UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

Arbeitsgruppe
Aktives Sehen

MARKOV RANDOM FIELD TERRAIN CLASSIFICATION FOR AUTONOMOUS ROBOTS IN UNSTRUCTURED TERRAIN

Vom Promotionsausschuss des Fachbereichs 4: Informatik
an der Universität Koblenz-Landau
zur Verleihung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)
genehmigte

DISSERTATION

von

Dipl.-Inform. Marcel Häselich

Koblenz - 2014

Datum der Einreichung:	27. Juni 2014
Datum der wissenschaftlichen Aussprache:	17. Dezember 2014
Vorsitzender des Promotionsausschusses:	Prof. Dr. Ralf Lämmel
Vorsitzender der Promotionskommision:	Prof. Dr. Ulrich Furbach
Erster Berichterstatter:	Prof. Dr. Dietrich Paulus
Zweiter Berichterstatter	Prof. Dr. Marcin Grzegorzek

Veröffentlicht als Dissertation der Universität Koblenz-Landau.

Zusammenfassung

Diese Doktorarbeit beschäftigt sich mit dem Problem der Terrainklassifikation im unstrukturierten Außengelände. Die Terrainklassifikation umfasst dabei das Erkennen von Hindernissen und flachen Bereichen mit der einhergehenden Analyse der Bodenoberfläche. Ein 3D Laser-Entfernungsmesser wurde als primärer Sensor verwendet, um das Umfeld des Roboters zu vermessen. Zunächst wird eine Gitterstruktur zur Reduktion der Daten eingeführt. Diese Datenrepräsentation ermöglicht die Integration mehrerer Sensoren, z.B. Kameras für Farb- und Texturinformationen oder weitere Laser-Entfernungsmesser, um die Datendichte zu erhöhen. Anschließend werden für alle Terrainzellen des Gitters Merkmale berechnet. Die Klassifikation erfolgt mithilfe eines Markov Zufallsfeldes für Kontextsensitivität um Sensorrauschen und variierender Datendichte entgegenzuwirken. Ein Gibbs-Sampling Ansatz wird zur Optimierung eingesetzt und auf der CPU sowie der auf GPU parallelisiert um Ergebnisse in Echtzeit zu berechnen. Weiterhin werden dynamische Hindernisse unter Verwendung verschiedener State-of-the-Art Techniken erkannt und über die Zeit verfolgt. Die berechneten Informationen, wohin sich andere Verkehrsteilnehmer bewegen und in Zukunft hinbewegen könnten, werden verwendet, um Rückschlüsse auf Bodenoberflächen zu ziehen die teilweise oder vollständig unsichtbar für die Sensoren sind. Die Algorithmen wurden auf unterschiedlichen autonomen Roboter-Plattformen getestet und eine Evaluation gegen von Menschen annotierte Grundwahrheiten von Karten aus mehreren Millionen Messungen wird präsentiert. Der in dieser Arbeit entwickelte Ansatz zur Terrainklassifikation hat sich in allen Anwendungsbereichen bewährt und neue Erkenntnisse geliefert. Kombiniert mit einem Pfadplanungsalgorithmus ermöglicht die Terrain Klassifikation die vollständige Autonomie für radgetriebene Roboter in natürlichem Außengelände.

Abstract

This thesis addresses the problem of terrain classification in unstructured outdoor environments. Terrain classification includes the detection of obstacles and passable areas as well as the analysis of ground surfaces. A 3D laser range finder is used as primary sensor for perceiving the surroundings of the robot. First of all, a grid structure is introduced for data reduction. The chosen data representation allows for multi-sensor integration, e.g., cameras for color and texture information or further laser range finders for improved data density. Subsequently, features are computed for each terrain cell within the grid. Classification is performed with a Markov random field for context-sensitivity and to compensate for sensor noise and varying data density within the grid. A Gibbs sampler is used for optimization and is parallelized on the CPU and GPU in order to achieve real-time performance. Dynamic obstacles are detected and tracked using different state-of-the-art approaches. The resulting information — where other traffic participants move and are going to move to — is used to perform inference in regions where the terrain surface is partially or completely invisible for the sensors. Algorithms are tested and validated on different autonomous robot platforms and the evaluation is carried out with human-annotated ground truth maps of millions of measurements. The terrain classification approach of this thesis proved reliable in all real-time scenarios and domains and yielded new insights. Furthermore, if combined with a path planning algorithm, it enables full autonomy for all kinds of wheeled outdoor robots in natural outdoor environments.

Acknowledgments

A thesis is a complex project over a long period of one's life. Mine covered almost half a decade and I owe thanks to a lot of people who accompanied and supported me during that time. I was only able to reach my goal with their support.

First of all my thanks goes to the person who made all this possible, my supervisor Prof. Dr.-Ing. Dietrich Paulus. He granted me all the freedom I needed to perform my research while his advice and ideas were equally of great help for this thesis.

Secondly, I would like to thank Prof. Dr.-Ing. Marcin Grzegorzek for his willingness to mentor my work. Never will I forget that one week in Poland, my first scientific excursion ever, where the Markov random fields idea was forged in the first place.

I would also like to express my sincere thanks to all my colleagues. Most notably Frank Neuhaus and Nicolai Wojke who suffered the most from my lifelong struggle with cmake and my proclivity for making the same mistakes again. Christian Winkens for sharpening the idea of integrating dynamic obstacles into terrain classification during a workshop in France. Detlev Droege for letting us turn his car into a robot so many times. And of course Dagmar Lang, for her critical eye and excellent reviewing skills. I would like to thank Nicolai a second time, this time for his great work in car detection and tracking. I as well owe a lot of thanks to students that contributed to this thesis: Denis Dillenberger and Frank Neuhaus for putting all the sensors and the platform into operation, and preparing all the stuff I used so frequently later on. Marc Arends for his pioneering work with Markov random fields. Nikolay Handzhiyski and Laura Haraké for their contribution to the robot's path planning. René Bing for creating the calibration pattern and the useful calibration graphical user interface. Michael Klostermann, Martin Prinzen, and Benedikt Jöbgen for their contribution to the pedestrian detection and tracking system. Furthermore, I would like to thank the employees of the Wehrtechnische Dienststelle 51 in Koblenz for the long-lasting research and development cooperation.

Also, my thanks go to Simon Eggert and Benjamin Knopp for always keeping the coffee flowing when fatigue threatened to overwhelm me.

And of course my parents: studying computer sciences would not have been possible without their support and financial assistance.

Finally, I am grateful to my wife Sabrina, for giving birth to our two wonderful children. Thank you for all the support you gave me and for enduring my moods before important deadlines.

Contents

Chapter 1

Introduction

Autonomous navigation and exploration in unstructured environments is a major challenge in robotics and an active field of research. In order to fulfill complex outdoor tasks, a robot needs enhanced sensing and interpreting abilities to detect drivable regions and to avoid collisions.

For an autonomous outdoor system, the ideal world is fully observable and every constraint is known. All properties and observations are deterministic and static and the robot possesses unlimited resources and time. The real world scenario addressed in this work embodies the exact opposite of such an ideal world: with its sensors, a robot perceives only excerpts of an unknown world. These sensor readings cannot even be trusted due to sensor noise or vibrations caused by egomotion on rough terrain. On top of that, obstacles can be dynamic and change over time or behave differently depending on their own unknown tasks or goals. Although the environment is continuous and the number of actions is infinite, the robot only possesses a limited amount of computing power. Results have to be available in real-time, e.g., to be able to avoid a collision with a tree or a car.

Advances in robotic platform robustness and especially the fast development and progression of new and improved sensors allow robots to proceed more and more to rough terrain. Over the past decade, a number of competitions have been initiated to demonstrate, evaluate, and expedite the capabilities of outdoor robots. In the *RoboCup Rescue* competition [15][1], for example, autonomous robots are instructed to navigate in a simulated disaster scenario to detect victims and create maps for rescue workers. The *DARPA Grand Challenges* [6] in 2004 and 2005, as well as the *DARPA Urban Challenge* [7] in 2007 also generated great improvements and grabbed public attention. The Grand Challenges required an autonomous robot to navigate safely through unstructured terrain with the aid

[1] Quotes in this work are separated into three different types. Works of other authors are referred to by [AES+08], for example. My own publications are referred to by [Häselich et al., 2011a] and [20] is an example of an internet source.

of *Global Positioning System* (GPS) waypoints. For the Urban Challenge, the scope was broadened by adding traffic and traffic rules in an urban scenario.

The requirements of an unstructured environment reach far beyond simply detecting and following a road. Unstructured objects like bushes or trees surround the robot and the road may only consist of dirt and grass or there may not even be a road. Hence, the terrain surface needs to be analyzed very thoroughly since the algorithms have to decide where the robot can drive next. This leads to the terrain classification problem in which the environment needs to be classified on the basis of available sensor data. The classification result can contain impassable regions which the robot has to avoid or passable regions where navigation is possible. Passable regions should in turn underlie a further differentiation considering the terrain surface in those regions. Thereby, the robot's path planning algorithm can choose to navigate either with longer trajectories over smooth terrain or take shortcuts over rough terrain surfaces. Both the terrain classification results and navigating from them have to meet a strict real-time criterion: for the terrain classification approach, this means that all computations have to be done in real-time. The resulting environment representation needs to be efficient enough so that a path planning algorithm can create trajectories on it in real-time, too.

Another research domain is the development of roadworthy, self-driving cars. In contrast to the scenario addressed in this work, automotive manufacturers are working on robot cars that will be able to autonomously navigate in traffic and urban scenarios. First projects in this area deal with selected applications. *Piloted parking* (cf. see [1, 2]), for example, is a scenario in which car owners are able to leave their car outside a car park. Once the vehicle is empty and the track is free of humans, the vehicle navigates autonomously to a specific parking space. From July to October, 2010, a convoy of four vehicles drove from Parma, Italy to Shanghai, China [21]. The cars followed a predefined route, used GPS for orientation, and large parts of the route were driven autonomously with few incidents. Other vehicles use 3D laser range finders (LRFs) to navigate in road scenarios [16, 18, 19]. A prefabricated 3D map, for example, created from a previous trip on the same route, is commonly used for these applications. In contrast to mapping approaches, terrain classification is directly coupled to the path planning domain. In addition, the goals of mapping algorithms are the creation, expansion, and maintenance of persistent environment maps. The situation is different with terrain classification where the classification result is discarded immediately or at least after a couple of seconds.

Algorithms and approaches developed in this work focus on wheeled outdoor robots. Two different platforms with widely varying properties are used to experiment, evaluate, and validate the algorithms. The first robot is a heavyweight 500 kg robot with all-wheel steering and all-wheel drive, the *Mustang MK IA*. Equipped with four powerful motors, the Mustang MK IA can be directed by steering angle and velocity, and achieves a top speed of up to 14 km/h. This platform was provided by the *Bundeswehr Technical Center for Engineer and*

General Field Equipment for developmental and research experiments. An image of the Mustang MK IA is shown at the upper part of Figure 1.1.

The second robot is a lightweight and agile robot called *Roach* with an all-wheel drive. Roach is constructed on a *FORBOT* chassis build by the manufacturer *Roboterwerk* [13]. Its motors accelerate the three wheels on each side via chain links enabling Roach to achieve a top speed of up to 20 km/h. This type of locomotion allows rotations and turns while staying on the same spot. The robot platform was acquired by the Deutsche Forschungsgemeinschaft (DFG) and provided to the Active Vision Group of Prof. Dr.-Ing. Dietrich Paulus under research contract PA 599/11-1. An image of Roach is shown in the lower part of Figure 1.1.

Both robots are equipped with multiple sensors. Light Detection And Ranging (LIDAR, in military context often referred to as LADAR) is frequently used in robotic systems. Modern LRFs deliver fast point measurement of most solid objects in their environment up to a range of several hundred meters, depending on the sensor. Localization and mapping approaches often make use of 2D LRFs, which are cheap, small, and well-suited for most indoor scenarios. Especially after the first DARPA, 3D LRFs started to conquer the market. Self-built rotating devices or 2D LRFs on pan-tilt units are able to gather a whole 3D point cloud instead of a single plane. The sound-system manufacturer Velodyne built a 3D LRF, the *Velodyne HDL-64E* [20]. It consists of 64 lasers rotating around the upright axis, gathering data at a frequency of up to 15 Hz. An image of the Velodyne HDL-64E is shown in Figure 1.2 at the center. A Velodyne HDL-64E is mounted on top of the Mustang MK IA. The data sheet of the Velodyne HDL-64E is given in Appendix A.

Later, due to customer demand, Velodyne developed and sold the *Velodyne HDL-32E*. The HDL-32E is a smaller and lighter version of the HDL-64E, consisting of only 32 lasers and rotating at a higher frequency of 20 Hz. An image of the Velodyne HDL-64E is shown in Figure 1.2 at the very left. A HDL-32E is mounted on top of Roach and was purchased and provided by the DFG, like the FORBOT. The data sheet of the Velodyne HDL-32E is given in Appendix B.

Since LRFs provide only distance measurements resulting in 3D points with intensity values, the robots are additionally equipped with different camera systems. One of the cameras that has been attached to the front is the *Logitech HD Pro Webcam C910* [10], allowing Full High Definition video capturing (up to 1920×1080 pixels). Another camera is the *Philips SPC1300NC*, of which two were deployed on the left and the right side of the robot, respectively. The two Philips SPC1300NC cameras allow a native video resolution of 640×480 pixels.

For pose determination, an inertial measurement unit (IMU), the *xSens MTi* [23], and an GPS receiver, the *Navilock NL-302U GPS* [17], were used. Furthermore, the radar-based ground sensor *Speed Wedge SW01* [11] is used for egomotion estimation.

Figure 1.1: The autonomous robots Mustang MK IA (top) and Roach (bottom).

Figure 1.2: Overview of the different deployed sensors. From the left to the right: Velodyne HDL-32E and Velodyne HDL-64E (images: [20]), Logitech HD Pro Webcam C910 (image: [10]), Philips SPC1300NC (image: [12]). And in the bottom row: xSens MTi (image: [23]) and Navilock NL-302U GPS (image: [17]).

1.1 Motivation

The scenario described so far can be summarized as complex and fault-prone and requires a fast processing and decision-making time. Due to the egomotion of the robot and possible sensor misreadings or occlusions, locations exist where the sensor data are far from perfect. Hence, an approach is sought that is fast enough to process the sensor data in time and that is able to cover for missing or flawed data.

Markov random fields (MRFs) allow a context sensitive classification of the terrain and, combined with a sophisticated data representation, a real-time classification can be implemented. Modern 3D LRFs provide a rich and thorough interpretation of the environment — if a system is able to process their data with the necessary efficiency. An example of the sensor data is shown in Figure 1.3. As illustrated in the image, the point clouds gathered by the sensor are large and detailed. Combined with the information available from the neighbors, it is

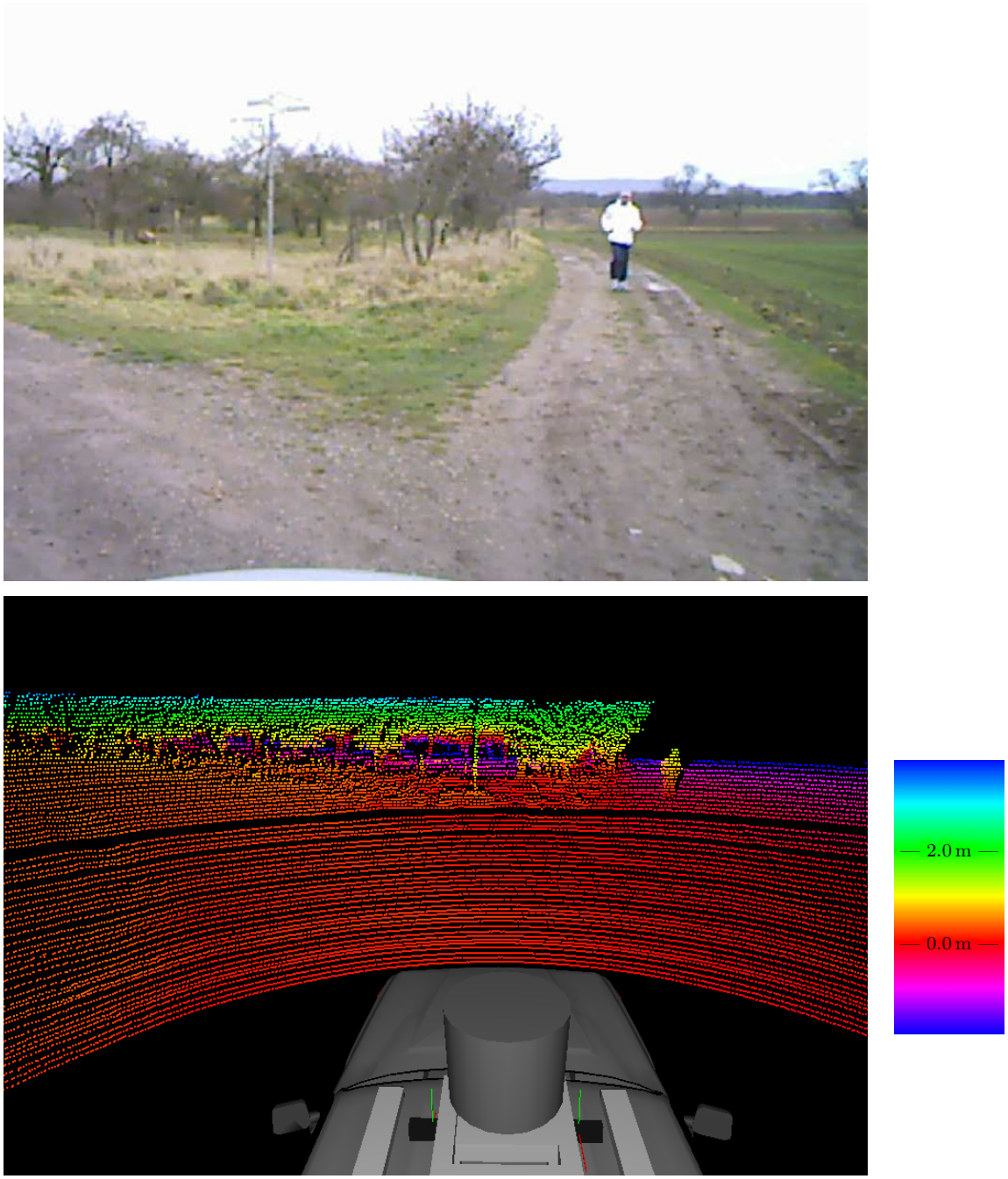


Figure 1.3: Sample data of a camera and a Velodyne HDL-64E. The upper part shows a camera image of a rural environment with trees and a pedestrian. In the lower part, the data of the Velodyne HDL-64E of the same scene is shown. The 3D points are colored according to their height values, as illustrated in the legend on the lower right.

exiting to see if the MRF will be able to yield improved classification results in adequate time.

The terrain classification can hence be divided into two different tasks: in the first part, an algorithm is sought that is able to realize the classification in a robust way given the available sensor data. And in the second part, this algorithm needs to be accelerated or improved in a way that it becomes fast enough to be used on any mobile robot platform without the necessity of overmuch hardware equipment.

Besides terrain surface and static obstacles, dynamic obstacles also need to be regarded. The detection and tracking of dynamic obstacles not only allows the avoidance of present obstacles, but also a prediction of where other objects might be in the future. Therefore, tracking is necessary to predict trajectories which leads towards advanced understanding and correct behavior in traffic situations. Furthermore, the decisions other moving objects make planning their trajectories imply regions with good terrain surface conditions.

1.2 Own Contribution

Scenario, hardware, and motivation were described in the last sections. The novelty and uniqueness of my own contribution is explained in the following paragraphs.

I developed a holistic approach for combined terrain classification on fused sensor data with a Markov random field and the detection and tracking of dynamic obstacles within the vicinity of the robot. Hence, I created the basis for autonomous navigation in complex, unstructured outdoor scenarios.

My own contribution can be split up into three different topics as visualized in Figure 1.4. As indicated in the image, the two topics *terrain classification* and *dynamic obstacles* yield a third one by combining their results. To the best of my knowledge, the last topic *integration of dynamic obstacles into terrain classification* has not been addressed before by other researchers.

Markov Random Field Terrain Classification of Fused Sensor Data
In this area, I created a novel approach that applies Markov random field theory to a new domain. A model successfully used in image segmentation was adapted to a modern 3D LRF fused with camera images of three color cameras and presented in [Häselich et al., 2011a]. A prerequisite for this fusion is the calibration of the cameras to the LRF. Here, I had a minor contribution in the form of a new calibration pattern combined with an existing approach from another domain (cf. [Häselich et al., 2012a]). Experiments revealed that the limitations of the sensor technology include a blindness in the area close to the robot. This is caused by the operating range of the Velodyne LRFs. Another major contribution of my work was the integration of sensor data over time which was presented in a journal article [Häselich et al., 2013a]. In addition, runtime was identified as a

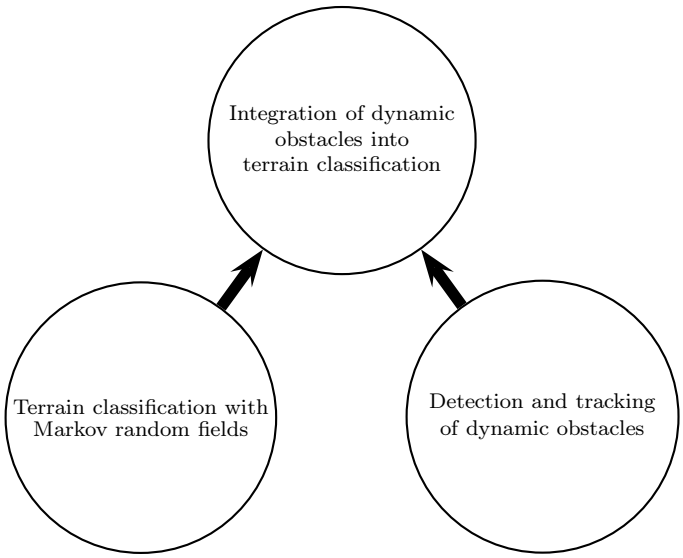


Figure 1.4: Research topics of this thesis.

crucial part of the system, especially w.r.t. path planning and collision avoidance. Therefore, I developed and compared different acceleration approaches, for example, parallelization strategies using the graphics card, in [Häselich et al., 2012b]. Overall, my contributions to this field can be seen as pioneering work and are an inspiration to others (e.g., Laible et al. [LKZ13] adapted my approach and modified it).

Detection and Tracking of Dynamic Obstacles

During my work, early experiments already revealed the importance of dynamic obstacles. Unlike static obstacles, especially the tracking of dynamic obstacles is a necessity for collision-free navigation. Vehicles were identified as primary threats for an autonomous robot and an approach to reliably track vehicles was presented in [Wojke and Häselich, 2012]. In this work, an existing approach created for urban scenarios was adapted, accelerated, and extended for unstructured environments. Pedestrians are also important obstacles for collision-free driving. In [Häselich et al., 2013b], an approach for pedestrian detection in color-camera images was presented. Computation time was identified as problematic part of camera-based detection and acceleration techniques did not yield adequate runtimes. Hence, focus was put on laser-based detection and tracking of pedestrians. This had the advantage that techniques from vehicle detection and tracking could be re-used and that both approaches (for vehicles and pedestrians) work on a single sensor. For the laser-based approach, existing techniques were modified and augmented for unstructured environments as presented in [Häselich et al., 2014]. Here, my work was application-driven as my major interest was not a precise detection and tracking system but rather the results of the tracking in order to

integrate them into terrain classification.

Integration of Dynamic Obstacles into Terrain Classification

A completely new domain emerged during my research. The parallel development of algorithms for terrain classification and dynamic obstacles eventually resulted in the combination of both. The idea was born soon after the first autonomous test drives. In [Häselich et al., 2011b] for example, I presented a novel approach to path planning primarily designed to demonstrate the efficiency of my terrain classification approach. Generic trajectory-templates were used to accelerate the planning process by directly using terrain cells for navigation. This, in the end, resulted in the idea to project trajectories from other obstacles into the terrain grid. Since the combination of both domains was only possible towards the end of my work, no publications were submitted so far but are planned for the near future. Furthermore, research in this new area seems highly promising and additional exploration of the topic will surely follow.

1.3 Outline

This thesis is structured as follows. Chapter 2 contains the basics of Markov random fields and specifies the notation used in the subsequent chapters. Related work reviews for the respective domains are given in the corresponding chapters. The same applies to the experiments conducted, which will explained and pictured in the respective sections. Chapter 3 provides a short overview on the underlying software framework and explains the interaction possibilities of the software modules. The application of the Markov random field theory and my approach solving the terrain classification problem is presented in Chapter 4. Dynamic obstacles and their special role for safe navigation are addressed in Chapter 5. Integration of dynamic obstacles into terrain classification is described in Chapter 6. My thesis concludes in Chapter 7 with a summary and new possible perspectives.

Chapter 2

Fundamentals of Markov Random Fields

Markov random fields, named after the Russian mathematician Andrei Andrejewitsch Markow (⋆1856, †1922), provide the possibility to model context-dependent relationships for versatile data representations and structures. Especially in image processing, MRF applications yield consistent solutions to various problems like denoising, data fusion, or segmentation. Stereo vision and optical flow are further examples where the value or assignment of an image pixel normally depends more on closer neighbors than on those further away.

Within the domain of terrain classification, a set of sites needs to be analyzed and classified based on the sensor data available at these sites. Sensor data are gathered from a moving vehicle by different sensor modalities in outdoor scenarios under varying lighting, weather, and surface conditions. Hence, they are noisy, incomplete due to occlusions or sensor errors, and difficult to interpret in a short period of time. Moreover, it is more likely to find another piece of road at a site surrounded by neighbors that are mainly road segments according to their associated sensor readings. Considering these conditions and the fact that terrain cells can be modeled in an equidistant grid, MRF theory provides an excellent solution to approach the terrain classification problem.

The following chapter specifies MRF notation based on the book of Stan Z. Li [Li09]. Identifiers, definitions, descriptions, and formulas are taken from Li's book and have been adapted to the domain of terrain classification. An overview on the similarities and the few differences between the notations from Section 2.1 to Section 2.7 and Li's notations is given in Appendix C. Thus, a consistent terminology and comparability with similar works is made possible. The content following has been modified and extended with examples and provides the basis for the contribution of my own work in the subsequent chapters.

2.1 Sites and Labels

Image pixels or grid cells are common data representations in vision and robotics. The terrain classification problem can be formulated as a labeling problem in which the solution is the assignment of a label to each terrain cell.

A *site* specifies a site in the terrain. As a simplification, the general definition of $\mathcal{S}$ often uses a one-dimensional indexing. Let $\mathcal{S}$ index a discrete set of m sites as

$$\mathcal{S} = \{1, \ldots, m\} \tag{2.1}$$

in which $1, \ldots, m$ are indices. A squared, equidistant, and finite grid for a 2D data representation of size $n \times n$ ($n \cdot n = m$) is called a *regular grid* and is defined as

$$\mathcal{S} = \{(i, j) \,|\, 1 \leq i, j \leq n\} \quad . \tag{2.2}$$

A *label* can be linked to each site. Let $\mathcal{L}$ be a set of labels. In the discrete case, a label assumes a discrete value in a set of M labels according to

$$\mathcal{L}_{\text{discrete}} = \{l_1, \ldots, l_M\} \quad . \tag{2.3}$$

For example, a label set could be $\mathcal{L} = \{\mathit{Obstacle}, \mathit{Non\text{-}obstacle}\}$. It can either be ordered (e.g., roughness values) or unordered (e.g., terrain classes). In the latter case, the similarity of "equal" or "unequal" is defined between any two labels.

2.2 The Labeling Problem

The labeling problem is to assign a label from $\mathcal{L}$ to each site in $\mathcal{S}$. The set

$$\mathcal{F} = \{\mathrm{f}_1, \ldots, \mathrm{f}_m\} \tag{2.4}$$

is called a *labeling* of sites $\mathcal{S}$ in terms of labels $\mathcal{L}$. The function f is a mapping from $\mathcal{S}$ to $\mathcal{L}$, that is,

$$\mathrm{f} : \mathcal{S} \mapsto \mathcal{L} \quad . \tag{2.5}$$

In the terminology of random fields, a labeling is called a *configuration*. Since all sites in the terrain grid have the same label set $\mathcal{L}$, the set of all possible labelings, the *configuration space*, is the Cartesian product

$$\mathbb{F} = \underbrace{\mathcal{L} \times \mathcal{L} \times \cdots \times \mathcal{L}}_{m \text{ times}} = \mathcal{L}^m \quad . \tag{2.6}$$

For m regular sites and M discrete labels, a total number of M^m possible configurations in $\mathbb{F}$ exists.

An objective function maps a configuration to a real number, measuring the quality in terms of some goodness or cost. The formulation of this function defines

the criteria for the optimal solution. Within configuration space, one solution $\mathcal{F}^*$ exists that receives either the best score or is tied for it: the *optimal configuration.*

In the context of a grid with regular sites and discrete labels, the objective function is an energy function that needs to be minimized. An energy function is defined by its form and its parameters. The form depends on assumptions about the solution $\mathcal{F}$ and the observed data $\boldsymbol{D}$ as well as the parameters $\boldsymbol{\tau}$. Since the parameters are part of the definition of the energy function $\mathrm{E}(\mathcal{F} \,|\, \boldsymbol{D}, \boldsymbol{\tau})$, the optimal solution $\mathcal{F}^*$ is

$$\mathcal{F}^* = \underset{\mathcal{F}}{\operatorname{argmin}} \ \mathrm{E}(\mathcal{F} \,|\, \boldsymbol{D}, \boldsymbol{\tau}) \quad . \tag{2.7}$$

2.3 Neighborhoods and Cliques

The sites in $\mathcal{S}$ are related to one another via a neighborhood system that is defined as

$$\mathcal{N} = \{\mathcal{N}_i \,|\, \forall i \in \mathcal{S}, \mathcal{N}_i \subset \mathcal{S}\} \tag{2.8}$$

where $\mathcal{N}_i$ is the set of sites neighboring i. A site is not neighboring to itself

$$i \notin \mathcal{N}_i \tag{2.9}$$

and the relationship is mutual

$$i \in \mathcal{N}_{i'} \Leftrightarrow i' \in \mathcal{N}_i \quad . \tag{2.10}$$

A *cell* $\mathfrak{c}_k \equiv (i, j)$ indexes a position in a grid of size $n \times n$ with $k = n{\cdot}i + j$. For a regular grid with sites $\mathcal{S}$, the set of neighbors of i is defined as the set of sites with a radius of $\sqrt{r}$ from i

$$\mathcal{N}_i = \{i' \in \mathcal{S} \,|\, [\mathrm{d}(\mathfrak{c}_{i'}, \mathfrak{c}_i)] \leq \sqrt{r}, i' \neq i\} \tag{2.11}$$

where $\mathrm{d}(A, B)$ denotes the Euclidean distance between A and B. Given an equidistant grid, the orders $\{\text{first}, \text{second}, \text{third}, \text{fourth}, \text{fifth}, \ldots\}$ correspond to the distances $\left\{1, \sqrt{2}, 2, \sqrt{5}, \sqrt{8}, \ldots\right\}$. Examples are shown in Figure 2.1.

Since the terrain grid is regular, the ordering of the elements in $\mathcal{S}$ is specified. When the sites $\mathcal{S} = \{(i, j) \,|\, 1 \leq i, j \leq n\}$ correspond to the cells of an $n \times n$ terrain grid, an internal site (i, j) has four neighbors given as

$$\mathcal{N}_{i,j} = \{(i-1, j), (i+1, j), (i, j-1), (i, j+1)\} \tag{2.12}$$

for the first-order neighborhood system, and

$$\begin{aligned} \mathcal{N}_{i,j} = \{&(i-1, j-1), (i-1, j), (i-1, j+1), (i, j-1), \\ &(i, j+1), (i+1, j-1), (i+1, j), (i+1, j+1)\} \end{aligned} \tag{2.13}$$

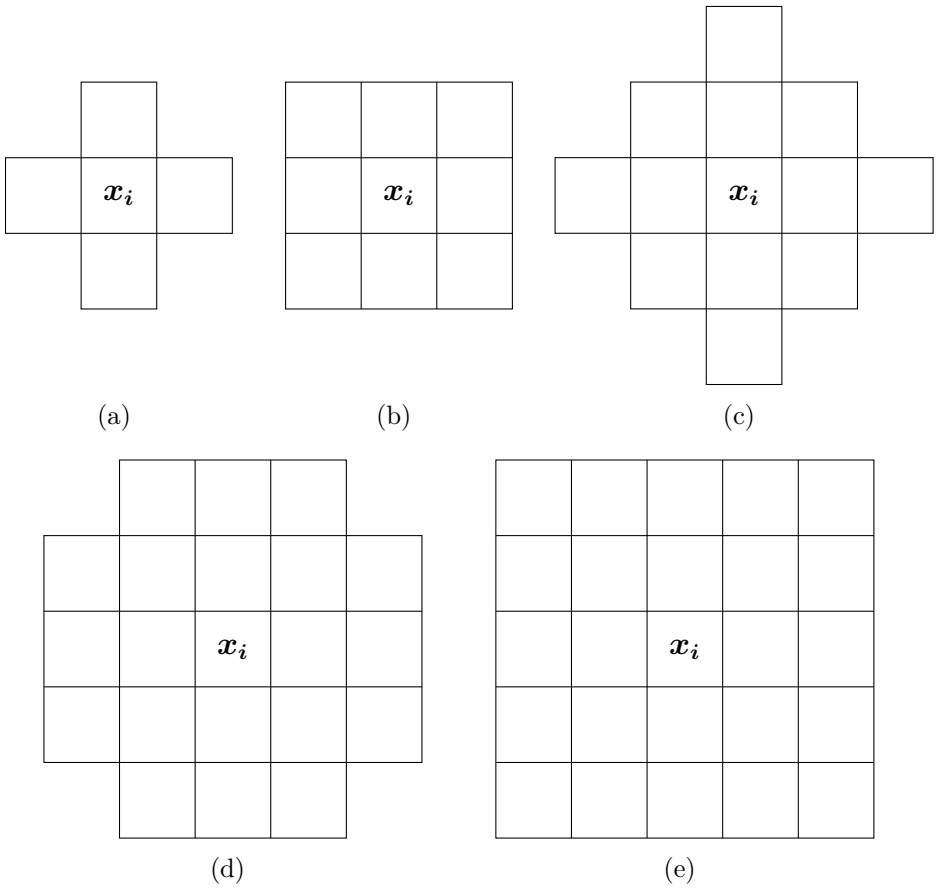


Figure 2.1: Neighborhoods of a site $\boldsymbol{x_i}$ on a regular grid. The image (a) shows the first-order neighborhood system, (b) the second-order, (c) the third-order, (d) the fourth-order, and (e) the fifth-order.

for the second-order neighborhood system, respectively.

The pair $(\mathcal{S}, \mathcal{N}) \equiv G$ constitutes a graph in which $\mathcal{S}$ contains the nodes and $\mathcal{N}$ defines the links between them according to the neighboring relationship. A *clique* $\mathcal{C}$ for $(\mathcal{S}, \mathcal{N})$ is a subset in $\mathcal{S}$ where the sites are fully connected. A *maximal clique* is a clique that cannot be extended by including any more adjacent elements of $\mathcal{S}$. Hence, it cannot be a subset of a larger clique and has the maximum extent. For a regular grid with a first- or second-order neighborhood, it consists either of a single-site $\{i\}$ (cf. Figure 2.2(a)), a pair of neighboring sites $\{i, i'\}$ (cf. Figure 2.2(b) and 2.2(c)), a triple of neighboring sites $\{i, i', i''\}$ (cf. Figure 2.2(d)), or a quadruple-sites $\{i, i', i'', i'''\}$ (cf. Figure 2.2(e)). Figure 2.2 shows all cliques up to the second-order neighborhood in all orientations, but they can be translated according to the capabilities of the underlying neighborhood system. Collections of these cliques are denoted by $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ as

$$\mathcal{C}_1 = \{i \,|\, i \in \mathcal{S}\} \quad , \tag{2.14}$$

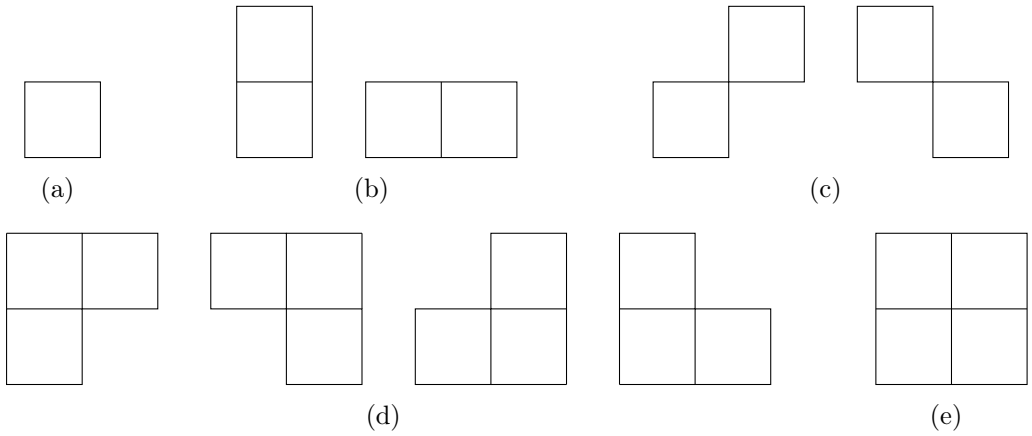


Figure 2.2: Cliques on a regular grid. Cliques for the first-order neighborhood system are shown in (a) and (b), cliques for the second-order neighborhood are shown in (c), (d) and (e).

$$\mathcal{C}_2 = \{\{i, i'\} \mid i \in \mathcal{S}, i' \in \mathcal{N}_i\} \quad , \tag{2.15}$$

$$\mathcal{C}_3 = \{\{i, i', i''\} \mid i, i', i'' \in \mathcal{S} \text{ are neighbors to one another}\} \quad , \tag{2.16}$$

$$\mathcal{C}_4 = \{\{i, i', i'', i'''\} \mid i, i', i'', i''' \in \mathcal{S} \text{ are neighbors to one another}\} \quad . \tag{2.17}$$

Regarding a first-order neighborhood, $\mathcal{C}_1$ and $\mathcal{C}_2$ form the clique $\mathcal{C}$ as

$$\mathcal{C}_{\text{first}} = \mathcal{C}_1 \cup \mathcal{C}_2 \quad . \tag{2.18}$$

For a regular grid with a second-order neighborhood, the clique additionally contains diagonal, triple, and quadruple sites

$$\mathcal{C}_{\text{second}} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3 \cup \mathcal{C}_4 \quad . \tag{2.19}$$

2.4 Markov Random Fields

Let $R = \{r_1, \ldots, r_m\}$ be a *family* (cf. [Sch09]) of random variables defined on the set $\mathcal{S}$ in which each random variable r_i takes a value f_i in $\mathcal{L}$. Here, R contains m random variables, one for each site in $\mathcal{S}$. As the random variables are unordered and may contain repetitions, they are indexed by the set of sites $\mathcal{S}$ as every random variable corresponds to exactly one site. This can be expressed as $(r_i)_{i \in \mathcal{S}}$, where r_i is a member of the family R, an element of the indexed set, and i is an index from the index set $\mathcal{S}$ (also referred to as index area). The family R is called a random field and $r_i = \mathrm{f}_i$ denotes the event that r_i takes the value f_i. The notation $\langle r_1 = \mathrm{f}_1, \ldots, r_m = \mathrm{f}_m \rangle$ denotes the joint event, abbreviated as $R = \mathcal{F}$, where $\mathcal{F} = \langle \mathrm{f}_1, \ldots, \mathrm{f}_m \rangle$ is a configuration of R corresponding to a realization of

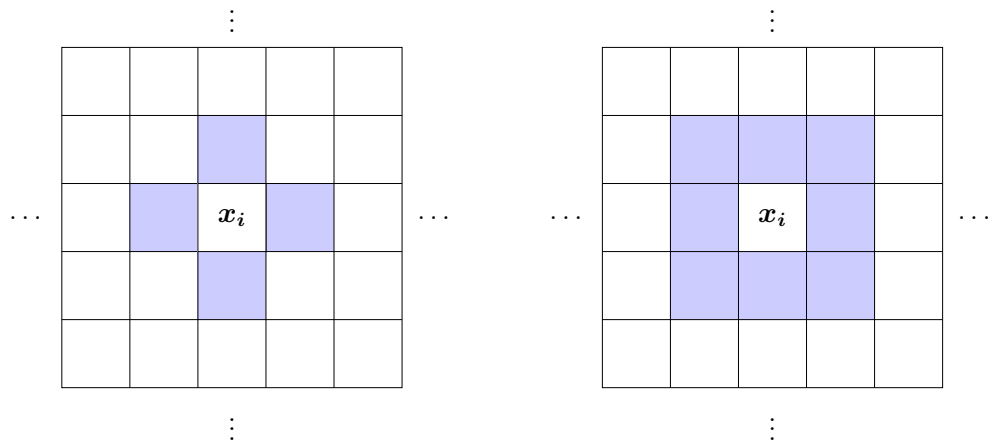


Figure 2.3: Illustration of the Markovianity. Given its neighborhood (depicted with blue sites in the images), a random variable at site $\boldsymbol{x_i}$ is independent from all remaining random variables. The left hand image (a) shows a neighborhood of four and the right hand image (b) a neighborhood of eight sites.

the field. In the case of a discrete set of labels, the probability that random variable r_i takes the value f_i is denoted by $P(r_i = \mathrm{f}_i)$, abbreviated $P(\mathrm{f}_i)$ and the joint probability is denoted as $P(R = \mathcal{F}) = P(r_1 = \mathrm{f}_1, \ldots, r_m = \mathrm{f}_m)$, abbreviated $P(\mathcal{F})$.

R is said to be an MRF on $\mathcal{S}$ w.r.t. a neighborhood system $\mathcal{N}$ if and only if the following two conditions are satisfied:

$$P(\mathcal{F}) > 0, \quad \forall\, \mathcal{F} \in \mathbb{F} \qquad \text{(Positivity)} \tag{2.20}$$

$$P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{S}\setminus\{i\}}) = P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{N}_i}) \qquad \text{(Markovianity)} \tag{2.21}$$

where $\mathbb{F}$ is the set of all possible configurations (cf. Equation 2.6), $\mathcal{S} \setminus \{i\}$ is the set difference, $\mathrm{f}_{\mathcal{S}\setminus\{i\}}$ denotes the set of labels at the sites in $\mathcal{S} \setminus \{i\}$, and

$$\mathrm{f}_{\mathcal{N}_i} = \{f_{i'} \,|\, i' \in \mathcal{N}_i\} \tag{2.22}$$

stands for the set of labels at sites neighboring i. Figure 2.3 illustrates this for a regular grid with a neighborhood of four and eight neighbors.

The two possible approaches for specifying an MRF are either in terms of the conditional probabilities $P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{N}_i})$ or in terms of the joint probability $P(\mathcal{F})$. In Li's book, he promotes the following three reasons argued by Besag [Bes74] for the joint probability approach:

1. No obvious method is available for deducing the joint probability from the associated conditional probabilities.

2. The conditional probabilities themselves are subject to some non-obvious and highly restrictive consistency conditions.

3. The natural specifications of an equilibrium of a statistical process is in terms of the joint probability rather than the conditional distribution of the variables.

The joint probability approach leads to the equivalence between Markov Random Fields and Gibbs distributions as stated by the famous but remarkably unpublished *Hammersley-Clifford theorem* [HC71] (or [Bes74, Cli90]).

2.5 Gibbs Random Fields

A set of random variables R is said to be a *Gibbs random field* (GRF) on $\mathcal{S}$ w.r.t. $\mathcal{N}$ if and only if its configurations obey a *Gibbs distribution* (also called the *Boltzmann distribution* [LL96]). A Gibbs distribution takes the form

$$P(\mathcal{F}) = \mathrm{Z}^{-1} \cdot \mathrm{e}^{-\frac{1}{\mathrm{T}}\mathrm{U}(\mathcal{F})} \tag{2.23}$$

where

$$\mathrm{Z} = \sum_{\mathrm{f} \in \mathbb{F}} \mathrm{e}^{-\frac{1}{\mathrm{T}}\mathrm{U}(\mathrm{f})} \tag{2.24}$$

is a normalizing constant called the *partition function*, T is a constant called the *temperature*, which is usually 1 unless otherwise stated, and $\mathrm{U}(\mathcal{F})$ is the energy function. The energy

$$\mathrm{U}(\mathcal{F}) = \sum_{c \in \mathcal{C}} \mathrm{V}_c(\mathcal{F}) \tag{2.25}$$

is a sum of *clique potentials* $\mathrm{V}_c(\mathcal{F})$ over all possible cliques $\mathcal{C}$. The value of $\mathrm{V}_c(\mathcal{F})$ depends on the local configuration on the clique c.

A GRF is said to be *homogeneous* if $\mathrm{V}_c(\mathcal{F})$ is independent of the relative position of a clique c in $\mathcal{S}$ (cf. Figure 2.2). Furthermore, it is said to be *isotropic* if V_c is independent of the orientation of c.

To calculate the Gibbs distribution, it is necessary to evaluate the partition function Z, which is the sum over all possible configurations in $\mathbb{F}$. Since a large number of elements in $\mathbb{F}$ exists for the combination of discrete $\mathcal{L}$ and $\mathcal{S}$ (cf. Equation 2.6), evaluation is prohibitive even for problems of moderate size. For m regular sites and M discrete labels, a total number of M^m configurations exists.

$P(\mathcal{F})$ measures the probability of the occurrence of a particular configuration $\mathcal{F}$. The more probable configurations are those with lower energies. The temperature T controls the sharpness of the distribution. When the temperature is high, all configurations tend to be equally distributed. Near zero temperature, the distribution concentrates around the global energy minimum. Given T and

$\mathrm{U}(\mathcal{F})$, it is possible to sample from the configuration space $\mathbb{F}$ according to $P(\mathcal{F})$ (cf. Section 2.9).

For discrete labeling problems such as the terrain classification problem, a clique potential $\mathrm{V}_c(\mathcal{F})$ can be specified by a number of *parameters*. For example, letting $\mathcal{F}_c = \{\mathrm{f}_i, \mathrm{f}_{i'}, \mathrm{f}_{i''}\}$ be the local configuration on a triple clique $c = \{i, i', i''\}$, $\mathcal{F}_c$ takes a finite number of states and therefore $\mathrm{V}_c(\mathcal{F})$ takes a finite number of values.

