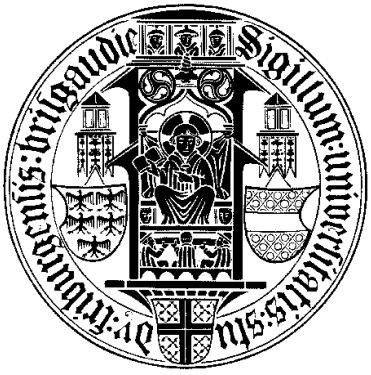
Dissertation zur Erlangung des Doktorgrades der Fakultät für Angewandte Wissenschaften der Albert-Ludwigs-Universität Freiburg im Breisgau

Mapping and Exploration for Search and Rescue with Humans and Mobile Robots

Alexander D. Kleiner



2007

Tag der Disputation:
20. Februar 2008

Dekan:
Prof. Dr. Bernhard Nebel, *Albert-Ludwigs-Universität Freiburg*

Gutachter:
Prof. Dr. Bernhard Nebel, *Albert-Ludwigs-Universität Freiburg*
Prof. Dr. Daniele Nardi, *University of Rome "Sapienza"*

Zusammenfassung

Katastrophenhilfe ist eine zeitkritische Aufgabe bei der Überlebende innerhalb der ersten 72 Stunden gerettet werden müssen. Ein Ziel der Rettungsrobotik ist die Unterstützung dieser Aufgabe mit kooperierenden Teams bestehend aus Menschen und Robotern. Dabei sollen das Katastrophengebiet flächendeckend erkundet und Positionen Überlebender an eine zentrale Befehlsstelle zur Planung von Rettungsmissionen gemeldet werden. Dies kann nur effizient erfolgen wenn Menschen und Roboter das Katastrophengebiet gemeinsam kartieren und gleichzeitig ihre Suche koordinieren. Roboter sollten dabei in der Lage sein, Teilprobleme, wie z.B. Navigation und das Aufspüren von Opfern, autonom durchzuführen. Aufgrund der unstrukturierten Beschaffenheit von Katastrophengebieten und der Unzugänglichkeit des Geländes ist die Bewältigung dieser Aufgaben außerordentlich schwierig. Des Weiteren müssen entwickelte Lösungen auch bei einem Totalausfall der Kommunikation dezentral funktionsfähig sein.

In dieser Dissertation wird ein vereinheitlichter Ansatz für Menschen und Roboter zur Lösung dieser Probleme vorgestellt. Dabei werden Kernprobleme, wie z.B. Positionsbestimmung auf unwegsamen Gelände, Kartenerstellung durch heterogene Teams und dezentrale Team Koordination mit eingeschränkter Kommunikation direkt behandelt. Es wird die Methode "RFID-SLAM" zur robusten und effiziente Kartierung großräumiger Gebiete vorgestellt, welche RFID Technologie zur Wiedererkennung von Orten einsetzt. Die vorgestellte Methode beinhaltet einerseits die fortwährende Positionsbestimmung von Menschen und Robotern, jeweils mittels Fußgänger-Koppelnavigation und rutschemfindlicher Radodometrie, und andererseits eine graphenbasierte Kartenoptimierung. Das Verfahren kann zentral oder dezentral erfolgen, wobei im Gegensatz zu herkömmlichen Methoden, ein wiederholtes Besuchen von Orten einzelner Teammitglieder zum optimieren der Karte nicht erforderlich ist. Aufbauend auf dieser Kartenrepräsentation werden für die Koordination größerer Rettungsteams eine deliberative Methode zur Bestimmung der Aufgabenverteilung und Multiagenten-Pfadplanung und eine Methode zur lokalen Suche unter der Verwendung des Speichers von RFIDs, vorgestellt.

Zur autonomen Navigation von Robotern auf unwegsamen Gelände und automatischen Erkennung von Menschen in Katastrophengebieten werden einerseits eine effiziente Methode zur Erstellung von Höhenkarten und andererseits ein Ansatz zur genetischen Optimierung von "Markov Random Field" Modellen präsentiert. Abschließend wird eine *human in the loop* Architektur vorgestellt, welche die Integration von gesammelten Daten einzelner Rettungseinheiten in ein Multiagentensystem erlaubt. In diesem Zusammenhang wird die zentrale Koordination großräumiger Katastrophenhilfe mittels Agententechnologie beschrieben.

Die in dieser Arbeit vorgestellten Verfahren wurden weitgehend in Outdoor-Szenarien und offiziellen Testszenarien zum Katastrophenschutz getestet. Sie waren ein wesentlicher Bestandteil von Systemen, die insgesamt mehr als zehn mal den ersten Platz bei internationalen Wettbewerben, wie z.B. den RoboCup Weltmeisterschaften, erreichten.

Abstract

Urban Search And Rescue (USAR) is a time critical task since all survivors have to be rescued within the first 72 hours. One goal in Rescue Robotics is to support emergency response by mixed-initiative teams consisting of humans and robots. Their task is to explore the disaster area rapidly while reporting victim locations and hazardous areas to a central station, which then can be utilized for planning rescue missions.

To fulfill this task efficiently, humans and robots have to map disaster areas jointly while coordinating their search at the same time. Additionally, robots have to perform subproblems, such as victim detection and navigation, autonomously. In disaster areas these problems are extraordinarily challenging due to the unstructured environment and rough terrain. Furthermore, when communication fails, methods that are deployed under such conditions have to be decentralized, i.e. operational without a central station.

In this thesis a unified approach joining human and robot resources for solving these problems is contributed. Following the vision of combined multi-robot and multi-human teamwork, core problems, such as position tracking on rough terrain, mapping by mixed teams, and decentralized team coordination with limited radio communication, are directly addressed. More specific, RFID-SLAM, a novel method for robust and efficient loop closure in large-scale environments that utilizes RFID technology for data association, is contributed. The method is capable of jointly improving multiple maps from humans and robots in a centralized and decentralized manner without requiring team members to perform loops on their routes. Thereby positions of humans are tracked by PDR (Pedestrian Dead Reckoning), and robot positions by slippage-sensitive odometry, respectively. The joint-graph emerging from these trajectories serves as an input for an iterative map optimization procedure. The introduced map representation is further utilized for solving the centralized and decentralized coordination of large rescue teams. On the one hand, a deliberate method for combined task assignment and multi-agent path planning, and on the other hand, a local search method using the memory of RFIDs for coordination, are proposed.

For autonomous robot navigation on rough terrain and real-time victim detection in disaster areas an efficient method for elevation map building and a novel approach to genetic MRF (Markov Random Field) model optimization are contributed. Finally, a human in the loop architecture is presented that integrates data collected by first responders into a multi-agent system via wearable computing. In this context, the support and coordination of disaster mitigation in large-scale environments from a central-command-post-perspective are described.

Methods introduced in this thesis were extensively evaluated in outdoor environments and official USAR testing arenas designed by the National Institute of Standards and Technology (NIST). Furthermore, they were an integral part of systems that won in total more than 10 times the first prize at international competitions, such as the RoboCup world championships.

To Wei Fang (魏方) and my family,
for their love and support,
who provided the foundation
of all my endeavors.

Acknowledgements

My work was accompanied and supported by many people which took influence by different means. In my opinion, there are basically three different categories of good influence: motivation/virtue, team work/learning experience, and entertainment, which are all equally important. Although these three categories overlap with either friends or colleagues, and in some cases influence came through simply exchanging a few words, and in other cases through years of joint work, I would like to thank them in this way.

First of all, I would like to thank my adviser Professor Bernhard Nebel who generously made the building of each robot and agent system that we designed financially possible, but more importantly, who motivated me. Furthermore, his integrity and correctness polarized my personal development from the very beginning, and I very much appreciate the liberties I received to work on my projects.

During my first years in Freiburg I very much enjoyed the time with the CS Freiburg robot soccer team, particularly the work with Thilo Weigel, Markus Dietl, and Steffen Gutmann. From the nights I spent with them in the laboratory or at RoboCup venues I learned, above all, endurance and determination, which I later carried on to many other teams.

During visits of Rescue Robotics related conferences and events I met a lot of highly active people in the community, which greatly motivated my work. Hence, I would like to thank Adam Jacoff for his restless efforts in setting up every year new performance metrics, even if it means to saw countless cubic meters of wood. Furthermore, he motivated robotics researchers by enabling discussions on disaster mitigation with emergency responders from the field. My work was also motivated by the commitment of Prof. Satoshi Tadokoro, Prof. Mike Lewis, and Prof. Daniele Nardi. I like to thank them for all the efforts they have undertaken for the community, and also for giving research directions.

Many thanks to Dirk Hähnel for the straightforward discussions we had on SLAM, and supporting our team while building-up the first rescue robot.

Many thanks to my friends and colleagues for the wonderful team work we experienced. First of all I would like to thank Christian Dornhege, who accompanied me for the longest time. Besides his broad knowledge and ability to solve problems practically, his diligence and discipline enabled many systems to function as they should. Furthermore, I like to thank Rainer Kümmerle and Bastian Steder for their brilliant work with *Rescue Robots Freiburg*. It is their individual skills that lay the basis for strong team work, particularly Basti's ability to solve any mechanical and electronic problem by

a hot-glue gun, cable fixer and shrink tubing, and Rainer's understanding of putting mathematics into source code. Thanks to all other team members contributing essential parts to the systems that we built with *ResQ Freiburg* and *Rescue Robots Freiburg* from 2004-2006: Tobias Bräuer, Michael Brenner, Jörg Stückler, Moritz Göbelbecker, Mathias Luber, Daniel Meyer-Delius, Johann Prediger, Michael Ruhnke, and Michael Schnell. Also many thanks to Dali Sun who walked several kilometers to collect PDR (Pedestrian Dead Reckoning) data evaluated in this thesis.

Vittorio Amos Ziparo (Don Vito) from Italy visited our laboratory for one year as a guest researcher. On the one hand his way of making jokes and taking things easy made working together very pleasant. On the other hand his strong motivation to bring things to an end, which I was certain about as he still did not want to sleep after two days and nights of intensive work, and his way of taking the consequences from decisions made, finally lead to the success we had.

I would like to thank Holger Kenn, for the nice team work we had, his optimistic thinking, and his directness. Anyway, he was the first person working on rescue robots who I met.

My thanks to all my friends and colleagues for the great time I had in Freiburg. Special thanks to Michael Brenner for all the meaningful discussions on multi-agent planning and also for enduring me in our office for so long. Thanks to Sebastian Kupferschmidt, Dapeng Zhang, and Malte Helmert for having table sports and beer garden events. Also many thanks to Gian Diego Tipaldi and Giorgio Grisetti for the late night excursions, as well as for teaching me on SLAM.

Finally, most thanks to my oldest friend Jörg Ziegler, my family Julia, Brigitte, and Dieter Kleiner, and most of all Wei Fang, for supporting me and enjoying life with me.

Table of Contents

1 Introduction

1.1 Motivation

Large scale urban disaster situations, such as earthquakes, floods or terrorist attacks pose an immense threat to modern urban civilization centers. Examples such as the Great Hanshin earthquake in Kobe, the Pacific Tsunami, the hurricane Kathrina or the terrorist attacks of 9/11 demonstrate that although disaster response plans were in place, these events pushed existing countermeasures over their limits, resulting in loss of human lives, and long-term adversary effects on the affected regions.

During Urban Search And Rescue (USAR), time is of utmost importance. Rescue teams have to explore large terrain within a short amount of time in order to locate survivors after a disaster. In this scenario, the number of survivors decreases drastically by each day due to hostile environmental conditions and the victims' injuries. Therefore, the survival rate depends significantly on the efficiency of rescue teams.

Furthermore, cooperation is a must during rescue operations for disaster mitigation. In general the problem is not solvable by a single unit, yet a heterogeneous team that dynamically combines individual capabilities in order to solve the task is needed [Murphy et al., 2000]. This is due to the structural diversity of disaster areas, variety of evidence on human presence that sensors can perceive, and the necessity to quickly and reliably examining targeted regions. Mixed teams not only offer the possibility to field such diverse capabilities, they also exhibit increased robustness due to redundancy [Dias et al., 2004], and increased performance due to parallel task execution.

Particularly the aspect of integrating mobile robots in human search teams, and thus to combine robotics technology with human expertise, attracted increasing attention in recent years. One goal there is to build *mixed-initiative* teams [Wang and Lewis, 2007], consisting of autonomous robots that are partially controlled by human incident commanders. Their task is to jointly create a map of the disaster area and to register victim locations, which are further utilized by human task forces to schedule rescue missions.

To deploy robots for exploration and mapping has multiple advantages. Miniature robots can enter either confined or toxic spaces that humans and search dogs cannot. Search dogs are of great assistance during emergency response. However, at the WTC at 9/11 they were hindered by rain, which limited their sense of smell to only 0.3 meters. While victims were assumed to be located at a depth of 30 meters they could

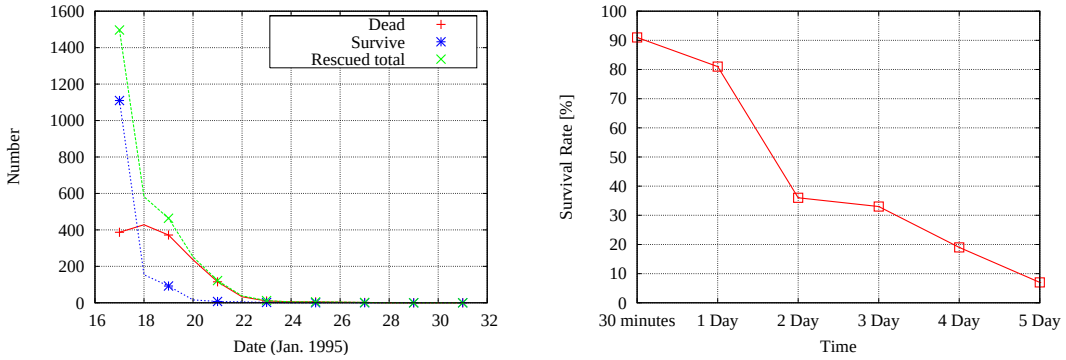
(a) (b)

Figure 1.1: First responders and destroyed building structures after the 9/11 incident. Courtesy of Robin Murphy.

not go deeper than half a meter into the rubble because there was no air [AAAI, 2002]. Furthermore, rescue safety and effectiveness is a serious issue. In the case of the Mexico City earthquake in 1985, 135 rescuers died, many of them while searching confined spaces that were flooded [Casper et al., 2000]. Figure 1.1 depicts collapsed building structures after 9/11.

Rescue workers have not enough time to thoroughly search the area since they are needed for rescuing as many victims as possible within the first 72 hours, which are therefore also called the *golden 72 hours*. After this time, the survival rate decreases drastically (see Figure 1.2). For example, it takes in average 10 trained professionals approximately 4 hours to remove a victim in a void space and 10 trained professionals 10 hours to remove an entombed victim [Administration and Agency, 1995]. Here, robotic technology can be allocated to find victims thereby enabling the humans to concentrate on the rescue. Also, robotic technology can influence rescue missions substantially by providing information on human activities in the field. Tom Haus, a firefighter at 9/11, pointed out these problems when I met him at the NIST emergency responder exercise. He depicted the difficulties first responders encounter in terms of situation awareness and keeping orientation. He noticed during a panel discussion: “We need a tracking system that tells us where we are, where we have been, and where we have to go to”.

A thorough analysis of the requirements for the performance of robots that can support USAR roles and tasks has been undertaken by the US authorities NIST (National Institute of Standards and Technology) and DHS (Department of Homeland Security) [Messina et al., 2005]. They acquired expert knowledge by polling members from twenty FEMA (Federal Emergency Management Agency) task forces and the US National Guard. Among other USAR tasks, they identified reconnaissance and primary search as the two highest priorities for applying robots.



(a) (b)

Figure 1.2: Disaster mitigation statistics: (a) death and survival after the Kobe earthquake, (b) survival rate from the Federal Emergency Management Agency (FEMA). Courtesy of Satoshi Tadokoro.

Due to the fact that Rescue Robotics faces extraordinary challenging problems, such as unstructured environments and hostile conditions, qualitative and quantitative performance metrics are indispensable. During the last years, researchers and organizations developed such performance metrics in order to asses the deployment of robots for disaster mitigation. Casper and Murphy analyzed experiences from a robot deployment at 9/11 [Casper and Murphy, 2003]. They concluded that robot portability and Human Robot Interfaces (HRI) are the most pressing needs. Burke et al. examined operator situation awareness during tele-operated search team interaction during a disaster rescue drill by analyzing communications. Pratt et al. photo documented with an iSENSYS IP3 micro air vehicle damage in multi-story buildings after hurricane Kathrina [Pratt et al., 2006]. They provide logged sensor data and useful hints on their homepage [CRASAR, 2007]. The special project for Earthquake Disaster Mitigation in Urban Areas (DDT Project), started by the Japanese Ministry of Education, Sports, Culture, Science and Technology in 2002, concentrates on the development of robotic solutions for reconnaissance of victim bodies, structural damage and environmental conditions to assist rescue teams in large-scale urban earthquake disasters [Tadokoro, 2005, Tadokoro et al., 2003]. Furthermore, NIST developed a USAR test-bed for the RoboCup/AAAI robot urban search and rescue competition with the goal to promote research in designing intelligent fieldable robots and to provide a benchmark for testing current robot platforms [Jacoff et al., 2000, Kitano and Tadokoro, 2001] (see Section 1.3).

One research challenge in Rescue Robotics is to acquire a map of the environment. This involves autonomously solving in real-time the problem of Simultaneous Localization and Mapping (SLAM), consisting of a continuous state estimation problem and a discrete data association problem. In disaster areas, these problems are extraordinarily challenging. On the one hand, state estimation is difficult due to the extremely unreli-

able odometry measurements usually found on robots operating on rough terrain. On the other hand, data association, i.e. to recognize locations from sensor data, is challenging due to the unstructured environment. Collapsed building structures, and limited visibility conditions due to smoke, dust, and fire, prevent an easy distinction of different places. This problem is also relevant to standard SLAM approaches, which recognize places by associating vision-based and LRF-based features. These extraordinary circumstances make it very difficult to apply common techniques from robotics. Many of these techniques have been developed under strong assumptions, for example, they require polygonal structures, such as typically found in office-like environments [Gutmann and Schlegel, 1996, Grisetti et al., 2002] or depend on predictable covariance bounds from pose tracking for solving the data association problem by validation gating [Dissanayake et al., 2001]. Moreover, SLAM methods work with the principle of map improvement through loop-closure, i.e. to improve the map globally each time places have been re-observed. However, when facing the reality of emergency response, firefighters will intentionally try to avoid performing loops, e.g. while they are searching for victims.

To facilitate teamwork among humans and robots is another challenge in Rescue Robotics. One difficulty that arises is that designing approaches for supporting emergency response from a single robot/human perspective is prone to yield suboptimal solutions. The joint performance of a team depends on assembling the right mixture of individual human and robot capabilities, and thus has to be designed as a whole. Another difficulty that arises is the exchange of information, such as map data and victim locations, e.g. requiring reliable methods for merging individually generated map pieces. While there has been extensive research on building maps from a single agent perspective, and also on combining multiple maps offline, there has been only little attention on addressing the problem of decentralized SLAM, i.e. to jointly improve a map online based on sporadic point-to-point communications between the agents. In emergency response scenarios, this issue is of major importance since communication bandwidth is a truly limited resource. This also affects the problem of team coordination, i.e. requiring approaches being operational even if communication is not possible at all. Finally, robots have to be capable of detecting victims in the field, which is rather difficult in real-time given arbitrary shapes and colors found in disaster areas.

1.2 Contribution

In this thesis a unified approach combining human and robot resources for supporting disaster mitigation is presented. Whereas the main contribution is on autonomous mapping and exploration of disaster areas, also solutions for human detection and top-level coordination of rescue teams are presented. Following the vision of building an overall approach of combined multi-robot and multi-human teamwork, core problems, such as

position tracking on rough terrain, joint route graph optimization of multiple units, and team coordination with limited radio communication are directly addressed. Conventional approaches for solving these problems are scrutinized, and improved solutions regarding efficiency and robustness in harsh environments are presented. The thesis describes the following contributions, which will be further explained in detail:

- The design and construction of low-cost robotic hardware for search and rescue research
- A method for slippage sensitive pose tracking on wheeled vehicles
- A method for visual odometry-based pose tracking on tracked vehicles
- RFID-SLAM: an algorithm for decentralized and centralized SLAM
- A local search method for decentralized team coordination
- A deliberative approach for centralized team coordination
- A method for building elevation maps on rough terrain in real-time
- A method for genetic optimization of MRF (Markov Random Field) models for real-time victim detection
- A *human-in-the-loop* architecture building upon the RoboCup Rescue simulation kernel for disaster mitigation

RFID-SLAM is a novel method for robust and efficient loop closure in large-scale environments that utilizes RFID tags for data association. RFID tags are small devices that carry a worldwide unique number that can be read from distant places, hence, offering a robust way to label and recognize locations in unstructured environments. Passive RFID tags do not require to be equipped with batteries since they are powered by the reader if they are within reading distance. Their reading and writing distance, which depends on the employed communication frequency, can be assumed to be within a range of meters.

RFID-SLAM is capable of jointly improving multiple trajectories from humans and robots and requires the autonomous deployment of RFID tags in the environment for building a network of connected locations. Thereby the pose of each human is automatically tracked by PDR (Pedestrian Dead Reckoning), which recognizes human footsteps analytically from acceleration patterns. For the purpose of robot tracking, a slippage-sensitive odometry that reduces odometry errors on slippery ground, as for example, if the robot navigates on rough terrain, has been developed.

With the centralized version of RFID-SLAM, humans and robots are estimating the distances between RFID tags by pose tracking and communicate them to a central station. The central station successively builds a joint graph from these estimates and corrects the joint network of all trajectories by minimizing the Mahalanobis distance [Lu and Milios, 1997], while utilizing RFID transponders for data association [Kleiner et al., 2006c]. One advantage of RFID-SLAM is that single agents are not required to perform loops on their routes since those are automatically generated by merging individual tracks of the team members. Furthermore, DRFID-SLAM, a decentralized version of RFID-SLAM, is contributed. DRFID-SLAM utilizes information sharing between humans and robots via the memory of RFIDs. Hence, does not depend on radio communication directly.

Team coordination can be generally decomposed into task assignment and multi-agent path planning. Whereas task assignment and multi-agent path planning were both intensively studied as separate problems in the past, there has been only little attention on solving both of them at once, particularly if large robot teams are involved. This is mainly due to the fact that the joint state space of the planning problem grows enormously with the number of robots. However, particularly in destroyed environments where robots have to overcome narrow passages and obstacles, path coordination is essential in order to avoid collisions and deadlocks. In this thesis a novel deliberative approach that reduces significantly the size of the search space by utilizing RFID tags as coordination points, while solving the problem of task assignment and path planning simultaneously, is contributed. The method utilizes the same RFID infrastructure as RFID-SLAM. Hence, global path planning is carried out on a graph topology, which is computationally cheaper than planning on global grid maps, as it is usually the case.

Furthermore, a local search method that uses the memory of RFIDs for labeling visited places in the field is contributed. Since data exchange is carried out via the memory of RFIDs, the method can also be applied if radio communication fails completely. Additionally, the local approach has the advantage that computational costs do not grow with the number of robots participating in the search.

To utilize RFID technology in emergency response scenarios has two further advantages: first, RFID tags that have been distributed into the environment can be used in a straightforward manner by humans as landmarks to follow routes towards victim locations and exits, i.e. they do not need to localize themselves within a metric map. Second, RFID tags can be used by human task forces to store additional information regarding nearby places for subsequent rescue teams. The idea of labeling locations with information that is important to the rescue task has already been applied in practice. During the disaster relief in New Orleans in 2005, rescue task forces marked building entrances with information concerning, for example, hazardous areas or victims inside the buildings [FEMA, 2003]. The RFID-based marking of locations is a straight forward extension of this concept.

In order to navigate autonomously in disaster areas, robots need to have an adequate representation of the environment. For this purpose, trajectories, as for example generated by pose tracking and RFID-SLAM, can be taken as a basis for subsequently integrating range measurements from a Laser Range Finder (LRF) into a consistent map. In the context of emergency response, robots have to cope with rough terrain, e.g. an unstructured environment containing obstacles of arbitrary shape in three dimensions. However, standard approaches for mapping, such as occupancy grid maps [Moravec, 1988], lead to a two-dimensional representation of the environment, hence not supplying an adequate representation in this context. Furthermore, pose tracking cannot reliably be performed from wheel odometry or scan matching only.

Therefore, an efficient method for building elevation maps in real-time, i.e. to map the environment while the robot is in motion, is contributed. The proposed approach tracks the three-dimensional pose of the robot by integrating measurements from an orientation sensor and a novel method of visual odometry combined with scan matching. The visual odometry tracks salient features with the KLT feature tracker [Tomasi and Kanade, 1991] over images taken by the camera, and computes from the tracked features the translation of the robot. Furthermore, the three-dimensional pose is updated from height observations that have been registered on the map. Given the three-dimensional pose, the height value of each map cell is estimated by integrating readings from a downwards tilted LRF. Due to the integration of the full three-dimensional pose, the method allows to create elevation maps online while the robot traverses rough terrain, as for example while driving over ramps and stairs.

To search for victims in disaster areas requires robots to perform subtasks, such as victim detection, partially or even fully autonomous. Due to the real-time constraint in rescue-like applications, only fast computable techniques are admissible. However, the detection rate of fast classifiers, such as *color thresholding*, *motion detection*, and *shape detection* turns out to be moderate, since they are typically tuned for specific features, as for example a specific face color at a particular illumination. Hence, in environments containing many diverse objects, they tend to produce a large number of detections, from which in the worst case most are *false-positives*, i.e. objects that are wrongly recognized as victims. One solution to this problem is to combine local evidences, i.e. evidences that are close to each other in the real world, and to reason on their true class label with respect to their neighborhood relations. Markov Random Fields (MRFs) provides a probabilistic framework for representing such local dependencies. However, inference in MRFs is computational expensive, and hence not generally applicable in real-time.

A novel approach for the genetic optimization of MRF models is contributed, which determines offline relevant neighborhood relations, for example the relevance of the relation between heat and motion, with respect to the data. These preselected types are then utilized for generating MRF models during runtime. First, the vertices of the MRF graph are constructed from the output of the weak classifiers. Second, edges between

these nodes are added if the specific type of nodes can be connected by an edge type that was selected during the optimization procedure. One advantage of the proposed approach is that it can easily be extended for other sources of evidence, such as CO_2 and audio noise detection.

In order to develop an overall strategy for disaster mitigation, information from single units, such as detected victims and traveled trajectories, has to be collected and analyzed centrally, e.g. at a central command post. Only a consistent view on the disaster situation allows to efficiently coordinate and schedule rescue missions performed by human and robot teams in the field. Therefore, solutions are needed that increase the *situation awareness* of an incident commander from a central perspective, and support the process of decision making.

The advantage of central data fusion is that the generated model of the disaster area can be taken as input for algorithms that analyze the current situation and compute optimal strategies for disaster mitigation and first responder risk reduction. Research in the field of Multi-Agent Systems (MAS) offers a rich set of solutions for this purpose. Since in a MAS the number of situations and joint actions performed by agents is typically large, close-to-disaster-reality simulations are needed for evaluating these methods. The *RoboCup Rescue* simulation system aims at simulating large-scale disasters and exploring new ways for the autonomous coordination of rescue teams [Kitano et al., 1999]. Here the goal is to provide a platform for developing software agents that react to disaster situations by coordinating police, ambulance and firefighters. This goal leads to challenges like the coordination of heterogeneous teams of hundreds of agents, the exploration of large-scale environments in order to localize victims, as well as the scheduling of time-critical rescue missions.

Systems that integrated human resources via agent proxies into a MAS are referred to as *human-in-the-loop* architectures. A human-in-the-loop architecture is contributed that integrates first responders via wearable computing into the RoboCup Rescue simulation system. The proposed wearable device allows to acquire disaster relevant data. Thereby locations of first responders can either be tracked by GPS positioning or PDR combined with RFID-SLAM. Finally, the acquisition of real-world data and the coordination of disaster mitigation in large-scale environments from a central-command-post-perspective, is demonstrated.

Methods introduced in this thesis were extensively evaluated in outdoor environments, as well as in official Urban Search and Rescue (USAR) test arenas designed by the National Institute of Standards and Technology (NIST) [Jacoff et al., 2001]. Furthermore, the proposed methods were an integral part of robotic or multi-agent systems that won in total more than 10 times the first prize at international competitions, such as the RoboCup world championships. Presented results show that these methods perform robustly and efficiently in the utilized benchmark scenarios.

1.3 RoboCup Rescue

Disaster mitigation is a serious social issue which involves a very large number of heterogeneous units in hostile environment. Due to the close-to-reality requirement for approaches intended to be applied in this domain, it is necessary to introduce benchmarks for evaluating them before they are deployed. After the Great Hanshin Earthquake 1996 in Kobe, the Japanese government decided to promote research related to problems during large-scale urban disasters. One of the outcomes of this initiative was the RoboCup Rescue competition. Using the same successful method of competition-based benchmarks that the Robot Soccer competition was applying, the RoboCup Federation added two new competitions to RoboCup in 2001, the Rescue Robot League and the Rescue Simulation League, with the main goals of promoting research and development of robot and multi-agent technology for disaster mitigation, and their thorough evaluation regarding real-world deployments [Tadokoro et al., 2000].

In the Rescue Robot league, physical robots are designed and tested in simulated disaster situations which are mainly designed by the U.S. National Institute for Standards and Technology (NIST) [Jacoff et al., 2003, Messina et al., 2005]. There, the aim of robots is to map an unknown environment and provide information about simulated disaster victims, such as their location and situation, their simulated medical condition and other helpful indications such as ID tags. The Rescue simulation league aims at simulating large-scale disasters and exploring new ways for the autonomous coordination of rescue teams [Kitano et al., 1999]. The goal of teams participating in the competition is to provide a software system that reacts to a simulated disaster situation by coordinating a group of agents. This goal leads to challenges like the coordination of heterogeneous teams, the exploration of a large-scale environment in order to localize victims, as well as the scheduling of time-critical rescue missions. Agents have only a limited amount of communication bandwidth they can use to coordinate with each other. The problem cannot be addressed by a single entity, but has to be solved by a true multi-agent system. Moreover, the simulated environment is highly dynamic and only partially observable by a single agent. Agents have to plan and decide their actions asynchronously in real-time.

Both leagues change elements of the competition and scoring functions from competition to competition to foster the development of new capabilities and sustain a continuous progress towards obtaining real-world usable systems, which is the long-term goal of the leagues. Comparing the results of RoboCup Rescue over the last years, one notices a remarkable progress in specific areas. Whereas early platforms in the robot league were mainly wheel-based, later developments included innovative tracked-based robots capable of climbing stairs and ramps by controlling three independent track segments [Koyanagi et al., 2006]. Approaches for Simultaneous Localization And Mapping (SLAM) have been improved from simple compass-based map integration [Birk, 2003] towards mapping of unstructured environments in three dimensions [Surmann

et al., 2004]. Furthermore, purely human operator based systems have been partially replaced by autonomous robot architectures [Kleiner et al., 2006b].

The simulation league developed techniques for multi-agent strategy planning and team coordination in an inherently distributed rescue effort [Kleiner and Ziparo, 2006, Kleiner et al., 2004], as well as on the development of information infrastructures and decision support systems for enabling incident commanders to efficiently coordinate rescue teams in the field. For example, Schurr introduced a system based on software developed in the Rescue Simulation league for the training and support of incident commanders in Los Angeles [Schurr et al., 2005].

1.4 Structure of the thesis

The thesis is structured as follows. In Chapter 2 design criteria for rescue robots are discussed and hardware platforms that were developed for the evaluation of the proposed methods, are introduced. In Chapter 3 methods for pose tracking of humans and robots in USAR-like scenarios are introduced and evaluated. The RFID technology-based SLAM approach (RFID-SLAM), which is based on pose tracking methods described in the previous chapter, is introduced and evaluated in Chapter 4. Here the problems of SLAM by robots, SLAM by humans, and the jointly mapping by humans and robots are addressed. Furthermore, a decentralized version of RFID-SLAM is presented.

An efficient method for building elevation maps in real-time is presented and evaluated in Chapter 5. This method is particularly tailored for online processing, thus facilitating autonomous robot behaviors while navigating on rough terrain. Chapter 6 introduces and evaluates a novel method for coordinating large robot teams in USAR scenarios. The method utilizes the memory of RFIDs for enabling communication-free team coordination. A method for detecting humans in USAR environments is introduced and evaluated in Chapter 7. The introduced approach allows real-time detection of victims based on optimized Markov Random Fields (MRFs). Finally, in Chapter 8 we discuss an interface for human responders allowing the integration of data that has been collected in the field into a Multi-Agent System (MAS). Furthermore, based on the central view from an incident commander generated from this data, methods for coordinating emergency responders are introduced.

1.5 Relevant publications and awards

Parts of the thesis have been published in the following journal articles, conference, symposium, and workshop proceedings:

Journals and Book Chapters

- **Real-time Localization and Elevation Mapping within Urban Search and Rescue Scenarios** joint work with C. Dornhege [Kleiner and Dornhege, 2007] (contained in Chapter 3, Chapter 4, and Chapter 5).
- **Towards heterogeneous robot teams for disaster mitigation: Results and Performance Metrics from RoboCup Rescue** joint work with S.Balakirsky, S.Carpin, M. Lewis, A. Visser, J.Wang, and V. A. Ziparo [Balakirsky et al., 2007] (contained in Chapter 6).
- **Game AI: The shrinking gap between computer games and AI systems** [Kleiner, 2005].

Conference Papers

- **Genetic MRF model optimization for real-time victim detection in Search and Rescue** joint work with R. Kümmerle [Kleiner and Kümmerle, 2007] (contained in Chapter 7).
- **Decentralized SLAM for Pedestrians without direct Communication** joint work with D. Sun [Kleiner and Sun, 2007] (contained in Chapter 3 and Chapter 4).
- **Behavior maps for online planning of obstacle negotiation and climbing on rough terrain** joint work with C. Dornhege [Dornhege and Kleiner, 2007a] (partially contained in Chapter 5).
- **Cooperative Exploration for USAR Robots with Indirect Communication** joint work with V.A. Ziparo, L. Marchetti, A. Farinelli, and D. Nardi [Ziparo et al., 2007a] (contained in Chapter 6).
- **RFID-Based Exploration for Large Robot Teams** joint work with V.A. Ziparo, B. Nebel, and D. Nardi [Ziparo et al., 2007b] (contained in Chapter 6).
- **RFID Technology-based Exploration and SLAM for Search And Rescue** joint work with J. Prediger, and B. Nebel [Kleiner et al., 2006c] (contained in Chapter 4).
- **Successful Search and Rescue in Simulated Disaster Areas** joint work with M. Brenner, T. Bräuer, C. Dornhege, M. Göbelbecker, M. Luber, J. Prediger, J. Stückler, and B. Nebel [Kleiner et al., 2005a] (contained in Chapter 8).
- **Approaching Urban Disaster Reality: The ResQ Firesimulator** joint work with T. A. Nüssle, and M. Brenner [Nüssle et al., 2004].

Workshop Papers

- **Mapping disaster areas jointly: RFID-Coordinated SLAM by Humans and Robots** joint work with C. Dornhege and D. Sun [Kleiner et al., 2007] (contained in Chapter 4).
- **Towards the Integration of Real-Time Real-World Data in Urban Search and Rescue Simulation** joint work with H. Kenn [Kenn and Kleiner, 2007] (contained in Chapter 8).
- **Wearable computing meets multiagent systems: A real-world interface for the RoboCupRescue simulation platform** joint work with N. Behrens, and H. Kenn [Kleiner et al., 2006a] (contained in Chapter 8).
- **Visual Odometry for Tracked Vehicles** joint work with C. Dornhege [Dornhege and Kleiner, 2006] (contained in Chapter 3).

Team Description Papers

- **RoboCupRescue - Simulation League Team RescueRobots Freiburg (Germany)** joint work with V.A. Ziparo [Kleiner and Ziparo, 2006] (contained in Chapter 6).
- **RoboCupRescue - Robot League Team RescueRobots Freiburg (Germany)** joint work with C. Dornhege, R. Kümmerle, M. Ruhnke, B. Steder, B. Nebel, P. Doherty, M. Wzorek, P. Rudol, G. Conte, S. Durante, and D. Lundstrom [Kleiner et al., 2006b] (contained in Chapter 2).
- **RoboCupRescue - Robot League Team RescueRobots Freiburg (Germany)** joint work with B. Steder, C. Dornhege, D. Höfler, D. Meyer-Delius, J. Prediger, J. Stückler, K. Glogowski, M. Thurner, M. Luber, M. Schnell, R. Kümmerle, T. Burk, T. Bräuer, and B. Nebel [Kleiner et al., 2005b] (contained in Chapter 2).
- **ResQ Freiburg: Team Description and Evaluation** joint work with M. Brenner, T. Bräuer, C. Dornhege, M. Göbelbecker, M. Luber, J. Prediger, and J. Stückler [Kleiner et al., 2004] (contained in Chapter 8).

Awards

Together with teams of students and colleagues we were able to gain the following awards at international competitions:

- In the **RoboCup Mid-sized league** (with CS Freiburg):

  - 1st place at the GermanOpen 2001
  - 1st place at the RoboCup world championship 2001 in Seattle
  - 2nd place at the GermanOpen 2002

- In the **RoboCupRescue Simulation league** (with ResQ Freiburg):
  - 1st place at the GermanOpen 2003
  - 1st place at the GermanOpen 2004
  - 1st place at the RoboCup world championship 2004 in Lisboa
  - Winner of the Infrastructure Competition at the RoboCup world championship 2004 in Lisboa

- In the **RoboCupRescue Robot league** (with RescueRobots Freiburg):
  - 2nd place at the GermanOpen 2005
  - Winner of the mobility award at the GermanOpen 2005 (Together with AIS Fraunhoffer)
  - 1st place of the "Best in class Autonomy" competition at the RoboCup world championship 2005 in Osaka
  - 1st place of the "Best in class Autonomy" competition at the RoboCup world championship 2006 in Bremen

- In the **RoboCupRescue Simulation league** (Virtual Robots) (with Vittorio Ziparo)
  - 1st place at the RoboCup world championship 2006 in Bremen

- In the **RoboCupRescue Simulation league** (Infrastructure competition) (with Nils Behrens and Holger Kenn)
  - Winner of the Infrastructure Competition at the RoboCup world championship 2006 in Bremen

- At the **Sick Robot Day 2007**, held by the *SICK* GmbH (with Jörg Müller and Christian Dornhege)
  - 1st place of the indoor competition

2 Rescue Robotics Hardware

In this chapter, hardware components for rescue robotics, which have been developed or used for experimental evaluations in this thesis, are introduced. On the one hand, design criteria for robot platforms operating in rescue scenarios are discussed. On the other hand, sensor devices, such as Laser Range Finders (LRFs) and IR cameras are introduced. Furthermore, RFID technology, which serves as a basis for algorithms proposed in this thesis, is discussed.

Finally, two robot platforms that have been designed and developed by *Rescue Robots Freiburg* at the University of Freiburg, and methods for Human Robot Interaction (HRI), are described. HRI methods introduced in this chapter are designed for facilitating the control of a team of heterogeneous robots by a single operator. The introduced robot platforms *Zerg* and *Lurker* are tailored for different types of tasks. Whereas the *Zerg* robot aims at large-scale exploration within a robot team, the *Lurker* robot has been particularly designed for overcoming rough terrain, such as climbing stairs and ramps.

The remainder of this chapter is structured as follows. In Section 2.1 general design criteria for rescue robot platforms are discussed. Sensors for navigation and perception that have been utilized for experiments proposed in this thesis, are introduced in Section 2.2 and Section 2.3, respectively. Section 2.4 introduces RFID technology and discusses its importance to rescue robotics. Finally, in Section 2.5 the developed robot platforms are proposed.

2.1 Design criteria for robot platforms

Designing robot platforms for Urban Search and Rescue (USAR) is a challenging problem. A detailed analysis of the requirements for the performance of robots that can support USAR roles and tasks has been undertaken by the US authorities NIST (National Institute of Standards and Technology) and DHS (Department of Homeland Security) [Messina et al., 2005]. They acquired expert knowledge by polling members from twenty FEMA (Federal Emergency Management Agency) task forces and the US National Guard. Among other roles for USAR robots, they identified reconnaissance and primary search, as the two highest priorities for applying robots. The final result of the survey has been utilized to establish performance metrics for USAR robots.

Among other things, the platform has to fulfill the following criteria. First, any tasks in disaster areas, such as finding victims, demands a high degree of versatility from the

(a) (b) (c)

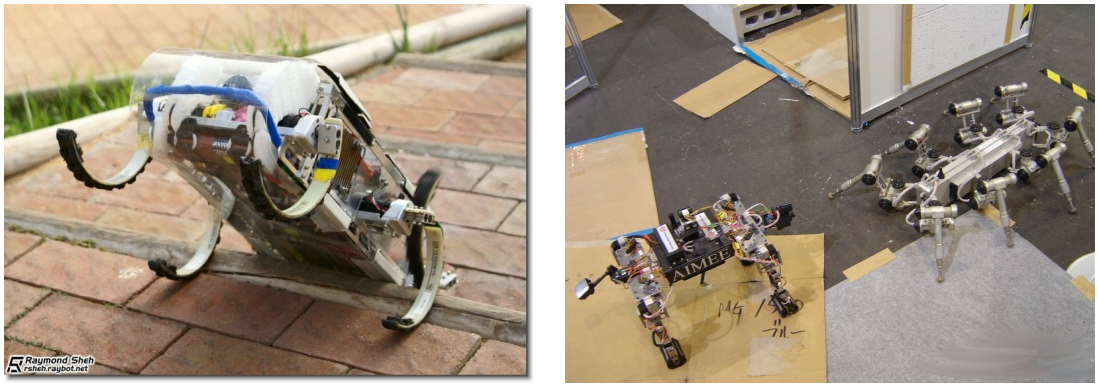
Figure 2.1: Rescue Robots during the NIST "Response Robot Exercise" evaluated by performance metrics for their USAR usability: (a) *IRobot PackBot EOD* with (b) tele-operation console. (c) The *AirRobot* tele-operated for flying reconnaissance missions. Courtesy of Raymond Sheh.

robots. For example, there are places only reachable by climbing, spots only accessible through small openings, and regions only observable from the air. Robots operating in such environments have to be designed with an adequate degree of mobility and size. Ideally, they compose a heterogeneous team combining very different individual capabilities, which is therefore capable of overcoming very different types of terrain. Second, robots and humans must work together in order to successfully accomplish their mission. Hence, robot platforms have to be designed that they are fitting into existing rescue operations, i.e. they have to actively support the work of first responders operating in the field. For example, a platform has to be real-time controllable within confined spaces no less than 30 meters far from the operator and being suitable for long-term deployment of at least 12 hours with rechargeable battery packs. Furthermore, the human-robot interface has to be intuitive usable by untrained personal via a limited-function controller and a user-friendly GUI (Graphical User Interface). Third, robots have to be designed for rugged use and being affordable by rescue organizations.

Figures 2.1 and 2.2 show robots evaluated during the NIST response robot exercise in Gaithersburg (2006), where a wide range of diverse robot platforms participated. Figure 2.1 (a) depicts the *PackBot EOD*, a robot equipped with a manipulator allowing to collect information, such as video images, thermo image signatures, and CO_2 emission, at locations that are difficult to access. Figure 2.1 (c) depicts the *AirBot*. This platform enables first responders to get an overview of a large terrain, or to seek access into

(a) (b)

Figure 2.2: Rescue Robots during the NIST "Response Robot Exercise" evaluated by performance metrics for their USAR usability: (a) The *Soryu* snake robot, and (b) the *BOZ* robot lifting a smaller *DragonRunner* robot to an upper location. Courtesy of Raymond Sheh.



(a) (b)

Figure 2.3: Legged rescue robots: (a) The R-Hex robot. (b) Walking machines from the University of Bremen. Courtesy of Raymond Sheh (a) and Adam Jacoff (b).

buildings from the roof. The *Soryu* snake (Figure 2.2 (a)) has been developed particularly for accessing collapsed buildings through interstices in floors. Furthermore, as depicted in Figure 2.2 (b) there might be tasks that are optimally solved by teamwork between robots. Finally, Figure 2.3 depicts legged rescue robots that are designed for overcoming rough terrain. Their design is mainly inspired by insects, such as ants and cockroaches, which show great mobility capabilities in their natural environment.

Up to now, robot autonomy, i.e. robots controlled by Artificial Intelligence methods, is only rarely found on truly fieldable systems. Rescue robots have been exclusively deployed under tedious human supervision. For example, single control commands, such as positioning a manipulator or moving a robot to a target location, are issued from a tele-operation console, as shown in Figure 2.1 (b). This causes a high burden on human operators and limits the number of robots they can deploy at the same time. Therefore, the ultimate goal is to increase the level of autonomy of rescue robots step-wise, thereby enabling a single operator to control a large team by a small set of high level commands, such as assigning an area that has to be explored, whereas the robots autonomously navigate to their target locations. This goal requires that robots are equipped with sensors for localization and mapping, such as Laser Range Finders (LRFs), and Inertial Measurement Units (IMUs), as well as sufficient computational power.

Requirements for autonomy have a great impact on the design of robot platforms. Unfortunately, most commercially available robot platforms are not particularly designed for autonomy support. On the one hand, an increase of computational capacities causes an increase in the robot's weight, which is, for example, a crucial problem in case of the *AirBot* robot platform. On the other hand, sensors for localization require a special mounting, e.g. LRFs require a free line of sight within an angle of 180° or even 270°, and IMUs have to be mounted outside magnetic fields, as they are for example caused by the motors. In turn, the implementation of these constraints affects the mobility of the platform significantly, or even demands its complete re-design. In Section 2.5 two robot platforms that were particularly developed for autonomy missions are described.

2.2 Sensors for navigation

2.2.1 Inertial Measurement Unit (IMU)

An Inertial Measurement Unit (IMU) is a closed system that is used to detect orientation and motion of a vehicle or human. It typically contains a combination of three accelerometers and three angular rate sensors (gyroscopes), which are placed such that their measuring axes are orthogonal to each other. Accelerometers measure the acceleration by determining inertia forces acting on a body, whereas a gyroscope measures changes of the body's orientation based on the principle of conservation of angular momentum. Gyroscope measurements are unreliable in the long term since they are subject to drift, which can partially be reduced if the IMU has a temperature sensor.

The combination of an IMU with a magnetometer (compass) or a Global Navigation Satellite System (GNSS) is referred to as Attitude and Heading Reference System (AHRS). Such systems typically contain a Kalman filter for fusing the data from multiple sensor sources, yielding an orientation vector that is comparably stable towards sensor drift and magnetic perturbations.

IMUs have been utilized traditionally for aircraft navigation. Due to the increasing miniaturization and decreasing price of these devices, they are more and more applied to car and robot navigation. More recent applications include human motion capture in the context of sports technology, computer animation, and support for the navigation of blind people.

Xsens MTi/MTx

The MTi is a miniature, gyro-enhanced AHRS and combines a tri-axial accelerometer and a tri-axial gyroscope with a tri-axial magnetometer (see Figure 2.4). From these internal sensors, the unit computes a three-dimensional orientation vector with a signal processor, which is updated internally at 512 Hz, and defined by the three Euler angles *yaw* (azimuth or compass angle), *roll* (bank), and *pitch* (elevation). Due to the simultaneous integration of gyro and compass data, the orientation vector is drift-free and stable towards minor perturbations caused by external magnetic sources. The ori-

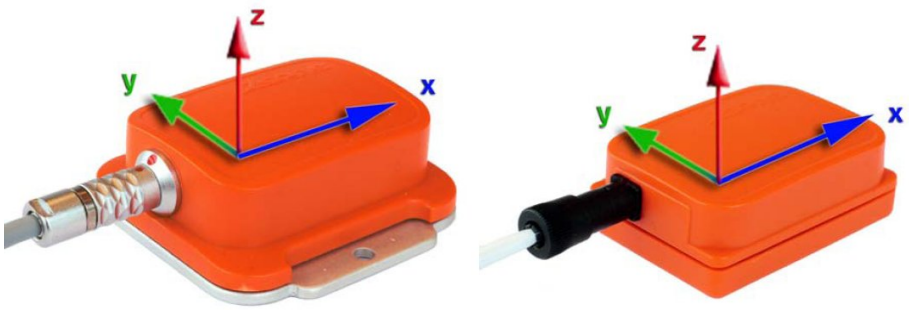


Figure 2.4: Two Inertial Measurement Units (IMUs): The *MTi* and *MTx* from *Xsens*. Courtesy of *Xsens*.

entation vector, the calibrated 3D acceleration, the 3D rate of turn (rate gyro), and the 3D earth-magnetic field data, are accessible via a *RS-232* interface at 120 Hz. Under static conditions, i.e. when the senor is not in motion, the roll and pitch components of the vector have an accuracy of <0.5°, and the *yaw* component an accuracy of <1.0°, whereas under dynamic conditions both have an accuracy of <1.0°.

Since the earth magnetic field is measured and used as a reference by the sensor, external magnetic sources, such as large amounts of ferrous material and electrical wires, as can be found in modern buildings, are serious sources of perturbations. As already mentioned, the unit compensates these errors with measurements from the gyro, which are not affected by magnetic fields. However, there exists a trade-off between the compensation of magnetic fields and the amount of drift that inevitably increases with increasing influence of the gyro. Hence, the sensor unit provides a special modus called

AMD (Adapt to Magnetic Disturbances), which can be activated in environments containing heavy magnetic fields.

The MTx from Xsens is a slightly modified version of the MTi, which is designed particularly for capturing human motion. The device mainly differs from the MTx by the casing's shape and weight and that accelerations are captured at a higher frequency.

2.2.2 Laser Range Finder (LRF)

Laser Range Finders (LRFs) are commonly used in robotics for localization and mapping due to their high accuracy and fast data data rate. They measure the distances to surrounding objects according to the Time Of Flight (TOF) principle. The distance to an object is computed from the time it takes to detect the laser beam that has been emitted by the sensor and reflected by the object. Typically, beams at half-degree resolution are emitted in a Field Of View (FOV) of 180°. A widely used model is the *LMS200* from *Sick* since this sensor is extremely robust and capable to detect objects within a range of 80 meters. However, this device greatly influences the mobility of a robot platform due to its heavy weight of 4.5 kg. A lightweight solution is the *URG-04LX* from *Hokuyo*, which weighs only 160 g. However, its limited range of only 4 meters makes it difficult to apply algorithms for localization and mapping since these require a rich set of features. A maximal range of 4 meters leads in many cases, as for example in outdoor situations, to a large number of *far readings*, i.e. measurements without reflections, whereas in narrow indoor situations, as for example the interior of a rubble pile, are appropriate scenarios for its deployment. The *S300* from *Sick* offers a compromise between both sensors since it is much lighter than the LMS200, but only measures distances up to 30 meters. Table 2.1 summarizes the technical specifications of LRFs that have been utilized during experiments introduced in this thesis.

	Sick LMS200	**Sick S300**	**Hokuyo URG-04LX**
Weight	4500 g	1200 g	160 g
Volume	$\approx 20\,cm^3$	$\approx 15\,cm^3$	$\approx 5\,cm^3$
FOV	180°	270°	240°
Max. Range	80 m	30 m	4 m
Max. Ang. Res.	0.25°	0.5°	0.36°
Accuracy	± 15 mm	± 30 mm	± 10 mm
Scans per second	30	20	10
Interface	RS-232/RS-422	RS-232/RS-422	RS-232/USB

Table 2.1: Comparison of laser range finders.

2.2.3 Global Navigation Satellite System (GNSS)

GNSS (Global Navigation Satellite System) refers to the general class of satellite-based positioning systems, such as *GPS*, *GLONASS*, and *Galileo*, whereas the term GPS virtually refers to the US-American Navstar System only. Since the Global Positioning System (GPS) is the most commonly used GNSS, it will be described in more detail.

GPS is composed of a nominal constellation of 24 satellites *, where the orbits of the satellites are arranged in a way that at least six satellites are always within line of sight from almost anywhere on Earth. GPS positioning is carried out by measuring the distance between the receiver and four or more GPS satellites. Since the transmission speed of radio signals is constant (nearly at the speed of light), the distance to a satellite can be computed from the measured time delay between the transmission and reception of the radio signal, i.e. from the signal's propagation time. This procedure requires extremely accurate clocks on both the receivers and transmitters side. However, receivers are typically equipped with low-cost crystal oscillator-based clocks since accurate atomic clocks, as they are used by the satellites, are too expensive for consumer devices. Therefore, the clock of the receiver has to be synchronized continuously with a satellite clock, requiring to track one satellite only for this purpose.

Each satellite transmits an individual Pseudo Random Code (PRC) from which the identity of the satellite and the time delay of the signal can be determined. Additional information, such as the exact locations of the satellites, is modulated on this signal and decoded by the receiver. From the known locations and distances of satellites the location of the receiver can be deduced by a technique known as trilateration. The technique can be visualized by the intersection of imaginary spheres, which have centers at each satellite location, and radii equal to the corresponding distance measurements. Theoretically, four satellite locations and distances are necessary to deduce the three-dimensional location of the receiver. However, under the assumption that the GPS receiver is located on the earth's surface, the correct location can be disambiguated reliably from only three satellites since one of the two concluded locations is generally outside the satellite orbit.

The farther the tracked satellites are from each other, the better the accuracy of the estimated position since the spheres are intersected by more acute angles. A value expressing the quality of the current satellite constellation is the Horizontal Dilution of Precision *HDOP* value, which is provided by most GPS receivers. In summary, the location of the receiver, denoted by the three unknown parameters *longitude* (East coordinate) and *latitude* (North coordinate), and *altitude*, can be deduced successfully from four satellites, including one satellite for synchronization of the receiver clock.

If there are sufficient satellites within line of sight of the receiver, atmospheric effects have the biggest influence on the accuracy of GPS positioning. Atmospheric effects

*Note that since January 2007 there are 29 broadcasting satellite in the GPS constellation in order to increase the positioning accuracy by redundancy.

might cause a variation of the radio signal's velocity. Particularly the ionosphere, which is 50 - 500 km located above the surface of the earth, contains ionized particles that influence propagation speeds. The fact that these effects are comparably durable within a particular region makes it possible to compensate them on the receiver's side by utilizing a reference measurement that has been taken in the same region. This procedure is known as Differential GPS (DGPS), and is carried out by stations, which periodically determine the error between a local range measurement to a satellite, and the range that has been computed from the accurately known positions of satellite and station.

The computed correction can be used for improving position accuracy of all receivers located in the station's region. The correction can be transmitted directly to the receiver by sending it over radio (which requires that the receiver is able to receive on an additional channel), or it can be received alternatively via an Internet connection.

Recently there has been a development of Satellite Based Augmentation Systems (SBAS) that transmit the corrections on the same frequency as the positioning satellites, having the effect that no additional receiver hardware is required. Correction data and positioning data are distinguished from each other by the Code Division Multiplex Access (CDMA) procedure. The European Geostationary Navigation Overlay Service (EGNOS), which mainly covers areas in Europe, consists of three geostationary satellites and a network of ground stations. EGNOS increases positioning accuracy of *GPS*, *GLONASS*, and *Galileo* from 10 - 20 meters down to 1 - 3 meters. Comparable systems, which are compatible with EGNOS, are in the United States the Wide Area Augmentation System (WAAS), and in Japan the MTSAT Space-based Augmentation System (MSAT).

Experiments presented in this work have been carried out with the *GPSlim236* GPS receiver from *Holux*, which is equipped with *SIRFstar III* technology. The receiver allows to track up to 20 satellites at an update rate of 1 Hz and has a position accuracy of 5 - 25 meters without DGPS. Furthermore, the receiver is able to process data from the *EGNOS* system, improving the horizontal position accuracy to <2.2 meters and vertical position accuracy to <5 meters at 95 % of the time.

Universal Transverse Mercator (UTM) conversion

GPS positions are denoted by ellipsoidal coordinates (latitude, longitude) based on the World Geodetic System (*WGS-84*). In order to utilize GPS data together with data generated from other sensors, it is necessary to transform the ellipsoidal coordinates to plane coordinates (x, y) (easting, northing). For this purpose there exist numerous conversion methods from geodetic applications, such as the Gauß-Krüger projection and the Universal Transverse Mercator (UTM) [Snyder, 1987] projection. For this thesis the UTM projection has been utilized due to its international use, e.g. by rescue organizations and the USGS (U.S. Geological Survey). The UTM projection separates the world between 80° S latitude and 84° N latitude into 60 vertical zones of a width

Figure 2.5: Map of Europe showing the latitude and longitude zones of the Universal Transverse Mercator (UTM) projection from 29 S to 38 W. Courtesy of the Demis company.

of 6°longitude (800 km), where each zone is centered over a meridian of longitude. Starting from the date line, zones are numbered from 1 to 60 increasing in an easterly direction. Furthermore, each vertical zone is separated into 20 latitude zones of a height of 8°, which are lettered starting from "C" at 80° S, increasing up until "X". Figure 2.5 shows the corresponding zones for Europe.

Within each Zone, locations are addressed by an easting and northing coordinate pair (x, y), with the point of origin at the intersection of the equator with the central meridian of the zone. In order to avoid negative easting numbers, an offset of 500,000 meters is added to each easting. Hence, locations east of the zone's central meridian have an easting value above 500,000 meters, whereas locations western of it have a value below 500,000 meters.

A similar procedure is applied to northing numbers. Northings in the northern hemisphere increase as going northward from the equator, which has an initial northing value of 0 meters, whereas in the southern hemisphere, northings decrease as going southward from the equator, which has then an initial value of 10,000,000 meters in order to avoid negative numbers. Therefore, it theoretically suffices to describe a zone by its vertical zone number and an indication whether the zone is located in the northern or in the southern hemisphere. For example, my office at the Computer Science Department at the University of Freiburg is located at the geographic position 48° 0'49.03" N, 7°50'1.96" E (in decimal 48.013621, 7.83388), which corresponds to the UTM zone 32U with an easting value of 413036 meters and a northing value of 5318471 meters. Besides the described calculation, the projection handles certain areas differently, as for example the areas around the poles, and also minimizes distortion by a scaling factor.

The interested reader might find further information on map conversions in the document published by J. S. Snyder [Snyder, 1987].

2.3 Sensors for detecting humans

2.3.1 Video and infra-red cameras

Vision sensors are essential for tele-operation since they are required for navigation as well as for gathering information on victims, such as identifying their state and the way they are buried. Moreover, they can be utilized for autonomous victim detection. This is conducted by applying image processing techniques on data generated from video and infra-red cameras. However, in the context of emergency response this is a challenging problem since only little assumptions can be made regarding color and shape of human body parts. Infra-red cameras, which are capable of detecting body heat, might misinterpret other heat sources, such as radiators, engines, and fire.

(a) (b) (c)

Figure 2.6: The CCD cameras (a) Sony DFW-500, and (b) Logitech QuickCam 4000 Pro. (c) The infra-red camera Thermal-Eye 3600AS.

Figure 2.6 shows camera models that have been utilized for victim identification in this thesis. The *Sony DFW-V500* camera incorporates a 1/3 type Progressive Scan Wfine CCD and an IEEE-1394 (Fire Wire) interface. Images are captured with 30 frames/sec at a maximal resolution of 640×480 pixels in the YUV 4:2:2 format. The camera has a weight of 305 g and requires an external power source. It offers high-quality images that are adequate for applying image processing techniques.

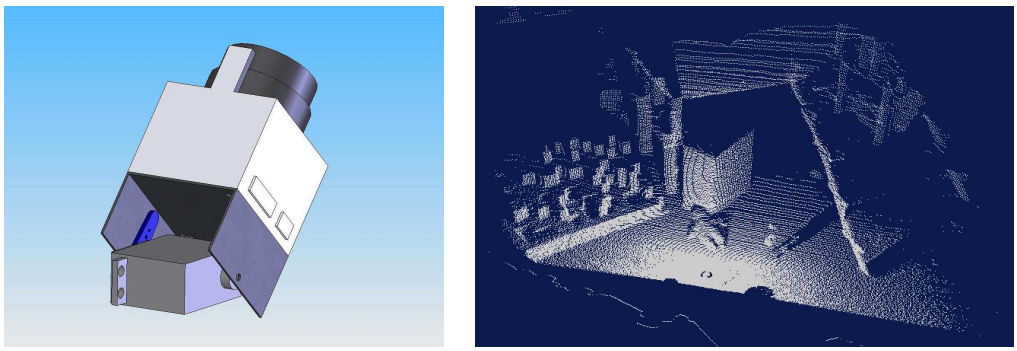
The *Logitech QuickCam 4000 Pro* also captures video images at 640×480 pixels, and can be connected via USB to a computer. This camera has been used for tele-operation and visual odometry (see Section 3.4). Due to its lightweight of less than 50 g (without housing) it can be mounted on a robot easily. Furthermore, the original lens of the camera has been replaced with a wide-angle lens yielding a FOV of 74°. However, the quality of the images turned out to be non-adequate for victim detection.

The *ThermalEye 3600AS* Infra-Red (IR) camera allows to detect human beings by their temperature within a Field Of View (FOV) of $\approx 50^\circ \times 37^\circ$, and a distance of up to 100 meters with a specified operating Temperature of $-20^\circ C$ to $85^\circ C$. The camera captures images with 30 frames/sec at a maximal resolution of 160×120 pixels, which are provided by an analog PAL signal that has to be digitized with additional hardware. Due to the modest power consumption of only 1.2 W and a weight of only 67.5 g, the camera can be mounted on nearly every robot platform. Each pixel of the camera image contains a gray value which is proportional to the measured heat at the real-world location corresponding to the pixel.

Before camera images can be used for victim detection, the camera has to be calibrated with respect to its intrinsic parameters, such as the focal length and the radial lens distortion. On color cameras these parameters are usually found by applying an analytical method [Tsai, 1986] that utilizes pixel to real-world correspondences as input. These correspondences are determined from pictures taken of a test pattern, such as the printout of a chess board [Bradski, 2000], with known feature distances. Then, relative displacements of features detected in the images, and those displacements measured on the test pattern, are brought into relation for computing a conversion function. In case of IR camera calibration, it is necessary to generate a test pattern that also appears on thermo images. This can be achieved by taking images from a heat reflecting metal plate covered with quadratic isolation patches in a chess board-like manner. The resulting images can then be processed by the calibration procedure in the same way as if taken from a color camera. Finally, the calibration procedure allows to determine the real-world distance and angle of features detected in the image. However, accurate depth information can only be obtained by stereo camera systems, since depth computation from monocular cameras requires that observed objects are located at the same hight.

2.3.2 3D Laser Range Finder (LRF)

In Rescue Robotics, the task-relevant environment is three-dimensional, and thus requires highly developed sensors for autonomous operation. For example, the detection of structures relevant for navigation, such as ramps, stairs, and stepfields, makes perception of three-dimensional objects indispensable. Also for the task of victim detection image processing techniques are not always sufficient on their own. Accurate depth measurements are needed in order to correctly group vision features according to their spacial neighborhood, and to reliably distinguish between background and foreground. Therefore, we developed a light-weight 3D Laser Range Finder (LRF) device for structure detection and mapping in the rescue domain. A picture of the sensor, and a 3D scan taken by this device can be seen in Figure 2.7. The LRF sensor is rotated by a strong servo that allows fast and accurate vertical positioning of the device. The device can be rotated vertically by more than 90 degrees, which is sufficient to generate 3D



(a) (b)

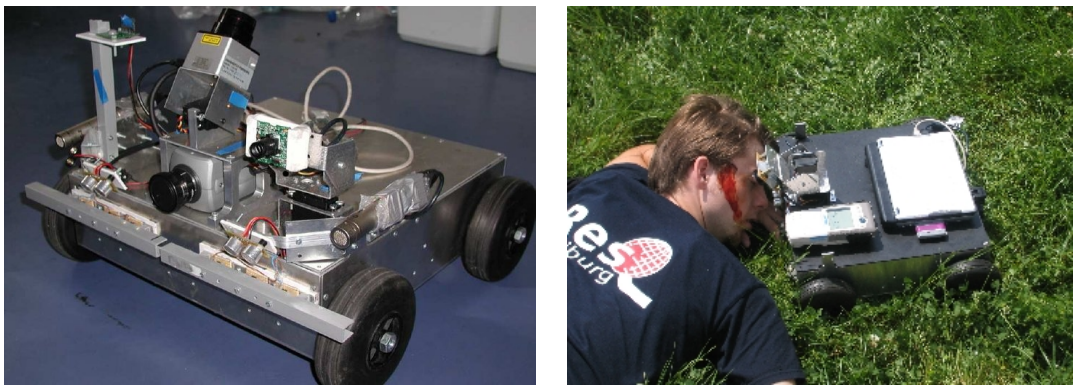
Figure 2.7: A hand-crafted light-weight 3D Laser Range Finder (LRF) device. (a) A model of the 3D LRF. (b) 3D scan taken in front of stepfield and baby doll.

scans from humans, and objects, such as stairs, ramps, and stepfields. The design differs from other 3D scanners in that it can be implemented by a simple "U-shaped" piece of metal and a servo, which altogether can be produced at approximately 100 USD (see Section 10.2.1).

