HYBRID, METRIC - TOPOLOGICAL, MOBILE ROBOT NAVIGATION

THÈSE N^{O} 2444 (2001)

PRÉSENTÉE AU DÉPARTEMENT DE MICROTECHNIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES TECHNIQUES

PAR

Nicola TOMATIS

ingénieur informaticien diplômé EPF
de nationalité suisse et orginaire d'Arvigo (GR)

acceptée sur proposition du jury:

Prof. R. Siegwart, directeur de thèse
Prof. J. Crowley, rapporteur
Prof. D. Floreano, rapporteur
Prof. I. Nourbakhsh, rapporteur
Dr S. Vestli, rapporteur

Lausanne, EPFL
2001

Acknowledgments

"Knowledge is in the end based on acknowledgment."

Ludwig Wittgenstein (1889–1951)

I would like to thank my supervisor, Prof. Roland Siegwart, for making this thesis possible. He provided support and guidance, while ensuring freedom during the whole work. I would also like to thank Prof. James Crowley, Prof. Dario Floreano, Prof. Illah Nourbakhsh and Dr. Sjur Vestli for their valuable comments and for accepting to be my co-examiners.

Thanks to Philippe Cattin and Dr. Nadine Tschichold who were my supervisors during my diploma thesis in Zurich. They helped me to believe in my skills and pushed me to make this Ph.D.

I also want to thank Kai Arras for having helped me to start so quickly in the field of mobile robotics. Furthermore, I am indepted to Prof. Illah Nourbakhsh who has become my second supervisor over the last one and half years. He believed in my ideas and guided me to the current results.

Thanks also to Roberto Brega at IfR, ETH Zurich for his consulting and support about the XO/2 operating system.

Special thanks to all the ASL team: They have become my second family since I have started my Ph.D. in Lausanne.

I wish to express my gratitude towards the Board of the Swiss Federal Institute of Technology (BSIT) for their financial support throughout the last three years.

Last but not least, thanks to my family and my friends for their constant support: Life is easier when you know there is always someone ready for you.

For this, I want especially to thank my parents Piero and Rina, if I am here now, it is firstly because of your help; my sister Laura, I know it is difficult to see each other more often, but you are always with me; my brother Luca and his wife Monica, Alessandro is the most beautiful gift you could make to me.

Finally, thanks to my love Eliane.

Abstract

"Logicians may reason about abstractions. But the great mass of men must have images. The strong tendency of the multitude in all ages and nations to idolatry can be explained on no other principle."

Thomas Babington Macaulay (1800–1859)

This thesis presents a recent research on the problem of *environmental modeling* for both *localization* and *map building* for wheel-based, differential driven, fully autonomous and self-contained mobile robots. The robots behave in an indoor office environment. They have a multi-sensor setup where the encoders are used for odometry and two exteroperceptive sensors, a 360° laser scanner and a monocular vision system, are employed to perceive the surrounding.

The whole approach is feature based meaning that instead of directly using the raw data from the sensor features are firstly extracted. This allows the filtering of noise from the sensors and permits taking account of the dynamics in the environment. Furthermore, a properly chosen feature extraction has the characteristic of better isolating informative patterns. When describing these features care has to be taken that the uncertainty from the measurements is taken into account.

The representation of the environment is crucial for mobile robot navigation. The model defines which perception capabilities are required and also which navigation technique is allowed to be used. The presented environmental model is both metric and topological. By coherently combining the two paradigms the advantages of both methods are added in order to face the drawbacks of a single approach. The capabilities of the hybrid approach are exploited to model an indoor office environment where metric information is used locally in structures (rooms, offices), which are naturally defined by the environment itself while the topology

of the whole environment is resumed separately thus avoiding the need of global metric consistency.

The hybrid model permits the use of two different and complementary approaches for localization, map building and planning. This combination permits the grouping of all the characteristics which enables the following goals to be met: *Precision*, *robustness* and *practicability*. Metric approaches are, per definition, precise. The use of an *Extended Kalman Filter* (EKF) permits to have a precision which is just bounded by the quality of the sensor data. Topological approaches can easily handle large environments because they do not heavily rely on dead reckoning. Global consistency can, therefore, be maintained for large environments. Consistent mapping, which handle large environments, is achieved by choosing a topological localization approach, based on a *Partially Observable Markov Decision Process* (POMDP), which is extended to simultaneous localization and map building.

The theory can be mathematically proven by making some assumptions. However, as stated during the whole work, at the end the robot itself has to show how good the theory is when used in the real world. For this extensive experimentation for a total of more than 9 km is performed with fully autonomous self-contained robots. These experiments are then carefully analyzed. With the metric approach precision with error bounds of about 1 cm and less than 1 degree is further confirmed by ground truth measurements with a mean error of less than 1 cm. The topological approach is successfully tested by simultaneous localization and map building where the automatically created maps turned out to work better than the *a priori* maps. Relocation and closing the loop are also successfully tested.

Version Abrégée

"Les logiciens pensent peut-être avec abstraction, mais la majorité des hommes a besoin d'images. La tendance d'une multitude d'âge et nations à l'idolâtrie ne peut être expliquée avec aucun autre principe."

Thomas Babington Macaulay (1800–1859)

Cette thèse présente une récente recherche dans le domaine de la *modélisation d'environnement* pour la *localisation* et la *construction de cartes* pour des robots mobiles complètement autonomes avec roues et *differential drive*. Les robots se déplacent dans un environnement typique de bureaux. Ils ont un système avec plusieurs capteurs: les encodeurs sont employés pour l'odomètrie et les deux capteurs exteroperceptifs, un scanneur laser et un système de vision monoculaire, sont utilisés pour percevoir les environs.

L'approche présentée est basée sur des *features*. Cela signifie que, au lieu d'utiliser directement les données brutes des capteurs, on extrait d'abord des caractéristiques de l'environnement. Cela permet de filtrer le bruit en provenance des capteurs et de faire face aux changements dynamiques dans l'environnement. Dans la représentation de ces *features* il ne faut pas oublier de tenir compte de l'incertitude des mesures.

La représentation de l'environnement est cruciale pour la navigation de robots mobiles. Non seulement elle définit quelle perception sera nécessaire, mais, en plus, elle détermine quelle technique de navigation pourra être employée. Le modèle présenté ici est métrique et topologique. En combinant de façon cohérente les deux paradigmes, les avantages des deux méthodes peuvent être combinés pour faire face aux défauts de chaque approche. Les qualités du système hybride

sont employées pour modéliser un environnement de bureaux, où l'information métrique est utilisée localement dans des structures qui sont définie naturellement par l'environnement (pièces, bureaux), tandis que la topologie globale de l'environnement est résumée séparément, permettant ainsi d'éviter de devoir maintenir la consistance métrique globale.

Le modèle hybride permet d'utiliser deux approches différentes et complémentaires pour la localisation, la construction de cartes et la planification. Cette combinaison permet de regrouper toutes les caractéristiques qui permettent d'arriver aux buts définis: *précision*, *robustesse* et *faisabilité*. L'utilisation d'un *filtre de Kalman* permet d'avoir une précision, qui est seulement limitée par la qualité des données des capteurs. Les approches topologiques peuvent faire face sans problèmes à des grands environnements parce qu'ils nécessitent pas de l'odomètrie. La consistance globale est donc plus facile à maintenir. Une création de cartes consistantes est obtenue en utilisant une approche topologique basée sur les *partially observable markov decision processes* (POMDP), qui est étendue pour la création de cartes.

La théorie proposée peut être prouvée mathématiquement en acceptant certaines suppositions. Cependant, le but final est que ce soit le robot qui nous montre les qualités de l'approche. Pour cela, une vaste expérimentation, pour un total de plus que 9 km, est faite avec des robots complètement autonomes. Ces expériences sont étudiées en détail pour en tirer des conclusions correctes. Avec l'approche métrique, la précision, obtenue en estimant la faute moyenne avec les *error bounds*, est de 1 cm et moins d'un degré. Elle est confirmée par des mesures de *ground truth* qui montrent une faute de moins d'un cm en moyenne. L'approche topologique a été testée avec succès, où même des taches comme la relocalisation et la fermeture de boucles dans l'environnement marchent correctement.

1

Introduction

"Imagination is more important than knowledge."

Albert Einstein (1879–1955)

This dissertation addresses fundamental questions concerning perception, environmental modeling and navigation for indoor mobile robots. The introduction gives a perspective of the presented work relative to the field of mobile robotics by starting from the definition of *robot* and then focusing on today's problems emphasizing on those addressed in this work.

1.1 The History of Robots

Since the presentation of the play *R.U.R.* (Rossum's Universal Robots) in 1920 the word *robot* entered the English language. The origin of the word is found in the Eastern European languages: Firstly, in Czech *robota* means compulsory labor or drudgery, but also the Old Church Slavonic word *rabota* meaning servitude and *rabî* meaning slave, have forged the English translation *robot*. R.U.R. written by the best known literally figures of liberated Czechoslovakia, Josef and Karel Capek, coined the word *robot* as meaning "mechanical man" for the first time. The play conceives a future in which all workers will be automated. Their ultimate revolt when they acquire souls and the ensuing catastrophe results in an exciting, vivid theatrical experience.

The concept of robot is therefore born with the oldest cliches of the science fiction field: The *Rogue Robot Plot*. From medieval stories of the Golem through Mary Shelley's *Frankenstein* the story is the same: A mad scientist works obsessively to create an artificial man ignoring the dark forebodings of his nearest and dearest. Once the artificial man is created it quickly escapes the mad scientist's

control, destroying him and possibly others as well before finally being destroyed itself. With rare exceptions this plot was repeated in every robot story published in the science fiction magazines in the 1930's and is still a theme in contemporary films including *2001: A Space Odyssey* (1968), *Blade Runner* (1982) and *Terminator* (1984).

Isaac Asimov represents the first revolution in *robotics* (term coined by Asimov in 1942). Asimov came to detest the Rogue Robot Plot for a number of reasons. From a technical standpoint he found it unbelievable that a robot would be constructed without built-in safeguards as other machines were. From a literary standpoint he grew weary of seeing the same plot repeated ad nauseam. Philosophically he disagreed with the Rogue Robot Plot's theme that *There Are Some Things Man Was Not Meant to Know*. Asimov though it was obvious that the principles governing the robot were of first importance for the security of the human being. Given this it was inevitable that once Asimov began writing for publication in the 1940's, he would write robot stories of his own for the specific purpose of attacking the Rogue Robot Plot. This led also to the well known *Three Laws of Robotics*:

1. A robot may not injure a human or allow a human to be injured.
2. A robot must follow any order given by a human that does not conflict with the First Law.
3. A robot must protect itself unless that would conflict with the First or Second Laws.

To evidence the importance of the computational principles, in the *Runaround* novel, Asimov describes how subtle definitions within the programs which control a robot lead to significant changes in the robot's overall behavior.

The next revolution, which is of more interest for this text, comes at the end of the 1960's when the first vehicles controlled primarily by programs which reasoned were built. Stanford Research Institute robot's, Shakey, is the first one. Between 1966 and 1972 the Shakey project faced basic problems of mobile robotics such as obstacle avoidance, object recognition and map building in adapted (very simple) environments by using video processing. Fantasy and dreams started to be faced by scientists. What has been created on paper in a few years by some writers has been discussed and studied 30 years by the research community. The success of artificial intelligence research in the 1960's and 1970's has inspired expectations in the domain of mobile robotics. But nature has shown that life (survival) has always been a sensory and then a reasoning problem. After some years of research it has been clear that it is comparatively easy to make computers exhibit adult-level performance in solving problems on intelligence tests, but it is difficult or even impossible to give them the skills of a one-year-old child when it comes to perception and mobility.

Hans Moravec, one of the pioneers of mobile robotics in the 1970's, was exactly of this opinion when he wrote his book: *Mind Children, The Future of Robot and Human Intelligence* in 1988 [Moravec88]. However, even by knowing this, he expected human-like performance in mobile robotics by the end of the last millennium. Now a new millennium has started and we know that Moravec made a mistake because there is still some work to do before achieving this goal.

1.2 Perception and Uncertainties

The interaction between the mobile robot and its surroundings is performed by means of exteroperceptive sensors. The most popular are ultrasonics sensors, vision systems and laser based devices. Ultrasonics were a very popular perception system in the 1970's and 1980's, primarily because of their low cost and easy integration. They are based, like laser devices, on a *time-of-flight* principle. The disadvantage when using such a sensor is the large beam angle which leads to a low angular resolution. CCD cameras are still popular because of their low cost but they always had limited success on autonomous systems due to the complexity in treating the vast amount of information. Today increasing processing power allows results which can be classified as interesting. Cameras can also give distance information when coupled in a stereo vision head, but in this case the complexity in processing is almost unacceptable with respect to other range sensors. In the last decade the sensors which have gained more acceptance in the research and industrial communities are laser based devices. Often equipped with a rotating mirror they allow the measurement of distances in a plane. The success of this type of sensors is mainly due to the decreasing cost and the high accuracy. In the last few years they have also been used for 3D laser measurements.

Measurements are, per definition, uncertain. By knowing the physical characteristics of the employed sensors uncertainties can be isolated, modeled and propagated up to the application level. This allows the combination of the use of probability theory to represent the uncertainty of the geometric elements of the environment and of the robot as well. Furthermore, probabilistic position estimators have turned out to be more reliable than those relying only on the first moment.

Perception always starts by acquiring raw data from the sensors. With this, features of different levels of perceptual abstraction can be extracted and used for the navigation of the mobile vehicle. Actually raw data can be directly used and have the advantage of being as general as possible. But, as with most sensors, they are credible only by processing great amounts and deliver low information. Navigation based on geometric features allow for compact and precise environmental models. However, the existence of these features becomes indispensable which represents a limitation of environment types. This can be viewed as a loss of

robustness but can be diminished by simultaneously employing geometric features from different sensors with complementary properties. Continuing in this direction, high level features such as corners, cross way or even doors and windows can be extracted combining more sensors data or by means of complex extraction schemes. This surely limits the environment type but permits a very compact and precise scene description and can be advantageous when a simple environment topology is helpful for the application. When the percepted features allows the definition and detection of the environment topology it is doubtful to talk about limitations of the approach.

1.3 Metric and Topological Environmental Modeling

Perceiving the environment remains a fundamental task for autonomous mobile systems. More precisely, localization and mapping in an unmodified environment belongs to the basic skills for mobile robot applications. In many potential service tasks the vehicle is operating in a structured or semi-structured surrounding. This property can be exploited by using the structures as frequently and reliably recognizable features for navigation. Topological, metric or hybrid navigation schemes can make use of different types of environment features on various levels of perceptual abstraction leading to different environmental models.

Current research has therefore diverged to different approaches: Metric, topological or hybrid navigation schemes have been proposed and studied. Approaches using purely metric maps are vulnerable to inaccuracies in both map-making and dead-reckoning abilities of the robot. Even by taking into account all relationships between features and the robot itself, the drift in the odometry makes the global consistency of the map difficult to maintain in large environments. Landmark-based approaches, which rely on the topology of the environment, can handle this problem better because they only have to maintain topological global consistency, not metric consistency. However, these approaches are either less precise than fully metric approaches, due to the discretization of the localization space, or computationally intractable for fully autonomous robots when fine grained grids are used. More recently approaches combining the topological and the metric paradigm (mainly grid-based) have shown that positive characteristics of both can be integrated to compensate for the weakness of each single approach.

1.4 Simultaneous Localization and Map Building

Another important issue in mobile robotics is map building. This is due to the fact that a priori maps are rarely available and, even when given, not in the format required by the robot. Furthermore, they are mainly unsatisfactory due to imprecision, incorrectness and incompleteness. Therefore, map building is not only a

desire which automates a work which would have to be performed by hand, it is instead a real need for real-world applications.

This task has high complexity because it requires the robot to be localized with respect to the portion of the environment which has already been mapped in order to build a coherent map. Some works have been conducted reducing the complexity by assuming a perfect odometry [Matthies88] but that is not an acceptable assumption for real applications.

In this case too metric, topological or hybrid navigation schemes have been proposed and studied. The first metric approach that has been published with a precise mathematical basis is the *stochastic map* [Smith88]. In this approach the state vector is considered as a non-separable entity containing the spatial relations and correlations between all the map features and the mobile robot. This approach suffers from instabilities and is NP-complete: The complexity explosion is exponential with respect to the number of features detected in the environment. However, some heuristics have been proposed in order to reduce the complexity in the covariance matrix [Leonard92]. But this can lead to problems which have been explicitly shown in [Wullschleger99] for the case where correlations are neglected.

Grid/place based approaches reduce the complexity, which remains exponential, but grows with the number of cells/places instead of the number of features in the environment. Nevertheless, as explained in Section 1.3, the grid approach does not allow the same precision as pure metric methods.

1.5 About this Work

In this section both the motivations and the contributions of this thesis are briefly presented. Then, as an extension of the contribution, the robots used during the experiments are presented with their main characteristics. Some explanations are given describing the operating system because it is an important tool for achieving the characteristics required by applications ready embedded systems. The last part of the section presents the structure of the thesis.

1.5.1 Problem Statement and Contributions

For localization and also for map building many approaches have been proposed. However, mobile robots navigating in unmodified human environments are very rare and almost no autonomous mobile vehicles are used in such environments for real applications. This means that the problem of indoor navigation is not yet solved in its whole entirety.

The contribution of this thesis is to develop a new navigation solution which has to fulfill the following characteristics:

- *Practice*. The theory has to be validated by extended empirical experiments with real robots in real environments.
- *Robustness*. The robot navigation has to require no human intervention (no lost situations, automatic exception handling, ...). Assuming that no collisions occur is unacceptable. The robot has to be allowed to collide and if it is not physically damaged it has to recover from such a situation. Approaches which work only in static environments are just academic games. The real environment is dynamic.
- *Precision*. The robot has to be as precise as possible. Its precision can just be bounded by the precision of the sensors, but not by the approach. With the current sensors this should allow for most of the typical applications (navigation in narrow environments, docking, ...)
- *Practicability*. The proposed approach has to work on-board, on-line with the limited computing and memory resources of the robots.

These characteristics are present in the whole thesis, where details concerning the implementation and the experiments have almost the same weight as the theory.

1.5.2 The robot systems

The three robots used in this work have been designed and built or partially build at the Autonomous Systems Lab, EPFL. They mainly have the same characteristics but differ in the details. Pygmalion is the first autonomous ASL robot. It has been designed and build by Kai Arras directly at ASL starting in 1997. Donald and Daffy Duck have been designed at ASL, but built externally by MRS, a spin-off of the Institute of Robotics at the Swiss Federal Institute of Technology, Zurich. They were delivered to the ASL at end of 1999.

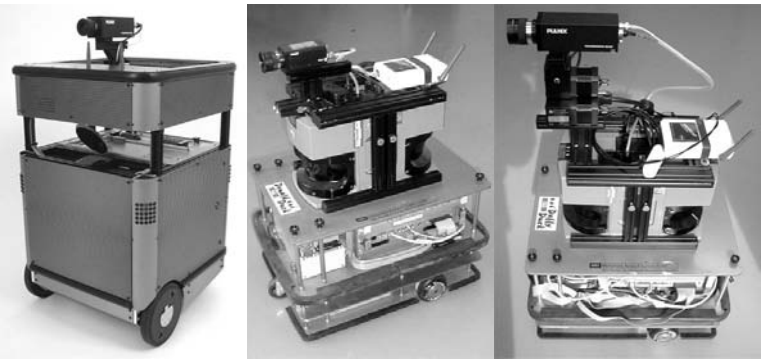

Figure 1.1: The three robots at the Autonomous Systems Lab: Pygmalion, Donald Duck and Daffy Duck.

1.5.2.1 Hardware Characteristics

Pygmalion

- MVME230x PowerPC 604 300 MHz VME Motorola card.
- VME-based six axis controller with IP-modules for analogue, digital I/O and encoder inputs.
- EC-motors with 1:100 harmonic drives with an encoder resolution of 50,000 pulses per revolution.
- Acuity AccuRange4000 laser scanner with a full view on the environment (minus four blind zones from the vertical supports of the load platform).
- Gray level CCD camera connected to PMC Bt848 based frame grabber.
- Bumpers are sub-divided into eight tactile zones at two heights.
- 8 batteries (12 V) for an autonomous operation up to 6 hours.
- Size: 45cm width x 45cm depth x 65 cm height.
- Weight: About 50 kg.

Donald and Daffy Duck

- MVME230x PowerPC 604 300 MHz VME Motorola card.
- Compact VME rack with 5, 12 and 24 V power supply.
- SmartCARD 2, 6 servo axes controller, 8 power amplifier, analog and digital I/O controller.
- DC-motors with encoder resolution of 500 pulses per revolution.
- 2 Sick LMS200 laser scanner with a full view on the environment (minus two blind zones between the two scanners).
- Gray level CCD camera connected to PMC Bt848 based frame grabber.
- 4 batteries (12 V) for an autonomous operation up to 4 hours.
- Size: 35cm width x 45cm depth x 60 cm height.
- Weight: About 35 kg.

1.5.2.2 The Development and Operating System

The application software is deployed on top of the XO/2 operating system [Brega98], [Brega00], [Tomatis01c]. XO/2 is an object-oriented, hard-real time system software and framework designed for safety, extensibility and abstraction. It takes care of many common issues faced by programmers of mechatronic products by hiding general design patterns inside internal mechanisms or by encapsulating them into easy-to-understand abstractions. Careful handling of the safety aspects has been the criterion by which the system has been developed. These mechanisms allow the system to maintain a *deus ex-machina* knowledge about the running applications thus providing higher confidence to the application programmer which, relieved from many computer-science aspects, can better focus his attention on the actual problem to be solved. Some more details are to be found in Appendix A.

1.5.3 Structure

This work is structured in five main sections:

- “Perception using Uncertain Information” discusses the importance of uncertainty modeling for probabilistic approaches, especially for the case of metric data.
- Chapter “Extracting Features” introduces the concept of features, presents the advantage and drawbacks which are present when using them and explains how to extract those features which will be used in the rest of the work. Some words concerning which kind of feature is adapted for metric, topological or hybrid navigation schemes are also given as an introduction for the next chapters.
- Perceiving the environment remains one of the fundamental tasks for autonomous systems. “Multi-Sensor Data Fusion” discusses how to integrate data information from different sensors on different levels on the perception pipe in order to improve the overall perception characteristics.
- The next chapter “Hybrid, Metric - Topological, Localization” analyzes the main characteristics of some well-known metric and topological approaches for localization focusing on their problems. A hybrid method which takes advantage of the positive characteristics of both the metric and topological paradigm is then presented and tested in detail.
- In the chapter “Hybrid Localization and Map Building” the method is extended to, and tested for, map building.

2

Perception using Uncertain Information

"As far as the laws of mathematics refer to reality, they are not certain, and as far as they are certain, they do not refer to reality."

Albert Einstein (1879–1955)

The interaction between mathematics and reality has always been difficult. The laws of mathematics give a precise result when each pre-condition is respected and the data is precisely defined. However, reality is difficult to measure and can, therefore, never be resumed by perfect data. This is the reason why, when working with reality, uncertainty models are introduced and applied to the measurement processes. By introducing these uncertainty models in mathematical laws, the quality of the results can be estimated.

However, somewhere an assumption concerning the origin and the magnitude of the uncertainty, or the errors in the measurement, has to be made.

2.1 Bottom-Up Sensor Modeling

In order to minimize the divergence between the models and reality care has to be taken when making the above mentioned assumption. By knowing the physical characteristics of the employed sensors the source of uncertainties can be identified. This source can be modeled, verified by experimentation and propagated through the extraction process up to the application level.

2.2 Odometry

Non-systematic odometry errors occur in two spaces: The joint space and the Cartesian space. With a differential drive kinematics the joint space is two-dimensional and includes the left and right wheel. Effects of wheel slippage, uneven ground and limited encoder resolution appear in this space. In [Chong97] a physically well-grounded model for this kind of errors is presented starting from the uncertain input $u(k+1) = [\Delta d_L, \Delta d_R]^T$ with $\Delta d_L, \Delta d_R$ as the distances travelled by each wheel and the diagonal input covariance matrix:

$$U(k+1) = \begin{bmatrix} k_L|\Delta d_L| & 0 \\ 0 & k_R|\Delta d_R| \end{bmatrix} \tag{2.1}$$

which relies on the assumption of proportionally growing variances per $\Delta d_L, \Delta d_R$ travelled. The odometry model for the first and second moment of the state vector $x = (x, y, \theta)^T$ is then:

$$\hat{x}(k+1|k) = f(x(k|k), u(k+1)) \tag{2.2}$$

$$P(k+1|k) = \nabla f_x P(k|k) \nabla f_x^T + \nabla f_u U(k+1) \nabla f_u^T \tag{2.3}$$

where $f(.)$ uses a piecewise linear approximation, $P(k|k)$ is the state covariance matrix of the last step and $\nabla f_{x,u}$ is the Jacobian of $f(.)$ with respect to the uncertain vectors $x(k|k)$ and $u(k+1)$. k_L and k_R are constants with units of meters.

The Cartesian space is spanned by the vector x encoding position and orientation of the vehicle. Effects of external forces (mainly collisions) occur in this space. Non-systematic Cartesian errors could be additionally modeled in Equation 2.3 by a 3×3 covariance matrix $Q(k+1)$ being a function of the robot displacement $\Delta x, \Delta\theta$ in the robot frame. In any case it is difficult to identify these models, i.e. to obtain rigorous values for k_L, k_R and especially the coefficients in $Q(k+1)$ which are valid for a range of floor types. Therefore, in this work only the joint space model (i.e. model presented in [Chong97]) will be taken into account.

2.3 Laser Range Finder

Laser equipped with a rotating mirror are known as lidars or laser range finders. They are based on a *time-of-flight* principle where the range distance is estimated

by measuring the time the laser beam requires to hit an obstacle and to be reflected back to the transmitter.

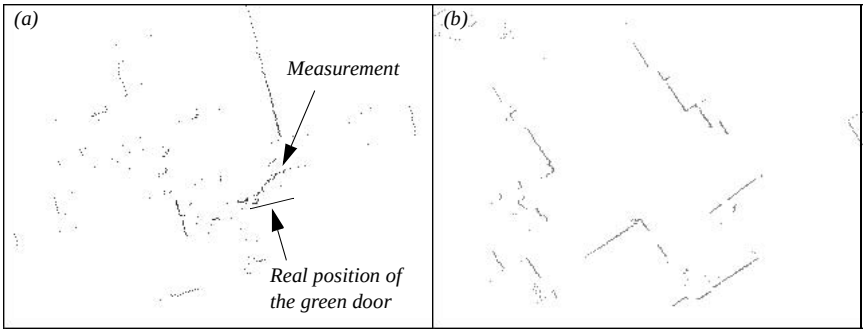

Figure 2.1: Two laser scan of the same room. (a) The scan from the Acuity AccuRange4000LIR has many outliers and gives strange measurements on green surfaces. (b) The Sick LMS200 is an industrial device.