2.6 The Hammersley-Clifford Theorem

An MRF is characterized by its local property (the Markovianity) whereas a GRF is characterized by its global property (the Gibbs distribution). The *Hammersley-Clifford Theorem* [HC71] establishes the equivalence between those two types of properties. The theorem states that R is an MRF on $\mathcal{S}$ w.r.t. $\mathcal{N}$ if and only if R is a GRF on $\mathcal{S}$ w.r.t. $\mathcal{N}$. Many proofs of the theorem exist [Bes74, Mou74, KS80] and Li conducts the proof in two parts. The first is as follows:

Let $P(\mathcal{F})$ be a Gibbs distribution on $\mathcal{S}$ w.r.t. the neighborhood system $\mathcal{N}$. Consider the conditional probability

$$P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{S}\setminus\{i\}}) \underset{\text{Bayes}}{=} \frac{P(\mathrm{f}_{\mathcal{S}\setminus\{i\}} \,|\, \mathrm{f}_i) P(\mathrm{f}_i)}{P(\mathrm{f}_{\mathcal{S}\setminus\{i\}})} \underset{\text{Product rule}}{=} \frac{P(\mathrm{f}_i, \mathrm{f}_{\mathcal{S}\setminus\{i\}})}{P(\mathrm{f}_{\mathcal{S}\setminus\{i\}})} = \frac{P(\mathcal{F})}{\sum_{\mathrm{f}'_i \in \mathcal{L}} P(\mathcal{F}')} \quad (2.26)$$

where $\mathcal{F}' = \left\{\mathrm{f}_1, \ldots, \mathrm{f}_{i-1}, \mathrm{f}'_i, \ldots, \mathrm{f}_m\right\}$ is any configuration that agrees with $\mathcal{F}$ at all sites except possibly i. Writing out $P(\mathcal{F}) = \mathrm{Z}^{-1} \cdot \mathrm{e}^{-\sum_{c\in\mathcal{C}} \mathrm{V}_c(\mathcal{F})}$ gives

$$P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{S}\setminus\{i\}}) = \frac{\mathrm{e}^{-\sum_{c\in\mathcal{C}} \mathrm{V}_c(\mathcal{F})}}{\sum_{\mathrm{f}'_i} \mathrm{e}^{-\sum_{c\in\mathcal{C}} \mathrm{V}_c(\mathcal{F}')}} \quad . \quad (2.27)$$

Divide $\mathcal{C}$ into two sets $\mathcal{A}$ and $\mathcal{B}$ with $\mathcal{A}$ consisting of cliques containing i and $\mathcal{B}$ cliques not containing i. Then Equation 2.27 can be written as

$$P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{S}\setminus\{i\}}) = \frac{[\mathrm{e}^{-\sum_{c\in\mathcal{A}} \mathrm{V}_c(\mathcal{F})}]\ [\mathrm{e}^{-\sum_{c\in\mathcal{B}} \mathrm{V}_c(\mathcal{F})}]}{\sum_{\mathrm{f}'_i}\{[\mathrm{e}^{-\sum_{c\in\mathcal{A}} \mathrm{V}_c(\mathcal{F}')}]\ [\mathrm{e}^{-\sum_{c\in\mathcal{B}} \mathrm{V}_c(\mathcal{F}')}]\}} \quad . \quad (2.28)$$

Because $\mathrm{V}_{\mathcal{C}}(\mathcal{F}) = \mathrm{V}_{\mathcal{C}}(\mathcal{F}')$ for any clique c that does not contain i, $\mathrm{e}^{-\sum_{c\in\mathcal{B}} \mathrm{V}_c(\mathcal{F}')}$ cancels from both the numerator and the denominator. This probability depends only on the potentials of the cliques containing i,

$$P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{S}\setminus\{i\}}) = \frac{\mathrm{e}^{-\sum_{c\in\mathcal{A}} \mathrm{V}_c(\mathcal{F})}}{\sum_{\mathrm{f}'_i} \mathrm{e}^{-\sum_{c\in\mathcal{A}} \mathrm{V}_c(\mathcal{F}')}} \quad , \quad (2.29)$$

and therefore, it depends on labels at the neighborhood of i. This proves that a GRF is an MRF.

For the proof that an MRF is a GRF, Li refers to the uniqueness of the GRF representation [Gri76, KSK76], which provides such a proof. Li conducts the second part of the proof as follows:

The choice of clique potential functions for a specific MRF is not unique and many equivalent choices exist that specify the same Gibbs distribution. However, a unique normalized potential, called the *canonical potential*, exists for every MRF [Gri76].

Let $\mathcal{L}$ be a countable label set. Whenever for some $i \in c$, f_i takes a particular value $l \in \mathcal{L}$, a clique potential function $\mathrm{V}_c(\mathrm{f})$ is said to be *normalized* if $\mathrm{V}_c(\mathrm{f}) = 0$. Griffeath [Gri76] established the mathematical relationship between an MRF distribution $P(\mathcal{F})$ and the unique canonical representation of clique potentials V_c in the corresponding Gibbs distribution [Gri76, KS80] as follows:

Let R be a random field on a finite set $\mathcal{S}$ with local characteristics $P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{S}\setminus\{i\}}) = P(\mathrm{f}_i \,|\, \mathrm{f}_{\mathcal{N}_i})$ and $b \subset c$. Then R is a Gibbs field with *canonical potential function* defined by

$$\mathrm{V}_c(\mathcal{F}) = \begin{cases} 0 & c = \emptyset \\ \sum_{b \subset c} (-1)^{|c \setminus b|} \ln P(\mathrm{f}^b) & c \neq \emptyset \end{cases} \quad , \tag{2.30}$$

where $\emptyset$ denotes the empty set, $|c \setminus b|$ is the number of elements in the set $c \setminus b$, and

$$\mathrm{f}_i^b = \begin{cases} \mathrm{f}_i & \text{if } i \in b \\ 0 & \text{otherwise} \end{cases} \tag{2.31}$$

is the configuration that agrees with $\mathcal{F}$ on set b but assigns the value 0 to all sites outside of b. For non-empty c, the potential can also be obtained as

$$\mathrm{V}_c(\mathcal{F}) = \sum_{b \subset c} (-1)^{|c \setminus b|} \ln P(\mathrm{f}_i^b \,|\, \mathrm{f}_{\mathcal{N}_i}^b) \tag{2.32}$$

where i is any element in b. Such a canonical potential function is *unique* for the corresponding MRF. Using this result, the canonical $\mathrm{V}_c(\mathcal{F})$ can be computed if $P(\mathcal{F})$ is known. Given $\mathrm{V}_c(\mathcal{F})$, the energy of the GRF (cf. Equation 2.25) can be computed. This proves that an MRF is a GRF.

2.7 The Ising and Potts Models

The Hammersley-Clifford theorem provides the basis for the specification of MRF joint distribution functions. Contextual constraints can be formulated on multiple labels, but the most efficient and commonly used form are pair-site clique potentials. Let $\mathcal{C}_1$ be the collection of first-order sites (cf. Equation 2.14) and let $\mathcal{C}_2$ be the collection of second-order sites (cf. Equation 2.15). With clique potential of up to two sites, the energy takes the form

$$\mathrm{U}(\mathcal{F}) = \sum_{\{i\} \in \mathcal{C}_1} \mathrm{V}_1(\mathrm{f}_i) + \sum_{\{i,i'\} \in \mathcal{C}_2} \mathrm{V}_2(\mathrm{f}_i, \mathrm{f}_{i'}) \quad , \tag{2.33}$$

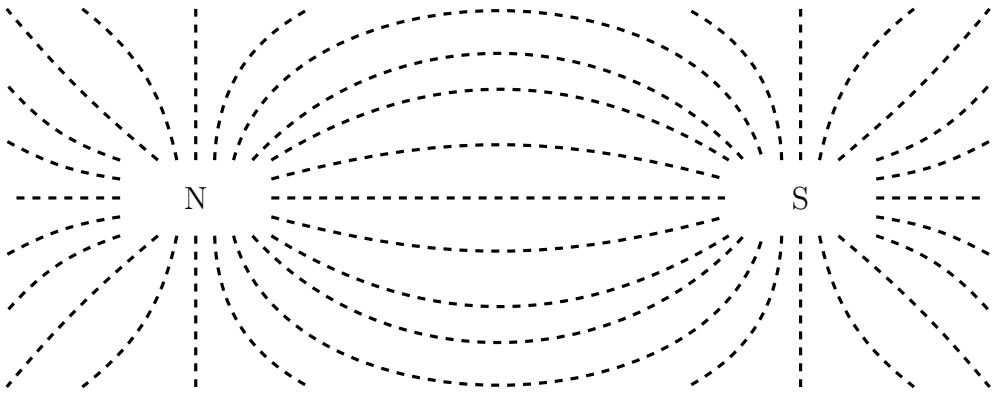


Figure 2.4: Example of a ferromagnetic field. Iron filings (depicted as black dashes) tend to align to magnetic poles with respect to their neighborhood.

where $\mathrm{V}_1(\mathrm{f}_i)$ is the clique potential at site i and $\mathrm{V}_2(\mathrm{f}_i, \mathrm{f}_{i'})$ the clique potential of the two sites i, i' with $i' \in \mathcal{N}_i$. Let $\mathrm{V}_1(\mathrm{f}_i) = \mathrm{f}_i \mathrm{G}_i(\mathrm{f}_i)$ for an arbitrary function $\mathrm{G}_i(\cdot)$ and let $\mathrm{V}_2(\mathrm{f}_i, \mathrm{f}_{i'}) = \beta_{i,i'} \mathrm{f}_i \mathrm{f}_{i'}$ for a constant $\beta_{i,i'}$ reflecting the pair-site interaction between i and i'. The energy then takes the form

$$\mathrm{U}(\mathcal{F}) = \mathrm{f}_i \mathrm{G}_i(\mathrm{f}_i) + \beta_{i,i'} \mathrm{f}_i \mathrm{f}_{i'} \quad . \tag{2.34}$$

Models in this general form are termed *auto-models* by Besag [Bes74]. An auto-model is said to be an *auto-logistic model* if every f_i takes a value in the discrete label set $\mathcal{L} = \{0, 1\}$ (or $\mathcal{L} = \{-1, +1\}$). The corresponding energy takes the form

$$\mathrm{U}(\mathcal{F}) = \sum_{\{i\} \in \mathcal{C}_1} \alpha_i \mathrm{f}_i + \sum_{\{i,i'\} \in \mathcal{C}_2} \beta_{i,i'} \mathrm{f}_i \mathrm{f}_{i'} \quad , \tag{2.35}$$

where $\beta_{i,i'} \mathrm{f}_i \mathrm{f}_{i'}$ can be seen as the interaction coefficient between neighboring terrain cells. When $\mathcal{N}$ is the nearest neighborhood system on a grid (4 neighbors in the terrain grid), the auto-logistic model is reduced to the *Ising model*. The Ising model is named after the German mathematician and physicist Ernst Ising ($\star$1900, †1998). It is given to Ising as research topic by his doctorate supervisor Wilhelm Lenz ($\star$1888, †1957), who invented it. Therefore, the model is sometimes referred to as the Lenz-Ising model. Ising uses the model to describe ferromagnetism in statistical mechanics to explain the coexistence and competition of ferromagnetic and anti-ferromagnetic interactions (cf. Figure 2.4) in solid bodies. The energy of the model is defined as *Hamiltonian*

$$\mathcal{H}_{\text{Ising}} = -\sum_{ij} J_{ij} S_i S_j - H \sum_i S_i \tag{2.36}$$

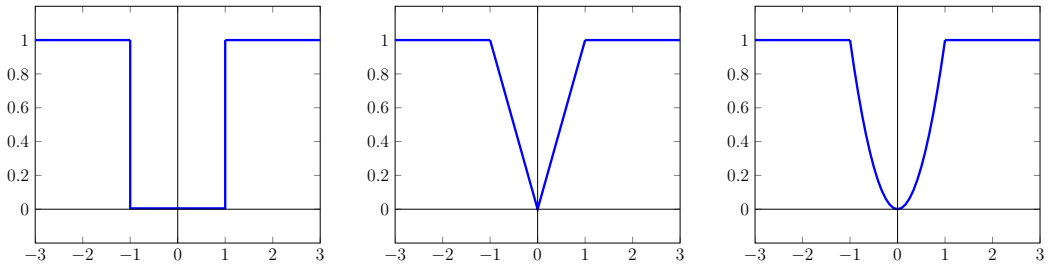


Figure 2.5: Examples of different smoothness cost functions. The basic Potts model is shown in (a), the Truncated linear model in (b), and the Truncated quadratic model is depicted in (c).

where $S_i = \pm 1$ is an Ising spin variable located at the i-th site and H is an external field intensity (a magnetic field). The identifiers i and j refer to sites and the sum over ij is taken over all nearest-neighbors J_{ij} on all sites of the grid, referred to as *coupling constant* in physics. Theoretical studies about the model are concerned with magnetic equilibrium called *spin glasses*. A spin glass is a disordered magnet where the signs are present equally often, which is called *spin symmetry*. The Ising model and the spin glass theory are active research topics (cf. [Kaw10, MG10]). Furthermore, the ideas and models regarding ferromagnetic to cooperative phenomena are applied to various other fields like biology or computer sciences (cf. [Mat81, SS12]).

In case of the terrain classification problem, the Ising model is not sufficient since the neighborhood interactions require more than the binary class relationship of ± 1. Therefore, the *Potts model* forms an extension of the Ising model for n-ary states. Several variations of the Potts model possess different properties, some of which are shown in Figure 2.5. The basic Potts model is defined as

$$\delta(l_1, l_2) = \begin{cases} -1 & \text{if } l_1 = l_2 \\ +1 & \text{else} \end{cases} \tag{2.37}$$

for two arbitrary labels $l_1, l_2 \in \mathcal{L}$ and is visualized in Figure 2.5(a). Its binary differentiation between being equal to label l_1 or not allows fast computations and makes it suitable for equally similar or dissimilar classes such as $\mathcal{L} = \{Building, Car, Tree\}$. The *Truncated linear model* shown in Figure 2.5(b) is well suited for classes that possess similarities like $\mathcal{L} = \{House, Cottage, Palm, Tree\}$ or for disparity values. It is called truncated since the maximum allowed positive return value is usually capped to control the penalty for very dissimilar classes. The *Truncated quadratic model* (cf. Figure 2.5(c)) has almost the same properties but behaves more tolerant of deviations between congruent classes.

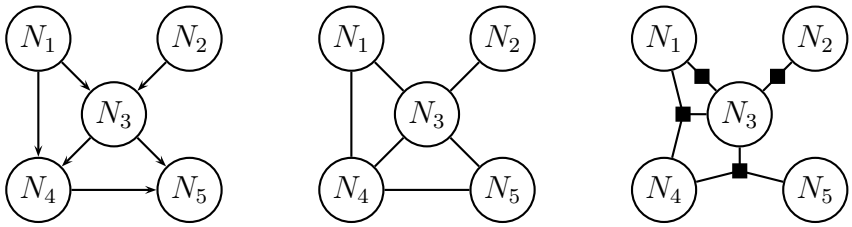


Figure 2.6: Examples of three different Probabilistic Graphical Models with five nodes. A Bayes network is shown on the left side next to a Markov random field in the middle. An example of a Factor graph is on the right side.

2.8 Probabilistic Graphical Models

MRFs belong to the framework of *Probabilistic Graphical Models* (PGMs) which arises from the combination of probability theory and graph theory. The framework spans famous methods such as Bayes networks, Markov random fields, and Conditional random fields. An example of different PGMs is shown in Figure 2.6. In the terminology of PGMs, a site is referred to as *node* N (e.g., $N_1, N_2, \dots$) and represents a random variable (e.g., $x_1, x_2, \dots$) or a vector of random variables (e.g., $\boldsymbol{x_1}, \boldsymbol{x_2}, \dots$). PGMs allow insights about dependence and independence within the graph by inspection. Bayes networks, for example, are also known as *probabilistic directed acyclic graphical models* and the presence of cycles can be directly spotted from the graph visualization. A Bayes network is suitable for modeling causal relationships between random variables, whereas an MRF is more suited to model loose neighborhood relations. Both PGMs can be transformed into a factor graph which is often used to model Conditional random fields. The factor graph representation allows a more detailed partitioning and enables a generalized representation by introducing additional factor nodes (depicted as black squares along the solid connecting lines in Figure 2.6(c)). For the modeling of an MRF as a factor graph, the factors would all be equal in the general case. MRFs also constitute a graph of multiple layers as shown in Figure 2.7. Nodes depicted in red represent the labels and are hidden variables. Their value, a terrain class, depends on the neighboring nodes which are directly linked to each other via a solid line and the corresponding observation node. Observation nodes are depicted in blue and represent terrain features computed from sensor data at the site they are connected to with a dotted line. The representation of an MRF as a graphical model provides an intuitive possibility to understand and model the properties of the random field. In the case of an MRF based on an equidistant regular grid, the modeling is feasible in a straightforward manner. Each site in the grid corresponds to a node in the graph and neighborhood relations are available from

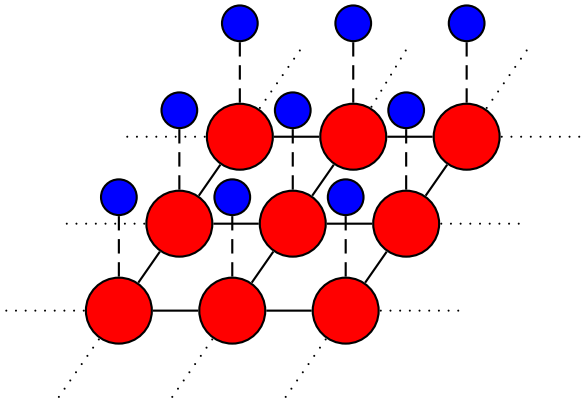


Figure 2.7: Markov Random Field visualization as a Probabilistic Graphical Model. Red nodes represent hidden variables (e.g., disparity values or terrain classes) and blue nodes represent observations (e.g., intensity values or terrain features). The connection of the observations to a site is illustrated by dashed lines and the neighborhood interaction is drawn as solid lines. A further extension of the field is indicated by the dots next to the nodes.

adjacent cells. While the MRF model using the grid structure is convenient and intuitive, the optimization is complex and needs to be well designed.

2.9 Optimization and Gibbs Sampling

Once the MRF is specified and data $\boldsymbol{D}$ are available, the optimal configuration $\mathcal{F}^*$ can be found according to Equation 2.7. As already stated before, a number of M^m configurations exists. Let $m = 40,000$ (e.g., a terrain grid of $100\,\mathrm{m} \times 100\,\mathrm{m}$ with a resolution of $50\,\mathrm{cm} \times 50\,\mathrm{cm}$ per terrain cell) and let $M = 5$ (e.g., some terrain classes). For data arriving at an exemplary frequency of 10 Hz, a brute force approach needs to compare $5^{40,000}$ configurations in 100 ms. Each of these configurations is calculated w.r.t. a cost function and the data $\boldsymbol{D}$ usually consists of features which need to be extracted within the same period of time.

A number of acceleration and sampling techniques exist to approach such complex optimization tasks. These approaches try to find the optimal configuration, for example, by changing a large number of labels simultaneously or utilizing sophisticated sampling techniques. This can be done by computing a minimal cut though a graph to achieve a maximal flow (max-flow min-cut theorem). Graph cuts [GPS89] provide an approach to this task for a source and a sink, which limits the algorithm to binary classification tasks. Approaches that extend graph cuts for more than two labels are e.g., the *alpha expansion* or the *alpha-beta swap* algorithms [BVR01]. The idea of these algorithms is to swap the labels of large homogeneous regions at once, yielding an acceleration of runtime.

Sampling strategies are another widespread approach to the optimization problem. A number of samples is drawn under an algorithm-specific set of rules

Algorithm 1 Gibbs sampling

initialize $s_1, \ldots, s_m$
for $\tau = 1, \ldots,$ MaxIteration **do**
  Sample $s_1^{(\tau+1)} \sim p(s_1 \mid s_2^{(\tau)}, s_3^{(\tau)}, \ldots, s_m^{(\tau)})$
  Sample $s_2^{(\tau+1)} \sim p(s_2 \mid s_1^{(\tau+1)}, s_3^{(\tau)}, \ldots, s_m^{(\tau)})$
  $\vdots$
  Sample $s_j^{(\tau+1)} \sim p(s_j \mid s_1^{(\tau+1)}, \ldots, s_{j-1}^{(\tau+1)}, s_{j+1}^{(\tau)}, \ldots, s_m^{(\tau)})$
  $\vdots$
  Sample $s_m^{(\tau+1)} \sim p(s_M \mid s_1^{(\tau+1)}, s_2^{(\tau+1)}, \ldots, s_{m-1}^{(\tau+1)})$
end for
return $\mathcal{F}$

to determine the outcome of configurations that are (almost) impossible to calculate. Various approaches exist that introduce genuine sampling strategies, e.g., by discarding unusable but quickly calculable calculate samples or by learning from past samples. Another way to accelerate the process is to interrupt the optimization algorithm and to accept a solution that is close to the optimal solution but not yet entirely correct.

Another very interesting sampling strategy is the Gibbs sampling, named after the American physicist Josiah Willard Gibbs (⋆1839, †1903), and is described by Stuart and Donald Geman in 1984 [GG84]. Gibbs sampling is applicable for problems with at least two dimensions and preferably used for high-dimensional problem spaces. Josiah Gibbs' idea, generally speaking, is based on the divide-and-conquer principle. Instead of sampling the next state all at once for the entire problem, the algorithm divides the problem space and samples each dimension separately. In textbooks, the behavior of the algorithm is sometimes illustrated as a snake or spiral which approaches the optimal configuration with steps from different courses, getting closer and closer with every new iteration. Let τ denote the current iteration of the algorithm and m is the amount of sites (Equation 2.1). For each τ during each iteration, each sample that is drawn is either from the previous iteration or from the current iteration. Considering the order during each iteration, arbitrary (e.g., random) or particular (e.g., sequential) drawings are possible. The example shown in Algorithm 1 follows a straightforward sequential order. During this process, new values for variables are used as soon as they are obtained. Here, $s_j^{(\tau)}$ denotes a sample drawn at site j from the previous iteration (an old sample), respectively the initialization, and $s_j^{(\tau+1)}$ denotes a sample from the current iteration (a new sample). For the example in Figure 2.8 one iteration from τ to $\tau + 1$ consists of the following steps:

(a) Initialization of $s_1 \ldots, s_9$.

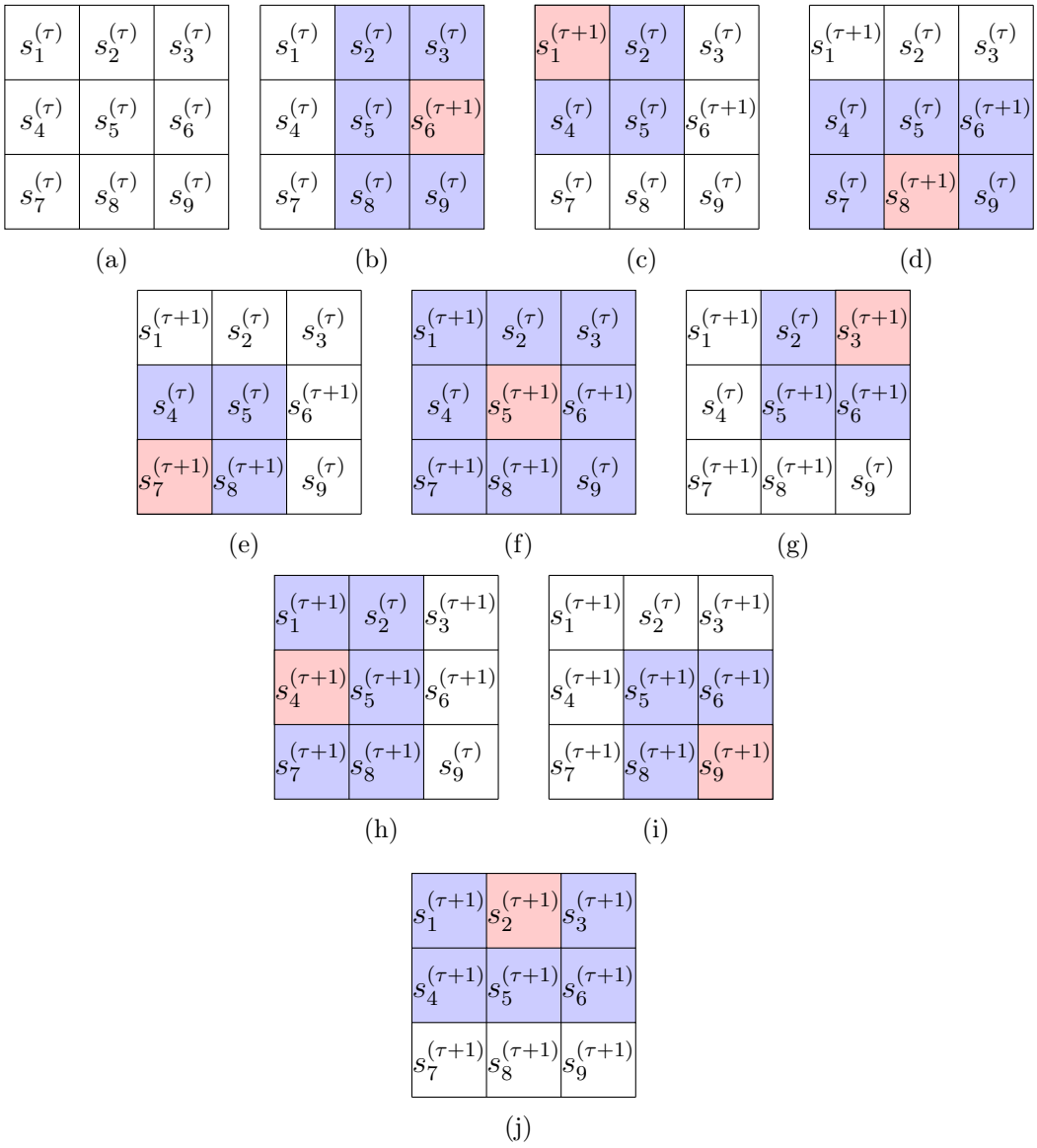


Figure 2.8: Exemplary iteration of Gibbs sampling. A set of nine labels is linked to a 3×3 grid with a second-order neighborhood system (cf. Figure 2.1 (b)). The initialization of $s_1 \ldots, s_9$ is shown in (a) and the single iteration steps are depicted in (b)-(j). A grid cell shaded in red means that this is the current sample which is drawn and grid cells shaded in blue represent cells that belong to the neighborhood of this cell.

(b) A new sample is drawn for s_6 as $s_6^{(\tau+1)} \sim p(s_6 \mid s_2^{(\tau)}, s_3^{(\tau)}, s_5^{(\tau)}, s_8^{(\tau)}, s_9^{(\tau)})$. For the given second-order neighborhood system, $s_6^{(\tau+1)}$ is drawn w.r.t. five neighbors from the initial respectively the previous iteration.

(c) A new sample is drawn for s_1 according to $s_1^{(\tau+1)} \sim p(s_1 \mid s_2^{(\tau)}, s_4^{(\tau)}, s_5^{(\tau)})$.

(d) A new sample is drawn for s_8 as $s_8^{(\tau+1)} \sim p(s_8 \mid s_4^{(\tau)}, s_5^{(\tau)}, s_6^{(\tau+1)}, s_7^{(\tau)}, s_9^{(\tau)})$. Note that four neighbors are from the previous iteration and one neighbor, $s_6^{(\tau+1)}$, is already from the new iteration meaning that is has been sampled shortly before and is immediately used in the same iteration.

(e) A new sample is drawn for s_7 according to $s_7^{(\tau+1)} \sim p(s_7 \mid s_4^{(\tau)}, s_5^{(\tau)}, s_8^{(\tau+1)})$. Samples $s_4^{(\tau)}$ and $s_5^{(\tau)}$ are old and $s_8^{(\tau+1)}$ is newly sampled.

(f) A new sample is drawn for s_5 w.r.t. eight neighbors, which would be the common case for a larger grids. The sample is drawn according to $s_5^{(\tau+1)} \sim p(s_1 \mid s_1^{(\tau+1)}, s_2^{(\tau)}, s_3^{(\tau)}, s_4^{(\tau)}, s_6^{(\tau+1)}, s_7^{(\tau+1)}, s_8^{(\tau+1)}, s_9^{(\tau)})$. Four old and four new samples are taken into account.

(g) A new sample is drawn for s_3 according to $s_3^{(\tau+1)} \sim p(s_3 \mid s_2^{(\tau)}, s_5^{(\tau+1)}, s_6^{(\tau+1)})$. $s_2^{(\tau)}$ is from the last iteration, the other two are already newly sampled.

(h) A new sample is drawn for s_4 w.r.t. five neighbors according to $s_4^{(\tau+1)} \sim p(s_4 \mid s_1^{(\tau+1)}, s_2^{(\tau)}, s_5^{(\tau+1)}, s_7^{(\tau+1)}, s_8^{(\tau+1)})$. From these five neighbors, only one is from the previous iteration.

(i) A new sample is drawn for s_9 according to $s_9^{(\tau+1)} \sim p(s_9 \mid s_5^{(\tau+1)}, s_6^{(\tau+1)}, s_8^{(\tau+1)})$. For the first time in this sampling iteration, all samples in the neighborhood are from the current iteration.

(j) A new sample is drawn for s_2 w.r.t. five neighbors according to $s_2^{(\tau+1)} \sim p(s_2 \mid s_1^{(\tau+1)}, s_3^{(\tau+1)}, s_4^{(\tau+1)}, s_5^{(\tau+1)}, s_6^{(\tau+1)})$. This is the last sample for the current iteration. Depending on the overall progress, the resulting samples may either be the final configuration due to some abortion criterion or because the optimal configuration is found or resulting samples serve as initialization for the next iteration.

The principle of the algorithm and it's effectiveness make it an attractive problem-solving approach for many application domains. Other advantages of the algorithm are its flexibility, as the sample drawing can be manipulated in many ways, and a lot of useful abortion criteria exist for this algorithm: maximum number of iterations; thresholding in advance between two iterations; desired closeness to optimal configuration, etc. Besides all its advantages, the algorithm also suffers from a so-called *burn-in phase*. This phase describes the first iterations

of the algorithm before samples are drawn from the stationary distribution. All applications have to pass through this phase which yields a negative impact on the runtime of the algorithm. Furthermore, samples from a late iteration may be contaminated by the values of early samples (from the burn-in phase), which is why in some cases it might be expedient to discard values from early iterations later and to re-sample them. For example, a burn-in threshold can be applied which takes effect after a certain number of iterations and discards all old samples to prevent contamination.

At this point, all basic requirements for understanding and solving the terrain classification problem have been explained and several examples in this section already introduced solutions for small grids. Additional literature exists on almost any of the presented basics, enhancing them or viewing them from another point of view. For example, Li [Li09] also describes the basics for continuous cases and presents the derivations of some of the statements described here for the discrete cases. A multitude of solutions is available for optimization. Bishop, for example, covers a variety of approaches in his book [Bis06] but algorithm collections also exist and are published in the web, in the form of software libraries (e.g., cf. [14]).

2.10 The Mahalanobis Distance

Another basic for this thesis is the *Mahalanobis distance*. Although it is not directly related to MRF theory, it will be frequently used in the successive chapters and provides a powerful tool for outdoor sensor data interpretation. The Mahalanobis distance is named after the Indian physicist Prasanta Chandra Mahalanobis (⋆1893, †1972). It measures the effect size (the statistical relative size) of unknown samples to a known set. The Mahalanobis distance D^2 is a unitless measure, scale-invariant, and is defined as

$$D^2 = (\boldsymbol{D} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\boldsymbol{D} - \boldsymbol{\mu}) \quad , \tag{2.38}$$

where $\boldsymbol{D}$ is the vector of data, $\boldsymbol{\mu}$ is the vector of mean values, and $\boldsymbol{\Sigma}$ is the covariance matrix.

For a simple example, consider a multivariate matrix $\boldsymbol{X}$ of multiple observations as

$$\boldsymbol{X} = \begin{pmatrix} -1.1 & 1 & 0.2 & -0.15 & -1 & -0.3 & 0.4 & 0.3 & 0.1 & 0.8 & -0.5 \\ -0.8 & 1 & 0.23 & -0.5 & 0.4 & -0.23 & 0.51 & 0.4 & -1.5 & 0.9 & -0.7 \\ 0.9 & -0.1 & 0.01 & -0.2 & 0.32 & -0.31 & -1 & 0.5 & -0.43 & -0.2 & \\ -0.3 & -0.21 & 0.12 & -0.1 & -0.13 & -0.01 & -1.1 & 0.5 & -0.35 & 0.6 & \end{pmatrix}$$

where the two rows represent two independent variables and the columns represent 21 observations of these. Observations are visualized as red dots in Figure 2.9 (a) in a two-dimensional plot, one axis for each variable.

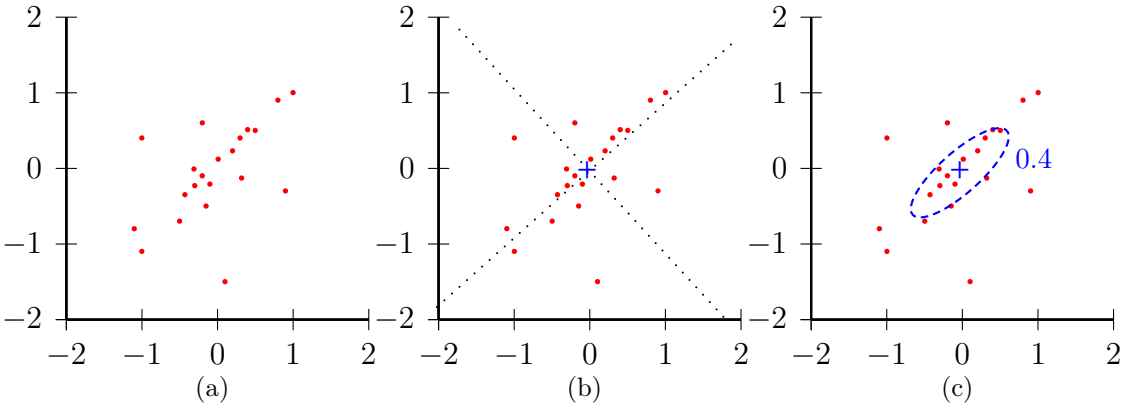


Figure 2.9: Example for the Mahalanobis distance. A set of random 2D samples is shown as red points in (a). The centroid of the samples is shown as a blue cross together with the two primary axes at (b). An example of a dashed blue ellipse for a Mahalanobis distance of 0.4 is shown in (c).

Given these data, the first step is to compute the mean vector and the covariance matrix (variance-covariance matrix). The mean vector $\boldsymbol{\mu}$ for $\boldsymbol{X}$ is

$$\boldsymbol{\mu} = \begin{pmatrix} -0.036 \\ -0.06 \end{pmatrix} \quad . \tag{2.39}$$

Covariance of two variables x and y is given by

$$cov = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \boldsymbol{\mu}_x)(y_i - \boldsymbol{\mu}_y) \quad , \tag{2.40}$$

where $\boldsymbol{\mu}_x$ and $\boldsymbol{\mu}_y$ denote the means of x and y respectively and yields the covariance matrix $\boldsymbol{\Sigma}$

$$\boldsymbol{\Sigma} = \begin{pmatrix} 0.858\ 114 & 1.065\ 1 \\ 1.065\ 1 & 1.851\ 86 \end{pmatrix} \quad . \tag{2.41}$$

The mean vector consists of the means of each variable and forms the centroid of the samples. In Figure 2.9 (b), the centroid is shown as a blue cross and the primary axis as dotted lines. The covariance matrix consists of the variances of the variables along the main diagonal. Covariances between each pair of variables populate the matrix at the other positions. In this simple 2D example, regions of constant Mahalanobis distances form a 2D ellipse around the centroid as illustrated by the dashed blue ellipse in Figure 2.9 (c) for a fixed Mahalanobis distance of 0.4. For more complex data, ellipsoids or hyperellipsoids replace ellipses.

Since the Mahalanobis distance is a unitless and scale-invariant similarity measure for multivariate data, it is very useful in robotics and can be applied to multiple problems in the following parts of this thesis.

The next chapter presents the software architecture, the underlying system, and how the terrain classification module interacts with other modules before the MRF terrain classification is described in Chapter 4.

Chapter 3

Software Architecture

In order to understand how the software components that will be described in the following chapters interact with each other, this chapter gives a brief overview of the underlying software architecture. The architecture used for this thesis was developed over several years and the following descriptions were previoulsy presented as joint work in [Thierfelder et al., 2011a], mainly written by Susanne Thierfelder and Viktor Seib. The software was designed and improved in previous robot projects to fulfill various complex tasks for heterogeneous autonomous robots. Robots find their way into numerous of application domains, most of them requiring a sophisticated software in order to fulfill different complex tasks in short periods of time. Many different robot architectures which focus on various scenarios have emerged over the past decade.

The *OROCOS* project ([Bru01],[4]) wishes to become an open and general-purpose robot control software package. Within the project, code is divided into different types of modules to manage the complexity of big software projects. The success of OROCOS will depend on the contributing researchers and engineers.

Miro [USEK02] is an object-oriented robot middleware for mobile robot applications. It was designed with the focus on multi-platform support and interoperability and can be applied to heterogeneous robot platforms.

An open-source component-based software engineering framework called *Orca* is proposed for mobile robotics by Brooks et al. [BKM+05]. The framework focuses on reusable components for mobile robotic systems in indoor and outdoor scenarios.

Farinelli et al. present *SPQR-RDK* [FGI05], a robot programming environment specifically designed for multi-platform usage. SPQR-RDK provides an integrated framework with technical prerequisites for remote control, inspection, and information sharing.

Player [CMG05] is a robotic framework introduced by Collett et al. The framework is an open source project and focuses on simplicity and flexibility.

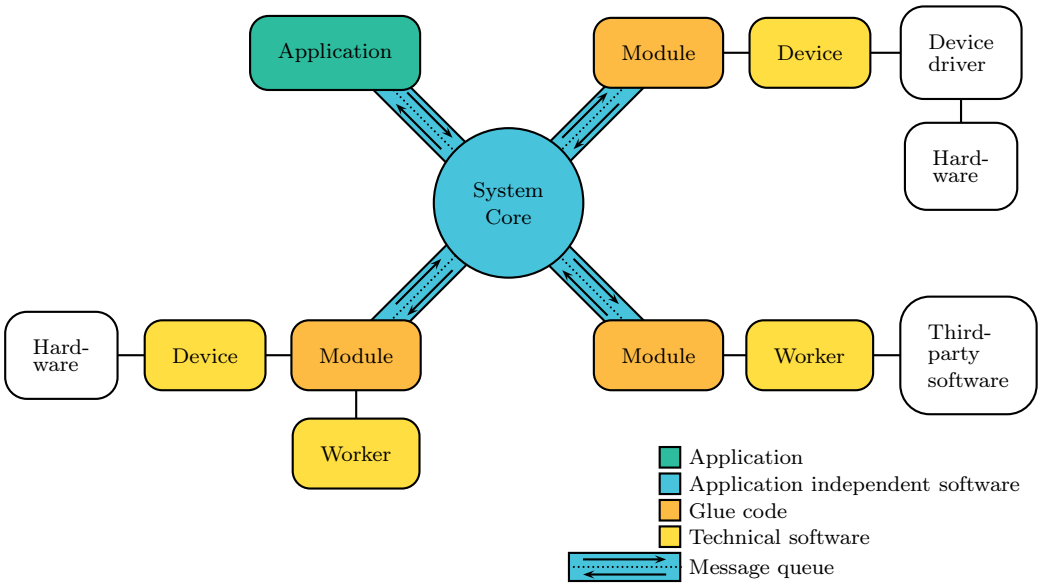


Figure 3.1: The architecture concept. Modules act as glue code and communicate with each other via messages distributed by an application independent system core. Further components such as devices or workers are attached to these modules to include hardware and algorithms. Image after [Thierfelder et al., 2011a].

Ando et al. propose the *OpenRTM-aist* [ASK+05] middleware as a software platform for robot system integration. Within OpenRTM-aist, components can be combined and handled as black boxes in order to construct complex systems.

A middleware framework called *MARIE* that uses a generic communication framework is presented in [CBL+06]. MARIE aims to create a flexible distributed component system that allows developers to design and share prototypes rapidly.

The robotic software framework *CLARAty* [Nes07] has been primarily developed by the Mars Technology Program for integrating robotic technologies from different programs and for deployment on NASA's research rover fleet.

A modular framework called *OpenRDK* focuses on rapid development of distributed robotic systems and is presented in [CCIN08]. OpenRDK's features include a multi-threaded multi-processes structure and a blackboard-type inter-module communication and data sharing.

A *Robot Operating System* (*ROS*,[22]) is presented by Quigley et al. [QCG+09]. ROS aims at simplifying robot software development by proving a modular, tools-based development environment.

The strong proliferation and progress of robotic middleware and framework architectures over the past decade proves the importance of these topic. Especially ROS and other projects show that the research in these areas is alive and progressing in various directions.

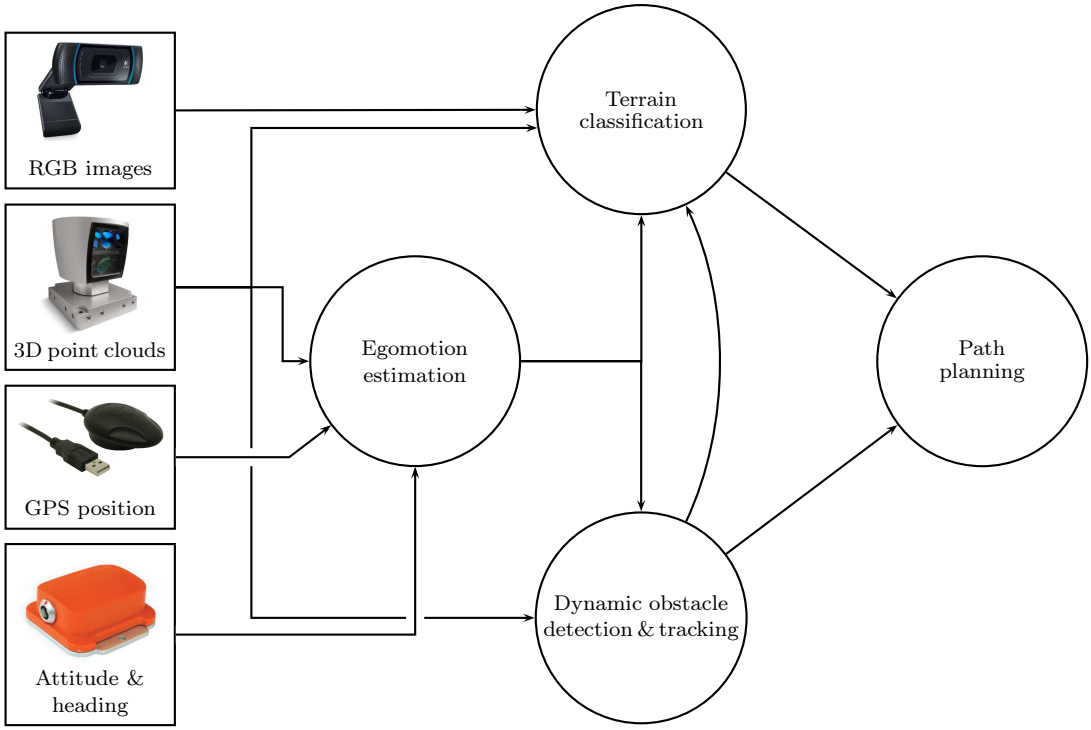


Figure 3.2: Data flow and overview of the different modules. Sensor data are processed by different modules which in turn deliver data for other modules. Arrows indicate message flow in the software.

The underlying software architecture for this thesis has been developed and advanced by the Active Vision Group and follows a strictly message-based software design. Source code is distributed over various independent modules that are connected to the system. Concurrency is achieved by executing every module in its own thread with isolated program blocks encapsulating a specific task. The center of the architecture is a generic, application-independent system core that serves as a dispatcher between connected components. Inter-module communication is realized via this system core by subscribing to messages relevant to their specific task, keeping modules independent from each other. All data that needs to be transferred between modules is encapsulated into messages. If a module wants to share information with another module, it has to send a message to the system core which delegates it to all subscribers of the message, following a loosely coupled design. The concept of the system [Pel11] is illustrated in Figure 3.1 and has its origins in the *Quasar windmill* [Sie02]. As indicated in the image, modules connect devices (hardware) and workers (algorithms) with the system core and thus act as glue code. Devices communicate with a certain piece of hardware of the robot and workers are small sets of program code which implement specific algorithms. Another invaluable property is the capability of

messages to serialize and deserialize themselves, allowing algorithm development and evaluation on recorded sensor data.

An overview of the different components of this thesis is given in Figure 3.2. As illustrated in the image, the 3D point clouds of the LRF are used for terrain classification, egomotion estimation, and dynamic obstacle detection and tracking. Camera images are used for terrain classification only and GPS and IMU data are subscribed by the egomotion estimation module. The terrain classification module receives incoming data from the sensors as well as from the modules responsible for the egomotion estimation and the detection and tracking of dynamic obstacles. Furthermore, the path planning module regards static obstacles and the terrain surface as well as trajectories of other dynamic obstacles as it operates on the data delivered by these two modules.

The architecture proved its reliability and robustness on different robotic platforms (cf. [Vetter et al., 2010, Thierfelder et al., 2011a, Lang et al., 2011]) and in a large number of scenarios (cf. [Thierfelder et al., 2011b, Hahn et al., 2011, Lang et al., 2012]). A more detailed description of the design and the underlying decision-making process is depicted in [Pel11] and a precise description of this architecture is published in [Thierfelder et al., 2011b]. In the next chapters, the contents and algorithms of the specific modules relevant for terrain classification will be described in detail.

Chapter 4

Markov Random Field Terrain Classification

Mobile autonomous systems need a detailed interpretation of the surrounding terrain in order to avoid obstacles and to judge the drivability of the surface. The task of a terrain classification algorithm is to provide a data representation and classification that satisfies these requirements in real-time. 3D point clouds produced by modern LRFs like the Velodyne HDL-64E and HDL-32E provide a rich and thorough picture of the environment in the form of 3D distance measurements. The vast amount of data of 3D LRFs cannot be directly used for path planning algorithms. Therefore, as a first step, a reduction of the huge point cloud is necessary and an efficient data representation is essential. Laser range measurements alone allow no differentiation between similar surfaces; therefore, color cameras are calibrated to the 3D LRF. Cameras allow for the determination of color and texture of the 3D points in the field of view of each camera and allow access to fused data in one coordinate system in case all sensors are calibrated to each other. The goal of the terrain classification is to determine the drivability of the surrounding terrain with an MRF in real-time based on the fused sensor data.

