KUNGL
TEKNISKA
HÖGSKOLAN

Simultaneous Localization and Mapping for Navigation in Realistic Environments

Guido Zunino

TRITA-NA-0203

Licentiate Thesis
Royal Institute of Technology
Numerical Analysis and Computer Science

Akademisk avhandling som med tillstånd av Kungl. Tekniska Högskolan framlägges till offentlig granskning för teknisk licentiatitexamen onsdag den 20 february 2002, kl 14.00 is sal E3, Lindstedtsvägen 3, Kungl. Tekniska Högskolan, Stockholm.

ISBN 91-7283-246-0, TRITA-NA-0203, ISSN 0348-2952
ISRN KTH/NA/R-02/03-SE

KTH, Stockholm 2002

Abstract

Navigating autonomously in a domestic environment is a problem that has attracted a great deal of interest in mobile robotics. A robotic system that operates in ordinary furnished rooms without the need of an engineered environment has many different applications such as service, cleaning and surveillance tasks or simply entertainment. Robotic systems that use artificial landmarks or pre-stored maps of the environment are available today. However, these systems are not very flexible. The user must in fact supply a map of the environment, which can be interpreted by the system. This thesis deals with the problem of Simultaneous Localization and Mapping (SLAM). The mobile robot builds a map of an unexplored environment while simultaneously using this map to localize itself. The feature based approach used in this thesis utilizes the Extended Kalman Filter (EKF) machinery to estimate the pose of the robot and the location of the features. This approach is referred to as *stochastic mapping*. Point features in the environment are robustly extracted from sonar data using triangulation techniques.

In addition, this thesis explores a method for recovering from the most common mode of failure of the *stochastic mapping* approach. This method allows the EKF algorithm to continue in a consistent manner after a failure has been detected.

Finally, the thesis presents a method for achieving more accurate navigation by using the architectural properties of most domestic environments. This method drastically improves navigation, when the *stochastic mapping* algorithm can not be used due to poor quality sensor data.

All the algorithms presented in this thesis have been tested and verified in real world experiments.

Keywords : mobile robots, sensor fusions, sonars, odometry, SLAM, Kalman filter, mapping, localization, navigation.

Acknowledgments

First of all I would like to thank my supervisor Professor *Henrik Christensen* who without his guidance and support I would not have be able to have done this research. I would also like to thank him for introducing me to mobile robotics and for taking the time to discuss problems and new ideas.
I wish also to express my sincere appreciation to the following people:

Olle Wijk and *Patric Jensfelt* for their support and knowledge in the area as well as the beneficial discussions about robotics. Their research has also been valuable to me in carrying out this thesis.

Philipp Althaus for supplying me with good friendship and the good advice he has given me during this time. He has been good company during our long coffee breaks.

Frank Hoffmann, *Lars Petterson* and *Marco Zucchelli* for being great colleagues and friends and for breaking up the long days with unmemorable lunch breaks talking about football.

Danica Kragić, *Mattias Lindström*, *Anders Orebäck*, *Girma Siele*, *Morten Stanberg John Folkesson*, *Jan-Olof Eklundh*, *Jeanna Al-Ayoubi*, *Emma Gniuli* and all the other present and former CAS and CVAP people who have contributed to the pleasant atmosphere here.

Simoncelli Maurizio who originally began this adventure in Sweden with me. Without Maurizio I would not have come to Sweden and I would have not had this wonderful experience.

This work has been carried out at the Centre for Autonomous Systems and sponsored by a NUTEK project within the Complex Systems programme. The financial support is gratefully acknowledged.

I would also like to express my gratitude to Electrolux for providing the department with the robotic platform which I used for my research.

Contents

List of Figures

Acronyms and Notations

CAS	Centre for Autonomous Systems
TOF	Time Of Flight
TBF	Triangulation Based Fusion
EKF	Extended Kalman Filter
SLAM	Simultaneous Localization and Mapping
RWHT	Range Weighted Hough Transform

c_s	Speed of sound in air ($\approx$340 m/s)
c	Speed of light
T	Triangulation point with position (x_T, y_T)
P_T	Covariance matrix for a triangulation point T
n_t	Number of triangulations supporting a triangulation point T
x_{s_i}	x position of sensor i
y_{s_i}	y position of sensor i
γ_i	Orientation of sensor i
δ	Opening angle of the sonar sensor
r	Reading form the sonar sensor
$\mathbf{x}$	Estimate of the augmented system state vector
$\mathbf{P}$	Covariance matrix of the system state estimate
x_r	Estimate of the robot pose $(x_r, y_r, \theta_r)^T$
x_i	Estimate of the i feature location $(x_i, y_i)^T$
P^{rr}	Covariance matrix of the robot pose
P^{ii}	Covariance matrix of the feature i
P^{ri}	Robot to feature i correlation matrix
P^{ij}	Feature i to feature j cross-correlation matrix
ρ	Distance from the feature of the measurement model
θ	Angle from the feature of the measurement model
$\mathbf{R}$	Covariance of the measurement
β	Number of unmatched measurement in the feature initiation process
ℓ	Length of the line feature
ξ	Angle of the line feature
AngleRef	Angle of the reference line for the *Geometric Constraint* algorithm
$U(a, b)$	Uniform distribution in the interval $[a, b]$
$N(m, \sigma)$	Normal distribution, with mean m and standard deviation σ

Chapter 1

Introduction

Navigation in a realistic environment is a fundamental requirement for obtaining an autonomous mobile robot. The critical importance and the theoretical challenges of such a problem has driven the robotics research community to an extensive study, however, many fundamental questions remain unanswered.

The navigation problem according, to Leonard and Durrandt-Whyte (Leonard and Durrant-Whyte, 1992), can be addressed as answering the questions: "Where am I?", "Where do I want to go?" and "How do I get there?". In order to give an answer to any of these questions, the platform needs to know what coordinate system and what environment these questions refer to. A coordinate system and the set of information about of the location of features in the environment is called a *map*. Another important question that needs to be answered is: "What is my map?". This question is of fundamental importance, as a matter of fact, knowing that the robot is at the point P at the position (1125,120) gives no information unless the robot knows which map it is referring these coordinates to. Answering the second and the third questions is of little value unless the system can relate these answers to a specific map. Since the necessity of being able to answer the first and the last question is of crucial importance, the problem of navigation is often addressed as the problem of mapping and localization. The work in this thesis is dedicated to the problem of navigation, that is localization and mapping, in domestic settings. The question "Where do I want to go?" is a problem which depends strictly on the task of the robot. If the task is simply to reach a determined goal point, this mission goal can be defined a priori.

However, if the task is more complex, such as complete coverage of the free space surface of the environment for a cleaning robot, it is necessary to have a more complex task planner. The solution of the question "How do I get there?" is a path-planning problem, which has been studied extensively in the literature.

Implementation of navigation system that uses artificial landmarks or *a priori* known maps of the environment, and accurate sensor systems to get precise measurements of the landmarks or map features, is straightforward for today's robots. Similarly, the task of building a map of the environment given the exact position of the robot is largely a solved problem. However, it is much harder to solve the complete problem simultaneously, enabling a mobile robot to build a map of an unexplored environment while simultaneously using this map to localize itself. The problem is known as Simultaneous Localization and Mapping (SLAM). The approaches to SLAM can be categorized into three main categories: (1) "grid-based", (2) "feature-based", and (3) "topological". The state of the art of these approaches are summarized in the following section.

1.1 Related work

The conceptually simplest approach to the SLAM problem is the *grid-based* approach. The pioneers of this approach are Moravec and Elfes (Moravec and Elfes, 1985). Their work has been extended in (Elfes, 1987; Moravec, 1983) and utilized by other researchers (Stewart, 1996; Yamauchi and Langley, 1997). This method is also referred as a "metric" approach by Thrun *et al.* (Thrun et al., 1998a) (Thrun et al., 1998b). In this approach the environment is divided in into a grid of cells of fixed size. Each cell contains the probability of being occupied by an object. Therefore a cell known to be occupied would be assigned the value of 1, while for a cell that is free from any object will be assigned the value of 0. This method generates a map of the environment referred to a certainty grid. To localize, after having created a new certainty grid of its local environment, the robot performs a search through a set of previously acquired global certainty grids. The pose of the robot which maximizes the correlation between the new and the old map is defined as the new estimate of the pose. Once the localization step is performed the new certainty grid is merged with the old one increasing the accuracy of the certainty grid. The major advantages of the *grid-based* approaches to SLAM are that they are intuitively simple, easy to implement, and

naturally extendible to higher dimensions (Stewart, 1996). Successful experimental works have been presented using *grid-based* methods (Burgard et al., 1999). The main disadvantages of *grid-based* approaches are the high computational cost of localization and the high storage requirement. Furthermore, these approaches have a weak theoretical foundation as data are smeared, thereby removing critical information or adding inconsistent information. The quality and the robustness of *grid-based* approaches are critically dependent on the sensor system used and how the sensory information is used to update the uncertainty grid.

Feature-based approaches to SLAM use easily identifiable elements in the environment, such as planes, corners and edges in an indoor environment, and build an internal representation (map) with the location of these landmarks (Chong and Kleeman, 1997; Feder et al., 1988) (Moutarlier and Chatila, 1989) (Smith and Cheesman, 1987). The first solution to this problem was provided by Smith, Self and Cheesman (Smith and Cheesman, 1987) who developed an Extended Kalman Filter (EKF) approach for building a "*stochastic map*" of spatial relationships. The *stochastic map* is a special way of organizing the states in an EKF (Bar-Shalom and Fortmann, 1988) (Gelb, 1973) in order to build and update a feature map of the environment from sensor measurements and dead reckoning of the robot. Localization is performed simultaneously by updating the pose of the vehicle. That is, when features are re-observed the measurements are used to update both the pose of the robot and the location of all the features in the map. The approach relies on the assumption of Gaussian distribution of the uncertainties. *Stochastic mapping* has the advantage of providing metrically accurate navigation robustly and the possibility of incorporating false returns, drop-out and data association ambiguities. Crucial requirements of *stochastic mapping* are the automatic extraction of featured from the environment, and a reliable data association. The main disadvantage of such an approach is the computational complexity of the algorithm when the number of features becomes large (Mendle, 1985). This problem, also referred as the *map scaling problem*, is due to the large number of vehicle-to-feature and feature-to-feature correlations that must be maintained as the size of the operational environment increases. Many researchers have addressed the problem of *stochastic mapping*, Moutarlier and Chatila (Moutarlier and Chatila, 1989) have implemented a framework similar to the one presented by Smith *et* al. (Smith and Cheesman, 1987) using laser range data. Methods for reducing the computational requirements have been proposed. Failures of strategies which ignore the correlations have been

demonstrated by Uhlmann *et* al. (Uhlmann et al., 1997) and Castellanos *et* al. (Castellanos et al., 1997). Leonard and Feder (Feder, 1999) developed a method for splitting the map into multiple globally-referenced submaps keeping the complexity bounded, Durrant-Whyte and Dissanayake (Durrant-Whyte and Dissanayake, 1999) presented a method for choosing the best features in the environment to best maintain the performance of the SLAM algorithm.

The *topological* approach to the SLAM problem attempts to create a graph-like description of the environment rather than a precise accurate metric map. In the *topological* description nodes correspond to "significant places" in the environment which are easy to distinguish, and arcs connecting these nodes correspond to sequences of actions that connect neighboring places. This approach has been proposed by Brooks (Brooks, 1989), Mataric (Mataric, 1990; Mataric, 1992; Mataric, 1997) and other researchers (Duckett and Nehmzow, 2001; Thau, 1997). The motivation of this approach was the belief that humans and animals do not produce accurate metric maps of the environments they operate in. Often these approaches are coupled with behavior-based architectures (Arkin, 1998) and utilize reactive rules. These methods are appropriate for navigation in simple environments, but they have not been successfully applied to complex and large environments. Furthermore, for many operations, metric information are essential for achieving the goal of the mission. Other methods use *topological* maps augmented with metric information (Jensfelt, 2001), in this approach a multiple layer hierarchy is considered, as the the environment becomes large the number of levels in the hierarchy can be increased. The lowest level could be rooms, the second floors and so on. Each submap maintains full correlation information and correlations between submaps are given through the use of a common coordinate system.

Having outlined the main approaches to mobile robot navigation, we now turn to the particular challenge of domestic navigation.

1.2 Domestic navigation

Autonomous mobile robotics is a fascinating research topic, for many reasons. The change from a computer on wheels that is merely able to sense some physical properties of the environment through its sensors into an intelligent agent, able to identify features, to detect patterns and regularities, to learn from experience, to localize, build maps and to nav-

igate requires the simultaneous application of many research disciplines. There are commercial applications of mobile robots. Transportation, surveillance, inspection, cleaning or household robots are just some examples. However, autonomous mobile robots have not yet made much impact upon industrial and domestic applications, mainly due to the lack of robust, reliable and flexible navigation and behavior mechanisms for operating in unmodified, semi-structured environments. This thesis is mainly focusing in domestic application, the mobile robot is required to perform different tasks guaranteeing a high degree of robustness.

Installing markers such as beacons, visual patterns or induction loops (guiding wires buried in the ground) would make this problem easy, but it is expensive, inflexible and sometimes outright impossible. Selling a domestic service robot with the requirement for the user to place artificial markers in his/her living-room or bedroom in order to operate it properly wouldn't be a good idea, in addition it is, in general, impossible that a map of the environment can be provided. On the other hand, equipping the robot with the most accurate and expensive sensors available, to achieve high precision performance, would make the product too expensive for a wide market area. Therefore, the robot should be able to operate without any kind of engineering of the environment itself using a minimum set of sensors which can guarantee robustness and an affordable price. It is of interest to investigate methods that allow automatic acquisition of maps of the environment and simultaneous use of these maps for robust localization and navigation. The test environment used for the evaluation of the methods proposed in the thesis is a 5×9 meters living-room set up at the Centre for Autonomous Systems, see figure 1.1.

1.3 Sensors and features

Most mobile robots for indoor applications have wheels. The movement of the robot, is usually measured through cheap sensors such as optical encoders mounted on the wheels and shaft to measure the number of rotations. By knowing a set of parameters, such as wheel radii and distance between the wheels, it is possible to compute the robot movement based on the information given by the encoders. This kind of information is referred to as *odometry*. The main disadvantage of odometry information is that, it is subjected to drift. This drift comes from two kinds of errors: *systematic errors* which are mainly due to imperfection in the structure

Figure 1.1: Regular living-room at the Centre for Autonomous systems.

of the robot, and *non-systematic errors* due to external causes such as wheel slippage when the robot is moving over a carpet, human intervention and so on. While *systematic errors* can be compensated for to some degree (Borenstein and Feng, 1996), *non-systematic errors* are harder to deal with since their appearance is of more random nature. From a localization perspective, odometry drift is highly undesirable and may prevent an autonomous robot from accomplishing its task. The use of more advanced internal sensors such as gyros or accelerometers can only help to reduce the odometry drift, bun not eliminate it. The only way to correct it is through the use of *exteroceptive sensors*. Differently from *interoceptive sensors*, *exteroceptive sensors* perform measurements corresponding to entities external to the robot. Using such measurements it is possible to compensate for the odometry drift. This thesis deals with *feature-based* navigation. The ease of extracting different kinds of features depends on the sensor used. An overview is therefore given of the sensory modalities available today and the kind of features detectable with such sensors.

1.3.1 Laser

The laser scanner is the sensor which is widely expanding in robotics. The dominating techniques for laser based range measurement are *time of flight* (TOF) techniques and *phase-shift* techniques. In a TOF system a short laser pulse is sent out and the time until it returns is measured. The distance from the object hit is given by $D = \frac{1}{2}cT$, where c is the speed of light and T is the round trip time. In *phase-shift* systems a continuous wave is transmitted. The idea is to compare the phase of the returned signal with a reference signal generated by the same source. Using doppler shift, the velocity of the target can be measured in addition to the distance to it. The laser sensor has been used by many researchers (Chatila, 1985; Hoppen et al., 1990; Buchberger et al., 1993; Borthwick et al., 1993; Forsberg et al., 1993; Weiss and von Puttkamer, 1995; Guttman et al., 1998; Fox et al., 1999; Jensfelt, 1999; Jensfelt, 2001) A laser scanner present many advantages:

- It is fast, i.e. the measurement, for domestic robotic tasks, can be considered as instantaneous. This means that one does not have to think about compensating for the motion of the platform.
- The range accuracy is fairly good. The new generation of laser scanners from SICK Electro-Optics has an accuracy better than 10

mm.

- The angular resolution of 0.5° or 0.25° depending on operation mode is far better than for the sonar.
- The data from the laser scanner can be interpreted directly as the range to an obstacle in a certain direction. This can be said also for sonar sensors, disregarding specular reflection and cross-talk, but not for a camera image, which takes much more effort to interpret.

Among the disadvantages with the laser scanners are:

- The sensor provides range information limited to a plane. If some obstacles lie above or under such a plane it can not be detected by the sensor.
- The sensor is still very expensive.
- Some materials appear as transparent to the laser, such as glass.

The laser scanner is the best sensor for extracting planar features, such as walls, due to the dense range data provided. Using the Hough transform (Hough, 1962)or the Range Weighted Hough Transform (RWHT) (Forsberg et al., 1993; Illingworth and Kittler, 1988) it is possible to robustly extract flat surface element from the environment. Other features easily detectable with laser are door posts (Jensfelt, 2001).

1.3.2 Sonar

Sonar sensors are widely used in robotics due to their affordable price. Several researcher have used sonar in robotic application, advanced configurations of sonar sensor have been presented (Kleeman, 1992; Kleeman and Kuc, 1994; Akbarally and Kleeman, 1995; Kleeman, 1999; Heale and Kleeman, 2000; Stanley and McKerrow, 1997; McKerrow and Zhu, 1996; Peremans, 1994; Audenaert et al., 1992; Barshan and Kuc, 1991; Barshan and Kuc, 1990; Kuc and Barshan, 1992), nevertheless, the most commonly used sonar device for robotic navigation is the Polaroid 6500. This sensor works as both transmitter and receiver, analogously to laser range finders, it uses the TOF technique. The success of applying sonar data in mobile robotics is quite dependent on the approach to classify the data. The main advantages of sonar sensor are:

- Low cost. The price of a Polaroid 6000 is less than $ 10.