2.3.1 Acuity AccuRange4000LIR

The AccuRange 4000 has a mirror's rotation frequency of 2.78 Hz yielding a 1° angular resolution with its maximal sampling frequency in calibrated mode of 1 kHz. It delivers range ρ and intensity i as analogue signals. The latter is the signal strength of the reflected beam and predominantly affects range variance. In order to have a good physically based uncertainty model of range variability accounting not only for the distance to the target but also for its surface properties, a relationship $\sigma_\rho = f(i)$ is sought. The uncertainty due to the encoders is usually low with respect to the beam spot size. Angular variability is therefore neglected. For range accuracy there are several factors which influence the extent of noise:

- The amplitude of the returned signal which is available as measurement.
- Drift and fluctuations in sensor circuitry. At the configured sampling frequency (1 kHz) this is predominant over thermal noise of the detection photodiode and resolution artifacts of the internal timers.
- Noise injected by the AD conversion electronics.

In [Adams99] a phase shift measurement principle has been examined yielding a range variance to amplitude relationship of the form:

$$\sigma_\rho^2 = a/V_r + b \tag{2.4}$$

where σ_ρ^2 is range variance and V_r the measured amplitude. After identification an inverse, slightly non-linear function was found.

For identification in this case an experiment was performed with a stationary target at about 1 meter distance [Arras99a]. The returned signal strength was varied systematically with a Kodak gray scale control patch where 10,000 readings were taken at each of the twenty gray levels.

In contrast to the model in [Adams99] an abrupt rise of noise below a certain amplitude can be observed (Fig. 2.2). This yields a simple relationship describable by two parameters: i_{min} allows the rejection of too uncertain range readings with $i < i_{min}$ and for measurement with $i > i_{min}$ a constant value for range variance $\sigma^2_{\rho_{const}}$ could have been found. This reduces our model for range variance to a constant value, independent of target distance and amplitude.

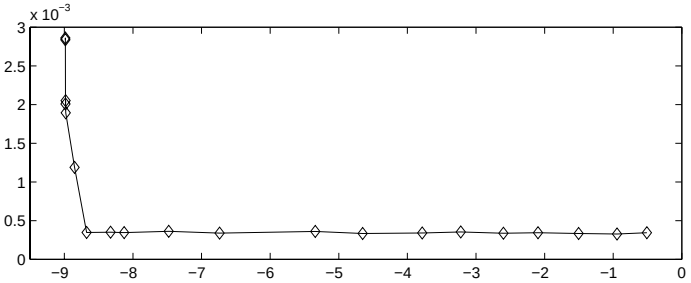

Figure 2.2: Range standard deviation (*y*-axis, in meters) against measured amplitude (*x*-axis, in Volts). 10,000 readings, measurements were conducted with a Kodak gray scale control patch.

Although this analysis leads to such a simple result it permits rejection of false, or very uncertain readings, by means of the amplitude measurements. This is very important since in many practical cases the sensor exhibits strong dependency upon the surface properties such as color and roughness. Moreover, the Acuity sensor is often subject to outliers. When the laser beam hits no target at all, and at normally occurring range discontinuities it returns an arbitrary range value, typically accompanied by a low signal strength.

2.3.2 Sick LMS200-30106

The LMS200 is an industrial device. Besides the protocol driver which is to be written it can be used practically plug-and-play. It delivers high quality range and angle information and comes with standard interfaces. The disadvantage is that this black-box character inhibits the above mentioned analysis of noise sources. Therefore, the constant measurement uncertainty given by the supplier is used for the range measurements, while as with the Acuity the angular variability is neglected.

2.4 CCD Camera

A CCD camera is a device which permits the measurement of the light intensity in an area of the environment. This is performed by employing an optic which allows the choice of an opening angle. The optic projects the image on a CCD sensor which measures the intensity (gray level or color). Imperfections in the optics and interference in the device electronics cause non-neglectable errors in the measured image. These errors have to be taken into account by a suitable model.

2.4.1 Sensor Modeling

2.4.1.1 The Pinhole Model

A vision task which is intended to extract accurate geometric information from a scene requires a calibrated vision system. For this the imaging device is normally modeled as an ideal pinhole camera as shown in Fig. 2.3.

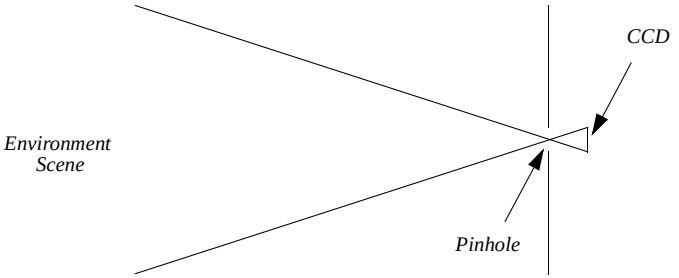

Figure 2.3: The pinhole camera model.

This requires the determination of a variety of camera parameters including its position and orientation (extrinsic parameters), image center, scale factor and lens focal length (intrinsic parameters). Due to the complexity of the lens system which cannot be captured by such a simple model distortion parameters of the imaging system have to be additionally introduced in order to determine the deviation from the pinhole camera model.

2.4.1.2 Distortion

The two principal forms of distortion considered in photogrammetry applications are radial and decentering (tangential) distortion.

A model for the correction of these distortions is given by Equation 2.5 and Equation 2.6. The coordinates (x', y') refer to the distorted location of the point in the uncorrected image and $(\bar{x}, \bar{y})$ to the corresponding corrected location.

$$\bar{x} = x' + x_c(k_1 r^2 + k_2 r^4 + k_3 r^6 + \ldots) + [p_1(r^2 + x_c^2) + 2p_2 x_c y_c][1 + p_3 r^2 + \ldots] \quad (2.5)$$

$$\bar{y} = y' + y_c(k_1 r^2 + k_2 r^4 + k_3 r^6 + \ldots) + [p_2(r^2 + y_c^2) + 2p_1 x_c y_c][1 + p_3 r^2 + \ldots] \quad (2.6)$$

where $x_c = x' - c_x$, $y_c = y' - c_y$, $r = \sqrt{x_c^2 + y_c^2}$, (k_1, k_2, k_3) are the parameters of radial distortion; (p_1, p_2, p_3) are the parameters of decentering distortion; and r is the radius of a point from the image center which is defined by the couple (c_x, c_y). A typical approach is to model only one or two distortion coefficients since higher order coefficients are comparatively insignificant [Weng91] and could even model noise in the calibration procedure instead of real camera characteristics.

2.4.1.3 The Model

The position of the vision device is modeled by means of the position of its center of projection $O(O_x, O_y, O_z)$. Being the orientation of the lens system and the CCD sensor imperfect, orientation has to be explicitly modeled. This can be performed by means of 3 rotation parameters. Therefore, a total of 6 parameters (extrinsic) describe the position and orientation of the CCD camera completely.

The intrinsic parameters are image center, scale factor and lens focal length. The image center parameters $H(H_x, H_y)$ define the distance between the center of the image projected on the CCD and the center of the CCD sensor itself. The scale factor S is a measurement of one pixel in meters. The focal length C is the distance between the center of projection $O(O_x, O_y, O_z)$ and the CCD sensor.

All these parameters are visualized in Fig. 2.4.

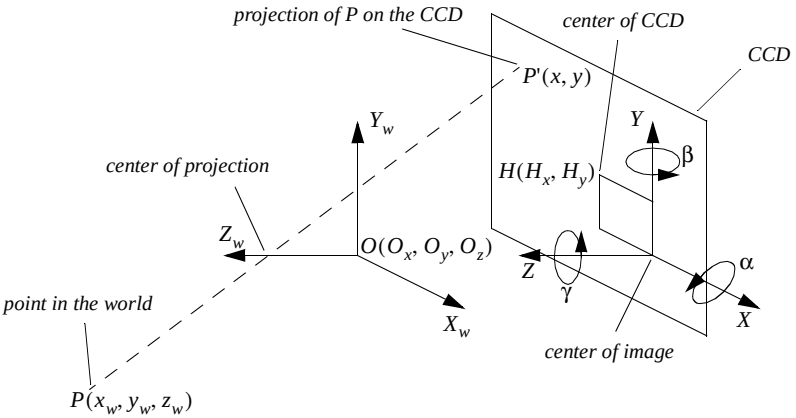

Figure 2.4: The general model for camera calibration.

The choice of the distortion parameters depends on the type of lens that is used. Lens with low distortion will require only a simple model with few parameters. A wide angle lens requires more parameters and could even not be properly modeled in the whole area due to imperfections in the construction which are caused by the complexity of building such a lens system.

The model for the calibration of a general vision system is given by the following equations where Equation 2.7 is used for the calibration in the *x*-direction and Equation 2.8 in the *y*-direction:

$$C \cdot \frac{(x_w - O_x) + \gamma(y_w - O_y) - \beta(z_w - O_z)}{\beta(x_w - O_x) - \alpha(y_w - O_y) + (z_w - O_z)} =$$

$$S\left\{x_c + x_c(k_1 r^2 + \ldots) + [p_1(r^2 + x_c{}^2) + 2p_2 x_c y_c][1 + p_3 r^2 + \ldots]\right\} \qquad (2.7)$$

$$C \cdot \frac{(-\gamma)(x_w - O_x) + (y_w - O_y) + \alpha(z_w - O_z)}{\beta(x_w - O_x) - \alpha(y_w - O_y) + (z_w - O_z)} =$$

$$S\left\{y_c + y_c(k_1 r^2 + \ldots) + [p_2(r^2 + y_c{}^2) + 2p_1 x_c y_c][1 + p_3 r^2 + \ldots]\right\} \qquad (2.8)$$

(x_w, y_w, z_w) is the position of a point in world-coordinates; the coordinates (x, y) refer to the distorted location of this point in the uncorrected image; $x_c = x - H_x$, $y_c = y - H_y$, $r^2 = x_c^2 + y_c^2$ and (k_1, k_2, k_3) are the parameters of radial distortion; (p_1, p_2, p_3) are the parameters of decentering distortion; r is the radius of a point from the image center; $H(H_x, H_y)$ is the image center.

Figure 2.5: A room of the Institute. The image loses its rectangular layout because it is compensated against radial distortion.

2.4.2 Implementation

The camera system is calibrated by combining the method presented in [Prescott97] with spatial knowledge from a test field. This provides a coherent set of extrinsic, intrinsic and distortion parameters.

2.4.2.1 The Model

For the implementation some adaptations have been conducted in order to obtain a simple and numerically stable model. The general model (Equation 2.7 and Equation 2.8) denotes a high dependence between the scale factor *S*, the focal length *C* and the O_z parameter of the center of projection. By using different test fields with different positions and orientations these dependences are reduced when solving the non-linear system. With the current setup it is not the case. The test filed is perpendicular to the *Z*-axis of the camera. Therefore, *S* and *C* have been defined as constant with the value given by the hardware supplier.

Furthermore, in [Arras99a], [Arras99b], [Arras00] and [Arras01a] the vision system has been used to extract vertical lines only. Then, in this case only the *x*-direction has been calibrated. It follows that O_y and the rotation on the *X*-axis (α) are not relevant. The rotation on the *Z*-axis (γ) has not been taken into account because vibrations and imprecision on the floor where the robot is navigating are of some magnitude higher than this error.

Another dependence which has been found is the one between O_x, H_x and β. To stabilize the conversion of the Gauss-Newton solution of the non-linear system the less relevant parameter (O_x) had been excluded from the final model.

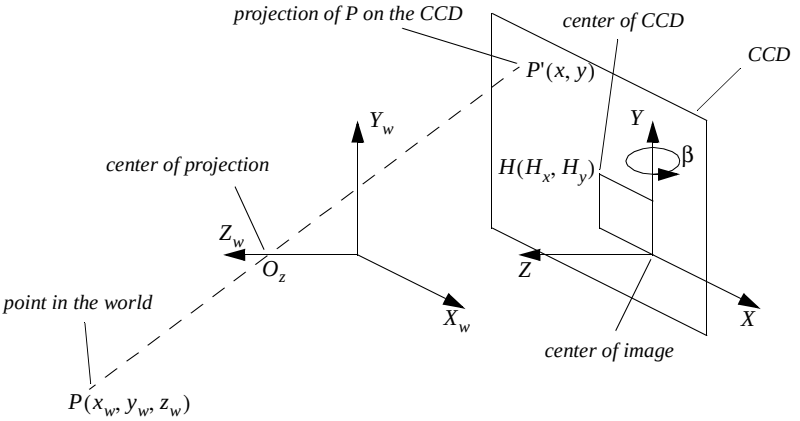

Figure 2.6: The camera model as used for the vision devices on the robots.

Even by neglecting the above mentioned parameters the calibration can be very precise if the camera is mounted horizontally on the robot with enough mechanical accuracy.

The resulting model is graphically described in Fig. 2.6.

The next step is the choice of the distortion model. By fitting various models with different numbers and types of distortion parameters, spherical aberrations resulted in being important, while tangential aberrations were not very relevant. To reduce the complexity of the parameter fitting phase and of the on-line image calibration only spherical distortion had been corrected.

By comparing the resulting Equation 2.9 with the general model described by Equation 2.7 and Equation 2.8 the achieved simplification can be easily seen:

$$C \cdot \frac{x_w - \beta(z_w - O_z)}{\beta x_w + (z_w - O_z)} = S[x_c + x_c(k_1 r^2 + k_2 r^4 + k_3 r^6 + k_4 r^8)] \qquad (2.9)$$

where (x_w, y_w, z_w) is the position of a point in world-coordinates; the coordinates (x, y) refer to the distorted location of this point in the uncorrected image; $x_c = x - H_x$, $y_c = y - H_y$ and $r^2 = x_c^2 + y_c^2$. k_1, k_2, k_3, k_4 are parameters of the radial distortion; r is the radius of a point from the image center; $H(H_x, H_y)$ is the image center.

The question is now how Equation 2.9 has to be used to obtain a calibrated image for processing. There are two main steps: Parameter fitting and image calibration.

2.4.2.2 Parameter Fitting

To find a set of parameters some measurements have to be taken. Eight parameters have to be found: The image center $H(H_x, H_y)$, the position of the center of projection O_z, the rotation β along the y-axis and the distortion parameters k_1, k_2, k_3 and k_4. The other variables in Equation 2.9 have to be measured. For this a test field is fixed to the robot so that its relative position to the robot is well-known. The world frame is set to the robot frame. It follows that O_z is the distance between the robot frame and the center of projection.

The parameter fitting is then made by means of the following steps:

- Image capturing and edge pixel extraction
- Measurement of the edge points in the image (x) and with the position of that point in world coordinates (x_w and z_w)
- The last step has as a result a non-linear over-determinate system which is solved with the Gauss-Newton method.

Note that in order to have a fully calibrated image exactly the same method can be used. Starting from Equation 2.8 a simplified equation for the *y*-direction can be found (as with Equation 2.9 for *x*) yielding a non-linear over-determined system with twice the number of equations as the current one.

Uncertainties from the test field geometry (uncertain measurements x_w and z_w) and those caused by noise in the camera and acquisition electronics (uncertain measurement *x*) are propagated through the camera calibration procedure onto the level of camera parameters.

2.4.2.3 Image Calibration

Image processing takes place on the image level. That means that the corrected position of a pixel in the image has to be calculated:

$$\bar{x} = x_c + x_c(k_1 r^2 + k_2 r^4 + k_3 r^6 + k_4 r^8) \tag{2.10}$$

where the coordinates (x, y) refer to the distorted location of this point in the uncorrected image; $\bar{x}$ is the corrected position; $x_c = x - H_x$, $y_c = y - H_y$, $r^2 = x_c^2 + y_c^2$, (k_1, k_2, k_3, k_4) are the parameters of the radial distortion; *r* is the radius of a point from the image center; $H(H_x, H_y)$ is the image center.

As soon as the information concerning the position in world coordinates is needed the angle in sensor coordinates of the projection of a calibrated pixel can be calculated by means of:

$$\varphi = \operatorname{atan}(\bar{x} \cdot S / C) \tag{2.11}$$

where *C* is the focal length, *S* the scale factor and $\bar{x}$ the calibrated position from Equation 2.10. Exactly the same procedure is employed for the *y*-direction.

In this case the sources of uncertainty are the calibration parameters and the electronics of the vision system. They are propagated through the calibration model yielding an uncertainty $\sigma_{\bar{x}}$ for each pixel.

2.5 Conclusions

In this chapter the models of the sensors, which are used in this thesis, have been briefly presented. The models use probability theory to represent the imprecision associated with the metric location of each measurement. This imprecision is defined as *uncertainty*. As explained in Section 2.1 a bottom-up approach has been adopted. The modeled uncertainties represent the measurement error source which will be propagated through the extraction process up to the application level. This will be presented in the next chapter for *primitive features* where error propagation is used to bring this source of uncertainty up to the extracted horizontal and vertical lines.

3

Extracting Features

"Mathematics … would certainly have not come into existence if one had known from the beginning that there was in nature no exactly straight line, no actual circle, no absolute magnitude."

Friedrich Nietzsche (1844–1900)

Even if mathematicians and robotics researchers may not agree with the statement of Nietzsche this sentence introduces one of the main problems of feature extraction: The reliance on the existence of the feature itself. Actually an alternative to the use of features is the integration of raw data directly which have the advantage of being as general as possible. But, with most sensors they are credible only by processing great amounts of data and are very sensitive to dynamic objects. The feature extraction step can then be seen as a filter which permits the handling of noise from the sensors and dynamics in the environment. Furthermore, when the type of feature is properly chosen feature extraction permits the better isolating of informative patterns.

This section as not to be taken as the presentation of new or revolutionary approaches to feature extraction. Feature extraction is not a goal of this work, it is just a mean to reach the goal of the thesis which is to develop a new navigation approach, as stated in Section 1.5. Here both primitive and high-level features are presented. During the whole text primitive features are also simply defined as features while high-level features are often referred to as landmarks.

3.1 Primitive Features

In this first section the extraction of some primitive features is explained in detail. Primitive features are defined as features which can be described by simple geometrical definitions. Points, lines and segments belong to this type of features. In this section it is described how to extract lines from points for both the laser scanner and CCD camera. Note that such features have precise geometrical characteristics which make them very appropriate for pure metrical navigation (see Section 5.1 for a definition of metric). An appropriate uncertainty model helps the use of these features which can then be better integrated in a probabilistic metric framework such as an *Extended Kalman Filter* (EKF) [Crowley89], [Leonard92] or the *Markov localization* [Fox98]. To accomplish this the uncertainties coming from the sensors measurements (points) are properly propagated up to the extracted feature.

3.1.1 Horizontal Lines

This section presents the approach for extracting horizontal lines from a 360° laser scanner data set which Kai Oliver Arras has published at the Mobile Robotics XII conference of SPIE in 1997 [Arras97].

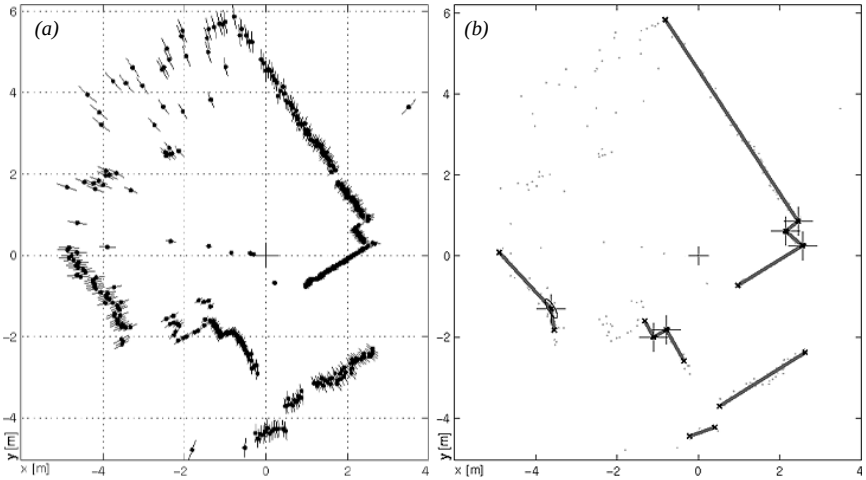

Figure 3.1: (a) A scan from the Acuity AccuRange4000LIR laser scanner. The data is represented with their range uncertainty. (b) The extracted segments which define the parameters of the infinite horizontal lines.

By extracting horizontal lines two main problems have to faced: Segmentation and fitting. Segmentation permits the determination of which data belong to which physical object (line in this case), while fitting means finding the best line to some given data. 'Best' is defined with respect to unweighted squared algebraic errors. Furthermore, a third aspect is taken into account: Valuable information would be lost if only a single observed segment is taken into account when other segments belonging to the same physical object are extracted. On the other hand two features which differ only slightly in one or more of their model parameters should be identified as being distinct within the limits which are given by sensor noise. This problem is defined here as segment merging. The result of these three steps can be seen in Fig. 3.1.

3.1.1.1 Data Segmentation

Segmentation of the range image can be performed by accumulating evidence in the model space as with the *Hough transform* [Hough62]. This leads to a clustering problem with *n* points which can be solved efficiently due to the particular character of the problem.

The model is fitted into n_f neighboring points and the covariance matrix is computed. This is conducted for all points of the scan. When adjacent groups of range readings lie on the same feature their associated points constitute a cluster in the model space corresponding to that feature. Clustering is now the task of finding these clusters. In a general case a clustering problem of this size where no *a priori* knowledge is available would lead to impracticably high computational times for an embedded system. However, with a laser scanner points on the same land-mark are *usually* consecutive points. Due to this underlying regularity in the acquisition process a distance measure in the model space is defined which is applied to n_m adjacent points:

$$d_i = \sum_j (x_j - x_w)^T (C_j + C_w)^{-1} (x_j - x_w) \tag{3.1}$$

where $j = i - (n_m - 1)/2, \ldots, i + (n_m - 1)/2$ and x_w is the weighted mean:

$$x_w = C_w \sum C_i^{-1} x_i,\ C_w^{-1} = \sum C_i^{-1} \tag{3.2}$$

Low distance indicates that the points involved have high model fidelity. If d_i is plotted against the measurement index regions of low value can be expected at the corresponding index places of the sought clusters. A threshold d_m is applied cutting off the regions of low distance. A contributing segment is now defined to be the set of measurement points whose representations in the model space satisfying $d_i \le d_m$.

3.1.1.2 Line Fitting

Line fitting is a problem with known solution which is linear in the parameters. One dimensional range data are acquired in polar coordinates. As explained in Section 2.3 for the laser range finders the angular uncertainty has been neglected. In this case a re-parametrization of the line model can be made such that the fit problem is lead back to the standard regression. In general, range data uncertainties are in both coordinates and the question is which errors are then to be minimized by the fitting algorithm. The perpendicular distances from the points to the line is a choice which makes sense from the geometrical point of view because the goal is to have a solution which keeps track of the spatial or geometrical character of the problem. This yields a non-linear regression problem which has to be solved for polar coordinates. By using the sensor models presented in Section 2.3 a weight for each measurement point can be determined and the line can be fitted in the generalized least squares sense. It can be shown that the solution is:

$$\tan 2\alpha = \frac{\frac{2}{\sum w_i}\sum\sum_{i<j} w_i w_j \rho_i \rho_j \sin(\theta_i + \theta_j) + \frac{1}{\sum w_i}\sum (w_i - \sum w_j) w_i \rho_i^2 \sin 2\theta_i}{\frac{2}{\sum w_i}\sum\sum_{i<j} w_i w_j \rho_i \rho_j \cos(\theta_i + \theta_j) + \frac{1}{\sum w_i}\sum (w_i - \sum w_j) w_i \rho_i^2 \cos 2\theta_i} \tag{3.3}$$

$$r = \frac{\sum w_i \rho_i \cos(\theta_i - \alpha)}{\sum w_i} \tag{3.4}$$

where (ρ_i, θ_i, w_i) is the measurement i; α and r are the parameters of the line model:

$$\rho \cos(\theta - \alpha) - r = 0 \tag{3.5}$$

which is also shown in Fig. 3.2.

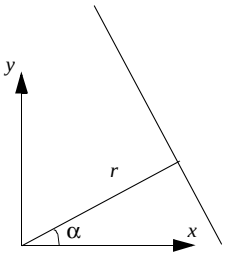

Figure 3.2: The line model.

However the identical Cartesian form of Equation 3.3 and Equation 3.4 can be used in order to have a more efficient implementation:

$$\tan 2\alpha = \frac{-2\sum w_i(\bar{y}_w - y_i)(\bar{x}_w - x_i)}{\sum w_i[(\bar{y}_w - y_i)^2 - (\bar{x}_w - x_i)^2]} \tag{3.6}$$

$$r = \bar{x}_w \cos\alpha + \bar{y}_w \sin\alpha \tag{3.7}$$

where:

$$\bar{x}_w = \frac{1}{\sum w_i}\sum w_i \rho_i \cos\theta_i \text{ and } \bar{y}_w = \frac{1}{\sum w_i}\sum w_i \rho_i \sin\theta_i \tag{3.8}$$

are the weighted means. Nevertheless, in order to determine the covariance matrix the result of Equation 3.3 and Equation 3.4 has to be known. By writing the model parameters as random variables represented by their first-order Taylor series expansion about the mean point and assuming independence of P and Q the first-order covariance estimate is:

$$C_t = m_{\rho\theta} C_{\rho\theta} m_{\rho\theta}^T = m_\rho C_\rho m_\rho^T + m_\theta C_\theta m_\theta^T \tag{3.9}$$

where:

$$m_{\rho\theta} = \begin{bmatrix} m_\rho & m_\theta \end{bmatrix} = \begin{bmatrix} \frac{\partial\alpha}{\partial P} & \frac{\partial\alpha}{\partial Q} \\ \frac{\partial r}{\partial P} & \frac{\partial r}{\partial Q} \end{bmatrix} \tag{3.10}$$

is the $p \times 2n$ Jacobian matrix containing all partial derivatives of the model parameters with respect to P and Q about the mean point and $C_{\rho\theta} = diag(C_\rho, C_\theta)$ the $2n \times 2n$ partitioned diagonal data covariance matrix with sub-matrices $C_\rho = diag\left(\sigma_{\rho_i}^2\right)$ and $C_\theta = diag\left(\sigma_{\theta_i}^2\right)$.