The following chapter starts with a discussion of related work in the field of terrain classification and MRFs in Section 4.1. Fusion of camera and LRF data is described in Section 4.2, followed by the application of the MRF in Section 4.3. Experimental results are presented in Section 4.4. Integration of egomotion is briefly explained in Section 4.5. An additional and elaborate evaluation in another domain, the 3D mapping, is demonstrated in Section 4.6. The chapter concludes in Section 4.7 with a proof of concept in the form of a path planning algorithm operating on the results of the terrain classification.

4.1 Related Work

MRFs are frequently used in the field of image segmentation, especially for segmentation tasks. Szirányi et al. [SZC+00] address the problem of the large amount of computing power required for Markovian approaches. They introduce a parallel architecture for MRF segmentation of images and show that the Markovian labeling approach can be implemented in fully parallel cellular network architectures.

Meas-Yedid et al. [MYTOM02] use an MRF clustering approach for color segmentation based upon color and spatial information. Their approach proves to be robust against noise, color changes, illumination changes, and blurring during the performed experiments.

An MRF image segmentation model that combines color and texture features is presented by Kato and Pong [KP06]. Segmentation is obtained by classifying pixels into different pixel classes, which are represented by multi-variate Gaussian distributions either computed from training data or estimated from the input image.

Qazi et al. [QAB+11] present a segmentation methodology with robust parametric approximations proposed for multichannel linear prediction error distribution. They use an energy term based on region size with the conventional Potts energy model (cf. Section 2.7) and present improved results in terms of percentage errors of color texture segmentation for high-resolution multispectral satellite images.

D'Angelo and Dugelay [dD10] provide an MRF description of an unsupervised color image segmentation algorithm. Their system is based on a color quantization of the image in the Lab color space and uses a fuzzy k-nearest neighbors algorithm.

Besides color images, RGB-D sensors such as the Microsoft Kinect allow us to record color, texture, and depth information in one data set. Herbst et al. [HRF11] use an RGB-D camera and apply a multi-scene MRF model to detect objects that moved between multiple visits to the same scene. By combining shape, visibility, and color cues they are able to detect objects without texture within the scenes.

An MRF that integrated high-resolution image data into low-resolution range data is presented by Diebel and Thrun [DT05b]. Their MRF exploits the fact that discontinuities in range and coloring tend to co-align and recovers the range data at the same resolution as the image data.

Various approaches exist to classify the terrain surrounding an autonomous mobile robot platform. Especially image- or laser-based strategies are often used when terrain drivability information is needed.

Image-based strategies either use a single, stereo or combined setup of digital and infrared cameras. Konolige et al. [KAB+06] and Alberts et al. [AES+08] both use stereo vision approaches to maneuver a vehicle through unstructured

environments. Stereo vision allows them to extract drivable regions from the camera video streams. Furthermore, Vernaza et al. [VTL08] present a camera-based terrain classification approach for the *DARPA LAGR* program. Their approach uses an MRF that classifies image data of a stereo system into obstacles or ground regions for an autonomous robot. Khan et al. [KKBZ11] present a comparison of multiple approaches to using color-cameras for terrain classification for outdoor mobile robots based on local features. Their approach uses random forests for classification and is able to perform in various weather conditions.

A *negative obstacle* is a non-drivable region underneath the ground level and represents a difficult challenge in non-urban environments. Thermal infrared images have the characteristic that negative obstacles remain warmer than the surrounding terrain in the night. Rankin et al. [RHM03] therefore combine thermal signatures and stereo range data in order to determine the terrain drivability. Morton and Olson [MO11] describe a terrain classifier for detecting both positive and negative obstacles. Their classifier utilizes the change in height and the distance between observations from adjacent surfaces as well as the point density in discretized cells.

Laser-based approaches either work with a 2D LRF, a 2D LRF on stepper motors, or a 3D LRF. Wurm et al. [WKSB09] use the laser remission values of a 2D LRF on a pan-tilt unit to classify the surface terrain based on the resulting 3D scans. In this way, they detect grass-like vegetation but prefer paved routes with their robot. Another approach for terrain classification is presented by Wolf et al. [WSFB05]. Their robot uses a 2D LRF which is oriented to the ground, records data while driving, and produces 3D maps using hidden Markov models. The authors are able to differentiate flat areas from grass, gravel, or other obstacles. In comparison to the approaches of Wurm et al. [WKSB09] and Wolf et al. [WSFB05], the Velodyne HDL-64E and HDL-32E produce substantially larger point clouds that need an even faster processing.

Vandapel et al. [VHKH04] segment 3D distance measurements and classify the segments into three different classes: terrain surface, clutter, and wires. Their approach works with a stationary 3D sensor, which returns detailed data, as well as on a mobile platform with a rotating 2D scanning mount.

Ye and Borenstein [YB03, YB04] present an algorithm for terrain mapping with a 2D LRF. Their LRF is mounted at a fixed angle to the ground in front of their robot and creates an elevation map while driving.

A color stereo camera and a 2D LRF are used by Manduchi et al. [MCTM04] to detect obstacles. The authors present a color-based classification system and an algorithm that analyses the laser data in order to discriminate between grass and obstacles.

Schenk and Csatho [SC07] fuse aerial images with 3D point clouds to construct surfaces of urban scenes. Surfaces are represented in a 3D object space coordinate system by patches that store the shape and the boundary of the corresponding surface region.

A stereo pair of digital cameras, an infrared camera, and two 2D LRFs on scanning mounts are used by Wellington et al. [WCS05]. The authors apply multiple MRFs that interact through a hidden semi-Markov model that enforces a prior on vertical structures. Their results show that including the neighborhood structure significantly improves obstacle classification.

Another approach to terrain classification is based on judging surface conditions from the vibrations caused by driving on different surfaces. Brooks et al. [BID05] present a method to classify terrain based on vibrations measured by an accelerometer. Their algorithm is able to distinguish between sand, gravel, and clay in real-time.

A Bayes filter is employed by Komma et al. [KWZ09] to predict terrain classes from a history of vibration signals. The authors observe that only an adaptive approach which automatically adjusts its parameters is reactive enough to detect both, high-frequent and low-frequent terrain class changes.

Coyle et al. [CCR11] present another approach to reaction-based terrain classification for the detection of sand, grass, gravel and asphalt. The authors present a method of interpolating point clouds that uses singular value decomposition (SVD), matrix logarithms, and Catmull-Rom splines to reduce the need to collect large data sets for algorithm training.

In contrast to vibration-based approaches, the terrain classification algorithm described in the subsequent chapters is able to perceive the surface conditions *before* the robot actually drives on the terrain. Considering regions like fields or grasslands with potholes, rain grooves, and rough surfaces, this is a major advantage over vibration-based approaches.

Another research topic covers the simultaneous localization and mapping (SLAM) problem, where LRFs are used to process the 2D and 3D distance measurements to build maps [GNB00, MT06] of the environment. In contrast to mapping, terrain classification needs to process the sensor data as fast as possible in real-time on the robot for a path planning algorithm to directly access data and perform tasks autonomously.

Many approaches related to autonomous navigation need precise pose predictions in order to produce accurate and robust results. Cameras, LRFs, GPS receivers and IMUs are, among others, commonly used for egomotion estimation.

Lu and Milios [LM97] present two iterative algorithms to register a 2D range scan in relation to a previous scan to compute relative robot positions in unknown environments. Their first algorithm matches data points with tangent directions in two scans and minimizes a distance function. Their second algorithm establishes correspondences between points in the two scans and then solves the point-to-point least-squares problem to compute the relative transformation between the two scans.

Eggert et al. [ELF97] presented a comparative analysis of four algorithms that compute the 3D rigid body transformation aligning two sets of 3D points. The four algorithms are evaluated in terms of accuracy and robustness, stability with

respect to degenerate data sets, and relative computation time under different conditions.

Droeschel et al. [DMH+09] present an approach to estimate the egomotion of a robot while moving. Their approach uses the coherence of depth and reflectance data of a Time-of-Flight camera and fuses the resulting motion estimate with data from an IMU in order to improve accuracy and robustness against distorted measurements.

Ohno et al. [OTS+03] are concerned with map-based outdoor navigation near buildings and use an extended Kalman filter to fuse odometry and Differential Global Positioning System (DGPS) measurement data. The authors present two correction methods to correct DGPS position and odometry errors.

Kim et al. [KKOO07] present a localization method using different sensors with bias data. An extended Kalman filter integrates odometry and DGPS data from their robot in a global coordinate frame.

An adaptive Kalman filter for GPS data processing based on the observation of residuals in real-time outdoor applications is presented by Reina et al. [RVNY07]. Their approach is based on a fuzzy indicator to define a scale factor for the predicted covariance matrix by observing the size of the residuals.

A sigma-point Kalman filter is used for integrated navigation by van der Merwe and Wan [vdMW04] and fuses GPS and IMU data.

Lamon and Siegwart [LS04] present a method for combining dead reckoning sensor information in order to provide an initial estimate of the six degrees of freedom of a rough terrain rover. The rover's wheel encoders and an inertial navigation system are fused in an extended information filter.

An extended Kalman filter presented by Voigt et al. [VNH+11] fuses binocular visual measurements and inertial cues for egomotion estimation of an aerial vehicle. Their approach relies on inertial data if visual feature constellation is degenerate and enables pose estimation at high frame rates.

Terrain classification in outdoor environments comprises multiple research areas that represent active research topics. The research areas described in this section are only a selective choice of publications and approaches. A successful terrain classifier needs to regard all aspects of these topics in order to perform a precise and fast classification. As a first step, a calibration is necessary to provide fused color camera and 3D LRF data in the same coordinate system.

4.2 Camera and Laser Data Fusion

Before the MRF is described, the fusion of the camera and laser data is explained in this section. Fusing these two sensor modalities requires all attached sensors to be registered in a joint reference frame. Position and orientation information is acquired using an extrinsic calibration.

4.2.1 Related Work

Calibration of cameras with LRFs is another active research topic in robotics. These approaches are distinguished by the type of the camera, e.g., pinhole or omnidirectional cameras, and the type of the LRF, e.g., 2D or 3D LRFs.

A comparison of four different calibration techniques for 3D range data and camera images is given by Cobzaş et al. [CZJ02]. The techniques cover point and line based retrieval of the rigid transformations and image based mapping between the data sets.

Zhang and Pless [ZP04] present a method for manually calibrating a 2D LRF with a camera from multiple views. Their approach registers the laser points on a planar checkerboard pattern with the camera image of the pattern.

Unnikrishnan and Herbert [UH05] extend the approach of Zhang and Pless for the calibration of a 3D LRF with a camera. They extract a plane from four manually selected vertices on the checkerboard pattern and register it with the automatically detected pattern of the image.

The method of Unnikrishnan and Herbert works similarly to the approach presented by Andreasson et al. [ATL07]. Andreasson et al. design a new calibration pattern that consists of a common checkerboard framed by gray duct tape which makes the frame detectable in the intensity values of their LRF.

Leonard et al. [LBH+07] present a method for multi-sensor calibration in their technical report of the DARPA Urban Challenge. Besides various other sensors, cameras and LRFs are also calibrated. They use a calibration object in the form of a pyramid with an attached camera that can detect fiducials on their vehicle to determine the position of the pyramid. The result is validated with a Computer-aided design (CAD) model of the vehicle.

A method to calibrate a camera to a 2D LRF is described by Caglioti et al. [CGM08]. Their calibration is based on the algorithm introduced by Colombo et al. [CCdB06] and uses coaxial circles. A calibration object is not required, but the laser beams need to be visible in the camera images.

Nŭñez et al. [NDJRD09] present a camera to 3D LRF calibration based on the approach of Horaud and Dornaika [HD95]. The authors use a squared checkerboard rotated by 45 ° and move their sensor platform around it. Inferences about the motion is extracted from an IMU.

A stereo camera is calibrated to a 2D LRF by Li et al. [LLD+07]. Their specially built calibration pattern is a black triangle which they use to find corresponding lines in the camera image and in the laser data.

Aliakbarpour et al. [ANP+09] propose a method to calibrate a stereo camera with a 2D LRF with the help of an IMU. The approach is based on the calibration of Svoboda et al. [SMP05], where calibration is performed with a moving light source. Aliakbarpour et al. use a common laser pointer as the light source and retrieve the orientation of the LRF from the IMU.

Mei and Rives [MR06] present several methods for calibrating a 2D LRF with an omnidirectional camera. In order to estimate the position of the camera in relation to the LRF, the laser beam needs to be visible in the camera image. The authors consider LRFs with invisible laser beams and extend the approach of Zhang and Pless [ZP04] for omnidirectional cameras.

A calibration of an omnidirectional camera to a 3D LRF without a calibration pattern is presented by Scaramuzza et al. [SHS07]. Manual selection of correspondences in the laser data and camera image allows the computation of the rigid transformations between the selected points.

The calibration method introduced by Pandey et al. [PMSE10] uses a Velodyne HDL-64E with an omnidirectional camera. Therefore, the authors also extend the approach of Zhang and Pless [ZP04] and use calibration patterns affixed to floors and walls.

4.2.2 Camera Calibration and Camera to Laser Calibration

Calibration of multiple color cameras to a 3D LRF is a complex task. For the MRF terrain classification, calibration represents an expedient to realize a fused sensor reference frame. The data of the three color cameras of the robot, mounted on the left, right, and front sides of the robot, needs to be fused into the coordinate system of the LRF. Therefore, a C++ implementation of the approach presented by Unnikrishnan and Herbert [UH05] has been used for this thesis.

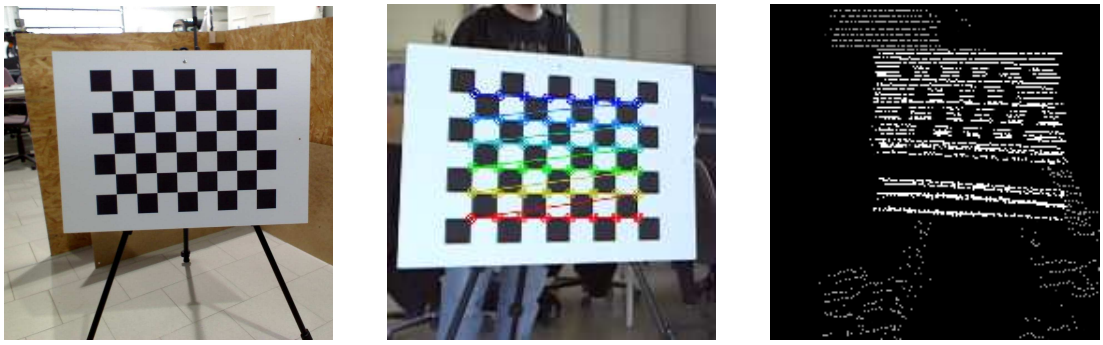
In the context of this fusion approach, multiple different calibration processes need to be performed subsequently to acquire the desired result. Initially, it is necessary to perform an intrinsic calibration separately for each sensor. This is a prerequisite so that the correctness of the particular sensor data is guaranteed for the following joint calibration.

Since the manufacturer Velodyne delivers calibrated 3D LRFs, a calibration by the customer is not required at the beginning. In the course of time, a recalibration of the sensor might become necessary. Vibrations to the sensor (e.g., caused by the robot driving over rough surfaces) are a plausible cause for a recalibration as well as different weather conditions, especially extreme heat and cold. The manufacturer offers the opportunity to send in the sensor for a recalibration at no charge. Since this requires shipping the sensor to the United States of America and the resulting absence of the sensor for multiple weeks, scientists and owners of a Velodyne LRF have developed their own intrinsic 3D LRF calibrations [GL10, PMSE10, AJGBHR+11, CCH+12, MKR12].Within the context of this thesis, the Velodyne HDL-64E was recalibrated by the manufacturer once, and the Velodyne HDL-32E was also been sent in once for a software update including advanced features. In addition, four lasers of the Velodyne HDL-64E became inoperative over time and have been disabled. Besides these hardware-related issues, a further recalibration of the intrinsic parameters of the 3D LRFs has not been necessary.

Intrinsic calibration of the cameras can be viewed with a few exceptions as a solved task. For the extraction of the camera parameters, the implementation of the camera calibration from Zhang [Zha00] and Bouguet [3] available in the Open Source Computer Vision [8] library is used.

After the application of the OpenCV camera calibration, intrinsic camera parameters are determined and calibration of the cameras to the 3D LRF is possible. The approach of Unnikrishnan and Herbert [UH05] provides a fast and feasible calibration. It enables a simultaneous calibration of the intrinsic camera parameters as well as the rigid transformation of the camera in relation to the LRF. For the calibration, a checkerboard is required and needs to be selected manually from the 3D data. Manual selection of the vertices for all three cameras takes several minutes since approximately 15 laser data/image pairs are required for a precise calibration. The approach requires an object that is both visible in the LRF and the camera data. On the one hand, Andreasson et al. [ATL07] point out that gray duct tape made their calibration frame detectable in the LRF data. On the other hand, teams of the DARPA challenges report that black vehicles disappear completely in the laser data. Based on these reports, various experiments with different material, surface structures, and colors, have been conducted. Grey duct tape and matt black metallic surfaces almost completely absorb the intensity data of the Velodyne HDL-64E and HDL-32E. Therefore, a calibration object made of a DIN A1 (841 mm×594 mm) aluminum plate with a matt black checkerboard fixed attached to a camera tripod was designed. The matt black checkerboard has the advantage that all calibration steps can be performed with a single checkerboard. For the calibration of the cameras to the 3D LRF, a large distance of at least 20 m from each camera to the LRF is necessary which is why a smaller checkerboard would be too small. A DIN A2 (594 mm×420 mm) checkerboard for example would yield an insufficient amount of distance measurements from the LRF and result in problematic detections for cameras with lower resolutions. The aluminum plate is weather resistant (e.g., against moisture humidity, extreme heat) and is easy to transport. A matt black surface has the advantage of being less susceptible against sunlight and other illumination influences in outdoor scenarios. Regarding the Velodyne HDL-64E and HDL-32E measurements, matt surfaces are distinguishable from white patches. This simplifies the calibration and lays the foundation for a completely automatic calibration based on reflectance values. Patch size and number can be estimated for the desired scenario w.r.t. the cameras or the desired maximum range of vision, respectively. Larger patches can be detected more easily farther away or with lower resolution cameras. The larger the patches, the lower the total amount of patches. A Larger number of detected patches yields a preciser calibration resulting in a quality loss if an insufficient number of patches is provided.

An image of the calibration object developed for this thesis is shown in Figure 4.1. It is dimensioned 841 mm × 594 mm and divided into 9 × 7 patches. While the calibration object is usable for both the intrinsic calibration of the



(a) (b) (c)

Figure 4.1: Views of the calibration object. The calibration pattern with the tripod is shown in (a). Camera readings with activated checkerboard detection from the point of view of one of the robots cameras is visualized in (b). Laser readings from the Velodyne HDL-64E are shown in (c) (after projection). The image (c) further depicts the typical missing horizontal readings on the calibration pattern where the two arrays of 32 lasers each of the Velodyne HDL-64E meet.

cameras and the camera to laser calibration, it turned out to be too small for some cameras. Especially for cameras with a low resolution, it is recommended to use an even larger calibration pattern of size DIN A0 (1189 mm × 841mm) in order to achieve the desired precision at larger distances. Another possibility is a modification of the calibration pattern's margin with a reflective surface. As observed by Andreasson [ATL07], reflective tape is visible to the LRF. During the experiments conducted with the calibration pattern developed for this thesis, it has been observed that reflectance values from gray duct tape are differentiable from other intensity values. A margin of gray duct tape therefore allows a margin extraction in the LRF measurements. Once the calibration object is available, the calibration is performed as follows.

Each camera requires an extrinsic calibration to assign it a position in the coordinate system of the LRF w.r.t. the coordinate system where the LRF is registered. The extrinsic parameters of each camera are given by the rotation matrix $\boldsymbol{R}$ and the transformation vector $\boldsymbol{t}$ w.r.t. a reference coordinate system. Under the premise that the 3D LRF is the center of the coordinate system, $\boldsymbol{R}$ and $\boldsymbol{t}$ of a point P_c from camera to laser coordinates are given by

$$\mathsf{P}_l = \boldsymbol{R}\,\mathsf{P}_c + \boldsymbol{t} \quad . \tag{4.1}$$

Let J be the set of pairs of laser scan and corresponding camera image. For each $J_i \in J$ one needs to find the orientation $(\boldsymbol{\theta}_{c,i},\boldsymbol{\theta}_{l,i})$ and the distance $(d_{c,i},d_{l,i})$ of the planes $\boldsymbol{\theta}_{c,i}\boldsymbol{x} - d_{c,i} = 0$ and $\boldsymbol{\theta}_{l,i}\boldsymbol{x} - d_{l,i} = 0$ to the camera w.r.t. the LRF.

The values $\boldsymbol{\theta}_{c,i}$ and $d_{c,i}$ are extracted from the extrinsic calibration of the camera with the help of OpenCV. In order to retrieve $\boldsymbol{\theta}_{l,i}$ and $d_{l,i}$, one has to locate

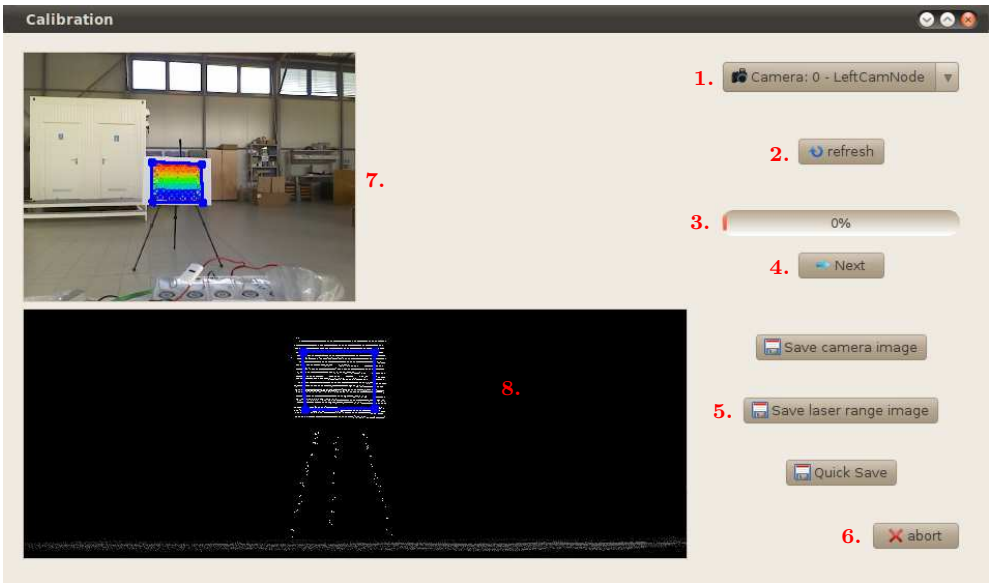


Figure 4.2: Visualization of the calibration process in the GUI. At the upper left side the camera image of the calibration pattern is shown and at the lower left side the scene from the LRF perspective. The control elements described in the text are numbered and shown on the right side.

the calibration pattern in the LRF data. The black patches create a checkerboard that is visible in the 3D intensity values (see Figure 4.1 (c)) as the LRF returns no values at these spots. Depending on which side of the LRF a camera is mounted, 50 % of the points on the averted side are discarded. Additionally, points that are too close to the LRF or which exceed the maximum range for calibration are culled. Afterwards, all remaining points are projected to a plane by discarding the depth value in the direction of the camera (front, left, right). This results in a depth-image from the point of view of the camera on the laser data where the checkerboard is visible. Once the vertices of the calibration pattern are available, projection is reversed to retrieve the corresponding points in the 3D data.

Now, one has to find the transformation that minimizes the discrepancy between position and orientation of the calibration pattern in the camera and laser data as

$$\underset{\boldsymbol{R},\boldsymbol{t}}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{j=1}^{m(i)} \|\boldsymbol{\theta}_{c,i}^{T}(\boldsymbol{R}\,\boldsymbol{x}_{i,j} + \boldsymbol{t}) - d_{c,i}\| \quad , \tag{4.2}$$

where n is the number of elements in J and $m(i)$ is the number of inliers (3D measurements on the calibration pattern) in the LRF data. At least 15 pairs of J_i are necessary for the calibration, which takes around six minutes for each camera. Since the calibration object can be folded (flexible tripod), it is very portable and the sensor platform can easily be mounted on different vehicles.

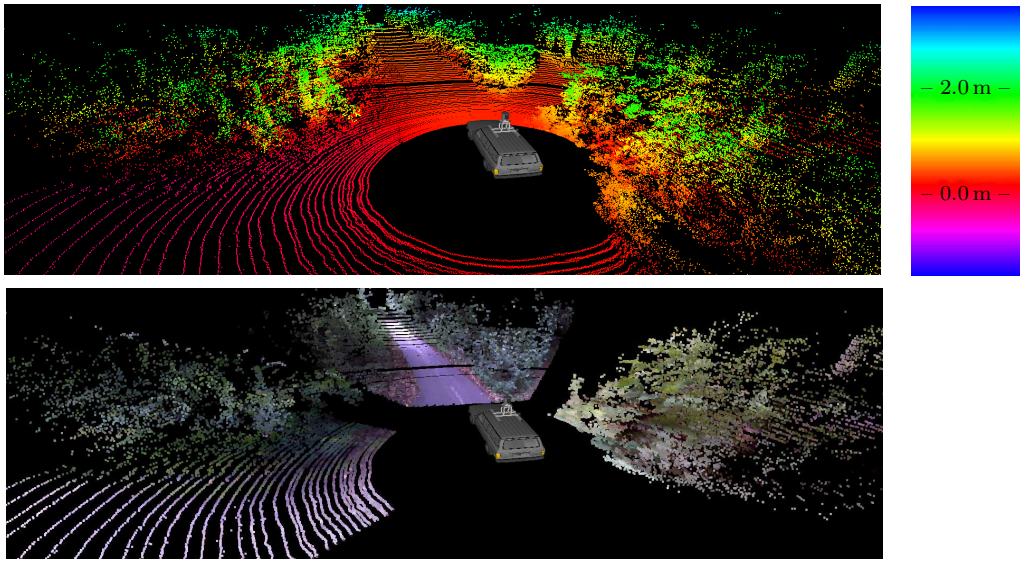


Figure 4.3: Example data of camera and Velodyne HDL-64E fusion. The upper part shows a camera image of a forest environment with trees and a road crossing. The 3D points are colored according to their height values, as illustrated in the legend on the upper right. In the lower part, the data of the Velodyne HDL-64E of the same scene are shown with the camera images projected into the same coordinate system.

The user can steer the single steps of the calibration process with the help of a graphical user interface (GUI). The calibration process is described with the aid of image Figure 4.2 which shows the calibration GUI.

The camera to be calibrated is selected in (1.) and automatically updates the camera image (7.) and the LRF data visualization (8.). The calibration pattern is automatically detected in the camera image and the LRF data visualization is projected and culled in the viewing direction of the camera. The *Refresh*-button (2.) generates the two visualizations and renewed detection but has no further influence on the calibration procedure. The detection of the calibration pattern in the LRF data visualization is performed by the user by selecting the four corners of the checkerboard pattern manually (8.). This step has to be repeated several times while moving the calibration pattern to different locations and updating the process using the *Next*-button. Each camera image with a set of LRF measurements forms a J_i, is saved, and will be used to compute the extrinsic parameters. A progress bar (3.) indicates the calibration progress. The *Save*-button (5.) stores the current J_i, e.g., to compare the results for different setups and the *Abort*-button (6.) discards everything. Once the progress bar reaches 100 %, the extrinsic parameters are calculated and printed out. The whole process takes around 20 minutes to calibrate all three cameras to the 3D LRF, including their intrinsic (re-)calibrations.

The results of the calibration and the developed calibration pattern are published in [Häselich et al., 2012a] and an example of the resulting fusion is presented in Figure 4.3. Calibration yields the foundation for an MRF terrain classification based on fused data.

Test drives with the robot on rough terrain revealed that the calibration needs to be repeated once in a while, probably caused by the vibrations while driving. After longer drives over rough terrain, a recalibration of all cameras to the LRF becomes necessary.

4.3 MRF Application

Classification of unstructured environments is a challenging task: sensor noise, manifold and complex vegetation, rough terrain, and vibrations while driving require a solid approach in order to realize an adequate terrain classification. Regarding these circumstances, a statistical approach can handle sensor noise and uncertainties quite well and for this reason an MRF on fused camera and LRF data is chosen, which will be described in the following. Since the developed algorithms will be used on different robot platforms, the availability of the cameras is optional. Hence, a greater versatility is achieved w.r.t. the robot platform and the available sensors as only a 3D LRF is required. Application of the MRF for terrain classification is separated into several subsections. First of all, data representation for the terrain is depicted in Section 4.3.1. Features for different terrain labels are described in Section 4.3.2 followed by the description of the annotation necessary for their estimation in Section 4.3.3. Finally, the Markov model of the terrain is explained in Section 4.3.4.

4.3.1 Terrain Data Representation

There are three requirements that need to be fulfilled when selecting the appropriate data representation for the terrain. The data representation has to be efficient so that the robot's algorithms are able to perceive the surrounding terrain in real-time. In addition, the data reduction needs to be precise enough to avoid any obstacle and does not unnecessarily block narrow passages. Finally, the drivability of the terrain should be considered in order to obtain the ability to prefer flat terrain segments like roads over rough terrain surfaces like fields. Therefore, a 2D grid is chosen with a resolution of $51.282\,\text{cm} \times 51.282\,\text{cm}$ for each terrain cell as data representation. The size of the grid is set from $40\,\text{m} \times 40\,\text{m}$ up to $100\,\text{m} \times 00\,\text{m}$ to achieve an optimal balance between computational effort of the MRF, information density of the 3D distance measurements, and the range of the LRFs. Since the grid is centered around the origin of the data, the LRF used, this data representation is feasible for the Velodyne HDL-64E and HDL-

32E. Considering collision avoidance and terrain drivability, a classification of the terrain cells is necessary. For the MRF approach, the following labels are used:

Street (gray) A terrain cell with an optimal surface in even ground, e.g., a street or a road segment.

Rough (brown) A terrain cell with a less preferable surface in rough terrain, e.g., a field or meadow. The autonomous system should avoid these regions as long as there is an alternative route on cells labeled *Street* which lead to the destination in adequate time.

Obstacle (red) A terrain cell that represents an impassable obstacle for the robot. A safety distance to these cells should be kept to guarantee collision-free navigation.

Unknown (black) A terrain cell with insufficient information, e.g., a region no sensor captured (rejection label).

A description of the features used for discrimination of the introduced labels follows in the next section.

4.3.2 Terrain Features

Features are essential for classification tasks. The following set of features describes the properties of the chosen terrain labels. Features are selected according to their low computational effort and their capability to separate the labels.

Laser-based features (L) The *roughness* f_{r} of a terrain cell describes it's surface structure and is, for example, suitable to distinguish between roads, meadows, or fields. Roughness is calculated by the *local height disturbance* [NDPP09]. *Height difference* f_{h} enables an efficient distinction between passable and impassable cells because a threshold can specify the maximal drivable height for each robot platform. According to Happold et al. [HOJ06], the distance between the lowest and highest 3D laser point within the cell is used for classification. This feature is based on the assumption that the main characteristic of an obstacle is its height.

Haralick features (H) Texture of different terrain labels varies and allows a differentiation. Computation of image features can easily become computationally expensive. Therefore, only three out of the 14 texture calculations proposed by Haralick et al. [HSD73] are used: *Angular Second Moment* f_{asm}, *Sum of Squares: Variance* f_{v} and *Inverse Difference Moment* f_{idm} (cf. Appendix D).

Homogeneity (FH) Another fast calculating feature is proposed by Knauer and Meffert [KM10] and expresses the texture homogeneity f_{fh} of a terrain cell. Based on the assumption that rough terrain surfaces of the label *Rough* possess a more heterogeneous texture than the terrain surface of label *Street*, this feature is incorporated in the terrain classification process.

Color (C) In addition to the distance and height measurements of the LRF and the texture features of the camera images, the color f_{c} is used to classify terrain cells.

This set of features can be divided into two categories. The first type is acquired from data gained by the LRF and contains the laser-based features (L). The second type consists of features computed from the camera images of the terrain and contains image-based features (H, FH and C). Algorithms are implemented in a way that if one or even all of the cameras fail (front-lighting, darkness, etc.) or are missing (e.g., change of robot platform), autonomous driving exclusively on the LRF data is still possible. Combination of all features yields the feature vector $\boldsymbol{f} = (f_{\mathrm{r}}, f_{\mathrm{h}}, f_{\mathrm{asm}}, f_{\mathrm{v}}, f_{\mathrm{idm}}, f_{\mathrm{fh}}, f_{\mathrm{c}})$ that integrates in the terrain classification process.

4.3.3 Acquisition of Terrain Features

In order to estimate feature parameters for the different labels, an annotation mode has been implemented. This mode allows computation of feature parameters from a manual classification of a user. Figure 4.4 shows the annotation mode in the software. A user selects a class label he wants to assign to a terrain cell (yellow rectangle in the lower image) and paints directly in the visualized data to classify it. In the lower image, an example annotation for the three labels *Street* (gray), *Rough* (brown), and *Obstacle* (red) is shown. Regarding data observability, the user annotation includes regions with incomplete data. Those region have two origins: missing sensor data (e.g., in regions where the cameras do not overlap or the two blocks of the 32 lasers of the Velodyne HDL-64E meet (cf. Figure 4.3)) or occlusions and hidden spots in the natural terrain (e.g., caused by opaque obstacles like trees or bushes). The result of the manual classification is extracted and saved in addition to previously made classifications. This has the advantage that the estimated parameters are based on real data acquired from the environment which are carefully classified by a human expert.

4.3.4 Markov Model of the Terrain

Selecting an appropriate Markov model of the terrain requires the examination of the underlying data representation for that terrain. Since the representation of the terrain is realized with a two-dimensional grid, the application of a two-dimensional MRF seems natural. The two-dimensional MRF on terrain cells

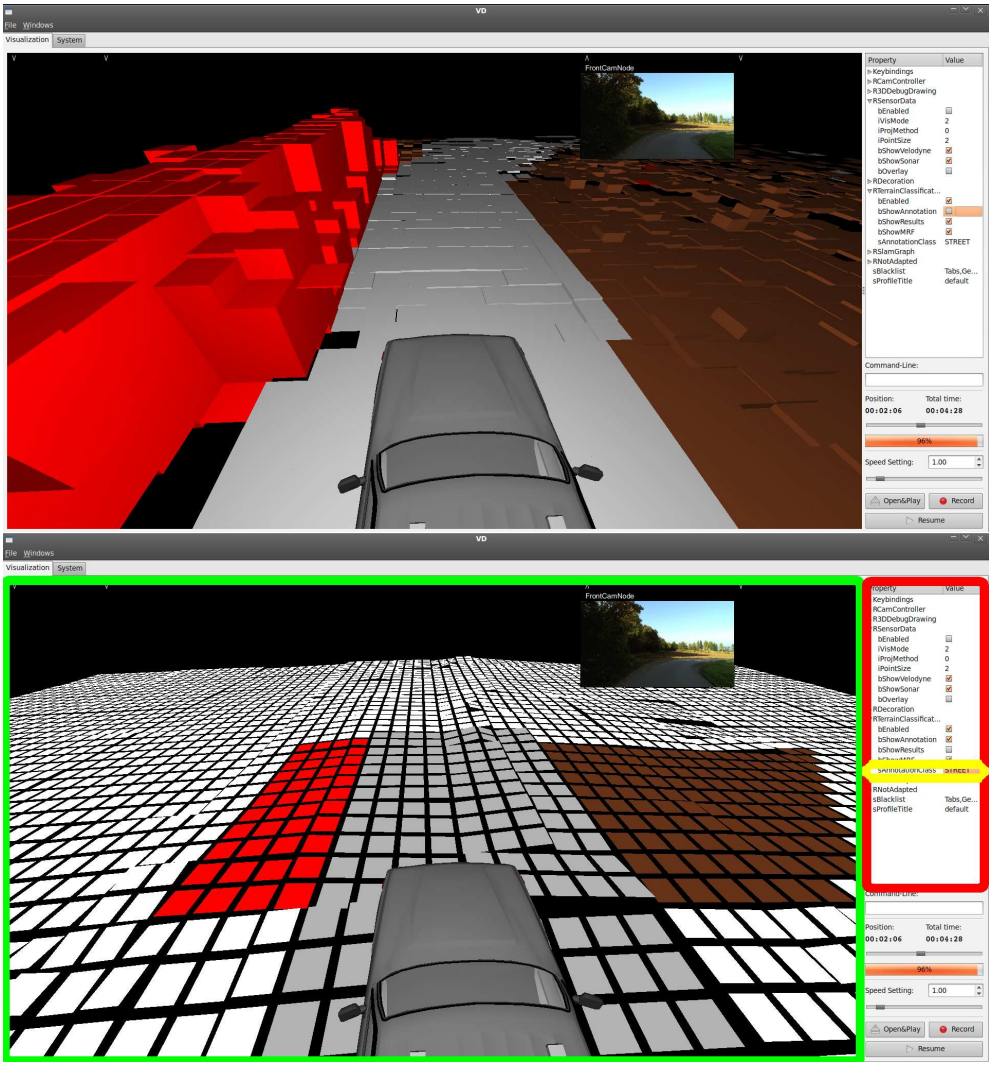


Figure 4.4: User interface of the annotation software with activated classification in normal mode (top) and in annotation mode (bottom). The lower image shows parameters for visualization (red rectangle), selection of terrain label (yellow rectangle), and visualization frame (green rectangle) in which the users annotates the data.

behaves similar to a two-dimensional MRF used for image segmentation based on neighborhood relationships. A Markov model of the terrain should also respect the context-sensitivity of neighboring cells and the acquired features. Therefore, the MRF does not exclusively regard neighborhood relationships, like e.g., the model described by Wolf et al. [WSFB05, WS08], but also available features. For this reason, an MRF model is chosen that prove reliable in image segmentation tasks, as described by Deng and Clausi [DC04]. Such a model is able to fulfill both requirements and can easily be adapted to the terrain classification problem by regarding each terrain cell as random state in the MRF. Here, the energy E of a GRF is calculated piece-wise by splitting it up into a component $\mathrm{E}_{\mathcal{N}}$ that models the relationship of the neighboring cells and another component E_f which models the computed features of a cell. Equivalence of MRFs and GRFs is given by the Hammersley-Clifford Theorem, cf. Section 2.6.

The neighborhood component $\mathrm{E}_{\mathcal{N}_{i,j}}$ of a cell $\mathfrak{c}_{i,j}$ at position (i,j) in a grid is defined as

$$\mathrm{E}_{\mathcal{N}_{i,j}} = \sum_{\lambda \in \mathcal{N}_{i,j}} \beta \cdot \delta(l_{i,j}, l_\lambda) \quad , \tag{4.3}$$

where $l_{i,j}$ is the terrain label assigned to the cell $\mathfrak{c}_{i,j}$ and l_λ is the terrain label assigned to a cell $\mathfrak{c}_\lambda$, which is part of the neighborhood of $\mathfrak{c}_{i,j}$. The function $\delta(\cdot)$ returns a value of -1 for $l_{i,j} = l_\lambda$ and $+1$ for $l_{i,j} \neq l_\lambda$ (cf. the Potts Model Section 2.7). β is used to weight a neighbor's impact according to its distance to the cell.

The corresponding feature component $\mathrm{E}_{f_{i,j}}$ of a cell $\mathfrak{c}_{i,j}$ is based on the assumption that the features of a cell follow a Gaussian distribution. As an energy its computation is defined as

$$\mathrm{E}_{f_{i,j}} = \sum_k \left(\frac{(f_{i,j_k} - \mu_{i,j_k})^2}{2\sigma_{i,j_k}^2} + \log\left(\sqrt{2\pi}\sigma_{i,j_k}\right) \right) \quad , \tag{4.4}$$

where f_{i,j_k} is the k-th feature of $\mathfrak{c}_{i,j}$ and μ_{i,j_k} and σ_{i,j_k} are the mean and the standard deviation respectively of the k-th feature of the label $l_{i,j}$ assigned to a cell $\mathfrak{c}_{i,j}$.

Combining the two calculations, the complete energy $\mathrm{E}_{i,j}$ of a cell $\mathfrak{c}_{i,j}$ can be calculated as

$$\mathrm{E}_{i,j} = \mathrm{E}_{\mathcal{N}_{i,j}} + \alpha \cdot \mathrm{E}_{f_{i,j}} \tag{4.5}$$

where α is a weighting constant used to control the influence of the different energy types.

For classification, the sum of all computed energies $\mathrm{E}_{i,j}$ needs to be minimized. This leads to a maximization of the posterior probability of the labeling of the terrain. The energy can be minimized by finding a label for each cell which fits best for the computed features and the labels of the neighbor cells (cf. the optimal solution from Equation 2.7). Optimization needs to be performed in real-time and

is described together with the experimental results in the next section since it is strongly coupled to the runtime of the algorithms.

4.4 Evaluation and Optimization

The conducted experiments compare the MRF terrain classification results with a ground truth. Ground truth is acquired by human experts, who annotate terrain cells per hand with the tool described in Section 4.3.3. By comparing results with their respective ground truth, true positive and false positive rates for each terrain label are computed. Classification is evaluated for different scenarios where the relative appearances of the terrain labels differs. The system used for all computations is a single laptop, an Intel(R) Core(TM) i7 QM with 1.73 GHz and 8 GB RAM, which is the computer that is used on the autonomous robots. An example is shown in Figure 4.5, where 3D data, fused data, and MRF results are visualized.

Table 4.1 shows the result for a scenario where the robot drives on a road through fields and small to medium sized vegetation. Table 4.2 shows the results for a forest scenario where the label *Rough* appears only in single terrain cells and not in larger regions. In both tables the use of laser-based features is denoted by L, selected Haralick features by H, the homogeneity feature by FH and color by C. Furthermore, TPR is the true positive rate and FPR the corresponding false positive rate. For both scenarios, the corresponding receiver operating characteristics (ROC curves) are shown in Figure 4.6. In all tested scenarios, results for detecting streets and obstacles are equally good. In case of Table 4.2, the results show that the detection of the less appearing label *Rough* does not work properly. The reason is the modeling of the terrain, in which it is assumed that terrain cells of one label show a tendency to group and not to exist as single cells. Hence, those rare *Rough* cells are overruled by the neighborhood component (cf. α in Equation 4.5 which remained unchanged for all the experiments). Furthermore, the use of image features does not significantly improve the quality of the results as expected. One reason for this may be that texture and color vary more than assumed, which leads to high values for the standard deviations of these features. Thus a high deviation from the mean value of such a feature does not change the computed energy value sufficiently to have impact on the classification. During the evaluation it could be observed that color values are able to improve classification results but suffer from various outdoor-specific effects. For example, weather conditions play an important role as wet surfaces change their color and clouds influence illumination. Front-lighting and changes in viewing angles of the cameras are also problematic, especially for low-cost camera solutions (cf. cameras presented in Section 1). In conclusion, color information from cameras provides no real benefit if used according to the presented approach. Hardware solutions are available as advanced camera systems, e.g., like high-dynamic-range or multi-

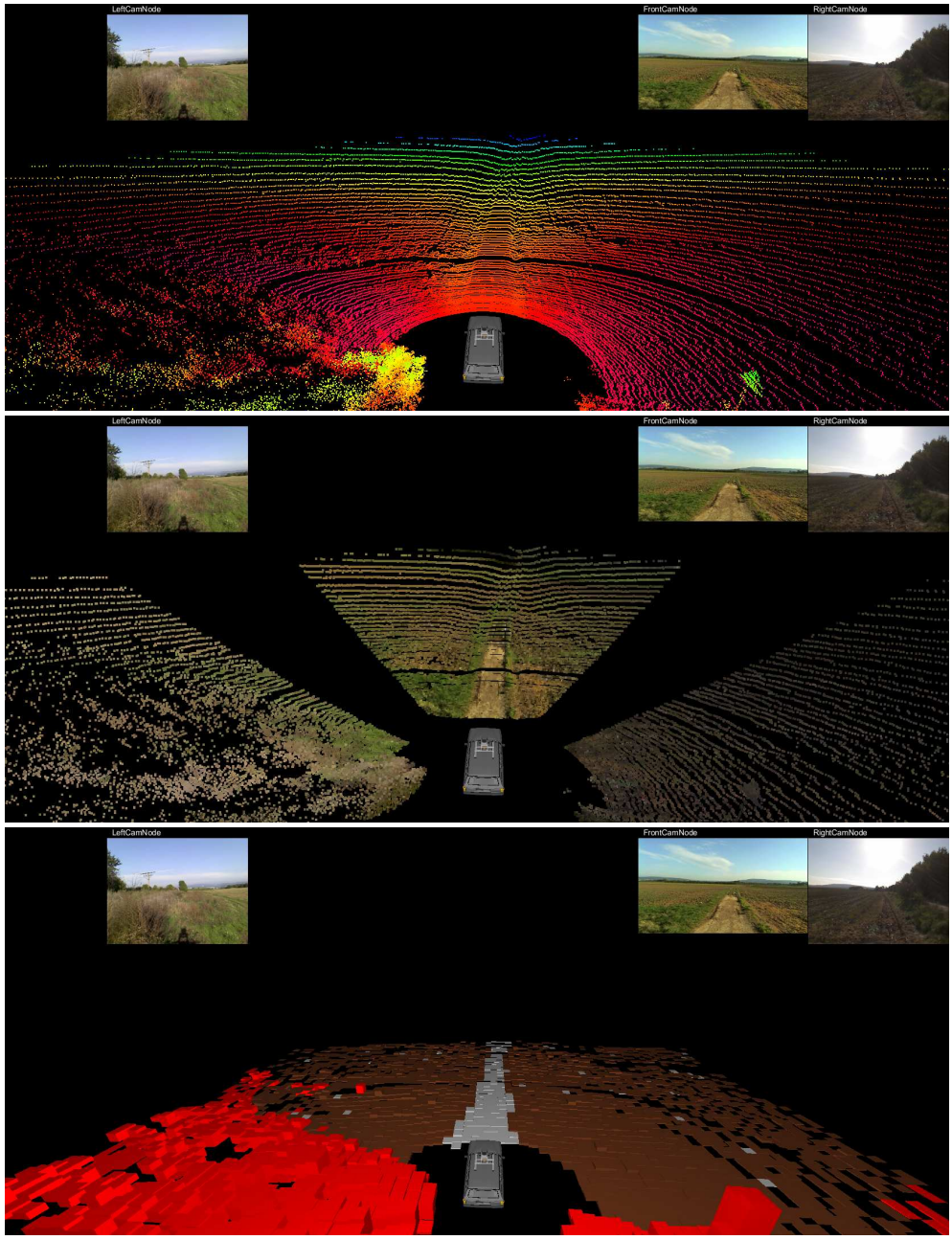


Figure 4.5: Example of an unstructured environment with three different representations. The images from the left, front and right camera are shown in the upper part of each representation. The upper representation shows the 3D point cloud provided by the Velodyne HDL-64E. The color of a point is determined by its height. On the representation in the middle, the LRF data are fused with all three cameras. This way color, texture, and 3D position are available in one coordinate system. The lower representation shows the result of the MRF application: classified terrain cells with drivability information. The labels *Unknown, Road, Rough,* and *Obstacle* are visualized in black, gray, brown, and red, respectively.

