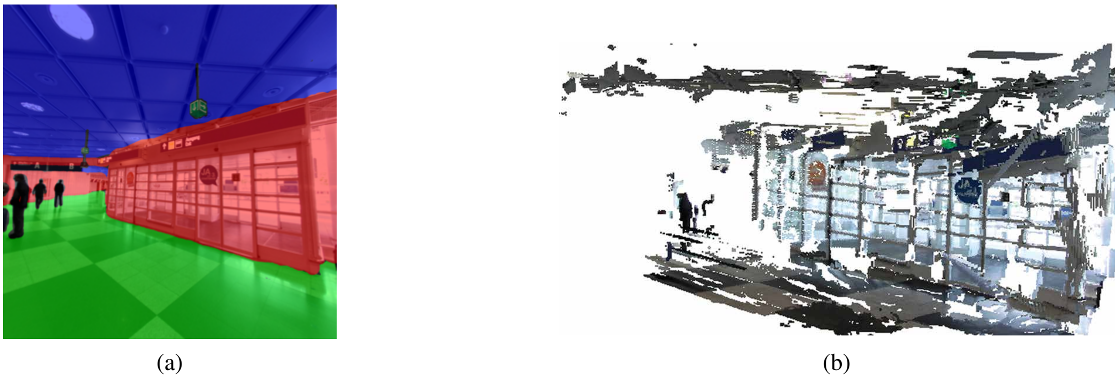
2.2.1.2 Camera Auto-Calibration

Camera auto-calibration is the determination of camera parameters from uncalibrated images of unstructured scenes. The main difference to conventional camera calibration methods is that no 3D points must be known in advance [Hartley and Zisserman, 2003]. Due to this higher degree of uncertainty, the results obtained by using auto-calibration are more sensitive to outliers than the ones obtained by conventional methods. The first solutions for the concept of auto-calibration are based on a single moving camera which means that the intrinsics are constant and at least two images are needed. A self-calibration method, which exploits the Kruppa equations, for camera poses with constant intrinsic parameters was introduced by [Maybank and Faugeras, 1992]. The method was then developed further by [Luong and Faugeras, 1997], where point correspondences from three images and the fundamental matrices are used for self-calibration and 3D reconstruction. A system of polynomial equations is derived from the Kruppa equations which is then solved by numerical continuation. [Pollefeys and Van Gool, 1997] proposed a method for self-calibration by estimating a plane at infinity and calculating constant camera intrinsics from the Image of the Absolute Conic (IAC). [Triggs, 1997] then presented the Absolute Quadric for recovering the intrinsic parameters. This concept was then used by [Pollefeys et al., 1998] for estimating varying intrinsic parameters from multiple images.

[Beardsley and Murray, 1992] first described the extraction of intrinsic camera parameters from three vanishing points. By the determination of three vanishing points within an image, the principal point and the focal length can be recovered sequentially. In [Cipolla et al., 1999], camera calibration using three vanishing points of an image is proposed. Their semi-automatic auto-calibration method uses building façades to determine three vanishing points. The user needs to select a set of parallel image lines in order to search for a correct vanishing point initialization. After initialization, the intrinsic parameters are recovered. The relative rotation between a camera pair is estimated using the calculated points on the plane at infinity and the translation is calculated by using further points of interest in a scene.

Calibrating a camera using a pedestrian was first introduced by [Lv et al., 2006]. Top and bottom points are determined in the images and three vanishing points are extracted. A geometric solution is used to obtain the intrinsic parameters afterwards. The extrinsic parameters with respect to one camera are calculated to compute the complete pose of a camera within a defined world coordinate system. Another approach for calibrating a camera from a pedestrian was introduced in [Krahnstoever and Mendonça, 2005]. Camera parameters are estimated by using a foot-to-head plane homology in combination with a Bayesian framework which is able to handle measurement uncertainties and outliers. In [Krahnstoever and Mendonça, 2006], it is shown that incorporating information about the motion of people observed for several frames can help to get a robust solution to the camera auto-calibration problem. A similar approach to [Lv et al., 2006] is proposed by [Junejo, 2009], where pedestrians need to walk on uneven terrains in order to extract camera parameters. A direct method for auto-calibrating a camera by observing a pedestrian is presented in [Kusakunniran et al., 2009]. Based on top and bottom points of a walking human, the pose matrix is estimated column by column by exploiting cross-ratio constraints. Having two vanishing points and a vanishing line, the third vanishing point can be calculated. The vanishing points are then directly used to calculate the pose of the camera. A camera calibration method for two cameras only is published by [Chen et al., 2007]. Recovering the intrinsic parameters is based on the algorithm presented by [Junejo, 2009]. The relative orientation is afterwards calculated using all vanishing points and the infinite homography. The vanishing points do not need to be orthogonal to each other and the intrinsic parameters are estimated by obtaining the infinite homography from all the extracted vanishing points. [Mičušík and Pajdla, 2010] published a surveillance camera calibration method based on foot and head points of pedestrians. By introducing the Quadratic Eigenvalue problem, extrinsic and intrinsic parameters are extracted as well as a foot-head homology is estimated.

[Pflugfelder and Bischof, 2010] presented a method for calibrating non-overlapping cameras from pedestrian observations by exploiting vanishing points from man-made environments and a non-linear optimization. Extrinsic parameters as well as smooth pedestrian trajectories in 3D space are calculated by using Singular Value Decomposition. [Zhang et al., 2008b] explain an auto-calibration method using the orientation of pedestrians and vehicles. The method extracts a vertical vanishing point from the main axis direction of their trunk, perpendicular to the ground plane. Two horizontal vanishing points are extracted by investigating the moving cars. Calibrating a camera network from a person's silhouette is described by [Sinha et al., 2004]. The RANdom SAmple Consensus (RANSAC) based method estimates a camera's parameters from the motion of a moving silhouette. [Puwein et al., 2011] published a method for calibrating a camera network from sports broadcasts. Correspondences among views are established in a coarse to fine fashion by collecting and matching both Maximally Stable Extremal Regions (MSER) and Scale Invariant Feature Transform (SIFT) features. The images used for the calibration procedure show hockey, football, or basketball games. The approach recovers both extrinsic and intrinsic parameters of a surveillance camera. In [Furukawa and Ponce, 2008] a camera's parameters are roughly found by initial feature matches between multiple views which are refined by finding additional matches in the neighborhood of the initial ones. The method additionally generates a 3D model of the scene. [Zhang et al., 2011] present a method for estimating the intrinsic parameters and the lens distortion parameters of a surveillance camera from low-rank textured images. A closed-form solution for gathering a camera's parameters was first introduced by [Wildenauer and Hanbury, 2012]. A monocular camera is calibrated by using vanishing points from Manhattan World scenarios. A RANSAC-based approach for estimating focal length and three vanishing points from a set of four lines is proposed. The resulting parameters are further refined by exploiting a Maximum Likelihood estimator.



(a)

(b)

Figure 2.2: (a) Humans do not have any troubles to estimate the 3D layout of the scene and label the 2D image based on the predefined classes *ground plane*, *ceiling* and *vertical*. (b) The 3D reconstructed scene obtained from the labeled image.

2.2.2 3D Reconstruction

When humans are looking on an image, they are immediately able to interpret the scene due to capturing the semantic and geometric context. Even when looking at the image shown in Figure 2.2a for the first time, a human brain does not have any troubles to assume the 3D layout of the scene without having any further information. Humans are able to roughly gather the position of the viewpoint where the image was taken, to localize the ground plane and the ceiling, to find vertical wall segments and even to distinguish between inside and outside the building although there is a reflexive door surface in the middle of the image. As can be seen, obtaining the 3D layout, detecting occluded objects and even gathering the 3D relationship among objects in the scene is something which is obviously beyond the visible 2D scene. A rough 3D surface normal layout estimation is shown in Figure 2.2a, where different colors represent the different classes *ground plane* (green), *ceiling* (blue) and *vertical* (red). Image parts, which are not known, are labeled as black regions. The 3D reconstruction of the scene using this labeled image can be seen in Figure 2.2b.

For the last few years, transferring this human ability to computers is one of the grand challenges in computer vision. Having the 3D geometry of a scene would help applications placed on top of this knowledge. Assuming a given ground plane is for example necessary for initializing a tracking sequence, as in [Shitrit et al., 2011, Ess et al., 2009, Leotta and Mundy, 2011], but it also increases the accuracy in tasks such as autonomous robot navigation [Leibe et al., 2007b] and automatic object manipulation [Petrovskaya et al., 2006]. Reconstructing a scene can be established by using a single image in combination with additional information (e.g. existing 3D models), or by using feature matches coming from multiple images.

2.2.2.1 Single View Labeling-based 3D Reconstruction

Reconstructing 3D models from a single 2D image is an ill-posed problem. Nevertheless, several methods split an image into segments, use semantic information, and choose a label for each pixel from a set of geometrically meaningful classes in order to overcome this problem. [Hoiem et al., 2005a, Hoiem et al., 2005b] both present an approach for automatically constructing a rough 3D model from a single 2D image. This is established by learning a statistical model of surface normal label classes. Extracting 3D information from a single 2D image showing a Manhattan world indoor environment is described by [Delage et al., 2006]. They assume to have a calibrated camera, extract edges, the ground plane and surface orientations from the images and obtain a final labeling by solving an MRF. Labeling the 3D layout of a scene is published in [Hoiem et al., 2007]. By combining multiple 2D cues (color, texture and perspective features of a patch) the classifier is trained on multiple indoor and outdoor still images using boosted decision trees. Each image is segmented using the approach presented by [Felzenszwalb and Huttenlocher, 2004]. For getting a higher accuracy, the method merges segments to obtain different segments in terms of size and shape. [Barinova et al., 2008] present an algorithm based on [Hoiem et al., 2007] for obtaining a 3D model out of a single 2D image. The 3D model is created by combining a number of patches belonging to vertical structures and patches on the ground plane. By combining the search for the ground-vertical boundary with geometric 3D modeling constraints, the best model is obtained by exploiting a Conditional Random Field (CRF). [Gould et al., 2009] obtain a holistic representation of the scene by finding semantic and geometric meaningful and consistent regions in the image. Mean-shift segmentation and merging patches in the segmentation process is used to obtain better results. Another segmentation and depth estimation framework using an MRF and semantic segmentation using meanshift is presented by [Liu et al., 2010a]. Bedroom sampling on still images by incorporating the geometric features of objects within a room is used to obtain a rough layout of the room by [Pero et al., 2011] and [Pero et al., 2012].

2.2.2.2 Multiple View Labeling-based 3D Reconstruction

Having multiple images enables generating a sparse 3D point cloud of the scene by using Structure from Motion (SfM) approaches. Labeling each pixel with a geometrically meaningful label using multiple images is established by combining semantic information with 3D information coming from the triangulated point cloud. [Brostow et al., 2008] published an approach for labeling 2D video sequences from outdoor scenes using sparse 3D point clouds. By using Delaunay Triangulation, a relief mesh is set up from the 3D points. Based on the orientation and location of the triangles, the traffic scene is segmented. For this approach, the features are purely calculated on the geometric observations. [Flint et al., 2011] present a method which incorporates stereo, monocular and 3D features to iteratively help in the segmentation process of indoor videos. Bayesian filtering with motion cues of possible hypotheses of box layouts of input videos showing indoor scenes is presented by [Tsai et al., 2011]. An approach using videos of street scenes for segmenting the scene is presented in [Xiao and Quan, 2009]. Reconstructed 3D points are manually but not perfectly labeled to help in the 2D segmentation process which is performed using an MRF. [Floros and Leibe, 2012] present an approach which combines spatial and temporal smoothness terms between corresponding pixels in a single higher-order CRF in order to obtain an image segmentation formulation.

2.2.2.3 Single View 3D Reconstruction

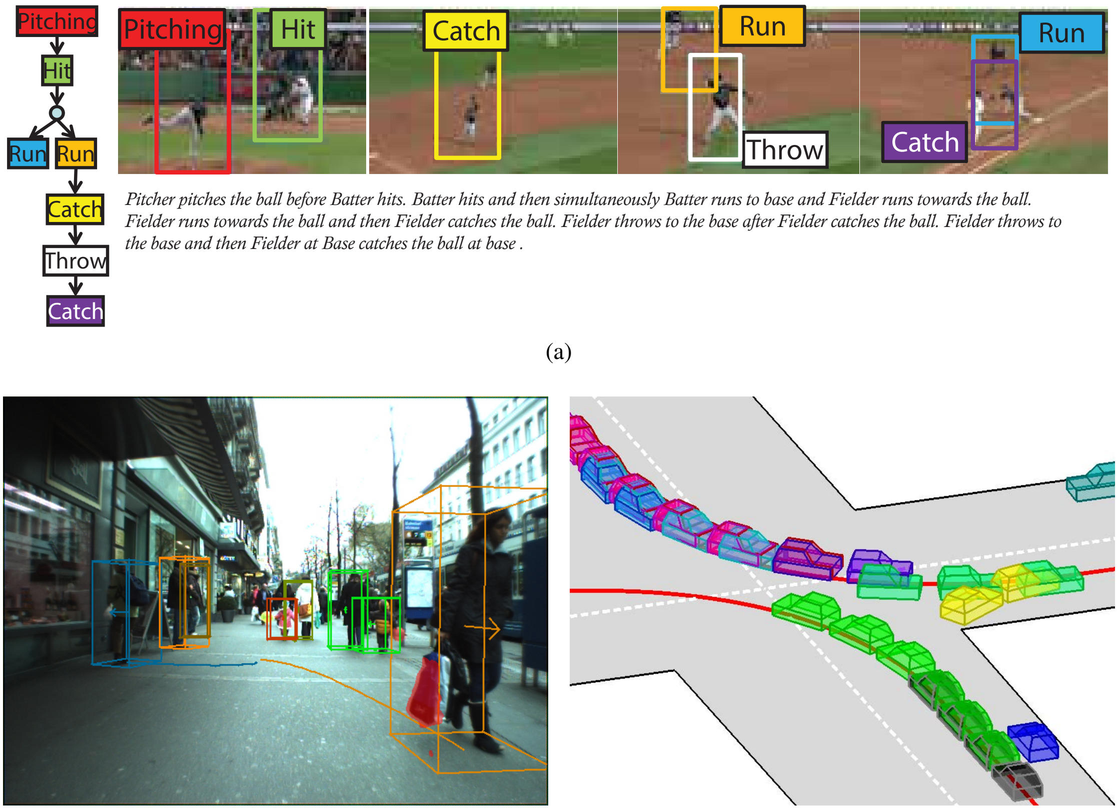
Going beyond labeling-based 3D reconstruction leads to reconstructing the scene from higher-level representation. Following [Lee et al., 2009], the layout of indoor Manhattan World scenes can be estimated from a single image. By connecting and sweeping line segments, the most likely box layout is found. [Hedau et al., 2009] present a novel approach on estimating the scene layout of cluttered rooms by fitting the most likely 3D box. By combining the clutter detection and vanishing point evaluations, the most likely configuration is found. [Gupta et al., 2010] describe an approach which allows estimating the 3D scene layout by combining volumetric reasoning (e.g. occlusions, arrangement of objects) with reasoning with mechanics (e.g. material density and internal energy). [Schwing et al., 2012] propose to estimate the 3D surface layout of an indoor scene by decomposing higher order potentials into pairwise potentials by incorporating integral images to geometry. This enables much faster runtimes by keeping the high accuracy of state-of-the-art methods. [Schwing and Urtasun, 2012] propose the first exact solution to obtain the 3D room layout using a cuboid in indoor environments. They do not use an MRF to perform inference but solve the problem by using a branch and bound formulation.

2.2.2.4 Multiple View 3D Reconstruction

A full 3D reconstruction pipeline from 2D image sequences is presented in [Pollefeys et al., 2004], which is based on one of the first SfM methods [Pollefeys et al., 1998]. Images are obtained from a hand-held camera and the method recovers both the camera parameters and the underlying 3D structure from uncalibrated image sequences. Two initial cameras are determined as a starting point for 3D reconstruction and both structure and motion are updated for every additional camera pose. Refinement for structure and motion is obtained by using bundle adjustment. Metric reconstruction is achieved by exploiting self-calibration. Dense disparity matching from rectified images in combination with a triangular mesh allow generating 3D surface models of the scene. A sparse 3D reconstruction framework for SfM was introduced by [Snavely et al., 2006]. The algorithm creates a sparse point cloud in combination with corresponding camera positions from a given image set. The reconstruction is done incrementally which means that not all images are considered at a time. Optimization is obtained by exploiting the bundle adjustment algorithm [Lourakis and Argyros, 2004]. A dense 3D reconstruction pipeline is published by [Furukawa and Ponce, 2007]. By using SIFT and Difference of Gaussian (DoG) features and multiple iterations of matching, expanding and filtering these matches, a dense model is obtained from multiple images. The key towards performance and geometric correct reconstruction lies in enforcing visibility constraints and photometric constraints. The framework is extended by [Furukawa et al., 2010] in order to overcome large collections of images. Before reconstructing the scene and finding corresponding feature matches between images, the method proposed in [Furukawa et al., 2010] clusters the input images. Reconstruction is then performed on each cluster where a fusion of all clusters in performed in the last step of the algorithm. Reconstructing a 3D model from 2D images using a plane sweep stereo reconstruction algorithm with multiple sweeping directions is proposed by [Gallup et al., 2007]. A 3D reconstruction pipeline using a single semantic segmentation and matching method is presented by [Mičušík and Kosecka, 2009]. Dense reconstruction of well-known touristic parts of cities is presented in [Agarwal et al., 2009] by using a parallel distributed system. Dense reconstruction processed on a single computer is presented by [Frahm et al., 2010]. Images from tourists are collected and matched by the method published by [Agarwal et al., 2009] to obtain the 3D model. [Frahm et al., 2010] automatically cluster the pictures based on the location of the building which is seen on the images. This enables the 3D reconstruction of the scene in less time with less computational effort. Automatic dense 3D reconstruction from 2D images using planar patches to recover both planar and non-planar structures was introduced by [Gallup et al., 2010]. Planes are detected using RANSAC and automatically linked for multi-view reconstruction. The final outcome is obtained by exploiting Graph Cuts [Boykov and Jolly, 2001]. [Wu et al., 2011] present a dense reconstruction pipeline which is working on a single input image and is based on exploiting the repetition of image patches. [Xiao and Furukawa, 2012] describe an algorithm for reconstruction and visualization of large scale indoor environments from various museums. By exploiting volumetric primitives and therefore doing a volumetric reconstruction instead of recovering a surface model, wall configurations are found and textured. [Häne et al., 2012] present a pipeline for piecewise planar depth map fusion and 3D reconstruction using a first-order primal dual optimization method instead of a higher order one.

2.2.2.5 Combining Recognition and 3D Reconstruction

Going one step further after reconstructing the 3D environment, the computer should know which object is at what location. Therefore, the objects need to be localized and put in 3D space. These two tasks can also be combined in order to increase the accuracy of both the detection and the 3D reconstruction task. [Bao et al., 2010] outline a method which first detects some specific objects in the scene (e.g. cups) and then reasons about the supportive plane between these objects. By optimizing both the detections and the supportive plane, incorrect detections are eliminated and missed detections are found. A SfM pipeline for estimating both camera parameters and 3D structure from point correspondences and corresponding object detections between several viewpoints is described by [Bao and Savarese, 2011]. This enables both geometric and semantic 3D reconstruction from multiple views. This approach was then extended by [Bao et al., 2012] by also using corresponding regions additionally to points and objects to reconstruct the scene. Labeling both visible and occluded regions was introduced by [Guo and Hoiem, 2012]. By using visible cues, semantic context and geometric features, the underlying structure is estimated. [Fouhey et al., 2012] present an algorithm for 3D reconstruction from observing a human action in a still image. According to the pose of the human, the geometric context is obtained. An extended approach is described by [Delaitre et al., 2012]. By observing humans over time, the underlying 3D structure can be estimated from human poses and object appearances. Joint dense 3D reconstruction and both 2D and 3D class segmentation is presented in [Häne et al., 2013]. The reconstruction is done in volumetric space where voxels are labeled as *occupied* or *empty*. Occupied voxels are then assigned a label from a discrete set of classes (e.g. ground, building, vegetation.). By incorporating class specific geoemtric priors and a smoothness constraint, an optimization leads to the final reconstruction and labeling result.



Pitcher pitches the ball before Batter hits. Batter hits and then simultaneously Batter runs to base and Fielder runs towards the ball. Fielder runs towards the ball and then Fielder catches the ball. Fielder throws to the base after Fielder catches the ball. Fielder throws to the base and then Fielder at Base catches the ball at base .

(a)

(b)

(c)

Figure 2.3: Examples for 3D reasoning frameworks. (a) Automatically generated storyline of a baseball match, taken from [Gupta et al., 2009]. Objects are detected in the scene and the storyline is generated based on detecting objects and events. (b) Pose estimation and tracking results taken from [Mitzel and Leibe, 2012]. Pedestrians are tracked using a 3D bounding box, carried items are marked red. (c) 3D scene layout estimation of a traffic scene, taken from [Geiger et al., 2011].

2.2.3 3D Reasoning

The Holy Grail in computer vision is to understand what an image is describing and to perform 3D scene reasoning about the scene in order to extract similar results and cues as humans are experiencing. Neuroscience studies have pointed out that a human's brain works with 3D representation of objects and scenes and somehow (unclear how) stores the 3D information to achieve 3D reasoning about the scene at the level humans have been experiencing. The field of 3D reasoning is a broad one covering an endless number of different events (e.g. human actions and 3D object movements). Nevertheless, all these high level event recognition and 3D reasoning methods incorporate 3D pose estimation and 3D object tracking as a first step, which can also be seen in Figures 2.3. Figure 2.3a shows the storyline of a baseball match, automatically generated by the method described in [Gupta et al., 2009]. Figure 2.3b shows the outcome of a 3D object tracking pipeline described in [Mitzel and Leibe, 2012]. Figure 2.3c presents the outcome of the algorithm described in [Geiger et al., 2011]. The method performs 3D scene layout estimation in combination with detection and tracking of traffic participants in 3D space. Although the tasks solve different problems in the domain of 3D reasoning, 3D pose estimation and 3D tracking of rigid and non-rigid objects serve as a basis for all three applications.

2.2.3.1 3D Pose Estimation

Using 3D information helps when performing object detection, classification and pose estimation. First, the appearance of objects varies substantially with the viewing angle and local features may be occluded in 2D. Second, using 3D information allows making some a-priori assumptions about the scene and relaxing the problem (e.g. a car is more likely to be located on a road than in the sky). By representing a 3D scene in 2D, the depth information gets lost and going back to 3D from a single viewpoint is therefore an ill-posed problem. In computer vision, the same object may be captured under varying lighting conditions and different poses. Therefore, traditional pose estimation algorithms extract features from multiple input images to cover at least a discrete set of variations in order to relax the problem of these intra-class variations [Fergus et al., 2005a, Fergus et al., 2005b]. These approaches were also extended to view-invariant detection methods by learning a sparse 3D object model from multiple training images [Payet and Todorovic, 2011, Ozuysal et al., 2009, Thomas et al., 2006].

Following [Savarese and Li, 2007], 3D shape models for pose estimation of rigid objects are estimated by learning a collection of features from multiple 2D training images. Visual features are combined with geometric ones in order to recover the pose of rigid objects.

Detailed 3D models of cars and motorcycles are first exploited by [Liebelt et al., 2008] for classification in still images. The training is performed using a synthetic camera orbiting around the models. Classification is done by comparing all possible projections to the input image. Combining these geometric features with appearance features learned from 2D images is demonstrated by [Liebelt and Schmid, 2010, Khan et al., 2010, Glasner et al., 2011]. An approach for matching vehicles in still images under large body transformations using detailed 3D models is described by [Guo et al., 2008a]. They gain an initial pose from meta-data, match the 2D image projection to the 2D model projection by using Chamfer distance and the Iteratively Closest Point algorithm. Rendering is done on manually labeled semantic parts, where occlusions in the rendering are handled by filling gaps using an MRF.

[Arie-Nachimson and Basri, 2009] published a way to construct implicit 3D shape models for pose estimation of vehicles. As an extension to [Savarese and Li, 2007], visual and geometric features are combined with visibility and transformation constraints. A rough shape model for 3D pose estimation of vehicles is published by [Li et al., 2009]. A Bayesian inference algorithm is exploited to generate a shape model from visual features coming from partial shapes, i.e. their corresponding geometric information. Following [Stark et al., 2010], a viewpoint estimator is trained on non-photorealistic rendered images of 3D CAD models. These models are mapped onto 2D image planes in order to obtain visual features. The classifier is then trained on these features in order to obtain a generalized pose estimator. [Guo et al., 2008b] describe an approach for 3D pose estimation in still images using Chamfer matching and projections of existing 3D models.

[Villamizar et al., 2011] utilize real images, Histogram of Oriented Gradients (HOG) features and RFs for training the pose estimator. Sun and Savarese propose a framework for object pose estimation using a general 3D model representation in [Xiang and Savarese, 2012]. Objects are detected with respect to its aspect configuration and the matching score to a learned configuration from multiple 3D models. Detecting an object using parts and the configuration of these parts was first introduced by [Felzenszwalb et al., 2010]. This approach was extended to discrete 3D pose estimation by [Pepik et al., 2012b]. The method explained by [Pepik et al., 2012a] provides a further extension to enable pose estimation on a continuous view sphere. Additionally to having a certain part configuration constraint in 2D, a part configuration learned from 3D models must also hold. The approach of [Liebelt et al., 2008] was extended to represent the object viewpoint on a continuous instead of a discrete viewsphere by [Schels et al., 2012].

The RF classifier is a popular method for pose estimation in computer vision due to the fact that it can handle large training data sets and several classes, it is robust against outliers and the classification or regression task is performed quickly, as is described in Section 3.3.2. Using classification-based RFs for pose estimation of humans was first introduced by [Rogez et al., 2008]. Human people are detected in a variety of real world images and a training set is built up to generate an RF. Classes are generated by encountering both the pose and the action of a human. Using regression instead of classification is described by [Gall et al., 2011b]. Objects are found by exploiting the generalized Hough transform. Detections of object parts individually vote for the localization of the complete object. By using depth images, regression is used for determining a human's head pose [Fanelli et al., 2011]. These features are also used by [Shotton et al., 2011] for obtaining a human's pose. Each body part casts votes for a single class. The final pose is estimated by generating confidence-scored 3D proposals of how the body parts are connected. Conditional RFs are used for pose estimation by [Sun et al., 2012] which allows incorporating relationships between output variables by a global latent variable.

2.2.3.2 3D Tracking

A polyhedral 3D vehicle model for classification, following a motion model over time, was first introduced by [Koller, 1993]. The parameters of the model are first found by edge fitting and the refinement is done by exploiting the tracking results. More complex deformable models are first described by [Ferryman et al., 1995]. The model parameters are learned from real data, where the parameter space is reduced using principal components analysis. The pose of a fixed model which is obtained from gray value image gradients rather than from edge segments is described by [Kollnig and Nagel, 1997]. Following [Lou et al., 2005], vehicles are tracked by exploiting a simple 3D model and a Kalman filter. The shape of the used 3D model is fixed and the pose is determined by exploiting edge fitting. The pose is then refined by introducing a feasible motion model. [Zhang et al., 2008a] use a fitness score and a particle filter for tracking the vehicle. The approach of [Liebelt et al., 2008] was extended to videos in [Toshev et al., 2009]. The area of the projected model is matched to a segmented foreground input image mask. The area overlap as well as the shape similarity is used as matching score. The temporal inference is established by introducing a CRF. Using a deformable model [Leotta, 2010] for vehicle tracking was first introduced by [Leotta and Mundy, 2011]. The authors are changing parts of the model online between consecutive frames and align the rendered projection with the input image. A Kalman filter is used to predict a pose from a frame to a subsequent one. A framework for estimating the scene layout of traffic scenes in combination with a 3D detection and tracking pipeline for traffic participants is presented by [Geiger et al., 2011]. The algorithm is applied on a single moving camera and the results are obtained by combining 3D object detections, object tracklets and semantic scene labels.

The task of 3D person tracking is tackled by [Urtasun et al., 2006]. Human pose and motion estimations are learned by using Gaussian Process Dynamical Models to overcome different human walking styles. Multi-camera person tracking in 3D space was presented by [Fleuret et al., 2008]. Persons are detection in each image and blobs are merged in order to obtain a 3D occupancy grid. Tracking is then done in 3D space. Person tracking using a stereo rig on a mobile device was introduced by [Ess et al., 2009]. The method jointly estimates camera position, stereo depth, and object positions. A real-time multi-person 3D detection and tracking system was introduced by [Mitzel et al., 2011]. A 3D depth occupancy map is used to track, add and delete individual objects over time. An extended approach of [Mitzel et al., 2011] for 3D tracking of pedestrians, carried items and other unknown objects is presented by [Mitzel and Leibe, 2012]. Objects are extracted from input images in combination with stereo depth maps and the 3D scene model is updated online. Different objects are categorized based on their 3D shape.

2.3 Innovative Aspects and Context of this Thesis

The main advantages and innovations of the proposed pipeline over existing work can be summarized as follows.

Calibration

The first proposed method for camera calibration extracts top and bottom points similarly to the algorithm described in [Lv et al., 2006]. Differently to existing approaches, these points are not only used for gathering intrinsic parameters of a single camera (e.g. [Lv et al., 2006], or [Kusakunniran et al., 2009]) but for estimating both intrinsic and extrinsic parameters of a whole network of cameras.

Different to existing approaches (e.g. [Lv et al., 2006] and [Junejo, 2009] who use pedestrians, or [Wildenauer and Hanbury, 2012] who use building façades of man-made environments), the second proposed method combines static (zebra-crossings) and dynamic (pedestrians) objects for calibrating a single surveillance camera. This combines the main advantage of conventional calibration techniques, namely obtaining a precise result, and the main advantage of auto-calibration techniques, namely increasing the flexibility. To sum up the method described by [Zhang et al., 2008b], a vertical vanishing point is extracted by observing pedestrians and two horizontal vanishing point are extracted by observing multiple vehicles' movements. Different to that approach, the proposed algorithm does not rely on a specific movement of the dynamic objects (e.g. the cars must not turn but move straight). In theory, there are no standardized dimensions of a zebra-crossing available. In practice, brighter areas of it are at least two meters long and exactly 50 centimeters wide. This knowledge is exploited for extracting metric information from the scene in order to overcome the problem of forcing the user to provide some a-priori information about the scene (e.g. the pedestrian's height, as illustrated by [Lv et al., 2006], or the camera's height, as by [Zhang et al., 2008b]).

3D Reconstruction

Conventional 3D reconstruction methods rely on obtaining discriminative feature matches between corresponding views [Snavely et al., 2006, Agarwal et al., 2009]. The obtained sparse 3D model is then densified e.g. by the method introduced by [Furukawa and Ponce, 2007] which is based on combining a variety of different feature descriptors for feature matching. Man-made environments contain flat and textureless surfaces where large areas provide the same color information (e.g. walls and floors). These areas deliver incorrect matches due to the lack of discrimination between feature descriptions. In order to describe such regions, semantic information is used for existing approaches. [Mičušík and Kosecka, 2010] present a method which reconstructs the scene from multiple views based on semantic information coming from a single superpixel segmentation method. [Hoiem et al., 2007] describe an approach for extracting 3D information from a single 2D image by segmenting an image and using color and texture information in combination with perspective features for each segment. The proposed framework is based on these methods but combines multiple segmentation algorithms in order to increase the 3D reconstruction accuracy. Conventional 3D reconstruction pipelines deliver correct feature matches for regions, where discrimination can be assured (e.g. texture on the ground or on the ceiling, chairs, windows). Different to existing work, where either point features or semantic information is used, the proposed 3D reconstruction pipeline combines information coming from semantic cues and 3D reasoning.

3D Reasoning

In [Leotta and Mundy, 2011], tracking is done by changing parts of a deformable 3D model and predicting the pose by a Kalman filter. The proposed approaches are not using a deformable model since on the one hand, having more parameters to optimize to get the shape of the car can fail due to multiple local minima. On the other hand, not having the strong prior knowledge about the vehicle's shape gives worse tracking results, as can be seen in the experiments. There is never happening a prediction of a pose from one frame to a subsequent one, as described in [Leotta and Mundy, 2011], but a set of poses for each frame is evaluated in order to find the best combination of vehicle types and poses over time. This brings the advantage of having multiple hypothesized poses available for each frame and a higher accuracy in terms of pose estimation. To sum up [Guo et al., 2008b], they work on single images, gain an initial pose from meta-data and match 2D images to 2D projection by conventional Chamfer distance. Rendering is done on manually labeled semantic parts. They exploit an MRF for connecting the right semantic parts of the vehicle. Different to pose estimation methods working on still images [Guo et al., 2008b], each frame is not optimized separately but temporal consistency in the video is exploited for temporal inference and refinement. This means it relaxes the computational cost of finding the best solution for each frame by exploiting temporal coherence as a strong prior. Using Fast Directional Chamfer Matching (FDCM) or RFs for vehicle pose estimation, as in this research, works superior in terms of speed, accuracy and practicality [Liu et al., 2010b], compared to using meta-data as in [Guo et al., 2008b]. The main drawback of the approach presented in [Toshev et al., 2009] is the inevitable perfect foreground segmentation for each frame. The proposed vehicle 3D pose estimation and classification approaches are able to overcome this problem by relying on edges for matching 3D models projections and input images instead of using background subtraction and foreground shape comparison.

2.4 Summary

The research discipline of 3D computer vision emerged from photogrammetry. This chapter first gives a brief overview of this development and describes related work from the field of 3D scene understanding. Related work is then split into the task of 3D scene understanding, namely calibration, 3D reconstruction and 3D reasoning.

First, related publications and methods for the task of camera calibration, which is divided into conventional calibration methods and auto-calibration approaches, are described.

Second, algorithms for 3D reconstruction from 2D images, which can be divided into extracting 3D information coming from a single view, or from multiple ones, are reviewed. Additional work for combining reconstruction and object recognition is presented.

Third, related work tackling the tasks of 3D vehicle pose estimation and 3D vehicle tracking are presented, as the proposed 3D reasoning pipeline is also dealing with these two tasks.

This chapter then also outlines innovative aspects of the proposed framework and shows how it is related to existing work.

CHAPTER 3

Theory and Background

As can be seen in Figure 3.1, pixels offer color or intensity information but a single pixel does not tell anything about the 3D scene or its 2D projected scene without any geometric information. To describe a scene, a geometric structure must be known in combination with the pixels. Geometry can either occur within an image plane or between image plane and 3D coordinate system.

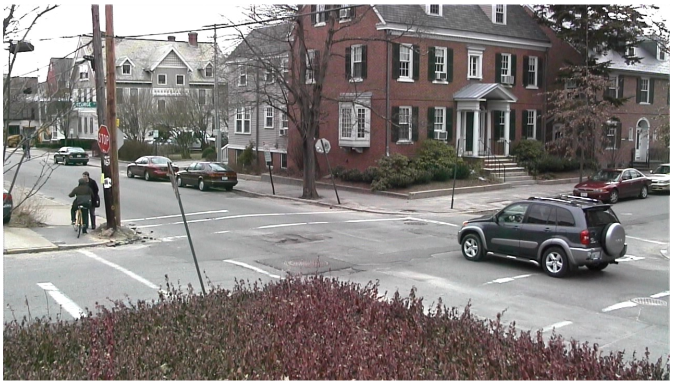


Figure 3.1: Random input image, which is taken from [Leotta and Mundy, 2011], used for demonstration purposes to show the outcome when applying different features and representations presented in the following parts of Chapter 3.

First, this chapter outlines geometric representations in both 2D space (Section 3.1) and in a 3D world (Section 3.2), as well as their integration in computer vision applications. The presented 2D image representations are for example used for 3D vehicle pose estimation described in Section 6.3, or 3D surface normal labeling presented in Section 5.1 and 3D reconstruction outlined in Section 5.2.

Second, a method for optimization, namely RFs, and a method for classification, namely MRFs, are described in Section 3.3. The theory for these two tools is provided as they are used

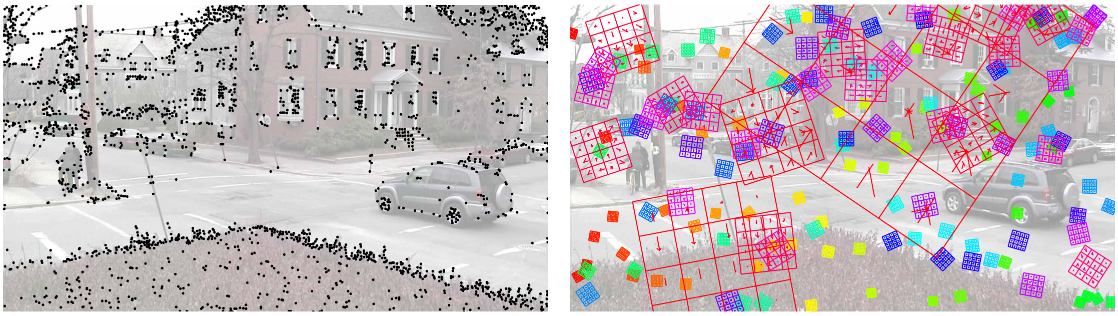

(a) (b)

Figure 3.2: Point features describing the structure of an image. (a) Harris corner detector, (b) SIFT features, where every 20^{th} feature is shown due to better visualization.

for the 3D vehicle pose estimation and classification implementation in Chapter 6 as well as for the dense 3D reconstruction algorithms outlined in Chapter 5.

This chapter therefore gives an overview on all methods and principles used in Chapters 4-6. Discussions on why the specific methods are chosen for the accompanying implementations are presented in the particular chapter.

3.1 2D Image Representations

Geometric information is fundamental when describing 2D image projections of 3D scenes. An image can only be described when the geometric information of one pixel is known. This means that a pixel's image coordinates and a geometric relationship to other pixels are present additionally to color or intensity values. This geometric information within an image plane is described by so called geometric features [Shapiro and Stockman, 2001]. Geometric features can be clustered based on their dimensionality which is needed for representing them. A dimension of zero is needed to describe point features, one dimension describes line or edge features and two dimensions describe region or area features. In computer vision, these features are the basis to perform higher level computer vision tasks (e.g. object detection, object matching, 3D reconstruction, scene understanding etc.) [Leotta and Mundy, 2011, Ma et al., 2003, Sonka et al., 2008].

3.1.1 Point Features

Features having a dimensionality of zero are also known as interest points [Hartley and Zisserman, 2003]. In general, when talking about interest in this context, the pixel should provide discriminative information when looking on a predefined geometric image structure around the pixel position and this information should be robust against defined changes (e.g. transformations, rotations, lighting, etc.) [Hartley and Zisserman, 2003, Gonzalez and Woods, 2007]. Figure 3.2 shows the outcome of two point feature detectors, namely the Harris corner detector, first described in [Harris and Stephens, 1988], (see Figure 3.2a) and SIFT, which was introduced in [Lowe, 2004] (see Figure 3.2b.

These two image descriptors are picked out for presentation in the following sections since Harris corner features are used for the implementation described in Section 6.3 and SIFT features are used for gathering a sparse point cloud from an image sequence, as described in Section 5.1.

3.1.1.1 Harris Corner Detector

One of these point features describing the structural information around a pixel is the Harris corner detector, introduced by [Harris and Stephens, 1988]. This detector searches for junctions of edges in an image and operates on intensity values rather than on color information. The detected point is therefore only described by a location (x, y), where the idea of the detector is the following. Having a patch of a certain size of an image, there will be less intensity change when this patch is shifted on a flat region. If this patch is shifted along an edge, there will be less intensity change along this edge, but if the window slides over a corner, there is a discontinuity between intensity values in all directions. From the mathematical point of view, this change between intensity values for a point (x, y) in image $\mathcal{I}$ and a shift (u, v) is computed by

$$\begin{aligned} E(u,v) &\approx \sum_{x,y} G(x,y,\sigma)\left[\mathcal{I}(x,y) + u\mathcal{I}_x + v\mathcal{I}_y - \mathcal{I}(x,y)\right]^2 \\ &= \sum_{x,y} G(x,y,\sigma)\left[\mathcal{I}_x^2 u^2 + 2\mathcal{I}_x\mathcal{I}_y uv + \mathcal{I}_y^2 v^2\right], \end{aligned} \tag{3.1}$$

where $\mathcal{I}_x$ and $\mathcal{I}_y$ are the partial derivatives of $\mathcal{I}$ in vertical and horizontal direction, $G(x, y, \sigma)$ is a Gaussian kernel with size σ. Corners are then found by analyzing the so called structure tensor of Equation 3.1 [Harris and Stephens, 1988].

3.1.1.2 Scale Invariant Feature Transform

Another point feature used in computer vision for describing and detecting the structural information of points of interest is called SIFT, which was introduced in [Lowe, 2004] by David Lowe in 2004. The main advantage of this detector is the fact that it is invariant to changes in translation, rotation and scaling. The SIFT feature is therefore described by a pixel location (x, y), a scale, and an orientation.

The first step of SIFT is to detect corner points in an image but in combination with enforcing invariance to scaling changes. The keypoints are detected in a cascaded filtering approach. This can be done by first filtering the input image with multiple Gaussian kernels of various sizes σ. An image $\mathcal{I}$ is filtered at scale space σ at position (x, y) by

$$\mathcal{L}(x,y,\sigma) = \mathcal{I}(x,y) \otimes G(x,y,\sigma). \tag{3.2}$$

After filtering the image $\mathcal{I}$ with multiple Gaussian filters, the DoG function is computed by

$$\mathcal{D}(x,y,\sigma) = \mathcal{L}(x,y,k\sigma) - \mathcal{L}(x,y,\sigma), \tag{3.3}$$

where k is a multiplicative factor. In the original implementation of [Lowe, 2004], three scale spaces are used and combined to one so called octave. Within one octave, the image dimensions are kept constant. Multiple octaves are used in the implementation, where the image dimensions are halved for each octave. To detect keypoints in the image, the maxima and minima are determined in the DoG images. Each sample point is therefore compared to its eight connected neighbors. The keypoint is then selected when the value is larger or smaller than the values from all its neighbors. As this comparison is not only established spatially but also over different scale spaces, each keypoint must be compared to 26 neighbors when three scale spaces are used. In this step, corners are also localized and both edge-like extrema and low contrast extrema are discarded due to the lack of discrimination. In [Lowe, 2004], low contrast extrema are determined by using an approximation of a second-order Taylor expansion on the DoG function, $\mathcal{D}(x, y, \sigma)$. Edge-like extrema can be determined by calculating the eigenvalues of a specific pixel location.

In a next step, the feature's orientation is determined in order to become invariant against rotational changes. For each window at the selected scale, centered at the point of interest, the principle gradient direction is computed. In practice, this is established by calculating a histogram from all orientations available within the window. For each keypoint, the magnitude m and orientation θ is therefore computed by

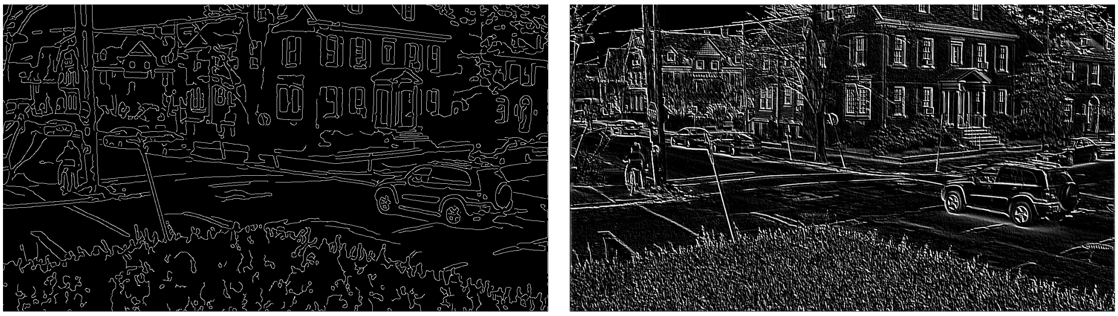
$$\begin{aligned} m(x,y) &= \sqrt{(\mathcal{L}(x+1,y,\sigma)-\mathcal{L}(x-1,y,\sigma))^2+(\mathcal{L}(x,y+1,\sigma)-\mathcal{L}(x,y-1,\sigma))^2} \\ \theta &= \arctan\left(\frac{\mathcal{L}(x,y+1,\sigma)-\mathcal{L}(x,y-1,\sigma)}{\mathcal{L}(x+1,y,\sigma)-\mathcal{L}(x-1,y,\sigma)}\right), \end{aligned} \tag{3.4}$$

where σ is chosen to be the closest scale to the scale of the keypoint. An orientation histogram having 36 bins is then computed, covering a range of 360 degrees. Each orientation is weighted by the magnitude m and by a Gaussian window of $G(x, y, \frac{3\sigma}{2})$. The dominant direction in the window corresponds to the highest peak in the histogram. When multiple peaks are found, multiple keypoints having the same location and the same scale but different dominant orientations are generated for the observed pixel location.

The feature description is then obtained by calculating 16x16 orientation histograms around the location of the keypoint. These are then weighted by a Gaussian filter in order to obtain a 4x4 grid of orientation histograms around the keypoint. Each histogram contains 8 orientation bins which means that a SIFT descriptor holds 128 elements [Lowe, 2004].

3.1.2 Edge Features

Apart from point features, the geometric structure of an image can also be described by so called edge features [Shapiro and Stockman, 2001, Ma et al., 2003]. Edges can arise from discontinuities in depth, discontinuities in surface orientation, discontinuities between different materials and discontinuities in terms of illumination [Barrow and Tenenbaum, 1980]. Ideally, an edge detector should find all these discontinuities which should lead to finding the boundaries between different objects, different surface orientations, different surface materials and when illumination changes occur [Barrow and Tenenbaum, 1980,Lindeberg, 1998]. The main problem is, that this is not always applicable due to e.g. occlusions or low contrast which then leads



(a) (b)

Figure 3.3: Edge features describing the structure of an image. (a) Canny edge detector using parameters $\sigma = 3$, $t_1 = 0.05$, $t_2 = 0.2$, (b) Sobel filter without applying a threshold.

to fragmented, missing and false edges. False edges in this case are edges which indicate a boundary where there is no boundary [Lindeberg, 1998]. In computer vision, detecting edges corresponds to calculating the first-order derivatives from the intensity values of an image in both horizontal and vertical direction to obtain the gradient images in both horizontal and vertical direction $\mathcal{I}_x$ and $\mathcal{I}_y$, respectively [Lindeberg, 1998, Ma et al., 2003].

In the following, the Sobel filter and the Canny edge detector are described. The Sobel filter is used for the implementation described in Section 6.3, the Canny edge detection is used several times, e.g. in Section 4.2 to locate the zebra-crossing, or in Section 6.1 and Section 6.2 to obtain the rendered images of the 3D models.

The Sobel filter is a convolution of the image $\mathcal{I}$ with the Sobel operator [Gonzalez and Woods, 2007] in vertical and horizontal direction, followed by calculating the magnitude and orientation of the edge. Figure 3.3b shows an image filtered with the Sobel operator, where no threshold is used for binarization of the output [Gonzalez and Woods, 2007].

Another approach for detecting edges in an image was introduced by [Canny, 1986], which in literature is known as the Canny edge detector. The detector is by now quite old but is still considered state-of-the-art in the field of computer aided image analysis. The output of the algorithm is shown for comparison to the Sobel operator in Figure 3.3a. The three parameters needed, σ and t_1, t_2, are describes in the following and set to $\sigma = 3$ and $t_1 = 0.05$, $t_2 = 0.2$ respectively for this example image. The method tries to overcome the aforementioned problem of finding incorrect or fragmented edges by searching for an optimal trade-off between noise reduction and edge localization. The first step of the algorithm is designed to localize edges in an image. The optimal function is described by four exponential terms which is time consuming to be solved and therefore can be approximated by a first-order derivative of a 2D Gaussian kernel. The image is therefore in a first step smoothed by convolving it with a Gaussian kernel of size σ which in practice is done by applying Equation 3.2. Second, the Sobel operator is applied and both magnitude and orientation are calculated [Canny, 1986, Gonzalez and Woods, 2007].

Since an edge should only be one pixel wide, the next step is known as edge thinning which in practice can be done by applying Non-Maximal Supression (NMS). Each potential edge pixel is evaluated and if the magnitude of the pixel in question is smaller than the magnitude of all its neighboring pixels, the point is not considered as an edge point. Neighboring pixels, which lie along the edge direction, can of course have a higher magnitude and are therefore not taken into consideration [Sonka et al., 2008, Canny, 1986].