- High range precision. The range estimate is usually within the 1% of the true value.

- They can operate in any brightness condition.

However, there are a number of disadvantages of sonars:

- Weak echoes: This fact becomes appearent when the power of the received signal is low, and it takes a long time for reaching the detection threshold. This results in range estimates that are too long (Leonard and Durrant-Whyte, 1992).

- Multiple reflections: The sound wave may be reflected by more than one object before reaching the receiver, causing false readings.

- Crosstalk: When several sonars send out identical sound waves, it might happen that a receiver receives the sound wave sent by another sensor.

- Bad angular resolution. Due to the wide main lobe (25°) the reflection can originate from anywhere along parts of a spherical surface (Wijk, 2001).

- Limited range sensing. Typically the Polaroid 6500 sensor can measure distances in the interval 0.2-10 m. However for small objects the detection distance is more like 0.2-5 m.

- Speed of sound constraint. Due to the fact that the speed of sound ($c_s \simeq 340$ m/s) is very slow (compared to the speed of light which is used in laser scanners) the fire rate of the sensor is limited.

Because of these problems, considering a single sonar measurement alone, the classification task become very hard. There are, however, techniques where, combining several readings taken during robot motion, make it possible to extract simple geometric features from the environment. A method called *Triangulation Based Fusion* (TBF) (Wijk and Christensen, 1998) allows the extraction of point features from the environment from sonar readings. Also line features can be extracted using sonar data (Althaus et al., 2001) even if the accuracy is not as good as with laser scanners.

1.3.3 Vision

One thing that sonar, laser scanners and other range-finding sensors, lack is the ability to use surface properties to localize and identify objects. Color or grey-scale images allow us to use a wider set of information to identify and localize features in the environment. Different researchers have used vision for mobile robot navigation (Taylor and Kriegman, 1995; Buffa et al., 1992; Murray and Jennings, 1997). The main advantages of vision sensors are:

- Large amount of information.
- Capability of getting 3D information about the environment.
- Cameras are passive sensors, they don't have to emit sound or light pulses as sonar and laser sensors.

The drawbacks are:

- High computing requirement to extract the information from the images.
- Vision is highly influenced by the lighting.
- It is still expensive.

When comparing vision versus range sensing (sonar and laser scanner), the information content is much more well defined in the latter case. It is, therefore, more straightforward to approach the problem of mapping and localization with range sensing than with vision. This thesis is focused on sonar sensors and point features due to the good cost/performance ratio.

1.4 Outline and Contribution

The thesis is divided in 7 chapters. Chapter 2 is a presentation of the robotic platform used in the experiments. Chapter 3 explains the TBF algorithm. The contribution of this chapter is the extension of the TBF algorithm such that it can be used with the Electrolux sonar sensor. Chapter 4 focus on the *stochastic mapping* approach. Chapter 5 presents the *recovery from failure* algorithm, another contribution of the thesis. In chapter 6 the *Geometric Constraint* algorithm is proposed which is the key contribution of the thesis. Chapter 7 present the conclusions of the thesis. Brief reviews of the different chapters follow below.

Chapter 2

Chapter 2 gives an overview on the two robotic platforms used in the experiments of this thesis. The first one is the Electrolux autonomous vacuum cleaner Trilobite. The second platform is the Nomadic SuperScout.

Chapter 3

This chapter presents the models of the two sonar systems used by the test platforms. A description of the TBF algorithm, method used for extracting point features from raw sonar data, is given. Finally the measurement model equations for both sensors are presented.

Chapter 4

This chapter deals with the Simultaneous Localization and Mapping problem. The *stochastic mapping* approach is used for estimating both the robot's pose and the location of the features in the map. Real world experiments, show the performance of the algorithm on the Nomadic SuperScout robot.

Chapter 5

In this chapter a variety of situations in which the *stochastic mapping* algorithm fails are presented. An algorithm for detecting these failures is proposed, and a method for recovering from these failures is developed. Using this approach the *stochastic mapping* algorithm can be restored in a consistent way. Real world experiments demonstrate the validity of the method.

Chapter 6

Chapter six presents a method called *Geometric Constraint* algorithm. Using the knowledge of the robot's behavior and the architectural proprieties of most domestic environments the method can be, successfully used to compensate the odometry drift. A set of experiments show the robustness of this method in a real domestic setting.

Publications

The two publications produced during this research are:

- Guido Zunino and Henrik I Christensen, "Navigation in Realistic Environments", International Symposium on Intelligent Robotic Systems (SIRS), Toulouse, France, July 2001.
- Guido Zunino and Henrik I Christensen, "Simultaneous Localization and Mapping in Domestic Environments ", IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Baden-Baden, Germany, August 2001.

Chapter 2

Domestic Robots

This chapter contains a presentation of the robots used in the experiment of this thesis. The two platforms used for this work are the Electrolux vacuum cleaner "Trilobite" and the Nomadic "SuperScout".

2.1 The Trilobite

The Trilobite (figure 2.1) is the worlds only series-manufactured, automatic vacuum cleaner. The first prototype was presented to the public in 1997 on the BBC TV program "Tomorrows World". The Trilobite has been developed by the Swedish Electrolux, and it is still having extensive research and development behind it. The product has a charging station where the vacuum cleaner docks by itself, three cleaning programs (normal, quick, and spot vacuuming), flexible drive wheel suspension and an LCD display. The Trilobite uses sonar sensors for navigation (high frequency is used in order not to bother animals), which improves its "sight", a new fan system and high-performance, environmentally friendly batteries (nickel-metal hydride). Thanks to a collaboration between the company and the Centre for Autonomous Systems, the Trilobite has become one of the testing platforms at our institute.

(a) (b)

Figure 2.1: (a) Electrolux Trilobite. (b) The Trilobite vacuuming in a bedroom.

2.1.1 Ultrasound navigation

The vacuum cleaner navigates using ultrasound. The sonar that emits the ultrasound vibrates at a rate of 60,000 Hz and is coated with a thin gold plate for best performance. The sonar system is composed of a wide angle transmitter (figure 2.2a), placed on the front part of the platform, and set of 8 microphones (figure 2.2b) for detecting the reflected echoes. The Trilobite has no problem avoiding collision with things placed on the floor, thanks to the wide opening angle the microphones can detect any kind of object toward the robot heading. Special magnetic strips are placed in doorways, near stairs and other openings. These act as a wall, keeping the Trilobite in the room. The machine contains sophisticated electronics and four motors, one for each drive wheel, one for the brush and one for the fan.

(a) (b)

Figure 2.2: (a)Ultrasound transmitter. (b)Microphone to receive the ultrasound echo.

2.1.2 Finds its own charger

The machine can find its own way back to the charger. If the cleaning has not been completed when the vacuum cleaner needs to be charged, it automatically moves to the recharger. The Trilobite continues cleaning when charging is finished, after about two hours. Once cleaning is completed, the machine returns to its charging station and switches to rest mode.

2.1.3 Trilobite's navigation

The commercial Trilobite, uses the following navigation strategy: it first explores the perimeter of the environment, using a wall following behavior, then, after the vehicle is back at the starting point it estimates the size of the environment and starts with a random path (see figure 2.3). The random strategy is fine for small environments, but it is sub-optimal for large and cluttered rooms. A smarter strategy would be preferable, implying the use of localization and mapping of the environment.

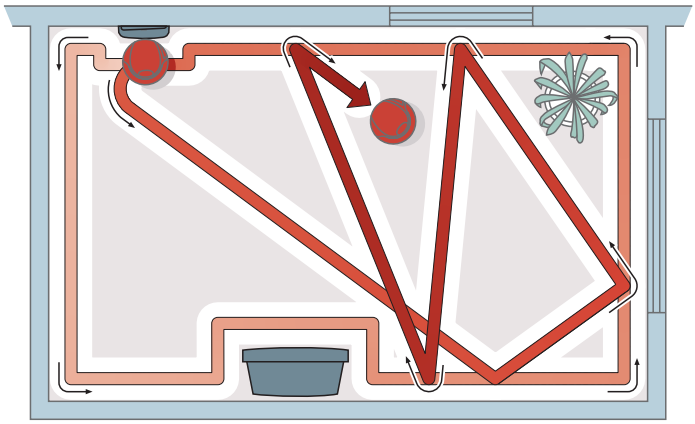


Figure 2.3: Trilobite navigation (courtesy of Electrolux).

2.2 Nomadic SuperScout

The Nomad SuperScout (figure 2.4) is an integrated mobile robot system with ultrasonic, tactile and odometry sensing. It uses a special multi-processor low-level control system that controls the sensing, motion, and communications. At a high level, the Scout is controlled by the on board PC computer communicating to remote workstations via radio modem. Alternatively, the Super Scout can be controlled via joystick directly from the user. The standard set of sensors with which the robot is equipped are: 16 Polaroid sonar sensors placed equidistantly around the robot, a set of microswitches to detect collision with obstacles and encoders on the driven wheels. There is, however, room for a wide set of external sensors, such as, laser scanners, cameras, infrared range scanners, sonar arrays. The SuperScout is widely used in the research community due to its relatively low cost, and to the large numbers of software tools dedicated to it. Nomadic technologies has provided a nice programming environment which allows the user to develop algorithms which can be tested with either the real or a simulated robot platform without any change in the program.

Figure 2.4: Nomadic SuperScout family at the Centre for Autonomous Systems.

Chapter 3

Feature detection

This chapter presents the methods used for extracting features from the environment using sonar sensors. A prerequisite for a proper design of these methods is a solid understanding of the sensors being used. To this end, sensor models are developed for the sonar systems on the Electrolux Trilobite as well as the Nomadic SuperScout. An outline is also given of the *Triangulation Based Fusion* algorithm on which the feature extraction method is built. Finally, the measurement models for the two robots are presented, with estimates of the quality of the features.

3.1 Triangulation on the Scout

The Nomadic Scout Robot, as explained in chapter 2, is equipped with the Polaroid 6500 Sonar Sensor. This sensor works as both transmitter and receiver. Due to the relatively slow speed of robots operating in domestic environments, it is here assumed that the sensor does not move between the time that the ultrasound pulse it is sent and the corresponding echo is received. This is not true in reality, but let us consider the robot moving at the maximum speed allowed (0.4 m/sec) firing toward an object 5 meters away from the robot. In this worst case scenario, the distance d_d between the position of the sensor when it fires and when it receives the echo is

$$d_d \leq 0.4m/sec \cdot \frac{2 \cdot 5m}{340m/sec} = 11.7mm.$$

This distance is neglected and a compensation for that is done in the range sensor model (3.2). Using this assumptions the sensor model for the Polaroid 6500 becomes:

Angular sensor model: Given a range reading from an edge located 0-5 m away from the sensor, the angular offset θ from the center beam axis is assumed to be uniformly distributed as:

$$\theta \sim U(-12.5^\circ, 12.5^\circ). \tag{3.1}$$

Range sensor model: Range readings from an object are assumed to be normally distributed around the true range distance $\bar{r}$ as:

$$r \sim N(\bar{r}, 0.01\bar{r} + 0.01m). \tag{3.2}$$

The $0.01m$ term in the variance of the gaussian distribution is added to compensate the fact that the sensor is not static during the measurement process.

Considering this model it is possible to use the following triangulation principle: consider a sonar reading which corresponds to a vertical edge in the environment (see figure 3.1a), due to the model of the Polaroid sonar sensor (3.1),(3.2) the object which this reading is generated from, can be anywhere along the $\pm 12.5^\circ$ corresponding beam arc. If the robot moves to another position and gets the reading from the same edge, the location of the edge can be extracted by computing the intersection point $T = (x_T, y_T)$ between the two arcs (figure 3.1b).

The equations used for determining the intersection point are:

$$(x_T - x_{s_i})^2 + (y_T - y_{s_i})^2 = r_i^2 \tag{3.3}$$

$$\arctan \frac{y_T - y_{s_i}}{x_T - x_{s_i}} \in \left[\gamma_i - \frac{\delta}{2}, \gamma_i + \frac{\delta}{2}\right], \tag{3.4}$$

where (x_s, y_s) denotes the sensor position, r the range reading, γ the sensor heading angle and δ the opening angle of the center sonar lobe. Equation (3.3) is solved to yield

$$x_T = x_{s_1} + \frac{1}{d_s^2}(d_{x_s} d_r^2 \pm |d_{y_s}| \sqrt{r_2^2 d_s^2 - d_r^4}) \tag{3.5}$$

$$y_T = y_{s_1} + \frac{1}{d_s^2}(d_{y_s} d_r^2 \pm |d_{x_s}| \sqrt{r_2^2 d_s^2 - d_r^4}), \tag{3.6}$$

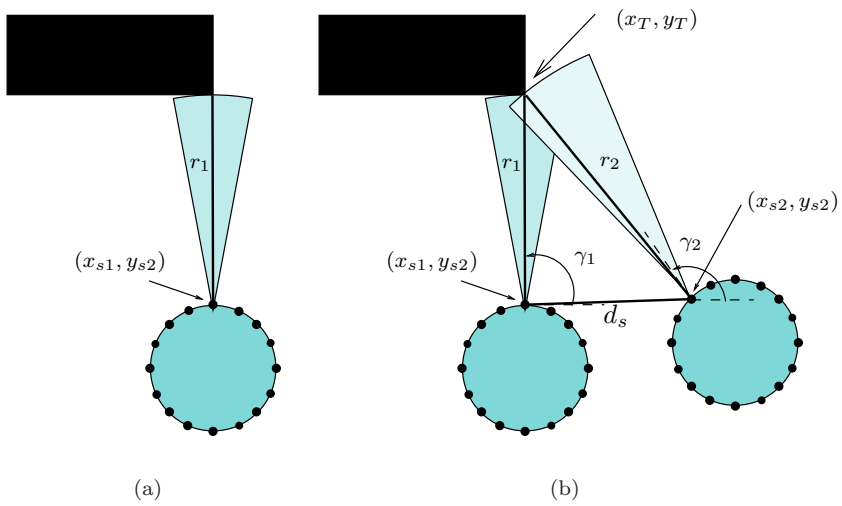


Figure 3.1: Basic triangulation principle.

with

$$\begin{aligned} d_{x_s} &= x_{s_1} - x_{s_2} \\ d_{y_s} &= y_{s_1} - y_{s_2} \\ d_s^2 &= d_{x_s}^2 + d_{y_s}^2 \\ d_r^2 &= \frac{r_1^2 - r_2^2 - d_s^2}{2}. \end{aligned}$$

3.2 Triangulation on the Trilobite

Let us now examine the model of the sonar system installed on the Electrolux Trilobite. This sensor is composed of a wide angle transmitter, placed in the front part of the vehicle, and by 8 microphones. Four out of the eight microphones are distributed on the horizontal plane, with angles respect to the robot heading, of $-90°$, $-45°$, $0°$ and $45°$ (see figure 3.2), the fifth microphone angled $90°$ is missing due to cost production reasons. The maximum range of these sensors is 2.5 meters. The other four sensors are placed at an angle of $45°$ with respect to the horizontal plane at angles $-67.5°$, $-22.5°$, $22.5°$ and $67.5°$. These sensors are

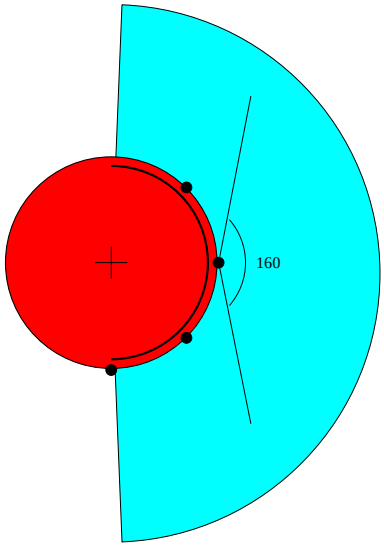


Figure 3.2: Opening angle of the microphones.

only used for obstacle avoidance due to their limited range (25 cm). As a result of this particular configuration the microphones can get echoes back from objects angled far apart from the heading of the microphone. Experimental tests showed that microphones detect echoes from object angled up to 80° with respect to the microphone. This is true if the object is within the lobe of the transmitter (see figure 3.2). The microphone placed at −90° cannot detect echoes from object on the back of the robot, since the transmitter fires the sonar pulse at the front of the Trilobite. The configuration thus allows multiple microphones to "see" one feature at the same time as shown in figure 3.3. As the transmitter and the receiver are not in the same physical position, the geometry of the problem becomes a bit more complicate than in the case of the Polaroid sensor. The distance r reported by the microphone is given by:

$$r = \frac{d_1 + d_2}{2}, \tag{3.7}$$

where d_1 is the closest distance from the transmitter to the object, and d_2 is the distance from the object to the microphone (see figure 3.4).

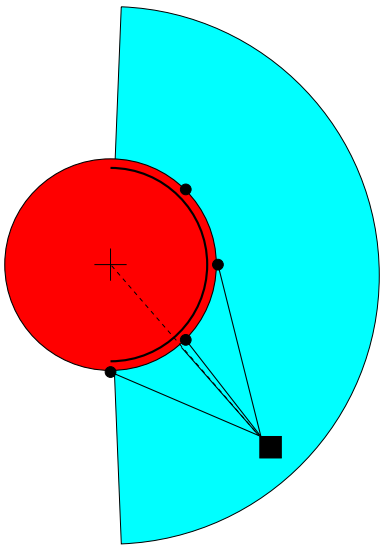


Figure 3.3: Feature detected by three microphones.

Modelling the transmitter as a point source placed in the center of the robot, given a reading r the reflecting object can be anywhere on the intersection of the opening angle of the microphone and the ellipse:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1, \tag{3.8}$$

where

$$\begin{aligned} a &= r + \frac{R}{2} \\ b &= \sqrt{a^2 - c^2} \\ c &= \frac{R}{2}. \end{aligned}$$

and R is the radius of the robot.