According to the noise model for laser range finders of Section 2.3 the term in Equation 3.9 which keeps track of angular uncertainties has been omitted yielding a parameter covariance matrix with elements:

$$\sigma_{\alpha\alpha} = \frac{1}{(D^2 + N^2)^2}\left[\sum w_i^2 [N(\bar{x}_w \cos\theta_i - \bar{y}_w \sin\theta_i - \rho_i \cos 2\theta_i) \right.$$
$$\left. D(\bar{x}_w \sin\theta_i + \bar{y}_w \cos\theta_i - \rho_i \sin 2\theta_i)]^2 \sigma_{\rho_i}^2\right] \tag{3.11}$$

$$\sigma_{rr} = \sum \left[\frac{w_i}{\sum w_j} \cos(\theta_i - \alpha) + \frac{\partial \alpha}{\partial P_i} (\bar{y}_w \cos\alpha - \bar{x}_w \sin\alpha) \right]^2 \sigma^2_{\rho_i} \tag{3.12}$$

$$\sigma_{\alpha r} = \sum \frac{\partial \alpha}{\partial P_i} \frac{\partial r}{\partial P_i} \sigma^2_{\rho_i} \tag{3.13}$$

where N and D are numerator and denominator of the right hand side of Equation 3.3 or Equation 3.6 respectively.

3.1.1.3 Segment Merging

The goal of this step is to merge observed segments which belong to the same physical object. Clustering techniques provide this treatment of data. Having gained a number n_s of segments from the method in the last section, or any other scheme providing propagation of the first two moments, the number of points in the model space has been significantly reduced. An *agglomerative hierarchical clustering* algorithm, which permits an efficient implementation, is utilized and due to its simplicity a short outline is given:

Having computed the symmetric $n_s \times n_s$ distance matrix D, the procedure starts with each point as a separate cluster:

1. Find the minimal distance d_{ij} of clusters Q_i and Q_j in D and while $d_{ij} \leq d_{\alpha_\varepsilon}$ proceed to
2. Merge Q_i and Q_j obtaining Q_{ij}. The number of clusters is decreased by one.
3. Update D by calculating the new distances from Q_{ij} to all other clusters. Only one column and row is changed. Then go back to 1.

The distance between two clusters is always the distance of two normally distributed vectors in the model space and we can refer back to the matching problem where the *Mahalanobis* distance is widely employed for that purpose:

$$d^2_{ij} = (x_i - x_j)^T (C_i + C_j)^{-1} (x_i - x_j) \tag{3.14}$$

If x_i and x_j belong to the same feature d^2_{ij} has a chi-squared distribution and an appropriate threshold $\chi^2_{\alpha_\varepsilon, 2}$ must be chosen. The final estimates of the parameters is directly available after exiting the clustering algorithm. The final segments are obtained by combining their measurement points provided that the segments are adjacent and belong to the same line.

3.1.2 Vertical Lines

Vertical lines have characteristics which are similar to those of the previously presented horizontal lines: They allow for a precise and simple geometrical characterization permitting their application for pure metric localization. However, they differ in many other aspects. First of all they cannot be extracted by a 360° laser scanner; the best suited sensor in this case is a camera. Then, vertical lines exhibit characteristics which are complementary to horizontal lines especially for models of indoor environments, where long and thin structures like hallways can often present few horizontal lines, while containing many vertical structures. Furthermore, they exhibit some other characteristics which are interesting from a practical point of view: If the camera has a horizontal position on the vehicle vertical structures have the advantage of being view invariant, opposed to horizontal ones. Furthermore, they require simple image processing which is important for a real-time implementation under conditions of moderate computing power. Nevertheless, vertical lines also have a main drawback for environmental modeling: They exhibit a very low mutual discriminance. The fact that in indoor environments there are many vertical structures can cause the detection many ambiguous vertical lines which are close together.

The more robust way to extract vertical lines would be to:

- Calibrate the image received from the camera
- Enhance edge pixels
- Thin the edge image
- Fit edge points to the line model

However, with embedded systems and especially for mobile robots computing power has to be used with care to ensure full autonomy. Therefore, some optimization with respect to computational efficiency are proposed for the implementation:

- Enhance edge pixels only in the x-direction for vertical lines (sub-optimal due to the spherical aberration which is not yet corrected by the calibration procedure)
- Thin the edge image by applying non-maxima suppression only in x (very fast thinning approach which can be used because edge enhancement has been performed only in x)
- Calibrate by means of a look-up table the remaining edge pixels instead of the whole image
- Count the edge pixels for each column and label as vertical edges with a pre-defined number of edge pixels (line fitting reduces to a 1D problem)

3.1.2.1 Edge Enhancement

Optimal edge enhancement can be achieved by using a well suited smoothing technique. The *Canny* edge detector is an edge enhancement method where smoothing is a central issue: Smoothing is performed by convoluting the image with a Gaussian function. It can be demonstrated that this approach is an optimal edge detection filter with respect to output to noise ration, localization precision and unique maximum. However, as often stated in this dissertation, computational time is an essential component for autonomous embedded systems. Therefore, it is preferable to use simple operators to approximate smoothing results. For this, masks can be used instead of convolution to approximate the gradient components more efficiently.

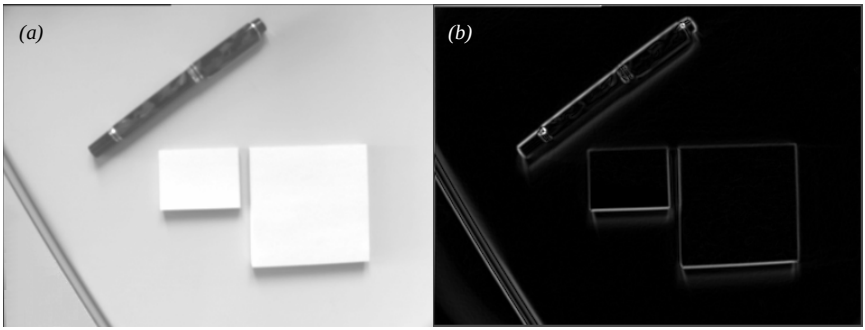

Figure 3.3: (a) A CCD gray scale image. (b) Its gradient magnitude image approximated by means of the Sobel filter.

For vertical lines detection the *Sobel* filter is used. It is composed of two *3 x 3* masks oriented in the row and column directions:

$$s_1 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, s_2 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.15}$$

These masks are applied to each pixel and the resulting gradient magnitude $|\vec{G}|$ is approximated as follows:

$$|\vec{G}| = \sqrt{s_1^2 + s_2^2} \tag{3.16}$$

While the gradient direction α is given by the approximation:

$$\alpha = \operatorname{atan}\left(\frac{s_1}{s_2}\right) \tag{3.17}$$

In Fig. 3.3 an image with its resulting gradient magnitude image is shown as an example.

For vertical lines this reduces to a single mask enhancing in the *x*-direction, meaning that only s_2 has to be calculated. Furthermore, this directly yields the gradient magnitude $|\vec{G}|^2$ while the gradient direction α is fixed. Vertical edge enhancement on an uncalibrated image, as presented here, is sub-optimal due to the fact that objects which are effectively physically vertical would be better detected in a calibrated image, where they would also be perfectly vertical. In this case radial distortion lowers the resulting gradient magnitude. Nevertheless, this allows the limitation of the calculation on bi-dimensional data to this simple edge enhancement and the next step which is thresholding.

3.1.2.2 Thresholding

This step allows the isolation of the pixels which are interesting for further computation. This is performed by choosing the *n* pixels with the highest gradient magnitude. The motivation for this approach is twofold:

- It permits the bounding of the computation to a fixed number *n* of edge pixels
- It dynamically adapts the threshold by intrinsically taking into account illumination effects

Pixels which have a gradient magnitude higher than the previously mentioned threshold build the edge pixel image (Fig. 3.4a).

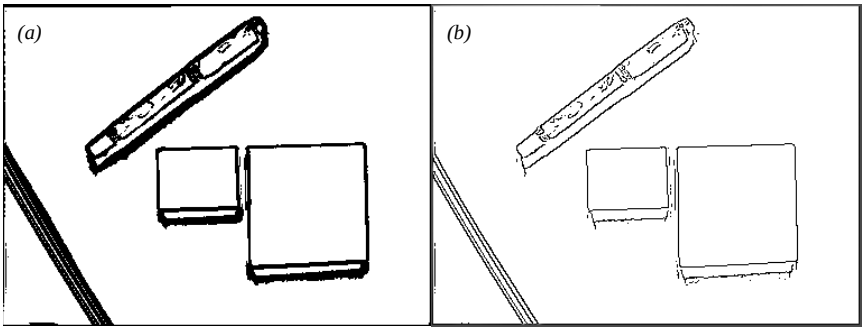

Figure 3.4: (a) The edge pixel image after thresholding. (b) Edge pixels after edge thinning.

3.1.2.3 Non-Maxima Suppression

In the edge pixel image shown in Fig. 3.4a edges are represented by structures with more than one pixel.

A thinning algorithm permits the elimination of pixels which are unnecessary for the representation of the edge. This is performed by choosing the pixels which have the highest gradient magnitude between those which represent the edge perpendicularly to the edge direction. The direction is given by the gradient direction α. This technique is defined as *non-maxima suppression* and is described in Fig. 3.5. The resulting image is shown Fig. 3.4b.

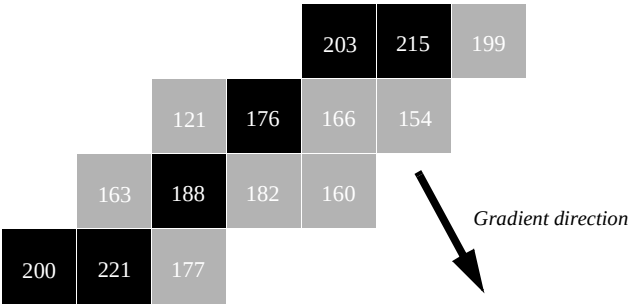

Figure 3.5: Non-maxima suppression permits the thinning of edges by choosing the pixel with the highest gradient magnitude between those which are on the same gradient direction.

3.1.2.4 Edge Pixel Calibration

As can be seen in Fig. 3.4b the remaining pixels represent only a very low subset of the whole image presented in Fig. 3.3a. Therefore, calibration is very efficiently applied at this moment. The *x* position of the remaining edge pixels is corrected leading to compensated position $\bar{x}$ by applying Equation 2.10:

$$\bar{x} = x_c + x_c(k_1 r^2 + k_2 r^4 + k_3 r^6 + k_4 r^8) \tag{3.18}$$

where the coordinates (x, y) refer to the distorted location of this point in the uncorrected image; $\bar{x}$ is the corrected position; $x_c = x - H_x$, $y_c = y - H_y$, $r^2 = x_c^2 + y_c^2$, (k_1, k_2, k_3, k_4) are the parameters of the radial distortion; r is the radius of a point from the image center; $H(H_x, H_y)$ is the image center.

3.1.2.5 Line Fitting

Line fitting reduces to simple histogram counting where the edge pixels belonging to the same image column are summed together. Columns representing peaks in the histogram and with a pre-defined number of edge pixels are labelled as ver-

tical edges. The angle representing the vertical line in sensor coordinates can then be calculated by means of:

$$\varphi = \text{atan}(\bar{x} \cdot S / C) \tag{3.19}$$

where C is the focal length, S the scale factor and $\bar{x}$ the calibrated position from Equation 3.18. Refer to Section 2.4 for more details.

This line extraction method is essentially a special case of the *Hough transform* [Hough62], where the model space is one-dimensional since the slope of the lines is kept constant. Fig. 3.6 shows some of the presented steps for a typical image taken during mobile robot navigation.

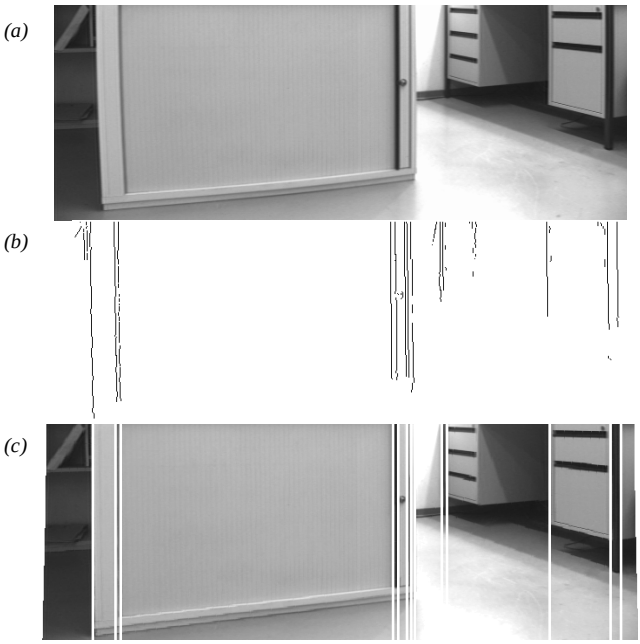

Figure 3.6: (a) Uncalibrated image. (b) The edge pixels are calibrated after edge enhancement, thresholding and thinning. (c) The resulting vertical lines on the calibrated image.

3.2 High-Level Features

High-level features are more complex than primitives and can, therefore, be intrinsically more distinctive for the environment. This characteristic can be exploited for topological localization where the distinctiveness of the landmarks

plays an important role. This fact has to be taken into account in the extraction procedure because the required information concerning the landmark is not the same as with the previously presented primitive features. Uncertainties in the position are less relevant in this case, while the probability of having effectively extracted the right kind of landmark is more important for those approaches (for example by the *Partial Observable Markov Decision Processes* (POMDPs)). This probability is defined as *extraction confidence*.

This section presents three different landmarks which are used for topological navigation. They are all extracted from a 360° laser scan.

3.2.1 Corners

Corners are landmarks which are well suited for both pure metric and topological localization. Approaches using them with an EKF are already available in the literature. However, the reason to present this type of landmark here is that they will be used later in Section 6.2 for topological localization and map building.

3.2.1.1 Corner Definition

A corner is defined as a point where two line segments meet each-other by forming a 90° angle. It is represented by its Cartesian position $(x_c, y_c)^T$ and the mean of the orientation α_c of the segments.

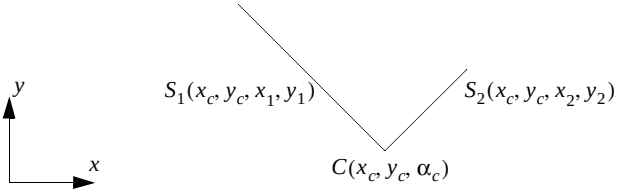

Figure 3.7: A corner is represented by a point where two segments meet each-other by forming a 90° angle and the mean of their orientation.

The corner parameter α_c can easily be determined by means of the segments parameters:

$$\alpha_c = \frac{\operatorname{atan2}\left(\frac{y_1 - y_c}{x_1 - x_c}\right) + \operatorname{atan2}\left(\frac{y_2 - y_c}{x_2 - x_c}\right)}{2} \tag{3.20}$$

3.2.1.2 Segmentation

Segmentation of high-level features has not a straightforward approach as with primitives where the feature model can directly be used for clustering. As explained in the previous section the corner model is basically defined by two segments. Therefore, as in Section 3.1.1 segmentation can be performed by accumulating evidence in the model space via the *Hough transform* [Hough62].

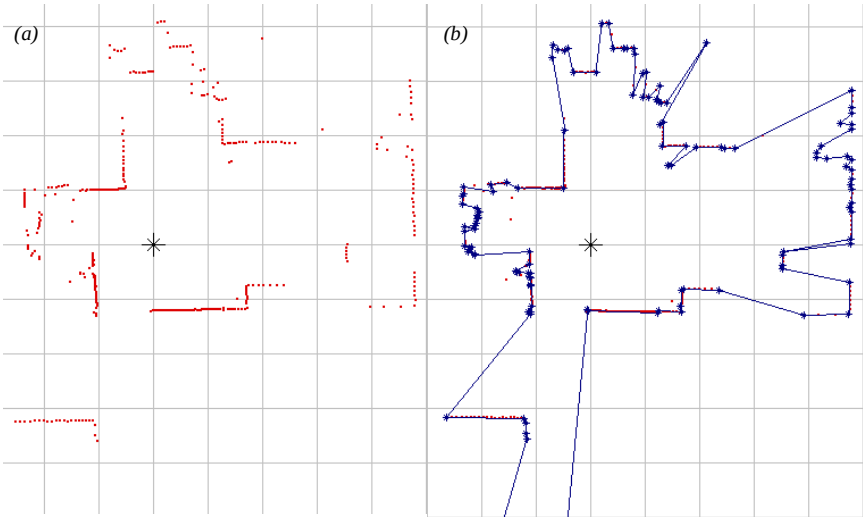

Figure 3.8: Segmentation of scan data by means of slope comparison to find segments for corner detection. (a) A scan from the Sick LMS200 laser scanner. (b) The extracted segments.

Given a scan with n (ρ_i, θ_i) measurements and their corresponding measurements (x_i, y_i), segmentation is performed by calculating the slope:

$$s_i = \operatorname{atan2}\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}}\right) \tag{3.21}$$

of each pair of consecutive scan points p_i and p_{i-1} and then traversing the slopes sequentially to find out which consecutive points belong to the same physical segment by applying the following slope and distance tests:

$$\begin{gathered} s_i - \bar{s}_{i-n,\,i-1} < \frac{max\Delta s}{\rho_i} \\ \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} < max\Delta d \end{gathered} \tag{3.22}$$

where $max\Delta s$ and $max\Delta d$ are thresholds established by experience, (x_i, y_i) and (x_{i-1}, y_{i-1}) the Cartesian coordinates of scan point *i* and *i-1*, and:

$$\bar{s}_{i-n,\,i-1} = \sum_{i-n}^{i-1} \frac{s_i}{n-1} \tag{3.23}$$

is the slope of the current segment.

The first test ensures that the point which passes the test is effectively on the same direction as the current segment. The $max\Delta s$ threshold is divided by ρ_i to ensure that points lying near the sensor are not erroneously segmented as a new object due to the discretization to 1 centimeter of the used laser scanner. The second test permits the correct separation of two consecutive segments with the same slope. The result of this segmentation approach can be seen in Fig. 3.8 where each detected segment is visualized.

To avoid unnecessary calculation due to noisy segments a further simple step is conducted before corner fitting: Segments composed by less than three points or defined by less than two points per meter are filtered out. The results of this further step can bee seen in Fig. 3.9.

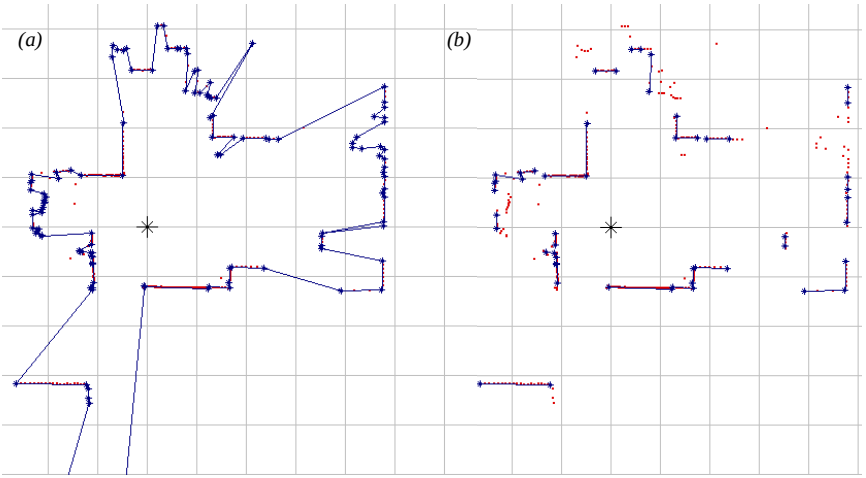

Figure 3.9: (a) Segments after segmentation of the laser data. (b) Segments resulting after having filtered out noise.

3.2.1.3 Corner Fitting

For corner fitting the characteristic deriving from the fact that laser data is sorted with respect to θ is exploited indirectly: Segments are also sorted and can, therefore, be sequentially traversed to find corners. Fitting reduces to a test ensuring that error for the 90° angle and the distance between the extremes of the segments which creates the corner, remain bounded within a suited threshold ε_s for the angle and ε_d for the distance:

$$\left| \operatorname{atan2}\left(\frac{y_1 - y_{c1}}{x_1 - x_{c1}}\right) - \operatorname{atan2}\left(\frac{y_2 - y_{c2}}{x_2 - x_{c2}}\right) \right| - 90 < \varepsilon_s$$
$$\sqrt{(x_{c1} - x_{c2})^2 + (y_{c1} - y_{c2})^2} < \varepsilon_d \tag{3.24}$$

where the meaning of x_{c1}, y_{c1}, x_{c2} and y_{c2} is explained in Fig. 3.10 and they are used to determine (x_c, y_c) in an approximated but highly efficient way, where:

$$x_c = \frac{x_{c1} + x_{c2}}{2} \text{ and } y_c = \frac{y_{c1} + y_{c2}}{2}. \tag{3.25}$$

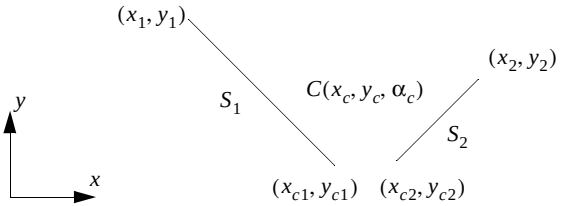

Figure 3.10: Two segments which may not share the corner point can build a corner if they fulfill Equation 3.24.

The result of the corner detection is shown in Fig. 3.11.

3.2.1.4 Extraction Confidence

In many POMDP approaches a table is used in order to estimate the probability of seeing and not seeing a certain type of landmark which is present at a certain location in the environment. The probability of seeing another type of landmark when the first one is physically present is also modeled in this way. This table is, in general, estimated by experience and then regarded as constant.

On-line estimation of these probabilities is, therefore, helpful in two ways:

- The estimation of the probabilities can be made for each landmark instead of each type of landmarks.

- The probability is updated dynamically during navigation allowing for dynamics in the environment.

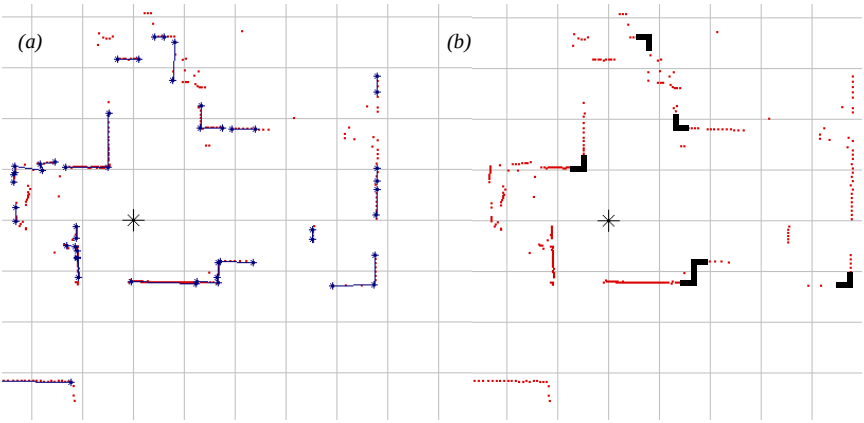

Figure 3.11: (a) Segments. (b) Detected corners.

Moving in this direction the extraction confidence has then to model the probability of seeing a certain landmark when it is present in a certain location. This can be seen as a physical characteristic of the landmark itself. For corners the size of the detected landmark is used. This size is not defined as the size of the segments, because a corner represented by a 1.95 meters and a 0.05 meter segments is more difficult to detect as a corner built by two 0.5 meters segments. Therefore for the extraction confidence p_l the area built by the segments is used (Fig. 3.12).

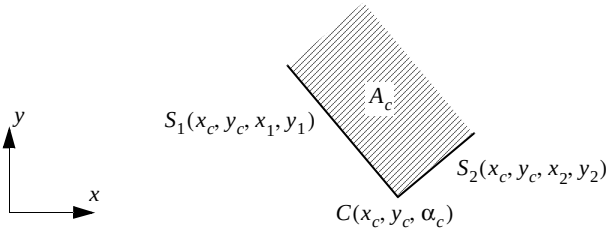

Figure 3.12: The extraction confidence is given by the area which can be built by S_1 and S_2. When this is more than one square meter, then p_l is set to 1.

This is calculated as follows:

$$A_c = \sqrt{(x_c - x_1)^2 + (y_c - y_1)^2}\sqrt{(x_c - x_2)^2 + (y_c - y_2)^2} \tag{3.26}$$

The extraction confidence p_l is then defined as follows:

$$p_l = \begin{cases} A_c, & A_c < 1 \\ 1, & A_c \geq 1 \end{cases} \tag{3.27}$$

3.2.2 Discontinuities

Discontinuities are landmarks which are typical for hallways. This can be seen in Fig. 3.13. By knowing the direction of the hallway they are very easy to extract: A step in the structures perpendicular to the hallway means that a discontinuity is present.

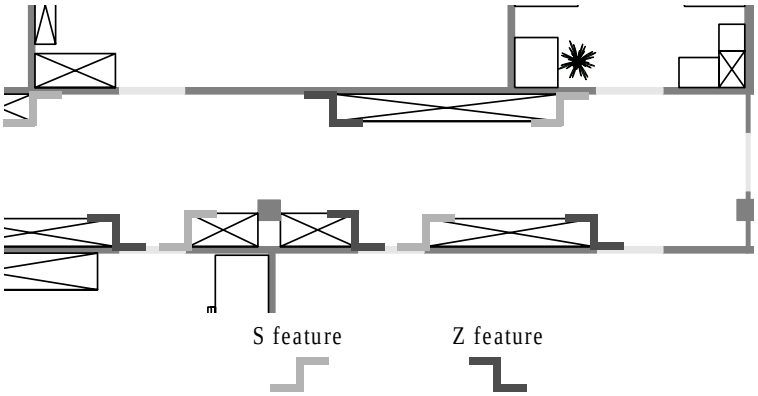

Figure 3.13: Discontinuities are typical features appearing in hallways. The can be characterized by their type, defined here as “S” and “Z”, and by the size of the step they create perpendicularly to the hallway.

3.2.2.1 Segment Based Model

Detecting steps along a hallway can be performed in many different ways. Here an approach using segments is presented. A discontinuity is given by two parallel segments which also have to be parallel to the hallway, creating a step. This can be seen in Fig. 3.14.

As with the corners in Section 3.2.1 some heuristics are used as a test to accept $D(x_d, y_d, \Delta_d)$ as a discontinuity:

$$\left|\operatorname{atan2}\left(\frac{y_1 - y_{d1}}{x_1 - x_{d1}}\right) - \operatorname{atan2}\left(\frac{y_2 - y_{d2}}{x_2 - x_{d2}}\right)\right| < \varepsilon_s \qquad (3.28)$$
$$\left|x_{d1} - x_{d2}\right| < \varepsilon_d$$

where the meaning of x_{c1}, y_{c1}, x_{c2} and y_{c2} is explained in Fig. 3.14 and they are used to determine discontinuity parameters (x_d, y_d, Δ_d) as follows

$$x_d = \frac{x_{d1} + x_{d2}}{2},\ y_d = \frac{y_{d1} + y_{d2}}{2} \text{ and } \Delta_d = \left|y_{d1} - y_{d2}\right| \qquad (3.29)$$

The constants ε_s and ε_d are estimated by experience.