The last step of the algorithm should decide whether or not the edge is strong enough. A strong edge in this context means that the magnitude is high and is therefore more likely to be a correct edge. Edges are filtered by introducing a hysteresis thresholding procedure which means that all values should be located between thresholds t_1 and t_2. Corresponding edges are found by tracing edges in the image. First, an element holding a magnitude $m > t_2$ is located in the image. By tracing the edge, all pixels having a magnitude $m > t_1$ are then considered to be part of the edge [Canny, 1986].

3.1.3 Region Features

Region features do not only cover points or lines within an image but densely a whole part of it [Sonka et al., 2008]. Similar to edge features, a region should represent all the pixel providing similar color, intensity, depth, illumination, texture, or other similar properties, where this similarity should hold up to a certain threshold [Leotta and Mundy, 2011]. Grouping the pixels is also known as segmentation where the desired outcome of the algorithm is image segments [Ma et al., 2003]. Different to edge features, pixels within one segment do not have to be spatially connected. Similar to edge features, segmentation methods can provide incorrect or missing segments. In the following, principles for segmenting an image into so called superpixels and segmenting the image into Background (BG) and Foreground (FG) are presented. There are many representative implementations developed both these two types of region features, where Figure 3.4 shows one superpixel segmentation method based on entropy rates [Liu et al., 2011] and one BG/FG segmentation method called frame differencing [Sonka et al., 2008]. In Figure 3.4a the image is split into 1300 superpixels. The threshold for binarization of Figure 3.4b is chosen to be an intensity value of 50.

The presented background subtraction technique is used for the implementations described in e.g. Section 4.1, Section 4.2 and Section 6.3. Superpixel segmentation methods are the basis for 3D surface normal labeling and 3D reconstruction algorithms presented in Chapter 5. Therefore, these two region features are presented in the following.

3.1.3.1 Superpixels

The name *Superpixel* and a corresponding algorithm was first introduced by Ren and Malik [Ren and Malik, 2003] in 2003. The idea of superpixels is to transform an image, consisting of pixels where one pixel is visually not meaningful, into perceptually meaningful regions [Achanta et al., 2012]. Superpixels can be seen as an oversegmentation of the image which means that dividing an image into superpixels is known as pre-segmentation [Hanbury, 2008]. Regardless of the algorithm to be applied on the image, superpixels reduce the computational complexity of subsequent image processing tasks compared to using all pixels. Different to other segmentation

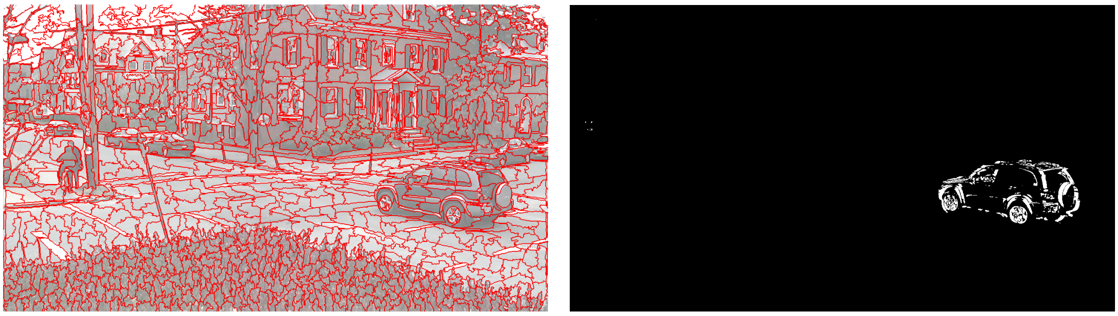

(a) (b)

Figure 3.4: Region features describing the structure of an image. (a) Entropy rate superpixel segmentation [Liu et al., 2011] producing 1300 segments, (b) BG / FG segmentation using frame differencing and a threshold of 50.

methods (e.g. BG/FG segmentation), pixels, which are grouped to superpixels, are also spatially connected.

Depending on a particular application, there are several algorithms which are able to split an image into superpixels each of them having advantages and disadvantages. In Section 5.1 of this dissertation, four different superpixel segmentation methods are used as a pre-processing step for 3D surface labeling. Each of these methods are described in the following in chronological order of the date they were published.

The first one was introduced in [Felzenszwalb and Huttenlocher, 2004] by Felzenszwalb and Huttenlocher in 2004. The segmentation is solved by using a graph-based approach, where each node represents a pixel and edges describe the dissimilarity between neighboring pixels. Nodes are clustered using the graph so that edges within a segment have low weights and edges between segments provide high weights. Each segment is defined by a minimum spanning tree. The internal difference is therefore defined as the maximum edge weight in the minimum spanning tree of the segment. A boundary between segments is then found by gathering the minimum weight of the edges connecting neighboring superpixels. The superpixels obtained are irregular in terms of size and compactness and the number of superpixels cannot be defined beforehand. The complexitiy of the algorithm changes linearly with the number of pixels in the graph which means it has a complexity of $O(n \log n)$, where n is the number of pixels.

The second method outlined in this section was first described in 2009 by [Levinshtein et al., 2009]. The approach is named *TurboPixels* and uses geometric flow to obtain a segmentation of the image. Superpixels are obtained by dilating multiple seed locations, based on the level-set geometric flow method, which are evenly distributed over the whole image plane [Osher and Sethian, 1988]. The geometric flow is calculated by using image gradients, where boundaries stop to grow when they get close to a region having a high gradient. This approach leads to evenly distributed superpixels having similar sizes and compactnesses where the number of superpixels can also be defined. According to the authors, the complexity of the algorithm is described by $O(n)$.

The third superpixel method is named Simple Linear Iterative Clustering (SLIC) and was first introduced by [Achanta et al., 2012, Achanta et al., 2010]. The clustering is performed by using k-means in 5D space, where these five dimensions are x, y on the image plane and L, a, b from the CIELAB color space. First, seed points are found by sampling on a regular grid in the image space. Resulting superpixels are roughly same sized and the desired number of superpixels can also be defined for this approach. The distance between a pixel and a potential segment center is found by using a distance measure which also incorporates the size of the superpixel. The complexity of this approach is $O(n)$.

The last algorithm described in this dissertation is based on entropy rates and was first introduced by [Liu et al., 2011] in 2011. Superpixels are obtained by finding the maximum of a graph-based energy function which consists of an entropy rate and a balancing term. The entropy rate is used to find the superpixel borders, which refer to the cuts in the graph, and the balancing term is used to get superpixels of approximately the same size. To obtain compactness and homogeneous superpixels, the entropy rate of a random walk on the graph is used and the optimization is done by using a greedy algorithm. For this approach, the number of desired superpixels can be defined, and the method has approximately a complexity of $O(n \log n)$, according to the authors.

3.1.3.2 Background Modeling

BG/FG segmentation is also known as motion estimation or background modeling, where the purpose is to determine moving parts [Shapiro and Stockman, 2001]. In order to find these moving parts, a reference frame must be provided which can be a background image or a subsequent frame. Moving segments are described as FG, pixels, which did not move between the two compared frames, are denoted as BG. The simplest way to divide the image into FG and BG regions is achieved by applying frame differencing and comparing intensity values of subsequent frames or between an input frame and a background image. If the intensity difference at a pixel location is above a certain threshold, the pixel is considered to be foreground, otherwise it is considered to be part of the background. The output of this method using two subsequent frames and a threshold of 50 can be seen in Figure 3.4b [Sonka et al., 2008].

Simple background subtraction by frame differencing suffers from a high level of noise since each pixel is treated separately. To overcome this problem, a Gaussian Mixture Model (GMM) can be used to introduce a statistical model for BG/FG segmentation [Stauffer and Grimson, 1999]. The GMM was first introduced by Stauffer and Grimson in [Stauffer and Grimson, 1999]. The goal of this model is to ignore noisy background variations by modeling the normal intensity variations of each image pixel over time. This modeling is done by using a Gaussian filter, more precisely a mixture of adaptive Gaussians. The mixture then handles both illumination changes and multiple surfaces, occurring at different time instances at a single pixel location. At time t, it is assumed that the history values $\mathbf{X} = \{X_1, X_2, X_t\}$ of a pixel are known, on which the model is applied on. At each time instance, Gaussians are evaluated to determine which pixels are most likely to correspond to the background and which ones do not. The GMM for pixel

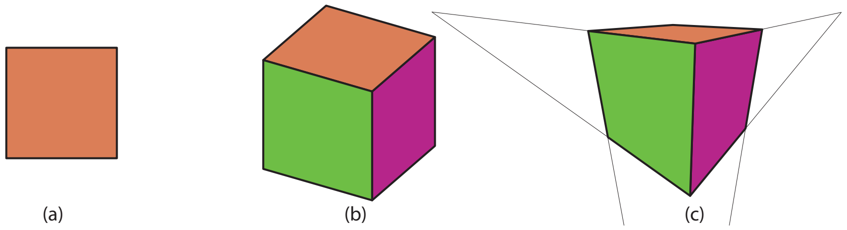


Figure 3.5: A 3D cube projected onto the image plane using (a) orthographic, (b) parallel and (c) perspective projection from three vanishing points.

values $\mathbf{X}$ at time instance t are modeled by

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t}\eta(X_t, \mu_{i,t}, \Sigma_{i,t}), \tag{3.5}$$

where K is the number of Gaussian distributions, $\mu_{i,t}$ and $\Sigma_{i,t}$ are the mean and the covariance matrix of the i^{th} Gaussian distribution, ω is the weight of a pixel belonging to the i^{th} Gaussian function, $\sum_{i=1}^{K} \omega_{i,t} = 1$ and eta is the $d-$dimensional Gaussian probability density function. The value X_t matches one of the existing Gaussian functions i if X_t is within $\sigma < 2.5$ and the parameters are updated in a next step. Gaussians are clustered in BG and FG functions, corresponding pixels are also marked as BG and FG pixels.

3.2 3D Computer Vision

Geometric information does not only describe the relationship between different pixels on the image plane but also between 2D pixels and 3D points [Hartley and Zisserman, 2003]. Capturing an image is established by projecting a 3D scene onto a 2D image plane. The 3D point lies along the camera ray going through the projection of this point. When only the projection is known, the problem of finding the corresponding 3D point is an ill-posed one since this point can be anywhere along the camera ray [Hartley and Zisserman, 2003]. This problem can be solved by using some a-priori information or by having corresponding points from different viewpoints.

As 3D computer vision methods are fundamental aspects throughout the whole presented framework, the following sections describe the geometric relationship between 2D image plane and 3D world coordinate plane as well as computer vision principles and methods to go back and forth between these two coordinate systems.

3.2.1 Perspective Projection

When projecting a 3D scene onto an image plane, this can be done either by using a parallel or a perspective projection [Gonzalez and Woods, 2007]. When using a parallel projection, all the projection lines are orthogonal to the projection plane and the projection center is at infinity. A famous representative of parallel projection types is the orthographic projection which is e.g.

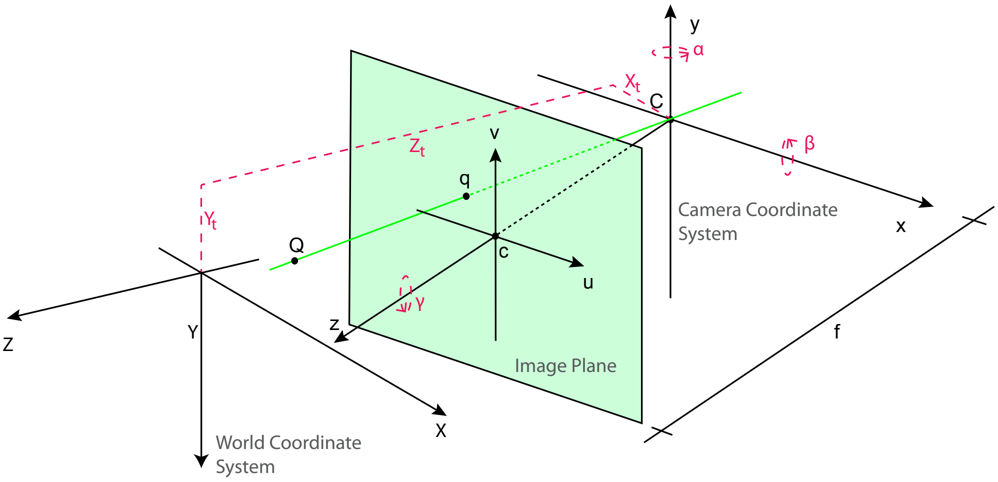


Figure 3.6: Pinhole camera. Mapping of a 3D point $\mathbf{Q}$ onto its 2D counterpart $\mathbf{q}$ on the image plane. The intrinsic parameters are described by f and $\mathbf{c}$, the extrinsics by $\mathtt{R} = \mathtt{R}_\alpha \mathtt{R}_\beta \mathtt{R}_\gamma$ and $\mathbf{t} = (X_t, Y_t, Z_t)^T$.

used for floor plans. It shows a 3D scene or a 3D object from several views. Different to orthographic projection, the perspective projection uses vanishing points which makes objects closer to the viewpoint look larger than objects farther away. Vanishing points represent the point on an image plane where parallel lines of a 3D scene converge. Depending on the viewpoint, the number of vanishing points can change between one and three. Figure 3.5 shows a cube projected on the image plane using (a) orthographic, (b) parallel and (c) perspective projection from three vanishing points [Ma et al., 2003].

When talking about 3D computer vision and projection in this work, it is meant to be a perspective projection between 3D scene and 2D image plane. The corresponding camera model is called pinhole camera model which in combination with a lens distortion model can be used as a fair approximation for conventional cameras [Hartley and Zisserman, 2003]. The principles are shown in Figure 3.6. For simplification, the image plane is mirrored along the z-axis of the camera coordinate system. Let $\mathbf{Q} = (x_Q, y_Q, z_Q)^T \in \mathbb{R}^3$ be a point in camera space and $\mathbf{q} = (u_q, v_q)^T \in \mathbb{R}^2$ its projected point on the image plane. The optical center is denoted as $\mathbf{C}$, the principal point on the image plane as $\mathbf{c} = (u_0, v_0)$. The distance between the optical center and the image plane is described by the focal length f of a camera. The projection of point $\mathbf{Q}$ is then given by the intersection of the projection line and the image plane, where the projection line must pass through both the optical center and the 3D point $\mathbf{Q}$ [Hartley and Zisserman, 2003].

When assuming that the origin of the image plane $\mathbf{c}$ corresponds with the origin of the camera coordinate system $\mathbf{C}$, the mapping between the projected point $\mathbf{q}$ of a point $\mathbf{Q}$ can be described by [Hartley and Zisserman, 2003]

$$\begin{pmatrix} u_q \\ v_q \end{pmatrix} = \frac{f}{z_Q} \begin{pmatrix} x_Q \\ y_Q \end{pmatrix} . \tag{3.6}$$

Due to camera geometrics, the assumption that the principal point corresponds with the camera coordinate system origin, a conversion between the two systems is necessary. This can be done by including the principal point in Equation 3.6. The principal point in combination with the focal length describe the relationship between the image plane and the camera reference frame and are also called intrinsic parameters. By re-formulating Equation 3.6, introducing the principle point and transforming the result in homogeneous coordinates, the 3x4 camera calibration matrix K is obtained. It is defined by

$$\mathtt{K} = \begin{bmatrix} f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{3.7}$$

In homogeneous coordinates, the mapping between camera coordinate system and image plane is then established by

$$\mathbf{q} = \mathtt{K}\mathbf{Q}. \tag{3.8}$$

When going from 3D space to 2D image coordinates, the point to be projected must not be defined by using camera coordinates but by using coordinates from the real 3D world. The pinhole camera can therefore also be extended by so called extrinsic parameters which define the relationship between camera plane and world coordinate system [Hartley and Zisserman, 2003]. The extrinsic parameters consist of a 3x3 rotation matrix R, obtained from the three angles denoted by α, β and γ in Figure 3.6, and a translation vector $\mathbf{t} = (X_t, Y_t, Z_t)^T$. Together with the camera intrinsics they set up the 3x4 camera matrix P. Let $\mathbf{Q} = (X_Q, Y_Q, Z_Q)^T \in \mathbb{R}^3$ now be described by world coordinates. The mapping between $\mathbf{Q}$ and $\mathbf{q}$ is then given by

$$\begin{aligned} \mathbf{q} &= \mathtt{P}\mathbf{Q} \\ &= \mathtt{K}\begin{pmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}\mathbf{Q}. \end{aligned} \tag{3.9}$$

3.2.2 Camera Auto-Calibration from Vanishing Points

Let $\mathbf{p} = (p_1, p_2, p_3)$, $\mathbf{q} = (q_1, q_2, q_3)$, $\mathbf{r} = (r_1, r_2, r_3)$ be three vanishing points which are orthogonal to each other. Vanishing points can be determined by first extracting line segments (e.g. by applying the Canny edge detector) from an image. Then, line segments are clustered based on their orientations and the three dominant orientations which hold the maximum number of lines are obtained by using RANSAC. Under the assumption of squared pixels, the IAC, denoted as ω, has the form of

$$\omega = \begin{bmatrix} \omega_1 & \omega_2 & \omega_4 \\ \omega_2 & \omega_3 & \omega_5 \\ \omega_4 & \omega_5 & \omega_6 \end{bmatrix} \tag{3.10}$$

After the calculation of three vanishing points, ω can be determined. By having three orthogonal vanishing points, the following equations arise.

$$\begin{aligned} \mathbf{p}^T \cdot \omega \cdot \mathbf{q} &= 0 \\ \mathbf{q}^T \cdot \omega \cdot \mathbf{r} &= 0 \\ \mathbf{p}^T \cdot \omega \cdot \mathbf{r} &= 0 \end{aligned} \tag{3.11}$$

By expanding these equations, a linear equation system can be formed by

$$\mathtt{A}^T = \begin{bmatrix} p_1 \cdot v_1 & p_1 \cdot r_1 & q_1 \cdot r_1 & 0 & 1 \\ p_1 \cdot q_2 + p_2 \cdot q_1 & p_1 \cdot r_2 + p_2 \cdot r_1 & q_1 \cdot r_2 + q_2 \cdot r_1 & 1 & 0 \\ p_2 \cdot q_2 & p_2 \cdot r_2 & v_2 \cdot r_2 & 0 & -1 \\ p_1 \cdot q_3 + p_3 \cdot q_1 & p_1 \cdot r_3 + p_3 \cdot r_1 & q_1 \cdot r_3 + q_3 \cdot r_1 & 0 & 0 \\ p_2 \cdot q_3 + p_3 \cdot q_2 & p_2 \cdot r_3 + p_3 \cdot r_2 & q_2 \cdot r_3 + q_3 \cdot r_2 & 0 & 0 \\ p_3 \cdot q_3 & p_3 \cdot r_3 & q_3 \cdot r_3 & 0 & 0 \end{bmatrix} \tag{3.12}$$

By assuming zero skew, which is equal to setting $\omega_2 = 0$, the elements of ω are found by solving

$$\mathtt{A} \cdot (\omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6)^T = \mathbf{0}^T. \tag{3.13}$$

The intrinsic parameters are directly related to ω by $(\mathtt{KK}^T)^{-1} = \omega$. By applying the Cholesky decomposition [Press et al., 1992], the intrinsic parameters can be extracted from the IAC [Hartley and Zisserman, 2003].

3.2.3 Epipolar Geometry

As can be seen in Section 3.2.1, projecting a 3D point onto an image plane is well defined [Hartley and Zisserman, 2003]. An ill-posed problem arises, when the corresponding 3D point should be found for a given 2D point, since the 3D point must lie somewhere on the projection ray but the exact location cannot be determined. This problem can only be solved when having (i) additional priors about the 3D scene (e.g. 3D models) or (ii) two or more corresponding 2D points from several views. The 3D relationship between two cameras is expressed by the so called epipolar geometry which can be described by the 3x3 fundamental matrix $\mathtt{F}$. Let i and j be the indices of two cameras within a camera network which can consist of two or more cameras. As can be seen in Figure 3.7, the epipolar geometry holds two important aspects when the relative orientation between two camera views is known [Hartley and Zisserman, 2003].

- Given two corresponding 2D image points $\mathbf{x}_i$ and $\mathbf{x}_j$, the 3D point $\mathbf{X}$ is described by the intersection of the projection line passing through $\mathbf{x}_i$ and $\mathbf{C}_i$ and the projection line passing through $\mathbf{x}_j$ and $\mathbf{C}_j$.
- When only one of the 2D image points, $\mathbf{x}_i$ or $\mathbf{x}_j$, is known, the search space of the corresponding point in the other view can be shrinked down since it must lie on the known epipolar line. The epipolar line of camera two is the line passing through C_i and the projection of C_j onto the image plane of camera two and vice versa. The projections of the cameras' centers onto the image plane of the other camera are called epipoles, represented by e_i and e_j in Figure 3.7.

Let $\mathbf{x}_i$ be a point from the first view and $\mathbf{x}_j$ be a point from the second view, then the fundamental matrix is defined by

$$\mathbf{x}_j^T \mathtt{F} \mathbf{x}_i = 0. \tag{3.14}$$

As can be seen from this equation, the fundamental matrix can be determined from eight corresponding points. When the intrinsic parameters $\mathtt{K}_i$ and $\mathtt{K}_j$ are already known, the fundamental

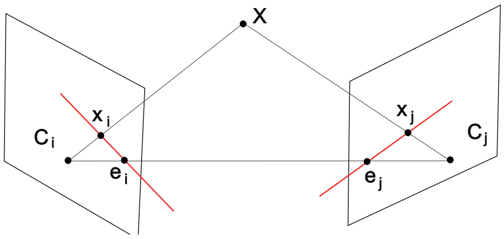


Figure 3.7: Epipolar geometry.

matrix in Equation 3.14 can be replaced by the essential matrix $\mathbf{E} = \mathtt{K}_i^T \mathtt{F} \mathtt{K}_j$ [Hartley and Zisserman, 2003]. The essential matrix can then be used to obtain the relative orientation $\Delta\mathtt{R}$ and $\Delta\mathbf{t}$ between the two cameras with indices i and j by

$$\mathtt{E}_{ij} = [\Delta\mathbf{t}_{ij}]_\times \Delta\mathtt{R}_{ij} = \Delta\mathtt{R}_{ij}[\Delta\mathtt{R}_{ij}^T \Delta\mathbf{t}_{ij}]_\times \tag{3.15}$$

The relative rotation and translation can then be extracted from the essential matrix [Ma et al., 2003]. As there are four solutions for $\Delta\mathtt{R}_{ij}$ and $\Delta\mathbf{t}_{ij}$, the solution where a positive depth is gathered for any projected point must be picked, as described in [Hartley and Zisserman, 2003]. The epipolar geometry is used in Chapter 4 of this dissertation for camera auto-calibration.

3.3 Optimization and Classification Methods

Due to changes in illumination, a changing viewpoint, changing shape and color of objects and also due to blur and noise in an image, similar scenes and objects look different when projected on various images [Sonka et al., 2008]. Since it is not feasible in practice in the field of computer vision to consider all of these variations, it is important for a number of applications (i) to be able to obtain similar objects or scenes for given ones and (ii) to be able to handle the mentioned local changes which means to enforce an image region to be consistent with neighboring regions in terms of time, space or context, without letting the local noise avoid obtaining the globally correct result. The following sections describe two methods used for such scenarios, namely the MRF to enforce spatial, temporal or contextual consistency and the RF to perform classification on image regions or objects afflicted with noise in the image.

Graphical models are used for performing inference in Section 6.1, 6.2 to get the best feasible combination of vehicle poses over time and in Section 5.1 to get the best combination of structure labels for 3D reconstruction. In Section 6.2, RFs are used to determine the pose of a car seen in video sequences, where the RF classifier is trained on existing 3D models.

3.3.1 Markov Random Field

A discrete Markov Random Field (MRF) is a statistical framework used in computer vision to analyze spatial or contextual dependencies among regions of an image [Nowozin and Lampert, 2011]. The framework is used for many tasks where a single label from a discrete set of predefined labels is assigned to a set of given regions in an image. The optimization of an MRF

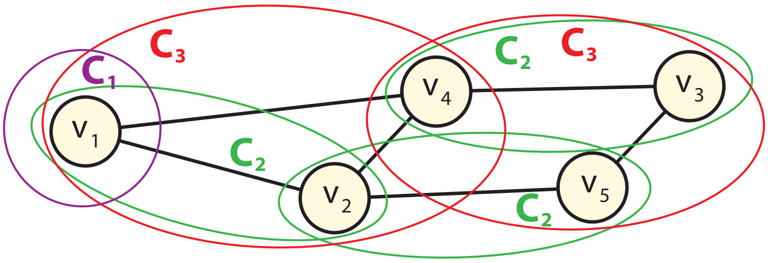


Figure 3.8: Sample graph and example cliques on the graphs, denoted as $\mathcal{C}_k$, where k is the size of the clique.

delivers the globally best combination of labels for all the regions which are analyzed. Depending on the application, the regions may be e.g. points, segments or objects and the labels may be a variety of e.g. object classes, depth values, or color and intensity values. The MRF is described by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2 \dots v_N\}$ are the N nodes or vertices on the graph (correspond to image regions) and $\mathcal{E} \subset \binom{\mathcal{V}}{2}$ describe the edges between these nodes [Nowozin and Lampert, 2011]. Figure 3.8 shows an example graph. On the graph, two nodes $v_i, v_j \in \mathcal{V}$ are neighbors if $(v_i, v_j) \in \mathcal{E}$. The neighborhood of a node v_i is therefore given by $\mathcal{N}(v_i) = \{v_j : (v_i, v_j) \in \mathcal{E}\}$, where $i \neq j$. In Figure 3.8, the neighbors of node v_2 are given by $\mathcal{N}(v_2) = \{v_1, v_4, v_5\}$. The model is a representative of the so called undirected graphical models which means that $v_i \in \mathcal{N}(v_i) \Leftrightarrow v_j \in \mathcal{N}(v_j)$. A clique is defined as an ordered and complete subset of nodes on the graph and can exist from one, two, three or more vertices. The maximal clique is then a clique consisting of the maximal number of nodes while the graph still remains correct and complete. In Figure 3.8, there can be formed multiple cliques, e.g. $\mathcal{C}_1 = \{v_1, v_2, v_3, v_4, v_5\}$, $\mathcal{C}_2 = \{(v_1, v_2), (v_2, v_4), \dots\}$ and $\mathcal{C}_3 = \{(v_1, v_2, v_4), (v_2, v_4, v_3), \dots\}$. In other words, all nodes in a clique must be neighbor of all other members in the clique. Note that not all possible cliques are shown in the Figure due to visibility constraints. Each node is then assigned a random variable from a set of them, denoted by $\mathbf{X} = \{X_1, X_2 \dots X_N\}$ which can take a label configuration $\mathbf{x} = \{x_1, x_2 \dots x_N\}$ from a set of M labels $\mathcal{L} = \{l_1, l_2 \dots l_M\}$. The set $\mathbf{X}$ is also known as random field. For this discrete label set, the probability that random variable X_i takes label x_i is denoted by $P(x_i)$, the joint probabilty that a configuration $\mathbf{x}$ is assigned to a set of random variables $\mathbf{X}$ is denoted by $P(\mathbf{x})$. The random field $\mathbf{X}$ is then said to be an MRF, if

$$\begin{aligned} P(x_i) &> 0 \qquad \forall x_i \\ P(x_i | \mathbf{x}_{\mathcal{V}\setminus\{i\}}) &= P(x_i | \mathbf{x}_{\mathcal{N}(i)}), \end{aligned} \tag{3.16}$$

which in other words means that the probability for a value taken from a set of labels to be assigned to a random variable must be greater than zero and that each random variable only depends on random variables through its neighbors on the graph. To formulate the global joint probability from multiple local functions, the MRF can be characterized by a Gibbs distribution, as stated in the Hammersley-Clifford Theorem [Hammersley and Clifford, 1971]. The joint distribution is then given by the product of all these non-negative functions over the maximal cliques of the graph, which can be written as

$$P(\mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{T} \sum_{C \in \mathcal{C}} V_C(x_C) \right), \tag{3.17}$$

where Z is a normalizing constant, called the partition function, T is a constant called the temperature and V_C is the corresponding clique potential. The energy function $E(\mathbf{x})$ can then expressed by

$$\begin{aligned} E(\mathbf{x}) &= \sum_{C \in \mathcal{C}} V_C(x_C) \\ &= \sum_{\{i\} \in \mathcal{C}_1} V_1(x_i) + \sum_{\{i,j\} \in \mathcal{C}_2} V_2(x_i, x_j) + \dots \end{aligned} \tag{3.18}$$

Optimization of an MRF equals to finding the maximum joint probability on the graph or performing inference on the graph [Nowozin and Lampert, 2011]. Maximizing the joint probability can be established by minimizing the energy, given in Equation 3.18. This equals to minimizing all the clique potentials and the most likely configuration $\hat{\mathbf{x}}$ is then given by

$$\begin{aligned} \hat{\mathbf{x}} &= \operatorname*{argmax}_{\mathbf{x}} P(\mathbf{x}) \\ &= \operatorname*{argmax}_{\mathbf{x}} \frac{1}{Z} \exp\left(-E(\mathbf{x}) \right) \\ &= \operatorname*{argmin}_{\mathbf{x}} E(\mathbf{x}). \end{aligned} \tag{3.19}$$

Performing exact inference is in practice a hard task and therefore time consuming. Fortunately, there are several implementations and algorithms which are able to perform an approximated optimization of the MRF, e.g. Loopy Belief Propagation (LBP) [Frey and MacKay, 1998], Iterated Conditional Modes (ICM) [Besag, 1986], or also graph cut [Boykov and Jolly, 2001]. When using graph cuts, it must be noticed that the cliques must be reduced to binary ones in order to perform a global optimization of the graph. Due to computational complexity, the algorithms are mostly used on computer vision problems which can be solved by double or triple cliques. An MRF is called pairwise if all the cliques in the random field are of size two. This model is the most common one used in the field of computer vision. Each node is represented by a pixel and the labeling is solved with pairwise clique potentials of neighboring pixels. The joint probability is then given by unary potentials $V_1(x_i)$ and binary potentials $V_2(x_i, x_j)$. The unary potential is defined by the observation af a pixel taking a certain label and the binary potential is called the smoothness term between these pairwise nodes. For further reading on graphical models see [Nowozin and Lampert, 2011].

3.3.2 Random Forest

Many tasks in computer vision (e.g. object detection and recognition, segmentation) rely on efficient and accurate algorithms for classification or regression [Gall et al., 2011a]. Classification, or clustering, is the task of separating a dataset in clusters or classes based on their

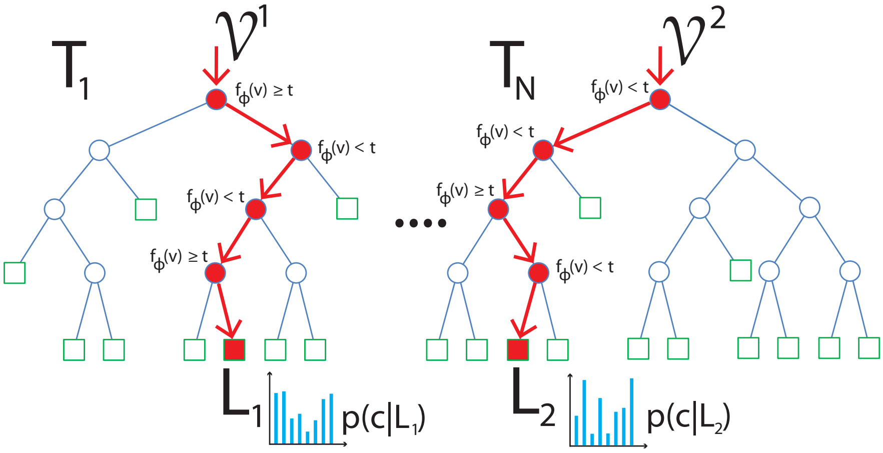


Figure 3.9: Principle of a random forest consisting of N binary decision trees, where each tree contains multiple split nodes and leaf nodes. For a classification task and given a set of input feature vectors $\mathcal{V}^\rangle$, each tree T_i gives back the probability of this set being classified as class c. For regression the response variables are continuously. The probability is found by going through the graph from top to bottom and evaluating all the split functions f_ϕ until the path terminates at leaf node L_i.

discrete or categorical response variable. In case the response variables are defined in continuous space, the task is called regression [Gall et al., 2011a]. Both classification and regression can be performed by a so called Random Forest (RF) which is a combination of N binary decision trees $\mathcal{T} = \{T_1, T_i \ldots T_N\}$ and was first introduced by Breiman in 2001 [Breiman, 2001]. The principle of a random forest can be seen in Figure 3.9.

A single tree T_i consists of split nodes (denoted as circles) and leaf nodes (denoted as rectangles) [Breiman, 2001]. The split node on the first depth level of the tree is called root node. Let $\mathcal{V} = \{\mathbf{v}_j\}$ be a given set of feature vectors. A feature vector $\mathbf{v}_j$ is then sent through the tree from top to bottom and evaluated at a split node for each depth level of the tree using a split function f_ϕ. The path of the feature vector is then terminated at the leaf node L_i of a tree. In order to perform classification tasks, the goal for each tree is to estimate the probability of $\mathbf{v}_j$ belonging to a certain class c, described by $p(c|L_i)$. When regression should be performed, the goal is to estimate the distribution over the continuous parameter $x \in \mathbb{R}^H$.

Figure 3.10 shows an example of a separation of features in $\mathbb{R}^2$ in two different classes. Each line represents one split function f_ϕ which can be seen as a weak learner [Gall et al., 2011a]. Merging two lines means going down one depth level of a tree. Having all lines merged is equal to having reached the terminal node L_i of a tree. As can be seen, the final merged line represents the line which best separates the data into the given number of classes. It can also be seen that split functions that do not help to obtain a better separation boundary are not considered in the final boundary decision. Having a decision boundary does in fact not always mean that all the features from the input data are classified correctly but most of them are. Obtaining a good decision boundary depends on how to set the learning parameters of the tree. It is described in

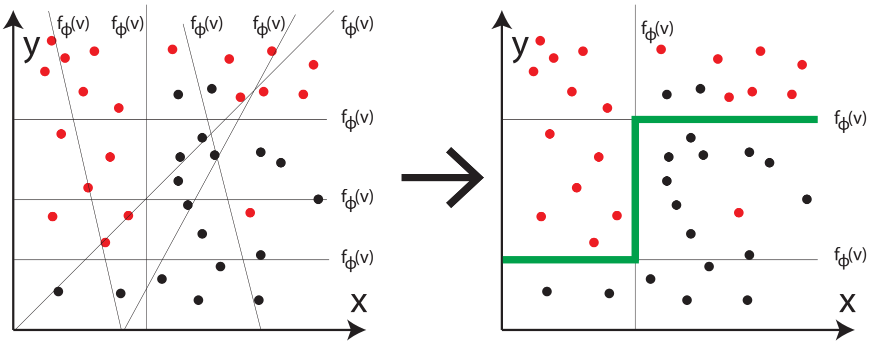


Figure 3.10: Binary decision tree example for multiple features in $\mathbb{R}^{\nvDash}$ and two classes, denoted by red and black points. By combining multiple weak learners f_ϕ, a stronger classifier is obtained and therefore a more discriminative boundary between the two classes is found.

the following how to set these parameters in order to obtain a good and globally valid decision boundary. In order to avoid overfitting, which can occur when estimating the decision boundary using a single tree, multiple trees are used.

In a first step, an RF must be trained. To handle the large amount of data and perform the training efficiently, randomness is introduced by

- training each tree i on a random subset $\mathcal{V}^i = \{\mathbf{v}_r\}, \mathbf{v}_r \in \mathcal{V}$ and
- randomly using split functions at each node.

Having a set of randomly picked split function parameters $\mathbf{\Phi} = \{\phi_k\}$, a split function is defined by f_ϕ, where $\phi \in \mathbf{\Phi}$. The split function is used to divide the set of input vectors $\mathcal{V}^i$ into $\mathcal{V}_l^i$ and $\mathcal{V}_r^i$, following the left and right branch of the tree respectively by

$$\begin{aligned} \mathcal{V}_l^i(\phi) &= \{\mathbf{v}_r | f_\phi(\mathbf{v}_r) < t\} \\ \mathcal{V}_r^i(\phi) &= \mathcal{V}^i \backslash \mathcal{V}_l^i, \end{aligned} \tag{3.20}$$

where t is a threshold [Gall et al., 2011a]. The splitting function parameters ϕ and the threshold t must be chosen so that the information gain g is maximized by exploiting the Shannon entropy $\mathcal{H}$ by

$$\begin{aligned} \hat{\phi} &= \underset{\phi}{\operatorname{argmax}}\, g(\phi, \mathcal{V}^i), \\ g(\phi, \mathcal{V}^i) &= \mathcal{H}(\mathcal{V}^i) - \sum_{s \in \{l,r\}} \frac{|\mathcal{V}_s^i(\phi)|}{\mathcal{V}^i} \mathcal{H}(\mathcal{V}_s^i(\phi)). \end{aligned} \tag{3.21}$$

The set $\mathcal{V}^i$ is split until a stopping criteria is met (e.g. maximum tree depth, minimum entropy gain) and the number of features used for evaluation depends on the task to be performed. By using only one feature, the tree is built up extremely randomized [Geurts et al., 2006]. Taking few features may introduce an underfitting, taking too many features may produce overfitting of the optimization task.

In order to perform classification or regression based on the trained RF, the given input data can vary from the training data. The final probability of a randomly picked feature vector $\mathbf{v}'$ voting for a class c is then obtained by averaging all probabilities obtained from the trees by

$$p(c|\mathbf{v}') = \frac{1}{N}\sum_{i=1}^{N} p(c|L_i). \tag{3.22}$$

Similarly, the probabilities can also be multiplied or probability distributions can be averaged or multiplied using the distributions from all the trees [Gall et al., 2011a]. To conclude, RFs describe an efficient and fast algorithm for classification and regression tasks which must be designed carefully due to the number of parameters which need to be set.

3.4 Summary

This chapter describes the basic principles, algorithms and tools which are used for the accompanying implementations described in Chapters 4-6. The aim of a computer vision based 3D scene understanding framework is to extract semantic information from 2D images in order to perform 3D reasoning about the scene. Since a single pixel does not have any semantic information, the geometric relationship to pixels within the image or to corresponding 3D points must be known.

First, different representations of such geometric information within an image are described. Point, line and region features are discussed and two representatives for all three feature types, which are also used in the following chapters, are described in detail. These representatives are the Harris corner detector and SIFT for point features, the Sobel filter and the Canny edge detector for edge features and superpixel segmentation methods as well as background modeling techniques for region features.

As geometric relations do not only occur between pixels in an image but also between points on an image plane and their corresponding 3D points, the second part of this chapter describes the basic relationship between image plane, camera plane and 3D scene. It also shows how points can be projected back and forth between these coordinate systems using multiple viewpoints. The section concludes with outlining the principles of epipolar geometry and how a camera can be calibrated from three orthogonal vanishing points detected in the image.

The third part of this chapter deals with classification and optimization algorithms which are applied to computer vision problems. In computer vision, images taken from the same scene but with e.g. different capturing devices, under different lighting conditions or from a different viewpoint may appear quite different in terms of color or intensity the pixels are holding. Apart from that, the geometric relationship can also slightly change. To deal with such variations in computer vision applications and to get globally correct results (e.g. for segmentation, object detection or recognition tasks), an optimization algorithm, namely MRFs, and a method for classification and regression, namely RFs, are described.

CHAPTER 4

Camera Auto-Calibration from Traffic Participants

This chapter describes two camera-auto calibration methods which exploit redundant scene information obtained by observing a traffic scene in order to increase the accuracy of such auto-calibration methods.

First, a new approach for automatically calibrating a camera network is described in Section 4.1. It gathers intrinsic and extrinsic parameters from a pedestrian, who is walking upright on the ground plane. By combining redundant information from multiple time instances, vanishing points are calculated from the pedestrian. Intrinsic parameters are estimated by exploiting these vanishing points. The same feature points of a pedestrian are also taken to calculate each camera's extrinsic parameters within a common coordinate system. The performed experiments outline the accuracy and the practicability. Major parts of this work are described in the accompanying publication [Hödlmoser and Kampel, 2010].

Second, a novel auto-calibration method for a single camera based on traffic scenes is described in Section 4.2. This scene must consist of one or more pedestrians and a zebra-crossing. Vanishing points are obtained from a combination of both static objects (zebra-crossing) and dynamic ones (pedestrians). A horizontal vanishing point and a vanishing line is extracted from a zebra-crossing. Pedestrians walking upright allow estimating the vertical vanishing point and the intrinsic parameters are determined. Extrinsic parameters are estimated by assuming the width of the black and white patterns of the zebra-crossing to be known. By combining static and dynamic calibration objects, the method can be made robust against outliers. The robustness against outliers, the practicability of the method and improved results in terms of accuracy compared to state-of-the-art algorithms are shown in the experiments. These are carried out by using synthetic and real data of different application scenarios. The method is published in [Hödlmoser et al., 2011b].

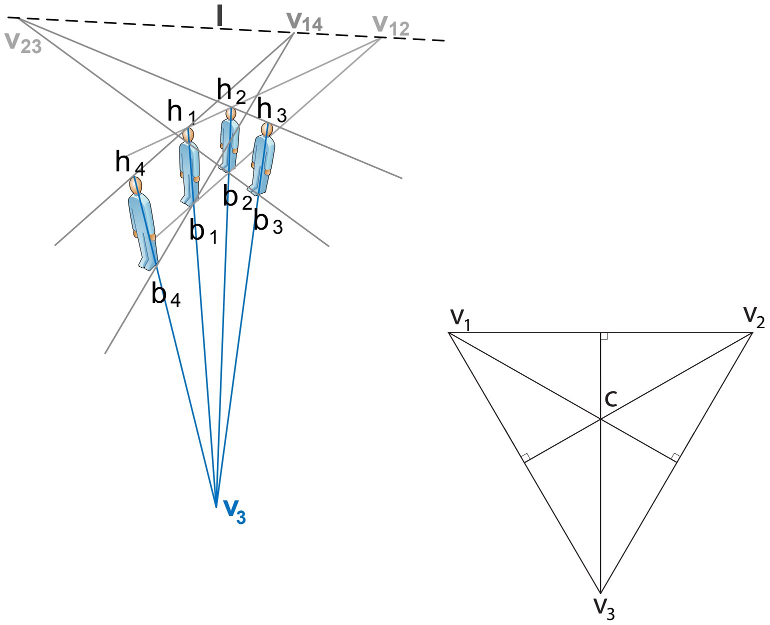
4.1 Multiple Camera Auto-Calibration Using Pedestrians

Humans are the most frequent object to be analyzed in surveillance scenarios [Leotta and Mundy, 2011]. Therefore, pedestrians are predestined to be used as calibration object for self-calibrating a traffic surveillance camera. In this section, a novel and practical approach to determine both intrinsic and extrinsic parameters for a network of surveillance cameras by only analyzing a pedestrian is presented. The method delivers a camera's intrinsic and extrinsic parameters from a pedestrian, who is walking upright on the ground plane. Calibration can then be done when having at least two different time instances of the walking pedestrian. By combining redundant information from more than two time instances, vanishing points are calculated. These vanishing points are determined by extracting top and bottom points of the person. Instead of exploiting a geometric solution as in [Lv et al., 2006], the intrinsic parameters are then extracted from the IAC. In a next step, the extracted top and bottom points are also used as input for gathering the pairwise relative orientations between all cameras within a network. To calculate the scaling factor for the absolute translation of all cameras, the user needs to predefine the height of the walking person.

4.1.1 Vanishing Point Estimation

Given a camera network consisting of $i = 1 \dots N$ cameras, calibrating the whole network includes three steps. First, the camera matrices $\mathtt{K}_i$ need to be determined for each camera. Second, pairwise relative rotations $\Delta\mathtt{R}_{ij}$ and translations $\Delta\mathbf{t}_{ij}$) between two cameras i,j are extracted. The last step is the determination of the absolute rotations $\mathtt{R}_i$ and translations $\mathtt{t}_i$ for each camera in a common 3D coordinate system.

This approach describes the extraction of intrinsic and extrinsic parameters for multiple cameras in a network from a sequence of a walking human. The pedestrian is observed for at least two frames over time $t = 1 \dots T$ and head and bottom (or foot) points are extracted. These points are then taken to calculate two vanishing points $\mathbf{v}_1$ and $\mathbf{v}_3$ and a vanishing line $\mathbf{l}$. A third vanishing point $\mathbf{v}_2$ can be determined by geometric relationships and the IAC can be formed. Figure 4.1b illustrates the triangle spanned by $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$. The orthocenter is denoted as $\mathbf{c}$. By applying the Cholesky decomposition [Press et al., 1992], the intrinsic parameters can be extracted from the IAC. The geometric relationship of the vanishing points and the vanishing line can be seen in Figure 4.1a. It shows the vanishing line $\mathbf{l}$ and a selection of vanishing points of four pairs of instances of a walking human. Head and bottom points are also used to recover the cameras' extrinsic parameters. To obtain a metric scaling of the scene and to extract the scaling factor for the absolute translation of all cameras, the user needs to predefine the height of the walking person. In order to obtain correct vanishing points, there are two constraints related to this approach, namely (i) head and foot points need to have a fixed 3D distance over time and (ii) all head and foot points need to be coplanar respectively. Both constraints are fulfilled by a pedestrian having a constant height walking on an even ground plane.



(a) (b)

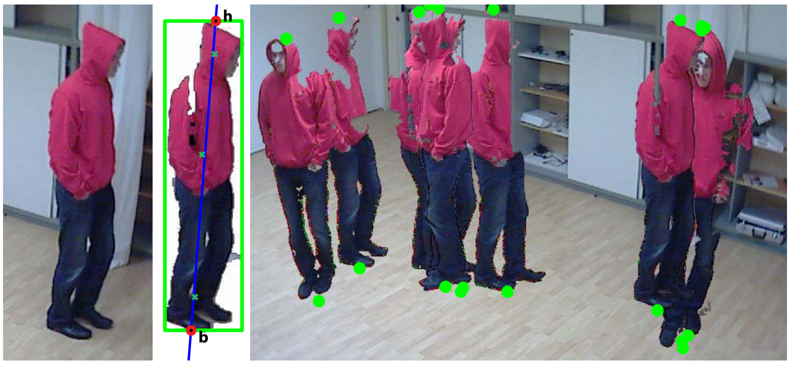
Figure 4.1: (a) Determination of the vanishing line $\mathbf{l}$ and a vertical vanishing point $\mathbf{v}_3$ by using all head and foot points of a walking pedestrian. (b) Triangle spanned by $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$.

4.1.1.1 Determination of Vertical Vanishing Point

To determine a head point $\mathbf{h}_t = (x_h, y_h)$ and a bottom point $\mathbf{b}_t = (x_b, y_b)$ of a walking human at time instance t, the human silhouette must be extracted in a first step which is crucial for all further calculations. A simple background subtraction is done to determine a rough silhouette of the pedestrian. The resulting foreground bounding box is used as input for the GrabCut algorithm [Rother et al., 2004] which performs an accurate extraction of the pedestrian's silhouette from the background.

The bounding box for the determined pedestrian is calculated and divided in an upper and a lower part. The center of mass is calculated for the whole bounding box, for the lower and for the upper part respectively. By using a least squares estimation, a line is fitted through the three centers of mass. This gives a vertical line of the walking human at time instance t. The head and foot points are then given by the intersection of the vertical line and the bounding box. Figure 4.2a shows a random input image from a video sequence used for the experiments, Figure 4.2b shows the extracted person within a rectangular bounding box, its three centers of mass denoted by crosses and its corresponding head and foot points denoted by circles. Figure 4.2c shows the pedestrian extracted for an image sequence and the corresponding head and foot points.

The goal of the next step is to estimate three vanishing points $\mathbf{v}_1 = (x_1, y_1)$, $\mathbf{v}_2 = (x_2, y_2)$ and $\mathbf{v}_3 = (x_3, y_3)$ from head and foot points. According to [Lv et al., 2006], a vertical vanishing point $\mathbf{v}_3$ can be calculated by using the head and foot points of a walking human. Theoretically,



(a) (b) (c)

Figure 4.2: (a) Example input image, (b) extraction of head and foot points and (c) extracted head and foot locations for a sequence of images.

all lines going through corresponding foot and head points of a pedestrian must converge to one vanishing point $\mathbf{v}_3$. Since head and foot location initialization may be noisy, an approximation of the intersection needs to be performed. Using cross ratio in 2D, the relationship between the vertical vanishing point and head and foot points of a pedestrian is given by [Kusakunniran et al., 2009]

$$x_3(y_h - y_b) + y_3(x_b - x_h) \quad = \quad (x_b y_h - x_h y_b). \tag{4.1}$$

This equation is underdetermined and cannot be solved when only having one instance of a pedestrian. When two or more instances of the pedestrian are given, an equation system $\mathbf{A}\mathbf{v}_3 = \mathbf{r}$, where $\mathbf{r}$ is the result vector, can be formed to solve the equation. A least squares solution can be gathered by solving

$$\mathbf{v}_3 = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{r}. \tag{4.2}$$

To make the solution more robust and to exploit redundant information, all $\binom{N}{2}$ possible combinations for N instances of the pedestrian are used for further calculations.