The angular and the range model of the sensor becomes:

Angular sensor model: Given a range reading from an edge located 0-2.5 m away from the sensor, the angular offset θ from the

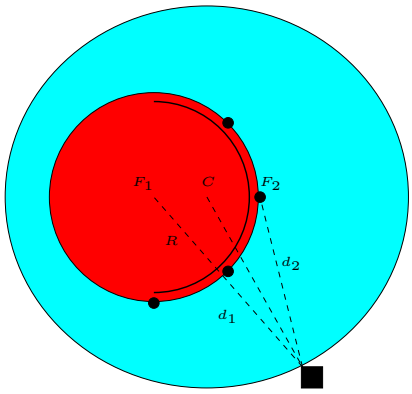


Figure 3.4: The reading r could be generate by an object placed anywhere along the ellipse arc.

axis of the microphone is assumed to be uniformly distributed as:

$$\theta \sim U(-80°, 80°). \tag{3.9}$$

For the microphone placed $-90°$ from the robot heading this angular offset is modeled as:

$$\theta \sim U(0°, 80°). \tag{3.10}$$

Range sensor model: Analogously to the polaroid sensor range readings from an object are assumed to be normally distributed around the true distance $\bar{r} = (d_1 + d_2)/2$ as:

$$r \sim N(\bar{r}, 0.01\bar{r} + 0.01m). \tag{3.11}$$

Following these considerations the basic triangulation principle, for the Trilobite, becomes an intersection between ellipse arcs. The intersection between the two ellipses is performed numerically. First the two ellipses are parameterized:

$$\begin{aligned}
x_{ell1} &= x_{c1} a_{ell1} \cos(\phi_{ell1}) \cos(t_1) r + b_{ell1} \sin(\phi_{ell1}) \sin(t_1) \\
y_{ell1} &= y_{c1} b_{ell1} \cos(\phi_{ell1}) \sin(t_1) r - a_{ell1} \sin(\phi_{ell1}) \cos(t_1) \\
x_{ell2} &= x_{c2} a_{ell2} \cos(\phi_{ell2}) \cos(t_2) r + b_{ell2} \sin(\phi_{ell2}) \sin(t_2) \\
y_{ell2} &= y_{c2} b_{ell2} \cos(\phi_{ell2}) \sin(t_2) r - a_{ell2} \sin(\phi_{ell2}) \cos(t_2).
\end{aligned}$$

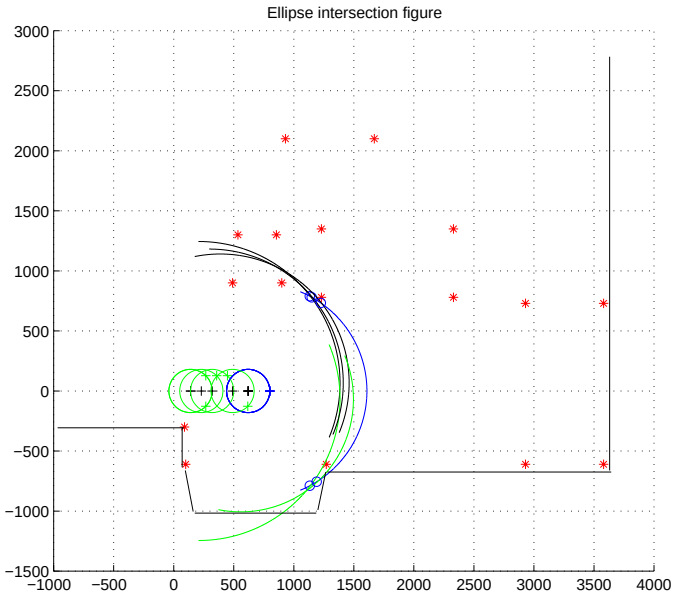


Figure 3.5: Through the intersection of the ellipse arcs generated by a set of readings the hypothesis for point features can be extracted.

Here (x_c, y_c) is the center of the ellipse, a and b are the axes, ϕ is the angle of the ellipse, and t is the parameter of the curve. Then the end points of the ellipses arcs are computed considering the sensors' angular models (3.9)-(3.10). If the intersection between the two arcs is allowed, the point is computed using the regula-falsi method (Asaithambi, 1995). This method is computational efficient, and converges in few iterations. The point $(\tilde{x}_T, \tilde{y}_T)$ extracted with this numerical method is not the real intersection point (x_T, y_T). The error is, however, bounded to 1 cm and it does not effect the algorithm as it will be explained in section 3.4. Figure 3.5 shows an example of readings from the Trilobite with the corresponding intersections of the ellipse arcs.

3.3 Triangulation Based Fusion Algorithm

The TBF technique (Wijk and Christensen, 2000) is a voting scheme for combining together sonar readings that have hit a mutual vertical edge in the environment. The assumptions for this algorithm are that the sonar sensors are mounted on the robot and that they are distributed on the same horizontal plane. In this case, a 2D assumption of the world is done and the vertical edges are considered as points. The original TBF algorithm was developed for the Polaroid 6500 Sonar Sensor, where the same physical sensor works both as the transmitter and the receiver, in this work it is applied also for sensors where the transmitter and the receiver are separated.

The TBF algorithm uses a the triangulation principle described in section 3.1 and section 3.2. Using a temporal buffer in which the position and the reading of each sonar is stored, it is possible to implement a recursive algorithm for grouping together sensor readings which have hit a mutual edge, and to estimate the edge position, referred as triangulation point $\hat{T} = (\hat{x}_T, \hat{y}_T)$, with a parameter n_t indicating the number of readings which contribute to the estimate. Details about the implementation can be found in (Wijk, 1998; Wijk and Christensen, 1998; Wijk, 2001). An intuitive illustration on how the TBF algorithm works is given in figure 3.6. In figure 3.6a the robot gets a reading from an object in the environment, the position estimate of the object is placed at the center of the sonar beam and the triangulation number n_t is set to zero. Then looping through the temporal buffer which contains the readings of all the sonars, the reading in figure 3.6b is matched to the first reading and the estimate of the feature position and the triangulation number is updated. When a third reading is matched to the first one (figure 3.6c) the position of the estimate is update using the following equations:

$$\hat{x}_T = \frac{1}{n_t + 1}(n_t \hat{x}_T + x_T^{tri}) \tag{3.12}$$

$$\hat{y}_T = \frac{1}{n_t + 1}(n_t \hat{y}_T + y_T^{tri}) \tag{3.13}$$

$$n_t = n_t + 1, \tag{3.14}$$

where (x_T^{tri}, y_T^{tri}) is the triangulation point extracted using equations (3.5)-(3.6) using the current reading and the first reading. By using a threshold on n_t it is possible to filter out all the hypotheses with low reading support which are, most probably, not generated by point features.

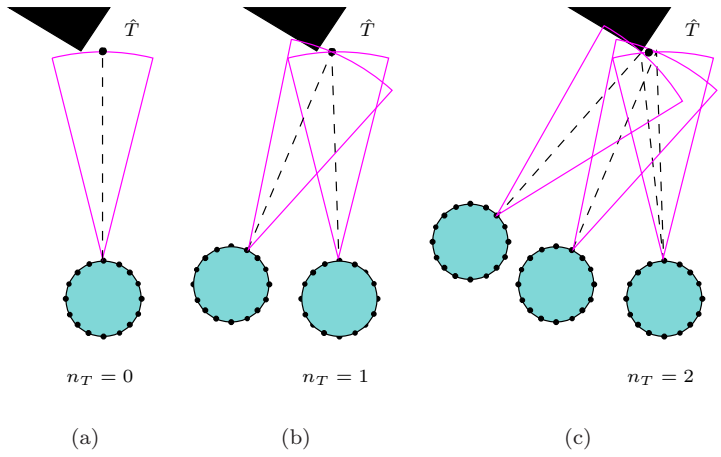
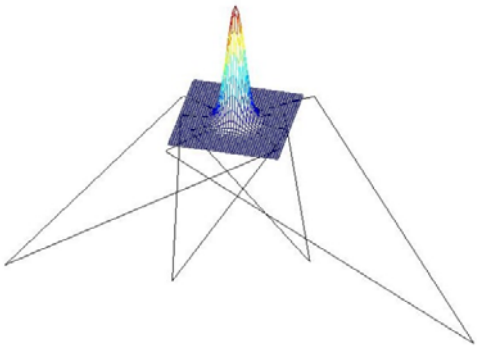


Figure 3.6: Basic triangulation principle.

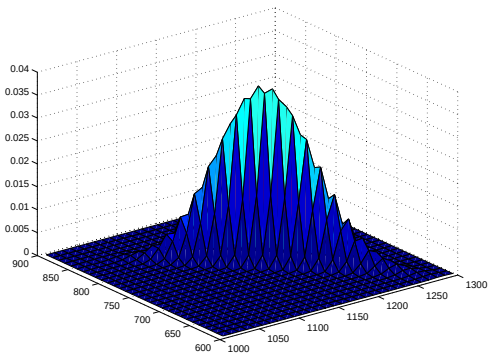
3.4 Measurement model

The algorithm presented in the previous section provides an estimate on the feature location, but does not provide any uncertainty bound of this estimate. The estimate might be not extremely precise, i.e. in the case of the Electrolux Trilobite it is affected by the error in the numerical intersection routine as explained in section 3.2. A more precise estimate of the triangulation point location is performed by placing a local grid, centered on the estimate position of the point $\hat{T} = (\hat{x}_T, \hat{y}_T)$, and updating this grid using the readings which support $\hat{T}$ with the sensor model (3.1), (3.2) or (3.9), (3.10), (3.11) depending on the robot used. Treating the readings as independent the fusion process consist of simple multiplications (Wijk, 2001). When all the readings have been fused to the grid, the cell in the grid with the maximum value is chosen as the refined estimation point $T = (x_T, y_T)$. The initial estimate $\hat{T}$ is, therefore, not required to be extremely precise, it just has to be in the neighborhood of the true point feature location. The covariance matrix P_T is given by

$$P_T = \sum_{i,j} \begin{pmatrix} (x_{ij} - x_T)^2 & (x_{ij} - x_T)(y_{ij} - y_T) \\ (x_{ij} - x_T)(y_{ij} - y_T) & (y_{ij} - y_T)^2 \end{pmatrix} p_{ij} \quad (3.15)$$



(a)



(b)

Figure 3.7: (a) Example of the local grid update. (b) Local grid obtained with readings from the Trilobite.

where (x_{ij}, y_{ij}) is the ij cell's position and p_{ij} is the associated probability.

The TBF algorithm supplies the measurement of triangulation points in cartesian coordinates (x_T, y_T), these measurements will now be referred as range and angle with respect to the robot pose (ρ, θ).

$$\rho = \sqrt{(x_T - x_r)^2 + (y_T - y_r)^2} \tag{3.16}$$

$$\theta = \arctan\left(\frac{y_T - y_r}{x_T - x_r}\right) - \theta_r. \tag{3.17}$$

The covariance of the measurement becomes

$$\mathbf{R} = J_m P_T J_m^T, \tag{3.18}$$

where

$$J_m = \begin{bmatrix} \frac{x_T - x_r}{r} & \frac{y_T - y_r}{r} \\ -\frac{y_T - y_r}{r^2} & \frac{x_T - x_r}{r^2} \end{bmatrix} \tag{3.19}$$

is the jacobian of the coordinate transformation.

3.5 Summary

In this chapter the models of the Polaroid sonar sensor and of the Trilobite's sonar sensor are developed. The Triangulation Based Fusion technique is utilized for detecting point features in the environment using these sensor's models. The technique used for extracting the error bound of the measurements is described. Finally the measurement model equations are presented. These measurement equation will be used in the methods developed in the rest of the thesis.

Chapter 4

Simultaneous Localization and Mapping

In this chapter the Extended Kalman Filter approach to the SLAM problem is presented. An overview of the current work on the field is presented, then the model of our platforms is described and used together with the measurement model described in chapter 3 for building the EKF framework. At the end of the chapter an experimental test is performed showing the performance of the algorithm in a standard domestic living-room.

4.1 Current Work

The seminal research effort in feature-based simultaneous localization and mapping was performed by (Smith and Cheesman, 1987) who published the *stochastic mapping* algorithm, which is an Extended Kalman Filter approach to SLAM. The state of the vehicle and the position of all the features in the map is collected in one single state vector, and when features in the map are re-observed, both the robot pose and the location of the features are updated.

A first implementation of the *stochastic mapping* algorithm, with real data was presented by Moutarlier and Chatila in (Moutarlier and Chatila,

1989), where the "relocation-fusion" approach is presented. The updated step introduced by (Smith and Cheesman, 1987) is divided into two steps. In the "relocation" step only the vehicle pose estimate is updated using the sensor measurements. Then in the "fusion" step linearization around the updated robot pose is done and the features are updated. Due to the high computation requirement of the *stochastic mapping* algorithm (a direct implementation of the EKF approach has $O(N^3)$ complexity), different approaches to the map scaling issue have been investigated in the research community.

Since the correlation terms render high the complexity of the algorithm, an attempt to reduce correlations is presented in (Leonard and Durrant-Whyte, 1991). The reduction is achieved by considering the fact that correlations are due to updating the robot pose with uncertain map features or updating the features with uncertain robot pose. Neglecting the correlations is motivated by performing an update only with confirmed robot poses and confirmed features. A robot pose or a feature is defined to be confirmed when the covariance is smaller than a threshold in which case it is approximated to zero. This approach requires that at least one feature is acquired before the robot motion has started in order to avoid uncertainty due to vehicle pose uncertainty.

A different approach to the cross-correlation problem is presented in the papers (Csorba and Durrant-Whyte, 1997; Csorba et al., 1997), in this approach a relative frame of reference is used, the positions of the features are given only in relation to each other and not to the absolute frame of reference. The main disadvantage of this approach is that there is not a good way to pass from the relative map to one map in the absolute frame of reference.

Another approximation to the full SLAM algorithm is proposed in (Dissanayake et al., 2000). In this approach it is pointed out that removing features from the state vector does not lead to any inconsistencies, but only to a loss of information. By choosing a subset of relevant features in the environment, the SLAM algorithm can run without compromising the overall performances.

In (Uhlmann, 1995) the Covariance Intersection (CI) method is applied to the SLAM problem. CI (also known as Gaussian Intersection) is applicable when the correlations are unknown, and provides a conservative estimate of the covariance.

One other approach to face the map scaling problem is the Decoupled Stochastic Mapping (DSM) technique presented in (Feder, 1999), where the environment is divided in sub-maps. Each of those maintains full

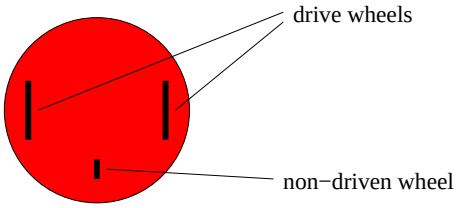


Figure 4.1: Differential drive system.

correlation information, and the correlations between sub-maps are given through the use of a common coordinate system.

Castellanos and Tardós (Castellanos and Tardós, 1999) have proposed a SLAM algorithm using a combination of many features. A common assumption is that a feature can only match a feature of the same type. This limitation is alleviated by introducing the "binding matrices" that relate different features. In this way, for example, a plane can be used to update an edge.

In this thesis the *map scaling* problem it is not directly approached since the environment in which the robot performs its task is relatively small and the number of the features extracted is small enough to run the SLAM algorithm in real time. The size of standard domestic environments is typically of few tens square meters, and the number of features necessary to perform efficient navigation in such environments it is not higher than 20-30.

4.2 Robot modelling

In this section we develop a simplified model of the vehicle and state our assumptions. This model and these assumptions are used throughout this thesis. In this work we have restricted the model to a two dimensional space, because for domestic environments the z-coordinate is not of relevance. Both the Trilobite and the SuperScout use a differential drive system. The differential drive is a two-wheeled drive system with independent actuators for each wheel. The name refers to the fact that the motion vector of the robot is sum of the independent wheel motions. The drive wheels are usually placed on each side of the robot as shown in figure 4.1, where the large black rectangles are the drive wheels. The small black rectangle is a non-driven wheel which forms a tripod-like sup-

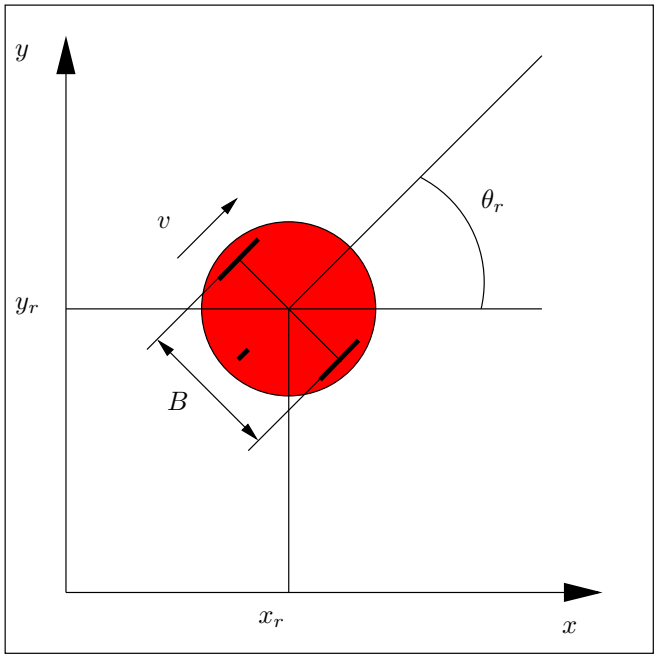


Figure 4.2: Definitions of the robot's state variables used in the model.

port structure for the body of the robot. Often, the passive wheel is a caster wheel, a small swivelled wheel used on office furniture.

In our implementation we denote the robot state estimate by $\mathbf{x}_r = [x_r\, y_r\, \theta_r]^T$ as shown in figure 4.2, and the control input $\mathbf{u} = [v_1 v_2]^T$, where v_1 and v_2 are the velocity of the right and of the left driving wheel respectively.