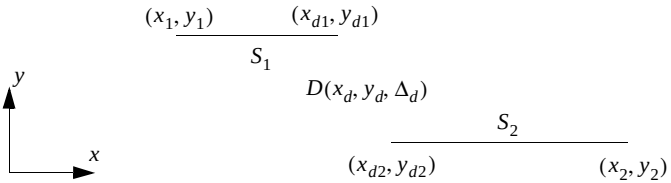

Figure 3.14: The two segments S_1 and S_2 build a discontinuity when they fulfill Equation 3.28 in a hallway which is parallel to x.

3.2.2.2 Extraction

For the extraction of discontinuities the segment extractor presented in Section 3.1.1 is used. However, before applying the extractor the laser scan is rotated to be aligned with the environment. The resulting segments can then directly be used for the tests of Equation 3.28. The result of the algorithm is shown in Fig. 3.15 with an example.

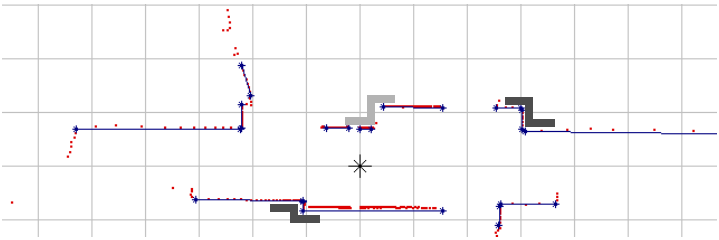

Figure 3.15: The segment extractor used in Section 3.1.1 is applied to the laser scan and couple of segments passing the test of Equation 3.28 are labelled as discontinuities.

3.2.3 Openings

Openings are the best suited landmark for topological localization because they are very representative for the topology of indoor environments. They are the best way to characterize the topology of the environment. In this section a simple approach for opening detection with laser scanner is presented. It is very computationally efficient but lacks robustness with respect to more complex methods. However, it is important to understand that while navigating topologically with a mobile robot the efficiency takes even more importance: Assuming that the presented method requires time t_1 ms and has probability p_1 of detecting an opening and that a second method has parameters t_2 and p_2 the probability of seeing the opening in a hallway is $t_v p_1 / t_1$ for the first method and $t_v p_2 / t_2$ for the second, where t_v is laps of time where the opening is visible. This means that for the case where the second method is twice as good as the first one, but requires four times the processing power, it would be better to employ the first one.

3.2.3.1 Segment Based Opening Extraction

This approach requires some assumptions which are acceptable only for typical indoors environments:

- The environment contains mainly orthogonal structures
- The scan can be aligned to the environment

Given these assumptions the approach is very simple:

- Align the scan to the environment
- Extract segments as in Section 3.2.1
- Calculate mean distance and max distance of segments along the two axes
- Apply an heuristic permitting to detect openings

The only steps which are new for this approach are the estimation of the mean and maximum distance.

For example, for positive x all the segments $s_i(x_1, y_1, x_2, y_2)$ respecting:

$$\begin{aligned} s_i(x_1, y_1, x_2, y_2) &\perp x \\ \frac{x_1 + x_2}{2} &\geq 0 \end{aligned} \tag{3.30}$$

are used to compute the positive x mean and maximum, where:

$$mean_{posx} = \sum_{s_i \perp x,\, s_i \in posx} \frac{(x_{1_i} + x_{2_i})/2}{s_i \perp x,\, s_i \in posx} \tag{3.31}$$

$$max_{posx} = max_{s_i \perp x,\, s_i \in posx} \frac{x_{1_i} + x_{2_i}}{2} \tag{3.32}$$

Given these two parameters an opening can be detected by simply applying the following test:

$$max_{posx} - mean_{posx} > open \tag{3.33}$$

where *open* is a constant determined by experience. The same equations are applied for negative *x*, positive and negative *y*. Examples of the result of the approach are given in Fig. 3.16 for an office and a hallway.

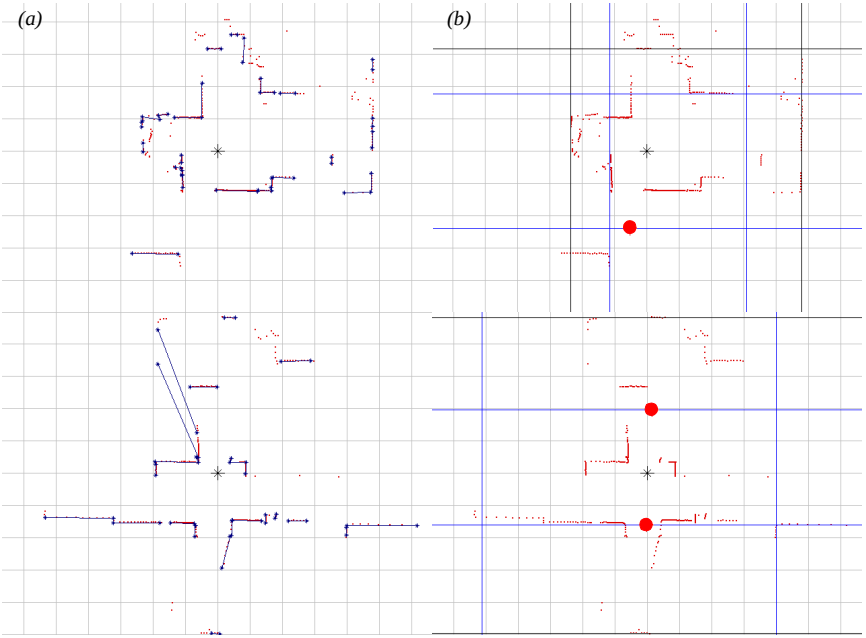

Figure 3.16: A simple approach for detecting openings (a) Segments extracted from a 360° laser scan in an office. (b) Segment mean and segment maximum allows the correct detection of the openings.

3.3 Conclusions

This chapter has presented the extraction of the features which will be used in the next chapters. Two main types of feature have been presented: Primitive (features) and high-level (landmarks). Primitive features permit a precise and simple

metric model which takes into account both the first and second moment. They are, therefore, very suitable for fully metric approaches as can be seen in the next chapter. High-level features present a higher mutual distinctiveness for the environment. This is perhaps less evident for corners which could also be modeled and used for fully metric navigation, but more for discontinuities and openings. High-level features are then very interesting for topological approaches where distinctiveness plays an important role since the approaches are mainly used for global localization.

Even if perception is not the main goal of this work the presented features are appropriate for the localization and map building approaches which will be presented later. It is also important to understand that features, beside the advantage of filtering raw data, are suitable only for environments where they are effectively present. This thesis addresses the problem of indoor navigation where the environment is mainly well structured and therefore compatible with the presented features. For outdoor, where the environment presents less structures which are easy to identify, the challenge remains more difficult. Another approach relying on other characteristics of the environment like f. e. colors instead of range, would have to be studied.

4

Multi-Sensor Data Fusion

"I was afraid that by observing objects with my eyes and trying to comprehend them with each of my other senses I might blind my soul altogether."

Socrates (469–399 BC)

400 years BC Socrates recognized that different sensors perceive the environment differently. A direct consequence of this is that environmental representation is directly dependent on sensors and perception. Another point is that merging perception from different sensors allows to correct, or at least to improve, the perception of a single one. This statement is actually twofold: Multi-sensor perception can allow a more robust perception of the surroundings, but can also permit the robust perception of an object as a combination of the perception of two sensors, even if the same object would be difficult to detect with each single sensor. These two cases are treated in the next two main sections where it is presented how to improve the perception by adding the information of a further sensor (Section 4.1) and how to detect high level features as a combination of the perception of two sensors (Section 4.2).

4.1 Fusion for Metric Localization: The Extended Kalman Filter

In this section the *Extended Kalman Filter* (EKF) is presented as a sensor data fusion framework. The EKF permits the easy integration of measures from many sensors by making some assumptions and following some simple rules. This work has been carried out with Kai Arras and the results are also presented in a journal paper [Arras01a].

4.1.1 Introduction

Kalman filter localization with line segments from range data has been investigated a decade ago [Crowley89], [Leonard92]. Vertical edges in combination with an EKF have been employed in [Chenavier92] and [Muñoz98]. The combination of these features is used in [Neira99] and [Pérez99]. In [Neira99] a laser sensor with an opening angle of 60° providing both range and intensity images was utilized while in a recent work, [Pérez99], the absolute localization accuracy of laser, monocular and trinocular vision was examined. Similar precision has been found for all the three sensor setups.

The Kalman filter acts as a position tracker whose performance is dependent on how fast the real distributions deviate from the idealized Gaussians due to the errors in the odometry. Since only a single distribution is maintained an incorrect matching of the local map onto the global map can lead to irreversible filter divergence. In such a case the robot has to be globally re-localized. On the other hand if the tracker operates fast enough, obtains precise and discriminant sensory information and no unmodeled events (i.e. collisions) take place, then it can reliably keep the robot localized because the errors remain bounded to the idealized statistical assumptions.

The contribution of this work is threefold: Firstly, the goal of the experiments with the multi-sensor setup is to determine the localization precision which is attainable with this approach. Secondly, in contrast to most of the contributions in the domain of mobile robot localization, this section presents results from extensive experiments where the practicability of this multi-sensor approach is verified under conditions which do not differ from those of typical applications. Finally, the paper introduces infinite horizontal and vertical lines as features, opposed to range-only information in [Leonard92], and segments of finite length in [Crowley89], [Castellanos96], [Neira99] and [Pérez99].

In a first set of experiments the improvement with respect to the precision when the vision information is added to the laser range finder is examined under controlled conditions. As with [Neira99] the uncertainty bounds are considered which, under the assumption of realistic uncertainty models, permit inference concerning the sought first moments of the robot position. For this the physically well grounded uncertainty models for odometry, the laser range finder and the vision system presented in Chapter 3 are used. The second experiment is the Computer 2000 event, an annual computer tradeshow in Lausanne where over four days visitors could give high-level navigation commands to the robot via a web interface. With a system up-time of 28 hours, an overall travel distance of 5 km and more than 140000 localization cycles long-term reliability under application-like conditions was the main concern.

4.1.2 The Extended Kalman Filter (EKF)

The Extended Kalman Filter [Maybeck90] is one of the most famous frameworks based on the *Bayes rule*. Other Bayes filters are the *Hidden Markov Models* [Fox98], the *Dynamic Bayes networks* and the *Partially Observable Markov Decision Processes* [Kaelbling95], [Koenig95], [Cassandra96]. The Bayesian rule permits the calculation of the *a posteriori* probability $p(A|B)$ by using the commutative characteristic of the *product rule*:

$$p(A \wedge B) = p(A|B)p(B) = p(B|A)p(A) \tag{4.1}$$

where $p(A)$ is the *a priori* probability and $p(A|B)$ describes the probability of A given all we know is B. The Bayesian rule is then given by:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)} \tag{4.2}$$

In order to use the rule the Kalman filter makes some assumptions which are also adopted by many other Bayesian approaches for localization. With these rules the information can be integrated recursively by using the *recursive Bayesian updating*. The assumptions are defined as follows:

- Firstly, the *Markov* assumption is made meaning that at any time the information collected until the current moment in time can be represented in the current state:

$$p(s_n | a_1, o_1, \ldots, a_{n-1}, o_{n-1}, a_n) = p(s_n | s_{n-1}, a_n) \tag{4.3}$$

  where s is a state, a an action and o an observation.

- Secondly, the Kalman filter assumes *independence* between measurements and errors meaning that the current observation depends only on the current state:

$$p(o_n | s_n, a_1, o_1, \ldots, a_{n-1}, o_{n-1}, a_n) = p(o_n | s_n) \tag{4.4}$$

  This is a strong assumption which actually never holds, making the filter difficult to stabilize. However, by being sufficiently conservative in the required models a stable solution can be found.

- Last, but not least, errors are assumed to be *Gaussian* and error propagation to be linear. This assumption is acceptable for range (Section 2.3 and Section 3.1.1) and angular measurements (Section 2.4 and Section 3.1.2), but is critical for the odometry (Section 2.2) where a growing uncertainty can show the limitation of the assumption.

These assumptions are very strong and cause instabilities in the filter which can lead to inconsistent estimates especially for simultaneous localization and map building, as has been shown in [Julier01].

An EKF localization cycle consists of five steps: State prediction, observation, measurement prediction, matching and estimation. These steps are briefly presented in this section and are shown in Fig. 4.1.

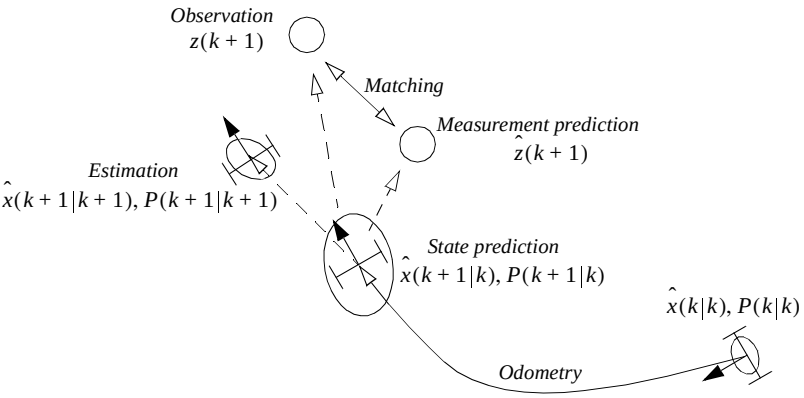

Figure 4.1: The five steps of the Kalman filter.

State Prediction. The state $\hat{x}(k+1|k)$ and its associated covariance $\hat{P}(k+1|k)$ are determined from odometry based on the previous state moments $\hat{x}(k|k)$ and $P(k|k)$. The odometry model has been described in Section 2.2.

Observation. The extracted feature parameters constitute the vector of observations $z(k+1)$ and its associated observation covariance matrix $R(k+1)$. Since measurement errors of sensors and features are assumed to be independent, $R(k+1)$ is blockwise diagonal.

Measurement Prediction. The modeled features in the map, ${}^{w}m$, are transformed into the frame of the observations. The first moments are computed by $\hat{z}(k+1) = h(\hat{x}(k+1|k), {}^{w}m)$ where $h(.)$ is the non-linear measurement model (the global-to-local transform). Error propagation is performed by a first-order approximation which requires the Jacobian ∇h with respect to the state prediction $\hat{x}(k+1|k)$.

Matching. Since the observation covariance matrix $R(k+1)$ is blockwise diagonal (independence assumption) matched pairings can be integrated in a manner which is advantageous for filter convergence. Furthermore, each pairing is integrated in an iterative way: *(i)* matching of the current best pairing, *(ii)* estimation and *(iii)* re-prediction of features not associated so far. This procedure has also been used in [Pérez99].

A pairing $p_{ij} = (z^{[j]}, \hat{z}^{[i]})$ is matched if it satisfies the validation test:

$$(z^{[j]} - \hat{z}^{[i]}) S_{ij}^{-1} (z^{[j]} - \hat{z}^{[i]})^T \leq \chi^2_{\alpha, n} \tag{4.5}$$

where S_{ij} is the innovation covariance matrix of the pairing and $\chi^2_{\alpha, n}$ a number taken from a χ^2-distribution with *n* degrees of freedom where α is the probability level on which the hypothesis of pairing correctness is rejected.

Estimation. Successfully matched observation and predictions yield the innovations $\upsilon(k+1) = z(k+1) - \hat{z}(k+1)$ and the covariance which is defined as $S(k+1) = \nabla h\, P(k+1|k) \nabla h^T + R(k+1)$. Finally, with the filter equations:

$$W(k+1) = P(k+1|k) \nabla h^T S^{-1}(k+1) \tag{4.6}$$

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + W(k+1)\upsilon(k+1) \tag{4.7}$$

$$P(k+1|k+1) = P(k+1|k) - W(k+1)S(k+1)W^T(k+1) \tag{4.8}$$

the posterior estimates of the robot position and associated covariance are computed.

4.1.3 Environmental Modeling

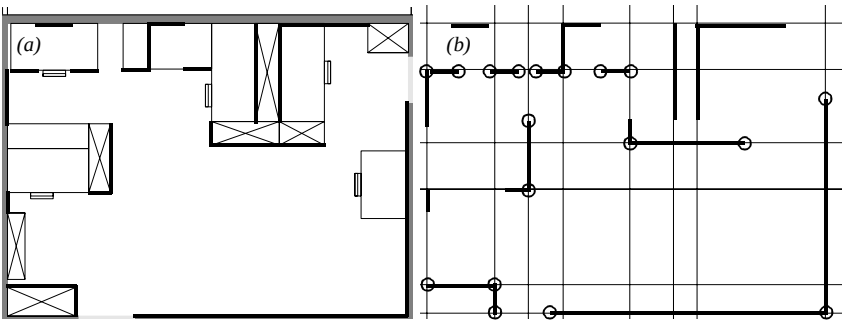

Figure 4.2: (a) An office of the Institute building. (b) The horizontal (lines) and vertical (circles) lines representing the office in the map. The black segments permit the observation of the correspondence between the two figures. This environmental model is extremely compact with a memory requirement of approximately 30 bytes per m^2.

The model of the environment used here is feature based. The approach is implemented for a typical indoor office environment. Such an environment is well

structured and, therefore, suitable for feature based approaches. The features are horizontal and vertical lines: Horizontal lines are extracted by a laser scanner as explained in Section 3.1.1 while for vertical lines the extractor presented in Section 3.1.2 is used. See also Fig. 4.2 for an example.

The features are hand measured and stored with constant uncertainty in an *a priori* map containing 191 infinite lines and 172 vertical lines for the $50 \times 30 m^2$ portion of the Institute building shown in Fig. 4.3. This is an environmental model of extreme compactness with a memory requirement of about 30 bytes $/ m^2$.

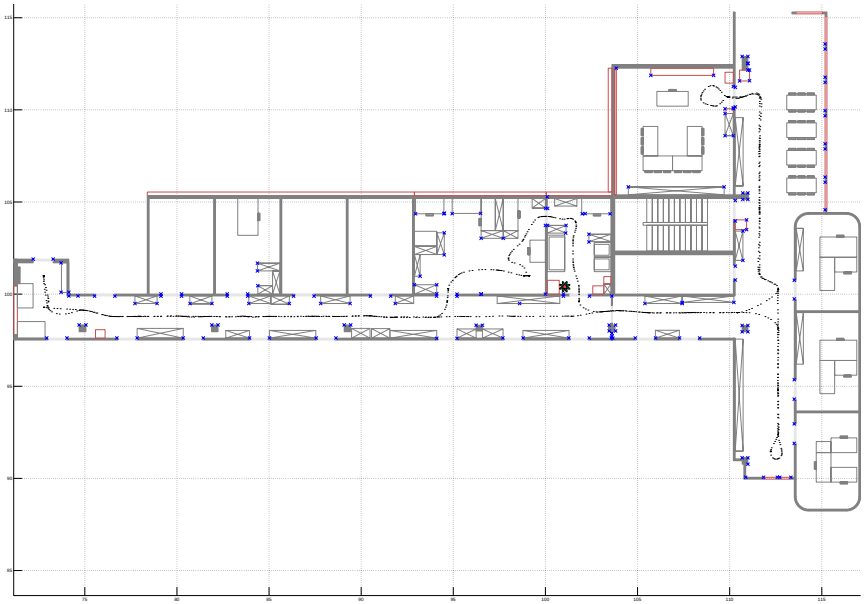

Figure 4.3: The test bed for the presented approach. The *a priori* map representing it contains 191 infinite horizontal lines and 172 vertical lines. The trajectory the robot made for the experiments of Section 4.1.6 is also shown.

4.1.4 Fusing Horizontal and Vertical Lines

By fusing two different types of features some special characteristics of the observation and matching step have to be adapted:

Observation. Both the extracted feature parameters $(\alpha, r)^T$ for horizontal lines and φ for vertical lines constitute the vector of observations $z(k+1)$ and its associated observation covariance matrix $R(k+1)$. Since measurement errors of sen-

sors and features are assumed to be independent all subsequent equations operate with 2×2-matrices for horizontal lines and scalars for vertical lines.

Matching. To optimize the filter convergence the laser observations are integrated first since they typically exhibit far better mutual discriminance making their matching less error-prone, followed by the vertical lines from the camera where often ambiguous matching situations occur. In [Pérez99] the same observations concerning feature discriminance have been reported. Starting from the same idea each pairing is integrated according to its quality in an iterative way as presented in Section 4.1.2, but for each sensor. The quality of a pairing of prediction $\hat{z}_{l,v}^{[i]}$ and observation $z_{l,v}^{[j]}$ is different for both sensors.

For the horizontal lines the quality criterion of a pairing is *smallest observational uncertainty* – not smallest Mahalanobis distance as in [Pérez99]. This renders the matching robust against small spurious and uncertain segments which have small Mahalanobis distances. The 'current best' pairing $p_{ij} = (z_l^{[j]}, \hat{z}_l^{[i]})$ is, therefore, the observation $z_l^{[j]}$ with $trace(R_l^{[j]}) = min_i$ which satisfies the validation test:

$$(z_l^{[j]} - \hat{z}_l^{[i]})\, S_{ij}^{-1}\, (z_l^{[j]} - \hat{z}_l^{[i]})^T \leq \chi^2_{\alpha,n} \tag{4.9}$$

where S_{ij} is the innovation covariance matrix of the pairing and $\chi^2_{\alpha,n}$ a number taken from a χ^2-distribution with $n = 2$ degrees of freedom. α is the probability level on which the hypothesis of pairing correctness is rejected.

The criterion for vertical lines is uniqueness. This permits the matching of unambiguous prediction-observation pairings allowing the filter to further converge before matching ambiguous pairings. Predictions $\hat{z}_v^{[i]}$ with a single observation $z_v^{[j]}$ in their validation gate are preferred and integrated according to their smallest Mahalanobis distance provided that they satisfy Equation 4.9 with $n = 1$ (subscript l become v). When there is no unique pairing anymore candidates with multiple observations in the validation region, or observations in multiple validation regions are accepted and chosen according to the smallest Mahalanobis distance.

4.1.5 On-the-Fly Localization

Localization in a *step-by-step* manner where the position estimation is performed only at a standstill is unsatisfactory for several reasons: The vehicle advances slowly and has a non-continuous movement which is not acceptable in many applications such as cleaning tasks. Furthermore, the position update rate is typically low with respect to the distance travelled. This can become critical because of the Gaussian and linear assumption of the EKF, especially for the odometry where these assumptions are acceptable only for small errors. There-

fore, continuous localization during motion – henceforth referred to as *on-the-fly localization* – is desirable. This confronts the researcher with difficulties which are present, but only hidden at low speeds, or step-by-step navigation. This includes efficiency of the implementation which should allow real-time capabilities of the method, resolution and uncertainties of time stamps the system can provide for sensory inputs and the need for compensating the distortion of sensory data caused by the vehicle movement. Time stamp quality imposes bounds on localization precision and feature matching rates whose influence is to be studied when a localization method shall prove its relevance for real-world applications.

4.1.5.1 Time Stamps

The main difference from the viewpoint of multi-sensor localization between step-by-step and on-the-fly navigation is that temporal relations of sensor observations, predictions and estimations of all sensors involved have to be maintained and related to the present. This is performed by assigning time stamps to observations and recording odometry into a temporary buffer. When sensor *A* performs its data acquisition the data receive a time stamp T_A and after feature extraction is completed the corresponding state prediction is read out from the odometry buffer. When the position estimate arrives from the Kalman filter it is valid at time T_A. Based on the odometry model presented in Section 2.2 a means is then needed to relate this old position estimate to the current position of time *t*. This is achieved by forward simulation of Equation 2.2 and Equation 2.3 from T_A to *t* using the encoders reads which are buffered for this purpose.

For a multi-sensor system care has to be taken that prediction and estimation results of one sensor are not overwritten by those of another sensor. This would be the case if each sensor would have its own EKF running independently from the others with its own cycle time yielding temporally nested updates. Nested updates are unfavorable since a slow outer update cycle (e.g. vision) overwrites the estimation results of faster running inner cycles (e.g. laser).

A sequential scheme of EKFs for each sensor is, therefore, required with the constraint that the estimates are integrated in the succession of their respective observation, enabling the steps to be performed in parallel. This is very interesting for those steps which uses resources not completely shared by all the sensors. For example acquisition of images from a CCD camera can be performed completely in parallel to any other step from the laser scanner because the frame grabber has its own processor which is able to interact with the camera and copy the new image via DMA (direct memory access) in a shared area of the main memory, without interrupting the main CPU.

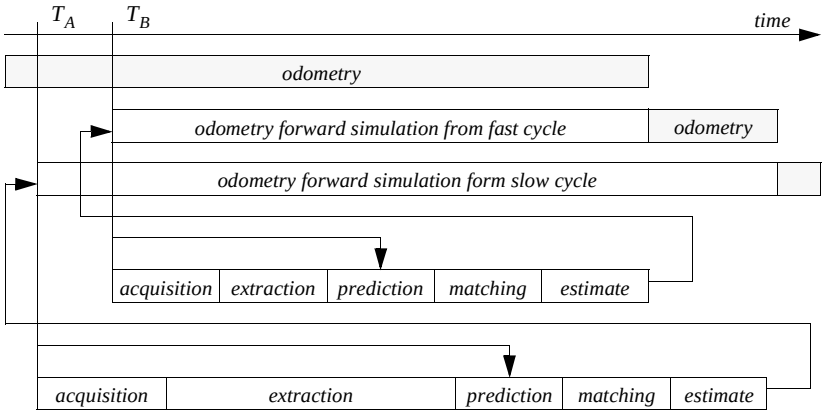

Figure 4.4: Independently running EKFs for multi-sensors are undesirable since sensors permitting fast estimates will sometimes have cycles completely nested by slow cycles from another sensor. This would completely erase the estimate of the fast estimate which is valid at T_B, when making the forward simulation of the odometry model from T_A.

4.1.5.2 Scan Compensation

The vehicle movement imposes a distortion on the raw data of the laser scans. This deformity depends on the ratio robot speed to mirror rotation velocity which is non-neglectable for the Acuity AccuRange4000LIR sensor (Section 2.3.1), because it has a mirror's rotation frequency of only 2.78 Hz. For the Sick LMS200 (Section 2.3.2) this problem is less critical, since it has a rotation frequency of 50 Hz. It is important to note that scans have to be compensated on the raw data level and not on the feature level. In the latter case the extraction method would operate with an artificially modified feature evidence.