Label/value	**L**	**L + H**	**L + FH**	**L + C**
Road/TPR	91.0 %	91.9 %	90.4 %	91.0 %
Road/FPR	1.5 %	1.2 %	1.2 %	1.4 %
Rough/TPR	74.6 %	73.8 %	75.2 %	75.3 %
Rough/FPR	1.2 %	1.1 %	1.3 %	1.4 %
Obstacle/TPR	92.8 %	91.8 %	92.9 %	92.6 %
Obstacle/FPR	3.6 %	4.4 %	3.2 %	3.1 %

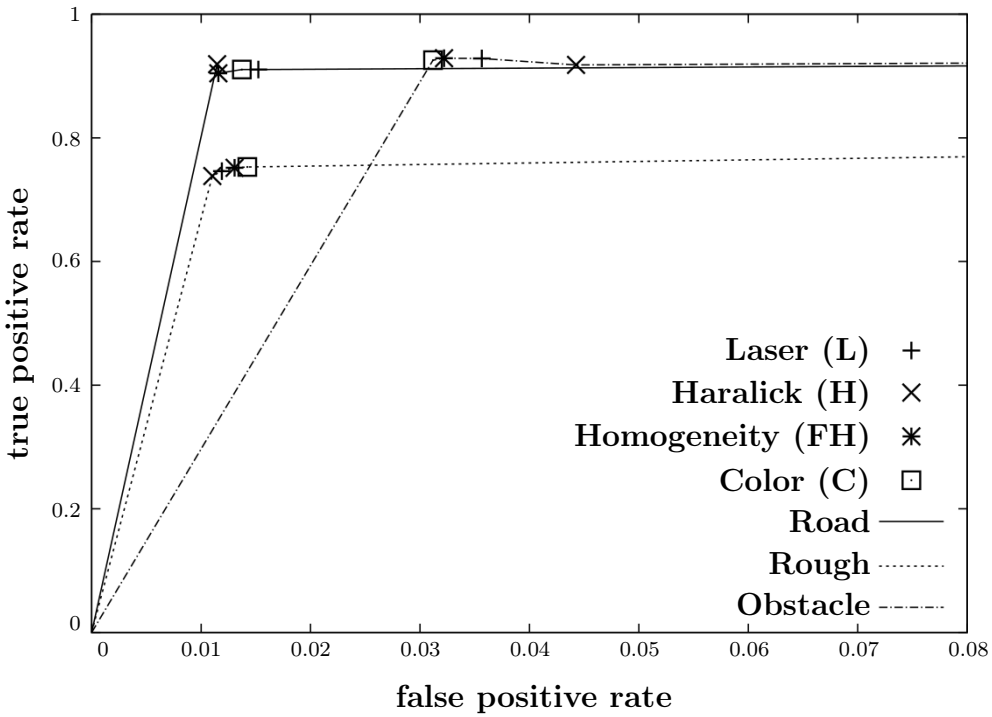
Table 4.1: Performance of the MRF terrain classification algorithm in a rural environment. True positive rate (TPR) and false positive rate (FPR) are presented for the three labels *Road*, *Rough* and *Obstacle*. Columns are separated by the features used: laser-based features (L), Haralick features (H), the homogeneity feature (FH), and the color information (C).

Label/value	**L**	**L + H**	**L + FH**	**L + C**
Road/TPR	95.2 %	91.7 %	95.0 %	95.0 %
Road/FPR	4.3 %	5.6 %	4.3 %	4.5 %
Rough/TPR	5.7 %	0.4 %	2.8 %	3.9 %
Rough/FPR	0.3 %	0.1 %	0.3 %	0.3 %
Obstacle/TPR	96.2 %	95.4 %	96.1 %	96.3 %
Obstacle/FPR	6.9 %	6.7 %	7.3 %	6.8 %

Table 4.2: Performance of the MRF terrain classification algorithm in a forest environment. Labels and features are used analogous to Table 4.1 and true positive and false positive rates are shown. This scenario is specified by very few cells of label *Rough*, which are further suppressed by the neighborhood relations within the MRF.

Mode	**Mean**	**Std. dev.**	**Max**	**Min**
L	62.7 ms	7.4 ms	82.2 ms	48.8 ms
L+H	744.9 ms	131.5 ms	994.0 ms	421.5 ms
L+FH	141.4 ms	11.2 ms	188.4 ms	121.7 ms
L+C	144.4 ms	12.6 ms	187.0 ms	117.3 ms

Table 4.3: Runtime results for the MRF application with different features from single sensor and fused sensor data within an area of 40 m × 40 m. All values represent long-term measurements and result from the direct application to real sensor data, without preprocessing or any form of data reduction. The terms L, H, FH, and C are equivalent to Table 4.1 and Table 4.2.



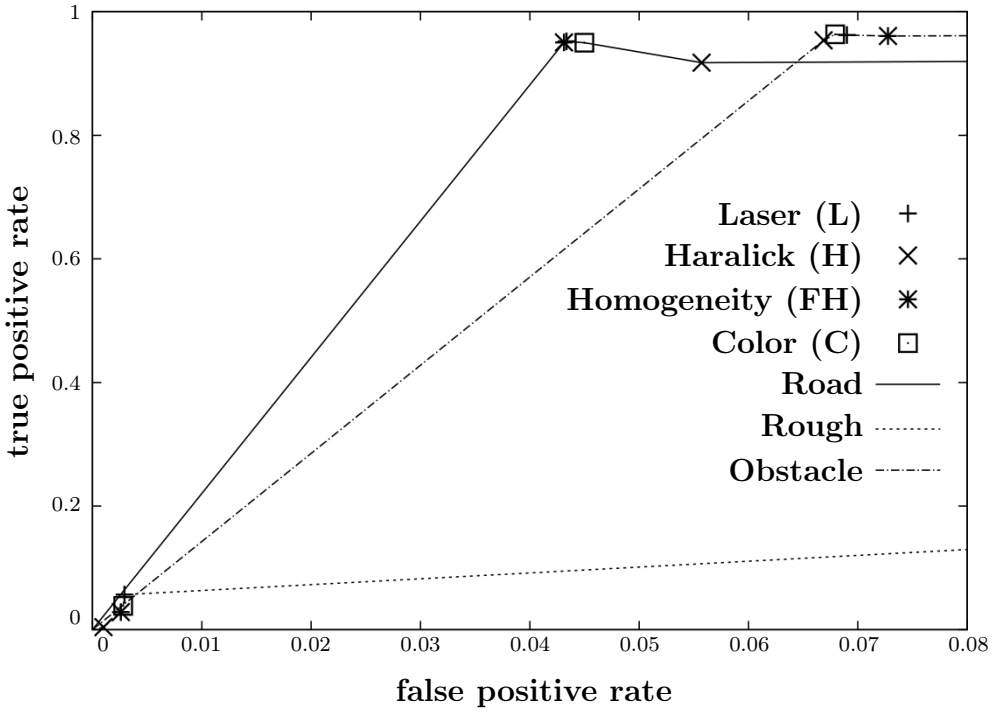


Figure 4.6: ROC curves corresponding to the values of Table 4.1 (upper figure) and Table 4.2 (lower figure). The three curves in both figures show the true positive rate in relation to the false positive rate for the three labels *Road*, *Rough* and *Obstacle*.

spectral cameras, or as software solutions by having a color checkerboard visible all the time or by calibrating color values before each operation of the robot.

Evaluation of the runtime, shown in Table 4.3, shows that using of image features multiplies the required calculation time. The table compares averages and standard deviations as well as maxima and minima of runtime for different configurations of the terrain classification. Using laser features exclusively leads to a fast runtime and good classification results. In all experiments the fixed values α and β were respectively set to 0.4 and 0.8 (cf. Equation 4.5 and 4.3) and the terrain was limited to 40 m × 40 m. Results of the MRF terrain classification are published in [Häselich et al., 2011a].

4.4.1 Gibbs Sampler Optimization

Configuration space is given by the number of labels to the power of the number of sites, cf. Equation 2.6. For a terrain grid of 40 m × 40 m up to 100 m × 100 m, a terrain cell resolution of 51.282 cm × 51.282 cm, and 4 different labels, this leads to a total number of $4^{(78\times78)}$ respectively $4^{(195\times195)}$ possible configurations. Of course, these large numbers represent only theoretical maxima since occlusion and regions without measurements significantly reduce the number of plausible solutions.

The terrain classification results presented in Section 4.4 and Table 4.3 are limited to a terrain of 40 m × 40 m. This means that the maximum range of the sensor data is decreased to 20 m in each direction (28.28 m diagonal). Considering the Velodyne HDL-64E LRF, a larger distance of up to 100 m × 100 m allows a more predictable path planning. Although the Velodyne HDL-64E and HDL-32E have a range of more than 50 m in each direction, the point cloud becomes too sparse at larger distances, especially w.r.t. the ground readings which are essential for surface classification. In order to further increase the processable area, system performance is optimized using multi-core CPU and general-purpose graphics processing unit (GPGPU) hardware. For each point cloud, the MRF needs around 63 ms (max. 82 ms) for a 40 m × 40 m grid (cf. Table 4.3). Most of this time is consumed by the optimization step in which the optimal labeling of the terrain regions is determined. Considering these properties, a grid size of 100 m × 100 m is desirable. The Velodyne HDL-64E has a frequency of up to 15 Hz that corresponds to a minimal update time of 67 ms for each point cloud gathered in a full rotation of the sensor. For the Velodyne HDL-32E, the same acceleration techniques can be used since the frequency of 20 Hz is in fact faster, but the point clouds are sparser. Therefore, it is necessary to accelerate the energy minimization in order to increase the size of the terrain covered and to ensure a terrain classification with the MRF in real-time. A longer range results in a more predictive behavior for a path planning algorithm and an improved runtime makes collision avoidance faster and therefore more secure at higher speeds. This section describes the runtime optimizations used to achieve this specific goal.

Energy minimization of an MRF can be done with a variety of algorithms with different advantages and disadvantages. In graph theory, the min-cut/max-flow problem [FF56, FF62] is often solved by Graph Cuts [GPS89] or Swendsen-Wang Cuts [SW87]. Those algorithms solve the problem for binary classification problems and hence realize the Ising model (cf. Section 2.7). The Wolff algorithm [Wol89] represents an extended version of the Swendsen-Wang Cuts and allows multiple labels to swap simultaneously. Approaches that extend Graph Cuts for more than two labels are e.g., the alpha expansion or the alpha-beta swap algorithms [BVR01]. Those algorithms mainly accelerate classification by changing clusters of labels simultaneously (neighboring spins technique). Unlike image segmentation, it is essential in the terrain classification domain to be able to maintain certain terrain cells even if they are an antipode to their whole neighborhood. Typical examples for such terrain cells in a natural environment are single small trees, separate tall but thin rocks, or poles on a field. When approaching these obstacles, it is essential for an autonomously navigating robot that these single terrain cells remain obstacles during the energy minimization in order to guarantee a collision-free navigation. For this reason the Gibbs sampler [GG84] (cf. Section 2.9) is chosen to minimize the energy of the MRF. Its sampling strategy allows it to influence the weight of the particular terrain labels separately w.r.t. the observed sensor measurements.

Considering an acceleration of the MRF energy minimization, Gonzalez et al. [GLGG11] propose two methods to construct parallel Gibbs samplers. One method uses graph coloring to construct a direct parallelization of the sequential Gibbs sampler and the other method is a complementary strategy which can be used when variables are tightly coupled.

Martens and Sutskever [MS10] introduce a Markov chain transition operator that updates all variables of a pairwise MRF in parallel by using auxiliary Gaussian variables. Their experiments show that the results are comparable to the sequential Gibbs sampler in terms of mixing speed, while being simple to implement and parallelize.

Pock et al. [PCBC09] propose a convex relaxation approach for the Potts model. Their approach uses an efficient primal dual projected gradient algorithm which also allows a fast implementation on parallel hardware but does not guarantee to find global minimizers of the Potts model.

Kato et al. [KBZ96] present a multi-scale MRF model that consists of a label pyramid in which the optimization is first performed at higher scales by a parallel relaxation algorithm. Then, the next lower scale is initialized by a projection of the result. On the one hand, the authors introduce an interaction scheme between two neighbor grids in this label pyramid and show that these connections allow for propagating local interactions more efficiently, yielding faster convergence in some cases. On the other hand, these interactions make the model more complex, requiring computationally more demanding algorithms.

A comparative study of further energy minimization methods for MRFs is given in detail by Szeliski et al. [SZS+08].

Two main strategies exist to accelerate the Gibbs sampler. The first one is a modification of the sampling algorithm itself, as proposed by other authors (see [MS10, PCBC09]). Secondly, the terrain grid representation allows a subdivision into smaller terrain pieces, which can then be processed by an unmodified Gibbs sampler. Though each of the sub-grids will suffer from the burn-in phase of the sampling process, the parallelization opportunity — especially on the graphics card — makes this approach very interesting.

The unmodified algorithm takes the initial grid and a temperature as input values and runs in a loop until convergence. It iterates over the width and height of the terrain according to the number of terrain cells in each direction. Within the loop, it computes the neighborhood and feature energies according to Equation 4.3 and Equation 4.4, and computes the energy of a possible label change for each label. This computation uses the exponential function of the negative energy (cf. [DC04]) divided by the current temperature value, which is decreased at the end of each iteration. Afterwards, random numbers are used to sample from the target distributions and a new label is chosen for each cell with respect to its energy value in case it yields to a minimization of the energy (cf. Section 2.9). This is exactly the approach which is used in Table 4.3 and will be evaluated as *Basic*-algorithm in the following subsection.

The first optimization concerns the energy computation step on the part of the feature energy. According to Equation 4.5, energy is computed as sum of the neighborhood and the feature energy and both energy values of a cell are computed during runtime. In contrast to the neighbors, features extracted from the 3D points contained in a cell will not change within the loop (within one sensor point cloud). Hence, it is more efficient to compute the features in advance and to access results during runtime. This improvement will be evaluated as *Feature*-algorithm in the following subsection. For example, for a terrain grid of 100 m × 100 m and a cell size of around 50 cm × 50 cm, this results in a memory usage of 40,000 floating point values. In addition, this acceleration is independent of available hardware. In order to take advantage of multiple CPUs, the terrain is split into slices. Hence, a number of slices is created corresponding to the number of terrain rows divided by the number of available threads. Each thread handles the number of terrain cells in each row assigned to it; for example, for the terrain of 100 m × 100 m and a CPU with eight cores, eight threads are created each handling 25 terrain rows with 200 cells of 50 cm × 50 cm in each row. This way, the whole terrain grid is processed at once by the multi-core processor. In order to avoid accessing neighbors of different instants of time, the terrain is copied and updated at the beginning of the loop and each thread retrieves the neighborhood information from the current copy. For the implementation, the boost library [5] is used. The CPU parallelization will be evaluated as *Multicore*-algorithm in the following subsection.

Graphics processing units (GPUs) are high-performance many-core processors that can be used to efficiently accelerate applications. In contrast to CPU parallelization, various properties of the graphics card play an important role and need to be considered carefully for an optimal implementation. In this work, the OpenCL 1.1 framework [9] is used. To take advantage of the GPU, the terrain grid is divided into squared work-groups. The work-groups are executed in clock cycles, one work-group per cycle per GPU core. A large work-group size is able to hide the memory latency while a small one decreases the amount of active threads. The amount of work-items within one work-group is restricted for each graphics card. Therefore, the optimal work-group size needs to be adjusted for each GPU individually in order to achieve the best performance. The Gibbs sampler is implemented as a GPU kernel that is executed for every work-item with the exception that the loop and the control of the temperature remains on the CPU. Since the graphics card limits the maximum number of work-items, the GPUs will not be able to process the complete terrain in one cycle. In fact, a total of four cycles are necessary to process all terrain cells, which results in an interesting comparison, since the *Multicore*-algorithm is able to process the whole terrain at once. Besides the splitting of the terrain into squared work-groups, the local memory of each GPU core is used and relevant terrain cells are copied to this fast GPU memory. This copying is done in parallel and the local memory of the GPU allows faster computations during runtime. For example, for a terrain of $100\,\mathrm{m} \times 100\,\mathrm{m}$ with an nVidia (R) Quadro (R) FX 880M, 256 work-groups are used each with 144 work-items that are processed by 48 CUDA (TM) cores. This general-purpose computing on the graphics processing unit will be evaluated as *GPGPU*-algorithm in the following subsection.

4.4.1.1 Evaluation

Evaluation of the runtime optimizations is again performed on a single laptop with an Intel(R) Core(TM) i7 QM with 1.73 GHz and 8 GB RAM and an nVidia(R) Quadro(R) FX 880M. MRF classification is performed on real sensor data of urban, rural, and forest scenarios. During the experiments, cell size and all parameters are fixed. Performance of the various algorithms is displayed in Table 4.4. In the domain of terrain classification, it is desirable that nearly all laser scans are processed in real-time. Therefore, the *soft real-time criterion* can be defined over the mean runtime of the algorithm which guarantees that most of the scans will be processed. In order to guarantee that all terrain classification will be processed in real-time, the maximum runtime of the algorithm must never exceed a threshold. The threshold is given by the maximum update frequency of the LRF, which defines the *hard real-time criterion*. In the table, mean values that meet the soft real-time criterion and maximum runtimes that meet the hard real-time criterion, both at 67 ms (15 Hz), are denoted by an asterix.

Algorithm	Grid size [m×m]	Mean [ms]	Std. dev. [ms]	Min [ms]	Max [ms]
Basic	20×20	15.1*	1.0	11.3	25.8*
	40×40	51.2*	2.1	40.2	116.9
	60×60	102.8	2.4	82.8	162.3
	80×80	165.2	3.0	140.5	282.0
	100×100	248.3	3.4	219.3	387.7
Feature	20×20	8.1*	0.7	5.8	14.4*
	40×40	32.6*	1.4	26.8	57.1*
	60×60	75.6	1.9	66.1	123.5
	80×80	125.1	2.2	108.6	161.2
	100×100	206.2	3.1	185.1	307.8
Multicore (8 CPU cores)	20×20	3.4*	0.3	1.9	8.9*
	40×40	13.6*	0.8	8.4	30.6*
	60×60	28.1*	1.2	18.3	46.2*
	80×80	43.8*	1.4	31.6	80.8
	100×100	67.4	1.6	54.2	124.2
GPGPU (48 CUDA (TM) cores)	20×20	4.1*	0.2	3.5	12.0*
	40×40	6.2*	0.3	5.1	14.5*
	60×60	11.0*	0.3	9.7	27.7*
	80×80	16.1*	0.6	14.1	39.5*
	100×100	23.6*	0.9	20.5	43.0*

Table 4.4: Runtime results of the different optimizations. All values represent long-term measurements and result from the direct application to real sensor data of urban, rural, and forest scenarios. Dimensions of the grid are given in square meters. Mean, standard deviation, minima and maxima are given in milliseconds. Mean values that meet the soft real-time criterion and maxima that meet the hard real-time criterion are denoted by an asterix.

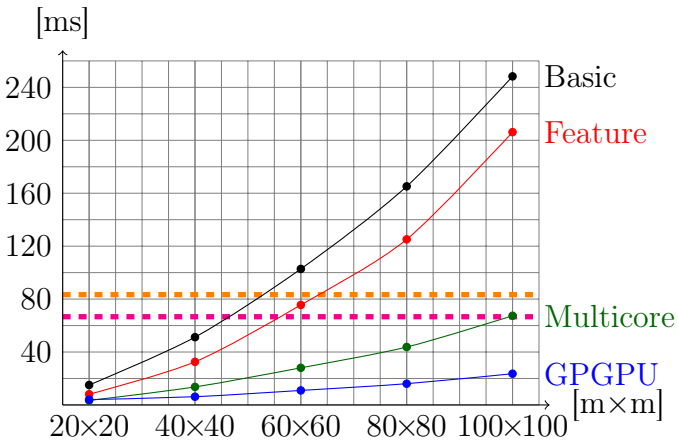


Figure 4.7: Representation of the mean runtimes as curves. The orange dashed line indicates that both parallelizations (Multicore and GPGPU) are fast enough to process the data of the Velodyne HDL-64E (mean update frequency of 12 Hz) for all grid sizes. The magenta dashed line represents an update frequency of 15 Hz and shows that only the GPGPU variant is fast enough for a hard real-time criterion on the 100 m × 100 m terrain.

A graphical comparison of the mean runtimes is given in Figure 4.7. The orange dashed line represents the measured mean update frequency of the Velodyne HDL-64E and hence can be used as guideline for a soft real-time criterion. In contrast to that, a hard real-time criterion defined by the maximum frequency of 67 ms can only be achieved by the GPGPU-algorithm with a maximum runtime of 43.0 ms. The magenta dashed line represents the maximum update frequency of 15 Hz of the Velodyne HDL-64E. Considering the 20 Hz update frequency of the Velodyne HDL-32E, each laser-scan contains much less 3D points so that the criteria can be applied analogously. Though the *Feature*-algorithm yields the weakest acceleration, it has the advantage of being hardware-independent and is used by the two parallelization approaches. At this point it is observable that even with a non-optimal configuration of work-items and terrain grid, the standard laptop GPU considerably outperforms the multi-core approach. Compared to other results, where the Gibbs sampler itself or other energy minimizing algorithms are parallelized, the GPGPU approach yields a sufficient acceleration, too. Reservations for the terrain segmentation approach where that each thread (either on the CPU or the GPU) has to deal with the burn-in phase of the sampler. Thus, it is interesting to see, that a parallelization, without a manipulation of the original Gibbs sampler, is able to process all data in real-time. Results of the Gibbs sampler optimization are published in [Häselich et al., 2012b].

4.5 Egomotion Estimation

In order to be able to integrate classification results over time, a local estimate of the robot's motion between different time steps is required. The software (cf. Section 3) already uses a simple extended Kalman filter to determine the dynamic state of the robot in a global coordinate frame. The filter fuses the information from the IMU with the measurements from the GPS receiver or, if GPS information is unavailable, with speed measurements acquired directly from the robot, using the radar-based speed sensor Speed Wedge SW01. The following Kalman state is used in order to encode the current state of the robot:

$$\boldsymbol{\mu}_t = \begin{pmatrix} x & y & z & v & \phi & \vartheta & \psi \end{pmatrix}^{\mathrm{T}} \tag{4.6}$$

Here, (x, y, z) and v describe the current position and velocity of the robot. The angles ϕ, ϑ, and ψ encode the robot's current orientation in roll, pitch, and yaw angles, respectively. All of these quantities are expressed in the NWU (north west up) coordinate frame. In the Kalman prediction phase, the commonly used assumption of a simple linear motion model with additive Gaussian noise ζ is made

$$\boldsymbol{\mu}_{t+1} = \begin{pmatrix} x_t + v_t \Delta t \cos\vartheta_t \cos\psi_t \\ y_t + v_t \Delta t \cos\vartheta_t \sin\psi_t \\ z_t + v_t \Delta t \sin\vartheta_t \\ v_{\mathrm{odo}} \\ \phi_t + \phi_{\mathrm{odo}} \Delta t \\ \vartheta_t + \vartheta_{\mathrm{odo}} \Delta t \\ \psi_t + \psi_{\mathrm{odo}} \Delta t \end{pmatrix} + \zeta \quad . \tag{4.7}$$

The values $\phi_{\mathrm{odo}}, \vartheta_{\mathrm{odo}}, \psi_{\mathrm{odo}}$ are the angular velocities in roll, pitch, and yaw direction, respectively, as measured by the IMU. The amount of time in milliseconds between two rotations of the 3D LRF is given by Δt. The scalar v_{odo} describes the current velocity of the robot, as measured by the robot's built-in speed sensors.

The next step would be to perform the Kalman update using the measurements obtained by the GPS receiver. For this application however, it suffices to have an estimate of the robot's motion between two subsequent time steps. In particular, it is not required to globally correct positioning information. While the previous methodology to estimate the relative transformation between subsequent robot poses yields very good guesses, it is generally not precise enough to be used exclusively. Therefore, a variant of the Iterative Closest Point (ICP) algorithm on a two-dimensional virtual scan was chosen to further refine these estimates.

Petrovskaya and Thrun [PT09] describe the extraction of a two-dimensional scan from data of a Velodyne HDL-64E using a piece-wise planar ground model. All readings are projected into a 3D grid in polar coordinates around the vehicle center (cf. Figure 4.8). For filtering noise, a representative reading is chosen for

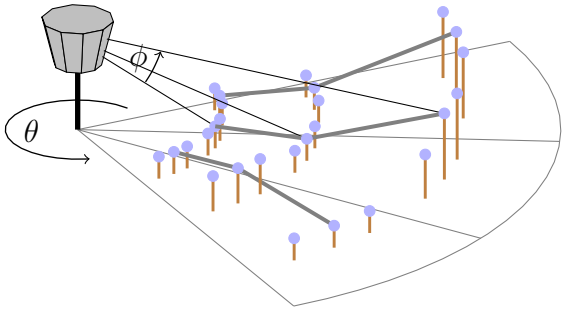


Figure 4.8: Projection of Velodyne HDL-64E readings into a 3D grid in spherical coordinates as described in [PT09]. Readings are illustrated as dots. Their height is indicated by a line that projects readings into the horizontal plane.

each cell as the median of distances. A model of ground elevation is established by comparing neighboring cells with the same horizontal bearing θ. If the slope is smaller than a threshold, the point is classified as ground reading. The distance to the closest obstacle in a target height of 50 cm to 200 cm is used as the entry in the virtual scan. With only mid-height objects apparent in the scan, overhanging foliage and small objects, such as curbs or small stones, are not considered for egomotion estimation. While this method has been developed in the context of vehicle detection and tracking (cf. Section 5.1.1), it works equally well for egomotion estimation.

In order to refine the results of the extended Kalman filter prediction, point sets are registered as $P = \{\mathsf{P}_i\}$ and $Q = \{\mathsf{Q}_j\}$ that correspond to readings within virtual scans acquired at the prior and the current time step. Therefore, a 2D variant of the ICP algorithm [Nüc09] is used that minimizes the error function

$$\rho(\boldsymbol{R}, \boldsymbol{t}) = \sum_{(i,j)\in C} \|\mathsf{P}_i - (\boldsymbol{R}\mathsf{Q}_j + \boldsymbol{t})\|^2 \quad , \tag{4.8}$$

with rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$. Here, C is a set of corresponding point index pairs determined by the nearest neighbor search. The transformation $(\boldsymbol{R}, \boldsymbol{t})$ that minimizes Equation 4.8 is computed using the direct method based on SVD. Therefore, the correlation matrix is computed as

$$\boldsymbol{H} = \sum_{(i,j)\in C} (\mathsf{P}_i - \bar{\mathsf{P}})(\mathsf{Q}_j - \bar{\mathsf{Q}})^{\mathrm{T}} \quad , \tag{4.9}$$

where $\bar{\mathsf{P}}$ and $\bar{\mathsf{Q}}$ are the means of points in the correspondence set:

$$\bar{\mathsf{P}} = \frac{1}{|C|} \sum_{(i,j)\in C} \mathsf{P}_i \qquad \bar{\mathsf{Q}} = \frac{1}{|C|} \sum_{(i,j)\in C} \mathsf{Q}_j \quad . \tag{4.10}$$

Next, rotation and translation are calculated separately. First, the rotation is determined as

$$\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^{\mathrm{T}} \quad , \tag{4.11}$$

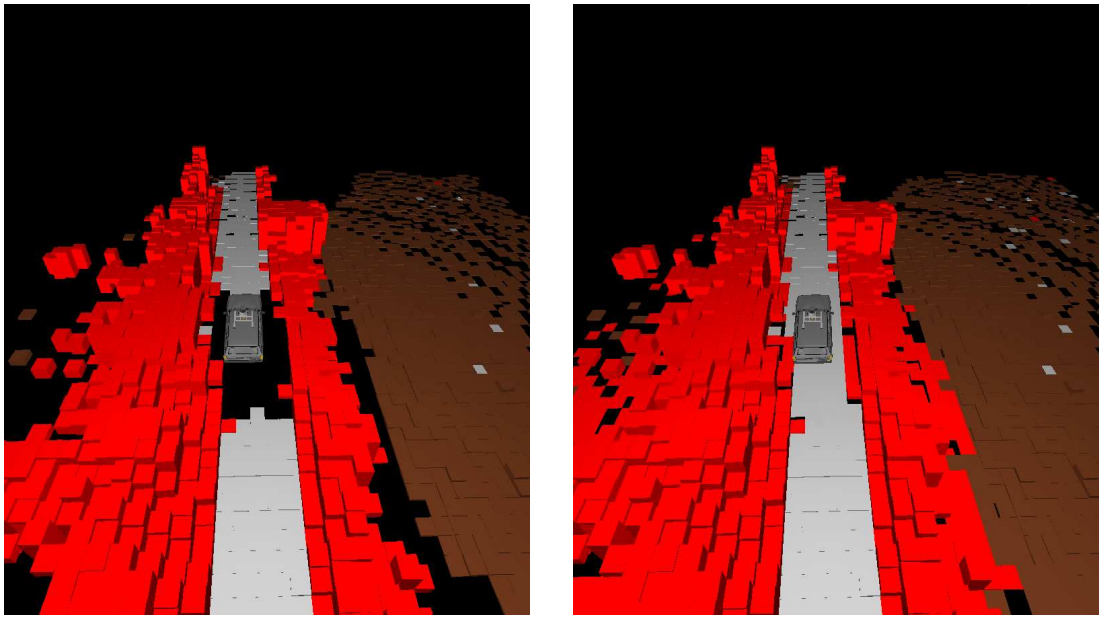


Figure 4.9: MRF classification with and without Egomotion estimation. The image on the left shows the MRF result on a single 3D scan while the image on the right shows an interpolation of data gaps in the MRF by accessing the previous terrain classification. Terrain cells in the direct neighborhood of the robot are included as well as previously visible regions on the right side of the robot behind some obstacles.

where matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ are obtained using SVD $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}}$. Then, the translation is computed as

$$\boldsymbol{t} = \boldsymbol{\mu}_p - \boldsymbol{R}\boldsymbol{\mu}_q \quad . \tag{4.12}$$

Using egomotion estimates, it is possible to transform the current grid structure into a preceding coordinate frame by rotating and translating the center of each cell. Therefore, the MRF is able to access previous classification results to obtain an estimate of terrain conditions where insufficient sensor data are available. Note that this methodology is analogous to the calibration approach presented in Section 4.2.2 where multiple camera images were rotated and translated into the coordinate frame of the 3D LRF. Here, multiple data sets from the same sensor from different points in time are rotated and translated into the coordinate frame of the latest data set. Results of this method are visualized in Figure 4.9. Due to self-occlusion and a minimum perceivable distance of approximately one meter, the area close to the vehicle is imperceptible for the robot. Using egomotion, classification results are carried over from the previous time step and data is available for continued navigation. This allows an interpolation in regions with few or no sensor measurements over small time gaps while larger regions (e.g., like holes or descents) are preserved. These results are published in [Häselich et al., 2013a] as joint work with Frank Neuhaus and Nicolai Wojke, who are mainly responsible

for the results presented in Section 4.5. The integration of a couple of sensor readings over time for the terrain classification task is strictly limited to a few laser scans due to the real-time criterion. In contrast to mapping approaches, a path planning algorithm needs the classified terrain grid as fast as possible. Therefore, the following section describes an evaluation performed on whole 3D maps, explicitly ignoring real-time criteria.

4.6 Evaluation on 3D Maps

Evaluation so far was carried out on single scans of around 120,000 3D points or as long-term measurements of performance and runtime. 3D maps are another research focus in outdoor robotics and provide an alternative to evaluate terrain classification. Using maps enables a more precise and convenient generation of ground truth and the evaluation can be performed on millions of points instead of thousands. For the process, precision is calculated as

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.13}$$

and recall is calculated as

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad . \tag{4.14}$$

The following evaluation aims to highlight strengths and limitations of the terrain classification algorithm by applying it to scenarios which diverge greatly from each other. Therefore, correctness and robustness of the algorithm were tested in the Koblenz city forest, on the Koblenz campus of the University of Koblenz-Landau, and on a farm road outside of Koblenz. Three data sets together with more than 137 million annotated 3D points were labeled by multiple human experts over several weeks. The MRF terrain classification usually operates with a configuration of cell size 0.51m $\times$ 0.51m and a grid size of 60m $\times$ 60m. For evaluation, grid size was altered and results were compared w.r.t. the particular data set. Configuration of MRF terrain classification itself remained fixed for all experiments, e.g., without any variations of feature or neighborhood component or the relation between the two (cf. Equation 4.5).

Two different types of evaluation are conducted. The first one is a *point-wise* evaluation, which means that every annotated point from the ground truth is compared to the class of the terrain cell that includes it. Since the result of the terrain classification is used for collision-free path planning, the algorithm should classify a terrain cell as an *Obstacle* if any *Obstacle* point lies within that cell. Any *Obstacle* point from the ground truth is considered a FP for the point-wise classification if the terrain cell is not an *Obstacle*. Any non-*Obstacle* point from the ground truth is in turn not considered a FP if the terrain cell is an *Obstacle*

and contains at least one *Obstacle* point from the ground truth. For all other cases, the class for the 3D point from the ground truth is compared to the class of the terrain cell it lies in. The second evaluation takes all other 3D points within a terrain cell into account and is therefore called the *cell-wise* evaluation. A terrain cell is considered a TP if the majority ($\geq 50\,\%$) of points from the ground truth share its class. The same exception for collision-free path planning from the point-wise evaluation applies here. Any *Obstacle* terrain cell is considered a TP for the cell-wise classification if the terrain cell contains at least one *Obstacle* point from the ground truth. Any non-*Obstacle* terrain cell is in turn not considered a FP if the terrain cell is an *Obstacle and* contains at least one *Obstacle* point from the ground truth.

The first scenario is a forest road in the Koblenz city forest. It was recorded on an uphill switchback road through a thick forest region where the class *Rough* is highly under-represented. Trees are strung together and besides the main road, only few patches of ground surfaces are visible to the sensors. Precision and recall are very high for the classes *Obstacle* and *Street* for all grid sizes. Results for the under-represented class *Rough* suffer from the mostly occluded forest surface. Additionally, the class is mainly present next to trees and is mostly overruled by the MRF neighborhood component; hence, recall values drop to 0.24 for the point-wise and to 0.43 for the cell-wise comparisons. A visual comparison of the ground truth against the result from the terrain classification algorithm is shown in Figure 4.10 and evaluation is presented in Table 4.5.

The second scenario is the university campus in Koblenz which is characterized by a semi-urban appearance. During map creation, several loops are successfully closed and the map is very compact due to the repetitive visits of the locations. This way, a change in the grid size not only influences the border regions of the single scans but whole areas are affected by this modification because they are observed from multiple directions. For example, the meadow marked with a green bullet in Figure 4.11 (b) shows this effect very clearly. Precision and recall are again high for the classes *Obstacle* and *Street* for all grid sizes. Recall for class *Rough* is around 0.45 for the point-wise and around 0.70 for the cell-wise evaluation. Precision is always above 0.60 in all experiments. A visual comparison is shown in Figure 4.11 and evaluation is presented in Table 4.6. The two scenarios indicate that the class *Rough* is the hardest to classify in regions with few fields, meadows, etc., which are the structures this class has been created for.

Results of the two previous scenarios motivate an environment where the class *Rough* is over-represented. Hence, the third and last scenario is a farm road with a lot of fields, meadows, and vegetation. This scenario is the most complex one due to rainy weather, puddles, translucent vegetation, and vibrations from the dirt road. A visual comparison is shown in Figure 4.12 and evaluation is presented in Table 4.7. The field in Figure 4.12 (b) marked with a green X, for example, is largely occluded by vegetation next to the road. Other difficulties arise from wet surfaces as some rays from the LRF are reflected and yield false

measurements. Overall, the MRF terrain classification approach performs well in this scenario. Recall and precision remain very high for the class *Obstacle* and the values for *Street* are good, too, with exception for precision in longer distances (grid sizes $\geq 60\,\mathrm{m}\times 60\,\mathrm{m}$). Precision for class *Rough* is always above 0.92 and recall is comparable to the other two scenarios.

In conclusion, the MRF terrain classification approach performs very well. Only the feature for the roughness classification requires enough points on the surfaces of the terrain cells to yield a precise result. The value of the terrain classification result for a path planning algorithm is described in the next section.

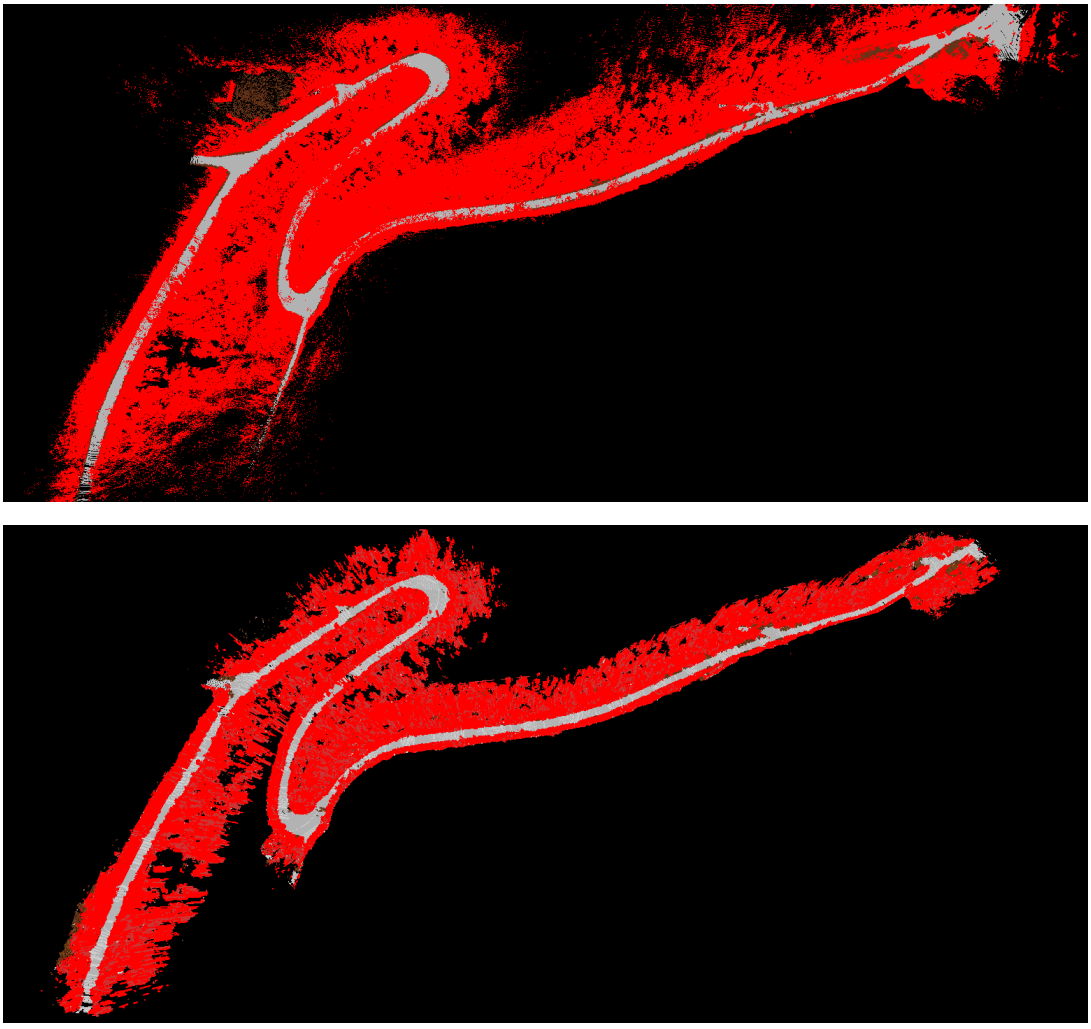


Figure 4.10: Visual comparison of the terrain classification on a forest road map. The map consists of over 26 million sensor measurements and is characterized by many trees along the sloped road and the ground surface is almost completely occluded. Human-annotated ground truth is shown in the upper and the output of the terrain classification algorithm is presented in the lower image. Evaluation is shown in Table 4.5.

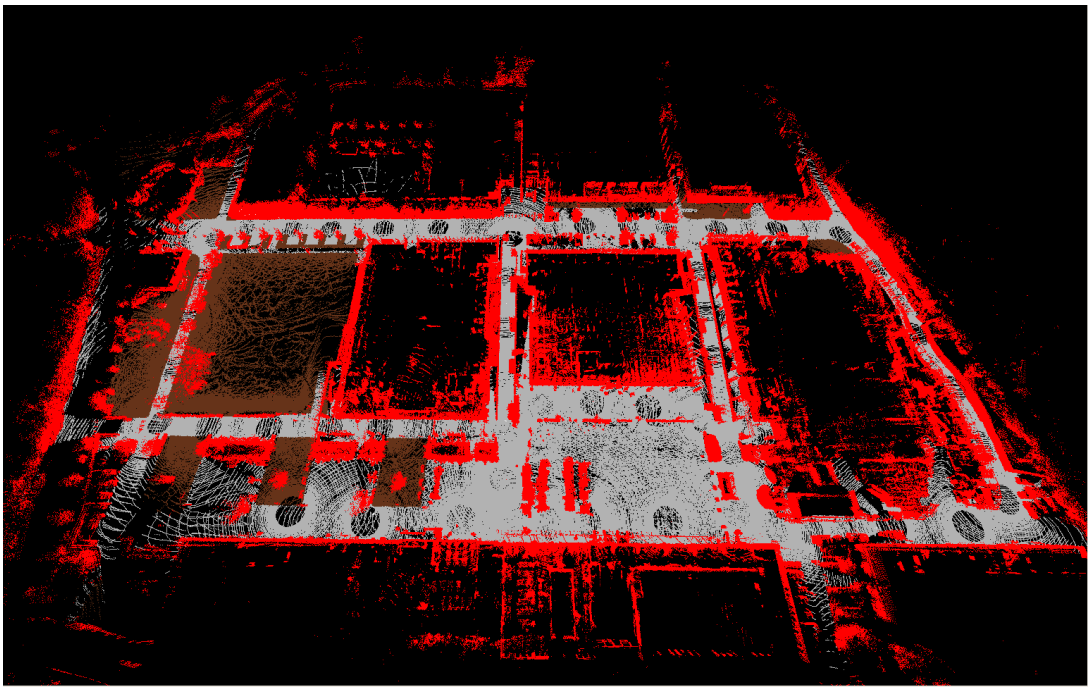
Name of data set:	Wald1Map	Number of Scans:	263
Number of Annotated Points:	26,288,749	Number of Points:	26,433,792

Grid Size [m×m]: 30×30	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 21,700,151				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	14,950,332	1,199,927	372,315	0.93	0.98	333,438	21,493	6,035	0.94	0.98
Street	3,984,321	293,956	899,287	0.93	0.82	121,645	5,060	14,099	0.96	0.90
Rough	322,804	123,988	1,029,202	0.72	0.24	19,626	10,314	16,953	0.66	0.54

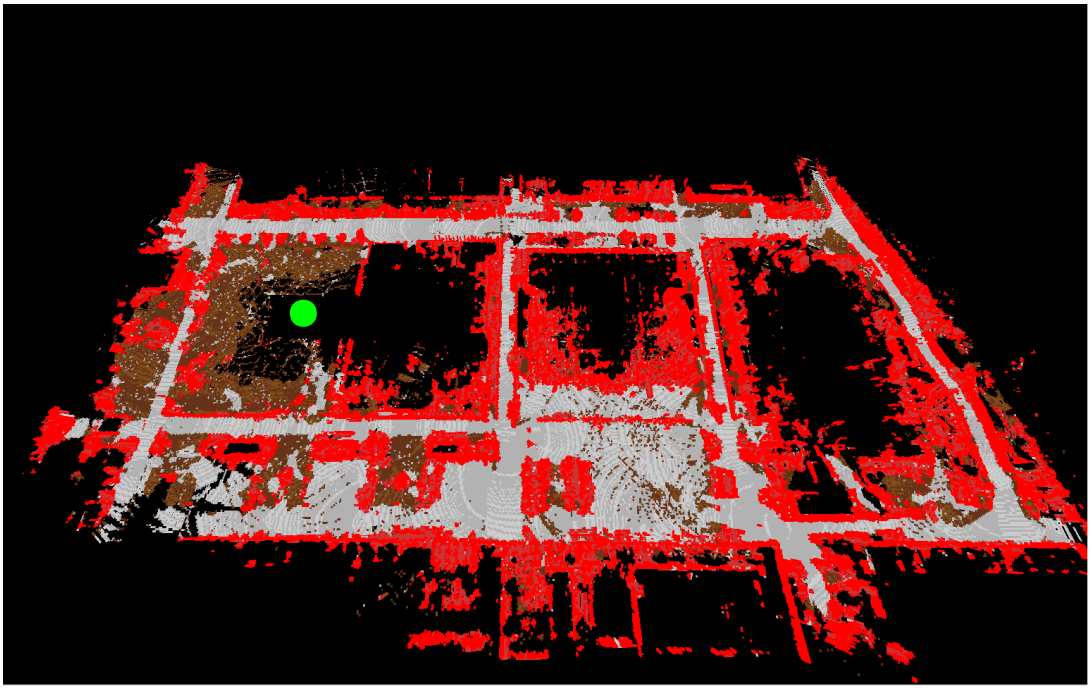
Grid Size [m×m]: 60×60	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 25,307,120				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	17,825,467	1,345,355	494,975	0.93	0.97	636,045	47,435	10,206	0.93	0.98
Street	4,354,376	327,368	969,382	0.93	0.82	194,235	10,767	27,949	0.95	0.87
Rough	369,539	161,694	1,149,871	0.70	0.24	30,169	18,080	38,347	0.63	0.44

Grid Size [m×m]: 100×100	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 26,165,412				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	18,413,829	1,366,655	651,715	0.93	0.97	723,626	53,347	11,782	0.93	0.98
Street	4,405,100	338,219	1,005,397	0.93	0.81	212,397	12,136	36,648	0.95	0.85
Rough	370,587	195,251	1,174,356	0.65	0.24	32,109	25,220	42,493	0.56	0.43

Table 4.5: Evaluation of the terrain classification on a large forest road map. The map consists of over 26 million sensor measurements and is characterized by many trees along the sloped road and the ground surface is almost completely occluded. Human-annotated ground truth is compared to the output of the MRF terrain classification algorithm for different terrain grid sizes.