2.3.3 Audio and CO_2 sensors

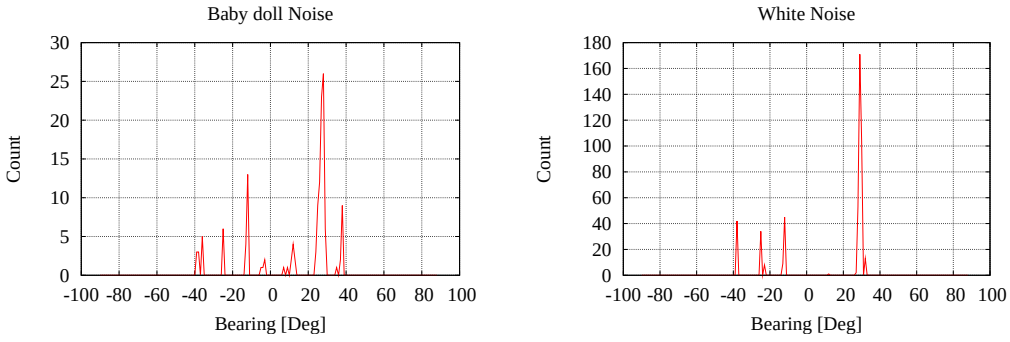
The locations of survivors after a disaster can be determined by audio through utilizing two microphones mounted with a fixed distance between them on the robot platform (see Figure 2.8 (a)). Given an audio source left, right or between both microphones, it is possible to measure the time difference, i.e. phase shift, between both signals. Kenn et al. use differential time of flight measurements through energy cross-spectrum evaluation of the sound signals for detecting the angular direction to multiple sound sources [Kenn and Pfeil, 2006]. They integrate all measurements into an occupancy grid map for localizing the victims in a global map. We utilized the Crosspower Spectrum Phase (CSP) approach, which allows to calculate the phase shift of both signals based on the Fourier transformation [Giuliani et al., 1994, Bennewitz et al., 2005, Omologo and Svaizer, 1996, Svaizer et al., 1997]. As shown in Figure 2.9, the bearing of the sound source can be successfully determined, even for different kinds of noise. To locate victims under harsh environmental conditions, e.g. in a noisy environment, is still challenging. However, the detection of audio signals close to victim locations provides a good indication of the victim's state, e.g. it might indicate that the victim is still alive and thus that a rescue mission is worth the effort.

Gases, such as CO_2, can be detected by the use of IR measuring after the NDIR (Non Dispersive Infra-Red) principle. Within a small tube, gases are absorbing infra-red light at a particular frequency. By shining an infra-red beam through the tube contain-



(a) (b)

Figure 2.8: The *Zerg* robot (a) equipped with two microphones for victim detection, and (b) measuring with a *Telaire 7001* sensor CO_2 emission of a victim during a TV recording in Freiburg.



(a) (b)

Figure 2.9: Examples for sound source detection by the Crosspower Spectrum Phase (CSP) approach. Both sound sources were located at a bearing of +30 degree. (a) Sporadic noise from a baby doll, and (b) continuous white noise are mainly detected at the right bearing.

ing CO_2, and measuring the amount of infra-red absorbed by the gas at the necessary wavelength, a NDIR detector is able to measure the volumetric concentration of the gas. Different kinds of wave lengths from IR light can be used for the analysis of different gases. The gas is either pumped or diffused into a tube, and the electronics measures the absorption of the characteristic wavelength of light. One drawback of this measurement principle is its slow response time for detecting a change of the CO_2 concentration. Therefore, robot navigation has to be interrupted for gaining reliable measurements.

We utilized the *Telaire 7001* NDIR carbon dioxide sensor, which measures CO_2 con-

centrations in the range of 0 to 4000 ppm at an accuracy of 50 ppm, and response time of < 60 seconds for 90% of the step change (see Figure 2.8 (b)).

The application of CO_2 concentration measurements in the context of urban search and rescue has been studied [Takashi and Hiroshi, 2003, Nguyen et al., 2004] intensively. Measurements of the CO_2 concentration can be utilized for survivor search. However, due to the volatility of CO_2 gas, particularly if the concentration is very low, survivor localization is possible under special conditions only. Gas sensors are more promising for determining the victim's state, and for the detection of hazardous locations, e.g. by sensing flammable and poisonous gases that can harm first responders during their missions.

2.4 RFID technology

RFID stands for Radio Frequency Identification, a term that describes small devices (RFID tags) that use radio signals to exchange data with a reader. The RFID tag transmits typically a number that is worldwide unique, or at least unique within in the context of the application. This has the effect that an object, to which the tag is attached to, can uniquely be identified from a database. In the usual context, RFID tags identify parcels in a post office or products sold in department stores. For example, since 2005 *Wal-Mart* requires from its top 100 suppliers to ship pallets labeled with RFID tags to its stores. The *Metro* group started the "Future Store Initiative", a stepwise application of RFID technology, ranging from automated stock maintenance in stores, to intelligent refrigerators in consumer households that monitor the expiration date of food based on data read from tags attached to the food items. In the long run, their goal is to completely replace the current bar code system, found on products sold today, by RFID technology. Furthermore, the size of RFID chips shrinks continuously. For example, *Hitachi* introduced 2003 the *μ-Chip* with a size of 0.4×0.4 mm, where the generation introduced in 2006 reaches a size of 0.15×0.15 mm. Their latest version, released in 2007, has a size of 0.05×0.05 mm. The progressing miniaturization of RFID technology has also the effect of reducing manufacturing costs since considerably more chips can be produced from a single wafer. The small size of RFIDs makes it also possible to attach them to bank notes [Lange, 2005], or to integrate them into carpets for simplifying indoor localization of cleaning robots. The German vacuum-cleaner company *Vorwerk* introduced in 2006 the "Smart Floor", a network of RFID chips embedded into a polyester fabric that can be placed under carpets [Co. KG, 2005]. The recent advent of the RFID technology leads to a steady increase of smart devices in our environment that provide digital data. Algorithms and methods applied in this context are generally recognized under the terms "Sensor networks", "Ubiquitous Computing", and "The Internet of things".

RFID tags can be classified as *active* and *passive* devices depending on whether they

are powered directly from an attached battery, or whether they are powered from an electromagnetic field emitted by the reading device. This section focuses on passive devices, which are produced cheaply and in large numbers. They consist mainly of an antenna, memory, and a sophisticated state machine (or even a microprocessor). The latter is required if tags handle encrypted communication or implement an anti-collision protocol. Anti-collision protocols are necessary if the application requires multiple tags to be read at the same time. RFID tags with anti-collision protocols wait for their turn when responding to a reader. In contrast to the "1-bit" Electronic Article Surveillance (EAS) tags, which are commonly found in libraries or stores for theft prevention, these advanced RFID tags are equipped with a read- and writable memory, which can be used to store additional information on the object the tag is attached to. It can be assumed that due to the continuously decreasing price of memory chips, the capacity of RFID tags will further increase in the future. The company MicroSensys offers passive RFID tags with up to 512 KBit memory, and develops currently RFIDs with memory capacities of up to 4 MBit [MicroSensys, 2007].

RFID tags are available for various communication frequencies within the Industrial Scientific Medical (ISM) band, as for example, at High Frequency (HF) 6.78 MHz, 13.56 MHz and 27.125 MHz, at Ultra High Frequency (UHF) 433.920 MHz, 869 MHz, 915 MHz, and at Microwave 2.45 GHz, 5.8 GHz, and 24.125 GHz. The choice of the communication frequency is of major importance for the application since the maximal communication range and communication bandwidth depends on it. The higher the frequency, the higher the communication bandwidth and communication range. Furthermore, the maximal range depends on the environment, e.g. the kind of material located between receiver and tag. For example, metal or water reduces the range significantly. Short-wave frequencies, as for example 13,56 MHz, lead typically to distances below three meters and thus are only applicable to objects within short range. Table 2.2 summarizes some RFID frequencies and applications they are utilized for.

Frequency	**Max. reading range**	**Typical applications**
LF (e.g. 100 KHz)	0.5 m	Pet identification and close reads of items with high water content
HF (e.g. 13.56 MHz)	3 m	Building access control, Product identification
UHF (e.g. 915 MHz)	9 m	Boxes and pallets
Microwave (e.g. 2.4 GHz)	>10 m	Vehicle identification of all sorts

Table 2.2: Communication frequencies and corresponding maximal reading ranges of RFID tags and their typical applications (original data from [Bhatt and Glover, 2006]).

2.4.1 RFIDs in robotic applications

For robotic applications, particularly in the context of localization and mapping, it is important to compute an estimate on the distance between the RFID tag and the reader. The Transceiver-Receiver (TR) separation, i.e. the distance between a detected RFID and the detector, can generally be estimated from the power of the signal. However, signal propagation in an indoor environment is perturbed by damping and reflections of radio waves. Since these perturbations depend on the layout of the building, the construction material used, and the number and type of objects in the building, modeling the relation between signal path attenuation and TR separation is a challenging problem.

Seidel and Rapport introduced a model for path attenuation prediction that can also be parameterized for different building types and the number of floors between transceiver and receiver [Seidel and Rapport, 1992]. This model has been evaluated for frequencies in the UHF domain, e.g. 914 MHz. RFID implementations operating in this domain are requiring a line of sight between the tag and the detector. This allows to adopt a simpler version of the model from Seidel and Rapport, based on the assumption that RFID detections are not possible through walls [Ziparo et al., 2007a]. The model relates the signal power P to distance d in the following way:

$$P(d)[dBm] = p(d_0)[dBm] - 10n \log \frac{d}{d_0}, \tag{2.1}$$

where $P(d_0)$ is the signal power at reference distance d_0 and n denotes the mean path loss exponent that depends on the structure of the environment. Seidel and Rapport determined for transmissions at 914 MHz a path loss of 31.7 dB at a reference distance of 1 meter. Furthermore, they determined for different building types characteristic values for n and the standard deviation σ of the signal.

TR has also been evaluated in order to improve the security of RFID data transmission [Hancke and Kuhn, 2005]. RFID tags are vulnerable to relay attacks if they are used for proximity authentication. Therefore, it is important to read preferably from tags that are close to the reader, which requires the reader hardware to be aware of the distance to the sender. Another application is to localize objects with attached RFIDs, such as books in a library. The basic idea is to build a system that guides the user's search by providing feedback on the distance to the searched object [Fishkin and Roy, 2003]. More details on RFID technology are found in [Finkenzeller, 2003].

2.4.2 Importance for Urban Search and Rescue (USAR)

RFID technology offers a great opportunity for disaster mitigation and USAR. The key problem in this scenario is to get an overview of the disaster, e.g. to be aware of the locations of first responders while generating a map augmented with disaster-relevant information, such as victim locations, hazardous areas, and passable openings. How-

ever, standard localization techniques, such as vision-based structure recognition, and GPS, are not well suited in these scenarios due to harsh environmental conditions. For example, smoke and dust particles lead to a limited field of view, and GPS fails completely within structures made of reinforced concrete. RFID technology can be utilized for labeling and re-observing locations, as well as for unambiguously communicating them to other team members. RFIDs can already be located at the disaster site, e.g. they may be attached to consumer products, or intentionally integrated into building structures. Alternatively, they can be deployed by rescue teams [Miller et al., 2006, Kleiner et al., 2006c], either by manually fixing them to relevant locations, or by automatically deploying them in masses from aerial vehicles. Additionally, RFID tags can be resistant under extremely hostile conditions. For example, Surface Acoustic Wave (SAW) technology allows to produce RFID transponders that are known to operate stable up to several hundreds of degrees, even up to $1000°C$ [Fachberger et al., 2006]. They can be attached to metal surfaces, and are operational within a range of 10 meters for identification, distance measurements, and temperature measurements (up to $400°C$) [Reindl et al., 2001, Research, 2007]. Hence, they can be deployed even under harsh conditions, e.g. in buildings set on fire.

Furthermore, the memory of RFIDs can be utilized by rescue teams for leaving behind information, such as nearby locations of victims, and nearby explored locations, supporting the search of other rescue teams in the field. The concept of labeling locations with information crucial for the rescue task has been already applied in real disaster response situations. During the disaster relief in New Orleans in 2005, rescue task forces marked buildings with information concerning, for example, hazardous materials or victims inside the buildings. A description of these kinds of markings can be found in a document published by the U.S Dep. of Homeland Security [FEMA, 2003].

2.5 Rescue Robots Freiburg

The work proposed in this thesis was extensively tested on two different robot platforms, which are a 4WD (four wheel drive) differentially steered robot for the autonomous team exploration of large office-like areas, and a tracked robot for mapping and overcoming three-dimensional obstacles, such as stairs and ramps. Both robot platforms were custom made at the University of Freiburg and designed under the following criteria (according to Section 2.1):

- Capability of autonomous operation.
- Lightweight design in order to gain a maximal degree of mobility.
- Manufacturing costs.
- Heterogeneous team of robots with different capabilities.

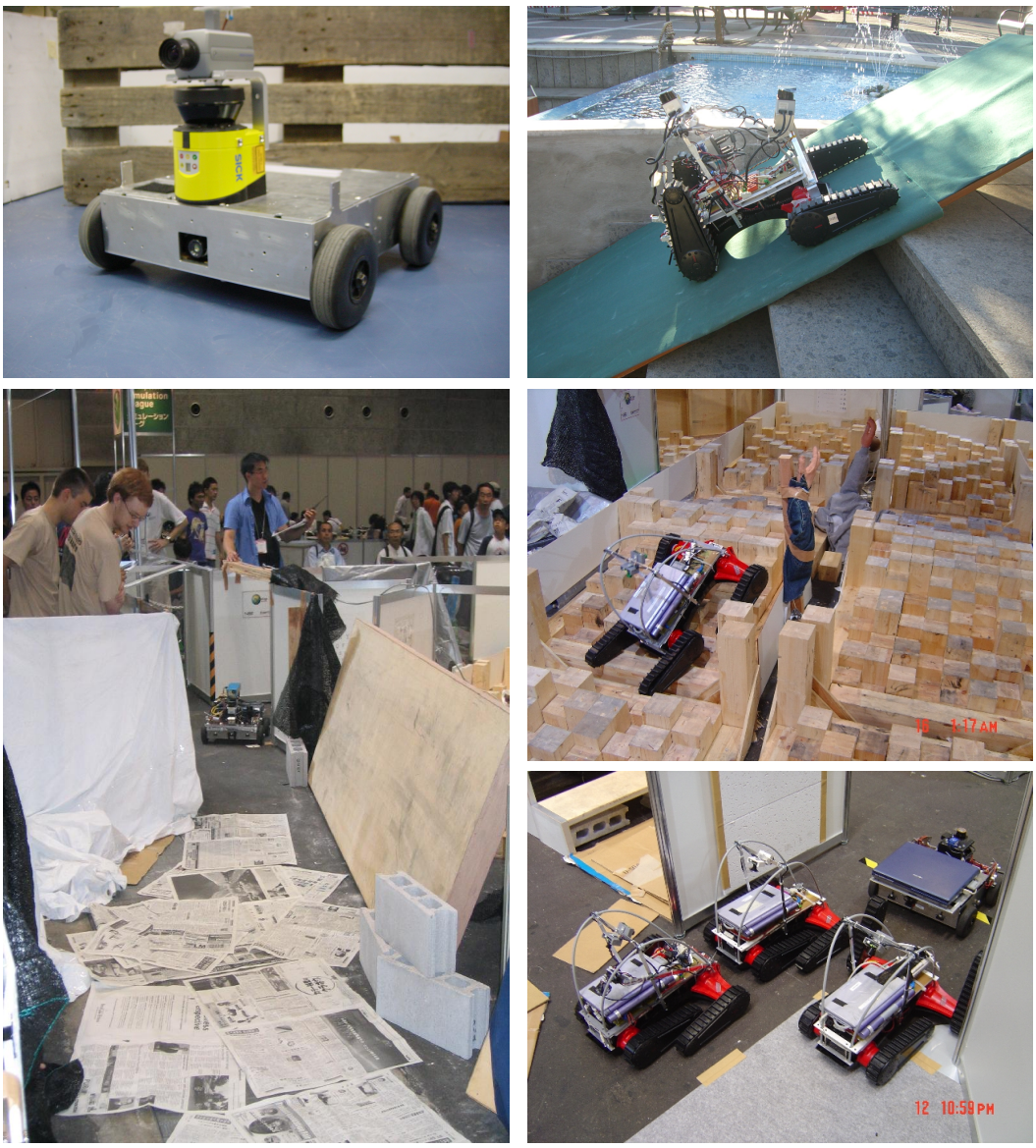


Figure 2.10: Robots of the team *Rescue Robots Freiburg*. Left column, top to bottom: The *Zerg* robot equipped with a *Sick S300* LRF, and navigating in the *yellow arena* during the *Autonomy Competition* at *RoboCupRescue* 2005 in Osaka. Right column, top to bottom: The *Lurker* robot climbing up a ramp during the *RescueCamp* 2006 in Rome, and searching for victims in the *red arena* at *RoboCupRescue* 2005. A team of robots waiting for mission start during *RoboCupRescue* 2005.

Both robot platforms process data from sensors with a *CardS12* module, which consists of a *MC9S12D64* Micro Controller Unit (MCU) with a 16-bit HCS12 CPU, 64 KB of

flash memory, 4 KB RAM, and 1 KB EEPROM. The module offers a large amount of peripheral connectivity, such as Analog-Digital Converters (ADCs), Pulse Width Modulation (PWM) ports, and digital I/Os. Furthermore, each robot carries a lightweight laptop which connects to the MCU via RS-232. The laptop processes the sensor data further and transmits it, if required, via wireless LAN to an operator station. Details of the mechanical design and schematics of the electronic components of both robots are found in the Appendix (Chapter 10).

2.5.1 *Lurker* robot

Figure 2.10 (right column) shows the tracked *Lurker* robot, which is based on the *Tarantula* R/C toy. Although based on a toy, this robot is capable of climbing difficult obstacles such as stairs, ramps, and random stepfields. This is possible due to its tracks, which can operate independently on each side, and the "Flippers" (i.e. the four arms of the robot), which can be freely rotated at 360°. The robot is controlled by relays one for each track (left-hand and right-hand side), and one for each flipper rotation (front and rear axis), which are connected with four digital outputs of the MCU. The base was heavily modified in order to enable autonomous operation. First, we added a 360° freely turnable potentiometer to each of the two axes for measuring the angular position of the flippers. Second, we added ten touch sensors to each flipper, allowing the robot to measure force when touching the ground or an object. The data from the ten touch sensors is multiplexed onto four wires that are bridged with sliding contacts at the rotating joints. The touch sensors, although simply constructed by push buttons, provide valuable feedback if the robot touches the ground or an obstacle. This information together with the angles of the flippers serves as a basis for autonomous skills, and also supports tele-operation if the information is visualized on the operator interface. Furthermore, the robot is equipped with a 3-DOF Inertial Measurement Unit (IMU) from *Xsens*, allowing drift-free measurements of the three Euler angles *yaw*, *roll*, and *pitch*, and two *Hokuyo URG-X004* Laser Range Finders (LRFs), one for scan matching (shown in Section 3.2), and one for elevation mapping (shown in Section 5.3), where the latter can be tilted in the pitch angle within 90°. For feature tracking, which will be elaborated on in Section 3.4, a *Logitech QuickCam Pro 4000* web cam [Logitech, 2006] has been utilized. The *Tarantula* toy later on has also been used by other groups for education and research [Sheh, 2006].

2.5.2 *Zerg* robot

Figure 2.10 (left column) shows the *Zerg* robot, a 4WD differentially steered platform, which was completely hand-crafted. The 4WD drive provides more power to the robot and therefore allows to drive up ramps and to operate on rough terrain (as far as possible with the utilized wheel-base). Each wheel is driven by a *Pitman GM9434K332* motor

with a *19.7:1* gear ratio and a shaft encoder. The redundancy given by four encoders allows to detect heavy slippage and situations in which the robot gets stuck, as will be shown in Section 3.3. In order to reduce the large odometry error that naturally arises from a four-wheeled platform, we also utilized an Inertial Measurement Unit (IMU) from *Xsens*. Moreover, the robot is equipped with a *Thermal-Eye* infrared thermo camera for victim detection. Localization and mapping is performed by a *Hokuyo URG-X004* LRF (indoor version), or a *Sick S300* (outdoor version).

2.5.3 Human-Robot Interaction (HRI)

One important goal of Rescue Robotics is to provide technology for assisting the tasks of humans during urban search and rescue. Human-Robot Interaction (HRI) is an active research field dealing with the problem of increasing the efficiency of controlling partially autonomous robot teams, while decreasing the cognitive load of the operator. Wang et al. demonstrated empirically the advantage of mixed initiative teams, e.g. partially autonomous robots controlled by a human operator [Wang and Lewis, 2007]. Caspar and Murphy studied human robot-interactions during The World Trade Center rescue response [Casper and Murphy, 2003]. They made eleven recommendations for the rescue field based on their findings from robotics, computer science, engineering, and psychology. Murphy provides an overview on how robots are currently used in urban search and rescue and discusses the HRI issues encountered over the past eight years [Murphy, 2004].

Inspired from HRI research recommendations, we developed two applications for allowing a single operator to control a heterogeneous team of *Zerg* and *Lurker* robots. We specifically designed a Graphical User Interface (GUI), which can be used to fully control any robot of the team (see Figure 2.11 (a)). The GUI is realized by a similar approach as proposed by the RoBrno team at RoboCup 2003 [Zalud, 2004]. Images from video cameras are shown in full size on the screen, and additional information is overplayed via a Head Up Display (HUD) in order to focus the operator's attention on crucial information. The kind of additional information that is displayed, and its transparency (alpha value), can be adjusted by keyboard commands. Robot control is carried out with a joypad that is connected to a portable Laptop. Besides images from the thermo camera and video camera mounted on the robot, the operator receives readings from other sensors, such as range readings from the LRF, compass measurements, and the battery state of the robot. Data from the LRF is used to simplify the operator's task of navigation, i.e. to estimate the distance to surrounding obstacles. The GUI facilitates to dynamically switch control between multiple robots that are actively sending data from the field at the same time.

Furthermore, we developed the strategic tool *IncidentCommander* for issuing high-level commands, e.g. to assign robots to target positions on the map, or to assign complex behaviors, such as stair climbing tasks, which then are autonomously executed by

(a) (b)

Figure 2.11: Human Robot Interaction (HRI): (a) The *RoboGui*, a graphical user interface for controlling and monitoring robots, visualizing images from IR and video camera, and readings from the LRF and IMU. (b) Joypad for operator control.

the commanded unit (see Figure 2.12). The *IncidentCommander* receives local maps

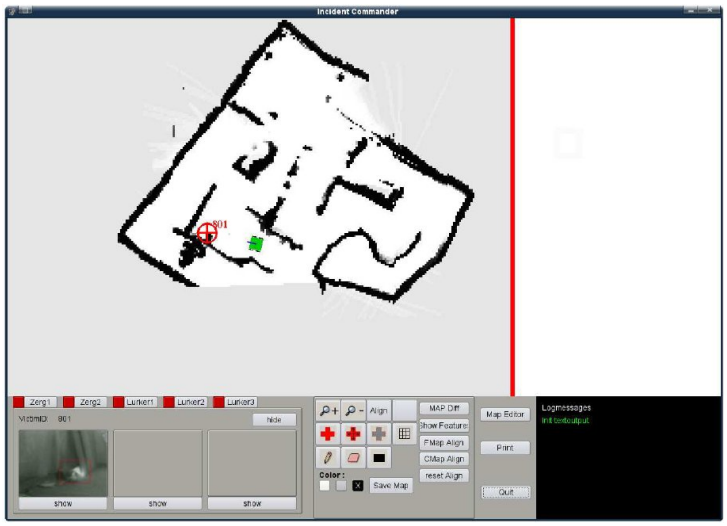


Figure 2.12: The *IncidentCommander*, a user interface for map-merging and assigning high-level commands to multiple robots, visualizing the map generated by the robots, and images from detected victims.

generated by the robots and allows to merge them into a global map, either manually or automatically, based on detected features, such as corners and corridors. Victims that are detected by the robots are automatically reported to the *IncidentCommander* by sending the victim's location on the map, as well as the according video and IR images taken at the victim's location. In Figure 2.12 the victim appears as a red cross augmented with a number on the map. If the victim is selected with the input device, all relevant images are shown. This allows for the operator to verify the victim detection visually, and to erase the marking on the map if the detection was a *false-positive*. Furthermore, the map can be modified manually if errors are detected.

3 Pose Tracking in Disaster Areas

3.1 Introduction

To incrementally build a map from scratch and to localize within this map requires good estimates of local pose displacements during locomotion. Pose tracking is the process of continuously tracking the pose of a moving object based on sensor readings. For each discrete time interval Δt, the pose tracker estimates from the last n sensor observations $o_t, o_{t-1}, o_{t-2}, \cdots, o_{t-n}$ the local displacement denoted by distance $d_{\Delta t}$ and angle $\theta_{\Delta t}$.

Pose displacements can be detected by an odometry sensor, e.g. measured by shaft encoders mounted on the robot's wheels, by a vision system, e.g measuring optical flow, or by scan matching algorithms applied to LRF (Laser Range Finder) measurements. However, wheel odometry becomes arbitrarily inaccurate if the robot has to drive on slippery ground or even has to climb over obstacles. Particularly on skid-steered platforms, e.g. tracked vehicles, odometry leads to large measurement errors in rotation. Vision-based systems depend on a constant flow of features, such as dark corner points, by which detection might be affected due to harsh visibility conditions, as for example smoke, dust, and fire. Scan matching techniques can only be applied reliably if the scanner continuously captures features, such as lines from corners and walls. They require polygonal structures as typically found in office-like environments [Gutmann and Schlegel, 1996, Grisetti et al., 2002], which are not necessarily present in unstructured environments.

As it is important for robots to track their pose for mapping and localization in rescue scenarios, it is even more important for human beings, e.g. first responders, to know their location in order to find and rescue victims. Pedestrian Dead Reckoning (PDR) stands for methods that estimate incrementally the position of a walking person, as for example in the context of Location Based Services (LBS) [Lechner et al., 2001], navigation for the Blind [Ladetto and Merminod, 2002b], and emergency responder tracking. PDR is typically utilized in urban areas where Global Navigation Satellite System (GNSS) fails due to signal damping or reflection caused by building structures. In urban environments, GNSS positioning is affected by the so-called multipath propagation problem [Grewal et al., 2001]. Buildings in the vicinity of the receiver can easily reflect GNSS signals, resulting in secondary path propagations with longer propagation time, which distort the amplitude and phase of the primary signal.

In this chapter, approaches for solving the pose tracking problem on robots and on

humans are introduced. They serve as a basis for RFID technology-based SLAM, which will be described in Chapter 4. Pose tracking on humans is computed from acceleration patterns measured by body-worn sensors. We adopted a method from Ladetto and colleagues [Ladetto et al., 2000] that recognizes human footsteps analytically from acceleration patterns. For this purpose, we utilized an Inertial Measurement Unit (IMU) that includes tri-axial accelerometers, gyroscopes, and magnetometers [*]. Utilizing IMUs has the advantage that the small and lightweight sensors can be attached to humans in an ergonomic way. They might be integrated into a helmet [Beauregard, 2006], a belt [Amft et al., 2004], firefighter jacket [Kleiner et al., 2006a], or even into shoes [Foxlin, 2005].

For pose tracking on robots, two novel approaches, one for wheeled robots and one for tracked robots, are contributed. Since wheel odometry becomes arbitrarily inaccurate if robots navigate on slippery ground or have to overcome smaller obstacles, a method for slippage-sensitive odometry has been developed. The introduced method, which is designed for 4WD robot platforms with over-constrained odometry, infers slippage of the wheels from differences in the measured wheel velocities. Inference is carried out by a decision tree that has been trained from labeled odometry data. Furthermore, we solve the problem on tracked robots by utilizing a consumer-quality camera for measuring translations, and an IMU for measuring rotations. The proposed method tracks salient features with the KLT feature tracker [Tomasi and Kanade, 1991] over images taken by the camera, and computes from the tracked features the translation of the robot. Translation tracking is computed from a voting of single feature trackings that are classified by a *tile coding* classificator [Sutton and Barto, 1998]. The method is tailored for a simplified motion model of the robot, requiring a set of discrete velocity commands.

All methods were extensively evaluated in outdoor environments, as well as in USAR test arenas designed by the National Institute of Standards and Technology (NIST) [Jacoff et al., 2001]. Our results show that the proposed methods perform robustly and efficiently in the utilized benchmark scenarios.

The remainder of this chapter is structured as follows. In Section 3.2 state-of-the-art techniques for pose tracking are described. Then, two solutions for rescue scenarios are contributed, which are dead reckoning with slippage sensitive odometry in Section 3.3, and pose tracking by visual odometry in Section 3.4. In Section 3.5 an existing method for PDR is discussed, and in Section 3.6 empirical results from indoor and outdoor experiments of the described approaches are presented. In Section 3.7 an overview on related work is given, and in Section 3.8 we conclude.

[*]Note that a magnetometer measures the earth magnetic field and hence offers the functionality of a digital compass.

3.2 Conventional techniques

3.2.1 Dead reckoning

Dead reckoning is the process of estimating a global position of a vehicle by continuously advancing a known position by incorporating bearing, speed, and traveled distance. For wheeled robots, dead reckoning is a method for determining the movement of the robot by measuring the revolution of the wheels. This is accomplished by shaft encoders, which are mounted directly on the wheel axes. They provide the number of counted ticks, which is proportional to the revolution of the wheel. Given the gear ratio n, the resolution of the decoder (ticks per revolution) R and the diameter of the wheels D, one can calculate a conversion factor c for the conversion between counted ticks and the linear wheel displacement:

$$c = \frac{\pi D}{nR}. \tag{3.1}$$

If N_r, N_l are the tick counts of the right and the left wheel, respectively, one can compute the right wheel displacement $d_r = cN_r$ and left wheel displacement $d_l = cN_l$. The overall traveled distance d and traveled angle α can be calculated from the wheel distances d_l and d_r after a simplified model from Borenstein [Borenstein et al., 1996]:

$$d = \frac{(d_r + d_l)}{2}, \quad \alpha = \frac{(d_r - d_l)}{B}, \tag{3.2}$$

where B denotes the distance between both wheels, also known as *wheel base*.

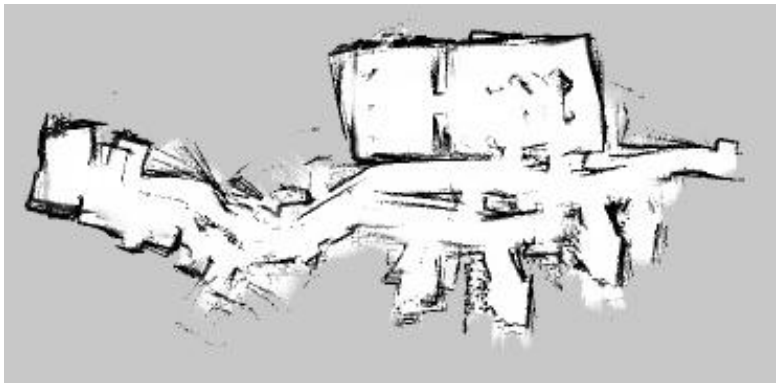


Figure 3.1: *Dead reckoning* on a skid-steering 4WD robot driving in a cellar: The resulting map is unusable for planning and navigation.

Dead reckoning is usually accompanied with measurement errors, for example caused by wheel slip during locomotion, which do accumulate over time. Figure 3.1 depicts the result from successively integrating range measurements from a laser range finder with

respect to the odometry pose of the robot. The resulting map is unusable for planning and navigation.

For any further processing of dead reckoning information it is essential to maintain a model of the accumulating errors. The traveled distance d and angle α is modeled by a Gaussian distribution $N(u, \Sigma_u)$, where $u = (d, \alpha)^T$ and Σ_u is a 2×2 covariance matrix expressing dead reckoning errors. Likewise uncertainty of the two-dimensional pose $l = (x, y, \theta)^T$ is modeled by a Gaussian distribution $N(\mu_l, \Sigma_l)$, where μ_l is the mean and Σ_l a 3×3 covariance matrix [Maybeck, 1990]. Given this representation, the pose at time t can be updated from input u_t as follows:

$$l_t = F(l_{t-1}, u_t) = \begin{pmatrix} x_{t-1} + \cos(\theta_{t-1})d_t \\ y_{t-1} + \sin(\theta_{t-1})d_t \\ \theta_{t-1} + \alpha_t \end{pmatrix}, \tag{3.3}$$

$$\Sigma_{l_t} = \nabla F_l \Sigma_{l_{t-1}} \nabla F_l^T + \nabla F_u \Sigma_u \nabla F_u^T, \tag{3.4}$$

where

$$\Sigma_u = \begin{pmatrix} d\sigma_d^2 & 0 \\ 0 & \alpha\sigma_\alpha^2 \end{pmatrix}, \tag{3.5}$$

$$\nabla F_l = \begin{pmatrix} 1 & 0 & -\sin(\theta_{t-1})d_t \\ 0 & 1 & \cos(\theta_{t-1})d_t \\ 0 & 0 & 1 \end{pmatrix}, \tag{3.6}$$

$$\nabla F_u = \begin{pmatrix} \cos(\theta_{t-1}) & 0 \\ \sin(\theta_{t-1}) & 0 \\ 0 & 1 \end{pmatrix}. \tag{3.7}$$

Note that the update shown in Equation 3.3 is an approximation and only applicable if the robot moves almost a straight-line within the time interval in which d and $alpha$ are measured. The result from incrementally applying the update on a skid-steering robot driving in a cellar is depicted in Figure 3.1. One method to improve pose estimates from wheel odometry, is to utilize an Inertial Measurement Unit (IMU) as described in Chapter 2. Particularly in case of skid-steered vehicles, such as a 4WD robot or tracked robot, the usage of an IMU is recommendable. Pose tracking from odometry data can be further improved by utilizing range measurements from the LRF by incremental scan matching.

3.2.2 Scan matching

The basic idea behind scan matching is to rotate and translate a scan in order to match a previously taken scan and thereby estimate the displacement $(\Delta x, \Delta y, \Delta\theta)$ of the robot. It has been shown that particularly in environments with vertical structures, such as

walls around an indoor soccer field, this can efficiently be performed [Gutmann et al., 2001]. Various methods were introduced by researches in the past which can generally be separated into *scan point-based* and *grid-based* methods. Among the scan point-based methods, Cox and colleges proposed a method particularly suited for polygonal environments [Cox, 1991]. Their method matches range readings with a priori given line mode. Lu and Milios proposed a method that can also be applied in non-polygonal environments [Lu and Milios, 1994]. Gutmann showed how to combine these two methods in order to improve their overall performance [Gutmann, 2000]. Grid-based methods have the advantage that they are able to filter-out erroneous scans by averaging them by an occupancy grid [Moravec and Elfes, 1985], however, their disadvantage is the high requirement of memory and computational resources. In this section, a grid-based scan matcher introduced by Hähnel [Hähnel, 2005] will be described in more detail since this method was implemented within the system described further on.

The technique determines from a sequence of previous scan observations $z_t, z_{t-1}, ..., z_{t-n}$ subsequently for each time point t an estimate of the robot's pose k_t. This is carried out by incrementally building a local grid map from the n most recent scans and estimating the new pose k_t of the robot by maximizing the likelihood of the scan alignment of the latest scan z_t in the current map. The robot pose $N(l_t, \Sigma_{l_t})$ is fused with the pose of the scan matcher $N(k_t, \Sigma_{k_t})$ by:

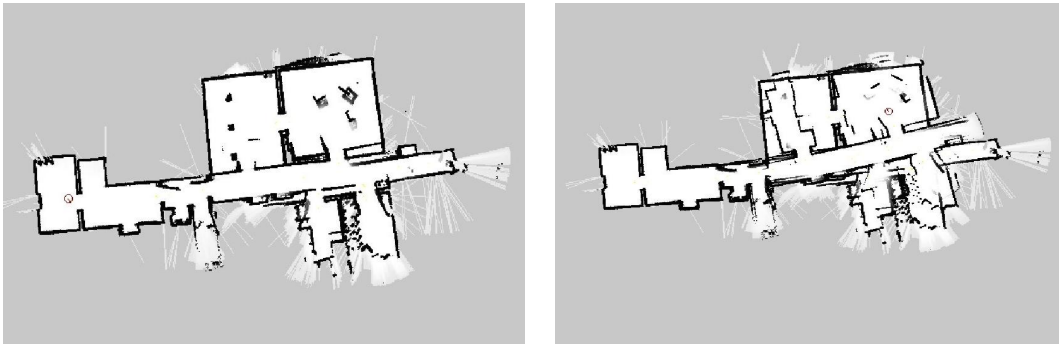
$$l_{t+1} = \left(\Sigma_{l_t}^{-1} + \Sigma_{k_t}^{-1}\right)^{-1} \left(\Sigma_{l_t}^{-1} l_t + \Sigma_{k_t}^{-1} k_t\right) \tag{3.8}$$

$$\Sigma_{l_{t+1}} = \left(\Sigma_{l_t}^{-1} + \Sigma_{k_t}^{-1}\right)^{-1} \tag{3.9}$$

The result from applying scan matching incrementally on laser range data collected by a robot driving in a cellar is depicted in Figure 3.2. Scan matching-based pose tracking leads to good results if the scanner captures sufficient features in the environment, such as walls and corners. However, if the LRF has a restricted field of view, e.g. a range limit of four meters as the LRF from *Hokuyo*, scanning leads to a large number of *far readings*. Far readings are measurements at maximum range leading to an insufficient amount of features. This problem can be solved by disabling scan matching and to continue pose tracking from odometry data only. However, to detect such situations is difficult since an increase of far readings does not reliably indicate the true amount of information provided by the scan.

3.3 Slippage sensitive wheel odometry

If the robot operates on varying ground, as for example concrete sporadically covered with newspapers, or if the robot gets stuck on obstacles, odometry errors are not linearly growing anymore, but are dependent on the particular situation. Therefore, we designed



(a) (b)

Figure 3.2: *Scan matching* with dead reckoning on a skid-steering 4WD robot driving in a cellar: (a) The first part of the resulting map is usable for planning and navigation. (b) As the robot proceeds, the map gets unusable due to constantly increasing positioning errors.

the *Zerg* robot with an over-constrained odometry for the detection of slippage of the wheels by utilizing four shaft-encoders, one for each wheel. From these four encoders, we recorded data while the robot was driving on varying ground, and labeled the data sets with the classes $C = (slippage, normal)$. This data was taken to learn a decision tree [Quinlan, 2003] with the inputs $I = (\Delta v_{Left}, \Delta v_{Right}, \Delta v_{Front}, \Delta v_{Rear})$, representing the velocity differences of the four wheels, respectively. For example, Δv_{Front} denotes the velocity difference of the two front wheels, and Δv_{Right} the velocity difference of the two wheels on the right hand side of the robot. Under normal operation, Δv_{Left} and Δv_{Right} should be close to zero, however under slippage, these differences increases. As depicted in Figure 3.3, the trained classifier reliably detects this slippage from the velocity differences.

Given the detection of slippage, the traveled distance d is computed from the minimum wheel velocity given by $v_t = \min\left(v_{LeftFront}, v_{RightFront}, v_{LeftRear}, v_{RightRear}\right)$, and the robot's pose is updated according to Equation 3.3, however, with $\sigma^2_{d_{slip}}$, within covariance matrix Σ_u, in order to increase uncertainty in translation. Note that the rotation update needs not to be modified since the traveled angle α is measured by the IMU, which is not affected by wheel slippage. The values for σ^2_d and $\sigma^2_{d_{slip}}$ have been determined experimentally. During extensive runs with slippage events, we recorded the true traveled distance, determined with scan matching, and the distance estimated by the odometry. The data set was labeled by the slippage detection and then was utilized for computing the Root Mean Square (RMS) error for determining the variances σ^2_d and $\sigma^2_{d_{slip}}$. We finally determined $\sigma_d = 0.816\frac{cm}{m}$ and $\sigma_{d_{slip}} = 24.72\frac{cm}{m}$. As will be shown in Section 3.6, the improved odometry reduces the error significantly, while maintaining appropriate covariance bounds.

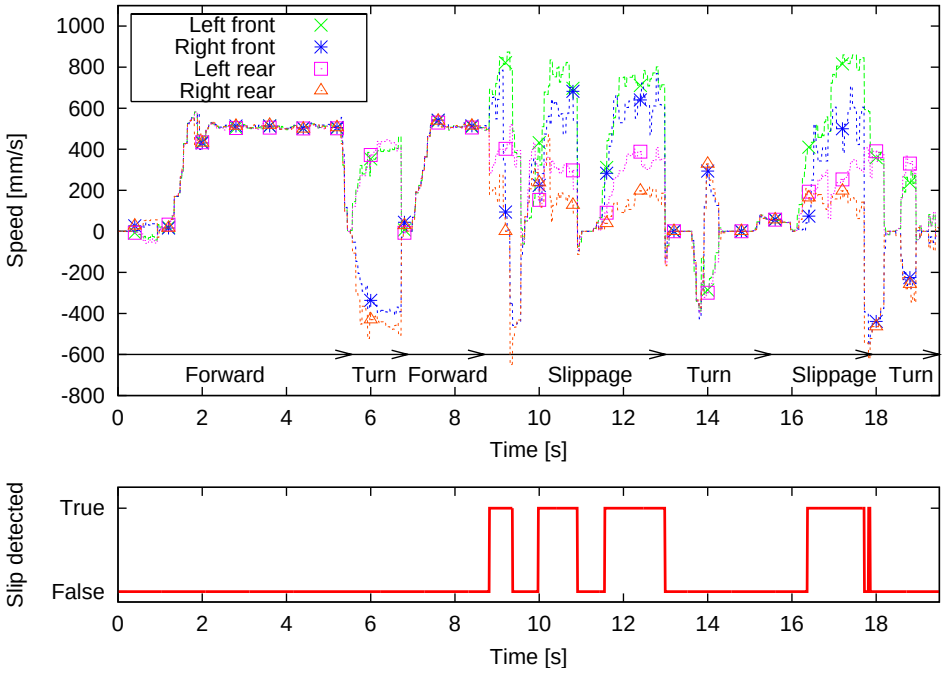


Figure 3.3: Slip detection on a 4WD robot: each line in the upper graph corresponds to the velocity measurement of one of the four wheels by shaft encoders. The black arrows indicate the true situation, e.g. driving forward, slippage, etc., and the red line in the lower graph depicts the automatic slip detection by the decision tree classifier, given the velocities as input.

3.4 Visual odometry based pose tracking

In this section a solution to pose tracking on tracked robots that can support SLAM during autonomous behaviors on three-dimensional obstacles is described. The introduced approach aims at an inexpensive, light-weight, and computational efficient implementation, as for example based on a single consumer-quality webcam. Since the robot is equipped with an IMU providing high accuracy orientation information, and the motion model on tracked robots can be simplified by restricting it to constant velocities, our goal is to distinguish reliably between *forward*, *backward*, and *none* translations, only. This limitation is motivated by the fact that computing the full metric information of the translation, e.g. to determine the scale, requires to extract depth information from camera observations, which typically can only be done by stereo vision. However, the problem can be relaxed if robots operate in planar environments [Weigel et al., 2002], or by utilizing metrical initialization points with known spacial displacement in the environment [Davison, 2003]. By focusing on determining the translation direction only,

we solve this problem for three-dimensional environments as the robot's known driving speed produces a reference that can be used to generate metrical pose information.

3.4.1 Feature tracking

Salient features are tracked over multiple images with the KLT feature tracker [Tomasi and Kanade, 1991], and utilized for calculating difference vectors that indicate the robot's motion. Since rotations are estimated by the IMU sensor, and the goal is to determine translations from the images, the camera is mounted to either the left or right side of the robot, yielding the effect that forward and backward motion results in horizontal tracking vectors pointing into the direction of the motion. The true translation of the robot is determined based on the individual voting of single translation vectors. Each vector votes for one of the possible translations according to a pre-trained *tile coding* classificator [Sutton and Barto, 1998].

In general, an image sequence can be described by a discrete valued function $I(x, y, t)$, where x, y describe the pixel position and t describes the time. It is assumed that features detected in an image also appear in the subsequent image, however translated by $d = (\xi, \eta)^T$, where ξ and η denote the translation in x and y:

$$I(x, y, t + \tau) = I(x - \xi, y - \eta, t) \tag{3.10}$$

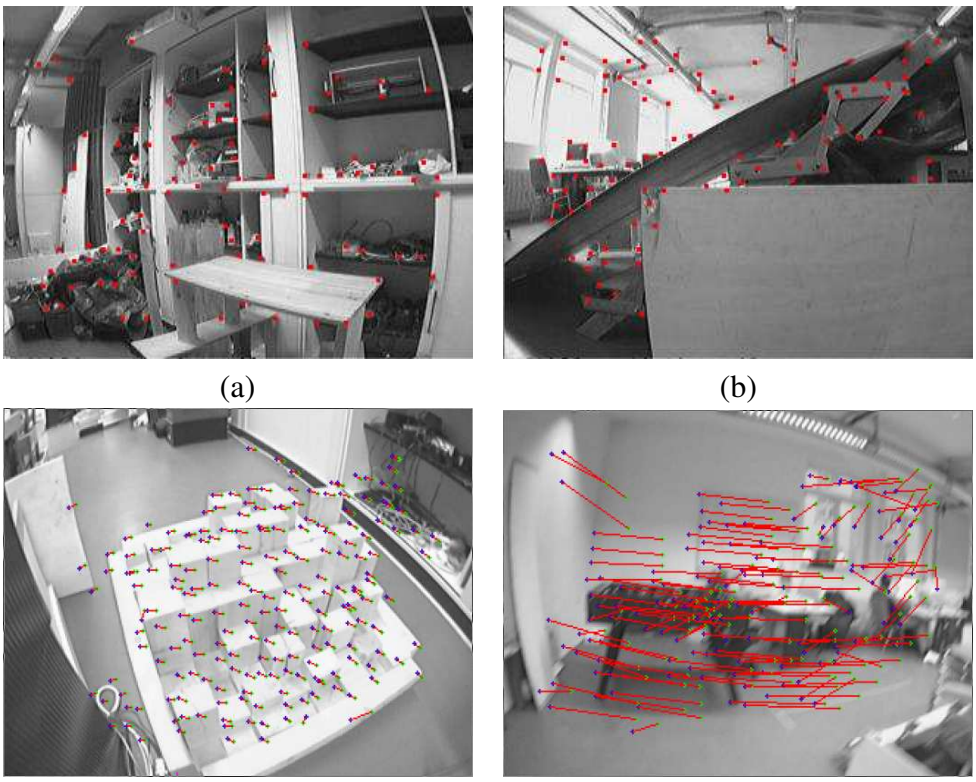
Usually, a feature tracker determines this translation by minimizing the squared error ϵ over a tracking window. For brevity we define $I(x, y, t + \tau)$ as $J(x)$ and $I(x - \xi, y - \eta, t)$ as $I(x - d)$, leading to the following error measure with a weighting function w [Tomasi and Kanade, 1991]:

$$\epsilon = \int_{\mathcal{W}} [I(x - d) - J(x)]^2 w dx. \tag{3.11}$$

To facilitate the process of feature tracking, the selection of appropriate features, i.e. features that can easily be distinguished from noise, is necessary. Hence, features that show light-dark changes, e.g. edges, corners, and crossings, are selected with high probability by the KLT feature tracker. In Figure 3.4, examples of KLT's adaptive feature selection and the tracking over a series of images are shown.

When traversing obstacles, the robot's motion is not exclusively a forward or backward motion. Instead, it is overlaid with noise that originates from slippage of the tracks and shaking of the robot's body due to rough terrain, leading to jitter effects. Since these effects usually do not accumulate over time, our method generates trackings over multiple frames, rather than performing single frame trackings only.

If trackings of the same feature coexist over more than two images, their corresponding translation vectors $d_i, d_{i+1}, ..., d_k$ are replaced by a single translation vector d_{ik}, consisting of the vector sum of all trackings between d_i and d_k. In order to reward trackings over multiple frames, a weight $w_{ik} = |k - i|$ is assigned to each tracking. These weights



(a) (b)

(c) (d)

Figure 3.4: KLT feature tracking: (a,b) Features (red dots) are adaptively selected within images. (c) Feature tracking over two subsequent images. The vectors between two corresponding features, shown by red lines, indicate the movement of the camera. (d) The tracking over a series of five images.

are used during the voting process, which will be described in Section 3.4.3.

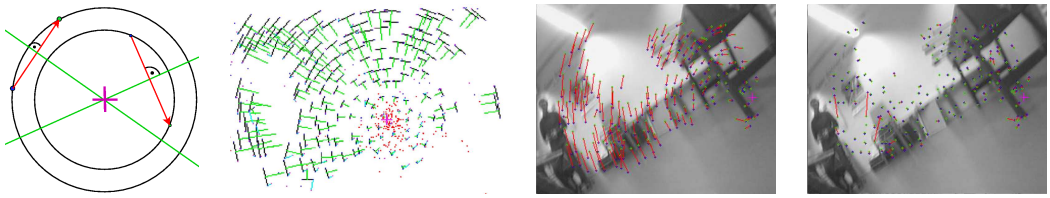
3.4.2 Filtering of rotations around the pitch angle

Since the focus lies on translation estimation, rotations have to be filtered-out in advance. It is assumed that translations are free of rotations around the yaw angle since otherwise the motion is considered as a rotation. However, during straight-line navigation, rotations might occur around the pitch angle while climbing over obstacles and movements of the flippers that change the robots pitch with respect to the ground. Due to the high variance in the latency time of the employed camera system (a web cam connected via USB 1.1), this can only be accurately achieved on the image data directly, rather than by combining image data with rotation angles from the IMU sensor. Given a feature tracking between two images of the form $(x_i, y_i)^T \rightarrow (x_j, y_j)^T$, which includes

a rotation around the point $(r_x, r_y)^T$ with angle α, one can derive a corresponding rotation free tracking $(x_i, y_i)^T \rightarrow (x'_j, y'_j)^T$ after the following equation, with given rotation matrix $R(.)$:

$$(x'_j, y'_j)^T = (r_x, r_y)^T + R(-\alpha) \cdot (x_j - r_x, y_j - r_y)^T \tag{3.12}$$

Therefore, in order to perform the filtering of rotations, one has to determine the rotation center $(r_x, r_y)^T$ and rotation angle α. Rotating points of different radii describe concentric circles around the rotation center. When considering two feature trackings whose features lie on a circle, one can see that the perpendicular bisectors of the two lines, given by the connecting start- and endpoint of the feature tracking, subtend in the rotation center, as shown in Figure 3.5 (a).



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

Figure 3.5: (a) The perpendicular bisectors (green) of the tracking vectors (red) subtend at the center of the circle (magenta). (b) Example of the Monte Carlo algorithm: The perpendicular bisectors (green) point to the center of rotation (magenta). Red dots depict the sampled intersection points. (c,d) Example of the rotation correction while the robot changes the angles of its front flippers. The feature vectors before (c) and after (d) the correction.

This property is exploited with a Monte Carlo algorithm for estimating the true center of rotation (see Figure 3.5 (b)). First, up to n possible centers of rotation are sampled from the set of feature trackings T by Algorithm 1. Second, all sampled centers of rotation are put into a histogram, where the final center is determined by the histogram's maximum.

Furthermore, one has to determine the rotation angle, which can be done by calculating the vector cross product. Given a feature tracking $(x_i, y_i)^T \rightarrow (x_j, y_j)^T$ rotated around $(r_x, r_y)^T$ by α, one can calculate the cross product by considering the start- and endpoint of the feature tracking as endpoints of vectors starting at the rotation center. Suppose $v_i = (x_i - r_x, y_i - r_y)^T$ and $v_j = (x_j - r_x, y_j - r_y)^T$ are vectors derived from tracking images I and J, respectively. Then, the angle between these vectors $\alpha = \angle(v_i, v_j)$ can be calculated from the vector product: $v_i \times v_j = \|v_i\| \cdot \|v_j\| \cdot \sin(\alpha)$. Given the rotation center $(r_x, r_y)^T$ from the previous estimation, one can determine the true rotation angle α by averaging rotation angles from all single feature trackings. Finally, it is necessary to prevent the algorithm from being executed on rotation-free sequences. This is achieved by adding a center of rotation to the histogram, only if it is located within the bounding

Algorithm 1: Sample up to n possible centers of rotation

Input: A set of feature trackings: T
Output: A set of calculated intersection points: C

```
C = ∅;
for i = 0; i < n; i++ do
    t1 ← selectRandomFeatureTracking(T);
    t2 ← selectRandomFeatureTracking(T);
    s1 ← calculatePerpendicularBisector(t1);
    s2 ← calculatePerpendicularBisector(t2);
    (cut, det) ← calculateIntersectionPoint(s1, s2);
    if det < minDeterminant then
        continue;
    end
    C ← C ∪ cut;
end
```

box of the image. Rotation centers that are far from the bounding box are most likely due to quasi-parallel feature translations, which in turn indicate a rotation-free movement. If the number of centers of rotation is below a threshold λ, the transformation of Equation 3.12 is not applied. We determined experimentally $\lambda = 10$ [Dornhege and Kleiner, 2006].

3.4.3 Classification

From the set of filtered translation vectors, one can determine the robot's translation. However, the translation vectors are given in the two dimensional image plane. The projection from translation vectors of the vision system to the robot's translation depends on the intrinsic parameters of the camera, e.g. focal length and lens distortion, and on the extrinsic parameters of the camera, e.g. the translation and rotation relative to the robot's center. This projection can either be determined analytically or by a mapping function. Due to the assumption of a simplified kinematic model, this mapping can be learned efficiently by a function approximator, classifying each vector into one of the classes $C = \{forward, backward, none\}$.

The learning is based on collected data which was automatically labeled during teleoperation runs under mild conditions, i.e. without heavy slippage. During a second phase, the data labeling has been verified manually by a human on a frame to frame basis. This procedure allows for the efficient labeling of thousands of trackings since single images contain several features. Each labeled tracking is described by the class assignment $c \in C$ and the vector $v = (x, y, l, \alpha)^T$, where x, y denotes the origin in the image, l denotes the vector length and α denotes the vector heading. Given the labeled

data, tile coding function approximation is used for learning the probability distribution

$$P(c \mid x, y, l, \alpha). \tag{3.13}$$

Tile coding is based on tilings which discretize the input space in each dimension. Shape and granularity of these discretizations can be adjusted according to the task. Multiple tilings are overlaid with a randomized offset in order to facilitate generalization. During learning, each tile is updated according to: $w_{i+1} = w_i + \alpha \cdot (p_{i+1} - w_i)$, where w_i is the weight stored in the tile, $p_i \in \{0, 1\}$ is the manually labeled class assignment, and α is the learning rate, which is set to $\frac{1}{m}$, where m is the number of overlaid tilings in order to ensure normalized probabilities [Sutton and Barto, 1998]. Based on the probability distribution induced by $P(c \mid x, y, l, \alpha)$ over C, each vector v_i votes individually for a class assignment c_i with respect to its location, length, and heading:

$$c_i = \underset{c \in C}{\operatorname{argmax}}\, P(c \mid x_i, y_i, l_i, \alpha_i) \tag{3.14}$$

Let $c_i^k = I(c_i = k)$ be the class indicator function, which returns 1 if $c_i = k$ and 0 otherwise. Then, the final classification a for each frame can be decided based on the maximal sum of weighted individual votes from each vector:

$$a = \underset{k \in C}{\operatorname{argmax}} \sum_{i=1}^{N} c_i^k \cdot w_i \tag{3.15}$$

Note that w_i increases according to the number of times the underlying feature has been successfully tracked by the feature tracker previously described.

In order to determine the distance d traveled between two images I and J, a constant translational velocity v_T of the robot [†] is assumed. Given time stamp t_j and t_i of image I and J, respectively, d can be calculated by:

$$d = \begin{cases} v_T \cdot (t_j - t_i) & \text{if class = forward} \\ -v_T \cdot (t_j - t_i) & \text{if class = backward} \\ 0 & \text{otherwise} \end{cases} \tag{3.16}$$

Finally, from the yaw angle θ of the IMU and the robot's last pose $(x_{old}, y_{old}, \theta_{old})^T$ the new pose of the robot is calculated by:

$$(x_{new}, y_{new}, \theta_{new})^T = (x_{old} + d \cdot \cos(\theta_{old}), y_{old} + d \cdot \sin(\theta_{old}), \theta_{old})^T \tag{3.17}$$

[†]Note that this value could also be automatically adjusted according to the vehicles current set-velocity.

3.5 Pedestrian Dead Reckoning (PDR)

In this section a robust method for estimating the distance displacement d and azimuth angle (heading) θ of walking persons, by utilizing a single IMU sensor that contains a tri-axial accelerometer, a tri-axial gyroscope, and a tri-axial magnetometer, will be described. According to Section 3.2.1, the displacement vector $u = (d, \theta)^T$ with covariance matrix Σ_u is further processed by Kalman-based dead reckoning, yielding the pose estimate $\hat{l} = (\hat{x}, \hat{y}, \hat{\theta})^T$ with covariance matrix $\Sigma_{\hat{l}}$.

3.5.1 Distance estimation

Generally, the traveled distance d of a walking person can only under difficulties be computed by double integration of the antero-posterior (forward-backward) acceleration [Ladetto et al., 2000, Foxlin, 2005]. This has two reasons: first, the alignment of the IMU device on the body of the person strongly influences the measurements, hence it has to be calibrated each time the position of the device has been changed. Second, the double integration quickly accumulates measurement errors, e.g. caused by abrupt acceleration changes during walking, leading to non-permissible positioning errors. An alternative approach is to detect the step occurrence from vertical and horizontal acceleration patterns merged with a physiological model of human walking. Experiments have shown that the maximal step frequency of humans is about 5 Hz, and that there are no acceleration peaks beyond 15 Hz. Therefore, according to the law of Shannon, accelerations of walking patterns can be reliably measured at a sample frequency of 30 Hz. There are several identification strategies, ranging from computational expensive Fourier analysis to simple *Zero-crossing*, which basically detects a step if the acceleration curve crosses the zero value in an ascending way. A method that shows very robust results within the least computation time has been introduced by Ladetto et al. [Ladetto et al., 2000]. Human walking generates a vertical acceleration with a maximum value if a foot is placed on the ground (see Figure 3.6). By detecting these maxima in the vertical acceleration curve within a fixed time interval, it is possible to detect and count the occurrence of foot steps. The method can be separated into three parts, which are step detection, step counting, and direction detection.

To detect the walking mode of the person, e.g. to distinguish between forward walking, lateral (side-wards) walking, and standing, is a challenging task due to the individual walking style of humans. Ladetto concluded that the mean variance over a short period of the vertical (up-down), and antero-posterior (forward-backward) acceleration correlates significantly with the walking mode and step frequency. The higher the frequency, the higher the variance. The occurrence of steps can thus be determined from the mean variance computed within a fixed time interval. Furthermore, a comparison between the mean variance of the antero-posterior acceleration and vertical acceleration allows to distinguish between lateral (side-wards) and forward movements. A lateral

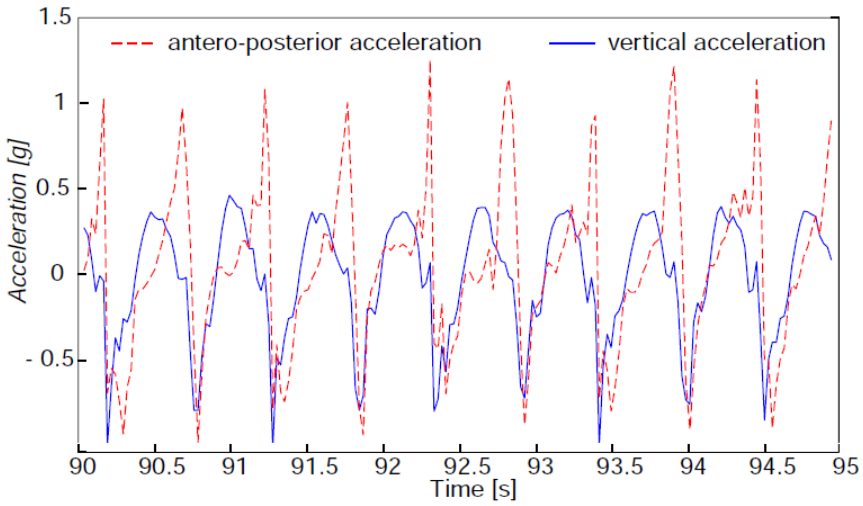


Figure 3.6: Vertical (up-down), and antero-posterior (forward-backward) acceleration patterns recorded during human walking. Courtesy of Quentin Ladetto.

movement is detected, if the variance of the vertical acceleration is higher than the one of the antero-posterior, and vice versa.

To detect the frequency of steps, one has basically to count the maxima in the antero-posterior acceleration curve. Depending on the individual walking style of a person, a step might cause two peaks located closely to each other, originating from the impacts of the heel and the sole with the ground, respectively, whereas the heel impact normally shows the bigger amplitude. In order to detect step maxima reliably, it is beneficial to filter-out signal disturbances in advance. This is achieved by applying a strong low-pass filter that removes high frequency components from the signal, e.g. frequencies above 5 Hz. Given the filtered signal with each local maxima representing a footstep, the step frequency is determined by time differencing the maxima. The traveled distance is computed by multiplying the frequency with the individual length of the person's steps. Since the step length greatly varies among different persons, we automatically calibrate this parameter from distances computed by GNSS positions if they are available [Ladetto et al., 2000].

Furthermore, the direction of walking (forward-backward, left-right) is determined by computing the angles between the maxima and the two zero-crossings of the signal before and after the maxima, respectively. For example, a backward motion is detected if the angle between the maxima and the previous zero-crossing is bigger than the angle between the maxima and the following zero-crossing. A more detailed description of the method is found in the work of Ladetto and colleagues [Ladetto et al., 2000, Gabaglio, 2003, Ladetto and Merminod, 2002b]. For the experiments described in

this thesis an implementation of Ladetto's work from Michael Dippold [Dippold, 2006] has been used.

3.5.2 Heading estimation

The compass and the gyroscope of the IMU are used in conjunction in order to compensate perturbations from local magnetic fields, as for example caused by ferromagnetic building materials such as reinforced concrete or steel [Ladetto and Merminod, 2002a, Ladetto et al., 2001]. This approach ideally combines the strength of both sensors and compensates their weaknesses at the same time. Whereas the compass provides absolute measurements of the azimuth, which are partially inaccurate due to local magnetic fields, the gyroscope measures, also under magnetic influence, the angular acceleration with high accuracy. However, long term measurements of the gyroscope are affected by an inherent drift error. Hence, drift errors of the gyroscope can be compensated by globally accurate azimuth angles from the compass, and local disturbances of the compass can be compensated with locally accurate measurements of the angular acceleration.

For the experiments presented in this thesis, an *MTi* IMU from *Xsens* was used (see Chapter 2, Section 2.2.1). The sensor unit provides a special modus called AMD (Adapt to Magnetic Disturbances) which can be activated for environments containing heavy magnetic fields. The device was attached to a test person for measuring the vertical and lateral acceleration, as well as the azimuth orientation (see the orange box in Figure 3.7). Based on an empirical evaluation, we modeled pose tracking uncertainty with $\sigma^2_{\hat{d}} = (0.05\,m)^2 d$, and $\sigma^2_{\hat{\theta}} = (15^\circ)^2$.

3.6 Experimental results

In this section, results from both simulated and real-robot experiments are provided. All real-robot experiments were carried out on the robot platforms described in Chapter 2 in outdoor scenarios, and testing arenas that are equal or similar to those proposed by NIST. Experiments on pedestrian dead reckoning were carried out with the *MTx* IMU from *Xsens* and a *SIRFstarIII* chip-based GPS device. Results from pose tracking on robots, specifically, wheeled pose tracking and visual odometry-based pose tracking, are presented in Sections 3.6.1 and 3.6.2, respectively, whereas results from dead reckoning on pedestrians are presented in Section 3.6.3.

3.6.1 Pose tracking under heavy slippage

The slippage detection method has been extensively evaluated on the *Zerg* robot. During this experiment, the robot performed different maneuvers, such as moving straight,

Figure 3.7: Test person with *Xsens MTi* and *Holux* GPS device for obtaining ground truth data.