4.1.1.2 Determination of Horizontal Vanishing Points

Having multiple instances of a pedestrian enables the calculation of pairwise vanishing point calculation between each pair of these instances. Similarly to estimating $\mathbf{v}_3$, all possible pairwise combinations are used in this step to exploit redundant information again in order to increase the robustness of the result against outliers. A vanishing point between two instances of a pedestrian at time instance t and time instance $t+1$ can be calculated by $\mathbf{v}_{t,t+1} = \overline{\mathbf{h}_t\mathbf{h}_{t+1}} \times \overline{\mathbf{b}_t\mathbf{b}_{t+1}}$. Theoretically, all vanishing points constructed from such pairs must lie on one line called the vanishing line. Since head and foot location initialization may be noisy and therefore not all vanishing points must lie on one line, the vanishing line is fitted by using the least squares algorithm.

As three vanishing points orthogonal to each other are needed for gathering the intrinsic parameters of a camera, $\mathbf{v}_3$ and $\mathbf{l}$ are used for the determination of two more vanishing points, $\mathbf{v}_1$ and $\mathbf{v}_2$. The second vanishing point $\mathbf{v}_1$ can be determined by taking an arbitrary point on the vanishing line. If the vanishing line has a gradient equal to zero, it is called the horizontal line. Before calculating $\mathbf{v}_2$, the vanishing line needs to be aligned with the horizontal line. This is established by rotating the vanishing line by the negative gradient of $\mathbf{l}$. By this alignment, the whole scene, including all points needed for further calculations, must also be rotated (i.e. $\mathbf{v}_1 = \mathbf{v}'_1, \mathbf{v}_2 = \mathbf{v}'_2, \mathbf{v}_3 = \mathbf{v}'_3$). The principal point $\mathbf{c} = (u_0, v_0)$ can safely be assumed to be the image center and $\mathbf{c}$ is rotated to obtain $\mathbf{c}'$. As $\mathbf{c}'$ is the orthocenter of the triangle spanned by the three vanishing points, $\mathbf{v}'_2$ can be calculated geometrically as also shown in Figure 4.1b. To obtain $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$, the scene is rotated back around the image origin by the negative gradient of the vanishing line.

4.1.2 Camera Parameter Estimation

After the calculation of all three vanishing points, the IAC ω can be determined. According to Section 3.2, ω can be determined from three orthogonal vanishing points under the assumption of squared pixels and zero skew. The intrinsic parameters are directly related to ω by $(\mathtt{KK}^T)^{-1} = \omega$. By applying the Cholesky decomposition [Press et al., 1992], the intrinsic parameters can be extracted from the IAC.

Head and foot points of the walking pedestrian are also used to recover the scene structure which in other words means the determination of the cameras' extrinsic parameters. Given a camera network consisting of $i = 1 \dots N$ cameras, the goal is to estimate each camera's absolute rotation $\mathtt{R}_i$ and translation $\mathbf{t}_i$ in a common coordinate system. In order to do so, the relative orientations between camera pairs are calculated sequentially. The orientation between two arbitrary cameras, providing the indices i and j, consists of a relative rotation $\Delta\mathtt{R}_{ij}$ and a relative translation $\Delta\mathbf{t}_{ij}$. As described in Section 3.2.3, the relative rotation and translation between two cameras having indices i and j are extracted from the essential matrix.

After pairwise calibration of the camera network, the absolute orientation for each camera must be determined. Therefore, one camera is assumed to be the common coordinate origin. Having two cameras i, j and their relative rotations $\Delta\mathtt{R}_{ij}$ and translations $\Delta\mathbf{t}_{ij}$, the absolute rotation $\mathtt{R}_j$ and absolute translation $\mathbf{t}_j$ of camera j is calculated by

$$\mathtt{R}_j = \Delta\mathtt{R}_{ij}\mathtt{R}_i,\ \mathbf{t}_j = \Delta\mathbf{t}_{ij} + \Delta\mathtt{R}_{ij}\mathbf{t}_i \tag{4.3}$$

Since the essential matrix is only determined up to a scale factor λ, this factor must be calculated to gain a scaled translation vector. Therefore, corresponding pedestrian head and foot points are triangulated in the normalized world coordinate system at time instance t to obtain head points $\mathbf{H}_t$ and foot points $\mathbf{B}_t$. The scale factor can be calculated by

$$\lambda = \frac{1}{T}\sum_{t=1}^{T}\frac{M}{||\mathbf{H}_t, \mathbf{B}_t||^2}, \tag{4.4}$$

where M is the real height of the pedestrian in metric 3D space which must be provided by the user. The rotation and translation parameters are optimized by running the sparse bundle

Cam #		f (pixels)	u_0 (pixels)	v_0 (pixels)	$\Delta\mathtt{R}$ (degrees)	$\Delta\mathbf{t}$ (mm)	μ RE (10^{-3}pixels)	σ RE (10^{-3}pixels)
01	CV	659.40	320.00	240.00	(00.00, 00.00, 00.00)	000.00	0.00	0.19
	RV	660.00	320.00	240.00	(00.00, 00.00, 00.00)	000.00	–	–
02	CV	659.46	319.98	239.99	(60.00, 00.00, 00.00)	371.35	0.00	0.25
	RV	660.00	320.00	240.00	(60.00, 00.00, 00.00)	371.30	–	–
03	CV	659.42	320.00	239.99	(110.00, 00.00,-00.00)	666.64	0.01	0.40
	RV	660.00	320.00	240.00	(110.00, 00.00, 00.00)	666.55	–	–
04	CV	659.79	319.95	239.99	(135.00,-00.00, 00.00)	782.55	0.01	0.24
	RV	660.00	320.00	240.00	(135.00, 00.00, 00.00)	782.54	–	–
05	CV	659.36	319.99	239.99	(179.99,-00.00,-00.00)	854.60	0.00	0.30
	RV	660.00	320.00	240.00	(180.00, 00.00, 00.00)	854.53	–	–
06	CV	659.43	320.03	240.00	(224.99, 00.00, 00.00)	786.95	0.01	0.28
	RV	660.00	320.00	240.00	(225.00, 00.00, 00.00)	786.93	–	–
07	CV	659.34	319.96	239.98	(275.01,-00.00, 00.00)	603.98	0.02	0.96
	RV	660.00	320.00	240.00	(275.00, 00.00, 00.00)	603.79	–	–
08	CV	659.36	320.03	240.00	(290.00,-00.00,-00.00)	492.78	0.00	0.30
	RV	660.00	320.00	240.00	(290.00, 00.00, 00.00)	492.76	–	–
09	CV	659.29	319.98	239.99	(304.99, 00.00,-00.00)	369.46	0.01	0.37
	RV	660.00	320.00	240.00	(305.00, 00.00, 00.00)	369.48	–	–
10	CV	659.36	319.97	239.99	(339.99, 00.00,180.00)	134.51	0.00	0.22
	RV	660.00	320.00	240.00	(340.00, 00.00,180.00)	134.52	–	–

Table 4.1: Comparison of calculated intrinsic / extrinsic parameters and their reference parameters of seven synthetic cameras. Additionally, reprojection errors are shown.

adjustment algorithm over all cameras within the network, as proposed in [Lourakis and Argyros, 2004]. This algorithm simultaneously minimizes the reprojection error and the error of all absolute orientations, when intrinsic parameters are fixed.

4.1.3 Experiments

To show the accuracy of the proposed method, an evaluation using synthetic and real data is performed in the following. As this section should show the precision of the algorithm, it is first evaluated by using a synthetically generated scene. Additionally, the impact of the noisy measurements of top and bottom points on the calibration results is shown. The algorithm is secondly tested on a real world scenario in order to show the practical applicability. Reprojection errors as well as a comparison between results from the proposed approach and both synthetic and real world ground truth data are presented.

4.1.3.1 Synthetic Data

For evaluation purposes, a 3D scene is generated using OpenGL and ten top and bottom points are randomly positioned within the scene. The distance between top and bottom locations is 56 millimeters, as the model figure of the experiments using real images is exactly that height. Ten cameras are located randomly within the observed scene of 360 degrees where all top and bottom points can be seen from each camera. All cameras have the same focal length and principal point. The positions of the cameras and the calculated camera positions are shown in Table 4.1. The relative rotation $\Delta\mathtt{R}$ and the relative translation $\Delta\mathbf{t}$ for each camera present the offset to reference camera one and are represented by the three angles pitch, roll, yaw and the

σ	f(pixels)	u_0(pixels)	v_0(pixels)	$\Delta\mathtt{R}$(degrees)	$\Delta\mathbf{t}$(mm)
0.5	638.36	323.50	240.27	(340.22 -00.08 179.76)	123.34
1.0	621.33	322.89	239.89	(339.76 00.39 173.98)	102.67
1.5	602.49	309.30	238.55	(340.59 -00.54 175.33)	99.31

Table 4.2: Intrinsics and relative orientation between reference camera one and camera ten when introducing noise.

Euclidean distance to the origin. As can be seen, both the intrinsic and the extrinsic parameters can be extracted precisely. Reference values (RV) from the OpenGL scene are compared to calculated values (CV). All cameras offer a difference between calculated and reference focal length of $< 0.12\%$. The maximum difference of the relative translation is 0.03%, measured at camera seven. Since the top and bottom locations are calculated having no noise, it can be seen that the mean and the standard deviation of the reprojection error (represented by μ RE and σ RE) is nearly zero. In a next step, the height of the synthetically generated distance between top and bottom locations is compared and evaluated. The mean height of all ten top/bottom pairs is 56.00 millimeters, having a standard deviation of 2.0248e-004.

To proof the robustness of the algorithm, the method is run again by using a varying standard deviation on the top and bottom points. Table 4.2 shows the results obtained when using a raising standard deviation on camera ten. As can be seen, intrinsic parameters and the relative rotations remain stable (maximum $\Delta f = 8.71\%$, maximum $\Delta u_0 = 3.34\%$, maximum $\Delta v_0 = 0.60\%$, maximum $\Delta\mathtt{R} = (0.07\%, 0.10\%, 3.34\%)$) whereas the distance between camera one and camera ten gets smaller when noise is introduced (maximum $\Delta\mathbf{t} = 26.17\%$).

The computation time of the whole calibration procedure using 2,3,4,5 and 10 cameras is 1.5, 2.7, 4.3, 5.6 and 13.8 seconds which means that by each camera used in the network, the computation time increases by approximately 1.4 seconds which is negligible in a calibration procedure.

4.1.3.2 Real Data

In the following, real data is used for evaluating the proposed calibration method. In order to obtain foot- and headpoints of pedestrians, the distance between pedestrians and camera centers must not be above 10-15 meters. As there is no dataset publicly available showing pedestrians from this distance but from multiple synchronized viewpoints, new datasets are generated for evaluation. The experiments are divided into two different setups. In a first setup, three cameras are capturing a moving model figure (Figure 4.3a), having a height of 56 mm. Three Unibrain Fire-i webcams are used in order to enable observing the scene from 360 degrees. In a second setup, two cameras (AXIS M1031-W) mounted on an indoor wall, are capturing a walking human (Figure 4.3b) having the height of 195cm.

For the experiments, ten subsequent moving positions of the model figure and ten subsequent walking positions of the pedestrian are used for calibrating the camera network. Table 4.3 shows the calculated intrinsic parameters using the proposed method. The ground truth (denoted as GT in the table) for the intrinsics is given by the data sheet of the used cameras. Since the

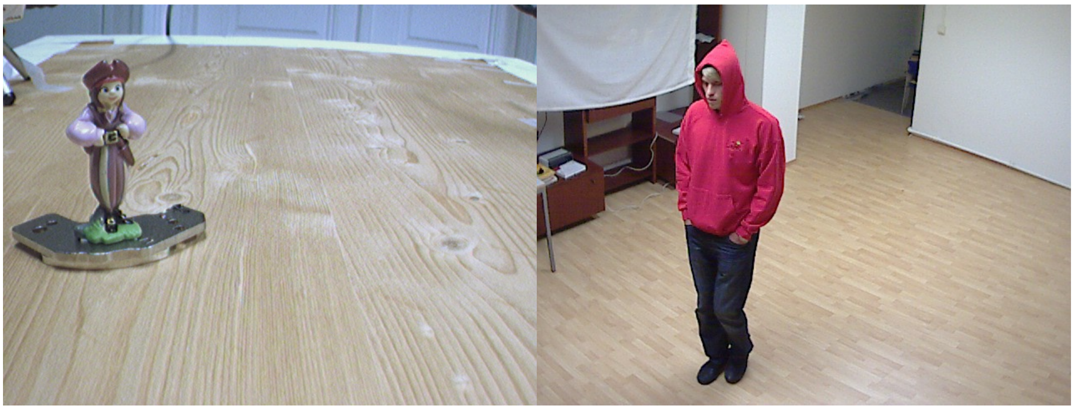

(a) (b)

Figure 4.3: Input data for the experimental setup using real data: (a) model figure and (b) walking human.

	f(pixels)	u_0(pixels)	v_0(pixels)	$\Delta\mathtt{R}$(degree)	$\Delta\mathbf{t}$(mm)	μ RE(pixel)	σ RE(pixel)
Model Figure							
GT	746.67	320.00	240.00	–	–	–	–
cam01	766.01	331.96	239.36	(00.00 00.00 00.00)	00.00	3.91	2.45
cam02	685.48	336.44	239.89	(246.38 17.78 21.42)	362.74	2.93	1.76
cam03	709.12	286.17	243.23	(135.34 -12.98 33.94)	412.53	3.81	3.01
Pedestrian							
GT	782.22	320.00	240.00	–	–	–	–
cam01	781.88	374.98	243.36	(00.00 00.00 00.00)	0000.00	3.83	3.05
cam02	783.51	303.31	241.66	(266.55 -36.40 -18.96)	4221.49	3.81	3.26

Table 4.3: Intrinsics and extrinsics of three real cameras compared to the ground truth, given by the camera's data sheets. Additionally, mean and standard deviation of the reprojection error are shown.

determination of top and bottom locations extracted from real world images is noisy, it can be seen that the mean reprojection error is relatively high in both cases (e.g. 3.91 pixels for camera 1 using the model figure and 3.83 pixels for camera 1 using the pedestrian). To show the accuracy of the proposed calibration method on real data, the height of the model figure and the pedestrian is calculated. All ten recovered top and bottom positions are used for calculating the mean height of all ten points which is $\mu = 56.00$ millimeters ($\sigma = 1.21$ millimeters) for the model figure and $\mu = 195.00$ centimeters ($\sigma = 4.17$ centimeters) for the pedestrian.

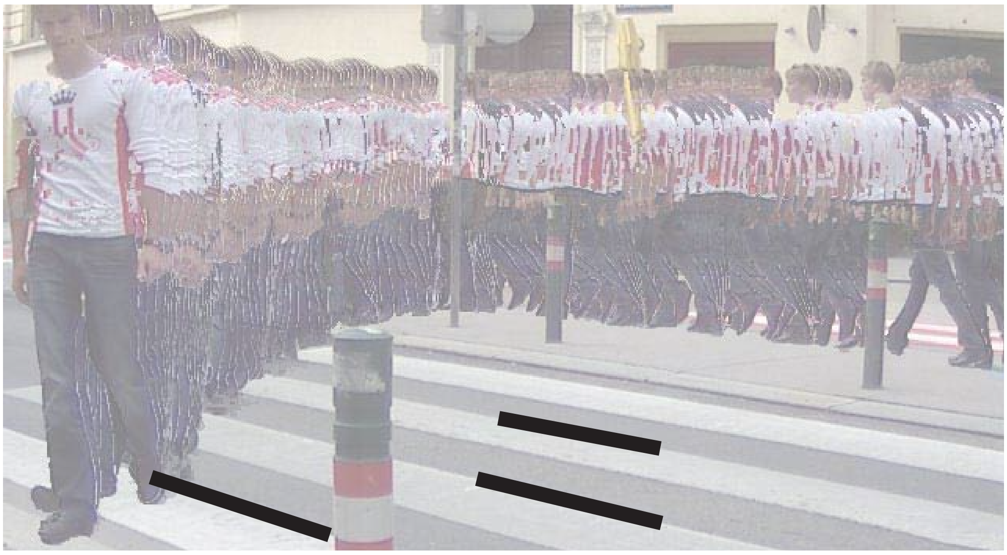


Figure 4.4: Camera calibration by analyzing a zebra-crossing and a pedestrian. A vertical vanishing point is extracted from the walking person, two horizontal vanishing points are determined by analyzing the zebra-crossing.

4.2 Single Camera Auto-Calibration Using Pedestrians and Zebra-Crossings

This section presents a calibration method which is able to gather intrinsic and extrinsic camera parameters from 2D images using only the a-priori assumption that one or more pedestrians are walking on or near a zebra-crossing. There are no restrictions to the humans walking in a certain manner, direction, or certain velocity. The approach is also able to classify between pedestrians and cars and is therefore of great use for surveillance applications. Figure 4.4 shows a sequence of one pedestrian walking on a zebra-crossing. The proposed method exploits the approach of [Se, 2000] and determines both a horizontal vanishing point and a vanishing line from the edges of a zebra-crossing. By having a person walking through the working volume, all three vanishing points get extracted from a scene and the camera' intrinsic and extrinsic parameters can be calculated. Exploiting redundant information from pedestrians and zebra-crossings eliminates two main problems in previously presented auto-calibration approaches, namely

1. a restriction in terms of constraints which cannot be fulfilled precisely in surveillance scenarios (e.g. cars must be driven in a straight manner, as presented in [Zhang et al., 2008b]) and
2. the need of a-priori information, e.g. camera's or pedestrian's height, like in [Lv et al., 2006] or [Zhang et al., 2008b].

Vanishing points are defined as the intersection of parallel lines, projected onto the image plane. When the parallel lines are also equally spaced, the vanishing line can be extracted. These two concepts are useful for auto-calibrating a camera. The edges of a zebra-crossing, which consists of alternating black and white patterns, are equally spaced and parallel in 3D space. Therefore, the first horizontal vanishing point $\mathbf{v}_1$ and the vanishing line $\mathbf{l}$ are extracted by analyzing a zebra-crossing. Since the main axes of all instances of a pedestrian's trunk which are observed over a sequence of frames and perpendicular to the ground plane, are parallel to each other, a vertical vanishing point $\mathbf{v}_2$ can be determined. The second horizontal vanishing point $\mathbf{v}_2$ can then be estimated by using the triangle spanned by three vanishing points. The image of the absolute conic can be formed using all three vanishing points. By applying the Cholesky decomposition [Press et al., 1992] the intrinsic parameters can be extracted from the image of the absolute conic. Extrinsics are estimated by determining the camera's height from the width of the zebra-crossing's bright area.

4.2.1 Vanishing Point Estimation

As can be seen in the previous section, cameras can be calibrated by using multiple instances of a walking pedestrian. One of the problems which arise is the sensitivity against poor segmentation of the pedestrian which leads to outliers in the calculation of vanishing points. Since the camera parameters are directly related to the vanishing points, the accuracy of the calibration result decreases with noisy measurements of the vanishing points. Most of the surveillance scenarios (e.g. in garages, on parking lots, or general urban traffic scenes) basically provide two types of moving objects, namely pedestrians and vehicles. When pedestrians are observed, they may cross the street on or near a zebra-crossing. In this section, a camera calibration method is presented which exploits this knowledge and uses both pedestrians and zebra-crossings in order to extract the camera's intrinsic and extrinsic parameters from a traffic surveillance scenario. When a camera is calibrated using static objects, it is known that the flexibility is limited but the accuracy is high. On the other hand, when a camera is calibrated using dynamic or moving object, the method is flexible and more general but the calibration result is noisier. It is shown in the experiments in Section 4.2.3 that the calibration results can be improved by combining static (pedestrians) and dynamic (zebra-crossing) objects and by exploiting the advantages of both these objects for calibration. Two constraints need to be fulfilled in order to extract vanishing points from surveillance scenarios using the described algorithm.

- Pedestrians are walking on an even ground plane with their body perpendicular to the plane.
- Zebra-crossings having equally spaced and parallel black (or darker) and white (or brighter) patterns are present somewhere in the observed scene.

As the mentioned properties arise in traffic surveillance scenarios, they can easily be fulfilled and used for gathering three vanishing points, perpendicular to each other and denoted by $\mathbf{v}_1 = (x_1, y_1)$, $\mathbf{v}_2 = (x_2, y_2)$, $\mathbf{v}_3 = (x_3, y_3)$.

4.2.1.1 Moving Object Detection and Classification

As static surveillance cameras are used for the experiments carried out in Section 4.2.3, simple background subtraction is taken for extracting the moving blobs within a scene. The blob's shadow is removed by using the normalized Red Green Blue (RGB) color model. Since blobs can be separated (e.g. between top and bottom of pedestrians), a morphological closing is performed for reunion. The center of mass is calculated for each line and for each row of the moving blob. Afterwards, two lines called lines of mass are fitted through all centers in horizontal and vertical direction. Figure 4.5 shows the vertical $(\mathbf{m})$ and horizontal $(\mathbf{m}')$ lines of mass of a walking person and a car. When observing a pedestrian, the difference between horizontal and vertical line of mass is smaller than the difference between the two lines of an observed car. This difference occurs due to a more cubic and regular shape of a car compared to the pedestrian. Therefore, a threshold to classify between a car and a pedestrian is introduced. In practice, this threshold is set to $45°$. As can be seen, the segmentation must not be precise (e.g. holes in the body of the pedestrian, clipped head) to get a correct classification result and the two lines of mass.

4.2.1.2 Extraction of Vertical Vanishing Point

When using a sequence of two or more instances of one or more walking pedestrians, indicated by $n = 1, 2 \ldots N$, multiple vertical lines of mass can be recovered, which are denoted by $\mathbf{m}_n$. Next to calculating these lines for each blob classified as a pedestrian, the vertical vanishing point $\mathbf{v}_3$ can be determined. When using synthetic data, all intersections of possible pairs of $\mathbf{m}_n$ must converge to one point $\mathbf{v}_3$. Since the initialization of $\mathbf{m}_n$ may be noisy when using real data, an approximation needs to be performed. Let $\mathbf{h} = (x_h, y_h)$ and $\mathbf{b} = (x_b, y_b)$ be the foot and head point of the first instance of the pedestrian, respectively. By exploiting cross-ratio, the coordinates of the vertical vanishing point $\mathbf{v}_3$ may be written as

$$\frac{y_3 - y_h}{x_3 - x_h} = \frac{y_b - y_h}{x_b - x_h}, \tag{4.5}$$

or

$$x_3(y_h - y_b) + y_3(x_b - x_h) = (x_b y_h - x_h y_b), \tag{4.6}$$

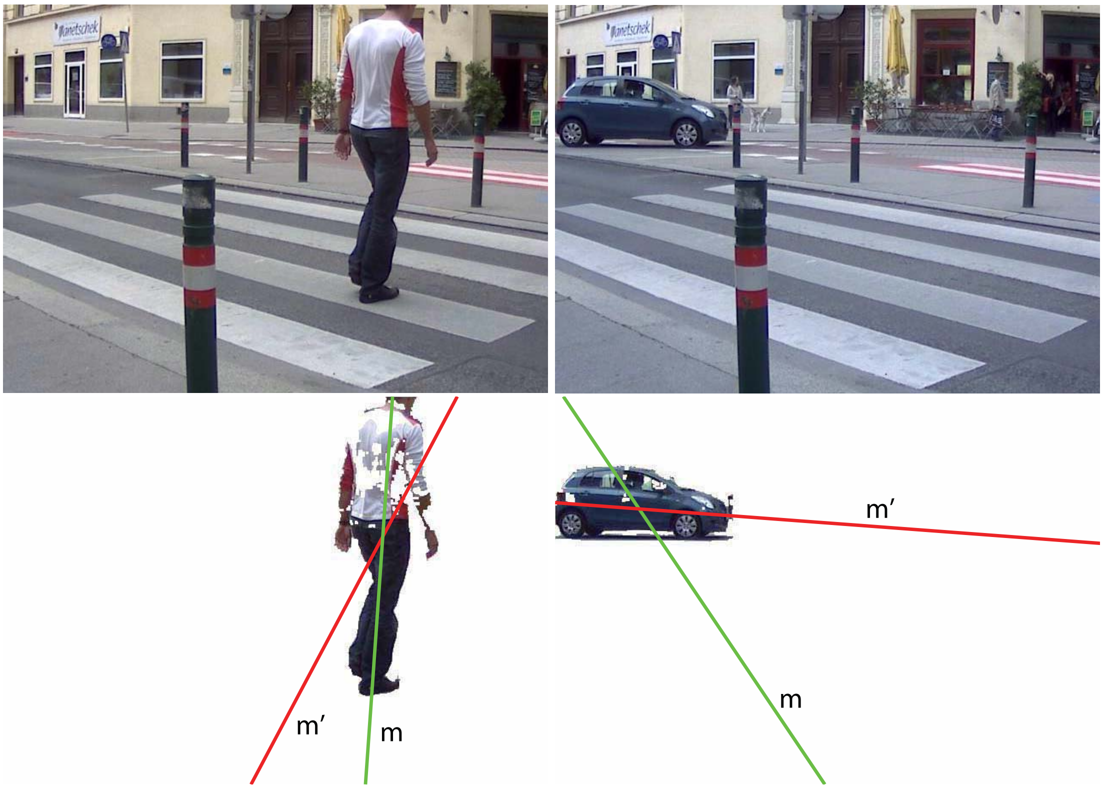
as described in [Kusakunniran et al., 2009]. When two or more instances of a pedestrian are given, an equation system $\mathbf{A} \cdot \mathbf{v}_3 = \mathbf{r}$, where $\mathbf{r}$ is the result vector, can be formed to solve the equation. A least squares solution is gathered by solving

$$\mathbf{v}_3 = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{r}. \tag{4.7}$$

Two different pedestrians or two different positions of the same pedestrian would be enough to determine a vertical vanishing point but due to more robust calculations, more than two positions should be used for camera calibration.

4.2.1.3 Extraction of Horizontal Vanishing Points

After the determination of the vertical vanishing point $\mathbf{v}_3$, the first horizontal vanishing point $\mathbf{v}_1$ and the vanishing line $\mathbf{l}$ need to be estimated. A zebra-crossing, which consists of black (or

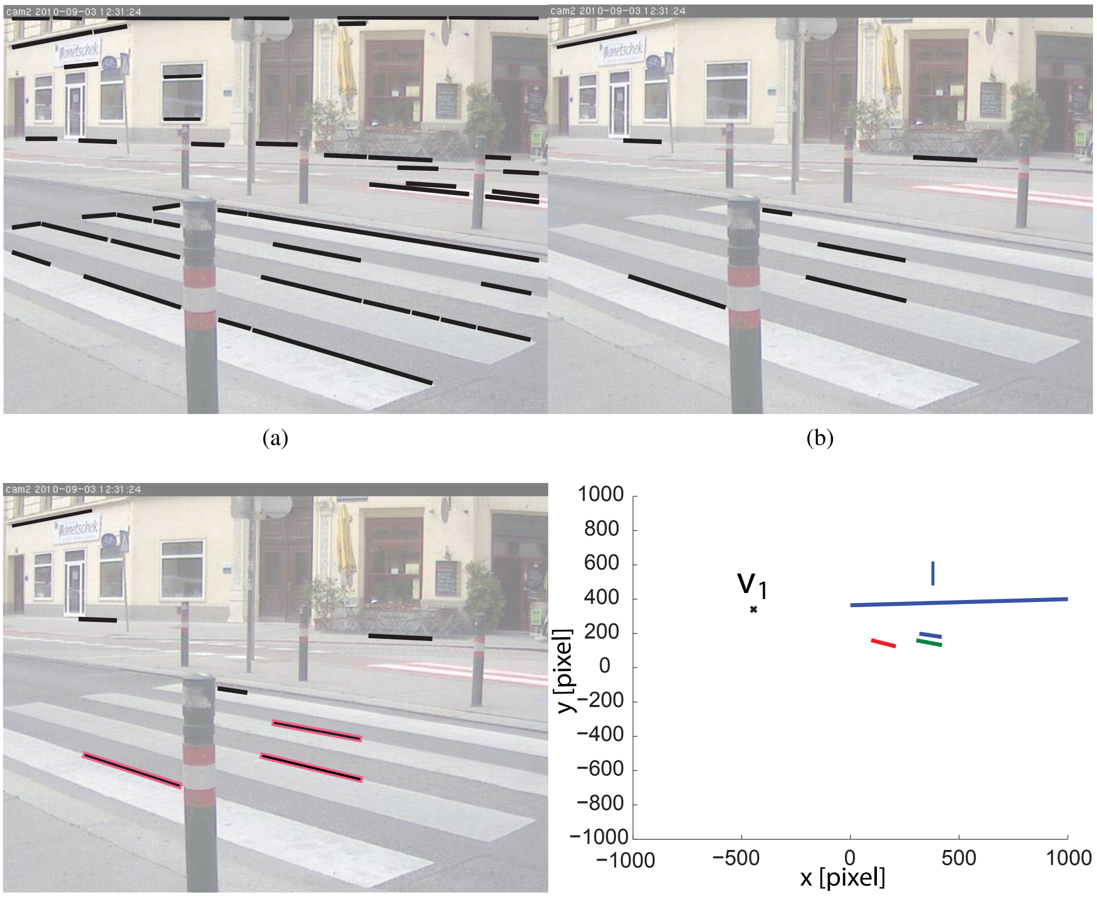


(a) (b)

Figure 4.5: Input image and classification of (a) a pedestrian and (b) a car. The difference between vertical and horizontal line of mass is $24.0°$ for the pedestrian and $51.7°$ for the vehicle.

darker) and white (or brighter) equally spaced patterns, is therefore analyzed. For the calculation of the vanishing line, at least three equally spaced parallel lines need to be extracted. In a first step, a Canny Edge Detector is exploited to determine the edges in the zebra-crossing image. Only positive gradients are taken for further calculations. When positive and negative gradients would be used, this would mean that the width of black and white patterns must be the same to have all parallel edges equally spaced. When only using positive gradients, all black or all white patterns must have the same width but not both colors.

In a second step, redundant edge segments are eliminated (e.g. one edge segment of a zebra-crossing can be separated in two line segments when a pedestrian is occluding parts of the edge). The shorter segment is removed and the longer one is kept for further calculations. The next step of the pipeline searches for all edge segments which support a common vanishing point $\mathbf{v}_1$. By exploiting RANSAC [Fischler and Bolles, 1981], the intersection of a random pair of edge segments is taken as an initial guess for $\mathbf{v}_1$. Only those segments where $\mathbf{v}_1$ is located within a certain distance to them are taken as inliers. In practice, this threshold distance is three pixels. The result having the most inliers is taken as the best one. The final vanishing point is



(a) (b)

(c) (d)

Figure 4.6: (a) All detected edge segments, (b) segments supporting $\mathbf{v}_1$ after removal of redundant data, (c) triplet of lines supporting $\mathbf{l}$, (d) the determined values for $\mathbf{l}$ and $\mathbf{v}_1$ shown in pixels on the image plane.

then estimated by only using inlier edges. For each inlier edge segment two points on the line are taken to get a least square estimation for $\mathbf{v}_1$ (see description for the estimation of $\mathbf{v}_3$ in Section 4.2.1.2).

It may occur that also segments, which are not part of the zebra-crossing, support a vanishing point but those lines get eliminated in the next step. The vanishing line $\mathbf{l}$ can only be calculated by a closed-form solution when using exactly three equally spaced and parallel edge segments. Having three different line segments $\mathbf{s}_i$, $\mathbf{s}_j$ and $\mathbf{s}_k$, which may not be consecutive, this closed form solution is given by [Se, 2000]

$$\mathbf{l} \propto [(\mathbf{s}_i \times \mathbf{s}_k) \cdot (\mathbf{s}_j \times \mathbf{s}_k)]\mathbf{s}_j + (k - i)[(\mathbf{s}_i \times \mathbf{s}_j) \cdot (\mathbf{s}_k \times \mathbf{s}_j)]\mathbf{s}_k. \tag{4.8}$$

RANSAC [Fischler and Bolles, 1981] is used to estimate the vanishing line. Therefore a triplet of

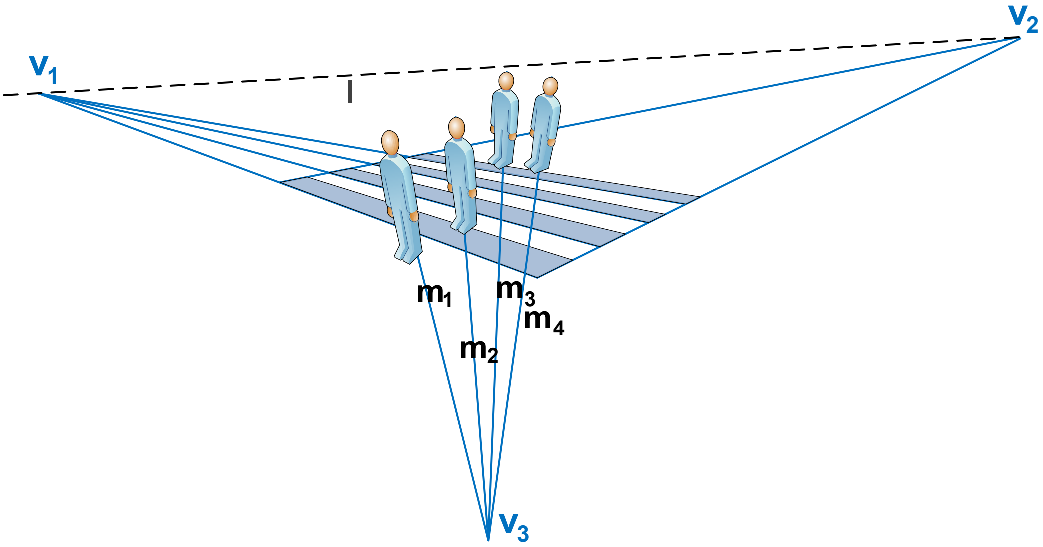


Figure 4.7: Interrelationship between $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, $\mathbf{l}$.

random edge segments is taken to calculate an initial guess for $\mathbf{l}$ using Equation 4.8. A verifying check if another edge segment $\mathbf{s}_l$ supports the current estimation of $\mathbf{l}$ can be established by calculating

$$\mathbf{E} = \mathbf{s}_i \times \mathbf{s}_l + (l - i)\mu\mathbf{l} \times \mathbf{s}_l \ ,$$

where

$$\mu = -\frac{\mathbf{s}_i \times \mathbf{s}_j}{\mathbf{l} \times \mathbf{s}_j} = -\frac{\mathbf{s}_i \times \mathbf{s}_k}{2 \cdot (\mathbf{l} \times \mathbf{s}_k)}. \tag{4.9}$$

In practice, $\mathbf{s}_l$ supports $\mathbf{l}$ when $|\mathbf{E}|$ is smaller than a threshold, chosen to be 0.0008. Figure 4.6 shows all detected edge segments, the segments supporting one common vanishing point, the three edge segments used for calculating the vanishing line, the estimated $\mathbf{l}$ and $\mathbf{v}_1$.

The third vanishing point, $\mathbf{v}_2$, can be calculated after the estimation of $\mathbf{v}_1$, $\mathbf{v}_3$ and $\mathbf{l}$. Figure 4.7 shows different instances of a pedestrian walking on a zebra crossing and the interrelationship of $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ and $\mathbf{l}$. If $\mathbf{l}$ has a gradient equal to zero, it is called the horizontal line. This gradient angle is known as the yaw angle γ of the camera. Before calculating $\mathbf{v}_2$, the vanishing line needs to be aligned with the horizontal line. This is established by rotating $\mathbf{l}$ by the negative gradient of the vanishing line. By aligning $\mathbf{l}$, the whole scene, including all points needed for further calculations, must also be rotated ($\mathbf{v}_1 = \mathbf{v}'_1, \mathbf{v}_3 = \mathbf{v}'_3$). The principal point $\mathbf{c}$ can safely be assumed to be the image center and $\mathbf{c}$ is rotated to obtain $\mathbf{c}'$. As $\mathbf{c}'$ is the orthocenter of the triangle spanned by the three vanishing points, $\mathbf{v}'_2$ can be calculated geometrically, as described in Section 4.1 and in [Lv et al., 2006]. To obtain $\mathbf{v}_2$, the scene is rotated back around the image origin by the gradient of $\mathbf{l}$.

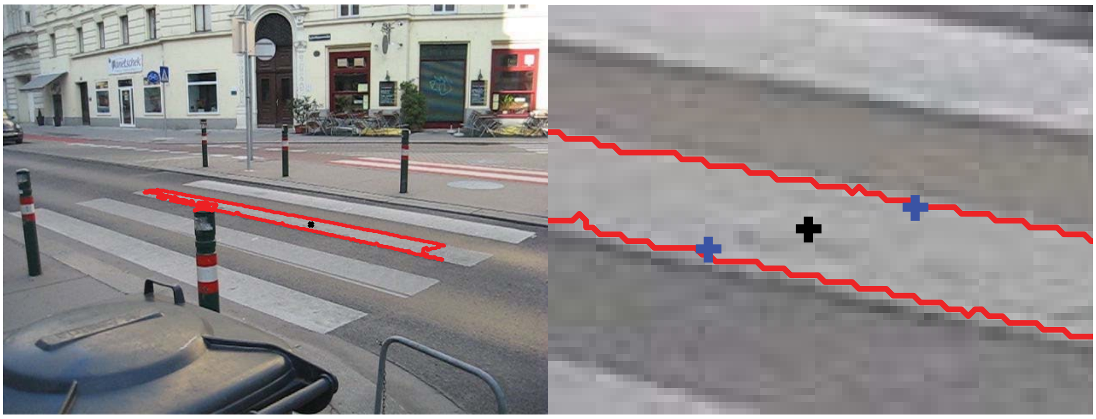

(a) (b)

Figure 4.8: (a) One extracted lighter pattern of a zebra-crossing and its centroid. (b) The two determined reference points, where the distance between them must be 50 centimeters.

4.2.2 Camera Parameter Estimation

By applying the Cholesky decomposition [Press et al., 1992], the intrinsic parameters $\mathtt{K}$, consisting of focal length f and principal point $\mathbf{c} = (u_0, v_0)$ are extracted from the IAC. Next to calculating the intrinsic parameters, the extrinsics, defined by a rotation matrix $\mathtt{R}$ and a translation vector $\mathbf{t}$, need to be determined. Under the assumption of zero skew and an aspect ratio of one, $\mathtt{KK}^T = \omega^{-1}$ can be rewritten as

$$\begin{bmatrix} \lambda_1 x_1 & \lambda_2 x_2 & \lambda_3 x_3 \\ \lambda_1 y_1 & \lambda_2 y_2 & \lambda_3 y_3 \\ \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix} = \mathtt{KR} \tag{4.10}$$

where λ_i are scaling factors. By rearranging Equation 4.10, these scaling factors are determined by solving the equation system

$$\begin{aligned} x_1^2\lambda_1^2 + x_2^2\lambda_2^2 + x_3^2\lambda_3^2 &= f^2u_0^2 \\ y_1^2\lambda_1^2 + y_2^2\lambda_2^2 + y_3^2\lambda_3^2 &= f^2v_0^2 \\ \lambda_1^2 + \lambda_2^2 + \lambda_3^2 &= 1 \end{aligned} \tag{4.11}$$

Three angles, namely α for roll, β for pitch and γ for yaw are then extracted from $\mathtt{R}$. The position and scaling within a common world coordinate system is described by a translation $\mathbf{t} = \mathtt{R}(0 \quad 0 \quad H)^T$, where H is the camera's height. To get the height in metric 3D space, a scaling needs to be determined. Brighter areas of a zebra-crossing do not have to be standardized but in practice, most of them are at least two meters long and exactly 50 centimeters wide. This knowledge is exploited and the brighter areas are extracted, where at least two border pixels of the area are located within a distance of two pixels to the extracted zebra-crossing edges. These edges are used to gather the vanishing line $\mathbf{l}$. The centroid of the areas is also extracted.

Next, for each extracted pattern two intersections of the area's border lines and the line, which has the vanishing point's orientation and goes through the centroid, has to be calculated. These intersection pairs are taken as reference points where the real world distance of each pair must be 50 centimeters. Figure 4.8 shows the determination of both reference points for one brighter pattern of the zebra-crossing. After the extraction of a lighter pattern, the two reference points, where the distance between them must be 50 centimeters, are determined. Since both points are located on the same plane, the 3D coordinates can be recovered from a single camera. As the camera matrix `K` and the rotation matrix `R` are known, the camera's height is increased and the pair of points is projected to 3D space until the estimated distance between the two points meets the reference distance of 50 centimeters. To gain a higher accuracy, the mean distance of intersection pairs of all bright patterns is taken.

4.2.3 Experiments

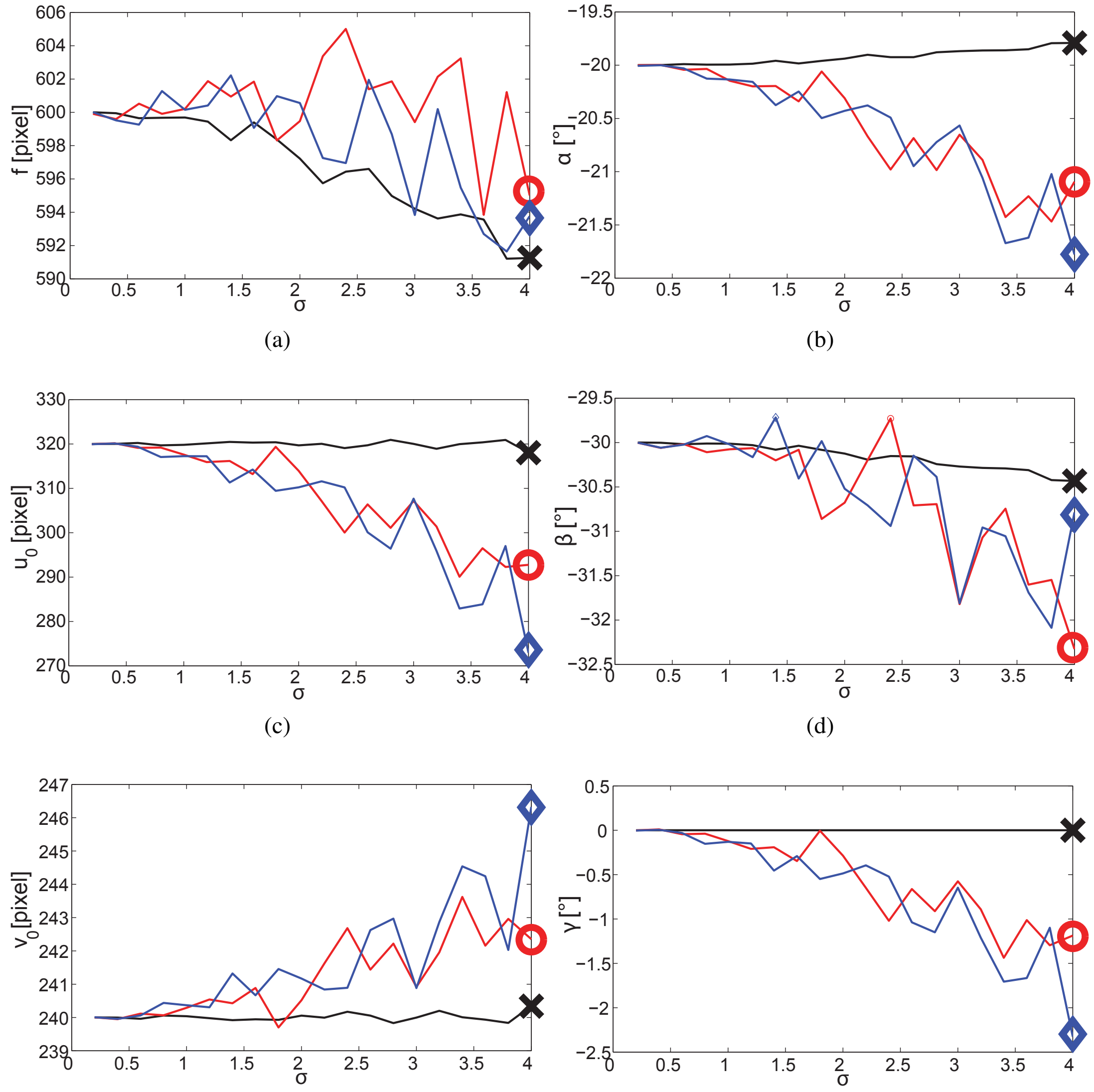
To show the practicability and the accuracy of the proposed approach, this section provides experiments using synthetic and real data in various scenarios.

4.2.3.1 Synthetic Data

As in practice the calibration parameters are affected by noisy input data (e.g. poor segmentation, distortions, changing lightning conditions), this effect can be simulated by using synthetic, noisy input data. A synthetic camera, having a focal length of 600 pixels, providing three angles, $\alpha = -20°$, $\beta = -30°$ and $\gamma = 0°$ is positioned at a height of $H = 20$ units from the ground plane. The image provides the dimensions 640×480 so the principal point is located at $u_0 = 320$, $v_0 = 240$. Two or more instances are needed in order to assure the least squared optimization to work properly and exactly three equally spaced lines are needed to calculate the vanishing line. In this test set, five pedestrians and three parallel and equally spaced lines, representing three equally spaced edges of a zebra-crossing, are used. All instances of the pedestrian and the edges of the zebra-crossing are positioned randomly within the scene, where persons are sitting on the ground plane. The distance between two edges is 15 units.

By introducing Gaussian noise, described by its parameter σ, the pixel locations are randomly moved in both vertical and horizontal direction. As it would be interesting to answer the question whether the edges of the zebra-crossing or the foot/head locations of the pedestrians are less robust against outliers, those two classes of points are distorted separately and in a combined fashion. Figure 4.9 shows the results for f, u_0, v_0, α, β and γ from top to bottom and left to right. The metrics are represented by the vertical axis. The horizontal axis represents the standard deviation σ which goes from 0-4 pixels by a stepsize of 0.2. The calibration is performed 1000 times at each noise level and the mean results are taken for comparison. The distorted foot/head points, distorted zebra-crossing edges and both locations distorted are represented by $\times$, $\circ$ and $\diamond$, respectively.

As can be seen, the output is not sensitive to pedestrian locations. At a maximum noise level of $\sigma = 4$, the relative errors of f, u_0, v_0, α, β and γ compared to the ground truth are 1.37%, 0.04%, 0.01%, $0.25°$, $0.51°$ and $0.00°$, respectively. In practice, the sensitivity regarding the zebra-crossing depends on the distance between the black and white patterns in the image space.