The vehicle displacement is compensated during a scan by transforming each range reading ${}^{S'}P$ acquired in the sensor frame S' into the non-stationary robot frame R' and then into the world frame W. Followed by a re-transform into the stationary robot frame R and finally into the desired reference frame of the scan S. For a compensation on-the-fly S must be the sensor frame at the start position of a new scan. By reading out odometry each time when a new range reading arrives, it is immediately transformed by the expression:

$$ {}^{S}P = {}^{R}_{S}T^{-1}\,{}^{W}_{R}T^{-1}\,{}^{W}_{R'}T\,{}^{R'}_{S'}T\,{}^{S'}P \tag{4.10} $$

where ${}^{R}_{S}T = {}^{R'}_{S'}T$. The 4×4 matrices T are homogeneous transforms casting the rotation and translation of the general transform into a single matrix. ${}^{R}_{S}T$ is the sensor-to-robot frame transform and ${}^{W}_{R}T$ the world-to-robot transform given by the actual robot position vector x. The compensated scan receives the time stamp of S, which is the time when the scan has been started recording.

4.1.6 Experiments

As explained in the introduction two different types of experiments are presented:

- The first type of experiments are under controlled conditions. These experiments have been performed with Pygmalion, one of the robots presented in Section 1.5.2, using the Acuity AccuRange4000LIR laser scanner and a progressive scan NTSC Pulnix camera (TM 9701). The robot moved by means of a position control which did not take into account local data (no obstacle avoidance). Experiments permit the estimation of the overall precision of the robot and the evaluation of the improvement achieved by adding the information from the CCD camera to the laser-only localization.
- The second set of experiments have been defined by the visitors of the 'Computer' tradeshow, an annual fair for computer hard- and software at the *Palais de Beaulieu* exposition center in Lausanne, Switzerland. The test platform was still Pygmalion, but for this event it was upgraded to the Sick LMS200 laser scanners and used the obstacle avoidance presented in Appendix A.3.

4.1.6.1 Experiments under controlled conditions

In the first set of experiments the robot was driving under controlled conditions: All runs have been performed in the evening which allowed the limitation of environmental dynamics and to control the illumination conditions because almost all corridors, offices and rooms in Fig. 4.3 are subject to sunlight which would influence the performance of the vision system. Global planning is achieved by a search algorithm in a graph which results in *via nodes* permitting to reach the goal. For local planning the robot was using a position controller for *non-holonomic* kinematics [Astolfi95] which ensures trajectories as reproducible as possible, permitting the better comparison of the two setups. The environment of Fig. 4.3 shows the floor plan of a $50 \times 30 m^2$ portion of the Institute building. In the laser-only mode and in the multi-sensor mode the trajectory has been driven five times. Care has been taken that both experiments had the same localization cycle time by limiting the implementation to 2 Hz resulting in about 950 cycles on the 140 m test trajectory. The average speed was 0.3 m/s and the maximal speed 0.6 m/s.

Table 4.1: Overall mean values of the error bounds, the number of matched horizontal lines n_l and matched vertical lines n_v per cycle, and the average localization cycle time $\overline{t_{exe}}$ under full CPU load.

	laser	laser and vision
$2\sigma_x$	1.31 cm	1.07 cm
$2\sigma_y$	1.35 cm	1.05 cm
$2\sigma_\theta$	0.92°	0.56°
$\bar{n}_l / \bar{n}_v$	2.73 / –	2.66 / 2.00
$\overline{t_{exe}}$	64 ms	411 ms

Results

The resulting 2σ-uncertainty bounds of the a posteriori position estimates are illustrated in Fig. 4.5. Both cases generally reflect a very high localization accuracy in all three state variables. Sub-centimeter precision is approached. Table 4.1 shows the overall means of error bounds $2\sigma_x$, $2\sigma_y$, $2\sigma_\theta$, number of matched horizontal lines $\bar{n}_l$, number of matches vertical lines $\bar{n}_v$, and execution times $\overline{t_{exe}}$. The error bounds define that, based on the uncertainty model, the robot is with a 95% probability within twice this value. The vision information contributes to a reduction of this uncertainty in *x* and *y* equally (-20%), but particularly in the orientation (-40%). This although the average number of matched vertical lines is moderate. The cycle times $\overline{t_{exe}}$ include sensor acquisition and the mean duration for calculating an iteration under full CPU load on the currently used hardware.

Discussion

Even carefully derived uncertainty bounds do not necessarily permit inference concerning the first moments since the estimation error could be arbitrarily large without being noticed (estimator inconsistency). We argue that the simple fact that the robot always succeeded in returning to its start point proves the correctness of these bounds. In fact, they are even conservative estimates since the true bounds could be better. Otherwise the robot would have gone lost due to a lack of matches caused by first moments drifted away from the true values. Ground truth information as in [Pérez99] would be preferable, but is impractical and expensive to obtain for long-term experiments such as those presented here. Positioning accuracy of the vehicle at the end point has been measured and further confirm the values in Table 4.1. These results are very similar to the accuracy reported in [Pérez99]. In [Gutmann98] the experiments yielded a maximal precision of approximately 5 centimeters for Markov localization whereas scan matching produced, in the best case, estimates of millimeter-accuracy.

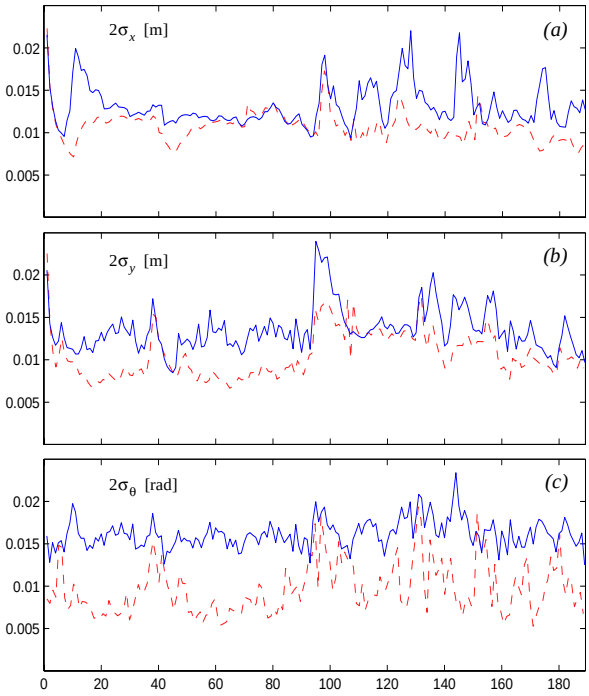

Figure 4.5: Averaged 2σ–error bounds of global x (a), y (b) and θ (c) a posteriori uncertainty during the test trajectory (showing only each 5th step). In each mode five runs have been made. Solid lines: Laser range finder only. Dashed lines: Laser and vision. In some cases the uncertainty in the multi-sensor mode is greater than for the single-sensor setup. This is possible since the values are averaged over five runs containing noise on the level of matched features.

Matching vertical lines is, due to their lack of depth information and their frequent appearance in compact groups, particularly error-prone. For example, door frames commonly have multiple vertical borders which, dependent on the illumination conditions, produce evidence for several closely situated vertical lines. In the matching stage they might be confronted with a large validation region, position bias from odometry, or time stamp uncertainty making the predicted model line difficult to identify. In such ambiguous matching situations incorrect pairings are likely to occur and, in fact, they have been occasionally produced in the multi-sensor experiments. But their effect remains weak since these groups are typically very compact.

However, this lack of discrimination in the presence of time stamp uncertainty is the main cause of reproducible failure of vision-only navigation (i.e. robot went lost). With the frame grabber in use it is difficult to identify the precise instant when the image is taken. Also odometry quantization, in our case 5 ms, became noticeable particularly during fast turns.

4.1.6.2 The Computer 2000 event

Figure 4.6: Over the four days of the Computer 2000 event visitors could define via web navigation goals in the corridors and offices of our Institute building at EPFL. The environment which was attainable by Pygmalion was $50 \times 30 m^2$ in size and contains twelve offices, two corridors, the seminar and the mail room. The robot was navigating for seven hours each day during normal office hours with typical environmental dynamics.

The 'Computer' tradeshow is an annual fair for computer hard- and software at the *Palais de Beaulieu* exposition center in Lausanne, Switzerland. Our laboratory was present over the four days, May 2nd to May 5th 2000, giving visitors the opportunity to control Pygmalion by means of the web-interface shown in Fig. 4.7. The robot itself was at EPFL in the environment illustrated in Fig. 4.3 and 4.7.

The Computer 2000 event was the final testbed for our localization system where we were mainly interested in long-term reliability under application-like conditions. The setup was active during normal office hours with an average of about 7 hours per day. The environment exhibited typical dynamics from people, doors, chairs and other robots, as well as daylight illumination. Several doors open into the corridor (see Fig. 4.6b). Travel speed has been limited to 0.4 m/s since the

robot shares its environment with persons some of them not familiar with robotics. In this case for local planning the obstacle avoidance strategy based on the laser range data was active during the event (see Appendix A.3, or [Persson00]). The web-interface (A.4) allows one to give global navigation commands (e.g. an office) and local navigation commands (a (x, y)-point, or an orientation) to the robot.

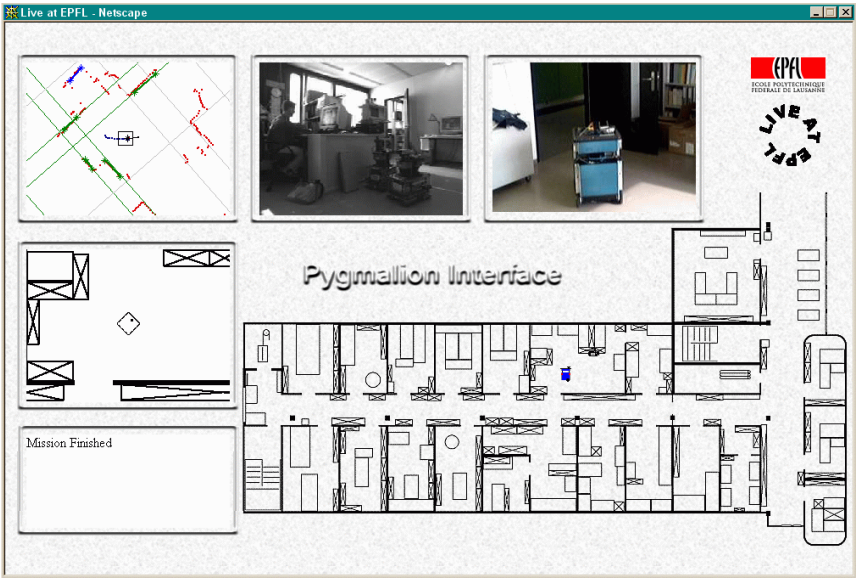

Figure 4.7: The Pygmalion web-interface, a plug-in-free Netscape application. It provides context-sensitive menus on the map and all sub-windows with intuitive click-and-move-there commands for robot goal definition. Four different real-time streams constitute the visual feedback on the current robot state: An external web-cam (top-right), an embarked camera (top middle), raw data from the laser range finder together with predicted, observed and matched features (top-left) and the robot animated in both its local model map (left middle) and global map (center). By clicking onto the map an office is defined as the destination. Clicking onto the camera image turns the robot and clicking on the laser scanner image defines a goal in the (x, y)-plane.

Results

The event statistics of Computer 2000 is shown in Table 4.2.

Table 4.2: Overall statistics for the Computer 2000 event.

Hours of operation	28
Environment size	$50 \times 30 m^2$
Environment type	office, unmodified
Overall travel distance	5013 m
Average speed	0.2 m/sec
Travel speed	0.4 m/sec
Number of missions	724
Number of localization cycles	145433
Number of lost-situation	0
Number of collisions	~ 10

A mission is either a global, or a local navigation command from the user interface. A lost situation is defined as a mission whose goal could not be attained due to a localization error and which required manual intervention to re-localize the robot. We do not count missions where the robot went lost due to a collision with an invisible object (e.g. glass door, or object lower than the beam height of the scanner) and where the robot was already lost (after such a collision).

When the vehicle is surrounded by people, or is in a less structured part of the environment it happens that there are no matched features during a certain period. We counted 14 of 724 missions where the robot had no matches over a period of 10 seconds, 21 missions where it had no matches over 5 seconds. None of them required manual intervention during, or after the mission.

Discussion

These positive results were further underlined by the feedback we received from the large number of visitors during Computer 2000. In particular, they enjoyed the easy-to-use interface which allowed anybody to control a mobile robot and discover our institute building over the internet.

Another important result of the Computer 2000 event is that under these experimental conditions vertical lines performed poorly. In addition to the shortcomings already stated in the first set of experiments, they are:

- Environmental dynamics: When navigating with a robot in a populated office environment, its free space and hence its vision sensor are often blocked by obstacles (e.g. co-workers, doors; see Fig. 4.6). This results in two independent problems. Firstly, a blocked sensor cannot contribute to the localization update. This is critical for the vision system which, in contrast

to the laser scanner, has a limited angle of view. Secondly, when avoiding these obstacles the robot turns typically with high angular velocities which in combination with the time stamp uncertainties and the low feature discrimination is likely to produce false matches.

- Illumination conditions: It is obvious that the illumination in an office environment cannot be controlled to suit best the needs of a vision sensor. In particular, both corridors have big windows at their end. The camera is heading to these windows from a relatively dark corridor when navigating in this direction.

Even by introducing a unary constraint taking into account the laser measurement to verify if a predicted vertical line is visible from the current robot position (see Fig. 4.8) the conclusion is, that with the presented setup, vertical lines are features of insufficient robustness with respect to environmental dynamics, time stamp uncertainty, changing illumination conditions and mutual discrimination. In general, a trade-off between the robustness of a feature and the computational effort to obtain it has to be found. This in turn influences how many features can be used for localization per time, or distance travelled. Vertical lines appeal by their simple availability and the modest requirements of computational power compared to many other visual features.

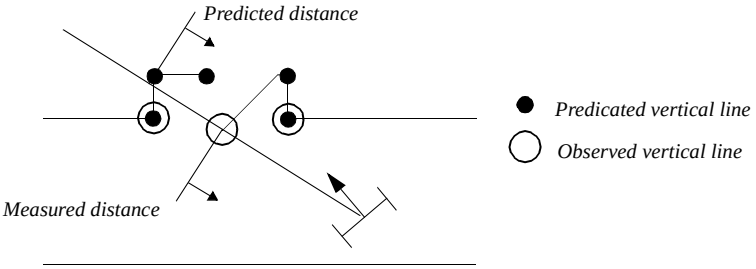

Figure 4.8: The observation of the open door looks like the predicted one for the vision system which has no range information. By comparing the predicted distance to the laser information this false match can be rejected. This can be defined as a unary constraint.

Finally, the necessity of this type of experiments has to be emphasized. They reflect the ‘real case’ and are, therefore, indispensable when a navigation approach shall prove its real-world performance. ‘Laboratory-experiments’ where conditions have been carefully controlled, only yield optimistic bounds for the practicability of a method. It is clear that these kind of experiments always remain in a certain dependency upon the specific robot setting. Especially a different

vision/frame grabber system, a turret keeping a constant camera orientation, or an omnidirectional camera system could have performed better under the same experimental conditions. However, we rate it as not surprising that during this work an approach which in the beginning seems promising, and which has been successively applied by several researchers before, turns out to be partially incompatible with the requirements of application-like conditions.

4.1.7 Conclusions

In this section an approach for localization fusing geometric feature data from a 360 degree laser range finder and a monocular vision system has been presented. The features allow for an extremely compact environmental model of only 30 bytes $/ m^2$ memory requirement. This is contrasted by a positioning accuracy close to sub-centimeter and small localization cycle times. These results were obtained with a fully self-contained autonomous system where long-term tests with an overall length of more than 6.4 km and 150000 localization cycles have been conducted.

Already with a moderate number of matched features the vision information was found to further increase this precision, particularly in the orientation. However, the limitations encountered with this feature motivate the introduction of constraint-based matching strategies as in [Muñoz98]. In spite of the reliability of the presented position tracking approach collisions can never be excluded completely, especially when the robot is employed in public places with such a simple sensor configuration. Furthermore, due to the fact that the Kalman estimator is unimodal (single hypothesis tracking) the robot has no means to detect and recover from such a lost situation without an arbitrary heuristic. This shows that the need of global localization (multi hypothesis tracking) can only be limited, but not avoided even by a very robust, but still unimodal, localization approach.

4.2 High-Level Features from Primitives

As explained in the introduction a multi-sensor setup can be used for a better perception of the environment, as in the last section, but also to better recognize more complex features. In this section a further improvement of the corner detector is presented. Then an approach focusing on door detection is shown as a further possible multi-sensor data fusion application.

4.2.1 Data Fusion: In Which Frame?

In this chapter vertical lines, as presented in Section 3.1.2, are the only features taken into account from the CCD camera. These features are defined by a single parameter φ, representing the angle of the extracted feature in the sensor frame.

As no range information is given there is no means to transform the detected feature in another frame. With the Kalman filter this was not a problem since the features are used separately in their own sensor frame. For object recognition the sensor data from each sensor have to be transformed in the same frame. Therefore, the fusion of vertical lines with other information from the laser scanner has to take place in the camera frame. All the other data are transformed in this frame and are fused there, as shown in Fig. 4.9. As soon as an object is detected its position can be transformed back into the robot frame.

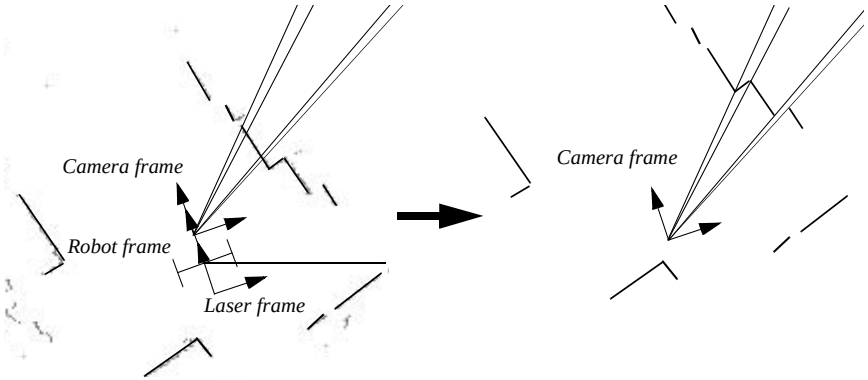

Figure 4.9: When using vertical lines which are defined by a single angular parameter φ in sensor coordinates, sensor data fusion for object recognition has to take place in the camera sensor frame. For this the laser features are transformed into the camera frame.

4.2.2 Corners

The idea is to try to extract the angle φ from the CCD camera in order to collect more evidence of the presence of the corner detected by the laser scanner. This is shown in Fig. 4.10, without many details concerning the corner model which has already been presented in Section 3.2.1.

This further information can be used for metric approaches in order to reduce the uncertainty of the detected corner $C(x_c, y_c, \alpha_c)$: As in the above presented approach for localization the angle information φ and its associated observation covariance matrix $R(k+1)$ can be integrated in the estimation of the corner position (not the robot position as in Section 4.1.4) to increase the precision and, therefore, reduce the estimation uncertainty.

The same approach can be used for topological approaches where the vertical line can be used to model the extraction confidence. This can be performed by

defining the extraction confidence as a volume, instead of an area as in Section 3.2.1, by using the vertical line as the third dimension.

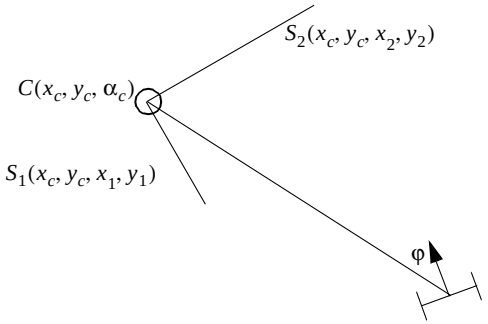

Figure 4.10: A corner as a combination of two segments and a vertical line. This model can be used for both metric (to increase the precision and reduce the uncertainty) and topological approaches (to increase the extraction confidence).

4.2.3 Doors

A similar approach can be used to detect more complicated landmarks such as, for example, doors. In the literature door detection (f.e. [Lanser92]) is a wide studied and highly complex problem. As stated in Chapter 3 feature extraction is not the goal of this work, however, this section permits the evaluation of the advantage of multi-sensor data fusion with a concrete example.

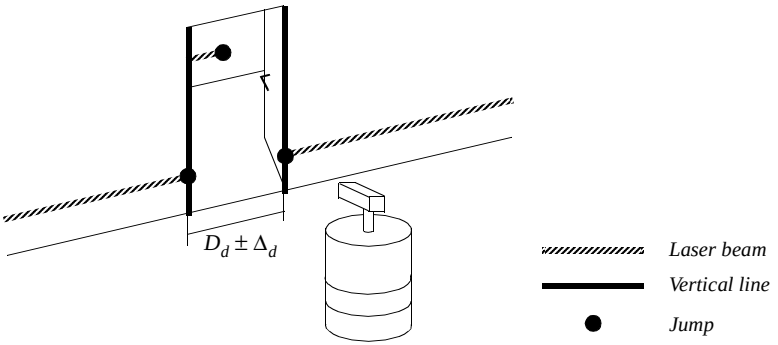

Figure 4.11: Extracting doors in a perfect world. When vertical lines and corresponding jumps are found within a pre-defined distance $D_d \pm \Delta_d$, it is very probable that a door as been detected.

4.2.3.1 Model

A door is modeled as a set of vertical lines which can appear at the same position where jumps in the range are present. When couples of vertical lines / jumps are at a predefined distance they can represent a door.

Actually the problem is much more complex due to occlusions which have to be taken into account. In Fig. 4.12 some cases are shown.

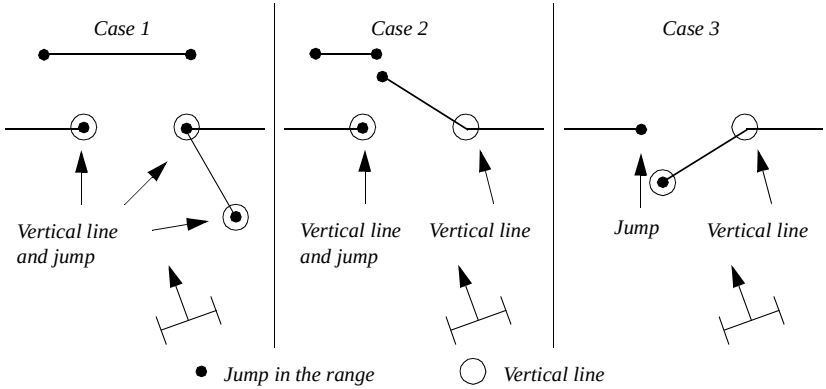

Figure 4.12: Model a door with simple features is complicated. Case 1 is the best case where no occlusions appear. Case 2 causes more problems because the door is not completely open. A closed door would even be worse. Case 3 presents an occlusion in combination with a semi-opened door.

4.2.3.2 Jump Detection with Range Data

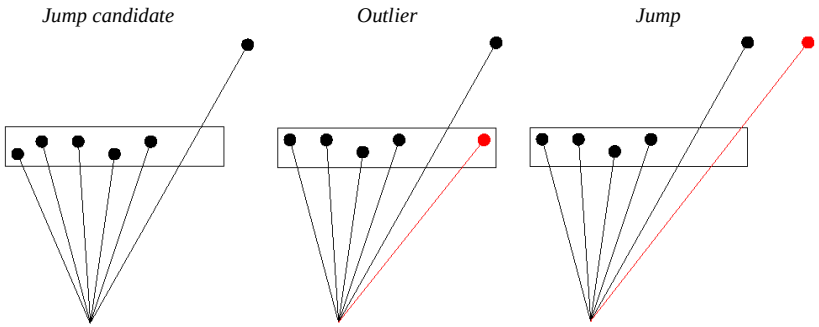

Figure 4.13: Jump detection with a simple approach permitting to reject some outliers.

Jump detection in a laser scan is a quite simple procedure. However, the scan has to have a good quality because, otherwise, outliers would create false jumps. Jumps can be detected by a segmentation algorithm as in Section 3.1.1 or Section 3.2.1 or by an even simpler algorithm: The jump can be detected by taking into account a subset of range points and searching in this subset. Outliers can furthermore be detected by means of simple heuristics as presented in Fig. 4.13.

Fig. 4.14 shows two examples of jump detection, one for a scan taken with the Acuity AccuRange4000LIR and another with the Sick LMS200.

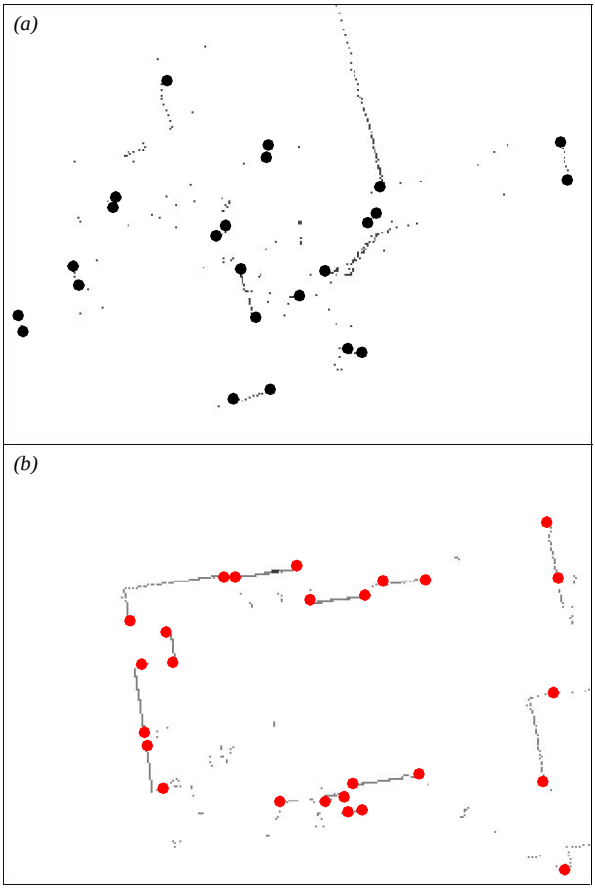

Figure 4.14: (a) Jump detection with a scan from the Acuity AccuRange4000LIR. Outliers are a problem, even by using a filtering approach. (b) With the Sick LMS200 the jumps are well extracted.

4.2.3.3 Experiments

In contrast to Section 4.1 only few experiments have been performed. They have to be interpreted as a means to learn the characteristics of this approach. They permit the inference of some statements which will be helpful for the implementation for a real application.