A general dynamic model for the vehicle can be defined as

$$\mathbf{X}_{r\,k+1} = \mathbf{f}(\mathbf{X}_{r\,k}, \mathbf{u}_k) + \mathbf{q}_k \tag{4.1}$$

where $\mathbf{f}$ is a nonlinear model of the vehicle, taking the current state $\mathbf{X}_{r\,k}$ and the control input $\mathbf{u}_k$ as input variables, $\mathbf{q}_k$ is the noise process with zero mean and covariance Q_k, which takes into account the unmodeled dynamics and noise. k signifies a discrete time index and the model is updated at times $t = kT$ for a constant period T. In this thesis we use the following definition of $\mathbf{f}$

$$\mathbf{f} = \begin{bmatrix} x_r + \Delta S \cos(\theta_r + \frac{\Delta\Theta}{2}) \\ y_r + \Delta S \sin(\theta_r + \frac{\Delta\Theta}{2}) \\ \theta_r + \Delta\Theta \end{bmatrix}, \tag{4.2}$$

were

$$\Delta S = \frac{v_1 + v_2}{2} \tag{4.3}$$

$$\Delta\Theta = \frac{v_1 - v_2}{B}. \tag{4.4}$$

The vehicle model of equation 4.1 does not take into account the real vehicle dynamics, thus the model is purely kinematic.

4.3 Extended Kalman Filter Approach

The *stochastic map* is a special way of organizing the states in an Extended Kalman Filter for the purpose of feature relative navigation. The much celebrated Kalman Filter (Kalman, 1960; Welch and Bishop, 1995) is a set of mathematical equations that provides an efficient computational recursive solution of the least square problem. The filter is very powerful in several aspects: it supports estimation of past, present and even future states, and it can do so even when the precise nature of the modeled system is unknown. The Extended Kalman Filter (EKF) is the technique of linearizing a non-linear dynamic system for use in a Kalman Filter. Using this framework, the measurements are used to create a map of the environment which, in turn, is used to localize the robot. In our implementation we use

$$\mathbf{x}_{k|k} = \mathbf{X}_k + \eta_k \tag{4.5}$$

to represent the full system state vector $\mathbf{x} = [\mathbf{x}_r^T\, \mathbf{x}_1^T\, \mathbf{x}_2^T \ldots \mathbf{x}_N^T]^T$, where $\mathbf{x}_r = [x_r\, y_r\, \theta_r]^T$ is the estimate of the robot position and $\mathbf{x}_i = [x_i\, y_i]^T$ is the estimate of the landmark state, respectively, $\mathbf{X}$ is the true state vector, and η is the error of the estimate. k is a time index, the subscript $\mathbf{x}_{k+1|k}$ will be used to denote the estimate of the state $\mathbf{x}$ at time $k+1$ given all the information up to time step k. The estimate error covariance,

$\mathbf{P_{k|k}} = E[\eta_k \eta_k^T]$, of the system state takes the form

$$\mathbf{P}_{k|k} = \begin{bmatrix} \mathbf{P}^{rr} & \mathbf{P}^{r1} & \dots & \mathbf{P}^{rN} \\ \mathbf{P}^{1r} & \mathbf{P}^{11} & \dots & \mathbf{P}^{1N} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}^{Nr} & \mathbf{P}^{N1} & \dots & \mathbf{P}^{NN} \end{bmatrix} \tag{4.6}$$

The sub-matrices, $\mathbf{P}^{rr}$, $\mathbf{P}^{ri}$ and $\mathbf{P}^{ii}$ are, respectively, the robot to robot, robot to feature and feature to feature covariances, the sub-matrices $\mathbf{P}^{ij}$ are the feature to feature cross-correlations. The robot and the map are represented by a single state vector $\mathbf{x}$ with the corresponding estimate error covariance $\mathbf{P}$ at each time step. Given the system equation of the robot system (4.1) and the system equations of the features

$$\mathbf{x}_{i_{k+1}} = \mathbf{x}_{i_k}, \tag{4.7}$$

an EKF is employed to estimate the state $\mathbf{x}$ and the covariance $\mathbf{P}$ given the measurement $\mathbf{z}$. The estimation occurs trough a prediction step, caused by the movement of the vehicle , and an update step, which takes place when features are re-observed . When a new feature is observed it must be added to the state vector and covariance. Thus the evolution of *stochastic mapping* is divided into three steps : vehicle movement, feature integration and feature re-observation. These steps are described next.

4.3.1 Vehicle movement

When the robot moves the resulting vehicle state estimate at time step $k+1$ is given by taking the expectation in the state transition model (4.1).

$$\mathbf{x}_{r_{k+1|k}} = E\left[\mathbf{f}(\mathbf{X}_k, \mathbf{u}_k)\right] \approx \mathbf{f}(\mathbf{x}_{r_{k|k}}, \mathbf{u}_k). \tag{4.8}$$

Since the stochastic process $\mathbf{q_x}$ has zero mean, the term becomes zero when evaluating the expected value over the equation (4.1) to determine $\mathbf{x}_{r_{k|k}}$. The covariance $\mathbf{P^{rr}}_{k|k}$, is propagated through the linearized state transition model of the EKF yielding $\mathbf{P^{rr}}_{k|k}$ given by

$$\mathbf{P^{rr}}_{k+1|k} = \mathbf{J}_x \mathbf{P^{rr}}_{k|k} \mathbf{J}_x^T + \mathbf{Q}_k, \tag{4.9}$$

where $\mathbf{Q}_k = E[\mathbf{q}_k \mathbf{q}_k^T]$ and $\mathbf{J}_x$ is the Jacobian of $\mathbf{f}$ with respect of $\mathbf{X}$ evaluated at $\mathbf{x}_{k+1|k}$. This step, referred to as *prediction step*, produces

an estimate of the robot position exclusively based on the model of the robot and the odometry, the covariance in this case will increase as shown in figure 4.3a. It is also necessary to update the robot-to-feature cross correlation according to (Moutarlier and Chatila, 1989)

$$\mathbf{P^{ri}}_{k+1|k} = \mathbf{J}_x \mathbf{P^{ri}}_{k|k}, \tag{4.10}$$

while the feature-to-feature cross correlation and the covariance for the features do not change by moving the robot.

4.3.2 New feature integration

When a new feature $\mathbf{L}_{new} = [\rho\ \theta]$ is observed and validated (see figure 4.3b) the new feature state $\mathbf{x}_{N+1}$ is incorporated in the system vector state (Moutarlier and Chatila, 1989)

$$\mathbf{x}_{N+1} = \mathbf{m}(\mathbf{x}_{k+1|k+1}, \mathbf{L}_{new}) \tag{4.11}$$

$$= \begin{bmatrix} x_{r_{k+1|k+1}} + \rho\cos(\theta_{r_{k+1|k+1}} + \theta) \\ y_{r_{k+1|k+1}} + \rho\sin(\theta_{r_{k+1|k+1}} + \theta) \end{bmatrix} \tag{4.12}$$

$$\mathbf{x}_{k+1|k+1} \leftarrow \begin{bmatrix} \mathbf{x}_{k+1|k+1} \\ \mathbf{x}_{N+1} \end{bmatrix}, \tag{4.13}$$

$$\mathbf{P}^{N+1N+1} = \mathbf{J}_{\mathbf{x_r}} \mathbf{P}^{rr}_{k+1|k+1} {\mathbf{J}_{\mathbf{x_r}}}^T + \mathbf{J_z} \mathbf{R} {\mathbf{J_z}}^T, \tag{4.14}$$

$$\mathbf{P}^{rN+1} = {\mathbf{P}^{N+1r}}^T = \mathbf{P}^{rr}_{k+1|k+1} {\mathbf{J}_{\mathbf{x_r}}}^T, \tag{4.15}$$

$$\mathbf{P}^{N+1i}_{k+1|k+1} = {\mathbf{P}^{iN+1}_{k+1|k+1}}^T = \mathbf{J}_{\mathbf{x_r}} \mathbf{P}^{ri}_{k+1|k+1}{}^T. \tag{4.16}$$

The matrices $\mathbf{J}_{\mathbf{x_r}}$ and $\mathbf{J_z}$ are the Jacobians of $\mathbf{m}$ with respect to the robot state $\mathbf{x}_r$ and to $\mathbf{L}_{new}$.

4.3.3 Feature re-observation

Whenever a feature present in the state vector is observed the update step of the EKF is used to update the state of the map including the robot pose (figure 4.3c). Defining

$$\overline{x}_i = x_{i_{k+1|k}} - x_{r_{k+1|k}}$$

$$\overline{y}_i = y_{i_{k+1|k}} - y_{r_{k+1|k}} \tag{4.17}$$

the observation model of the equation (3.16) for the feature i takes the form

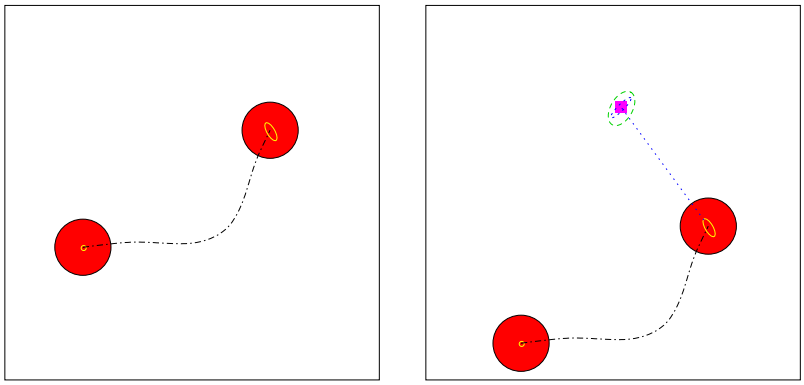
$$\begin{aligned} \mathbf{z}_{ik+1} &= \begin{bmatrix} \rho_{ik+1} \\ \theta_{ik+1} \end{bmatrix} = \begin{bmatrix} \sqrt{\overline{x}_i^2 + \overline{y}_i^2} \\ \arctan \frac{\overline{y}_i}{\overline{x}_i} - \theta_{r_{k+1|k}} \end{bmatrix} + \mathbf{n}_{zi} && (4.18) \\ &= \mathbf{h}_i(\mathbf{x}_{k+1|k}) + \mathbf{n}_{zi}. && (4.19) \end{aligned}$$

The noise process $\mathbf{n}_{zi}$ is assumed to be white Gaussian with covariance $\mathbf{R}_i$. If the N features are observed the observation model becomes

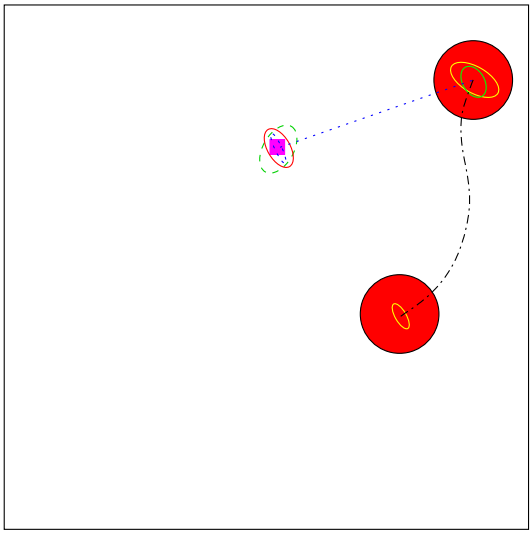
$$\begin{aligned} \mathbf{z}_{k+1} &= \begin{bmatrix} \mathbf{z}_{1k+1} \\ \vdots \\ \mathbf{z}_{Nk+1} \end{bmatrix}, && (4.20) \\ \mathbf{h} &= \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_N \end{bmatrix}, \\ \mathbf{R}_{k+1} &= \begin{bmatrix} \mathbf{R}_1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{R}_N \end{bmatrix}. \end{aligned}$$

With the Jacobian of $\mathbf{h}$ given by H_x the update step of the EKF becomes (Moutarlier and Chatila, 1989)

$$\begin{aligned} \mathbf{x}_{k+1|k+1} &= \mathbf{x}_{k+1|k} + \mathbf{K}_{k+1}(\mathbf{z}_{k+1} - \mathbf{h}(\mathbf{x}_{k+1|k})), && (4.21) \\ \mathbf{P}_{k+1|k+1} &= (\mathbf{I} - \mathbf{K}_{k+1}\mathbf{H}_x)\mathbf{P}_{k+1|k}, && (4.22) \\ \mathbf{K}_{k+1} &= \mathbf{P}_{k+1|k}\mathbf{H}_x^T(\mathbf{H}_x\mathbf{P}_{k+1|k}\mathbf{H}_x^T + \mathbf{R}_{k+1})^{-1}. && (4.23) \end{aligned}$$



(a) (b)



(c)

Figure 4.3: (a)Illustration of the prediction step in stochastic mapping, (b)Feature integration step. (c)Feature re-observation step.

4.3.4 Data association

A critical aspect for the SLAM algorithm in a real world scenario is the data association. Due to the fact that the detection of features in cluttered environments does not always provide clean data, data association, track initiation and track deletion is performed. The objective of data association is to assign measurements to the features from which they originate and reject spurious measurements. The *measurement-to-feature* association is performed using a gating approach in the innovation space (Bar-Shalom and Fortmann, 1988), incorporating both measurement uncertainty and robot uncertainty. The robot's uncertainty is transformed into measurement space and added to the measurement uncertainty. The innovation matrix $\mathbf{S_i}$ for the feature i is given by:

$$\mathbf{S}_i = \mathbf{H}_{\mathbf{x_i}} \begin{bmatrix} \mathbf{P}^{rr} & \mathbf{P}^{ri} \\ \mathbf{P}^{ir} & \mathbf{P}^{ii} \end{bmatrix} \mathbf{H}^{\mathbf{T}}_{\mathbf{x_i}} + \mathbf{R_i}, \tag{4.24}$$

where

$$\mathbf{H}_{\mathbf{x_i}} = \frac{d\mathbf{h_i}([\mathbf{x}_r \mathbf{x}_i])}{d[\mathbf{x}_r \mathbf{x}_i]}. \tag{4.25}$$

Defining the innovation $\nu_i = \mathbf{z}_i - \mathbf{h}_i(\mathbf{x}_i)$ the validation gate is given by:

$$\nu_i^T \mathbf{S}_i^{-1} \nu_i \leq \lambda. \tag{4.26}$$

For a system with 2 degrees of freedom, a value of $\lambda = 9.0$ yields the region of minimum volume that contains the measurement with a probability of 98.9% (Bar-Shalom and Fortmann, 1988). Such a validation procedure defines where a measurement is expected to be found. The initiation of a new feature is performed using a nearest neighbor gating technique described in (Feder, 1999). In performing the track initiation, all the measurements that have not been matched with any feature in the map, are stored. Basically, any measurement which has not been associated to an existing feature is a potential new feature. At each time step a search of clusters of $\beta \geq N$ of unmatched measurement is performed. For each of this clusters a new track is initiated. The cluster is defined as a measurement which gate according to the equation (4.26) with all the other measurements in the cluster. A track is deleted if the robot has moved more than a certain distance d since the first measurement of the correspondent cluster was detected. In our system, due to the relatively low probability of false return and a not very high probability of detection

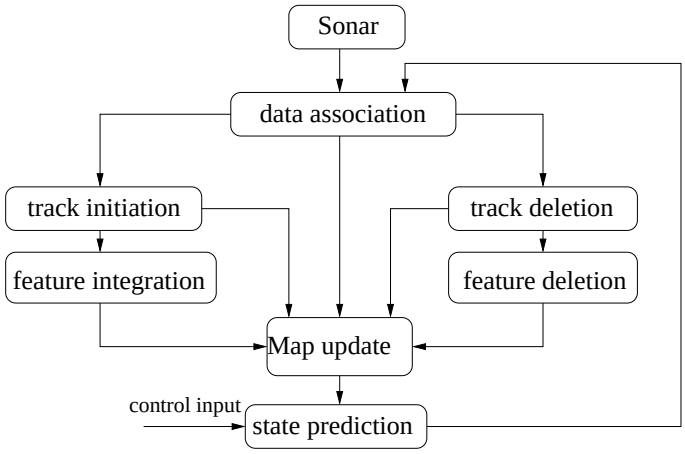


Figure 4.4: Structure of the SLAM algorithm.

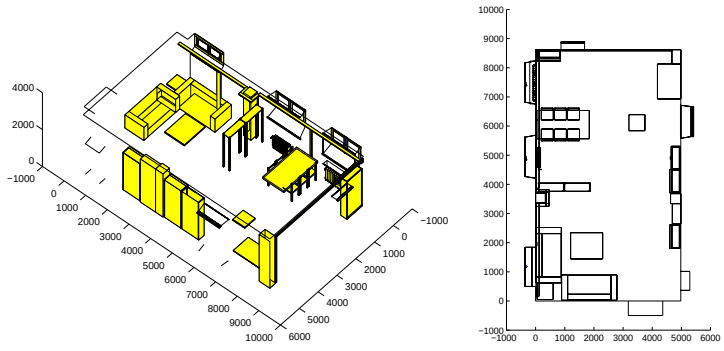
(about 50%) a value of N=4 is chosen. Since the environment is assumed to be static the deletion of features in the map is not taken into account. However, in chapter 5 a method for deleting features, which have been placed wrongly due to failures of the algorithm, is presented. The most common situation where false features are added into the state vector is when the robot is not in the pose it believes to be. The new feature acquired will, therefore, be placed inconsistently into the map.

4.4 Experimental results

To evaluate the performance of the EKF algorithm in a real domestic setting, an experimental test of the *stochastic mapping* algorithm is presented. The test environment is a regular living room set up in our laboratories at the Centre for Autonomous Systems (see figure 1.1). The room is the size of about 5×9 meters, a CAD model of the living-room is shown if figure 4.5. This model has been built for the evaluation of the experiments, it is an extensive model of the environment, containing all the furniture in it. Due to the limited range of the feature detection algorithm, and in order to get a more distributed set of features in the room a small table has been set in the middle of the living-room (figure 4.5b). This "trick" helps the algorithm, because it adds features in an area which would have been empty otherwise. Features lying in the cen-

ter of the room are, in fact, easily detectable from many positions by the robot. This fact helps the algorithm to converge faster. In the following experiment an eight shape trajectory is performed. In figure 4.6 is shown the position estimate of the robot, with the ellipses representing the covariance of the robot's position. The ground truth is obtained using a laser pose tracker, a process which accurately supply the position and orientation of the robot. This information is just used for evaluation purposes, and is not used by the algorithm. Figure 4.8 shows the error and the 2σ bounds of the state variables x,y and θ of the robot.