(a)



(b)

Figure 4.11: Visual comparison of the terrain classification on a campus map. The map consists of over 64 million sensor measurements and is characterized by buildings, asphalt and road segments, some meadows, and small vegetated areas. Human-annotated ground truth is shown in (a) and the output of the terrain classification algorithm is presented in (b). Evaluation is shown in Table 4.6.

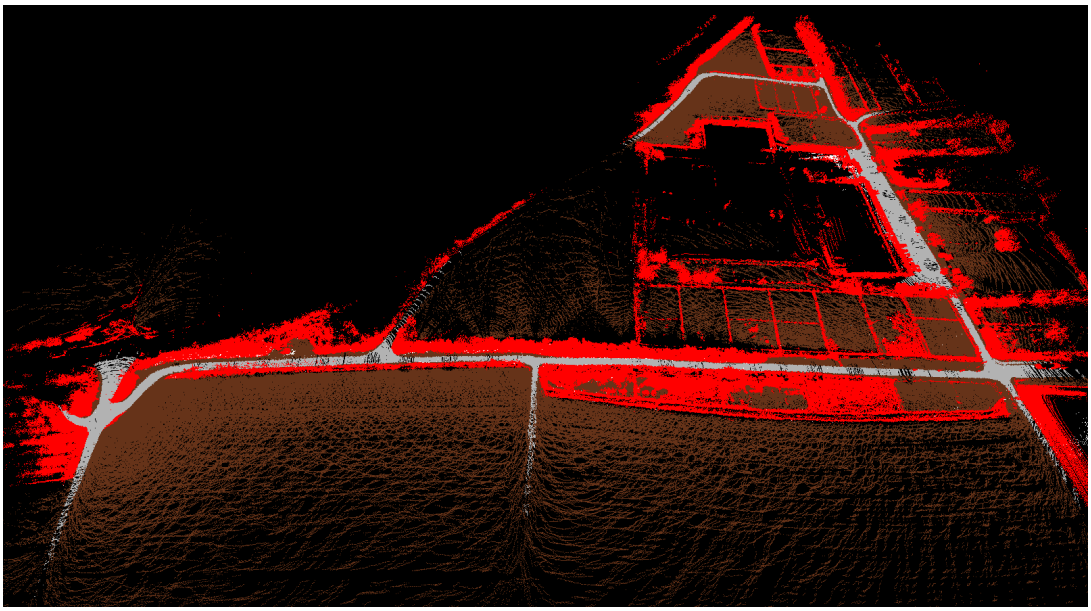
Name of data set:	megaLoopCampus	Number of Scans:	644
Number of Annotated Points:	63,038,659	Number of Points:	64,134,528

Grid Size [m×m]: 30×30	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 52,066,300				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	24,310,875	3,204,860	1,658,402	0.88	0.94	346,671	56,938	30,879	0.86	0.92
Street	17,280,858	1,606,462	3,850,540	0.91	0.82	604,101	39,133	94,417	0.94	0.86
Rough	1,589,828	983,717	2,287,426	0.62	0.41	121,348	75,559	46,778	0.62	0.72

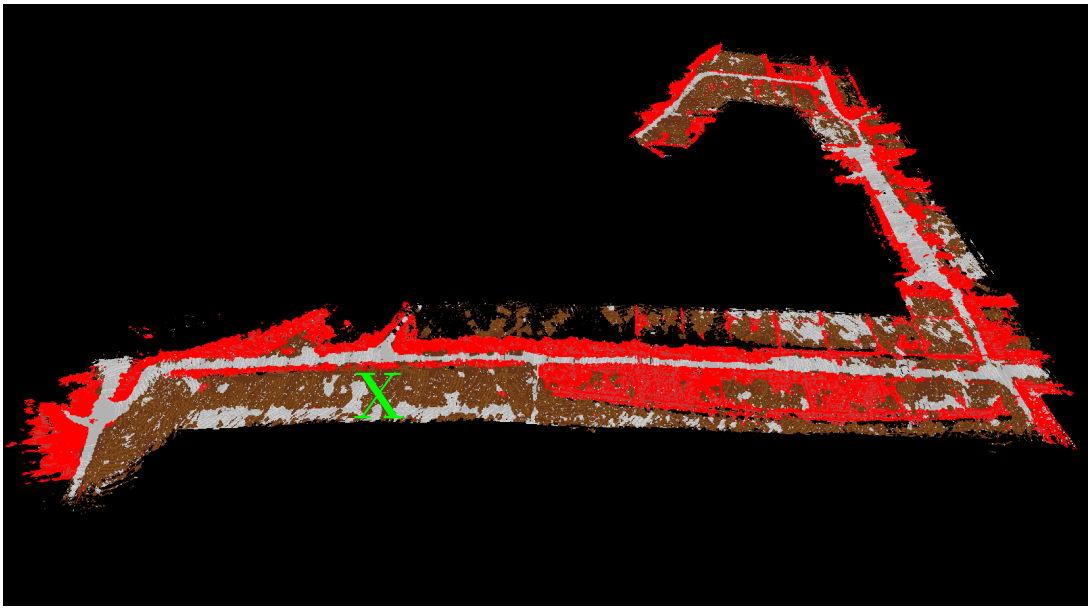
Grid Size [m×m]: 60×60	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 60,486,169				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	29,408,823	3,641,866	1,865,557	0.89	0.94	692,713	124,771	50,980	0.85	0.93
Street	19,102,630	1,781,887	4,413,649	0.91	0.81	983,515	78,095	194,465	0.93	0.83
Rough	2,070,151	1,262,633	2,535,066	0.62	0.45	239,296	140,748	98,613	0.63	0.71

Grid Size [m×m]: 100×100	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 63,001,171				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	31,146,895	3,739,610	2,061,640	0.89	0.94	901,388	149,155	58,523	0.86	0.94
Street	19,287,803	1,846,793	4,629,425	0.91	0.81	1,043,899	96,356	248,088	0.92	0.81
Rough	2,152,302	1,383,994	2,631,202	0.61	0.45	269,038	180,658	120,002	0.60	0.69

Table 4.6: Evaluation of the terrain classification on a large campus map. The map consists of over 64 million sensor measurements and is characterized by buildings, asphalt and road segments, some meadows, and small vegetated areas. Human-annotated ground truth is compared to the output of the MRF terrain classification algorithm for different terrain grid sizes.



(a)



(b)

Figure 4.12: Visual comparison of the terrain classification on a farm road map. The map consists of over 48 million sensor measurements and is characterized by rainy weather, puddles, and translucent vegetation compromising the field of view. Human-annotated ground truth is shown in (a) and the output of the terrain classification algorithm is presented in (b). Evaluation is shown in Table 4.7.

Name of data set:	Feldweg2Map	Number of Scans:	426
Number of Annotated Points:	48,192,619	Number of Points:	48,917,376

Grid Size [m×m]: 30×30	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 35,950,715				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	14,528,459	2,423,306	888,066	0.86	0.94	333,927	51,965	26,099	0.87	0.93
Street	6,868,690	3,066,429	3,485,905	0.69	0.66	193,183	91,750	10,383	0.68	0.95
Rough	4,292,002	248,935	5,193,304	0.95	0.45	301,586	25,042	132,648	0.92	0.69

Grid Size [m×m]: 60×60	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 43,619,222				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	17,753,915	2,750,770	1,158,561	0.87	0.94	629,002	112,859	60,820	0.85	0.91
Street	7,457,607	3,961,874	3,588,889	0.65	0.68	297,954	309,534	28,542	0.49	0.91
Rough	6,563,356	460,816	6,385,987	0.93	0.51	840,454	63,602	397,006	0.93	0.68

Grid Size [m×m]: 100×100	Cell Size [m×m]: 0.51×0.51					Classified Points [No.]: 46,751,857				
Label	Point-wise					Cell-wise				
	TP [No.]	FP [No.]	FN [No.]	Precision	Recall	TP [No.]	FP [No.]	FN [No.]	Precision	Recall
Obstacle	18,908,187	2,836,187	1,412,879	0.87	0.93	785,886	138,567	76,399	0.85	0.91
Street	7,492,263	4,379,626	3,645,895	0.63	0.67	308,685	452,815	42,460	0.40	0.88
Rough	7,495,081	566,796	7,077,159	0.93	0.51	1,162,399	83,640	556,536	0.93	0.68

Table 4.7: Evaluation of the terrain classification on a large farm road map. The map consists of over 48 million sensor measurements and is characterized by rainy weather, puddles, and translucent vegetation compromising the field of view. Human-annotated ground truth is compared to the output of the MRF terrain classification algorithm for different terrain grid sizes.

4.7 Spline Templates as Proof of Concept

Another way to evaluate the result of the MRF terrain classification algorithm is to analyze the result w.r.t. their usefulness for a path planning algorithm. A large number of navigation approaches have emerged over the past years that are specialized in off-road scenarios and focus on path planning on rough terrain surfaces [LMDM98, CMLL00, YSS00, FPM+05, KAB+06, ORMG07, AES+08]. Generic approaches (e.g., [HK07]) and grid-based strategies (e.g., [YB04, PS05]) also yield interesting results. Inspired by the success of the methods, the following generic grid-based strategy was developed to maneuver a robot autonomously through rough terrain based on the MRF terrain classification.

The data structure introduced in Section 4.3.1 can be used by a path planning algorithm to achieve fast runtimes. For this purpose, a structure element called *spline template* has been invented for the path planning of the robot. A spline template consists of a spline which represents a common approach to vehicle trajectory generation and a safety distance d which defines a corridor. Within this corridor, none of the terrain cells that share an overlapping area must be obstacles. This situation is illustrated in Figure 4.13 for two exemplary spline templates. The current location of the robot is depicted as a blue dot and its preliminary destination as a red dot. Terrain cells marked in green must be free from obstacles in order to guarantee a collision-free navigation. Each spline template can be connected to another, allowing a generic trajectory generation over arbitrary distances. The selection of the preliminary destinations can be realized using the Dijkstra's algorithm [Dij59] or the A* Algorithm [HNR68], for example. For the Mustang MK IA robot described in Section 1, a set of around 30 spline templates is used for three speed intervals, yielding around 100 different templates. Therefore, the path planning algorithm can access spline templates with different bends matching the robots current egomotion speed. The major advantage of this method is that each spline template can be calculated in advance w.r.t. the properties of each robot. The wheel base of each robot differs, allowing different steering angles, and the size of the robot directly influences the safety distance d, thus influences the corridor width. With all spline templates computed in advance, runtime usage is reduced to efficient terrain cell inspections. Instead of intersecting a trajectory with obstacles, only relevant cells are tested if the corridor is free from terrain cells classified as *Obstacle*.

Table 4.8 shows the performance of the spline template-based path planning approach in different outdoor tests. The measured computation times prove the efficiency of the MRF terrain classification. Results on the spline template-based path planning approach are presented in [Häselich et al., 2011b]. In the next section, approaches to detect and track dynamic obstacles in unstructured natural environments are presented.

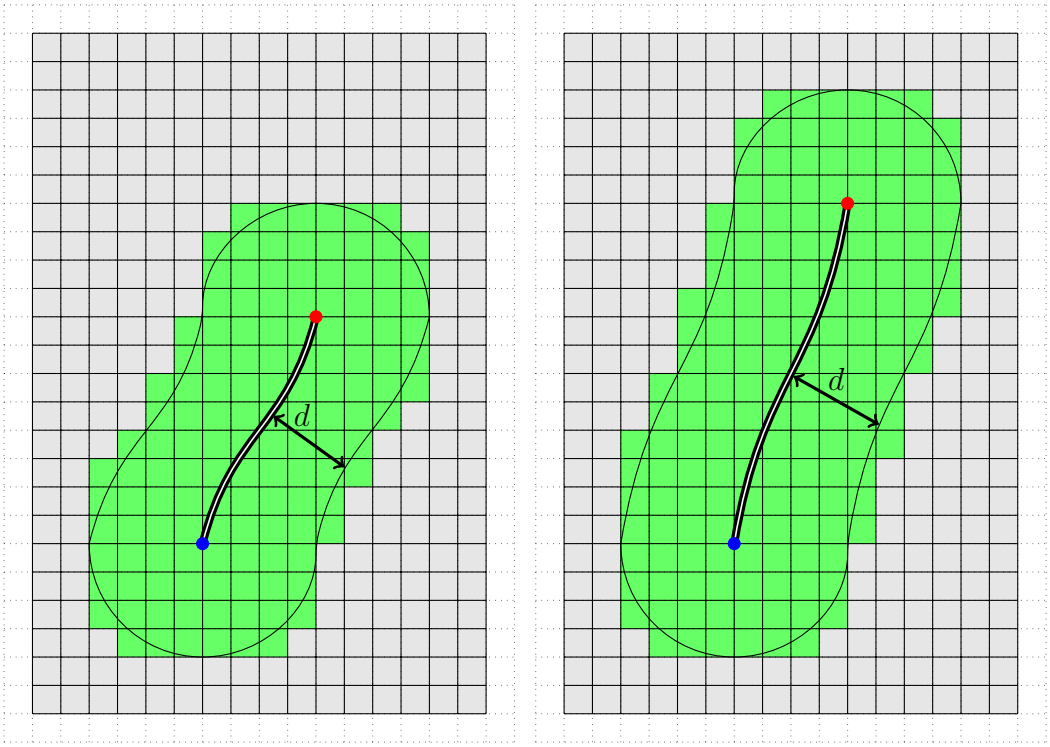


Figure 4.13: Example illustration of two spline templates for two different velocities. A spline template consists of a feasible vehicle trajectory which is depicted as double lines in the figure. Together with a safety distance d, it defines a corridor which has to be free of obstacles. It is depicted as a surrounding shell in which all relevant terrain cells are marked in green. The starting point is marked with a blue dot and the destination is marked with a red dot. The spline template on the left side can be used for slower velocities while the spline template on the right side is less curvy and can be used for faster velocities. The dotted lines at the borders indicate that the spline template is only applied to a small region within the whole terrain grid.

		Preprocessing		**Path planning**	
Scenario	**Duration**	**Mean**	**Std. Dev.**	**Mean**	**Std. Dev.**
	[min:sec]	[ms]	[ms]	[ms]	[ms]
campus	5:39	36.69	4.59	1.11	0.38
asphalt road	20:40	40.10	5.89	1.65	0.92
forest	12:45	52.73	7.42	1.38	0.63
farm/dirt road	11:34	43.93	7.56	1.61	0.92
city	26:53	39.62	5.59	2.42	1.71

Table 4.8: Computation time of a path planning algorithm using the terrain classification result in different scenarios. The table shows the performance of the preprocessing in form of the terrain classification and the path planning with spline templates on different kinds of surfaces and in various scenarios.

Chapter 5

Detection and Tracking of Moving Objects

Besides static obstacles and different terrain conditions, there are also dynamic obstacles in the environment of the robot which need to be detected as such. A successfully detected and tracked dynamic obstacle allows a prediction in which direction the object will move in order to prevent unintended behavior or even accidents. Different types of dynamic obstacles exist of which vehicles and pedestrians play an important role in the terrain classification context. The following section is divided into three parts. Section 5.1 describes a detection and tracking approach for dynamic vehicles in 3D LRF data. Pedestrian detection and tracking is split up into a camera-based (Section 5.2.2) and a laser-based approach (Section 5.2.3).

5.1 Detection and Tracking of Dynamic Vehicles in 3D LRF Data

Detection and tracking of dynamic vehicles is an essential task for collision-free autonomous navigation of outdoor robots. Unlike obstacles regarded in the terrain classification problem, dynamic vehicles are characterized by their motion. One aspect covers a reactive and foresightful path planning. Another less obvious aspect deals with the fact that other vehicles drive on the best available surfaces, too. Hence, it is possible to infer surface conditions from the trajectories of other vehicles, which will be exploited later (cf. Section 6). Being able to perceive other traffic participants is therefore a prerequisite for an advanced terrain classification in this scenario. In urban areas, vehicle movement is restricted to areas such as roads and parking lots. Generally, information about the road network can be used for background separation and motion prediction. In contrast, such information is not always available in unstructured environments, making vehicle

detection and tracking more challenging. The approach to vehicle detection and tracking developed for this thesis is independent of any prior knowledge of the environment, in particular of the road network. It handles vehicles of various appearance, size, and speed as well as partial and full occlusions over short periods of time. This approach uses the data of the Velodyne HDL-64E and is able to detect and track vehicles within 360° in large distances around the robot. The high frequency of the LRF allows algorithms to detect changes in a split second. For this purpose, an existing approach of Petrovskaya and Thrun [PT09] that was designed for the DARPA Urban Challenge in 2007 is extended. The approach heavily relies on road network information for background subtraction in order to reduce the number of false positive detections and to reduce the computational load of continuing vehicle hypothesis verification. Existing work is advanced to unstructured environments without limiting search space to areas of likely vehicle occurrence. An overview on related work in this field is presented in Section 5.1.1. In unstructured environments, potential candidates for dynamic objects occur almost everywhere due to changes caused by inaccurate egomotion estimates as well as leaves, bushes, etc. The applied sensor data interpretation is presented in Section 5.1.2. Therefore, a foreground separation is executed using temporal and geometric cues and described in Section 5.1.3. In order to deal with sensor noise and sporadic range readings, the two-dimensional virtual scan of Petrovskaya and Thrun [PT09] is extended to a Gaussian environment model that captures the local point distribution in the neighborhood of each reading. Following previous work [Vu09, ZT98], line and corner features are used to guide vehicle localization in areas of change. Vehicle tracking is briefly depicted in Section 5.1.4 and is carried out using a Rao-Blackwellized Particle Filter as described in [PT09]. The approach is evaluated on publicly available and own data sets in Section 5.1.6.

5.1.1 Related Work

Over the past few decades, a large number of detection and tracking approaches have been developed, especially for the DARPA Grand and Urban Challenges. Existing approaches can be divided into appearance-based and motion-based methods.

Appearance-based methods on the one hand allow an identification of dynamic objects even if they are currently not moving. Sun et al. [SBM06] give a survey on vision-based vehicle detection and tracking over the last 15 years. Most vision-based approaches locate potential vehicles based on visual cues. Hypothesis are then validated using template-matching or binary classification.

During the Grand and Urban Challenges, Broggi et al. [BCC+10] use a wide baseline trinocular camera system to generate range images of up to 50 m in front of their robot. After filtering ground readings, remaining points are clustered to generate object hypothesis. During the VisLab Intercontinental Challenge, the robot "Porter" is equipped with two stereo camera systems [BBS+10]. The

cameras provide information for line marking as well as vehicle and pedestrian detection. Both systems are supported by multiple LRFs for terrain analysis and long range obstacle detection.

Wender and Dietmayer [WD07] generate vehicle hypothesis in planar laser scans by fitting corner features to point clusters. For verification in camera images, two separate classifiers for lateral views and for frontal views of vehicles are trained. Transformation is obtained from the candidate's corner approximation in the planar laser scan.

In an approach described by Miller et al. [MCH+09], the output of a commercial vision-based vehicle tracking system is integrated into a multi-sensor environment representation. Camera images are used for candidate generation rather than for validation. However, vision is only a small fraction of the system that is built around a core of multiple laser and radar sensors.

Dietmayer et al. [DSS01] use only LRF data to detect traffic participants exclusively based on their geometry. The authors also fit oriented bounding boxes to point clusters in planar laser scans in order to obtain geometric extends.

Morris et al. [MHM08] use a combination of 2D and 3D scans to identify vehicles in cluttered environments. The 2D scans are used for hypothesis generation which are further examined in the 3D scans. A linear *support vector machine* (SVM) is trained to discriminate vehicles from clutter.

Lai and Fox [LF10] use a SVM as well. They discriminate between multiple classes of objects based on geometric features generated from the data of a Velodyne HDL-64E.

Motion-based methods on the other hand create candidates in areas of change and therefore only detect moving objects. Explicit discrimination based on appearance is avoided in order to to overcome high intra-class variance. Instead, motion is used as a predominant feature for object detection.

Wang [Wan04] detects moving objects as a pre-filter for simultaneous localization and mapping. The environment is represented by two distinct occupancy grid maps: one for static and one for dynamic objects. In a related project, Wang et al. [WTS03] use a motion-based approach to detect pedestrians and vehicles in planar laser scans. For object tracking, an extended Kalman filter with Multiple Hypothesis Tracking [CH96] is used.

Coué et al. [CPL+06] provide a Bayesian formulation of occupancy maps that associates a velocity with each grid cell. Cell movements are predicted using a linear velocity model. Gindele et al. [GBSD09] formulate *Bayesian Occupancy Filter Using Prior Map Knowledge* (BOFUM) to incorporate a priori knowledge about the street layout. In order to reduce computational expenses for BOFUM, Brechtel et al. [BGD10] introduce a probabilistic sampling-based solver.

A comprehensive collection of the scientific results obtained at the DARPA Urban Challenge by participating finalist teams is given in [BIS09]. The robot "Boss" of team Tartan Racing [UAB+08] is equipped with a combination of in total 18 laser and radar sensors, including a Velodyne HDL-64. Lasers are used

to cluster points, whereas radars are used for speed estimations. In close and medium range, line and corner features are fit to object clusters. Depending on the geometric approximation, objects are tracked using the bicycle model [KWSD04] or a constant acceleration model.

Team MIT [LHT+08] uses a combination of 12 planar LRFs and a Velodyne HDL-64E for environment perception. Additionally, 15 radar sensors cover the area around the vehicle in order to obtain precise velocity estimates in long distances. Environment is represented by an array of cells where each cell stores information about the occupancy of the area it represents.

The Stanford Racing Team [MBB+08] is another participant of the DARPA Urban Challenge. Petrovskaya and Thrun [PT09] use a model-based approach for vehicle detection and tracking from 3D laser data by extracting a two-dimensional planar scan. Their model is used for precise motion estimation instead of vehicle classification and their approach does not require data association.

5.1.2 Sensor Data Interpretation

The system presented here is based on the approach of Petrovskaya and Thrun [PT09]. The authors introduce a measurement model for vehicle tracking with planar LRF that allows direct interpretation of range data. This section describes the measurement model as well as the method used to generate a two-dimensional scan representation from data of a 3D LRF.

Vehicle tracking is a problem in two-dimensional space as vehicles always remain in contact with the ground. However, the 3D data of the Velodyne HDL-64E can be used to filter ground readings as perceived by planar LRFs in uneven terrain. Petrovskaya and Thrun [PT09] describe the extraction of a two-dimensional virtual scan from Velodyne HDL-64E data with a simple ground model. Therefore, all readings are projected into a 3D grid in polar coordinates around the vehicle center (cf. Figure 4.8). To filter noise, a representative reading is chosen for each cell as the median of distances. A model of ground elevation is established by comparing neighboring cells with the same horizontal bearing θ. If the slope is smaller than a threshold, the point is classified as ground reading. The distance to the closest obstacle in a target height from 50 cm to 200 cm is used as entry in the virtual scan (cf. Section 4.5).

The measurement model [PT09] approximates vehicle geometry as a rectangular shape of non-zero depth (cf. Figure 5.1). The likelihood of range readings is modeled as piece-wise constant function over the domain of range readings $[r_{\min}, r_{\max})$. Naturally, the likelihood of a reading falling onto the vehicle surface is highest. Due to self-occlusion, there are at maximum two sides visible at a time. Rays that extend through the visible vehicle surface into and beyond the vehicle interior are assigned a low likelihood. Similarly, range readings that fall short, but not in the close vicinity of the vehicle, receive a penalty. However, the likelihood of short readings remains high, as occlusion is common in dynamic

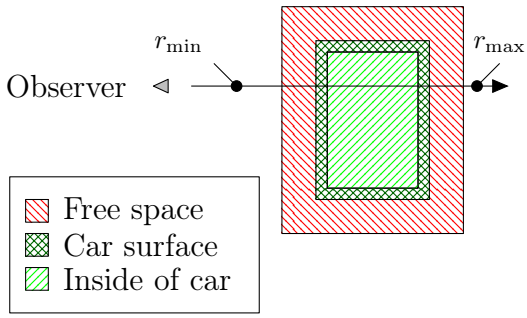


Figure 5.1: Geometric regions involved in measurement model computation as described by Petrovskaya and Thrun [PT09].

environments, caused by objects such as road signs, pedestrians, or other vehicles. By assigning a minimum likelihood to readings in the close vicinity of the vehicle, the measurement model enforces a region of free space.

5.1.3 Foreground Separation

Natural unstructured environments are characterized by a low signal to noise ratio. While LRFs provide precise geometric measurements in indoor environments, in the outdoors, vegetation causes sporadic and large changes due to its scattered appearance. Additionally, environmental factors such as wind cause small and continuous changes. Errors in data reduction of full 3D point clouds to 2D virtual scans reinforce the effect.

While Petrovskaya and Thrun [PT09] restrict vehicle detection to the area close to the road, this approach explicitly does not use road network information to reduce the number of false positive detections and to reduce the computational load of continuing vehicle hypothesis verification. Consequently, a foreground model that incorporates geometric as well as temporal cues is introduced. With interest only in moving vehicles (as static vehicles are already covered by the terrain classification), the foreground is defined as all points that fall onto approximate piece-wise planar object structures that exhibit motion.

Movement is perceived as change in the environment. To distinguish movement from background noise, a model of Gaussian distributions is created. Let z_i be the range reading of the i-th ray in the virtual scan. Then z_i is the median of the set of readings $C_{i,j}$ of the j-th cell in the i-th cone of the spherical grid, where j is the index of the cell with the closest obstacle. The normal point distribution is computed from the set of readings $C_{i,j}$ and its eight-neighborhood in the spherical grid to obtain samples $X_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ at time t for all rays in the virtual scan.

At time steps $t-1$ and t, samples X_{t-1} and X_t are compared to detect change. If sample X_{t-1} is classified as background and sample X_t is classified as foreground, a change in the environment is accepted. If both samples are classified

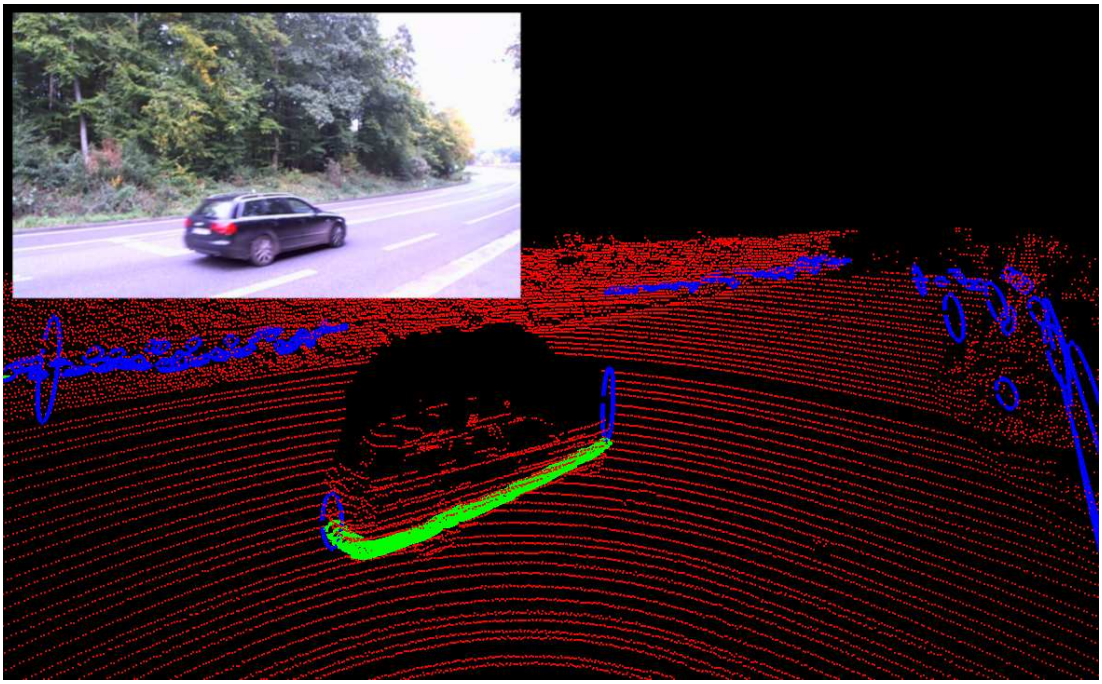


Figure 5.2: A vehicle on a rural highway as perceived in the model of Gaussian distributions. Covariances of local point distributions are drawn as ellipses (blue: background; green: foreground).

as foreground, their means are compared using the squared Mahalanobis distance (cf. Equation 2.38). A change in the environment is accepted if D^2 is greater than a threshold t_d.

To account for situations where the normal point distribution is an imprecise approximation, the size of the largest eigenvalue of the covariance matrix is thresholded. A sample is classified as background if the largest eigenvalue of covariance $\mathbf{\Sigma}_t$ is larger than a threshold t_{σ^2}. Hence, translucent and scattered objects are not considered in vehicle detection.

Figure 5.2 shows a vehicle on a rural highway. The vehicle has a piece-wise planar surface with high point density and is therefore considered as foreground. Vegetation and clutter in the distance are classified as background due to their scattered appearance. The Gaussian environment model is computed from the data of the 3D spherical grid rather than the 2D virtual scan. Accordingly, planar structures in the 2D virtual scan are not considered foreground if readings sporadically hit the surface (e.g., a mid-height fence). A visualization of the Mahalanobis distance for the scene displayed in Figure 5.2 is given in Figure 5.3.

Line and corner features yield an accurate description of the local vehicle geometry. They support the underlying rectangular geometry of the measurement model. To acquire local shape estimates, a region-growing segmentation is performed in areas of change to obtain point clusters of local geometry. Then, line

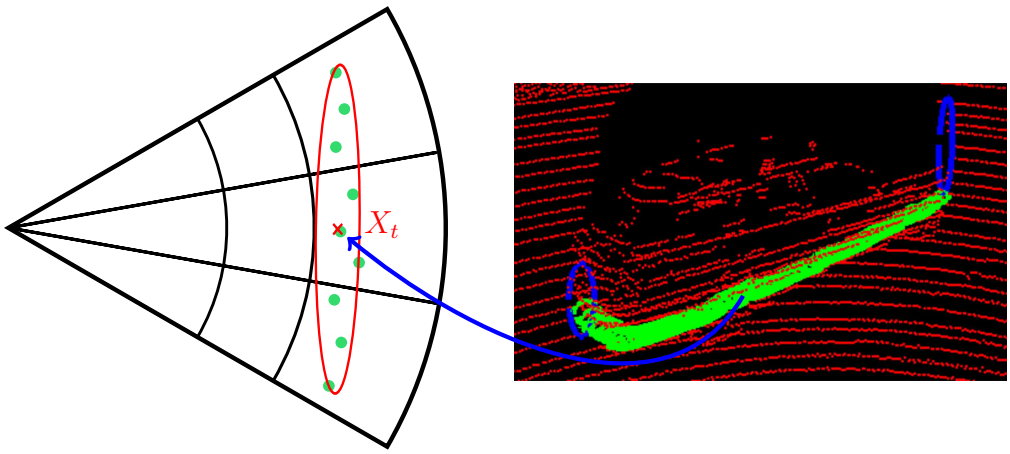


Figure 5.3: Illustration of the Mahalanobis distance of a virtual scan. The means of two foreground samples from consecutive instants of time are compared. The Mahalanobis distance is illustrated for one exemplary point in the left image where the point corresponds to a measurement on the surface of the vehicle on the right image (indicated by the blue arrow).

and corner features are fitted using a *random sample consensus* (RANSAC)-based [FB81] approach. The following two models are fit to data:

Two-sided model: Due to self-occlusion, there are at maximum two sides visible at a time. A set of two perpendicular lines is used to describe corner configurations.

One-sided model: If there is only one side visible, a single line is used to describe all readings within the point cluster. The one-sided model is a degenerate case of the two-sided model, where the position of the shorter side is free.

If the data does not fit either of the two models, the point cluster is disregarded.

Fitting a line $\boldsymbol{l} = (n_x, n_y, -d)^{\mathrm{T}}$ with normal $\boldsymbol{n} = (n_x, n_y)^{\mathrm{T}}$ and distance to origin d requires two points. As error function, the absolute Euclidean distance

$$\rho(\overline{\mathsf{P}}, \boldsymbol{l}) = |\overline{\mathsf{P}} \cdot \boldsymbol{l}| \tag{5.1}$$

between point $\overline{\mathsf{P}} = (\mathsf{P}_\mathsf{x}, \mathsf{P}_\mathsf{y}, 1)^{\mathrm{T}}$ and line $\boldsymbol{l}$ is used. Corner estimation requires three non-collinear points. The first two points define a line $\boldsymbol{l_1}$ and the third point defines the position of the orthogonal line $\boldsymbol{l_2}$ that passes through the point and its perpendicular foot on $\boldsymbol{l_1}$. The error is computed as the minimum error between $\overline{\mathsf{P}}$ and lines $(\boldsymbol{l_1}, \boldsymbol{l_2})$ as

$$\rho(\overline{\mathsf{P}}, (\boldsymbol{l_1}, \boldsymbol{l_2})) = \min(\rho(\overline{\mathsf{P}}, \boldsymbol{l_1}), \rho(\overline{\mathsf{P}}, \boldsymbol{l_2})) \quad . \tag{5.2}$$

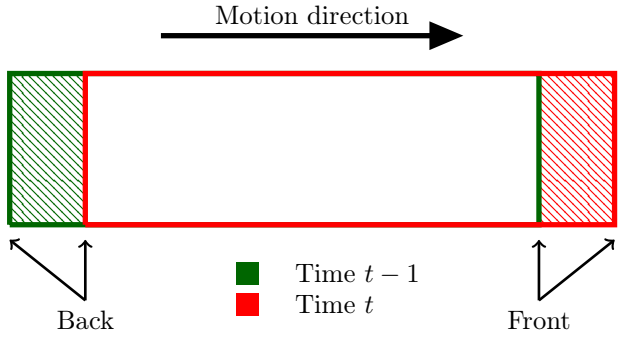


Figure 5.4: Areas involved in motion evidence score computation [PT09].

The error threshold is configured to equal the vehicle surface width of the measurement model. RANSAC is configured to terminate according to the adaptive stopping criterion where the number of iterations is updated based on the highest inlier ratio observed so far [HZ03].

The framework of RANSAC for (quasi-)degenerate data (QDEGSAC) [FP06] is used to deal with quasi-degenerate configurations where few readings fall onto the shorter side. In a series of RANSAC runs, QDEGSAC identifies degenerate configurations and selects the best fitting model based on the ratio of inliers to outliers.

5.1.4 Vehicle Detection and Tracking

The success of previous studies encourages the use of motion as a predominant feature for the detection of moving objects. Therefore, a motion-based approach is used for vehicle detection and tracking in unstructured environments.

First, vehicle candidates are separated from background. Then, candidates are localized that are not yet explained by an existing track. In two consecutive frames, candidates are validated based on their motion pattern.

Given local shape estimates from the foreground test, vehicle localization is necessary to handle partial occlusions where the object is split up into multiple clusters. Similar to [PT09], the Scaling Series algorithm [PKTN07] is used for vehicle localization. In order to derive a more relaxed version of the measurement model, vehicle surface width is inflated proportional to the sample radius.

With local shape estimates available, the vehicle pose is roughly known. For corner configurations, the pose is known up to a 90° ambiguity in orientation. Considering line configurations, an additional search for the position along the given shape estimate is performed.

Search space is defined by two spheres that resolve the 90° ambiguity in orientation. With an approximation of shape available, it is not necessary to consume the entire free space in early iterations, because position and orientation

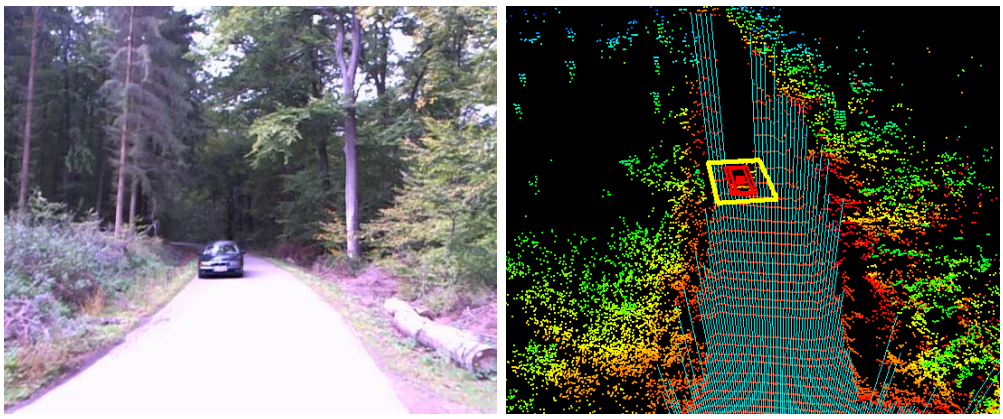


Figure 5.5: Camera view and laser data with a virtual scan of the detection and tracking of a driving vehicle. The example scenario shows the successful detection and tracking of a driving vehicle (yellow) in a difficult unstructured forest region.

are roughly known. In total, the algorithm runs four iterations until the target vehicle surface width is reached.

For track validation, the *motion evidence* score of Petrovskaya and Thrun [PT09] is used. The motion evidence score describes how likely change was caused by a moving vehicle, considering the areas that altered due to movement. Figure 5.4 shows a moving vehicle in time steps $t-1$ and t. The area in front of the vehicle at time $t-1$ was previously free and became occupied. The area behind the vehicle in time step t was previously occupied and became free. The motion evidence score requires knowledge of the candidate's speed. However, when a candidate is first localized, the motion pattern can be tested against a minimum velocity $v_{\min}$ to quickly rule out changes that do not fit the model.

For vehicle tracking, a Rao-Blackwellized Particle Filter is used that samples directly from the measurement model. A detailed description is available in the original paper [PT09]. Experiments revealed that spurious measurements in the background are not hindering tracking performance. Therefore, all points in the virtual scan are used for vehicle tracking, including points that were previously classified as background. An example of a successfully tracked vehicle in a forest environment is shown in Figure 5.5. The approaching vehicle is shown in the camera image on the left and the tracking is shown as a yellow rectangle in the virtual scan on the right.

5.1.5 Egomotion Estimation

Egomotion estimation is another essential aspect of the system. With an imprecise or unavailable estimation, the movement of the robot causes false alarms in form of vehicle candidates from stationary objects. Therefore a localization module is integrated in the software framework (Chapter 3) to solve this task. A

	0 m–30 m	30 m–50 m	50 m–80 m
Average detection time [No. frames]	3.38	3.50	11.33
Minimum detection time [No. frames]	3	3	10
Maximum detection time [No. frames]	5	5	12
Number of misses [No.]	0	0	6

Table 5.1: Number of frames required for vehicle detection and number of total misses in 0 m–30 m, 30 m–50 m, and 50 m–80 m.

description of the localization module developed by Volk [Vol09] and used in this approach is available in Section 4.5.

5.1.6 Evaluation

The system is evaluated with regard to detection times in ranges 0 m–30 m, 30 m–50 m, and 50 m–80 m. The test sequence contains in total eight vehicles that drive towards the robot with velocities between 60 km/h–80 km/h. To enable multiple detections in different ranges, the tracker is configured to drop tracks after one frame. Table 5.1 summarizes the results. In the range of 50 m–80 m only two out of eight vehicles were detected. Detection times ranged between 10 and 12 frames. Due to the limited angular resolution of the Velodyne HDL-64E, only few readings fall onto the vehicle surface in long distances. Furthermore, inaccuracies in ground detection cause noisy measurements that are disregarded by the foreground model. In ranges 0 m–30 m and 30 m–50 m all vehicles are successfully detected. Detection times average between three and four frames. The maximum detection time is five frames for both ranges. With at least three frames required for vehicle detection, the system performs close to the theoretical minimum in ranges up to 50 m.

The overall performance is further evaluated on the publicly available DARPA Urban Challenge data set [HAO+11]. The data set depicts the DARPA Urban Challenge finals from the perspective of team MIT's robot "Talos". Evaluation is carried out by counting total number of detections and misses on mission files 2 and 3. Both missions sum up to a total of 4 hours and 40 minutes of autonomous driving over a distance of 62.7 km. Vehicles were considered for detection if they were within 50 m of the robot and visible in the virtual scan, e.g., not fully occluded by other objects. Moving vehicles are counted as missed in two situations: 1. if the vehicle is not detected and 2. if the track is not maintained during close contact. Specifically, vehicles must be tracked when entering an intersection and when passing the ego-vehicle in close range. Qualitative measures, such as minimum and maximum detection times, are omitted due to missing ground truth.

Table 5.2 summarizes the results. A total of 361 out of 371 vehicles are correctly identified and tracked. Several misses occurred where vehicles are driving

Mission	Duration [s]	Distance [km]	Detected [No.]	Missed [No.]
2	5,428	21.5	153	6
3	10,414	41.2	218	4

Table 5.2: Summary of overall performance on the DARPA Urban Challenge data set of Huang et al. [HAO+11].

	Runtime: mean / standard deviation in ms				
Scenario	Scan	Detection	Validation	Tracking	Total
City center	23 / 3	5 / 3	12 / 13	13 / 9	54 / 15
Forest	26 / 4	12 / 5	15 / 14	0 / 0	53 / 16

Table 5.3: Mean and standard deviation of runtimes in a city and in a forest scenario.

in the opposite direction on a divided highway. The view onto the other side of the road is obstructed by evenly spaced trees on the median. Consequently, vehicles are partially occluded with only few readings falling onto the vehicle surface. Furthermore, it has been revealed that the initial detection of slow moving objects is challenging. Several misses are caused by slow moving traffic in the middle of the road or close to an intersection. Near misses occur at intersections with approaching vehicles that are overly careful (to slow). Generally, these vehicles are detected when entering rather than on approaching the intersection. However, once these vehicles are detected, tracking remains stable. Occasionally, the robot approaches an intersection where another vehicle is already waiting. As vehicle localization is initiated in areas of change, the vehicle is not detected until it moves off the stop line.

For runtime evaluation two data sets are analyzed. The first data set was recorded during a trip through the congested city center of Koblenz. It is characterized by frequent traffic and partial as well as full occlusions, with up to ten vehicles being tracked simultaneously. The second data set was recorded in a natural forest environment. The data are characterized by vegetation and clutter. However, there was no encounter with other traffic participants. Table 5.3 summarizes runtimes on a 2.53 GHz Intel Core 2 Duo. The system performs equally well in both scenarios with runtimes of in average under 55 ms per frame, which is lower than the update frequency of the Velodyne HDL-64E. These results are published in [Wojke and Häselich, 2012].

5.2 Detection and Tracking of Pedestrians

Autonomous robots need to perceive pedestrians in their environment in order to avoid collisions or to interact with them. Unlike vehicles, pedestrians possess a

much more arbitrary movement set since they are not bound to a certain driving direction. Furthermore, they are smaller and move much slower than vehicles, making them very hard or even impossible to detect by motion only. Therefore, shape-based approaches are widespread since they allow detection and tracking of stationary pedestrians and techniques can be applied to several sensor types.

5.2.1 Related Work

Systems using camera images for pedestrian detection and tracking in outdoor scenarios are widespread and an active research community is addressing this topic. In the past years, a large number of approaches emerged and significant progress could be observed. A recent publication of Dollar et al. [DWSP12] especially addresses this situation and presents an elaborate evaluation of sixteen state-of-the-art detectors. Evaluation yields that despite the steady progress over the last years, there is is still room for improvement in case of low resolution images or for partially occluded pedestrians.

Regarding the components of current detectors, the trainable system of Papageorgiou and Poggio [PP00] can be seen as pioneering work for many approaches. Their approach computes a feature vector from a detection window that is classified afterwards by a SVM [CV95] with a polynomial kernel. The authors present a large number of features and select relevant ones manually to detect faces, persons and cars.

Viola et al. [VJS03] present an approach that is based on their previous work on facial recognition [VJ11]. Their approach uses AdaBoost [FS95] combined with decision trees for learning. Classification is distributed on a cascade of true/false decisions and is further accelerated by computation principles for rectangular sums presented by Crow et al. [Cro84].

Dalal and Triggs [DT05a] present the "Histograms of Oriented Gradients" (HOG) feature which is able to encode the shape of a pedestrian in a robust way. It is computed from normalized three-dimensional histograms quantized in position and gradient orientation. Dalal and Triggs use a linear SVM for classification. Prisacariu and Reid [PR09] demonstrate the real-time capability of the HOG feature without losing quality using GPGPU techniques. In a subsequent approach, Dalal and Triggs [DTS06] introduce the feature "Histograms of Oriented Flow" (HOF). This feature adopts the principle of HOG to the optical flow [HS81] to increase the classification quality.

Detection quality of HOG-based approaches can be increased through combination with complementary features and other learning approaches. Wojek et al. [WWS09] focus on moving images and examine the quality of HOG, HOF and Haar-Wavelet features in combination with different learning algorithms. Besides the combination of linear SVM and HOG introduced by Dalal and Triggs, the histogram intersection kernel SVM presented by Maji et al. [MBM08] is examined by Wojek et al. In addition, AdaBoost and the extension MLPBoost [CMBR12] are

used. The comparison shows that movement information is helpful for detecting pedestrians moving sideways.

Walk et al. [WMSS10] present a pedestrian detection system that uses a histogram intersection kernel SVM in combination with HOG, HOF, and a new feature called "Color Self-Similarity" (CSS). The CSS feature encodes the color similarity of all cells within a detection window and is supposed to describe color relations from objects and background.

Zhu et al. [ZYCA06] integrate AdaBoost with HOG features to realize a human detection system. The authors use the work of Porikli [Por05], who showed that the fast computation of rectangular sums on integral images is portable to histograms. An integral image is created for each histogram bin and the resulting integral histogram can be used to compute arbitrary rectangular histograms. The detector of Zhu et al. achieves an acceleration of factor 70, but is unable to uphold the quality of the original HOG variant.

Dollar et al. [DTPB09] generalize this idea and present the "Integral Channel Features". In their approach, the authors investigate various channels and use the boost approach to determine relevant features. For the detection of pedestrians, eight channels are used: gradient magnitude, gray-value image and gradient histogram (six channels).