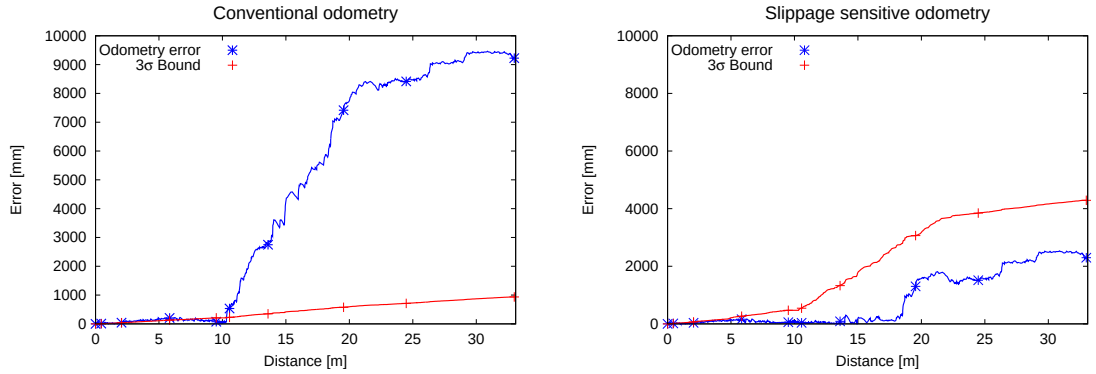
turning, and accelerating while driving first on normal and then on slippery ground. Afterwards, each situation has been manually labeled with one of the six classes *slip-straight*, *slip-turn*, *slip-accelerate*, *noslip-straight*, *noslip-turn*, and *noslip-accelerate*. Table 3.1 summarizes the results of the classification, where bold numbers indicate the correct classification, i.e. *true-positives*. The method is able to reliably detect slippage even while the robot is accelerating or performing turns.

True situation \ Classification		Slip	No Slip
Straight	No Slip	10 (0.5%)	**2051 (99.5%)**
	Slip	**2363 (90.1%)**	236 (8.9%)
Turn	No Slip	28 (0.9%)	**3226 (99.1&)**
	Slip	**2684 (96.4%)**	102 (3.6%)
(De-)Acceleration	No Slip	75 (14.9%)	**426 (85.1%)**
	Slip	126 **(98.5%)**	2 (1.5%)

Table 3.1: Classification accuracy of the slippage detection under different maneuvers of the robot. Bold numbers indicate the correct classifications, i.e. *true-positives*.

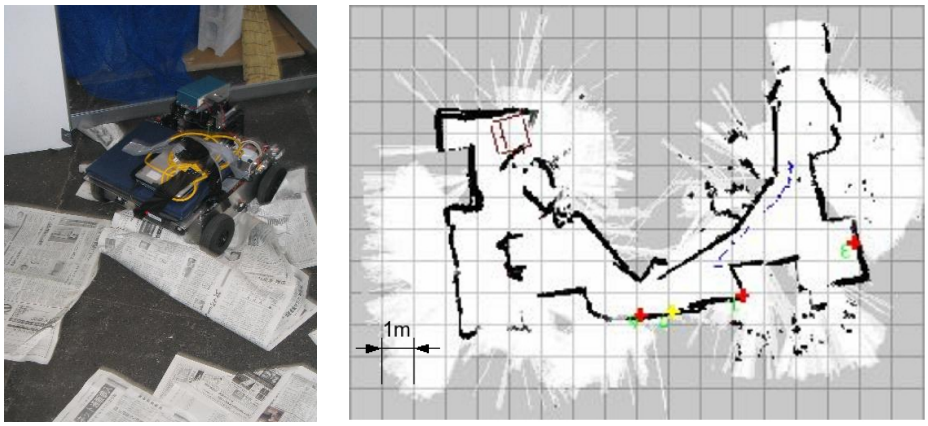
In order to evaluate the slippage detection-based improvement of the odometry, we conducted experiments for the comparison of both improved and conventional odometry and their covariance bounds. Figure 3.8 shows the performance of slippage sensitive odometry compared to conventional odometry. It can be seen in Figure 3.8 (a) that the

error of the conventional odometry increases drastically during slippage (taking place between 10 and 20 meters). Moreover, the covariance bound significantly underestimates the error. However, in the same situation, slippage sensitive odometry is capable of reducing the error (Figure 3.8 (b)), while providing valid covariance bounds.



(a) (b)

Figure 3.8: Conventional odometry (a) compared to slippage sensitive odometry (b) during the event of slippage (between 10 and 20 meters): In contrast to conventional odometry, improved odometry reduces the position error (blue line) and provides valid covariance bounds (red line) during slippage.



(a) (b)

Figure 3.9: *Zerg* robot during the final of the *Best in Class autonomy* competition at RoboCupRescue 2005 in Osaka: (a) slipping on newspapers and (b) the autonomously generated map. Red crosses mark locations of victims which have been found by the robot.

The approach of slipping detection has been utilized during the RoboCup Rescue

competition. Figure 3.9 depicts the *Zerg* robot during the final of the "Best in Class Autonomy" competition, held in the NIST arena for Urban Search and Rescue (USAR) [Jacoff et al., 2001] during RoboCup 2005. In this scenario, robots had to explore an unknown area within 20 minutes autonomously, to detect all victims, and finally to deliver a map sufficient for human teams to locate and rescue the victims. Conditions for exploration and SLAM were intentionally made difficult. For example, the occurrence of wheel slip was likely due to newspapers and cardboards covering the ground, which was partially made of steel and concrete. Stone bricks outside the robot's field of view caused the robot to get stuck, and walls made of glass caused the laser range finder to frequently return far readings. We applied computer vision techniques on images generated by a thermo (IR) camera in order to estimate the relative locations of victims, if they were detected withing the camera's FOV. More details on victim detection are given in Chapter 7. As shown in Figure 3.9, the system was able to cope with these difficulties and also to build reliable a map augmented with victim locations detected by the robot. Finally, the system won the autonomy competition in 2005.

3.6.2 Pose tracking on tracked vehicles

The approach of visual odometry was extensively tested on both the tracked robot *Lurker*, operating on three-dimensional obstacles, and the wheeled robot *Zerg*, operating on flat surfaces. Experiments with the *Zerg* robot have the advantage that the visual odometry, and conventional wheel odometry can be compared directly to ground truth data. They allow a comparison of both methods under the same circumstances and show, that the visual odometry produces results of comparable accuracy to wheel odometry. On this robot, position ground truth was determined by LRF-based scan matching, whereas ground truth on three-dimensional obstacles was measured manually.

Run	Trav. dist. [m]	Vis. odo. [cm/m]	Wh. odo. [cm/m]
lab_1 (2D)	91.53	7.82 ± 1.84	6.17 ± 1.54
lab_2 (2D)	73.72	8.25 ± 2.46	7.59 ± 1.94
cellar (2D)	108.91	19.30 ± 13.65	21.22 ± 12.40
ramp (3D)	6.36	13.28 ± 9.2	-
palette (3D)	2.37	22.08 ± 8.87	-

Table 3.2: Relative error of the visual odometry and conventional wheel odometry compared to ground truth data (either manually measured for 3D runs or estimated by scan matching for 2D runs).

Table 3.2 gives an overview on the measured mean and standard deviation of the relative distance error from visual odometry and wheel odometry on both robot platforms. Since the method has been mainly developed for tracked vehicles, the *Zerg's* kinematic

has been modified in order to be similar to that of the evaluated tracked vehicle, i.e. to allow only a subset of possible velocities, which are in case of the *Lurker* robot: *stop*, *forward*, and *backward*. The results clearly show that on the *Zerg* platform the visual odometry reaches an accuracy comparable to the conventional odometry. In the cellar environment, the visual odometry turned out to be even slightly superior, which can be explained by the higher degree of wheel slippage that we noticed in this environment. Comparing the graphs in Figures 3.10 (a) and (b) of the cellar environment with those of the lab in Figures 3.10 (c) and (d) it becomes clear that the cellar environment is much harder for both algorithms. The wheel odometry usually suffers from wheel slippage while the visual odometry has to cope with the fact that the white walls do not provide good features. Figures 3.10 (a) and (c) depict the accumulation of the distance error of the wheel odometry and Figures 3.10 (b) and (d) show the distance error of the visual odometry in the cellar and lab environment, respectively. The real advantage of the visual odometry, however, is revealed if the robot operates on three-dimensional obstacles.

The results in Table 3.2 indicate that the introduced method, when applied while operating on three-dimensional obstacles, provides a usable estimate of the robot's motion. Figures 3.11 (a) and (b) depict the driven distance during locomotion over three-dimensional obstacles compared to ground truth distances. The results indicate that, in case of tracked robots, the *tile coding* classification and voting applied to a simple kinematic model lead to sufficiently accurate results. From log files it has been determined that during the *cellar* run 87% (96%), the *ramp* run 81% (93%), and the *palette* run 94% (99%) of the classifications detected the correct motion of the robot, where numbers in brackets denote the voting-based improvement. While processing an image resolution of 320×240 on a *IntelPentiumM*, 1.20 GHz, we measured an average processing time of 24.08 ± 0.64 ms for the complete processing without KLT feature tracking. This leads, together with the feature tracker, to a maximal frame rate of 5.34 ± 1.17 Hz. If processing an image resolution of 160×120, the complete processing without KLT feature tracking needs 8.68 ± 0.3 ms and allows a total frame rate of 17.27 ± 1.81 Hz. Experiments proposed in this thesis were carried out with the higher resolution. However, experiments with the lower resolution showed that these results lead to a comparable accuracy, too.

The following two experiments demonstrate the application of visual odometry in the context of SLAM on tracked vehicles negotiating obstacles. We conducted the experiments on the *Lurker* robot by utilizing the visual odometry together with the scan matching algorithm. During the first experiment, the LRF sensor was automatically controlled by the measured pitch orientation of the robot in order to stay continuously at a horizontal position. This allows the LRF to perceive the environment independently from the robot's orientation, i.e. to return the same laser scan at the same locations also if the orientation differs. The result is shown by the image series in Figure 3.12 (a-f). In (a) and (d) an overview on both obstacles is given, whereas (b) and (e) depict

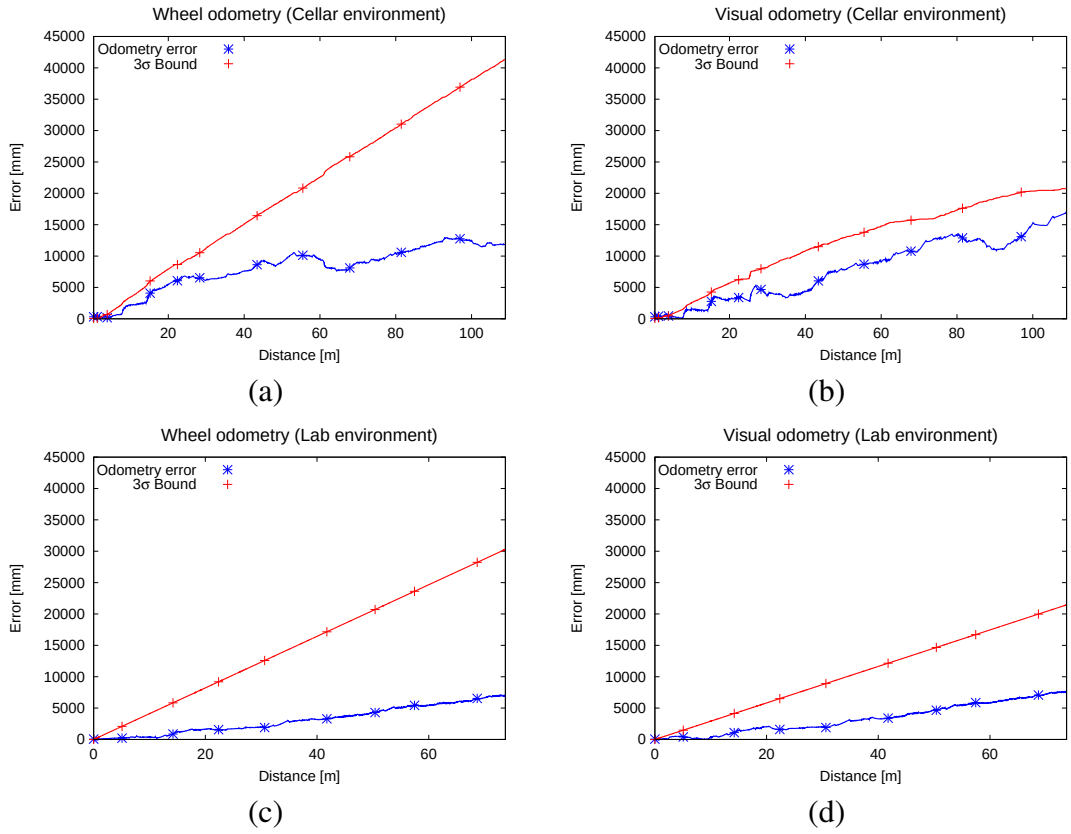
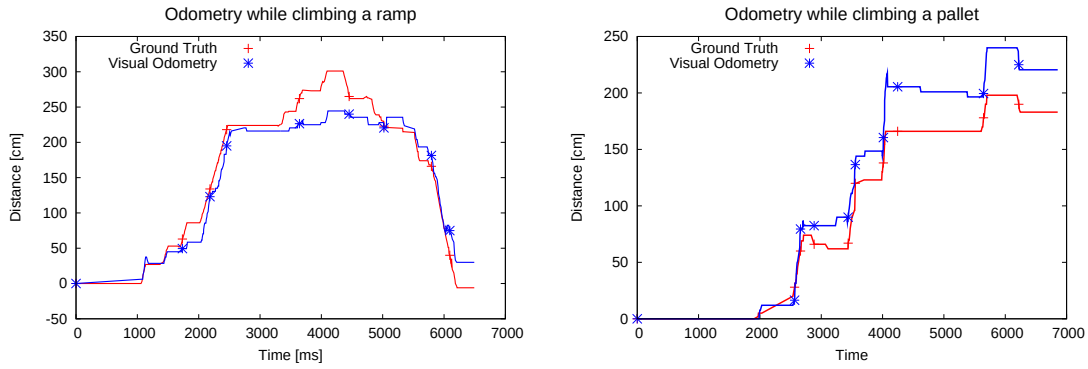


Figure 3.10: The accumulating distance error of the visual odometry method compared to ground truth data: (a,b) measured in a cellar of $15\,m \times 50\,m$, (c,d) measured in the robotic lab, a $5\,m \times 5\,m$ squared area, (a) and (c) show results from the wheel odometry, (b) and (d) display the errors originating from visual odometry.

the generated features, and (c) and (f) show the generated maps, at the corresponding positions, respectively.

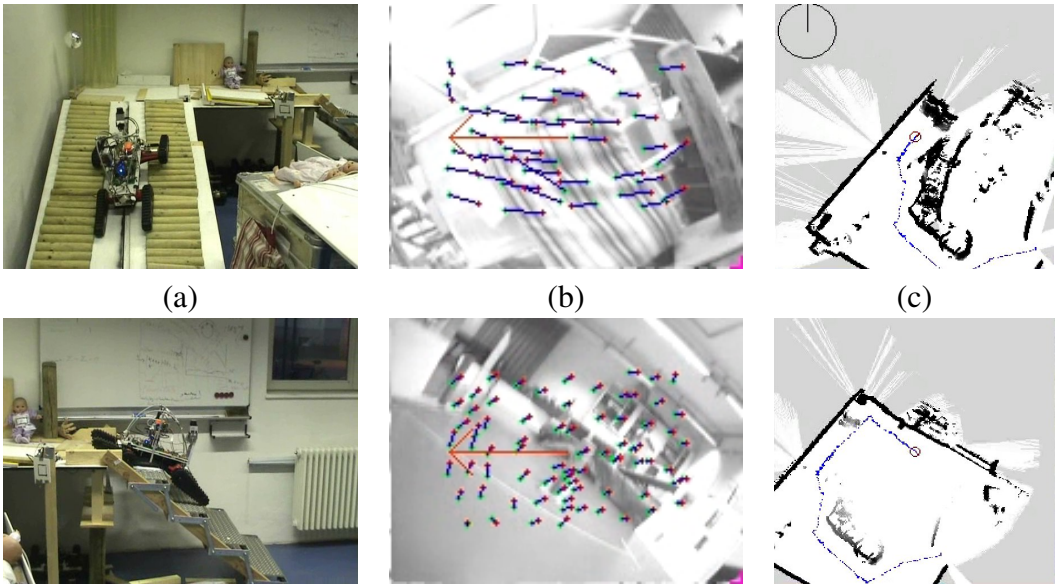
As shown by the "black wall" in front of the robot (Figure 3.12 (c)), there are situations in which the 2D LRF cannot provide sufficient evidence for the robot's motion. In this particular case, measurements of the laser are nearly independent of the robot's location on the ramp, whereas the visual odometry (see Figure 3.12 (b)) provides clear motion evidence. Note that the map representation shown in (c) and (f) does not suffice for the particular task since it does not distinguish between measurements of different height values at the same location, i.e. parts of the map in (c) have been deleted in the subsequent map (f). This problem can be solved by utilizing elevation maps, as discussed in Chapter 5.

In another experiment, the influence of visual odometry on elevation mapping has been evaluated. Figure 3.13 depicts two elevation maps of the same ramp, one with



(a) (b)

Figure 3.11: The distance of the visual odometry method compared to manually generated ground truth data: Results from driving forward and backward on a ramp (a), and climbing over a wooden palette (b). The red curve (crosses) indicates the manually measured ground truth, and the blue curve (asterisks) indicates the distance estimated by visual odometry, respectively.



(a) (b) (c)

(d) (e) (f)

Figure 3.12: *Lurker* robot performing vision-based pose tracking while overcoming three-dimensional obstacles: (a,d) the three-dimensional obstacles, (b,e) the features and classified directions generated from a side camera, and (c,f) the grid maps generated at these locations, respectively.

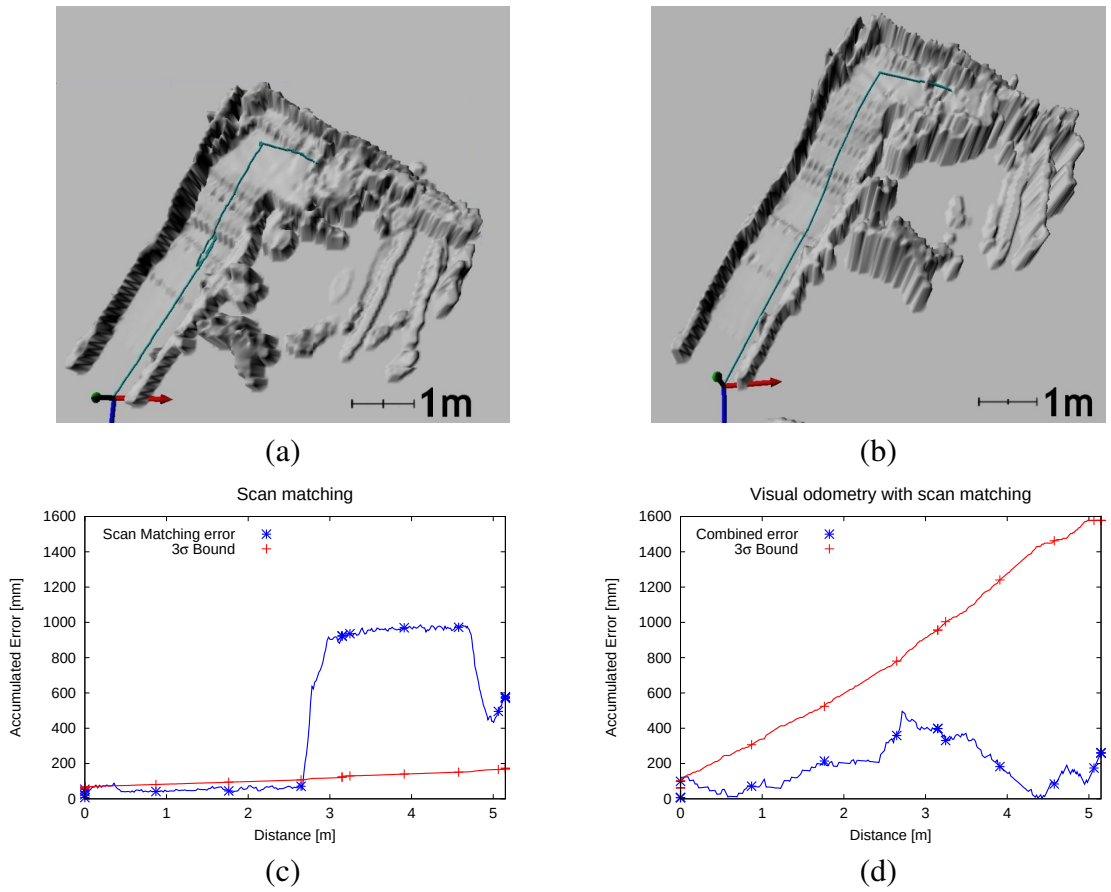


Figure 3.13: Comparing elevation mapping based on scan matching only (a) and scan matching combined with visual odometry (b). The scan matchings small error (c) grows rapidly out of its usual error bound, when the robot drives on the ramp, while the visual odometry (d) is not influenced by this effect. Scan matching without visual odometry support does not correctly reflect the true length of the ramp, because insufficient motion evidence causes the map to be compressed partially.

support of visual odometry, and one without. The corresponding error graphs show that scan matching cannot correctly reflect the robot's motion when the robot drives on the ramp between 2.75 m and 4.75 m. Usually, estimating the robot's pose based on two dimensional scan alignment can be done reliably, but this is no longer possible when the two-dimensional reference system changes. The result is a rapid error growth (Figure 3.13 (c)) that leads to distortions in the map due to the incorrect pose assumption. Fusing distance estimates from the visual odometry, that are not influenced by this effect, into the scan matchings pose clearly reduces the error. Mapping based on scan matching only yields a compressed map since in this environment 2D laser scans

do not provide sufficient information on the motion of the robot, whereas generating a map based on visual odometry reveals the true size of the ramp, which was verified by measuring the ramp's dimensions manually.

3.6.3 Pose tracking of human walking

The PDR implementation described in Section 3.5 has been extensively tested in outdoor scenarios. Outdoor scenarios have the advantage that ground truth data can be generated from differential GPS positioning yielding an error within a few meters. Hence, results from dead reckoning can be compared directly with the true trajectory of the person.

In order to evaluate the setting, we recorded trajectories of a pedestrian walking for approximately 3 kilometers around a lake in Freiburg. During these experiments, the test person was naturally walking at different speeds, either in a forward or side wards direction, as well as stopping at certain locations. The experiments did not include unusual motions, such as crouching or climbing. Figure 3.14 shows the resulting trajectories from different experiments. The red line indicates the ground truth generated from GPS readings, and the green line depicts the trajectory generated with the PDR module. As expected, the PDR trajectory diverges from ground truth, the longer the person walks. Particularly in Figure 3.14 (b), after a walk of approximately 3 kilometers, the PDR position has nearly an error of 240 meters. In Chapter 4 it will be shown how these trajectories can be improved by utilizing observed correspondences of RFIDs.

3.7 Related work

Borenstein et al. introduced a method for improving the odometry on differential-drive robots [Borenstein, 1996]. A method for odometry improvement and optimization of motor control algorithms on 4WD robots has been introduced by Ojeda et al. [Ojeda and Borenstein, 2004]. They applied "Expert Rules" in order to infer the occurrence of wheel slip.

The approach of visual odometry was extensively studied in the past. Davison and colleagues developed a system for visual EKF-based SLAM using only a single camera [Davison, 2003]. Their system was mainly evaluated in desk-like environments. However, in contrast to the system introduced in this chapter, their approach requires a calibration of the camera at start-up in order to estimate the three-dimensional real-world locations of the features. Corke and colleagues introduced a solution for a planar rover equipped with an omni-directional vision system [Corke et al., 2004]. In contrast to the proposed work, which also aims at indoor applications, they assume that the robot operates in an open space, as it is usually the case on planetary analog environments. Milella and Siegwart proposed a system that computes the 6 DOF ego motion of an all-

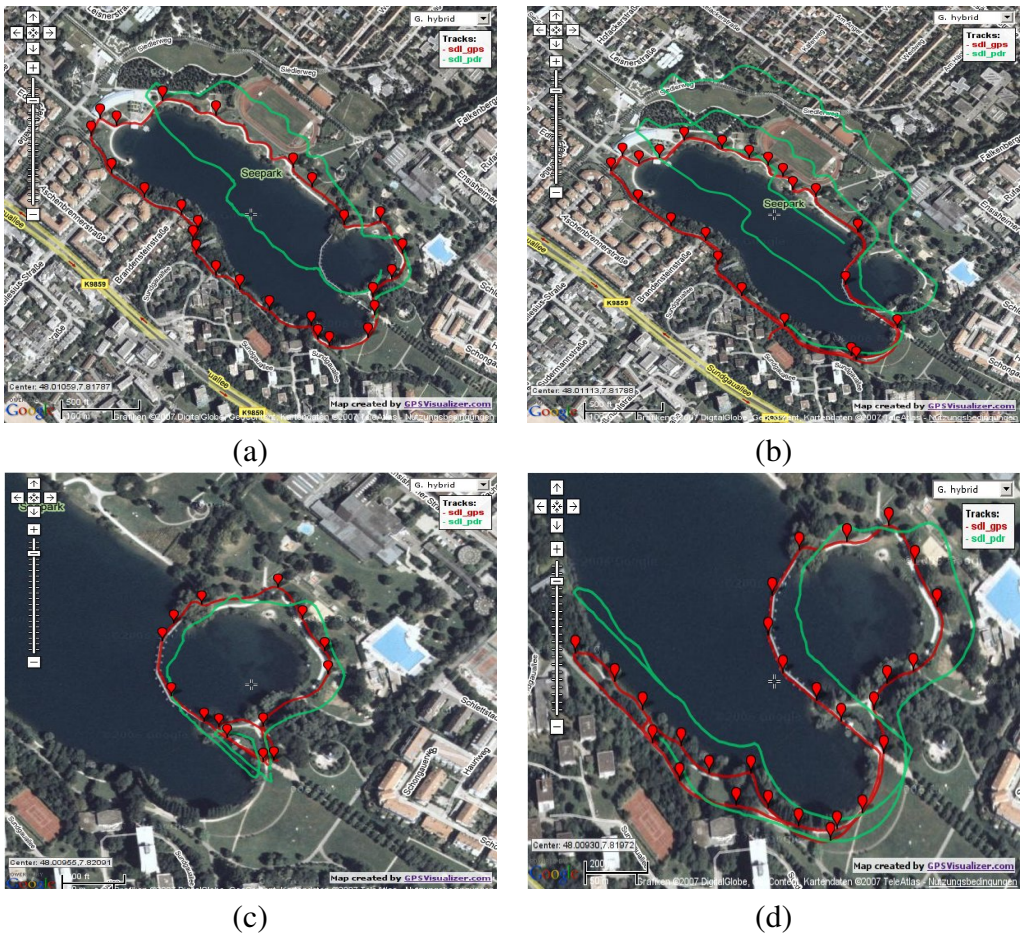


Figure 3.14: Comparison between PDR (green trajectory) with GPS ground truth (red trajectory). Trajectories have been recorded while walking around a lake in Freiburg

terrain robot based on the Iterative Closest Points (ICP) method processing data from a 3D stereo vision system [Milella and Siegwart, 2006]. Nister and colleagues presented a system for visual odometry that works with both mono and stereo vision [Nister et al., 2004]. Their results show that data processing of a stereo system leads to a highly accurate estimate of the robot's pose, which was also confirmed by the work of Helmick and colleagues [Helmick et al., 2004]. The use of a stereo system generally has the advantage that the depth information of features tracked by the vision system can be utilized for computing the velocity of the robot. Results proposed in this chapter show that with the simplified kinematics of tracked robots, a single but lightweight camera solution can also lead to sufficiently accurate pose estimates.

PDR methods were extensively studied in the past. Human motion has been tracked by vision sensors [Zhu et al., 2006], as well as based on the analysis of acceleration

patterns [Amft et al., 2004, Foxlin, 2005, Ladetto et al., 2000, Judd, 1997]. Zhu and colleagues proposed a system that combines data from an IMU sensor and GNSS device with dead reckoning information generated from a stereo vision package [Zhu et al., 2006]. However, since their system requires the user to wear two cameras mounted on a backpack, it cannot be applied in narrow scenarios. Due to the computational power of small devices and due to the high demand from a wide range of social and military applications, the development of light-weight solutions for pedestrian dead reckoning has been significantly accelerated in the past. The approach introduced by Ladetto and colleagues [Ladetto et al., 2000] has been commercialized and is available as a hardware implementation, namely the *Personal Navigation Module* (PNM) [Vectronix, 2007]. The PNM, which is completely integrated into a box that weighs 34 grams and measures $43 \times 34 \times 25$ mm, was developed by the *Ecole Polytechnique Fedalrale de Lausanne* (EPFL) together with the *Leica Vectronix AG*. It is worth mentioning that the system won the Swiss Technology Award in 2003. A comparable system is the *Dead Reckoning Module* (DRM), which was developed by Tom Judd [Judd, 1997] at the *Point Research Cooperation* and the fifth generation (DRM5) is commercially available from *Honeywell* [Honeywell, 2007]. The system was evaluated in the context of disaster response for the localization of first responders [Miller et al., 2006]. During these experiments persons were tracked with the DRM while walking through buildings of different types.

3.8 Conclusion

In this chapter methods for pose tracking on tracked and wheeled robots, as well as for pedestrian walking, were introduced. Methods for tracking robots were mainly evaluated in the testing arenas proposed by NIST for Urban Search and Rescue. Experimental results showed that under modest computational requirements good results have been achieved, even under the harsh environmental conditions found in the arenas of the NIST benchmark. While pose tracking based on slippage sensitive odometry enabled mapping during heavy slippage, the visual odometry method allowed to improve pose estimates significantly while navigating on rough terrain.

As results from PDR experiments showed, the tracking of human beings based on accelerometers leads to promising results. However, also these results are only preliminary compared to the situations first responders are exposed to while rescuing victims after a real disaster. They might crouch, run, and climb within very different and individual modes, generating a wide range of ambiguous sensor values. Also in this area further research has to be undertaken in order to capture the whole range of human motion. Nevertheless, results as they were proposed in this chapter are sufficient for dead reckoning in scenarios that do not require special modes of motion.

4 RFID Technology-based SLAM

4.1 Introduction

Truly autonomous robot navigation in an unexplored environment, as well as efficient victim search performed by humans, requires to incrementally build a map from observations while keeping track of positions at the same time. This problem is generally referred to as Simultaneous Localization And Mapping (SLAM) and can be decomposed into a state estimation problem and a data association problem. The techniques described in Chapter 3 solve the state estimation problem by continuously tracking the pose of pedestrians and robots. These methods suffer from the problem of error accumulation, i.e. the variance of pose estimates increases according to the length of the traveled trajectory. Therefore, it is required to recognize previously visited places in order to correct the trajectory estimated by the pose tracker.

However, in unstructured environments, data association is still a challenging problem. Many existing techniques have been developed under strong assumptions, for example, they require polygonal structures, as they are typically found in office-like environments [Gutmann and Schlegel, 1996, Grisetti et al., 2002] or depend on predictable covariance bounds from pose tracking for solving the data association problem by validation gating [Dissanayake et al., 2001]. For example, it is not guaranteed that LRF-based methods work reliably if far-reading measurements or ambiguous patterns prevent distinguishable features. Data association from camera images requires at least partially stable illumination conditions, and might lead to unsatisfactory results in highly dynamic environments. Particularly in the context of disaster response, these methods can only be applied limitedly. Firefighters at 9/11 reported that they had major difficulties to orientate themselves after leaving collapsed buildings due to limited visibility, e.g. caused by smoke and fire, as well as due to the lack of recognizable structures, such as corridors and doorways. Furthermore, there might be places that are only accessible by robots, making it necessary to integrate humans and robots into one team for mapping the area after a disaster. Finally, SLAM works with the principle of map improvement through loop-closure, however, when facing the reality of emergency response, firefighters will intentionally try to *avoid* performing loops, e.g. while they are searching for victims.

These requirements and extraordinary circumstances make it very hard to apply common techniques from robotics. An alternative solution to the data association problem

is to utilize RFID technology. RFID tags have a worldwide unique number, and thus offer a robust way to label and to recognize locations in harsh environments. Their size is already below 0.5 *mm*, as shown by the μ-chip from *Hitachi* [Hitachi, 2003], and their price is lower than 13¢ [AlienTechnology, 2003]. Passive RFID tags do not require to be equipped with a battery since they are powered by the reader if they are within a certain distance. Their reading and writing distance, which depends on the employed communication frequency, can be assumed to be within a range of meters (see Chapter 2).

In this chapter, RFID-SLAM, a novel method for SLAM is contributed. The method utilizes RFID technology for data association, allowing robust and efficient loop closure in large-scale environments. Specifically the RFID approach enables information sharing between pedestrians and robots, facilitating it for individual team members to improve their map without performing loops. Furthermore, a decentralized variant of RFID-SLAM (DRFID-SLAM) is proposed, which allows agents to jointly improve their maps without requiring radio communication.

In the proposed approach, RFIDs are actively deployed by robots or humans at adequate locations, as for example narrow passages that are likely to be passed. The displacements between RFID tags are estimated by pose tracking methods, and utilized for building a joint RFID graph from multiple human and robot trajectories, which is globally optimized by minimizing the Mahalanobis distance [Lu and Milios, 1997]. The emerging graph structure can be considered as a topological map containing metric information describing the displacement between the nodes. The advantage is that robots and humans are able to compute their global metric position efficiently based on a sparse graph representation. Finally, the jointly corrected RFIDs are used as global constraint points for interpolating the trajectory of each single agent. Pose tracking on robots is carried out from slippage-sensitive wheel odometry and IMU data (see Chapter 3, Section 3.3), and pose tracking by humans is based on the IMU sensor only (see Chapter 3, Section 3.5).

The introduced method was evaluated extensively by human and robot experiments, which were performed indoors and outdoors. The results show that the method is capable of closing large loops within a few seconds, and moreover, allows to correct loop-free trajectories if they are shared between the agents.

The remainder of this chapter is organized as follows. In Section 4.2 conventional SLAM approaches are addressed. Centralized RFID-SLAM is discussed in Section 4.3, and the distributed version of the approach (DRFID-SLAM) will be presented in Section 4.4. In Section 4.5 results from experiments are proposed, and in Section 4.6 related approaches are discussed. Finally, in Section 4.7 the conclusion is presented.

4.2 Conventional techniques

4.2.1 EKF-based SLAM

Extended Kalman Filter (EKF) based SLAM is a well-known method for continuously updating the pose of a robot from motion commands, known as the *prediction step*, and landmarks observations, known as the *observation step* [Bailey, 2002, Durrant-Whyte et al., 1996, Dissanayake et al., 2001]. The prediction step deals with the motion of the robot by incrementally applying pose tracking (see Chapter 3), which continuously increases the uncertainty of the pose estimate according to the error model of the odometry. The observation step occurs if a landmark is detected in the environment. This step improves the overall state estimate if a previously stored landmark can be re-observed, whereas when a landmark is observed for the first time, it is added to the memory through an initialization process called *state augmentation*.

The basic idea behind EKF-based SLAM is to correlate the pose of the robot with the locations of landmarks by storing both in a single state vector and covariance matrix. It has been shown by the work of Durrant-Whyte that particularly the correlations between landmarks formulated within the single state vector contribute to the convergence of the method [Durrant-Whyte et al., 1996]. Under the assumption that landmarks are static in the environment, their correlation improves monotonically as more observations are made [Dissanayake et al., 2001].

Let the pose of the robot be the vector $l = (x, y, \theta)^T$ with 3×3 covariance matrix Σ_l and the locations of n landmarks by the vector $m = (x_1, y_1, x_2, y_2, \cdots, x_n, y_n)^T$ with $n \times n$ covariance matrix Σ_m, then the single state vector s is defined by:

$$s = \begin{pmatrix} l \\ m \end{pmatrix} \tag{4.1}$$

$$\Sigma_s = \begin{pmatrix} \Sigma_l & \Sigma_{lm} \\ \Sigma_{ml} & \Sigma_m \end{pmatrix}. \tag{4.2}$$

It is assumed that at the beginning of the recursive procedure the robot starts at location $(0, 0)$ with an empty set of observed landmarks, i.e. $l = 0$ and $\Sigma_s = \Sigma_l = 0$. At each motion update the prediction step is carried out by applying pose tracking according to the input $u_t = (d_t, \alpha_t)$ with covariance matrix Σ_u:

$$s_t = G(s_{t-1}, u_t) = \begin{pmatrix} F(l_{t-1}, u_t) \\ m_{t-1} \end{pmatrix} \tag{4.3}$$

$$\Sigma_{s_t} = \nabla G_s \Sigma_{s_{t-1}} \nabla G_s^T + \nabla G_u \Sigma_u \nabla G_u^T, \tag{4.4}$$

where

$$\nabla G_s = \begin{pmatrix} \nabla F_l & 0 \\ 0 & I \end{pmatrix} \tag{4.5}$$

$$\nabla G_u = \begin{pmatrix} \nabla F_u \\ 0 \end{pmatrix}, \tag{4.6}$$

and F_{lu}, ∇F_l, and ∇F_u are defined by Equations 3.3, 3.6, and 3.7, respectively. Note that this update does not affect the estimated locations of landmarks m and covariance matrix Σ_m since landmarks are assumed as stationary. However, the update modifies the partial covariance matrix Σ_l representing the uncertainty of the robots pose, as well as its cross-correlations Σ_{ml} and Σ_{lm} with landmark locations.

From an observation $z = (r, \phi)$ of a landmark within range r and bearing ϕ with 2×2 covariance matrix Σ_z, the state vector is updated as follows: first, the observation is associated to one of the landmarks stored in the state vector. This is carried out by either utilizing a similarity measure, e.g. based on features detected from the observation, or by selecting the landmark which is closest to the estimated global location of the observation, based on the Mahalanobis distance, also known as *validation gating*. Note if the landmark is unknown, i.e. cannot be associated to a known one, the state vector is augmented with the new observation. Second, based on the current estimates of associated landmark $m_i = (x_i, y_i)$ and robot pose $l = (x, y, \theta)$, the observation is predicted by the following measurement function:

$$H_i(s) = \begin{pmatrix} \sqrt{(x_i - x)^2 + (y_i - y)^2} \\ \tan^{-1}\left(\frac{y_i - y}{x_i - x}\right) - \theta \end{pmatrix}. \tag{4.7}$$

Third, the state vector is updated from the observation. This is carried out by computing the *innovation* v and its 2×2 covariance matrix Σ_v by applying the law of error propagation:

$$v_i = z - H_i(s) \tag{4.8}$$

$$\Sigma_{v_i} = \nabla H_s \Sigma_s \nabla H_s^T + \Sigma_z, \tag{4.9}$$

where the Jacobian ∇H_s is given by:

$$\nabla H_s = \begin{pmatrix} -\frac{\Delta x}{d} & -\frac{\Delta y}{d} & 0 & 0 & \cdots & 0 & \frac{\Delta x}{d} & \frac{\Delta y}{d} & 0 & \cdots & 0 \\ \frac{\Delta y}{d^2} & -\frac{\Delta x}{d^2} & -1 & 0 & \cdots & 0 & -\frac{\Delta y}{d^2} & \frac{\Delta y}{d^2} & 0 & \cdots & 0 \end{pmatrix} \tag{4.10}$$

with $\Delta x = x_i - x$, $\Delta y = y_i - y$, and $d = \sqrt{\Delta x^2 + \Delta y^2}$. Whether an association is accepted or rejected is determined by computing the Mahalanobis distance between the measured

and predicted observation:

$$M_i = v_i^T \Sigma_{v_i}^{-1} v_i, \tag{4.11}$$

An observation is accepted if $M_i < \lambda$, where λ denotes the gate threshold. If an observation is accepted, the following Kalman update is applied:

$$s_t = s_{t-1} + K v_i \tag{4.12}$$

$$\Sigma_{s_t} = \Sigma_{s_{t-1}} - K \Sigma_v K^T, \tag{4.13}$$

where the Kalman gain K is given by:

$$K = \Sigma_s \nabla H_s \Sigma_{v_i}. \tag{4.14}$$

The computation of the covariance matrix Σ_s requires $O\left(n^2\right)$ operations, where n denotes the number of landmarks. Hence, the computational cost increases quadratically with each new landmark the robot observes. Due to the correlations between landmarks and the pose of the robot, the map continuously improves by each observation. However, convergence is only guaranteed in the linear case, and fails if the linearizion performed by the Jacobian matrices is inaccurate, for example if the robot turns around quickly. However, the major drawback of the procedure is its sensitivity to incorrect data associations of landmarks [Neira and Tard, 2001]. Suppose the robot follows a larger loop in the environment and travels for a long distance without re-observing landmarks. Consequently, the uncertainty of the pose estimate increases by continuously applying the prediction step. If re-observing a landmark at the end of the loop, data association might fail since the observation at the predicted pose lies not within the Mahalanobis distance of the landmark stored in the state vector. Therefore, if the robot's pose is highly uncertain compared to the density of re-observable landmarks, loop-closure and hence convergence of the map cannot be guaranteed.

For the method as it has been described so far, it is assumed that observations take place sequentially, and if there are multiple observations at a discrete time step t, updates from observations are performed after a sequence, one update for each observation. Unfortunately, this strategy does not reflect landmark correlations for specific locations. Bailey introduced a *batch update* procedure that significantly reduces the data association problem by correlating landmarks observed at a single pose [Bailey, 2002]. He has shown by an empirical evaluation that batch updates lead to more reliable data association even under extremely noisy pose estimates from the robots odometry. However, landmarks are associated based on features detected by a laser range finder requiring a minimal amount of structure found in the environment.

4.2.2 FastSLAM

A remarkable reduction of the computational complexity of EKF-based SLAM is achieved by the *FastSLAM* method, which was introduced by Montemerlo and colleagues [Montemerlo et al., 2002]. Their approach is based on the work of Murphy [Murphy, 2000], who observed that, if the trajectory of the robot is known, the problem of determining the landmark locations can be decomposed into independent estimation problems, one for each landmark. Moreover, this factored representation is exact since landmark observations taken by the robot are conditionally independent. Consequently, FastSLAM solves the SLAM problem by solving collections of independent landmark estimation problems conditioned on individual trajectory estimates of the robot. The individual trajectory estimates are maintained by a particle filter, where each particle represents one possible map of the environment, consisting of a trajectory and N Kalman filters for estimating the locations of N landmarks. Hence, the computational complexity is $O(MN)$, where M is the number of particles and N the number of landmarks. They also introduced a more efficient implementation based on a tree-like data structure, leading to a computational complexity of $O(M \log K)$.

Another advantage of the method is its ability to track multiple hypotheses of data associations. In contrast to EKF-based SLAM, which decides once the association between a predicted and known landmark by the validation gate shown by Equation 4.11, FastSLAM allows to track simultaneously multiple hypotheses, each represented by a particle, whereas unlikely associations are filtered out by a re-sampling process. However, this method also might fail to converge given an arbitrary noisy pose estimate of the robot. Furthermore, it is unclear how many particles have to be utilized under harsh environmental conditions as they are present in areas after a disaster. In the next section it will be shown that RFID technology offers a straight-forward solution to the data association problem.

4.3 Centralized RFID-SLAM

RFID-SLAM [Kleiner et al., 2006c] is a procedure for mapping an area based on the odometry trajectories and RFID observations from multiple robots and humans in the field. For the sake of simplicity, robots and humans are denoted as "field agents", which communicate their observations to a central station. More specifically, the agents estimate distances between RFID locations by the pose tracking methods described in Chapter 3, and communicate these estimates back to the station, which centrally combines and corrects all trajectories. Furthermore, it is assumed that RFID detections are within a range below one meter allowing to cover corridors and doorways while providing sufficient positioning accuracy.

From the correspondences of observed RFID tags and their estimated displacements,

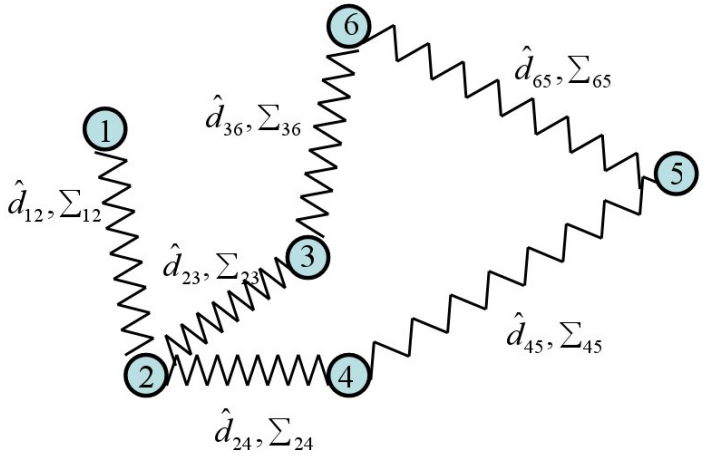


Figure 4.1: The spring-mass analogy to the generated RFID graph. Vertices represent RFIDs (masses) and edges between them represent measured distances with covariances (springs).

the station computes the globally consistent RFID locations by minimizing the Mahalanobis distance [Lu and Milios, 1997]. These corrected locations are finally used as global constraint points for interpolating single agent trajectories. The global optimization method can be illustrated by considering the analogy to a spring-mass system (see Figure 4.1). Consider the locations of RFIDs as masses and the measured distances between them as springs, where the uncertainty of measurements corresponds to the elasticity of the springs. Then, finding a globally consistent map is equivalent to finding an arrangement of the masses that requires minimal energy.

The reminder of this section is structured as follows. In Section 4.3.1 the construction of RFID graphs from communicated displacement estimates is discussed. The linear and non-linear optimization of RFID graphs is shown in Section 4.3.2 and Section 4.3.3, respectively. In Section 4.3.4 the interpolation of trajectories, and in Section 4.3.5 the RFID sensor model will be introduced.

4.3.1 Building RFID graphs

Each time an RFID has been observed, a message is generated that contains the ID of the previously visited RFID i and the currently visited RFID j, as well as an estimate of the local displacement between both RFIDs. The local displacement is estimated by a Kalman filter, integrating data from pose tracking (Chapter 3). If an RFID has been detected, the Kalman Filter is reset in order to estimate the *relative* displacement to the subsequent tag on the trajectory. The noisy measurement of the local displacements is denoted by $\hat{d}_{ij} = d_{ij} + \Delta d_{ij}$. It is assumed that the error Δd_{ij} is normally distributed and

thus can be modeled by a Gaussian distribution with zero mean and 3×3 covariance matrix Σ_{ij}. The local displacement $\hat{d}_{ij}$ is defined by the vector $(\Delta\hat{x}_{ij}, \Delta\hat{y}_{ij}, \Delta\hat{\theta}_{ij})^T$, where $\Delta\hat{x}_{ij}$ and $\Delta\hat{y}_{ij}$ denote the relative spacial displacement, and $\hat{\theta}_{ij}$ the relative orientation change.

The central station incrementally builds a global graph from all displacement estimates communicated by the field agents through utilizing the unique ID of RFIDs for data association. The constructed graph $G = (V, E)$ consists of vertices V and edges E, where each vertex represents an RFID tag, and each edge $\left(V_i, V_j\right) \in E$ represents an estimate $\hat{d}_{ij}$ with covariance matrix Σ_{ij} between two RFID tags associated with vertices V_i and V_j, respectively. The graphs from all field agents are unified in the following way: on the one hand, if the same vertex has been observed twice, a loop has been detected on the graph. A detected loop is modeled by a *pseudo* edge between the same RFID node with respect to the RFID antenna's sensor model, which will be further described in Section 4.3.5. On the other hand, if two or more field agents observe the same edge, i.e. their trajectory overlaps between two or more neighboring RFIDs, both observations are merged by an Extended Kalman Filter (EKF) [Maybeck, 1979].

We denote the true pose vectors of $n + 1$ RFID nodes by $l_0, l_1, \ldots, l_n$, and the function calculating the true displacement $(\Delta x_{ij}, \Delta y_{ij}, \Delta\theta_{ij})^T$ between a pair of nodes $\left(l_i, l_j\right)$ is denoted as measurement function d_{ij}. The goal is to find the true locations of the l_{ij} given the set of measurements $\hat{d}_{ij}$ and covariance matrices Σ_{ij}. This can be achieved according to the maximum likelihood concept by minimizing the following Mahalanobis distance:

$$l_{0:n} = \arg\min_{l_{0:n}} \sum_{i,j} \left(d_{ij} - \hat{d}_{ij}\right)^T \Sigma_{ij}^{-1} \left(d_{ij} - \hat{d}_{ij}\right), \tag{4.15}$$

where $l_{0:n}$ denotes the concatenation of poses $l_0, l_1, \ldots, l_n$. Since measurements are taken relatively, it is assumed without loss of generality that $l_0 = 0$ and $l_1, \cdots, l_n$ are relative to l_0. Moreover, the graph is considered as fully connected, and if there does not exist a measurement between two nodes, the inverse covariance matrix Σ_{ij}^{-1} is set to zero.

4.3.2 Linear optimization

If poses are modeled without orientation angle θ, then the optimization problem can be solved linearly. Assume the two connected RFID locations $l_i = (x_i, y_i)^T$ and $l_j = (x_j, y_j)^T$, where l_i follows l_j on the trajectory. Then, Equation 4.15 can be solved by inserting $d_{ij} = l_i - l_j$:

$$l_{0:n} = \arg\min_{l_{0:n}} \sum_{i,j} \left(l_i - l_j - \hat{d}_{ij}\right)^T \Sigma_{ij}^{-1} \left(l_i - l_j - \hat{d}_{ij}\right). \tag{4.16}$$

In order to solve the optimization problem analytically, Equation 4.16 has to be transformed into matrix form. The measurement functions for all edges can be described by

the following matrix equation:

$$\mathbf{d} = \mathbf{hl}, \tag{4.17}$$

where vector $\mathbf{l}$ denotes the concatenation of pose vectors $l_0, \ldots, l_n$ with the dimension nd, vector $\mathbf{d}$ denotes the concatenation of all pose differences $d_{ij} = l_i - l_j$. and matrix $\mathbf{h}$ is an incidence matrix with elements $\{-1, 0, 1\}$. For example, a fully connected graph consisting of the three nodes l_0, l_1, l_2 has the following form:

$$\begin{pmatrix} d_{01} \\ d_{02} \\ d_{10} \\ d_{12} \\ d_{20} \\ d_{21} \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 1 & -1 \\ 0 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} l_1 \\ l_2 \end{pmatrix}. \tag{4.18}$$

Consequently, Equation 4.16 can be rewritten in matrix form by

$$\mathbf{l} = \arg\min_{\mathbf{l}} \left(\hat{\mathbf{d}} - \mathbf{hl}\right)^T \Sigma_{\hat{\mathbf{d}}}^{-1} \left(\hat{\mathbf{d}} - \mathbf{hl}\right), \tag{4.19}$$

where $\mathbf{hl}$ denotes the measurement function as described by Equation 4.17, $\hat{\mathbf{d}}$ denotes the concatenation of observations $\hat{d}_{ij}$, and $\mathbf{\Sigma}_{\hat{\mathbf{d}}}^{-1}$ denotes the inverse covariance matrix of $\hat{d}$, consisting of the inverse sub-matrices Σ_{ij}. Finally, the minimization problem can be solved by

$$\mathbf{l} = \left(\mathbf{h}^T \mathbf{\Sigma}_{\hat{\mathbf{d}}}^{-1} \mathbf{h}\right)^{-1} \mathbf{h}^T \mathbf{\Sigma}_{\hat{\mathbf{d}}}{}^{-1} \hat{\mathbf{d}}\,, \tag{4.20}$$

and the covariance of $\mathbf{l}$ can be calculated by

$$\mathbf{\Sigma_l} = \left(\mathbf{h}^T \Sigma_{\hat{\mathbf{d}}}^{-1} \mathbf{h}\right)^{-1}. \tag{4.21}$$

As shown by Lu and Milios, the computation of Equations 4.20 and 4.21 can be simplified under the assumption of statistical independence between distance measurements d_{ij}. Consequently, the $nd \times nd$ matrix $\mathbf{h}^T \mathbf{\Sigma}_{\hat{\mathbf{d}}}^{-1} \mathbf{h}$ can be replaced with the matrix $\mathbf{g}$, which can be computed by:

$$\begin{aligned} g_{ii} &= \sum_{j=0}^{n} \Sigma_{ij}^{-1}\,, \quad i = 1, \ldots, n, \\ g_{ij} &= -\Sigma_{ij}^{-1}\,, \quad i = 1, \ldots, n\,,\ j = 1, \ldots, n\,, i \neq j. \end{aligned} \tag{4.22}$$

Note that $\Sigma_{ij} = \Sigma_{ji}$, and the $d \times d$ matrices Σ_{ij} have to be invertible. Furthermore, the nd-dimensional vector $\mathbf{h^T \Sigma_l^{-1} \hat{d}}$ can be replaced by the vector $\mathbf{b}$, where the d-dimensional

sub-vectors can be computed by

$$b_i = \sum_{j=0;\ j\neq i}^{n} \Sigma_{ij}^{-1} \hat{d}_{ij}\,, \quad i = 1, \ldots, n. \tag{4.23}$$

Note that $\hat{d}_{ij} = \hat{d}_{ji}$. Finally, the position estimate and covariance can be written as

$$\mathbf{l} = \mathbf{g}^{-1}\mathbf{b} \tag{4.24}$$

$$\mathbf{\Sigma_l} = \mathbf{g}^{-1} \tag{4.25}$$

Equation 4.24 and Equation 4.25 can be solved in $O\left(n^3\right)$, where n is the number of nodes, since the $nd \times nd$ matrix $\mathbf{g}$ has to be inverted. However, in practice, this computation can be performed much more efficiently since matrix $\mathbf{g}$ is a sparse matrix. This is due to the fact that only a little amount of nodes in the graph are connected with each other*. Therefore, most covariances Σ_{ij} are set to zero. There are numerous packages available that have been developed for the efficient computation of sparse matrices, as for example the *Sparse matrix multiplication package (SMMP)* [Bank and Douglas, 1993].

4.3.3 Non-linear optimization

The method described in Section 4.3.2 requires that the measurement functions d_{ij}, and the measurements $\hat{d}_{ij}$ are linear. However, particularly in the context of robot localization, poses are typically modeled with an angular component, e.g. $l_i = (x_i, y_i, \theta_i)^T$. One "trick" in Kalman filtering based methods is to linearize the measurement function with a Taylor series. After the linearization, Equations 4.24 and 4.25 from Section 4.3.2 can be applied in the same way as if the measurement function would be linear †. Therefore, the goal is to determine the linearized measurement functions d'_{ij} and linearized measurements $\hat{d}'_{ij}$, in order to solve the optimization problem by Equation 4.24. Note that the measurements $\hat{d}'_{ij}$ are already linearized if they have been computed by EKF-based pose tracking, as described in Chapter 3. In the following, the linearization of the measurement function will be discussed.

Assume the two connected locations $l_i = (x_i, y_i, \theta_i)^T$ and $l_j = (x_j, y_j, \theta_j)^T$ with the local displacement $d_{ij} = (\Delta x, \Delta y, \Delta\theta)^T$, where l_i follows l_j on the trajectory. Then, the displacement between these locations can be determined by the following transforma-

*In the context of robot navigation node connections are naturally constrained by the environment.

†Under significant angular changes the linearization procedure might lead to errors, requiring the optimization process to be applied iteratively.

tion [Smith and Cheeseman, 1986]:

$$\Delta x = (x_i - x_j)\cos\theta_j + (y_i - y_j)\sin\theta_j \tag{4.26}$$
$$\Delta y = -(x_i - x_j)\sin\theta_j + (y_i - y_j)\cos\theta_j \tag{4.27}$$
$$\Delta\theta = \theta_i - \theta_j, \tag{4.28}$$

which will be denoted by

$$d_{ij} = l_i \ominus l_j. \tag{4.29}$$

Let $\Delta d = \hat{d}_{ij} - d_{ij}$ denote the observation error, which is a function of the two poses l_i and l_j since $d_{ij} = l_i \ominus l_j$. It is assumed that the estimates $\hat{l}_i = \left(\hat{x}_i, \hat{y}_i, \hat{\theta}_i\right)$ and $\hat{l}_j = \left(\hat{x}_j, \hat{y}_j, \hat{\theta}_j\right)$ of these poses are given from pose tracking. Then, the observation error can be linearized by the following Taylor series [Lu and Milios, 1997]:

$$\Delta d_{ij} \;\approx\; \hat{d}_{ij} - (\hat{l}_i \ominus \hat{l}_j) + \hat{K}_j^{-1}(\Delta l_i - \hat{H}_{ij}\Delta l_j), \tag{4.30}$$

where

$$\hat{K}_j^{-1} = \begin{pmatrix} \cos\hat{\theta}_j & \sin\hat{\theta}_j & 0 \\ -\sin\hat{\theta}_j & \cos\hat{\theta}_j & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{4.31}$$

$$\hat{H}_{ij} = \begin{pmatrix} 1 & 0 & -\hat{y}_i + \hat{y}_j \\ 0 & 1 & \hat{x}_i - \hat{x}_j \\ 0 & 0 & 1 \end{pmatrix}, \tag{4.32}$$

$$\Delta l_i = \hat{l}_i - l_i;\; \Delta l_j = \hat{l}_j - l_j \tag{4.33}$$

If replacing matrix $\hat{H}_{ij}$ with the product $\hat{H}_i^{-1}\hat{H}_j$, Equation 4.30 can be transformed to

$$-\hat{H}_i\hat{K}_j\Delta d_{ij} \;=\; \hat{H}_i\hat{K}_j((\hat{l}_i \ominus \hat{l}_j) - \hat{d}_{ij}) - (\hat{H}_i\Delta l_i - \hat{H}_j\Delta l_j), \tag{4.34}$$

which is equal to

$$\Delta d'_{ij} = \hat{d}'_{ij} - d'_{ij}, \tag{4.35}$$

if substituting the following terms

$$\Delta d'_{ij} = -\hat{H}_i\hat{K}_j\Delta d_{ij} \tag{4.36}$$
$$\hat{d}'_{ij} = \hat{H}_i\hat{K}_j((\hat{l}_i \ominus \hat{l}_j) - \hat{d}_{ij}) \tag{4.37}$$
$$d'_{ij} = \hat{H}_i\Delta l_i - \hat{H}_j\Delta l_j. \tag{4.38}$$

Hence, the linearized measurement function d'_{ij} with covariance matrix Σ'_{ij} is given by:

$$d'_{ij} = \hat{H}_i \Delta l_i - \hat{H}_j \Delta l_j \tag{4.39}$$

$$\Sigma'_{ij} = \hat{H}_i \hat{K}_i \Sigma_{ij} \hat{K}_j^T \hat{H}_i^T \tag{4.40}$$

$$\tag{4.41}$$

which can be utilized for formulating the linearized optimization problem

$$\mathbf{l} = \arg\min_{\mathbf{l}} \sum_{i,j} \left(d'_{ij} - \hat{d}'_{ij}\right)^T \Sigma'^{-1}_{ij} \left(d'_{ij} - \hat{d}'_{ij}\right), \tag{4.42}$$

and can be solved in the same way as proposed in Section 4.3.2. Since the linearization leads to errors, the procedure has to be applied iteratively, where the result from the nth computation serves as a starting point for the $n+1$th computation. After the optimization has been performed, the coordinates of the optimal poses can be determined as follows: assume l'_i and Σ'_i are the poses and covariances resulting from the linearized optimization. Then, the final locations of the corrected poses can be determined by:

$$l_i = \hat{l}_i - \hat{H}_i^{-1} l'_i \tag{4.43}$$

$$\Sigma_i = (\hat{H}_i^{-1}) \Sigma'_i (\hat{H}_i^{-1})^T \tag{4.44}$$

4.3.4 Trajectory interpolation

The corrected RFID network can be used as a basis for correcting the odometry trajectory of each agent. This is carried out by utilizing the corrected RFID locations as constraint points for the correction of the trajectory. One possibility to do this, is to augment the noisy odometry trajectories with these constraints, e.g. by connecting them with *pseudo edges*, and to correct them once more with the method that has been described in the last section. However, since these constraint points are already globally consistent, it turns out to be much more efficient to perform a local interpolation of trajectory poses between the corrected RFIDs.

Given a sequence of corrected RFID locations $r_{1:t}$ and an uncorrected trajectory $x_{1:t}$ the corrected trajectory $y_{1:t}$ is computed by interpolating each pose $x_k \in x_{1:t}$ between the two closest RFIDs. In order to correct each pose x_k of the trajectory, we first determine the corrected RFID locations r_i and r_j of the two closest RFIDs before and after the pose, with $j \geq k \geq i$, and the uncorrected poses x_i and x_j at corresponding time, respectively. Finally, the corrected pose is computed by:

$$y_k = x_k - \frac{\left(w_1 (x_i - r_i) + w_2 \left(x_j - r_j\right)\right)}{w_1 + w_2}, \tag{4.45}$$

where weights w_1 and w_2 are computed by $w_1 = \left|r_j - x_k\right|$ and $w_2 = |r_i - x_k|$. Experimental results in Section 4.5 will show that this method together with the RFID graph optimization allows to efficiently and robustly correct large trajectories.

4.3.5 Sensor model

Each time a loop has been detected on the trajectory, i.e. an RFID has been observed twice, a *pseudo* edge is added to the corresponding RFID node. We model this edge by accounting for the spatial expansion of the utilized RFID antenna.

For experiments, two different RFID antennas, one for humans and one for robots, were utilized. The antenna of the robot was mounted in parallel to the ground allowing to detect RFIDs lying beneath the robot, whereas the antenna of the human was carried manually. An evaluation of the robot RFID system has shown that RFID tags lying beneath the robot within the antenna's expansion (a rectangle of $\approx 20 \times 15\,cm$), are constantly detected, also if the robot travels with high velocity. The antenna carried by the human detects RFIDs within a range of approximately $30\,cm$ if the antenna is aligned to the bearing of the RFID tag. However, it is not possible to tell the exact position of the detection within this range. Consequently, the observation uncertainty is modeled according to the maximal detection range.

It is assumed that, in the average case, RFIDs are detected within the antenna's center, and that RFID detections can occur at arbitrary orientations of the human or robot. Therefore, a "loop-closing" edge is modeled with distance $\hat{d}_{ii}$ set to $(0, 0, \Delta\theta)^T$, where $\Delta\theta$ denotes the angle difference between the two pose estimates of the RFID. Under the assumption that RFIDs are detected at 95% probability if they are within maximal reading range d_M, we model the according covariance matrix by:

$$\Sigma_{ii} = \begin{pmatrix} 2^2 d_M^2 & 0 & 0 \\ 0 & 2^2 d_M^2 & 0 \\ 0 & 0 & \sigma_\theta^2 \end{pmatrix}, \tag{4.46}$$

where σ_θ^2 is the linearized variance of the angle, and $2^2 d_M^2$ the variance of the normal distribution over the interval $[-2d_M; 2d_M]$. Note that σ_θ^2 has to be linearized according to the procedures described in Section 4.3.3. Finally, The robot antenna was modeled with $d_M = 20\,cm$, and the antenna carried by the human with $d_M = 30\,cm$.

4.4 Decentralized RFID-SLAM (DRFID-SLAM)

In emergency response scenarios, communication links between agents or to a central station might be cut-off temporarily or constantly. One obvious reason is that in building structures radio waves are not able to penetrate walls if they are made of reinforced

concrete. Otherwise, it might also be likely that first responders, which search for victims on large terrain, are simply out of communication range. Moreover, they might visit the same location at different time, and thus are also not able to communicate. Therefore, methods deployed in this context have to be robust towards total loss of communication. Decentralized RIFD-SLAM (DRFID-SLAM) is particularly designed for situations in which radio communication is impossible. Note that this distinguishes the approach from other methods that require at least *low-bandwidth* communication channels between the agents [Reece and Roberts, 2005, Nettleton et al., 2003, Kim et al., 2004]. Furthermore, we assume that RFIDs have sufficient memory for storing RFID graph structures as described in Section 4.3.

The basic idea of DRFID-SLAM is to utilize the memory of RFID tags for learning the topology of the surrounding RFID graph. When agents pass a tag, they update their knowledge on the graph topology by reading data from the tag's memory, as well as synchronize the tag's memory by writing graph data they have collected on their trajectory. By this, agents are able to learn a graph larger than their own trajectory, i.e. extended by trajectories of other agents. The union of single agent trajectories leads to graphs that contain loops, hence allowing agents to correct their trajectories, even if they do not contain loops. Figure 4.2 depicts an example of DSLAM of three agents successively passing RFIDs. The third agent is finally able to close a loop consisting of partial tracks from the first and second agent.