It can be observed from these figures that the uncertainty of the robot's state increases when the robot is exploring unknown areas (see $T \in [0-360]$ and $T \in [500-700]$ in figure 4.8), this is expected since no measurements of old features are performed. However, the uncertainty drops when the robot goes back to pre-explored locations and re-observes features in the map. The final feature map is represented in figure 4.7, with the ellipses centered in the feature position estimate, representing the 2σ uncertainty of the estimate. Note that features 4, 9, 14 and 19 in figure 4.7 correspond to actual objects not included in the CAD model of the living-room such as mobile robots and various equipment.



(a) (b)

Figure 4.5: (a)CAD 3D model of the CAS living-room. (b)2D projection of the model.

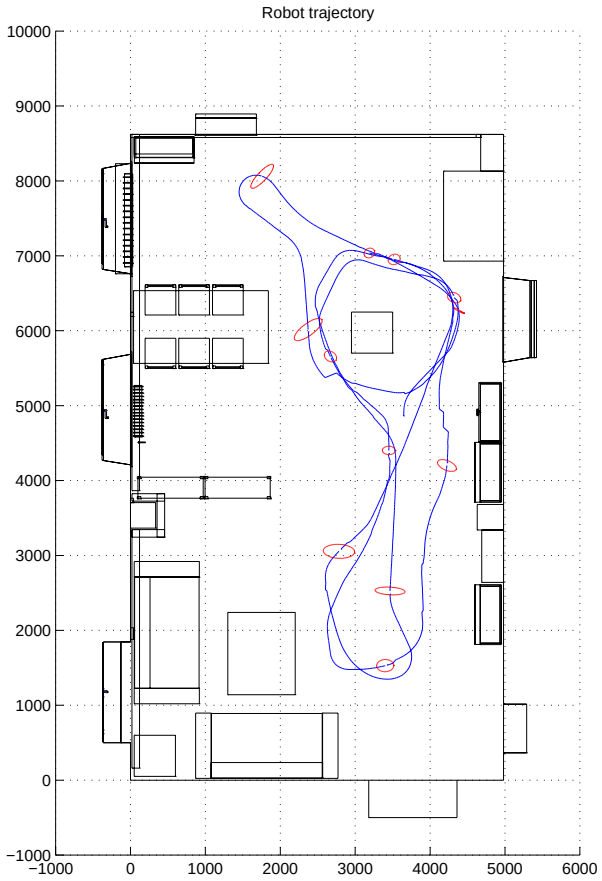


Figure 4.6: Trajectory of the robot in the living room.

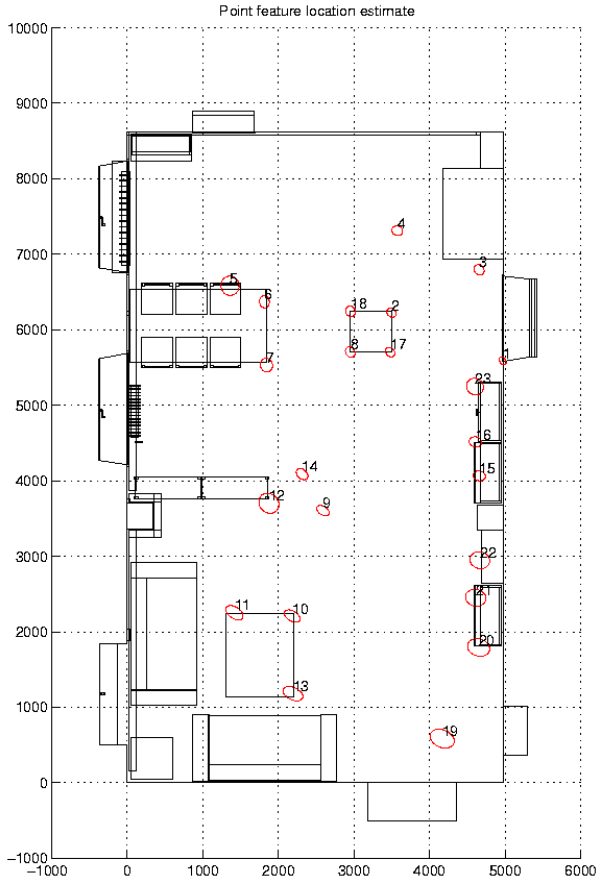
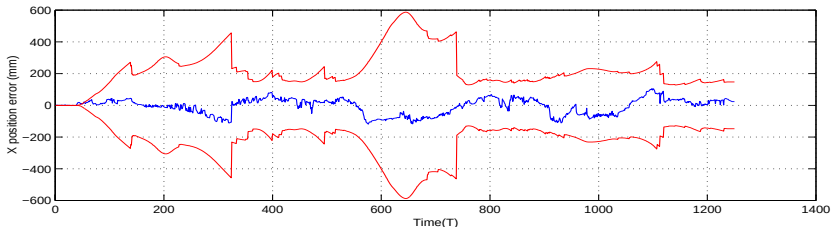
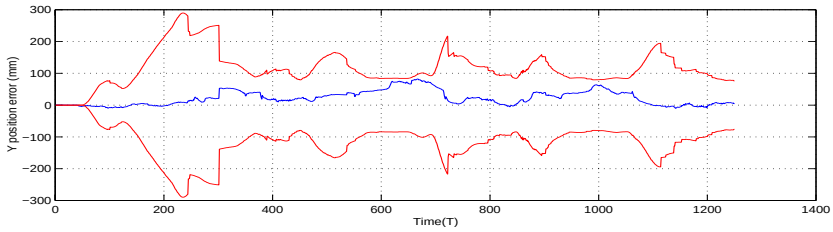


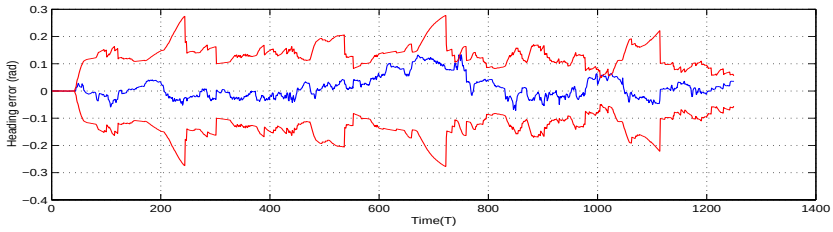
Figure 4.7: Landmarks position estimate with the 2σ covariance ellipses.



(a)



(b)



(c)

Figure 4.8: Error and the 2σ bounds,(a) x-position, (b) y-position and (c) heading.

4.5 Summary

In this chapter the Simultaneous Localization and Mapping problem was studied. A feature-based approach was proposed, using point features extracted with sonar data. The model of the robot was developed and used in the Extended Kalman Filter framework for estimating the pose of the vehicle and the location of the point features in the map. An experimental test was presented showing satisfactory performances in a real domestic environment. There is still room for improvements. The data associations is the most delicate part in the algorithm, improving it would allow better performance. This problem becomes significant in situations where not many features can be detected by the robot. In this situations, due to the few update steps the uncertainty of the robot pose estimate becomes bigger and bigger, and matching measurements to the features from which they originate becomes very hard. This problem is faced in the next chapter.

Chapter 5

Failure Recovery

5.1 Failures of the EKF approach

This section describes the different type of failures that the EKF approach can suffer. There are three basic modes of failure of the stochastic map approach:

- **divergence due to data association errors**
- **map slippage**
- **unexpected perturbation of the robot**

5.1.1 Data association failure

The first failure mode occurs when a measurement is associated to an incorrect feature, that might happen when features are very close to each other and the uncertainty in vehicle position is big making data association more ambiguous. This failure mode is the most troublesome because the algorithm believes to provide accurate information. The system state vector is, then, updated with erroneous data and the error will drift outside the bound defined by the estimate covariance. Figure 5.1 shows the result of a data association error the vehicle's state estimate moves outside the estimated error bound. One way of resolving data association errors on-the-fly is to compare the estimated vehicle state after the update step to the estimate produced by the odometry alone. If the state estimate of the EKF algorithm is outside of the odometry

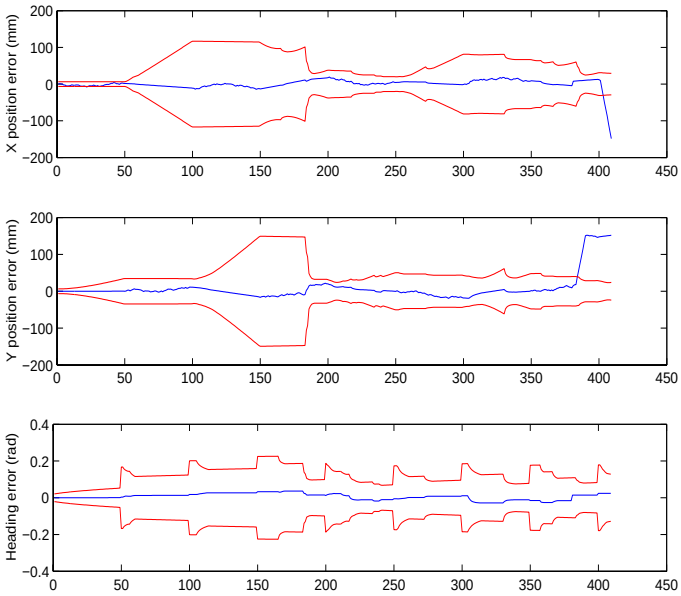


Figure 5.1: Example of divergence due to bad data association.

error bound a data association fault may have occurred. This is based on the following assumptions: the odometry estimate will *always* bound the true error, this is not, entirely, true since the odometry model does not account for everything, but in general it is the case; and the EKF update should *always* improve the dead-reckoning, if there were no data association errors. Therefore, if the correction in the robot pose provided by the update step of the EKF place the robot outside of the uncertainty bound estimated by the odometry model, the inconsistency is probably due to bad data association.

5.1.2 Map slippage failure

The second mode occurs when the robot's position is close to the error bounds and due to the linearization of the non-linear transformations all the features are re-mapped in new locations which are slightly shifted from the original map. When this is repeated the map will slip more and more causing an inconsistent map estimate.

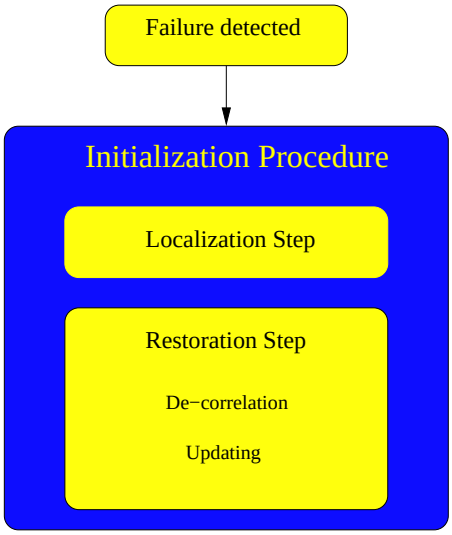


Figure 5.2: Steps involved in the recovery from a failure.

5.1.3 Failure due to unexpected perturbation

The third mode occurs when the robot is effected by a strong perturbation, which drives the actual error outside the error bound. This fact might occur very often in domestic environments, the robot can run over carpets or electrical cable, in some situations the robot might collide with an obstacle or it can be moved by a human, also the small size and light weight (Electrolux vacuum cleaner) this problem might occur frequently.

5.2 Detecting a failure

A simple method for detecting a failure mode is implemented (see figure 5.3). If the robot does not get any measurements from a location where a landmark is expected, a warning flag is set and a counter is incremented, whenever a measurement is matched with a feature the counter is reset to zero. If the counter reaches a threshold value M the robot considers itself lost and performs the initialization procedure. Due to the low sensitivity of the TBF algorithm in detecting features in front of or behind the robot.

Failure detecting algorithm

```
Select the set of features Γ
Loop over features F_i ∈ Γ
   Loop over all measurements m_j
      if (m_j ↔ F_i)
         Reset the counter C
         Exit from the loop
      else
          Increment the counter C
if (C ≥ M)
    Failure detected
```

Figure 5.3: Failure detecting algorithm.

The failure detecting algorithm considers only those features which are in determined positions to respect to the robot pose as explained in figure 5.4.

This algorithm turns out to be quite robust although it is rather simple. The principle is that whenever the robot does not get measurements from features which are expected to be seen that means that its position estimate is inconsistent. There are different situations in which the algorithm fails, the most evident are:

- The robot pose estimate is divergent, but, by coincidence, some measurements match expected features resetting the counter (see figure 5.5 a). This situation might happen in cluttered environments, where lots of measurements are performed. This fact will, just, delay the failure detection since it is not realistic that the situation will happen constantly over the time.

- The pose estimate is convergent, but the features expected to be seen are not detected by the robot. This can happen in rare occasions due to objects standing in between the robot an the feature, not detected when the feature was initiated (see figure 5.5 b). This fact will make the robot performing an unnecessary recovery step. This will waste computing resources, and supply a final estimate not as accurate as it would be without the unnecessary recovery.

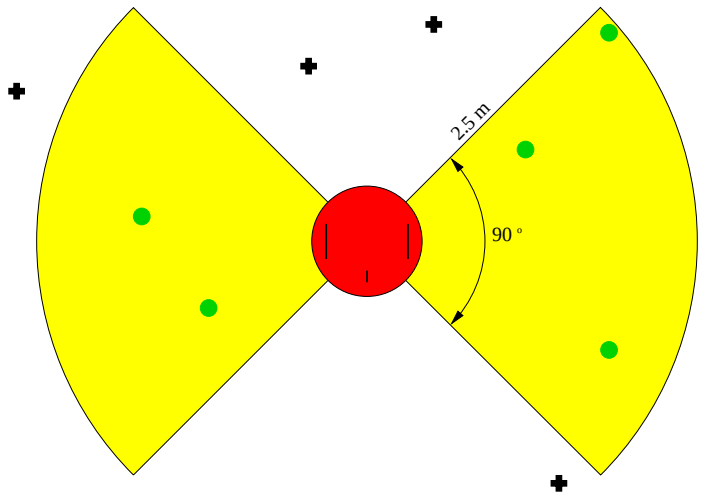
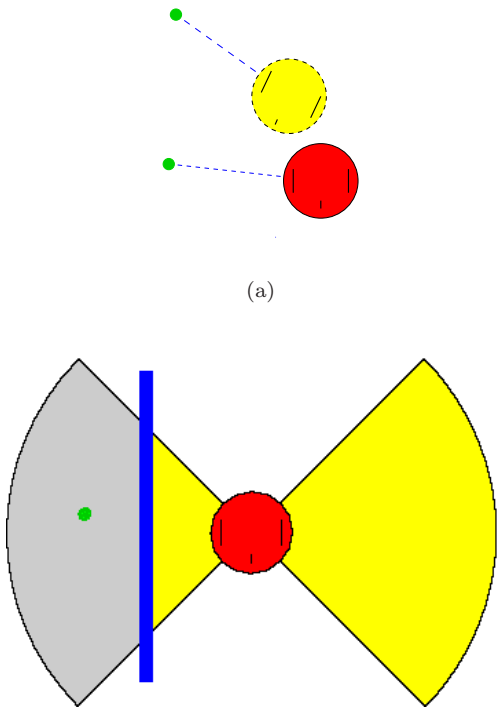


Figure 5.4: This figure indicates which features the algorithm considers detectable by the robot. The features marked with a cross are not taken into account by the algorithm.

However, the final estimate will be consistent. This situation is also uncommon to happen, since the robot can easily validate its position trough other features.

5.3 Recovery from a failure

The initialization procedure (see figure 5.2) is divided into a localization step and a restoration step. The robot must localize itself with respect to the last consistent map collected, and then it has to restore the stochastic mapping algorithm in order to be able to continue with a consistent estimate of the state vector. Considering that the location of the features in the map is modified by the algorithm, only after an update step, a copy of the state vector is saved before the robot performs the updating step and, at the same time a warning flag is set. The robot will continue to move, executing the *stochastic mapping* algorithm. Eventual new features are integrated to the state vector and updating steps due to measurements to these features are normally performed. At this stage, it is still impossible to validate the consistency of the map, the robot could



(a)

(b)

Figure 5.5: The two main mode where the failure detecting algorithm fails.

have started to add new features and to perform measurement of these features inconsistently with respect of the original map. Whenever the robot detects a feature collected before the warning flag was set to 1, the map is considered to be consistent, and the warning flag is unset. At this point, before the next update step, the vector state is saved and the map is assured to be consistent. The algorithm is represented in figure 5.6.

Consistency of the reference map

```
if WarningFlag == 0 and UpdateStep
   Save the state vector x_ref and
   the covariance matrix P_ref
   WarningFlag=1
   MostRecLand=N
else if (WarningFlag == 1 and
        FeatureMatched < MostRecLand)
   WarningFlag=0
```

Figure 5.6: Steps involved for storing $\mathbf{x}_{ref}$ and P_{ref}.