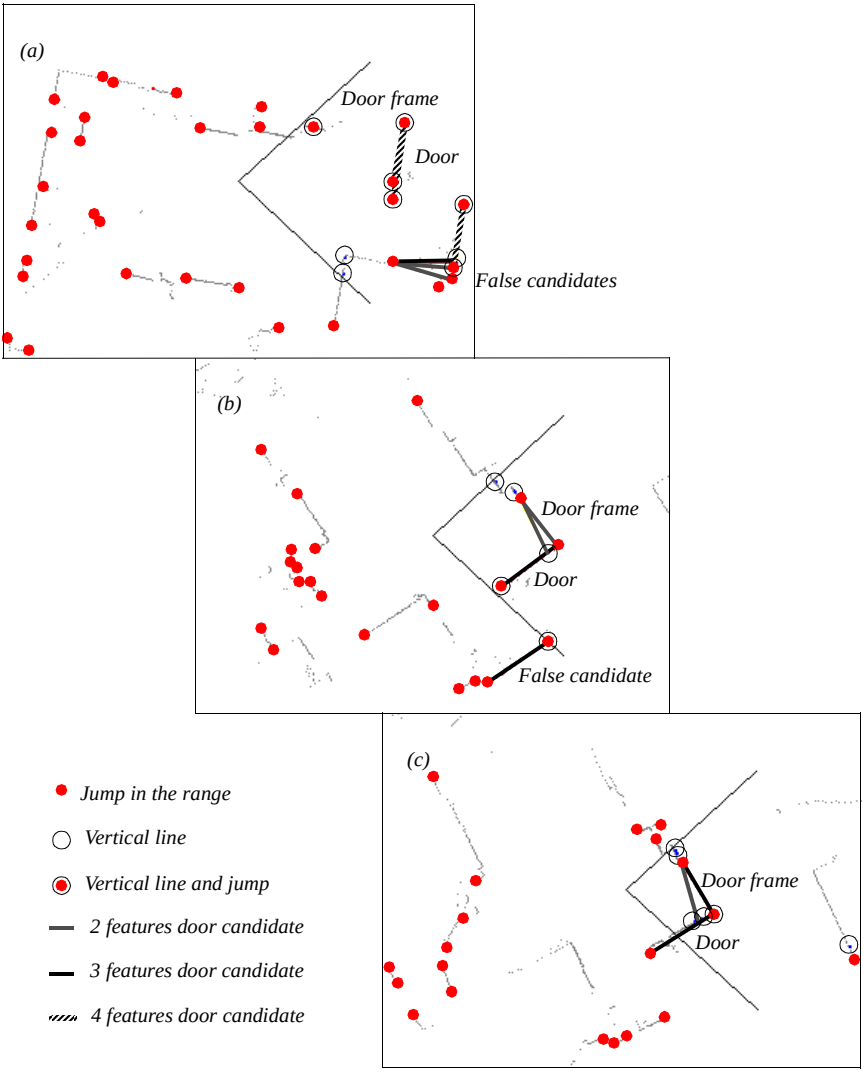

Figure 4.15: Three example of door detection with a multi-sensor setup.

Results

In Fig. 4.15 three examples of the experiments are shown. Door candidates are those set of two features containing at least one vertical line and having the pre-defined distance $D_d \pm \Delta_d$. As explained in Fig. 4.11 the best case is when both the jump and vertical line are detected as in Fig. 4.15a where a 4 features candidate is present for the door. However, in the same image a false 4 features candidate is also present. In Fig. 4.15b and Fig. 4.15c, no 4 features candidates are found but all the candidates represent either the door frame, or the opened door correctly.

Discussion

This approach has shown how information from different sensor can be integrated in order to better perceive an object. Doors have been modeled by a simple set of features with a pre-defined distance between them. The experiments show that doors can be detected, but it is difficult to distinguish them from false candidates. The lack of discriminance between correct and incorrect candidates can be reported to the lack in the model of both the jump and line features. By analyzing the results in more detail, it can be seen that, by taking into account the size of the jumps and the height of the vertical lines, the candidates could be discriminate much better. For example the 2 and 3 features false candidates of Fig. 4.15a could be rejected by taking into account the size of the vertical lines which are given by a small furniture. Nevertheless, with this model it is impossible to reject the incorrect 4 feature candidate of Fig. 4.15a which is a furniture with the same size of the modeled doors and, therefore, looks like a closed door for the system.

4.3 Conclusions

In this chapter, the Kalman filter has been presented as a means to integrate information from different sensors for position estimation. By following some simple rules avoiding that prediction and estimation results of one sensor are overwritten by those of another sensor, information from any arbitrary sensor can be integrated for the metric estimation of the robot pose. Long-term experiments have been performed with a fully self-contained autonomous system demonstrating empirically, but in an exhaustive way, the quality of the presented approach.

Another possible application of multi-sensor data fusion is object recognition and measurement. Two examples have been presented where both corners and doors are extracted and measured by taking into account information from both the laser scanner and the vision system.

5

Hybrid, Metric - Topological, Localization

"Nothing exists until or unless it is observed."

William Burroughs (1914–97)

"Where am I?" is the question the robot will have to answer in this chapter. In order to do this it has to observe its surroundings. As soon as it sees something the machine has to recognize what is the object it sees and relate it to its own knowledge. In this chapter we want to give the robot the knowledge concerning the features in the environment in the form of a map which exists because it has been observed and measured by the human.

The approach illustrated in this chapter is also summarized in a paper [Tomatis01b] which has been presented at the IEEE International Conference on Robotics and Automation (ICRA) in Seoul, Korea in May 2001.

5.1 Introduction

Successful navigation of embedded systems for real applications relies on the *precision* that the vehicle can achieve, the *robustness* against lost situations and the *practicability* of their algorithms with the limited resources of the autonomous system. This problem has been faced with approaches which can be separated into two categories: *Metric* and *topological*.

In the literature there is some confusion concerning the definition of the metric and topological concepts. Therefore, before analyzing the related work the two paradigm are explicitly defined as they will be used throughout this work:

- Metric approaches permit the robot to estimate its (x, y, θ) position. In this category a further distinction can be made by distinguishing between continuous and discrete (grid-base) representation of the environment.
- Topological based methods represent the position of the robot as a location in the environment, but without precise metric information.

5.1.1 Related Work

Metric approaches based on Kalman filtering [Maybeck90] permit high accuracy and low complexity when using line based features from ultrasonic sensors [Crowley89], [Leonard92], CCD camera [Chenavier92], or both laser scanner and CCD camera [Pérez99], [Arras01a]. Furthermore, maps for this type of approach are very compact and directly extensible with feature information from different sensors. They have also proven their robustness in large, application-like experiments such as in Section 4.1. Nevertheless, their solution remains unimodal (single hypothesis tracking). This means that an unmodeled event (i.e. collision) could cause a *lost situation* (i.e. a situation where the robot cannot match features correctly anymore because the difference between the estimate and the real position is too high) from which the system is unable to recover. Some recent works, such as [Arras01b], propose *Multi Hypotheses Tracking* (MHT) with EKFs. These approaches are then precise and robust. However, when the robot has to relocalize itself localization during motion is infeasible due to the computational complexity of this task. Furthermore, the question of how to deal with this kind of multi hypotheses when planning remains open.

Metric approaches using raw range data for *scan matching* are also unimodal. They find out the best alignment of the local measurement in the global map where both maps are a set of sensed points [Lu94], [Gutmann96]. The robot position is also represented by a single Gaussian which makes scan matching a local method and a simplified Kalman filter is used to calculate the posterior state. Compared to feature-based approaches scan matching techniques usually operate with memory-intensive maps since the environment model consists in raw range data recorded from a set of reference positions.

Other metric approaches, such as those based on *Markov localization* [Fox98], are multimodal (i.e. robust against lost situations) and precise when small grids are used, but require off-board processing because of their computational complexity. Even if the *Monte Carlo localization* [Dellaert99] proposes a promising improvement for the efficiency by using a sampling-based method (particle filter) which can represent arbitrary distributions a trade-off between precision and real-time embedded computability remains to be found as soon as multimodality is introduced in metric approaches.

On the other hand, topologically based methods are very robust but lack in precision. The main idea of only using the topology of an environment [Kuipers87] causes a loss of the metric paradigm which is indispensable for guaranteeing precision. Nevertheless, the robustness of these approaches has often been proven: The success of Dervish at the 'AAAI robot contest' with its *State Set Progression* in 1994 [Nourbakhsh98] followed by the formalization of the *Partial Observable Markov Decision Process* (POMDP) [Kaelbling95], [Koenig95], [Koenig98] represented the start of this type of approach. POMDP methods are used and improved for optimal and computational acceptable decision making [Cassandra96].

The main characteristics of all these approaches are summarized in Table 5.1. The first row permits the observation of how much data from the sensor is used for each approach. Those using features reject some raw data, however, as explained in Chapter 3, this can be viewed as a filtering step permitting the handling of dynamic in the environment. To distinguish between the meanings of feature and landmark is not easy. As in Chapter 3, primitive features (Section 3.1) are defined as feature while high-level features (Section 3.2) are defined as landmarks. A '+' in precision means that the accuracy of the approach is just bounded by the quality of the sensor data not by the approach itself. Robustness is defined, in this case, as the quality of avoiding lost situations (multi hypotheses tracking). Montecarlo localization is a special case because, in order to be practicable, the number of particles in the filter have to be reduced after bootstrapping, making it relatively vulnerable to lost situations. As can be observed in Table 5.1 currently there are no approaches fulfilling all the previously presented characteristics: *Precision*, *robustness* and *practicability*.

Table 5.1: A brief resume of the best known approaches for mobile robot localization. '+' means good, '-' bad, '=' something between.

	Full metric EKF	MHT EKF	Topological POMDP	Grid based Markov	Particle filter Monte Carlo
Sensor information	features	features	landmarks	raw data	raw data
Amount of used data	=	=	-	+	+
Env. dynamic	+	+	+	-	-
Precision	+	+	-	=	+
Robustness	-	+	+	+	=
Practicability	+	=	+	-	=

In order to have an approach permitting the fulfillment of all the requirements stated before characteristics from both the metric and topological universe may be combined together. In [Castellanos00] a basically metric based approach with

absolute localization is extended to include a localization relative to a local reference frame. This results in a two level abstraction which remains metric, but embodies a topology in the constellation of the local frames. In [Thrun96] the approach consists of extracting a topological map from a grid map with a Voronoi-based method. In this case the topological navigation permits a gain in efficiency with respect to the metric grid based navigation. In [Duckett99] an approach for both global localization and position tracking using a topological map augmented with metric information is presented. The method is based on histogram matching of ultrasonic range data.

This chapter also proposes an integration of both the metric and topological paradigms, in order to gain the best characteristics of both universes. However, instead of adding new characteristics to an already existing approach, as with the previously presented approaches, here two complementary approaches are combined together in a natural way. Furthermore, a goal which is not directly treated in this chapter, is to have a solution which is extensible to simultaneous localization and map building. This will be presented in the next chapter.

5.2 Environmental Modeling

The presented model embodies both metric and topological representations. The metric model consists of infinite lines which belong to the same place. These places are related to each other by means of a topological map which is composed of nodes representing topological locations and edges between nodes. The model of the environment is then characterized by two different levels of abstraction (see Fig. 5.1):

- Places are defined by local metric maps which allows metric navigation within the neighborhood.
- To move from one place to another the system switches from metric to topological when leaving the place, navigates by means of a topological state estimator within the global topological map and switches back to metric when reaching the goal location.

The only requirement specific to this model is to have a *detectable metric feature* when travelling from a topological node to a metric place. This permits the system to determine the transition point where the change from topological to metric has to be executed and allows robust initialization of the metric localization (i.e. relocation). Given this metric feature local metric maps can be placed anywhere in the environment. The robot can then navigate topologically in the large and as soon as it needs precision for its tasks it switches to metric. Furthermore, global metric consistency is not needed because local metric maps are only topologically related to each other, not metrically.

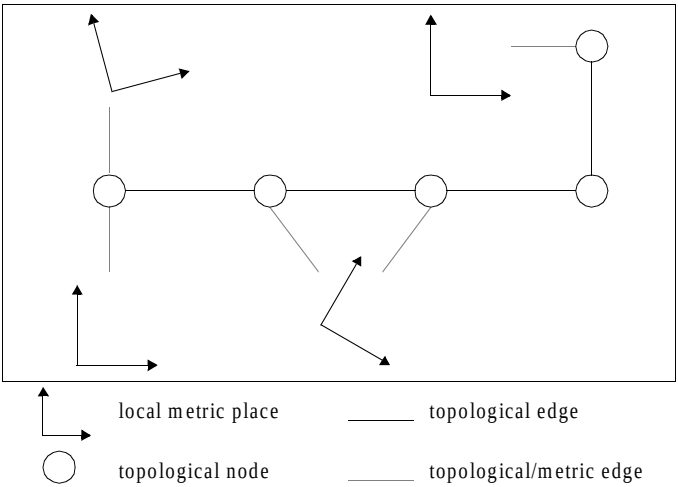

Figure 5.1: The environment is represented by places given by their metric maps and nodes representing topological locations. When travelling from a node to a place, the system switches from topological to metric and vice-versa.

Metric localization is performed with a widely tested implementation of the EKF presented in Section 4.1.2. As explained previously this approach guarantees high precision with low complexity and memory requirements. Topological navigation uses a POMDP based state estimator [Cassandra96]. This permits efficient planning in the large, has an advantageous symbolic representation for man-machine interaction and is robust against lost situations thanks to its multimodality. By looking at Table 5.1 it can be observed that these two approaches are the only ones which are practicable for fully autonomous, self-contained embedded systems. Furthermore, they are complementary in their characteristics (precision and robustness) meaning that if the best characteristics of both approaches can be combined in a coherent approach all the suitable characteristics would be fulfilled.

5.2.1 Implementation Related Assumptions

Although the presented model is general and flexible it is preferable to make some environment dependent assumptions when implementing it on a real robot. In our case the experimental test bed is a part of the Institute building. This environment is mainly composed of offices, meeting rooms and hallways. It seems an acceptable assumption to expect that the robot will have to be very precise in the rooms where most of its tasks have to be executed. While navigating in the large

(i.e. hallways) precision with respect to the features is less important but robustness and global consistency take an important role. Therefore, the current implementation uses local metric maps for offices and rooms in general and a global topological map which connects them together.

Furthermore, only four directions of traveling are employed: N, E, S, W. This implies the assumption that the environment is orthogonal which is the case for most office buildings including the Institute building where the robot is functional. It is important to understand that the above mentioned limitation is not an inherent loss of generality because it is only a simplification for the current implementation and not a general requirement of the algorithm.

Due to the fact that this implementation is the first attempt to validate the presented model empirically the perception is arbitrarily chosen by taking into account the characteristics of the environment. The features employed are then as simple as possible and might be sub-optimal, especially for landmark detection. Nevertheless, they permit extensive testing of the proposed hybrid (metric/topological) approach by means of a single sensor: A 360° laser scanner.

5.2.2 Local Metric Maps

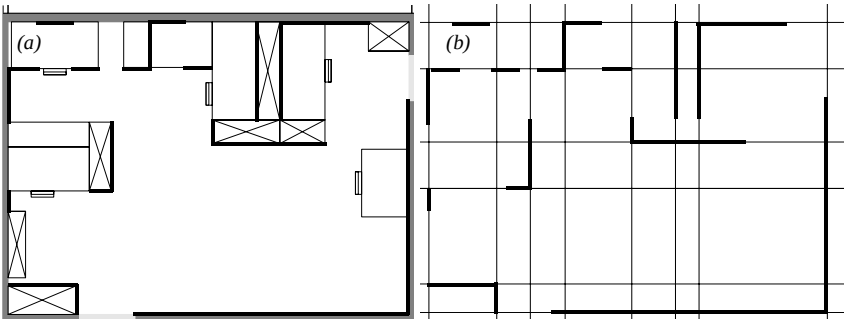

Figure 5.2: (a) An office of the Institute building. (b) The lines representing it in the local metric map. The black segments enable the observation of the correspondence between the two figures. This environment model is extremely compact with a memory requirement of approximately 10 bytes per m^2.

The features used for metric environmental representation are infinite lines. As already stated in Section 3.1.1 they are less informative than line segments but permit a very compact representation of structured geometric environments requiring approximately 10 bytes per m^2. In Fig. 5.2 the same office which has been presented in Section 4.1.3 is shown with the lines used for its local metric

map. There are two main differences with respect to the model presented in Section 4.1.3: Firstly, here only horizontal lines are used and, more importantly in this case the lines in Fig. 5.2 represent a complete (local) map. For the fully metric model of Section 4.1.3 they were just a sub-set of the global metric map.

5.2.3 Global Topological Map

The topological map can be viewed as a graph. In each node the information concerning the visible landmarks and the way to reach the connected locations/ places is stored. The landmarks for the topological representation are typical for hallways in office environments:

- Discontinuities perpendicular to the direction of travel in the hallway. They are characterized by the form (S and Z) and the size of the step as presented in Section 3.2.2.
- Openings, detected with the approach proposed in Section 3.2.3, are used for both state estimation and model transition. Therefore, they are fundamental for this approach.
- Hallways perpendicular to the direction of travel creating an intersection are also very helpful landmarks because of their distinctiveness.

In Fig. 5.3 the graph representing the topological model is viewed for a portion of a hallway of the Institute. Landmarks are linked to the closest node from which they are visible.

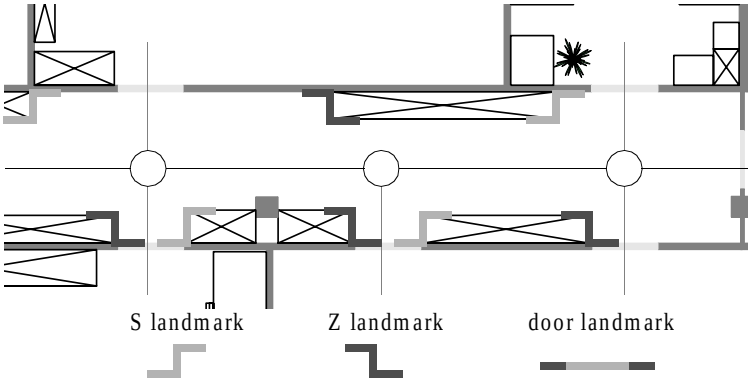

Figure 5.3: The topological map is represented by a graph. Each node contains the information concerning the visible landmarks and the way to reach the adjacent nodes.

5.3 Metric Navigation

This section is just a summary of the approach presented in Section 4.1 because for local metric maps exactly the same approach is used. The only main difference is that, in this case, only horizontal lines (not horizontal and vertical) are used. The main characteristics are summarized here in order to have a direct overview:

- Localization: The EKF presented in Section 4.1.2 is used. It allows the integration of information from the sensors during motion. Prediction is given by the odometry model presented in Section 2.2. Horizontal lines are extracted by means of the approach of Section 3.1.1.
- Global planning: In this case global planning is actually just global in the local map meaning that some points in the local map are defined to better reach the doors and not completely visible places.
- Local planning: The robot navigates locally by means of a motion control algorithm (Appendix A.3) which fulfills the role of both position controller and obstacle avoidance: It reaches the given (x, y, θ), or (x, y) goal by planning a collision free path (with respect to the current local data), and reacting to the dynamic environment either by merely replanning the path, or by changing heading direction and replanning when an object appears in front of the robot.

5.4 Topological Navigation

For topological navigation a *Partially Observable Markov Decision Process* (POMDP) model is used. A detailed description of this approach is to be found in [Cassandra96] and [Kaelbling95].

5.4.1 Localization

The topological state estimator calculates the probability of being in a state, given the last probability distribution and the current observation and action. However, it does not take into account the heading of the robot. This information is necessary for two main reasons: For local planning the robot has to know the heading and the state estimator itself has to know the robot direction in order to distinguish between the four directions of travel allowed by the topological model (see Section 5.2.3).

5.4.1.1 State Estimator

Given a finite set of environment states S, a finite set of actions A and a state transition model T the model can be defined by introducing partial observability. This includes a finite set O of possible observations and an observation function OS, mapping S into a discrete probability distribution over O. $T(s, a, s')$ repre-

sents the probability that the environment makes a transition from state s to state s' when action a is taken. $OS(o, s)$ is the probability of making an observation o in state s. The probability of being in state s' (belief state of s') after having made observation o while performing action a is then given by the following equation:

$$SE_{s'}(k+1) = \frac{OS(o, s') \sum_{s \in S} T(s, a, s') SE_s(k)}{P(o|a, SE(k))} \tag{5.1}$$

where $SE_s(k)$ is the belief state of s at the last step, $SE(k)$ is the belief state vector of last step and $P(o|a, SE(k))$ is a normalizing factor.

Set of States S

As already explained at the beginning of this chapter the goal here is to realize localization within a given map. Therefore, the set of states S is given by human measurement and is shown in the graph of Fig. 5.5.

Set of Actions A

The environment is simple enough to allow the use of few actions. These are *follow mid-line* and *wall following*. In the narrow horizontal hallway of Fig. 5.5 just *follow mid-line* is used while in the other hallway *wall following* is used in order to remain on the line of the graph in Fig. 5.5.

Observation Function OS

Single landmarks are integrated sequentially. The function $OS(o, s)$ just gives back the probability of being in state s when landmark o is extracted. This information is stored in a table which defines the probability of seeing the extracted landmark o when o is effectively present in s, but also the probability of seeing it when there is not such a landmark in s as explained in Table 5.2 via an example.

Table 5.2: The *OS* function uses a table of this kind. If in state s there is a door and a Z feature then when a door and an S feature are observed, *OS* is equal to 0.5 (door is door) . (0.2 + 0.1) (Z feature is nothing, or S feature) . (0.2 + 0.1) (nothing, or Z feature is an S feature) = 0.045.

Map \ Observation	Z feature	S feature	Door	Hallway	Nothing
Z feature	0.6	0.1	0.05	0.05	0.2
S feature	0.1	0.6	0.05	0.05	0.2
Door	0.05	0.05	0.5	0.2	0.2
Hallway	0.05	0.05	0.2	0.5	0.2
Nothing	0.2	0.2	0.2	0.2	0.2

Transition Model *T*

$T(s, a, s')$ permits the estimation of the probability of arriving in state *s'* after having performed action *a* from state *s* by taking into account the relative displacement given by odometry starting from the last state estimation. This can be performed because of the fact that locations (nodes) have a similar distance between them.

5.4.1.2 Heading Estimator

The heading of the robot is estimated by performing a weighted mean of each observed line which is either horizontal, or vertical with respect to the environment. The success of this method is due to the fact that in a rectilinear office building the vast majority of flat surfaces are aligned with the principal building directions. Lines are matched by means of the validation test used in Section 4.1.2:

$$(z^{[i]} - \hat{z}^{[j]})\, S_{ij}^{-1}\, (z^{[i]} - \hat{z}^{[j]})^T \leq \chi^2_{\alpha, n} \tag{5.2}$$

where, in this case, prediction $\hat{z}^{[j]}$ is directly the odometry state vector variable θ from the odometry model of Section 2.2 and $\chi^2_{\alpha, n}$ is a number taken from a χ^2 distribution with $n = 1$ degrees of freedom. This can be viewed as a Kalman filter for heading only.

5.4.2 Global Planning

Since it is computationally intractable to compute the optimal POMDP control strategy for a large environment [Cassandra96] simple sub-optimal heuristics are introduced. For the system presented here the *most likely state* policy has been adopted: The world state *s* with the highest probability is assumed to be correct and the action *a* which would be optimal for that state is executed. However, sometimes the estimation can be unconfident, meaning that the probability distribution has not a relevant peak. This is calculated by means of the confidence function $C(SE(k))$ which is a simple scoring function taking into account the most likely state, the cumulated probability of its neighborhood states and the ratio between the cumulated probability and the second most likely state. The tests for the scoring function have been tuned by experience and are:

$$max_s s > 0.5 \tag{5.3}$$

$$sum_s > 0.9 \tag{5.4}$$

$$\frac{sum_s}{max_{s \setminus max_s s}\, s} > 2 \tag{5.5}$$

where the sum of the neighbor states sum_s is defined by:

$$sum_s = \sum_{s' \equiv s \cdot next} s' \tag{5.6}$$

The confidence function is then defined as:

$$C(SE(k)) = \begin{cases} high & score \equiv 3 \\ confident & score \equiv 2 \\ uncofident & score \leq 1 \end{cases} \tag{5.7}$$

When unconfident states take place if the optimal action is the same for all the states with high probability the action will be executed directly, otherwise the system searches for the best acceptable action for all the high probability states which permits information gain for the state estimator (ex.: Reverse direction and follow mid-line).

5.4.3 Local Planning

As explained previously, in this environment topological navigation takes place in long hallways. Therefore, only a few actions are needed. *Follow mid-line* and *wall following* permit the robot to navigate in the global environment. All actions are implemented by using the motion control algorithm presented in Appendix A.3. This approach requires a metric goal which is calculated relative to the current position by taking into account the longest extracted line segments.

5.5 Switching Model

Switching model means not only that the robot changes its localization approach, but also its navigation technique. In general, the robot switches to metric when it expects to need some more precision. It switches back to topological as soon as it requires to make a long tour. As explained in Section 5.2.1, for the current implementation the robot switches to metric when it enters a room and back to topological when it leaves it.

5.5.1 Topological to Metric

Due to the fact that the topological navigation method is multimodal, the confidence before switching to the unimodal metric navigation is very critical. In contrast to pure topological navigation a false state when entering a local metric place would cause the robot to search a goal position in the false local metric map where the goal could even be inaccessible. Therefore, switching to metric is executed

only when the estimator confidence is high (see Equation 5.7) and the robot is in front of a door. If it is not the case the best action for gaining more information is taken.

When switching from topological to metric another important problem has to be faced: The Kalman filter has to be initialized (i.e. $\hat{x}(0|0)$ and $P(0|0)$ are unknown). Such a task is often referred to as the *relocation* problem. Although this is a complex problem it can be simplified for this approach. As explained in Section 5.2 a detectable metric feature (doors in this case) between a node and a place permits knowing when to switch and gives an approximation of the robot position with respect to the local metric map. The first two moments of the measurement are used to initialize the Kalman filter and permit a fast convergence of the filter.

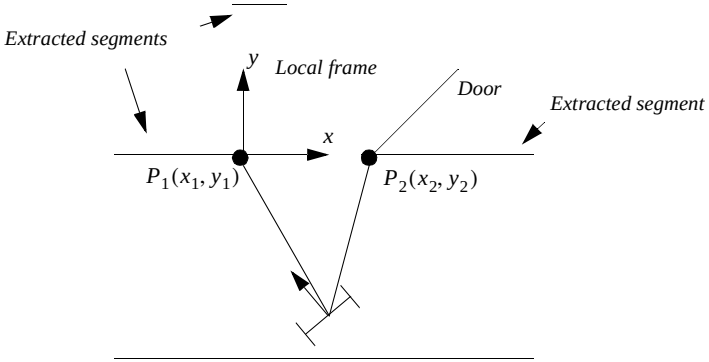

Figure 5.4: The robot is highly confident and has reached and detected a door. It measures P_1 and P_2 relative to its position. By knowing the position of these points in the current local metric map it can initialize the EKF.