(a) (b)

(c) (d)

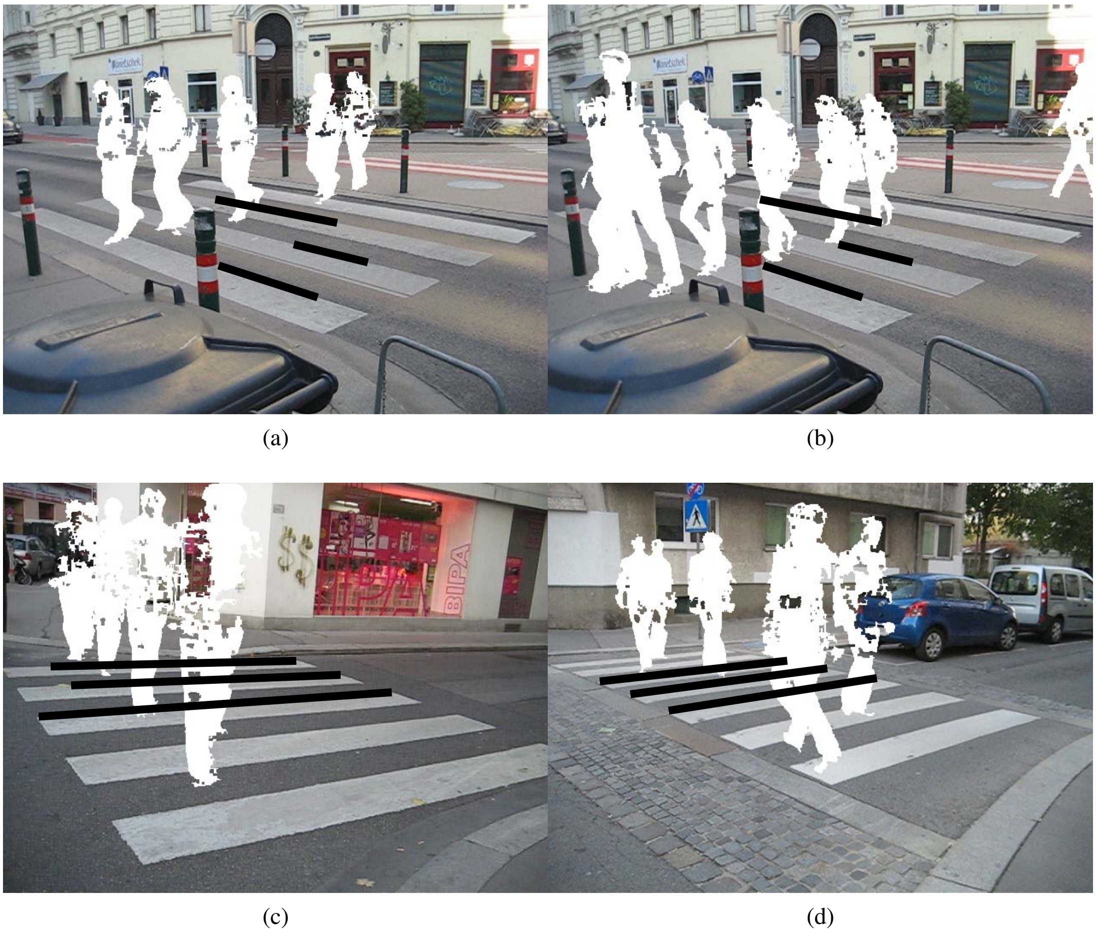
(e) (f)

Figure 4.9: Synthetic data. From top to bottom and left to right: The results for f, u_0, v_0, α, β and γ by using distorted foot/head points ($\times$), distorted zebra edges ($\circ$) and both pixel values distorted ($\diamond$). The pixels are shifted by σ in both vertical and horizontal direction.

The wider the distance is, the more robust the calibration setup is. Therefore, the calibration procedure is more robust when the camera's height is greater than the distance to the first pattern of the zebra-crossing. When both top/bottom locations of the pedestrians and the edges of a zebra-crossing are distorted, maximum relative deviations to the ground truth data of 1.10%, 14.20%, 2.82%, $1.00°$, $0.23°$ and $1.68°$ for f, u_0, v_0, α, β and γ, are obtained respectively. As expected, the curves of all metrics show that the accuracy decreases by increasing the noise level. As the camera's height has a maximum relative deviation of 0.02% when both zebra edges and pedestrian locations are distorted, H and $\mathbf{t}$ are not taken into account for evaluation.

4.2.3.2 Real Traffic Scene

To show the practicability of the proposed approach, the algorithm is also tested on real world data. There is no dataset publicly available which provides both a scene showing pedestrians in combination with zebra-crossings and a correct calibration. Therefore, a Canon Digital IXUS 60 camera, having a focal length of 665 pixels, captures two image sequences with a resolution of 640x480 pixels. A pedestrian is walking near or on a zebra-crossing. Four sequences, labeled PED01-PED04 and shown in Figure 4.10, are taken for evaluation purposes.



(a) (b)

(c) (d)

Figure 4.10: Real data. All instances of pedestrians used for (a) PED01, (b) PED02, (c) PED03 and (d) PED04, shown in one image. For PED01 and PED02, the same zebra-crossing is used, PED03 and PED04 offer different ones.

	f (pixels)	u_0 (pixels)	v_0 (pixels)	$(\alpha, \beta, \gamma)^T$ (degrees)	$\mathbf{t}$ (cm)	H (cm)
CCTfM	662.1	318.6	239.8	–	–	–
[Lv et al., 2006]	678.8	193.8	220.0	–	–	–
PED01	665.3	317.6	240.5	-45.4 -10.8 -02.8	-07.9 159.3 -38.2	168.0
PED02	625.4	327.7	238.5	-43.3 -11.6 -02.8	-07.9 158.6 -41.1	176.0
PED03	656.7	314.8	239.7	33.8 -04.5 06.4	17.5 156.4 -38.6	143.0
PED04	675.7	356.6	245.9	43.8 -09.6 01.3	03.8 161.9 -31.4	146.0

Table 4.4: Comparison of calculated intrinsic/extrinsic parameters using PED01-PED04 to the results obtained by using the CCTfM and the method of [Lv et al., 2006] (using sequence PED01). The ground truth focal length is 665 pixels, the measured camera heights are 164/166/152/153 centimeters for PED01-PED04, respectively.

The three lines represent the pencil of zebra-crossing edges used for calculating the vanishing line. PED01 and PED02 show the same zebra-crossing, PED03 and PED04 offer a different one. Due to privacy issues, pedestrians are shown as white silhouettes. The camera is located at a measured height of $H = 164/166/152/153$ centimeters (cm) above the ground plane for PED01-PED04, respectively.

To evaluate the accuracy of the intrinsic parameters, the results are compared to the results obtained by using the CCTfM [1] and an implementation based on [Lv et al., 2006], where PED01 was used as input. Table 4.4 presents the intrinsic parameters for the CCTfM, [Lv et al., 2006] as well as estimations of intrinsics, extrinsics and the camera height for PED01-PED04 using the proposed approach. As can be seen, all obtained results are close to the ground truth data but the results for the principal point using the method described in this work are closer than the results obtained by [Lv et al., 2006]. This occurs due to segmentation uncertainties which have more effect on the method of [Lv et al., 2006] than on the proposed one. As can be seen from the experiments' results, the focal length is more sensitive to outliers than the principal point which occurs due to poor pedestrian segmentation (e.g. clipped head or legs) and the resulting calculation errors for each $\mathbf{v}_n$. As can be seen in Figure 4.10, PED03 provides the greatest yaw angle of $\gamma = 6°$ and a smaller tilt angle than PED04.

---

[1] `http://www.vision.caltech.edu/bouguetj/calib_doc/`, last retrieved on 21.03.2013.

4.2.3.3 Real Traffic Scene Measurements

In a second step, real world measurements are compared to the results obtained by the described method to evaluate the estimated extrinsic parameters. The calibration parameters of PED01 are used for this purpose and two types of measurements are determined. This experiments should also show how calibration parameters influence the accuracy of the measurements in 3D space.

First, vertical ones (labeled h_i, $i = 1 \dots 7$), where bottom points are located on the ground plane, having the 3D coordinates $(x_i, y_i, 0)$ and top points located at (x_i, y_i, z_i) are measured. Second, horizontal distances (labeled d_i, $i = 1 \dots 13$) are calculated, where both start and end points are positioned on the ground plane and therefore share the same z-coordinate. With this restrictions it is possible to recover the 3D point from a single projection matrix, as described in [Wang et al., 2005]. The uncertainty analysis takes into account image coordinates, distorted by a Gaussian noise of $\sigma = 5$ in both vertical and horizontal direction. All metrics are presented in centimeters. Figure 4.11 illustrates the measurements projected onto a real world image plane.

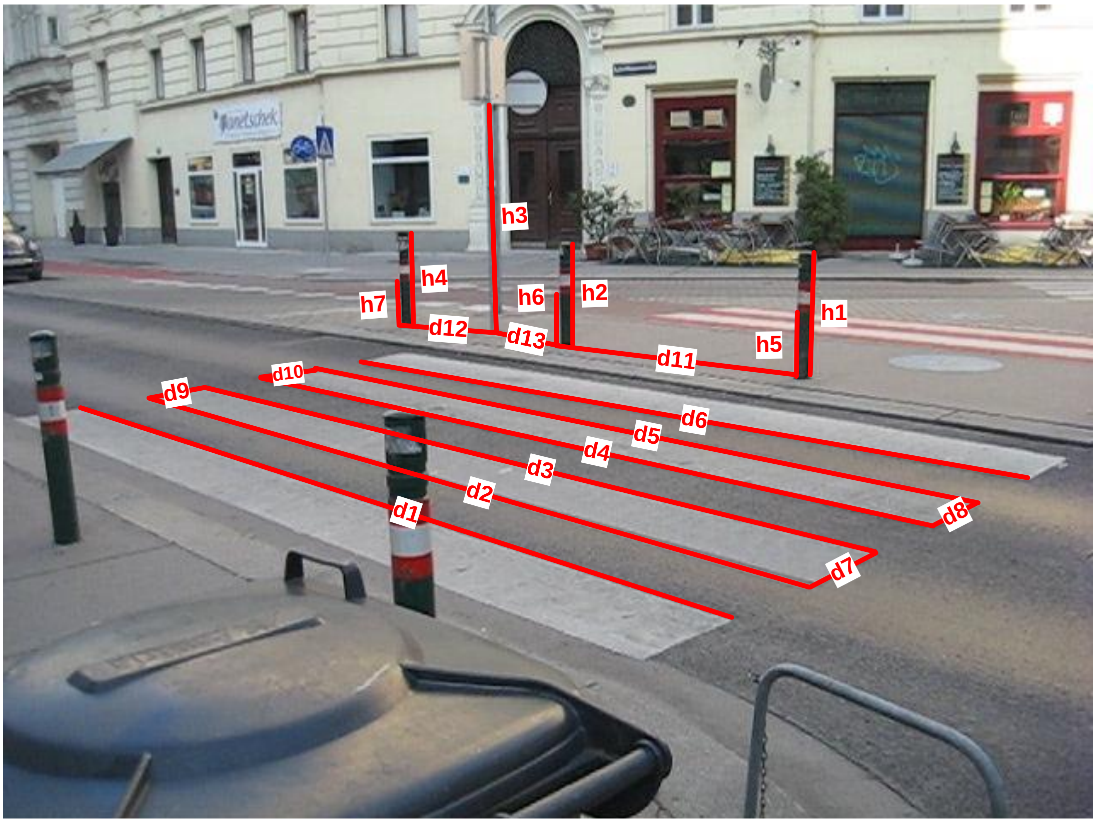


Figure 4.11: Vertical (d_i) and horizontal (h_i) measurements (in centimeters) compared to estimated values of the proposed method.

As can be seen, the mean error for all distances between measured and calculated values is

Distance	d_1	d_2	d_3	d_4
Measured	600	600	600	600
Calculated	558.56±12.1	543.14±15.5	554.30±12.0	540.65±12.6
Distance	d_5	d_6	d_7	d_8
Measured	600	600	50	50
Calculated	535.58±13.6	543.61±13.9	49.81±4.1	44.35±6.3
Distance	d_9	d_{10}	d_{11}	d_{12}
Measured	50	50	245	123
Calculated	48.67±4.1	50.02±6.4	240.12±16.3	124.71±10.8
Distance	d_{13}	h_1	h_2	h_3
Measured	118	90	90	200
Calculated	111.18±10.0	84.06 ±2.0	85.47±3.6	198.23±3.5
Distance	h_4	h_5	h_6	h_7
Measured	90	50	50	50
Calculated	89.57±3.34	46.05±2.79	47.36±3.66	47.64±3.66

Table 4.5: Measurements h_1-h_7 and $d_1 - d_{13}$, shown on the background image of PED01.

5.4%, the standard deviation is 3.7%. In this experiment, the distance between the 3D points and the camera center range from 500.5 and 1438.3 centimeters. Although Zhang et al. only used distances shorter than 2.3 meters and located near the camera, the proposed approach reaches slightly better results compared to the measurements introduced in [Zhang et al., 2008b] (mean error 6.3%, standard deviation 4.6%). Using longer distances results in higher relative error rates because due to measurement uncertainties far distances have a higher error weighting than near ones. This phenomenon is also visible in Table 4.5, where longer distances or distances far away from the camera have a higher uncertainty than shorter or near measurements.

4.3 Summary

This chapter describes two camera auto-calibration methods which try to exploit redundant information in order to increase the parameter estimation accuracy compared to state-of-the-art approaches.

First, an approach, which obtains both intrinsic and extrinsic parameters for a whole camera network from multiple observations of a pedestrian over time, is described. In surveillance scenarios, pedestrians are predestined to be used as input for auto-calibrating a traffic surveillance camera. The user needs to predefine the height of the walking person to determine the scaling of all cameras' absolute translations. As shown in the experiments, the accuracy of the method only depends on the noise of top and bottom locations.

Second, a novel self-calibration method for fixed surveillance cameras based on zebra-crossings and pedestrians is outlined. These two objects are included in traffic surveillance scenarios. An important advantage of the algorithm is that it needs no user input to completely recover both intrinsic and extrinsic parameters. This is necessary for camera calibration and re-calibration where physical access is impossible. One vertical vanishing point is extracted from the pedestrians' lines of mass and in combination with the horizontal vanishing point and the vanishing line, obtained from three equally spaced zebra-crossing edges, the camera can be calibrated. The suggested system requires zebra crossings to work which limits the number of scenarios. Nevertheless, it is useful for traffic surveillance applications, where 3D information is needed to improve e.g. pedestrian detection or tracking, pose estimation or object classification. The proposed system is both robust and accurate which is demonstrated by experimenting on synthetic and real world data and by comparing it to state-of-the-art calibration methods.

As stated in [Sun and Cooperstock, 2005], a mean value of 0.88 pixels on the distorted image plane results in a mean distance between the 3D test points and the optical ray generated from their projections of 3.58 millimeters. The 3D test points used in their experiments have a rather small distance ranging from 25 to 55 centimeters measured from the camera center and the cameras are calibrated manually. This shows that obtaining accurate camera parameters is important for real 3D world measurements. Calibrating a camera manually is a tedious work and when a network consists of hundreds of cameras, where each of them also needs to be re-calibrated whenever it is moved, this cannot be tackled manually anymore. The proposed method, which is exploiting a zebra-crossing and pedestrians, achieves a mean error of 5.4% between measured 3D distances (ranging from 50 to 600 centimeters) and projected ones. This means an improvement of approximately 1% compared to the mean error reported by [Zhang et al., 2008b]. The mean error of 3.58 obtained by [Sun and Cooperstock, 2005] equals to 6.5% when using the largest possible distance of 55 centimeters. As can be seen, also their mean measurement error is approximately 1% larger, although in the proposed setup the distance between the 3D points and the camera center is ranging from 5.0 to 14.4 meters, which is farther away from the camera center compared to the settings used by [Sun and Cooperstock, 2005].

CHAPTER 5

3D Reconstruction from Semantic and Geometric Information

This chapter outlines how 2D images obtained at man-made indoor scenes can be split into *ground plane*, *ceiling* and *vertical* regions. It is also shown how this information in combination with point based local features can be used to obtain a dense 3D model of the scene. As is described in the following, combining this semantic information with the outcome of feature-based 3D reconstruction pipelines overcomes the problem of finding feature matches on textureless surfaces (e.g. walls, non-Lambertian planar patches,...).

First, a novel method for 3D surface normal layout estimation is outlined in Section 5.1. The image is first split into superpixels using multiple segmentation methods. Superpixels give the opportunity to obtain a semantically richer representation of a scene then single point-based features do. Automatically analyzing the content of multiple connected pixels can be seen as a similar procedure compared to a human person capturing semantic and geometric context within a scene. Each segment is then assumed to be part of either the *ground plane*, the *ceiling*, or a *vertical* wall. Segmentation uncertainties are compensated by using multiple superpixel segmentation methods. After having multiple hypothesis for each pixel's surface orientation, the global best label is obtained by using an MRF. The proposed algorithm delivers more accurate results compared to state-of-the-art 3D surface normal labeling methods. Major parts of this method are published in [Hödlmoser and Mičušík, 2013].

Second, a novel algorithm for 3D reconstruction of man-made environments is proposed in Section 5.2. The 3D reconstruction is based on the labeling outcome of the method presented in Section 5.1. Such man-made environments suffer from textureless and non-Lambertian surfaces, where conventional, feature-based 3D reconstruction pipelines fail to obtain good feature matches. To overcome this problem, the semantic information of multiple superpixel methods is exploited once more to estimate both a 3D point and a 3D surface normal for each pixel. By applying a semantic classifier on each of the segmentation methods, a likelihood is obtained for each segment to be located on the ground plane, on the ceiling or on a vertical wall segment. The global best surface normal orientations for all pixels are then obtained by solving an MRF.

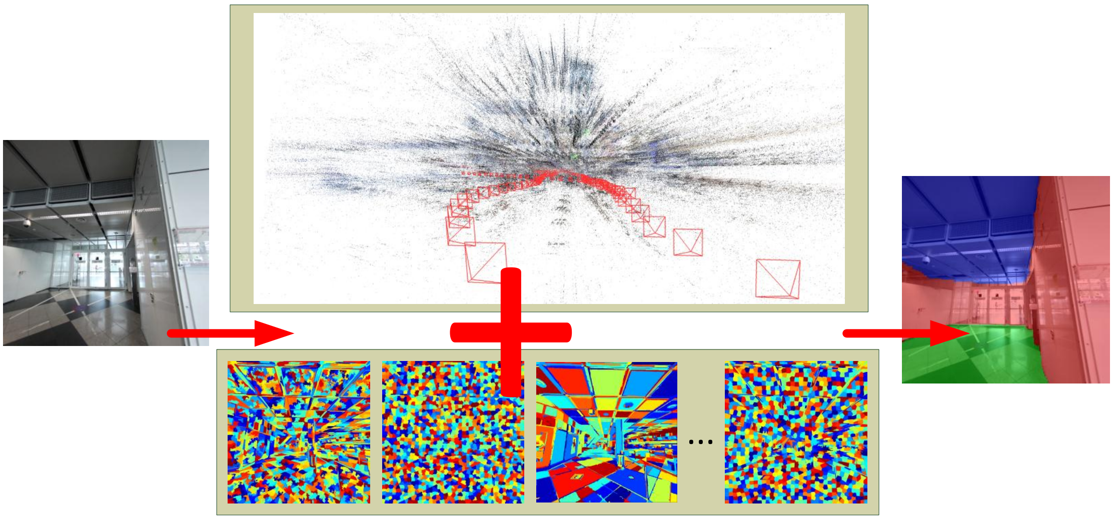


Figure 5.1: Proposed scene layout estimation by combining multiple superpixel segmentation methods and geometric reasoning.

The 3D reconstruction is then obtained by combining the outcome of the semantic surface normal orientation estimation with a sparse point cloud obtained by a conventional feature-based 3D reconstruction pipeline. It is also shown, that the proposed method outperforms state-of-the-art dense 3D reconstruction pipelines. This section contains material which is published in [Hödlmoser et al., 2013a].

5.1 Semantic 3D Surface Labeling of Complex Indoor Scenes

If humans would look on a single pixel of an image without having any other kind of information, they would not be able to reason about the context or the geometric scene. In computer vision, images may be described by point features, so called low level descriptors which do not tell anything about the semantic context. Semantically, it would be the most meaningful representation to use the occurring objects and their geometric relationships in the scene. To obtain such a representation, superpixels can be exploited. In terms of semantic richness, superpixels are located between low level features and high level objects. Superpixels treat semantically connected pixels as a single patch. As in [Wang et al., 2011, Mičušík and Kosecka, 2010], a single superpixel segmentation method is taken as a preprocessing step for further applications. As there is no superpixel generation method, which works for all kind of different scenarios and environments, the resulting segmentation may generate incorrect or missing patches. This would mean that the application on top of it would be based on an incorrect initialization and cannot recover anymore.

The approach described in this section therefore tries to overcome this problem and proposes a novel method which estimates the 3D scene layout of a scene. As a prerequisite, it is assumed that each superpixel segment in a 2D image can be represented by a planar patch in its corresponding 3D environment. The goal is then to classify each pixel in *vertical plane*, *ground plane* and *ceiling* which is equal to finding the discrete 3D surface normal of the patch, as can be seen in Figure 5.1. The workflow can be seen in Figure 5.2, where the contribution presented this section is two-fold and described in the following.

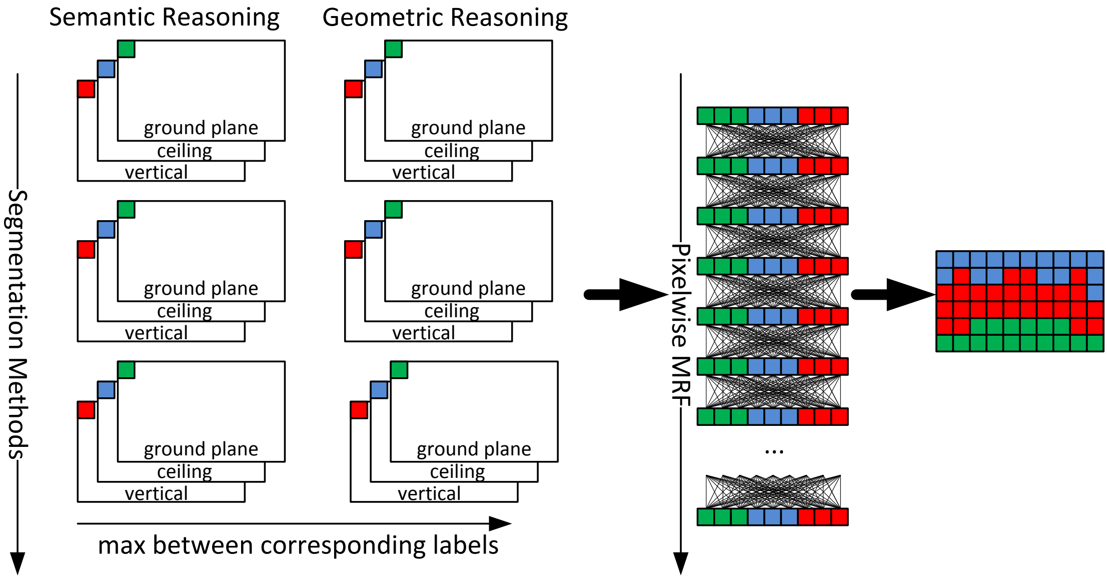


Figure 5.2: Workflow of the proposed algorithm. For each segmentation method, semantic and geometric features are calculated in order to obtain a likelihood for each segment being labeled *ground plane*, *ceiling*, or *vertical*. The final label for each pixel is then obtained by using a pixel-wise MRF.

- The presented approach combines the strengths of several superpixel segmentation methods to build a stronger classifier for pixel-wise labeling of an image in *vertical plane*, *ground plane* and *ceiling*. Different to applications which use different quantization steps for the same segmentation method [Hoiem et al., 2007, Ladicky et al., 2009], this method combines different superpixel segmentation methods to form a stronger classifier. The idea is then that pixels which are assigned the same label multiple times are more likely to have a certain label than pixels where the outcome is changing for different segmentation methods.
- Semantic reasoning is combined with geometric reasoning in order to improve the 3D surface normal labeling accuracy. For estimating the label of each superpixel, a modified version of the method described in [Hoiem et al., 2007] is used. This method is referred to as *semantic reasoning*. A sparse point cloud is generated using SfM and geometric features are calculated from it. Semantic features are combined with geometric ones and an MRF is exploited to do a pixel-wise classification.

5.1.1 Proposed Framework

The 3D surface normal of a scene is estimated by combining the advantages of multiple superpixel segmentation methods and evaluating both semantic and geometric features. In a first step, multiple images are taken to obtain a sparse point cloud and corresponding 3D camera positions using a conventional SfM pipeline [Snavely et al., 2006]. Afterwards, the whole labeling pipeline is performed on a single image. This image is then segmented into semantically meaningful parts using superpixel segmentation methods. As can be seen in Figure 5.3, there exists a variety of different segmentation methods, where each of them uses different cues to get semantically meaningful regions. These four methods are used since they provide a vast variation of the segments' shapes and sizes. They also provide different results when using changing parameters. This knowledge is exploited to obtain more accurate results by combining the strengths of different methods. By applying the approach of [Hoiem et al., 2007], the likelihood for each segment being located on the *ground plane*, *ceiling* or on a *vertical* wall segment is obtained. The segmented image is processed by [Hoiem et al., 2007] which delivers a likelihood for each segment having an orientation label $l \in \mathcal{L} = \{\text{ground plane}, \text{ceiling}, \text{vertical}\}$. The likelihood u for a single pixel y_i within this segment and at image location i, is then given by

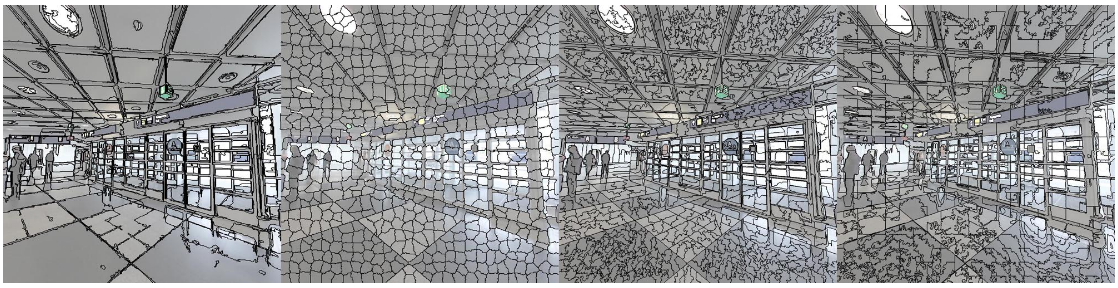


Figure 5.3: Multiple superpixel segmentation methods deliver different results in terms of shape and size of the segments. From left to right: [Felzenszwalb and Huttenlocher, 2004], [Levinshtein et al., 2009], [Liu et al., 2011], [Achanta et al., 2012].

$$u(y_i) = P(l|y_i). \tag{5.1}$$

Geometric features are then calculated for each segment using the sparse point cloud. As the goal is to classify each pixel according to the label set $\mathcal{L}$, it is assumed that each segment in the image can be represented by a planar patch. All geometric features are obtained relatively to the camera's orientation. The camera's gravity vector $\mathbf{g}$ is therefore determined from the vertical vanishing point. It is further known that the ground plane's and ceiling's surface normal are aligned with this gravity vector. Surface normals of vertical patches must be perpendicular to the gravity vector. Only those segments, where four or more reprojected 3D points are located, are labeled. Using RANSAC, a plane is fitted to the 3D points. Figure 5.4 shows the features used for labeling the segments after fitting the plane. In the following, the surface normal of a patch is denoted as $\mathbf{n}$.

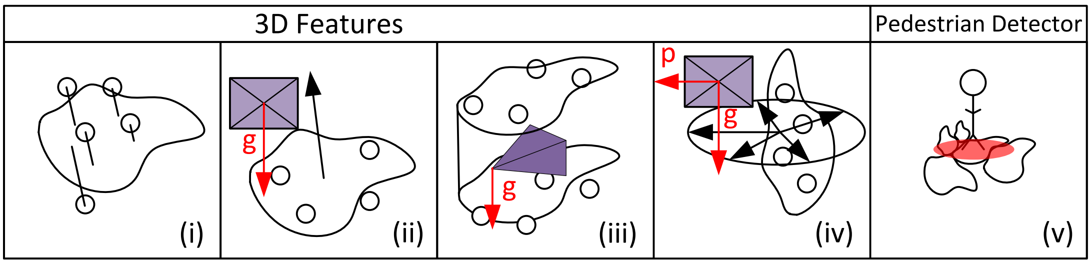


Figure 5.4: Geometric features with respect to the camera's gravity vector $\mathbf{g}$ and a random perpendicular vector $\mathbf{p}$ used for geometric reasoning. (i) Planarity, (ii) ground plane / ceiling orientation, (iii) ground plane position, (iv) vertical plane orientation, (v) segments where footpoints of pedestrians are located are labeled as ground plane.

- **Planarity likelihood** P_{plan}(Figure 5.4(i)): By calculating the Euclidean distances $\mathbf{d}$ between the 3D points and the plane, the planarity likelihood P_{plan} is calculated by

  $$P_{\text{plan}} = \frac{Q(\mathbf{d})_{25} + Q(\mathbf{d})_{75}}{Q(\mathbf{d})_{50}}, \tag{5.2}$$

  where $Q(\mathbf{d})$ are the quantiles of $\mathbf{d}$.

- **Horizontality likelihood** P_{hor} (Figure 5.4(ii)): By determining the surface normal difference between the normal of the plane in question $\mathbf{n}$ and the gravity vector $\mathbf{g}$, the likelihood for the plane being horizontal is obtained by

  $$\begin{aligned} \alpha &= \min(\cos^{-1}(\mathbf{g}\cdot\mathbf{n}), \pi - \cos^{-1}(\mathbf{g}\cdot\mathbf{n})) \\ P_{\text{hor}} &= exp(-|\alpha|\ \pi/180). \end{aligned} \tag{5.3}$$

- **Ground plane / ceiling likelihood** $P_{\text{gp}}/P_{\text{cei}}$ (Figure 5.4(iii)): This feature indicates if the patch in question is more likely to be located on the ground or on the ceiling. Given the center point of the plane in question $\mathbf{a}$ and the camera center $\mathbf{b}$, the likelihood for the segment being located on the ground plane is then given by

  $$P_{\text{gp}} = \begin{cases} 1 & \text{if } \mathbf{a} \text{ below } \mathbf{b} \\ 0.2 & \text{else} \end{cases}, \tag{5.4}$$

  P_{cei} is set up vice versa.

- **Verticality likelihood** P_{ver}(Figure 5.4(iv)): The patch's surface normal $\mathbf{n}$ is rotated around the camera's gravity vector $\mathbf{g}$ with a stepsize of $r = 0^\circ \ldots 5^\circ \ldots 360^\circ$ to obtain $\mathbf{n}_r$. By defining $\beta = [\beta_0 \ldots \beta_{360}]$, the verticality likelihood is then obtained by

  $$\begin{aligned} \beta_r &= \min(\cos^{-1}(\mathbf{p}\cdot\mathbf{n}_r), \pi - \cos^{-1}(\mathbf{p}\cdot\mathbf{n}_r)) \\ P_{\text{ver}} &= exp(-|\beta|\ \pi/180), \end{aligned} \tag{5.5}$$

  where $\mathbf{p}$ is any random perpendicular vector to the camera's gravity vector.

- **Pedestrian Likelihood $P_{\textbf{ped}}$**(Figure 5.4(v)): This feature distinguishes between ground plane and ceiling. Pedestrians are detected using [Felzenszwalb et al., 2010]. It is assumed that the lower boundary of the bounding box is a person's foot point $\mathbf{f}$. To gain robustness, an ellipse (height 5 pixels, width 10 pixels) is defined which also takes neighboring segments into account. The likelihood that a pedestrian is located on a given segment s is defined by
$$P_{\text{ped}} = \begin{cases} \lambda & \text{if } \mathbf{f} \in s \\ \lambda/2 & \text{else} \end{cases}, \tag{5.6}$$
where λ is a multiplier constant in order to increase the likelihood for the segment to be located on the ground plane.

The final likelihoods for the labels are calculated by

$$v(y_i) = P(l|y_i) = \begin{cases} P_{\text{plan}} \cdot P_{\text{hor}} \cdot P_{\text{gp}} \cdot P_{\text{ped}} & \text{if } l \ = \ \text{ground plane} \\ P_{\text{plan}} \cdot P_{\text{hor}} \cdot P_{\text{cei}} & \text{if } l \ = \ \text{ceiling} \\ P_{\text{plan}} \cdot P_{\text{ver}} & \text{else} \end{cases}. \tag{5.7}$$

5.1.1.1 Pixel-wise Labeling

To get a spatial consistent result for the whole image, the label for each pixel is determined independently of the segments of an image. The superpixel segmentation methods in the previous step can be seen as multiple soft priors to the final labeling problem. The solution to this problem corresponds to finding the configuration of a Gibbs distribution with maximal probability which is equivalent to finding the maximum posterior (MAP) configuration of an MRF. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph described by vertices $\mathcal{V}$ which in this case are represented by the N pixels of the image and edges $\mathcal{E}$. As described in Section 3.3.1, when having a set of random variables $\mathbf{X} = \{X_1, X_2, \ldots X_N\}$ and a label configuration $\mathbf{x} = \{x_1, x_2, \ldots x_N\}$ which can take values from the discrete set of labels $\mathcal{L}$, the energy term E of the pairwise MRF is defined by

$$E(\mathbf{x}) = \sum_{i \in \mathcal{V}} \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}(i)} \psi_{i,j}(x_i, x_j), \tag{5.8}$$

where $\mathcal{N}(i)$ is the neighborhood of node i, ψ_i is the unary potential in the graph and $\psi_{i,j}$ is the pairwise potential, or smoothness term, between neighboring pixels. These terms are defined to be

$$\begin{aligned} \psi_i(x_i) &= 1 - \max(u(y_i), v(y_i)) \\ \psi_{i,j}(x_i, x_j) &= \begin{cases} 0.5 & \text{if } x_i = x_j, \\ 1 & \text{if } x_i \neq x_j \end{cases} \end{aligned} \tag{5.9}$$

The MAP configuration $\hat{\mathbf{x}}$ is then found by

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}}\, E(\mathbf{x}). \tag{5.10}$$

In the proposed implementation an 8-connectivity is chosen so that each pixel has 8 neighboring pixels.

5.1.2 Experiments

The proposed algorithm is being evaluated on two datasets. The *Airport* dataset provides an image sequence of 270 images having a resolution of 1001x1001 pixels. The dataset is taken in a challenging indoor airport environment. 100 of the images in the dataset are manually segmented using the labels *ground plane*, *ceiling* and *vertical*. Objects which cannot be classified by the method are not considered in this evaluation (e.g. people, columns) and are marked as black region in the following. The second dataset was taken from [Hedau et al., 2009], which is referred to as the *Rooms* dataset. It provides 314 images with varying resolutions showing cluttered indoor scenes. Ground truth labels are also *ground plane*, *ceiling*, *vertical* and objects are also excluded (e.g. beds, chairs). Since this dataset only provides still images, geometric reasoning cannot be applied.

To exploit consistent pixel-wise labeling, different superpixel methods are applied and the labels are estimated by exploiting geometric reasoning and semantic reasoning using [Hoiem et al., 2007]. To get different superpixel methods, the methods of [Felzenszwalb and Huttenlocher, 2004], [Levinshtein et al., 2009], [Liu et al., 2011] and [Achanta et al., 2012] are used. These four methods are chosen because they provide a vast variation of their segments' shapes and sizes. Each of those methods is performed multiple times using different parameter sets to obtain varying segmentation results. Hoiem's baseline method (with and without an MRF optimization) is compared to the proposed method which is performed six times using varying parameters with and without incorporating information coming from geometric reasoning. This results in eight different runs:

- Hoiem's base method [Hoiem et al., 2007] which uses [Felzenszwalb and Huttenlocher, 2004] for segmentation
- Hoiem's method in combination with an MRF (Hoiem et al. + MRF)
- Hoiem's method applied on four segmentation methods with one parameter set and no geometric reasoning (4SP1N)
- Hoiem's method applied on four segmentation methods with one parameter set and geometric reasoning (4SP1Y)
- Hoiem's method applied on four segmentation methods with two parameter sets and no geometric reasoning (4SP2N)
- Hoiem's method applied on four segmentation methods with two parameter sets and geometric reasoning (4SP2Y)
- Hoiem's method applied on four segmentation methods with three parameter sets and no geometric reasoning (4SP3N)
- Hoiem's method applied on four segmentation methods with three parameter sets and geometric reasoning (4SP3Y)

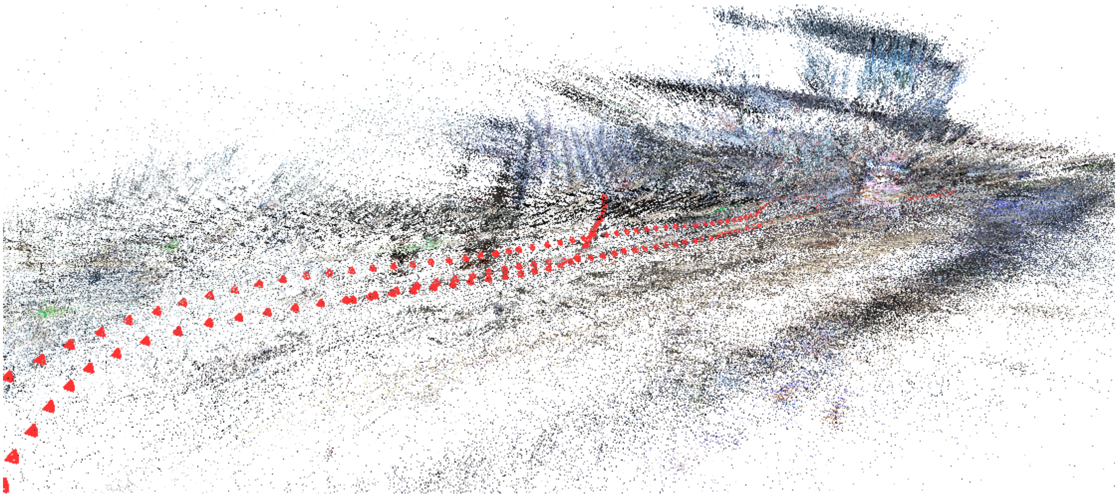


Figure 5.5: Screenshot of the sparse 3D point cloud and the camera positions from the *Airport* dataset.

Airport Dataset								
	Hoiem et al., 2007	Hoiem et al., 2007+MRF	Proposed 4SP1N	Proposed 4SP1Y	Proposed 4SP2N	Proposed 4SP2Y	Proposed 4SP3N	Proposed 4SP3Y
ground plane	63.57	76.77	80.58	80.40	82.83	82.40	82.83	**82.92**
ceiling	36.98	58.20	61.61	64.91	62.54	66.25	62.63	**66.82**
vertical	17.98	23.07	25.37	25.04	25.84	25.62	**25.94**	25.64
global	54.36	70.49	73.42	74.85	74.64	76.22	74.67	**76.48**
average	39.51	52.68	55.85	56.78	57.07	58.09	57.13	**58.46**

Rooms Dataset [Hedau et al., 2009]					
	Hoiem et al. 2007	Hoiem et al. 2007+MRF	Proposed 4SP1N	Proposed 4SP2N	Proposed 4SP3N
ground plane	63.54	71.60	73.06	74.07	**74.91**
ceiling	37.35	**46.12**	42.60	43.19	43.53
vertical	**80.77**	80.63	79.53	80.10	80.10
global	84.87	85.00	85.05	85.81	**85.92**
average	60.55	66.11	65.06	65.78	**66.18**

Table 5.1: Percentage of correctly classified pixels per label for *Airport* and *Rooms* [Hedau et al., 2009] dataset.

The labels for the MRF outcome of all methods are *ground plane*, *ceiling* and *vertical*. On the *Airport* dataset, the SfM pipeline presented in [Snavely et al., 2006] is run to obtain a sparse 3D model of the scene and the camera positions. In combination with Bundler, SIFT features are used to obtain points of interest to match in the images. It uses a modified version of bundle adjustment to obtain both a sparse point cloud and the camera positions. A screenshot of the reconstructed scene can be seen in Figure 5.5.

5.1.2.1 Quantitative Experiments

As the proposed method is calculating a corresponding label for each pixel, the percentage of correctly classified pixels is compared to the ground truth images. For both the *Airport* dataset and the dataset of [Hedau et al., 2009], pixels which are left blank in the ground truth image are not taken into account for the comparison. Having a ground truth labeled image $\mathcal{G}$ and a resulting image $\mathcal{R}$, the accuracy for label l is determined by $\frac{|\mathcal{G}_l \cap \mathcal{R}_l|}{|\mathcal{G}_l \cup \mathcal{R}_l|}$, where $| \cdot |$ refers to the number of pixels of label l in an image. The percentage of correctly classified pixels for each label can be seen in Table 5.1.

***Airport* Dataset:** As can be seen, there is an improvement of approximately 20% between using [Hoiem et al., 2007] and the same method using an MRF for pixel labeling. It can also be seen that there is an improvement between using only the superpixel method proposed by [Felzenszwalb and Huttenlocher, 2004] and using all described methods having multiple parameter sets. The difference between incorporating geometric reasoning and not incorporating is up to 4%, no matter if a segmentation algorithm is applied once or multiple times. The improvement is obviously higher between using a variety of superpixel methods than between a variety of different parameter sets for each method, regardless of incorporating geometric reasoning or not.

***Rooms* Dataset:** There is also an improvement between using a single superpixel segmentation method and multiple ones for this dataset. Since the scenes shown in the images are not as complex as the ones shown in the images of the *Airport* dataset, the method of [Hoiem et al., 2007] delivers better results and the improvement when processing the frame using 4SP1N, 4SP2N, or 4SP3N is not as obvious as for complex scenes. Nevertheless, an average accuracy improvement of almost 6% can be obtained when using multiple segmentation methods. For the *ground plane* label, an improvement of over 10% is reached.

5.1.2.2 Qualitative Experiments

Figure 5.6 shows qualitative results of the proposed method using different parameter settings and a comparison to the state-of-the-art method proposed by [Hoiem et al., 2007]. Each row shows a different image of the scene. The first image of each row shows the manually labeled ground truth data, where black regions indicate objects which are not taken into account in the evaluation. The following columns show the results for [Hoiem et al., 2007], [Hoiem et al., 2007]+MRF, 4SP1N, 4SP1Y, 4SP2N, 4SP2Y, 4SP3N, 4SP3Y. The labels *ground plane*, *ceiling* and *vertical* are indicated by the colors green, blue, and red, respectively. As can be seen, there is clearly an improvement between only considering a single segmentation method and incorporating multiple ones. It is also visible that there is an improvement in labeling the surfaces by incorporating geometric reasoning about the scene.

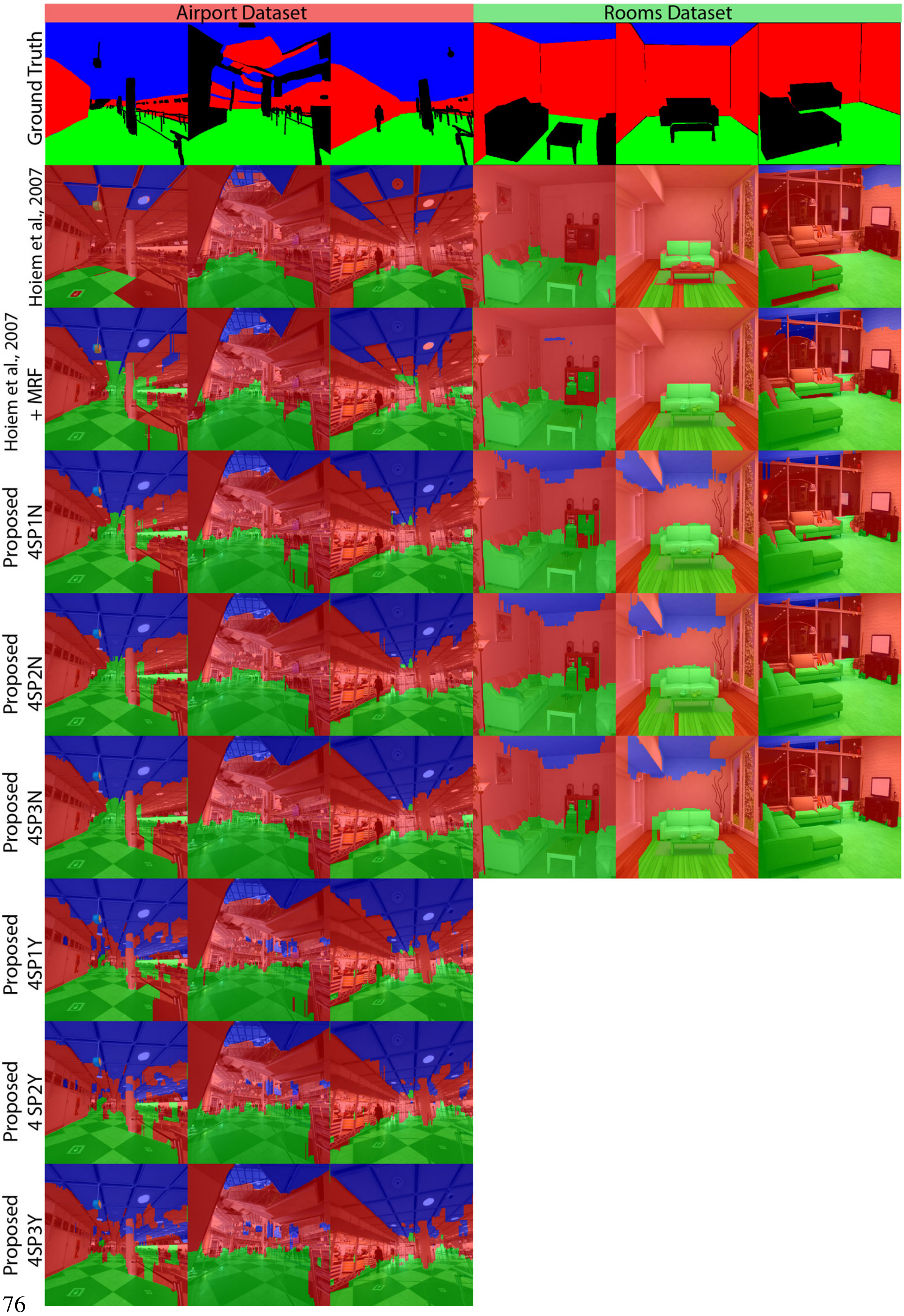


Figure 5.6: Qualitative results. First image of each row shows ground truth (excluded objects are marked black), following columns show results for [Hoiem et al., 2007], [Hoiem et al., 2007]+MRF, 4SP1N, 4SP1Y, 4SP2N, 4SP2Y, 4SP3N, 4SP3Y. Green= ground plane, blue=ceiling, red=vertical.

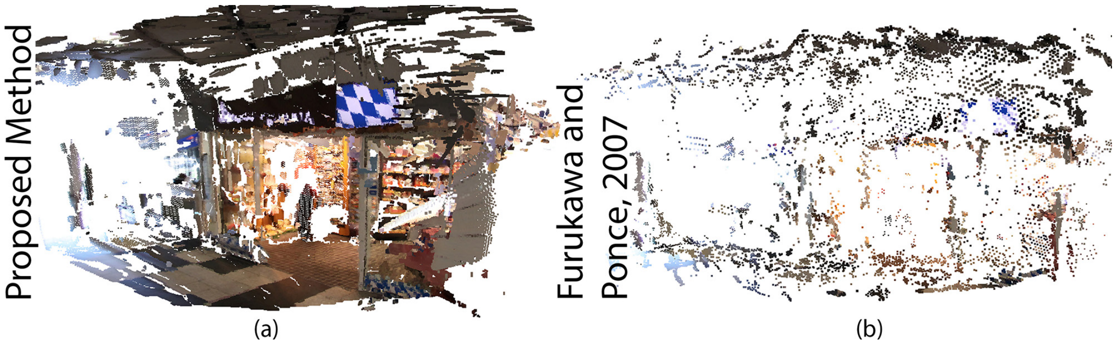


Figure 5.7: (a) Proposed semantic patch-based 3D reconstruction results compared to (b) conventional feature-based 3D reconstruction results [Furukawa and Ponce, 2007].

5.2 3D Reconstruction of Complex Indoor Scenes

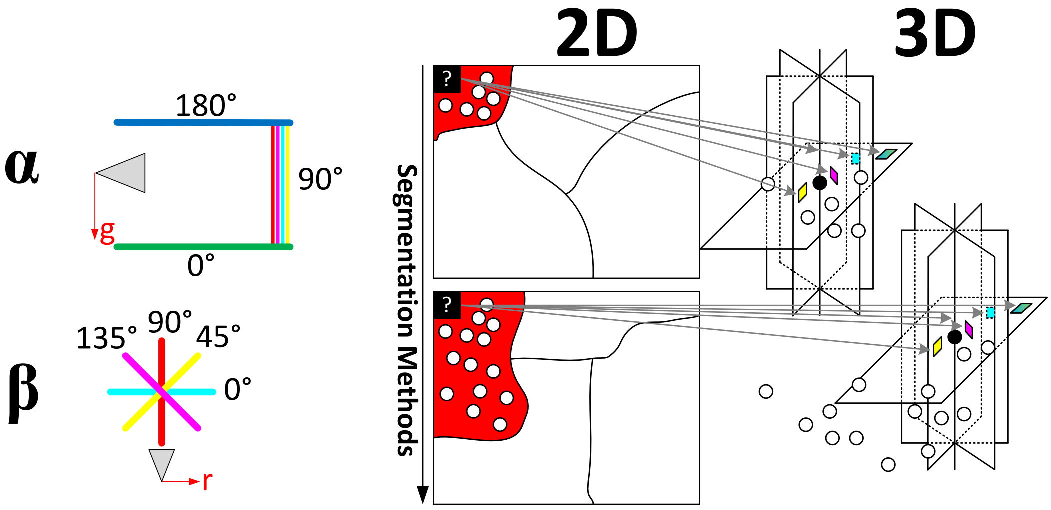
Reconstructing a dense 3D model from a single moving camera capturing a real world environment is usually done by generating a sparse point cloud obtained by triangulation [Snavely et al., 2006] followed by a densification [Furukawa and Ponce, 2007], where both steps rely on discriminative feature matches. In case of man-made environments, this approach is not feasible because of incorrect matches which can occur between corresponding camera views due to similar features obtained from flat and textureless surfaces (e.g. walls, floors). Nevertheless, these conventional 3D reconstruction pipelines deliver correct but sparse 3D point clouds where discriminative features can be extracted (e.g. posters on the wall, texture on the ground and on the ceiling).