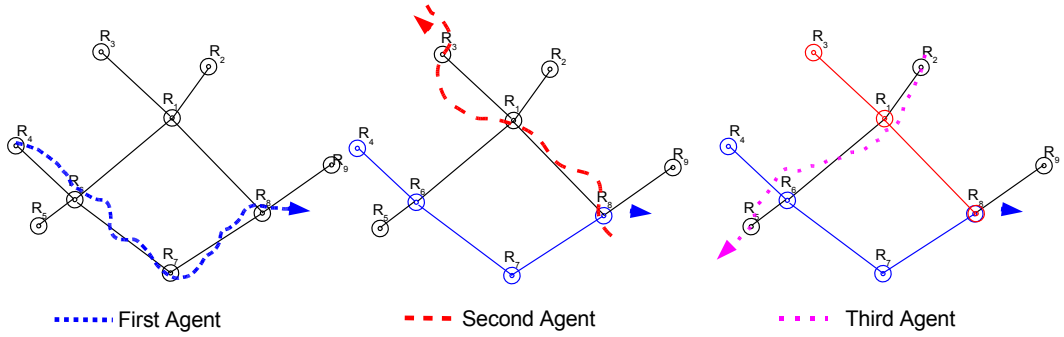


Figure 4.2: Example for decentralized SLAM of three agents: RFID nodes $R_1, R_2, \cdots, R_9$ successively learn the trajectories of the passing agents. The third agent learns at node R_1 the trajectories from the first and second agent, enabling loop-closure for map improvement when reaching node R_6.

The procedure can be considered as a step-wise information propagation through the network of RFIDs, where each visited RFID learns about nearby RFID nodes from the trajectory of its visitor, as well as from trajectories that were previously stored by other agents in the nodes passed by the visitor. Therefore, after a sufficient amount of trajectories has been traveled, a single RFID node memorizes the whole infrastructure

of the network. Note that for supporting map improvements, RFIDs do not necessarily have to store the whole network structure.

One problem arising in the context of distributed SLAM is the double counting of information propagated via different paths, i.e. rumor propagation. While this problem typically requires to apply probabilistic methods, such as Bounded Covariance Inflation (BCI) [Reece and Roberts, 2005], and Covariance Intersection (CI) [Nettleton et al., 2003], it can be solved rather trivially with RFIDs. Here the basic idea is to manage displacement estimates locally in the memory of RFIDs, i.e. to distinguish each estimate by location and time where it has been observed.

We distinguish edges as *managed* and *propagated* in the memory of RFIDs and introduce a counter C_i attached to each RFID, which is locally updated each time an agent writes information concerning a managed edge. Edges that are managed by node i are all the adjacent edges that have i in common, i.e. those edges that directly connect to i. Consider the case that an agent travels from RFID i to RFID j and generates thereby a new estimate $\left(\hat{d}_{ij}, \Sigma_{ij}\right)$. Then, RFID j is the managing node, i.e. manages the edge between i and j. Consequently, the agent stores in RFID j a new entry $e_{ij} = \left\langle i, j, \hat{d}_{ij}, \Sigma_{ij}, C_i \right\rangle$ consisting of the IDs of the two visited RFIDs i and j, their displacement estimate, and the current count value of the RFID, which is incremented after writing. Obviously, there can be two nodes managing the same edge since agents can either travel from i to j or from j to i. Therefore, we have to distinguish both cases during propagation, which is solved by defining $\langle i, j, \cdots \rangle \neq \langle j, i, \cdots \rangle$, i.e. the managing node corresponds to the second node in the sequence.

Algorithm 2: Updating the graph R_k of visited RFID k

Input: New entry e_{ik}, a set of graphs $S = \left\{R_j : j \in P\right\}$, where P denotes the RFID trajectory of the agent
Output: Updated graph R_k

Update managed edges:
$R_k \leftarrow R_k \cup \left\langle i, k, \hat{d}_{ik}, \Sigma_{ik}, C_k \right\rangle$;
$C_k \leftarrow C_k + 1$;

Update propagated edges:
foreach $R_j \in S$ **do**
  foreach $e \in R_j$ **do**
    if $e \in R_k$ **then**
      continue;
    $R_k \leftarrow R_k \cup e$;
  end
end

Propagated edges are managed edges after they have been copied by agents. If visiting a node, the agent performs a union operation between all entries stored in the memory of the RFID, denoted by graph R_i, and its own graph, denoted by A_i, and propagates

them further to other RFIDs. The double counting of identical estimates is prevented by ensuring that there always exists only one unique entry for each combination of i, j, and C_j and unique sequence $< i, j >$ within each RFID, where C_j is the counter value of RFID j during the creation of the managed edge. The complete procedure for updating RFID nodes is shown by Algorithm 2. Note that the agents update their own graphs A_i by the same procedure, while considering propagated edges only.

The decentralized information exchange facilitates the building of RFID graph structures as described in Section 4.3. Hence, agents are able to improve their trajectories if they detect a new loop after performing the union procedure. However, the possibility to close loops from joining trajectories depends on the number of agents that visit a node, as well as on the sequence trajectories have been traveled. For example, in the situation depicted in Figure 4.2, the first agent does not benefit from the tracks of the others. The performance of the proposed approach can be further improved by extending the information propagation via radio communication. Each time agents are within communication range, they can unify their graphs, i.e. to exchange propagated edges, by the procedure described in Algorithm 2. Thereby, they are possibly able to connect large graphs, accelerating the map joining process.

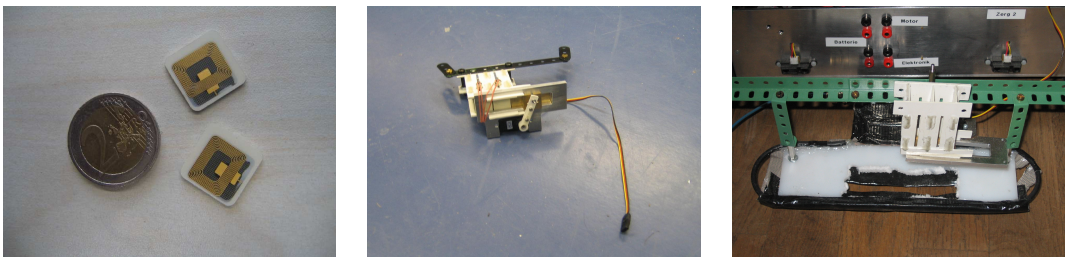
4.5 Experimental results

Experiments on centralized and decentralized RFID-SLAM were conducted with a team of humans, a team of robots, and finally jointly by humans and robots. All experiments were carried out in both indoor and outdoor scenarios, where the *Zerg* robot platform, described in Chapter 2, has been deployed.

4.5.1 Experimental setup

The robot platform utilized for experiments is equipped with an RFID system consisting of an RFID reader and antenna. The active distribution of RFID tags is carried out by a custom-built actuator based on a metal slider that can be moved by a conventional servo (Figure 4.3(b)).

The slider is connected with a magazine that maximally holds around 50 tags. Each time the mechanism is triggered, the slider moves back and forth while dropping a single tag from the magazine. The device is constructed in a way that for each trigger signal only one tag is released. A software module triggers the device at adequate locations, which are determined according to the existing density of RFIDs, i.e. maintaining a maximal defined density of RFIDs, and also, if operating in indoor scenarios, with respect to the structure of the environment. For example, narrow passages, such as doorways and corridors, are likely to be passed by the robot, thus RFIDs are deployed with high probability in these kind of environmental structures. The width of



(a) (b) (c)

Figure 4.3: A novel mechanism for the active distribution of RFID tags. (a) The utilized RFID tags. (b) The mechanism with servo. (c) The mechanism together with the antenna mounted on the *Zerg* robot.

the free space surrounding the robot is computed from the distance between the obstacles located most left and most right to the robot. They are found on a line which goes through the center of the robot, and which is orthogonal to the robot's orientation.

The antenna of the reader is mounted parallel to the ground, allowing to detect RFIDs lying beneath the robot. In order to enable the robot to perceive the deployment of RFIDs, the deploy device has been mounted above the RFID antenna, forcing deployed RFIDs to pass directly through the antenna. We utilized Ario RFID chips from *Tagsys* (Figure 4.3 (a)) with a size of $1.4 \times 1.4\,cm$, 2048 *Bit* RAM, and a response frequency of 13.56 MHz. For the reading and writing of these tags, we employed a Medio S002 reader, likewise from *Tagsys*, which operates within a range of approximately 30 cm while consuming less than $200\,mA$. Figure 4.3 (c) shows the complete construction consisting of deploy device and antenna mounted on the *Zerg* robot. For a detailed description of the RFID sensor model see Section 4.3.5.

Pose tracking on the *Zerg* platform was carried out by slippage-sensitive odometry, as described in Section 3.3, and the *Xsens MTi* IMU, as described in Chapter 2. For pose tracking on pedestrians, we utilized the *MTx* IMU from *Xsens*, which has been attached to a test person for measuring the vertical and lateral acceleration, as well as the azimuth orientation (see the orange box in Figure 3.7 (a)). Based on an empirical evaluation, we modeled pose tracking uncertainty on pedestrians with $\sigma^2_{\hat{d}} = (0.05\,m)^2 d$, and $\sigma^2_{\hat{\theta}} = (15°)^2$, and on robots with $\sigma_d = 0.816\frac{cm}{m}$ and $\sigma_{d_{slip}} = 24.72\frac{cm}{m}$.

During outdoor experiments, ground truth data was obtained by a GNSS device. For this purpose, we used the *GPSlim236* GPS receiver from *Holux*, which is equipped with *SIRFstar III* technology. The receiver allows to track up to 20 satellites at an update rate of 1 Hz and has a position accuracy of 5 – 25 meters without DGPS. Furthermore, the receiver is able to process data from the *EGNOS* system [‡], improving the horizontal

[‡]EGNOS stands for European Geostationary Navigation Overlay Service and provides Differential GPS (DGPS) via satellites that broadcast correction signals on the same frequency as GNSS satellites.

position accuracy towards < 2.2 meters and vertical position accuracy towards < 5 meters at 95% of the time. Note that we were able to also obtain ground truth in semi-indoor scenarios since RFIDs where intentionally placed outdoors.

4.5.2 RFID-SLAM with robots

The described method for RFID technology-based SLAM was tested extensively with data generated by a simulator [Kleiner and Buchheim, 2003] and on the *Zerg* robot platform. The simulated robot explored three different building maps, the *small* map, *normal* map, and *large* map of the sizes 263 m^2, 589 m^2 1240 m^2, while automatically distributing RFID tags in the environment. Figure 4.4 (a-c) shows the averaged results from 100 executions of RFID-SLAM on the three maps at five different levels of odometry noise. We measured a computation time of 0.42 seconds on the small map, 2.19

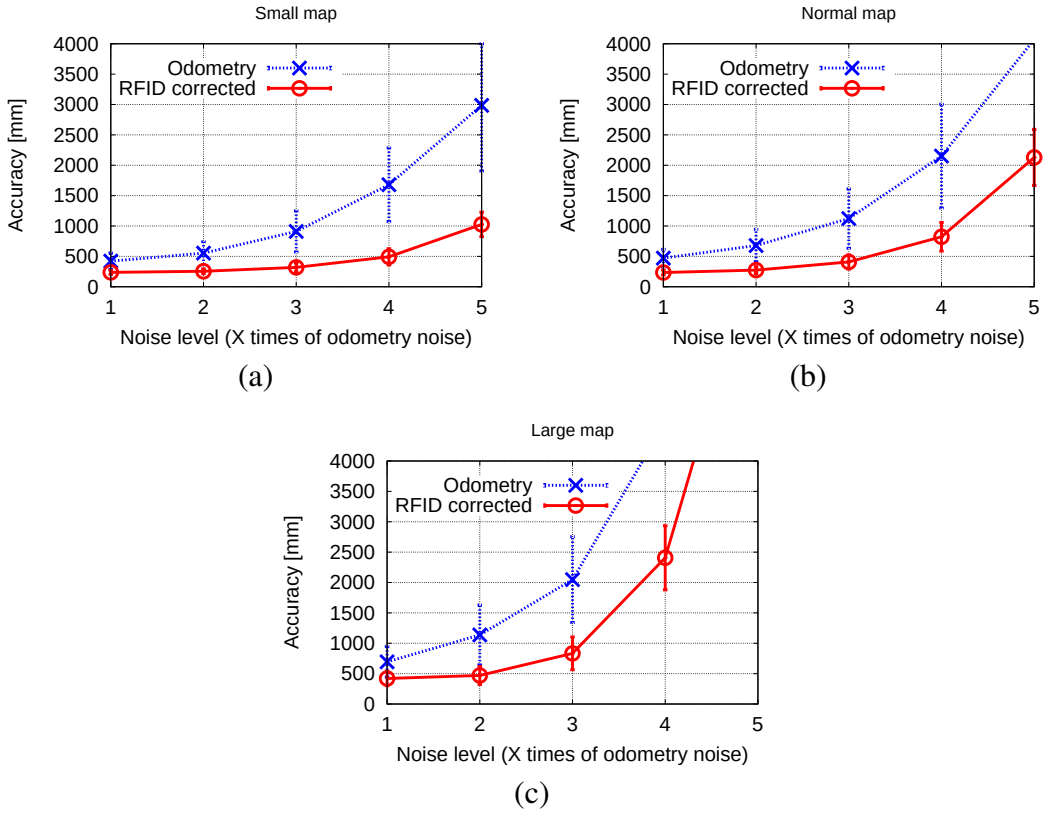


Figure 4.4: (a) - (c) Result from applying RFID-based SLAM at different levels of odometry noise on (a) the small map (263 m^2), (b) the normal map (589 m^2), and (c) the large map (1240 m^2).

seconds for the normal map, and 13.87 seconds for the large map, with a Pentium4

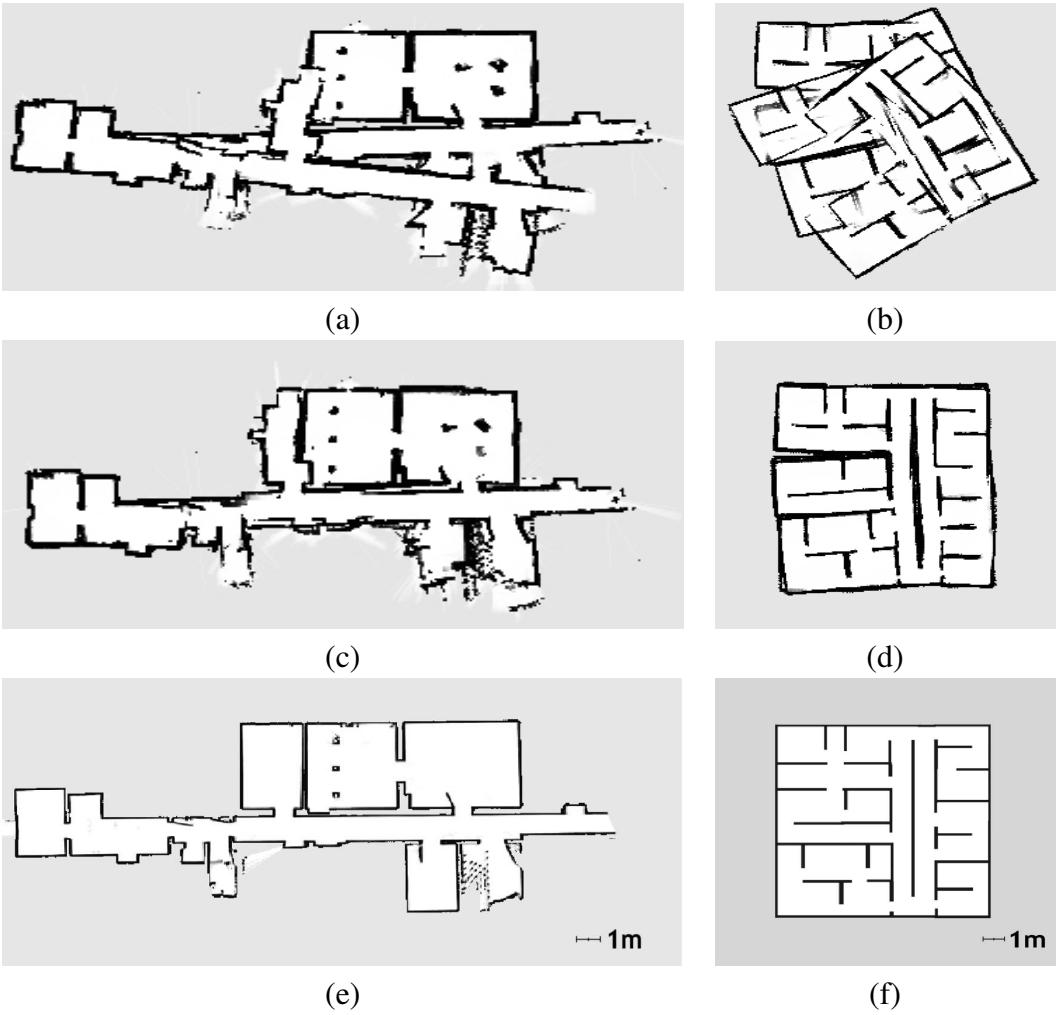


Figure 4.5: (a,c,e) Result from RFID-SLAM of a robot driving in a cellar: (a) the noisy map, (c) the corrected map, and (e) the ground truth created by iterative scan matching. (b,d,f) Result from applying the RFID-SLAM to data generated in the simulation. (b) The small map with odometry noise, (d) the corrected map, and (f) the ground truth taken directly from the simulator's map editor. Note that the cellar's ground truth map displays unoccupied rooms. The rectangular structures that can be seen in the upper left room in the constructed maps (a) and (c) originate from crates stored in those rooms.

2.4 GHz. The small map before and after the correction is shown in Figure 4.5 (b,d). To achieve this result, the robot distributed approximately 50 RFIDs.

In order to evaluate the performance of RFID-SLAM in a real environment, we collected data from a robot autonomously exploring a cellar for 20 minutes while detecting RFID tags on the ground. The robot continuously tracked its pose as described in Sec-

tion 3.3. As depicted in Figure 4.5, the non-linear method successfully corrected the angular error based on RFID data association. The correction was based on approximately 20 RFID tags.

Figure 4.6: Result from applying RFID-SLAM outdoors while driving with $1m/s$ on a parking lot. Trajectories are visualized with GoogleEarth, showing RFID locations estimated by the odometry (red), the ground truth (blue), and corrected by RFID-SLAM (green).

Additionally, we conducted an outdoor experiment with the *Zerg* robot driving with an average speed of 1 m/s on a parking lot. The odometry was generated from the wheel encoders (translation) and IMU (rotation). Furthermore, the robot detected RFIDs with the antenna described above. We obtained position ground truth from both Differential GPS (DGPS) and manual measurements, whereas faulty GPS positions, e.g. due to multi-path propagations close to buildings, were corrected from the manual measurements. Figure 4.6 shows the RFID locations estimated by the odometry (red), the ground truth (blue), and corrected by RFID-SLAM (green). The corrected trajectory has a mean Cartesian error of 1.8 ± 3.1 m, compared to the uncorrected trajectory, which has a mean Cartesian error of 8.3 ± 8.5 m.

Furthermore, we evaluated the influence of the number of detected RFIDs with respect to the stability of the SLAM approach. Figures 4.7 (a), (b), and (c) show the Cartesian error, the cross-track error (XTE), and the along-track error (ATE) at decreasing number of RFIDs, respectively. The route graph optimization consistently improves the accuracy of the trajectory, even with a comparably little number of RFIDs, e.g.

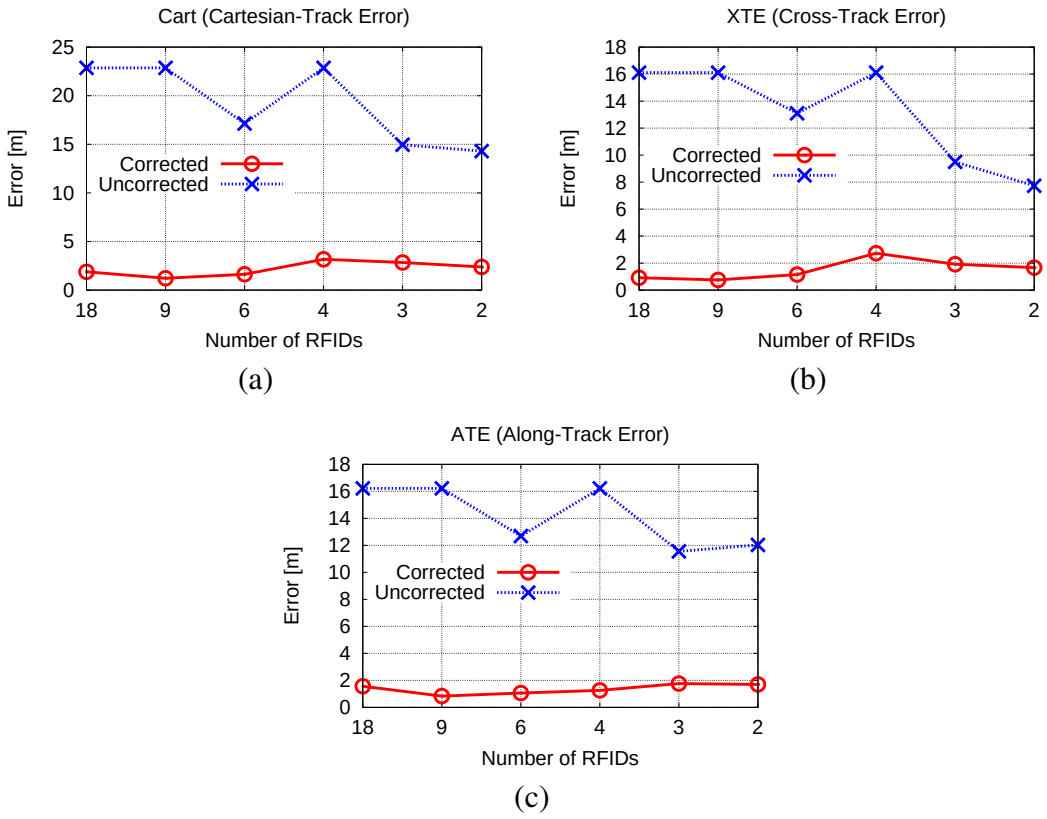
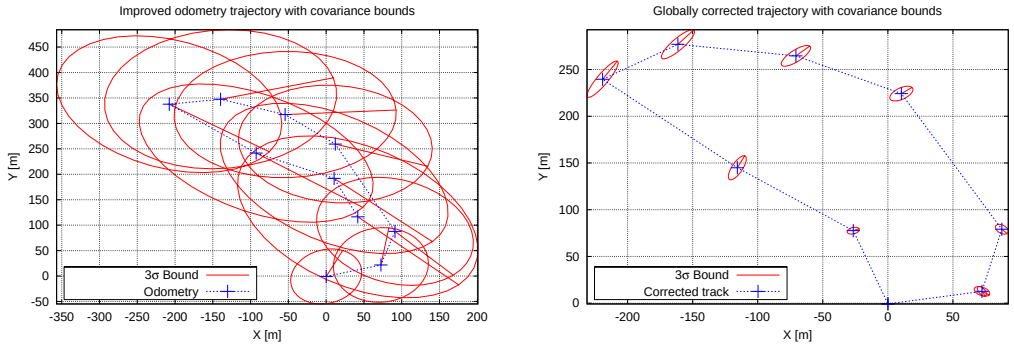


Figure 4.7: Result from applying RFID-SLAM outdoors while driving with 1 m/s for approximately 1 km on a parking lot with the *Zerg* robot. (a) the Cartesian error, (b) the cross-track error (XTE), and (c) the along-track error (ATE) with respect to the number of utilized RFIDs.

one RFID each 500m. The correction of 18 RFIDs took 2.1 seconds on a PentiumM 1.7 MHz, and the interpolation of the odometry trajectory took 0.2 seconds.

In order to evaluate the scalability of the approach in large-scale environments, a second outdoor experiment was conducted. During this experiment, the robot was driving a total distance of more than 2.5 km with an average speed of 1.58 m/s. Note that this velocity requires human beings to walk comparably fast in order to follow the robot. Furthermore, the robot was heavily shaking from fast navigation over uneven ground, such as road holes, small debris, and grass. Also during this experiment, pose tracking was carried out based on data from the wheel encoders (translation) and IMU (rotation), and position ground truth has been obtained from DGPS. The optimization yielded an average Cartesian error of 9.3 ± 4.9 m, compared to the uncorrected trajectory, which has an average Cartesian error of 147.1 ± 18.42 m. The correction of 10 RFIDs took 0.3 seconds on a PentiumM 1.7 MHz, and the interpolation of the odometry trajectory took



(a) (b)

Figure 4.8: Covariance bounds from applying RFID-SLAM outdoors: (a) before and (b) after the correction.

Figure 4.9: Result from applying RFID-SLAM outdoors while driving with 1.58 m/s for more than 2.5 km with a *Zerg* robot showing the odometry trajectory (red), the ground truth trajectory (blue), and the corrected RFID trajectory (green).

2.4 seconds. Figure 4.8 shows the covariance bounds during EKF-based dead reckoning of the improved odometry (a), and after the global optimization (b). Figure 4.9 shows the trajectory of the odometry (red), the ground truth (blue), and the corrected RFID graph (green). Note that for the sake of readability, Figure 4.8 only shows the first loop of the performed trajectory.

4.5.3 RFID-SLAM with humans

In this section experiments conducted by a team of humans and a human-robot team are shown. Note that multiple trajectories were performed by a single person and data was recorded for later processing.

Semi-Indoor Experiment

The semi-indoor experiment was carried out on the campus of the University of Freiburg, including many accessible buildings which were entered by the test person. We measured that some of these buildings contain magnetic fields disturbing the angle estimate of the PDR method, as for example, metal stairs or metal doors, such as shown in Figure 3.7. Figure 4.10 provides an overview of the area, which was generated by *GoogleEarth*. During this experiment, the test person traveled six trajectories with different starting and ending locations, while performing pose tracking with the PDR method described previously, and while distributing and re-observing RFIDs. In order to visualize the PDR trajectories, we utilized the accurate starting locations taken from the ground truth data and projected each PDR trajectory with respect to its starting location on the map. In Figure 4.10 each trajectory is shown with a different color. Position accuracy decreases with increasing length of the traveled trajectory.

All trajectories were collected and merged into a single graph for applying the centralized method described in Section 4.3. The corrected edges between RFIDs are shown by the black lines in Figure 4.10, as well as the corrected locations of the RFIDs (small squares). Furthermore, we computed the average Cartesian error with respect to ground truth. Figure 4.11 depicts these errors according to each pedestrian. It shows the uncorrected trajectory, the single trajectory corrected on its own, and the trajectory corrected from the unified graph. The correction based on the unified graph yields best results for each pedestrian.

Outdoor Experiment

The outdoor experiment was carried out in a larger urban area, containing mainly residential buildings with up to six floors. Due to the multipath propagation problem in this environment, e.g. narrow streets and high buildings, the recorded GPS tracks had to be manually corrected for providing ground truth data. Figure 4.12 (a) depicts the recorded

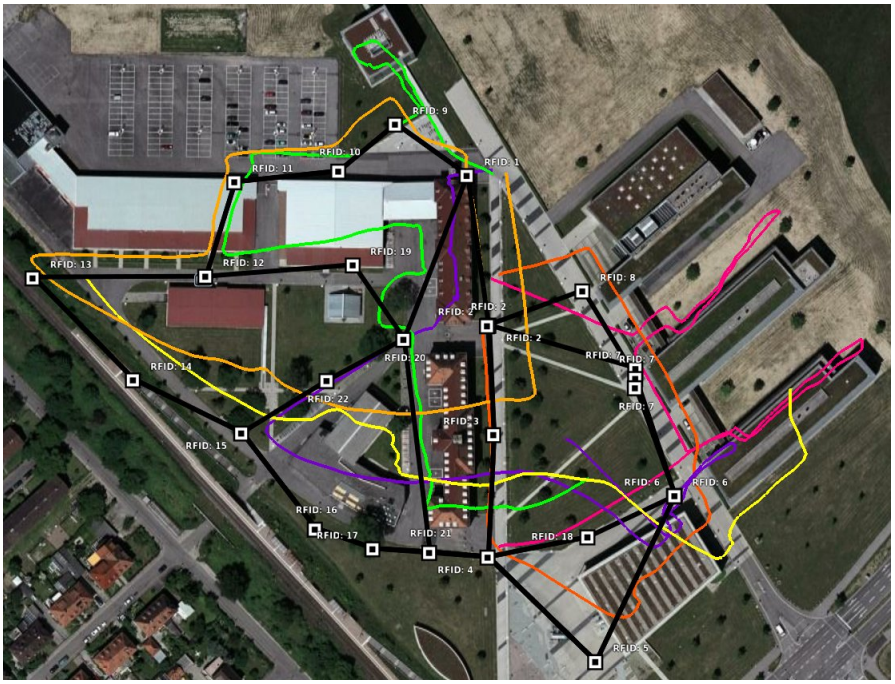


Figure 4.10: Result from the semi-indoor experiment: each line indicates the pose tracking of the trajectory of a single pedestrian. Black lines and squares show the corrected graph of RFIDs.

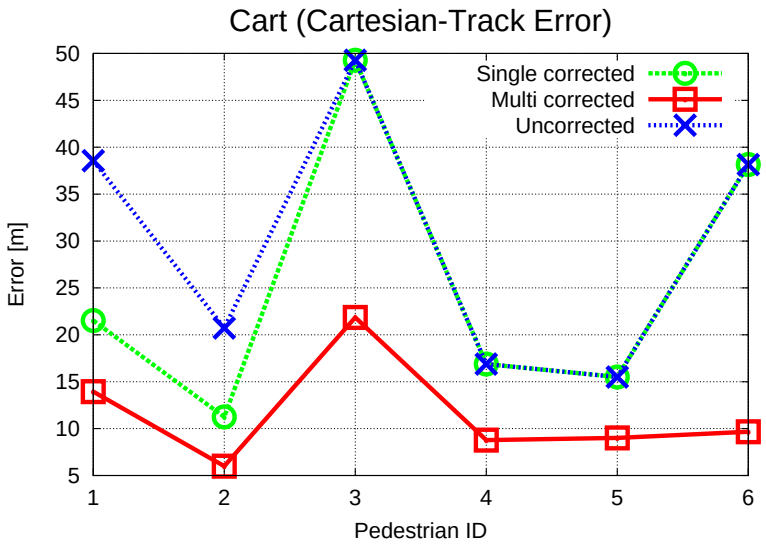


Figure 4.11: Result from the semi-indoor experiment: the average Cartesian error.

PDR trajectories. During this experiment, the pedestrians walked approximately 9000 meters, while deploying 20 RFID tags. Figure 4.12 (b) depicts the uncorrected RFID graph constructed by centrally merging all PDR trajectories and RFID observations, and Figure 4.12 (c) shows the graph after the correction. It can be seen in Figure 4.12 (d) that the corrected graph is close to ground truth. We also computed the average Cartesian error with respect to ground truth, shown in Figure 4.13. Also during this experiment the correction based on the unified graph yields better results for each single pedestrian.

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

Figure 4.12: Result from correcting pedestrian trajectories in an urban environment (the City of Freiburg). (a) Pedestrian trajectories, where each color indicates the recorded track of a single pedestrian. (b) The uncorrected RFID graph, generated from the PDR tracks. (c) The corrected RFID graph. (d) Manually improved GPS ground truth.

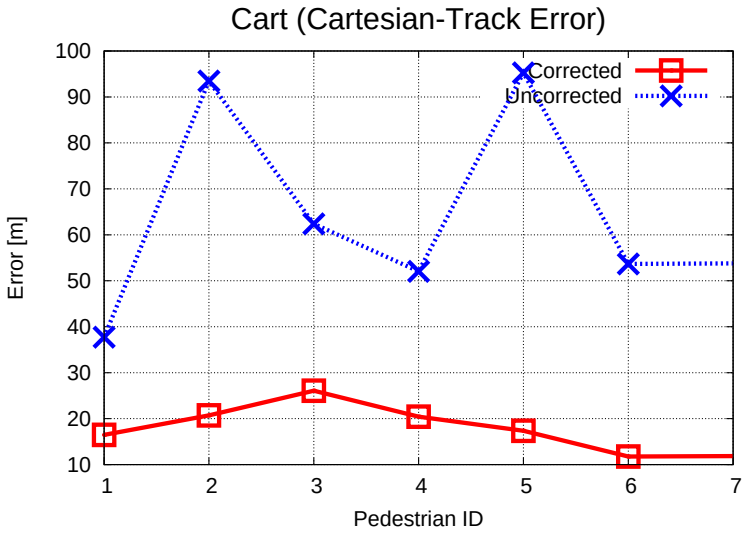


Figure 4.13: Result from the outdoor experiment: the average Cartesian error.

Decentralized SLAM Experiment

The last experiment was conducted in order to evaluate the decentralized version of RFID-SLAM. This was carried out by simulating the incremental execution of trajectories recorded during the outdoor experiment previously described. We executed all 720 possible sequences of six trajectories and computed the mean Cartesian error. After the execution of each trajectory the graph learned by the according agent and the graphs learned by passed RFIDs were computed. Therefore, the first pedestrian had no advantage at all, the second pedestrian benefited from data propagated into RFIDs by the first, and the third benefited from data propagated by the first and the second, and so on. Note that graphs learned by the RFIDs do not necessarily consist of complete trajectories, instead they contain the union of trajectories from the agents until they synchronized this node. Figure 4.14 depicts the average Cartesian error after correcting the trajectory of each pedestrian based on the graph generated from the union of all visited RFIDs. The more pedestrians traveled through the environment, the better the pose estimate of a single pedestrian can be improved. Hence, the last pedestrian was able to achieve a nearly two times higher pose accuracy than the first one by closing loops that were constructed by joining graphs from other pedestrians.

4.5.4 RFID-SLAM jointly with humans and robot

In another experiment we jointly corrected the odometry trajectories from a human and a robot exploring the same area while detecting RFIDs. During this experiment, pedestrian and robot performed pose tracking with the PDR method and slippage sensitive

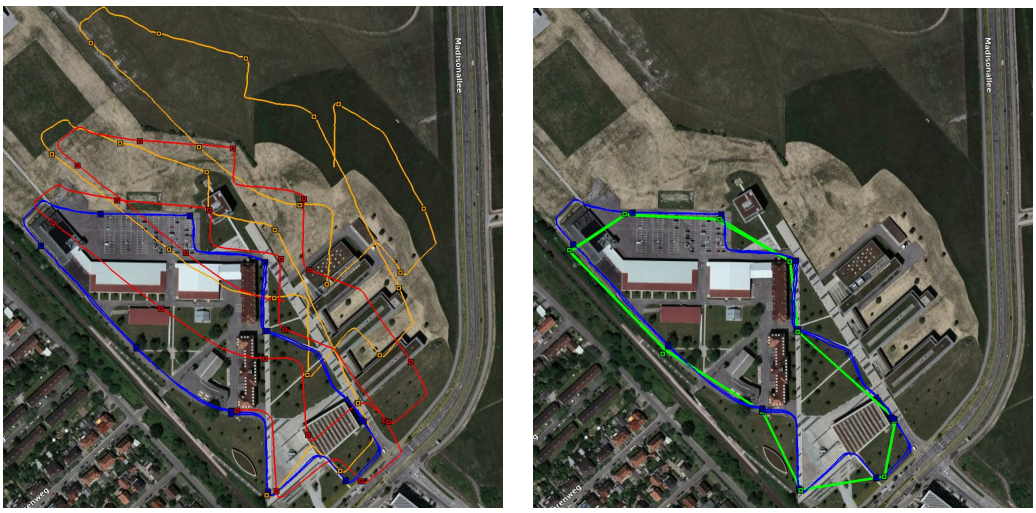
Figure 4.14: Result from decentralized SLAM: Average Cartesian pose error of the i-th pedestrian walking after his predecessors which left information locally in RFIDs. Averages are computed from all possible sequences of six pedestrians.

odometry described in Chapter 3. For an area of approximately 900 m^2, 10 RFIDs were used. Table 4.1 depicts the average Cartesian error, the average cross-track error (XTE),

	Cart. Err. [m]	XTE Err. [m]	ATE Err. [m]
Rob. Odo.	147.10 ± 36.85	139.59 ± 35.99	46.11 ± 20.69
Ped. Odo.	56.63 ± 24.38	44.22 ± 24.52	32.31 ± 15.82
Ped. Corr.	14.27 ± 12.7	8.40 ± 11.67	10.68 ± 10.84
Rob. Corr.	9.37 ± 9.90	5.57 ± 9.55	6.52 ± 9.23
Both Corr..	5.64 ± 4.77	2.50 ± 4.23	4.33 ± 4.50

Table 4.1: Average positioning errors of odometry, single, and joint correction.

and average along-track error (ATE) of the original robot odometry (Rob. Odo.), the original pedestrian odometry (Ped. Odo.), their single corrected trajectories (Ped. Corr, Rob Corr.), and the jointly corrected trajectory (Both Corr.). The simultaneous correction of both trajectories improved the accuracy significantly. Figure 4.15 (a) shows both odometry trajectories compared to ground truth (blue line), and Figure 4.15 (b) shows the corrected RFID graph (green line). The small squares indicate RFID observations.



(a) (b)

Figure 4.15: Result from correcting trajectories from robot odometry (orange line) and pedestrian odometry (red line) jointly: The corrected RFID graph (green line) lies close to ground truth (blue line). Small squares indicate RFID observations.

4.6 Related work

During the last decade, methods for SLAM have been applied in many different scenarios, for example in outdoor [Bailey, 2002, Ramos et al., 2007] and underwater environments [Newman et al., 2003]. Inspired by the fundamental work of Smith et al. [Smith et al., 1988], early work on SLAM was mainly based on the Extended Kalman Filter (EKF) [Dissanayake et al., 2001] which updates the state vector after each measurement in $O\left(n^2\right)$. Based on the observation that landmark estimates are conditionally independent given the robot's pose, Montemerlo et al. introduced *FastSLAM* which reduces the computational complexity of EKF-based SLAM to $O(nk)$, where k is the number of robot trajectories considered at the same time [Montemerlo et al., 2002]. The framework has been further extended to using evidence grids [Hähnel et al., 2003, Grisetti et al., 2005]. Stachniss et al. introduced a method for improving map quality gained by *FastSLAM* during exploration by inducing actions on the robot for actively closing loops, e.g. to cause the robot to re-enter already visited places [Stachniss et al., 2004]. Thrun et al. introduced an approach following the idea of representing uncertainty with an information matrix instead of a covariance matrix [Thrun et al., 2004]. By exploiting the sparsity of the information matrix, the algorithm, called Sparse Extended Information Filter (SEIF), allows updates of the state vector in constant time. Another variant of SLAM, the *Treemap* algorithm, has been introduced by Frese [Frese, 2006]. This method divides a map into local regions and subregions and the landmarks of each re-

gion are stored at the according level of the tree hierarchy. Lu and Milios introduced a method for globally optimizing robot trajectories by building a constraint graph from LRF and odometry observations [Lu and Milios, 1997]. The method described in this thesis is closely related to their work, but the modification enables efficient route graph corrections by decomposing the problem into pose tracking, optimization, and interpolation. In contrast to incrementally full state updates performed by EKF-based methods after each observation, the decomposition reduces the computational requirements during runtime to a minimum, thus allowing the efficient optimization of even large-scale environments. Whereas existing methods typically rely on a high density of landmarks, the RFID-based approach is tailored for very sparse landmark distributions with reliable data association.

In connection with radio transmitters, the SLAM problem has mainly been addressed as "range-only" SLAM [Kehagias et al., 2006, Djugash et al., 2005, Kurth et al., 2003, Kantor and Singh, 2002] since the bearing of the radio signal cannot accurately be determined. RFIDs have been successfully utilized already for localizing mobile robots [Hähnel et al., 2004, Bohn and Mattern, 2004] and emergency responders [Kantor et al., 2003, Miller et al., 2006]. Hähnel and colleagues [Hähnel et al., 2004] successfully utilized Markov localization for localizing a mobile robot in an office environment. Their approach deals with the problem of localization in a map previously learned from laser range data and known RFID positions, whereas the work presented in this paper describes a solution that performs RFID-based localization and mapping simultaneously during exploration. Also sensor networks-based Markov localization for emergency response has been studied [Kantor et al., 2003]. In this work, existing sensor nodes in a building are utilized for both localization and computation of a temperature gradient from local sensor node measurements. Bohn and colleagues examined localization based on super-distributed RFID tag infrastructures [Bohn and Mattern, 2004]. In their scenario tags are deployed beforehand in a highly redundant fashion over large areas, e.g. densely integrated into a carpet. They outline the application of a vacuum-cleaner robot following these tags. Pedestrian dead reckoning has also been combined with WLAN perception-based localization [Retscher and Kealy, 2006]. Miller and colleagues examined the usability of various RFID systems for the localization of first responders in different building classes [Miller et al., 2006]. During their experiments, persons were tracked with a Dead Reckoning Module (DRM) while walking through a building. They showed that the trajectories can be improved by utilizing the positions of RFID tags detected in the building. While these map improvements were carried out with only local consistency, the approach presented in this work yields a globally consistent map improvement.

Efficient solutions for decentralized SLAM have been introduced in the past [Reece and Roberts, 2005, Nettleton et al., 2003, Kim et al., 2004]. These approaches assume that at least low-bandwidth radio communication between the agents is possible. They solve the *double counting problem*, e.g. the multiple propagation of equal information

via different paths, by probabilistic methods, such as Bounded Covariance Inflation (BCI) [Reece and Roberts, 2005], and Covariance Intersection (CI) [Nettleton et al., 2003]. The decentralized method proposed in this work shows that this problem can be solved also by managing estimates locally in the memory of RFIDs.

4.7 Conclusion

RFID-SLAM allows the efficient generation of globally consistent maps, even if the density of landmarks is comparably low. For example, the method corrected an outdoor large-scale map within a few seconds from odometry data and RFID perceptions only. This was partially achieved due to reliable pose tracking based on slippage sensitive odometry, but also due to the data association solved by RFIDs. Solving data association by RFIDs allows to speed-up the route graph corrections by decomposing the problem into optimization and interpolation. Furthermore, a novel method for distributed SLAM by exploiting the memory capacity of RFIDs, has been contributed. The result shows that sharing information between single agents allows to globally optimize their individual paths, even without explicit need for communication. This has the advantage that, particularly in the context of disaster response, this method can also be applied if communication is disturbed by building debris and radiation. If communication is available, the process can be further accelerated by exchanging graph data between the agents directly.

Moreover, the introduced method allows to apply SLAM without requiring pedestrians and robots to perform loops while executing their primary task. Due to the joining of routes via RFID connection points, loops automatically emerge, which is a necessary requirement if applying SLAM in the real-world, particularly when humans are involved in USAR (Urban Search And Rescue) situations. Finally, the joint correction of human and robot trajectories, has been demonstrated. The result shows that sharing information between independent agents, i.e. humans and robots, allows to correct their individual paths globally.

Besides, RFID-SLAM offers many advantages, particularly in a disaster response scenario. One practical advantage is that humans can be integrated easily into the search since the exchange of maps can be carried out via the memory of RFIDs [Kleiner and Sun, 2007]. Furthermore, they can communicate with RFIDs by a PDA and leave behind information related to the search or to victims. The idea of labeling locations with information that is important to the rescue task, has already been applied in practice. During the disaster relief in New Orleans in 2005, rescue task forces marked buildings with information concerning, for example, hazardous materials or victims inside the buildings [FEMA, 2003]. The RFID-based marking of locations is a straight forward extension of this concept.

In future work, we will evaluate RFID technology operating in the UHF frequency

domain, which allows reading and writing within distances of meters. As we have already demonstrated in former work [Ziparo et al., 2007b], the combination of an RFID route graph representation with local mapping opens the door to efficient large scale exploration and mapping. Additionally, we will deal with the problem of building globally consistent elevation maps, as described in Chapter 5, by utilizing RFID technology-based route graph optimization for loop-closure.

5 Real-time Elevation Mapping

5.1 Introduction

The motivation for the elevation mapping method contributed in this chapter is to provide a basis for enabling robots to continuously plan and execute skills on rough terrain. Therefore, the method has to be computational efficient and capable of building elevation maps that are sufficiently accurate for structure classification and behavior planning, as we have already demonstrated in former work [Dornhege and Kleiner, 2007b]. Building globally consistent elevation maps, e.g. by loop closure, is computationally hard in large-scale environments since it requires to keep the whole map in memory. Therefore, the goal of the proposed method is not to build globally consistent maps, but maps that are locally consistent in the vicinity of the robot for continuous planning and navigation. In a further processing step, e.g. offline, a global map can be generated by merging locally consistent map patches according to a globally consistent route graph, as for example, computed by the RFID-SLAM method introduced in Chapter 4. Nevertheless, in the following we will denote poses as global in order to distinguish between the local and global coordinate frame of the robot.

Elevation mapping is carried out with a Kalman filter-based approach by integrating range measurements from a downwards tilted laser range finder. The map is incrementally build in real-time while the mobile robot explores an uneven surface. The method tracks the three-dimensional pose of the robot by integrating the robot's orientation, and the two-dimensional pose generated from visual odometry and scan matching (see Chapter 3). Furthermore, the three-dimensional pose is updated from height observations that have been registered on the map. Given the three-dimensional pose, the height value of each map cell is estimated by a Kalman filter that integrates readings from a downwards tilted LRF. Due to the integration of the three-dimensional pose, the method allows to create elevation maps while the robot traverses rough terrain, as for example driving over ramps and stairs.

The proposed approach was extensively evaluated in USAR test arenas designed by the National Institute of Standards and Technology (NIST) [Jacoff et al., 2001]. The results show that the method allows to reliably generate elevation maps in real-time while the robot explores an unknown area.

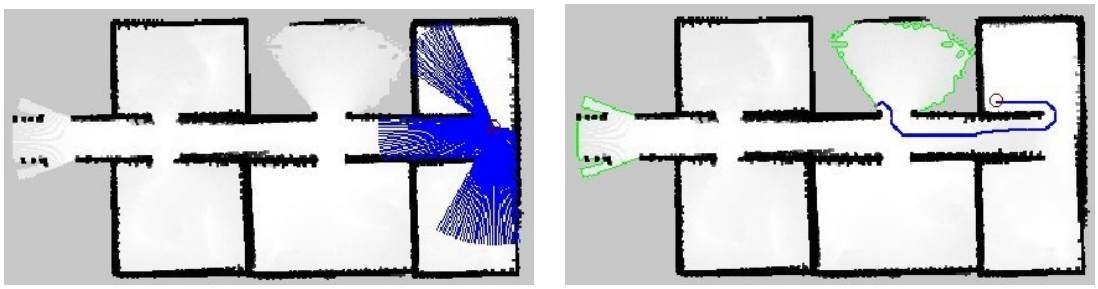
The remainder of this chapter is structured as follows. In Section 5.2 state-of-the-art techniques for mapping are described. In Section 5.3 the method for real-time elevation

mapping is proposed, and in Section 5.4 empirical results from mapping experiments are presented. Finally, in Section 5.5 an overview on related work is given, and in Section 5.6 the conclusion presented.

5.2 Conventional techniques

5.2.1 Occupancy grid maps

Occupancy grid maps are a data structure for fusing multiple sensor measurements into a discretized metric representation. They were introduced by Elfes and Moravec [Moravec and Elfes, 1985, Elfes, 1989], and have mainly been utilized for integrating range measurements from sonar and laser sensors for building a two-dimensional map of the environment. Basically, the representation partitions the space around the robot into equally sized rectangular cells. Each cell can take on values between 0 and 1, describing the probability that the corresponding location in the real world is occupied by an obstacle. Figure 5.1 depicts an occupancy grid generated in an office-like environment. The darker a cell, the higher the probability of occupancy. It is usually assumed that unexplored regions have an occupancy probability of 0.5, which corresponds to the large gray region in the figure. Elfes and Moravec developed an update procedure for



(a) (b)

Figure 5.1: (a) Occupancy grid map of an office-like environment, incrementally generated from laser range data (blue). (b) The robot plans to the next frontier cell (green) [Yamauchi, 1997].

computing for each map cell c_{xy} the occupancy probability $P\left(occ_{xy} \mid o_{1:t}, l_{1:t}\right)$, given past sensor observations $o_{1:t} = o_1, \cdots, o_t$, and robot poses $l_{1:t} = l_1, \cdots, l_t$. This is carried out by applying Bayes' rule under the assumption of independence between single sensor readings. Consequently, the occupancy probability of cell c_{xy} can be updated according

to the following procedure:

$$P\left(occ_{xy} \mid o_{1:t}, l_{1:t}\right) = 1 - \left[1 + \frac{P\left(occ_{xy} \mid o_t, l_t\right)}{1 - P\left(occ_{xy} \mid o_t, l_t\right)} \frac{1 - P\left(occ_{xy}\right)}{P\left(occ_{xy}\right)} \frac{P\left(occ_{xy} \mid o_{1:t-1}, l_{1:t-1}\right)}{1 - P\left(occ_{xy} \mid o_{1:t-1}, l_{1:t-1}\right)}\right]^{-1} \quad (5.1)$$

The procedure can be applied incrementally each time new sensor readings are available. In order to avoid numerical instabilities for probabilities close to zero or one, Equation 5.2.1 can be transformed to the *log odds* representation [Thrun et al., 2005]. Note that in static environments the prior probability $P\left(occ_{xy}\right)$ is typically assumed to be 0.5, thus the corresponding term in Equation 5.2.1 can be omitted. The update requires the global pose of the robot, which can be computed by methods described in Chapter 3 and Chapter 4. Furthermore, the occupancy probability $P\left(occ_{xy} \mid o_t, l_t\right)$ is computed based on a *inverse sensor model*, which is specific for the utilized type of sensor. In case of laser range finders, the sensor model is simply defined by a piecewise linear function:

$$P\left(occ_{xy} \mid o_t, l_t\right) = \begin{cases} P_{prior} & \text{if } c_{xy} \text{ is not covered by the beam} \\ P_{occ} & \text{if } c_{xy} \text{ is at the end of the beam} \\ P_{free} & \text{if } c_{xy} \text{ is before the end of the beam,} \end{cases} \quad (5.2)$$

where typically $P_{prior} = 0.5$, $P_{occ} = 0.8$, $P_{free} = 0.2$.

It can be seen in Figure 5.1 that the incremental update of the map reveals the structure of the environment, e.g. doors and hallways. The representation can be utilized efficiently for exploration and planning. Frontier cell-based exploration [Yamauchi, 1997] is a method that selects exploration targets based on the occupancy values of each cell and its next neighbors. Frontier cells are unoccupied cells and neighbor at least one cell that is unexplored. They are extracted from the grid map and taken as target locations for a trajectory planner. Figure 5.1 (b) depicts the generated frontier cells (green), and a path (blue) planned to reach one of these cells. Planning can be carried out by either applying value iteration or A* planning [Russell and Norvig, 2003] on the grid map, after growing obstacles on the grid with a Gaussian kernel in order to account for the size of the robot.

The described method has been widely used in office-like environments. In unstructured environments, a three-dimensional representation is needed since the surface might be uneven, and obstacles can be located at different levels of height. Moravec extended the approach for building occupancy grids in three-dimensions from stereo vision [Moravec, 1996]. However, since the computational requirements significantly increase in three-dimensions, the method does not scale in large environments.

5.3 Building elevation maps in real-time

An elevation map is represented by a two-dimensional array storing for each global location (x^g, y^g) a height value h with variance σ_h^2. In order to determine the height for each location, endpoints from the LRF readings are transformed from robot-relative distance measurements to global locations, with respect to the robot's global pose, and the *pitch* (tilt) angle of the LRF (see Figure 5.2).

This section is structured as follows. In Section 5.3.1 the update of single cell values relative to the location of the robot is described, in Section 5.3.2 the filtering of the map with a convolution kernel is shown, and in Section 5.3.3 an algorithm for the estimation of the robot's three-dimensional pose from dead reckoning and map observations is proposed.

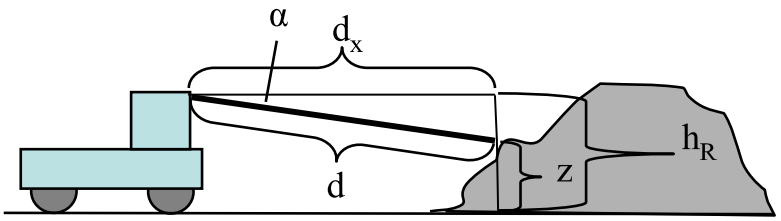


Figure 5.2: Transforming range measurements to height values.

5.3.1 Single cell update from sensor readings

Our goal is to determine the height estimate for a single cell of the elevation map with a Kalman filter [Maybeck, 1979], given all height observations of this cell in the past. We model height observations z_t by a Gaussian distribution $N\left(z_t, \sigma_{z_t}^2\right)$, as well as the current estimate $N\left(\hat{h}\left(t\right), \sigma_{\hat{h}(t)}^2\right)$ of each height value. Note that the height of cells cannot be observed directly, and thus has to be computed from the measured distance d and LRF *pitch* angle α. Measurements from the LRF are mainly noisy due to two error sources. First, the returned distance depends on the reflection property of the material, ranging from very good reflections, e.g. white sheet of paper, to nearly no reflections, e.g. black sheet of paper. Second, in our specific setting, the robot acquires scans while navigating on rough terrain. This will lead to strong vibrations on the LRF, causing an oscillation of the laser around the servo-controlled *pitch* angle. Consequently, we represent measurements from the LRF by two normal distributions, one for the measured distance $N(\mu_d, \sigma_d)$, and one for the *pitch* angle $N(\mu_\alpha, \sigma_\alpha)$.

The measurements from the LRF are transformed to robot-relative locations (x^r, y^r). First, we compute the relative distance d_x and the height z of each measurement accord-

ing to the following equation (see Figure 5.2):

$$\begin{pmatrix} d_x \\ z \end{pmatrix} = F_{d\alpha} \begin{pmatrix} d \\ \alpha \end{pmatrix} = \begin{pmatrix} d\cos\alpha \\ h_R - d\sin\alpha \end{pmatrix}, \tag{5.3}$$

where h_R denotes the height of the LRF mounted on the robot. Second, from distance d_x and the horizontal angle β of the laser beam, the relative cell location (x^r, y^r) of each cell can be calculated by:

$$x^r = d_x \cos\beta \tag{5.4}$$

$$y^r = d_x \sin\beta \tag{5.5}$$

Equation 5.3 can be utilized for computing the normal distributed distance $N(\mu_{d_x}, \sigma_{d_x})$, and height $N(\mu_z, \sigma_z)$, respectively. However, since this transformation is non-linear, $F_{d\alpha}$ has to be linearized by a Taylor series at μ_{d_x}, μ_z:

$$\begin{pmatrix} \mu_{d_x} \\ \mu_z \end{pmatrix} = F_{d\alpha} \begin{pmatrix} d \\ \alpha \end{pmatrix} \tag{5.6}$$

$$\Sigma_{d_x z} = \nabla F_{d\alpha} \Sigma_{d\alpha} \nabla F_{d\alpha}^T \tag{5.7}$$

$$\textit{with} \quad \nabla F_{d\alpha} = \begin{pmatrix} \cos\alpha & -d\sin\alpha \\ -\sin\alpha & -d\cos\alpha \end{pmatrix} \tag{5.8}$$

$$\textit{and} \quad \Sigma_{d\alpha} = \begin{pmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\alpha^2 \end{pmatrix}. \tag{5.9}$$

Then, the height estimate $\hat{h}$ can be updated from observation z_t, taken at time t, with the following Kalman filter:

$$\hat{h}(t) = \frac{1}{\sigma_{z_t}^2 + \sigma_{\hat{h}(t-1)}^2} \left(\sigma_{z_t}^2 \hat{h}(t-1) + \sigma_{\hat{h}(t-1)}^2 z_t \right) \tag{5.10}$$

$$\sigma_{\hat{h}(t)}^2 = \frac{1}{\frac{1}{\sigma_{\hat{h}(t-1)}^2} + \frac{1}{\sigma_{z_t}^2}}. \tag{5.11}$$

Equation 5.10 cannot be applied if the tilted LRF scans vertical structures since they lead to different height measurements for the same map location. For example, close to a wall the robot measures the upper part, far away from the wall the robot measures the lower part. We restrict the application of the Kalman Filter by the Mahalanobis distance. If the Mahalanobis distance between the estimate and the new observation is below a threshold c, the observation is considered to be taken from the same height. We use $c = 1$, which has the effect that all observations with a distance to the estimate that is below the standard deviation $\sigma_{\hat{h}}$, are merged. Furthermore, we are mainly interested

in the maximum height of a cell, since this is exactly what elevation maps represent. These constraints lead to the following update rules for cell height values:

$$\hat{h}(t) = \begin{cases} z_t & \text{if } z_t > \hat{h}(t) \wedge d_M\left(z_t, \hat{h}(t)\right) > c \\ \hat{h}(t-1) & \text{if } z_t < \hat{h}(t) \wedge d_M\left(z_t, \hat{h}(t)\right) > c \\ \frac{1}{\sigma^2_{z_t} + \sigma^2_{\hat{h}(t-1)}} \left(\sigma^2_{z_t} \hat{h}(t-1) + \sigma^2_{\hat{h}(t-1)} z_t\right) & else, \end{cases} \tag{5.12}$$

and variance $\sigma^2_{\hat{h}(t)}$ with:

$$\sigma^2_{\hat{h}(t)} = \begin{cases} \sigma^2_{z_t} & \text{if } z_t > \hat{h}(t) \wedge d_M\left(z_t, \hat{h}(t)\right) > c \\ \sigma^2_{\hat{h}(t-1)} & \text{if } z_t < \hat{h}(t) \wedge d_M\left(z_t, \hat{h}(t)\right) > c \\ \frac{1}{\frac{1}{\sigma^2_{\hat{h}(t-1)}} + \frac{1}{\sigma^2_{z_t}}} & else, \end{cases} \tag{5.13}$$

where d_M denotes the Mahalanobis distance, defined by:

$$d_M\left(z_t, \hat{h}(t)\right) = \sqrt{\frac{\left(z_t - \hat{h}(t)\right)^2}{\sigma^2_{\hat{h}(t)}}}\,. \tag{5.14}$$

The cell update introduced so far assumes perfect information on the global pose of the robot. However, since we integrate measurements from the robot while moving in the environment in real-time without loop-closure *, we have to account for positioning errors from pose tracking that do accumulate over time. Continuous Kalman updates without regarding pose uncertainty due to robot motion between update steps will successively reduce the cell's variance leading to variance estimates which highly underestimate the actual uncertainty about a cell's height. Thus we increase the variance in the Kalman propagation step based on the robot's motion to reflect the true uncertainty in the variance estimate. The height itself is not changed as the old estimate still gives the best possible estimation for a cell's height. We assume that the positioning error grows linearly with the accumulated distance and angle traveled. Hence, observations taken in the past loose significance with the distance the robot traveled after they were made:

$$\hat{h}(t) = \hat{h}(k) \tag{5.15}$$

$$\sigma^2_{\hat{h}(t)} = \sigma^2_{\hat{k}(t)} + \sigma^2_d d(t-k) + \sigma^2_\alpha \alpha(t-k), \tag{5.16}$$

where t denotes the current time, k denotes the time of the last height measurement at the same location, $d(t-k)$ and $\alpha(t-k)$ denotes the traveled distance and angle within the

*Note that global localization errors can be reduced by loop-closure, i.e. by re-computing the elevation map based on the corrected trajectory, which, however, can usually not be applied in real-time.

time interval $t - k$, and $\sigma_d^2, \sigma_\alpha^2$ are variances that have to be determined experimentally according to the utilized pose tracker. Since it would be computationally expensive to update the variances of all grid cells each time the robot moves, updates according to Equation 5.16 are only carried out on variances before they are utilized for a Kalman update with a new observation. The traveled distances can be efficiently generated by maintaining the integral functions $I_d(t)$ and $I_\alpha(t)$ that provided the accumulated distance and angle for each discrete time step t, respectively. Then, for example, $d(k - t)$ can be calculated by $I_d(k) - I_d(t)$. The integrals are represented by a table indexed by time t with a fixed discretization, e.g. $\Delta t = 1\ s$.

5.3.2 Map filtering with a convolution kernel

The limited resolution of the LRF occasionally leads to missing data in the elevation map, e.g. conspicuous by surface holes. Furthermore, the effect of "mixed pixels" might lead to phantom peaks within the elevation map [Ye and Borenstein, 2003], which frequently happens if the laser beam hits edges of objects and the returned distance measure is a mixture of the distance to the object and the distance to the background. Therefore, the successively integrated elevation map has to be filtered.

In computer vision, filtering with a convolution kernel is implemented by the convolution of an input image with a kernel in the spatial domain, i.e. each pixel in the filtered image is replaced by the weighted sum of the pixels in the filter window. The effect is that noise is suppressed and the edges in the image are blurred at the same time. We apply the same technique on the elevation map in order to reduce the errors described above. Hence, we define a convolution kernel of the size of 3×3 cells, where each value is weighted by its certainty and distance to the center of the kernel. Let $h(x + i, y + j)$ denote a height value relative to the kernel center at map location (x, y), with $i, j \in \{-1, 0, 1\}$. Then, the weight for each value is calculated as follows:

$$w_{i,j} = \begin{cases} \frac{1}{\sigma^2_{h(x+i,y+j)}} & \text{if } |i| + |j| = 0 \\ \frac{1}{2\sigma^2_{h(x+i,y+j)}} & \text{if } |i| + |j| = 1 \\ \frac{1}{4\sigma^2_{h(x+i,y+j)}} & \text{if } |i| + |j| = 2 \end{cases} \tag{5.17}$$

Consequently, the filtered elevation map h_f can be calculated by:

$$h_f(x, y) = \frac{1}{C} \sum_{i,j} h(x + i, y + j) w_{i,j}, \tag{5.18}$$

where $C = \sum w_{i,j}$.

5.3.3 Estimation of the three-dimensional pose

So far the incremental procedure for updating elevation map cells relative to the coordinate frame of the robot has been shown. In order to update map cells globally, the three-dimensional pose of the robot has to be considered, which is described by the vector $l = (x, y, h, \theta, \phi, \psi)^T$, where θ denotes the *yaw* angle, ϕ denotes the *pitch* angle, and ψ denotes the *roll* angle, respectively. We assume that IMU measurements of the three orientation angles are given with known variance. The position (x, y, h) is estimated by dead reckoning, which is based on the *pitch* angle and traveled distance measured by visual odometry (see Chapter 3) and scan matching. However, since these measurements are estimating the relative displacement δ with respect to the three-dimensional surface, δ has to be projected onto the plane, as depicted in Figure 5.3. Given the input

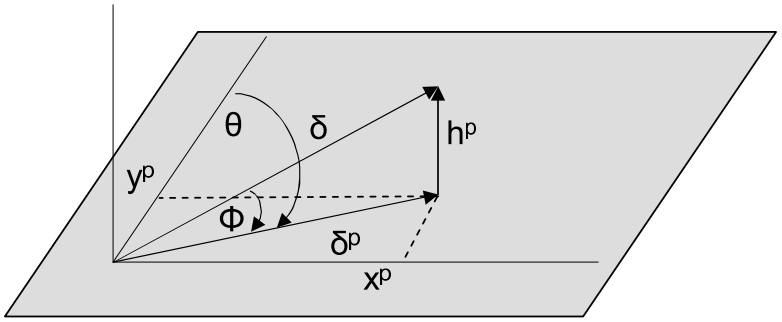


Figure 5.3: Dead reckoning of the projected Cartesian position (x_p, y_p, h_p) from *yaw* angle θ, *pitch* angle ϕ, and traveled distance δ.

$u = (\theta, \phi, \delta)^T$ represented by the Gaussian distribution $N(\mu_u, \sigma_u)$, the projected position $l = (x^p, y^p, h^p)^T$, represented by the Gaussian distribution $N(\mu_l, \Sigma_l)$, can be calculated as follows:

$$\begin{pmatrix} x_t^p \\ y_t^p \\ h_t^p \end{pmatrix} = F_{lu} \begin{pmatrix} x_{t-1}^p \\ y_{t-1}^p \\ h_{t-1} \\ \phi \\ \theta \\ \delta \end{pmatrix} = \begin{pmatrix} x_{t-1}^p + \delta \cos\theta \cos\phi \\ y_{t-1}^p + \delta \sin\theta \cos\phi \\ h_{t-1}^p + \delta \sin\phi \end{pmatrix} \tag{5.19}$$

$$\Sigma_{lu} = \nabla F_{lu} \Sigma_{lu} \nabla F_{lu}^T \tag{5.20}$$

$$\Sigma_{lu} = \nabla F_l \Sigma_l \nabla F_l^T + \nabla F_u \Sigma_u \nabla F_u^T, \tag{5.21}$$

where

$$\nabla F_l = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{5.22}$$

$$\nabla F_u = \begin{pmatrix} -\delta \cos\theta \sin\phi & -\delta \sin\theta \cos\phi & \cos\theta \cos\phi \\ -\delta \sin\theta \sin\phi & \delta \cos\theta \cos\phi & \sin\theta \cos\phi \\ \delta \cos\phi & 0 & \sin\phi \end{pmatrix}, \tag{5.23}$$

$$\nabla F_u = \Sigma_u = \begin{pmatrix} \sigma_\phi^2 & 0 & 0 \\ 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & \sigma_\delta^2 \end{pmatrix}, \tag{5.24}$$

$$\Sigma_l = \begin{pmatrix} \sigma_{x^p}^2 & \sigma_{x^p y^p}^2 & \sigma_{x^p h^p}^2 \\ \sigma_{x^p y^p}^2 & \sigma_{y^p}^2 & \sigma_{y^p h^p}^2 \\ \sigma_{x^p h^p}^2 & \sigma_{y^p h^p}^2 & \sigma_{h^p}^2 \end{pmatrix}. \tag{5.25}$$

Equation 5.19 allows to predict the current height of the robot. However, due to the accumulation of errors, the accuracy of the height estimate will decrease continuously. Therefore, it is necessary to update this estimate from direct observation. For this purpose, we utilize the height estimate $\left(\hat{h}(t), \sigma_{\hat{h}(t)}^2\right)$ at the robot's position from Equation 5.12 and 5.13, respectively. Then, the new estimate can be calculated by Kalman-fusing $\left(\hat{h}(t), \sigma_{\hat{h}(t)}^2\right)$ with the predicted height estimate $\left(h^p, \sigma_{h^p}^2\right)$ analogous to Equation 5.10.