Based on the Integral Channel Features, Dollar et al. [DBP10] publish a speed-optimized extension. Since gradients and gradient histograms are not scale-invariant, the authors approximate the necessary pyramid of features from different scales by using neighborhood information. Although detection quality is affected by this approximation, pedestrians can be detected in multiple images per second.

Felzenszwalb et al. [FGMR10] propose a part-based approach for pedestrian detection. Their approach uses the HOG feature as well and defines a two-dimensional star pattern to model the part-whole relation of the pedestrians. Felzenszwalb et al. choose a latent SVM which is able to encode the relative position of the parts as hidden variables.

Breitenstein et al. [BRL+09] present a tracking-by-detection approach for multiple persons using particle filters in color image series. Their implementation allows for the tracking of large numbers of moving pedestrians in 2D space without camera calibration or knowledge of the ground plane.

Modern 3D LRFs like the Velodyne HDL-64E enable detection and tracking within a 360 degree field of view around the robot. Recently, a number of approaches used the data of LRFs to detect and track moving objects in outdoor scenarios.

Scholer et al. [SBS+11] use a Velodyne HDL-64E LRF to detect and track people in 3D point clouds. Their approach allows for the tracking of partially and fully occluded persons over a certain length of time in indoor areas.

Spinello et al. [SLA11] present a combined bottom-up top-down detector for pedestrians in Velodyne HDL-64E data in urban outdoor environments. Their

approach is independent from ground plane assumptions and the described detector uses a layered person model. For tracking, a multi-target multi-hypothesis approach is used and their system achieves good results within a limited range of 20 m.

Navarro-Serment et al. [NSMH10] use geometric and motion features to detect and track pedestrians while driving in outdoor regions. Their algorithm detects objects using a virtual 2D slice and then classifies each object using statistical pattern recognition techniques. Kidono et al. [KMW+11] extend features of [NSMH10] by a slice feature and by reflection intensities of their Velodyne HDL-64E. Their approach classifies pedestrians with a SVM in road environments and is able to deal with low spatial resolution targets.

Thornton et al. [THM08] present a multi-sensor approach including a 3D LRF for human detection and tracking in cluttered environments.

A probabilistic person detector on multiple layers of 2D laser range scans classified using AdaBoost [FS95] is presented by Mozos et al. [MKH10]. Each layer detects a different body part and the conducted experiments reveal robust detection rates in cluttered environments.

Premebida et al. [PLN09] present a laser-based pedestrian detection system and focus on information extraction from LRFs. In their work, the authors explore and describe the potential of LRFs in pedestrian classification and present results with different classification techniques using automotive and industrial LRFs.

Carballo et al. [COY08] fuse multiple 2D LRFs on two layers to detect pedestrians in uncluttered indoor environments. Gidel et al. [GCB+10] present a pedestrian detection and tracking approach using a LRF system on multiple layers, too. The proposed detector fuses the information from four layers and tracking is performed with a particle filter. Sato et al. [SHT+10] use a LRF with six scanning layers to track pedestrians with a Kalman filter in urban environments. The approach maps objects with a certain height to a grid map and uses global nearest neighbor-based data association.

Despite the steady progress over the past years, there is still room for improvements. In particular, detection is disappointing at low resolutions, for partially occluded pedestrians, and especially in unstructured outdoor environments where uncertainties often occur. Furthermore, performance in terms of runtime and memory consumption is not yet optimal. In the following two sections, two different approaches are presented for camera- and laser-based pedestrian detection and tracking focused on unstructured environments and their specific challenges.

5.2.2 Pedestrian Detection in Camera Images

A key challenge for an image-based pedestrian detection system is the high variability of pedestrians. Persons differ in their form, pose, clothing, and color and hence possess a high inter-class variance. Other difficulties arise from the out-

door environment. The area around the sensor is unknown and heterogeneous and parts of the pedestrians can be occluded. Unknown ambient light conditions complicate the detection and may cause erroneous or falsified sensor data, especially for color values. In the following section, a complete system for pedestrian detection is presented according to state-of-the-art techniques. HOG [DT05a] is used to encode the shape and appearance of a pedestrian. Walk et al. [WMSS10] present CSS, a feature that describes color relations within a detection window and forms a well-suited addition to the gradient histograms. The approach of Walk et al. is modified and integrated into the robotic framework (Section 3). Computation of the color similarities is identified as a cost-intensive part and a new structure element that accelerates the computations is presented. This section is structured as follows. In Section 5.2.2.1, the pedestrian detector is described followed by the classifier in Section 5.2.2.2. The training procedure is depicted in Section 5.2.2.3, and evaluation is presented in Section 5.2.2.4.

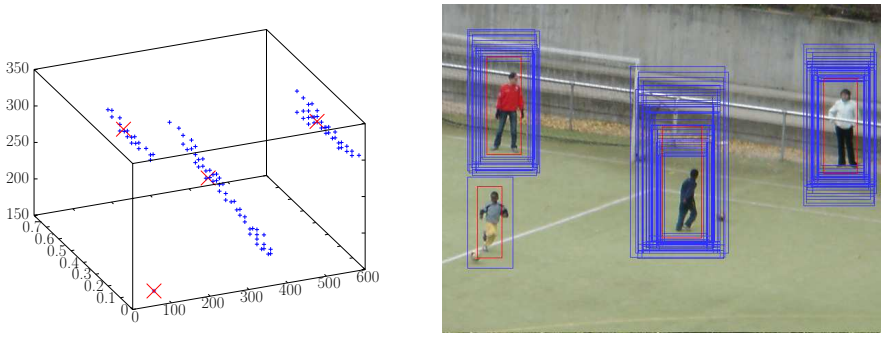
5.2.2.1 Detector

The search for pedestrians in an input image uses a sliding window framework with a fixed window size. Each window needs a binary classification if a pedestrian is sufficiently contained. A fixed window size alone results in a pedestrian detection that is limited to pedestrians of a fix size and is not scale-invariant. Hence, the input image is transformed into multiple scales and the fixed-size window operates on each scale with the given offset.

Classification of all windows on all scales yields a number of detections that accumulate around the pedestrians (cf. blue rectangles in Figure 5.6 (b)). Those multiple detections for a single target need to be aligned. Different approaches exist to realize such a thinning or *non-maximum suppression*. Dollar et al. [DTPB09] for example perform a pairwise comparison of overlapping windows and discard the ones with the lowest decision value. Here, the mean shift algorithm [CM02] is used as described by Dalal et al. [DTS06]. The mean shift algorithm assumes that its input points result from random samples of a single or multiple Gaussian distributions. In an iterative process, the algorithm estimates the mean of these distributions using the Mahalanobis distance. Figure 5.6 (a) shows an example in which detections are depicted as blue plus-signs and estimated means as red crosses. The corresponding bounding boxes are visualized in Figure 5.6 (b) where the red boxes are the ones selected by the non-maximum suppression.

5.2.2.2 Classifier

The task of the classifier is to perform a binary classification between pedestrian and non-pedestrian as precisely as possible. Therefore, a SVM is trained for the detection windows. A linear Soft-margin SVM [CV95, TK09] is used with a feature vector that will be described in Section 5.2.2.3. Given the two classes ω_1



(a) (b)

Figure 5.6: Non-maximum suppression using the mean shift algorithm. In the left image detections are marked as blue plus-signs and estimated means are red crosses, corresponding to the bounding boxes of the same color in the right image.

and ω_2 with corresponding feature vectors $\boldsymbol{x}_i$ with $i = 1, ..., l$, their affiliation is encoded by y_i defined as

$$y_i = \begin{cases} -1, & \text{if } \boldsymbol{x}_i \text{ belongs to } \omega_1 \\ +1, & \text{if } \boldsymbol{x}_i \text{ belongs to } \omega_2 \end{cases} . \tag{5.3}$$

For this task, the LIBLINEAR-library from Fan et al. [FCH+08] is used which is optimized for linear SVMs with a plenty of data.

5.2.2.3 Training Procedure and Feature Vector

For the training of the SVM, the INRIA data set [DT05a] is used which consists of 615 positive and 1218 negative samples. Positive samples are extracted from annotations and negative samples are generated at random at all scales from images without pedestrians. In order to reduce the number of false positives, an extended bootstrapping is performed in which the SVM is retrained with difficult samples. The model created in the first iteration is therefore applied on images without pedestrians on the whole scale space. Since all detections are false positives, they are added as such to the model. This step is repeatable and lowers the number of false negatives, but also influences the number of true positives. True positives that were close to the hyperplane before an iteration might not be detected again by the classifier afterwards. It can be observed that two bootstrapping phases adequately reduce the false negatives while preserving the number of true positives.

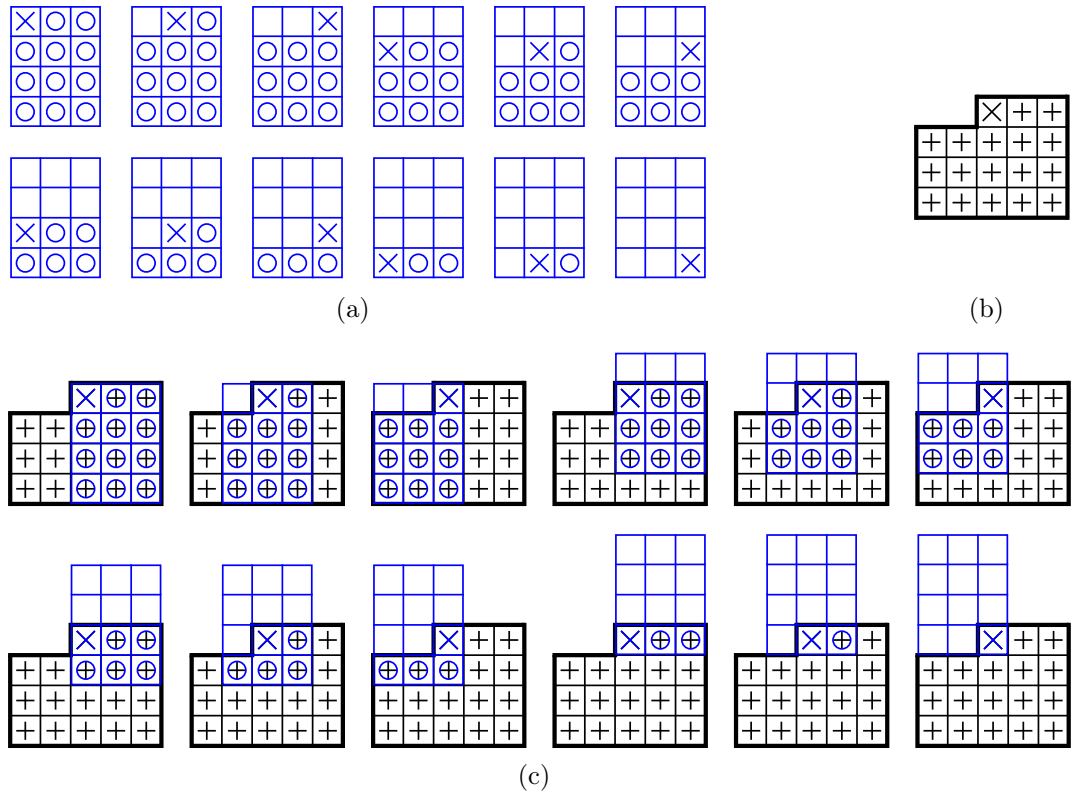


Figure 5.7: Application and benefit of the new structure element. Similarity computations for the CSS are shown in (a) and the structure element that is used to accelerate the computations is visualized in (b). The current cell is marked with an X, elements contained in the structure element are depicted as plus signs, and cells that are used for computation are marked with a circle. Extraction of the similarity computations from the new structure element is illustrated in in (c).

Features encode occurring patterns as multi-dimensional feature vectors. For the previously described detection windows, the HOG and CSS features are computed.

The HOG feature introduced by Dalal and Triggs [DT05a] is able to bundle the gradient information and still allows variations in the pose. It encodes the rough shape of an object and is tolerant to small changes of the shape. HOG-blocks can be precomputed for the entire input image and individual detection windows can be evaluated by the concatenation of all contained HOG-blocks. A detailed description of the HOG feature used in this work is available in [DT05a].

The second feature used in this work is the CSS-feature introduced by Walk et al. [WMSS10]. Complementary to the shape description of the HOG-feature, CSS encodes the color information. Color is a difficult feature in outdoor scenarios and for pedestrians as well. Pedestrians wear different clothes and their skin color can also vary. Images are taken from different cameras which produce different

color values. Furthermore, those images can be taken under various lighting and weather conditions. Those conditions complicate the use of color as a feature. Dollar et al. [DTPB09] have shown that integral channel features are able to record a peak in the Hue channel in the area of the head, but not for the rest of the body. CSS circumvents this limitation by considering color similarities instead of accessing the color values directly.

For the computation of the CSS-feature, the image is initially converted to the HSV space. The detection window is subdivided into quadratic cells of size $\zeta \times \zeta$ and for each cell a three-dimensional color histogram of 3×3×3 is constructed. The histogram is filled with the pixels contained in the cell and a trilinear interpolation is performed. Afterwards, for each cell the color similarity to any other cell of the window is computed. The feature is able to describe the similarity of body parts and the similarity of background regions, which are interrupted by the appearance of pedestrians (cf. Walk et al. [WMSS10]).

Computation of the similarities for one window is illustrated in Figure 5.7 (a). In the image, a circle represents the computation of a similarity with the cell marked with an X.

Inspection of the sliding window reveals an acceleration possibility, which is to dissolve redundancies by precomputations. The approach works as follows. Firstly, the whole image is divided into cells of size $\zeta \times \zeta$. The goal is to compute all relevant similarities for all cells in advance. The window size determines the region called *support* that influences the feature of a single cell. For the example from Figure 5.7 (a) with a detection window of 3×4 cells, the support is constructed as the structure element shown in Figure 5.7 (b). The support $\mathrm{sp}_x \times \mathrm{sp}_y$ w.r.t. the number of cells of the detection window in x- and y-direction is defined as $\mathrm{sp}_x = 2c_x - 1$ and $sp_y = c_y$.

It is now possible to create a buffer for each cell of the image that stores the support information (cf. the example in Figure 5.7 (b)). Figure 5.7 (c) visualizes how all the similarities displayed in Figure 5.7 (a) can be extracted from the structure element shown in Figure 5.7 (b). Application of the structure element is possible for all cell windows w.r.t. boundary treatment. The evaluation described in the next chapter will reveal an acceleration of factor four without any loss in quality.

5.2.2.4 Evaluation

Evaluation of the implemented pedestrian detection system follows the instructions of Dollar et al. [DWSP12] in order to achieve a precise comparison with existing detectors. Ground truth consists of images from *InriaTest* [DT05a] and *TudBrussels* [WWS09] data sets (for validation) data sets. InriaTest contains 288 images of varying size, illumination, and scenery and is a loose collection of holiday photographs. The 508 images of TudBrussels are gathered from a camera mounted on a car driving through the city of Brussels, Belgium. In this data

Detector	Resolution								
	320×240			640×480			1280×960		
	μ	σ	fps	μ	σ	fps	μ	σ	fps
HOG-Dalal	n. a.	n. a.	n. a.	4.18s	n. a.	0.24	18.52s	n. a.	0.05
rhog8	0.17 s	0.01 s	5.69	0.94 s	0.02 s	1.07	4.17 s	0.09 s	0.24
rhog6	0.23 s	0.01 s	4.29	1.39 s	0.02 s	0.72	6.61 s	0.10 s	0.15
rhog6css8combo	1.50 s	0.01 s	0.67	13.89 s	0.03 s	0.07	78.99 s	0.11 s	0.01
rhog6css8structcombo	0.49 s	0.01 s	2.05	3.61 s	0.03 s	0.28	19.09 s	0.11 s	0.05

Table 5.4: Comparison of the different detector runtimes reflecting the runtime optimization achieved with the new structure element.

set, pedestrians are annotated in various poses from a size of 50 pixels upward, sometimes occluded by vehicles and other objects. Both image data sets are completely disjoint from the training data. In order to determine if the rectangle surrounding a detected pedestrian corresponds to the rectangle of a pedestrian annotated in the ground truth, the PASCAL-criteria [EVGW+10] is used. The criteria states that two rectangles sufficiently overlap if they share at least 50 % of their area, resulting in a true positive (TP) matching between detected pedestrian and ground truth. Bounding boxes of detected pedestrians that fail this criteria are classified as false positives (FP), vice versa as false negatives (FN) for bounding boxes of the ground truth.

In the case of multiple detected bounding boxes corresponding to a single ground truth annotation, only the one with the highest decision value will be used while the others are classified as FP. This in turn means that an effective approach has to conduct a very effective non-maximum suppression to reduce the impact of these false positives on the classification quality.

Quality of the detector is described by an evaluation curve whose linear x-axis describes the false positives per image and whose logarithmic y-axis encodes the miss-rate. False positives per image are computed as

$$\frac{\mathrm{FP}}{\#\,\mathrm{images}} \tag{5.4}$$

and the miss-rate is defined as

$$\frac{\mathrm{FN}}{\mathrm{TP}+\mathrm{FN}} \quad . \tag{5.5}$$

Figure 5.8 shows the results for HOG versus HOG + CSS classifiers on both data sets. The key in the lower part shows the miss-rate at 10^{-1} false positives per image as reference point.

As observed by Dalal and Triggs [DT05a], the cell size of the HOG feature is directly linked to the object that is to be detected. A cell size of 6 of the

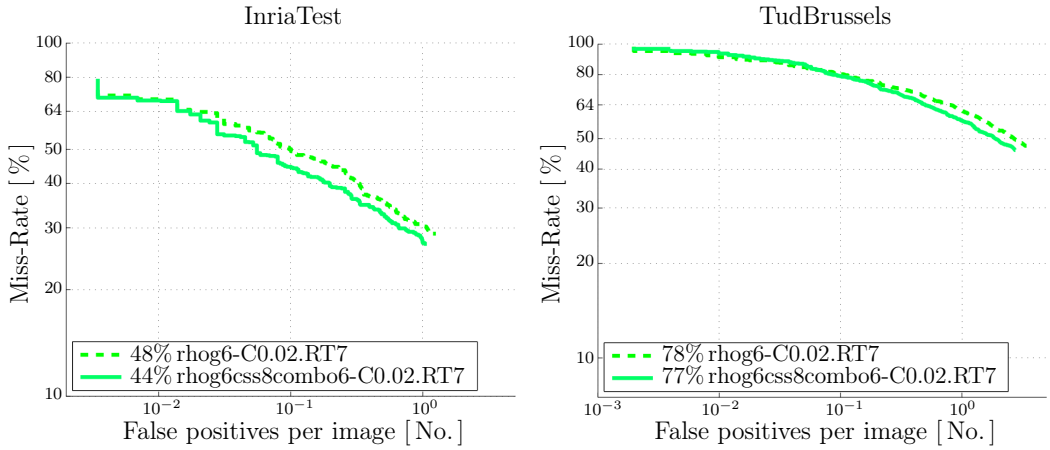


Figure 5.8: Comparison of the HOG-based classification versus the HOG + CSS-based variant. Evaluation was performed on INRIA (*InriaTest*) [DT05a] and TudBrussels (*TudBrussels*) [WWS09] data sets.

HOG feature yielded the best results during the experiments. Linear SVMs have a single parameter C which has a strong influence on the classification quality. In this work, a C-value of 0.02 yielded the best results. The cell size of the CSS feature is another parameter of the classifier. For the experiments, 8×8 pixels per CSS cell were used. The number of retraining phases (RT) is the subject of controversial discussion in literature. For example, Dalal and Triggs [DT05a] consider two retrainings appropriate while Walk et al. [WMSS10] take up the position that a large number should be carried out. In this work, two retraining phases (RT) mainly lead to convergence for the HOG feature alone whereas the combination of features requires seven retrainings.

Each combination of parameters can be seen as a new detector which is why the name of each classifier is formed from these parameters, e.g., as rhog6(HOG cell size)+css8(CSS cell size)-C0.02(SVM C-value)+RT7(# retrainings). The classifier is compared to the state-of-the-art approaches in Figure 5.9. Curves are generated from the script of Dollar et al. [DWSP12].

Runtime of the system is evaluated on the TudBrussels data set (508 images, resolution 640×480) and the result is shown in Table 5.4. Since runtime mainly depends on the size of the images, interpolated sizes of 320×240 and 1280×960 are used as additional material. Runtimes include loading data, detection itself, and non-maximum suppression. A detailed description of HOG-Dollar is available in [DWSP12]. The two variants rhog6 and rhog8 represent the feature with 6×6 and 8×8 pixels per cell. HOG features combined with the CSS features are represented by rhog6css8combo6 and rhog6css8structcombo, where the latter is the variant with the new structure element described in Section 5.2.2.3. The values for HOG-Dollar are taken from the original paper and the system used for

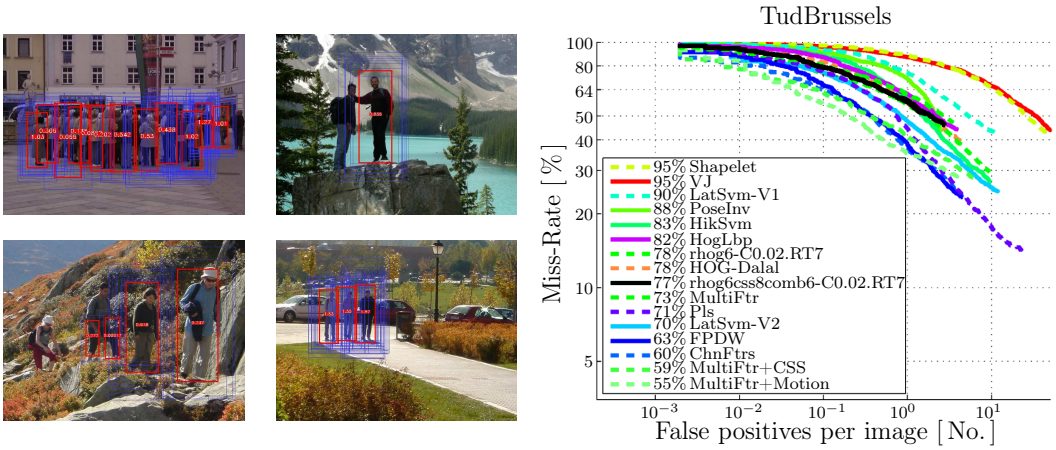


Figure 5.9: A selection of some detection results and a comparison with other state-of-the-art approaches. The legend follows the notation of Dollar et al. [DWSP12].

all other measurements is an Intel(R) Quad Core(TM) with 2.66 GHz and 8 GB RAM. These results are published in [Häselich et al., 2013b].

5.2.3 Pedestrian Detection and Tracking in 3D LRF Data

The following part describes a detection and tracking approach for pedestrians in 3D LRF data. It is divided by descriptions of ground removal (Section 5.2.3.1), clustering (Section 5.2.3.2), feature extraction (Section 5.2.3.3), classification (Section 5.2.3.4), measurement model (Section 5.2.3.5), tracking (Section 5.2.3.6), and evaluation (Section 5.2.3.7). The main focus here is a tolerant classifier with a solid measurement model combined with a rapid extinction of particles over time. This way, many candidates are accepted but rejected very early if they are not validated continuously.

5.2.3.1 Ground Removal

The task of ground removal is to separate obstacle points from ground points (cf. [KMW+11, NSMH10]) in order to reduce the computational load and to perform a first selection of candidates. In unstructured environments, an assumption of a planar ground surface with a fixed ground height is likely to fail. Planar ground regions are often interrupted by hills, ditches, and other surface variations. In this approach, 3D data are inserted into a three-dimensional grid with a resolution of 0.1 m×0.1 m×0.1 m per cell. Afterwards, cells are sorted according to their height, computed from the highest and the lowest point in each cell, from high to low. Each of those candidate cells is not only classified w.r.t. the contained points, but also w.r.t. the neighboring cells. A sliding window with a height

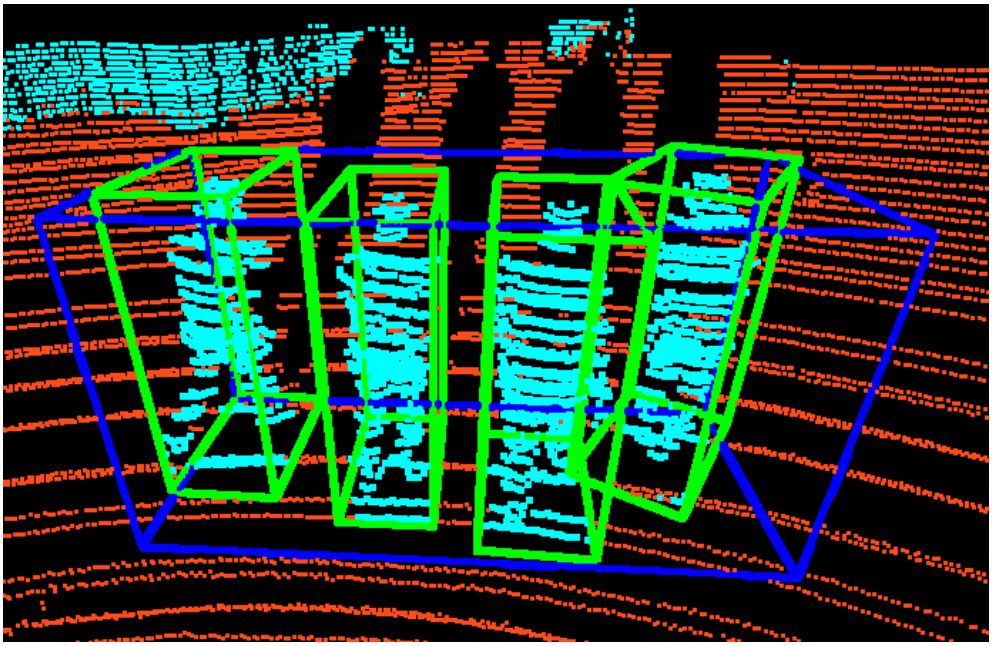


Figure 5.10: Accepted pedestrian candidates (green frames) extracted from a large segmented group (blue frame). The clustering algorithm separates large objects into smaller ones.

threshold is used to calculate the absolute difference between the highest and the lowest point among all points of the current cell and in addition all points of the neighboring cell. Based on the resulting values, a cell is classified as *Empty*, *Ground* or *Obstacle*.

5.2.3.2 Clustering

The segmentation algorithm yields groups of cells which contain obstacles. Afterwards, those groups need to be prepared for feature extraction and classification. On the one hand, the clustering algorithm needs to cluster obstacle cells to groups of sufficiently large 3D distance measurements which represent pedestrian candidates. On the other hand, the algorithm needs to split up large groups of 3D points in order to separate pedestrians from other obstacles which are close to them, including trees, buildings and other pedestrians. Unstructured outdoor scenes are more complex than urban environments considering different elevation levels in rough terrain, diverse vegetation, and sensor noise.

The following approach consists of two steps and aims to find a high number of pedestrian candidates even if they are located close to other objects. Thus, the next step is to further analyze clusters that are too large to represent a candidate, which in turn yields increased runtime. In order to keep the overall runtime low, several algorithms are inspected. Approaches like k-means [Mac67]

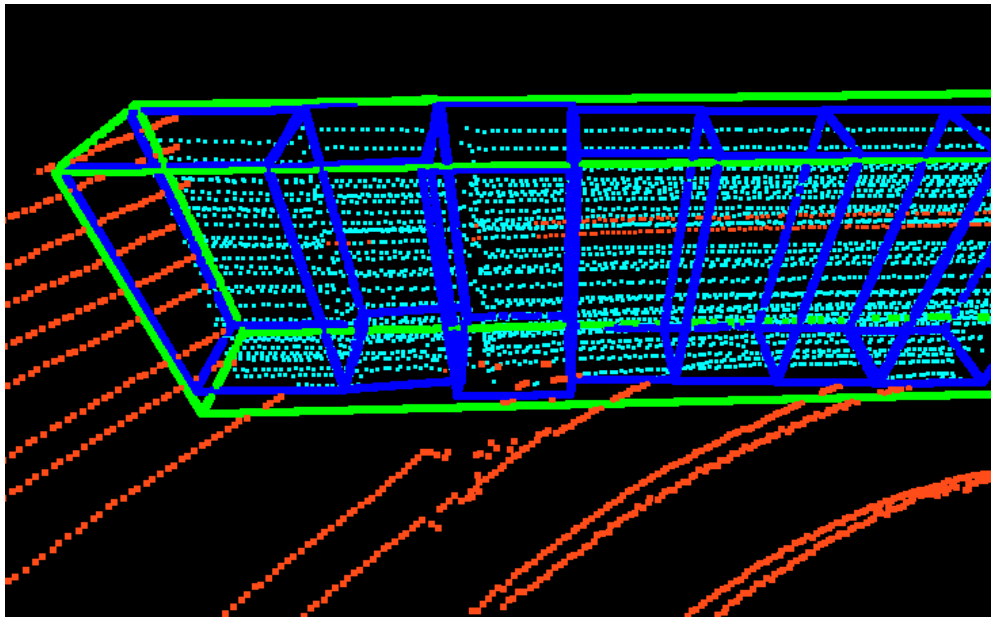


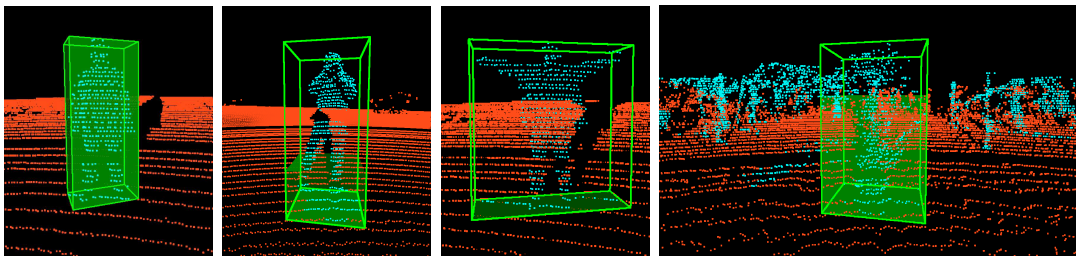
Figure 5.11: Rejected pedestrian candidates (blue frames) from a wall (green frame). The clustering algorithm re-merged the smaller boxes. The large object is also rejected because it is now again too large.

exhibit fast runtimes but the original algorithm requires knowledge of the number of clusters k in advance and an imprecise initial configuration of medians may drastically increase runtime. The k-means++ extension [AV07] improves the initial distribution and provides faster runtimes. The problem of an unknown k is solved by the dp-means algorithm [KJ12], which starts with $k = 1$ and increments it (if a cluster grows too large), and in addition exhibits fast runtimes. An example of a resulting separated cluster is shown in Figure 5.10 where the replaced cluster is framed in blue and the new clusters are framed in green.

The aforementioned step results in an over-clustering of large obstacles, which are now separated w.r.t. the extent of pedestrians. Hence, the next step is to re-merge nearby clusters without a sufficient gap between them. An example of re-merged candidate clusters is shown in Figure 5.11. Here, the new smaller clusters cannot be separated adequately and the original group is maintained. This strategy has the advantage of finding pedestrian candidates close to any other obstacles at the expense of an increased runtime and the possibility of more false positives.

5.2.3.3 Features

Pedestrian feature vector per cluster consists of 8 different features introduced in the literature, where f_1 and f_2 are presented by Premebida et al. [PLN09] and to describe the number of points included in a cluster and the minimum distance



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

Figure 5.12: Selected detection results with confidence displayed as green filling. A standing pedestrian with a confidence of $> 99\,\%$ is shown in (a) and a running pedestrian with a confidence of $\approx 35\,\%$ in (b). The pedestrian with a confidence of $< 1\,\%$ in (c) outstretches his arms while wearing an open jacket and is a good example of the limitations of the detector. In (d), a tree in a forest region falsely classified as pedestrian with a confidence of $\approx 65\,\%$.

of the cluster to the sensor. Navarro-Serment et al. [NSMH10] apply a Principal component analysis (PCA) to the clusters, which represents f_3 to f_7. Those five features are: the 3D covariance matrix of a cluster, the normalized moment of inertia tensor, the 2D covariance matrix in different zones (cf. [NSMH10]), the normalized 2D histogram for the main plane, and the normalized 2D histogram for the secondary plane. In another approach, Kidono et al. [KMW+11] introduce two additional features. The first one, the slice feature of a cluster, forms the last feature f_8 and aims to differentiate pedestrians from false positives in the shape of trees or poles. A cluster is partitioned into slices along the z-axis and for each slice the first and second largest eigenvalue is calculated. As the descriptive power of the slice feature decreases over longer distances, only a rough estimate remains. The other feature introduced by Kidono et al. considers the distribution of the reflection intensities in the cluster. Since the Velodyne HDL-64E used for this thesis is not calibrated w.r.t. the intensities, this feature could not be integrated.

5.2.3.4 Classification & Training

The task of the classifier is to perform a binary classification between pedestrian and non-pedestrian as precisely as possible for each candidate cluster. As proposed by Kidono et al. [KMW+11], a SVM is used with radial basis function (RBF) Kernel [CL11, CV95] together with the feature vector described previously. For training the SVM, pedestrians are annotated in different data sets from the campus Koblenz of the University of Koblenz-Landau. Positive samples are extracted from annotations and negative samples are generated at random from clusters that have not been marked as pedestrians by a human annotator. The LIBSVM library from Chang and Lin [CL11] is used to compute a separating hyperplane which discriminates between pedestrians and non-pedestrians. The

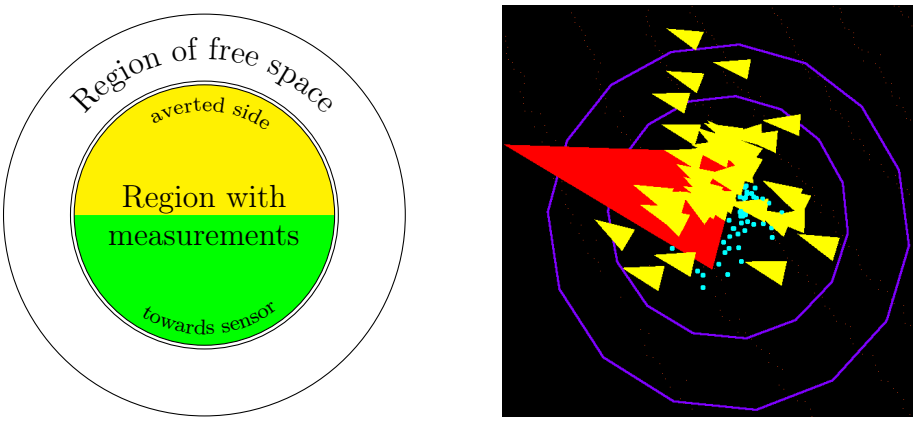


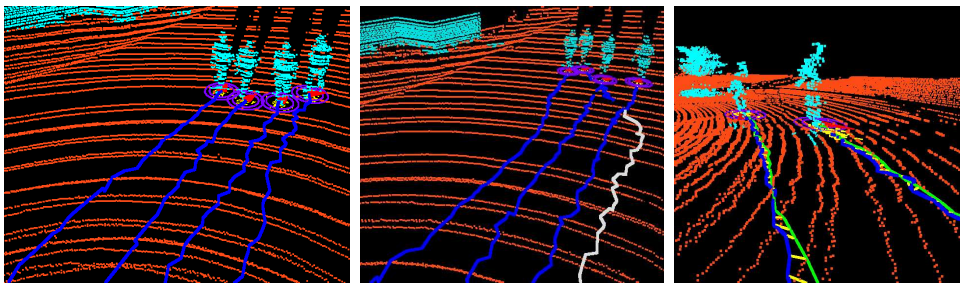
Figure 5.13: Cylindrical measurement model of the particle filter. The left image outlines the different regions of the model. The right image shows the model applied to real sensor data with displayed particle hypothesis (cyan: range readings; yellow: particle; red: best particle).

classifier returns a vector of confidence values in $[0, 1]$ for each class representing how likely it is that a candidate belongs to the class (cf. Figure 5.12). The probability of the highest rated class is used as the input value for the tracking system (cf. Section 5.2.3.6) and can further be used to reject candidates with low probability.

5.2.3.5 Measurement Model

The initial idea for the tracking is to re-use virtual 2D scans again inspired by Petrovskaya and Thrun [PT09]. Many pedestrians could successfully be tracked with an adopted measurement model and a high resolution 2D scan. Unlike in urban scenarios, varying inclination angles of the terrain and many occlusions (e.g., caused by vegetation) occur and are problematic. Hence, a new measurement model was developed focussed on a sophisticated interaction of the detection algorithm with the particle filter. This allows for discarding the hypotheses early and enables to deal efficiently with false detection that occur frequently in unstructured environments.

The measurement model approximates pedestrian geometry as a cylindrical shape of non-zero depth (cf. Figure 5.13). The likelihood of range readings is modeled according to three different regions omitting the height value. A region of free space in form of an outer cylinder is modeled around an inner cylinder with one half facing the sensor and the other half facing away. The majority of range readings are expected to fall in the region facing the sensor (green cylinder-half). The minority is expected on the other side (yellow cylinder half) as humans represent solid objects and laser rays passing a human occur infrequently, e.g., during limb movement. The region around the pedestrian is expected to contain few



(a) (b) (c)

Figure 5.14: Tracking results in selected situations. Image (a) shows four pedestrians walking in the same direction with blue lines on the ground visualizing active tracks. In (b), one of the pedestrians changed his direction and crossed paths with the others. Since he became entirely occluded once, he had to be re-detected and the previous track was stored and displayed in white. The third image (c) shows a comparison of tracked persons (blue tracks) with given ground truth (green tracks). Correspondences with tracking results and ground truth are shown as yellow lines.

to none points. Hence, the measurement model separates neighboring obstacles adequately while taking pedestrians very close to other objects into account, too.

5.2.3.6 Particle Filter & Particle Extinction

Tracking is performed using a Rao-Blackwellized particle filter [DdMR00], again with 40 particles for each target where target extent is estimated separately for each positional hypothesis. Each pedestrian hypothesis consists of a 2D position, an orientation, a rotation angle w.r.t. the sensor, a velocity, and a circular extent. Velocity compensates for pedestrian movement by applying a model of constant velocity due to the small possible change within one rotation of the LRF. An example of four successfully tracked pedestrians is shown in Figure 5.14. The image shows how pedestrians are tracked over time (blue lines) and that one pedestrian is lost once due to occlusions (white line). In case of a false detection, e.g.,, caused by a tree or a bush, a particle filter would be initiated and remain on the target until it disappears from view. Since false detections occur inevitably in the target domain, a solution was sought to handle them. Confidence-based approaches (in images [BRL$^+$09, SGVG10]) additionally grade system estimations in order to reinforce correct inferences. In this approach, a particle filter is discarded if either no re-detection occurs for a predefined time or if re-detections occur but the confidence of the SVM is insufficiently low to maintain the target. The first criteria allows continuous tracking in case of occlusions for a short period of time and the second criteria ensures that no particle filter remains on invalid targets for a longer period of time. In other words, the idea is that many low-confidence detections or few high-confidence detections are both able to maintain a target, even if the tracker yields inaccurate results.

Data set	TP [No.]	FP [No.]	FN [No.]	Precision [%]	Recall [%]
Gravel terrain (12 Hz.)	2358	37	221	98.46	91.43
Outdoor area (12 Hz.)	3149	1434	1633	68.71	65.85
Polyterrasse (5 Hz.)	1849	846	886	68.61	67.61
Tannenstrasse (5 Hz.)	1348	2501	1116	35.02	54.71

Table 5.5: Pedestrian detection results in 3D LRF data in different environments.

5.2.3.7 Evaluation

Evaluation was performed on a laptop with an Intel(R) Core(TM) i7 QM with 1.73 GHz and 8 GB RAM. Training of the SVM is performed on data sets gathered in vegetated areas on the university campus in Koblenz and the training data sets are completely disjoint from all evaluation data sets. During the experiments, all parameters are fixed and pedestrians were annotated within a range of up to 20 m. The first two data sets have been recorded in Koblenz. *Gravel terrain* represents a gravel parking lot on rough terrain with a number of puddles, some bushes and an overgrown lamp post. The *Outdoor area* data set has been recorded on a road turn next to vegetated hillside with two buildings and a tree. Two Velodyne HDL-64E data sets with pedestrian ground truth are published by Spinello et al. [SATS10, SLA11]. The first data set *Polyterrasse* contains pedestrians and bicycles in an urban area of Zurich, Switzerland. *Tannenstrasse*, the second data set, has been recorded in the downtown area and contains additional traffic participants such as trams and cars. Tables 5.5 and 5.6 show the detection and the tracking performance, respectively, on the data sets. The true positives (TP) denote the number of correctly detected/tracked pedestrians, the false positives (FP) denote the number of incorrectly detected/tracked objects (false alarms), and the false negatives (FN) denote the missed targets. Precision and Recall are calculated as before, cf. Equation 4.13 respectively Equation 4.14. Runtimes of the algorithm are summarized in Table 5.7.

The presented approach yields interesting as well as unexpected results. Firstly and as expected, the algorithm performs well on the Gravel terrain data set which is less crowded than the other data sets. On the Outdoor area data set, performance is affected by the additional candidates which reduce the precision. The decrease of the Velodyne HDL-64E frequency from 12 to 5 Hz has an unexpected but substantial influence on the SVM classification as the point density of a pedestrian drastically changes. Once detected, the particle filter tracks the pedestrians with 5 Hz, but the missing detections affect the tracking especially on the Tannenstrasse data set. Recall rates in crowded data sets are expected to be low due to the additional candidates required to separate pedestrians standing close to vegetation. In order to increase the recall values, one would have to exclude difficult

candidates (which is antithetical to the whole idea of this approach) or exchange the classifier itself for another method. Considering the runtime results, the algorithms show an overall good performance. The bottleneck of this approach is the segmentation step, which is affected by the additional candidates created by the segmentation. These results are published in [Häselich et al., 2014].

Data set	TP [No.]	FP [No.]	FN [No.]	Precision [%]	Recall [%]
Gravel terrain (12 Hz.)	2545	165	34	93.91	98.68
Outdoor area (12 Hz.)	4556	6448	226	41.40	95.27
Polyterrasse (5 Hz.)	2476	1454	259	63.00	90.53
Tannenstrasse (5 Hz.)	1802	7218	662	19.98	73.13

Table 5.6: Pedestrian tracking results in 3D LRF data in different environments. Noticeable are in particular the high recall values in comparison with the average detection recall values exhibited in Table 5.5.

Data set		Mean [ms]	Std. dev. [ms]	Min. runtime [ms]	Max. runtime [ms]
Gravel terrain 0 min 45 s 14 Hz	Segmentation	63.23	5.21	53.25	144.78
	Detection	9.76	4.74	0.00	25.43
	Tracking	1.48	0.96	0.00	4.31
	Overall	74.88	7.54	55.63	159.96
Outdoor area 12 min 46 s 14 Hz	Segmentation	141.03	57.08	96.89	349.27
	Detection	31.70	11.16	3.97	73.47
	Tracking	32.89	26.49	0.00	159.34
	Overall	206.56	64.66	115.84	582.29
Polyterrasse 2 min 59 s 5 Hz	Segmentation	63.86	5.05	48.70	87.85
	Detection	16.25	8.32	12.90	43.89
	Tracking	6.60	4.69	0.00	43.26
	Overall	87.53	12.32	57.54	137.17
Tannenstrasse 1 min 39 sec 5 Hz	Segmentation	551.63	199.05	76.65	2208.51
	Detection	44.47	14.15	7.68	100.10
	Tracking	96.53	74.86	0.00	591.14
	Overall	693.31	214.32	108.20	2390.22

Table 5.7: System runtimes of the pedestrian detection and tracking approach in 3D LRF data in different environments.

Chapter 6

Integration of Dynamic Obstacles into Terrain Classification

Figure 6.1: Where is the road? It runs between the two red dots. Although no road surface is directly visible, it can be inferred from the knowledge that cars drive on roads.

A complete and optimized terrain classification approach on fused sensor data is presented in Chapter 4 and the detection and tracking of vehicles and pedestrians is depicted in Chapter 5. So far, these algorithms work separately from each other and fulfill specific disjoint tasks on the robot. Nevertheless, the output from one of these algorithms can be very valuable for the other. Consider Figure 6.1, which shows an image recorded with the robot's front camera. If someone would ask a human being where the road runs along in this image, the answer would probably be something like: it runs between the two end-points indicated by the red dots in the image. But how can humans detect the road, the terrain surface in general, without actually seeing it? The answer is simple: Once humans detect the cars in this scenario, they use their knowledge that cars drive on roads and draw

inferences. This kind of inference is possible by knowing the semantics of cars or other dynamic obstacles and using it in another domain. The knowledge that cars drive on roads works exactly the same way: if a road surface is visible and detected as such, tracking of a car can be refined as it is more likely that a vehicle will follow the road rather than leaving it. This applies to pedestrians in an inferior way, too, with the knowledge that they tend to prefer flat terrain surfaces over rough terrain surfaces. For this thesis, only the first kind of semantic knowledge is relevant as the goal is to improve terrain classification and not to create a precise tracking approach. Therefore, in the following chapter, a system is described which integrates vehicles and pedestrians in the terrain classification approach in order to improve the terrain surface classification with the MRF.

6.1 Trails and Extrapolation

The task is now to integrate the information from the tracking system introduced in Chapter 5 into the terrain classification process. Therefore, *trails* are introduced. A trail belongs to a single target and contains all relevant available tracking results and information. It contains the target's pose, velocity, and dimensions. For a pedestrian, the trail T consists of

$$T_{\mathrm{P}} = \langle (\boldsymbol{x}, v, d)_0, (\boldsymbol{x}, v, d)_1, \ldots, (\boldsymbol{x}, v, d)_t \rangle \quad , \tag{6.1}$$

where $\boldsymbol{x}$ is the x, y-position and θ-orientation, v the movement speed, and d the diameter of the cylinder enclosing the pedestrian. Within the sequence, all entries are ordered by the time of their occurrence beginning with the oldest. Height is omitted as pedestrians are assumed to stay permanently in contact with the terrain surface. Correspondingly, the trail for a vehicle is given by

$$T_{\mathrm{V}} = \langle (\boldsymbol{x}, v, w, l)_0, (\boldsymbol{x}, v, w, l)_1, \ldots, (\boldsymbol{x}, v, w, l)_t \rangle \quad , \tag{6.2}$$

where $\boldsymbol{x}$ and v are identical with Equation 6.1 and width w is the diameter perpendicular to the orientation and length l the diameter parallel to the orientation for the rectangular vehicle model. Note that all regions of free space, anchor points, or other auxiliary variables are removed at this point. All trails of all currently tracked targets are assigned to them and are provided by the tracking system. If the tracker loses the target, the trail is marked accordingly and unbound from the target as the tracker will be discarded. Trails of lost targets are not discarded likewise as they still contain all the information where other traffic participants moved in the past. Therefore, the tracking module keeps track of all lost trails and provides them together with all currently tracked objects.