5.3.1 Localization step

The first step consists in an absolute localization by matching recently collected landmarks against a reference map of feature (Wijk and Christensen, 2000). The robot moves performing measurements on a new coordinate system $(x^{(2)}, y^{(2)})$, with the origin chosen in the robot pose when the failure is detected. Consider now the situation where the mobile robot has got a reference map $\chi_{ref}^{(1)} = \{x_{ref,1}^{(1)}, \ldots, x_{ref,N}^{(1)}\}$ and a set of K landmarks recently collected $\chi_{land}^{(2)} = \{x_{land,1}^{(2)}, \ldots, x_{land,K}^{(2)}\}$. The reference map, given in the initial system coordinate $(x^{(1)}, y^{(1)})$, is the reference map $(\mathbf{x}_{ref}, P_{ref})$ stored by the algorithm described in the previous section. The two sets $\chi_{ref}^{(1)}$ and $\chi_{ref}^{(2)}$ are represented in two different coordinate systems (see figure 5.7) $(x^{(1)}, y^{(1)})$ and $(x^{(2)}, y^{(2)})$ the relation between these coordinate systems will be a linear transformation τ involving a rotation and a translation. Given $\chi_{ref}^{(1)}$ and $\chi_{land}^{(2)}$, τ is obtained by solving a graph matching problem. Details of this implementation can be found in (Wijk, 1998). The localization step returns an estimate of the robot state $\hat{\mathbf{x}}_L$ with the corresponding error covariance $\mathbf{P}_L$. At this point the *Stochastic mapping* algorithm can not be restored by simply replacing $\mathbf{x}_{r_{ref}}$ and $P_{rr_{ref}}$ with $\hat{\mathbf{x}}_L$ and $\mathbf{P}_L$, because the resulting covariance matrix can no longer be guaranteed to be positive definite (Feder, 1999). Such replacement, in facts, violates the physical meaning of a covariance matrix and thus violates the consistency of the sub-map. To

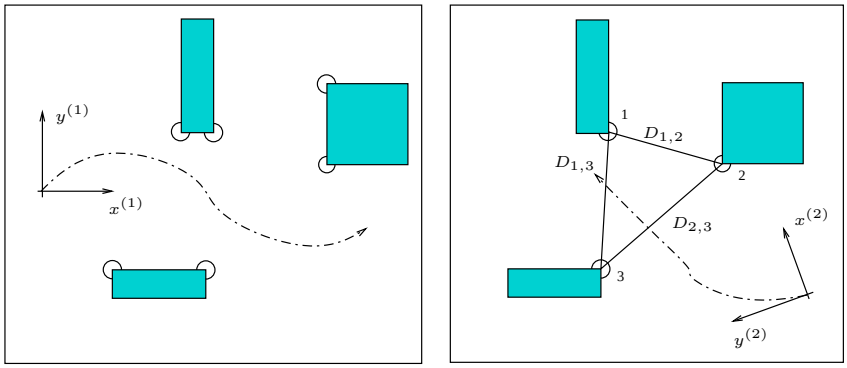


Figure 5.7: Matching of measurements with landmarks in the reference map.

overcome this problem a strategy similar to the one used in (Feder, 1999) is implemented.

5.3.2 Restoration step

The restoration step consists of two sub-steps: de-correlation and updating. In the de-correlation step the robot state estimate of the reference map is randomized, its covariance is highly inflated and the feature covariance is doubled:

$$\mathbf{x}_{ref}[-] \leftarrow \begin{bmatrix} \phi_{ref} \\ \mathbf{x}_{f_{ref}} \end{bmatrix}, \tag{5.1}$$

$$\mathbf{P}_{ref}[-] \leftarrow \begin{bmatrix} \mathbf{P}^{rr}_{ref} + \mathbf{\Phi}_{ref} & \mathbf{P}^{rf}_{ref} \\ \mathbf{P}^{fr}_{ref} & 2\mathbf{P}^{ff}_{ref} \end{bmatrix}, \tag{5.2}$$

where ϕ_{ref} is a random value uniformly distributed over the reference map and $\mathbf{\Phi}_{ref}$ represents a covariance larger than the size of the reference map. The updating step consists of an EKF update using $\hat{\mathbf{x}}_L$ as a measurement with covariance $\mathbf{P}_L$:

$$\mathbf{K} = \mathbf{P}_{ref}[-]\mathbf{H}^T(\mathbf{H}\mathbf{P}_{ref}[-])\mathbf{H}^T + \mathbf{P}_L)^{-1}, \tag{5.3}$$

$$\mathbf{x}[+] = \mathbf{x}_{ref}[-] + \mathbf{K}(\hat{\mathbf{x}}_L - \mathbf{H}\mathbf{x}_{ref}[-]), \tag{5.4}$$

$$\mathbf{P}[+] = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}_{ref}[-](\mathbf{I} - \mathbf{K}\mathbf{H})^T + \mathbf{K}\mathbf{P}_L\mathbf{K}^T. \tag{5.5}$$

where $\mathbf{H}$ is the $3 \times (3+2N)$ matrix $[\mathbf{I}\ \mathbf{0}]$. Ultimately with these steps the robot can restore the stochastic map algorithm. Let's now motivate why this strategy has been used and first consider other possible strategies which are not successful. One idea would be using the EKF update equations 5.4-5.5 considering:

$$\mathbf{x}_{ref}[-] \leftarrow \begin{bmatrix} \hat{\mathbf{x}}_L \\ \mathbf{x}_{f_{ref}} \end{bmatrix}, \tag{5.6}$$

$$\mathbf{P}_{ref}[-] \leftarrow \begin{bmatrix} \mathbf{P}_L & \mathbf{P}_{ref}^{rf} \\ \mathbf{P}_{ref}^{fr} & \mathbf{P}_{ref}^{ff} \end{bmatrix}. \tag{5.7}$$

This strategy maintains positive semi-definiteness, but, it violates the independence assumption of the Kalman Filter and produces overconfident estimates both of the robot pose and of the features state. Performing a vehicle update in this way also updates the state of the features greatly reducing their covariances, even though no new information has been obtained. Thus that's the reason of randomizing the position of the vehicle and multiplying by 2 the $\mathbf{P}_{ref}^{ff}$ before the Kalman update.

5.4 Experiments

An experiment has been performed in the living-room, which shows a situation where the robot performs a recovery from a failure. The robot starts at the position $(3600, 4800)$, see figure 5.8 to explore the unknown environment, during the run we deliberately made an angle perturbation to the robot. As shown in the figure 5.10 the x-position error grows going outside the boundaries. Since the robot doesn't get measurements from the features which are expected to be detected, it detects the failure. Once entering in the failure mode, the robot starts the initialization procedure. At the time iteration 920 (see the correction in figure 5.8a the vehicle performs a localization step, it matches landmarks number 2,12 and 9 (see figure 5.9) on the reference map, with a set of latest measurements stored into a temporary map. In this case the estimation of the robot position and the corresponding covariance supplied from the localization step are:

$$\hat{\mathbf{x}}_L = \begin{bmatrix} 3723 \\ 3677 \\ 4.57 \end{bmatrix}, \mathbf{P}_L = \begin{bmatrix} 55237 & 0 & 0 \\ 0 & 38988 & 0 \\ 0 & 0 & .305 \end{bmatrix}.$$

These data are entered into the equations 5.4-5.5 and the restoration step is completed, then the robot continues with the standard stochastic algorithm. Figure 5.9 shows the estimate of the feature position with the corresponding 2σ uncertainty. The resulting feature map results satisfactory with all the features mapped inside the 2σ bounds. Note that features 5,10,15,16,17 in figure 5.9 correspond to actual objects not included in the CAD model of the living room such as mobile robots and various equipment.

Figure 5.8: Trajectory of the robot in the living room.

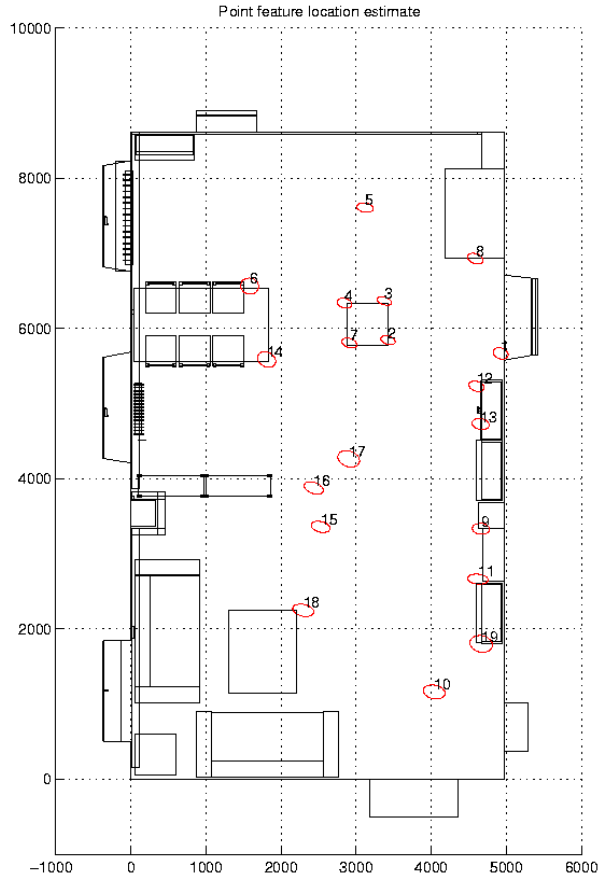
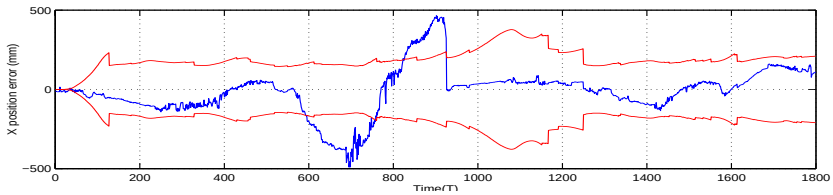
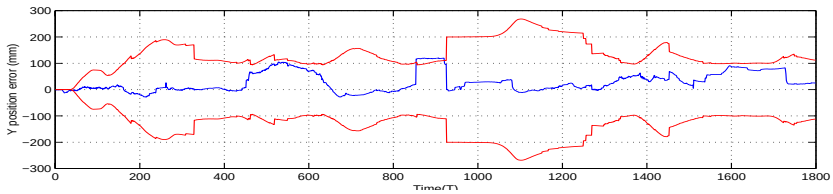


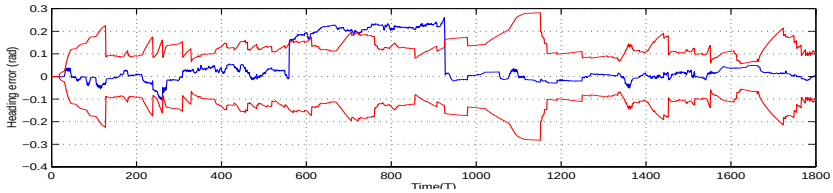
Figure 5.9: Landmarks position estimate with the 2σ covariance ellipses.



(a)



(b)



(c)

Figure 5.10: Error and the 2σ bounds of the x-position, y-position and heading.

5.5 Summary

In this chapter the most common mode of failures of the "stochastic mapping" algorithm were described. A method for recovering from these failures was presented, the problem can be divided in three major steps, detecting the failure, re-localizing the robot using consistent information and restoring the EKF algorithm with a guaranteed consistency. The performance of this method was shown to be robust in a medium size domestic environment.

Chapter 6

Geometric Constraints

This chapter explains how it is possible to use the architectural properties of most domestic environments in order to obtain more accurate navigation. The platform used for the experiments is the autonomous vacuum-cleaner Trilobite from Electrolux.

6.1 Why using architectural constraints

As pointed out in chapter 2, the Trilobite performs a first exploration of the environment, using a wall following behavior (see figure 2.3). During this phase, due to the sensor configuration (see chapter 3), three out of the four microphones available will detect echoes from the wall next to the robot as shown in figure 6.1. This fact will make the wall following behavior very efficient, but, on the other hand, environment mapping and position tracking becomes extremely difficult using point features. The quality and the number of measurements is strongly dependent on the number of sensor available, as explained in chapter 3. Without the possibility of compensating for the odometry drift, in realistic conditions, the estimated trajectory of the robot after the initial run might look like the one shown in figure 6.2. In both the runs the platform starts close to a wall, but it is clear that the drift starts from the very beginning, the robot start sliding over a threshold at the door in the upper left corner of the living-room, and it accumulates further drift over the time. Thus, the final estimate of the robot position is affected by a large error. Furthermore, the estimated positions of the point features collected dur-

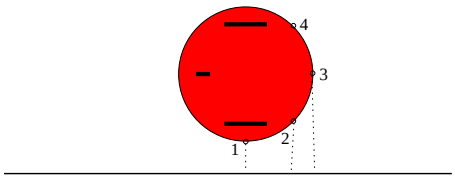


Figure 6.1: Readings from the microphones when the robot is following a wall.

ing this run have rather large errors (see "+'s" in figure 6.3). Using the "stochastic mapping" approach, does not help so much, since the high uncertainty of the robot pose estimate and the feature position estimate would make the data association problem extremely hard to solve. As an example, let us consider the initial run in figure 6.3, in that particular case the uncertainty of the robot is so big that any measurement could be matched to any feature in the map. One idea to improve the initial estimate of the map is to use the knowledge about the behavior of the robot. We know that the robot is following walls, and we know that the vacuum cleaner operates in domestic environments. Being aware of the most common architectural characteristics of regular indoor environment can help to improve the initial map. Another consideration that can be done, is the fact that very often, vases, electric plugs, cables, are placed in the corners of the rooms. Large perturbations in the odometry is therefore more likely to occur in these regions than others.

6.2 Geometric constraints

Let us now examine which are the most common similarities in domestic environment. Most of the rooms in standard apartments have straight walls, the major walls are perpendicular to each other and, typically, the larger pieces of furniture are placed either perpendicular or parallel to the walls. Bookshelves or couches, for example, are very often aligned with the walls. Other furnishing with orientations not compatible with the 90° constraint are usually either small objects or large pieces placed not in the proximity of a wall. An example is a sofa, which is usually placed either at a wall and aligned with it or with random orientation in the middle of the room. In summary, the three most common architectural proprieties of domestic environments are:

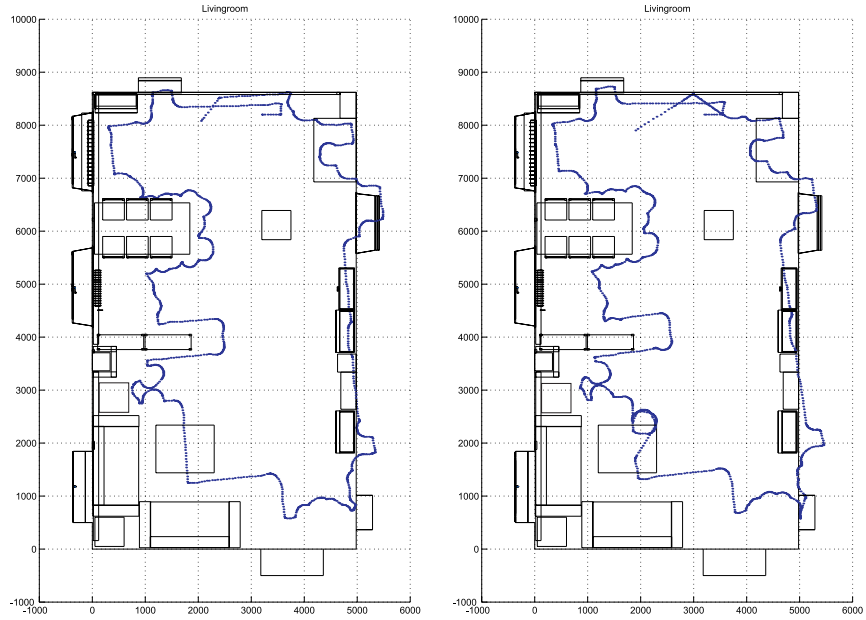


Figure 6.2: Two examples of trajectories for the Trilobite during the wall-following behavior according to the odometry.

- Primary walls are straight.
- Primary walls are either parallel or perpendicular to each other.
- The larger pieces of furniture standing at a wall are aligned with it.

6.3 Compensate for odometry drift using the geometric constraints

Given the observations stated in the previous sections, a method for compensating the odometry drift is presented. Let us first consider the simple example of figure 6.4. The vacuum cleaner get an angular perturbation (5°) in the lower right corner of the room, and the position estimate given by the odometry model (dashed line) accumulates an error with respect

Figure 6.3: Uncertainty of the Trilobite after the wall-following is completed.

to the real robot position (solid line). By using the geometric constrains listed in the previous section, this perturbation can easily be compensated for. The angular drift can be immediately noticed following the right wall by observing the discrepancy, between the 90° constraint and the angle detected using odometry.

6.4 The algorithm

A pseudo code of the algorithm for compensating the odometry drift is shown in figure 6.5. In words the pseudo code can be explained as follow:

1. The *FollowingStraightWall* flag is set when the robot is following a straight wall. This is sensed by checking different parameters. First the readings from the microphones 1 and 2 (see figure 6.1), these reading must fall within appropriate thresholds. Second the differ-

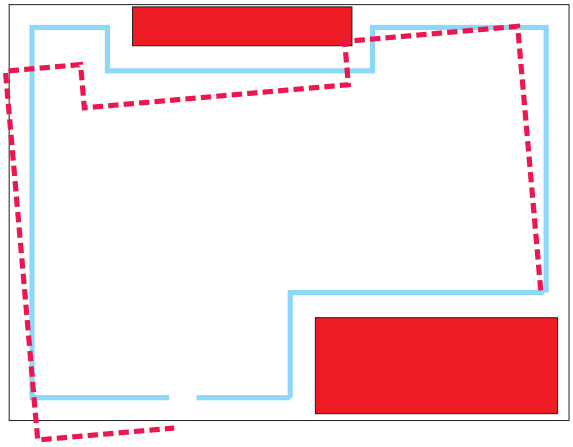


Figure 6.4: Example of the consequences of an angular error during the wall following.

Geometric Constraint Algorithm

```
if FollowingStraightWall
   k=k+1
   if endWallEncountered
      Extract line parameter ℓ
      if (ℓ ≥ minLength)
         Extract line parameters (ρ, ξ)
         if firstLine
            Set RefAngle = ξ
         else if | ξ − RefAngle |≤ angleTh
            Backtrace the robot pose from the
            starting point of the wall compensating
            for the angular error
      k=0
```

Figure 6.5: Steps involved for compensating odometry drift using architectural constraints.

ence between the robot's current orientation and its precedent one $\theta_k - \theta_{k-1}$ must be less than a 3° threshold. This is because we want to be sure that the robot is moving straight, allowing a minimal oscillation in the heading. A final check is done with the orientation of the robot at the beginning of the *FollowingStraightWall* hypothesis $\theta_k - \theta_0$, this threshold is set to 5°. This eliminates the possibility to mistake a slightly curved wall for a straight one. Whenever one of these conditions is not fulfilled the *endWallEncountered* is set and in the following step a new Hypothesis is initiated.