The global location (x^g, y^g) of a measurement, i.e. the elevation map cell for which the height estimate $\hat{h}(t)$ will be updated, can be calculated straightforwardly by:

$$x^g = x^r + x^p \tag{5.26}$$

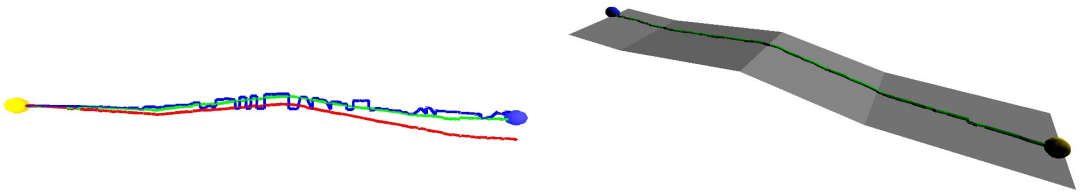
$$y^g = y^r + y^p \tag{5.27}$$

5.4 Experimental results

Elevation mapping was evaluated on the *Lurker* robot (see Chapter 2), which is capable to autonomously overcome rough terrain containing ramps and rolls. The system was successfully demonstrated during the RoboCup Rescue autonomy competition in 2006, where the robot won the first prize. The testing arena, which was used for the exper-

iments presented in this section, was designed by NIST during the Rescue Robotics Camp 2006 with the same degree of difficulty as presented at RoboCup'06, e.g. also containing rolls and ramps. During all experiments, the robot was equipped with an IMU sensor, a side camera for visual odometry, and two LRFs, one for scan matching and one for elevation mapping. The latter sensor was tilted downwards by 35°.

Figure 5.4 depicts the Kalman filter-based pose estimation of the robot, as described in Section 5.3.3. For this experiment, conditions were made intentionally harder. Map smoothing was turned off, which had the effect that missing data, due to a limited resolution of 2D scans, lead to significant holes on the surface of the map. Furthermore, we added a constant error of $-2°$ to pitch angle measurements of the IMU. As shown in Figure 5.4, the Kalman filter is able to cope with such errors, and finally produced a trajectory close to ground truth (indicated by the gray surface).



(a) (b)

Figure 5.4: Evaluation of the Kalman filter for tracking the robot's height. (a) Height values predicted from the IMU (red line) are merged with height values taken from the generated map (blue line). Errors from inaccuracies in the map, as well as a simulated continuously drift error of the IMU sensor are successfully reduced (green line). (b) Merged trajectory compared to ground truth (gray ramp).

In order to quantitatively evaluate the visual odometry support for elevation mapping, we recorded hight estimates while the robot was autonomously exploring an area, and finally climbing-up an open ramp. Figure 5.5 shows the results of the Kalman filter-based height estimation (blue line with crosses) in comparison to the manually measured ground truth (black line with triangles). As shown in Figure 5.5, the Kalman filter computes continuously an estimate close to ground truth, which stays consistently within the expected covariance bounds.

Figure 5.6 shows the final result from applying the proposed elevation mapper during the Rescue Robotics camp in Rome 2006. Figure 5.6 (a) depicts an overview on the arena, and Figure 5.6 (b) shows the calculated height values. The height of each cell is indicated by a gray value, the darker the cell, the bigger the elevation. Figure 5.6 (c) depicts the variance of each height cell, going from pink (hight variance) to yellow (low variance) The current position of the robot is indicated by a blue circle in the lower left corner. The further away cell updates on the robot's trajectory, the lower their variance

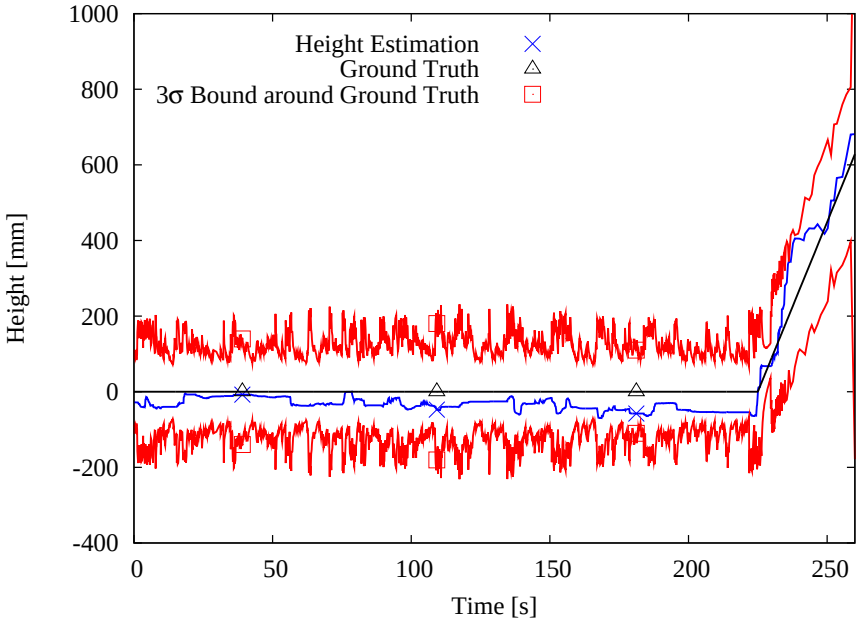
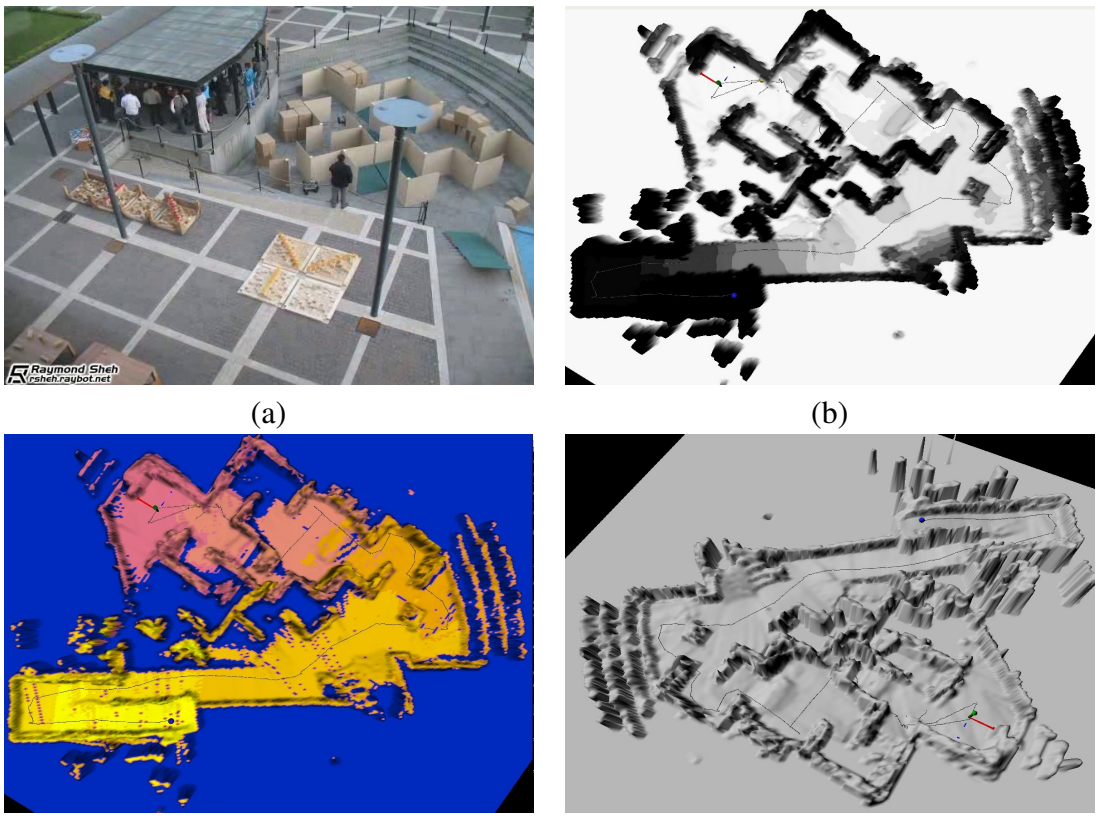


Figure 5.5: Quantitative evaluation of the Kalman-filtered height estimation. During this experiment the robot started exploration on the floor and then climbed a ramp at around 220 s. The graph shows the ground truth (black line with triangles) in comparison to the height estimation by the Kalman filter (blue line with crosses).

(see Section 5.3.1). Figure 5.6 (d) shows a 3D visualization of the generated elevation map. Structures, such as the long ramp at the end of the robot's trajectory, and the stairs, can clearly be identified. We measured on an $AMD64X2$ 3800+ a total integration time (without map smoothing) of $1.88 \pm 0.47\,ms$ for a scan measurement with 683 beams, including $0.09 \pm 0.01\,ms$ for the 3D pose estimation. Map smoothing has generally the time complexity of $O\left(N^2M\right)$, where N is the number of rows and columns of the map and M the size of the kernel. We measured on the same architecture $34.79 \pm 14.84\,ms$ for smoothing a map with $N = 300$ and $M = 3$. However, this can be improved significantly during runtime by only smoothing recently modified map cells and their immediate neighbors within distance M.

5.5 Related work

Elevation maps are indispensable, particularly for robots operating in unstructured environments. They have been utilized on wheeled robot platforms [Pfaff et al., 2007a, Wolf et al., 2005], on walking machines [Krotkov and Hoffman, 1994, Gassmann et al., 2003] and on car-like vehicles [Thrun et al., 2006, Ye and Borenstein, 2003, Pfaff et al., 2007b].



(a) (b)

(c) (d)

Figure 5.6: Elevation mapping during the Rescue Robotics Camp 2006 in Rome: (a) The arena build-up by NIST, (b) the corresponding digital elevation model (DEM) build by the *Lurker* robot, going from white (low altitude) to black (hight altitude), (c) the variances of each height value according to the robots position (lower left corner), going from pink (hight variance) to yellow (low variance), (d) the 3D perspective rotated 180°.

These methods differ in the way how range data is acquired. If data is acquired from a 3D scan [Pfaff et al., 2007a, Krotkov and Hoffman, 1994], it usually suffices to employ standard error models, which reflect uncertainty from the measured beam length. Data acquired from a 2D LRF, e.g. tilted downwards, requires more sophisticated error models, such as the compensation of pose uncertainty [Thrun et al., 2006], and handling of missing data by map smoothing [Ye and Borenstein, 2003].

Other researchers work focused on the building of maps from 3D scans by registering each scan on the existing map with the Iterative Closest Points (ICP) algorithm [Surmann et al., 2003]. In Surmann's setting, the robot acquires a 3D scan, plans and navigates based on this scan with a next best view planner, and stops again for acquiring the next scan from another position. They extended their work with a method for generat-

ing Prolog-based semantic descriptions of the 3D point clouds, which they utilized for optimizing the 3D model by Prolog unification [Nüchter et al., 2003]. Generally, full 3D data processing is still only limitedly applicable in real-time. The acquisition of 3D data, as well as 3D data registration, is still time consuming, thus requiring interruptions of continuous navigation. Furthermore, the sporadic data acquisition makes it difficult to apply the method in dynamic environments.

In contrast to previous work, the proposed approach deals with the problem of building elevation maps in real-time, allowing the robot continuous planning and navigation. Furthermore, the assumption that the robot has to be situated on a flat surface while mapping rough terrain has been relaxed.

5.6 Conclusion

We have contributed a solution for the real-time building of elevation maps from LRF data. It has been demonstrated that the generated elevation maps provide a good trade-off between computational efficiency and structural representation. They can reliably be generated in real-time while the robot is continuously in motion. This has partially been achieved by the introduced method for visual odometry, which significantly improved the accuracy of scan matching. As former work has shown, the generated elevation maps can be utilized for structure classification, and the planning of skill execution [Dornhege and Kleiner, 2007b], thus enabling a mobile robot to truly autonomously explore rough terrain.

To explore terrain based on a local world model might be sufficient for a range of tasks. However, many real-world situations require the robot to continuously build a global representation of the world in order to perform tasks globally efficient. For example, the problem of searching victims after a disaster requires a global representation of a large-scale environment. To represent such environments by grid map-like data structures becomes intractable if the map size increases. Therefore, in future work, we will deal with the problem of building globally consistent elevation maps by utilizing RFID technology-based route graph optimization for loop-closure (see Chapter 4). Here the basic idea is to anchor locally generated elevation maps in the corrected graph. These map patches can then be loaded into memory if they are within range of the robots current location. Furthermore, the saved data can be utilized offline for generating a global elevation map representation of the environment.

6 RFID Technology-based Multi-Robot Exploration

6.1 Introduction

During rescue operations for disaster mitigation cooperation is a must [Jennings et al., 1997]. In general the problem is not solvable by a single agent, but a heterogeneous team is needed that dynamically combines individual capabilities in order to solve the task. This requirement is due to the structural diversity of disaster areas, various sources of human presence that sensors can perceive, and to the necessity of quickly and reliably examining the targeted regions.

In the urban search and rescue context, there exists no standard robot platform that is capable of solving all kinds of challenges demanded by the environment. For example, there are places only reachable by climbing robots, spots only accessible through small openings, and regions only observable by aerial vehicles. Multi-robot teams do not only offer the possibility to field such diverse capabilities, they also exhibit increased robustness due to redundancy, and superior performance due to parallel task execution [Arai et al., 2002]. This latter aspect is as important as the first one, since the time to complete a rescue mission is literally of vital importance. Therefore, to approach the task of designing a multi-robot system for rescue from single-robot solutions is prone to yield suboptimal solutions. The joint performance of a team depends on the right mixture and coordination of the individual robot capabilities, and thus has to be designed as a whole. One fundamental problem in this context is the coordination of a team during an exploration task.

To coordinate a team of robots for exploration is a challenging problem, particularly in large-scale areas, as for example the devastated area after a disaster. This problem can be generally decomposed into task assignment and multi-agent path planning. Whereas task assignment and multi-agent path planning were both intensively studied as separate problems in the past, there has been only little attention on solving both of them at once, particularly if large robot teams are involved. This is mainly due to the fact that the joint state space of the planning problem grows enormously with the number of robots. However, particularly in destroyed environments where robots have to overcome narrow passages and obstacles, path coordination is essential in order to avoid collisions and deadlocks.

In this chapter a novel deliberative approach that reduces significantly the size of the search space by utilizing RFID tags as coordination points, while solving the problem of task assignment and path planning simultaneously, is contributed. The method utilizes the same RFID infrastructure as RFID-SLAM (Chapter 4), requiring robots to deploy RFID tags autonomously in the environment for building a network of reachable locations. Hence, global path planning is carried out on a graph topology, which is computationally cheaper than planning on global grid maps, as it is usually the case.

Furthermore, a local search method that uses the memory of RFIDs for labeling visited places in the field is contributed. Each RFID holds a set of poses that have been visited by robots in the vicinity of the tag. Since data exchange is carried out via the memory of RFIDs, the method can also be applied if radio communication fails completely. Additionally, the local approach has the advantage that computational costs do not grow with the number of robots participating in the search. Note that the correct association of visited poses to RFID locations requires to determine detection range and bearing from signal strength and antenna orientation, respectively.

The implemented system can be considered as two-layered, consisting of a *local mechanism* and a *global mechanism*. The *global mechanism* monitors the local exploration, and possibly restarts it at different locations, if the overall performance can be improved. This is carried out by assigning new target locations and generating a global multi-robot plan to reach them. Global planning is carried out in configuration time-space on the RFID network, and genetic sequence selection is applied as a global task assignment strategy.

For the purpose of navigation and obstacle avoidance, robots plan their trajectories with A* [Russell and Norvig, 2003] on a dynamically updated grid map (see Section 5.2), which is limited in size and shifted according to robot motion. Due to the limited size, grid updates from laser range data and navigation planning can be achieved without significant computational costs.

The proposed methods were evaluated in the USARSim [Balakirsky et al., 2006] simulation environment, which serves as basis for the *Virtual Robots* competition at RoboCup [Balakirsky et al., 2007], where our team one the first prize of the competition. The results show that the RFID tag-based exploration works for large robot teams, particularly under limited computational resources.

The remainder of this chapter is organized as follows. The RFID technology-based local and global exploration method and their application to rescue missions is discussed in Section 6.2 and Section 6.4, respectively. In Section 6.3 a more detailed description of the topological RFID map generated by the robots during exploration is provided. In Section 6.5 results from experiments are shown and in Section 6.6 related approaches are discussed. Finally, in Section 6.7 the conclusion is presented.

6.2 Coordinated local exploration

In this section a novel coordination mechanism [Kleiner et al., 2006c, Ziparo et al., 2007a] is contributed, which allows robots to explore an environment with low computational overhead and communication constraints. In particular, the computational costs do not increase with the number of robots. The key idea is that the robots plan their path and explore the area based on a local view of the environment. Team coordination is realized through the use of indirect communication via RFID tags. In Section 6.2.1 the method for efficient navigation based on a local world view is presented, and in Section 6.2.2 a novel coordination mechanism based on target selection and indirect communication is introduced.

6.2.1 Navigation

To efficiently and reactively navigate, robots continuously plan their trajectories based on a local representation of the environment, which is maintained within an occupancy grid [Moravec, 1988] limited in size. For example, in our implementation, the local map was limited to a four meter side square with 40 mm resolution. The occupancy grid is shifted according to wheel odometry and IMU data, and continuously updated from measurements of the LRF. This allows to overcome the accumulation of odometry errors, while having some memory of the past. The exploration process periodically selects targets, as shown in the following section, and continuously performs A* search [Russell and Norvig, 2003] on the occupancy grid in order to generate plans for reaching them. The continuous re-planning allows the robot to reactively avoid newly perceived obstacles, and to deal with unforeseen situations caused by errors during path following. A* planning is implemented with the Euclidean distance heuristic and the state expansion selects all neighbors of a cell that are not obstructed (i.e. have an occupancy value lower than a given threshold). The cost function c takes the length of the path and its distance to obstacles into account:

$$c\left(s_{i+1}\right) = c\left(s_i\right) + d\left(s_{i+1}, s_i\right) * \left(1 + \alpha * occ\left(s_{i+1}\right)\right) \tag{6.1}$$

where $occ\left(s\right)$ is the probability of occupancy of grid cell s, $d\left(.\right)$ is the Manhattan distance, and α is a factor for varying the cost for passing nearby obstacles. Furthermore, the grid is convoluted with a Gaussian kernel, forcing plans to be within a minimum distance to nearby obstacles.

While navigating in the environment, robots maintain a Local RFID Set (LRS), which contains all perceived RFIDs that are within range of the local occupancy grid. RFIDs which are out of the antenna's detection range, but still within the range of the grid, are shifted according to the motion of the robot. Additionally, the occupancy grid is augmented with RFIDs from a topological RFID graph representation, which will be

further described in Section 6.4, facilitating the processing of nearby RFIDs that have not been observed directly. On the basis of this information, new RFIDs are released in the environment in order to maintain a predefined density of RFID tags (in our implementation we took care of having the RFIDs at one meter distance from each other).

RFIDs that are within the local occupancy grid are utilized for avoiding collisions between the robots. Each robot tracks its own pose by integrating measurements from the wheel odometry and the IMU sensor with an Extended Kalman filter (EKF). As commonly known, the accuracy of this estimate decreases due to the accumulation of positioning errors, which can, for example, be prevented by performing data association, as shown in Chapter 4. However, since the proposed approach aims at a computational efficient implementation, pose estimates are not globally improved during the multi-robot exploration. Instead, local displacements between robots are synchronized via RFID tags. As will be shown, this method leads to pose estimates that are sufficiently accurate for collision-free navigation. Note that after the execution, the global map can still be improved offline, for example, by methods described in Chapter 4.

If two robots detected the same RFID tag in the past, the estimates of their mutual displacement $d_{R_1R_2} \approx l_{R_1} - l_{R_1}$ were synchronized by utilizing their local pose estimates at this RFID: Let $l_{R_1}(t_1)$ and $l_{R_2}(t_2)$ denote the individual pose estimates of robot R_1 and R_2 while detecting the same RFID tag at time t_1 and time t_2, respectively. Then, the new displacement between both robots can be calculated by $d_{R_1R_2} = l_{R_1}(t_1) - l_{R_1}(t_2)$. Furthermore, each robot can estimate poses within the reference frame of other robots by utilizing the latest displacement and the individual pose estimate of the other robot at time t. For example, R_2's pose estimate of R_1 is given by: $\hat{l}_{R_1}(t) = l_{R_1}(t) - d_{R_1R_2}$. Note that this procedure assumes the existence of a synchronized clock and requires the robots to keep their trajectory in memory.

Known locations of other robots can be utilized on the planning level for avoiding collisions between them. This is carried out by adding an extra cost to occupancy grid cells that are close to these locations. If a robot detects that a teammate with a higher priority is close, it stops until the other has moved out of the way. The simplest choice for a priority scheme is to assign a ranking of the robots at start-up time, e.g. to assign to each robot a unique ID. However, this method might fail if, for example, the robot with the highest priority is unable to move first since it is blocked by any other robot with a lower priority. There exist analytical solutions to this problem, as for example described by Jaeger and Nebel [Jaeger and Nebel, 2001], which require as input a dependency graph consisting of one node for each robot and one arc for each blockage between two robots. Since it is hard to decide from sensor data, such as vision or laser range readings, whether robots are blocked, we decided to utilize a simpler approach: If the conflict cannot be solved, the priority scheme is randomly shuffled by the robot with the highest priority, and communicated between robots that are involved in the conflict. Subsequently, the robots try to execute the new sequence. Note that this method, as any other method at this stage, is unable to solve a deadlock, which is a situation in which

any sequence of movements would not solve the conflict.

6.2.2 Local exploration

The fundamental problem of multi-robot exploration is how to select targets for the path planner that minimize the overlapping of explored areas. This involves, first, choosing a set of target locations $F = \{f_j\}$, second, computing a utility value $u\left(f_j\right)$ for each target location $f_j \in F$, and third, selecting the best target based on the utility value, for which the path planner can find a trajectory.

The set of targets F is identified by extracting frontier cells F [Yamauchi, 1997] from the occupancy grid. A frontier cell is defined as a cell that has not been explored, i.e. that has not been observed by the laser range finder, and also neighbors an observed cell that is unoccupied, i.e. has a occupancy probability below a certain threshold. Unexplored cells are those with prior probability, e.g. 0.5, which is initially set for all cells. Finally, the set is ordered based on the following utility calculation:

$$U\left(f_j\right) = -\gamma_1 F_a\left(|\phi - \theta_r|\right) - \gamma_2 F_v\left(l_{f_j}\right), \tag{6.2}$$

$$with \quad \phi = \tan^{-1}\left(\frac{x_r - x_{f_j}}{y_r - y_{f_j}}\right), \tag{6.3}$$

where ϕ denotes the angle between robot location l_r and frontier cell location l_{f_j}, θ_r denotes the orientation of the robot, and γ_1 and γ_2 are two parameters which control the trade-off between direction persistence and exploration. The function F_a denotes a fuzzy function yielding values in the interval $[0, 1]$, the smaller the angle, the higher the return value of the function. F_v is a function returning a value the higher the more locations in the vicinity of l_{f_j} have been already explored. As will be shown, the information about visited locations is read from RFIDs $r \in LRS$, which are within range of the local occupancy grid.

The angle factor F_a can be thought as an inertial term, which prevents the robot from changing direction too often (which would result in an inefficient behavior). If the robot has full memory of his perceptions (i.e. a global occupancy grid), the angle factor would be enough to allow a single robot to explore the area. However, due to the limitation of the occupancy grid, knowledge on previously visited areas gets lost, leading to inefficient exploration. This problem is solved by memorizing explored regions in the memory of RFIDs. If robots are within writing distance of RFIDs, they write their trajectory into the tags memory. More specifically, they memorize visited poses p from their trajectory (discretized at a lower resolution compared to the occupancy grid). The influence radius, e.g. the maximal distance in which poses are added, depends mainly on the memory capacity of the RFID tag and can be adjusted accordingly. In our implementation, poses were added within a radius of 4 meters. Note that poses are converted from the robot's coordinate frame to an RFID tag relative coordinate frame in order to

be independent of the individual coordinate frame of each robot. This has the additional advantage that, given the resolution and maximal relative distance, their location can be efficiently encoded into memory.

Moreover, a value $count\,(p)$ [Svennebring and Koenig, 2004] is associated with each pose $p = \left(x_p, y_p\right)$ in the memory of the RFID and is incremented by the robots every time the pose is added. These poses are then used to compute F_v:

$$F_v\left(l_{f_j}\right) = \sum_{r \in LRS} \sum_{p \in P_r} \frac{count\,(p)}{d\left(l_{f_j}, p\right)} \tag{6.4}$$

where P_r is the set of poses associated with the RFID r, and $d\,(.)$ denoted the Euclidean distance. It is worth noticing that robots writing and reading from RFIDs not only maintain memory of their own past, but also of the other robots implementing thereby a form of indirect communication and coordination of exploration. Thus, both multi-robot navigation and exploration do not require direct communication. This feature is very useful in all those scenarios (e.g. disaster scenarios) where wireless communication may be limited or unavailable. The most important feature of the approach, as presented up to now, is that the computation costs do not increase with the number of robots. Thus, in principle, there is no limit, other than the physical one, to the number of robots constituting a team.

6.3 Multi-robot topological maps

After the occurrence of a real disaster it is crucial for first responders to locate and rescue victims as fast as possible. In this context, the main task of a robot team is to explore and map the destroyed area, and more importantly, to mark locations of victims within the generated map. As has been already shown in Chapter 4, RFID tags can be used for building a topological representation of the environment. The advantage of building topological RFID maps is that this corresponding infrastructure can also be utilized by human first responders after the robots have explored the terrain. Instead of having to localize themselves within a metric map, which can be rather difficult if the environment is unstructured, they can be equipped with wearable devices, such as RFID reader and PDA, for following a path of RFID tags to task-relevant places, such as nearby victim locations, unexplored regions, or the exit of a building.

During execution of the local exploration method, described in Section 6.2, robots successively distribute RFIDs in the environment, which basically represent navigation points that are connected with each other via routes traveled by the robots. This connectivity network is taken as a basis for building a topological map consisting of vertices that are RFID tags, starting location and victim locations, and edges that are connections between them (see Figure 6.1 for an example). Each vertex stores the direction

and distance to all other vertexes to which an edge exists. This graph can be used to plan the shortest path to the next victim location and back to the exit of the building, e.g. by utilizing the Dijkstra [Dijkstra, 1959] algorithm or A* [Russell and Norvig, 2003] planning. In Chapter 8 a wearable device for first responders and methods for computing optimal routes will be described in more detail.

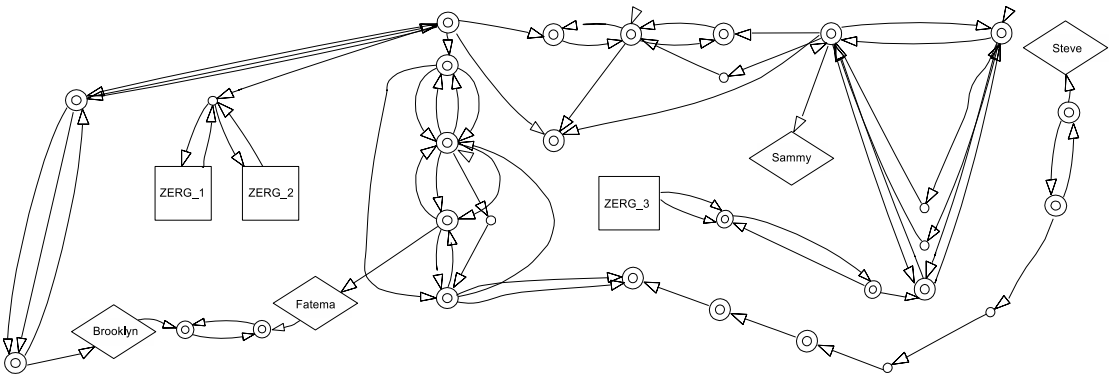


Figure 6.1: Topological map of the disaster area generated by a team of robots in USARSim environment: rectangles denote robot start locations, diamonds detected victims, and circles RFID locations.

During the distribution of RFID tags, robots measure between two tags i and j the relative displacement $d_{ij} = (\Delta x_{ij}, \Delta y_{ij})^T$ with covariance matrix Σ_{ij} if they have been passed. The relative displacement between two tags is estimated by a Kalman filter, which integrates pose estimates from the robot's wheel odometry, and an Inertia Measurement Unit (IMU). It is assumed that the yaw angle of the IMU is aligned to magnetic north, i.e. that IMU measurements are supported by a compass. If the robot passes a tag or victim location, the Kalman Filter is reset in order to estimate the *relative* distance to the subsequent tag on the robot's trajectory. Accordingly, each robot generates a graph with multiple measurements between RFID nodes, which can be exchanged with other robots if they are within communication range. Due to the unique identification of RFID tags, these graphs can be merged easily to a single graph consisting of the unification of all vertices and edges, where the overall displacement D_{ij} between two vertices i and j can be computed from the weighted average of collected measurements from all robots:

$$D_{ij} = \frac{1}{C} \sum_{k} \sum_{k_{ij}} d_{k_{ij}} \Sigma_{k_{ij}}^{-1}, \tag{6.5}$$

where

$$C = \sum_{k} \sum_{k_{ij}} \Sigma_{k_{ij}}^{-1} \tag{6.6}$$

and k_{ij} indicates the measurement between node i and j collected by robot k. Note if

there does not exist a measurement between two nodes, the elements of the corresponding covariance matrix are set to zero. The method described above was applied for building multi-robot maps during the RoboCup Rescue *Virtual Competition*. The generated map can be further improved by applying RFID-SLAM, as described in Chapter 4.

6.4 Global exploration monitoring

Due to the lack of lookahead of local exploration, robots may stay too long in local minima, resulting in redundant coverage of already explored areas. For example, this might be the case if the utility function $U\left(l_{f_j}\right)$ builds a plateau over a larger region, i.e. if the *count* values of poses in a larger region are nearly the same, and unexplored passages are only reachable with high costs from the *angle* factor F_a.

In order to avoid such a phenomenon, a novel monitoring approach has been developed, which periodically restarts the local exploration in more convenient locations. This method requires direct communication and a computational overhead, which grows with the number of agents. However, it improves the exploration ability of the robots significantly and it is robust to failures. In fact, if the communication links fail or the monitoring process itself fails, the robots will continue the local exploration as previously described. The remainder of this section is structured as follows. In Section 6.4.1 the problem is formalized. Then, in Section 6.4.2 three algorithms for solving this problem are introduced, and finally, in Section 6.4.3 the monitoring agent a central unit that computes and monitors all global actions of the agents is introduced.

6.4.1 Problem modeling

Basically, the problem is to find optimal target locations for all robots, and a multi-robot plan for reaching them. As a basis for solving this problem, a graph $G = (V, E)$, where V is the set of landmark locations, e.g. RFIDs, and E the set of connections between them. The graph is extracted from the topological map introduced in Section 6.3. Each node consists of a unique identifier of the RFID and its estimated position. Moreover, a set of frontier nodes $U \subset V$ and a set of current robot RFID positions $SL \subset V$ is defined. In general, $|U| > |R|$, where R is the set of available robots. A robot path (i.e plan) is defined as a set of pairs composed by a node $v \in V$ and a time-step t:

Definition 6.4.1 *A* single-robot plan *is a set* $P = \{<v, t> \mid v \in V \wedge t \in T\}$*, where* $T = \{0, \ldots, |P| - 1\}$*. P must satisfy the following properties:*
a) $\forall v_i, v_j, k \ <v_i, k> \in P \wedge <v_j, k+1> \in P \Rightarrow (v_i, v_j) \in E$,
b) $<v, 0> \in P \Rightarrow v = sl_i \in SL$
c) $<v, |P_i| - 1> \in P \Rightarrow v \in U$

where property a) states that each edge of the plan must correspond to an edge of the graph G. Properties b) and c) enforce that the first and the last node of a plan must be the location of a robot and a goal node, respectively. For example, the single-robot plan going from RFID R1 to RFID G1, depicted in Figure 6.2, is represented as $P_1 = (<R1, 0>, <N1, 1>, <N2, 2>, <G1, 3>)$. The previous definition implies

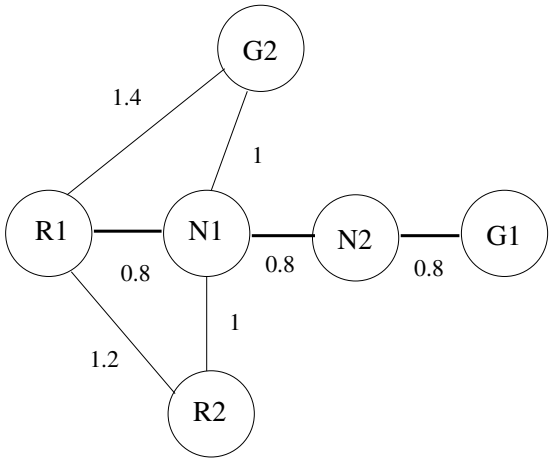


Figure 6.2: A simple graph showing a plan from $R1$ to $G1$ (bold edges).

that passing any two nodes, which are connected by an edge in the graph G, takes approximately the same amount of time. Recall that nodes represent RFIDs which are deployed approximately at the same distance one from the other, and edges represent shortest connection between them. Thus, the difference of time required for traveling between any two connected RFIDs is negligibly small, if robots drive at the same speed.

Definition 6.4.2 *A* multi-robot plan *P is a n-tuple of single-robot plans $(P_1, \ldots, P_n)$ such that:*
a) the plan with index i belongs to robot i,
b) $\forall i, j \in R \ <v', |P_i| - 1> \in P_i \wedge <v'', |P_j| - 1> \in P_j \Rightarrow v' \neq v''$
c) $\forall i, j \in R \ <v', 0> \in P_i \wedge <v'', 0> \in P_j \Rightarrow v' \neq v''$

Thus, a multi-robot plan is a collection of single-robot plans for each robot such that they all have different goals and different starting positions. A distinguishing feature of multi-robot plans with respect to single-agent ones is interaction. In fact, single-robot plans can interfere with each other leading to inefficiencies or even failures:

Definition 6.4.3 *Two single-robot plans P_i and P_j of a multi-robot plan P are said to be in* conflict *if $P_i \cap P_j \neq \emptyset$. The set of states $C_{P_i} = \bigcup_{i \neq j} P_i \cap P_j$ are the conflicting states for P_i.*

Moreover, *deadlocks* can occur in the system. In this setting a deadlock can arise if there is a circular wait or if a robot is willing to move to an already achieved goal of another robot.

Definition 6.4.4 *A multi-robot plan is said to have a* deadlock *if there is a circular wait or an infinite wait. The wait set for a multi-robot plan P is defined as: $WS_P = \{(i, j, t) \mid i, j \in R \ \wedge t \in T \wedge P_i(t+1) = P_j(t)\}$. A* circular wait *for a multi-robot plan P occurs if: $\exists$ a sequence of k distinct robots $(r_1, \ldots, r_k)$ and a time t such that $(r_i, r_{i+1}, t) \in WS_P$ for $i = 0 \ldots k-1$ and $(r_k, r_1, t) \in WS_P$. An* infinite wait *for a multi-robot plan P occurs if $\exists t_1, t_2 \in T \ \exists i, j \in R \mid P_i(t_1) = P_j(t_2) \wedge t_2 = |P_j| - 1 \wedge t_1 \geq t_2$.*

Consequently, the cost measure $c(.)$ for a multi-robot plan P is defined as follows:

$$c(P) = \begin{cases} \infty & \text{if deadlock} \\ \max\limits_{i \in R} cost(P, i) & \text{else} \end{cases} \tag{6.7}$$

where $cost(P, i)$ is the cost of executing i's part of the multi-robot plan P. We assume that the agents execute the plans in parallel, thus the score of the multi-robot plan is the maximum among the single-robot ones. Let $P_j(t)$ be a function that returns the RFID node of a plan P_j at a time index t, and $d(.)$ the Euclidean distance between two RFIDs. Then, $cost(P, i)$ can be computed from the sum of the Euclidean distances between the RFIDs of the plan plus the conflicts cost:

$$cost(P, i) = \sum_{t=0}^{|P_i|-2} d(P_i(t), P_i(t+1)) + confl(P, i) \tag{6.8}$$

where

$$confl(P, i) = \sum_{j \neq i} \sum_{<v,t> \in P_i \cap P_j} wait(P_j, t), \tag{6.9}$$

and

$$wait(P_j, t) = d(P_j(t-1), P_j(t)) + d(P_j(t), P_j(t+1)), \tag{6.10}$$

where the wait cost $wait(P_j, t)$ reflects the time necessary for robot j to move away from the conflict node. By Equation 6.9 costs for waiting are added if at least two robots share the same RFID node at the same time. This is a worst case assumption since conflicts in the final multi-robot plan are solved by the local coordination mechanism which forces robots only to wait if there are other robots with higher priority. We abstract this feature from our model since the priority ordering is periodically randomized in order to solve existing dead-locks, making it impossible to predict whether a robot will have to wait or not. Finally, the task assignment and path planning problem can be formulated as an optimization problem of finding a plan P^* that minimizes the cost function $c(P)$.

6.4.2 Global task assignment and path planning

We experimented with three different techniques in order to solve the task assignment and path planning problem. The first two approaches are inspired by Burgard et al. [Burgard et al., 2005]. The third approach can be seen as an extension of Bennewitz et al. [Bennewitz et al., 2001]. All of the previously cited approaches rely on a grid based representation while our approach is graph-based. The experimental results show that the third approach outperforms the first and the second, and is actually the one we adopted in the implementation of the full system. For this reason, only a brief overview of the first two approaches is given, but a more detailed description of the third.

All the approaches have a common pre-calculation. We compute the pairwise shortest paths [Dijkstra, 1959] for each node in U. This is a fast computation (i.e. $O((|E| + |V|log(|V|))|U|)$) which speeds up the plan generation processes presented in the following.

Greedy Approach

Given the information produced by the Dijkstra algorithm and an empty multi-robot plan, we identify the robot $r_{best} \in R$ which has the shortest path to reach a goal $g_{best} \in U$. The path computed by the Dijkstra algorithm from r_{best} to g_{best}, with its time values is added to the multi-robot plan. We then update the sets $R = R - \{r_{best}\}$ and $U = U - \{g_{best}\}$. The process is iterated until $R = \emptyset$ (see [Burgard et al., 2005] for more details).

Assignment Approach

Task assignment is a common approach in multi-robot systems to coordinate tasks between the agents. Here we utilize a genetic algorithm permuting over possible goal assignments to robots and use the plans computed by the Dijkstra algorithm. We then use the previously defined cost function as the fitness function (see [Bennewitz et al., 2001] for more details).

Sequential Approach

The last presented approach is based on sequential planning. We use, similar to the assignment approach, a genetic algorithm to permute possible orderings of agents $O = o_1, \ldots, o_n$. We then plan the ordering and use the previously defined cost function as the fitness function.

The sequential planning is based on A* [Russell and Norvig, 2003] and is done individually, following the given sequence, for each agent in order to achieve the most convenient of the available goals U. Every time when an agent o_i plans, the selected goal is removed from U and the computed plan added to the set of known plans P. The planning tries to avoid conflicts with the set of known plans P by searching through

time/space, where the state space S is defined as $S = V \times T$. This huge state space can be greatly simplified since we are only interested in the time of the conflicting states C_{P_j} (Definition 6.4.3). From the planning point of view the information relative to the time of non-conflicting states is irrelevant and thus all these states can be grouped by time using the special symbol *none*. The resulting set of non-conflicting states NC is defined as $NC = \{<v, none> \mid v \in V\}$ and has the cardinality of V (i.e. $|NC| = |V|$). Thus, the reduced search space is $ST = C_{P_j} \cup NC$.

During the search, the nodes are expanded in the following way: We look for the neighboring nodes of the current one given the set E of edges in G. For each of them we check if there is a conflict. If this is the case, we return the corresponding node from C_{P_j}, otherwise the one from NC.

In order to implement A* we have to provide a cost function g and a heuristic function h defined over ST. We define the cost function $g(s)$ for agent i as the single-robot plan cost function $cost(P, i)$. The multi-robot plan P will consist of the plans already computed with the path found up to s. Obviously the agent o_j will be able to detect conflicts at planning time only for those agents o_i with $i < j$ for which a plan has already been produced. Finally, the heuristic h (i.e. Dijkstra heuristic) is defined as follows:

$$h(< s, t >) = \min_{g \in G} d_{dij}(s, g) \tag{6.11}$$

where $d_{dij}(s, g)$ is the distance from s to g pre-computed by the Dijkstra algorithm, which is independent of time t. It is important to notice that A* will find the optimal solution since the heuristic is admissible, i.e. it underestimates the true costs, for a single robot plan given the subset of already computed paths and ignoring the others (for which no plan was found yet).

For example, let us consider the simple weighted graph depicted in Figure 6.2. $R1$ and $R2$ are respectively the location of two robots $r1$ and $r2$. $G1$ and $G2$ are the goals. In this example, the sequence $<r1, r2>$ has been selected and $r1$ has already produced the following plan: $(< R1, 0 >, < N1, 1 >, < N2, 2 >, < G1, 3 >)$. Now $r2$ has to plan. The only remaining goal is $G2$ since $G1$ has already been selected by $r1$. At first, according to the topology and the already defined plan for $r1$, from $<R2, none>$ the nodes $<R1, none>$ and $<N1, 1>$ can be reached. In fact, if we simulate the plan of $r1$, moving to $R1$ will entail no conflict and would have just the cost of traveling the distance; thus $g(<R1, none>) = 1.2$. In the other case, moving to $R2$ at time 1, will conflict with $r1$ who is moving there at the same time. In this case, our model will tell us that we have to wait for $r1$ to first reach $N1$ (with a cost of 0.8) and then leave it (with a cost of 0.8). Then, we would be able to reach $N1$ with a cost of 1. Thus the total cost of reaching $N1$ at time 1 will be $g(<N1, 1>) = 2.6$ (i.e. 0.8 + 0.8 + 1).

Moreover, the heuristic values for these states (obtained by pre-computing the Dijkstra algorithm) are: $h(<R1, none>) = 1.4$ and $h(<N1, 1>) = 1$. Thus, according to the well-known formula $f(n) = g(n) + h(n)$, $<R1, none>$ will be selected. Similarly,

nodes <$N1$, *none*> and <$G2$, *none*> will be expanded next and the planning process will continue until the plan (<$R2, 0$>, <$R1, 1$>, <$G2, 2$>) is found. Notice that, <$N1$, *none*> is different from <$N1, 1$> since $r2$ will already have moved away from $N1$ at time-step 2.

6.4.3 Monitoring agent

The monitoring agent MA constructs the map represented as the graph G online and identifies the frontier RFIDs, which are RFIDs at incompletely explored areas. Moreover, MA will monitor the local exploration and possibly identify, with one of the previously described techniques, a multi-robot plan in order to move robots to locations offering better performance expectations for the local exploration.

At execution time the robots send to MA their RFID locations (i.e. the nearest RFID they can perceive). Every time a robot changes its RFID position from r_i to r_{i+i}, MA updates the set SL of current robot locations and updates the graph as follows: $E = E \cup \{(r_i, r_{i+1})\}$ and $V = V \cup \{r_i\} \cup \{r_{i+1}\}$.

The monitoring process collects information continuously regarding the unexplored area in the vicinity of the RFIDs based on the local occupancy grid to identify the frontier RFIDs U. Roughly, the robot knows how many RFIDs, given the defined deployment density, should be placed per square meter and which is the density they perceive. Therefore, they are able to compute an estimate on how much the area has been explored in the proximity of the RFIDs they visit.

MA periodically evaluates the position of the robots on the graph and their distance from the frontier nodes U. If this value exceeds a given threshold, it stops the robots and computes a new multi-robot plan. Once a valid plan has been produced, MA starts to drive the robots by assigning to each of them the next RFID prescribed by their plans. Robots plan between RFIDs by using A* on the occupancy grid and the teammate avoidance, as described in Section 6.2.1, but with RFID locations as goals rather than frontier cells. If the plan execution fails due to unobservable RFIDs or obstructed connections, a failure message is sent to MA. MA will consequently remove the node and its edges from the graph G and re-plan. When the target RFID is reached, a task accomplished message is sent to MA, which will assign another task or send a global plan termination message. In the latter case, the robots will start again with local exploration.

Further, during the multi-robot plan execution, the planner monitors exploration for detecting unforeseen situations. For example, if a robot does not send an accomplished task message or an RFID position for a long time, it is considered lost and removed from the robot list. During multi-plan execution deadlocks might occur. This is due to the fact that the current system does not allow to predict the exact order in which tasks will be accomplished. This issue is currently solved by re-planing the assignment if a deadlock occurs, or by restarting local exploration if no better plans can be found.

6.5 Experiments

We utilized a simulated model of the *Zerg* robot, that captures the same physical properties as the real one, e.g. a four wheel drive, an RFID tag release device, an RFID antenna, Inertial Measurement Unit (IMU), and LRF. The sensors of the model are simulated with the same parameters as the real sensors, expect the real RFID reading and writing range. Without loss of generality, these ranges have been extended to two meters, since this parameter mainly depends on the size of the transmitter's antenna, which can also be replaced.

6.5.1 Evaluation of the local approach

The local approach has been tested intensively with the USARSim simulator ([Carpin et al., 2006a, Carpin et al., 2006b, Balakirsky et al., 2006]) for various environments generated by the National Institute of Standards and Technology (NIST). They provide both indoor and outdoor scenarios of the size bigger than $1000\,m^2$, reconstructing the situation after a disaster. Since USARSim allows for the simulated deployment of heterogeneous robot types in a wide range of different scenarios, it offers an ideal performance metric for comparing multi-robot systems. Furthermore, the effort of evaluating large robot teams is greatly reduced. The local approach was applied in USARSim during the RoboCup'06 *Virtual Robots Competition*, where our team won the first prize [*]. During this competition, virtual teams of autonomous or tele-operated robots had to find victims within 20 minutes while exploring an unknown environment. The simulation system was capable of simulating up to 12 robots at the same time. Most of the other teams applied frontier cell-based exploration on global occupancy grids. In particular: selfish exploration and map merging [Nevatia et al., 2006] (IUB), map merging and local POMDP planning [Pfingsthorn et al., 2006] (UVA), operator-based frontier selection and task assignment (SPQR), and tele-operation (STEEL) and (GROK). The competition was held in three preliminary rounds, two semifinals and two finals. In the following the results in terms of exploration and victim discovery of our team *RRFreiburg* compared to the results of other teams at the competition are presented.

Exploration was evaluated based on the totally explored area of each team. These values were automatically computed from the logs of the server hosting the simulator. Table 6.1 gives an overview of the number of deployed robots, and the area explored by each team. It also provides additional information such as the ratio between the totally explored area and the summed distance traveled by all robots (*Area/Length*). Furthermore, the average area explored by a single robot of each team is shown (*Area/#robots*). The result shows clearly that our team was able to deploy the largest robot team, while exploring an area bigger than any other team. Due to the modest computational re-

[*] Virtual Rescue Robots Freiburg: www.informatik.uni-freiburg.de/~rescue/virtual

		RRFreiburg	GROK	IUB	SPQR	STEEL	UvA
Preliminary 1	# Robots	12	1	6	4	6	1
	Area [m^2]	**902**	31	70	197	353	46
	Area/Length	0,65	**1,33**	0,49	0,89	0,98	0,82
	Area/#robots	**75,17**	31	11,67	49,25	58,83	46
Preliminary 2	# Robots	12	1	4	4	6	8
	Area [m^2]	**550**	61	105	191	174	104
	Area/Length	0,35	**0,92**	0,7	0,9	0,83	0,56
	Area/#robots	45,83	**61**	26,25	47,75	29	13
Preliminary 3	# Robots	10	1	5	7	6	7
	Area [m^2]	**310**	59	164	44	124	120
	Area/Length	0,38	**1,02**	0,64	0,88	0,53	0,41
	Area/#robots	31	**59**	32,8	6,29	20,67	17,14
Semifinal 1	# Robots	8	1	6	4	6	6
	Area [m^2]	**579**	27	227	96	134	262
	Area/Length	0,23	0,68	**0,88**	0,67	0,7	0,72
	Area/#robots	**72,38**	27	37,83	24	22,33	43,67
Semifinal 2	# Robots	8	1	6	5	6	7
	Area [m^2]	**1276**	82	139	123	139	286
	Area/Length	0,51	**1,03**	0,91	0,99	0,51	0,71
	Area/#robots	**159,5**	82	23,17	24,6	23,17	40,86
Final 1	# Robots	8	-	8	-	-	-
	Area [m^2]	**1203**	-	210	-	-	-
	Area/Length	0,47	-	**0,93**	-	-	-
	Area/#robots	**150,38**	-	26,25	-	-	-
Final 2	# Robots	8	-	6	-	-	-
	Area [m^2]	**350**	-	136	-	-	-
	Area/Length	0,2	-	**0,53**	-	-	-
	Area/#robots	**43,75**	-	22,67	-	-	-

Table 6.1: Exploration Results from RoboCup '06

sources needed by the local approach, we were able to run 12 robots on a single Pentium4, 3 GHz.

Victim points were awarded for both locating victims and providing extra information. Table 6.2 summarizes the victim information reported by each team. In particular, the table shows the number of victims identified, the bonus for reporting the status (i.e 10 points for each victim status) extra information such as pictures (up to 20 points for each report), and accurate localization of the victims (up to 20 points).

Figure 6.3 (a-b) depicts the joint trajectory of each team generated during the semifinal and final and (c-d) shows the single trajectory of each robot of our team on the same map, respectively. The efficiency of the RFID-based coordination is documented by the differently colored trajectories of each single robot.

		RRFreiburg	GROK	IUB	SPQR	STEEL	UvA
Preliminary 1	reported victims	**5**	2	1	1	2	0
	status bonus	**20**	0	10	0	**20**	0
	victim bonus	-	-	-	-	-	-
Preliminary 2	reported victims	2	1	1	**3**	2	1
	status bonus	20	0	10	**30**	20	0
	victim bonus	-	-	15	-	**30**	-
Preliminary 3	reported victims	**5**	3	3	1	**5**	3
	status bonus	**50**	0	0	0	**50**	0
	victim bonus	-	45	30	-	**60**	-
Semifinal 1	reported victims	**8**	4	7	6	7	6
	status bonus	**70**	20	30	50	60	30
	victim bonus	-	60	**105**	45	90	-
	localization bonus	**80**	20	70	48	70	60
Semifinal 2	reported victims	**16**	4	5	9	14	2
	status bonus	80	0	40	60	**140**	0
	victim bonus	-	45	**75**	**75**	0	-
	localization bonus	**160**	40	50	90	140	20
Final 1	reported victims	**10**	-	3	-	-	-
	status bonus	**60**	-	30	-	-	-
	victim bonus	-	-	**60**	-	-	-
	localization bonus	**100**	-	30	-	-	-
Final 2	reported victims	**10**	-	5	-	-	-
	status bonus	**80**	-	10	-	-	-
	victim bonus	-	-	**30**	-	-	-
	localization bonus	**100**	-	50	-	-	-

Table 6.2: Victim scoring results from RoboCup '06

6.5.2 Evaluation of the global approach

Efficiency in terms of conflict detection and joint path length optimization was evaluated on the basis of both artificially generated, and robot team generated RFID graphs. The artificially generated graphs, consisting of approximately 100 nodes, are weakly connected in order to increase the difficulty of the planning problem, whereas the graph generated by the robots, consisting of approximately 600 nodes, represents a structure naturally arising from an office-like environment.

Figure 6.4 depicts the result from evaluating greedy assignment, genetic optimized assignment, and sequence optimization on these graphs. Each method was applied with a fixed number of randomized goals and starting positions 10 times. We experimented with different sizes of robot teams ranging from 2 to 20. The abrupt ending of the curves indicate the size of the agent team, at which no more solutions could be found, i.e. the scoring function returned infinity. Note that for all the experiments, the genetic algorithm was constrained to compute for no more than one second.

The result makes it clear that sequence optimization helps to decrease both the over-

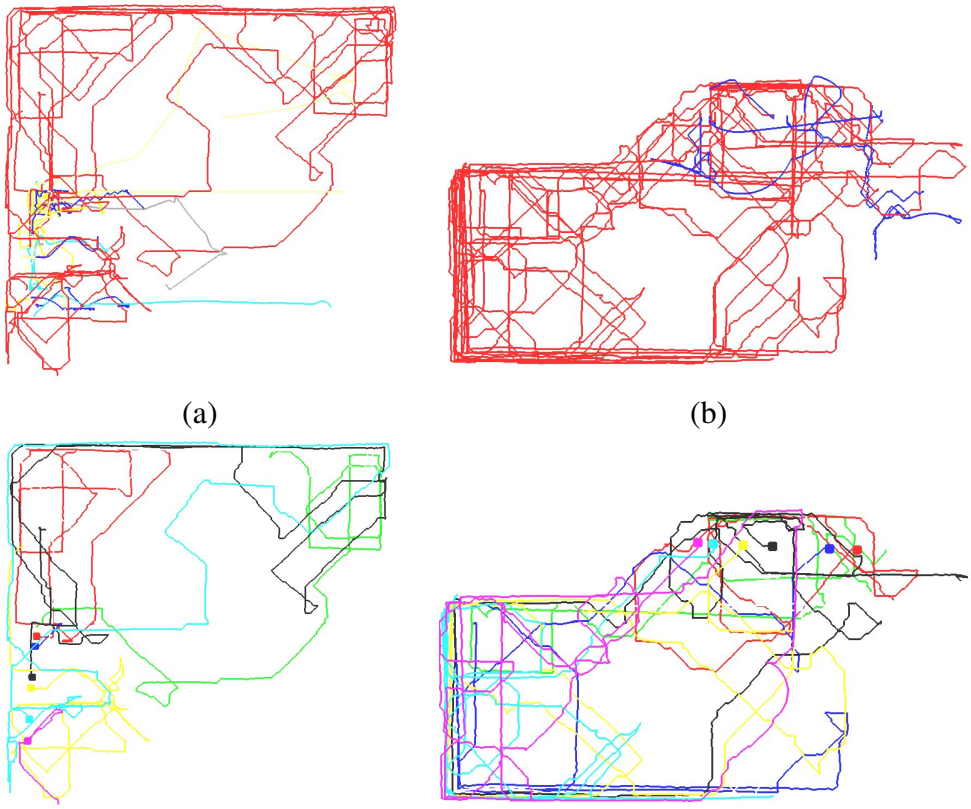


Figure 6.3: Exploration scoring trajectories recorded during the finals: (a,b) Comparison between our approach (red line) and all other teams. (c,d) Coordinated exploration of our robots with each robot represent by a different color.

all path costs and the number of conflicts between single robot plans. Moreover, the method yields solutions with nearly no conflicts on the graph dynamically generated by the robot team (see Figure 6.4 (c)). In order to compare the global and local approach in terms of the explored area, we conducted two experiments on a large map, for 40 minutes each. Due to the global approach, the robots were able to explore 2093 m^2 of the map, in contrast to the team executing the local approach, exploring only 1381 m^2 of the area. The trajectories in Figure 6.5 indicate that this was mainly because the robots running the local approach were not able to overcome the local minima in the long hall. With the global approach, the robots discovered the passage leading to the large area beneath the hall.

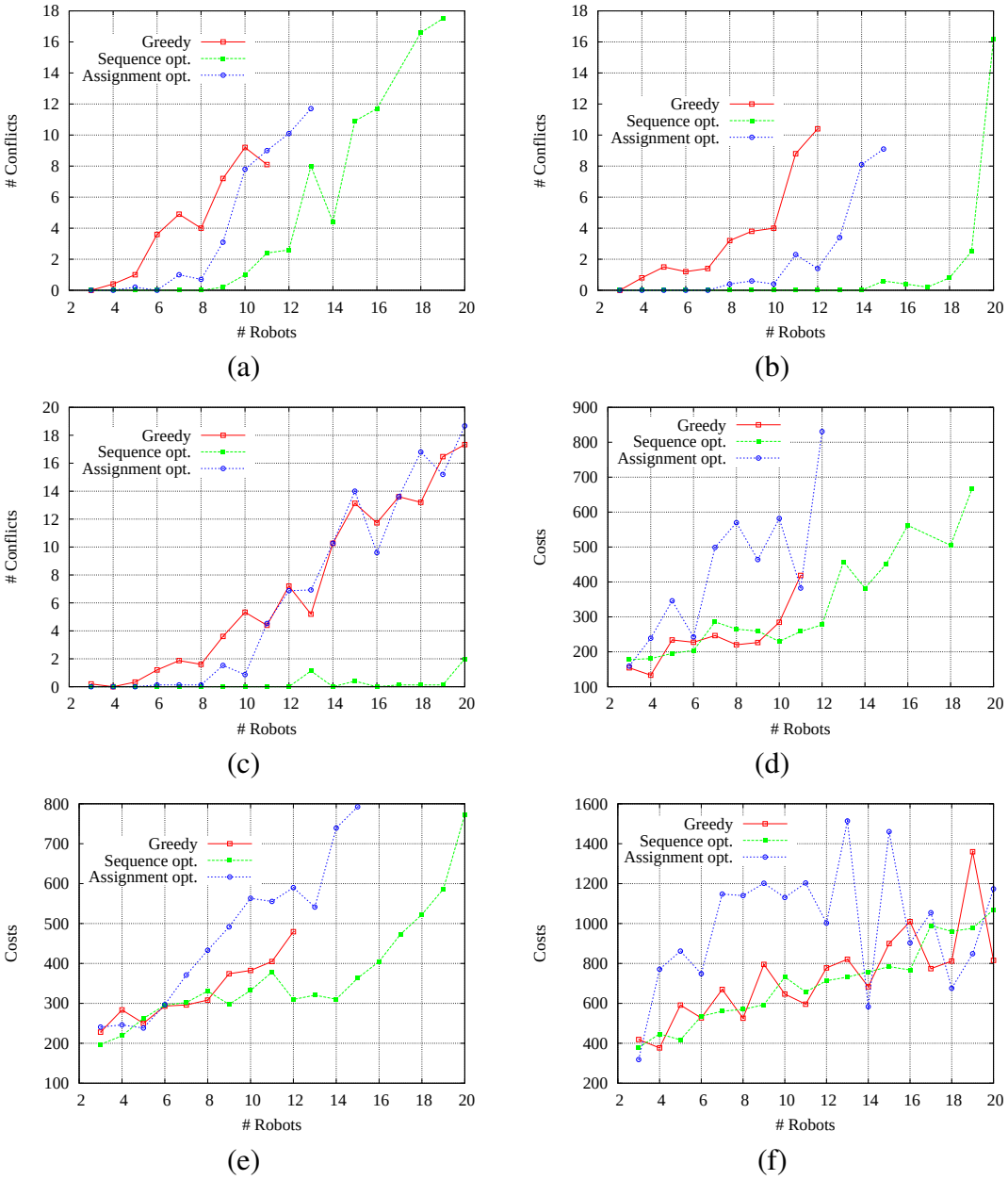
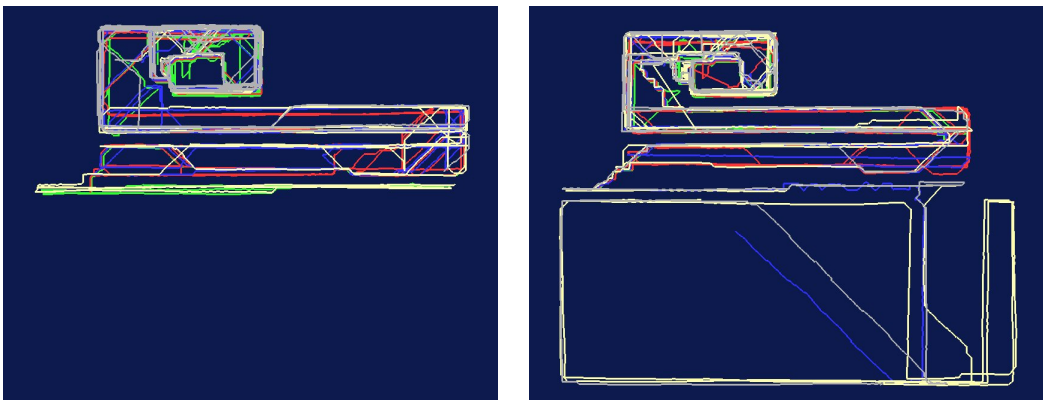


Figure 6.4: Comparing the number of conflicts (a-c) and travel costs (d-f) of the three approaches on different RFID graphs: (a,d) narrow office-like environment, (b,e) narrow outdoor area, (c,f) graph generated from RFIDs deployed by the robots on a USARSim map.



(a) (b)

Figure 6.5: Comparing the locally and globally coordinated exploration. During local exploration (a) robots get stuck in a local minima. The global approach (b) allows the robots to leave the local minima and to explore a larger area

6.6 Related work

There has been a wide variety of approaches to robotic exploration, whereas most of them neither consider the optimization of multi-robot plans, nor do they directly address the problem of reliable map sharing between robots. Furthermore, they require for map exchange a minimal amount of communication bandwidth.

Zlot et al. proposed an approach for coordinating multi-robot exploration guided by a market economy [Zlot et al., 2002]. Their method trades exploration tasks using single-item first-price sealed-bid auctions between the robots. Although the system is partially functional with low bandwidth communication, it is unclear by which extend exploration efficiency decreases if communication is significantly perturbed. Methods for local exploration have already been successfully applied in the past [Balch and Arkin, 1994, Svennebring and Koenig, 2004]. It basically has been shown that multi-robot terrain coverage is feasible without robot localization and communication. In their scenario, robots coordinate their tracks by leaving physical markings in the environment. However, it is unclear whether these markings can be reliably detected by the robots. Furthermore, physically markings are not desirable in every scenario. Burgard et al. [Burgard et al., 2005] contributed a method for greedy task assignment based on grid mapping and frontier cell exploration [Yamauchi, 1997]. Their approach trades off the information gain if reaching frontier cells with the cost of moving to them. They do not consider conflicts between single robot plans, and require robots to start their mission close to each other with knowledge about their initial displacement. Also the idea of compressing maps for reducing the amount of data during communication has been considered [Meier et al., 2005]. In this work, the approximation of grid maps

by polygons has been evaluated. The work by Bennewitz and colleagues [Bennewitz et al., 2001] focuses on the optimization of plans taken by multiple robots at the same time. They select priority schemes by a hill-climbing method that decides in which order robots plan to their targets [Erdmann and Lozano-Perez, 1987]. Plans are generated in the configuration time space by applying A* search on grid maps. Farinelli et al. considered a scenario where tasks to be accomplished are perceived by the robots during mission execution. [Farinelli et al., 2006]. The paper presents an asynchronous distributed mechanism based on Token Passing for allocating tasks in a team of robots.

The coordinated movement of a set of vehicles has also been addressed in other domains, such as in the context of operational traffic control [Hatzack and Nebel, 2001], and the cleaning task problem [Jaeger and Nebel, 2001].

6.7 Conclusion

In this chapter a novel method for coordinated exploration of large robot teams has been contributed. The approach, which is based on RFID technology for indirect communication, is composed of two parts. First, distributed local search, with the notable properties of not requiring direct communication and scaling with the number of agents. Second, global task assignment and multi-robot path planning for monitoring the local exploration, and restarting it in better locations.

The usage of RFIDs allows to build a significantly smaller representation of the environment compared to grid based approaches [Nevatia et al., 2006, Pfingsthorn et al., 2006, Burgard et al., 2005, Bennewitz et al., 2001]. The experimental results from RoboCup show that RFID-based coordination scales with large robot teams exploring large environments.