Note that a single target may create several trails as tracking may lose a target, e.g., due to occlusions, multiple times or a target may leave and re-enter the range of the sensor several times. An example for pedestrians is shown in

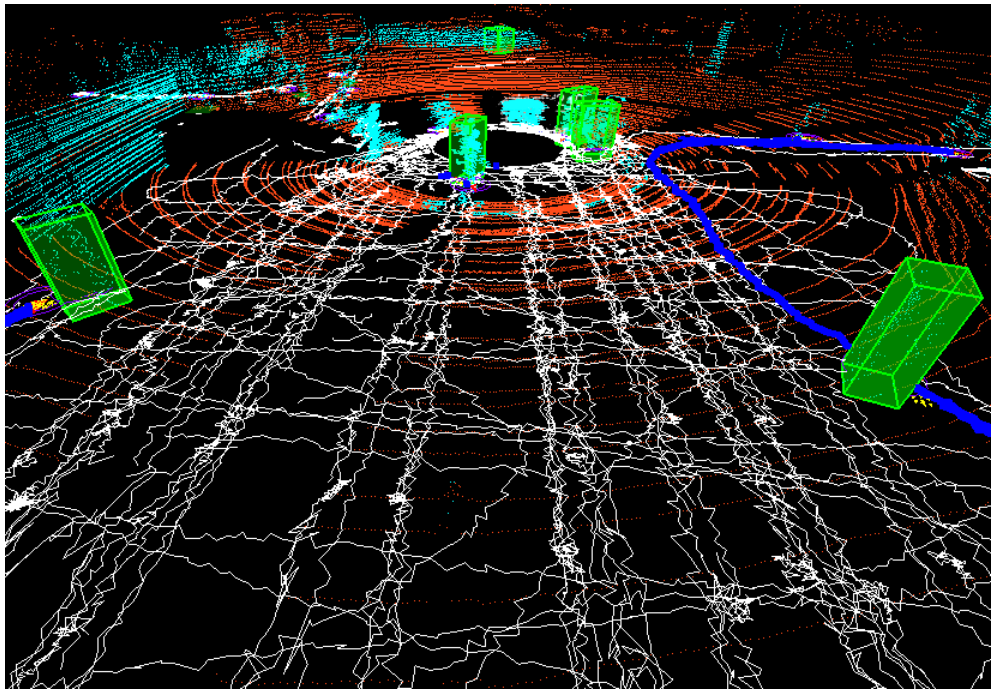


Figure 6.2: Tracking results from pedestrian tracking (cf. Section 5.2.3) with visualized trails. Trails are gathered pose and geometry information of successfully tracked targets. Detections are highlighted as green bounding-boxes, active (alive) trails are drawn in blue, and lost (dead) trails are shown in white.

Figure 6.2. Green boxes in the image are current pedestrian detections by the approach described in Section 5.2.3.4. Trails are visualized as white lines for lost trails and blue lines for active trails yet linked to a target being tracked. In this example, lost tracks accumulate over a long period of time resulting in many trails that encode the information where pedestrians moved in the past.

Considering vehicle tracking, a prediction can be made where a vehicle is going to move and where it will probably be located in the future. This information is not only interesting for collision avoidance (anticipatory driving) but is also relevant for terrain classification as the information where vehicles are going to drive allows inferences about the targeted terrain. Prediction is performed for all tracked vehicles as T' according to

$$T'_{\mathrm{V}} = \langle (\boldsymbol{x}, v, w, l, u)_{t+1}, (\boldsymbol{x}, v, w, l, u)_{t+2}, \ldots, (\boldsymbol{x}, v, w, l, u)_{t+\mathrm{MaxPrediction}} \rangle \quad , \quad (6.3)$$

where T'_{V} is consistent with T_{V} with exception for u, which is a measure for uncertainty in range $[0, \ldots, 1]$. Values for T'_{V} are computed as follows

$$\begin{aligned} x_{t+1} &= \Delta t \cdot v_t \cdot \cos(\theta_t) \\ y_{t+1} &= \Delta t \cdot v_t \cdot \sin(\theta_t) \\ \theta_{t+1} &= \Delta t \cdot \Delta\theta \\ u_{t+1} &= \max(1 - (\tfrac{\Delta_{0,t+1}}{\mathrm{MaxPrediction}}), 0) \end{aligned} \quad . \quad (6.4)$$

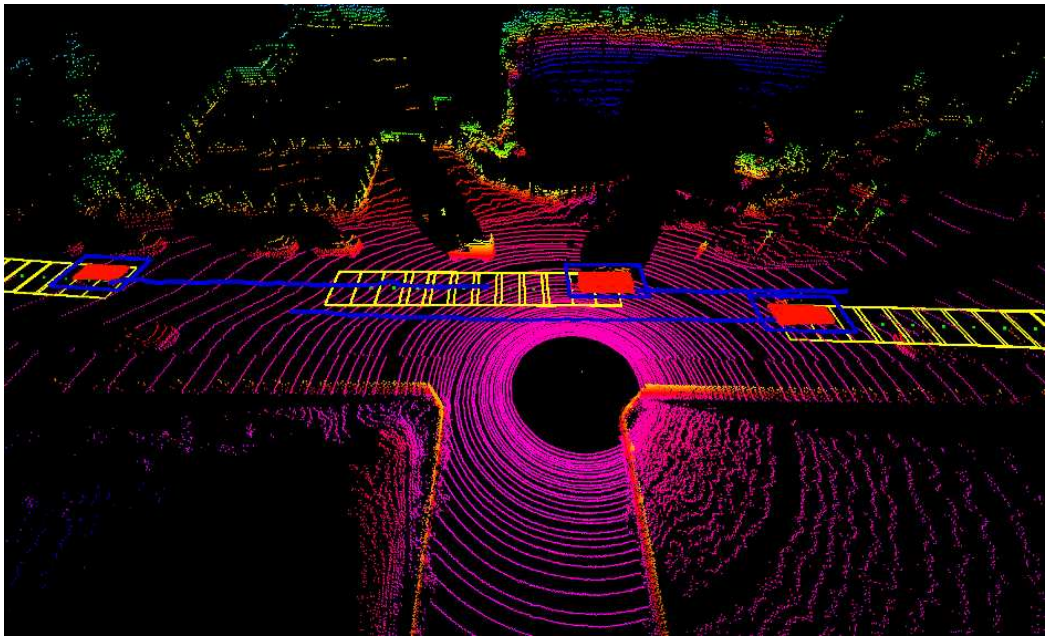


Figure 6.3: Tracking results from vehicle tracking (cf. Section 5.1) with visualized trails. Trails are gathered pose and geometry information of successfully tracked targets. Tracked vehicles are shown as red rectangles, vehicle trails as blue lines, and predictions as yellow rectangles for each extrapolation.

This linear extrapolation depends on the previous state as well as on Δt, which is the amount of time in milliseconds between two predictions, or between the first prediction and the current state, respectively. The time of the prediction from the current state to the current prediction is denoted by $\Delta_{0,t+1}$. For the fifth extrapolation step and a LRF update frequency of 12 Hz for example, $\Delta_{0,t+1}$ is $(4+1) \cdot \frac{1}{12}\,\mathrm{s} = 417\,\mathrm{ms}$. Values for v, w, l are not altered during prediction. The threshold MaxPrediction has two functions. On the one hand, it defines the temporal limit in which prediction makes sense and, on the other hand, it is used to scale the predictions by starting from 1 and linearly decreasing to 0. As MaxPrediction is set to 3 seconds, prediction for a car, for example, driving 50 km/h covers a trajectory of 41.6 m. The tracker performs a prediction for every vehicle. In case the target is lost, only T_V survives while T'_V is discarded together with the tracker. An example for tracked vehicles is shown in Figure 6.3. Here, vehicle trails T_V are shown as blue lines and predictions T'_V in form of extrapolations are visualized as yellow bounding boxes. The three cars passing a crossing are marked by red rectangles. Note that the detection and tracking approach for vehicles only considers moving objects which means that, for example, parked cars are considered stationary obstacles and do not contribute to the integration. Prediction is not performed for pedestrians as their cylindrical model is independent from their motion direction.

6.2 Integration into Markov Random Field Terrain Classification

With the introduction of trails, position and geometry information of pedestrians and vehicles is now available for terrain classification in form of T_P, T_V, and T'_V. In other words, for each site within the terrain grid, inspection is possible if other traffic participants use this cell for their navigation. The next step is to perform inference with this additional knowledge; more specifically, the task is to improve classification quality utilizing the new information. Therefore, trails need to be integrated into the terrain classification process. Reviewing Equation 4.5 reveals that the two energy terms E_f and $\mathrm{E}_\mathcal{N}$ are separable and fulfill disjoint tasks to create a combined energy value.

The term E_f encodes the feature component which describes how likely a terrain cell belongs to a terrain class. This is done by comparing observed features against normal distributions of features learned manually in advance. Hence, this part is a feature-based cell-wise classification of present observations.

The term $\mathrm{E}_\mathcal{N}$ encodes the neighborhood component which describes how well a terrain cell fits into its neighborhood. This is done by comparing the current label of the terrain cell against the labels of the cell's neighborhood. In contrast to the feature component, the neighborhood component is completely independent of all present sensor observations. It rather models the assumption that sites tend to group or, more precisely, it encodes the prior that sites neighboring each other are likely to have to the same label (often referred to *smoothness prior*). This a-priori neighborhood assumption is realized via the Potts model which provides a solution for the neighborhood interaction.

The information where other traffic participants move is the result of another algorithm from present and past observations. But those tracking results alone are useless for terrain classification. Their usefulness for the MRF terrain classification approach arises from the a-priori knowledge that vehicles drive on roads and pedestrians walk on sidewalks. Therefore, T_P, T_V, and T'_V are not considered additional features but rather knowledge that needs to be encoded in the prior. Hence, the Potts model needs to be revisited again. For two arbitrary labels $l_1, l_2 \in \mathcal{L}$ the Potts model from Equation 2.37 is defined as

$$\delta(l_1, l_2) = \begin{cases} -1 & \text{if } l_1 = l_2 \\ +1 & \text{else} \end{cases} , \tag{6.5}$$

which is the *Kronecker delta* (also known as *Kronecker's delta*) that is named after Leopold Kronecker (⋆1823, †1891), a German mathematician. In order to

integrate the additional knowledge into the model, the Kronecker delta for the terrain classification approach is extended to

$$\delta'(l_1, l_2, (i,j)_{l_1}) = \begin{cases} -1 & \text{if } l_1 = l_2 \wedge l_1 = \textit{Obstacle} \\ -1 - \mathrm{I}(T_\mathrm{P}, T_\mathrm{V}, T'_\mathrm{V}, (i,j)_{l_1}) & \text{if } l_1 = l_2 \wedge l_1 = \textit{Street} \\ -1 + \mathrm{I}(T_\mathrm{P}, T_\mathrm{V}, T'_\mathrm{V}, (i,j)_{l_1}) & \text{if } l_1 = l_2 \wedge l_1 = \textit{Rough} \\ +1 & \text{else} \end{cases}, \tag{6.6}$$

where $(i,j)_{l_1}$ is the site that the label l_1 is assigned to (cf. Equation 2.1) and the function $\mathrm{I}(\cdot)$ enforces an additional prior for the two classes *Street* and *Rough*. The prior integrates trail and prediction information from the tracking approaches as

$$\begin{aligned} \mathrm{I}(T_\mathrm{P}, T_\mathrm{V}, T'_\mathrm{V}, (i,j)_{l_1}) \;=\; & \sum_{m}^{T_\mathrm{P}} \mathrm{i}(m_{\text{geometry}}, (i,j)_{l_1}) \\ + \; & \sum_{n}^{T_\mathrm{V}} \mathrm{i}(n_{\text{geometry}}, (i,j)_{l_1}) \\ + \; & \sum_{o}^{T'_\mathrm{V}} \mathrm{i}(o_{\text{geometry}}, (i,j)_{l_1}) \cdot (1 - o_u) \end{aligned}, \tag{6.7}$$

where $m_{\text{geometry}}, n_{\text{geometry}}, o_{\text{geometry}}$ represent the geometry of the vehicles, or the pedestrians in the coordinate system of the terrain classification w.r.t. their pose and model, respectively. The function $\mathrm{i}(\cdot)$ returns 1 if there is an intersection of the geometry with the site the that label l_1 is assigned to, and otherwise 0. In case of multiple intersections from different points in time within a trail, the function returns only 1 and ignores the number of total intersections. Furthermore, to account for the uncertainty in prediction, a site affected by the prediction for a vehicle has its intersection value multiplied with the uncertainty value u from the tracking system. In case of multiple intersections from different points in time within a prediction, the function returns the lowest uncertainty value among the intersecting elements of T'_V. Note that obstacles are not changed since $l_1 = l_2$ from Equation 6.5 and $l_1 = l_2 \wedge l_1 = \textit{obstacle}$ from Equation 6.6 both return the same value.

Another design choice concerns the case when a terrain cell is affected by multiple different dynamic obstacles. For example, a pedestrian walked over a terrain cell which a car also used before and there is a prediction that a second car is likely to drive towards that cell. In this case, the function $\mathrm{I}(\cdot)$ could return the highest value from the respective candidates, which would yield a value 1 in this example. For my approach, I decided to accumulate the results from multiple targets which yields a value of 1+1+0.7 for the example. Thereby, a strong prior is enforced on a terrain cell if multiple dynamic obstacles choose it for navigation.

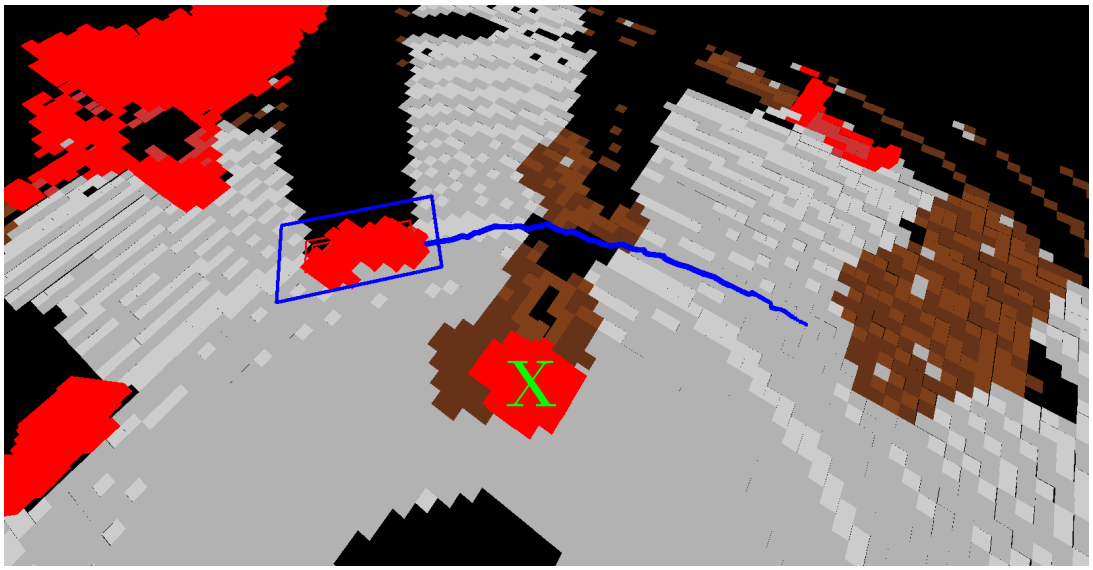
Within the context of terrain classification, the Kronecker delta can be thought of as a tensor and the differences of the models are as follows. The original Ising model is a $2 \times 2 \times 1$ tensor and, in fact, a degenerate case of the Potts model

which is a $M \times M \times 1$ tensor where M is the number of labels. The extension δ' of the Kronecker delta results in a $M \times M \times (o+1)$ tensor where o is the number of additional cases. For the MRF terrain classification approach, the Kronecker delta is a $3 \times 3 \times 3$ tensor as there are three labels *Obstacle*, *Street*, and *Rough* and two additional cases for the labels *Street* and *Rough* as those are possibly affected by dynamic obstacles. Note that the rejection label *Unknown* is omitted and that the label *Obstacle* is unchanged and still part of the default layer of $\delta(\cdot)$. The next section depicts a comparison between the original $\delta(\cdot)$ and $\delta'(\cdot)$ on selected examples where a change in classification is possible.

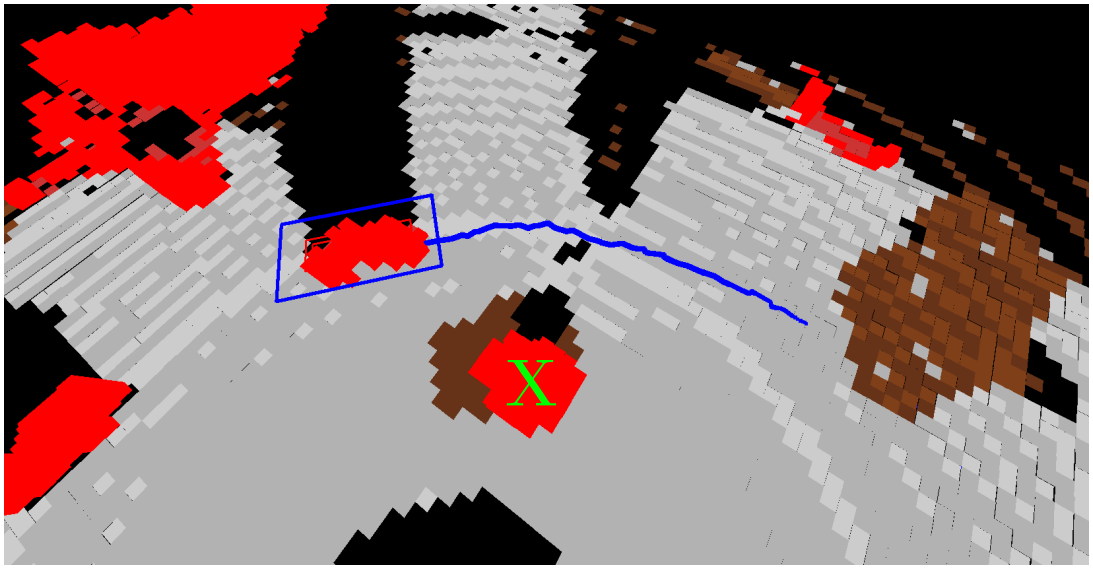
6.3 Experiments

Experiments are performed on a gravel area that possess a semi-rough surface. Semi-rough means that the roughness feature does not yield distinct results as the surface is generally flat but for example, the tires of vehicles leave wheel rots and the surface condition therefore varies. Regions with low data density occur separately and in combination with ambiguous surface conditions due to smaller obstacles in the test scenario.

Evaluation is not possible for this approach as no ground truth exists for this specific case. Generation of ground truth by human experts is also not possible here as the effect from the new model introduced in Section 6.2 only occurs in very specific regions and if dynamic obstacles actually use this region. For example, in a region with ambiguous features and low data density, a human expert cannot possibly annotate labels, or at least not without errors, because there are insufficient data he can rely on. Therefore, he could only guess or maybe already know the place, in which case the new information would be useless to the expert as it will not contain anything new. The question here is how can one rate the change in classification between the MRF terrain classification with and without the integration of dynamic obstacle trails and predictions. In order to assess the original terrain classification and the terrain classification using the new model from Equation 6.6, a visual comparison is provided in Figure 6.4 and 6.5. Figure 6.4 shows a vehicle trail and the difference in terrain classification. In Figure 6.4 (a), the original terrain classification is used and in Figure 6.4 (b), the new model is applied. The scenario is well-suited for the comparison as the bush marked with a green cross yield sparse roughness data in the region it occludes. With the new model, the region behind the bush is classified differently according to the information that a vehicle already drove on the respective terrain cells. The same observation can be made in Figure 6.5 where multiple pedestrians walk within the vicinity of the 3D LRF. Here, a difference in classification occurs at three small regions where the knowledge that pedestrians walked over the corresponding terrain cells is integrated into the terrain classification approach.

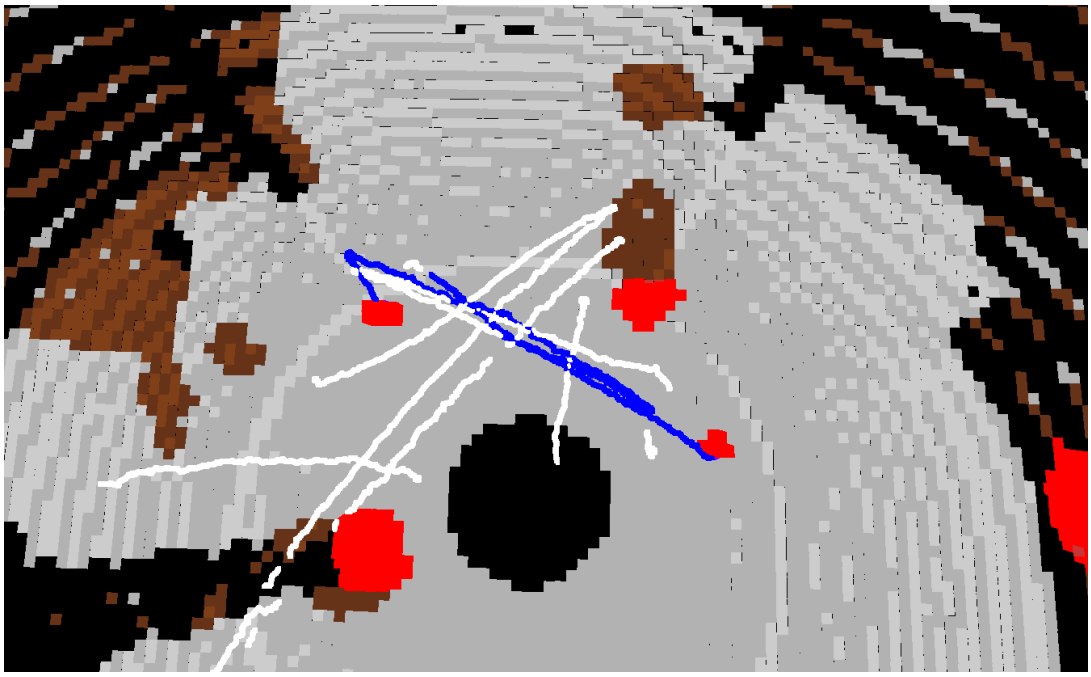


(a)

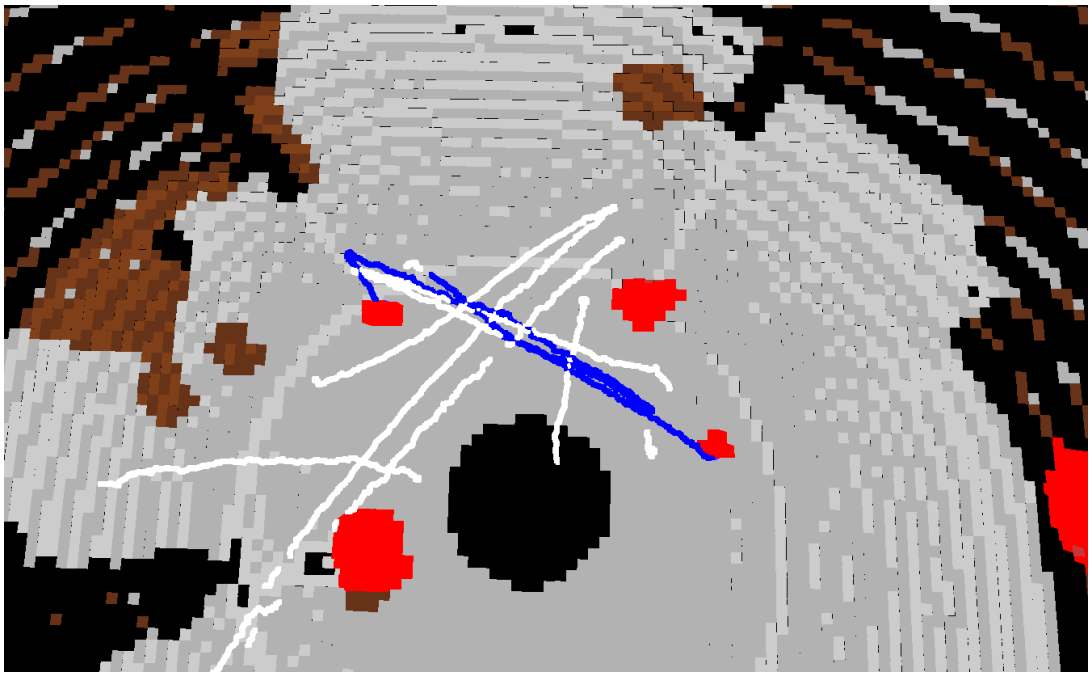


(b)

Figure 6.4: Comparison of activated and deactivated integration of moving vehicles into terrain classification. The upper image (a) shows the result of the MRF terrain classification approach in gravel terrain and the lower image (b) integrates the knowledge from dynamic obstacles. Bounding box and trail of the vehicle are depicted in blue. The scene is well-suited to demonstrate the result of the integration as the affected terrain cells contain only sparse data due to the bush highlighted with a green cross.



(a)



(b)

Figure 6.5: Comparison of activated and deactivated integration of moving pedestrians into terrain classification. The upper image (a) shows the result of the MRF terrain classification approach in gravel terrain and the lower image (b) integrates the knowledge from dynamic obstacles. Active trails of pedestrians currently tracked are depicted in blue and tracks of lost targets are shown in white. The image highlights the integration effect at three small regions where the labels changed according to the new information.

Chapter 7

Conclusion

The last chapter of this thesis reviews and summarizes my research and my results. It is split up into several subsections which state the problem, the methodology, describe the experiments, outline the results and discuss them, and conclude with recommendations for further research in this field.

Problem statement

In this thesis I address the problem of terrain classification in unstructured terrain. The environment is perceived by various sensors and autonomous navigation requires a correct interpretation of these data. Sensors include 3D LRFs, cameras, IMUs, and GPS receivers. As the natural scenario is complex, fault-prone, and requires fast processing of sensor data, a robust and fast solution is necessary. Furthermore, dynamic obstacles also need to be taken into account, in order to avoid collisions with vehicles or pedestrians.

Methodology

MRFs have been used frequently in image analysis and other fields and are characterized by their context-sensitivity. The MRF terrain classification approach presented in this thesis requires a 3D LRF and optionally up to three cameras. Further sensors like the IMU or a second 3D LRF also contribute to the result but are not necessary. In a first step, a terrain grid is applied for data reduction. For all grid cells, features are computed and classification with additional respect to the MRF neighborhood component is conducted. The resulting grid contains obstacles and free regions, of which the latter are further categorized according to their surface conditions. Other traffic participants in natural environments involve vehicles and pedestrians. Vehicles are detected and tracked in the data of the 3D LRF. Pedestrians are detected in camera images and are detected and tracked in the 3D LRF data. Tracking for all candidates is performed with a particle filter with different measurement models for the respective targets. Eventually, the results from the algorithms are combined to enhance and improve terrain classification by integrating the dynamic obstacles and thus using the knowledge that vehicles drive on streets and pedestrians prefer flat terrain.

Experiments
Evaluation was performed for all novel algorithms presented in this thesis. Starting with terrain classification, the MRF approach was initially evaluated against single scans annotated by hand. In the domain of terrain classification of 3D distance measurements, no ground truth was publicly available and hence had to be generated by human experts. The algorithm yielded good results in different test scenarios except for the label *Rough*, which was the least present label in specific scenarios and exhibited a reduced true positive rate there. Runtime of the terrain classification was evaluated in long-term test runs and the presented optimizations revealed adequate accelerations, e.g., factor 10 for the GPGPU parallelization. The MRF terrain classification approach was further evaluated on 3D maps of different challenging environments. Since the maps consist of millions of 3D points with annotated ground truth from human experts, experiments were extensive and further focused on variations. These variations differ from map to map and aim to highlight the strengths and limitations of the approach. Overall, the algorithm performed well and exhibited different new aspects and also some weaknesses in this extensive evaluation. Noticeable is especially the classification quality of the class *Obstacle*, which remained very high in all test cases. Evaluation of the approaches to detect and track dynamic obstacles could be performed on different data sets available to the public. The approach to vehicle detection and tracking in 3D LRF data was evaluated on data sets from the DARPA Urban Challenge. In all test cases, the algorithm performed well and exhibited excellent runtimes. The approach to color image-based pedestrian detection performed well on publicly available data sets, but exhibited only mediocre runtimes, even if several acceleration techniques were applied. Laser-based pedestrian detection and tracking was likewise evaluated on data sets available to the public. The approach performed well during the experiments except for runtime and precision in very crowded environments. Feasibility of an improved classification by integrating dynamic obstacle trails and predictions into the MRF terrain classification was proven in several comparative experiments in selected regions.

Discussion
The presented approaches to terrain classification and detection and tracking of dynamic obstacles in unstructured environments yield good to very good results in all scenarios and exhibit fast runtimes. Generally, the feasibility of successfully classifying obstacles and terrain with an MRF is proven. Limitations reveal while using fused data of color cameras and 3D LRF. Here, a straightforward approach does not yield adequate increase of classification quality to justify the increased runtime of the algorithms. For improvement, the quality of the camera images could be significantly increased by hardware purchases which in turn requires even faster/better algorithms. Enhancing the cameras for other sensor modalities, e.g., thermal or multispectral imaging, would also increase results of the algorithms. An improved calibration, for example, in form of auto-calibration of the cameras to the 3D LRF, would also contribute to the classification quality. Another ap-

proach could consider software-sided solutions for the existing camera hardware. For example, a color-calibration pattern permanently in the field of view of each camera could be used to deal with illumination and color-related issues. Usage of color-cameras in the outdoors is complex and fault-prone. Considering time of day, weather, or seasonal change for example, the same meadow that was green the last time perceived by the robot's cameras a couple of month ago is now white from snow. Researchers are already searching for solutions, e.g., [CN13] gather *experiences* of all possible phenomena and handle sensor data accordingly. The detection and tracking of vehicles in the 3D data yields very good results but is limited to the first obstacle in a certain direction. Therefore, occluded vehicles are not detected since they are irrelevant for collision detection, but could contribute to other algorithms, for example, the knowledge that there might be another street. Pedestrian detection in camera images is the most advanced research area encountered in this work. Only runtime and detection quality of low resolution images need further improvement in this field. Omnidirectional cameras are another alternative to consider as their field of view overlaps completely with the 3D LRF in order to work on fused data for pedestrian detection. Detection and tracking of pedestrians in 3D LRF data with the approach presented in this thesis yielded good results. Here, runtime issues occur in very crowded environments due to the amount of pedestrian candidates. The reason is my design decision to accept more candidates than other approaches and to drop them early by a close interaction with the tracker. A change of the classifier or a stricter segmentation might reduce runtime in crowded environments while maintaining high detection values.

Recommendations

Despite the success of the developed approaches there is still room for improvements. Besides the main direction I followed for my thesis, from terrain classification via detection and tracking of dynamic obstacles to the integration of both, there are several other directions worth considering. The two most interesting ones in my opinion concern the advance of the terrain classification from 2D to 3D and the development of a multi-sensor auto-calibration. Extension to a full 3D terrain classification from a 2D grid with height information (which in fact is a 2.5D Grid) yields an increased runtime but offers several new opportunities. These opportunities include the ability to drive trough tunnels, handle complex buildings like parking garages, and to better deal with vegetation hanging from above like leaves and branches. An approach to auto-calibration for arbitrary sensor configurations would also bring a lot of benefits to the community and is a challenging task. A possible solution could be to identify distinct objects, respectively their patterns or features, for pairs of sensors. In case such an object is perceived by a pair of sensors, this information could be used re-calibrate their relative position towards each other online w.r.t. the detected position and, where applicable, the orientation of the object.

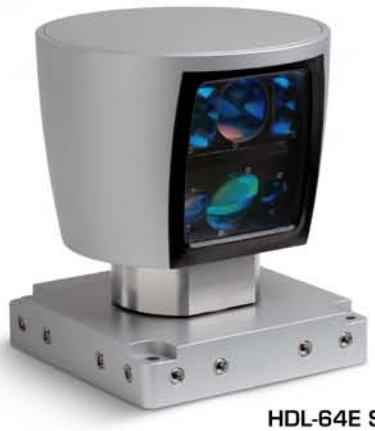
Appendix A

Data Sheet of the Velodyne HDL-64E

Taken from http://velodynelidar.com/lidar/hdldownloads/downloads.aspx.

High Definition Lidar
HDL-64E S2

Velodyne now offers an improved high definition lidar scanner designed for autonomous vehicle navigation, mapping, surveying, industrial automation, and other uses. The S2 version of the HDL-64E provides improved accuracy and a higher data rate than the original version.


HDL-64E S2

High Field of View, High Frame Rate

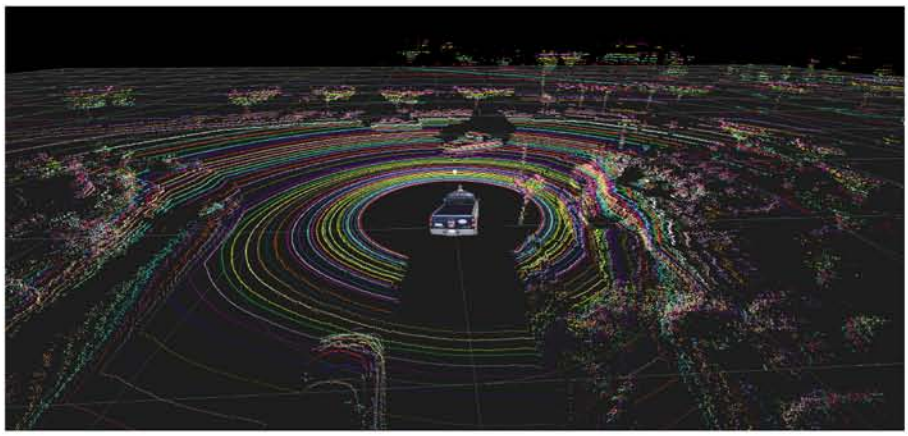
With its full 360° HFOV by 26.8° VFOV, the HDL-64E S2 provides significantly more environmental information than previously available. With its 5-15 Hz user-selectable frame rate and over 1.3 million points per second output rate, the HDL-64E S2 provides all the distance sensing data you'll ever need. The unit's development has been focused on high data rate, high robustness, accuracy and simple 100 MBPS Ethernet interfacing to the end user.

Patent Pending 64-Laser One-Piece Design

Traditional lidar sensors have relied upon a single laser firing into a mechanically actuated mirror, providing only one plane of view. The HDL-64E S2's patented one-piece design uses 64 fixed-mounted lasers to measure the surrounding environment, each mechanically mounted to a specific vertical angle, with the entire unit spinning. This approach dramatically increases reliability, FOV, and point cloud density.

The HD Lidar Concept

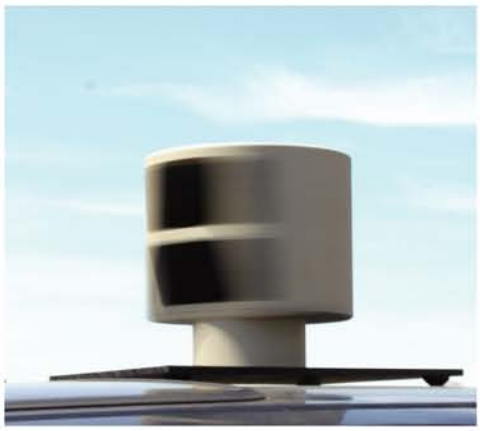
Velodyne's unique HD Lidar technology lets you focus your efforts on control algorithms, image parsing and application-specific processing instead of multi-sensor mounting, debugging and integration. A prototype of the HDL-64E S2 was successfully used in the 2005 DARPA Grand Challenge and the HDL-64E played an essential role in the 2007 DARPA Urban Challenge for multiple prominent teams.



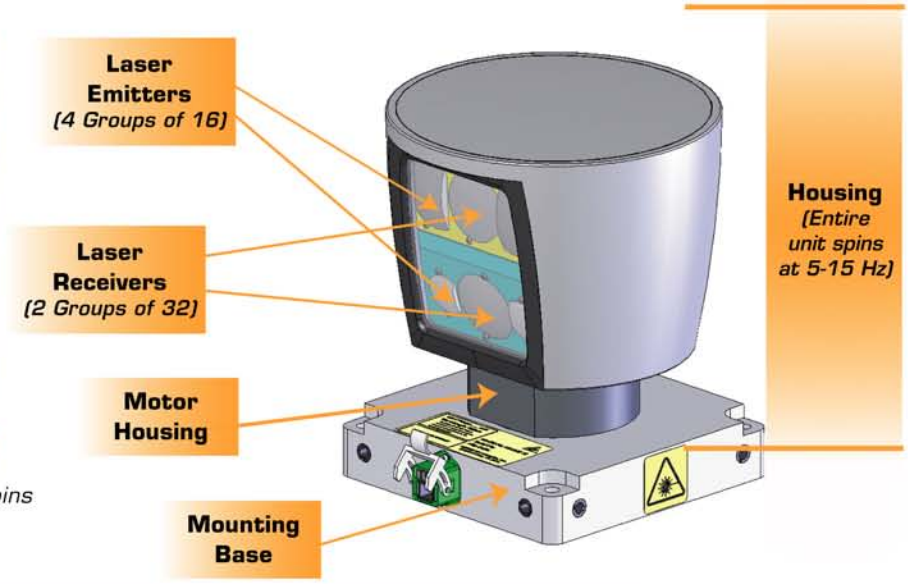
Actual point cloud image from HDL-64E S2 showing vehicle at intersection and other vehicles in vicinity along with road features.

High Definition Lidar

The HDL-64E S2 provides high definition 3 dimensional information about the surrounding environment.



HDL-64E S2 mounted atop vehicle. Unit spins up to 900 RPM (15 Hz) to gather data.



Specifications	
Sensor:	• 64 lasers/detectors • 360 degree field of view (azimuth) • 0.09 degree angular resolution (azimuth) • 26.8 degree vertical field of view (elevation) - +2° up to -24.8° down with 64 equally spaced angular subdivisions (approximately 0.4°) • <2 cm distance accuracy (one sigma) • 5-15 Hz field of view update (user selectable) • 50 meter range for pavement (~0.10 reflectivity) • 120 meter range for cars and foliage (~0.80 reflectivity) • >1.333 M points per second • Operating temperature - 10° to 50° C • Storage temperature - 10° to 80° C
Laser:	• Class 1 - eye safe • 4 x 16 laser block assemblies • 905 nm wavelength • 5 nanosecond pulse • Adaptive power system for minimizing saturations and blinding
Mechanical:	• 15V ± 1.5V @ 4 amps • <29 lbs. • 10" tall cylinder of 8" OD diameter • 300 RPM - 900 RPM spin rate (user selectable) • Environmental Protection IP67
Output:	• 100 MBPS UDP Ethernet packets

Velodyne Lidar, Inc.
345 Digital Drive, Morgan Hill, CA 95037
lidar@velodyne.com

408.465.2800

Appendix B

Data Sheet of the Velodyne HDL-32E

Taken from http://velodynelidar.com/lidar/hdldownloads/downloads.aspx.

High Definition LiDAR™
HDL-32E

Stylishly small, ruggedly built with an unrivaled field of view, Velodyne's HDL-32E LiDAR sensor was designed to exceed the demands of the most challenging real-world autonomous vehicle, mobile mapping, and other industrial applications.

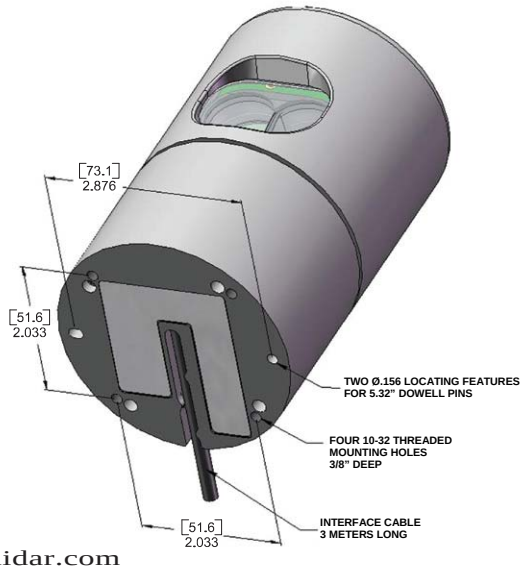
The HDL-32E measures only 5.7" high by 3.4" in diameter and weighs less than two kilograms. Its diminutive size and weight make it ideal for all LiDAR applications, in particular those with constrained form-factors and pricing requirements.

HDL-32E

Unprecedented Field of View and Point Density

The HDL-32E's innovative laser array enables navigation and mapping systems to observe more of their environment than any other LiDAR sensor. The HDL-32E utilizes 32 lasers aligned from +10° to -30° to provide an unmatched vertical field of view, and its patent pending rotating head design delivers a 360 horizontal field-of-view natively. The HDL-32E generates a point cloud of approximately 700,000 points per second with a range of 70 meters and typical accuracy of ±2cm at 10 Hz. The resulting comprehensive point cloud coverage within a single data stream makes the HDL-32E an indispensable part of any sensor suite.

The HDL-32E's operating temperature range spans from -10° C to +60° C and has an IP rating of 67. Its hardened structure makes it perfect for vehicles that operate in the most unforgiving of environments.



High Definition LiDAR

The HDL-32E provides high definition 3-dimensional information about the surrounding environment.

Specifications	
Laser:	• Class 1 - eye safe • 905 nm wavelength • Time of flight distance measurement • Measurement range 70 m [1m to 70m)
Sensor:	• 32 laser/detector pairs • +10.67 to -30.67 degrees field of view (vertical) • 360 degree field of view (horizontal) • 10 Hz frame rate • Operating temperature -10° to +60° C • Storage temperature - 40° to 105° C • Accuracy: <2 cm (one sigma at 25 m) • Angular resolution (vertical) ~ 1.33° • Angular resolution (horizontal) ~ 0.16° at 600 rpm
Mechanical:	• Power: 12V @ 2 Amps • Operating voltage: 9-32 VDC • Weight: <2 kg • Dimensions: 5.9" height x 3.4" diameter • Shock: 500 m/sec² amplitude, 11 msec duration • Vibration: 5 Hz to 2000 Hz, 3 Grms • Environmental Protection: 1P67
Output:	• Approximately 700,000 points/second • 100 Mbps Ethernet connection • UDP packets - distance - rotation angle • Orientation - internal MEMS accelerometers and gyros for six-axis motion external correction • GPS time-synchronized with included GPS Receiver

97-0038c HDL-32E Data Sheet, Mar 2012

Velodyne LiDAR, Inc.
345 Digital Drive
Morgan Hill, CA 95037

408.465.2800

Appendix C

Notation Similarities and Differences

The following table contains an overview on the similarities and the few differences between the notations from the book of Stan Z. Li [Li09] and this thesis. The intention was to adopt the MRF terminology as close to the textbook as possible in order to preserve a consistent terminology and easier comparability with similar works.

Identifier	Li's book	This thesis
The set of sites.	$\mathcal{S}$	$\mathcal{S}$
The set of labels.	$\mathcal{L}$	$\mathcal{L}$
The set of cliques.	$\mathcal{C}$	$\mathcal{C}$
A label $\in \mathcal{L}$.	ℓ	l
A clique $\in \mathcal{C}$.	c	c
A mapping from $\mathcal{S}$ to $\mathcal{L}$.	f	f
The configuration (labeling) of a random field.	f	$\mathcal{F}$
The configuration/solution space.	$\mathbb{F}$	$\mathbb{F}$
A random field.	F	R
A neighborhood system.	$\mathcal{N}$	$\mathcal{N}$
A Graph.	$\mathcal{G}$	G
An energy function.	$E(\cdot)$	E$(\cdot)$
The partition function.	$Z(\cdot)$	Z$(\cdot)$
A clique potential function.	$V(\cdot)$	V$(\cdot)$
A temperature constant.	T	T
An energy value in a Gibbs distribution.	U	U

Appendix D

Haralick Features

Haralick et al. [HSD73] introduce the famous 14 statistical texture features computed from the co-occurence matrix $\boldsymbol{C}$ of an $n \times m$ image:

$$\boldsymbol{C}_{\delta_x,\delta_y}(i,j) = \sum_{p=1}^{n}\sum_{q=1}^{m}\begin{cases}1, & \text{if } \mathrm{im}(p,q) = i \text{ and } \mathrm{im}(p+\triangle x, q+\triangle y) = i \\ 0, & \text{otherwise}\end{cases}$$

where $p(i,j)$ is the (i,j)-th entry of the normalized *gray-level co-occurence matrix* (also referred to as *gray tone spatial dependency matrix*). The partial probability density functions are denoted by p_x and p_y.