This section presents a method which combines a conventional 3D reconstruction pipeline with a patch-based 3D surface normal labeling system in order to overcome the problem of finding discriminative features in man-made environments. As clearly visible in Figure 5.7, (a) incorporating patch-based semantic information in the proposed pipeline gives a more planar and complete model compared to (b) exploiting point-based feature matches only, as proposed by [Furukawa and Ponce, 2007].

In a first step, a sparse point cloud is therefore generated from multiple input images and the 3D camera positions are calculated by using the method described in [Furukawa and Ponce, 2007]. According to [Furukawa and Ponce, 2007], the described method can also generate dense 3D models which is only true when finding discriminative features. In case of man-made environments, the outcome is also sparse, as can be seen in the experiments section. For the following steps, the algorithm is operating on a single input image.

The input image is segmented into semantic meaningful parts using superpixel methods. It is assumed that each segment can be modeled by a planar patch. By using color, texture, perspective features and a boosted decision tree, the 3D plane normal for each segment is estimated by [Hoiem et al., 2007]. Since multiple pixels are classified in a global fashion, it is referred to Hoiem's method as *semantic labeling* in the following parts of this paper.

In order to be able to perform a classification, this normal is chosen from a given set of



(a) (b)

Figure 5.8: (a) Multiple segmentation methods generate multiple 3D points for each pixel. (b) Each pixel can be labeled with one of the labels within the discrete set of orientations.

discrete normal directions. Superpixel methods are designed to perform well under a certain environmental setting (e.g. indoor/outdoor setting, specific lighting, defined color, or object pose). Since this specific setting cannot cover all possible variations in an image (e.g. different lighting conditions, different geometric relationships between objects seen in a scene), a superpixel segmentation method may also deliver incorrect or missing segments. To compensate these errors, redundant information is exploited and the image is segmented using multiple superpixel methods [Felzenszwalb and Huttenlocher, 2004, Levinshtein et al., 2009, Liu et al., 2011, Achanta et al., 2012].

Each pixel is then assigned a possibility to belong to a certain normal orientation class out of the given disrete set. In order to find the global best configuration and therefore the global best normal orientation for each pixel, an MRF is used. By combining the normal estimation with the sparse 3D point cloud, planes are fitted through the 3D points and the cloud is densified. The contribution of this section is therefore two-fold: first, semantic information is used to compensate missing and incorrect feature matches at textureless and non-Lambertian surfaces in man-made environments. Second, redundant information in terms of multiple segmentation methods is used to exploit the advantages of each single one in order to obtain a higher accuracy for both the surface labeling and 3D reconstruction results.

5.2.1 3D Point Cloud Densification Pipeline

The goal of the proposed pipeline is to do both, classifying each pixel according to the labels *ground plane*, *ceiling* and *vertical*, defined by an angle α and *vertical-*β (where $\alpha = 90^\circ$ corresponds to all possible vertical orientations β) and gathering a 3D point for each 2D pixel by only considering a single input image and a sparse point cloud of the scene. The angle α can take values of a discrete set $\mathcal{L}_1 = \{0°, 90°, 180°\}$ and describes the orientation difference between the camera's gravity vector $\mathbf{g}$ and the plane normal $\mathbf{n}$. The orientations $\alpha = 0°$ can be seen as the *ground plane* of the scene, $\alpha = 90°$ can be seen as any *vertical structure*, $\alpha = 180°$ corresponds to the *ceiling*. The angle β can take values from a discrete set $\mathcal{L}_2 = \{0°, 45°, 90°, 135°\}$ and describes the orientation difference between the camera's right vector $\mathbf{r}$ and the plane normal $\mathbf{n}$.

The workflow for obtaining all possible 3D points for each 2D pixel is illustrated in Figure 5.8. The pixel in question is shown as rectangle in the image for demonstration purposes. The segment holding the pixel in question is then projected in 3D space using all 3D points providing a 2D projection within the segment's 2D area. The pixel in question is projected onto the 3D segments for each segmentation method. Note that the center point (denoted as black circle in the image) of the 3D segments varies according to the considered 3D points. This means that multiple corresponding 3D points are available for each 2D pixel.

The workflow of the proposed optimization algorithm can be seen in Figure 5.9. First, a sparse 3D point cloud in combination with corresponding camera positions are generated from multiple input images by using the method described in [Furukawa and Ponce, 2007]. Second, multiple 3D points are generated for each pixel using multiple defined surface normal orientations. Each normal direction, within a discrete set of normals, is then assigned a certain likelihood which is gathered from semantic and geometric reasoning, as can also be seen in Figure 5.9. The globally best result and therefore the best fitting corresponding 3D point is obtained by pixel-wise optimization using an MRF. As the likelihoods from [Hoiem et al., 2007] cannot be compared with likelihoods from geometric reasoning, the optimization is established in two steps in order to obtain the optimized results $\mathbf{v}$, $\mathbf{w}$ and their combination $\hat{\mathbf{x}}$. Note that due to visibility constraints in Figure 5.9, the lines going from pixels in the segmented image to pixels in the optimized image are only samples and not all pixels of the final outcome are connected to an input. In the accompanying implementation, all pixels from the resulting image are labeled.

After generating the point cloud, the algorithm is operating on a single image, which is split into semantically meaningful parts. It is assumed that each segment in the image can be represented by a planar patch. The main problem in this step is that segmentation methods are designed for specific environmental settings (e.g. certain lighting conditions, specific objects or scenes, ...), which means that they may provide incorrect or missing segments when these settings or conditions are not met. To exploit the advantages from several segmentation methods in order to increase the accuracy of the 3D reconstruction pipeline, each frame is segmented by using multiple segmentation methods.

The segmented image is then further used as input for the next step, where the method of [Hoiem et al., 2007] delivers a likelihood for each segment having an orientation α. The likelihood u for a single pixel y_i within the segment s and at image location i is then given by

$$u(y_i) = P(\alpha | y_i). \tag{5.11}$$

As the original implementation differentiates between left and right wall segment, these two likelihoods are combined to obtain the likelihood for the label *vertical* ($\alpha = 90°$). The results obtained by from [Hoiem et al., 2007] are improved by also calculating the vanishing points in

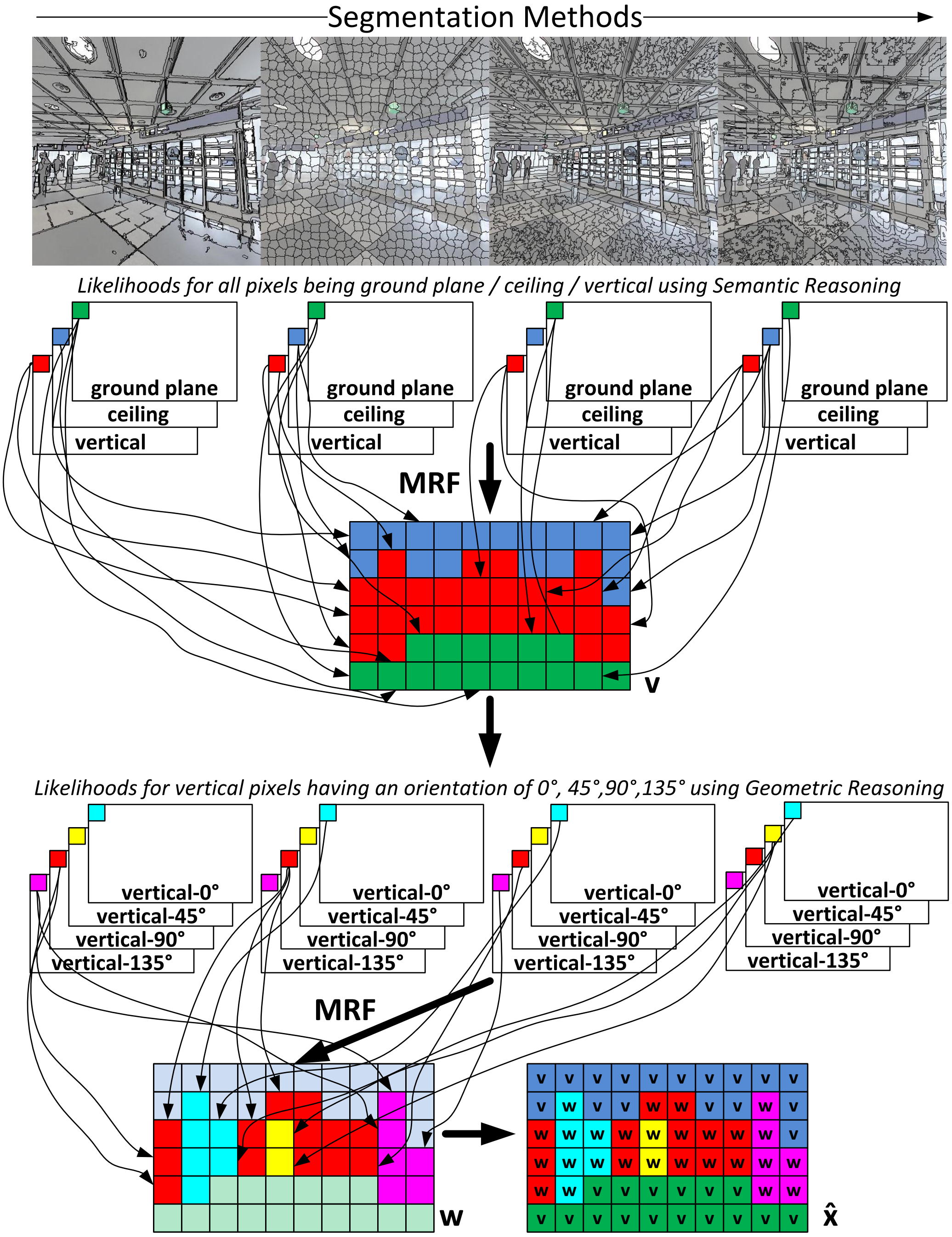


Figure 5.9: Workflow of the proposed optimization which is performed in two steps.

the image. This prevents from labeling pixels above the horizon line as *ground plane* and pixels below the horizon line as *ceiling*.

The next step is referred to as *geometric reasoning*. As the segmented image is already available, geometric reasoning is also performed at this stage. Therefore, all projected 3D points which are within the area of a segment are investigated for each segment. If a certain number of points (5 in the experiments) is found, a plane is fitted through all these 3D points by exploiting RANSAC. The 3D center point of the segment is obtained by gathering the median of all points marked as inlier in the plane fitting step. In case there are not enough points, a segment's normal is set as the median from its 3 neighboring segments' ones. The angle $\phi(\beta)$ between the normal $\mathbf{n}$ of the fitted plane and the normal $\mathbf{n}(\beta)$ of the plane in question is then calculated by

$$\begin{aligned} d(\beta) &= \frac{\cos^{-1}(\mathbf{n} \cdot \mathbf{n}(\beta))}{\|\mathbf{n}\| \cdot \|\mathbf{n}(\beta)\|} \\ \phi(\beta) &= \min(d(\beta), \pi - d(\beta)). \end{aligned} \tag{5.12}$$

To get a spatial consistent result for the whole image, the label for each pixel is determined independently of the segments of an image. The segmentation methods therefore serve as soft priors to the final labeling problem. The solution to this problem corresponds to finding the configuration of a Gibbs distribution with maximal probability which is equivalent to finding the maximum posterior (MAP) configuration of an MRF. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph described by vertices $\mathcal{V}$ which in this case are represented by the pixels of the image and edges $\mathcal{E}$. When having a set of random variables $\mathbf{X} = \{X_1, X_2, \dots X_N\}$ and a label configuration $\mathbf{x} = \{x_1, x_2, \dots x_N\}$ which can take values from the discrete set of labels $\mathcal{L}_1$, the energy term E of the pairwise MRF is defined by

$$E(\mathbf{x}) = \sum_{i \in \mathcal{V}} \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{i,j}(x_i, x_j), \tag{5.13}$$

where $\mathcal{N}_i$ is the neighborhood of node i, ψ_i is the unary potential in the graph and $\psi_{i,j}$ is the pairwise potential, or smoothness term, between neighboring pixels. The unary potentials and smoothness terms are set to

$$\psi_i(x_i) = exp(-u(y_i) \cdot \lambda \cdot d_s) \tag{5.14}$$

$$\psi_{i,j}(x_i, x_j) = 1 - exp\left(-\left|\frac{(\alpha(y_i) - \alpha(y_j))}{180}\right|\right), \tag{5.15}$$

where $u(y_i)$ is obtained by using Hoiem's method, d_s is the 3D Euclidean distance between the center point of segment s (which contains y_i) and the camera center, λ is a normalizing constant and $\alpha(y_i)$ is the surface normal orientation at pixel y_i. Normalization is reached by dividing through 180. The distance d_s is used to increase the likelihood for pixels which are closer to the camera center. When using 4 different segmentation methods, this leads to $4 \cdot |\mathcal{L}_1| = 12$ labels a pixel can obtain. The MAP configuration $\mathbf{v} = \{v_1, v_2, \dots v_N\}$ is then found by

$$\mathbf{v} = \operatorname*{argmin}_{\mathbf{x}} E(\mathbf{x}). \tag{5.16}$$

In the proposed implementation an 8-connectivity is chosen so that each pixel has eight neighboring pixels. The MRF is then solved by using ICM. After having defined if the pixel is located on the ground, the ceiling, or a wall segment, the next goal is to find out the orientation of the wall. Since unary potentials between Hoiem's approach and geometric reasoning cannot be directly compared, a second pairwise MRF is solved between neighboring pixels to find this orientation. The label configuration $\mathbf{x}$ can take values from the discrete set of labels $\mathcal{L}_2$. Unary and pairwise terms are set to

$$\psi_i(x_i) \;=\; 1 \;-\; exp\left(-\frac{\phi(\beta)}{180}\right) \tag{5.17}$$

$$\psi_{i,j}(x_i, x_j) \;=\; 1 \;-\; exp\left(-\left|\frac{(\beta(y_i) - \beta(y_j))}{180}\right|\right), \tag{5.18}$$

where $\beta(y_i)$ is the surface normal orientation for pixel y_i. Once more, when using 4 different segmentation methods, this leads to $4{\cdot}|\mathcal{L}_2| = 16$ labels a pixel can obtain. The final configuration $\mathbf{w} = \{w_1, w_2, \ldots w_N\}$ is then found by solving Equation 5.16. In the last step, $\mathbf{v}$ and $\mathbf{w}$ are combined to obtain the final labeling $\hat{\mathbf{x}}$ by

$$\hat{\mathbf{x}} = \begin{cases} v_i & \text{if } \alpha(y_i) \in \{0^\circ, 180^\circ\} \\ w_i & \text{else} \end{cases} \tag{5.19}$$

For both classification steps, not only a 2D segmentation of the scene is obtained but also corresponding 3D points for each 2D pixel. As stated previously, the 3D center point of each segment is obtained and a plane, where the orientation corresponds to the 2D orientation label obtained in the optimization step, is fitted through it. As the orientation is known at this stage, it is also known from which 2D segmentation the best configuration comes from. For each 2D pixel, the 3D point corresponding to the globally best segmentation result is therefore used.

For demonstration purposes, in the following experiment section all points which are too far away from the camera image plane are sorted out. Therefore, a threshold of ten times the distance between the current and the subsequent camera center is chosen.

5.2.2 Experiments

Experiments are conducted using an indoor dataset holding 270 images showing the indoor environment of an Airport having multiple height layers and complex geometric structures. The images are taken by a person walking on the ground plane with a Canon EOS 5D Mark II. A sparse point cloud and the 3D camera positions are obtained by using all the images in the dataset and the method of Furukawa, described in [Furukawa and Ponce, 2007]. A screenshot of the sparse 3D reconstructed model can be seen in Figure 5.10. Note that the cloud is denser than the one shown in Figure 5.5 in Section 5.1 since multiple local feature descriptors are used. Segmentation is done by using the algorithms described in [Felzenszwalb and Huttenlocher, 2004, Levinshtein et al., 2009, Liu et al., 2011, Achanta et al., 2012].

5.2.2.1 Quantitative Experiments

To show the relevance of the MRF in this approach, the 2D labeling results of Hoiem's method are compared to the proposed method. This means that the label configuration can only take

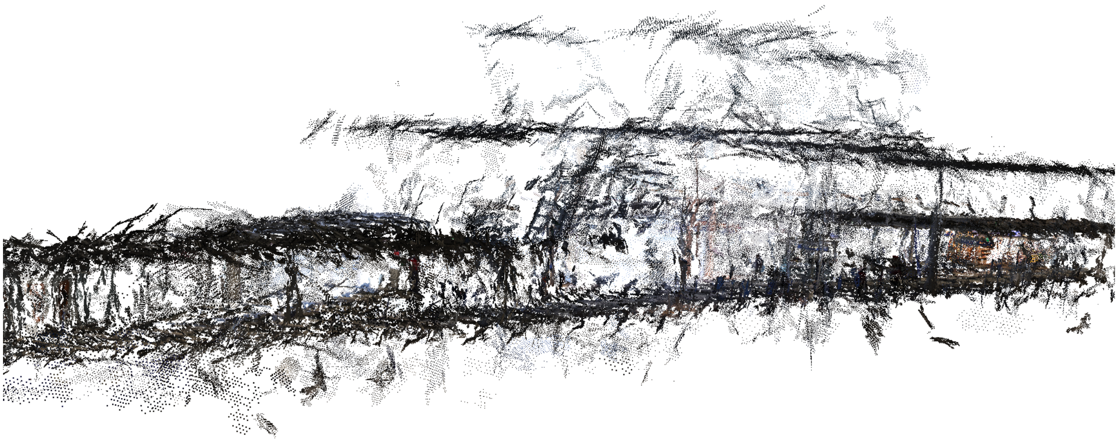


Figure 5.10: Screenshot of the sparse 3D point cloud obtained by [Furukawa and Ponce, 2007].

values of $\alpha \in \mathcal{L}_1$. First, 100 ground truth images are labeled manually, where pixels which are left blank in the ground truth image are not taken into account for the comparison. Note that these pixels are marked in black in Figure 5.11. Having a ground truth labeled image $\mathcal{G}$ and a resulting image $\mathcal{R}$, the accuracy p_l for label l is determined by

$$p_l = \frac{|\mathcal{G}_l \cap \mathcal{R}_l|}{|\mathcal{G}_l \cup \mathcal{R}_l|}, \tag{5.20}$$

where $|\cdot|$ refers to the number of pixels assigned to a certain discrete angle α. The percentage of correctly classified pixels for each label can be seen in Table 5.2, Figure 5.11 shows some sample results for the basic approach of [Hoiem et al., 2007], the proposed MRF approach using a single segmentation method (SM) [Felzenszwalb and Huttenlocher, 2004] and the proposed MRF approach using four segmentation methods (MM). The ground plane is labeled in green, the ceiling in blue and vertical segments in red. As can be seen, using multiple segmentation methods increases the accuracy of labeling the pixel correctly. Note that the results are different compared to the results presented in Section 5.1 due to using (i) different segmentation parameters and (ii) different unary and binary potentials in the MRF.

	Hoiem [Hoiem et al., 2007]	Proposed (SM)	Proposed (MM)
ground plane	68.39	85.54	87.41
ceiling	43.19	75.44	79.61
vertical	22.11	36.39	39.94
global	60.17	82.64	85.19
average	44.56	65.97	68.99

Table 5.2: Percentage of correctly classified pixels per label for the *Airport* dataset.

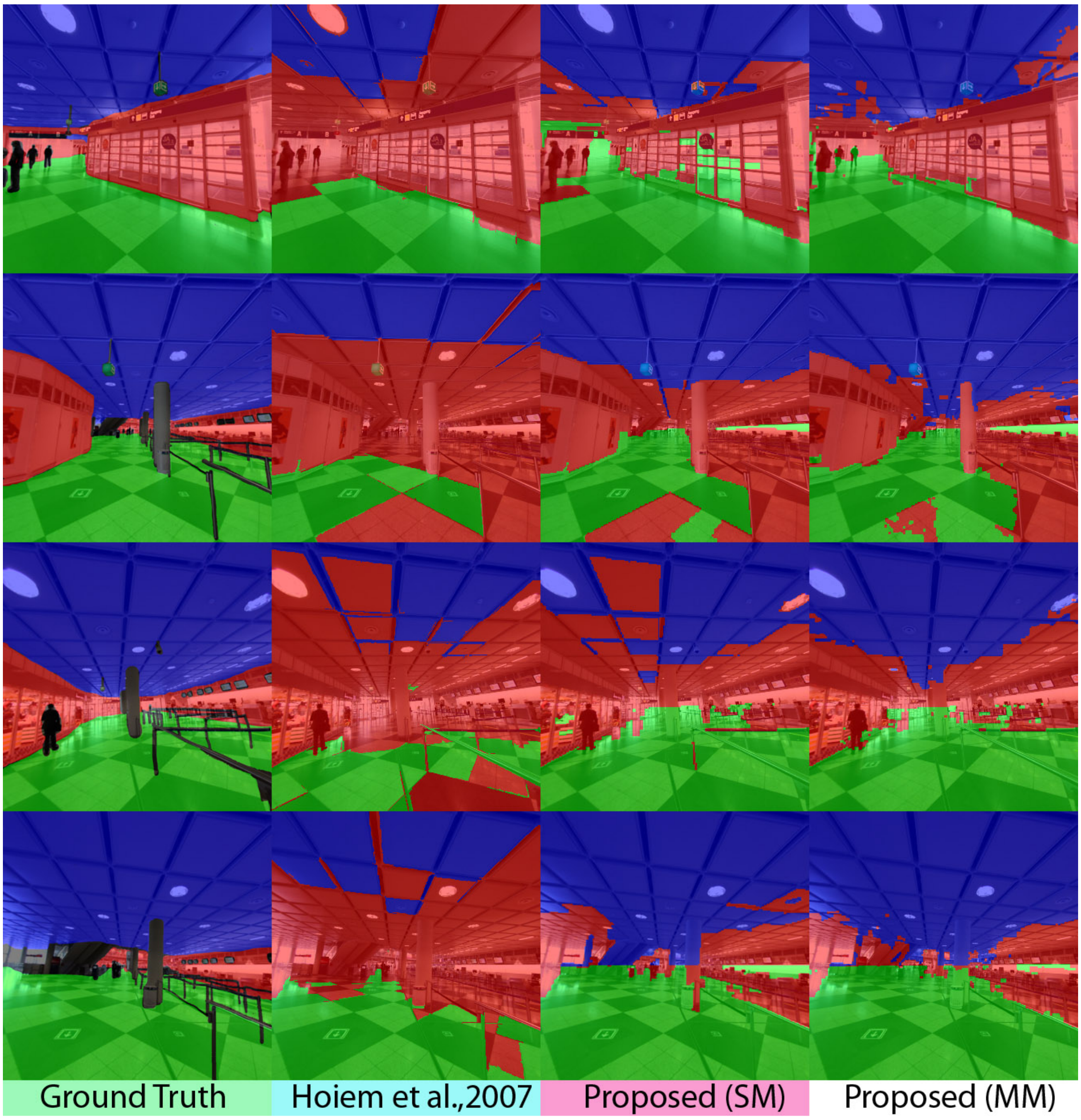


Figure 5.11: Sample labeling results where ground plane = green, ceiling = blue, vertical = red.

5.2.2.2 Qualitative Experiments

After having the labeled images, the reconstruction is established by using the method described in Section 5.2.1. Figure 5.12 shows two sample 3D model results rendered from novel viewpoints. As can be seen, planar patches are reconstructed densely where segments closer to the camera center are reconstructed denser than those patches farther away. As demonstrated by the quantitative experiments, using multiple segmentation methods improves the 2D labeling

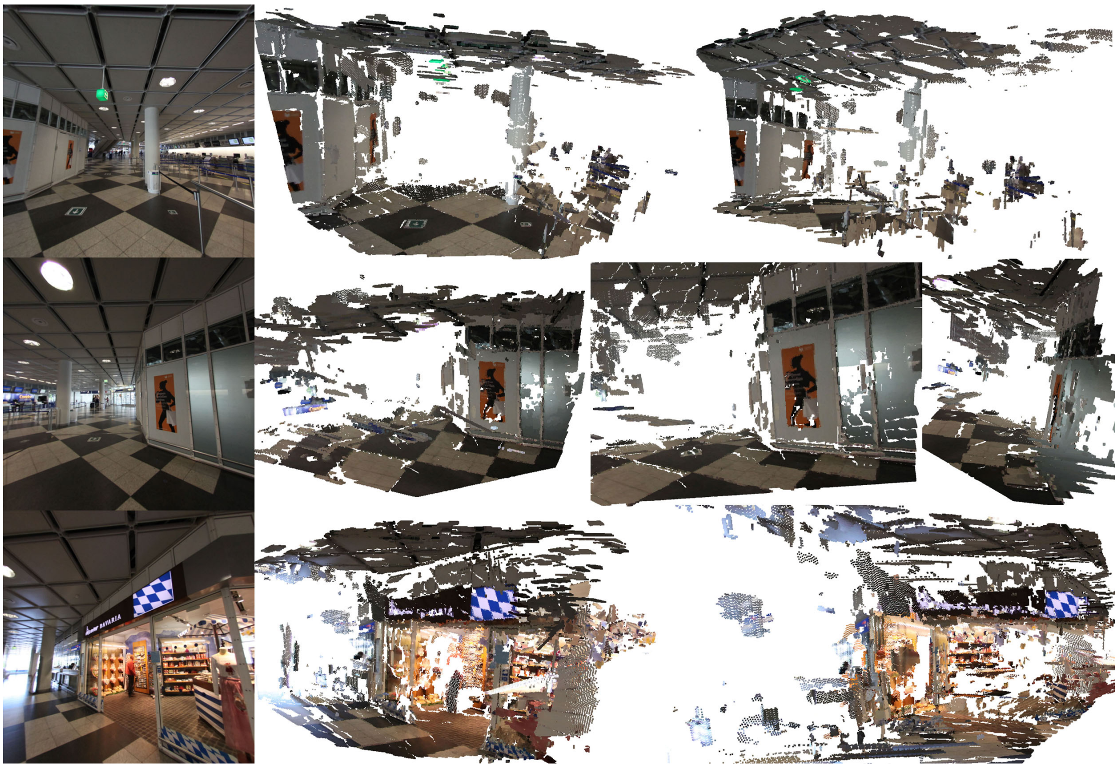


Figure 5.12: 3D models rendered from novel viewpoints using the results obtained by the proposed method.

accuracy. Figure 5.13 shows a qualitative comparison between 3D reconstruction results using a single superpixel method (SM) [Felzenszwalb and Huttenlocher, 2004] and multiple ones (MM) [Felzenszwalb and Huttenlocher, 2004, Levinshtein et al., 2009, Liu et al., 2011, Achanta et al., 2012]. As can be seen, using multiple segmentation methods improves the results of both, the 2D labeling as well as the 3D reconstruction. It is also clearly visible in the 2D labeling images that more errors occur at patches which are farther away from the camera center.

As the goal of this method is to densify a sparse point cloud, Figure 5.13 shows several sample results, where each row shows the reconstruction results for one image. Each column presents the input image, the reconstructed model using the proposed method and the resulting 3D point cloud obtained when using Furukawa's method [Furukawa and Ponce, 2007]. As can be seen, the outcome of the proposed method is much denser and provides a more complete model compared to the outcome of Furukawa's method.

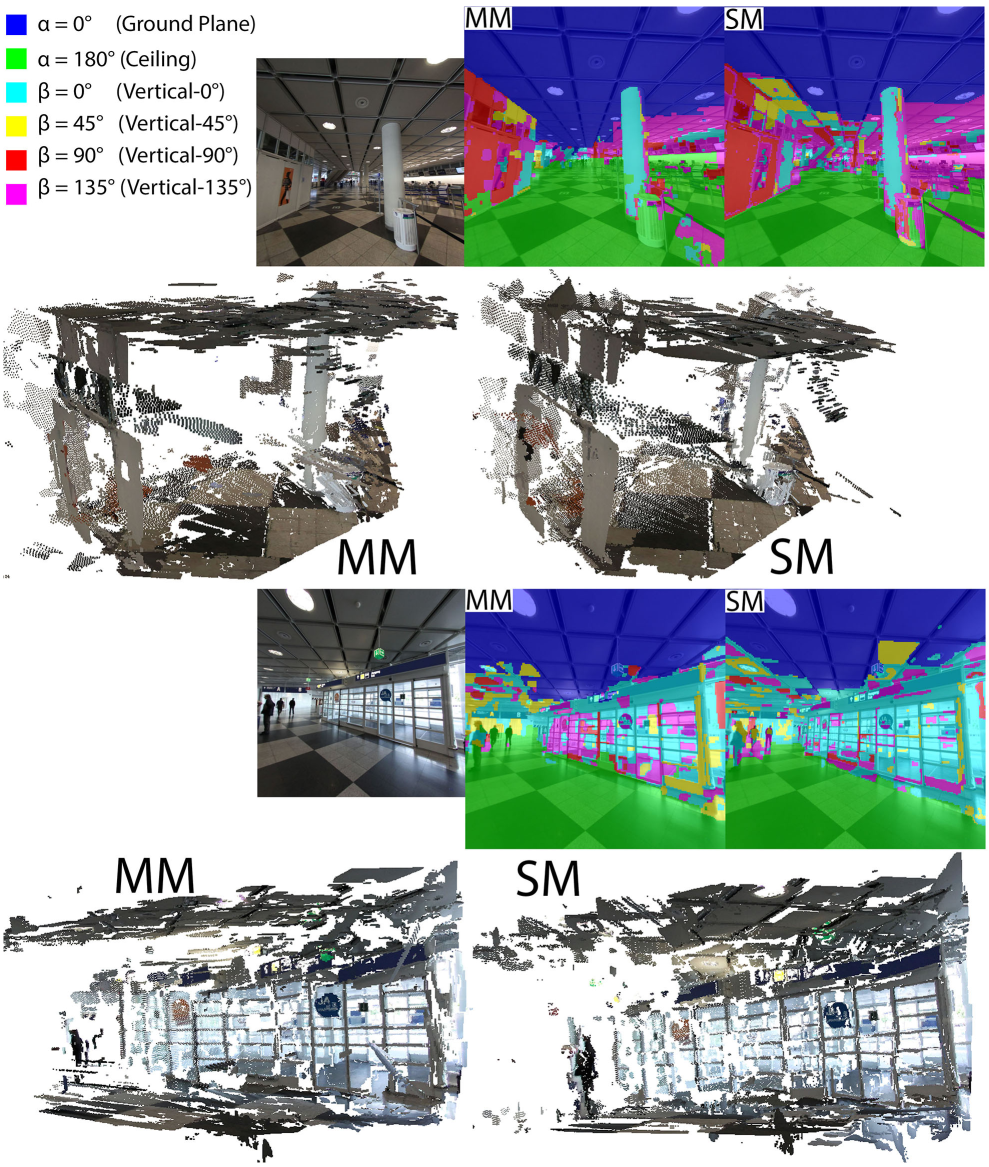


Figure 5.13: Qualitative comparison between 2D labeling and 3D models obtained by using a single segmentation method (SM) and four different ones (MM).

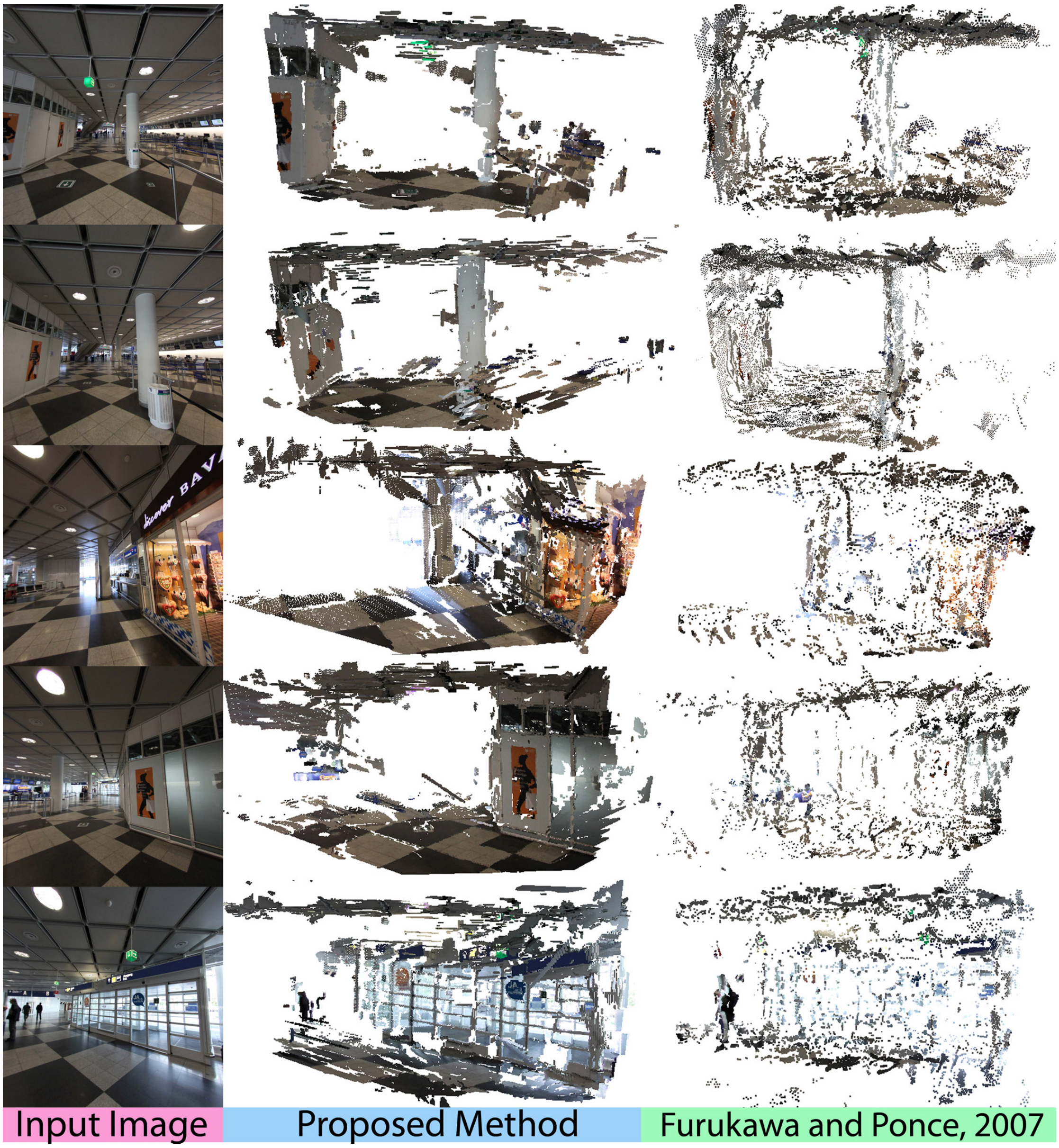


Figure 5.14: Qualitative results. Each row presents results using the input image shown in column 1. The following columns show the reconstructed scene using the proposed method and Furukawa's method [Furukawa and Ponce, 2007].

5.3 Summary

Conventional feature-based dense 3D reconstruction pipelines generate a sparse 3D point cloud and densify the cloud by analyzing the spatial neighborhood of the feature matches. In case of man-made indoor environments, matching is not possible anymore because of incorrect matches between corresponding views due to feature similarity from flat and textureless surfaces. The work described in this chapter tries to overcome this problem by combining point-based features with semantic patch-based 2D information.

First, a framework for estimating the 3D scene layout of a scene is presented. A semantic meaningful patch is obtained by using different cues. Depending on these cues, the resulting patches of different segmentation methods vary in shape and size. By combining the strengths of several superpixel segmentation methods, it is possible to obtain a stronger classifier for labeling each pixel's surface orientation. The labeling accuracy is improved by incorporating geometric features, obtained from 3D point clouds of the scene. The most likely label is then obtained by exploiting an MRF. Experiments on novel and existing datasets show superior results of the proposed approach compared to state-of-the-art methods.

Second, a 3D reconstruction pipeline is presented which performs densification of sparse point clouds obtained from man-made environments. A sparse point cloud and corresponding 3D camera positions are therefore obtained from conventional methods using multiple input images. A single image is then segmented and a likelihood for each segment having a certain 3D surface orientation is calculated by using semantic information. Multiple segmentation methods are used to exploit the advantages of each single one in order to obtain a higher accuracy for both this labeling step and the subsequent reconstruction step which is performed in combination with the sparse point cloud. As can be seen in the experiments section, the proposed method achieves denser and more accurate results compared to state-of-the-art labeling and 3D reconstruction pipelines.

CHAPTER 6

3D Reasoning Using Existing 3D Models

This chapter describes two approaches for 3D pose estimation of vehicles from videos using existing CAD models. The tasks 3D pose estimation and tracking are chosen since they are forming the basis for 3D reasoning frameworks [Geiger et al., 2011, Geiger et al., 2012]. The method is processing images showing vehicles since these objects represent the second most important subject to be analyzed in computer vision besides persons [Leotta and Mundy, 2011]. The advantage of analyzing vehicles over pedestrians is a varying shape and texture when the object is seen from different viewpoints. This makes the matching between 3D models and 2D images more robust. Existing 3D models are exploited since this gives the advantages that (i) the training can be performed without manually labeling the images and (ii) hidden parts of the object in question can also be analyzed since the dimensions and other features (e.g. color, shape,. . .) of the object are also known in advance. Using 3D models prevents from generating 3D scenes which are physically not feasible. As can for example be seen in Figure 6.1, the connectivity of the configuration of the building is ambiguous. The poses are then discretized and ranked based on a matching score between the 3D projection and an input frame. Highly ranked outliers are determined over time by exploiting an optimization framework. This chapter also provides a comparison between matching these 3D models to monocular image sequences and matching them to data coming from a stereo setup.

First, Section 6.1 describes a new framework for classification and pose estimation of vehicles in videos by assuming their given 3D models. First, the estimation of a vehicle's pose and type is cast as a solution of a continuous optimization problem over space and time. Due to a non-convexity of this problem, good initial starting points are important. It is proposed to obtain them by a discrete temporal optimization reaching a global optimum on a ranked discrete set of possible types and poses. In order to efficiently reduce the search space of potential 3D model types and poses for each frame for the discrete optimizer, the ranking of the discretized poses is done by using FDCM. An MRF is used to exploit temporal consistency between consecutive frames. Quantitative and qualitative experiments on a variety of videos with vast variation of

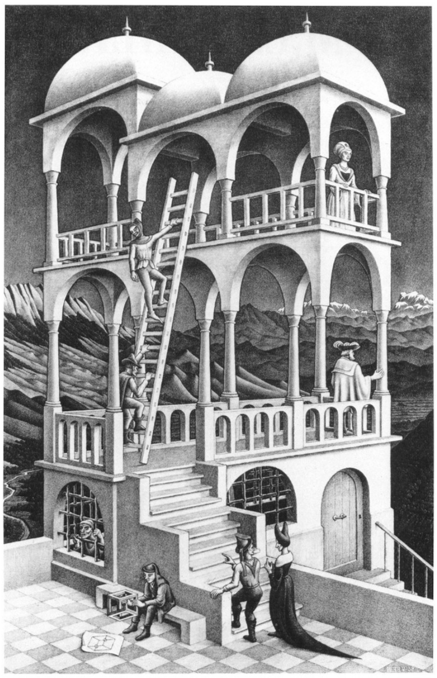


Figure 6.1: Image of the lithographic print named *Belvedere* by M. C. Escher, first printed in 1958. The configuration of the tower looks visually plausible but contains a visual illusion caused by ambiguous connectivities which results in a building which is physically not feasible.

vehicle types show superior results to state-of-the-art methods. Major parts of this method are published in [Hödlmoser et al., 2012].

Second, a computational efficient framework for the same task of vehicle classification and pose estimation is demonstrated in Section 6.2. The advantage of using FDCM is the robustness against outliers which comes with the drawback that similar model compared to the vehicle seen in the image must also exist in the dataset. To overcome this computational problem, a general pose estimator is trained on multiple models. It is not necessary anymore that an exact counterpart of the vehicle seen in the video must also be present in the dataset. The pose estimator then ranks all possible poses for each frame. For both methods, the final best matching pose is obtained by evaluating the temporal smoothness between consecutive poses in an image sequence. Redundancy for both methods is obtained by (i) rendering multiple 3D models from different viewpoints and (ii) estimating the best fitting pose for each frame depending on subsequent and previous frames. As can be seen from the experiments, which are conducted on a variety of videos with a vast variation of vehicle types, the proposed framework achieves similar results in less computational time compared to state-of-the-art methods. This algorithm is also published in [Hödlmoser et al., 2013b].

Third, a novel classification and pose estimation procedure, which shows how information coming from two different views can be aggregated, is presented in Section 6.3. The exper-

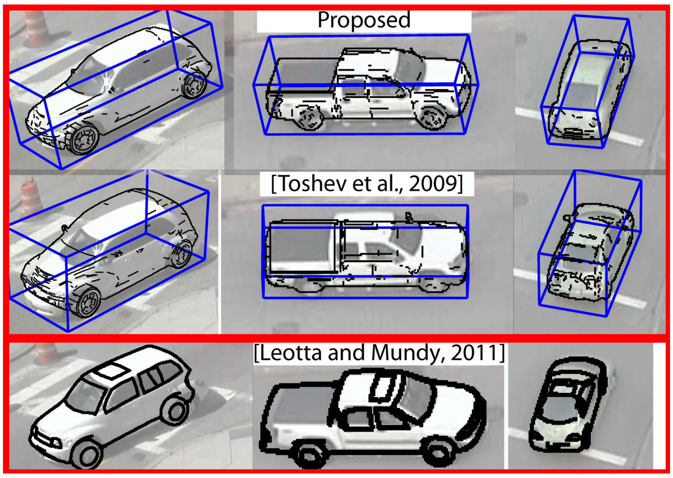


Figure 6.2: Pose estimation can be significantly improved over [Toshev et al., 2009] by processing in continuous space (columns 1, 2), by reducing incorrect classifications due to incorrect scales (column 3) and can be improved over deformable model approaches [Leotta and Mundy, 2011] by using existing 3D models.

iments show how exploiting this redundant information improves the classification and pose estimation performance over only considering the information coming from a single camera. The accompanying work is also published in [Hödlmoser et al., 2011c].

6.1 Vehicle Classification and Pose Estimation Using Chamfer Matching

This section presents a framework for estimating a vehicle's pose and its type by ranking possible poses and types for each frame and exploiting temporal coherence between consecutive frames for refinement. The contribution is two-fold. First, it is proposed to define the problem of estimating a vehicle pose and type as a solution over all possible poses and types along a sequence as a continuous optimization in space and time. To solve the problem in continuous space, starting points are required due to the non-convex objective function which are proposed to be obtained by an initial discrete optimizer reaching a global optimum on a discrete set of ranked types and poses. Hence, the proposed strategy is referred to as a discrete-continuous optimization method. As a second contribution, to cope with computational complexity of the generic class of methods, a novel way to efficiently reduce the search space over the vehicle poses and types for the discrete optimizer is presented. It is shown how a state-of-the-art object detector and FDCM, an OpenGL renderer, the Ackermann principle in the vehicle motion model, and a tree structured MRF to get fast and globally optimal inference in the initial phase can be combined to improve current approaches [Toshev et al., 2009, Leotta and Mundy, 2011]. As can be seen in Figure 6.2, using continuous optimization eliminates two main problems of

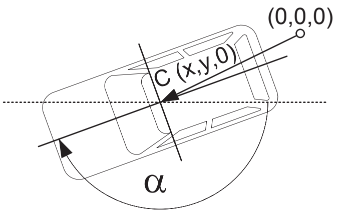


Figure 6.3: Vehicle, described by orientation α and centroid on the ground plane $C = (x, y, 0)$.

e.g. [Toshev et al., 2009], namely i) discrete pose estimation errors (rows 1,2:columns 1,2) and ii) one scaling of an incorrect model may fit better than the best discrete scaling of the correct model (rows 1-2:column 3). By exploiting 3D models with known dimensions, the approach also improves pose estimation over methods using generic models (row 3).

First, a 2D object detector [Felzenszwalb et al., 2010] is applied to obtain some initial hypotheses of the vehicle positions in the images. This brings the advantage of not having a noisy localization compared to background subtraction. Then, it is assumed to have a calibrated setup and a known ground plane (as others, e.g. [Leotta and Mundy, 2011]), to lift the hypotheses to 3D space. This gives a rough estimate of the vehicle's 3D trajectory. After that, a variety of poses which are used to render all 3D models are hypothesized and the FDCM algorithm [Liu et al., 2010b,Liu et al., 2010c] is applied to rank them. FDCM, which works superior in terms of speed and accuracy compared to conventional Chamfer matching [Liu et al., 2010b], uses object contours which have been shown to be robust in case of low resolution images or texture-less objects [Leotta and Mundy, 2011,Payet and Todorovic, 2011]. Exploiting edges for matching the projections of 3D models to 2D images gives the advantage of not dealing with any noisy foreground segmentation, as in [Toshev et al., 2009]. Finally, this work proposes a tree-structured MRF model to compute an optimal pose trajectory in the discrete domain which forces the vehicle to only move with feasible and constant motion. This serves as a reliable initial point for solving the final continuous optimization problem. The optimization is done by exploiting a least squares method, where the distances between projected 3D model points and corresponding 2D edge points in the input images are used to improve the pose estimation result.

Yet, there is no existing work where classification and pose estimation is done continuously in terms of space and time which enables higher accuracy compared to state-of-the-art methods. This is also shown in Section 6.2.4.

This work describes an efficient approach for accurately detecting a vehicle's pose and its type in continuous space, given a calibrated video sequence and a set of 3D models containing a large variety of vehicles. As [Leotta and Mundy, 2011], it is assumed, that given the ground plane, the vehicle's pose is parameterized by $\mathbf{p} = (x, y, \alpha)$ as shown in Figure 6.3. The car's centroid on the ground plane is therefore denoted as $\mathbf{C} = (x, y, z = 0)$, its orientation is described by the angle α.

The whole pipeline can be summarized in the following steps. First, the object in each single frame is detected by using [Felzenszwalb et al., 2010] which does not provide any 3D information. By re-projecting the 2D detection in 3D space, a rough guess on the pose of the object is estimated. By using FDCM, the k most similar models are then obtained by measuring the distance between the projected edges of all 3D models and the edges in the scene, generated by a Canny edge detector. These steps are described in Section 6.1.1. Small variations are then applied on the model's pose, the similarity is measured by combining area overlap and FDCM and all the projections are put in an MRF. By introducing a simple motion model and solving the MRF using a forward-backwards algorithm, the best fitting projection sequence for a given video sequence is determined. The temporal alignment is explained in Section 6.1.2. A final pose refinement, described in Section 6.1.2.2, is applied to get an optimal result in continuous space. Figure 6.4 shows the whole workflow of the proposed framework.