Moreover, we extensively experimented with three approaches for solving the task assignment and planning problem. The first two are genetic and greedy task assignment techniques [Burgard et al., 2005], whereas the third is a variant of sequential planning [Bennewitz et al., 2001]. It has been shown that genetic optimization, particularly sequence optimization, outperforms the greedy assignment in terms of conflict reduction and path length. With increasing number of robots, greedy assignment fails to provide executable solutions, whereas sequence optimization is able to solve problems with up to 20 robots. Finally, qualitative experiments with the full system have shown that the global method might lead to the exploration of areas that are more than double in size as achieved by the local approach.

In Chapter 4 experiments have been conducted that demonstrated the capability of the real-robot platform *Zerg* to autonomously deploy and detect RFID tags in a cellar environment. Since the simulated robots are running the same software modules as the real robots, we are confident that the proposed exploration system is transferable to the real-robot platform.

There are several issues that have to be considered by future work. First, the approach has to be evaluated on a real robot team exploring a larger area. Second, the plan execution phase of the approach can be further improved. It is considered to represent the generated multi-robot plans by *Petri* nets, allowing to verify online plans generated from task assignment and path planning [Ziparo and Iocchi, 2006]. Specifically multi-robot plans are represented as a discrete event system with RFIDs as resources that may be owned by one robot at the time. Given this representation, plan executions can be simulated for computing a more precise plan evaluation. On the one hand, this facilities the optimization of robot waiting times if they are involved within conflicts. On the other hand, it enables deadlock detection during the planning phase. Finally, the model allows to compute deadlock-free execution policies that can replace the randomized priority ordering.

7 Victim Detection

7.1 Introduction

Robots deployed for exploration and mapping of an area after a disaster are required to perform subtasks autonomously as much as possible. One primary goal is to deploy, under the surveillance of a human operator, a team of robots for coordinated victim search. This requires robots to perform subtasks, such as victim detection, partially or even fully autonomously.

The National Institute of Standards and Technology (NIST) develops test arenas for the simulation of situations after a disaster [Jacoff et al., 2001]. In this real-time scenario, robots have to explore an unknown area autonomously within 20 minutes and to detect victims therein. There might be "faked" victim evidence, such as printed images of human faces, non-human motion, and heat sources that do not correspond to victims. The heat blanket in the fourth row of Figure 7.4, for example, would be wrongly reported as victim by most heat-seeking robots. Note that this example is particularly difficult due to the large size of the thermo signature, as well as the closely located evidence given by the skin-like color of the blanket, the face, and motion.

Due to the real-time constraint in rescue-like applications, only fast computable techniques are admissible. During RoboCup'05 and RoboCup'06 the *RescueRobots Freiburg* team successfully applied simple but fast classifiers for victim detection, such as *color thresholding*, *motion detection*, and *shape detection* on images taken by an infrared and color camera, respectively. However, the detection rate of these classifiers turns out to be moderate, since they are typically tuned for specific objects found in the environment. Hence, in environments containing many diverse objects, they tend to produce a large number of evidence frames, from which in the worst case most are *false-positives*, i.e objects that are wrongly recognized as victims. One solution to this problem is to combine local evidences, i.e. evidences that are close to each other in the real world, and to reason on their true class label with respect to their neighborhood relations. Markov Random Fields (MRFs) provide a probabilistic framework for representing such local dependencies. However, inference in MRFs is computationally expensive, and hence not generally applicable in real-time.

In this chapter a novel approach for the genetic optimization of the building process of MRF models is contributed. The genetic algorithm determines offline relevant neighborhood relations, for example the relevance of the relation between evidence types

heat and motion, with respect to the data. These pre-selected types are then utilized for generating MRF models during runtime. First, the vertices of the MRF graph are constructed from the output of the weak classifiers. Second, edges between these nodes are added if the specific type of nodes can be connected by an edge type that has been selected during the optimization procedure.

Experiments carried out on test data generated in environments of the NIST benchmark indicate that compared to the utilized Support Vector Machine (SVM) based classifier, the optimized MRF models reduce the false-positive rate. Furthermore, the optimized models turned out to be up to five times faster than the non-optimized ones at nearly the same detection rate.

The remainder of this chapter is structured as follows. In Section 7.2 the underlying vision system is introduced, Section 7.3.1 explains the MRF model, and in Section 7.3.2 the genetic model selection approach is introduced. In Section 7.5 related work is discussed. Finally, results from experiments are presented in Section 7.4, and the chapter is concluded in Section 7.6.

7.2 Vision data processing

The utilized vision system is part of the rescue robot *Zerg*, shown in Figure 7.1 (a), which is equipped with a *Hokuyo* URG-X004 Laser Range Finder (LRF), a *ThermalEye* Infra-Red (IR) camera, and a *Sony DFW-V500* color camera. The LRF is capable of measuring distances up to 4000 *mm* within a field of view (FOV) of 240° and the FOV of the IR and color camera are 50° and 70°, respectively. Please see Section 2.5 for further details.

In order to combine evidence from thermo and color images, we first project their pixels onto the 3D range scan, and second determine pixel-pixel correspondence by interpolating from best matching yaw and pitch angles found in both projections. Before camera images are projected onto the scan, they are linearized with respect to the intrinsic parameters of the camera. On color cameras, these parameters are usually calibrated from pixel to real-world correspondences generated by a test pattern, such as the printout of a chess board [Bradski, 2000]. In case of IR camera calibration, it is necessary to generate a test pattern that also appears on thermo images. This has been achieved by taking images from a heat reflecting metal plate covered with quadratic isolation patches in a chess board-like manner.

From both images three different evidence types are generated, which are *color*, *motion*, and *shape*, respectively. Each evidence type is represented by a rectangular region described by the position and size (u, v, w, h) on the image, number of pixels included, and the real world position (x, y, z) of the center.

Color pixels are segmented by fast thresholding [Bruce, 2000] in the YUV color space. In case of the IR camera, only the luminance (brightness) channel is used since

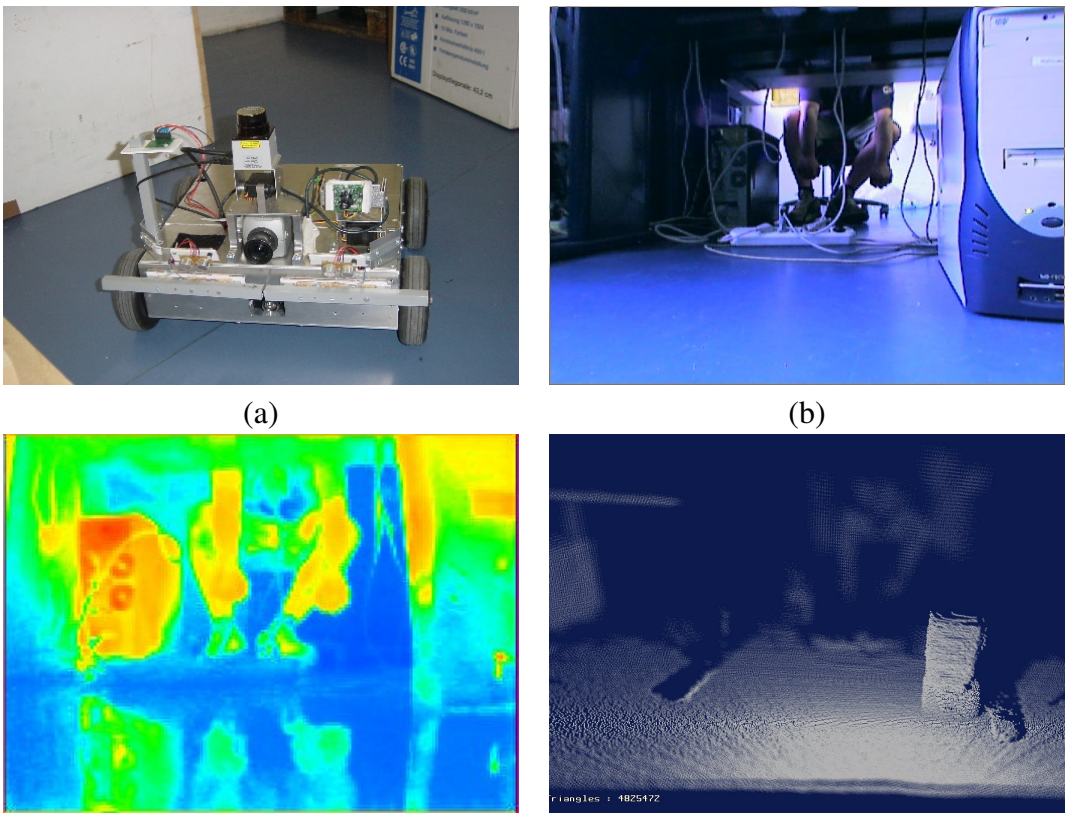


Figure 7.1: The autonomous rescue robot *Zerg* (a) and vision data from the same scene (b)-(d): A color image taken by the CCD camera (b), a thermo image taken by the IR camera (c), and a 3D scan taken by the 3D scanner (d).

thermo images are represented by single values per pixel, which are proportional to the detected temperature. Pixels within the same color class are merged into *blobs* by run length encoding, and represented by rectangular regions.

Motion is detected by background subtraction of subsequent images. Let I_t be an image at time t from a sequence of images $\mathbf{I}$ with $I_0 = Background$. Then, the difference between an image and the background can be calculated by $AVG_t = (1-\beta)\,AVG_{t-1} + \beta I_t$, $DIFF_t = AVG_t - I_t$, where AVG is the running average over all images and β a factor controlling the trade-off between latency and robustness. Pixels labeled as foreground are also merged into groupings by run length encoding and are represented by a set of rectangular regions.

Shape detection is currently limited to the detection of human faces. We use the *openCV* [Bradski, 2000] implementation of the method from Viola and colleagues [Viola and Jones, 2001], which has been further improved by Lienhart [Lienhart and

Maydt, 2002]. The method utilizes a cascade of *haar*-like features that are trained and boosted from hundreds of sample images scaled to the same size. Since the classifier was mainly trained from images with faces aligned in the vertical direction, we rotated images for allowing the detection of faces aligned horizontally.

7.3 Genetic model optimization

7.3.1 Markov Random Fields (MRFs)

The series of images in Figure 7.4 (a) clearly show that single evidences are not sufficient to uniquely identify victims. Therefore, it is necessary to consider neighborhood relations in order to reduce false-positive detections. Markov Random Fields (MRFs) provides a probabilistic framework for representing local dependencies. A MRF is defined by an undirected graph $\mathcal{G} = (\mathrm{Y}, \mathcal{E})$, where Y is a set of discrete variables $\mathrm{Y} = \{Y_1, \ldots, Y_N\}$, and $\mathcal{E}$ is a set of edges between them. Each variable $Y_i \in \{1, \ldots, K\}$ can take on one of K possible states. Hence, $\mathcal{G}$ describes a joint distribution over $\{1, \ldots, K\}^N$.

According to the approach of Anguelov et al. [Anguelov et al., 2005], we utilize *pairwise* Markov networks, where to each node a potential $\phi(y_i)$ and to each undirected edge $\mathcal{E} = \{(ij)\}\,(i < j)$ between two nodes a potential $\phi\left(y_i, y_j\right)$ is associated. Consequently, the pairwise MRF model represents the joint distribution by:

$$P_\phi(\mathrm{y}) = \frac{1}{Z} \prod_{i=1}^{N} \phi_i(y_i) \prod_{(ij)\in\mathcal{E}} \phi_{ij}(y_i, y_j), \tag{7.1}$$

where Z denotes a normalization constant, given by $Z = \sum_{\mathrm{y}'} \prod_{i=1}^{N} \phi_i(y'_i) \prod_{(ij)\in\mathcal{E}} \phi_{ij}(y'_i, y'_j)$. A specific assignment of values to Y is denoted by y and represented by the set $\left\{y_i^k\right\}$ of $K \cdot N$ indicator variables, for which $y_i^k = I(y_i = k)$. In order to foster the associativity of the model, we reward instantiations that have neighboring nodes, which are labeled by the same class. This is enforced by requiring $\phi_{ij}(k, l) = \lambda_{ij}^k$, where $\lambda_{ij}^k > 1$ for all $k = l$, and $\phi_{ij}(k, l) = 1$ otherwise [Taskar et al., 2004]. Inference is carried out by solving the *maximum a-posterior (MAP)* inference problem, i.e. to find $\arg\max_{\mathrm{y}} P_\phi(\mathrm{y})$.

For our specific problem, we define the node potentials $\phi(y_i)$ by a vector of features which indicates the quality of the node's corresponding evidence frame. These features are the size of the evidence frame, the number of included pixels, and the real world distance taken from the 3D range measurement. Likewise we define the edge potentials $\phi\left(y_i, y_j\right)$ by a vector of features that indicates the quality of neighborhood relations. Edges are build from combinations of the evidence types introduced in Section 7.2. In our case there exist 36 possible edge types, given the 6 different types of evidence. For

example, the edge type *isWarmSkin* describes the combination of the features *heat* and *skinColor*. The feature vector of an edge includes the type of the edge and the real-world distance measurement between both nodes. Some edge features are shown in Table 7.1.

	IR-Heat	**IR-Motion**	**IR-Face**
IR-Heat	*isSameType*	*isWarmMotion*	*isWarmFace*
IR-Motion	-	*isSameType*	*isMovingFace*
IR-Face	-	-	*isSameType*
Color-Skin	-	-	-
Color-Motion	-	-	-
Color-Face	-	-	-
	Color-Skin	**Color-Motion**	**Color-Face**
IR-Heat	*isWarmSkin*	*isWarmMotion*	*isWarmFace*
IR-Motion	*isMovingSkin*	*isWarmMotion*	*isMovingFace*
IR-Face	*isSkinFace*	*isMovingFace*	*isWarmFace*
Color-Skin	*isSameType*	*isMovingSkin*	*isSkinFace*
Color-Motion	-	*isSameType*	*isMovingFace*
Color-Face	-	-	*isSameType*

Table 7.1: Feature combinations of evidence types *heat/skin*, *motion*, and *face*, generated from color images and thermo images, respectively.

The MRF graph is dynamically constructed for each image from the video stream during runtime (see Figure 7.2 for an example). First, we generate from the image data six sets of evidence frames, as described in Section 7.2. From these sets, six types of MRF nodes are generated by calculating the feature vectors for each node potential, where each node type corresponds to one type of evidence. Second, edges between nodes are generated. Each node connects to the four closest neighbors in its vicinity if they are within close real-world distance, which was maximally 60*cm* in our implementation. Finally, for each edge a feature vector for its edge potential is calculated.

For the sake of simplicity, we represent potentials by a log-linear combination $\log \phi_i(k) = \mathrm{w}_n^k \cdot \mathrm{x}_i$ and $\log \phi_{ij}(k,k) = \mathrm{w}_e^k \cdot \mathrm{x}_{ij}$, where x_i denotes the node feature vector, x_{ij} the edge feature vector, and w_n^k and w_e^k the row vectors according to the dimension of node features and edge features, respectively. Consequently, we can denote the MAP inference problem $\arg\max_{\mathrm{y}} P_\phi(\mathrm{y})$ by:

$$\arg\max_{\mathrm{y}} \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathrm{w}_n^k \cdot \mathrm{x}_i) y_i^k + \sum_{(ij)\in\mathcal{E}} \sum_{k=1}^{K} (\mathrm{w}_e^k \cdot \mathrm{x}_{ij}) y_i^k y_j^k. \tag{7.2}$$

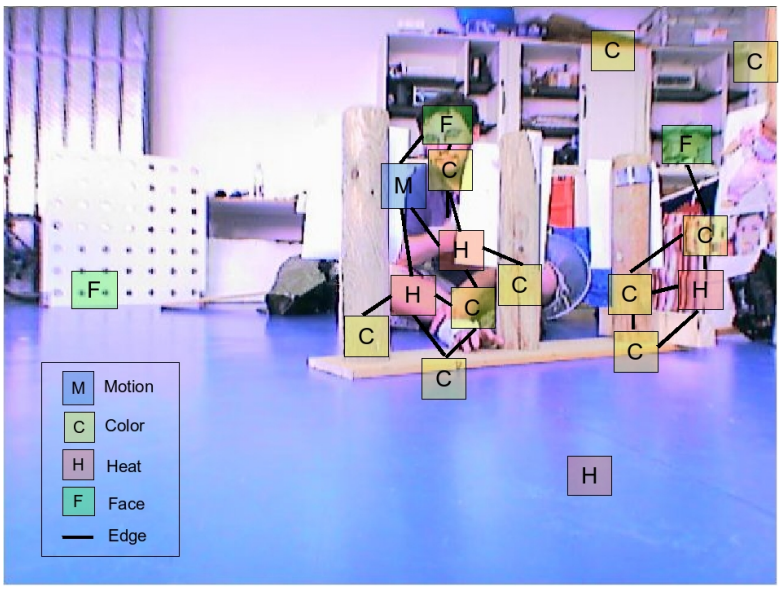


Figure 7.2: MRF graph online constructed from features detected in the thermo and color images. Note that the number of shown features has been reduced for the sake of readability.

Equation 7.2 can be solved as a linear optimization problem by replacing the quadratic term $y_i^k y_j^k$ with the variable y_{ij}^k and adding the linear constraints $y_{ij}^k \leq y_i^k$ and $y_{ij}^k \leq y_j^k$. Hence, the linear programming formulation of the inference problem can be written as:

$$\begin{aligned} \max \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathrm{w}_n^k \cdot \mathrm{x}_i) y_i^k + \sum_{(ij)\in\mathcal{E}} \sum_{k=1}^{K} (\mathrm{w}_e^k \cdot \mathrm{x}_{ij}) y_{ij}^k \qquad (7.3) \\ \text{s.t. } y_i^k \geq 0, \quad \forall i,k; \qquad \sum_k y_i^k = 1, \quad \forall i; \\ y_{ij}^k \leq y_i^k, \quad y_{ij}^k \leq y_j^k, \quad \forall ij \in \mathcal{E}, k, \end{aligned}$$

which can, for example, be solved by the *Simplex* method. Furthermore, it is necessary to learn the weight vectors for the node and edge potential from data, which has been carried out by utilizing the *maximum margin* approach recommended by Taskar and colleagues [Taskar et al., 2003].

7.3.2 Model selection

In general, solving the MAP inference problem, as shown in Section 7.3.1, is NP-hard [Shimony, 1994]. Also in case of the considered two-class problem one notices an increase in computation time if the number of nodes and edges, and thus size of the

linear programming problem, grows.

Computation time is an important issue if applying the detection method, for example, to live video streams taken by a camera in a Search and Rescue scenario. Furthermore, the efficiency of specific evidence correlations (edge types in the MRF graph) depends on the scenario where the classifier is applied. For example, heat sources might be a stronger evidence for human bodies in an outdoor scenario as it would be in an indoor scenario with many heat sources, such as PCs and radiators. Therefore, our goal is to reduce the computation time of the MRF model by selecting edge types that significantly improve the classifier with respect to the data. For example, measurements from different sensor types are generally more significant than measurements from a single sensor type. The two edges between the motion node and the two heat nodes in Figure 7.2 are more valuable than a single connection between heat nodes only. However, by selecting the four closest nodes both heat nodes would be connected.

Therefore, we examined the contribution of specific edge types to the overall detection rate. This has been carried out by learning MRF models with different sets of activated edge types while measuring accuracy and computation time needed for inference. Figure 7.3 summarizes the result, where each data point corresponds to a specific combination of edge types. MRF inference needs between 2 ms and 16 ms, depending

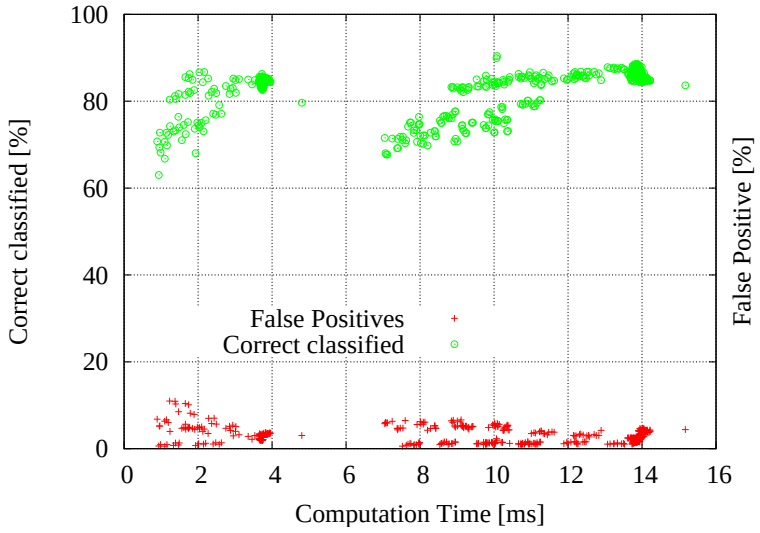


Figure 7.3: Comparing model complexity (computation time) with the percentage of correctly classified vision features (green), and the percentage of false positives (red). Each data point corresponds to a MRF model with specific types of edges activated.

on the combination of activated edge features. Surprisingly, a higher amount of computation time does not necessarily yield better classifier performance. Good classifier performance can be achieved already at a much smaller computation time than needed

for computing models containing all types of edges if the significant edge types are activated.

However, since the complexity of exhaustive search is in $O(2^n)$, and learning a single classifier takes a comparably high amount of time, finding optimal edge types is intractable in the general case. Therefore, we applied a genetic algorithm for selecting the most efficient combinations of edges. The scoring function for guiding the search has been defined by the trade-off between classifier performance and computation time:

$$S = U - \alpha C(p_i), \tag{7.4}$$

where U corresponds to the utility metric, $C(.)$ denotes a cost function reflecting the model complexity, p_i denotes the *ith* permutation, and α is a parameter regulating the trade-off. Depending on the needs of the specific application, U can be computed, for example, from the negative *false-positive* rate, the total number of correctly classified evidence frames, and the percentage of the correctly classified area. Without loss of generality, we decided to use the area-based utility metric since it enforces the detection of body silhouettes rather than frames on their own. The series in Figure 7.4 (c) depicts this metric for true positives by the blue cluster, which was build by the union of all true positive evidence frames in the image.

The scoring function is utilized as fitness function for the genetic algorithm (GA). Solutions, i.e. specific combinations of edge types, are represented for the genetic optimization as a binary string. Each edge type is represented by a bit, and set to *true* or *false* regarding the activation of the corresponding edge type. In order to guarantee that good solutions are preserved within the genetic pool, the so-called *elitism* mechanism, which forces the permanent existence of the best found solution in the pool, has been used. Furthermore, we utilized a simple one-point-crossover strategy, a uniform mutation probability of $p \approx 1/n$, and a population size of 10. In order to avoid that solutions are calculated twice, all computed solutions are memorized by their binary string in a hash map.

7.4 Experiments

We generated more than 6000 labeled examples from video streams recorded in a NIST arena-like environment and split them into three folds. The training data contained true evidence which is exclusively generated from human bodies and false evidence, which is generated from artificial sources, such as a heat blanket, laptop power supply, printouts of faces, moving objects, and objects with skin-like texture, such as wood. Figure 7.4 (a) depicts some examples from the training data. Each color frame corresponds to an evidence type and green frames correspond to face detection, orange frames to heat detection, red frames to color detection, and yellow frames to motion detection. Note

that the training data contained intentionally many cases in which the vision system produces ambiguous evidences. From this data, MRF models were trained by K-fold cross-validation, with $K = 3$.

In order to evaluate the model selection, we reduced the total set of edge types from 32 to 9 since in our case some feature combinations represent the same concept, as for example, the combinations of *heat* from the thermo images together with *face* from the color images, and *face* and *heat* both from the thermo image. The genetic model selection was evaluated by multiple runs with varied parameter α and varied scoring metric (Equation 7.4), e.g. based on the false-positive rate, total error, and total area. In average, the genetic selection yielded the optimal solution after considering 60 ± 7 models, which is more than eight times faster than performing exhaustive search over all 512 possible models.

For the selection of the final MRF model, we utilized the area-based scoring metric with $\alpha = 2.0$. The genetic algorithm selected a classifier which activates, for example, the edge types $motion \wedge face$, $heat \wedge skin$, $heat \wedge face$, $heat \wedge motion$, and forbids $motion \wedge skin$, as well as all edge types between the same kind of nodes. Finally, the selected classifier reached an accuracy of 87.9% at 2.3 ms, in contrast to the classifier with all edges activated (88.76 % at 12.59 ms) and the classifier with no edges activated (71.33 % at 1.14 ms). Figure 7.4 shows some examples of the successful application of the classifier even to hard cases, such as small finger movements, and test persons completely surrounded by faked evidences.

	SVM			MRF		
	False Pos.	False Neg.	Err. [%]	False Pos.	False Neg.	Err. [%]
Human	23	433	39.4	26	143	14.6
Faked	758	0	11.3	151	0	2.3
Both	703	2689	32.8	484	836	12.8
Total	1484	3122	21.0	661	979	7.5

Table 7.2: Comparison of the SVM and MRF classifier: Numbers denote the amount of wrongly classified evidences in images containing humans, faked evidence, and both.

We compared the performance of the optimized MRF model with a Support Vector Machine (SVM) [Vapnik, 1995] based classifier. The SVM was trained on the same features as they were generated for the MRF model, shown in Section 7.2, however, without encoding of the links between the features. In Table 7.2 the performance for classifying single evidence frames of both classifiers is reported. The results have been partitioned into three sets showing the performance on examples containing human evidence, faked evidence, and both. The result indicates that the optimized MRF model performs better in terms of false-positive classifications, particularly in situations con-

taining exclusively faked data.

In the context of Search and Rescue it is desirable to reach a high true-positive rate on each image, i.e. humans are detected reliably, and a low false-positive rate, i.e. no victim alarm from faked evidence. This is not directly expressed by the percentage of correctly classified frames since one wrongly detected frame within an image suffices to trigger the false alarm, even all the others are detected correctly. Therefore, we counted the true-positive and false-positive rate for both classifiers *image-wise*, i.e. images are counted as true-positive and false-positive if there is a single correct and a single wrong evidence found, respectively. It turned out that for images containing human evidence both MRF and SVM reported a victim correctly in 100% of the cases, whereas in images containing faked evidence, the SVM wrongly reported a victim in 60% and the MRF in 13% of the cases. Note that the result of the MRF model is comparably good, since the training data also contained images with more than three different types of faked evidence at the same time, which makes in this case a distinction from human beings impossible.

7.5 Related work

Human body detection and tracking has been mainly carried out based on background subtraction [Wren et al., 1997, Beleznai et al., 2004, Han and Bhanu, 2003] and feature-based detection [Viola et al., 2003, Cutler and Davis, 2000]. *Pfinder* is a system for real-time detection and tracking of human bodies based on background subtraction [Wren et al., 1997]. The background model, which is continuously updated, utilizes Gaussian distributions in the YUV color space at each pixel. Humans are modeled by blobs that are characterized by color, size, and the corresponding Gaussian distributions. It adopts a Maximum A Posterior (MAP) approach in order to assign pixels either to blobs or to the background. Beleznai et al. introduced a system for human body detection that considers the intensity difference between an input frame and a reference image based on the mean shift procedure [Beleznai et al., 2004]. Their method can be applied in real-time since all computation is performed on integral images. Han et al. examined the problem of image registration of both thermal and color images [Han and Bhanu, 2003]. They propose a genetic algorithm-based hierarchical search approach for image registration, which works on simple body silhouettes generated from background subtraction of both image types. Viola et al. are using a classifier trained on human shape and motion features [Viola et al., 2003]. The detector uses images as inputs and efficiently extracts simple rectangular features using integral images. A cascade of classifiers is created to achieve superior detection and low false positive rates. Each stage of the cascade is trained on true and false positives from the previous stage using *Adaboost*. Cutler et al. presented a technique for detecting periodic motions patterns, such as walking [Cutler and Davis, 2000]. They apply time-frequency analysis based

on the short-time Fourier transform (STFT), and autocorrelation for robust periodicity detection and analysis.

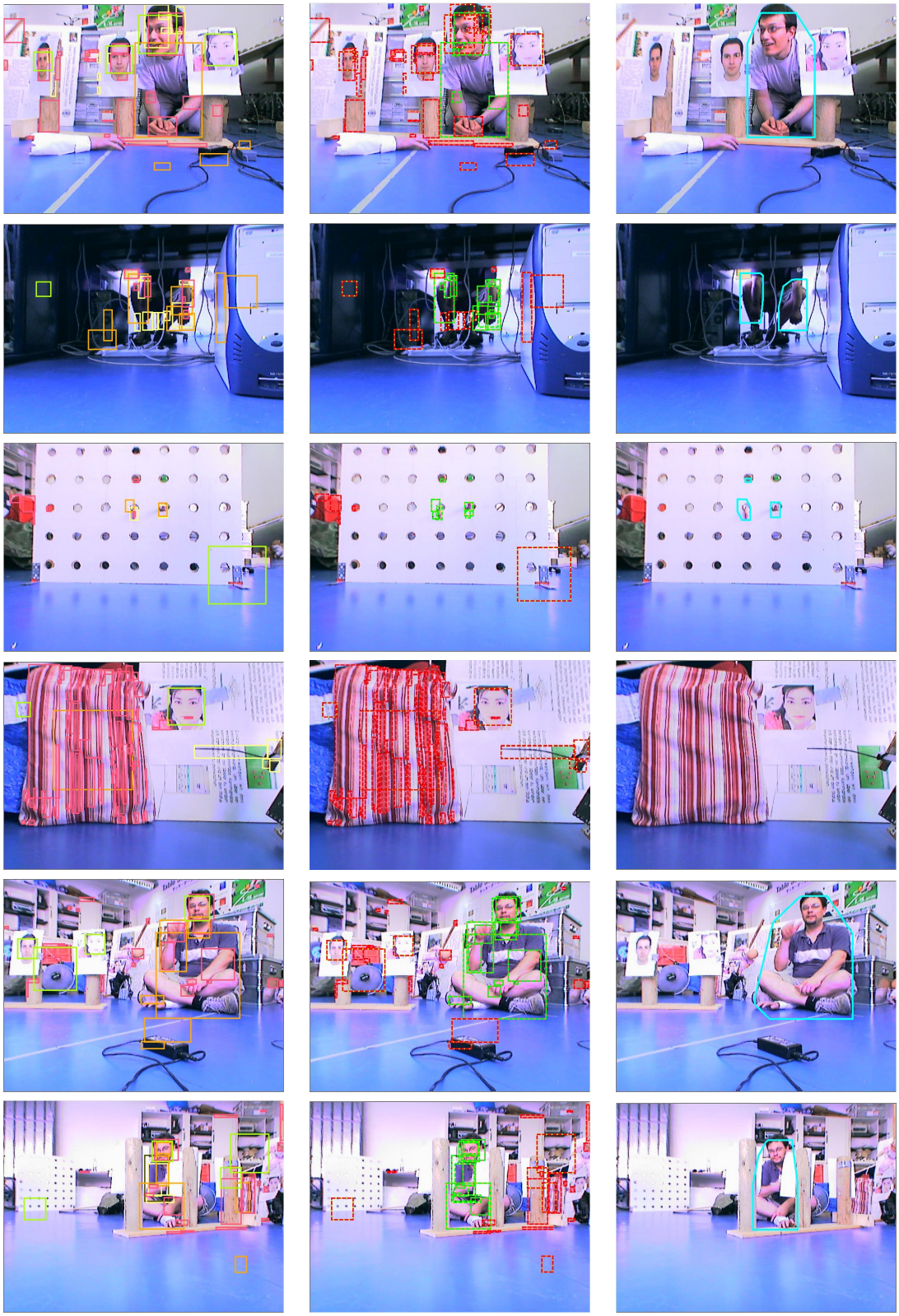
Support Vector Machines (SVMs) have been utilized for detecting human motion [Cao et al., 2004, Sidenbladh, 2004]. For example, Cao et al. presented a real-time system for motion detection using SVMs [Cao et al., 2004]. They utilized sets of filtered images, each encoding a short period of motion history, as input to the SVM. However, the described system seems not to be suitable for harsh environments, since it requires human motion to take place in the center of the image and trajectories have to be performed perpendicular to the optical axis of the camera. MRFs have been successfully applied to pedestrian tracking [Wu and Yu, 2006] and face detection [Dass et al., 2002]. For example, Dass et al. consider the spatial distribution of gray levels within images of human faces [Dass et al., 2002]. Feature selection has been examined in various contexts, e.g. Kohavi et al. provides a good overview on various methods [Kohavi and John, 1997].

In the context of rescue robotics, Bahadori et al. studied various techniques from computer vision that have been applied to human body detection [Bahadori and Iocchi, 2003]. Nourbakhsh et al. utilized a sensor fusion approach for incorporating the measurements from a microphone, IR camera, and conventional CCD camera [Nourbakhsh et al., 2005]. They assigned to each sensor a confidence value indicating the certainty of measurements from this sensor and calculated the probability of human presence by summing over all single sensor observation probabilities, weighted by their confidence value. However, in contrast to the proposed MRF approach, their method does not reflect local dependencies of detected evidence in the model.

7.6 Conclusion

We contributed a system that creates MRF models in real-time from motion, color, and shape evidence, detected by a CCD camera and IR camera, respectively. In order to reduce computational complexity during inference, the building process of models was optimized by a genetic algorithm, which decides offline relevant edge types with respect to the data. Finally, the selected classifier was five times faster than the model with all edge types activated, while gaining optimal performance in terms of the complexity trade-off, and near-optimal performance in terms of accuracy. We compared the optimized model with a SVM and showed that the false-positive rate was significantly reduced, which is an important aspect when considering victim detection in the context of rescue robotics. From an image-wise evaluation of the classifier it can be concluded that the approach reliably detects victims if present and only in hard cases, i.e. if the number of faked evidences is high, false-positives occur. The classifier performance could be further improved by introducing temporal relations, i.e. by adding edges between evidences found in preceding images from the video stream.

The proposed approach can be considered as a general framework for combining human evidence detected by sensors. For example, it can easily be extended for incorporating other types of human evidence, such as audio noise, e.g. tapping, and CO_2 emission. Also given these evidence types, it will be interesting to determine correlations between them that significantly contribute to the classifier's performance. Furthermore, we will consider in future work the extension of the class variable by classes describing the victim's state, e.g. *aware* and *unconscious*. These classes can also be determined from correlations between different evidence types. For example, no detection of motion, but detection of CO_2 emission indicates the victim's unconscious.



(a) (b) (c)

Figure 7.4: Examples from the test data: (a) Detected evidences: skin color (red), heat (orange), motion (white), face (green), (b) the same evidences after classification and (c) all positive evidences clustered into areas.

8 Human In The Loop

8.1 Introduction

One challenge in disaster response is to join local observations made by first responders in the field, such as blocked or unblocked connections, and found victims, at a central command post in order to coordinate and schedule search and rescue missions. "Human in the loop" stands for Multi-Agent-Systems (MAS) that directly combine perceptions and actions from humans with the decision making of the agents. In fact, humans are typically connected to the MAS via so-called proxies. Researchers in the field of Multi-Agent Systems (MAS) developed a rich set of solutions for efficiently coordinating agents, as well as simulators for evaluating these methods. The *RoboCup Rescue* simulation system aims at simulating large-scale disasters and exploring new ways of autonomous coordination of rescue teams [Kitano et al., 1999]. The goal here is to provide a software system that reacts to simulated disaster situations by coordinating a group of simulated agents such as police, ambulance and firefighters. This goal leads to challenges like the coordination of heterogeneous teams with more than 30 agents, the exploration of large-scale environments in order to localize victims, as well as the scheduling of time-critical rescue missions.

Applying the simulation system to the real world is problematic since it lacks the interface to the real world information. Currently, the simulation system relies on carefully designed special-purpose map data and observations generated by the simulation itself, e.g. agent motion is computed by a traffic simulator. However, by using wearable computing technology, we can provide real-world observations to the simulation system. This has three uses: First, it can be used to record real-world data from real-world intervention scenarios and by this, assess the validity of the simulation. Second, it can be used to observe the reaction of the multi-agent systems to real-world data. Third, it is a step towards using both the simulation system and the multi-agent systems to support incident commanders and responders in training and real interventions by providing autonomous decision support and faster-than-real-time simulation for strategy decisions.

In this chapter, we propose a wearable computing device for acquiring disaster relevant data, such as locations of victims and blockades, and show the process of successive data integration into the RoboCup Rescue simulation system. We assume that the locations of first responders are automatically tracked, which can either be carried out

by GPS positioning or PDR combined with RFID-SLAM (see Chapters 3 and 4). Communication between wearable computing devices and the server is carried out based on the open *GPX* protocol [TopoGrafix, 2004] for GPS data exchange, which has been extended for additional information relevant to the rescue task. By providing examples, we show how the data can be integrated and how rescue missions can be optimized by solutions developed on the RoboCup Rescue simulation platform. For this purpose, we introduce solutions that were developed by the *ResQ Freiburg 2004* team [Kleiner et al., 2004], the winner of RoboCup 2004 *. Besides planning and learning techniques, this package includes methods for the efficient coordination of victim search, and an any-time rescue sequence optimization based on a genetic algorithm.

The utilized multi-agent system was extensively evaluated during the Rescue simulation competition in 2004. The proposed results provide a comparison of the agent team with other teams, particularly in terms of exploration and mission scheduling. The efficiency of these methods combined with the demonstrated real-world data integration indicates that wearable computing technology combined with MAS technology can serve as a powerful tool for Urban Search and Rescue (USAR). The proposed interface was demonstrated during the *Infrastructure* competition at RoboCup 2006, where it was awarded the first prize [Kleiner et al., 2006a].

The remainder of this chapter is structured as follows. The interface between human rescue teams and the rescue simulator is proposed in Section 8.2. In Section 8.3 we introduce agent based solutions for search and rescue that can be applied to collected real-world data. In Section 8.4 we propose results from experiments on integrating real-world data into the Rescue simulation system, as well as results from evaluating the deployed multi-agent system. Finally, we discuss in Section 8.5 related work and conclude in Section 8.6.

8.2 Interfacing human first responders

In wearable computing, one main goal is to build devices that support users during their primary task with little or no obstruction. Apart from the usual challenges of wearable computing [Starner, 2001a, Starner, 2001b], in the case of emergency response, the situation of the responder is a stressful one. In order to achieve primary task support and user acceptance, special attention has to be given to user interface design. For this application, users need the possibility to enter information about their observations, and need feedback from the system, which recorded and transmitted the information. Furthermore, the user needs to receive task-related instructions from the command center, such as assignments to victim search tasks.

The implementation has to cope with multiple unreliable communication systems,

*Note that the Open Source version of the software is freely available [ResQ Freiburg, 2004].

such as existing cell phone networks, special-purpose ad-hoc communication, and existing emergency-response communication systems. As the analysis of the different properties of these communication systems is beyond the scope of this chapter, we will therefore abstract from them and assume an unreliable IP-based connectivity between the mobile device and a central command post. This assumption is motivated by the fact that both infrastructure-based mobile communication networks and current ad-hoc communication systems can transport IP-based user traffic. For example, the German company *IABG* implemented *HiMoNN* (High Mobility Network Node), a mobile ad-hoc network for catastrophe management allowing rapid information transfer and secure communications via IP [IABG mbH, 2007].

The situation of the device and its user is also characterized by harsh environmental conditions related to emergency response, such as fire, smoke, floods, wind, and chemical hazards. The device has to remain operable under such conditions and additionally has to provide alternative means of input, e.g. via gestures or textile devices, and output, e.g. head-mounted display and audio, under conditions that affect human sensing and action abilities. Moreover, the system has to be integrated into the processes of emergency response, i.e. having no impact on response times of units and therefore should be integrated into the existing gear of responders.

The wearable system has to track the location of the person continuously, which can either be carried out by GPS positioning, if the person walks outdoors, or by the PDR method described in Chapter 3, if the person walks indoors or close to buildings. Furthermore, RFID tags, which are already present or which are actively distributed, have to be detected, and crucial information, such as data on victims, has to be logged by the system. If communication is possible, recorded data has to be transmitted to a central command post, which utilizes all trajectories and RFID observations for jointly optimizing the map by the RFID-SLAM method described in Chapter 4. Figure 8.1 depicts the functional modules of the wearable system.

8.2.1 Preliminary test system

In order to analyze the properties of the communication and localization system, and to test the software interface to the central server, a preliminary test system has been implemented, for which three requirements have been dropped: The design criteria according to harsh environmental conditions, alternative input methods and HMD-based user interface, and the PDR-based indoor localization capability. Note that the latter has been already extensively evaluated in Chapter 4.

The communication and localization system is independent of the user requirements except for the fact that the system has to be portable. Therefore, we chose a mobile GPS receiver device and a GSM cell phone device as our test implementation platform. The GPS receiver uses the bluetooth [IEEE, 2002] personal area network standard to connect to the cell phone. The cell phone firmware includes a Java VM, based on the

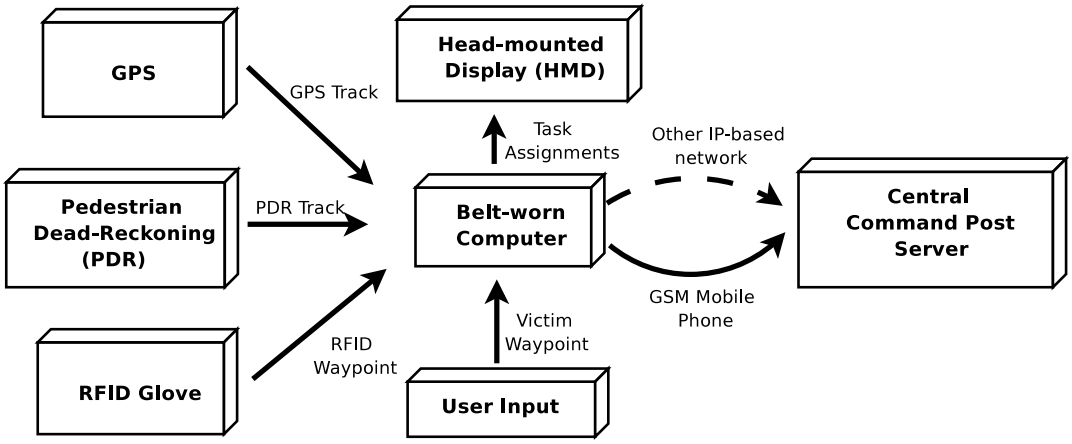


Figure 8.1: System diagram of the full system based on a GSM phone for data transmission connected to a belt-worn wearable computer, RFID reader, positioning device (either GPS or PDR), and a HMD.

J2ME standard with JSR82 extensions, allowing Java applications, which are running on the VM, to present their user interface on the phone, and to communicate directly with nearby blue-tooth devices, as well as with Internet hosts via the GSM network GPRS standard.

The implementation of the test application is straightforward: It regularly decodes the current geographic position from the NMEA data stream provided by the GPS receiver and sends this information to the (a priori configured) server IP address of the central command center. The utilized protocol between the cell phone and the command center is based on the widely used GPX [TopoGrafix, 2004] standard for GPS locations. Among other things, the protocol defines data structures for *tracks* and *waypoints*. A track is a time stamped sequence of visited locations, whereas a waypoint describes a single location of interest, e.g. the peak of a mountain. We extended the protocol in order to augment waypoint descriptions with information specific to disaster situations. These extensions allow rescue teams to report the waypoint-relative locations of RFIDs, road blockades, building fires, and victims. Currently, the wearable device automatically sends the user's trajectory to the command center, whereas additional information, such as the locations of victims or hazards, can be entered manually. A detailed description of the protocol extension can be found in Appendix 10.1.

8.2.2 Wearable emergency-response system

In order to fulfill all stated requirements, the full system was designed with additional hard- and software components. The system uses a wearable CPU core, the so-called

qbic belt-worn computer [Amft et al., 2004] (see Figure 8.2 (a)). It is based on a ARM CPU running the Linux operating system, has a bluetooth interface, and can be extended via USB and RS232 interfaces. The wearable CPU core runs the main application program. For localization, the same mobile GPS receiver as in the test system is used, but can be replaced by a non-bluetooth serial device for increased reliability. For communication, the system can use multiple technologies, the GSM cell phone used for the test system, can be one of those [†].

As already stated, the design of the user interface is crucial for this application. Therefore, we use a glove as wearable user input device [Lawo et al., 2006] and a wireless link between the user interface device and the wearable computer. Such an interface has already been designed for other applications, such as aircraft cabin operation [Nicolai et al., 2005]. Also the detection of RFIDs is carried out via the glove (see Figure 8.2 (e)). This facilitates the automatic detection of RFIDs when the user touches objects, such as a door frame or door knob, which were labeled with RFID tags.

The primary output device is a head-mounted display that can be integrated into existing emergency-response gear, such as firefighter helmets and masks (see Figure 8.2 (d)). In applications where headgear is not commonly used, the output can be provided also through a body-worn display device or audio output. The application software driving the user interface is based on the so-called *WUI toolkit* [Witt et al., 2006], which uses an abstract description to define user interface semantics independent of the input and output devices used. The application code is therefore independent of the devices, and robust towards different instances of the implementation, i.e. with or without head-mounted display. The WUI toolkit can also take context information into account, such as the user's current situation, in order to decide by which device and in what form output and input are provided.

8.3 Multi Agent Systems (MAS) for disaster management

As has been shown in Chapter 4, the information collected by the central station can be used for computing a globally consistent map. This map representation can be utilized further for the automatic optimization of search and rescue strategies by Multi-Agent Systems (MAS) methods. The communication protocol described in Section 8.2 has been particularly tailored according to the protocol of the RoboCup Rescue simulation kernel. This is motivated by the long-term goal of providing real-world data to the MAS community, as well as to facilitate the development of robust MAS methods for

[†] As we assumed IP-based connectivity, flexible infrastructure-independent transport mechanisms such as MobileIP [Perkins, 2002] can be used to improve reliability over multiple independent and redundant communication links.

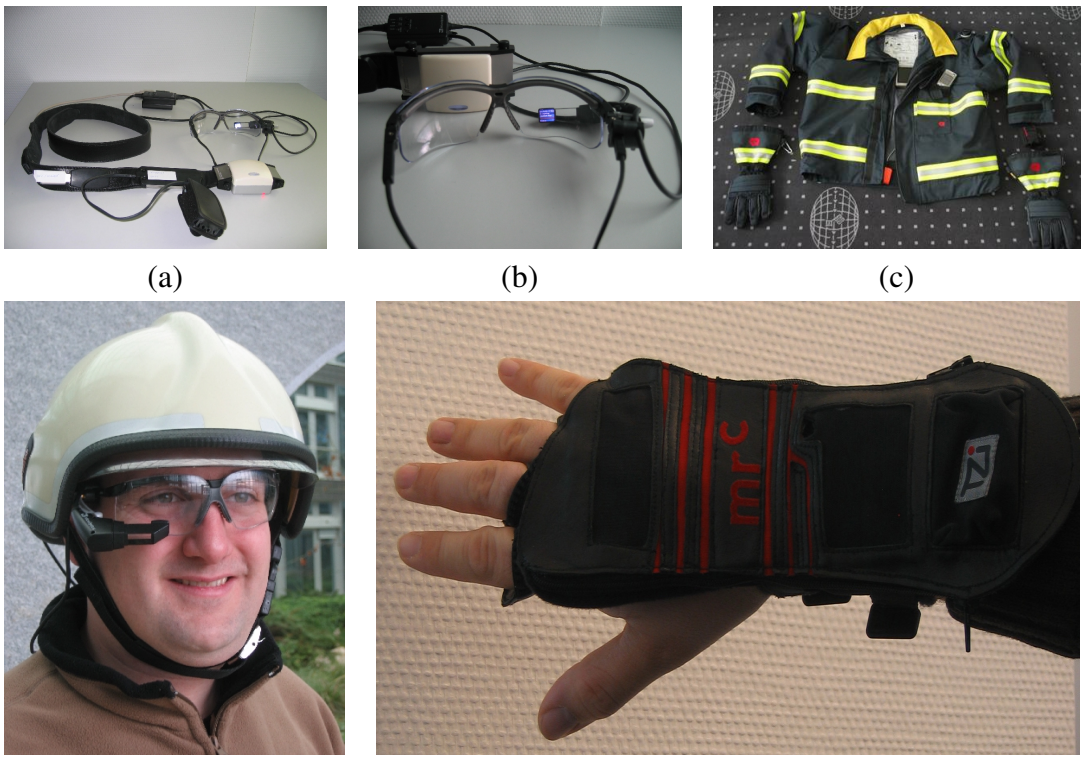


Figure 8.2: All parts of the wearable emergency-response system: (a) The *qbic belt-worn computer*, (b) the Head-Mounted Display (HMD), (c) a firefighter jacket equipped with an IMU sensor for PDR, (d) the HMD worn by a test person, (e) a wireless RFID reader device, integrated into a glove.

disaster management. We utilized the code base of the *ResQ Freiburg* team [Kleiner et al., 2005a] for evaluating MAS techniques in the context of urban search and rescue.

In the following the integration of data captured by the interface, which has been discussed in Section 8.2, into the RoboCup Rescue simulation kernel is described. Moreover, the optimization of search and rescue tasks based on this data is discussed along with methods for the optimization of rescue sequences.

8.3.1 Real-world data integration

Generally, it is assumed that the wearable device of each human responder continuously reports the latest recorded positions in form of *track* messages to the command center, if communication is possible. Additionally, the rescue team might provide information for specific locations, as for example, indicating the successful exploration of an area, or the detection of a victim by triggering the according *waypoint* message at the current

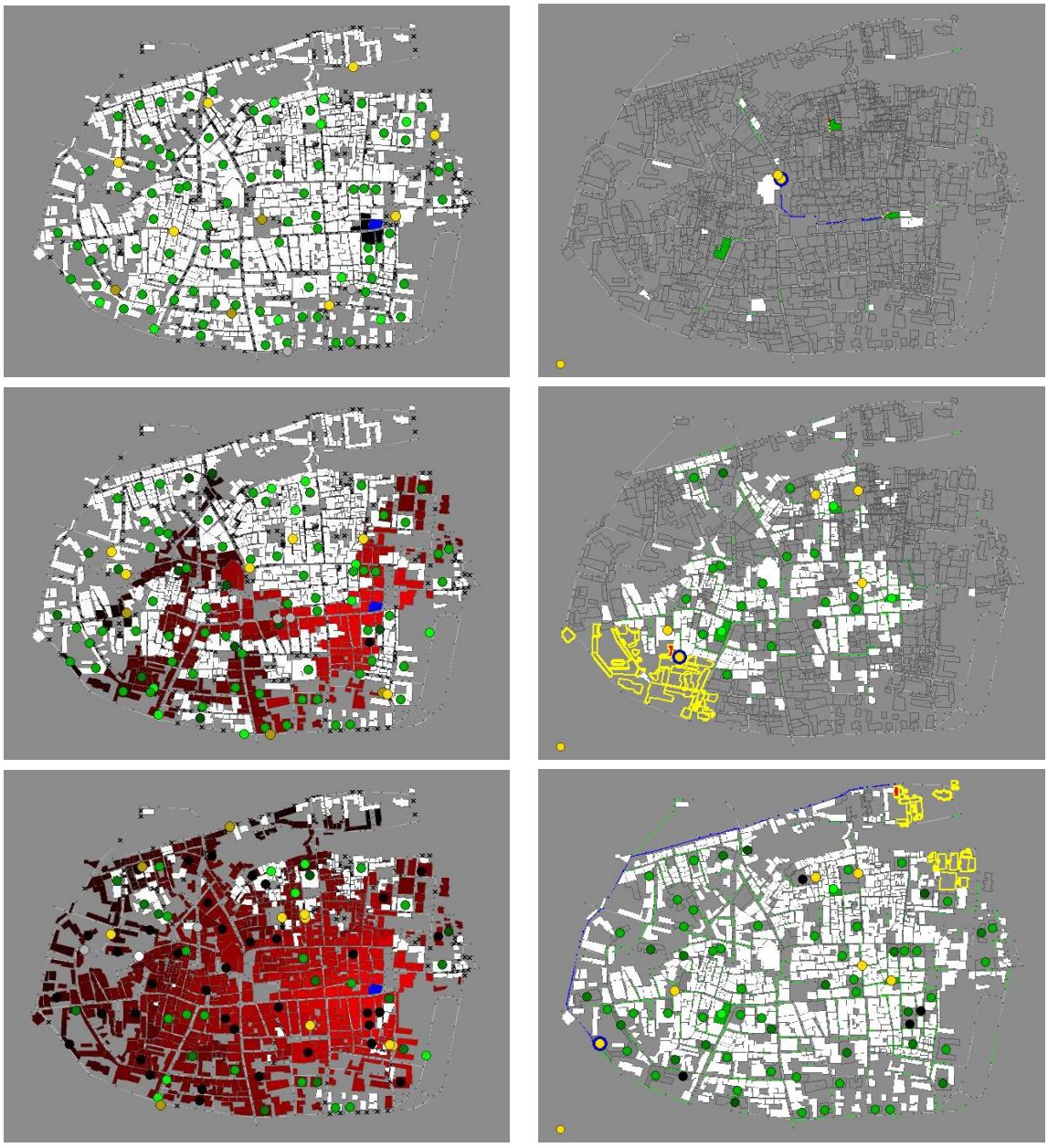
location.

Based on an initial road map, the MAS continuously integrates all collected track data, and information on the status of roads, e.g. whether they are passable at high costs or even impassable, as reported by the responders in the field. More specific, each connection between two locations is augmented with a weight that is computed from the distance, the amount of blockage, the number of crossings, and the number of other agents known to be located on the same road. The emerging graph structure is used as an input for the Dijkstra algorithm [Dijkstra, 1959] computing a connectivity matrix that stores for each human responder location the distance costs to any other location on the map. If places are unreachable, costs are infinite. In the worst case, the computation is in $O(m + nlog(n))$ for a single responder, where n is the total number of locations and m is the number of connections between them. The knowledge on the reachability between locations allows the system to recommend "safe" routes to rescue workers, as well as to optimize their target assignments by taking travel costs into account. Note that the connectivity matrix has to be re-computed if new information on the connectivity between two locations is available, e.g. either if the blockage of a connection has been observed, or if a formerly blocked connection has been cleared.

In case the procedure is applied with simulated agents, movement interactions are simulated by the *Traffic simulator*, and observations are according to a simulated disaster dynamic carried out by the *Collapse simulator* and *Fire simulator* [Nüssle et al., 2004], which are an integral part of the rescue simulation system. The sequence in Figure 8.3 (a) shows the continuous update of the connectivity matrix for the blue region in the simulated City of Foligno.

8.3.2 Coordinated victim search

Victim search can only be coordinated efficiently if rescue teams synchronize information on explored areas. Therefore, the protocol described in Section 8.2 implements messages for reporting the clearance of explored areas, as well as the detection of a victim. The command center utilizes this information for efficiently assigning rescue teams to unexplored locations that are reachable. The sequence in Figure 8.3 (b) shows the subsequently increasing knowledge on the status of the exploration, being transmitted to the control center by agents in the field. Regions that are marked with a yellow border indicate exploration targets assigned by the command center according to a global exploration strategy, which is described in the following.



(a) (b)

Figure 8.3: Online data integration of information reported by agents exploring the simulated City of Foligno: (a) The connectivity between the blue region and other locations increases over time due to the clearance of blocked roads, showing unreachable locations (white color), and reachable locations (red color). The brighter the red color, the better the location is reachable. (b) The subsequently increasing knowledge on the status of the exploration, given the information communicated by agents in the field, showing passable roads (green color), explored regions with victim detection (green color) and without (white color). Regions marked with a yellow border are exploration targets recommended by the command center agent.

District exploration.

District exploration is a multi-agent behavior for coordinated victim search. The behavior guarantees that at any time each agent has an assignment to an unexplored district on the map. One problem arising when defining districts after a fixed pattern, e.g. by overlaying equally sized rectangles, is that districts might be differently sized due to an inhomogeneous distribution of roads and buildings. Furthermore, places within the same district might not be interconnected, making it unfeasible for agents to entirely explore the district they were assigned to. Districts are more effective if they are build according to the building distribution, and according to the connectivity of locations on the map, e.g. they should consist of locations with a high degree of inter-connectivity. The connectivity value of two locations depends on the number of alternative paths that can be found between them, the number of lanes, the degree of blockage of each connection, and the degree of uncertainty on the state of the connection.

We utilized *agglomerative* [Bock, 1974] and *KD-tree* [Bentley, 1975] clustering in order to calculate a near optimal separation of the map into districts, and to compute a value expressing the approximative meta-connectivity between them. These methods calculate from a given connectivity graph $G = \langle V, E \rangle$, where V denotes the set of locations, and E the set of connections between them, a hierarchical clustering. According to a distance metric, each edge $e \in E$ has a weight attached, which is considered during the clustering procedure. KD-tree clustering is a *Divide and Conquer* method, which can be efficiently performed in $O(n \log n)$, whereas agglomerative clustering, a *Dynamic Programming* method, can be carried out in $O\left(n^2\right)$[‡] [Eppstein, 2000]. However, in KD-tree clustering the distance metric has to be a L_n-norm, e.g. the Euclidean or Manhattan distance, whereas in agglomerative clustering, the distance metric can be any arbitrary function.

Agglomerative clustering methods are distinguished by the way how the distance between two clusters is computed. For example, in *Single Linkage* the minimum distance between objects o_i and o_j from cluster C_i and C_j is taken as distance between the clusters:

$$d\left(C_i, C_j\right) = \min_{s,t} \left\{ d(s,t) \mid o_s \in C_i, o_t \in C_j \right\}, \tag{8.1}$$

in *Complete Linkage* the maximum is taken by:

$$d\left(C_i, C_j\right) = \max_{s,t} \left\{ d(s,t) \mid o_s \in C_i, o_t \in C_j \right\}, \tag{8.2}$$

and in *average linkage* the average of all connections between objects from cluster C_i

[‡]Note that that this bound only applies for *Single Linkage*, which will be described in the following.

and C_j by:

$$d\left(C_i, C_j\right) = \frac{\sum_{s \in C_i} \sum_{t \in C_j} d\left(s, t\right)}{n_i n_j} \tag{8.3}$$

We utilized *Single Linkage* in order to divide city maps into districts for exploration assignments. One significant advantage of agglomerative clustering is that the state of a connection, e.g. partial or full blockage, length and width of the road etc., can be considered directly for building the hierarchy. Figure 8.4 compares KD-tree clustering with agglomerative clustering on the map of Bremen. The figure shows that blocked roads (black crosses) are considered during agglomerative clustering, whereas KD-tree clustering, based on the Euclidean distance metric, separates the city map in nearly equally sized districts containing places that are unreachable.

(a) (b)

Figure 8.4: Comparison between (a) KD-tree clustering and (b) agglomerative clustering on the map of Bremen. Blocked roads (black crosses) are considered during agglomerative clustering, while KD-tree clustering, based on the Euclidean distance metric, separates the city map into nearly equally sized districts containing places that are unreachable from each other.

Since agglomerative clustering represents the true connectivity of places, it is a better choice for district exploration. The hierarchical clustering, represented by a binary tree, provides at each level a partitioning of the map into n districts, reflecting the reachability of locations, e.g. locations which are highly connected are likely to be found within the same cluster. Given the cluster tree, one can select a tree level according to the size of the search team, i.e. such that $n \geq m$, where m denotes the number of agents and n the size of the clustering. Due to the fact that blockades on the map and hence the map's connectivity is unknown to the agents in advance, the clustering has to be revised each time new information arrives.

Active Exploration.

Exploration within districts can be accelerated significantly if exploration targets are selected with respects to the expected utility of observed places, e.g. computed from the agent's sensor model and traveling costs. Active exploration is a method that considers exploration utility U and traveling cost C when selecting target locations [Burgard et al., 2005]. In a search and rescue scenario, the expected utility for visiting places can be computed by joining observations from agents, such as audio noise and CO_2 concentration, into a single occupancy grid (see Section 5.2), while considering the senor model, e.g. maximal detection range and field of view, of each sensor.

We utilized a simplified occupancy grid map for incorporating visual and auditory sensor observations from multiple agents. Note that also *negative* information, e.g. no evidence detection at a certain location, as well as *positive* information can be incorporated for increasing the efficiency of the search. Finally, from the set of locations L_D that are within the agent's district, a target location l_t is decided based on the trade-off between utility $U(l)$ and travel cost $T(l)$:

$$l_t = \underset{l \in L_D}{argmax}\, U(l) - \alpha * T(l) \tag{8.4}$$

where α is a constant regulating the trade-off between the estimated travel costs and the exploration utility, which has to be determined experimentally. The estimated travel costs $T(l)$ are taken from the connectivity matrix, computed by the Dijkstra algorithm.

8.3.3 Rescue sequence optimization

Since time is a critical issue during real disaster response, the survival rate of human victims depends on the optimal schedule of a rescue mission. For example, after a car accident on a highway, it is common practice for ambulance teams to decide the sequence of rescuing victims according to the degree of their injuries and accessibility. During a large-scale disaster, such as an earthquake, the efficient scheduling of rescue teams is even more important since there are probably many victims and often an insufficient number of rescue teams. Furthermore, the time needed for rescuing victims might significantly vary depending on the collapsed building structures trapping them. Therefore, it is important that rescue teams inform the station about the state of victims, e.g. conscious or unconscious, the degree of injuries, and their particular situation, e.g. surface, trapped, void, or entombed. This information facilitates optimizing the schedule of rescue missions by the station for maximizing the overall number of survivors.

In RoboCup Rescue, victims are reported with the three variables *damage*, *health* and *buridness*, expressing an individual's damage due to fire or debris, the current health that continuously decreases depending on damage, and the difficulty of rescuing the victim. From this variables, one can estimate an upper bound for the time necessary to rescue

a victim, and a lower bound for the time the victim will survive in the current state. We utilized Adaptive Boosting (Ada Boost) [Freund and Schapire, 1996] for learning the regression of a victim's survival time, and linear regression for predicting the time needed for performing the rescue task, given the mentioned parameters. Furthermore, the travel times needed for reaching victim locations have been computed by averaging over sampled travel times needed by individual agents.

It can be assumed that during a real disaster response scenario expert knowledge can be acquired for providing rough estimates on the resources needed for victim rescue, i.e. first responders might be able to estimate whether the removal of debris needs minutes or hours. Furthermore, the continuous sampling of first responder trajectories, e.g. acquired by PDR methods or GPS, can be utilized for automatically estimating travel times for different means of transport, e.g. by car or by feet, between specific locations [Liao et al., 2007].

Given estimates on the survival time of victims, and the time needed for rescuing them, it is possible to compute the total number of survivors for any arbitrary rescue sequence. More specific, one can compute for each rescue sequence $S = \langle t_1, t_2, ..., t_n \rangle$ of n rescue targets a utility $U(S)$ that is equal to the number of survivors. Consequently, the process of determining the optimal rescue sequence that maximizes $U(S)$ can be considered as an optimization problem. However, exhaustive search over all possible $n!$ sequences is already for a little number of targets intractable. The *Hungarian Method* [Kuhn, 1955] is a polynomial algorithm for solving scheduling tasks, and finds the optimal task assignment in $O\left(mn^2\right)$, where n is the number of workers, and m the number of tasks. However, the method requires that the time needed until a task is finished does not influence the overall outcome. This is clearly not the case for rescue tasks, since victims will not survive if they are rescued too late. Alternatively, *Greedy* heuristics can be deployed, for example, to schedule rescue missions first that require the least amount of time to be accomplished, or to schedule urgent rescue missions first, e.g. those with the least amount of victim survival time. However, it seems to be obvious that better solutions might be found by trading-off both of them.

We utilized a Genetic Algorithm (GA) [Holland, 1975] for the online optimization of rescue sequences. Rescue sequences are represented by integer strings, e.g. $S = \langle 2, 5, 1, 3, 4 \rangle$, encoding each individual solution in a genetic pool. The genetic pool is initialized with heuristic solutions, for example, sequences that *greedily* prefer targets that can be rescued within a short time or urgent targets that have only little chance of survival. The fitness function of solutions is given by the sequence utility $U(S)$, which itself is computed by simulating the expected outcome of the sequence. In order to guarantee that solutions in the genetic pool are at least as good as the heuristic solutions, the so-called *elitism* mechanism, which forces the permanent existence of the best found solution in the pool, was used. Furthermore, we utilized a simple one-point-crossover strategy, a uniform mutation probability of $p \approx 1/n$, and a population size of 10. One important property of the proposed method is that it can be executed as an *anytime*

algorithm. At any time the method provides a solution that is at least as good as the greedy solution, but potentially better with increasing computation time.

8.4 Experiments

8.4.1 Data integration from wearable devices

The integration of real-world data into the RoboCup rescue system was evaluated by successively processing data transmitted by a test person. The test person was equipped with the wearable interface described in Section 8.2, while walking several trajectories in an urban area in the City of Bremen (see Figure 8.5). During this experiment, the mobile device continuously transmitted the trajectory via GPRS to a central server. Furthermore, each time the test person had visual contact with a simulated victim [§], it reported via the described GPX protocol a *victim found* waypoint message to the server.