5.5.2 Metric to Topological

Changing from metric to topological reduces to a metric navigation to the initialization position of the robot for the current local place. There the initialization of the states of the graph representing the global map takes place.

5.6 Experimental Results

For the experiments Donald Duck, the robot presented in Section 1.5.2, has been used. It runs in a fully autonomous mode and is connected via radio ethernet only for data visualization purposes via web and data logging for statistical purposes. The approach has been tested during the day, under normal conditions, in

the portion of the Institute building shown in Fig. 5.5. This environment is not only complex, but also highly dynamic due to the presence of the coffee room and the secretarial office which increases the presence of humans in the hallways.

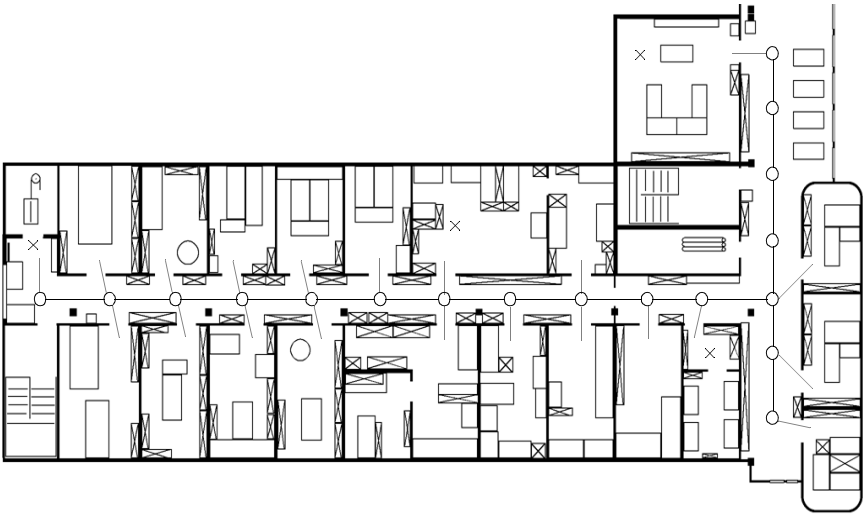

Figure 5.5: The map of the test environment with the graph representing the topological map. The crosses represent the start, or end of the randomly generated test missions.

5.6.1 Experiments

Test missions are generated randomly where the start and end point correspond to the crosses of Fig. 5.5. The robot is localized with respect to the local metric map at the start position. By leaving the room it switches to topological navigation. When it reaches the goal place it initializes the Kalman filter and navigates metrically to the goal point. There the position error is measured.

5.6.2 Results

Donald Duck performed 25 randomly generated missions and achieved an overall success rate of 96%. A mission is classified as successful when the robot reaches the goal point and is localized with respect to the local metric map in the goal place. The overall distance travelled is 1.15 km. Note that 0.95 km have been performed in topological mode which is multimodal and, therefore, robust against lost situations. The ‘mean error at goal point’ is the average of the differences between the robot estimate and its real position at the goal point. It is only 9 mm

demonstrating the accuracy of the Kalman approach. 23% of the estimates are unconfident. This seems relative high but the control strategy always solves them. Nevertheless, that remains sub-optimal because in four missions it causes robot navigation for information gaining only.

In one experiment the robot entered the wrong local place: Due to the noise in the robot perception (false extractions and occlusions) and the similarity between the goal location and its neighbor location, the robot thought it was at the goal location (where it had to switch to metric) when it was in front of the neighboring office door. Because the robot was confident about its state it switched to metric and entered the incorrect office. With this, the robot is unable to fulfill its mission because it uses the local metric map of the goal office for another room. With the current implementation, the only mean to recover from such a situation would be to detect the failure, exit the room and switch to topological. However, the detection of such a failure with the EKF remains unsolved. This case represents the main limitation of the presented approach which is, therefore, explicitly discussed in Section 5.7.1.

Table 5.3: Summary of the experimental results. The results demonstrate the feasibility of this hybrid approach for office environments.

number of missions	25
success rate	96%
number of state estimates	788
unconfident state rate	23%
total travel distance	1.15 km
topological travel distance	0.95 km
metric travel distance	0.2 km
mean error at goal point	9 mm

5.7 Discussion

For this system a natural integration of the metric and topological paradigms is proposed. The approaches are completely separated into two levels of abstraction. Metric maps are used only locally for structures (rooms) which are naturally defined by the environment. There, a fully metric localization and navigation method is adopted. Topological navigation and localization are used to connect the local metric maps which can be far away from each other. This intuitive way to gain in robustness while maintaining precision at the goal/task point is similar to the behavior humans have: When going to a room where a task has to be per-

formed the human localizes itself roughly with respect to doors, hallways, crosses and so on but in the goal room he measures exactly his position with respect to the coffee machine where he has to put his cup. Even the heuristics presented in Section 5.4.2 take inspiration from man's experience: If the person searching for the coffee machine is not sure about the door he has to enter, he will go back some meters, or continue in the hallway to gain information in order to be sure not to enter in the office of the big boss, which is the next one after the coffee room.

A further aspect of this approach is the emphasis in compactness of the environmental representation. This proves to be very efficient when calculating the robot position. Fig. 5.2 shows an office which is metrically modeled with only 14 lines while the whole environment ($50 \times 30 m^2$) is represented topologically by less than 20 nodes (Fig. 5.5).

5.7.1 Limitations

The main limitation is by switching from topological to metric where the loss of multimodality can cause inconsistent EKF initialization, as happened in one of the 25 test missions. However, this can be faced by extending the use of the topological navigation in the metric places too, in parallel to the EKF localization. For this an approach for the perception in rooms has to be developed. Furthermore, in general perception has to be improved especially for the topological localization where the discontinuties turned out to be sub-optimal when extracted with segments.This is not the goal of this thesis, however, possible solutions for a better perception could be [Lamon01] or [Ulrich00]. These approaches permit the characterization of the environment by means of a vision system which uses a conic mirror in order to have an omni-directional view of the surroundings. By taking into account local distinctive color and feature patterns and their relative angle position, they permit to distinguish between different locations. This allows the use of a topological approach also in less structured approaches and should even be introduced for outdoor localization.

5.8 Conclusion

This chapters has presented a new hybrid approach for localization. The metric and topological parts are completely separated into two levels of abstraction. Together they permit a very compact and computationally efficient representation of the environment for mobile robot navigation. Furthermore, this combination permits both precision with the non-discrete metric estimator and robustness by means of the multimodal topological approach. The success rate over the 1.15 km of the 25 tests missions is 96% meaning that only one mission was not fulfilled. The mean error at the goal point is only 9 mm. The 23% of unconfident states are

uncritical in the experiments. Nevertheless, they cause a loss of time when travelling for gaining further information.

In the next chapter this approach will be extended to permit simultaneous localization and map building.

6

Hybrid Localization and Map Building

"Make visible what, without you, might perhaps never have been seen."

Robert Bresson (born 1907)

Map building is the task of creating the knowledge concerning the environment which will then be used for localization. This knowledge is then stored in a map.

Mapping is very important in the real world because *a priori* maps are rarely available and, even when given, not in the format required by the robot. Furthermore, they are mainly unsatisfactory due to imprecision, incorrectness, incompleteness and dynamic changes. Therefore, map building is not only a desire which automatizes a work which would have to be performed by hand, but they are a real need for application-like scenarios.

This chapter shows how the environment model presented in Section 5.2 can be used for simultaneous localization and map building. The first part of this approach, with only a map and few localization tests, will be presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) in Maui, Hawaii, in October 2001 [Tomatis01d] while at the Fourth European Workshop on Advanced Mobile Robots (Eurobot) in Lund, Sweden the whole work with map comparisons and the extension which allows the closing of loops will be illustrated almost at the same period [Tomatis01e].

6.1 Introduction

Research in localization and automatic mapping has recently lead to successful approaches. However, solutions for consistent mapping allowing precise and robust localization in unmodified, dynamic, real-world environments have not yet been found. The problem is highly complex due to the fact that it requires the robot to remain localized with respect to the portion of the environment which has already been mapped in order to build a coherent map.

Current research has diverged to different approaches: Metric, topological, or hybrid navigation schemes have been proposed and studied. Approaches using purely metric maps are vulnerable to inaccuracies in both the map-making and odometry abilities of the robot. Even by taking into account all relationships between features and the robot itself in large environments the drift in the odometry makes the global consistency of the map difficult to maintain. Landmark-based approaches which rely on the topology of the environment, can better handle this problem because they only have to maintain topological global consistency and not metric consistency. However, these approaches are either less precise than fully metric approaches, due to the discretization of the localization space, or computationally intractable for fully autonomous robots when fine grained grids are used.

More recently, approaches combining the topological and the metric paradigms (mainly grid-based) have shown that the positive characteristics of both can be integrated to compensate for the weakness of each single approach. They are presented in the next section.

6.1.1 Related Work

After the first precise mathematical definition of the *stochastic map* [Smith88] early experiments [Crowley89], [Leonard92], have shown the quality of fully metric simultaneous localization and map building: The resulting environment model permits highly precise localization which is only bounded by the quality of the sensor data [Arras01a]. However, these approaches suffer from some limitations. Firstly they rely strongly on odometry. For automatic mapping this makes the global consistency of the map difficult to maintain in large environments where the drift in the odometry becomes too important. Furthermore, they represent the robot position with a single Gaussian distribution. This means that an unmodeled event (i.e. collision) could cause divergence between the ground truth and the estimated position from which the system is unable to recover (lost situation). In [Castellanos97] it has been shown that by taking into account all the correlations the global consistency is better maintained. However, this is not sufficient as confirmed by a recent work [Castellanos00] where a solution is pro-

posed by extending the absolute localization to include a localization relative to local reference frames.

Metric approaches using raw range data for *scan matching* are also unimodal [Lu94], [Gutmann96]. The robot position is also represented by a single Gaussian where a simplified Kalman filter is used to calculate the posterior state. Compared to feature-based approaches scan matching techniques usually operate with memory-intensive maps since the environment model consists of raw range data recorded from a set of reference positions.

On the other hand topological approaches [Kuipers87] can handle multi hypothesis tracking and have a topological global consistency which is easier to maintain. The robustness of such approaches has firstly been proven by the application of the *state set progression* [Nourbakhsh98] which has then been generalized to the POMDP approach [Cassandra96], [Gutierrez96]. For automatic mapping in [Koenig95] the *Baum-Welch algorithm* has been used for model learning. In contrast to the above mentioned topological approaches [Kunz99] proposes a topological approach which heavily relies on odometry in order to better handle dynamic environments. Other recent works try to resume the topology of the environment by means of Voronoi diagrams [VanZwyns01]. All these approaches are robust, but have the drawback of losing in precision with respect to the fully metric ones: The robot position is represented by a location without precise metric information. To overcome this, the *Markov localization* [Fox98] has been proposed: A fine grained grid guarantees both precision and multimodality. However, this approach remains computationally intractable for current embedded systems. A more efficient alternative has recently been proposed, but with the *Monte Carlo localization* [Dellaert99] the number of particles in the filter has to be carefully chosen (trade-off between computational intensive and multi hypotheses). Furthermore, it has not yet been extended for simultaneous localization and map building.

Metric and topological approaches are converging, such as [Castellanos00], [Dellaert99] and [Fox98], to hybrid solutions by adding advantageous characteristics of the opposing world. Moving in this direction, [Arleo99] proposes a hybrid approach which employs neural network to reduce the complexity, but relies on the assumption that obstacles are either parallel, or perpendicular to each other. In [Thrun96] the approach consists of extracting a topological map from a grid map by means of a Voronoi-based method while [Thrun98] proposes to use the *Baum-Welch algorithm*, as in [Koenig95], but to build a topologically consistent global map which also permits closing the loop for the global metric map. For this

approach they use an alternative name for the Baum-Welch algorithm: *Expectation Maximization* (EM).

Table 6.1: A brief resume of the best known approaches for map building.

	Stochastic Map	Scan Matching	Topological POMDP	Occupancy Grid	EM
Sensor information	features	raw data	landmarks	raw data	raw data
Amount of used data	=	+	-	+	+
Env. dynamic	+	-	+	-	-
Consistency	=	-	+	+	+
Precision	+	+	-	=	+
Robustness	-	=	+	+	=
Practicability	=	=	+	=	-

As can be seen in Table 6.1, as for localization (Section 5.1), there are no solutions for simultaneous localization and map building grouping all the requirements for a successful approach for application-like scenarios. Stochastic map allows the construction of a precise map, but guarantees a consistent result only when the odometry error to sensor information ratio is sufficiently low and under the assumption that the odometry model is always true (i.e. no unmodeled collisions). The main drawbacks of scan matching reside in its sensibility to the dynamics in the environment due to the use of raw data (no filtering through feature extraction), the reliance on the odometry and the vast amount of data which is required to represent the map. Topological approaches permit the easier construction of consistent maps because they do not require metric consistency. However, they do not allow precision. Occupancy grid approaches are one of the simplest approaches to map building. Nevertheless, their precision is bounded to the grid size while the grid size defines their practicability. The EM algorithm permit the rebuilding of a map which may converge to a consistent solution by starting up from scratch at each step. This means that the approach is simply unfeasible for on-line implementation.

In general, approaches relying on odometry (all but the topological one in Table 6.1) can work only if their perception and mapping approach can tackle the errors in odometry. This means that for small environments they are all suitable, but for large environments care has to be taken in order to converge to a consistent map.

This chapter proposes a new alternative to the above mentioned approaches. In this case too a natural integration of both the metric and topological paradigms is proposed. The effectiveness of such an approach for localization has already been

shown in the last chapter. In this chapter it is extended to include an automatic mapping approach which permits the handling of loops in the environment. Table 6.1 shows that the only metric approach capable of handling of dynamics in the environment while being precise end practicable, is the *stochastic map*. Furthermore, when used for small environments, this approach permits also the building of consistent maps with the highest precision which is only bounded by the quality of the measurements. For the metric approach the *stochastic map* [Smith88] is then proposed as a natural extension to simultaneous localization and map building when using the *Extended Kalman filter* (EKF). The *Partially Observable Markov Decision Process* (POMDP) state estimator presented in Section 5.4 is extended to allow environmental model learning (map building).

6.2 Environment Modeling

The main idea of the environmental representation is the same as in the last chapter. Here it is briefly summarized.

The environment is described by a global topological map which permits moving in the whole environment and local metric maps that the robot can use as soon as it needs further localization precision.

In order to switch from topological to metric a detectable metric feature is needed to determine the transition point and to initialize the metric localization (i.e. *relocation*). This is the only specific requirement for this approach. Given this transition feature a metric place can be defined everywhere in the environment.

Switching to topological does not require any specific characteristic: The robot navigates metrically to the initialization position for the current local place where it restarts its topological navigation.

6.2.1 Global Topological Map

Landmarks which are helpful for the topological model, are those which distinguish between locations in the environment. As stated in the conclusions of the hybrid localization approach (Section 5.8) the perception for the topological approach was sub-optimal. Here a better one is presented where the used landmarks are the following:

- Corners which are characterized by their orientation.
- Openings which are also used for model transition.

Corners have the advantage of being able to represent discontinuities as in Section 5.2.3, but also to permit the matching of just a part of a discontinuity if it is not completely detected. Furthermore, they permit a better description of the environment by modeling more physical objects.

The topological map can be viewed as a graph. Topological locations are represented by nodes containing the information concerning the way to reach the connected topological location/metric place. Furthermore, in contrast to the structure proposed in Section 5.2.3 the list of the landmarks lying between two locations is represented by a list between the two nodes. In Fig. 6.1 the graph representing the topological model is viewed for a portion of the environment.

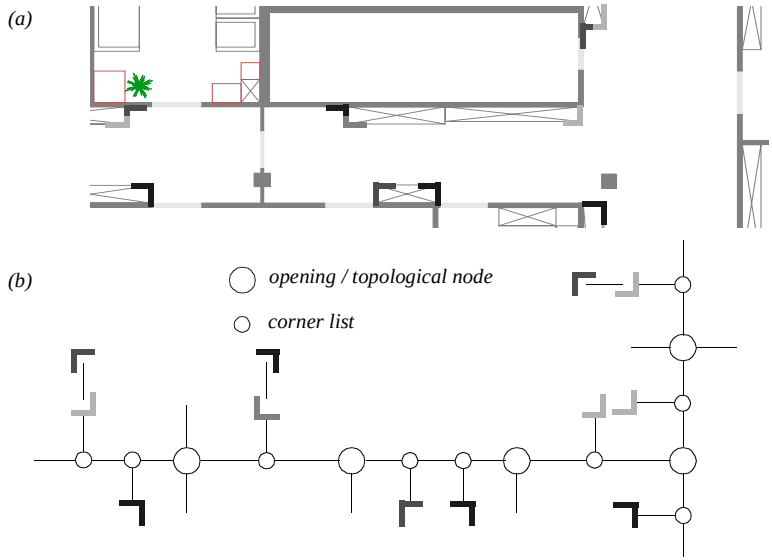

Figure 6.1: (a) A portion of a hallway with the extracted corner and opening features. (b) The topological map is represented by a graph. It contains nodes connected to each other with the list of corner features lying between them. Openings (topological nodes) can either be a transition to a room, or be a connection to another hallway.

Landmarks are extracted by the algorithms presented in Section 3.2. Note that, because the used sensor is a 360 degree laser scanner, an observation contains many landmarks which are transformed in a graph compatible to the environment model as shown in Fig. 6.2.

6.2.2 Local Metric Maps

The metric environmental representation is exactly the same as presented in Section 5.2.2.

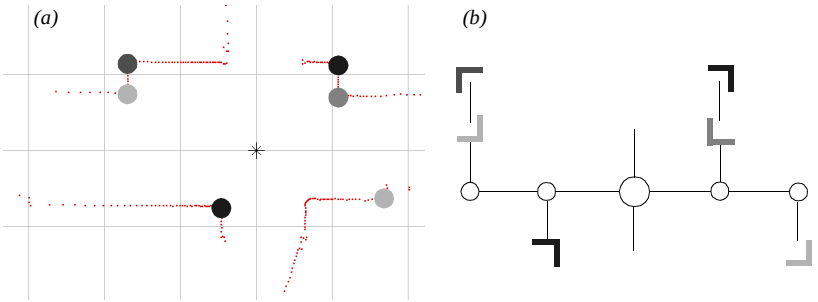

Figure 6.2: (a) Laser data and the extracted features. (b) The resulting observation graph.

6.3 Localization and Map Building

The environment models allow the use of two different navigation methods with complementary characteristics. The metric localization permits a very precise positioning at the goal point [Arras01a], [Tomatis01b] whereas the topological one [Cassandra96], [Tomatis01b] guarantees robustness against getting lost due to the multimodal representation of the robot's location.

6.3.1 Map Building Strategy

As explained in Section 6.2 the environment model is composed of a global topological map and a set of local metric maps. Given a metric transition feature, local metric maps can be everywhere in the environment. Therefore, a suitable strategy has to be adopted.

This strategy relies on some application and environment dependent assumptions. They are presented in the next section.

6.3.1.1 Implementation Related Assumptions

The current experimental test bed is still the same part of the Institute building. This environment is rectilinear and mainly composed of offices, meeting rooms and hallways. Therefore, only four travel directions are employed: N, E, S, W. However, this limitation is not an inherent loss of generality because it is not a general requirement of the algorithm.

For many possible application scenarios it can be expected that the robot will have to be very precise in rooms where most of its tasks have to be executed (e.g. docking for power recharging; manipulation tasks with objects on a table; human-robot interaction, ...). While navigating in the large (i.e. hallways) precision with

respect to the features is less important, but robustness and global consistency take an important role. As a result of this, the two different levels of abstraction are used in combination of the different type of environmental structures:

- While navigating in hallways the robot firstly creates and then updates the global topological map
- When it enters a room it creates a new local metric map

These two environmental structures are recognized by means of the laser sensor: Narrow and long open spaces are assumed to be hallways while other open spaces will be defined as rooms. For this, thresholds which are environment dependent have been found.

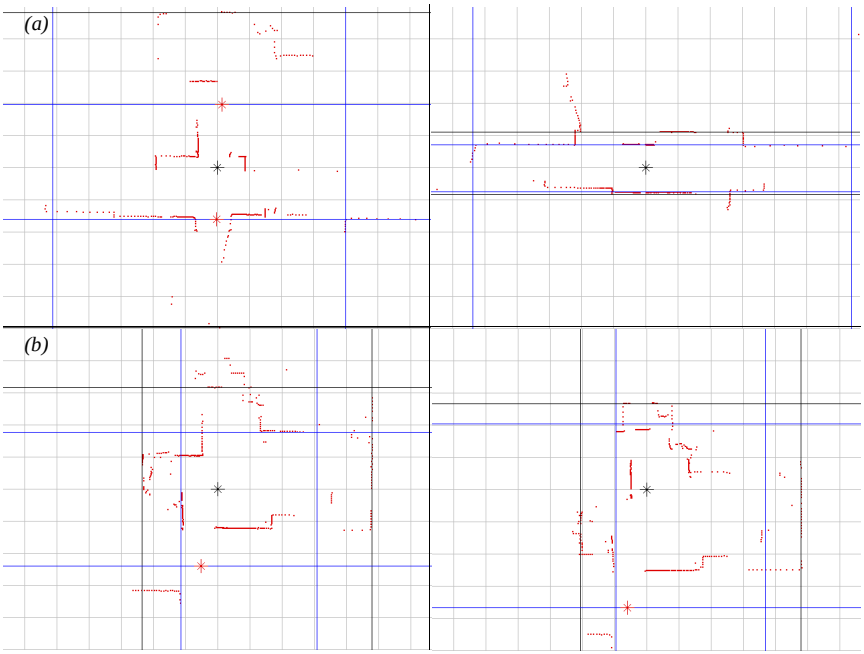

Figure 6.3: An arbitrary heuristic permit the distinction between (a) Hallways and (b) Rooms.

6.3.2 Exploration Strategy

The proposed exploration strategy is simple: The robot first explores all the hallways in a depth-first way. It then explores each room it encountered by backtracking. Note that, in general, for each hallway the room exploration reduces to a linear list traversal. Rooms with multiple openings cause two special cases which are treated in the next paragraphs.

6.3.2.1 Rooms with an opening to another room

In this case the robot continues building the current metric map. However, this can lead to the next case if the neighbor room has an opening to a hallway.

6.3.2.2 Rooms with an opening to a hallway

Due to the metric navigation mode during room exploration the robot knows the direction of the opening and can, therefore, deduce if it opens to the same hallway, a known one, or a new one. In the case of known hallways the robot simply goes back to the hallway it was coming from and continues its exploration. This could cause having two metric maps for the same metric place, one for each opening. In the case of a new hallway the exploration continues in a hallway using depth-first search.

6.3.3 Topological Localization and Map Building

In this section the *state estimator* is presented in detail in order to focus on the differences with respect to the one presented in the last chapter (Section 5.4.1.1). The *heading estimator* is the same as before and is, therefore, just briefly summarized. The approach to *map building* is then presented with a further section focusing on the problem of how to handle *loops* in the environment.

6.3.3.1 State Estimator

Given a finite set of environment states S, a finite set of actions A and a state transition model $T(s, a, s')$ the model can be defined by introducing partial observability. This includes a finite set O of possible observations and an observation function $OS(o, s, a)$, not $OS(o, s)$ as in the last chapter. The probability of being in state s' (*belief state* of s') after having made observation o while performing action a is then given by the equation:

$$SE_{s'}(k+1) = \frac{OS(o, s', a) \sum_{s \in S} T(s, a, s') SE_s(k)}{P(o|a, SE(k))} \tag{6.1}$$

where $SE_s(k)$ is the belief state of s for the last step, $SE(k)$ is the belief state vector of last step and $P(o|a, SE(k))$ is a normalizing factor.

Set of States S

As a consequence of the map building strategy which suggests the creation of local metric maps in rooms the transition between hallways and rooms takes a primary role in the model. Due to this fact nodes (states) are places at each position in the hallways where openings to a room are detected. They are shown with black dots in Fig. 6.6 for the current test environment.

Set of Actions *A*

The environment is the same as in Chapter 5. However, in this case only a single action, *follow mid-line*, is used for each direction. This can be done because, as stated in Section 6.3.1, as soon as the robot enters a large open space it switches to metric.

Observation Function *OS*

This is one of the main differences with respect to the last chapter. Here landmarks are not integrated sequentially by the state estimator. The observation function *OS* is made robust by the fact that an observation is composed of many landmarks (Fig. 6.2) raising its distinctiveness. This characteristic is used to permit a sort of graph matching, as shown in Fig. 6.4.

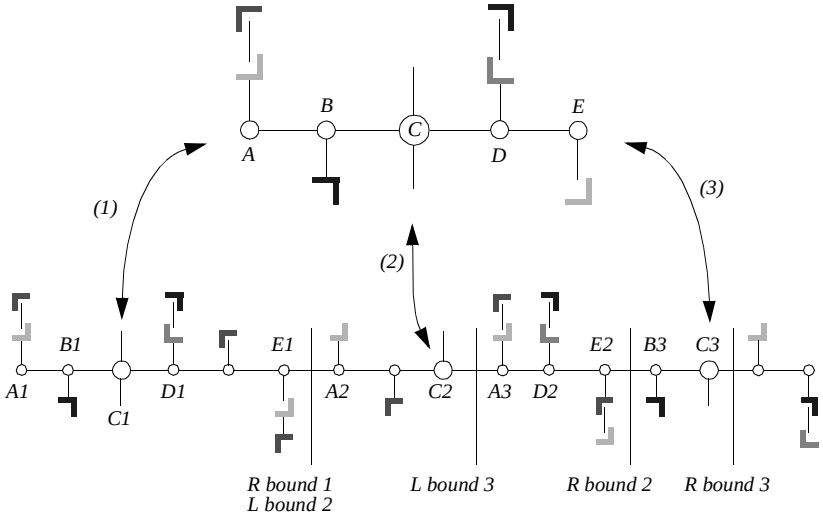

Figure 6.4: The observation function *OS* is a sort of graph matching.

For each state *s* the probability $OS(o, s, a)$ is then calculated by:

$$OS(o,s,a) = \prod_{l \in o \wedge l \in s} p_{lmap} \cdot \prod_{l \in o \wedge l \notin s} 1 - p_l \cdot \prod_{l \notin o \wedge l \in s} 1 - p_{lmap} \quad (6.2)$$

where p_l is the landmark extraction confidence (Section 3.2.1 and Section 6.3.3.4) and p_{lmap} the confidence of a landmark in the map (Section 6.3.3.4). As it is not guaranteed that at each moment the robot sees all the landmarks of state *s* only the map landmarks which can be matched with the observation graph are taken into account (see bounds in Fig. 6.4).