6.1.1 Single Frame Vehicle Matching

Matching vehicles in image sequences can be a hard task since these objects provide specular but large texture-less surfaces, different shapes and different colors. The most stable features which can be used are their edges. For comparing a 3D model to an input frame the model therefore needs to be rendered. This equals to detecting visible edges which can arise from sharp edges between two adjacent faces of the 3D model and from the silhouette of its projection. For speed reasons, the whole rendering pipeline is computed on the Graphics Processing Unit (GPU). Given a vehicle's pose and a calibrated camera, the visibility constraint is therefore exploited to obtain the face which is the closest visible to the camera center for each pixel. For finding all the sharp edges of a model, the normal vectors are calculated for adjacent faces, $\mathbf{n}_i$ and $\mathbf{n}_j$. A sharp edge is found, when $|\mathbf{n}_i \cdot \mathbf{n}_j| \leq$ threshold, meaning that $\mathbf{n}_i$ and $\mathbf{n}_j$ are pointing in different directions. The threshold is empirically chosen to be 0.95 ($= 20.00°$). A visible edge must therefore pass the visibility and the sharp edge constraint. First, the vehicle needs to be roughly located in the image. As can be seen in [Toshev et al., 2009] it is clearly not enough to perform background subtraction to obtain an accurate foreground mask due to highlights and shadows in the scene. A state-of-the-art object detector [Felzenszwalb et al., 2010] is used which returns a bounding box for the 2D vehicle location. Afterwards, the bounding box's centroid is re-projected on a horizontal plane at a certain height above the ground plane. As proposed for track initialization in [Leotta and Mundy, 2011], this height is assumed to be one meter. The gathered 3D point is projected on the ground plane and aligned with the 3D model's centroid on the ground plane. This gives a rough estimate of x and y. The procedure is applied to the first subsequent frames. By fitting a spline through these points, an initial guess for the orientation α is obtained.

The goal of the next step is to determine the class of the vehicle as well as the refined pose. Since the type of the target vehicle in the frame is unknown, it would be necessary to apply small variations of the initial pose guess ($\pm 5°$ in Section 6.1.3) for all the models in the data set, render all poses and perform a similarity search using FDCM to obtain the matching score s. For additional speed-up, this step is performed only on the k best fitting modes where the k best models are found by ranking the FDCM scores of the models in the initial pose. It is empirically found that keeping the hypotheses for the best two models is sufficient for the experiments. For the determination of the matching score using FDCM, let $\mathbf{U}$ and $\mathbf{E}$ be the edge maps of the rendered model and the input image respectively. FDCM maps the edge pixels in $\mathbf{U}$ and $\mathbf{E}$ to an

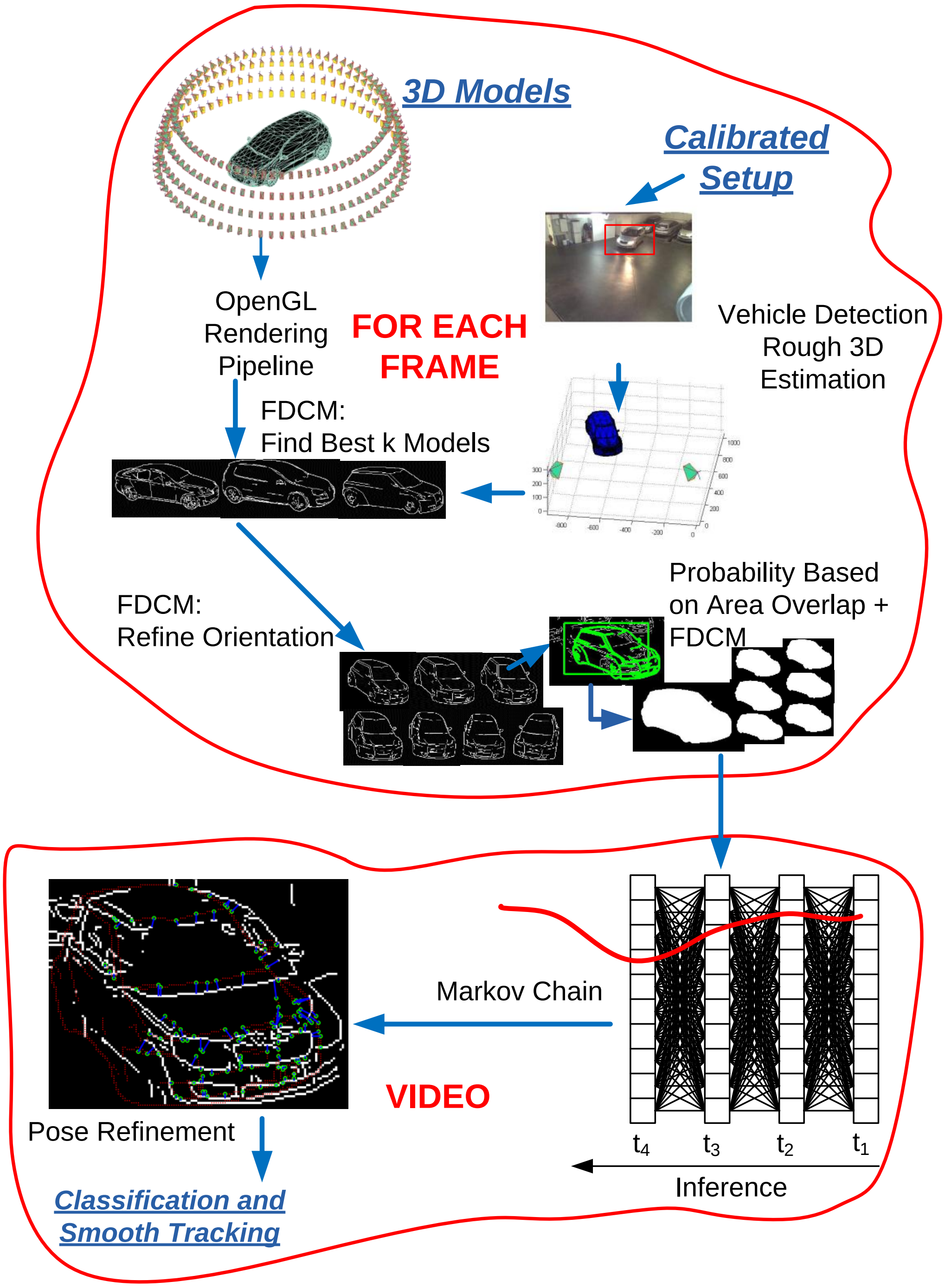


Figure 6.4: Framework application flow.

orientation augmented space. The alignment cost between the two edge maps is then given by

$$d(\mathbf{U},\mathbf{E}) = \sum_{u_i \in U} \min_{e_j \in E} ||u_i - e_j||_2 + \lambda |\angle(u_i) - \angle(e_j)| \tag{6.1}$$

where $\angle(\cdot)$ refers to the orientation of the edge pixel and λ is the weighting factor on the orientation term. The best alignment $\hat{d}$ is then the rigid transformation in 2D image space $\mathbf{r} \in SE(2)$ minimizing $\hat{d} = \min_{\mathbf{r} \in SE(2)} d(\mathbf{U_r}, E)$. Here, $\mathbf{U_r}$ is used to denote the transformation of the edge map $\mathbf{U}$ with the parameter $\mathbf{r}$. In [Liu et al., 2010b], it is shown that via line-segment approximation of the edge maps the FDCM cost can be evaluated efficiently using an integral distance transform structure. Such an approximation is particular beneficial for this approach since the structure of vehicles generally follows some straight line pattern. Moreover, the prior that a vehicle lies evenly on the ground further eliminates the need to search for an in-plane rotation and speeds up the matching process. The matching score is then updated by setting

$$s_{l,\mathbf{p}} = 1 - \hat{d}_{l,\mathbf{p}}, \tag{6.2}$$

where $l = 1 \dots k$, $\hat{d}_{l,\mathbf{p}}$ is the normalized FDCM matching score within $[0, 1]$ for model l, rendered with pose $\mathbf{p}$. FDCM returns the similarity score between the edge image and the rendered 3D model as well as its 2D location. It therefore shifts the projection on the image plane to get the best possible match which results in a re-projection error due to not caring about projective correctness. To handle this, the best k models are rendered by aligning the 2D output location of the FDCM with the vehicle's centroid again and use pose variations $\mathbf{q}$ (in the experiments $\mathbf{p}$ ± 80 centimeters and ± 2°). Given the shifted but projective incorrect model projection $A_l^{\mathbf{p}}$ and a projective correct model projection area $B_l^{\mathbf{q}}$, the similarity score for a pose is calculated by combining the output of FDCM and the area overlap by

$$s_l^{\mathbf{q}} = \frac{s_{l,\mathbf{p}} + \left(\frac{|A_l^{\mathbf{p}} \cap B_l^{\mathbf{q}}|}{|A_l^{\mathbf{p}} \cup B_l^{\mathbf{q}}|} \right)}{2}. \tag{6.3}$$

6.1.2 Temporal Model Alignment

After having ranked all possible poses for the whole image sequence, the best matching pose for each frame is then found in a batch process, which is described in the following. As stated previously, the proposed approach consists of a discrete pose estimation and a continuous pose refinement.

6.1.2.1 Discrete Pose Estimation

Consider an input vehicle sequence $\mathcal{V} = \{v_1 \dots v_t, \dots, v_N\}$ and multiple model sequences $\mathcal{M}_l = \{m_{l,1}, \dots, m_{l,t}, \dots, m_{l,N}\}$, where l is obtained using the method described in Section 6.1.1, v_t is the projection of the vehicle in the input video and $m_{l,t}$ is the projection of model l at time t. The goal is to find the best matching model sequence $\hat{\mathcal{M}}$ which means to find the best fitting model at each time instance t. This implies that i) the whole sequence provides the same

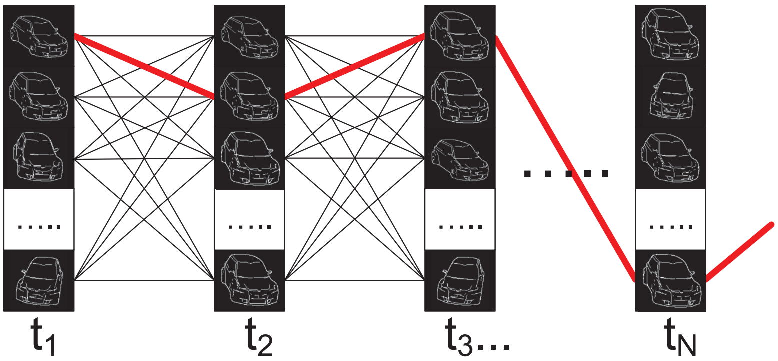


Figure 6.5: Temporal inference for ranked projections.

model type and the vehicle moves with ii) constant and iii) feasible motion. This problem can be solved by calculating the sequential inference as shown in Figure 6.5.

The inference can be calculated by using an MRF which is a chain-structured undirected graphical model, where the joint distribution for $\mathcal{M}_l$ given $\mathcal{V}$ for each 3D model l is determined by

$$P(\mathcal{M}_l|\mathcal{V}) = \frac{1}{Z(\mathcal{V})} \prod_t^N F(m_{l,t}|\mathcal{V}) F(m_{l,t}, m_{l,t-1}|\mathcal{V}), \tag{6.4}$$

where $Z(V)$ is a partition function guaranteeing a probability distribution, $F(m_{l,t}|\mathcal{V})$ is the matching score between a model projection $m_{\mathbf{q}}$, where $\mathbf{q}$ denotes a pose variation, and vehicle at time instance t. $F(m_{l,t}, m_{l,t-1}|\mathcal{V})$ denotes the transition of model l between consecutive frames. The matching score term for each frame is simply determined by

$$F(m_{\mathbf{q}}|v_t) = s_l^{\mathbf{q}}. \tag{6.5}$$

It is assumed that a vehicle moves continuous over time. According to [Siegwart and Nourbakhsh, 2004] and [Scaramuzza et al., 2009], the Ackermann steering principle ensures a feasible vehicle movement by applying different but defined turning radii for the inner and outer wheels of the car. The principle is shown in Figure 6.6. Combining continuous and feasible motion leads to

$$F(m_{l,t}, m_{l,t-1}|\mathcal{V}) = \exp(-(\|\mathbf{p}_{l,t-1} - \mathbf{p}_{l,t}\|^2 + \lambda_2(\varphi - \frac{\theta}{2}))), \tag{6.6}$$

where λ_2 is a weighting constant guaranteeing an equal impact of both terms on the final outcome.

For solving Equation (6.4) and finding the best fitting $\mathcal{M}_l$ for a given $\mathcal{V}$, it is necessary to determine the path having the maximum probability through the graph and compute both Equation (6.5) for each of the model projections and Equation (6.6) for each edge, where an edge should be from each projection of frame at time $t-1$ to each one at time t. Since the model type determined in this step must not change over time, Equation (6.4) is solved for each

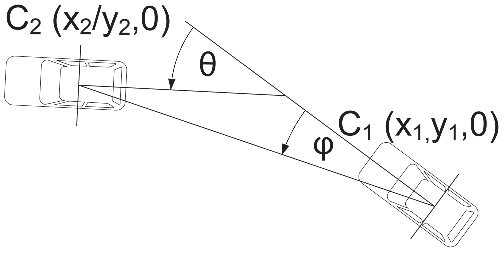


Figure 6.6: Ackermann steering principle where $\varphi = \frac{\theta}{2}$.

vehicle type l. After establishing the whole graph for each type, the problem becomes a labeling problem, where it is needed to find the best matching projection sequence

$$\hat{\mathcal{M}}_l = \underset{\mathcal{M}_l}{\operatorname{argmax}}\, P(\mathcal{M}_l|\mathcal{V}) \tag{6.7}$$

for each class l. This inference can exactly be solved by a forward-backwards algorithm. The vehicle type w for the best fitting model sequence is then obtained by

$$w = \overset{k}{\underset{l=1}{\operatorname{argmax}}} \left(P(\hat{\mathcal{M}}_l|\mathcal{V}) \right). \tag{6.8}$$

The optimized discrete sequence is then given by

$$\hat{\mathcal{M}} = \hat{\mathcal{M}}_w. \tag{6.9}$$

6.1.2.2 Continuous Pose Refinement

Using the MRF, it is only possible to choose the best fitting projection out of discretely rendered 3D models for each frame which means that the results of the previous step depend on the step-size of the discretization. To refine the pose, a continuous optimization of the car's parameters is carried out in 3D space. It is therefore proposed to use a least squares method where the distances between projected 3D model points and image edge points are minimized. At time t, Q_t 3D model points $\mathbf{A}_t$ are considered for which their projected 2D points $\mathbf{a}_t$ are part of a rendered edge. For all points $\mathbf{a}_t$, the Euclidean distance is used to find the closest corresponding points $\hat{\mathbf{a}}_t$ in the input edge image. These corresponding points are shown in Figure 6.7. To assure a smooth movement of the vehicle, the distance between two consecutive poses is additionally minimized over time. Given a camera matrix $\mathtt{K}$, its rotation and translation $\mathtt{R}$ and $\mathbf{t}$, the error for a set of discretely optimized pose vectors $\hat{\mathcal{P}} = \{p_1, \ldots, p_t, \ldots, p_N\}$ for a set of model projections $\hat{\mathcal{M}}$ is calculated by

$$\varepsilon(\hat{\mathcal{P}}) = \sum_{t=1}^{N} \frac{\lambda_3}{Q_t} \sum_{i=1}^{Q_t} \|\mathtt{K}[\mathtt{R}\ \mathbf{t}]\mathbf{A}_{t,i} - \hat{\mathbf{a}}_{t,i}\|^2 + \sum_{t=2}^{N} \|\mathbf{p}_{t-1} - \mathbf{p}_t\|^2, \tag{6.10}$$

where λ_3 is a weighting constant. Note that projected 3D points $\mathbf{A}_{t,i}$ have been converted from homogeneous to 2D coordinates. The iterations are performed by updating the poses

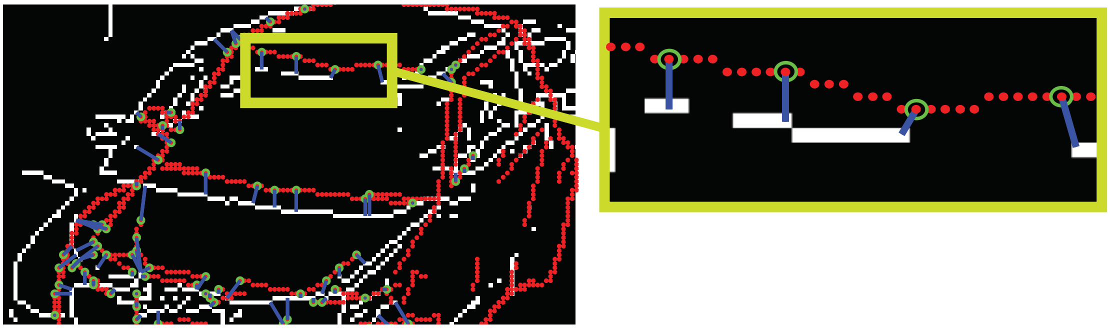


Figure 6.7: Corresponding points between projected edges of the model and edge image.

$\hat{\mathcal{P}}^{j+1} = \hat{\mathcal{P}}^{j} + \Delta\hat{\mathcal{P}}$. The pose update $\Delta\hat{\mathcal{P}}$ is determined by $\mathrm{J}_{\varepsilon}^{T}\mathrm{J}\varepsilon\Delta\hat{\mathcal{P}} = \mathrm{J}_{\varepsilon}^{T}\varepsilon$, where J_{ε} is the Jacobian matrix determined for each ε of Equation (6.10). At every iteration, the points $\mathbf{a}_{t,i}$, $\hat{\mathbf{a}}_{t,i}$ are calculated for each time instance t and the squared error between all pairs as well as between consecutive poses are estimated. After this step, the final projection set $\hat{\mathcal{M}}$ is updated according to $\hat{\mathcal{P}}$. Figure 6.8 shows the results of the pose estimation using simply FDCM (top row), the improved pose estimation result using an MRF (middle row) and the final result using the proposed discrete-continuous optimization (bottom row).

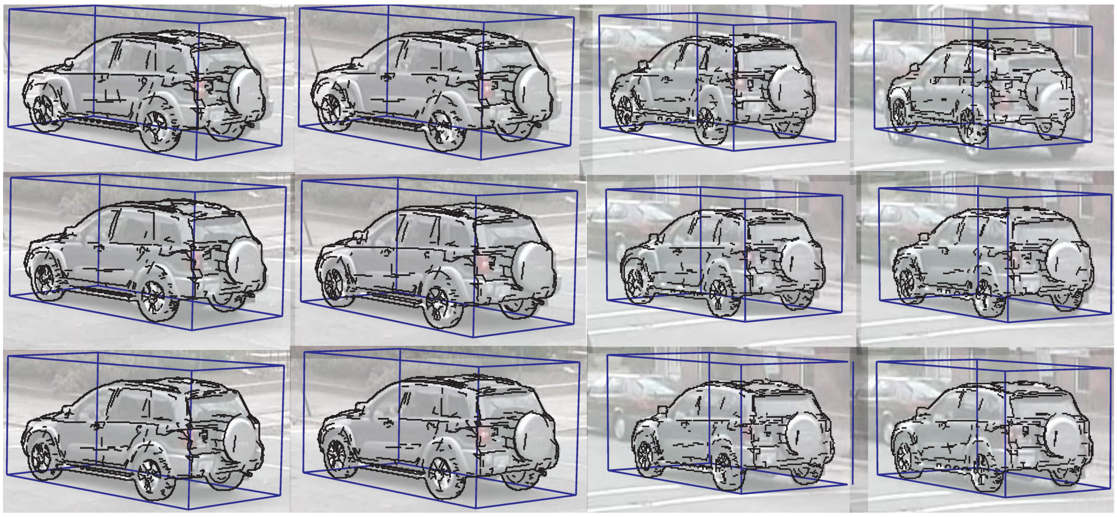


Figure 6.8: Pose estimation using FDCM only (top row), combining FDCM and MRF (middle row), combining FDCM, MRF and continuous optimization (bottom row).

6.1.3 Experiments

Experiments are carried out using eight calibrated real world data sequences. These show six different types of vehicles, provide between 30-150 frames, have a resolution between 640x480 (sequence 1) and 1280x720 (sequences 2-8, taken from [Leotta and Mundy, 2011]) pixels and are captured from different viewpoints. To obtain a classification rate, six corresponding CAD models are downloaded from the Internet for the cars shown in the videos. The dimensions of the models are taken from the manufacturer's specifications. Figure 6.9 shows the six models used which are from left to right: Chevrolet Silverado 2500HD, Chrysler PT Cruiser, VW Beetle, Toyota RAV4, Chevrolet Blazer and Skoda Fabia.

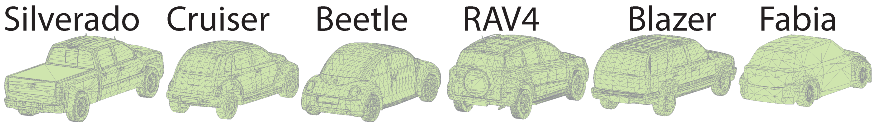


Figure 6.9: Models used for the experiments.

6.1.3.1 Quantitative Experiments

The output for each step of the proposed pipeline (using FDCM only (No Opt), applying the MRF (MRF) and applying pose refinement (Opt)) is shown and compared them to [Toshev et al., 2009] [Leotta and Mundy, 2011]. For generating ground truth data, the 2D area of the vehicle is manually segmented as foreground for each frame. Since Toshev's approach is not publicly available, it is re-implemented and the manual segmentation is used as input. This perfect foreground localization prevents incorrect classifications due to a bad segmentation. To assure fairness in the comparison, the re-implementation of the method is done in such a way that comparable performance on similar videos is obtained as presented in [Toshev et al., 2009]. The dataset is generated with an azimuth, elevation and distance stepsize of 5°, 10° and 50 centimeters, respectively. Leotta's approach [Leotta and Mundy, 2011] is publicly available. Due to the blur in sequence 1, this approach cannot be used on it and therefore this sequence is excluded for the quantitative experiments. First, the overlap between the ground truth region and the projected 3D model is compared. As this metric is used for the evaluation of the detected pose, it is only valid when correctly classified frames are used. In the experiments, a correct classification is obtained for all sequences, whereas Toshev's method gets the correct class in sequences 3-6 and 8. Since Leotta uses a deformable model, it cannot be directly used for classification and therefore all frames are labeled as correctly classified. The graph in Figure 6.10 shows the vehicle detection rate for correctly classified frames at a certain amount of overlap for all the steps of the proposed implementation (No Opt, MRF, Opt) as well as for Toshev's and Leotta's approach. As can be seen, the proposed method is able to outperform both Toshev's and Leotta's method. Toshev's approach uses all scales of a discretely rendered model which are available in the dataset. This can lead to misclassifications, where the incorrect, discrete scale of an incorrect model may fit better than the next best discrete scale of the correct model. As

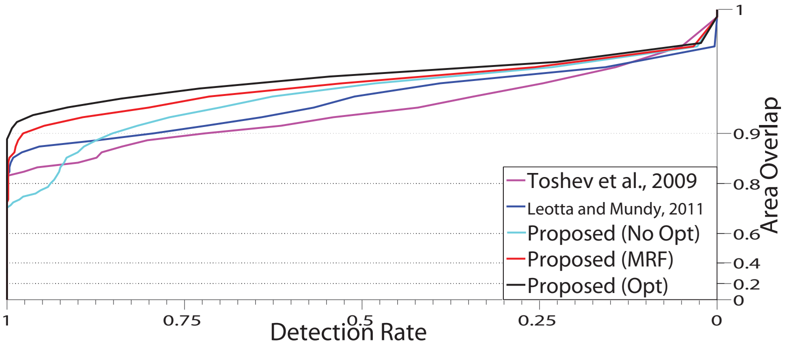


Figure 6.10: Area overlap over all sequences for correctly classified vehicles.

	Proposed(Opt)	Proposed(MRF)	Proposed(No Opt)
μ(cm)	24.68	25.68	28.63
σ(cm)	11.93	13.24	16.38
max(cm)	66.29	87.08	126.08

	Toshev et al., 2009	Leotta and Mundy, 2011
μ(cm)	35.47	28.75
σ(cm)	31.49	15.06
max(cm)	288.45	120.17

Table 6.1: Offsets of the vehicle's center on the ground plane compared to ground truth.

can be seen in Figure 6.10, this does not directly influence the overlap between ground truth and the detected model. Therefore the offset of the vehicle's centroid on the ground plane is also compared to the ground truth data (Table 6.1). As can be seen, the proposed algorithm clearly outperforms both methods when using the optimized results. Leotta's performance is worse due to the deformable model and Toshev's method is worse since it does not incorporate ground plane estimation but only classifies each single frame based on the area overlap between training data and the input frame. The algorithm described in this section provides a maximum deviation to the ground truth of 66.29 cm, Leotta 120.17 cm and Toshev even up to 288.45 cm. Figure 6.11 shows the trajectory of the detected vehicle's centroid over a whole sequence for each method. The vehicle's starting position is denoted by $\times$, $*$, $\star$, $\bigtriangledown$, $\diamondsuit$, $\circ$ and $+$ for Opt, MRF, No Opt, Leotta, Toshev, ground truth and Opt(model n/a) respectively. The results of sequences 1-8 are shown in columns 1-4, ordered from left to right and top to bottom. As can be seen, there are more jumps in space between consecutive frames using Toshev's method than using the proposed one which assumes the 3D model to be located on the ground plane. The last column shows a comparison between the proposed optimized result with and without having the best matching vehicle in the data set.

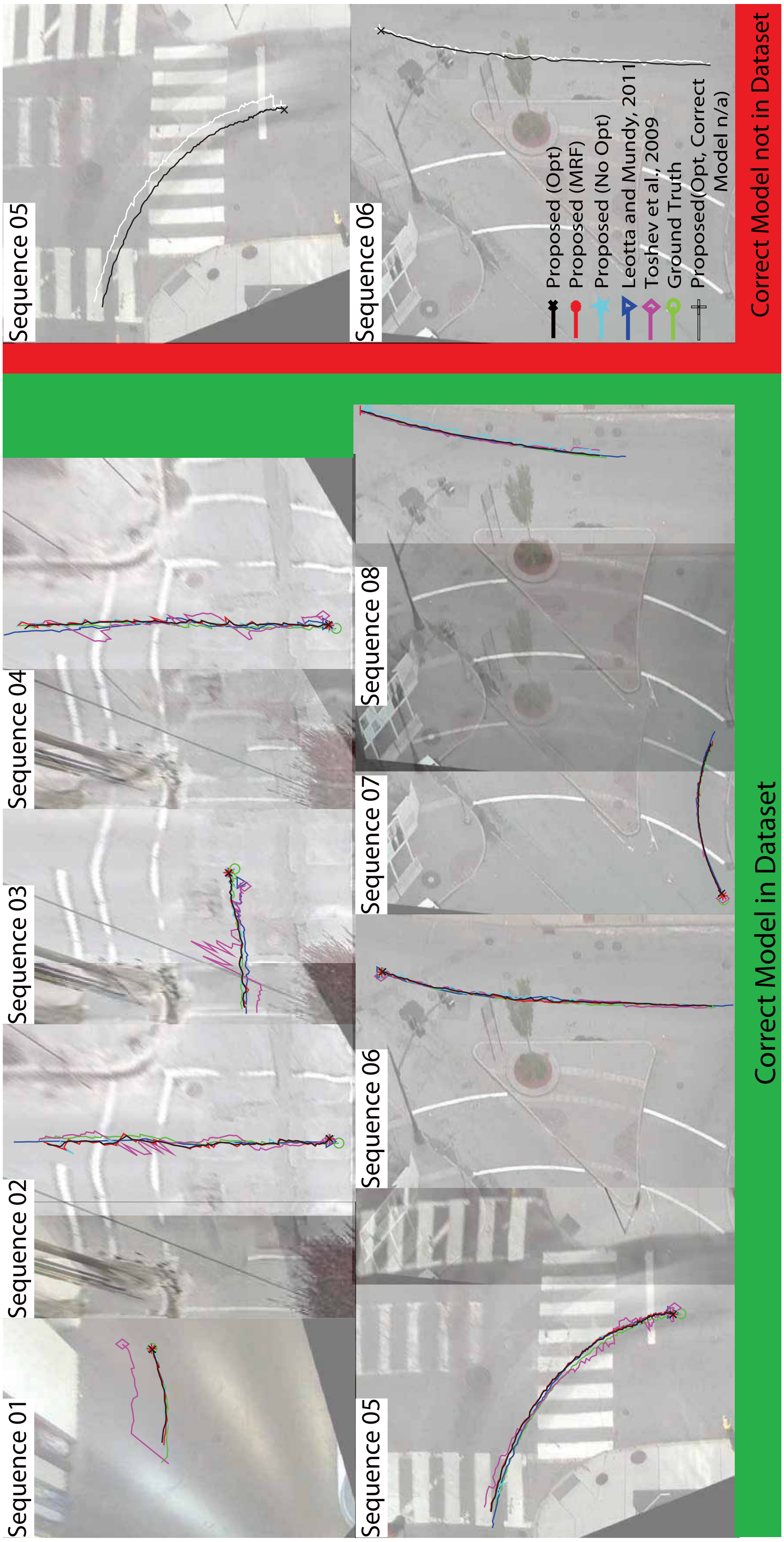


Figure 6.11: Columns 1-4: 3D tracks of car's centroid on warped top-view for all sequences. Proposed method (No Opt, MRF, Opt), [Toshev et al., 2009], [Leotta and Mundy, 2011], ground truth. Column 5: Optimized output with and without best matching model in dataset.

6.1.3.2 Qualitative Experiments

Figure 6.12 shows qualitative example results using the proposed approach. One row shows one sequence, where the leftmost image of each row presents the first frame of it and the proposed optimized projected track. The following frames of the row provide the refined pose of the detected class, projected onto the image plane. For viewing purposes, the region containing the vehicle is cropped. Rows 1-8 show result images for sequences 1-8, where the correct 3D model is within the dataset. The bounding box and the projection of the best fitting 3D model is plotted. Note that also blurry images (e.g. sequence 1) and occlusions (sequence 6, frame 3) are handled over time by the proposed framework. Rows 9-10 show results where the correct model is not in the dataset.

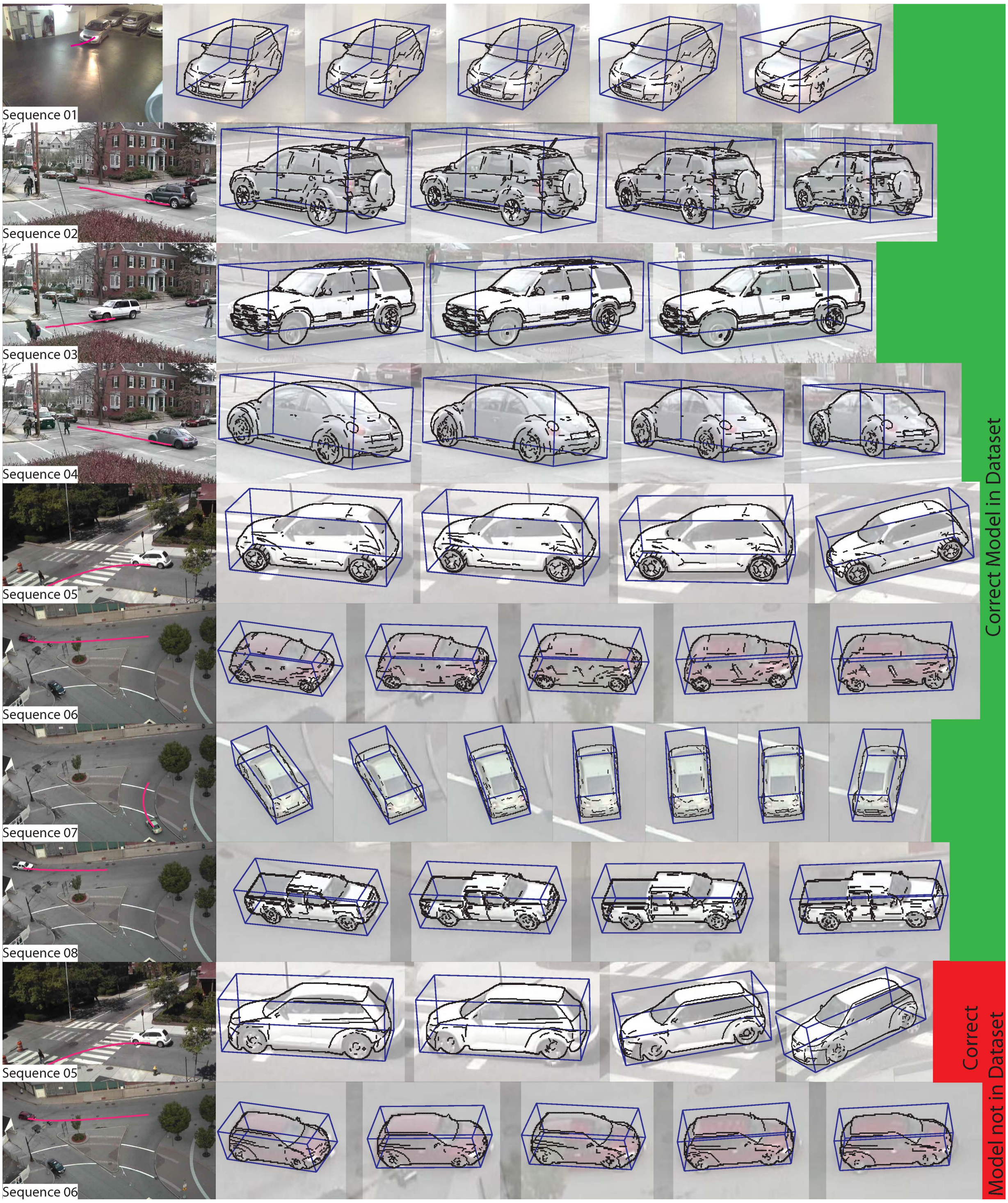


Figure 6.12: Qualitative results. One row describes one sequence where leftmost image shows first frame of it and the proposed optimized projected track. For rows 1-8, correct model is in dataset, for rows 9-10 it is not.

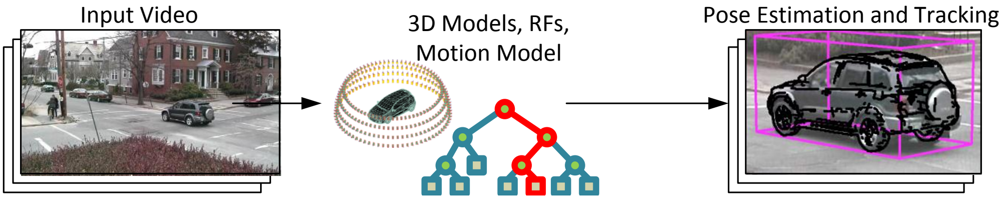


Figure 6.13: A computationally effective 3D model based vehicle detection and tracking framework.

6.2 Vehicle Classification and Pose Estimation Using Random Forests

Similarly to the framework described in Section 6.1, this method also tries to overcome both an object's intra-class variations as well as the ill-posed problem between a monocular image stream and its 3D reconstruction by using a video stream in combination with 3D models having known dimensions for pose estimation and tracking in videos.

As a contribution, an RF ensemble which is trained on a set of existing 3D models and used to rank the vehicles' possible poses and locations in real world input frames (see Figure 6.13), is introduced. Different to the method described in Section 6.1 [Hödlmoser et al., 2012], the pose estimation does not rely on the correct 3D model being in the training set. A generic classifier is trained on multiple models, whereas [Hödlmoser et al., 2012] needs to have the correct model available and evaluates all models in various poses to obtain the correct pose. The synthetic camera orbiting around the model gives correctly labeled training images based on the viewpoints and provides the advantage that the proposed method does not suffer from incorrectly classified training images due to manual pose estimation in real world images. Simple but discriminative principle gradient features are introduced to describe the training images as well as the input images. These features give the opportunity to describe synthetically rendered objects without using depth, texture or color information. Note that of course, the models can also be rendered with color and texture information but the synthesized appearances differ from the real world counterparts due to illumination and other ambient factors. To date, RFs trained on existing 3D models have never been used to overcome the problem of pose estimation. Different to [Fanelli et al., 2011], regression based RFs are not used for pose estimation which gives the opportunity to keep possible poses for each frame and remove outlier poses at a later stage over time and therefore enables making the whole pipeline more robust. A classification based approach is therefore used to rank all possible poses and an MRF is incorporated to ensure temporal consistency between poses of consecutive frames, as proposed by [Hödlmoser et al., 2012].

As in Section 6.1, it is assumed, that given the ground plane, the vehicle's pose is parameterized by $\mathbf{p} = (x, y, \alpha)$. The car's centroid on the ground plane is therefore denoted as $\mathbf{C} = (x, y, z = 0)$, its orientation is described by the angle α. Tracking a vehicle can be seen as finding the perfect pose in continuous space for each frame and connecting subsequent poses by exploiting a feasible motion model over time. Due to computational complexity, evaluating all possible poses is not feasible in practice, so the problem is cast as the determination of a set of poses in discrete space. Having a video which provides N frames and given a discrete input vehicle sequence $\mathcal{S} = \{\mathbf{s}_1 \dots \mathbf{s}_t \dots \mathbf{s}_N\}$ the goal is to find the pose sequence $\mathcal{R} = \{\mathbf{p}_1 \dots \mathbf{p}_t \dots \mathbf{p}_N\}$. This is established by finding the best matching model projection at each time instance t as well as determining the best transition between consecutive frames. This means the model must move in 3D space by following a feasible motion. Finding a solution to this problem is done by calculating the sequential inference which is established by using an MRF, a chain-structured undirected graphical model. The pose for a current frame is therefore inferred from past and future poses in a batch process. Given $\mathcal{S}$, the joint distribution for a model sequence $\mathcal{R}$ is denoted by

$$P(\mathcal{R}|\mathcal{S}) = \frac{1}{Z(\mathcal{S})} \prod_{t}^{N} Y(\mathbf{p}_t|\mathcal{S}) Y(\mathbf{p}_t, \mathbf{p}_{t-1}|\mathcal{S}), \tag{6.11}$$

where $Y(\mathbf{p}_t|\mathcal{S})$ is the matching score between a vehicle pose and the vehicle shown in the video at time instance t, $Y(\mathbf{p}_t, \mathbf{p}_{t-1}|\mathcal{S})$ describes the transition of the model between consecutive frames and $Z(\mathcal{S})$ assures a probability distribution. The following sections explain how to get both terms of Equation 6.11 for this specific tracking problem.

6.2.1 Discrete Vehicle Pose Definition

In order to rank poses for each frame, it is first needed to specify which poses are possible and how similar poses are clustered to the same class. The 3D model is therefore placed at the origin of the coordinate system and orbit a synthetic camera around it. Vehicles may provide a vast variety of dimensions and shapes but all of them provide some common features if seen from the same viewpoint. It is therefore proposed to tag all projections of multiple models seen from the same viewpoint with a common class label. As proposed by [Liebelt et al., 2008], the camera therefore orbits around the object in discrete space to decrease computational complexity by varying azimuth and elevation angle, as well as the distance between the synthetic camera and the coordinate center, as can be seen in Figure 6.14. To avoid misclassifications, the 3D model is sampled by a azimuth stepsize of $5°$, an elevation stepsize of $10°$ and a distance stepsize of one meter. These parameters are set to be an empirically found trade-off between having an accurate enough pose estimation whilst avoiding too fine granularity. Regression based RFs are not used in the proposed method. Using regression has the advantage of penalizing poses based on their distance to a correct result but the problem is that this method does not provide any confidence how well a certain pose fits the input frame. When the regression based RF obtains a completely incorrect pose, the proposed framework will not recover anymore which will yield to an incorrect result. Using classification instead gives the opportunity to count the number of trees voting for a specific class, take this as the pose confidence and remove outliers over time. Decreasing the stepsize means increasing the number of classes. Having too many classes is equal to using regression with the same penalty for all incorrect poses which also results in increasing the number of misclassifications and may not be handled by further steps anymore.

Vehicles provide low-textured but specular surfaces. Following the vehicle pose estimation and tracking methods described by [Leotta and Mundy, 2011], [Payet and Todorovic, 2011] and

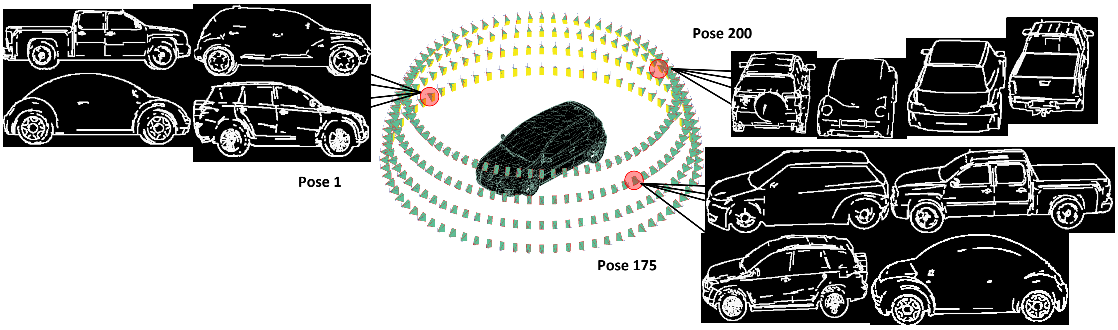


Figure 6.14: Discrete rendering of 3D models placed at the origin of the coordinate system. Multiple models seen from the same viewpoint are clustered into the same class. A class is defined by azimuth, elevation angle and the distance from the synthetic camera to the origin.

[Hödlmoser et al., 2012], edges are the most informative and stable features to use in this case. In the proposed framework, synthetic 3D model projections are therefore used. The training data is a set of synthetically rendered 3D models using a synthetic camera with a given focal length. Edges must be shown when sharp edges between adjacent faces or edges from the projection's silhouette occur. This rendering is performed on the GPU due to computational complexity. Contour edges are found by projecting all faces onto the 2D image plane. Sharp edges are drawn when normals of adjacent faces of a 3D model are pointing in different directions. For obtaining realistic looking projections, this difference is empirically chosen to be 20°.

6.2.2 Random Forests for Pose Estimation

For each frame, all model projections are ranked based on how well they fit to the input frame. This can efficiently be done by exploiting RFs. An RF is a classifier which consists of multiple decision trees and can be used for solving multi-class labeling problems. It was first introduced by [Amit and Geman, 1997] and extended in [Breiman, 2001]. By evaluating a feature using all the trees in the forest, each tree votes for a class. Each of those features of course only provides a weak assumptions about which class it belongs to but by evaluating multiple features and cleverly building up the decision forest, the whole classifier is proven to be robust [Shotton et al., 2011, Lepetit et al., 2005].

Let an RF, built up by $b = 1 \dots B$ of trees, classify between $l = 1 \dots L$ classes. Each tree consists of split nodes, built up by a feature θ and a threshold as well as leaf nodes. The threshold is used for following the tree to its left or right branch. Each of the trees then votes for a single class l. Then, the whole vote distribution of the forest is used for ranking the poses.

6.2.2.1 Random Forest Training

The authors in [Villamizar et al., 2011] utilize real images, HOG features and RFs for training a pose estimator. Different to the proposed approach, they are not using 3D models. It is therefore proposed to use principle gradient edge features instead of HOG features because it is empirically found that these features work better for the 2D to 3D model matching problem. In the proposed framework, multiple principle gradient features describe a 3D model projection. Synthetically generated data is a synthesized copy of its real world counterpart without any noise, occlusions, or variations. The classification of real world images can therefore be improved by intentionally varying the training images on which the classifier is trained. This can be done by changing the threshold for sharp edges, by introducing noise, and by varying the 2D location of the projection. All these variations can be seen in Figure 6.15, where the introduction of noise is performed by simply placing the model projection on a different background image.

First, the gradient direction image $\mathcal{G} = \arctan(\frac{\mathcal{G}_x}{\mathcal{G}_y})$ is determined, where each pixel represents an angle in radians and $\mathcal{G}_x$ and $\mathcal{G}_y$ are the derivatives of an input image in both horizontal and vertical direction. Since the proposed method uses a synthetic camera, all training images are aligned to each other. Given a pixel location $\mathbf{X} = (x, y)$, two points $\mathbf{X}_1 = (x_1, y_1)$ and $\mathbf{X}_2 = (x_2, y_2)$, both having a random offset ϕ to $\mathbf{X}$ in both directions are determined. Given a certain blocksize (20 pixels in the experiments), the mean gradient directions $\psi(\mathbf{X}_1)$ and $\psi(\mathbf{X}_2)$ of each block are calculated. The feature at location $\mathbf{X}$ is then computed by

$$\theta(\mathbf{X}, \phi) = \mod \left(\arctan \left(\frac{\sum_{i=1}^{2} \sin(\psi(\mathbf{X}_i))}{\sum_{i=1}^{2} \cos(\psi(\mathbf{X}_i))} \right), 2\pi \right). \tag{6.12}$$

To cover variations between classes, $M = 500$ features are roughly even distributed randomly over the area covered by all 3D model projections to obtain random features $\theta(\mathbf{X}, \phi_1) \dots \theta(\mathbf{X}, \phi_M)$. To build up an RF, each tree within the same RF chooses a location $\mathbf{X}$ at random but common over all nodes of the tree. For each of the nodes of the tree, $\mathbf{X}_1$ and $\mathbf{X}_2$ are chosen randomly. As stated in Section 6.2.1, the models are rendered using a stepsize of one meter for the distance between the model and the camera center. It is proposed to generate an RF for each distance in order to keep misclassifications and therefore incorrect poses ranked high on a minimum. This gives $L = 288$ classes for each RF within the proposed framework.

6.2.2.2 Single Frame Pose Estimation

The RF can be used for localizing the vehicle in 2D. However, it is time-consuming because all decision trees at each pixel location have to be evaluated. Hence the RF is not used for localization but a state-of-the-art object detector [Felzenszwalb et al., 2010]. It is applied at each frame and returns a bounding box for the 2D vehicle location. The detector by its nature also provides a rough pose estimation but it does not provide any 3D information. Hence, the detector's pose output cannot be used in this case. As can be seen in [Toshev et al., 2009] it is obviously not enough to perform background subtraction to obtain an accurate foreground mask due to highlights and shadows in the scene.

The vehicle's pose and its location on the ground plane are determined by exploiting an RF ensemble. Therefore, $a = 1 \dots A$ RFs having $b = 1 \dots B$ trees are used, where each tree is trained with a different distance between camera and vehicle model. To increase efficiency, the number of RFs to be evaluated is minimized by determining the rough distance between the car and the camera center. For the experiments, the threshold between the rough distance between

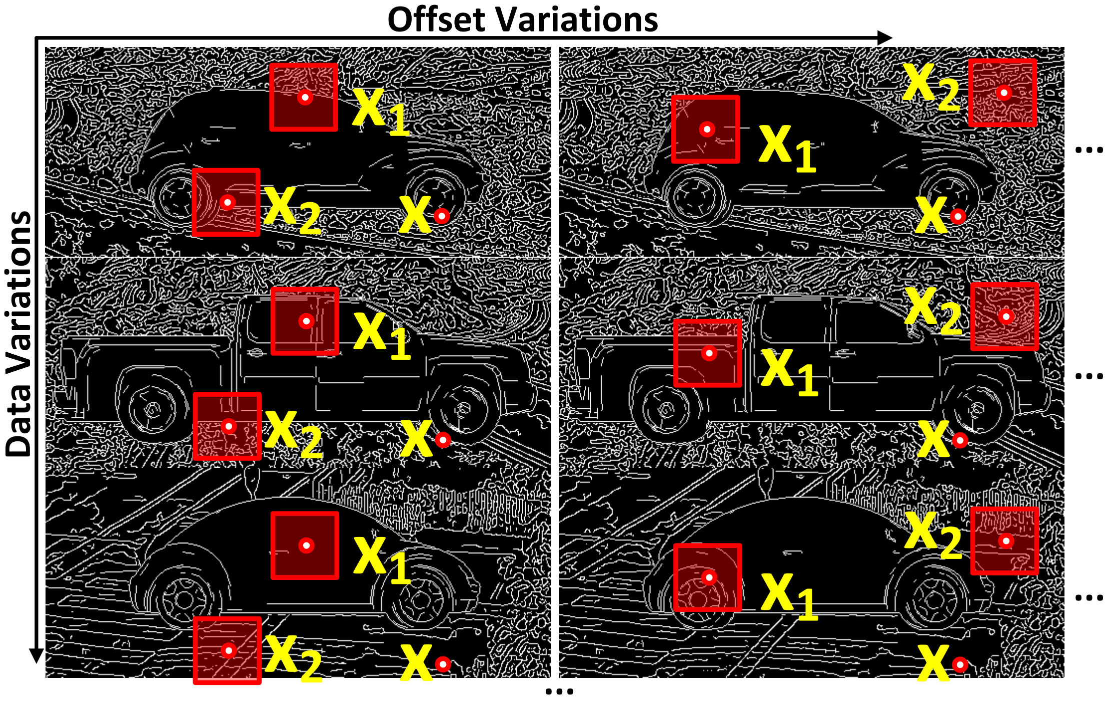


Figure 6.15: Feature Extraction for a variety of different projections but for the same class. A class is defined by the same viewpoint, where robustness to outliers is gained by placing the projection on different backgrounds, changing the threshold for sharp edges at the rendering process, shifting the model in 2D and using a variety of different 3D models.

camera and 3D model and the trained distance of the RFs is chosen to be ± 1.4 meters, so that only the closest three RFs are being evaluated for each frame.