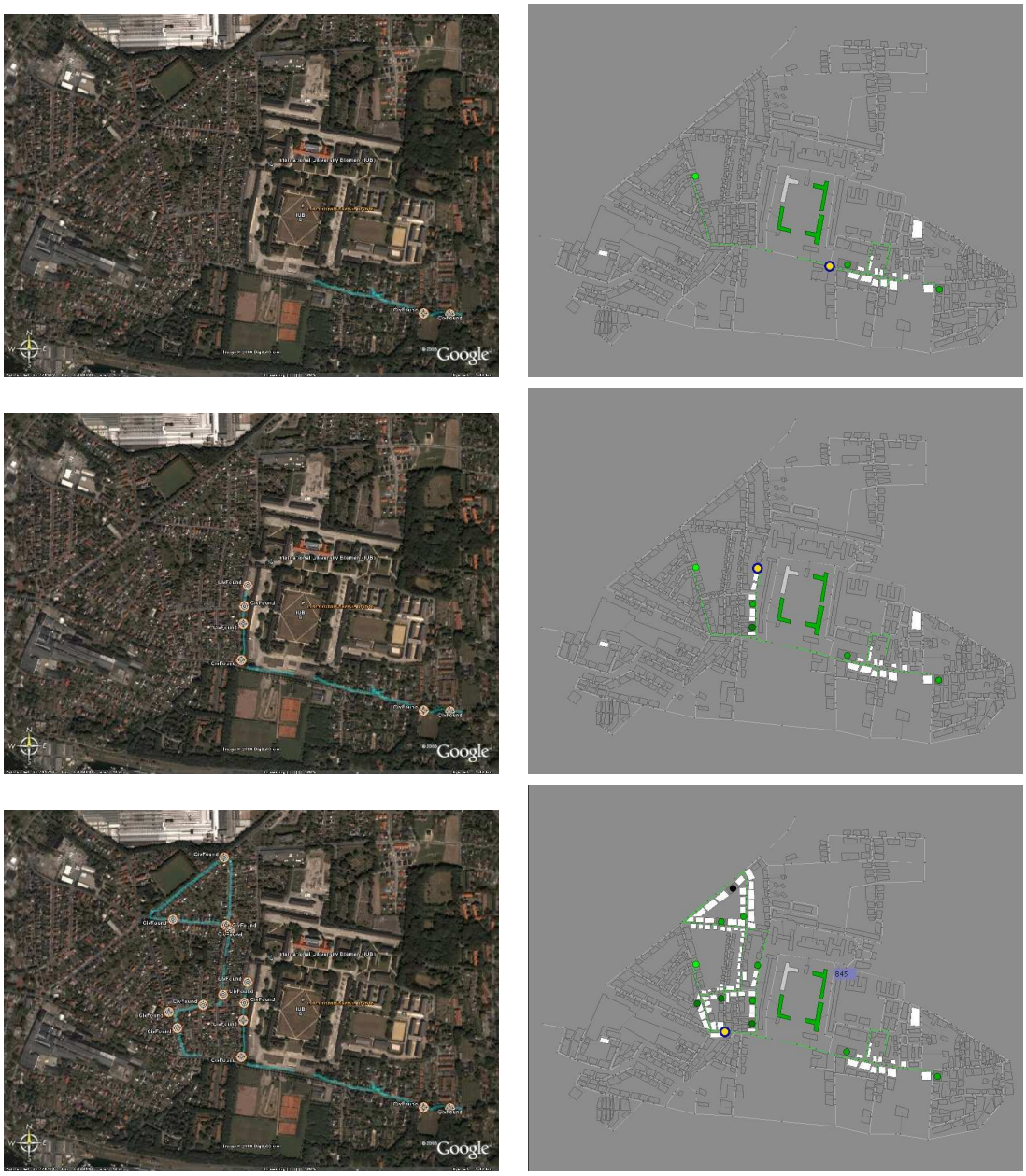
In order to integrate the data into the rescue system, the received data, encoded by the extended GPX protocol that represents location by latitude and longitude, was converted into a grid-based representation. We utilized the Universal Transverse Mercator (UTM) [Snyder, 1987] projection system, which provides a zone for any location on the surface of the Earth, where coordinates are described relatively to this zone (see Section 2.2.3). By calibrating maps from the rescue system to the point of origin of the UTM coordinate system, locations from the GPS device can directly be mapped. In order to cope with erroneous data, we decided to simply ignore outliers, i.e. locations far from the track, that were detected based on assumptions made on the test person's maximal velocity. Figure 8.5 (b) shows the successive integration of the received data into the rescue system and Figure 8.5 (a) displays the same data plotted by *GoogleEarth*. Note that GPX data can be displayed directly by GoogleEarth without requiring any pre-processing.

8.4.2 Coordinated victim search

The efficiency of the proposed method for exploration and victim search was evaluated by analyzing log files recorded during the RoboCup Rescue simulation competition in 2004. During this competition, 20 international teams competed on different maps simulated by the rescue system. The methods described in this chapter were an integral part of the *ResQ Freiburg* team which achieved the first place of the competition. Other teams applied, for example, Reinforcement Learning (DAMAS-Rescue and RoboAkut), and MAS group forming methods (NITRescue04).

During emergency response, the newest information is the most valuable. For example, the earlier victim locations are known, the better their rescue can be scheduled.

[§]Note that arbitrary locations in Bremen were selected as victim locations and were visually marked.



(a) (b)

Figure 8.5: Successive integration of data reported by a test person equipped with a wearable device. (a) The real trajectory and observations of victims plotted with GoogleEarth (victims are labeled with "civFound"). (b) The same data integrated into the rescue system (green roads are known to be passable, white buildings are known as explored, and green dots indicate observed victims).

Figure 8.6 shows the number of civilians found during each cycle on the *RandomMap* during the final and semi-final, respectively. The results confirm the efficiency of the

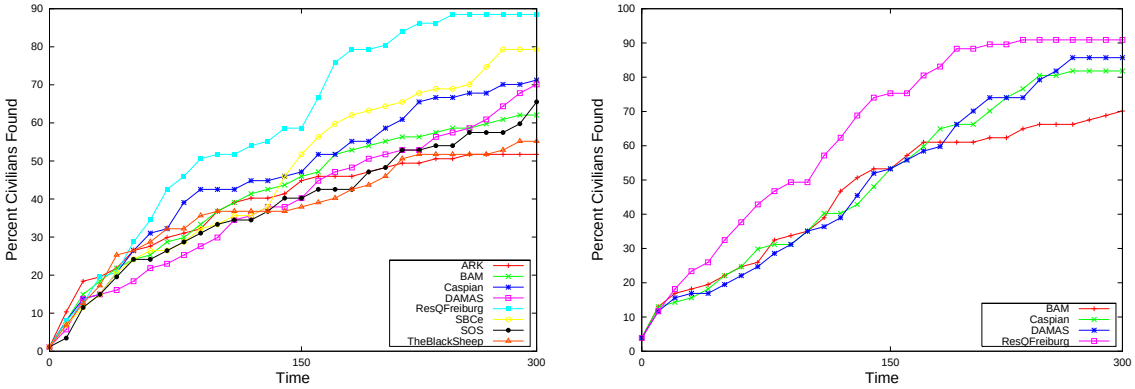


Figure 8.6: The number of civilians found by exploration on a randomly generated map during the (a) final and (b) semi-final.

exploration strategy of *ResQ Freiburg*. At any given time our agents knew about more civilians than the agents of any other team.

Table 8.1 shows the percentage of buildings that were searched by the agents [¶], and Table 8.2[‖], shows the percentage of found civilians during each run. Results are shown from all rounds of all teams that passed the preliminaries during the competition. All values are according to the state of the world at the last round, e.g. the percentage of explored buildings at round 300. Bold numbers denote the best results that have been achieved during the respective rounds. The results show that *Caspian* explored most of the buildings. However, *ResQ Freiburg* was able to find more victims, although they explored less buildings, which indicates a higher degree of exploration efficiency.

8.4.3 Victim rescue sequence optimization

In order to evaluate the performance of the online genetic algorithm, we executed multiple simulation runs, once with Greedy optimization and once with GA-based optimization for scheduling rescue missions. Figure 8.7 depicts the difference between a greedy rescue target selection, i.e. to prefer targets that can be rescued fast, and the GA-based selection optimization. It can be seen that GA-based optimization clearly increases the number of rescued civilians, even the time for making decisions is limited. Within each

[¶]Note that full communication of visited locations as well as exploitation of a sensor model was assumed.

[‖]Note that civilians are considered as being found if one of the agents was within their visual range.

	ResQ	Damas	Caspian	BAM	SOS	SBC	ARK	B.Sheep
Final-VC	83,48	83,24	**87,02**	67,27	N/A	N/A	N/A	N/A
Final-Random	69,62	72,62	**78,13**	49,92	N/A	N/A	N/A	N/A
Final-Kobe	89,19	92,97	89,73	**94,19**	N/A	N/A	N/A	N/A
Final-Foligno	84,15	85,25	**86,73**	74,29	N/A	N/A	N/A	N/A
Semi-VC	69,39	72,86	**77,42**	45,08	52,01	52,87	47,92	59,72
Semi-Random	**78,91**	68,73	71,91	54,36	59,36	70,27	46,18	46,18
Semi-Kobe	85,41	96,22	92,97	95,54	66,62	97,30	**99,46**	91,89
Semi-Foligno	74,75	89,12	84,98	62,49	65,35	**92,53**	79,08	20,74
Round2-Kobe	87,16	90,68	95,00	91,76	80,54	94,19	**99,46**	92,43
Round2-Random	81,18	80,94	88,61	84,53	60,67	**94,24**	82,61	87,89
Round2-VC	83,40	70,18	84,58	40,44	67,74	87,88	N/A	**89,54**
Round1-Kobe	87,43	90,27	94,05	96,08	96,62	97,70	**97,84**	80,95
Round1-VC	85,37	90,48	95,28	94,26	N/A	97,72	**100,00**	91,35
Round1-Foligno	83,78	**90,05**	**90,05**	60,00	54,65	88,57	67,37	13,00
Number of Wins:	1	1	**4**	1	0	2	4	1
AVG %:	81,66	83,83	**86,89**	72,16	67,06	87,33	79,99	67,37
STD %:	5,82	9,98	**7,87**	22,21	13,87	14,59	21,84	30,80

Table 8.1: Percentage of explored buildings.

	ResQ	Damas	Caspian	BAM	SOS	SBC	ARK	B.Sheep
Final-VC	97,22	94,44	**100,00**	81,94	N/A	N/A	N/A	N/A
Final-Random	**90,91**	85,71	81,82	70,13	N/A	N/A	N/A	N/A
Final-Kobe	**98,77**	97,53	95,06	**98,77**	N/A	N/A	N/A	N/A
Final-Foligno	**96,67**	**96,67**	**96,67**	72,22	N/A	N/A	N/A	N/A
Semi-VC	77,92	77,92	**85,71**	45,45	53,25	53,25	50,65	63,64
Semi-Random	**88,51**	73,56	72,41	63,22	67,82	80,46	52,87	55,17
Semi-Kobe	**100,00**	**100,00**	**100,00**	98,61	79,17	**100,00**	**100,00**	97,22
Semi-Foligno	90,12	95,06	86,42	81,48	83,95	**97,53**	85,19	30,86
Round2-Kobe	98,89	98,89	97,78	95,56	91,11	**100,00**	**100,00**	98,89
Round2-Random	**98,89**	95,56	**98,89**	81,11	70,00	96,67	85,56	94,44
Round2-VC	**92,22**	78,89	90,00	45,56	72,22	88,89	N/A	87,78
Round1-Kobe	94,29	**100,00**	**100,00**	98,57	**100,00**	**100,00**	94,29	78,57
Round1-VC	**100,00**	**100,00**	**100,00**	97,14	N/A	**100,00**	**100,00**	98,57
Round1-Foligno	**100,00**	97,14	94,29	77,14	74,29	92,86	77,14	14,29
Number of Wins:	**9**	4	7	1	1	5	3	0
AVG %:	**94,60**	92,24	92,79	79,06	76,87	90,97	82,85	71,94
STD %:	**7,17**	10,53	9,03	20,75	13,73	14,69	19,35	30,25

Table 8.2: Percentage of found civilians.

minute, the GA was able to compute approximately $300,000$ solutions on a *1.0GHz Pentium4* computer.

Finally, Table 8.3 shows the number of civilians saved by each team: *ResQ Freiburg* saved more than 620 civilians during all rounds, which are 35 more than the second best and 59 more than the third best in the competition.

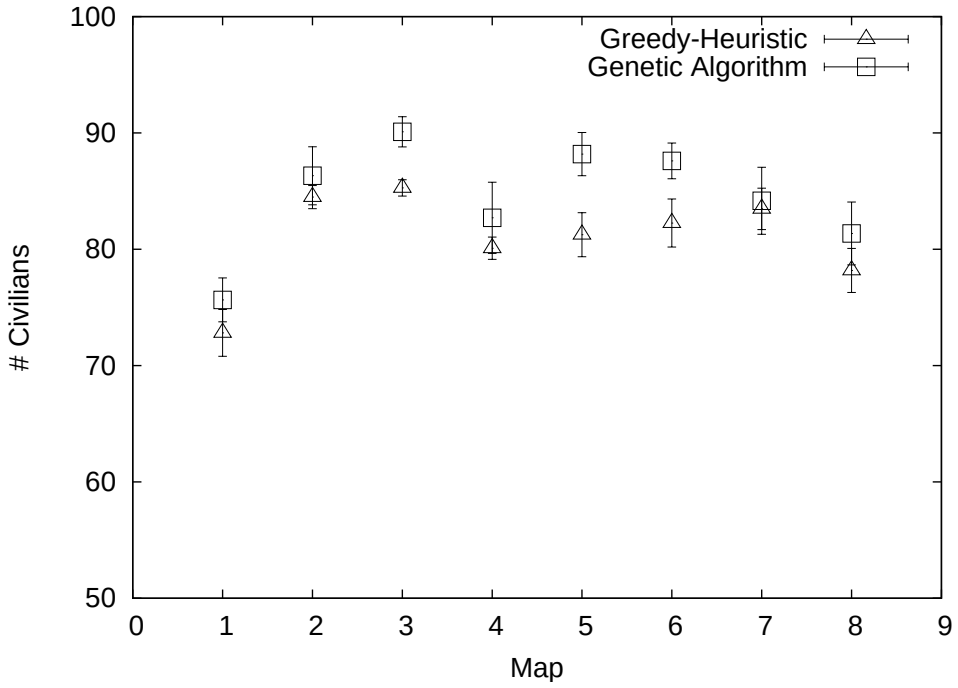


Figure 8.7: The number of civilian survivors after applying Greedy rescue sequence optimization and GA-based sequence optimization in different simulated cities.

	ResQ	Damas	Caspian	BAM	SOS	SBC	ARK	B.Sheep
Final-VC	42	43	**52**	34	N/A	N/A	N/A	N/A
Final-Random	**32**	25	29	16	N/A	N/A	N/A	N/A
Final-Kobe	**46**	45	**46**	30	N/A	N/A	N/A	N/A
Final-Foligno	**66**	54	50	29	N/A	N/A	N/A	N/A
Semi-VC	**18**	15	17	12	11	12	12	14
Semi-Random	22	**26**	16	14	20	14	15	15
Semi-Kobe	**57**	47	54	52	20	39	34	44
Semi-Foligno	37	**46**	44	43	42	28	29	24
Round2-Kobe	**57**	37	43	50	43	35	28	43
Round2-Random	**52**	48	39	45	47	44	50	37
Round2-VC	31	33	32	24	37	**51**	N/A	34
Round1-Kobe	45	**51**	47	43	47	31	25	34
Round1-VC	**62**	**62**	55	57	N/A	51	54	44
Round1-Foligno	**53**	**53**	37	33	37	41	30	23
#Wins:	**9**	5	2	0	0	1	0	0
Σ TOTAL:	**620**	585	561	482	304	346	277	312
Σ SEMI+PREM	**434**	418	384	373	304	346	277	312

Table 8.3: Number of saved civilians.

8.5 Related Work

Nourbakhsh and colleagues utilized the MAS *Retsina* for mixing real-world and simulation-based testing in the context of Urban Search and Rescue [Nourbakhsh et al., 2005]. Wickler et al. introduced *To Do* lists within the <*INCA*> constraint model to coordinate activities in an emergency response scenario [Wickler et al., 2006]. They utilize a Hierarchical Task Network (HTN) planner that can be used to synthesize courses of action automatically. Their system has also been used by Siebra and Tate to support coordination of rescue agents in the RoboCup Rescue simulation [Siebra and Tate, 2005].

Oh et al. applied agent technologies in post-disaster planning contexts by an agent-based decision support system that can identify good candidate locations for temporary housing [Oh et al., 2006]. Schurr and colleagues [Schurr et al., 2005] introduced the *DEFACTO* system, which enables agent-human cooperation and was evaluated in the firefighting domain with the RoboCup Rescue simulation package. U et al. proposed a generalized agent model for the RoboCup Rescue simulation system [U and Reed., 2006]. Their extension facilitates agents that have multiple roles, and can change their capabilities during a simulation, enabling a richer mix of agent behaviors.

Takahashi discusses the practical usage of disaster management systems for local governments by an evaluation of RoboCup Rescue simulation data, with the intention of guiding future research topics towards the development of practical disaster simulation systems [Takahashi, 2006]. Noda and Meguro proposed the standard protocol *MISP* and a simple database system *DaRuMa* for the purpose of collecting and sharing disaster information about damaged areas for supporting decision-making in rescue processes [Noda and Meguro, 2006]. Kuwata et al. introduced a simulation framework, based on RoboCup Rescue simulation, named *RoboCupRescue Human-In-The-Loop Agent Simulation (RCR-HITLAS)*, in which humans can act as agents with various roles. [Kuwata et al., 2006]. The purpose of the system is to measure the performance of user interfaces and decision support systems used by humans that humans, as well as their skills.

8.6 Conclusion

We introduced the preliminary design of a wearable computing device in conjunction with a multi-agent based disaster simulator, which both can be utilized for optimizing USAR missions. It has been generally demonstrated that the system is capable of integrating trajectories and observations captured during real-world experiments into the RoboCup Rescue simulation platform. As shown by the results from simulation runs, the integrated data can serve as a basis for coordinating exploration, e.g. by directing teams to unexplored regions, as well for scheduling rescue missions. The utilized MAS approaches were evaluated during various simulation runs, and have also been com-

pared with strategies developed by other teams during the RoboCup Rescue simulation competition in 2004.

The proposed interface can be extended further by additional sensors, such as CO_2 detectors, temperature sensors, and cameras, for facilitating the transmission of images and data that document the degree of damages or the state of victims at certain locations. The overall goal in this context is to design as system that integrates all observations from both human responders and robot rescue teams, while reducing the burden, i.e. the cognitive load, of the user. The strategic overview of the scenario allows incident commanders to efficiently combine the capabilities of individual team members by developing multi-robot and multi-human plans, given the available resources.

In Chapter 4 the automatic fusion of maps generated by robot and humans has been demonstrated. In future research, we intend to extent this approach by integrating features extracted from local observations taken by wearable devices or sensors mounted on robots. Furthermore, we will consider the evaluation of the coordination mechanism in real scenarios, e.g. to guide first responders via the wearable computing device during an exercise by automatically generated plans. It would be particularly interesting to compare this approach with conventional methods as they are used in emergency response nowadays, i.e. to determine the impact of agent technology in time critical missions.

Future work in the Rescue simulation league aims at opening the system to resources available on the Internet, such as map data from *Google Maps*, and any other freely available location-specific data. This direction has the primary goal to close the loop between disaster reality and disaster simulation.

9 Summary And Future Work

To transfer robot technology from laboratory experiments into the real world for solving socially significant problems requires research and development being undertaken close to reality. An exchange of practical knowledge with experts from the field, such as emergency response personnel, is indispensable. Only their experience can facilitate the development of deployable autonomous robot platforms, as well as their integration into human operation. Performance metrics, such as benchmark arenas, are one promising approach to this goal.

Current benchmarking scenarios do clearly not yet capture the whole magnitude of problems that robots encounter after a real disaster. In this scenario, mapping methods are required that also function when robots operate in confined spaces within collapsed structures, climb-up walls, or fly autonomously over the terrain. The idea behind benchmarking autonomous robot systems in USAR (Urban Search And Rescue) scenarios is to continuously increase the level of difficulty year by year in order to promote stepwise research in this field. Up to now, methods from autonomous robotics for USAR are just at their advent and robots that have been deployed after a real disaster were mainly tele-operated by human beings.

Lessons learned from early stage performance metrics already showed that assumptions holding under mild conditions, such as office-like environments, do not necessarily hold under harsh conditions. For example, conventional SLAM methods assume a constant growth of the positioning error during pose tracking, e.g. the longer the track the larger the variance. Unstructured environments have the property that conditions can arbitrarily change, e.g. the underground might be smooth floor, slippery, or even rough terrain, requiring robots to be aware of their context. Data association requires robust visual features in order to function reliably, which are not constantly found under those conditions. Furthermore, robotic methods must not interfere with task-specific goals of operations in the field. For example, conventional mapping algorithms require the repeated visiting of places in order to refine the map by loop closure, which turns out to be a suboptimal behavior during victim search and exploration.

Finally, the deployment of robotics technology is only helpful if it complements human task execution. Here the ultimate goal to is to optimally coordinate human and robot rescue teams, i.e. to distribute tasks among them according to their individual skills. Therefore, methods have to address data exchange issues directly, such as the exchange of maps and observations, while coping with communication perturbations, or even temporarily loss of communication.

In the light of these considerations three core problems were addressed in this thesis, which are localization, mapping, and exploration with a strong focus on human integration and teamwork of both robots and humans.

9.1 Summary

In Chapter 2 hardware for Rescue Robotics was introduced and design criteria for robot platforms built for the deployment in emergency response scenarios were discussed. According to these criteria, two robot platforms were contributed that have been built for the development and evaluation of algorithms presented in this thesis. Furthermore, different sensor types for localization, mapping, and victim search were discussed.

The problem of tracking humans and robots was subject of Chapter 3. A novel method for slippage sensitive pose tracking on wheeled robot platforms and a novel method for visual odometry on tracked platforms were contributed. While these methods were particularly designed for two specific application scenarios, which are the rapid mapping of a large-scale environment by wheeled robots, and the mapping of rough terrain by tracked robots, they can basically be considered as independent modules that can be used for different tasks. They were intensively tested under harsh environmental conditions, and evaluated within USAR benchmark scenarios, such as the RoboCup Rescue autonomy competition. Furthermore, an existing method for tracking the pose of humans was evaluated and discussed. Results showed that human pose tracking is feasible over larger distances, accompanied with increasing positioning errors.

In Chapter 4 RFID-SLAM, a novel method for SLAM, was contributed. This method allows the efficient generation of globally consistent maps, even if the density of landmarks is comparably low. For example, the method corrected an outdoor large-scale map within a few seconds from odometry data and RFID perceptions only. This has been partially achieved due to reliable pose tracking based on slippage sensitive odometry, but also due to the data association solved by RFIDs. Solving data association by RFIDs allows to speed-up the route graph corrections by decomposing the problem into optimization and interpolation. Furthermore, we demonstrated the joint correction of trajectories from human and robot teams. The joint correction of robot and human trajectories is of major importance since it is very likely that robots and humans explore different areas during emergency response. For example, robots will be deployed in hazardous places or places lacking of air, which humans cannot access. We showed that RFID-SLAM allows to improve maps by loop closure without requiring humans and robots to perform loops while executing their primary task. Due to the joining of routes via RFID connection points, loops automatically emerge. This is a necessary requirement if applying SLAM in USAR situations. In such situations, emergency responders have clear goals that have to be accomplished within a short amount of time, for exam-

ple, they have to explore the environment for victims, and therefore intentionally try to avoid to visit places repeatedly.

Finally, a decentralized version of RFID-SLAM that utilizes the memory capacity of RFIDs was contributed. The results show that sharing information between single agents allows to optimize globally their individual paths, even if they are not able to communicate directly. This has the advantage, particularly during disaster response, that this method can be applied if communication is disturbed by building debris and radiation. If communication is temporarily available, the process can be accelerated by exchanging data between the agents directly.

In the disaster response scenario RFID-SLAM offers other practical advantages. For example, humans can communicate with RFIDs via a PDA and leave behind information related to the search, such as discovered victims or hazardous places. The idea of labeling locations with information that is important to the rescue task has already been applied in the past. During the disaster relief in New Orleans in 2005, rescue task forces marked buildings with information concerning, for example, hazardous materials or victims inside the buildings [FEMA, 2003]. The RFID-based marking of locations is a straight forward extension of this concept.

In Chapter 5 a method for the building of elevation maps from readings of a tilted LRF was contributed. In contrast to former work, the proposed method integrates all six degrees-of-freedoms of the robot in real-time, allowing the mapping of rough terrain while the robot navigates over obstacles such as ramps. The quantitative evaluation of conducted indoor and outdoor experiments, partially in testing arenas proposed by NIST for USAR, showed that the proposed method can be deployed in real-time while leading to a robust mapping of the environment. The result was supported by the visual odometry method for pose tracking, which significantly improved the accuracy of scan matching. As we showed in another work, the generated elevation maps can be utilized for structure classification and the planning of skill execution [Dornhege and Kleiner, 2007b].

In Chapter 6 a novel method for coordinated exploration of large robot teams was contributed. The approach, which is based on RFID technology for indirect communication, is composed of two parts. First, distributed local search, with the notable properties of not requiring direct communication and scaling with the number of agents. Second, global task assignment and multi-robot path planning for monitoring the local exploration and restarting it at better locations. A novel graph structure based on RFIDs was introduced, which allows a significantly smaller representation of the environment compared to grid based approaches. The experimental results from RoboCup showed that RFID-based coordination scales-up with large robot teams exploring large environments. Moreover, the global task assignment based on genetic sequence optimization was evaluated. We showed that this method allows to improve the performance of the local approach for robot teams consisting of up to 20 robots. Due to a unified exchange of data, e.g. as has been shown with RFID-SLAM, the method can also be extended for

integrating humans into the search.

The problem of real-time detection of victims in disaster areas was discussed in Chapter 7. In order to scale-up the detection rate of weak classifiers, a system for creating optimized MRF models was contributed, which computes offline relevant feature combinations with respect to the data. The optimized classifier was five times faster than the model with all feature combinations activated, while gaining optimal performance in terms of the complexity trade-off, and near-optimal performance in terms of accuracy. We compared the optimized model with a SVM classifier and showed that the false-positive rate was significantly reduced, which is an important aspect when considering victim detection in the context of emergency response. The proposed approach can be considered as a general framework for combining human evidence detected by sensors. For example, it can easily be extended for incorporating other types of human evidence, such as audio noise, e.g. tapping and CO_2 emission.

In the last chapter, Chapter 8, the preliminary design of a wearable computing device in conjunction with a multi-agent based disaster simulator for optimizing USAR mission scheduling, was contributed. It was generally demonstrated that the system is capable of integrating trajectories and observations captured during real-world experiments into the RoboCup Rescue simulation platform. As shown by the results from simulation runs, the integrated data can serve as a basis for coordinating exploration, e.g. by directing teams to unexplored regions as well as for scheduling rescue missions. The proposed interface can be extended further by additional sensors, such as CO_2 detectors, temperature sensors, and cameras, for facilitating the transmission of images and data that document the degree of damages or the state of victims at certain locations. The overall goal in this context is to design a system that integrates all observations from both human responders and robot rescue teams at an incident commander perspective. The strategic overview of the scenario allows incident commanders to efficiently combine the capabilities of individual team members by developing multi-robot and multi-human plans, given the available resources.

9.2 Future Work

As results from PDR experiments showed, the tracking of human beings based on accelerometers leads to promising results. However, also these results are only preliminary compared to the situations first responders are exposed to while rescuing victims after a real disaster. They might crouch, run, and climb within very different and individual modes, generating a wide range of ambiguous sensor values. Further research has to be undertaken in order to capture the whole range of human motion.

The proposed method for RFID-SLAM has shown to work robustly in various scenarios. However, one drawback of this method is given by the low-range detection of RFIDs due to the utilized technology. In future work, we will evaluate RFID technology

operating in the UHF frequency domain, allowing reading and writing within distances of meters, and to extend the proposed approach accordingly.

The approach of RFID technology-based coordination has been tested mainly in simulated disaster areas. In future work, it will be evaluated on a team of real robots equipped with long-range RFID detectors while exploring an outdoor area. Furthermore, the plan execution phase of the approach will be improved. It is considered to represent the generated multi-robot plans by *Petri* nets, allowing to verify online plans generated from task assignment and path planning. Given this representation, parallel plan executions can be simulated beforehand for predicting unsafe situations. On the one hand, this will facilitate the optimization of robot waiting times if they are involved within conflicts. On the other hand, it will enable deadlock detection during the planning phase.

The proposed victim detection method serves as a generalized framework and hence can be further extended. In future work we will consider to extend the binary class variable by classes describing the victim's state, such as *aware* and *unconscious*. These classes can also be determined from correlations between different evidence types. For example, the detection of CO_2 emission and heat without the detection of motion can indicate an unconscious victim.

Most importantly, future work will deal with the problem of combining RFID route graph optimization with local elevation mapping in order to facilitate the building of large and consistent elevation maps in real-time. Here the basic idea is to anchor locally generated elevation maps within the corrected graph. These map patches can then be loaded into memory if they are within range of the robot's current location. Furthermore, the saved data can be utilized offline for generating a global elevation map representation of the environment. This representation opens the door for solving real-time mapping and navigation in large-scale environments, which is an indispensable requirement for future projects and benchmarks. For example, the $600.000 US\$$ endowed *TechX* challenge, a competition organized by the Singaporean national Defense, Science and Technology Authority (DSTA), requires a one hour fully autonomous exploration of a larger area containing buildings and rough outdoor terrain. Robots have to locate targets in buildings and return to their starting location. Such benchmarks demand highly sophisticated robot systems fulfilling their mission robustly during a long time operation, while having to rely on themselves completely.

It can be expected for the future that rescue robots continuously improve according to the increasing difficulty of benchmarking scenarios. Moreover, it is planned by the RoboCup Rescue community to introduce standardized map representations and communication protocols, which will be an important milestone towards implementing heterogeneous robot teams. A common format could serve not only for comparisons between different approaches, but also as a useful tool to exchange spatial information between heterogeneous agents, as well as to compare more rigorously results coming from simulation and real world systems. Furthermore, it makes it possible to provide

floor plans of building structures to rescue teams in advance, as is common practice during real rescue missions. Here, the long term goal is to develop communication standards in the simulation league and to transfer them to the robot league, after they have been extensively tested.

10 Appendix

10.1 Rescue communication protocol

```
<xsd:complexType name="RescueWaypoint">
<xsd:annotation><xsd:documentation>
    This type describes an extension of GPX 1.1 waypoints.
    Waypoints in the disaster area can be augmented
    with additional information, such as observations of fires,
    blockades, victims, and RFIDs.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
       <xsd:element name="Agent"
              type="RescueAgent_t" minOccurs="0" maxOccurs="1" />
       <xsd:element name="Fire"
              type="RescueFire_t" minOccurs="0" maxOccurs="unbounded" />
       <xsd:element name="Blockade"
              type="RescueBlockade_t" minOccurs="0" maxOccurs="unbounded" />
       <xsd:element name="VictimSoundEvidence"
        type="RescueVictimSoundEvidence_t"  minOccurs="0" maxOccurs="unbounded"/>
       <xsd:element name="Victim"
              type="RescueVictim_t" minOccurs="0" maxOccurs="unbounded" />
       <xsd:element name="RFID"
              type="RescueRFID_t" minOccurs="0" maxOccurs="unbounded" />
       <xsd:element name="Exploration"
              type="RescueExploration_t" minOccurs="0" maxOccurs="1" />
    </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="RescueVictim_t">
<xsd:annotation><xsd:documentation>
    This type describes information on a victim
    relatively to the waypoint.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="VictimDescription"
                     type="xsd:string"   "minOccurs="0" maxOccurs="1"/>
        <xsd:element name="VictimSurvivalTime"
                     type="xsd:integer"   "minOccurs="0" maxOccurs="1"/>
        <xsd:element name="VictimRescueTime"
                     type="xsd:integer"   "minOccurs="0" maxOccurs="1"/>
```

```
        <xsd:element name="VictimProximity"
                     type="Meters_t" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="VictimBearing"
                     type="Degree_t" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="VictimDepth"
                     type="Meters_t" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="RescueRFID_t">
<xsd:annotation><xsd:documentation>
    This type describes the observation of an RFID tag, which can
    further be used by the station to correct the agents trajectory.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="ID"
                     type="xsd:integer"   "minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="RFIDProximity"
                     type="Meters_t" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="RFIDBearing"
                     type="Degree_t" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="RFIDHeight"
                     type="Meters_t" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>



<xsd:complexType name="RescueFire_t">
<xsd:annotation><xsd:documentation>
    This type describes the observation of fire
    relatively to the waypoint.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="FireDescription"
                     type="xsd:string"   "minOccurs="0" maxOccurs="1"/>
        <xsd:element name="FireProximity"
                     type="Meters_t" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="FireBearing"
                     type="Degree_t" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="RescueBlockage_t">
<xsd:annotation><xsd:documentation>
    This type describes detected road blockages
    relatively to the waypoint.
```

```
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="BlockageDescription"
                     type="xsd:string"   "minOccurs="0" maxOccurs="1"/>
        <xsd:element name="BlockageProximity"
                     type="Meters_t" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="BlockageBearing"
                     type="Degree_t" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="RescueVictimSoundEvidence_t">
<xsd:annotation><xsd:documentation>
    This type describes evidence on hearing a victim
    relatively to the waypoint.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="VictimEvidenceRadius"
                     type="Meters_t" minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>


<xsd:complexType name="RescueExploration_t">
<xsd:annotation><xsd:documentation>
    This type describes the area that has been exploration
    around the waypoint.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="ExploredRadius"
                     type="Meters_t" minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="RescueAgent_t">
<xsd:annotation><xsd:documentation>
    This type describes the observant agent.
</xsd:documentation></xsd:annotation>
    <xsd:sequence>
        <xsd:element name="AgentName"
                     type="xsd:string"   "minOccurs="0" maxOccurs="1"/>
        <xsd:element name="AgentTeam"
                     type="xsd:string" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>
```

```
<xsd:simpleType name="Meters_t">
  <xsd:annotation><xsd:documentation>
    This type contains a distance value measured in meters.
  </xsd:documentation></xsd:annotation>
  <xsd:restriction base="xsd:integer"/>
</xsd:simpleType>

<xsd:simpleType name="Degree_t">
  <xsd:annotation><xsd:documentation>
    This type contains a bearing value measured in degree.
  </xsd:documentation></xsd:annotation>
  <xsd:restriction base="xsd:integer"/>
</xsd:simpleType>
```

10.2 Construction drawings

10.2.1 3D Laser Range Finder (LRF)

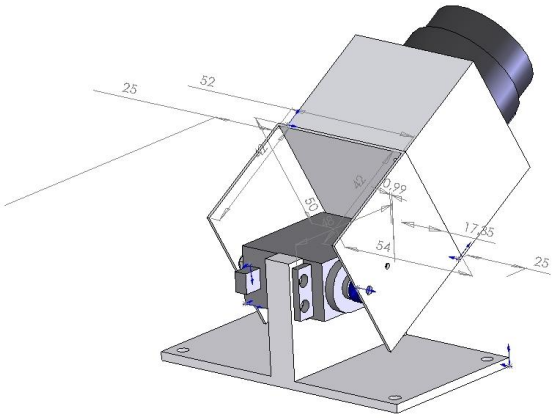


Figure 10.1: Perspective drawing of the 3D LRF mechanics.

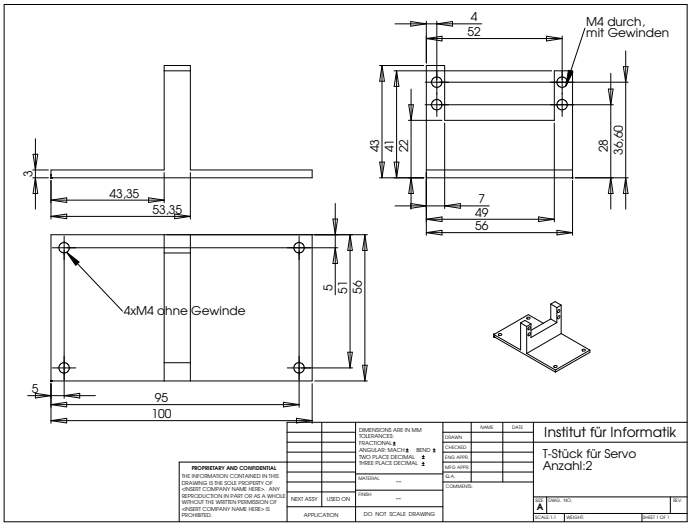


Figure 10.2: Mechanical drawing 3D LRF: servo mount.

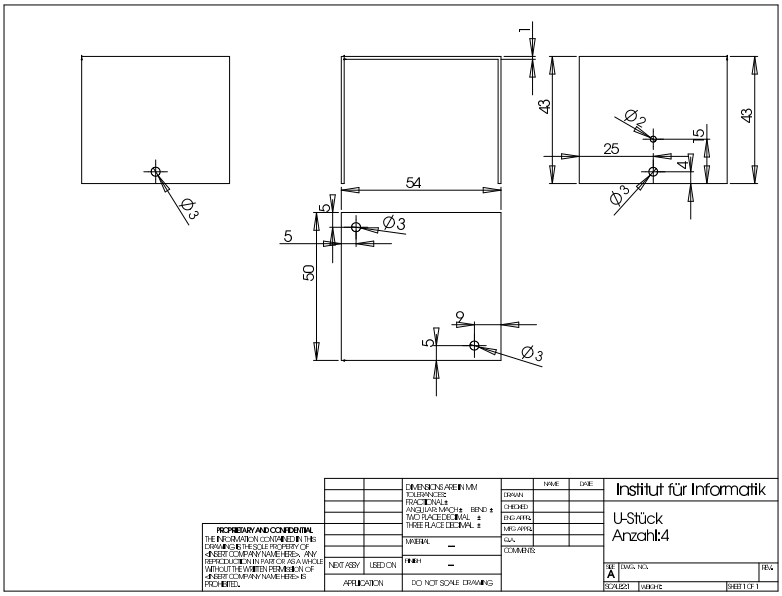


Figure 10.3: Mechanical drawing 3D LRF: U-shaped piece.

10.2.2 Zerg robot

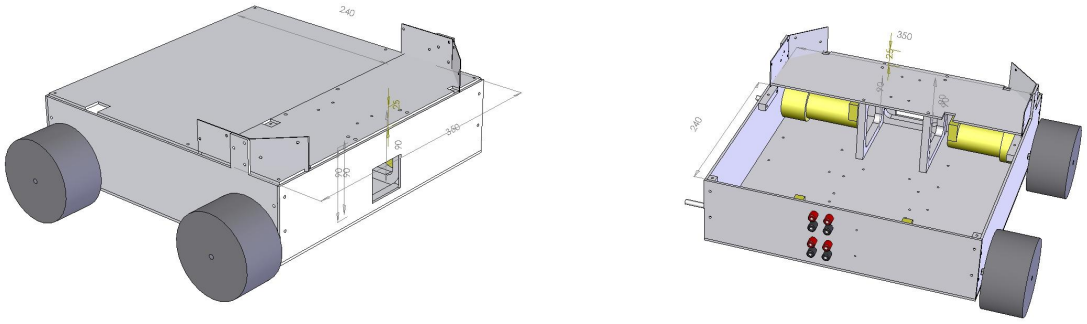


Figure 10.4: Perspective drawings Zerg robot mechanics.

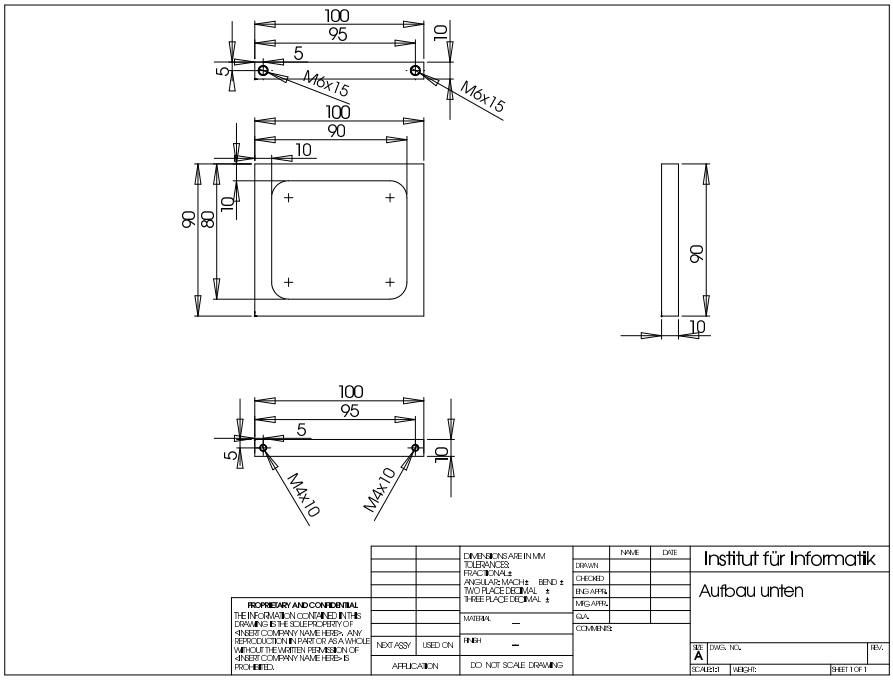


Figure 10.5: Mechanical drawing Zerg robot: inner support frame.

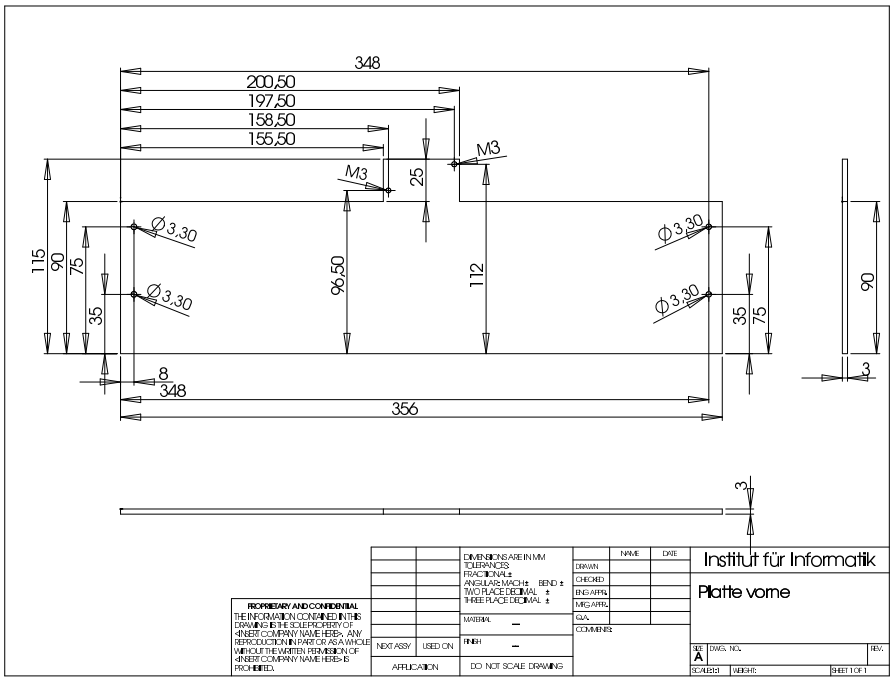


Figure 10.6: Mechanical drawing Zerg robot: front part.

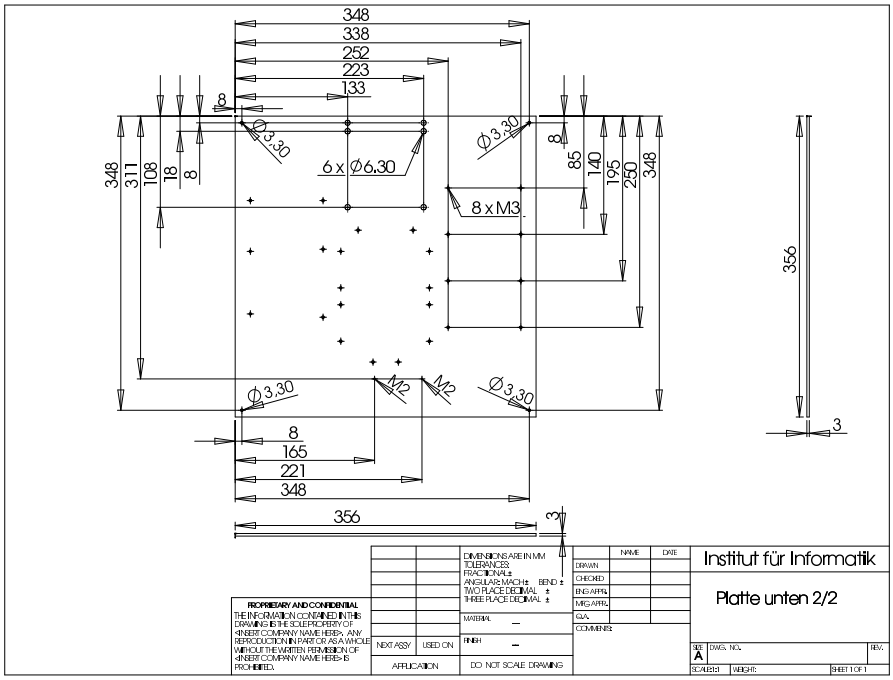


Figure 10.7: Mechanical drawing Zerg robot: bottom part.

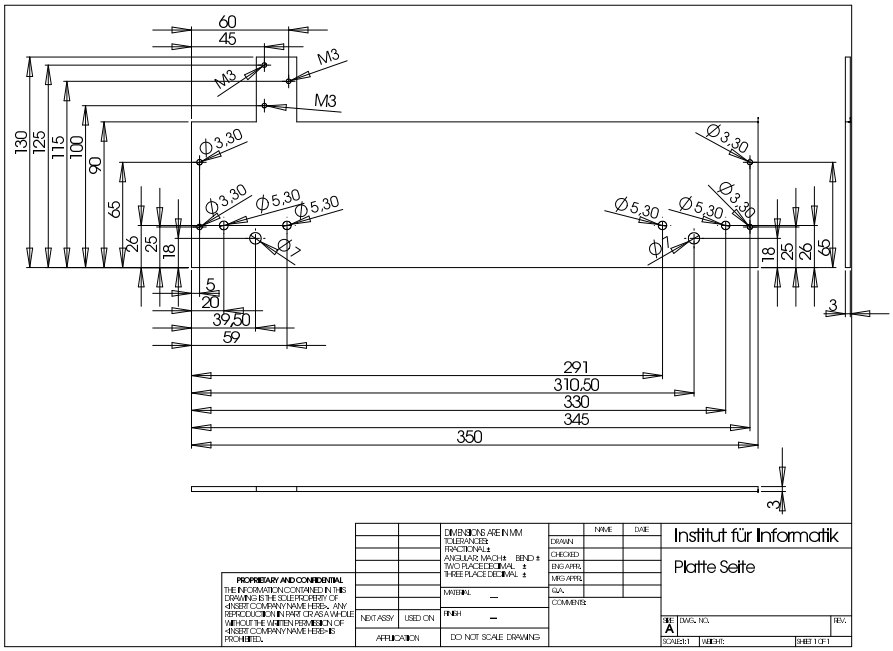


Figure 10.8: Mechanical drawing Zerg robot: side part.

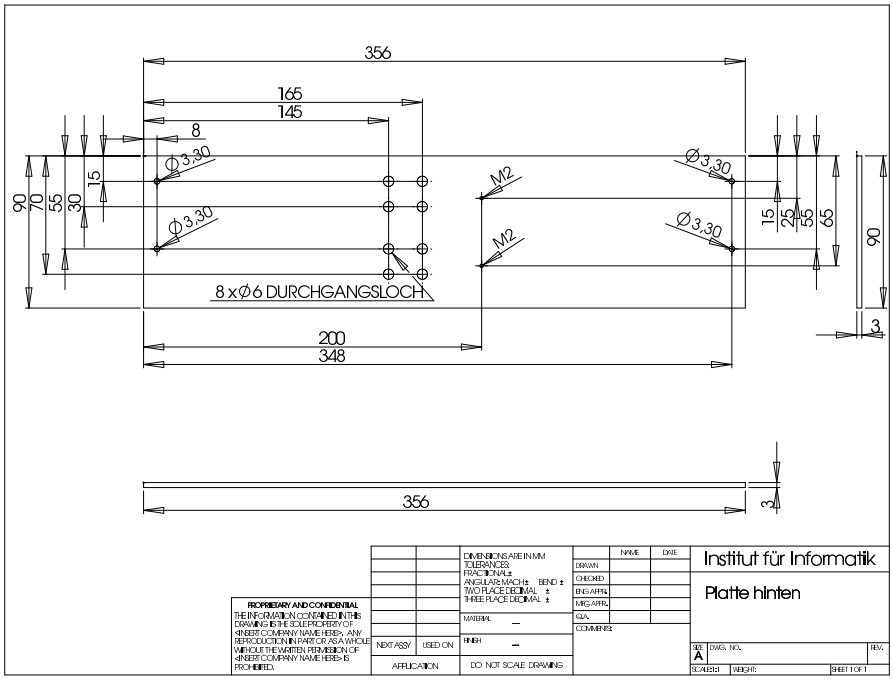


Figure 10.9: Mechanical drawing Zerg robot: back part.

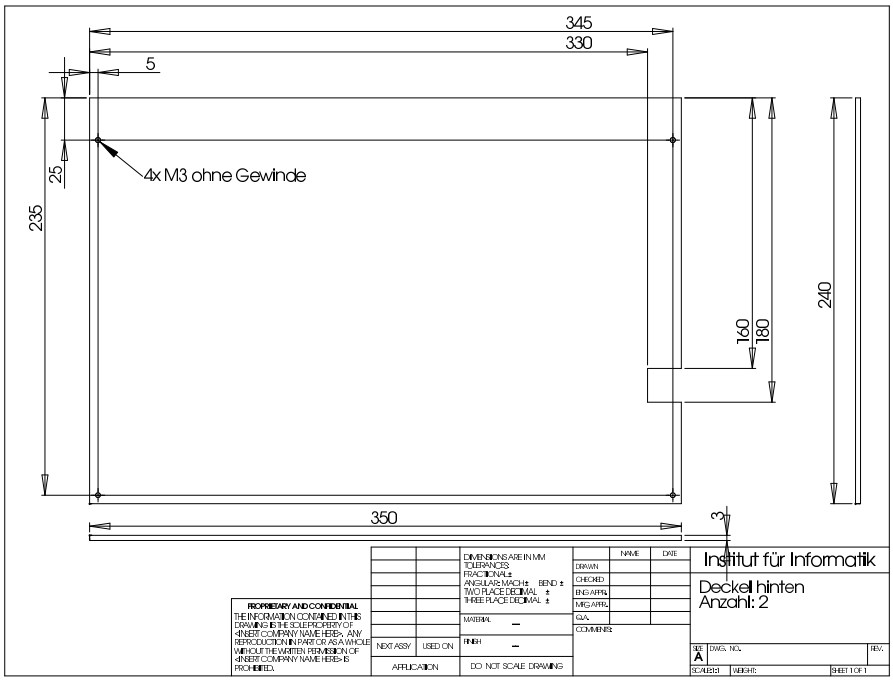


Figure 10.10: Mechanical drawing Zerg robot: cover back part.

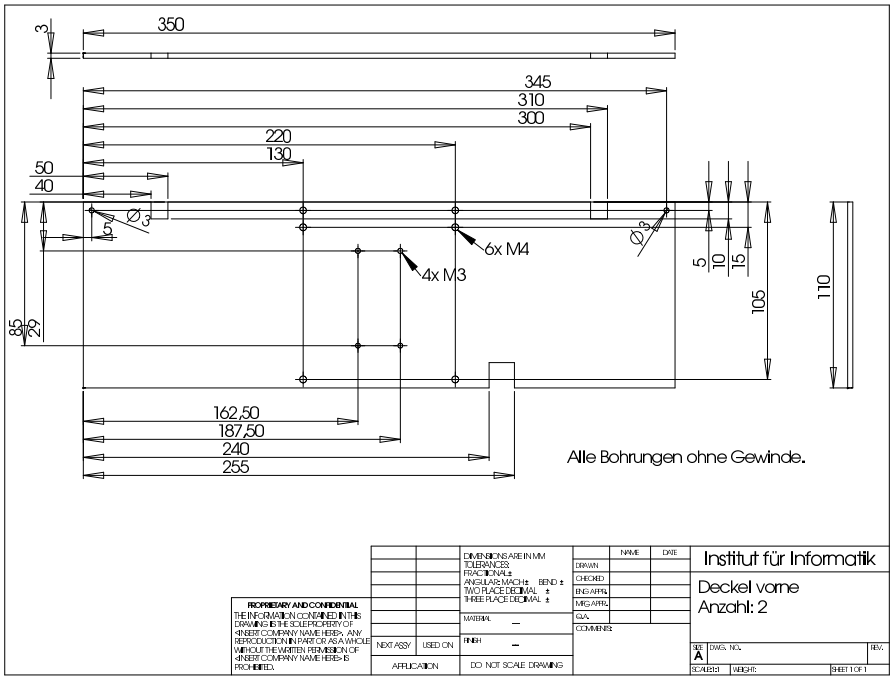


Figure 10.11: Mechanical drawing Zerg robot: cover front part.

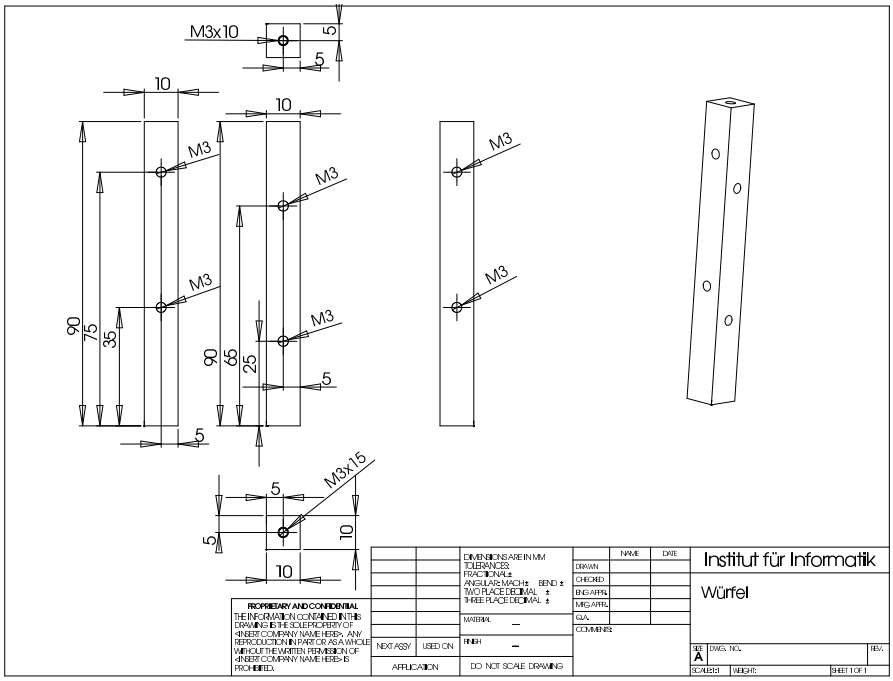


Figure 10.12: Mechanical drawing Zerg robot: assembling cube.

10.2.3 RFID tag releaser

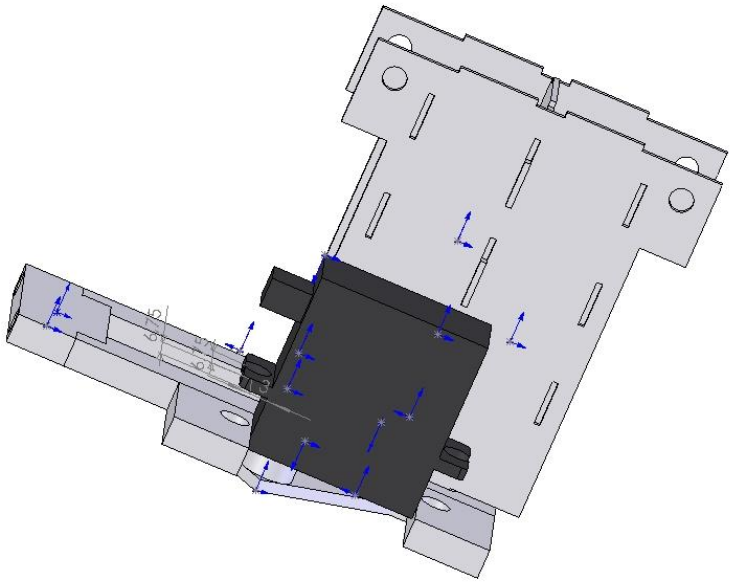


Figure 10.13: Perspective drawing of the RFID tag releaser.

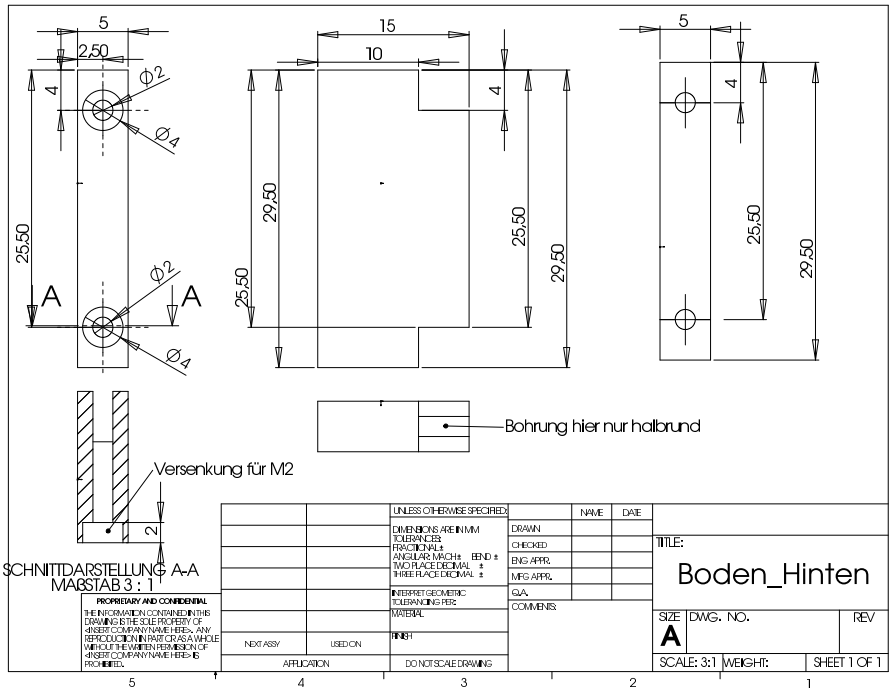


Figure 10.14: Mechanical drawing RFID tag releaser: bottom back part.

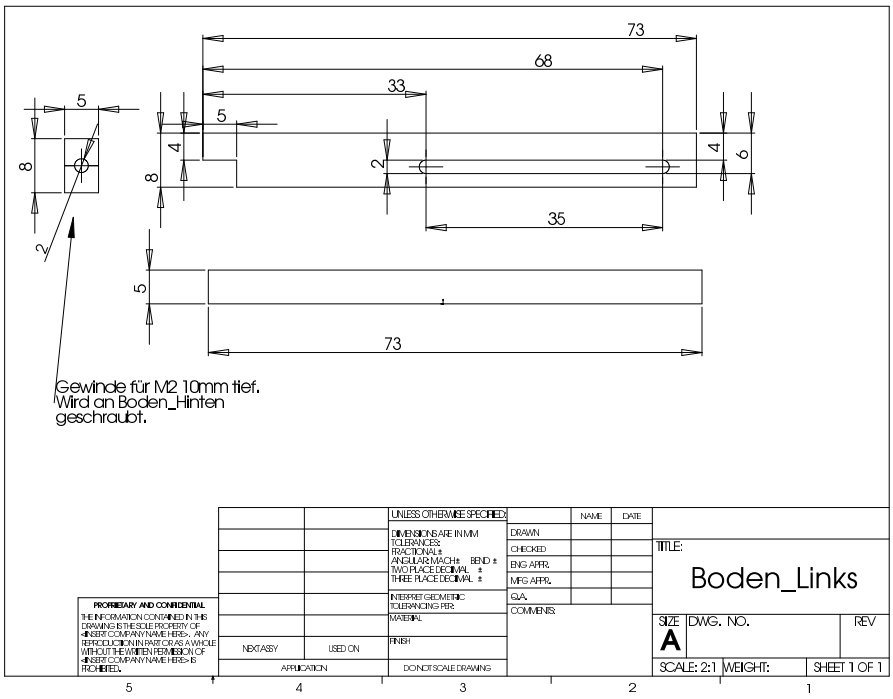


Figure 10.15: Mechanical drawing RFID tag releaser: bottom left part.

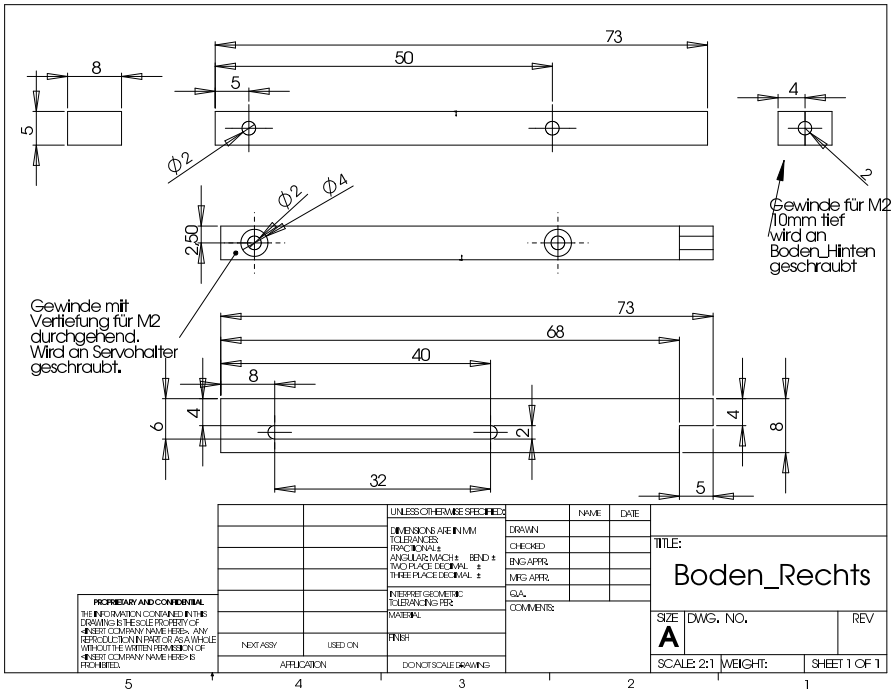


Figure 10.16: Mechanical drawing RFID tag releaser: bottom right part.

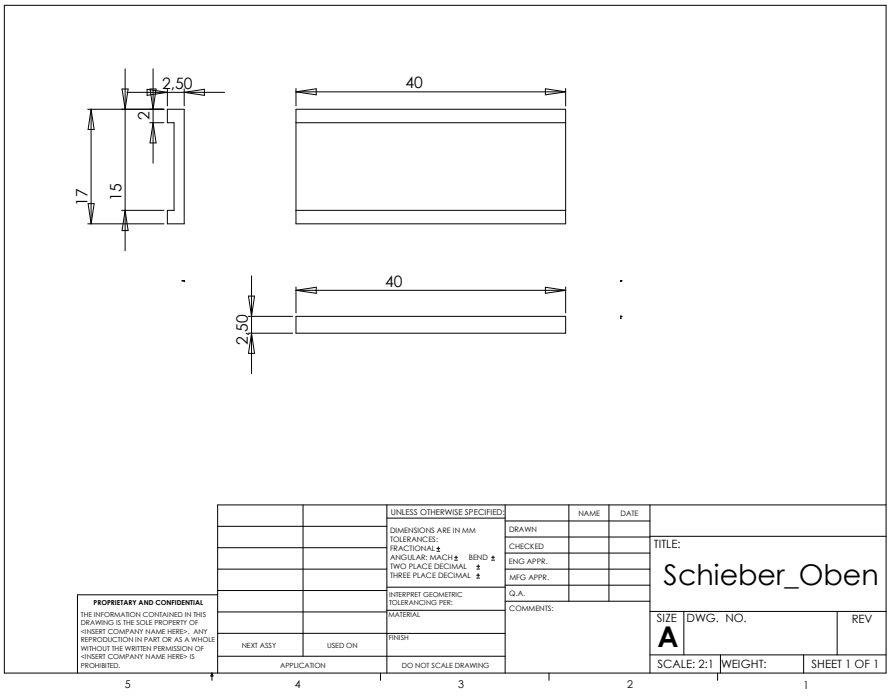


Figure 10.17: Mechanical drawing RFID tag releaser: slider upper part.