2. A crude estimate of the length of the wall is done by calculating the distance ℓ between the robot position when the *FollowingStraight-Wall* hypothesis was initiated and the robot position where the *endWallEncountered* flag is set.

3. If this distance ℓ is greater or equal than the threshold *minLength* the straight line is validated and associated to either a wall or a larger piece of furniture. The threshold used in the experiments is $0.8m$, which fits very well in the test areas, a more detailed investigation could be made in order to determine the average length of furniture which are placed by the walls.

4. The parameters of the line (ρ, ξ) are the computed by using the Hough Transform using the odometry information.

5. If the line extracted is the first one, the orientation of this wall is used as the reference angle *RefAngle* for future updating. The walls extracted after this one must fulfill the geometric constraints, which means that they should be either parallel or perpendicular to the reference wall.

6. If the extracted line is not the first one the following validation is performed: The angle of the line must be compatible with the geometric constraints. The angle difference :

$$\begin{aligned} \epsilon = \min \Bigg(& \left| \xi - RefAngle \right|, \left| \xi - \left(RefAngle + \frac{\pi}{2} \right) \right|, \\ & \left| \xi - (RefAngle + \pi) \right|, \left| \xi - \left(RefAngle + \frac{3\pi}{2} \right) \right| \Bigg) \end{aligned} \tag{6.1}$$

must be less or equal than a certain threshold *angleTh*. The *angleTh* threshold is not constant, but it is related on the angular

uncertainty of the robot accumulated since the last geometric correction, with the upper bound set to 25°. This option allows the algorithm to work correctly even in presence of walls which are not perpendicular or parallel to the reference one. For example, if a straight line angled more than 25° with respect to the 90° constraint is detected none alignment to the reference wall is performed.

7. If the line extracted fulfills the constraints stated in the previous steps, the trajectory is assumed to be generated by following one of the primary walls of the room or a furniture aligned to one of these walls. The angular error ϵ is assumed to be generated during the robot's rotation performed immediately before the *FollowingStraightWall* flag was set[1]. The pose estimate of the vehicle is, therefore, compensate taking into account ϵ.

One requirement for improving the performance of the algorithm is that the robot starts the wall following behavior in the proximity of a primary wall in order to set the *RefAngle* variable as soon as possible.

6.5 Experimental results

A set of experiments were performed in our living-room. Figures 6.6 and 6.7 show the improvements of the robot's pose estimate using the geometric constraint algorithm with respect to the odometry (figure 6.2). This experiment is performed with the living-room under "nice" condition, without cables on the floor, and with the furniture placed in such a way to avoid difficult situation to the robot. The odometry drift is therefore limited. In figure 6.6 it is pointed out the point-to-point correspondence between the odometry estimate and the geometric constraint algorithm in the points where the "90°" corrections are performed. Figure 6.8 is a zoom of figure 6.6 showing the corrections around a bookshelf in the middle of the living-room. In both the runs, the robot starts the wall following behavior close to the wall on the top of the living-room, this wall is considered by the algorithm as the *"reference"* wall. All the other straight lines detected by the algorithm, must follow the 90° constraint.

[1] This is just an approximation. The angular drift can be occurred earlier.

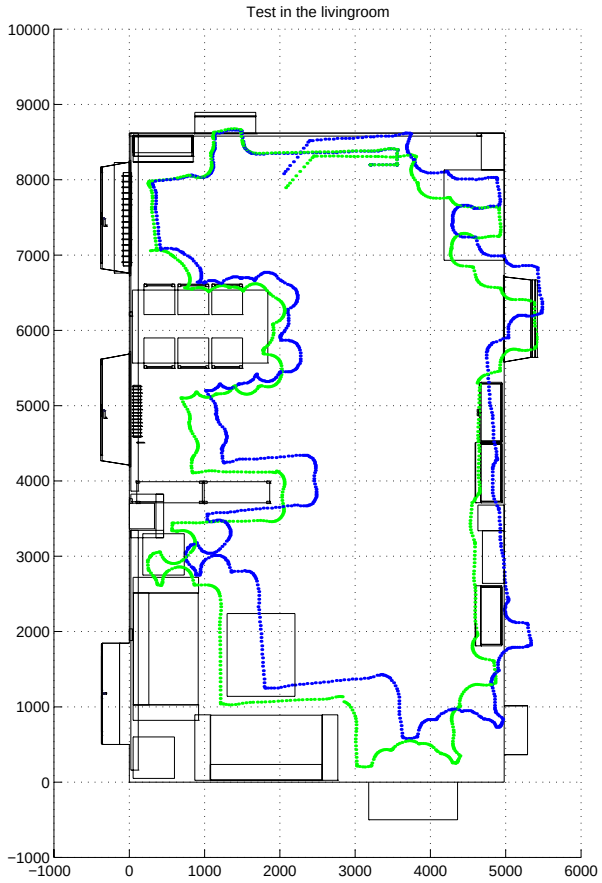


Figure 6.6: First trajectory of the Trilobite during the wall-following behavior correcting the odometry information (darker dots) with the geometric constraints (lighter dots).

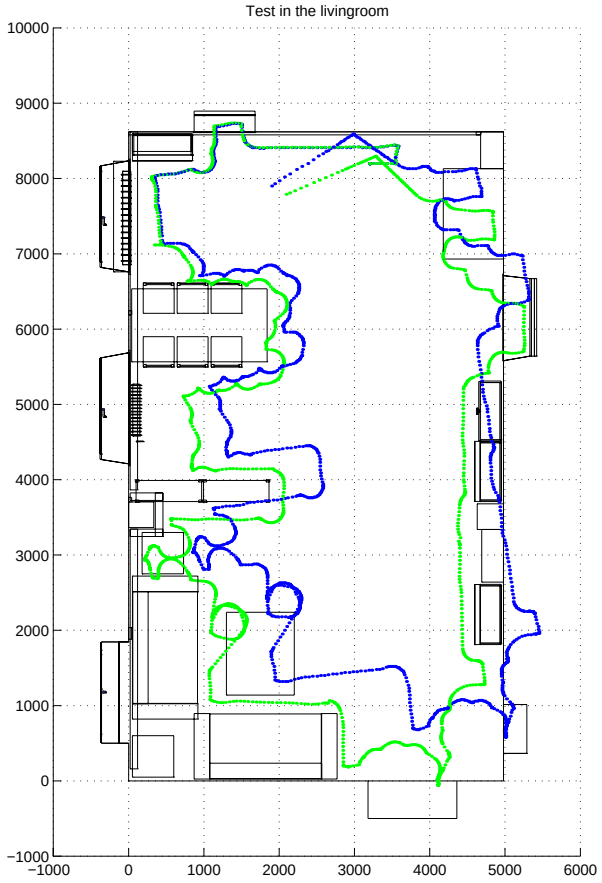


Figure 6.7: Second trajectory of the Trilobite during the wall-following behavior correcting the odometry information with the Geometric Constraints algorithm.

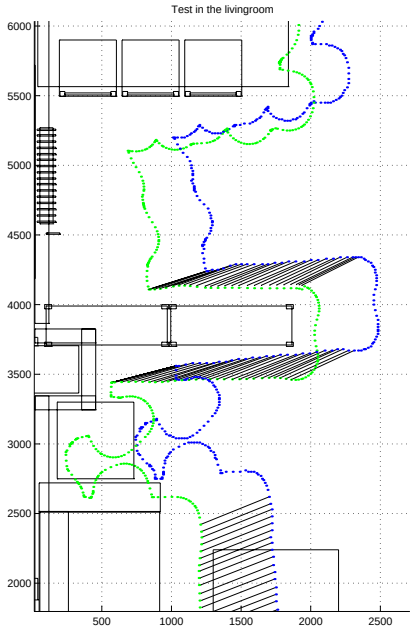


Figure 6.8: Detailed trajectory of the Trilobite around a bookshelf. The darker dots represent the estimate given by the odometry, the lighter dots the position estimate given by the Geometric Constraints algorithm.

Another set of experiments (see figures 6.9 and 6.10) are performed in the living-room under "normal" conditions. No tricks have been done in order to facilitate the robot mission. The chairs at the dining table are not perfectly aligned with the table, and the electric cables of the lamps and other equipment are on the floor. This situation is harder than the previous one since the robot is subjected to bigger slippage, and the odometry alone produces a degraded estimate position of the robot. The pose of the robot, estimated using the geometric constraints is, for both the experiments much better than the estimate supplied from the odometry only.

In figure 6.11 the set of straight lines detected by the algorithm during the experiment of figure 6.9 is shown. The dashed line is the first one detected and its orientation is taken as the *RefAngle*. The other lines are placed according to the constraints imposed by the algorithm.

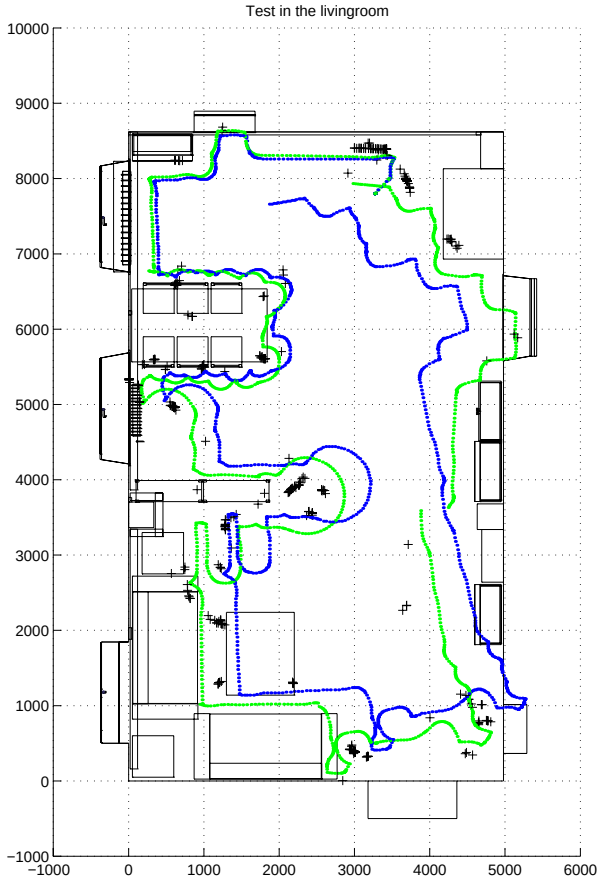


Figure 6.9: First trajectory of the Trilobite during the wall-following behavior in the living-room under "normal" conditions. The odometry estimate is represented by the darker dots, while the algorithm estimate is represented by the lighter dots.

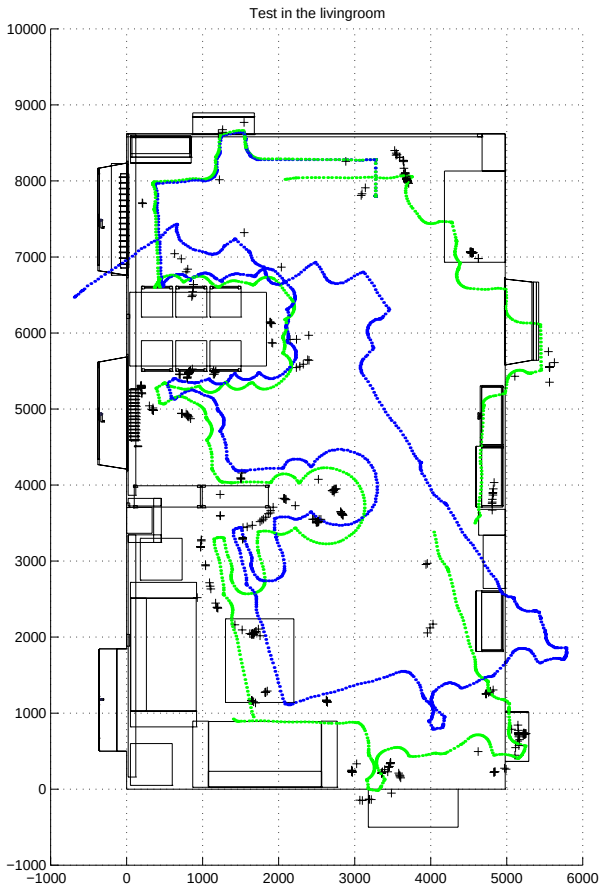


Figure 6.10: Second trajectory of the Trilobite during the wall-following behavior in the living-room under “normal” conditions.

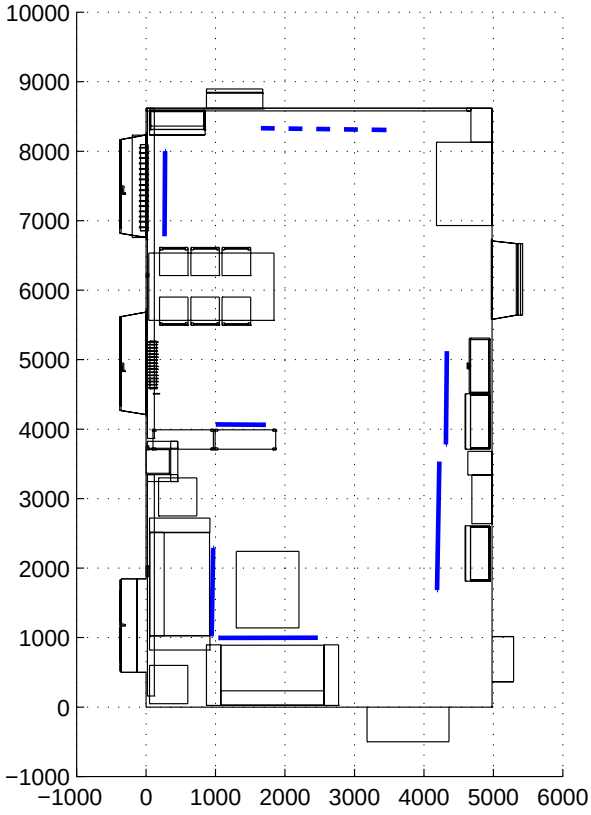


Figure 6.11: Lines features extracted from the Geometric Constraint algorithm during the experiment shown in figure 6.9.

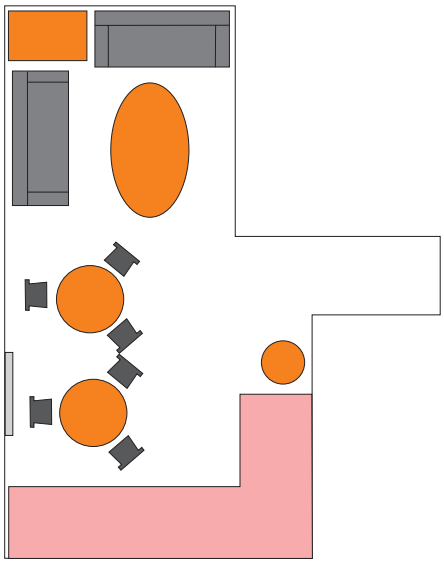


Figure 6.12: Plant of the kitchen.

To prove the robustness of the method another set of experiments has been performed in a different environment. The new environment is the kitchen at the Centre for Autonomous Systems (a sketch of the plant is shown in figure 6.12). In the first experiment, shown in figure 6.13, the robot does not accumulate a big odometry drift, therefore the pose estimate supplied by the algorithm (lighter dots) is very close to the one supplied by the odometry alone (darker dots). In the second experiment the robot gets a perturbation while it is following the edge of the sofa, and, while the error of the odometry estimate gets very big, the pose estimate supplied by the Geometric Constraints algorithm is very satisfactory. Figure 6.14 shows the two trajectory estimates. The line features extracted by the algorithm are shown in figure 6.15.

6.6 Further directions

The *Geometric constraints* concept introduces a set of possible future investigations. Models for more complicated architectural settings can be developed, the user could be given the option to choose between different

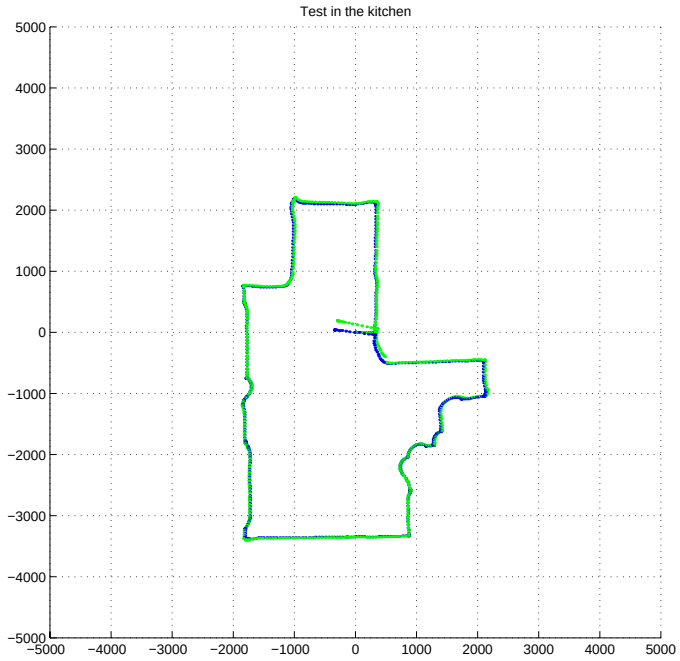


Figure 6.13: First trajectory of the Trilobite during the wall-following behavior in the kitchen. The odometry estimate is represented by the darker dots, while the algorithm estimate is represented by the lighter dots.

kind of domestic environmental styles. For example some room could have a wall inclined 45° with respect to the others, and this could be easily incorporate in the algorithm. Other features, such as circular walls could also be inserted in the algorithm.