The vehicle's pose and its location on the ground plane are determined by exploiting an RF ensemble. In this work, $a = 1 \dots A$ RFs are used having $b = 1 \dots B$ trees where each RF is trained with a different distance between camera and vehicle model. To increase efficiency, the number of RFs to be evaluated is minimized by determining the rough distance between the car and the camera center. For the experiments, a threshold between the rough distance between camera and 3D model and the trained distance of the RFs are chosen to be ± 1.4 meters, so that only the closest three RFs are being evaluated for each frame. A feature vector $\mathbf{\Theta}(\mathbf{X}) = \theta(\mathbf{X}, \phi_1) \dots \theta(\mathbf{X}, \phi_M)$ is then extracted with the same offsets $\phi_1 \dots \phi_M$ as in the training stage. The vehicle's 2D bounding box in the gradient image $\mathcal{G}$ of the input image is aligned with the training images and the features are calculated for the 2D bounding box of each video frame. Each tree from the RF ensemble holds a vote distribution $P_{a,b}(l|G, \mathbf{\Theta}(\mathbf{X}))$ for label l. The probability for label l is then given by

$$P(l|G, \mathbf{\Theta}(\mathbf{X})) = \frac{1}{A} \sum_{a=1}^{A} \frac{1}{B} \sum_{b=1}^{B} P_{a,b}(l|G, \mathbf{\Theta}(\mathbf{X})). \tag{6.13}$$

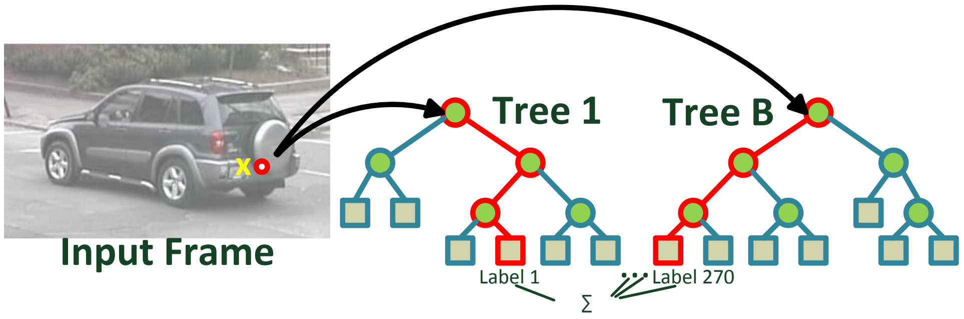


Figure 6.16: Pose estimation by classification using RFs and multiple features extracted at pixel $\mathbf{X} = (x, y)$.

The extraction of a feature at location $\mathbf{X}$ and finding the correct class by exploiting an ensemble of RFs is illustrated in Figure 6.16. Split nodes are denoted by ○, leaf nodes as □. To solve the first part of Equation 6.11, the matching score at time instance t between a model projection $\mathbf{p}_t$ and an input vehicle $\mathbf{s}_t$ is given by

$$Y(\mathbf{p}_t|\mathbf{s}_t) = P(l|G, \mathbf{\Theta}(\mathbf{X})). \tag{6.14}$$

6.2.3 Temporal Model Alignment

After having a matching score, the transition term $Y(\mathbf{p}_t, \mathbf{p}_{t-1}|\mathcal{S})$ must be determined. Ideally, the vehicle should move with constant speed which is established by using slow speed and applying a L_2 norm minimization. A feasible motion model is ensured by applying the Ackermann steering principle, [Scaramuzza et al., 2009]. The principle is defined by $\varphi = \frac{\gamma}{2}$, where φ and γ are the angles of the vehicle's inner and outer turning radius, respectively. As described in Section 6.1, forcing the vehicle to move with constant and feasible motion is therefore described by

$$Y(\mathbf{p}_t, \mathbf{p}_{t-1}|\mathcal{S}) = exp(-(\|\mathbf{p}_{t-1} - \mathbf{p}_t\|^2 + \lambda_2(\varphi - \frac{\gamma}{2}))), \tag{6.15}$$

where λ_2 assures equal weighting between the impact of constant and feasible motion, $\mathbf{p}$ is the car's pose. For solving Equation (6.11) and finding the best fitting sequence $\hat{\mathcal{R}}$ for a given $\mathcal{S}$, it is necessary to determine the Maximum A Posteriori Probability (MAP) configuration through the graph and compute both Equation (6.14) for each of the model projections and Equation (6.15) for each edge, where an edge should be from each projection of frame at time $t-1$ to each one at time t. The final MAP of the MRF is then given by

$$\hat{\mathcal{R}} = \operatorname*{argmax}_{\mathcal{R}} P(\mathcal{R}|\mathcal{S}). \tag{6.16}$$

6.2.4 Experiments

The algorithm is tested on five sequences from [Leotta and Mundy, 2011] showing a variety of different vehicles from different viewpoints. They provide a resolution of 1280x720 pixels and a calibrated setup. A number of 32 RFs consisting of 200 trees are generated with an azimuth angle ranging from 0° - 360°, stepsize 5°, an elevation angle ranging from 0° to 30°, stepsize 10° and a distance ranging from 16-32 and 125-140 meters, stepsize one meter. A number of 9 different existing 3D models are used to train the pose estimator (see Figure 6.17) where the dimensions are taken from the manufacturers' specifications.

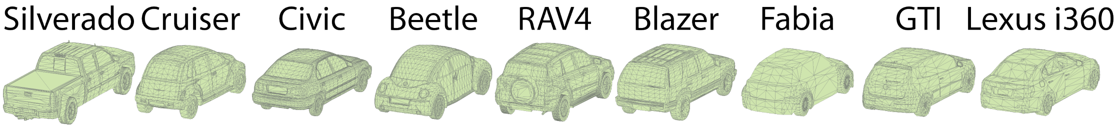


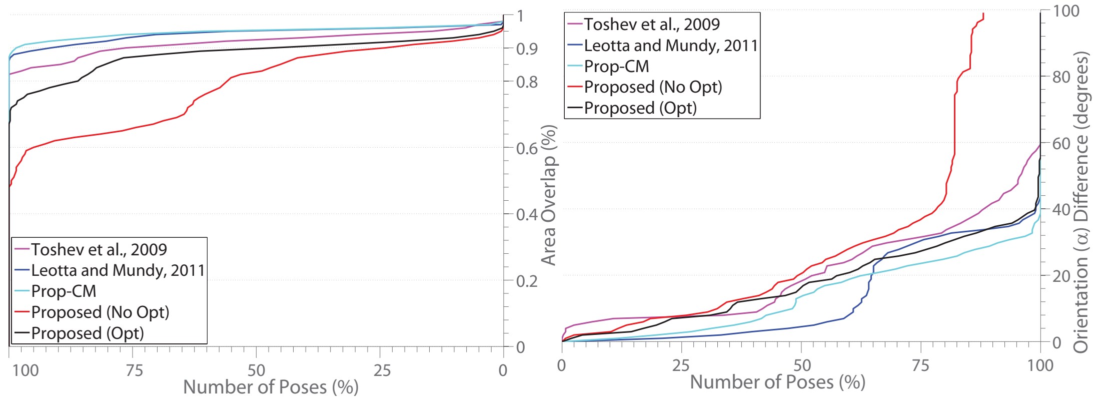
Figure 6.17: Models used for training the pose estimator.

6.2.4.1 Quantitative Experiments

The output for each step of the pipeline (RF only (No Opt) and applying MRF (Opt)) is shown and compared to Toshev's [Toshev et al., 2009] and Leotta's [Leotta and Mundy, 2011] results as well as to the approach described in Section 6.1, which in the following is referred to as the chamfer matching approach (Prop-CM). For generating ground truth data, the 2D area of the vehicle is manually segmented as foreground for each frame. Since the approach for pose estimation using 3D models of [Toshev et al., 2009] is not publicly available, it is being re-implemented. For Toshev's method the manual foreground segmentation is used as input data which prevents incorrect classifications due to a bad segmentation. To assure fairness in the comparison, the method is re-implemented such that comparable performance on similar videos is obtained. The dataset of [Toshev et al., 2009] is generated with the same parameters as the proposed one. Leotta's approach [Leotta and Mundy, 2011] is publicly available.

First, the overlap between the ground truth region and the projected 3D model is compared. As this metric is used for the evaluation of the detected pose, it is only valid when the correct type of the vehicle is projected onto the image plane. When the vehicle's pose is known and a correct 3D model is available, its type can be determined by rendering all vehicle types with the determined pose. The best matching vehicle type is obtained by using FDCM which is an edge-based matching method and known to be robust against intra-class variations [Liu et al., 2010c], [Hödlmoser et al., 2012]. The same models are used for determining the type and for training the pose estimator. In the experiments, a correct vehicle type for all sequences is obtained when using the proposed method, [Toshev et al., 2009] misclassified the vehicle shown in sequence 1.

Figure 6.18a shows the overlap rate for the proposed implementation (No Opt, Opt) as well as for Toshev's, Leotta's and Prop-CM. As can be seen, the described method obtains similar results compared to state-of-the-art algorithms. Toshev's approach uses all scales of a discretely rendered model. This can lead to misclassifications, where the incorrect, discrete scale of an incorrect model may fit better than the next best discrete scale of the correct model. As can be seen in Figure 6.18a, this does not directly influence the overlap between ground truth and detected model. Therefore the offset of the vehicle's centroid and its orientation α on the ground plane



(a) (b)

Figure 6.18: (a) Area overlap between ground truth and results from all methods over all sequences for correctly classified vehicles. (b) Difference of orientation α over all sequences between all methods and ground truth data.

Location Distance on Ground Plane to Ground Truth (cm)

	Proposed(Opt)	Proposed(No Opt)	Toshev et al., 2009	Leotta and Mundy, 2011	Prop-CM
μ	30.15	49.12	42.94	29.85	25.38
σ	26.85	38.52	33.21	16.36	11.98
max	202.67	332.24	288.45	120.17	66.29

Orientation Distance on Ground Plane to Ground Truth (degrees)

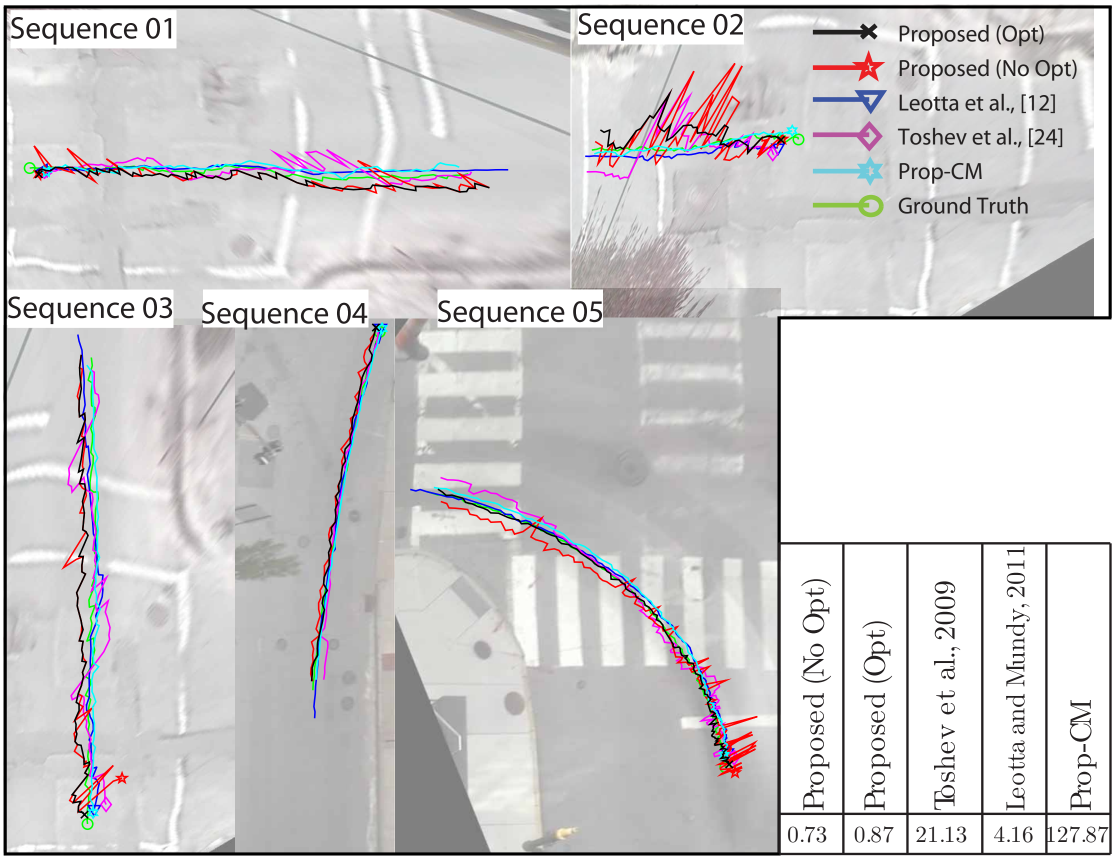
	Proposed(Opt)	Proposed(No Opt)	Toshev et al., 2009	Leotta and Mundy, 2011	Prop-CM
μ	17.60	28.72	20.96	13.20	13.06
σ	11.90	38.82	14.58	14.49	11.05
max	55.04	120.17	58.95	43.30	37.68

Table 6.2: Offsets of the vehicle's center (top) and orientation α (bottom) on the ground plane compared to ground truth .

are compared to the ground truth data (see Table 6.2). The proposed method clearly outperforms Toshev's implementation and gets similar results compared to Leotta's implementation both in terms of mean location and orientation error. Toshev's method is worse since it does not incorporate a ground plane estimation but only classifies each single frame based on the area overlap between training data and the input frame.

Figure 6.18b shows the percentage of poses being below a certain difference between the calculated orientation α and the ground truth data. The proposed, optimized pose estimator obtains comparable results to all other methods. More than 50% of all poses yield an error smaller than $15°$.

Figure 6.19a shows the trajectory of the vehicle's centroid over a whole sequence for each



(a) (b)

Proposed (No Opt)	Proposed (Opt)	Toshev et al., 2009	Leotta and Mundy, 2011	Prop-CM
0.73	0.87	21.13	4.16	127.87

Figure 6.19: (a) Tracks of the car's centroid on a warped top-view image for all sequences. Comparison between the proposed approach (No Opt, Opt), as well as [Toshev et al., 2009], [Leotta and Mundy, 2011], the method described in Section 6.1, Prop-CM, and ground truth. (b) Mean processing time for one frame in seconds.

method for all sequences from left to right. The vehicle's starting position is denoted by $\times$, $\star$, $\bigtriangledown$, $\diamondsuit$, $\hexagon$, $\circ$ for Opt, No Opt, Leotta, Toshev, Prop-CM and ground truth, respectively. As can be seen, there are more jumps in space between consecutive frames using Toshev's method than using the one described in this section since the proposed algorithm assumes the 3D model to be located on the ground plane. The main advantage of the proposed approach is that comparable results to state-of-the-art methods are reached in much less computational time. The mean processing time for one frame can be seen in Figure 6.19b.

As can be seen, the algorithm described in this section provides a faster runtime than the approach presented in Section 6.1 since there is no need to evaluate all models in order to obtain a matching score between 3D model and input video. For this comparison, only the time needed for pose estimation for each method is used, where detecting and tracking the vehicle

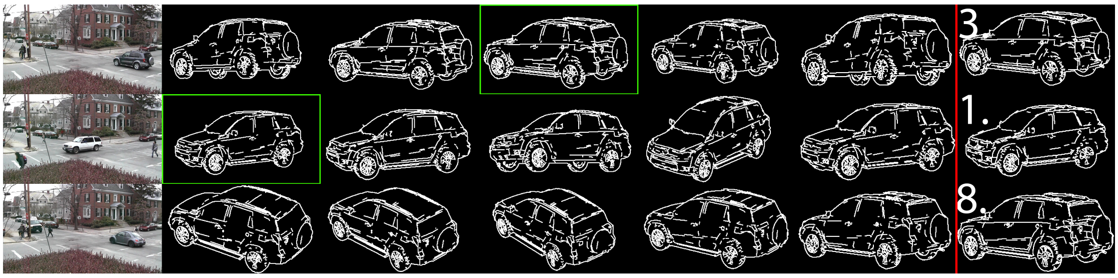


Figure 6.20: Best five poses for three random frames ranked from left to right. The rightmost image shows the pose and its rank chosen from the MRF. Note that the proposed method uses a random model type for visualization.

are excluded. The test is performed on an Intel i5, 2.4 GHz and 8GB RAM, the method outlined in this section, the algorithm described in Section 6.1 and Toshev's are implemented in Matlab, Leotta's in C++.

6.2.4.2 Qualitative Experiments

Figure 6.20 shows from left to right the best matching five poses for random frames. The rightmost image shows the best matching pose and its corresponding rank taken from the MRF. As can be seen, the best pose must not be ranked first to get a smooth result but all highly ranked poses are similar to the correct one. Figure 6.21 and Figure 6.22 show qualitative example results using the proposed approach. One sequence is represented by five rows where each column corresponds to the method outlined in this section, [Toshev et al., 2009], [Leotta and Mundy, 2011] and Prop-CM. The upper left image of each sequence presents its first frame and the proposed, optimized projected track. The following frames of each column provide the refined pose in combination with the vehicle type estimated by FDCM projected onto the image plane. For viewing purposes, the region showing the vehicle is cropped. As can be seen, comparable results are obtained for all evaluated methods. The last five rows in Figure 6.22 show incorrect model types projected onto the image plane.

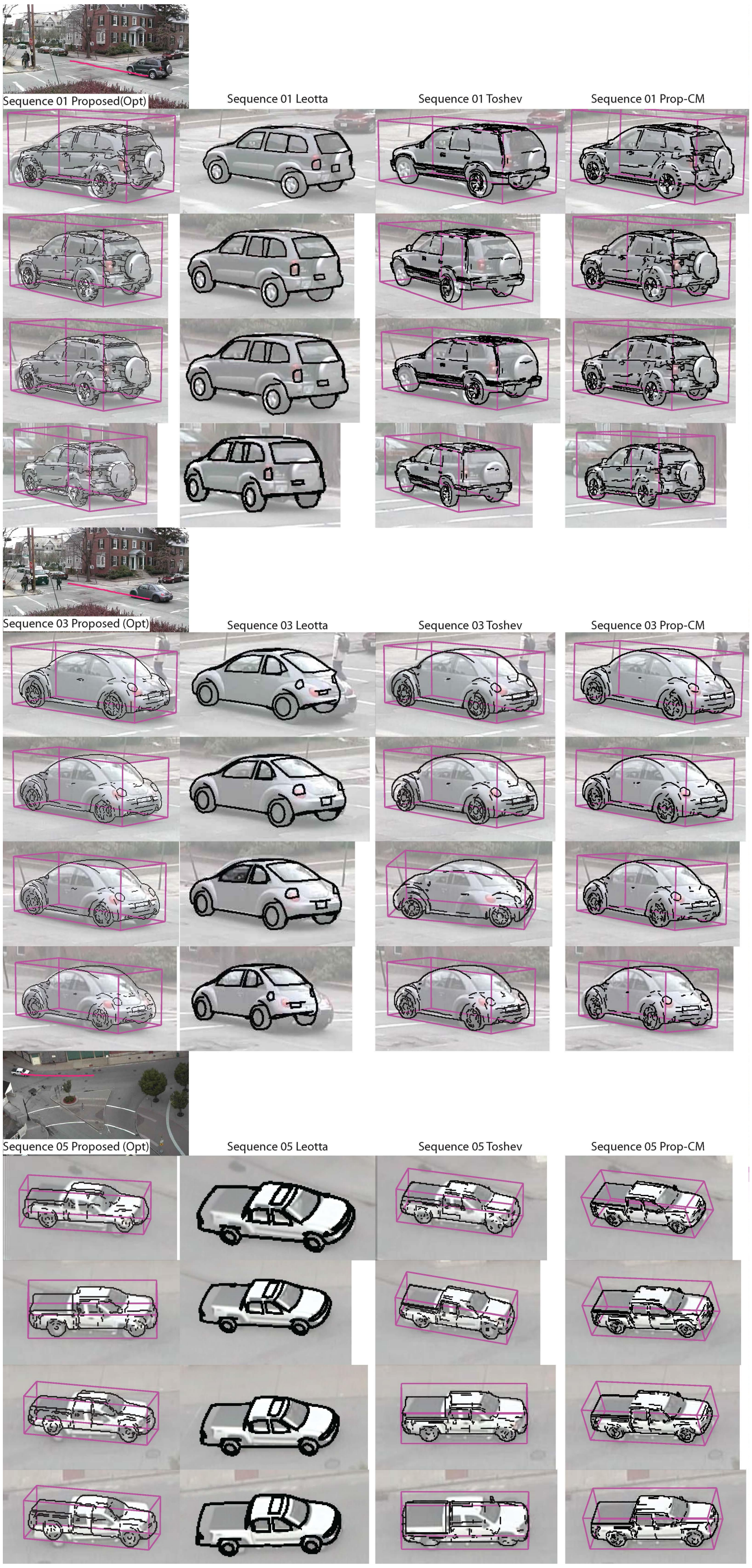


Figure 6.21: Classification and pose estimation results. One sequence is represented by five rows where each column shows results using different methods. The upper leftmost image of each sequence shows its first frame and the novel optimized projected track.

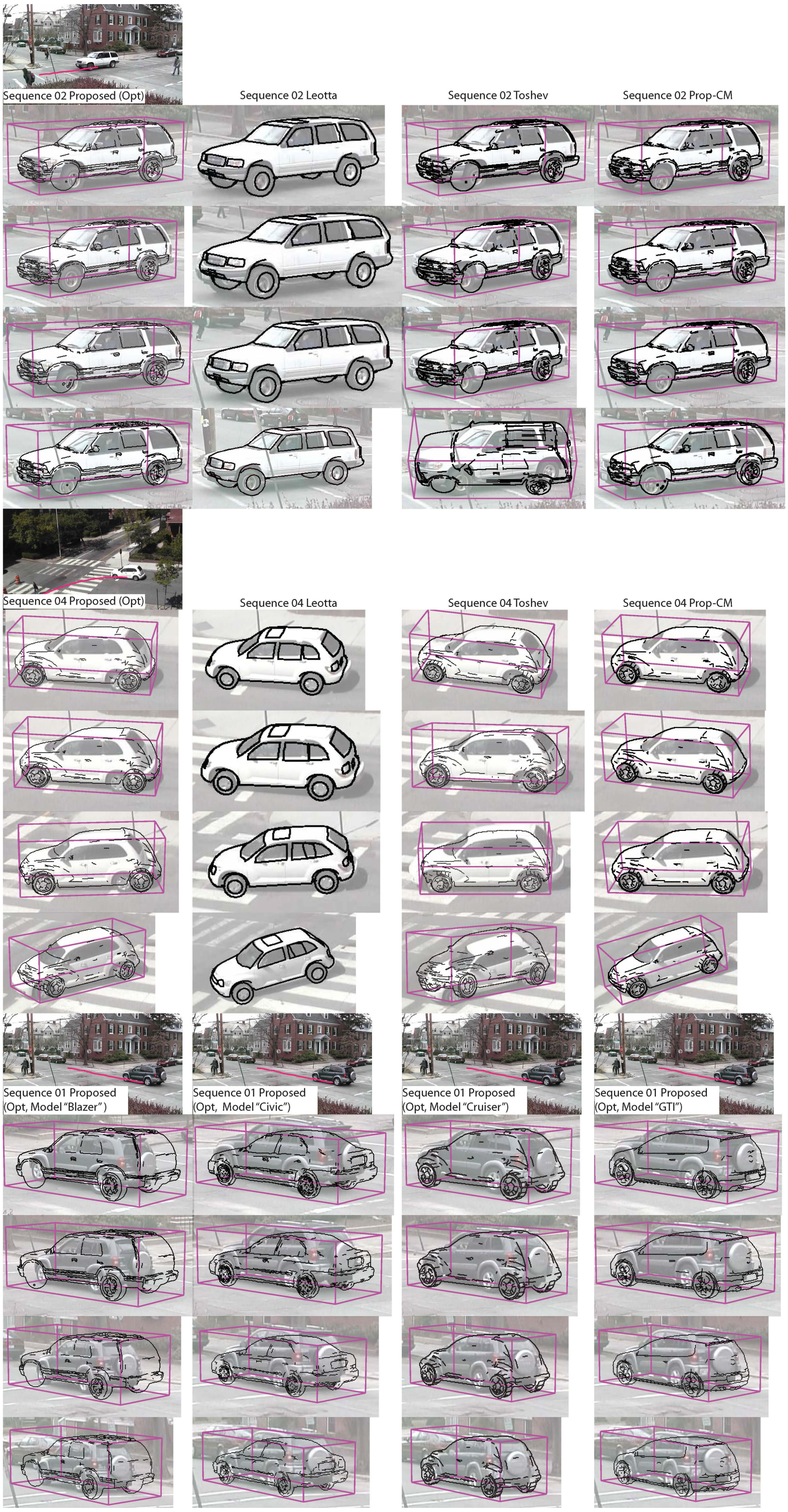

Sequence 02 Proposed (Opt)
Sequence 02 Leotta
Sequence 02 Toshev
Sequence 02 Prop-CM
Sequence 04 Proposed (Opt)
Sequence 04 Leotta
Sequence 04 Toshev
Sequence 04 Prop-CM
Sequence 01 Proposed (Opt, Model "Blazer")
Sequence 01 Proposed (Opt, Model "Civic")
Sequence 01 Proposed (Opt, Model "Cruiser")
Sequence 01 Proposed (Opt, Model "GTI")

Figure 6.22: Classification and pose estimation results. One sequence is represented by five rows where each column shows results using different methods. The upper leftmost image of each sequence shows its first frame and the proposed optimized projected track. The last five rows show a variety of incorrect model projected onto the image plane.

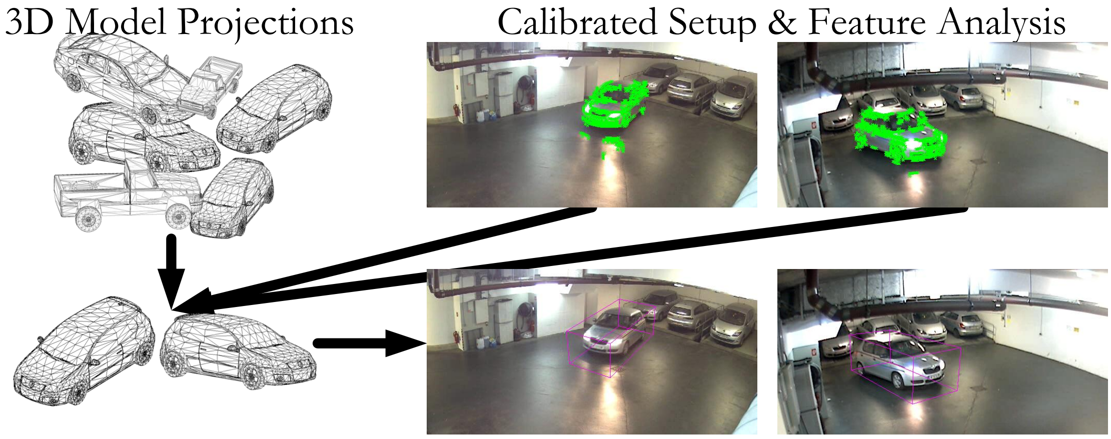


Figure 6.23: Classification and pose recovery by using projections of 3D models (left) and by analyzing the tracked features (top right). The Results (bottom right) are enhanced by aggregating information simultaneously over multiple synchronized and calibrated cameras.

6.3 Vehicle Classification and Pose Estimation from Two Views

Object classification and pose recovery are used in a wide range of applications (e.g. surveillance, content based image retrieval, and robotics). Detecting or classifying an object within a generic scene is a challenging problem in the area of computer vision because the same object can occur in front of different backgrounds, under varying lightning conditions and in different poses. These problems are known as intraclass variations. Traditional object classification algorithms extract features from input images, where objects of the desired class are visible to learn a classifier [Thomas et al., 2006]. This can be established by taking advantage of the huge amount of data available on the Internet. By using thousands of images, intra-class variations will be included in the classifier but only discrete viewpoints can be learned. The tremendous increase of computing power enables the generation of all different poses of an object from existing 3D models which eliminates the problem of intraclass variations.

This section therefore present a novel method for traffic object classification and pose estimation based on existing 3D models and shows how to efficiently combine multiple cues from different viewpoints in order to minimize the pose estimation and classification error. As surveillance scenarios may consist of cars and pedestrians, in a first step, the cameras are calibrated using pedestrians walking through the observed scene, as proposed in [Hödlmoser and Kampel, 2010]. Since the camera parameters are then known, the 3D models from an existing dataset can be projected from 3D space onto 2D image planes in the training stage. Features are extracted from the video stream to be classified. These features are then used to find the best fitting model in its best fitting projection within the training set. The calibrated setup consists of two cameras which allows to improve the classification and pose estimation results by assuring spatial consistency. Figure 6.23 shows the determination of the correct pose and class of a detected object in a video stream.

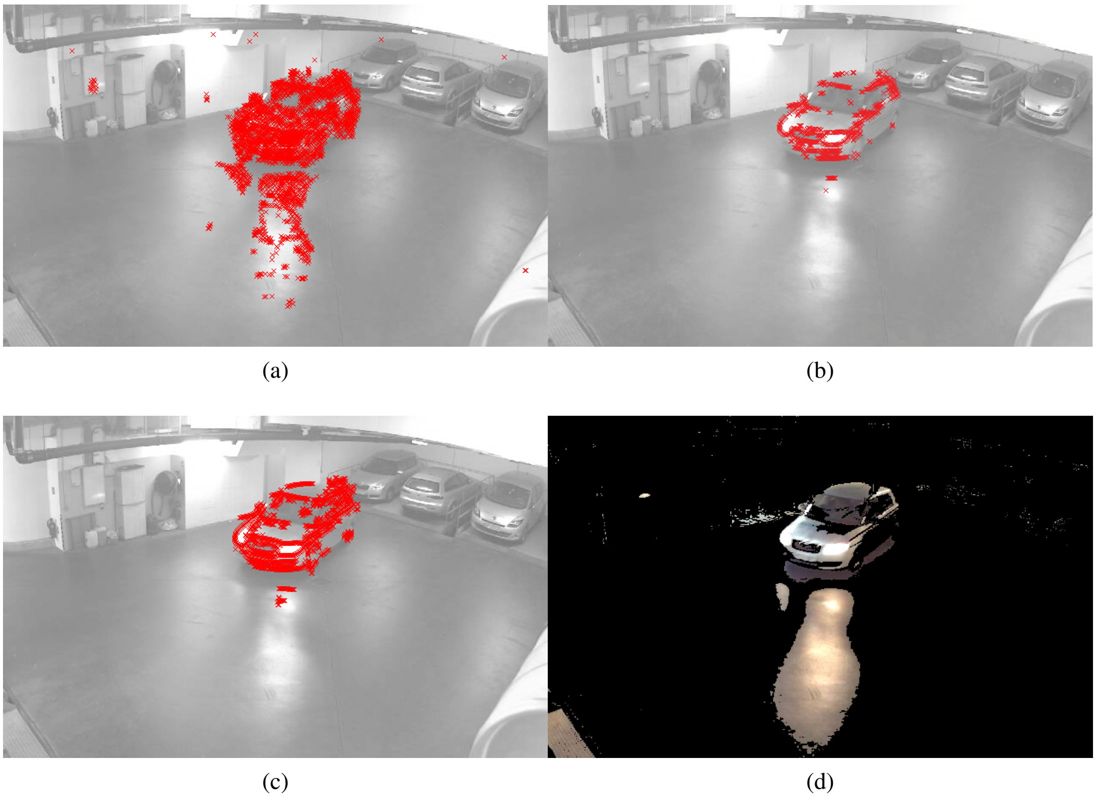

(a)
(b)
(c)
(d)

Figure 6.24: (a) Sample frame and Harris features, (b) features kept after applying Sobel filter, (c) final foreground features using Canny edge image. As can be seen in (d), background subtraction cannot handle highlights and shadows.

The contribution of this section is twofold. First, a novel framework is presented which is based on the approach of [Toshev et al., 2009] and is used for object classification and pose estimation using synthetic 3D models. After the determination of a vehicle's area in the image, all projections of the 3D models are compared to this blob in order to find the best fitting one. Second, this approach also considers a setup consisting of two cameras and shows how aggregating information from both cameras simultaneously enhances the classification results. In contrast to [Gill and Levine, 2009], where spatial consistency is meant to be consistency between detected parts of an object, in this section the term is used for consistency of 2D projections of an object between different viewpoints.

6.3.1 Moving Object Detection

The first part within the framework is related to the determination of moving objects within the scene. Since traffic scenarios consist of objects providing shadows and highlights due to poor lighting conditions (e.g. in a garage or at night), it is not possible to determine the moving

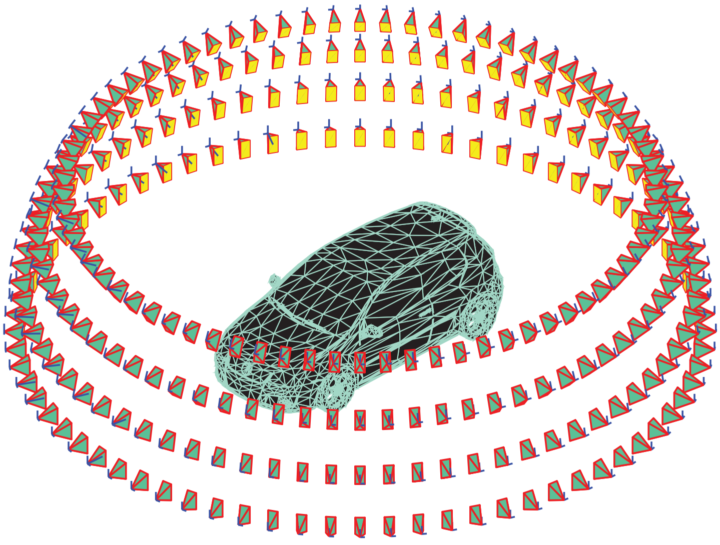


Figure 6.25: Training stage camera positions at a distance of 9 meters from the origin and discrete azimuth and elevation values.

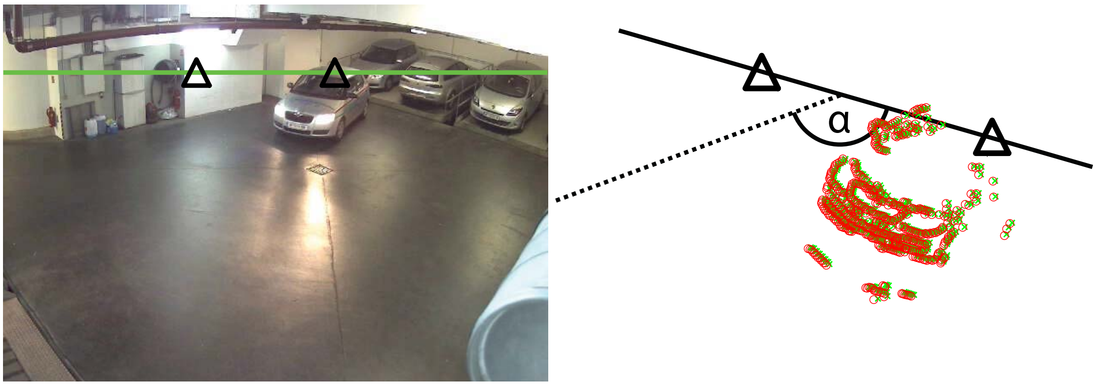
object correctly by simply subtracting the background or learning the background using a GMM (see Figure 6.24). It is therefore proposed to take the structure of the detected motion segments into account to get rid of highlight and shadow regions. Harris corner features [Harris and Stephens, 1988] are extracted and tracked over time by using a Kanade-Lucas-Tomasi feature tracker. Features, which provide the same oriented motion over time, are kept and new features are added when they occur. Since a moving object provides more structure than shadow or highlight areas, an edge image of the current frame is obtained by using a Sobel filter. Areas with less structural features are removed and others are kept by only taking those Harris features located within a certain distance near an edge. In practice, this distance is set to 1 pixel. As the goal is to increase the number of features to be tracked and consequently get a better pose estimation result, another edge image is calculated by exploiting the Canny edge filter which is more sensitive to structural changes than the Sobel filter. All points on an edge located within a distance of 1 pixel to a feature point are taken as input for the classification and pose estimation in the next step. Figure 6.24 shows (a) a sample frame providing all moving Harris corner features, (b) the features kept after applying the Sobel filter and (c) the resulting features taken for classification and pose estimation located on the edges of the output using a Canny edge detector. The foreground obtained by using background subtraction only is shown in (d).

6.3.2 Training, Classification and Pose Estimation

Increasing computational power [Hennessy and Patterson, 2011] allows the determination of all possible camera's view sphere 2D projections of a 3D model within a certain amount of time.

In practice, model projections are calculated from discrete camera positions by positioning the object at the coordinate origin and orbiting the camera around the object, as proposed in [Liebelt et al., 2008], in order to reduce the complexity and computational effort. The orbit of the camera is described by three parameters, namely the two angles azimuth, elevation and the distance from the origin. Figure 6.25 shows a random 3D model and all camera positions using a distance of 9 meters and discrete values for azimuth and elevation.

Classification and pose estimation of moving objects is done by finding the best fitting 2D projection for a given set of detected point features. To minimize the computational complexity, all tracked feature points between two consecutive frames are projected on the ground plane. As can be seen in Figure 6.26, (a) two random points having the same vertical value (denoted as $\triangle$) are projected on (b) the ground plane. The line going through both of them provides an azimuth angle of 0° which equals to an azimuth angle of 0° in the training stage. By calculating the orientation difference between the tracked points and the line going through these two points, the azimuth angle used for training, denoted as α, can be approximated. This angle can then be directly used for comparison to the projections of the 3D models from the training data set. Figure 6.26 shows the determination of the azimuth angle α. The solid line represents an azimuth angle of zero, the dashed line represents the mean orientation of all tracked feature points, which are represented by $\times$ and $\circ$ for two subsequent frames.



(a) (b)

Figure 6.26: Determination of the difference angle α between azimuth=0 and the vehicle orientation found from the 2D tracked feature points (a) in the 2D image and (b) on the 3D ground plane.

To narrow down the search space of valid projections, those projections providing an azimuth angle greater than $\alpha \pm 30°$ are excluded from further calculations. The center of mass of all detected feature points is then calculated and the center of mass of all 2D projections is placed on the same position. Having the feature locations $\mathbf{f}$ and n model projections $\mathbf{A}$, the index p of

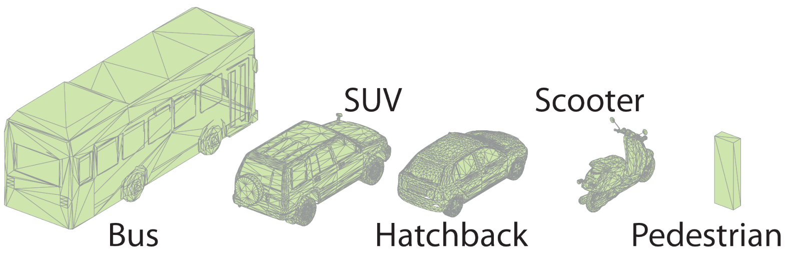


Figure 6.27: Test set used for the experiments.

the best fitting one is gathered by

$$p = \underset{i=1\ldots n}{\operatorname{argmax}} \left(\frac{|\mathbf{f} \cup \mathbf{A}_i|}{|\mathbf{f} \cap \mathbf{A}_i|} \right). \tag{6.17}$$

Apart from determining the most likely projections for each frame and each camera separately using Equation 6.17, these results can be enhanced by considering spatial consistency over all cameras. As the relative positions of all cameras are known, a projection of one camera view can be projected into all other camera views. This approach finds the global best class for the moving object and varying classifications can be excluded. In practice the first most likely projections for each camera are taken and the corresponding projections are calculated for all other views in order to speed up the matching procedure. The final result is obtained by finding the maximum correlation between model projection and input frame over all cameras by calculating the index p of the best fitting pose by

$$p = \underset{i=1\ldots n}{\operatorname{argmax}} \sum_{j=1}^{k} \left(\frac{|\mathbf{f}_j \cup \mathbf{A}_{(i,j)}|}{|\mathbf{f}_j \cap \mathbf{A}_{(i,j)}|} \right), \tag{6.18}$$

where k which is set to 2 in the experiments, denotes the number of cameras and n is set to 3 for evaluation purposes in order to speed up the matching procedure.

6.3.3 Experiments

To show the practicability of the proposed framework, it is tested on real world data where two cameras which have an overlapping view and a wide baseline, are mounted in a garage scenario. To calibrate the setup it is decided to use the self-calibration method proposed in [Hödlmoser and Kampel, 2010]. For all the experiments the same training set in combination with the calibration parameters is used. The set consists of five 3D models (4 vehicles and a pedestrian) where an azimuth angle covering a range between 0 and 360 degrees having a stepsize of 5, elevation between 0 and 30 degrees with a stepsize of 10 and a distance ranging from 9 to 17 meters with a stepsize of 1 meter is used. This results in 12960 different projections. The dimensions of the vehicles are taken from manufacturers' information. The test set is shown in Figure 6.27. All CAD models, except the pedestrian which is constructed by hand, are downloaded from the Internet.

	Hatchback	Bus	SUV	Scooter	Pedestrian	No Match
Single Viewpoints	73.2%	4.4%	8.0%	1.6%	0.0%	12.7%
Two Viewpoints	78.8%	1.3%	7.3%	0.0%	0.0%	12.7%

Table 6.3: Quantitative evaluation. Percentage for frames classified as a certain vehicle type. As ground truth data, each frame should be labeled as hatchback.

6.3.4 Quantitative Experiments

A number of four synchronized videos are taken from the same stereo camera setup where each video is between 50 and 120 frames showing the same hatchback. From all the moving objects which should be detected as hatchback, 4.4% and 1.3% are classified as bus, 8.0% and 7.3% are classified as SUV, 73.2% and 78.8% are classified as hatchback, 1.6% and 0% are classified as scooter, 0% and 0% are classified as pedestrian, without and with using spatial consistency, respectively. In both cases, for 12.7% of all frames no matching projection can be found due to fewer features. The results are summarized in Table 6.3 which shows the percentage of frames labeled as a certain vehicle type.

6.3.5 Qualitative Experiments

Figures 6.28 and 6.29 show input frames in columns 1 and 3 in combination with its best fitting projections calculated by the proposed algorithm in the columns besides them. Each two rows represent synchronized cameras having an overlapping field of view. Column 2 of Figure 6.28 and Figure 6.29 shows results by analyzing each camera separately, column 4 shows results by considering calibration data and therefore spatial consistency. As can be seen, two problems can be tackled using redundant data from both cameras, namely obtaining different classes in synchronized frames from different camera views and rough or incorrect pose estimations. If no correct projection is within the first three most likely ones after analyzing each view separately, the calibration data cannot be used to correct the results.

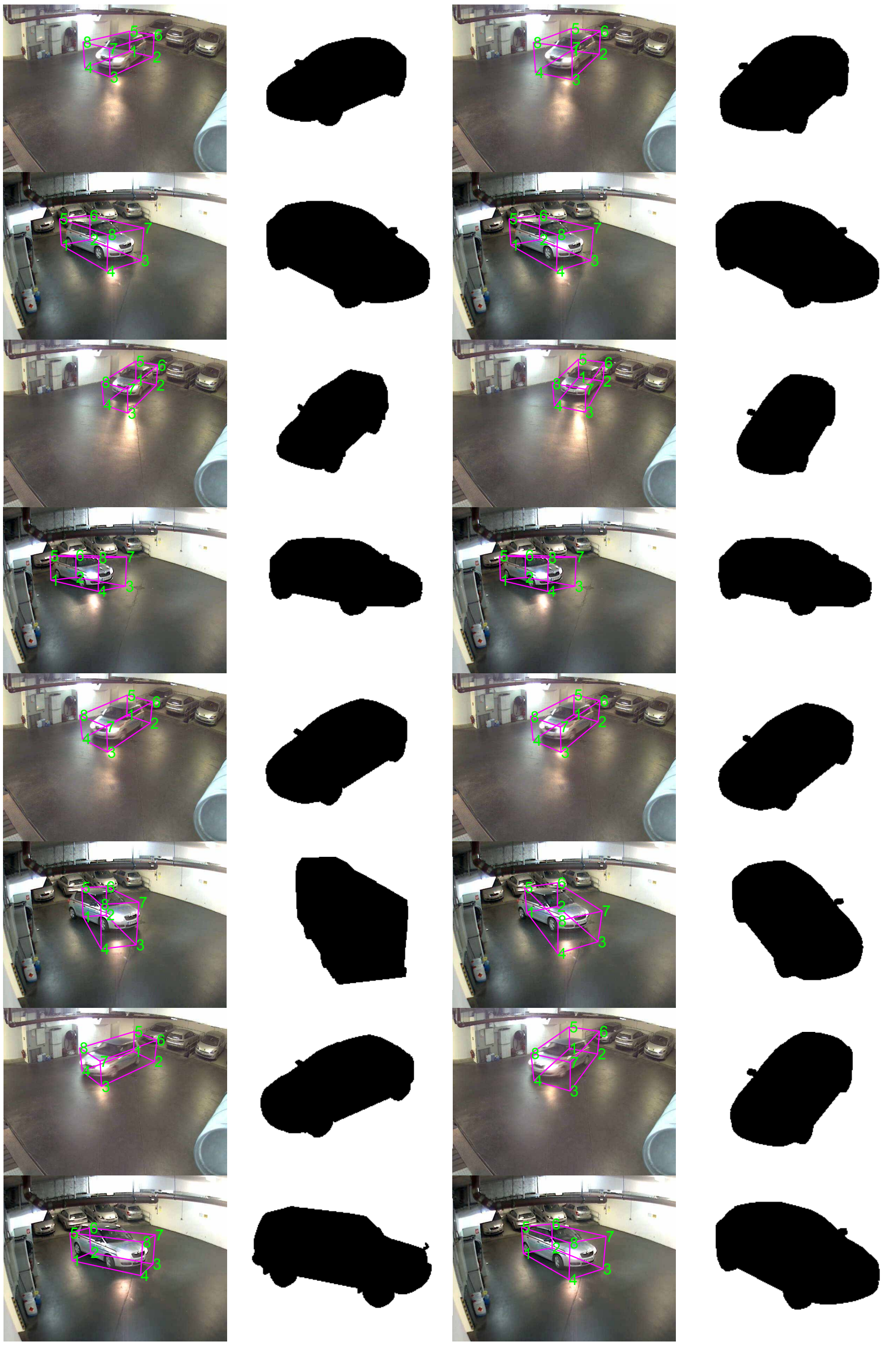


Figure 6.28: Input frames are shown in columns 1 and 3 in combination with its best fitting projections besides them. Each two rows represent synchronized cameras having an overlapping field of view. Column 2 shows results from analyzing each camera separately, column 4 shows results when considering calibration data. Note that the projections are enlarged for viewing purposes.