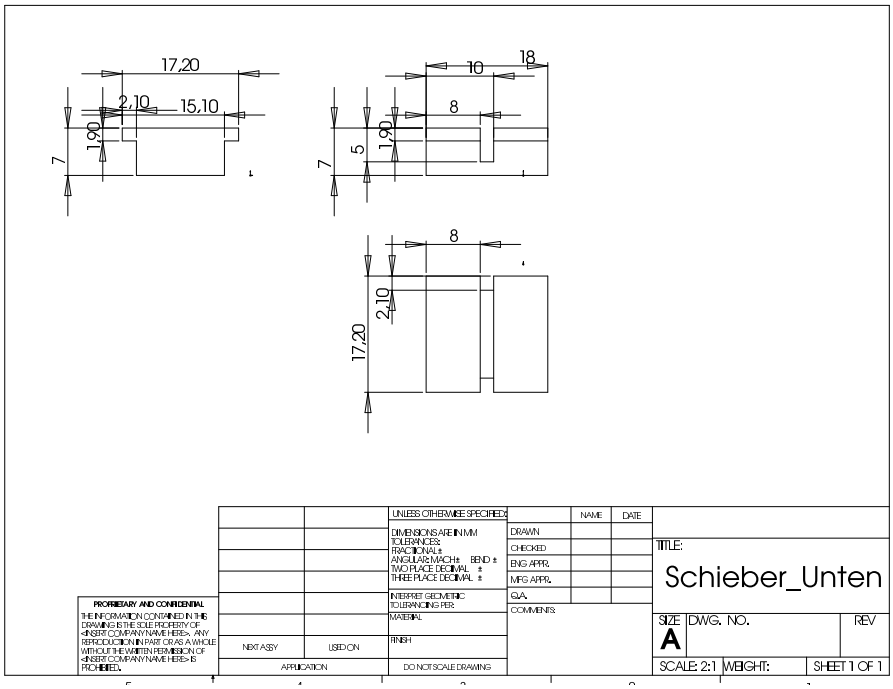


Figure 10.18: Mechanical drawing RFID tag releaser: slider lower part.

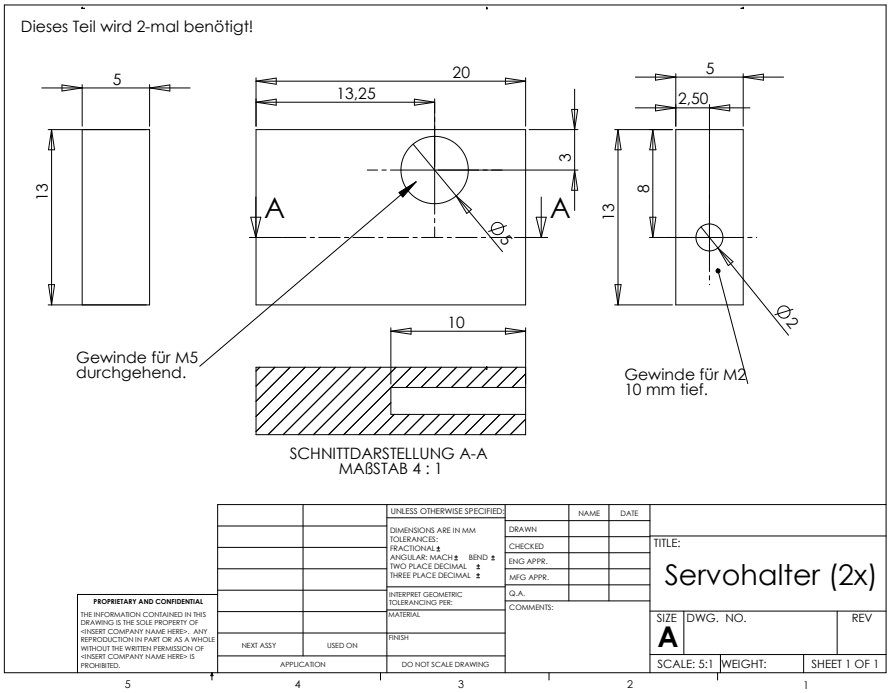


Figure 10.19: Mechanical drawing RFID tag releaser: servo mount.

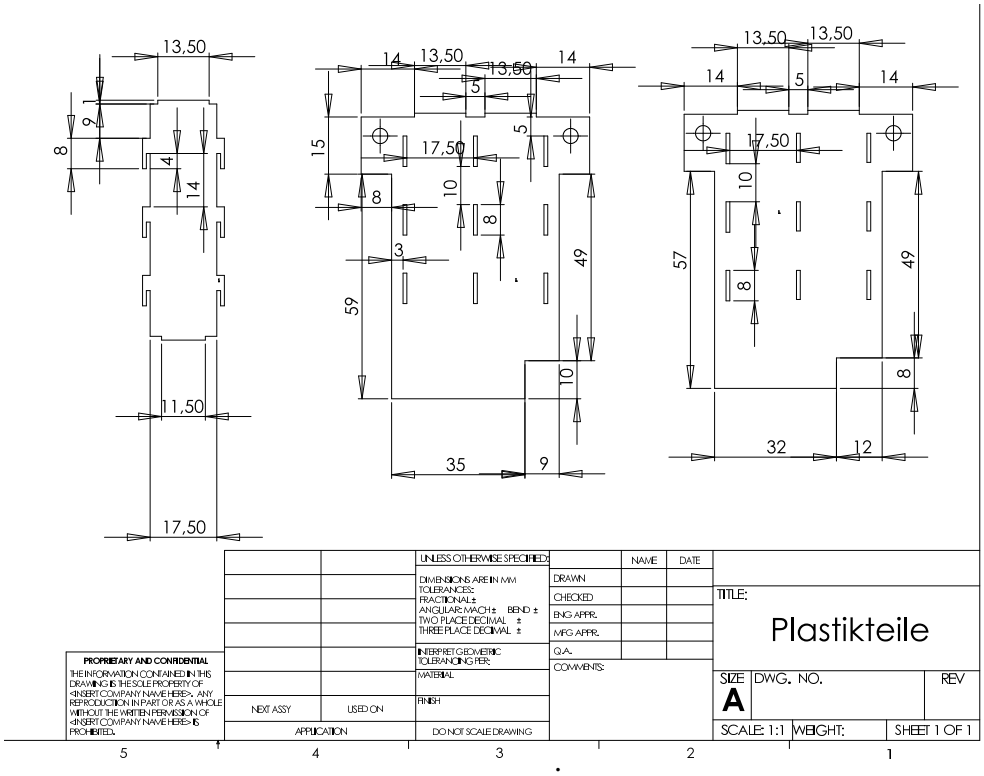
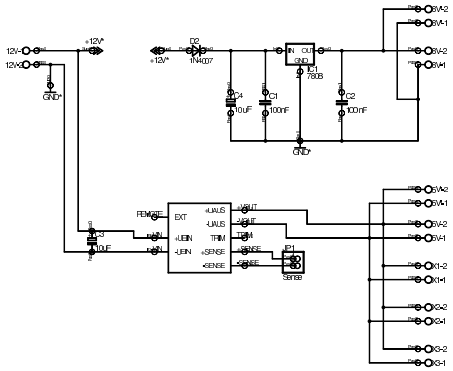
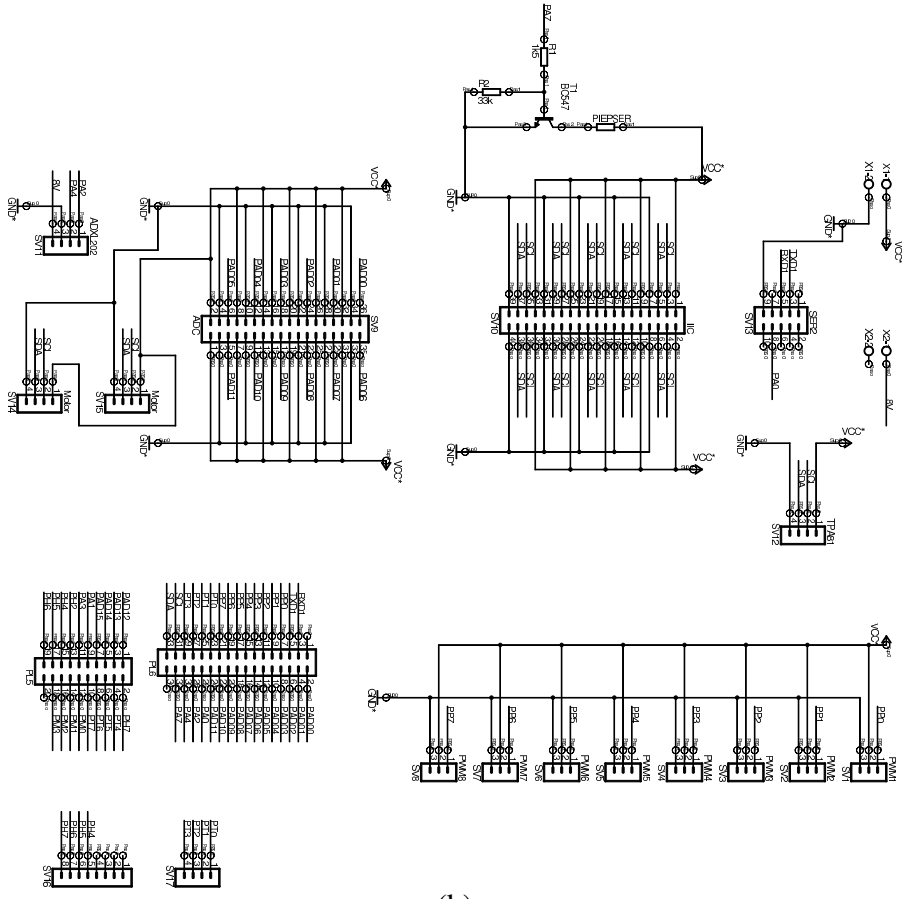


Figure 10.20: Mechanical drawing RFID tag releaser: RFID tag container.

10.3 Schematics



(a)



(b)

Figure 10.21: Schematics of (a) the *Power Supply Board*, and (b) the *Sensor board* of the Zerg robot.

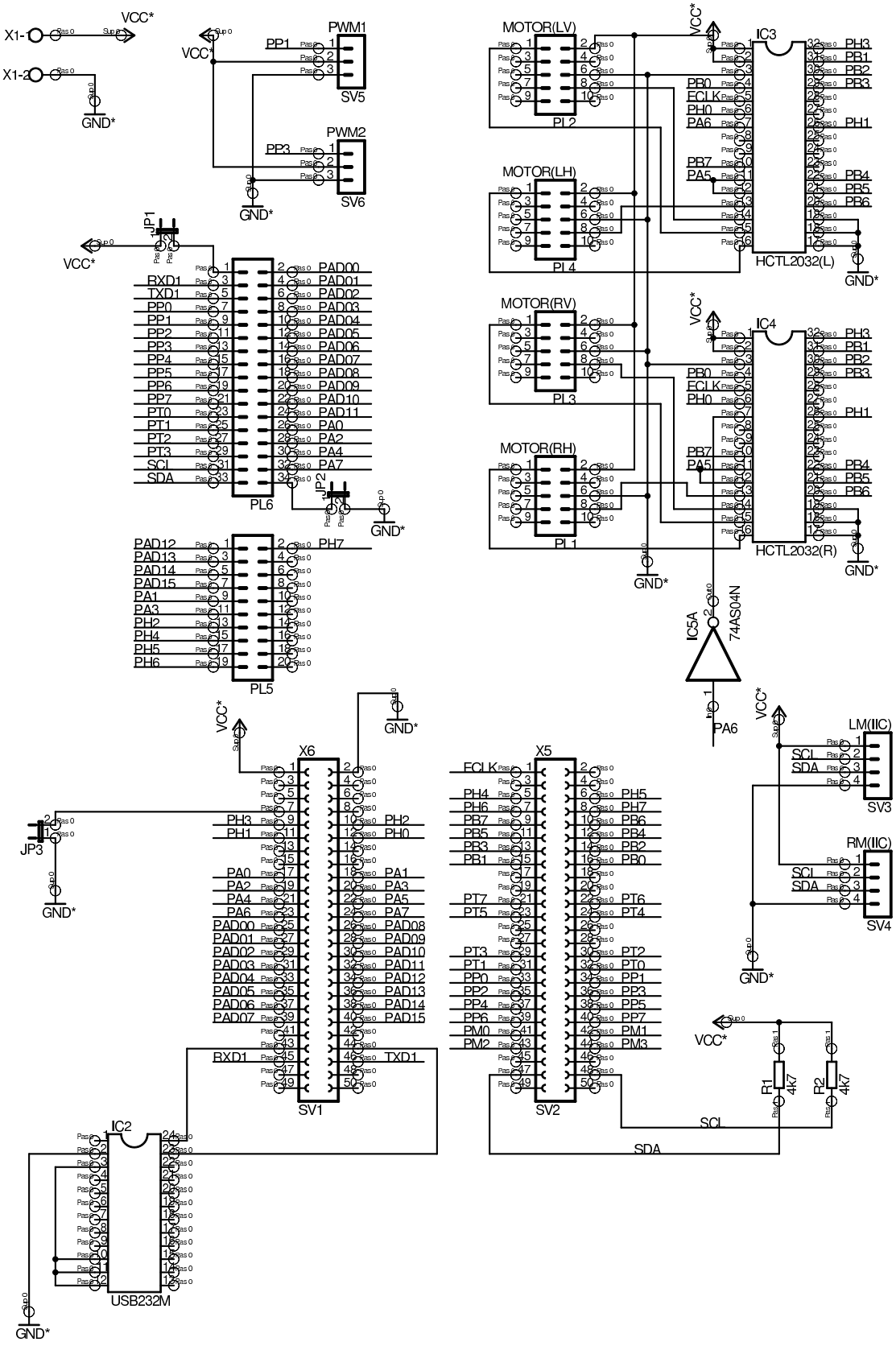


Figure 10.22: Schematics of the *Micro Controller Board* of the Zerg robot.

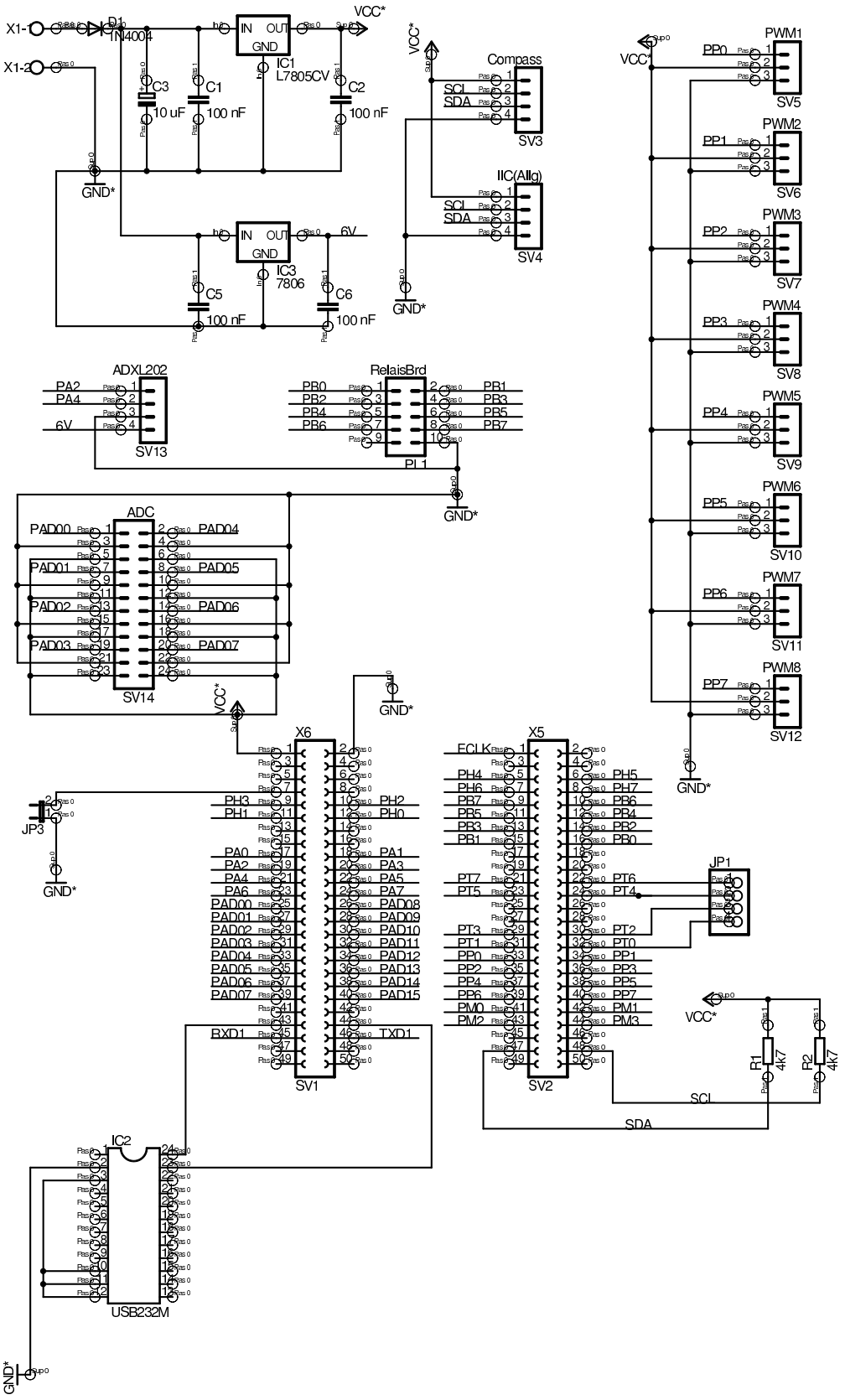
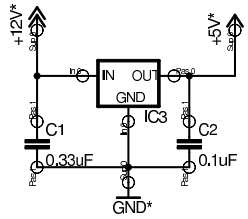


Figure 10.23: Schematics of the *Micro Controller Board* of the Lurker robot.



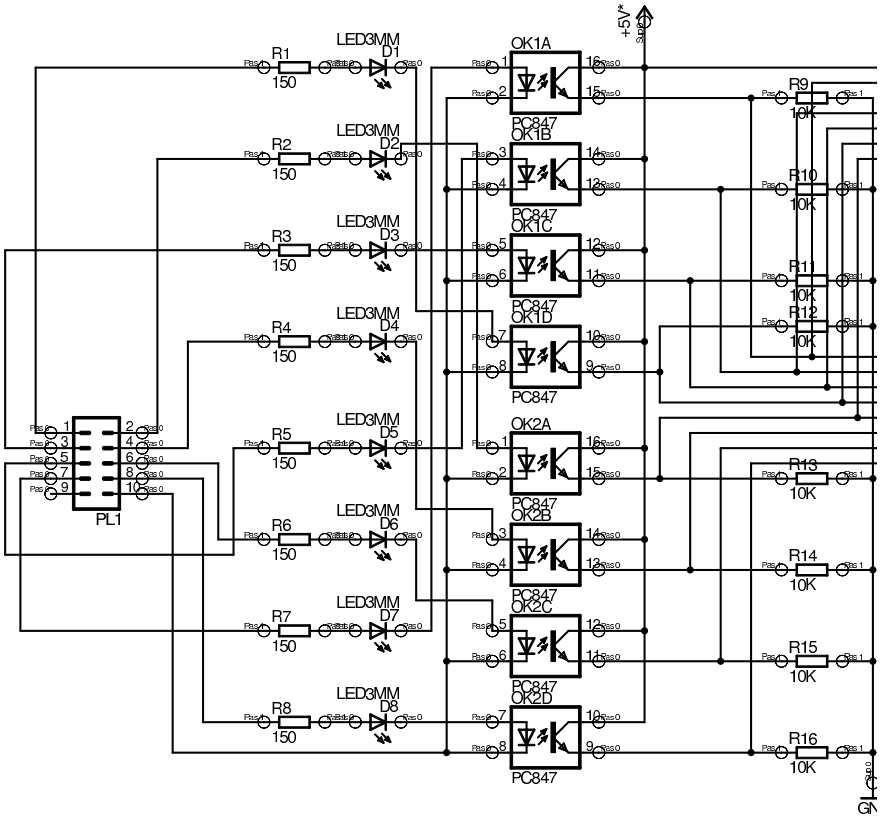


Figure 10.24: Schematics of the *Relais board* of the Lurker robot.

List of Figures

List of Tables

Bibliography

[AAAI, 2002] AAAI (2002). In the aftermath of september 11 what roboticists learned from the search and rescue efforts. `http://www.aaai.org/Pressroom/Releases/release-02-0910.php`.

[Administration and Agency, 1995] Administration, U. S. F. and Agency, F. E. M. (1995). Technical rescue program development manual. Technical report.

[AlienTechnology, 2003] AlienTechnology (2003). Alien technology. `http://www.alientechnology.com/`.

[Amft et al., 2004] Amft, O., Lauffer, M., Ossevoort, S., Macaluso, F., Lukowicz, P., and Tröster, G. (2004). Design of the QBIC wearable computing platform. In *15th International Conference on Application-Specific Systems, Architectures and Processors (ASAP '04)*, Galveston, Texas.

[Anguelov et al., 2005] Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., and Ng, A. (2005). Discriminative learning of markov random fields for segmentation of 3D range data. In *IEEE Computer Society Conference on Vision and Pattern Recognitionv (CVPR)*, San Diego, California.

[Arai et al., 2002] Arai, T., Pagello, E., and Parker, L. (2002). Guest editorial advances in multirobot systems. *IEEE Transactions on Robotics and Automation*, 18(5):655–661.

[Bahadori and Iocchi, 2003] Bahadori, S. and Iocchi, L. (2003). Human body detection in the robocup rescue scenario. In *RoboCup 2003: Robot Soccer World Cup VII*.

[Bailey, 2002] Bailey, T. (2002). *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, Australian Centre for Field Robotics, Australia.

[Balakirsky et al., 2007] Balakirsky, S., Carpin, S., Kleiner, A., Lewis, M., Visser, A., Wang, J., and Ziparo, V. (2007). Towards heterogeneous robot teams for disaster mitigation: Results and performance metrics from robocup rescue. *Journal of Field Robotics*, 24(11-12):943–967.

[Balakirsky et al., 2006] Balakirsky, S., Scrapper, C., Carpin, S., and Lewis., M. (2006). USARSim: providing a framework for multi-robot performance evaluation. In *Proc. of the Performance Metrics for Intelligent Systems (PerMIS) Workshop*.

[Balch and Arkin, 1994] Balch, T. and Arkin, R. C. (1994). Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52.

[Bank and Douglas, 1993] Bank, R. E. and Douglas, C. C. (1993). Sparse matrix multiplication package (SMMP). *Advances in Computational Mathematics*, 1:127–137.

[Beauregard, 2006] Beauregard, S. (2006). A helmet-mounted pedestrian dead reckoning system. In Herzog, O., Kenn, H., Lawo, M., Lukowicz, P., and Tröster, G., editors, *3rd International Forum on Applied Wearable Computing (IFAWC'03)*, pages 79–91. VDE Verlag.

[Beleznai et al., 2004] Beleznai, C., Frühstück, B., and Bischof, H. (2004). Human detection in groups using a fast mean shift procedure. In *Proc. IEEE Int. Conf. on Image Processing*, pages 349–352.

[Bennewitz et al., 2001] Bennewitz, M., Burgard, W., and Thrun, S. (2001). Optimizing schedules for prioritized path planning of multi-robot systems. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, volume 1, pages 271–276.

[Bennewitz et al., 2005] Bennewitz, M., Faber, F., Joho, D., Schreiber, M., and Behnke, S. (2005). Integrating vision and speech for conversations with multiple persons. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 2523–2528, Canada.

[Bentley, 1975] Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517.

[Bhatt and Glover, 2006] Bhatt, H. and Glover, B. (2006). *RFID Essentials*. O'Reilly.

[Birk, 2003] Birk, A. (2003). RoboCupRescue - robot league team IUB rescue, Germany. In *RoboCup 2004 (CDROM Proceedings), Team Description Paper, Rescue Robot League*, Bremen, Germany.

[Bock, 1974] Bock, H. (1974). *Automatic Classification*. Vandenhoeck and Ruprecht, Göttingen.

[Bohn and Mattern, 2004] Bohn, J. and Mattern, F. (2004). Super-distributed RFID tag infrastructures. In *Proceedings of the 2nd European Symposium on Ambient Intelligence (EUSAI 2004)*, number 3295 in Lecture Notes in Computer Science (LNCS), pages 1–12, Eindhoven, The Netherlands. Springer-Verlag.

[Borenstein, 1996] Borenstein, J. (1996). Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation*, 12(6):869–880.

[Borenstein et al., 1996] Borenstein, J., Everett, H., and Feng, L. (1996). Where am i? sensors and methods for mobile robot positioning. Technical report, University of Michigan.

[Bradski, 2000] Bradski, G. (2000). The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 25(11):120, 122–125.

[Bruce, 2000] Bruce, J. (2000). Realtime machine vision perception and prediction. Master's thesis, Carnegie Mellon University.

[Burgard et al., 2005] Burgard, W., Moors, M., Stachniss, C., and Schneider, F. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–378.

[Cao et al., 2004] Cao, D., Masoud, O., Boley, D., and Papanikolopoulos, N. (2004). Online motion classification using support vector machines. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2291–2296.

[Carpin et al., 2006a] Carpin, S., Lewis, M., Wang, J., Balakirsky, S., and Scrapper., C. (2006a). Bridging the gap between simulation and reality in urban search and rescue. In *Robocup 2006: Robot Soccer World Cup X*. Springer, LNAI.

[Carpin et al., 2006b] Carpin, S., Stoyanov, T., Nevatia, Y., Lewis, M., and Wang., J. (2006b). Quantitative Assessments of USARSim Accuracy. In *Proc. of the Performance Metrics for Intelligent Systems (PerMIS) Workshop*.

[Casper et al., 2000] Casper, J., Micire, M., and Murphy, R. (2000). Issues in intelligent robots for search and rescue. In *SPIE Ground Vehicle Technology II*, volume 4, pages 41–46.

[Casper and Murphy, 2003] Casper, J. and Murphy, R. (2003). Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Transactions On Systems Man And Cybernetics Part B Cybernetics*, 33(3):367–385.

[Co. KG, 2005] Co. KG, V. T. G. . (2005). Vorwerk is presenting the first carpet containing integrated RFID technology. Press release.

[Corke et al., 2004] Corke, P. I., Strelow, D., and Singh, S. (2004). Omnidirectional visual odometry for a planetary rover. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, volume 4, pages 4007–4012.

[Cox, 1991] Cox, I. J. (1991). Blanche: Position estimation for an autonomous robot vehicle. In Iyengar, S. S. and Elfes, A., editors, *Autonomous Mobile Robots: Control, Planning, and Architecture (Vol. 2)*, pages 285–292. IEEE Computer Society Press, Los Alamitos, CA.

[CRASAR, 2007] CRASAR (last seen 2007). Return to Katrina. `http://crasar.csee.usf.edu/Research/Projects/Katrina_SGER/index.php`.

[Cutler and Davis, 2000] Cutler, R. and Davis, L. S. (2000). Robust real-time periodic motion detection, analysis, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):781–796.

[Dass et al., 2002] Dass, S., Jain, A., and Lu, X. (2002). Face detection and synthesis using markov random field models. In *Proc. of the Int. Conf. on Pattern Recognition (ICPR)*, volume 4, pages 201–204, Washington, DC, USA. IEEE Computer Society.

[Davison, 2003] Davison, A. J. (2003). Real-time simultaneous localisation and mapping with a single camera. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, page 1403, Washington, DC, USA. IEEE Computer Society.

[Dias et al., 2004] Dias, M., Zinck, M., Zlot, R., and Stentz, A. (2004). Robust multirobot coordination in dynamic environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, volume 4, pages 3435–3442.

[Dijkstra, 1959] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

[Dippold, 2006] Dippold, M. (2006). Personal dead reckoning with accelerometers. In Herzog, O., Kenn, H., Lawo, M., Lukowicz, P., and Tröster, G., editors, *3rd International Forum on Applied Wearable Computing (IFAWC'03)*, pages 177–183.

[Dissanayake et al., 2001] Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H., and Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241.

[Djugash et al., 2005] Djugash, J., Singh, S., and Corke, P. (2005). Further results with localization and mapping using range from radio. In *Proc. of the Int. Conf. on Field and Service Robotics (FSR)*, Pt. Douglas, Australia.

[Dornhege and Kleiner, 2006] Dornhege, C. and Kleiner, A. (2006). Visual odometry for tracked vehicles. In *Proc. of the IEEE Int. Workshop on Safety, Security and Rescue Robotics (SSRR)*, Gaithersburg, Maryland, USA.

[Dornhege and Kleiner, 2007a] Dornhege, C. and Kleiner, A. (2007a). Behavior maps for online planning of obstacle negotiation and climbing on rough terrain. In *Video Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, San Diego, California.

[Dornhege and Kleiner, 2007b] Dornhege, C. and Kleiner, A. (2007b). Behavior maps for online planning of obstacle negotiation and climbing on rough terrain. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 3005–3011, San Diego, California.

[Durrant-Whyte et al., 1996] Durrant-Whyte, H., Rye, D., and Nebot, E. (1996). Localisation of automatic guided vehicles. In *Robotics Research: The 7th International Symposium (ISRR'95)*, pages 613–625. Springer Verlag.

[Elfes, 1989] Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57.

[Eppstein, 2000] Eppstein, D. (2000). Fast hierarchical clustering and other applications of dynamic closest pairs. *J. Exp. Algorithmics*, 5:619–628.

[Erdmann and Lozano-Perez, 1987] Erdmann, M. and Lozano-Perez, T. (1987). On multiple moving objects. *Algorithmica*, 2(2):477–521.

[Fachberger et al., 2006] Fachberger, R., Bruckner, G., Reindl, L., and Hauser, R. (2006). Tagging of metallic objects in harsh environments. In *Sensoren und Messsysteme - 13. ITG-/GMA Fachtagung*, University of Freiburg.

[Farinelli et al., 2006] Farinelli, A., Iocchi, L., Nardi, D., and Ziparo, V. (2006). Assignment of dynamically perceived tasks by token passing in multirobot systems. *Proceedings of the IEEE*, 94(7):1271–1288.

[FEMA, 2003] FEMA (2003). National urban search and rescue (USR) response system - field operations guide. Technical Report US&R-2-FG, U.S. Department of Homeland Security. `http://www.fema.gov/emergency/usr/resources.shtm`.

[Finkenzeller, 2003] Finkenzeller, K. (2003). *RFID-Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley and Sons, 2nd edition.

[Fishkin and Roy, 2003] Fishkin, K. and Roy, S. (2003). Enhancing RFID privacy via antenna energy analysis. Technical Report IRS-TR-03-012, Intel Research Seattle.

[Foxlin, 2005] Foxlin, E. (2005). Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Comput. Graph. Appl.*, 25(6):38–46.

[Frese, 2006] Frese, U. (2006). Treemap: An $O(logn)$ algorithm for indoor simultaneous localization and mapping. *Autonomous Robots*, 21(2):103–122.

[Freund and Schapire, 1996] Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *International Conference on Machine Learning (ICML)*, pages 148–156.

[Gabaglio, 2003] Gabaglio, V. (2003). GPS/INS integration for pedestrian navigation. *Geodätisch-geophysikalische Arbeiten in der Schweiz*, 64:161.

[Gassmann et al., 2003] Gassmann, B., Frommberger, L., Dillmann, R., and Berns, K. (2003). Real-time 3D map building for local navigation of a walking robot in unstructured terrain. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, volume 3, pages 2185–2190.

[Giuliani et al., 1994] Giuliani, D., Omologo, M., and Svaizer, P. (1994). Talker localization and speech recognition using a microphone array and a cross-powerspectrum phase analysis. In *Proc. of the ICSLP*, pages 1243–1246.

[Grewal et al., 2001] Grewal, M., Weill, L. R., and Andrews, A. P. (2001). *Global Positioning Systems, Inertial Navigation, and Integration*. John Wiley & Sons.

[Grisetti et al., 2005] Grisetti, G., C., S., and W., B. (2005). Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 667–672, Barcelona, Spain.

[Grisetti et al., 2002] Grisetti, G., Iocchi, L., and Nardi, D. (2002). Global Hough localization for mobile robots in polygonal environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, volume 1, pages 353–358.

[Gutmann and Schlegel, 1996] Gutmann, J. and Schlegel, C. (1996). Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *Proceedings of the 1st Euromicro Workshop on Advanced Mobile Robots*, pages 61–67. IEEE Computer Society Press.

[Gutmann, 2000] Gutmann, J.-S. (2000). *Robuste Navigation autonomer mobiler Systeme*. PhD thesis, Albert-Ludwigs-Universit"at Freiburg. ISBN 3-89838-241-9.

[Gutmann et al., 2001] Gutmann, J.-S., Weigel, T., and Nebel, B. (2001). A fast, accurate, and robust method for self-localization in polygonal environments using laser-range-finders. *Advanced Robotics*, 14(8):651–668.

[Hähnel, 2005] Hähnel, D. (2005). *Mapping with Mobile Robots*. PhD thesis, Universität Freiburg, Freiburg, Deutschland.

[Hähnel et al., 2004] Hähnel, D., Burgard, W., Fox, D., Fishkin, K., and Philipose, M. (2004). Mapping and localization with RFID technology. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, volume 1, pages 1015–1020.

[Hähnel et al., 2003] Hähnel, D., D., Burgard, W., Fox, D., and Thrun, S. (2003). A highly efficient fastslam algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 206–211, Las Vegas, USA.

[Han and Bhanu, 2003] Han, J. and Bhanu, B. (2003). Detecting moving humans using color and infrared video. In *Proc. of the IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 228–233.

[Hancke and Kuhn, 2005] Hancke, G. and Kuhn, M. (2005). An RFID distance bounding protocol. In *First Int. Conf. on Security and Privacy for Emerging Areas in Communications Networks*, pages 67– –73.

[Hatzack and Nebel, 2001] Hatzack, W. and Nebel, B. (2001). Solving the operational traffic control problem. In *Proceedings of the 6th European Conference on Planning (ECP'01)*.

[Helmick et al., 2004] Helmick, D., Chang, Y., Roumeliotis, S., Clouse, D., and Matthies, L. (2004). Path following using visual odometry for a mars rover in high-slip environments. In *IEEE Aerospace Conference*, volume 2, pages 772–789.

[Hitachi, 2003] Hitachi (2003). *Mu Chip Data Sheet*. `http://www.hitachi-eu.com/mu/products/mu_chip_data_sheet.pdf`.

[Holland, 1975] Holland, J. H. (1975). *Adaption in Natural and Artificial Systems*. University of Michigan Press.

[Honeywell, 2007] Honeywell (2007). Dead-reckoning module DRM 5. Available on: `http://www.ssec.honeywell.com/magnetic/products.html`. Referenced 2007.

[IABG mbH, 2007] IABG mbH (2007). Iabg homepage. `http://www.iabg.de`.

[IEEE, 2002] IEEE (2002). The IEEE standard 802.15.1: Wireless personal area network standard based on the bluetooth v1.1 foundation specifications.

[Jacoff et al., 2000] Jacoff, A., Messina, E., and Evans, J. (2000). A standard test course for urban search and rescue robots. In *Proc. of the Performance Metrics for Intelligent Systems (PerMIS) Workshop*, pages 253–259.

[Jacoff et al., 2001] Jacoff, A., Messina, E., and Evans, J. (2001). Experiences in deploying test arenas for autonomous mobile robots. In *Proc. of the Performance Metrics for Intelligent Systems (PerMIS) Workshop*, Mexico City, Mexico.

[Jacoff et al., 2003] Jacoff, A., Messina, E., Weiss, B., Tadokoro, S., and Nakagawa, Y. (2003). Test arenas and performance metrics for urban search and rescue robots. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 27–31, Las Vegas.

[Jaeger and Nebel, 2001] Jaeger, M. and Nebel, B. (2001). Decentralized collision avoidance, deadlock detection, and deadlock resolution for multiple mobile robots. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*.

[Jennings et al., 1997] Jennings, J., Whelan, G., and Evans, W. (1997). Cooperative search and rescue with a team of mobile robots. In *Proc. of the IEEE Int. Conf. on Advanced Robotics (ICAR)*, pages 193–200.

[Judd, 1997] Judd, D. C. T. (1997). A personal dead reckoning module. In *Proceedings of the Institute of Navigation's GPS Conference*, pages 169 – 170.

[Kantor and Singh, 2002] Kantor, G. and Singh, S. (2002). Preliminary results in range-only localization and mapping. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, volume 2, pages 1818–1823, Washington D.C., USA.

[Kantor et al., 2003] Kantor, G., Singh, S., Peterson, R., Rus, D., Das, A., Kumar, V., Pereira, G., and Spletzer, J. (2003). Distributed search and rescue with robot and sensor team. In *Proc. of the Int. Conf. on Field and Service Robotics (FSR)*, pages 327–332. Sage Publications.

[Kehagias et al., 2006] Kehagias, A., Djugash, J., and Singh, S. (2006). Range-only SLAM with interpolated range data. Technical Report CMU-RI-TR-06-26, Robotics Institute, Carnegie Mellon University.

[Kenn and Kleiner, 2007] Kenn, H. and Kleiner, A. (2007). Towards the integration of real-time real-world data in urban search and rescue simulation. In Löffler, J. and Klann, M., editors, *Int. Workshop on Mobile Information Technology for Emergency Response - MobileResponse 2007*, volume 4458 of *LNCS*, pages 106–115. Springer.

[Kenn and Pfeil, 2006] Kenn, H. and Pfeil, A. (2006). Localizing victims through sound and probabilistic grid maps in urban search and rescue scenario. In Noda, I., Jacoff, A., Bredenfeld, A., and Takahashi, Y., editors, *Robocup 2005: Robot Soccer World Cup IX*, pages 312–322. Springer.

[Kim et al., 2004] Kim, J., Ong, S., Nettleton, E., and Sukkarieh, S. (2004). Decentralised approach to unmanned aerial vehicle navigation: without the use of GPS and preloaded map. *Journal of Aerospace Engineering, Professional Engineering Publishing*, 218(6):399–416.

[Kitano and Tadokoro, 2001] Kitano, H. and Tadokoro, S. (2001). A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22:39–52.

[Kitano et al., 1999] Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., and Shimada, S. (1999). RoboCup Rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *IEEE Conf. on Man, Systems, and Cybernetics(SMC-99)*.

[Kleiner, 2005] Kleiner, A. (2005). Game AI: The shrinking gap between computer games and ai systems. *Ambient Intelligence: The evolution of technology, communication and cognition towards the future of human-computer interaction*, 6:143–155.

[Kleiner et al., 2006a] Kleiner, A., Behrens, N., and Kenn, H. (2006a). Wearable computing meets multiagent systems: A real-world interface for the RoboCupRescue simulation platform. In Jennings, N., Tambe, M., Ishida, T., and Ramchurn, S., editors, *First International Workshop on Agent Technology for Disaster Management at AAMAS06*, pages 116–123, Hakodate, Japan. AAMAS Press.

[Kleiner et al., 2004] Kleiner, A., Brenner, M., Bräuer, T., Dornhege, C., Göbelbecker, M., Luber, M., Prediger, J., and Stückler, J. (2004). ResQ Freiburg: Team description and evaluation. In *RoboCup 2004 (CDROM Proceedings), Team Description Paper, Rescue Simulation League*, Lisbon, Portugal. (1st place in the competition).

[Kleiner et al., 2005a] Kleiner, A., Brenner, M., Bräuer, T., Dornhege, C., Göbelbecker, M., Luber, M., Prediger, J., Stückler, J., and Nebel, B. (2005a). Successful search and rescue in simulated disaster areas. In Bredenfeld, A., Jacoff, A., Noda, I., and Takahashi, Y., editors, *Robocup 2005: Robot Soccer World Cup IX*, volume 4020 of *Lecture Notes in Computer Science*, pages 323–334. Springer.

[Kleiner and Buchheim, 2003] Kleiner, A. and Buchheim, T. (2003). A plugin-based architecture for simulation in the F2000 league. In Polani, D., Browning, B., Bonarini, A., and Yoshida, K., editors, *RoboCup 2003: Robot Soccer World Cup VII*, volume 3020 of *Lecture Notes in Computer Science*, pages 434–445. Springer.

[Kleiner and Dornhege, 2007] Kleiner, A. and Dornhege, C. (2007). Real-time localization and elevation mapping within urban search and rescue scenarios. *Journal of Field Robotics*, 24(8–9):723–745.

[Kleiner et al., 2006b] Kleiner, A., Dornhege, C., Kümmerle, R., Ruhnke, M., Steder, B., Nebel, B., Doherty, P., Wzorek, M., Rudol, P., Conte, G., Durante, S., and Lundstrom, D. (2006b). RoboCupRescue - robot league team RescueRobots Freiburg (Germany). In *RoboCup 2006 (CDROM Proceedings), Team Description Paper, Rescue Robot League*, Bremen, Germany. (1st place in the autonomy competition).

[Kleiner et al., 2007] Kleiner, A., Dornhege, C., and Sun, D. (2007). Mapping disaster areas jointly: RFID-coordinated slam by humans and robots. In *Proc. of the IEEE Int. Workshop on Safety, Security and Rescue Robotics (SSRR)*, pages 1–6, Rom, Italy.

[Kleiner and Kümmerle, 2007] Kleiner, A. and Kümmerle, R. (2007). Genetic MRF model optimization for real-time victim detection in search and rescue. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 3025–3030, San Diego, California.

[Kleiner et al., 2006c] Kleiner, A., Prediger, J., and Nebel, B. (2006c). RFID technology-based exploration and SLAM for search and rescue. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 4054–4059, Beijing, China.

[Kleiner et al., 2005b] Kleiner, A., Steder, B., Dornhege, C., Höfer, D., Meyer-Delius, D., Prediger, J., Stückler, J., Glogowski, K., Thurner, M., Luber, M., Schnell, M., Kümmerle, R., Burk, T., Bräuer, T., and Nebel, B. (2005b). RoboCupRescue - robot league team RescueRobots Freiburg (Germany). In *RoboCup 2005 (CDROM Proceedings), Team Description Paper, Rescue Robot League*, Osaka, Japan. (1st place in the autonomy competition, 4th place in the teleoperation competition).

[Kleiner and Sun, 2007] Kleiner, A. and Sun, D. (2007). Decentralized SLAM for pedestrians without direct communication. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 1461–1466, San Diego, California.

[Kleiner and Ziparo, 2006] Kleiner, A. and Ziparo, V. (2006). RoboCupRescue - simulation league team RescueRobots Freiburg (Germany). In *RoboCup 2006 (CDROM Proceedings), Team Description Paper, Rescue Simulation League*, Bremen, Germany. (1st place in the competition).

[Kohavi and John, 1997] Kohavi, J. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence Journal*, 97:273–324.

[Koyanagi et al., 2006] Koyanagi, E., Oba, Y., Kato, J., Sakurada, N., Yoshida, T., and Hayashibara, Y. (2006). RoboCupRescue 2006 - robot league team toin pelican (japan). In *RoboCup 2006 (CDROM Proceedings), Team Description Paper, Rescue Robot League*, Bremen, Germany.

[Krotkov and Hoffman, 1994] Krotkov, E. and Hoffman, R. (1994). Terrain mapping for a walking planetary rover. *IEEE Transactions on Robotics and Automation*, 10:728–739.

[Kuhn, 1955] Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quaterly*, 2:83–97.

[Kurth et al., 2003] Kurth, D., Kantor, G., and Singh, S. (2003). Experimental results in range-only localization with radio. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, volume 1, pages 974–979, Las Vegas, USA.

[Kuwata et al., 2006] Kuwata, Y., Takahashi, T., Ito, N., and Takeuchi, I. (2006). Design of human-in-the-loop agent simulation for disaster simulation systems,. In *Proc. of the Third International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster (SRMED)*, pages 9–14.

[Ladetto et al., 2001] Ladetto, Q., Gabaglio, V., and Merminod, B. (2001). Combining gyroscopes, magnetic compass and GPS for pedestrian navigation. In *International Symposium on Kinematic Systems in Geody, Geomatics and Navigation (KIS 2001)*, pages 205–212, Banff, Canada.

[Ladetto and Merminod, 2002a] Ladetto, Q. and Merminod, B. (2002a). Digital magnetic compass and gyroscope integration for pedestrian navigation. In *9th Saint Petersburg International Conference on Integrated Navigation Systems*, Russia.

[Ladetto and Merminod, 2002b] Ladetto, Q. and Merminod, B. (2002b). In step with INS: Navigation for the blind, tracking emergency crews. *GPSWorld*, 13(10):30–38.

[Ladetto et al., 2000] Ladetto, Q., Merminod, B., Terrirt, P., and Schutz, Y. (2000). On foot navigation: When GPS alone is not enough. *Journal of Navigation*, 53(02):279–285.

[Lange, 2005] Lange, E. (2005). Sicherheits-innovationen: Banknoten der zukunft. *Die Bank - Zeitschrift für Bankpolitik und Praxis*, 9.

[Lawo et al., 2006] Lawo, M., Witt, H., Kenn, H., Nicolai, T., and Leibrand, R. (2006). A glove for seamless computer interaction - understand the WINSPECT. In Kenn, H., Glotzbach, U., and Herzog, O., editors, *The Smart Glove Workshop*, number 33.

[Lechner et al., 2001] Lechner, W., Baumann, S., and Legat, K. (2001). Pedestrian navigation - bases for the next mass market in mobile positioning. In *Proceedings of Locellus 2001*, pages 79–90, Munich, Germany.

[Liao et al., 2007] Liao, L., Patterson, D. J., Fox, D., and Kautz, H. A. (2007). Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311–331.

[Lienhart and Maydt, 2002] Lienhart, R. and Maydt, J. (2002). An extended set of haar-like features for rapid object detection. *Proc. of the IEEE Int. Conf. on Image Processing (ICIP)*, 1:900–903.

[Logitech, 2006] Logitech (2006). *Logitech QuickCam Pro 4000*. `http://www.logitech.com/index.cfm/products/details/US/EN,CRID=2204,CONTENTID=5042`.

[Lu and Milios, 1994] Lu, F. and Milios, E. (1994). Robot pose estimation in unknown environments by matching 2D range scans. In *IEEE Computer Society Conference on Vision and Pattern Recognitionv (CVPR)*, pages 935–938.

[Lu and Milios, 1997] Lu, F. and Milios, E. (1997). Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349.

[Maybeck, 1979] Maybeck, P. S. (1979). *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Academic Press.

[Maybeck, 1990] Maybeck, P. S. (1990). *The Kalman filter: An introduction to concepts*. Cox and Wilfong.

[Meier et al., 2005] Meier, D., Stachniss, C., and Burgard, W. (2005). Coordinating multiple robots during exploration under communication with limited bandwidth. In *Proc. of the European Conference on Mobile Robots (ECMR)*, pages 26–31, Ancona, Italy.

[Messina et al., 2005] Messina, E., Jacoff, A., Scholtz, J., Schlenoff, C., Huang, H., Lytle, A., and Blitch, J. (2005). Statement of requirements for urban search and rescue robot performance standards. Technical report, Department of Homeland Security, Science and Technology Directorate and National Institute of Standards and Technology.

[MicroSensys, 2007] MicroSensys (2007). Homepage. `http://www.microsensys.de/`.

[Milella and Siegwart, 2006] Milella, A. and Siegwart, R. (2006). Stereo-based ego-motion estimation using pixel tracking and iterative closest point. In *Proc. of the IEEE Int. Conf. on Computer Vision Systems (ICVS)*, page 21, Washington, DC, USA. IEEE Computer Society.

[Miller et al., 2006] Miller, L., Wilson, P. F., Bryner, N. P., Francis, Guerrieri, J. R., Stroup, D. W., and Klein-Berndt, L. (2006). RFID-assisted indoor localization and communication for first responders. In *Proc. of the Int. Symposium on Advanced Radio Technologies*.

[Montemerlo et al., 2002] Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2002). FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, Menlo Park, CA, USA. American Association for Artificial Intelligence.

[Moravec, 1988] Moravec, H. (1988). Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2):61–74.

[Moravec and Elfes, 1985] Moravec, H. and Elfes, A. (1985). High resolution maps from wide angle sonar. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 116–121.

[Moravec, 1996] Moravec, H. P. (1996). Robot spatial perception by stereoscopic vision and 3D evidence grids. Technical Report CMU-RI-TR-96-34, Carnegie Mellon University.

[Murphy, 2000] Murphy, K. (2000). Bayesian map learning in dynamic environments. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press.

[Murphy, 2004] Murphy, R. (2004). Human-robot interaction in rescue robotics. *Systems, Man and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 34(2):138–153.

[Murphy et al., 2000] Murphy, R., Casper, J., Micire, M., and Hyams, J. (2000). Mixed-initiative control of multiple heterogeneous robots for USAR. Technical Report CRASAR-TR2000-11, Center for Robot Assisted Search & Rescue, University of South Florida, Tampa, FL.

[Neira and Tard, 2001] Neira, J. and Tard, J. (2001). Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897.

[Nettleton et al., 2003] Nettleton, E., Thrun, S., and Durrant-Whyte, H. (2003). Decentralised SLAM with low-bandwidth communication for teams of airborne vehicles. In *In Proc. of the Int. Conf. on Field and Service Robotics*, Lake Yamanaka, Japan.

[Nevatia et al., 2006] Nevatia, Y., Mahmudi, M., Markov, S., Rathnam, R., Stoyanov, T., , and Carpin, S. (2006). VIRTUAL-IUB: the 2006 IUB virtual robots team. In *Robocup 2006: Robot Soccer World Cup X*, Bremen, Germany.

[Newman et al., 2003] Newman, P. M., Leonard, J. J., and Rikoski, R. R. (2003). Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar. In *Int. Symposium on Robotics Research*, Sienna, Italy.

[Nguyen et al., 2004] Nguyen, H. M., Takamori, T., Kobayashi, S., Takashima, S., and Ikeuchi, A. (2004). Development of carbon dioxide sensing system for searching victims in large scale disasters. In *SICE Annual Conference*, volume 2, pages 1358–1361.

[Nicolai et al., 2005] Nicolai, T., Sindt, T., Kenn, H., and Witt, H. (2005). Case study of wearable computing for aircraft maintenance. In Herzog, O., Lawo, M., Lukowicz, P., and Randall, J., editors, *2nd International Forum on Applied Wearable Computing (IFAWC'02)*, pages 97–110. VDE Verlag.

[Nister et al., 2004] Nister, D., Naroditsky, O., and Bergen, J. (2004). Visual odometry. In *IEEE Computer Society Conference on Vision and Pattern Recognitionv (CVPR)*, volume 1, pages 652–659.

[Noda and Meguro, 2006] Noda, I. and Meguro, J. (2006). Data representation for information sharing and integration among rescue robot and simulation. In *International Joint Conference SICE-ICASE*, pages 3449–3454, Busan, Korea.

[Nourbakhsh et al., 2005] Nourbakhsh, I., Lewis, M., Sycara, K., Koes, M., Yong, M., and Burion, S. (2005). Human-robot teaming for search and rescue. *IEEE Pervasive Computing*, 4(1):72–78.

[Nüchter et al., 2003] Nüchter, A., Surmann, H., Lingemann, K., and Hertzberg, J. (2003). Semantic scene analysis of scanned 3D indoor environments. In *Proceedings of the 8th International Fall Workshop Vision, Modeling, and Visualization (VMV)*, pages 215–222, Munich, Germany. IOS Press.

[Nüssle et al., 2004] Nüssle, T. A., Kleiner, A., and Brenner, M. (2004). Approaching urban disaster reality: The ResQ firesimulator. In Nardi, D., Riedmiller, M., Sammut, C., and Santos-Victor, J., editors, *RoboCup 2004: Robot Soccer World Cup VIII*, volume 3276 of *Lecture Notes in Computer Science*, pages 474–482. Springer.

[Oh et al., 2006] Oh, J., Hwang, J., and Smith, S. (2006). Agent technologies for post-disaster urban planning. In Jennings, N., Tambe, M., Ishida, T., and Ramchurn, S., editors, *First International Workshop on Agent Technology for Disaster Management at AAMAS06*, pages 24–31, Hakodate, Japan. AAMAS Press.

[Ojeda and Borenstein, 2004] Ojeda, L. and Borenstein, J. (2004). Methods for the reduction of odometry errors in over-constrained mobile robots. *Autonomous Robots*, 16:273–286.

[Omologo and Svaizer, 1996] Omologo, M. and Svaizer, P. (1996). Acoustic source localization in noisy and reverberant environment using csp analysis. In *Proc. of*

the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), pages 921–924.

[Perkins, 2002] Perkins, C. (2002). IP mobility support for IPv4. RFC.

[Pfaff et al., 2007a] Pfaff, P., Triebel, R., and Burgard, W. (2007a). An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *The International Journal of Robotics Research - Special Issue FSR*, 26(2):217–230.

[Pfaff et al., 2007b] Pfaff, P., Triebel, R., Stachniss, C., Lamon, P., Burgard, W., and Siegwart, R. (2007b). Towards mapping of cities. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Rome, Italy.

[Pfingsthorn et al., 2006] Pfingsthorn, M., Slamet, B., Visser, A., and Vlassis, N. (2006). UvA rescue team 2006 RoboCup Rescue - simulation league. In *Robocup 2006: Robot Soccer World Cup X*, Bremen, Germany.

[Pratt et al., 2006] Pratt, K., Murphy, R., Stover, S., and Griffin, C. (2006). Requirements for semi-autonomous flight in miniature UAVs for structural inspection. In *Submitted to AUVSI Unmanned Systems North America*.

[Quinlan, 2003] Quinlan, J. R. (2003). Induction of decision trees. *Machine Learning*, 1(1):81–106.

[Ramos et al., 2007] Ramos, F. T., Nieto, J., and Durrant-Whyte, H. (2007). Recognising and modelling landmarks to close loops in outdoor SLAM. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2036–2041.

[Reece and Roberts, 2005] Reece, S. and Roberts, S. (2005). Robust, low-bandwidth, multi-vehicle mapping. In *Eighth IEEE Int. Conf. on Information Fusion*, Philadelphia, Penn.

[Reindl et al., 2001] Reindl, L. M., Pohl, A., Scholl, G., and Weigel, R. (2001). Saw-based radio sensor systems. *IEEE Sensors Journal*, 1:69–78.

[Research, 2007] Research, C. T. (2007). `www.ctr.at`.

[ResQ Freiburg, 2004] ResQ Freiburg (2004). Source code. Available on: `http://gkiweb.informatik.uni-freiburg.de/~rescue/sim04/source/resq.tgz`. release September, 2004.

[Retscher and Kealy, 2006] Retscher, G. and Kealy, A. (2006). Ubiquitous positioning technologies for modern intelligent navigation systems. *The Journal of Navigation*, 59:91–103.

[Russell and Norvig, 2003] Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education.

[Schurr et al., 2005] Schurr, N., Marecki, J., Scerri, P., Lewi, J. P., and Tambe, M. (2005). *Programming Multiagent Systems*, chapter The DEFACTO System: Coordinating Human-Agent Teams for the Future of Disaster Response, page 296. Springer.

[Seidel and Rapport, 1992] Seidel, S. Y. and Rapport, T. S. (1992). 914 MHz path loss prediction model for indoor wireless communications in multi-floored buildings. *IEEE Trans. on Antennas and Propagation*, 40(2):207–217.

[Sheh, 2006] Sheh, R. (2006). The Redback: A low-cost advanced mobility robot for education and research. In *Proc. of the IEEE Int. Workshop on Safty, Security and Rescue Robotics (SSRR)*.

[Shimony, 1994] Shimony, S. E. (1994). Finding maps for belief networks is np-hard. *Artificial Intelligence*, 68:399–410.

[Sidenbladh, 2004] Sidenbladh, H. (2004). Detecting human motion with support vector machines. In *Proc. of the Int. Conf. on Pattern Recognition (ICPR)*, volume 2, pages 188–191.

[Siebra and Tate, 2005] Siebra, C. and Tate, A. (2005). Integrating collaboration and activity-oriented planning for coalition operations support. In *In Proceedings of the 9th International Symposium on RoboCup*, Osaka, Japan.

[Smith et al., 1988] Smith, R., Self, M., and Cheeseman, P. (1988). Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, 1:167–193.

[Smith and Cheeseman, 1986] Smith, R. C. and Cheeseman, P. (1986). On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4):56–68.

[Snyder, 1987] Snyder, J. P. (1987). *Map Projections - A Working Manual. U.S. Geological Survey Professional Paper 1395*. United States Government Printing Office, Washington, D.C.

[Stachniss et al., 2004] Stachniss, C., Haehnel, D., and Burgard, W. (2004). Exploration with active loop-closing for FastSLAM. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, pages 1505–1510, Sendai, Japan.

[Starner, 2001a] Starner, T. (2001a). The challenges of wearable computing: Part 1. *IEEE Micro*, 21(4):44–52.

[Starner, 2001b] Starner, T. (2001b). The challenges of wearable computing: Part 2. *IEEE Micro*, 21(4):54–67.

[Surmann et al., 2003] Surmann, H., Nüchter, A., and Hertzberg, J. (2003). An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Journal Robotics and Autonomous Systems*, 45(3–4):181–198.

[Surmann et al., 2004] Surmann, H., Worst, R., Hennig, M., Lingemann, K., Nüchter, A., Pervoelz, K., Tiruchinapalli, K. R., Christaller, T., and Hertzberg, J. (2004). RoboCupRescue - robot league team KURT3D, Germany, team description paper, rescue robot league competition. In *RoboCup 2004: Robot Soccer World Cup VIII*, Portugal, July.

[Sutton and Barto, 1998] Sutton, R. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press.

[Svaizer et al., 1997] Svaizer, P., Matassoni, M., and Omologo, M. (1997). Acoustic source location in a three-dimensional space using crosspower spectrum phase. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, page 231, Washington, DC, USA. IEEE Computer Society.

[Svennebring and Koenig, 2004] Svennebring, J. and Koenig, S. (2004). Building terrain-covering ant robots: A feasibility study. *Auton. Robots*, 16(3):313–332.

[Tadokoro, 2005] Tadokoro, S. (2005). Special project on development of advanced robots for disaster response (DDT project). In *IEEE Workshop on Advance Robotics and its Social Impacts (ARSO'05)*.

[Tadokoro et al., 2000] Tadokoro, S., Kitano, H., Takahashi, T., Noda, I., Matsubara, H., Shinjou, A., Koto, T., Takeuchi, I., Takahashi, H., Matsuno, F., Hatayama, M., Nobe, J., and Shimada, S. (2000). The RoboCup-Rescue project: A robotic approach to the disaster mitigation problem. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*.

[Tadokoro et al., 2003] Tadokoro, S., Matsuno, F., Onosato, M., and Asama, H. (2003). Japan national special project for earthquake disaster mitigation in urban areas. In *First International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*.

[Takahashi, 2006] Takahashi, T. (2006). Requirements to agent-based disaster simulations from local government usage. In Jennings, N., Tambe, M., Ishida, T., and Ramchurn, S., editors, *First International Workshop on Agent Technology for Disaster Management at AAMAS06*, pages 78–84, Hakodate, Japan. AAMAS Press.

[Takashi and Hiroshi, 2003] Takashi, N. and Hiroshi, U. (2003). Detection of carbon dioxide in urban search and rescue as an application for miniaturized mass spectrometerst. *Journal of the Mass Spectrometry Society of Japan*, 51(1):264–266.

[Taskar et al., 2004] Taskar, B., Chatalbashev, V., and Koller, D. (2004). Learning associative markov networks. In *Proc. of the Int. Conf. on Machine Learning (ICML)*.

[Taskar et al., 2003] Taskar, B., Guestrin, C., and Koller, D. (2003). Max-margin markov networks. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada.

[Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. MIT Press.

[Thrun et al., 2004] Thrun, S., Liu, Y., Koller, D., Ng, A., Ghahramani, Z., and Durrant-Whyte, H. (2004). Simultaneous mapping and localization with sparse extended information filters. *International Journal of Robotics Research*, 23(7–8):693–716.

[Thrun et al., 2006] Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Stanley, the robot that won the DARPA grand challenge. *Journal of Field Robotics*, 23(9):655–656.

[Tomasi and Kanade, 1991] Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University.

[TopoGrafix, 2004] TopoGrafix (2004). GPX - the GPS exchange format. Available on: `http://www.topografix.com/gpx.asp`. release August, 9th 2004.

[Tsai, 1986] Tsai, R. Y. (1986). An efficient and accurate camera calibration technique for 3D machine vision. In *IEEE Computer Society Conference on Vision and Pattern Recognitionv (CVPR)*, pages 364–374.

[U and Reed., 2006] U, V. R. and Reed., N. E. (2006). Enhancing agent capabilities in a large rescue simulation system. In Jennings, N., Tambe, M., Ishida, T., and Ramchurn, S., editors, *First International Workshop on Agent Technology for Disaster Management at AAMAS06*, pages 71–77, Hakodate, Japan. AAMAS Press.

[Vapnik, 1995] Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

[Vectronix, 2007] Vectronix (2007). Personal dead reckoning module. Available on: http://www.vectronix.ch/. Referenced 2007.

[Viola and Jones, 2001] Viola, P. and Jones, M. J. (2001). Rapid object detection using a boosted cascade of simple features. In *IEEE Computer Society Conference on Vision and Pattern Recognitionv (CVPR)*.

[Viola et al., 2003] Viola, P., Jones, M. J., and Snow, D. (2003). Detecting pedestrians using patterns of motion and appearance. In *IEEE International Conference on Computer Vision*, volume 2.

[Wang and Lewis, 2007] Wang, J. and Lewis, M. (2007). Human control of cooperating robot teams. In *Human-Robot Interaction Conference (HRI)*.

[Weigel et al., 2002] Weigel, T., Gutmann, J.-S., Dietl, M., Kleiner, A., and Nebel, B. (2002). CS Freiburg: Coordinating robots for successful soccer playing. *IEEE Transactions on Robotics and Automation*, 18(5):685–699.

[Wickler et al., 2006] Wickler, G., Tate, A., and Potter, S. (2006). Using the <I-N-C-A> constraint model as a shared representation of intentions for emergency response coordination. In Jennings, N., Tambe, M., Ishida, T., and Ramchurn, S., editors, *First International Workshop on Agent Technology for Disaster Management at AAMAS06*, pages 2–9, Hakodate, Japan. AAMAS Press.

[Witt et al., 2006] Witt, H., Nicolai, T., and Kenn, H. (2006). Designing a wearable user interface for hands-free interaction and maintenance applications. In *Fourth Annual IEEE International Conference on Pervasive Computer and Communication (PerCom)*.

[Wolf et al., 2005] Wolf, D., Sukhatme, G., Fox, D., and Burgard, W. (2005). Autonomous terrain mapping and classification using hidden markov models. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2026–2031.

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

[Wu and Yu, 2006] Wu, Y. and Yu, T. (2006). A field model for human detection and tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5):753–765.

[Yamauchi, 1997] Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA '97)*.

[Ye and Borenstein, 2003] Ye, C. and Borenstein, J. (2003). A new terrain mapping method for mobile robots obstacle negotiation. In Gerhart, G. R., Shoemaker, C. M., and Gage, D. W., editors, *Unmanned Ground Vehicle Technology V.*, volume 5083, pages 52–62.

[Zalud, 2004] Zalud, L. (2004). Orpheus - universal reconnaissance teleoperated robot. In *RoboCup 2004: Robot Soccer World Cup VIII*, pages 491–498.

[Zhu et al., 2006] Zhu, Z., Oskiper, T., Naroditsky, O., Samarasekera, S., Sawhney, H. S., and Kumar, R. (2006). An improved stereo-based visual odometry system. In *Proc. of the IEEE Int. Workshop on Safty, Security and Rescue Robotics (SSRR)*, Gaithersburg, Maryland, USA.

[Ziparo and Iocchi, 2006] Ziparo, V. and Iocchi, L. (2006). Petri net plans. In *Proc. of ATPN/ACSD Fourth International Workshop on Modelling of Objects, Components, and Agents*.

[Ziparo et al., 2007a] Ziparo, V., Kleiner, A., Marchetti, L., Farinelli, A., and Nardi, D. (2007a). Cooperative exploration for USAR robots with indirect communication. In *Proc.of 6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV '07)*.

[Ziparo et al., 2007b] Ziparo, V., Kleiner, A., Nebel, B., and Nardi, D. (2007b). RFID-based exploration for large robot teams. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 4606–4613.

[Zlot et al., 2002] Zlot, R., Stentz, A., Dias, M., and Thayer, S. (2002). Multi-robot exploration controlled by a market economy. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 3016–3023.