Angular Second Moment	$f_1 = \sum_i \sum_j \{p(i,j)\}^2$
Contrast	$f_2 = \sum_{n=0}^{N_{g-1}} n^2 \left\{ \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} p(i,j) \right\}$ where $\lvert i-j\rvert = n$
Correlation	$f_3 = \frac{\sum_i \sum_j (ij)p(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y}$ where $\mu_x, \mu_y, \sigma_x, \sigma_y$ are the means and standard deviations of p_x and p_y
Sum of Squares: Variance	$f_4 = \sum_i \sum_j (i-\mu)^2 p(i,j)$
Inverse Difference Moment	$f_5 = \sum_i \sum_j \frac{1}{1+(i-j)^2} p(i,j)$

Sum Average $$f_6 = \sum_{i=2}^{2N_g} i p_{x+y}(i)$$

Sum Variance $$f_7 = \sum_{i=2}^{2N_g} (i - f_8)^2 p_{x+y}(i)$$

Sum Entropy $$f_8 = \sum_{i=2}^{2N_g} p_{x+y}(i) \log\{p_{x+y}(i)\}$$

Entropy $$f_9 = -\sum_i \sum_j p(i,j) \log(p(i,j))$$

Difference Variance $$f_{10} = \sum_{n=0}^{N_{g-1}} i^2 p_{x-y}(i)$$

Difference Entropy $$f_{11} = \sum_{n=0}^{N_{g-1}} p_{x-y}(i) \log\{p_{x-y}(i)\}$$

Information Measure of Correlation 1 $$f_{12} = \frac{HXY - HXY1}{\max(HX, HY)}$$

where $$HXY = -\sum_i \sum_j p(i,j) \log(p(i,j))$$

HXY and HY are entropies of p_x and p_y

$$HXY1 = -\sum_i \sum_j p(i,j) \log\{p_x(i) p_y(i)\}$$

Information Measure of Correlation 2 $$f_{13} = (1 - \exp[-2.0(HXY2 - HXY)])^{\frac{1}{2}}$$

where $$HXY2 = -\sum_i \sum_j p_x(i) p_y(j) \log\{p_x(i) p_y(j)\}$$

Maximal Correlation Coefficient $$f_{14} = (\text{Second largest eigenvalue of } Q)^{\frac{1}{2}}$$

where $$Q(i,j) = \sum_k \frac{p(i,k)p(j,k)}{p_x(i)p_y(k)}$$

List of Abbreviations

CAD	Computer-aided design
CPU	central processing unit
CSS	color self-similarity
DGPS	Differential Global Positioning System
FN	false negative
FP	false positive
FPR	false positive rate
GPGPU	general-purpose graphics processing unit
GPS	Global Positioning System
GPU	graphics processing unit
GRF	Gibbs random field
GUI	graphical user interface
HOF	Histograms of Oriented Flow
HOG	Histograms of Oriented Gradients
ICP	Iterative Closest Point
IMU	inertial measurement unit
LADAR	Light Detection And Ranging
LIDAR	Light Detection And Ranging
LRF	laser range finder
MRF	Markov random field
NWU	north west up
PCA	Principal component analysis
PGM	Probabilistic Graphical Models
QDEGSAC	random sample consensus for (quasi-)degenerate data
RANSAC	random sample consensus
RBF	radial basis function
ROC	receiver operating characteristics
ROS	Robot Operating System
SLAM	simultaneous localization and mapping
SVD	singular value decomposition
SVM	support vector machine
TP	true positive
TPR	true positive rate

List of Notations

$\mathcal{S}$	set of sites
i or (i, j)	a site either in one-dimensional or two-dimensional indexation
m	number of sites
$\mathcal{L}$	set of labels
l	label
M	number of labels
f	mapping from $\mathcal{S}$ to $\mathcal{L}$
$\mathcal{F}$	sequence of mappings f, called the configuration of a random field
$\boldsymbol{D}$	term describing available data in a general form, can be further specified as observed sensor data or features for example
$\mathbb{F}$	configuration space of a random field, contains all possible configurations $\mathcal{F}$. Also called the solution spac
E	energy function
U	energy value (in a Gibbs distribution)
Z	partition function, a normalization constant in the Gibbs distribution
$\mathcal{N}$	neighborhood system
$\mathcal{N}_i$	set of sites $\in \mathcal{S}$ neighboring site i
$\mathcal{C}$	set of cliques
c	element of $\mathcal{C}$
b	another element of $\mathcal{C}$, $b \subset c$
$\mathcal{A}$	clique set $\subset \mathcal{C}$
$\mathcal{B}$	another clique set $\subset \mathcal{C}$
V	clique potential
T	temperature value
R	random field
d	distance value
β	constant reflecting the pair-site interaction between two sites
G	graph
N	node in a graph
s	sample
S	set of samples
im	image
P or Q	point
$\boldsymbol{R}$	rotation matrix
$\boldsymbol{t}$	transformation vector
t	time

J	set of pairs of laser scan and corresponding camera image
$\boldsymbol{C}$	co-occurence matrix
$p(i, j)$	entry of the co-occurence matrix $\boldsymbol{C}$ at the i-th row and j-th column
μ	mean
σ	standard deviation
$\boldsymbol{H}$	correlation matrix
$\boldsymbol{\Sigma}$	covariance matrix
bel	belief
$\mathfrak{c}$	terrain cell
$\boldsymbol{z}$	virtual scan, a vector of distance measurements in a 360° order
$C_{i,j}$	set of sensor readings of the j-th cell in the i-th cone of a virtual scan
$\boldsymbol{x} = x, y, \theta^{\mathrm{T}}$	pose consisting of position and orientations
$\boldsymbol{q}$	particle
w	importance weight of a particle in a particle filter
T_{P}	sequence of pose, velocity, and geometry of past pedestrian states
T_{V}	sequence of pose, velocity, and geometry of past vehicle states
T'_{V}	sequence of pose, velocity, and geometry of predicted future vehicle states including an additional uncertainty in range $[0, \ldots, 1]$
I	function accumulating all trail and prediction values for a site
$\mathcal{H}$	Hamiltonian
S	Spin
egomotion	refers to the proper motion of the host vehicle
v	velocity

Quotes

[AES+08]	Example of a reference
[Häselich et al., 2011a]	Example of author's own publication
[20]	Example of an internet source

List of Tables

List of Figures

Own publications

[Hahn et al., 2011] Hahn, R., Lang, D., Häselich, M., and Paulus, D. (2011). Heat Mapping for Improved Victim Detection. In *Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 116–121, Kyoto, Japan.

[Häselich et al., 2011a] Häselich, M., Arends, M., Lang, D., and Paulus, D. (2011a). Terrain Classification with Markov Random Fields on fused Camera and 3D Laser Range Data. In *Proceedings of the 5th European Conference on Mobile Robotics*, pages 153–158, Örebro, Sweden.

[Häselich et al., 2013a] Häselich, M., Arends, M., Wojke, N., Neuhaus, F., and Paulus, D. (2013a). Probabilistic Terrain Classification in Unstructured Environments. *Journal of Robotics and Autonomous Systems*, 61(10):1051–1059.

[Häselich et al., 2012a] Häselich, M., Bing, R., and Paulus, D. (2012a). Calibration of Multiple Cameras to a 3D Laser Range Finder. In *Proceedings of the 2012 IEEE International Conference on Emerging Signal Processing Applications*, pages 25–28, Las Vegas, USA.

[Häselich et al., 2012b] Häselich, M., Eggert, S., and Paulus, D. (2012b). Parallelized Energy Minimization for Real-Time Markov Random Field Terrain Classification in Natural Environments. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, pages 1823–1828, Guangzhou, China.

[Häselich et al., 2011b] Häselich, M., Handzhiyski, N., Winkens, C., and Paulus, D. (2011b). Spline Templates for Fast Path Planning in Unstructured Environments. In *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3545–3550, San Francisco, USA.

[Häselich et al., 2014] Häselich, M., Jöbgen, B., Wojke, N., Hedrich, J., and Paulus, D. (2014). Confidence-Based Pedestrian Tracking in Unstructured Environments Using 3D Laser Distance Measurements. In *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4118–4123, Chicago, USA. accepted.

[Häselich et al., 2013b] Häselich, M., Klostermann, M., and Paulus, D. (2013b). Pedestrian Detection in Outdoor Images using Color and Gradients. In *Proceedings of the 6th European Conference on Mobile Robotics*, pages CD–Rom, Barcelona, Spain.

[Lang et al., 2011] Lang, D., Häselich, M., Prinzen, M., Bauschke, S., Gemmel, A., Giesen, J., Hahn, R., Haraké, L., Reimche, P., Sonnen, G., von Steimker, M., Thierfelder, S., and Paulus, D. (2011). RoboCup 2011 - RoboCup Rescue team resko@Uni-Koblenz (Germany). Technical report, University of Koblenz-Landau.

[Lang et al., 2012] Lang, D., Winkens, C., Häselich, M., and Paulus, D. (2012). Hierarchical Loop Detection for Mobile Outdoor Robots. In *Proceedings of SPIE, Intelligent Robots and Computer Vision XXIX: Algorithms and Techniques*, pages 83010P–83010P–11, San Francisco, USA.

[Thierfelder et al., 2011a] Thierfelder, S., Gossow, D., Navarro Luzón, C., Nowack, S., Merten, N., Friedmann, S., Weiland, L., Mies, D., Giesen, J., Häselich, M., Lang, D., Seib, V., and Paulus, D. (2011a). RoboCup 2011 - RoboCup @home team homer@Uni-Koblenz (Germany). Technical report, University of Koblenz-Landau.

[Thierfelder et al., 2011b] Thierfelder, S., Seib, V., Lang, D., Häselich, M., Pellenz, J., and Paulus, D. (2011b). Robbie: A Message-based Robot Architecture for Autonomous Mobile Systems. In *INFORMATIK 2011 - Informatik schafft Communities*, Berlin, Germany.

[Vetter et al., 2010] Vetter, S., Pellenz, J., Lang, D., Häselich, M., Fuchs, C., and Paulus, D. (2010). RoboCup 2010 - RoboCup Rescue team resko@Uni-Koblenz (Germany). Technical report, University of Koblenz-Landau.

[Wojke and Häselich, 2012] Wojke, N. and Häselich, M. (2012). Moving Vehicle Detection and Tracking in Unstructured Environments. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pages 3082–3087, St. Paul, USA.

Bibliography

[AES+08] J. Alberts, D. Edwards, T. Soule, M. Anderson, and M. O'Rourke. Autonomous Navigation of an Unmanned Ground Vehicle in Unstructured Forest Terrain. In *Proceedings of the ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems*, pages 103–108, Edinburgh, Scotland, 2008.

[AJGBHR+11] G. Atanacio-Jiménez, J.-J. González-Barbosa, J.B. Hurtado-Ramos, F.J. Ornelas-Rodríguez, H. Jiménez-Hernández, T. García-Ramirez, and R. González-Barbosa. LIDAR Velodyne HDL-64E Calibration Using Pattern Planes. *International Journal of Advanced Robotic Systems*, 8:70–82, 2011.

[ANP+09] H. Aliakbarpour, P. Nŭñez, J. Prado, K. Khoshhal, and J. Dias. An Efficient Algorithm for Extrinsic Calibration between a 3D Laser Range Finder and a Stereo Camera for Surveillance. In *Proceedings of the 14th International Conferencce on Advanced Robotics*, pages 3854–3859, Munich, Germany, 2009.

[ASK+05] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, and W.-K. Yoon. RT-Middleware: Distributed Component Middleware for RT (Robot Technology). In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3933–3938, 2005.

[ATL07] H. Andreasson, R. Triebel, and A. Lilienthal. *Non-iterative Vision-based Interpolation of 3D Laser Scans*, pages 83–90. Springer, Germany, 2007.

[AV07] D. Arthur and S. Vassilvitskii. K-means++: The Advantages of Careful Seeding. In *Proceedings of the Annual ACM-SIAM Sym-*

posium on Discrete Algorithms, pages 1027–1035, New Orleans, Louisiana, 2007.

[BBS+10] A. Broggi, L. Bombini, C. Stefano, P. Cerri, and R. I. Fedriga. Sensing Requirements for a 13,000 km Intercontinental Autonomous Drive. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 500–505, 2010.

[BCC+10] A. Broggi, A. Cappalunga, C. Caraffi, S. Cattani, S. Ghidoni, P. Grisleri, P. P. Porta, M. Posterli, and P. Zani. TerraMax Vision at the Urban Challenge 2007. *IEEE Transactions on Intelligent Transportation Systems*, 11(1):194–205, 2010.

[Bes74] J. Besag. Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):192–236, 1974.

[BGD10] S. Brechtel, T. Gindele, and R. Dillmann. Recursive Importance Sampling for Efficient Grid-Based Occupancy Filtering in Dynamic Environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3932–3938, 2010.

[BID05] C. A. Brooks, K. Iagnemma, and S. Dubowsky. Vibration-based Terrain Analysis for Mobile Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3415–3420, Barcelona, Spain, 2005.

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006.

[BIS09] M. Buehler, K. Iagnemma, and S. Singh, editors. *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, volume 56 of *Springer Tracts in Advanced Robotics*. Springer Press, 2009.

[BKM+05] A. Brooks, T. Kaupp, A. Makarenko, S. B. Williams, and A. Orebäck. Towards Component-Based Robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 163–168, 2005.

[BRL+09] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Gool. Robust Tracking-by-Detection using a Detector Confidence Particle Filter. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1515–1522, Kyoto, Japan, 2009.

[Bru01] H. Bruyninckx. Open robot control software: the OROCOS project. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2523–2528, 2001.

[BVR01] Y. Boykov, O. Veksler, and Zabih. R. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1222–1239, 2001.

[CBL+06] C. Côté, Y. Brosseau, D. Létourneau, C. Raïevsky, and F. Michaud. Robotic Software Integration Using MARIE. *International Journal of Advanced Robotic Systems*, 3(1):55–60, 2006.

[CCdB06] C. Colombo, D. Comanducci, and A. del Bimbo. Camera Calibration with Two Arbitrary Coaxial Circles. In *Proceedings of the 9th European Conference on Computer Vision*, pages 265–276, Graz, Austria, 2006.

[CCH+12] C.-Y. Chen, H.-J. Chien, P.-S. Huang, W.-B. Hong, and C.-F. Chen. Intrinsic Parameters Calibration for Multi-beam LiDAR using the Levenberg-Marquardt Algorithm. In *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, pages 19–24, New York, USA, 2012.

[CCIN08] D. Calisi, A. Censi, L. Iocchi, and D. Nardi. OpenRDK: A Modular Framework for Robotic Software Development. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1872–1877, 2008.

[CCR11] E. Coyle, E.G. Collins, and R.G. Roberts. Speed independent terrain classification using Singular Value Decomposition Interpolation. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 4014–4019, Shanghai, China, 2011.

[CGM08] V. Caglioti, A. Giusti, and D. Migliore. Mutual Calibration of a Camera and a Laser Rangefinder. In *VISAPP (Workshop on Robot Perception)*, pages 33–42, Funchal, Portugal, 2008.

[CH96] I. Cox and S. Hingorani. An Efficient Implementation of Reid's Multiple Hypothesis Tracking Algorithm and Its Evaluation for the Purpose of Visual Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):138–150, 1996.

[CL11] C.-C. Chang and C.-J. Lin. LIBSVM: A library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.

[Cli90] P. Clifford. Markov Random Fields in statistics. In G. Grimmett and D. Welsh, editors, *Disorder in Physical Systems: A Volume in Honour of John M. Hammersley*, pages 19–32. Oxford University Press, Oxford, UK, 1990.

[CM02] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[CMBR12] G. D. C. Cavalcanti, J. P. Magalhaes, R. M. Barreto, and T. I. Ren. MLPBoost: A Combined AdaBoost / Multi-Layer Perceptron Network Approach for Face Detection. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 2350–2353, Seoul, Korea, 2012.

[CMG05] T.H.J. Collett, B.A. Macdonald, and B. Gerkey. Player 2.0: Toward a Practical Robot Programming Framework. In *Proceedings of the Australasian Conference on Robotics and Automation*, pages –, 2005. Online proceedings.

[CMLL00] D. Coombs, K. Murphy, A. Lacaze, and S. Legowik. Driving Autonomously Offroad up to 35 km/h. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 186–191, Dearborn, USA, 2000.

[CN13] W. Churchill and P. Newman. Experience-based Navigation for Long-term Localisation. *International Journal of Robotics Research*, 32:1645–1661, 2013.

[COY08] A. Carballo, A. Ohya, and S. Yuta. Fusion of Double Layered Multiple Laser Range Finders for People Detection from a Mobile Robot. In *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 677–682, Seoul, Korea, 2008.

[CPL+06] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière. Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application. *International Journal of Robotic Research*, 25(1):19–30, 2006.

[Cro84] F. C. Crow. Summed-Area Tables for Texture Mapping. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques*, pages 207–212, New York, USA, 1984.

[CV95] C. Cortes and V. Vapnick. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[CZJ02] D. Cobzaş, H. Zhang, and M. Jägersand. A Comparative Analysis of Geometric and Image-Based Volumetric and Intensity Data Registration Algorithms. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 2506–2511, Washington, USA, 2002.

[DBP10] P. Dollar, S. Belongie, and P. Perona. The Fastest Pedestrian Detector in the West. In *Proceedings of the British Machine Vision Conference*, pages 68.1–68.11, London, UK, 2010.

[DC04] H. Deng and D. A. Clausi. Unsupervised image segmentation using a simple MRF model with a new implementation scheme. *Pattern Recognition*, 37(12):2323–2335, 2004.

[dD10] A. d'Angelo and J.-L. Dugelay. A Markov Random Field Description of Fuzzy Color Segmentation. In *Proceedings of the 2nd International Conference on Image Processing Theory, Tools and Applications*, pages 270–275, Paris, France, 2010.

[DdMR00] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183, Stanford, USA, 2000.

[Dij59] E. Dijkstra. A Note on Two Problems in Connection with Graphs. *Numerical Mathematics*, 1(1):269–271, 1959.

[DMH+09] D. Droeschel, S. May, D. Holz, P. G. Ploeger, and S. Behnke. Robust Ego-Motion Estimation with ToF Cameras. In *Proceedings of the 4th European Conference on Mobile Robotics*, pages 187–192, Dubrovnik, Croatia, 2009.

[DSS01] K. Dietmayer, J. Sparbert, and D. Streller. Model Based Object Classification and Object Tracking in Traffic Scenes from Range Images. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 25–30, 2001.

[DT05a] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 886–893, San Diego, USA, 2005.

[DT05b] J. Diebel and S. Thrun. An Application of Markov Random Fields to Range Sensing. In *Advances in Neural Information Processing Systems*, pages 291–298, Vancouver, Canada, 2005.

[DTPB09] P. Dollar, Z. Tu, P. Perona, and S. Belongie. Integral Channel Features. In *Proceedings of the British Machine Vision Conference*, pages 91.1–91.11, London, UK, 2009.

[DTS06] N. Dalal, B. Triggs, and C. Schmid. Human Detection Using Oriented Histograms of Flow and Appearance. In *Proceedings of the European Conference on Computer Vision*, pages 428–441, Graz, Austria, 2006.

[DWSP12] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.

[ELF97] D. W. Eggert, A. Lorusso, and R. B. Fisher. Estimating 3-D Rigid Body Transformations: a Comparison of Four Major Algorithms. *Machine Vision Applications*, 9:272–290, 1997.

[EVGW+10] M. Everingham, L. J. Van Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[FB81] M. Fischler and R. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[FCH+08] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C. J. Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9(1):1871–1874, 2008.

[FF56] L. R. Ford and D. R. Fulkerson. Maximal Flow Through a Network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[FF62] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[FGI05] A. Farinelli, G. Grisetti, and L. Iocchi. SPQR-RDK: a modular framework for programming mobile robots. In *Proceedings of International RoboCup Symposium*, pages 653–660, 2005.

[FGMR10] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[FP06] J.-M. Frahm and M. Pollefeys. RANSAC for (Quasi-)Degenerate Data (QDEGSAC). In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 453–460, 2006.

[FPM+05] A. Fonseca, L. Pimenta, R. Mesquita, R. Saldanha, and G. Pereira. Path planning for mobile robots operating in outdoor environments using map overlay and triangular decomposition. In *Proceedings of the International Congress of Mechanical Engineering*, Orlando, USA, 2005.

[FS95] Y. Freund and R. E. Schapire. A Decision-Theoretic Generalization of Online Learning and an Application to Boosting. In *Proceedings of the European Conference on Computational Learning Theory*, pages 23–37, Santa Cruz, USA, 1995.

[GBSD09] T. Gindele, S. Brechtel, J. Schröder, and R. Dillmann. Bayesian Occupancy Grid Filter for Dynamic Environments Using Prior Map Knowledge. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 669–676, 2009.

[GCB+10] S. Gidel, P. Checchin, C. Blanc, T. Chateau, and L. Trassoudaine. Pedestrian Detection and Tracking in an Urban Environment Using a Multilayer Laser Scanner. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):579–588, 2010.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distribution, and bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

[GL10] C. Glennie and D.D. Lichti. Static Calibration and Analysis of the Velodyne HDL-64E S2 for High Accuracy Mobile Scanning. *Remote Sensing*, 2:1610–1624, 2010.

[GLGG11] J. Gonzalez, Y. Low, A. Gretton, and C. Guestrin. Parallel Gibbs Sampling: From Colored Fields to Thin Junction Trees. *Journal of Machine Learning Research - Proceedings Track*, 15:324–332, 2011.

[GNB00] J. Guivant, E. Nebot, and S. Baiker. Autonomous navigation and map building using laser range sensors in outdoor applications. *Journal of Robotic Systems*, 17(10):3817–3822, 2000.

[GPS89] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact Maximum A Posteriori Estimation for Binary Images. *Journal of the Royal Statistical Society. Series B (Methodological)*, 51(2):271–279, 1989.

[Gri76] D. Griffeath. Introduction to Random Fields. In Kemeny et al. [KSK76], pages 425–458.

[HAO+11] A. Huang, M. Antone, E. Olson, L. Fletcher, D. Moore, S. Teller, and J. Leonard. A High-rate, Heterogeneous Data Set from the DARPA Urban Challenge. *International Journal of Robotics Research*, 29(13):1595–1601, 2011.

[HC71] J. M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. Unpublished; see however [Cli90], 1971.

[HD95] R. Horaud and F. Dornaika. Hand-eye Calibration. *International Journal of Robotics Research*, 14:195–210, 1995.

[HK07] T. Howard and A. Kelly. Optimal Rough Terrain Trajectory Generation for Wheeled Mobile Robots. *International Journal of Robotics Research*, 26(1):141–166, 2007.

[HNR68] P. Hart, N. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum-Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, SSC-4(2):100–107, 1968.

[HOJ06] M. Happold, M. Ollis, and N. Johnson. Enhancing Supervised Terrain Classification with Predictive Unsupervised Learning. In *Proceedings of Robotics: Science and Systems*, page Online proceedings: http://www.roboticsproceedings.org/rss02/p06.html, Philadelphia, USA, 2006.

[HRF11] E. Herbst, X. Ren, and D. Fox. RGB-D Object Discovery via Multi-Scene Analysis. In *Proceedings of the 2011 IEEE/RSJ international Conference on Intelligent Robots and Systems*, pages 4850–4856, San Francisco, USA, 2011.

[HS81] B. K. P. Horn and B. G. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17(1-2):185–203, 1981.

[HSD73] R. Haralick, K. Shanmugam, and I. Dinstein. Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 3(6):610–621, 1973.

[HZ03] R. I. Hartley and A Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[KAB+06] K. Konolige, M. Agrawal, R. Bolles, C. Cowan, M. Fischler, and B. Gerkey. Outdoor Mapping and Navigation using Stereo Vision.

In *Proceedings of the International Symposium on Experimental Robotics*, pages 179–190, Rio de Janeiro, Brazil, 2006.

[Kaw10] H. Kawamura. Two models of spin glasses – Ising versus Heisenberg. *Journal of Physics: Conference Series*, 233(1):–, 2010. Electronic Journal.

[KBZ96] Z. Kato, M. Berthod, and J. Zerubia. A Hierarchical Markov Random Field Model and Multi-Temperature Annealing for Parallel Image Classification. *Computer Vision, Graphics and Image Processing: Graphical Models and Image Processing*, 58:19–37, 1996.

[KJ12] B. Kulis and M. Jordan. Revisiting k-means: New Algorithms via Bayesian Nonparametrics. In *Proceedings of the International Conference on Machine Learning*, pages 513–520, Edinburgh, Scotland, 2012.

[KKBZ11] Y.N. Khan, P. Komma, K. Bohlmann, and A. Zell. Grid-based visual terrain classification for outdoor robots using local feature. In *Proceedings of the 2011 IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems*, pages 16–22, Paris, France, 2011.

[KKOO07] Y.-S. Kim, B. K. Kim, K. Ohba, and A. Ohya. Localization of Outdoor Mobile Robot with Multi-Path Bias Detection. In *Proceedings of the IEEE International Conference on Mechatronics and Automation*, pages 705–710, Harbin, China, 2007.

[KM10] U. Knauer and B. Meffert. Fast Computation of Region Homogeneity with Application in a Surveillance Task. In *Proceedings of ISPRS Commission V Mid-Term Symposium Close Range Image Measurement Techniques*, pages 337–342, Newcastle, UK, 2010.

[KMW+11] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura. Pedestrian recognition using high-definition LIDAR. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 405–410, Baden-Baden, Germany, 2011.

[KP06] Z. Kato and T Pong. A Markov Random Field Image Segmentation Model for Color Textured Images. *Image and Vision Computing*, 24(10):1103–1114, 2006.

[KS80] R. Kindermann and J.L. Snell. *Markov Random Fields and Their Applications*. American Mathematical Society, 1980.

[KSK76] J. G. Kemeny, J. L. Snell, and A. W. Knapp. *Denumerable Markov Chains*. Springer, 1976.

[KWSD04] N. Kämpchen, T. Weiss, M. Schaefer, and Dietmayer.K. IMM Object Tracking for High Dynamic Driving Maneuvers. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 825–830, 2004.

[KWZ09] P. Komma, C. Weiss, and A. Zell. Adaptive bayesian filtering for vibration-based terrain classification. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3307–3313, Kobe, Japan, 2009.

[LBH+07] J. Leonard, D. Barrett, J. How, S. Teller, M. Antone, S. Campbell, A. Epstein, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, T. Jones, O. Koch, Y. Kuwata, K. Mahelona, D. Moore, K. Moyer, E. Olson, S. Peters, C. Sanders, J. Teo, and M. Walter. Team MIT Urban Challenge Technical Report. Technical report, Massachusetts Institute of Technology, Cambridge, USA, 2007.

[LF10] K. Lai and D. Fox. Object Recognition in 3D Point Clouds Using Web Data and Domain Adaptation. *International Journal of Robotic Research*, 29(8):1019–1037, 2010.

[LHT+08] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, O. Koch, Y. Kuwata, D. Moore, E. Olson, S. Peters, J. Teo, R. Truax, M. Walter, D. Barrett, A. Epstein, K. Maheloni, K. Moyer, T. Jones, R. Buckley, M. Antone, R. Galejs, S. Krishnamurthy, and J. Williams. A Perception-Driven Autonomous Urban Vehicle. *Journal of Field Robotics*, 25(10):727–774, 2008.

[Li09] S. Li. *Markov Random Field Modeling in Computer Vision*. Springer, 2009.

[LKZ13] S. Laible, Y. Khan, and A. Zell. Terrain Classification with Conditional Random Fields on Fused 3D LIDAR and Camera Data. In *Proceedings of the 6th European Conference on Mobile Robotics*, pages CD–Rom, Barcelona, Spain, 2013.

[LL96] L. D. Landau and E. M. Lifshitz. *Statistical Physics*. Butterworth Heinemann, 1996.

[LLD+07] G. Li, Y. Liu, L. Dong, X. Cai, and D. Zhou. An algorithm for extrinsic parameters calibration of a camera and a laser range

finder using line features. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3854–3859, 2007.

[LM97] F. Lu and E. Milios. Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans. *Journal of Intelligent and Robotic Systems*, 18(3):249–275, 1997.

[LMDM98] A. Lacaze, Y. Moscovitz, N. Declaris, and K. Murphy. Path Planning for Autonomous Vehicles Driving Over Rough Terrain. In *Proceedings of the IEEE ISIC/CIRA/ISAS Joint Conference*, pages 50–55, Gaithersburg, USA, 1998.

[LS04] P. Lamon and R. Siegwart. Inertial and 3D-odometry Fusion in Rough Terrain - Towards Real 3D Navigation. In *Proceedings of the 2011 IEEE/RSJ international Conference on Intelligent Robots and Systems)*, pages 1716–1721, Sendai, Japan, 2004.

[Mac67] J. MacQueen. Some Methods for Classification and Analysis of MultiVariate Observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley, USA, 1967.

[Mat81] H. Matsuda. The Ising Model for Population Biology. *Progress of Theoretical Physics*, 66(3):1078–1080, 1981.

[MBB+08] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Hähnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and Sebastian Thrun. Junior: The Stanford Entry in the Urban Challenge. *Journal of Field Robotics*, 25(9):569–597, 2008.

[MBM08] S. Maji, A. C. Berg, and J. Malik. Classification Using Intersection Kernel Support Vector Machines is Efficient. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2245–2260, Anchorage, USA, 2008.

[MCH+09] I. Miller, M. Campbell, D. Huttenlocher, A. Nathan, F. Kline, P. Moran, N. Zych, B. Schimpf, S. Lupashin, E. Garcia, J. Catlin, M. Kurdziel, and H. Fujishima. Team Cornell's Skynet: Robust Perception and Planning in an Urban Environment. In *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, pages 257–304, Berlin, Heidelberg, Germany, 2009.

[MCTM04] R. Manduchi, A. Castano, A. Talukder, and L. Matthies. Obstacle Detection and Terrain Classification for Autonomous Off-road Navigation. *Autonomous Robots*, 18(1):81–102, 2004.

[MG10] N. B. Murphy and K. M. Golden. The Ising model and critical behavior of transport in binary composite media. *Journal of Mathematical Physics*, 53(6):–, 2010. Electronic Journal.

[MHM08] D. Morris, R. Hoffman, and S. McLean. Ladar-Based Vehicle Detection and Tracking in Cluttered Environments. In *Proceedings of the 26th Army Science Conference*, 2008.

[MKH10] O. Mozos, R. Kurazume, and T. Hasegawa. Multi-Part People Detection Using 2D Range Data. *International Journal of Social Robotics*, 2(1):31–40, 2010.

[MKR12] F.M. Mirzaei, D.G. Kottas, and S.I. Roumeliotis. 3D LIDAR-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. *International Journal of Robotics Research*, 31:452–467, 2012.

[MO11] R. D. Morton and E. Olson. Positive and Negative Obstacle Detection using the HLD Classifier. In *Proceedings of the 2011 IEEE/RSJ international Conference on Intelligent Robots and Systems*, pages 1579–1584, San Francisco, USA, 2011.

[Mou74] J. Moussouris. Gibbs and Markov Random Systems with Constraints. *Journal of Statistical Physics*, 10(1):11–33, 1974.

[MR06] C. Mei and P. Rives. Calibration between a Central Catadioptric Camera and a Laser Range Finder for Robotic Applications. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 532–537, Orlando, USA, 2006.

[MS10] J. Martens and I. Sutskever. Parallelizable Sampling of Markov Random Fields. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 517–524, Sardinia, Italy, 2010.

[MT06] M. Montemerlo and S. Thrun. Large-Scale Robotic 3-D Mapping of Urban Structures. In *Proceedings of the 9th International Symposium on Experimental Robotics*, pages 141–150, Singapore,Singapore, 2006.

[MYTOM02] V. Meas-Yedid, S. Tilie, and J.-C. Olivo-Marin. Color Image Segmentation Based on Markov Random Field Clustering for Histological Image Analysis. In *Proceedings of the 16th International Conference on Pattern Recognition*, pages 796–799, Quebec City, Canada, 2002.

[NDJRD09] P. Nŭñez, P. Drews Jr, R. Rocha, and J. Dias. Data Fusion Calibration for a 3D Laser Range Finder and a Camera using Inertial Data. In *Proceedings of the 4th European Conference on Mobile Robots (ECMR)*, pages 31–36, Mlini/Dubrovnik, Croatia, 2009.

[NDPP09] F. Neuhaus, D. Dillenberger, J. Pellenz, and D. Paulus. Terrain Drivability Analysis in 3D Laser Range Data for Autonomous Robot Navigation in Unstructured Environments. In *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation*, pages 1686–1689, Catalonia, Spain, 2009.

[Nes07] I.A. Nesnas. CLARAty: A Collaborative Software for Advancing Robotic Technologies. In *Proceedings of the NASA Science and Technology Conference*, pages –, 2007. Online proceedings.

[NSMH10] L. Navarro-Serment, C. Mertz, and M. Hebert. Pedestrian Detection and Tracking Using Three-dimensional LADAR Data. *International Journal of Robotics Research, Special Issue: Seventh International Conference on Field and Service Robots*, 29(12):1516–1528, 2010.

[Nüc09] A. Nüchter. *3D Robotic Mapping: The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom*. Springer Press, 2009.

[ORMG07] M. W. Otte, S. G. Richardson, J. Mulligan, and G. Grudic. Local path planning in image space for autonomous robot navigation in unstructured environments. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 212–240, San Diego, California, 2007.

[OTS+03] K. Ohno, T. Tsubouchi, B. Shigematsu, S. Maeyama, and S. Yuta. Outdoor Navigation of a Mobile Robot between Buildings Based on DGPS and Odometry Data Fusion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1978–1984, Taipei, Taiwan, 2003.

[PCBC09] T. Pock, A. Chambolle, H. Bischof, and D. Cremers. A Convex Relaxation Approach for Computing Minimal Partitions. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 810–817, Miami, Florida, 2009.

[Pel11] J. Pellenz. *Aktive Sensorik für autonome mobile Systeme*. PhD thesis, University of Koblenz-Landau, Koblenz, Germany, March 2011.

[PKTN07] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng. Touch Based Perception for Object Manipulation. In *Proceedings of Robotics Science and Systems Conference*, Atlanta, USA, 2007.

[PLN09] C. Premebida, O. Ludwig, and U. Nunes. Exploiting LIDAR-based Features on Pedestrian Detection in Urban Scenarios. In *Proceedings of the International IEEE Conference on Intelligent Transportation Systems*, pages 405–410, St. Louis, USA, 2009.

[PMSE10] G. Pandey, J. McBride, S. Savarese, and R. Eustice. Extrinsic Calibration of a 3D Laser Scanner and an Omnidirectional Camera. In *7th IFAC Symposium on Intelligent Autonomous Vehicles*, Lecce, Italy, 2010.

[Por05] F. Porikli. Integral Histogram: A Fast Way To Extract Histograms in Cartesian Spaces. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 829–836, San Diego, USA, 2005.

[PP00] C. Papageorgiou and T. Poggio. A Trainable System for Object Detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[PR09] V. Prisacariu and I. Reid. fastHOG - a real-time GPU implementation of HOG. Technical report, Department of Engineering Science, Oxford University, Oxford, UK, 2009.

[PS05] R. Philippsen and R. Siegwart. An Interpolated Dynamic Navigation Function. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3782–3789, Barcelona, Spain, 2005.

[PT09] A. Petrovskaya and S. Thrun. Model Based Vehicle Detection and Tracking for Autonomous Urban Driving. *Autonomous Robots, Special Issue: Selected papers from Robotics: Science and Systems 2008*, 26(2–3):123–139, 2009.

[QAB+11] I.-U.-H. Qazi, O. Alata, J.-C. Burie, M. Abadi, A. Moussa, and C. Fernandez-Maloigne. Parametric Models of Linear Prediction Error Distribution for Color Texture and Satellite Image Segmentation. *Computer Vision and Image Understanding*, 115:1245–1262, 2011.

[QCG+09] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source Robot Operating System. In *IEEE International Conference on Robotics and Automation:Workshop on Open Source Software*, 2009.

[RHM03] A. Rankin, A. Huertas, and L. Matthies. Negative obstacle detection by thermal signature. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 906–913, Las Vegas, USA, 2003.

[RVNY07] G. Reina, A. Vargas, K. Nagatani, and K. Yoshida. Adaptive Kalman Filtering for GPS-based Mobile Robot Localization. In *Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics*, pages 1–6, Rome, Italy, 2007.

[SATS10] L. Spinello, K. Arras, R. Triebel, and R. Siegwart. A Layered Approach to People Detection in 3D Range Data. In *Proceedings of the AAAI Conference on Artificial Intelligence: Physically Grounded AI Track*, Atlanta, USA, 2010.

[SBM06] Z. Sun, G. Bebis, and R. Miller. On-Road Vehicle Detection: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5):694–711, 2006.

[SBS+11] F. Scholer, J. Behley, V. Steinhage, D. Schulz, and A.B. Cremers. Person Tracking in Three-Dimensional Laser Range Data with Explicit Occlusion Adaption. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1297–1303, Shanghai, China, 2011.

[SC07] T. Schenk and B. Csatho. Fusing Imagery and 3D Point Clouds for Reconstructing Visible Surfaces of Urban Scenes. In *Proceedings of the IEEE GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas*, pages 11–13, Paris, France, 2007.

[Sch09] K.D. Schmidt. *Maß Und Wahrscheinlichkeit*. Springer, 2009.

[SGVG10] S. Stalder, H. Grabner, and L. Van Gool. Cascaded Confidence Filtering for Improved Tracking-by-Detection. In *Proceedings of*

the European Conference on Computer Vision, pages 369–382, Crete, Greece, 2010.

[SHS07] D. Scaramuzza, A. Harati, and R. Siegwart. Extrinsic Self Calibration of a Camera and a 3D Laser Range Finder from Natural Scenes. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4164–4169, San Diego, California, 2007.

[SHT+10] S. Sato, M. Hashimoto, M. Takita, K. Takagi, and T. Ogawa. Multilayer lidar-based pedestrian tracking in urban environments . In *Intelligent Vehicles Symposium*, pages 849–854, San Diego, USA, 2010.

[Sie02] J. Siedersleben. Quasar: Die sd & m Standardarchitektur. Technical report, sd & m Research, Munich,Germany, 2002.

[SLA11] L. Spinello, M. Luber, and K. Arras. Tracking People in 3D Using a Bottom-Up Top-Down Detector. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1304–1310, Shanghai, China, 2011.

[SMP05] T. Svoboda, D. Martinec, and T. Pajdla. A Convenient Multi-Camera Self-Calibration for Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 14(4):407–422, 2005.

[SS12] S. Saremi and T. J. Sejnowski. Hierarchical model of natural images and the origin of scale invariance. *Proceedings of the National Academy of Sciences of the United States of America*, 110(8):3071–3076, 2012.

[SW87] R. H. Swendsen and J. S. Wang. Nonuniversal Critical Dynamics in Monte Carlo Simulations. *Physical Review Letters*, 58:86–88, 1987.

[SZC+00] T. Szirányi, Josiane Zerubia, LászLó Czúni, D. Geldreicch, and Z. Kato. Image Segmentation Using Markov Random Field Model in Fully Parallel Cellular Network Architectures. *Real-Time Imaging*, 6(3):195–211, 2000.

[SZS+08] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1068–1080, 2008.

[THM08] S. Thornton, M. Hoffelder, and D. Morris. Multi-sensor Detection and Tracking of Humans for Safe Operations with Unmanned Ground Vehicles. In *IEEE Workshop on Human Detection from Mobile Platforms*, pages 103–112, Pasadena, USA, 2008.

[TK09] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, 2009.

[UAB+08] C. Urmson, J. Anhalt, H. Bae, J. Bagnell, C. Baker, R. Bittner, T. Brown, M. Clark, M. Darms, D. Demitrish, J. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, M. Likhachev, B. Litkouhi, A. Kelly, M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y. Seo, S. Singh, J. Snider, J. Struble, A. Stentz, M. Taylor, W. Whittaker, Z. Wolkowicki, W. Zhang, and J. Ziglar. Autonomous Driving in Urban Environments: Boss and the Urban Challenge. *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge*, 25(1):425–466, 2008.

[UH05] R. Unnikrishnan and M. Hebert. Fast Extrinsic Calibration of a Laser Rangefinder to a Camera. Technical report, Robotics Institute, Pittsburgh, USA, 2005.

[USEK02] H. Utz, S. Sablatnog, S. Enderle, and G. Kraetzschmar. Miro - Middleware for Mobile Robot Applications. *IEEE Transactions on Robotics and Automation*, 18(4):493–497, 2002.

[vdMW04] R. van der Merwe and E. A. Wan. Sigma-Point Kalman Filters for Integrated Navigation. In *Proceedings of the 60th Annual Meeting of The Institute of Navigation*, Dayton, USA, 2004.

[VHKH04] N. Vandapel, D. Huber, A. Kapuria, and M. Hebert. Natural Terrain Classification using 3-D Ladar Data. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5117–5122, New Orleans, USA, 2004.

[VJ11] P. A. Viola and M. J. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 511–518, Kauai, USA, 2011.

[VJS03] P. A. Viola, M. J. Jones, and D. Snow. Detecting Pedestrians Using Patterns of Motion and Appearance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 734–741, Nice, France, 2003.

[VNH+11] R. Voigt, J. Nikolic, C. Huerzeler, S. Weiss, L. Kneip, and R. Siegwart. Robust Embedded Egomotion Estimation. In *Proceedings of the 2011 IEEE/RSJ international Conference on Intelligent Robots and Systems*, pages 2694–2699, San Francisco, USA, 2011.

[Vol09] A. Volk. Verwendung von 3D Laserscan-Daten zur Verbesserung der Selbstlokalisation eines mobilen Systems im Outdoor-Bereich. Master's thesis, University of Koblenz-Landau, Institute for Computational Visualistics, Active Vision Group, Koblenz, 2009.

[VTL08] P. Vernaza, B. Taskar, and D. Lee. Online, self-supervised terrain classification via discriminatively trained submodular Markov random fields. In *IEEE International Conference on Robotics and Automation*, pages 2750–2757, Pasadena, USA, 2008.

[Vu09] T. Vu. *Vehicle Perception: Localization, Mapping with Detection, Classification and Tracking of Moving Objects*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, France, September 2009.

[Wan04] C. Wang. *Simultaneous Localization, Mapping and Moving Object Tracking*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, April 2004.

[WCS05] C. Wellington, A. Courville, and A. Stentz. Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection. In *Proceedings of Robotics Science and Systems*, pages 1–8, Cambridge, USA, 2005.

[WD07] S. Wender and K. Dietmayer. 3D Vehicle Detection Using a Laser Scanner and a Video Camera. In *Proceedings of Sixth European Congress on Intelligent Transport Systems*, pages 105–112, 2007.

[WKSB09] K. Wurm, R. Kümmerle, C. Stachniss, and W. Burgard. Improving robot navigation in structured outdoor environments by identifying vegetation from laser data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1217–1222, St. Louis, USA, 2009.

[WMSS10] S. Walk, N. Majer, K. Schindler, and B. Schiele. New Features and Insights for Pedestrian Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1030–1037, San Francisco, USA, 2010.

[Wol89] U. Wolff. Collective Monte Carlo Updating for Spin Systems. *Physical Review Letters*, 62:361–364, 1989.

[WS08] D. Wolf and G. Sukhatme. Semantic Mapping Using Mobile Robots. *IEEE Transactions on Robotics*, 24(2):245–258, 2008.

[WSFB05] D. Wolf, G. Sukhatme, D. Fox, and W. Burgard. Autonomous Terrain Mapping and Classification Using Hidden Markov Models. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2026–2031, Barcelona, Spain, 2005.

[WTS03] C. Wang, C. Thorpe, and A. Suppe. Ladar-based detection and tracking of moving objects from a ground vehicle at high speeds. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 416–421, 2003.

[WWS09] C. Wojek, S. Walk, and B. Schiele. Multi-Cue Onboard Pedestrian Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 794–801, Miami, USA, 2009.

[YB03] C. Ye and J. Borenstein. A new terrain mapping method for mobile robots obstacle negotiation. In *Proceedings of the UGV Technology Conference at the SPIE AeroSense Symposium*, pages 21–25, Orlando, Florida, 2003.

[YB04] C. Ye and J. Borenstein. A Method for Mobile Robot Navigation on Rough Terrain. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3863–3869, New Orleans, USA, 2004.

[YSS00] A. Yahja, S. Singh, and A. Stentz. An Efficient On-line Path Planner for Outdoor Mobile Robots. *Robotics and Autonomous Systems*, 33(1):129–143, 2000.

[Zha00] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1330–1334, 2000.

[ZP04] Q. Zhang and R. Pless. Extrinsic Calibration of a Camera and Laser Range Finder. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 2301–2306, Sendai, Japan, 2004.

[ZT98] L. Zhao and C. Thorpe. Qualitative and Quantitative Car Tracking from a Range Image Sequence. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 496–501, 1998.

[ZYCA06] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1491–1498, New York, USA, 2006.

Internet resources

[1] AUDI AG. Die Fahrerassistenzsysteme von morgen. https://www.audi-mediaservices.com/publish/ms/content/de/public/hintergrundberichte/2012/03/08/0/die_fahrerassistenzsysteme0.standard.gid-oeffentlichkeit.html, April 2013.

[2] AUDI AG. Parkhauspilot. https://www.audi-mediaservices.com/publish/ms/content/de/public/hintergrundberichte/2012/02/29/0/parkhauspilot.standard.gid-oeffentlichkeit.html, April 2013.

[3] Bouguet, J.Y. Camera Calibration Toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/calib_doc, June 2013.

[4] Bruyninckx, H. The Orocos Project - Smarter control in robotics & automation. http://www.orocos.org, February 2013.

[5] Dawes, B. and Abrahams, D. and Rivera, R. Boost C++ Libraries. http://www.boost.org, July 2013.

[6] Defense Advanced Research Projects Agency. Darpa Grand Challenge. http://www.darpagrandchallenge.com, February 2013.

[7] Defense Advanced Research Projects Agency. Darpa Urban Challenge. http://archive.darpa.mil/grandchallenge, February 2013.

[8] itseez. OpenCV (OPEN SOURCE COMPUTER VISION). http://www.opencv.org, June 2013.

[9] Khronos Group. OpenCL. http://www.khronos.org/opencl, July 2013.

[10] Logitech Europe S.A. HD Pro C910. http://www.logitech.com/en-us/support/hd-pro-webcam-c910, February 2013.

[11] MSO Meßtechnik und Ortung GmbH. Radarsensor Speed Wedge. http://www.mso-technik.de/sensor/speedwedge.html, June 2013.

[12] Philips Electronics N.V. SPC1300NC. http://www.p4c.philips.com/cgi-bin/dcbint/cpindex.pl?ctn=SPC1300NC/00, February 2013.

[13] Roboterwerk GmbH. FORBOT: Outdoor robot platform for research and science. http://www.roboterwerk.de/en, February 2013.

[14] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. Middlebury MRF Energy Minimization Page. http://vision.middlebury.edu/MRF/eccv06/, July 2006.

[15] Tadokoro, S. and Noda, I. and Nardi, D. Rescue Robots Competition. http://www.robocuprescue.org, February 2013.

[16] Thrun, S. What we're driving at. http://googleblog.blogspot.de/2010/10/what-were-driving-at.html, April 2013.

[17] Tragant Handels und Beteiligungs GmbH. Navilock GPS Receiver - NL-402U GPS. http://www.navilock.de/produkte/G_60095/merkmale.html, February 2013.

[18] C. Urmson. Toyota Motor Corp. and Lexus Showcase Advanced Active Safety Research Vehicle at CES - Lexus Research Vehicle Part of Evolving Automated Technologies That Aim to Make Driving Safer. http://pressroom.lexus.com/releases/toyota+lexus+advanced+active+safety+research+vehicle+ces+jan7.htm, April 2013.

[19] Urmson, C. The self-driving car logs more miles on new wheels. http://googleblog.blogspot.de/2012/08/the-self-driving-car-logs-more-miles-on.html, April 2013.

[20] Velodyne Lidar Inc. Velodyne Lidar. http://www.velodyne.com/lidar, February 2013.

[21] VisLab srl. The VisLab Intercontinental Autonomous Challenge. http://viac.vislab.it, April 2013.

[22] Willow Garage. ROS (Robot Operating System). http://www.ros.org, February 2013.

[23] Xsens Europe. Xsens MTi - Miniature Attitude and Heading Sensor. http://www.xsens.com/en/general/mti, February 2013.