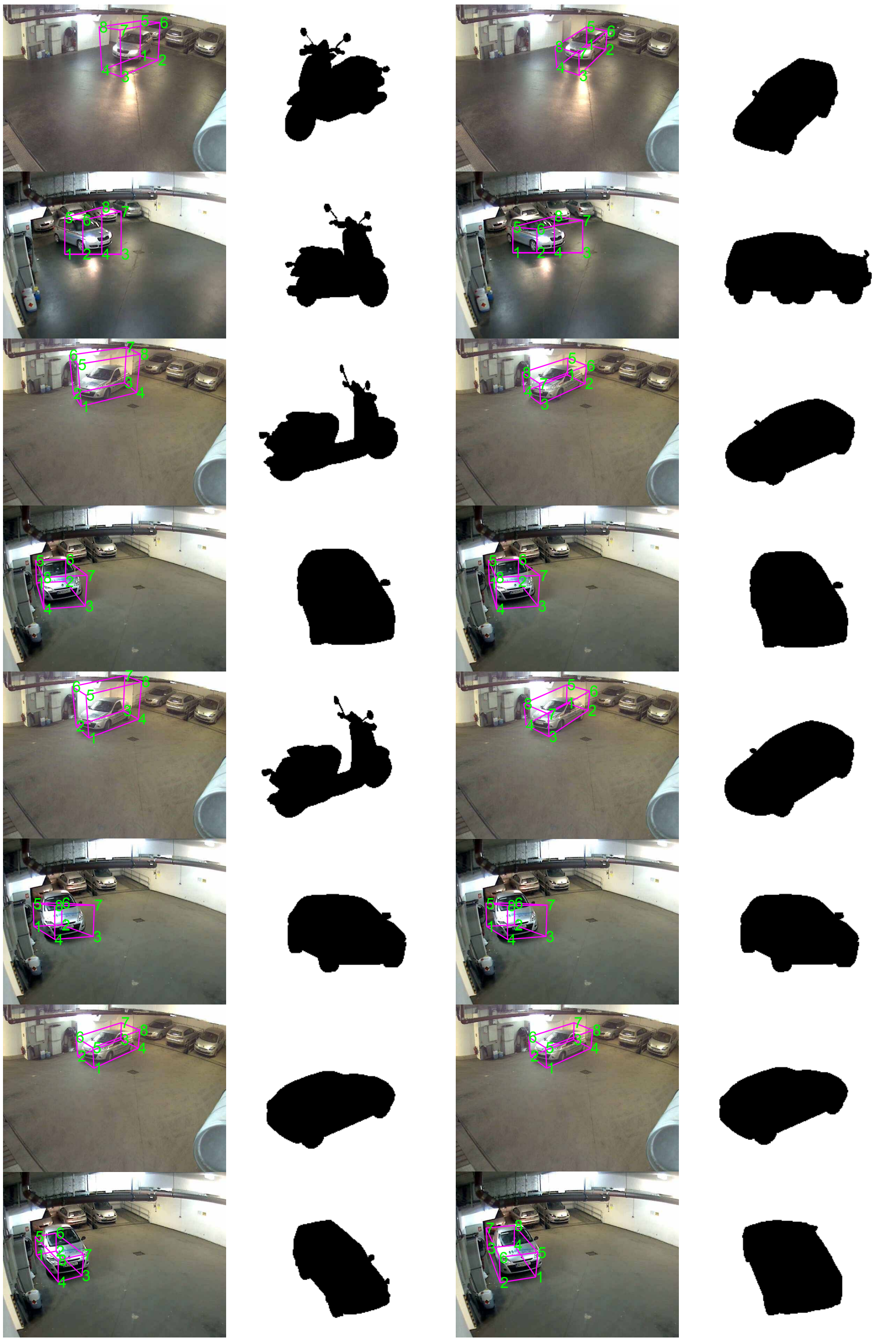


Figure 6.29: Input frames are shown in columns 1 and 3 in combination with its best fitting projections besides them. Each two rows represent synchronized cameras having an overlapping field of view. Column 2 shows results from analyzing each camera separately, column 4 shows results when considering calibration data. Note that the projections are enlarged for viewing purposes.

6.4 Summary

As 3D reasoning is a broad field in computer vision, this section presented a specific task within that area, namely object classification and pose estimation. Vehicles are used since they are the most important objects besides pedestrians in visual surveillance applications.

First, this chapter describes the task of vehicle classification and pose estimation in videos as a continuous optimization problem which can practically be solved by discrete-continuous optimization. Good starting points which are essential for the ensuing continuous optimization problem, are obtained by a discrete optimization reaching a global optimum on the given discrete set. In both the quantitative and qualitative results it is shown that the proposed method clearly outperforms state-of-the-art methods from both approaches, namely those using deformable models and those using models with known dimensions. It could also be seen that the proposed method handles two drawbacks of purely discrete solutions, namely huge pose estimation errors and preferring an incorrect model at an incorrect scale over the correct one.

Second, a framework for estimating the pose of vehicles in videos by exploiting existing 3D models having known dimensions in combination with a random forest is presented. The random forest classifier is trained on a synthetic set of existing 3D models, rendered from discrete viewpoints. Different to existing approaches, a generic pose estimator is trained on a variety of 3D models which does therefore not rely on having a corresponding 3D model in the training set to the vehicle shown in the input frame. A novel edge based feature is introduced to match 2D projections to 2D input frames. These features give the opportunity to describe synthetically rendered objects and compare them to their real world counterparts. An MRF ensures a feasible vehicle motion between consecutive frames. As can be seen from the experiments, similar results compared to state-of-the-art methods are obtained but the proposed method dramatically outperformed them in terms of processing time.

Third, it is shown that combining information from different improves the results of estimating the pose and type of an object. A novel method for traffic object classification and pose estimation is presented based on finding the best fitting projection of existing 3D models to moving 2D object projections in the scene. Additionally, a setup consisting of two cameras is considered and it is shown how aggregating information simultaneously from multiple cameras raises the classification performance compared to results obtained by analyzing each camera separately.

CHAPTER 7

Conclusion and Future Work

Reasoning in 3D space is necessary for humans to tackle the tasks of detecting occlusions and occluded objects, extracting metric information from 2D images, navigating through everyday life and many others. The holy grail of computer vision is to obtain even more reliable and consistent 3D information out of 2D images than humans are able to achieve. In this thesis, the task of 3D scene understanding using computer vision is split into three parts, namely camera calibration, 3D reconstruction and 3D reasoning.

When capturing a 3D scene onto a 2D image plane, there is an endless number of different influences which can change the representation of the scene (e.g. lighting conditions, different objects involved in the scene, different tasks performed between these objects,...). As a result, an algorithm performing a computer vision task is in practice tailored to handle a certain amount of influences and can therefore be applied on a specific image region, or with a specific environmental setting in order to obtain the maximum accuracy.

The main contribution of this thesis lies in showing that by combining different methods and algorithms related to 3D scene understanding, which in this work is referred to as ***redundancy***, the final outcome of a computer vision pipeline can be improved. Redundant information is therefore exploited in order to overcome the specificity of a single algorithm which furthermore increases the robustness of the framework and finally improves the task of 3D understanding in computer vision applications and methods.

As the first part of a 3D scene understanding framework is camera calibration, two methods for calibrating static surveillance cameras from traffic participants and road markings are presented. The first proposed method performs calibration of a camera network from a pedestrian. Both intrinsic and extrinsic parameters are extracted by observing the person over time and extracting foot and head points. Three vanishing points are extracted from these points and the intrinsic parameters are determined by exploiting the Image of the Absolute Conic (IAC). The relative rotation and translation between two cameras in a network are then also determined from foot and head points. The second proposed method describes a calibration procedure for a single static traffic surveillance camera from pedestrians and a zebra-crossing. Intrinsic parameters are obtained from vanishing points, extrinsic parameters are determined by assuming a certain width of the zebra-crossing pattern. When a camera is calibrated using static objects, it is known that the flexibility is limited but the accuracy is high. On the other hand, when a camera is calibrated using dynamic objects, the method is more flexible but the results are noisier. It is shown in the experiments carried out with a variety of real and synthetic scenes that by combining pedestrians (dynamic objects) and zebra-crossings (static objects) the advantages of both objects can be combined in order to obtain higher calibration accuracy.

The second part is referred to as 3D reconstruction. Conventional dense 3D reconstruction methods obtain a sparse 3D model by extracting and matching corresponding features and by densifying the model by searching for corresponding features in the spatial neighborhood of good matches. Man-made indoor environments suffer from flat and textureless surfaces, where these conventional, feature-based 3D reconstruction pipelines fail to estimate the 3D scene layout due to wrong or missing matches. This thesis therefore proposes a 3D reconstruction pipeline which combines conventional approaches with 3D reasoning coming from semantic information. First, a sparse 3D model is generated from multiple images in a sequence by using Structure from Motion (SfM). Second, each image is processed separately and split in semantic meaningful segments using existing Superpixel segmentation methods. Each segment is assumed to be described by a planar patch and a surface normal is estimated for each segment. Multiple methods are used in order to exploit redundancy which means that multiple orientation possibilities are available for each pixel. The final surface normal orientation is then gathered by finding the most probable solution for each pixel out of all possibilities by solving a pixel-wise Markov Random Field (MRF). The dense 3D model is then obtained by combining the surface orientation information with the sparse 3D point cloud. The experiments carried out show that the combination of semantic and geometric 3D reasoning provides a much denser reconstruction and a more complete model compared to state-of-the-art dense 3D reconstruction pipelines. It is also demonstrated that using multiple segmentation methods increases the accuracy of the 3D surface normal estimation compared to using a single one.

The final part of a 3D scene understanding pipeline is known as 3D reasoning. There is a variety of different applications and tasks covered by 3D reasoning (e.g. human actions and interactions among objects, extraction of metric information, occlusion estimation,...). Nevertheless, 3D pose estimation and 3D tracking form the basis for many of these tasks. As vehicles are the most important objects to be analyzed in surveillance applications besides pedestrians, this work presents a method for 3D vehicle pose estimation, classification and tracking. Two different approaches are outlined. The first framework obtains the vehicle's class and its pose by exploiting existing 3D models and measuring the similarity between discretely rendered 3D model projections and an input frame using Fast Directional Chamfer Matching (FDCM). The global best result for a whole video sequence is then obtained by solving an MRF. The unary term is defined as the matching score, the binary term is defined as the distance and orientation offset between 3D model positions of subsequent frames. The conducted experiments validate that having multiple solutions for each frame and finding a global best result afterwards outperform state-of-the-art 3D vehicle tracking and pose estimation approaches. The second proposed method tries to overcome the drawback of the first approach which is the computational complexity due to the fact that the matching similarity for multiple projections and for each 3D model in a dataset must be determined in order to find the best matching one. This is solved by creating a pose estimation classifier from multiple 3D models using a Random Forest (RF). Projections from the same viewpoint but from a different model type are combined to form a single class in a training stage. The matching similarity between a trained class and an input frame is then determined in the matching stage. Similarly to the first approach, the global best result is generated by using an MRF. Experiments show that similar accuracy but much faster runtimes can be achieved compared to state-of-the-art methods. The last part of this work outlines that having information coming from different viewpoints improves the accuracy of estimating the 3D pose and type of the vehicle. Therefore, a method for 2D vehicle classification and pose estimation and a solution for combining the results from multiple viewpoints are presented.

Although this thesis presented a well designed framework for 3D scene understanding, the single tasks of camera calibration, 3D reconstruction and 3D reasoning are not working in real time at the moment. Additionally to improving the algorithms' runtimes, future research objectives, which are described in the following, should be followed.

Camera Calibration: The current camera calibration procedures are limited in terms of the number of cameras included in a network. The first approach deals with calibrating a single camera, the second method is able to calibrate a network of cameras, where all of them observe the same scene. The future research direction in this field should therefore deal with this scalability problem in order to tackle the task of calibrating a couple of thousand non-overlapping cameras. This can for example be handled by accurately tracking traffic participants over time and by re-identifying them among different views.

3D Reconstruction: The presented 3D reconstruction procedure is done in two steps. The first step divides the image into vertical and horizontal regions, the second one splits the vertical parts into multiple surface normal orientations selected from a set of discrete orientations. In future, these two steps should be performed together. This can be established by replacing the pairwise MRF by an optimization framework processing a clique of three or more pixels which was also proposed by [Woodford et al., 2008]. Having a clique formed by three or more pixels enables penalizing the clique depending on the structure's planarity. By also introducing a semantic label in terms of a surface orientation to the cliques, the accuracy of the 3D reconstruction can be further improved due to solving both problems in parallel.

3D Reasoning: The proposed pipeline for 3D vehicle pose estimation and classification depends on a 2D detector [Felzenszwalb and Huttenlocher, 2004], serving as initialization for the matching procedure. The first approach presented in this thesis performs the matching between 2D image and the rendered 3D models by using FDCM which means that a matching score is generated for each projection. Therefore, only a certain number of vehicles can be processed due to practical time limitations. As is shown in the thesis, this problem can be handled by using an RF trained on multiple vehicle models. The drawback, which comes along with the runtime improvement, is a slight decrease in accuracy. Therefore, a combination of both approaches should be designed. When FDCM is working fast enough, first the initial 2D detection can be skipped and second a higher number of vehicles can be processed compared to the current implementation. Third, as FDCM is known to be robust against variations, the method of 3D pose estimation can be extended to perform action recognition using non-rigid objects (e.g. persons), where classes are not designed as different vehicle types anymore but as a variation of actions.

Bibliography

[Achanta et al., 2012] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Suesstrunk, S. (2012). SLIC Superpixels Compared to State-of-the-art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI).*

[Achanta et al., 2010] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Susstrunk, S. (2010). Slic superpixels. Technical Report 149300, EPFL.

[Agarwal et al., 2009] Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R. (2009). Building rome in a day. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 72–79.

[Alexandrov and Gorsky, 1991] Alexandrov, V. and Gorsky, N. (1991). *From humans to computers: cognition through visual perception*. World Scientific Publishing. World Scientific, Singapore.

[Amit and Geman, 1997] Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588.

[Arie-Nachimson and Basri, 2009] Arie-Nachimson, M. and Basri, R. (2009). Constructing implicit 3d shape models for pose estimation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1341–1348.

[Bao et al., 2012] Bao, S. Y.-Z., Bagra, M., Chao, Y.-W., and Savarese, S. (2012). Semantic structure from motion with points, regions, and objects. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2703–2710.

[Bao and Savarese, 2011] Bao, S. Y.-Z. and Savarese, S. (2011). Semantic structure from motion. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2025–2032.

[Bao et al., 2010] Bao, S. Y.-Z., Sun, M., and Savarese, S. (2010). Toward coherent object detection and scene layout understanding. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 65–72.

[Barinova et al., 2008] Barinova, O., Konushin, V., Yakubenko, A., Lee, K., Lim, H., and Konushin, A. (2008). Fast automatic single-view 3-d reconstruction of urban scenes. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 100–113.

[Barrow and Tenenbaum, 1980] Barrow, H. and Tenenbaum, J. (1980). Interpreting line drawings as three-dimensional surfaces. In *Proc. of Conference on Artificial Intelligence*, pages 11–14.

[Beardsley and Murray, 1992] Beardsley, P. and Murray, D. (1992). Camera calibration using vanishing points. In *Proc. of British Machine Vision Conference (BMVC)*, pages 417–425, Leeds, UK.

[Besag, 1986] Besag, J. (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B (Methodological)*, 48(3):259–302.

[Boykov and Jolly, 2001] Boykov, Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 105–112.

[Brandenberger, 1948] Brandenberger, A. (1948). *Fehlertheorie der inneren Orientierung von Steilaufnahmen*. PhD thesis, ETH Zurich.

[Breiman, 2001] Breiman, L. (2001). Random forests. In *Machine Learning*, pages 5–32.

[Brostow et al., 2008] Brostow, G., Shotton, J., Fauqueur, J., and Cipolla, R. (2008). Segmentation and recognition using structure from motion point clouds. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 44–57.

[Brown, 1956] Brown, D. (1956). The simultaneous determination of the orientation and lens distortion of a photogrammetric camera. Technical Report RCAMTP Data Reduction Technical Report W 33 AFMTC-TR 56-20, Patrick Air Force Base.

[Brown, 1971] Brown, D. (1971). Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866.

[Buch et al., 2009] Buch, N., Orwell, J., and Velastin, S. (2009). 3d extended histogram of oriented gradients (3dhog) for classification of road users in urban scenes. In *Proc. of British Machine Vision Conference (BMVC)*.

[Canny, 1986] Canny, J. (1986). A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 8(6):679–698.

[Chen et al., 2007] Chen, T., Del Bimbo, A., Pernici, F., and Serra, G. (2007). Accurate self-calibration of two cameras by observations of a moving person on a ground plane. In *Proc. of IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, pages 129–134, London, UK.

[Cipolla et al., 1999] Cipolla, R., Drummond, T., and Robertson, D. (1999). Camera calibration from vanishing points in image of architectural scenes. In *Proc. of British Machine Vision Conference (BMVC)*, pages 382–391.

[Clarke and Fryer, 1998] Clarke, T. and Fryer, J. (1998). The development of camera calibration methods and models. *The Photogrammetric Record*, 16(91):51–66.

[Criminisi et al., 2000] Criminisi, A., Reid, I., and Zisserman, A. (2000). Single view metrology. *International Journal of Computer Vision (IJCV)*, 40(2):123–148.

[Delage et al., 2006] Delage, E., Lee, H., and Ng, A. (2006). A dynamic bayesian network model for autonomous 3d reconstruction from a single indoor image. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2418–2428.

[Delaitre et al., 2012] Delaitre, V., Fouhey, D., Laptev, I., Sivic, J., Gupta, A., and Efros, A. (2012). Scene semantics from long-term observation of people. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 284–298.

[Ess et al., 2009] Ess, A., Leibe, B., Schindler, K., and Van Gool, L. (2009). Robust multiperson tracking from a mobile platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(10):1831–1846.

[Faig, 1975] Faig, W. (1975). Calibration of close-range photogrammetry systems: Mathematical formulation. *Photogrammetric Engineering and Remote Sensing*, 41(12):1479–1486.

[Fanelli et al., 2011] Fanelli, G., Gall, J., and Van Gool, L. (2011). Real time head pose estimation with random regression forests. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 617–624.

[Faugeras, 1992] Faugeras, O. (1992). What can be seen in three dimensions with an uncalibrated stereo rig. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 563–578.

[Felzenszwalb et al., 2010] Felzenszwalb, P., Girshick, R., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32:1627–1645.

[Felzenszwalb and Huttenlocher, 2004] Felzenszwalb, P. and Huttenlocher, D. (2004). Efficient graph-based image segmentation. *International Journal on Computer Vision (IJCV)*, 59(2):167–181.

[Fergus et al., 2005a] Fergus, R., Fei-Fei, L., Perona, P., and Zisserman, A. (2005a). Learning object categories from google's image search. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

[Fergus et al., 2005b] Fergus, R., Perona, P., and Zisserman, A. (2005b). A sparse object category model for efficient learning and exhaustive recognition. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Ferryman et al., 1995] Ferryman, J., Worrall, A., Sullivan, G., and Baker, K. (1995). A generic deformable model for vehicle recognition. In *Proc. of British Machine Vision Conference (BMVC)*.

[Finsterwalder, 1899] Finsterwalder, S. (1899). *Die geometrischen Grundlagen der Photogrammetrie*. Jahresbericht der Deutschen Mathematiker-Vereinigung. Teubner.

[Fischler and Bolles, 1981] Fischler, M. and Bolles, R. (1981). Random, sample cconsensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

[Fleuret et al., 2008] Fleuret, F., Berclaz, J., Lengagne, R., and Fua, P. (2008). Multicamera people tracking with a probabilistic occupancy map. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(2):267–282.

[Flint et al., 2011] Flint, A., Murray, D., and Reid, I. (2011). Manhattan scene understanding using monocular, stereo, and 3d features. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

[Floros and Leibe, 2012] Floros, G. and Leibe, B. (2012). Joint 2d-3d temporally consistent semantic segmentation of street scenes. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2823–2830.

[Fouhey et al., 2012] Fouhey, D., Delaitre, V., Gupta, A., Efros, A., Laptev, I., and Sivic, J. (2012). People watching: Human actions as a cue for single view geometry. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 732–745.

[Frahm et al., 2010] Frahm, J.-M., Georgel, P. F., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.-H., Dunn, E., Clipp, B., and Lazebnik, S. (2010). Building rome on a cloudless day. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 368–381.

[Frey and MacKay, 1998] Frey, B. and MacKay, D. (1998). A revolution: Belief propagation in graphs with cycles. In *Proc. of Conference on Neural Information Processing Systems (NIPS)*, pages 479–485.

[Furukawa et al., 2010] Furukawa, Y., Curless, B., Seitz, S., and Szeliski, R. (2010). Towards internet-scale multi-view stereo. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1434–1441.

[Furukawa and Ponce, 2007] Furukawa, Y. and Ponce, J. (2007). Accurate, dense, and robust multi-view stereopsis. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Furukawa and Ponce, 2008] Furukawa, Y. and Ponce, J. (2008). Accurate camera calibration from multi-view stereo and bundle adjustment. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Gall et al., 2011a] Gall, J., Razavi, N., and Van Gool, L. (2011a). An introduction to random forests for multi-class object detection. In *Theoretical Foundations of Computer Vision*, pages 243–263.

[Gall et al., 2011b] Gall, J., Yao, A., Razavi, N., Van Gool, L., and Lempitsky, V. (2011b). Hough forests for object detection, tracking, and action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(11):2188–2202.

[Gallup et al., 2007] Gallup, D., Frahm, J.-M., Mordohai, P., Yang, Q., and Pollefeys, M. (2007). Real-time plane-sweeping stereo with multiple sweeping directions. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Gallup et al., 2010] Gallup, D., Frahm, J.-M., and Pollefeys, M. (2010). Piecewise planar and non-planar stereo for urban scene reconstruction. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1418–1425.

[Geiger et al., 2012] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361.

[Geiger et al., 2011] Geiger, A., Wojek, C., and Urtasun, R. (2011). Joint 3d estimation of objects and scene layout. In *Neural Information Processing Systems (NIPS)*, pages 1467–1475.

[Geurts et al., 2006] Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1):3–42.

[Gill and Levine, 2009] Gill, G. and Levine, M. (2009). Multi-view object detection based on spatial consistency in a low dimensional space. In *Proc. of German Association for Pattern Recognition Symposium (DAGM)*, pages 211–220.

[Glasner et al., 2011] Glasner, D., Galun, M., Alpert, S., Basri, R., and Shakhnarovich, G. (2011). Viewpoint-aware object detection and pose estimation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1275–1282.

[Gonzalez and Woods, 2007] Gonzalez, R. and Woods, R. (2007). *Digital Image Processing*. Prentice Hall, 3rd edition.

[Gould et al., 2009] Gould, S., Fulton, R., and Koller, D. (2009). Decomposing a scene into geometric and semantically consistent regions. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1–8.

[Guo and Chellappa, 2010] Guo, F. and Chellappa, R. (2010). Video metrology using a single camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(7):1329–1335.

[Guo and Hoiem, 2012] Guo, R. and Hoiem, D. (2012). Beyond the line of sight: Labeling the underlying surfaces. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 761–774.

[Guo et al., 2008a] Guo, Y., Rao, C., Samarasekera, S., Kim, J., Kumar, R., and Sawhney, H. (2008a). Matching vehicles under large pose transformations using approximate 3d models and piecewise mrf model. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Guo et al., 2008b] Guo, Y., Rao, C., Samarasekera, S., Kim, J., Kumar, R., and Sawhney, H. (2008b). Matching vehicles under large pose transformations using approximate 3d models and piecewise mrf model. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Gupta et al., 2010] Gupta, A., Efros, A., and Hebert, M. (2010). Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 482–496.

[Gupta et al., 2009] Gupta, A., Srinivasan, P., Shi, J., and Davis, L. (2009). Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2012–2019.

[Hammersley and Clifford, 1971] Hammersley, J. and Clifford, P. (1971). Markov random fields on finite graphs and lattices. *Unpublished manuscript*.

[Hanbury, 2008] Hanbury, A. (2008). How do superpixels affect image segmentation? In *Proc. of Iberoamerican Congress on Pattern Recognition (CIARP)*, pages 178–186.

[Häne et al., 2013] Häne, C., Zach, C., Cohen, A., Angst, R., and Pollefeys, M. (2013). Joint 3d scene reconstruction and class segmentation. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. to appear.

[Häne et al., 2012] Häne, C., Zach, C., Zeisl, B., and Pollefeys, M. (2012). A patch prior for dense 3d reconstruction in man-made environments. In *Proc. of International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DimPVT)*, pages 563–570.

[Harris and Stephens, 1988] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proc. of the Alvey Vision Conference*, pages 147–151.

[Hartley, 1992] Hartley, R. (1992). Estimation of relative camera positions for uncalibrated cameras. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 579–587.

[Hartley and Zisserman, 2003] Hartley, R. and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

[Hauck, 1883] Hauck, G. (1883). Neue constructionen der perspective und photogrammetrie: Theorie der trilinearen verwandtschaft ebener systeme. *Journal für reine und angewandte Mathematik*, 95:1–35.

[Hedau et al., 2009] Hedau, V., Hoiem, D., and Forsyth, D. (2009). Recovering the spatial layout of cluttered rooms. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

[Heikkilä and Silvén, 1997] Heikkilä, J. and Silvén, O. (1997). A four-step camera calibration procedure with implicit image correction. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1106–1112.

[Hennessy and Patterson, 2011] Hennessy, J. and Patterson, D. (2011). *Computer Architecture: A Quantitative Approach*. The Morgan Kaufmann Series in Computer Architecture and Design. Elsevier Science, 4th edition.

[Hödlmoser et al., 2011a] Hödlmoser, M., Bober, C., and Kampel, M. (2011a). Vehicle guidance implemented on a single-board computer. In *Proc. of Demos at International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DimPVT)*.

[Hödlmoser and Kampel, 2010] Hödlmoser, M. and Kampel, M. (2010). Multiple camera self-calibration and 3d reconstruction using pedestrians. In *Proc. of International Symposium on Visual Computing (ISVC)*, pages 1–10.

[Hödlmoser and Mičušík, 2013] Hödlmoser, M. and Mičušík, B. (2013). Surface layout estimation using multiple segmentation methods and 3d reasoning. In *Proc. of Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*.

[Hödlmoser et al., 2011b] Hödlmoser, M., Mičušík, B., and Kampel, M. (2011b). Camera auto-calibration using pedestrians and zebra-crossings. In *Proc. of IEEE International Conference on Computer Vision Workshop on Visual Surveillance (ICCV-VS)*, pages 1697–1704.

[Hödlmoser et al., 2011c] Hödlmoser, M., Mičušík, B., and Kampel, M. (2011c). Exploiting spatial consistency for object classification and pose estimation. In *Proc. of IEEE International Conference on Image Processing (ICIP)*, pages 993–996.

[Hödlmoser et al., 2013a] Hödlmoser, M., Mičušík, B., and Kampel, M. (2013a). Sparse point cloud densification by combining multiple segmentation methods. In *Proc. of International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DV)*.

[Hödlmoser et al., 2012] Hödlmoser, M., Mičušík, B., Liu, M.-Y., Pollefeys, M., and Kampel, M. (2012). Classification and pose estimation of vehicles in videos by 3d modeling within discrete-continuous optimization. In *Proc. of International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DimPVT)*, pages 198–205.

[Hödlmoser et al., 2013b] Hödlmoser, M., Mičušík, B., Pollefeys, M., Liu, M.-Y., and Kampel, M. (2013b). Model-based vehicle pose estimation and tracking in videos using random forests. In *Proc. of International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DV)*.

[Hoiem et al., 2005a] Hoiem, D., Efros, A., and Hebert, M. (2005a). Automatic photo pop-up. *ACM Transactions on Graphics*, 24(3):577–584.

[Hoiem et al., 2005b] Hoiem, D., Efros, A., and Hebert, M. (2005b). Geometric context from a single image. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 654–661.

[Hoiem et al., 2007] Hoiem, D., Efros, A., and Hebert, M. (2007). Recovering surface layout from an image. *Internation Journal on Computer Vision (IJCV)*, 75(1).

[Junejo, 2009] Junejo, I. (2009). Using pedestrians walking on uneven terrains for camera calibration. *Proc. of International Conference on Machine Vision Applications (MVA).*

[Khan et al., 2010] Khan, S., Cheng, H., Matthies, D., and Sawhney, H. (2010). 3d model based vehicle classification in aerial imagery. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR).*

[Koller, 1993] Koller, D. (1993). Moving object recognition and classification based on recursive shape parameter estimation. In *Proc. of International Conference on Artifical Intelligence (ICAI).*

[Kollnig and Nagel, 1997] Kollnig, H. and Nagel, H.-H. (1997). 3d pose estimation by directly matching polyhedral models to gray value gradients. *International Journal of Computer Vision*, 23(3):283–302.

[Krahnstoever and Mendonça, 2005] Krahnstoever, N. and Mendonça, P. (2005). Bayesian autocalibration for surveillance. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1858–1865.

[Krahnstoever and Mendonça, 2006] Krahnstoever, N. and Mendonça, P. (2006). Autocalibration from tracks of walking people. In *Proc. of British Machine Vision Conference (BMVC)*, pages 107–116.

[Kruppa, 1913] Kruppa, E. (1913). Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *Sitzungsberichte der Mathematisch Naturwissenschaftlichen Kaiserlichen Akademie der Wissenschaften*, 122:1939–1948.

[Kumar et al., 2008] Kumar, R. K., Ilie, A., Frahm, J.-M., and Pollefeys, M. (2008). Simple calibration of non-overlapping cameras with a mirror. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR).*

[Kurillo et al., 2009] Kurillo, G., Li, Z., and Bajcsy, R. (2009). Framework for hierarchical calibration of multi-camera systems for teleimmersion. In *Proc. of IMMERSCON'09*, pages 1–6.

[Kusakunniran et al., 2009] Kusakunniran, W., Li, H., and Zhang, J. (2009). A direct method to self-calibrate a surveillance camera by observing a walking pedestrian. In *Proc. of Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 250–255, Melbourne.

[Ladicky et al., 2009] Ladicky, L., Russell, C., Kohli, P., and Torr, P. (2009). Associative hierarchical crfs for object class image segmentation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 739–746.

[Laussedat, 1899] Laussedat, A. (1899). *La Métrophotographie*. Gauthier-Villars.

[Lee et al., 2009] Lee, D., Hebert, M., and Kanade, T. (2009). Geometric reasoning for single image structure recovery. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Leibe et al., 2007a] Leibe, B., Cornelis, N., Cornelis, K., and Van Gool, L. (2007a). Dynamic 3d scene analysis from a moving vehicle. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[Leibe et al., 2007b] Leibe, B., Cornelis, N., Cornelis, K., and Van Gool, L. (2007b). Dynamic 3d scene analysis from a moving vehicle. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Leotta, 2010] Leotta, M. (2010). *Generic, Deformable Models for 3-D Vehicle Surveillance*. PhD thesis, Brown University, Providence, RI.

[Leotta and Mundy, 2011] Leotta, M. and Mundy, J. (2011). Vehicle surveillance with a generic, adaptive, 3d vehicle model. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(7):1457–1469.

[Lepetit et al., 2005] Lepetit, V., Lagger, P., and Fua, P. (2005). Randomized trees for real-time keypoint recognition. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 775–781.

[Les and Les, 2008] Les, Z. and Les, M. (2008). *Shape Understanding System: The First Steps toward the Visual Thinking Machines*, volume 86 of *Studies in Computational Intelligence*. Springer.

[Levinshtein et al., 2009] Levinshtein, A., Stere, A., Kutulakos, K., Fleet, D., Dickinson, S., and Siddiqi, K. (2009). Turbopixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(12):2290–2297.

[Li et al., 2009] Li, Y., Gu, L., and Kanade, T. (2009). A robust shape model for multi-view car alignment. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2466–2473.

[Liebelt and Schmid, 2010] Liebelt, J. and Schmid, C. (2010). Multi-view object class detection with a 3d geometric model. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Liebelt et al., 2008] Liebelt, J., Schmid, C., and Schertler, K. (2008). Viewpoint-independent object class detection using 3d feature maps. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Lindeberg, 1998] Lindeberg, T. (1998). Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):117–156.

[Liu et al., 2010a] Liu, B., Gould, S., and Koller, D. (2010a). Single image depth estimation from predicted semantic labels. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Liu et al., 2011] Liu, M.-Y., Tuzel, O., Ramalingam, S., and Chellappa, R. (2011). Entropy rate superpixel segmentation. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2097–2104.

[Liu et al., 2010b] Liu, M.-Y., Tuzel, O., Veeraraghavan, A., and Chellappa, R. (2010b). Fast directional chamfer matching. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Liu et al., 2010c] Liu, M.-Y., Tuzel, O., Veeraraghavan, A., Chellappa, R., Agrawal, A., and Okuda, H. (2010c). Pose estimation in heavy clutter using a multi-flash camera. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*.

[Longuet-Higgins, 1987] Longuet-Higgins, H. (1987). Readings in computer vision: issues, problems, principles, and paradigms. *A computer algorithm for reconstructing a scene from two projections*, pages 61–62.

[Lou et al., 2005] Lou, J., Tan, T., Hu, W., Yang, H., and Maybank, S. (2005). 3-d model-based vehicle tracking. *Transactions on Image Processing*, 14:1561–1569.

[Lourakis and Argyros, 2004] Lourakis, M. and Argyros, A. (2004). The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Technical Report TechReport 340, Institute of Computer Science-FORTH.

[Lowe, 2004] Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110.

[Luong and Faugeras, 1997] Luong, Q.-T. and Faugeras, O. (1997). Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision (IJCV)*, 22(3):261–289.

[Lv et al., 2006] Lv, F., Zhao, T., and Nevatia, R. (2006). Camera calibration from video of a walking human. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(9):1513–1518.

[Ma et al., 2003] Ma, Y., Soatto, S., Kosecka, J., and Sastry, S. (2003). *An Invitation to 3D Vision: From Images to Geometric Models*. Springer Verlag.

[Marr, 1983] Marr, D. (1983). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Company.

[Martynov et al., 2011] Martynov, I., Kamarainen, J.-K., and Lensu, L. (2011). Projector calibration by inverse camera calibration. In *Proc. of Scandinavian Conference on Image Processing (SCIA)*, pages 536–544.

[Maybank and Faugeras, 1992] Maybank, S. and Faugeras, O. (1992). A theory of self-calibration of a moving camera. *International Journal of Computer Vision (IJCV)*, 8(2):123–151.

[Meydenbauer, 1892] Meydenbauer, A. (1892). *Das photographische Aufnehmen zu wissenschaftlichen Zwecken insbesondere das Messbild-Verfahren*. Untes Verlagsanstalt.

[Mitzel and Leibe, 2012] Mitzel, D. and Leibe, B. (2012). Taking mobile multi-object tracking to the next level: People, unknown objects, and carried items. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 566–579.

[Mitzel et al., 2011] Mitzel, D., Sudowe, P., and Leibe, B. (2011). Real-time multi-person tracking with time-constrained detection. In *Proc. of British Machine Vision Conference (BMVC)*, pages 104.1–104.11.

[Mičušík and Kosecka, 2009] Mičušík, B. and Kosecka, J. (2009). Piecewise planar city 3d modeling from street view panoramic sequences. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2906–2912.

[Mičušík and Kosecka, 2010] Mičušík, B. and Kosecka, J. (2010). Multi-view superpixel stereo in urban environments. *International Journal on Computer Vision (IJCV)*, 89(1):106–119.

[Mičušík and Pajdla, 2010] Mičušík, B. and Pajdla, T. (2010). Simultaneous surveillance camera calibration and foot-head homology estimation from human detections. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1562–1569.

[Nowozin and Lampert, 2011] Nowozin, S. and Lampert, C. (2011). Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(3-4):185–365.

[Osher and Sethian, 1988] Osher, S. and Sethian, J. (1988). Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of Computational Physics*, 79(1):12–49.

[Ozuysal et al., 2009] Ozuysal, M., Lepetit, V., and Fua, P. (2009). Pose estimation for category specific multiview object localization. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Payet and Todorovic, 2011] Payet, N. and Todorovic, S. (2011). From contours to 3d object detection and pose estimation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*.

[Pepik et al., 2012a] Pepik, B., Gehler, P., Stark, M., and Schiele, B. (2012a). 3d2pm - 3d deformable part models. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 356–370.

[Pepik et al., 2012b] Pepik, B., Stark, M., Gehler, P., and Schiele, B. (2012b). Teaching 3d geometry to deformable part models. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3362–3369.

[Pero et al., 2012] Pero, L. D., Bowdish, J., Fried, D., Kermgard, B., Hartley, E., and Barnard, K. (2012). Bayesian geometric modeling of indoor scenes. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Pero et al., 2011] Pero, L. D., Guan, J., Brau, E., Schlecht, J., and Barnard, K. (2011). Sampling bedrooms. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2009–2016.

[Petrovskaya et al., 2006] Petrovskaya, A., Khatib, O., Thrun, S., and Ng, A. (2006). Bayesian estimation for autonomous object manipulation based on tactile sensors. In *ICRA*, pages 707–714.

[Pflugfelder and Bischof, 2010] Pflugfelder, R. and Bischof, H. (2010). Localization and trajectory reconstruction in surveillance cameras with nonoverlapping views. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(4):709–721.

[Pollefeys et al., 1998] Pollefeys, M., Koch, R., and Van Gool, L. (1998). Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 90–95.

[Pollefeys and Van Gool, 1997] Pollefeys, M. and Van Gool, L. (1997). A stratified approach to metric self-calibration. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 407–412.

[Pollefeys et al., 2004] Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., and Koch, R. (2004). Visual modeling with a hand-held camera. *International Journal of Computer Vision (IJCV)*, 59(3):207–232.

[Press et al., 1992] Press, W., Flannery, B., Teukolsky, S., and Vetterling, W. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press.

[Puwein et al., 2011] Puwein, J., Ziegler, R., Vogel, J., and Pollefeys, M. (2011). Robust multi-view camera calibration for wide-baseline camera networks. In *Proc. of IEEE Workshop on Applications of Computer Vision (WACV)*, pages 321–328.

[Ren and Malik, 2003] Ren, X. and Malik, J. (2003). Learning a classification model for segmentation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 10–17.

[Rogez et al., 2008] Rogez, G., Rihan, J., Ramalingam, S., Orrite, C., and Torr, P. (2008). Randomized trees for human pose detection. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Rother et al., 2004] Rother, C., Kolmogorov, V., and Blake, A. (2004). Grabcut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314.

[Savarese and Li, 2007] Savarese, S. and Li, F.-F. (2007). 3d generic object categorization, localization and pose estimation. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1–8.

[Scaramuzza et al., 2009] Scaramuzza, D., Fraundorfer, F., and Siegwart, R. (2009). Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*.

[Schaaf, 1992] Schaaf, L. (1992). *Out of the Shadows: Herschel, Talbot, and the Invention of Photography*. Yale University Press.

[Scheimpflug, 1904] Scheimpflug, T. (1904). Improved method and apparatus for the systematic alteration or distortion of plane pictures and images by means of lenses and mirrors for photography and for other purposes. *GB Patent No. 1196*.

[Schels et al., 2012] Schels, J., Liebelt, J., and Lienhart, R. (2012). Learning an object class representation on a continuous viewsphere. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3170–3177.

[Schwing et al., 2012] Schwing, A., Hazan, T., Pollefeys, M., and Urtasun, R. (2012). Efficient structured prediction for 3d indoor scene understanding. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2815–2822.

[Schwing and Urtasun, 2012] Schwing, A. and Urtasun, R. (2012). Efficient exact inference for 3d indoor scene understanding. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 299–313.

[Se, 2000] Se, S. (2000). Zebra-crossing detection for the partially sighted. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 211 – 217.

[Shapiro and Stockman, 2001] Shapiro, L. and Stockman, G. (2001). *Computer Vision*. Prentice Hall.

[Shitrit et al., 2011] Shitrit, H. B., Berclaz, J., Fleuret, F., and Fua, P. (2011). Tracking multiple people under global appearance constraints. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 137–144.

[Shotton et al., 2011] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1297–1304.

[Siegwart and Nourbakhsh, 2004] Siegwart, R. and Nourbakhsh, I. (2004). *Introduction to Autonomous Mobile Robots*. MIT Press.

[Sinha et al., 2004] Sinha, S., Pollefeys, M., and McMillan, L. (2004). Camera network calibration from dynamic silhouettes. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 195–202.

[Snavely et al., 2006] Snavely, N., Seitz, S., and Szeliski, R. (2006). Photo tourism: Exploring photo collections in 3d. In *ACM Transactions on Graphics*, pages 835–846.

[Sonka et al., 2008] Sonka, M., Hlavac, V., and Boyle, R. (2008). *Image Processing, Analysis, and Machine Vision*. Thomson, 3rd edition.

[Stark et al., 2010] Stark, M., Goesele, M., and Schiele, B. (2010). Back to the future: Learning shape models from 3d cad data. In *Proc. of British Machine Vision Conference (BMVC)*, pages 1–11.

[Stauffer and Grimson, 1999] Stauffer, C. and Grimson, E. (1999). Adaptive background mixture models for real-time tracking. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2246–2252.

[Sturm, 2011] Sturm, P. (2011). A historical survey of geometric computer vision. In *Proc. of International Conference on Computer Analysis of Images and Patterns (CAIP)*, pages 1–8.

[Sturm and Maybank, 1999] Sturm, P. and Maybank, S. (1999). On plane-based camera calibration: A general algorithm, singularities, applications. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1432–1437.

[Sun et al., 2012] Sun, M., Kohli, P., and Shotton, J. (2012). Conditional regression forests for human pose estimation. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Sun and Cooperstock, 2005] Sun, W. and Cooperstock, J. (2005). Requirements for camera calibration: Must accuracy come with a high price? In *Proc. of IEEE Workshop on Applications of Computer Vision (WACV)*, pages 356–361.

[Sutor, 1939] Sutor, J. (1939). *Bestimmung der inneren Orientierung und Verbildung aus Rundbildern*. PhD thesis, Technische Hochschule Berlin.

[Thomas et al., 2006] Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., and Van Gool, L. (2006). Towards multi-view object class detection. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Thompson, 1968] Thompson, E. (1968). The projective theory of relative orientation. *Photogrammetria*, 23:67–75.

[Tissandier, 1886] Tissandier, G. (1886). *La photographie en ballon*. Gauthier-Villars.

[Toshev et al., 2009] Toshev, A., Makadia, A., and Daniilidis, K. (2009). Shape-based object recognition in videos using 3d synthetic object models. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Triggs, 1997] Triggs, B. (1997). Autocalibration and the absolute quadric. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 609–614.

[Tsai et al., 2011] Tsai, G., Xu, C., Liu, J., and Kuipers, B. (2011). Real-time indoor scene understanding using bayesian filtering with motion cues. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 121–128.

[Tsai, 1987] Tsai, R. (1987). A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344.

[Urtasun et al., 2006] Urtasun, R., Fleet, D., and Fua, P. (2006). 3d people tracking with gaussian process dynamical models. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 238–245.

[Villamizar et al., 2011] Villamizar, M., Grabner, H., Moreno-Noguer, F., Andrade-Cetto, J., Van Gool, L., and Sanfeliu, A. (2011). Efficient 3d object detection using multiple pose-specific classifiers. In *Proc. of British Machine Vision Conference (BMVC)*, pages 20.1–20.10.

[von Sanden, 1908] von Sanden, H. (1908). *Die Bestimmung der Kernpunkte in der Photogrammetrie*. PhD thesis, Georg-August-Universität Göttingen.

[Wang et al., 2005] Wang, G., Hu, Z., Wu, F., and Tsui, H.-T. (2005). Single view metrology from scene constraints. *Image and Vision Computing (IVC)*, 23(9):831 – 840.

[Wang et al., 2011] Wang, S., Lu, H., Yang, F., and Yang, M.-H. (2011). Superpixel tracking. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 1323–1330.

[Wildenauer and Hanbury, 2012] Wildenauer, H. and Hanbury, A. (2012). Robust camera self-calibration from monocular images of manhattan worlds. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2831–2838.

[Woodford et al., 2008] Woodford, O., Torr, P., Reid, I., and Fitzgibbon, A. (2008). Global stereo reconstruction under second order smoothness priors. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Wu et al., 2011] Wu, C., Frahm, J.-M., and Pollefeys, M. (2011). Repetition-based dense single-view reconstruction. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3113–3120.

[Xiang and Savarese, 2012] Xiang, Y. and Savarese, S. (2012). Estimating the aspect layout of object categories. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3410–3417.

[Xiao and Furukawa, 2012] Xiao, J. and Furukawa, Y. (2012). Reconstructing the world's museums. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 668–681.

[Xiao and Quan, 2009] Xiao, J. and Quan, L. (2009). Multiple view semantic segmentation for street view images. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pages 686–693.

[Yoneyama et al., 2005] Yoneyama, A., Yeh, C.-H., and Kuo, J. (2005). Robust vehicle and traffic information extraction for highway surveillance. *Journal on Applied Signal Processing*, pages 2305–2321.

[Zach, 2007] Zach, C. (2007). *High-Performance Modeling From Multiple Views Using Graphics Hardware*. PhD thesis, Graz University of Technology.

[Zhang et al., 2008a] Zhang, Z., Li, M., Huang, K., and Tan, T. (2008a). 3d model based vehicle localization by optimizing local gradient based fitness evaluation. In *Proc. of IEEE International Conference on Pattern Recognition (ICPR)*.

[Zhang et al., 2008b] Zhang, Z., Li, M., Huang, K., and Tan, T. (2008b). Practical camera auto-calibration based on object appearance and motion for traffic scene visual surveillance. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

[Zhang et al., 2012] Zhang, Z., Liu, W., Metsis, V., and Athitsos, V. (2012). A viewpoint-independent statistical method for fall detection. In *Proc. of IEEE International Conference on Pattern Recognition (ICPR)*.

[Zhang et al., 2011] Zhang, Z., Matsushita, Y., and Ma, Y. (2011). Camera calibration with lens distortion from low-rank textures. In *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2321–2328.

[Zhang and Zhang, 2000] Zhang, Z. and Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22:1330–1334.

Nomenclature

BG	Background
CAD	Computer-Aided Design
CCTfM	Camera Calibration Toolbox for Matlab
CRF	Conditional Random Field
DoG	Difference of Gaussian
FDCM	Fast Directional Chamfer Matching
FG	Foreground
GMM	Gaussian Mixture Model
GPU	Graphics Processing Unit
HOG	Histogram of Oriented Gradients
IAC	Image of the Absolute Conic
ICM	Iterated Conditional Modes
LBP	Loopy Belief Propagation
MAP	Maximum A Posteriori Probability
MRF	Markov Random Field
MSER	Maximally Stable Extremal Regions
NMS	Non-Maximal Supression
RANSAC	RANdom SAmple Consensus
RF	Random Forest
RGB	Red Green Blue

SfM	Structure from Motion
SIFT	Scale Invariant Feature Transform
SLIC	Simple Linear Iterative Clustering

Notations

Symbol	Used Font	Description
$a, b, c, d \ldots A, B, C, D$	Default	Scalars, both uppercase and lowercase letters
$\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d} \ldots \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$	Bold	Vectors, both uppercase and lowercase letters
$\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}$	Monospace	Matrices, only uppercase letters
$\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}$	Calligraphy	Sets of elements, only uppercase letters

Personal Information

Name	Michael Hödlmoser
Nationality	Austria
Date of Birth	July 03, 1985

Education

Since 11/2009	PhD candidate, Computer Science, Visual Computing. Vienna University of Technology, Institute of Computer Aided Automation, Computer Vision Lab. Advisors: Priv. Doz. Dr. Martin Kampel Prof. Dr. Marc Pollefeys Dr. Branislav Mičušík
09/2004 – 06/2008	DI(FH) (∼ MSc) in Information Technology and Systems Management, Salzburg University of Applied Sciences. Passed with credits.
09/2006 – 01/2007	Study abroad program at Hawai'i Pacific University, Honolulu, Hawai'i, USA. Passed with distinction.

Academic Work Experience

Since 09/2009	Research associate. Vienna University of Technology, Computer Vision Lab. Funded by: • FFG Dissertation Fellowship *CAPRI*, 2011-2013 • FFG Project *miniSPOT.net*, 2010-2012 • FFG Project *IObserveNG*, 2009-2010
12/2012	Visiting PhD student. Advisor: Prof. Dr. Marc Pollefeys. ETH Zurich, Computer Vision and Geometry Group, Zurich, Switzerland.
08/2011 – 09/2011	Visiting PhD student. Advisor: Prof. Dr. Marc Pollefeys. ETH Zurich, Computer Vision and Geometry Group, Zurich, Switzerland.
02/2009 – 08/2009	Tutor. Image and Video Processing. Vienna University of Technology, Institute of Software Technology and Interactive Systems, Vienna.
07/2010	Attended the International Computer Vision Summer School (ICVSS'10) in Scicli, Italy.

Industrial Work Experience

Since 01/2011	Research associate and video processing engineer. Responsible for realization of EU FP7 project *RHEA* and FFG project *CAPRI*, CogVis Software and Consulting GmbH, Vienna.
02/2009 – 08/2009	Part-time C++ software engineer. Implementation of new features for authentication software using smart cards. Vasco Data Security Austria, Vienna.
10/2008 – 01/2009	Part-time software engineer. Implementation of object detection algorithms on a DSP using C and Assembler. OnDemand Microelectronics AG, Vienna.
02/2008 – 07/2008	Preparation of diploma thesis entitled *Analysis, Design and Implementation of Scan2Email on a GSM / GPRS Fax Machine*. Sagem Communications Austria, Vienna.
07/2007 – 01/2008	Professional practical training. R&D department Bernecker+Rainer. Porting the B+R specific Soft-CNC, ARNC0, to Windows for simulation purposes, Eggelsberg.