Another way to improve navigation, would be using an active strategy. Once terminated the wall following behavior, the vacuum cleaner starts to cover the inside of the room with a random strategy. In this phase due to the low number of point features collected during the initial run and the low range of the sensor, position tracking become very difficult, and the uncertainty of the robot's pose tends to grow rapidly. A way to re-localize the robot, when the uncertainty grows over a definite threshold would be switching to the wall following behavior and recognizing the

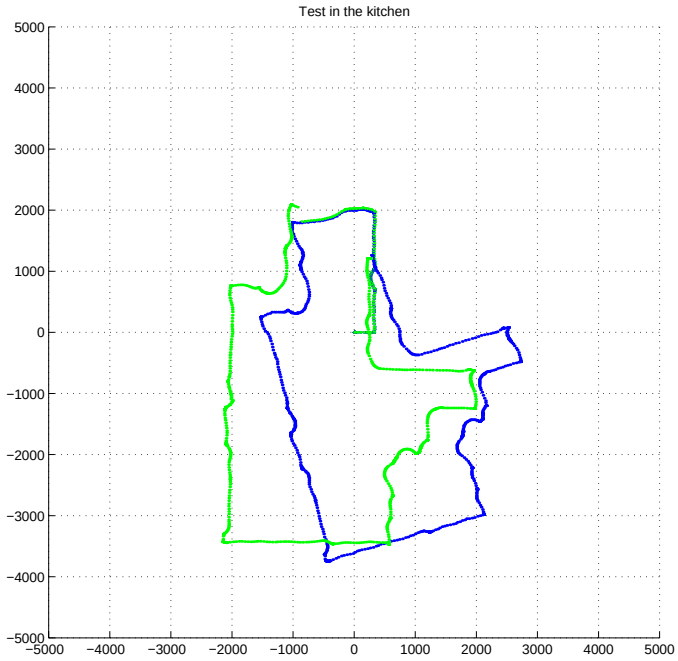


Figure 6.14: Second trajectory of the Trilobite during the wall-following behavior in the kitchen.

walls using the length ℓ and the orientation ξ. In this way localization with respect to the first estimate of the map becomes straightforward.

6.7 Summary

This chapter has presented a method for improving the position estimate of the robot given by odometry using the knowledge of how the most common domestic environments are set up. The *Geometric Constraints* algorithm for detecting straight walls and correcting the robot pose according to the architectural constraints is presented. A set of experiments performed in different conditions, show how this method produces nice results, limiting the use of sensors to the minimum.

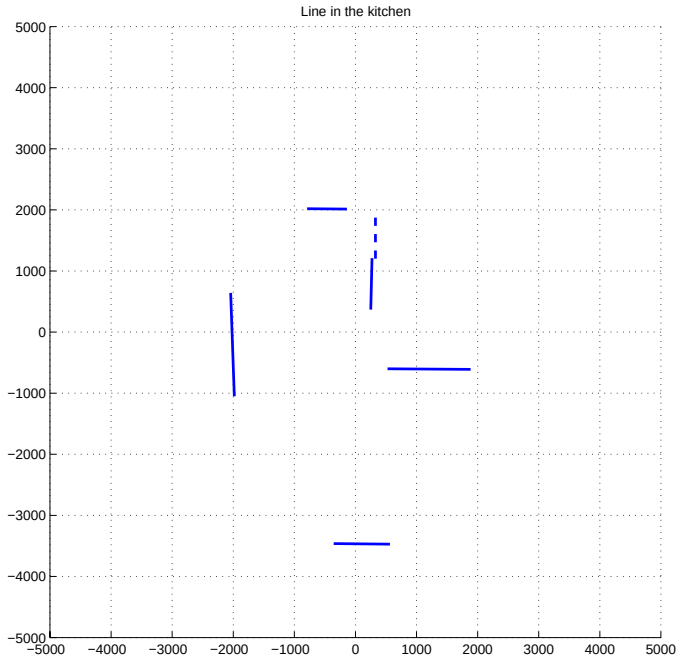


Figure 6.15: Lines features extracted from the Geometric Constraint algorithm during the experiment shown in figure 6.14.

Chapter 7

Conclusions and Directions for Future Research

This thesis presents an application of the feature based SLAM algorithm called *stochastic mapping* on a real robot operating in a domestic environment. Real world experiments show the validity of the algorithm and the capability of the robot to operate in medium size natural indoor environments. The features used by the algorithm are point features originating from vertical edges in the environment extracted with the *Triangulation Based Fusion* technique using sonar data. An interesting topic for future research is to improve performance in large-scale environments by extending the algorithm to also use other kinds of features, for example lines. It is also crucial to improve the data association, which is the most delicate part in the algorithm. The problem of the computational complexity of stochastic maps has not been investigated in the thesis since the environments considered for domestic applications are quite small. If the robot is intended to operate in very large scale environments, it is possible to reduce the computational requirements, methods for achieving this are currently being investigated in the mobile robotics community.

An algorithm for detecting and recovering from the failures of the *stochastic mapping* approach has been presented. Failures of the *stochastic mapping* algorithm can occur frequently in environments where, due to the lack of point features, the uncertainty of the robot pose becomes

very big or in cluttered environments where data association becomes very ambiguous. The algorithm proposed in this thesis enables the system to notice when a failure in the EKF occurs. The *stochastic mapping* algorithm can safely proceed by means of a localization step, where the robot re-localizes itself with respect to the map acquired previously, and a restoration step, where the algorithm restores a consistent estimate of the state vector of the system. Future research should investigate more efficient methods for detecting the failures such as considering multiple hypotheses.

The thesis presents also the *Geometric Constraint* algorithm. Using the knowledge of the most common architectural proprieties of domestic environments, this method allows robotic platforms to achieve accurate navigation with a minimum of sensing capabilities. This allows for robust navigation on inexpensive platforms which can then be mass-produced. More work can be done on expanding the model of a domestic environment to a richer one including different architectural styles and more geometric features. It is also of interest to investigate how the *Geometric Constraint* algorithm can be used in combination with active strategies to improve navigation.

Navigation in realistic environments requires the simultaneous application of many research disciplines. This thesis has focused on the localization and mapping problem using a single sensor modality and a single layer topological representation. The main reason for this choice was the desire to check how far we could get, keeping the cost of the hardware and the complexity of the algorithms low. The results obtained with the two robotic platforms in real world experiments show that more accurate navigation has been obtained. However, there are many open scenarios to investigate, some of those are listed below:

- Topological representation of the environment. A multi-level hierarchy such as, buildings, floors, rooms would enable the robot to perform more complex tasks and to increase flexibility.

- Integration of other sensor modalities can improve navigation. For example adding a gyro to the current system would reduce the odometry drift. This, with a minimum cost increase, would enable the robot to operate more robustly in large environments even when few features are detected. A camera, would also improve the navigation, and, it could also be used for expanding the set of applications of the domestic robot. An example is surveillance, the user could connect to the robot through the web from a remote site and

monitor his/her home, by driving the robot in different locations and looking at the video stream.

- More advanced traversal strategies of the environment, would improve the mapping and navigation problem and would be of extreme value for some applications such as autonomous cleaning. For this specific task the floor's surface must be completely covered by the robot. Random strategies are satisfactory for very small environments. However, for larger areas the navigational pattern has to be structured to achieve satisfactory performance.

- Mobiles robots run with battery power, for long term operations it is, therefore, necessary docking to recharge stations. Methods for automatic docking must be implemented in order to make a domestic robot autonomous.

Bibliography

Akbarally, H. and Kleeman, L. (1995). A sonar sensor for accurate 3d target localisation and classification. In *IEEE Conference on Robotics and Automation*, volume 3, pages 3003–3008.

Althaus, P., Christensen, H. I., and Hoffmann, F. (2001). Using the dynamical system approach to navigate in realistic real-world environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1023–1029.

Arkin, R. C. (1998). *Behavior-Based Robotics*. MIT Press.

Asaithambi, N. S. (1995). *Numeric Analysis. Theory and Practice*. Saunders College Publishing.

Audenaert, K., Peremans, H., Kawahara, Y., and Campenhout, J. V. (1992). Accurate ranging of multiple objects using ultrasonic sensors. *Proc. of the 1992 IEEE Int. Conf. on Robotics and Automation*, pages 1733–1738.

Bar-Shalom, Y. and Fortmann, T. E. (1988). *Tracking and Data Association*. Academic Press.

Barshan, B. and Kuc, R. (1990). Differentiating sonar reflections from corners and planes employing an intelligent sensor. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(6):560–569.

Barshan, B. and Kuc, R. (1991). Bat-like sonar system strategies for mobile robots. *Proceedings of IEEE Conference on Systems, Man and Cybernetics*, pages 905–910.

Borenstein, J. and Feng, L. (1996). Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation*, 12(6):869–880.

Borthwick, S., Stevens, M., and Durrant-Whyte, H. (1993). Position estimation and tracking using optical range data. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2172–2177.

Brooks, R. A. (1989). *The whole iguana*. MIT Press.

Buchberger, M., Jörg, K. W., and von Puttkamer, E. (1993). Laserradar and sonar based world modeling and motion control for fast obstacle avoidanceof an autonomous robot mobot-iv. *Proc. of the IEEE International Conference on Robotics and Automation*, 1:534,540.

Buffa, M., Faugeras, O., and Zhang, Z. (1992). A complete navigation system for a mobile robot, using real-time stereovision and the delaunay triangulation. *Proceedings of MVA92, IAPR Workshop on Machine Vision Applications*.

Burgard, W., Cremers, A. B., Fox, D., Haehnel, D., Lakemeyer, G., Shulz, D., Steiner, W., and Thrun, S. (1999). Experiences with an interactive museum tourguide robot. *Artificial Intelligence*.

Castellanos, J. A. and Tardós, J. D. (1999). *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer.

Castellanos, J. A., Tardós, J. D., and Schmidt, G. (1997). Building a global map of the environment of a mobile robot: The imprtance of correlations. *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1053–1059.

Chatila, R. (1985). Mobile robot navigation: Space modeling and decisional processes. *Proc. of 3rd International Symposium on Robotics Research*, pages 373–378. Gouvieux, France.

Chong, K. S. and Kleeman, L. (1997). Sonar based map building in large indoor environments. Technical report, Department of Electrical and Computer Systems Engineering,Monash University.

Csorba, M. and Durrant-Whyte, H. (1997). A new approach to map building using relative postion estimates. *Proc. of SPIE*, pages 115–125.

Csorba, M., Uhlmann, J. K., and Durrant-Whyte, H. (1997). A sub optimal algorithm for automatic map building. *Proc. of the American Control Conference*, pages 537–541.

Dissanayake, G., Durrant-whyte, H., and Bailey, T. (2000). A computationally efficient solution to the simultaneous localization and map building (slam) problem. *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1009–1014.

Duckett, T. and Nehmzow, U. (2001). Mobile robot self-localisation using occupancy histograms and a mixture of gaussian location hypotheses. *J. Robotics and Autonomous Systems*, pages 119–130.

Durrant-Whyte, H. F. and Dissanayake, M. W. M. G. (1999). Toward deployment of large scale simultaneous localisation and map building. *Control Theory and Applications, IEEE Proceedings*, 142:385–400.

Elfes, A. (1987). Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249–265.

Feder, H. J. S. (1999). *Simultaneous Stochastic Mapping and Localization*. PhD thesis, Massachuset Institute of Technology.

Feder, H. J. S., Leonard, J. J., and Smith, C. M. (1988). Adaptive concurrent mapping and localization using sonar. *Proc. IEEE Int. Workshop on Intelligent Robot and Systems*.

Forsberg, J., hman, P. A., and Wernersson, A. (1993). The hough transform inside the feedback loop of a mobile robot. *Proc. of the IEEE International Conference on Robotics and Automation*, 1:791–798.

Fox, D., Burgard, W., and Thrun, S. (1999). Markov localization for mobile robots indynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427.

Gelb, A. C. (1973). *Applied Optimal Estimation*. The MIT Press.

Guttman, J.-S., Burgard, W., Fox, D., and Konolige, K. (1998). An experimental comparison of localization methods. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2:736–743.

Heale, A. and Kleeman, L. (2000). A real time dsp sonar echo processor. In *In Proc IEEE/RSJ International Conference on Intelligent Robots and Systems*, Takamatsu, Japan.

Hoppen, P., Knieriemen, T., and von Puttkamer, E. (1990). Laser-radar based mapping and navigation for an autonomous mobile robot. *Proc. of the IEEE International Conference on Robotics and Automation*, 2:948–953.

Hough, P. (1962). A method and means for recognizing complex patterns. U.S. Patent, Number 3,069,654.

Illingworth, J. and Kittler, J. (1988). A survey of the hough transform. *Computer Vision, Graphics and Image Processing*, 43:87–116.

Jensfelt, P. (1999). Localization using laser scanning and minimalistic environmental models. Licentiate Thesis, Automatic Control Dep, Royal Institute of Technology, Stockholm, Sweden.

Jensfelt, P. (2001). *Approaches to Mobile Robot Localization in Indoor Environments*. PhD thesis, Automatic Control Dep, Royal Institute of Technology, Stockholm, Sweden.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transaction of the ASME-Journal of Basic Engineering*, pages 35–45.

Kleeman, L. (1992). Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead-reckoning. In *IEEE Conference on Robotics and Automation*, volume 3, pages 2582–2587.

Kleeman, L. (1999). Fast and accurate sonar trackers using double pulse coding. In *In Proc IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1185–1190, Kyongju, Korea.

Kleeman, L. and Kuc, R. (1994). An optimal sonar array for target localization and classification. In *IEEE Conference on Robotics and Automation*, volume 4, pages 3130–3135.

Kuc, R. and Barshan, B. (1992). Bat-like sonar for guiding mobile robots. *IEEE Control Systems Magazine*, pages 4–12.

Leonard, J. and Durrant-Whyte, H. (1991). Simultaneous map building and localization for an autonomous mobile robot. *Proc. of the International Workshop on Intelligent Robots and Systems*, 3:1442–1447.

Leonard, J. and Durrant-Whyte, H. (1992). *Directed Sonar Sensing for Mobile Robot Navigation*. Boston: Kluwert Accademic Publisher.

Mataric, M. J. (1990). Environment learning using a distribuited representation. *Proc. IEEE Int. Conf. Robotics and Automation*, page 402.

Mataric, M. J. (1992). Integration of representation into gloal-driven behavior-based robots. *IEEE Trans. Robotics and Automation*, 8(3):304–312.

Mataric, M. J. (1997). Behavior-based control: Examples from navigation, learning and group behavior. *Journal of Experimental and Theoretical Artificial Intellicence*, 9(2-3).

McKerrow, P. and Zhu, S. (1996). Modelling multiple reflection paths in ultrasonic sensing. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 284–291.

Mendle, J. (1985). *Computational requirements of a discrete Kalman Filter*. Kalman Filtering: Theory and Application. IEEE Press.

Moravec, H. P. (1983). The stanford cart and the cmu rover. *IEEE Transaction on Industrial Electronics*.

Moravec, H. P. and Elfes, A. (1985). High resolution maps from wide angle sonar. *Proc. IEEE Int. Conf. Robotics and Automation*.

Moutarlier, P. and Chatila, R. (1989). Stochastic multi-sensory data fusion for mobile robot location and environment modeling. *5th Int. Symposium on Robotic Research*, pages 207–216.

Murray, D. and Jennings, C. (1997). Stereo vision based mapping and navigation for mobile robots. *Proc. of the IEEE International Conference on Robotics and Automation*.

Peremans, H. (1994). A maximum likelihood algorithm for solving the correspondence problem in tri-aural perception. *IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 485–492.

Smith, R. C. and Cheesman, P. (1987). On the representation of spatial uncertainty. *Int. J. Robotics Research*, 5(4):56–68.

Stanley, B. and McKerrow, P. (1997). Measuring range and bearing with a binaural ultrasonic sensor. *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 565–571.

Stewart, W. K. (1996). Three-dimensional stochastic modelling using sonar sensing for undersea robotics. *Autonomous Robots*, 3:121–143.

Taylor, C. and Kriegman, D. (1995). Vision-based motion planning and exploration algorithms for mobile robots. *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*.

Thau, R. S. (1997). *Reliably mapping a robot's environment using fast vision and local, but not global, metric data*. PhD thesis, Massachusetts Institute of Technology.

Thrun, S., Fox, D., and Burgard, W. (1998a). A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53.

Thrun, S., Gutmann, J.-S., Fox, D., Burgard, W., and Kuipers, B. J. (1998b). Integrating topological and and metric maps for mobile robot navigation: A statistical approach. *AAAI-98*.

Uhlmann, J. K. (1995). *Dynamic Map Building and Localization: New Theoretical Foundations*. PhD thesis, University of Oxford, Robotics Research Group, Departement of Engineering Science.

Uhlmann, J. K., Julier, S. J., and Csorba, M. (1997). Nondivergent simultaneous map building and localisation using covariance itersection. navigation and control technology for unmanned systems ii. Technical report.

Weiss, G. and von Puttkamer, E. (1995). A map based on laserscans without geometric interpretation. *Intelligent Autonomous Systems*, pages 403–407.

Welch, G. and Bishop, G. (1995). An introduction to the kalman filter. Technical Report TR95-041, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, USA.

Wijk, O. (1998). Navigation of mobile robots using natural landmarks extracted from sonar data. Licentiate Thesis, Automatic Control Dep, Royal Institute of Technology, Stockholm, Sweden.

Wijk, O. (2001). *Triangulation Based Fusion of Sonar Data with Application in Mobile Robot Mapping and Localization*. PhD thesis, Automatic Control Dep, Royal Institute of Technology, Stockholm, Sweden.

Wijk, O. and Christensen, H. (1998). Extraction of natural landmarks and localization using sonars. *International Symposium on Intelligent Robotic Systems*, pages 231–240.

Wijk, O. and Christensen, H. (2000). Localization and navigation of a mobile robot using natural point landmarks extracted from sonar data. *Journal of Robotics and Autonomous Systems 31*, pages 31–42.

Yamauchi, B. and Langley, P. (1997). Place recognition in dynamic environments. *Journal of Robotic Systems*, 14(2):107–120.