The action a in the *OS* function enables one to know how to align the observation graph to the map graph. This is the main advantage by using just four directions of travel (Section 6.3.1.1). If the environment would not have been rectilinear all directions would have to be allowed and all possible alignments would have to be taken into account for matching.

Transition Model T

$T(s, a, s')$ permits the estimation of the probability of arriving in state s', after having performed action a from state s by taking into account the relative displacement given by odometry starting from the last state estimation. When no openings are visible $T(s, a, s) = 0.99$ while $T(s, a, s') = 0.01$ for $s \neq s'$. When the robot encounters an opening the most probable state s' is searched by comparing the travelled distance d, measured starting from s, to the information saved in state node s during map building. In this case $T(s, a, s') = 0.99$ while $T(s, a, s'') = 0.01$ for $s'' \neq s'$.

6.3.3.2 Heading Estimator

The heading estimator is the same as in Section 5.4.1.2. Each observed line which is either horizontal, or vertical with respect to the environment is used to calculate the heading. These lines are detected by means of the validation test:

$$(z^{[i]} - \hat{z}^{[j]})\, S_{ij}^{-1}\, (z^{[i]} - \hat{z}^{[j]})^T \leq \chi^2_{\alpha, n} \tag{6.3}$$

where prediction $\hat{z}^{[j]}$ is directly the odometry state vector variable θ and $\chi^2_{\alpha, n}$ is a number taken from a χ^2 distribution with $n = 1$.

6.3.3.3 Control Strategy

Since it is computationally intractable to compute the optimal POMDP control strategy for a large environment [Cassandra96] a simple sub-optimal heuristic is used. As in the last chapter the *most likely state* policy has been adopted: The world state with the highest probability is found and the action which would be optimal for that state is executed. However, here a better suited confidence value is used. It is calculated by mean of the unconfident function $U(SE(k))$ which is the entropy of the probability distribution over the states of the map. The POMDP is confident when:

$$U(SE(k)) = -\sum_{s} SE_s(k) \log SE_s(k) < U_{max} \tag{6.4}$$

where U_{max} is determined by experience. When the robot is unconfident, it follows the hallway in the direction where it expects to find more information. As explicitly stated in Section 5.5.1 what has to be avoided at any cost is to switch

from the multimodal topological navigation to the unimodal metric navigation when the robot is unconfident about its location. Otherwise it could enter a false local metric place and, therefore, be lost. If such a problem occurs a solution for detecting this situation and exiting the current local place would be required in order to allow the robot to relocate itself by means of the topological approach.

6.3.3.4 Map Building

Instead of using a complex scheme for model learning as in [Koenig95] and [Thrun98] where an extension of the *Baum-Welch* algorithm is adopted, here the characteristics of the observation graph (Fig. 6.2) are used. When the robot feels confident concerning its position (Equation 6.4) it can decide if an extracted landmark is new by comparing the observation graph to the node in the map corresponding to the most likely state. This can happen either in an unexplored portion of the environment, or in a know portion where new landmarks appear due to the environmental dynamics. As explained in Section 3.2.1 the landmarks come with their extraction confidence p_l. This characteristic is used to decide if the new landmark can be integrated in the map. When an opening landmark is extracted it is integrated in the map as a new state node (Fig. 6.1) with a rough measure of the distance to the last state node. Furthermore, for each integrated landmark the confidence p_l is used to model the probability of seeing that landmark the next time p_{lmap}. When it is re-observed the probability in the map is averaged with the confidence of the extracted one. If the robot does not see an expected landmark the probability $1 - p_{lmap}$ is used instead:

$$p_{lmap}(t_i) = \sum_{i=1}^{n} \frac{p_l(t_i)}{n} \tag{6.5}$$

$$\text{where, } p_l(t_i) = \begin{cases} p_l(t_i), & observed \\ 1 - p_{lmap}(t_{i-1}), & \neg observed \end{cases} \tag{6.6}$$

When the confidence p_{lmap} decreases and is below a minimum the corresponding landmark is deleted from the map. This allows for dynamics in the environment where landmarks which disappear in the real world will be deleted from the map too.

6.3.3.5 Closing the Loop

The problem of *closing the loop* can be defined as the question of how to know when a location has already been explored, meaning that the environment contains a loop and that the loop in the map has to be also closed (Fig. 6.5).

In [Thrun98] this is achieved by means of the EM algorithm which ensures global consistency. This information is then used to correct the global metric map which eventually converges to a global consistent map.

The current approach differs in two main aspects:

- Instead of closing the loops only by means of the perception loops are detected and closed by means of the *localization information*.
- Loops have to be closed only in the topological map because the metric model is represented by many disconnect local metric maps.

Loops can also exist in a local metric map. However, due to the fact that these maps are supposed to be small, the drift in odometry does not cause any relevant problem to the local consistency.

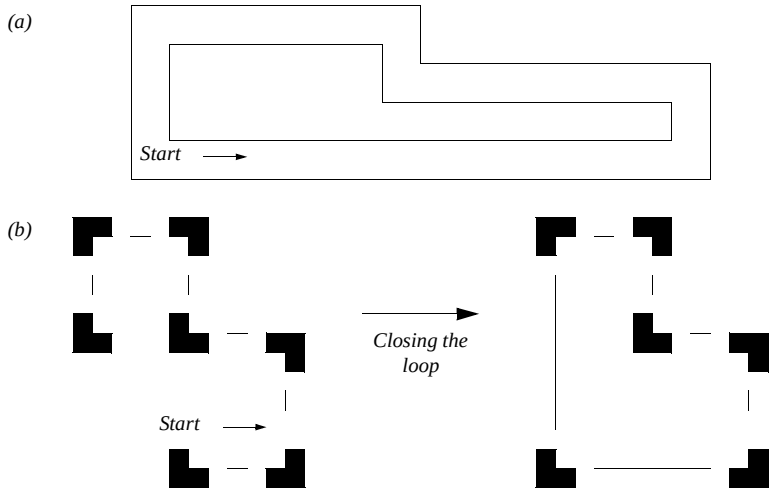

Figure 6.5: (a) A loop in the environment. (b) Its map. Closing the loop in automatic mapping requires the system to know that a location has already been visited.

The current method works as follows: The robot does not try to recognize if a single perception of the environment has already been seen somewhere else. However, as soon as the robot creates the map for a part of the environment which has already been visited the probability distribution starts diverging to two peaks: One for the position in the map which is currently being created; Another for the previously created location representing the same physical place. The algorithm starts tracking the two highest probabilities as soon as the POMPD becomes unconfident because this is the first clue indicating a divergence of the probability distribution. A loop can then easily be detected when the distribution has converged

into two peaks which move in the same way. The position where the loop has to be closed can then be detected by turning off the automatic mapper and backtracking with localization until the distribution re-converges to a single peak. This should also be the start point of the exploration (or a local start point).

6.3.4 Metric Localization and Map Building

This section briefly describes the main characteristics of the *stochastic map* approach [Smith88] which permits using an *Extended Kalman Filter* [Crowley89], [Leonard92] for localization.

With this approach both the robot position $x_r = (x, y, \theta)'$ and the features $x_i = (\alpha, r)'$ are represented in the system state vector:

$$x = \begin{bmatrix} x_r \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \quad C(x) = \begin{bmatrix} C_{rr} & C_{r1} & \cdots & C_{rn} \\ C_{1r} & C_{11} & \cdots & C_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ C_{nr} & C_{n1} & \cdots & C_{nn} \end{bmatrix} \tag{6.7}$$

This represents the uncertain spatial relationship between objects in the map which is changed by three actions:

- Robot displacement
- Observation of a new object
- Re-observation of an object already existing in the map

6.3.4.1 Robot Displacement

When the robot moves with an uncertain displacement $\boldsymbol{u}$ given by its two first moments (u, C_u) which are measured by the odometry, the robot state is updated to $g(x_r, \boldsymbol{u})$. The updated position and uncertainty of the robot position are obtained by error propagation on g:

$$x_r(k+1) = g(x_r(k), u) = x_r(k) \oplus u \tag{6.8}$$

$$C_{rr}(k+1) = G \begin{bmatrix} C_{rr}(k) & C_{ru}(k) \\ C_{ur}(k) & C_u \end{bmatrix} G^T \tag{6.9}$$

where G is the Jacobian of g with respect to x_r and $\boldsymbol{u}$ and $\oplus$ is the compounding operator which is defined as follows: Given two spatial relationships x_{ij} and x_{jk}, the resultant relationship x_{ik} is given by Equation 6.10.

$$\overrightarrow{x_{ik}} = \overrightarrow{x_{ij}} \oplus \overrightarrow{x_{jk}} = \begin{bmatrix} x_{jk}\cos\theta_{ij} - y_{jk}\sin\theta_{ij} + x_{ij} \\ x_{jk}\sin\theta_{ij} + y_{jk}\cos\theta_{ij} + y_{ij} \\ \theta_{ij} + \theta_{jk} \end{bmatrix} \tag{6.10}$$

G is the Jacobian of g with respect to x_r and $\boldsymbol{u}$.

6.3.4.2 New Object

When a new object is found a new entry must be made in the system state vector. A new row and column are also added to the system covariance matrix to describe the uncertainty in the object's location and the inter-dependencies with the other objects. The new object (x_{new}, C_{new}) can be integrated in the map by computing the following equations of uncertainty propagation:

$$x_{N+1}(k) = g(x_r(k), x_{new}) = x_r(k) \oplus x_{new} \tag{6.11}$$

$$C_{N+1N+1}(k) = G_{x_r} C_{rr}(k) G_{x_r}^T + G_{x_{new}} C_{new} G_{x_{new}}^T \tag{6.12}$$

$$C_{N+1i}(k) = G_{x_r} C_{ri}(k) \tag{6.13}$$

6.3.4.3 Re-Observation

Let x_{new} be the new observation in the robot frame. The measurement equation is defined as:

$$z = h(x_r, x_{new}, x_i) = g(x_r, x_{new}) - x_i \tag{6.14}$$

x_{new} is temporarily included in the state to apply the EKF. However, if prediction x_i satisfies the validation test:

$$(x_{new} - x_i) S_{newi}^{-1} (x_{new} - x_i)^T \leq \chi_{\alpha,n}^2 \tag{6.15}$$

where $S_{newi} = C_{newnew} + C_{ii} - C_{newi} - C_{inew}$, $\chi_{\alpha,n}^2$ is a number taken from a χ^2 distribution with $n = 2$ degrees of freedom and α the level on which the hypothesis of pairing correctness is rejected then x_{new} is a re-observation of x_i.

6.3.4.4 Extended Kalman Filter

When a spatial relationship is re-observed the updated estimate is a weighted average of the two estimates calculated by means of an EKF. It permits the updating of a sub-set of the state vector while maintaining the consistency by means of the covariance matrices. A measurement equation $z = h(x_1, \ldots, x_m)$ is considered as a function of m relationships included in x. All of the n estimates x_i of the

state vector x are updated by a value which is proportional to the difference $\delta = z - \hat{z}$ between the ideal measurement z and the actual measurement $\hat{z}$:

$$x_i(k+1) = x_i(k) + \Gamma_{iz}\Gamma_{zz}^{-1}\delta \tag{6.16}$$

$$\Gamma_{iz} = E[x_i\delta^T] \; \Gamma_{iz} = \sum_{j=1}^{M} C_{ij}H_{xj}^T \tag{6.17}$$

$$\Gamma_{zz} = E[\delta\delta^T] \; \Gamma_{zz} = \sum_{j=1}^{M}\sum_{k=1}^{M} H_{x_j}C_{jk}H_{x_k}^T \tag{6.18}$$

where H_{x_j} is the Jacobian matrix of h with respect to x_j.

The variance and covariance C_{ij} are also updated:

$$C_{ij}(k+1) = C_{ij}(k) - \Gamma_{iz}\Gamma_{zz}^{-1}\Gamma_{jz}^T \tag{6.19}$$

6.4 Experimental Results

The approach has been tested in the $50 \times 30 m^2$ portion of the Institute building shown in Fig. 6.6 with four different types of experiments for a total of more than 1.5 km. For the experiments Donald Duck has been used.

6.4.1 Map Building

In this section the automatic mapping capabilities of this approach are tested and evaluated. Note that the environment is arbitrarily closed (Fig. 6.6) so that the exploration procedure is finite. Furthermore, local metric maps are taken from the *a priori* map used in [Arras01a] because the current implementation of the *stochastic map* runs only off-line.

For this evaluation five maps generated by complete explorations of the environment shown in Fig. 6.6 are compared to evaluate their quality with respect to consistency and completeness. In order to evaluate the topological mapper first maps are compared before the backtracking step. By knowing which door is open during the exploration it can be extrapolated how many state nodes should be extracted (see the black dots in Fig. 6.6). Their position (odometry) and type (opening, or hallway) are stored during exploration to check whether the resulting model is consistent with the real environment. For the other features (corners) each resulting map is compared to the others to calculate the average amount of differences between a couple of maps. The results are presented in Table 6.2.

One of the problems encountered during the exploration is the difficulty of distinguishing between opening and hallway. This leads to a mean of 1.2 false detection for each experiment. In one experiment a state (opening) was not extracted at

all. Nevertheless, by visiting all the openings when traversing the environment by backtracking to add the local metric maps these errors are detected and corrected.

Table 6.2: Comparison of five maps generated by complete explorations of the environment shown in Fig. 6.6.

Number of explorations	5
Total travelled distance	343 m
Number of states in the environment	13
Mean detected states	12.8 / 98%
Mean confused hallway/opening	1.2 / 9.2%
Mean detected features	78
Mean different features	18 / 23%

For the corner features it is more difficult to define which feature really exists in the environment. What can better be seen is the difference between two maps. The mean amount of extracted corners for each exploration is 78; an average of 18 of these are noisy, or dynamic features which are not always extracted. This means that almost 77% of the features are constant in the five maps showing that the perception delivers valuable information to the mapper.

6.4.2 Localization

The quality of a map can also easily be estimated by testing it for localization. One of the maps created in the last section has been used here for localization.

To test the topological localization 25 randomly generated test missions for a total of approximately 900 m and 28000 estimates are performed. The robot knows in which state it is in the start point. A mission is successful when the robot reaches its goal location, is in front of the opening and is confident about its position. There it switches to the metric approach. To have more information concerning the experiments each state transition is stored in a log file with all the information permitting one to know if each state transition detected by the localization took place physically. The results are presented in Table 6.3. Even if all the missions are successful the log file permits the detection of 21 false state transitions which caused 404 false estimates in *B* and *B'* (Fig. 6.6) where the peak probability moved forward and backward between two neighbor states. These false estimates represent only 1.4% of the total meaning that the system recovers quite quickly from these errors. Nevertheless, the robot also had confident false estimates (0.5%) which can cause a mission failure if the goal state is estimated when the robot is in front of a another opening.

Table 6.3: Localization experiments. All the test missions have been successfully performed. However, the robot also made false state transitions which caused some false estimates (1.4%). This happened only by B and B' in Fig. 6.6. The reason that lead to a success rate of 100% is that the system always recovered from its error without estimating the goal location in front of a false opening. Nevertheless, the robot had also confident false estimates (0.5%) which could cause mission failure.

Number of missions	25
Success	25 / 100%
Total travelled distance	899 m
Mean travel distance	36 m
Mean travel speed	0.31 m/s
Total real state transitions	181
False state transitions	21 / 12%
Total estimates	27870
Unconfident states	3413 / 12%
False estimates	404 / 1.4%
Confident false estimates	149 / 0.5%

6.4.3 The Bootstrapping Problem

In the literature the problem of turning up a robot and letting it localize itself is defined as the *bootstrapping* (or *first-location*) problem. Another similar problem is the *kidnapping* (or *relocation*) problem which is the action of recovering from a lost situation (i.e. a situation where the robot is not in the position where it thinks it is). Actually, these two problems are very similar: If a robot is able to solve the bootstrapping problem it is also able to relocate itself in general.

With the same map as for the localization experiments a total of 10 bootstrapping experiments are started from a randomly defined position in the environment with an overall constant belief state (i.e. bootstrap/lost situation). The goal is to measure which distance, or amount of state transitions are required in order to converge to a correct confident state estimate. To avoid false interpretations the robot is required to travel 3 state nodes further without estimate errors to fulfill the test. In Table 6.4 the 10 tests are resumed briefly.

Table 6.4: The bootstrapping problem (i.e. overall constant belief state). The robot requires from 1 to 4 states to recover depending on the distinctiveness of the part of the environment where it is moving.

Number of experiments	10
Total travelled distance	250 m
Mean distance for recovering	13.7 m
Min / max distance for recovering	1.21 / 20.31 m
Mean number of state for recovering	2.11
Min / max state for recovering	1 / 4

As expected the robot can always recover. Its policy is simple: Go forward until recovery, or end of hallway; If end of hallway, turn. The system requires a minimum of 1 and a maximum of 4 states to recover. The interesting point is that this difference in the results is position dependent and repeatable. For example, the crossing between the two hallways permits the recovery with a single state because it is globally distinctive for the environment in Fig. 6.6. On the other hand, the right part of the horizontal hallway seems to be more distinctive than the left one where the robot requires the maximum amount of states to recover.

6.4.4 Closing the Loop

In the test environment there are no large loops. In order to test the proposed approach a loop is artificially created by displacing the robot during the exploration as shown in Fig. 6.6.

As explained in Section 6.3.3.5, it can be assumed that when two peaks appear and move in the same way for three subsequent state transitions a loop has been discovered. In all the other experiments this has effectively never appeared, showing that this a good test for loops. This experiment has been performed three times. Each time the probability distribution has effectively diverged into two peaks allowing the detection of the loop. In order to close the loop the robot has turned off the mapping algorithm and has gone back until the distribution has converged to a single confident peak. This took place where the map has been started (*1* in Fig. 6.6) proving that the loop could be closed correctly.

6.5 Discussion

In contrast to the hybrid methods in the literature for this system a natural integration of the metric and topological paradigm is proposed. The approaches are

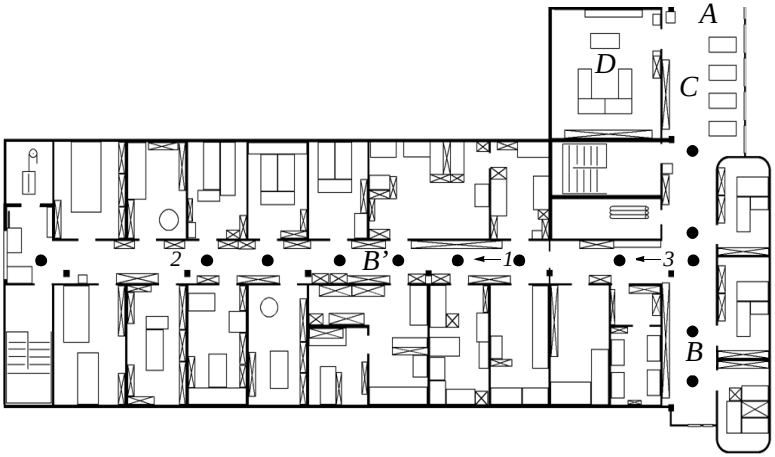

Figure 6.6: The test environment. It is complex, dynamic and artificially closed in *A* so that the exploration procedure is finite. Black dots are the places where the automatic mapper is expected to extract state nodes (the other doors are closed). In *B* and *B'* the robot had problems distinguishing between the two neighbor locations. *C* and *D* are detected as rooms and represented by a single local metric map. A large loop does not exist in this environment. Therefore, for the experiments in Section 6.4.4 a loop is 'artificially created' by starting the exploration in 1, stopping it in 2, taking the robot manually to 3 and resuming.

completely separated into two levels of abstraction. Metric maps are used only locally for structures (rooms) which are naturally defined by the environment. There, a fully metric method is adopted.

As has been shown in [Castellanos97] for such small environments where the drift in the odometry remains non-critical, *stochastic map* allows for precise and consistent automatic mapping. The metric localization is used but not explicitly tested here because the used EKF has already been extensively tested with a total of 6.4 km as shown in Section 4.1. Furthermore the metric localization approach has also been tested with this hybrid method for localization on the same robot in the last chapter where ground truth measurements at goal position resulted in an average error of less than 1 cm.

The topological approach is used to connect the local metric maps which can be far away from each other. With this the robot can take advantage of the precision

of a fully metric, EKF navigation, added to the robustness in the large of the POMDP approach. All this by maintaining a compactness of the environment representation and a low complexity which allows an efficient implementation of the method on a fully autonomous system. This hybrid approach also shows its practicability for environments with loops. In this case the loop is closed in the global topological map based on the information from the topological localization while the metric information remains local and does not, therefore, require further processing. This is in contrast to [Thrun98] where off-line calculation is required to correctly close the loop in the metric map.

6.5.1 Limitations

The limitation presented in Section 5.7.1 which arises when the robot changes its navigation approach from topological to metric, is still present here. The loss of multimodality could cause inconsistent EKF initialization. However, the better suited perception used in this chapter allows the avoidance of such a problem in the experiments: The noise in the feature extraction is not enough to cause any problem to the state estimator and its confidence function. Nevertheless, with the current setup an error can not be excluded. The solution of extending the topological localization to work permanently (meaning that it works also in parallel to the metric one in local places) which has already been proposed in Section 5.7.1, has, therefore, to be implemented in order to guarantee full consistency in the localization.

Another limitation is in the proposed approach for closing loops: If the environment is highly symmetric in the large and therefore, contains multiple large parts of the environment which looks similar, loops could be erroneously closed. This problem could be faced by taking into account rough relative metric information (distance and uncertainty) permitting the rejection of metric inconsistent loops candidates.

6.6 Conclusion

This chapter has shown how to extend the method presented in Chapter 5 to a hybrid approach for both localization and map building. The metric and topological parts are, as before, completely separated into two levels of abstraction. Together they allow a very compact and computationally efficient representation of the environment for mobile robot navigation. Furthermore, this combination permits both precision with the non-discrete metric estimator and robustness by means of the multimodal topological approach.

The approach is validated empirically by extensive experimentation for a total of more than 1.5 km. Map building is tested by performing five complete explo-

rations of the environment and comparing the resulting maps. This comparison demonstrates that the maps are consistent with respect to the environment and that the perception permits the extraction of precious information. For localization the success rate over the 0.9 km of the 25 tests missions is 100%. Nevertheless, a precise analysis of the state transitions shows that, between neighbor states, false state estimate occurs (1.4%) and sometimes are even treated as confident (0.5%). The relocation performance of the topological method has been shown with 10 successful experiments where the belief state converges with 1 to 4 states depending on the distinctiveness of the part of the environment where the robot is navigating. It has been shown how loops can be closed on the localization level instead of the perception level. This is easily performed by using the multi hypothesis tracking characteristic of the POMDP approach for detection and backtracking for closing the loop.

7

Conclusions

"What is the good of drawing conclusions from experience? I don't deny we sometimes draw the right conclusions, but don't we just as often draw the wrong ones?"

G. C. Lichtenberg (1742–99)

This thesis has presented recent research on the problem of *environmental modeling* for *localization* and *map building* for wheel-based, differential driven, fully autonomous and self-contained mobile robots which navigate in an indoor office environment. The robots have a multi-sensor setup where the encoders are used for odometry and two exteroperceptive sensors, a 360° laser scanner and a monocular vision system, are employed to perceive the surroundings.

The whole approach is feature-based. This allows the filtering of noise from the sensors and permits dealing with dynamics in the environment. Furthermore, a properly chosen feature extraction permits the better isolation of informative patterns. When describing these features care has to be taken that uncertainty from the measurements is taken into account because, as Albert Einstein told us: *"As far as the laws of mathematics refer to reality, they are not certain, and as far as they are certain, they do not refer to reality."*

The representation of the environment is crucial for mobile robot navigation. The model defines which perception capabilities are required and also which navigation technique is allowed to be used. The presented environmental model is both metric and topological. By coherently combining the two paradigms the advantages of both methods are added in order to face the drawbacks of a single approach. The capabilities of the hybrid approach have been exploited to model an indoor office environment where metric information is used locally in struc-

tures (rooms, offices) which are naturally defined by the environment itself while the topology of the whole environment is resumed separately thus avoiding the need of global metric consistency.

The hybrid model permits the use of two different and complementary approaches for localization, map building and planning. This combination permits the grouping of all the characteristics which enable the presented goals to be met: *Precision*, *robustness* and *practicability*. Metric approaches are, per definition, precise. The use of an *Extended Kalman Filter* (EKF) permits precision which is just bounded by the quality of the sensor data. Topological approaches can easily handle large environments because they do not heavily rely on dead reckoning. Global consistency can, therefore, be maintained in this kind of environment. Consistent mapping which handle large environments has been achieved by choosing a topological localization approach allowing *multi hypotheses tracking* which has been extended to simultaneous localization and map building.

All that has been mentioned above works well in theory and can be mathematically proven by making some assumptions. However, as stated during the whole work, at the end the robot itself has to tell us if the theory works well. For this, experiments for a total of more than 9 km have been performed with fully autonomous self-contained robots. These experiments have then been carefully analyzed. With the metric approach, precision with mean errors of approximately 1 cm and less than 1 degree has been estimated by measuring the error bounds and then confirmed by ground truth measurements with a mean error of less than 1 cm. The topological approach has also been successfully tested by simultaneous localization and map building where the automatically created maps turned out to work even better than the *a priori* maps. Relocation and closing the loop have also been successfully tested.

7.1 Contribution

The main contribution of this thesis has been to develop a new environmental representation and robot navigation approach while trying to fill the gap between pure academic theory and industry based productivity. Due to the fact that real world maps are mainly unsatisfactory due to imprecision, incorrectness, incompleteness and dynamic changes, localization and map building have been looked as a single complex problem. Mobile robot navigation, and especially simultaneous localization and map building, has been faced from a theoretical point of view, with an evaluation of the techniques available in the scientific world. This evaluation has lead to a hybrid solution which is quite easy to explain and prove from a theoretical point of view because it relies on two well-known approaches. The scientific challenge has been the coherent combination of the two theories

with their own advantages. The engineer task has been to put the resulting theory into a real robot and let it show that it really works.

This work represents the first solution which combine topological and metric by means of a coherent environment model by using two separate and complementary approaches.

The main contribution in this sense is to have taken into account the following statements:

- *Real Robots*: All the experiments are performed with *real robots*. All the used data come from *sensors on the robots*. All the calculations are conducted *on-board*, *on-line* and *on-the-fly* (during robot motion).
- *Real World*: The robots behave in real environments. These environments are indoor and mainly office-like. Such an environment is *dynamic*. The static assumption is unacceptable. As soon as the robot moves, and especially in dynamic environments, *collisions* cannot always be avoided. Approaches which rely on the fact that collisions never take place are unacceptable.

Given these preconditions the achieved goals are:

- *Precision*: In order to perform a specific task (e.g. docking for power recharging; manipulation tasks with objects on a table; human-robot interaction, ...) the robot has to precisely know its position.
- *Robustness*: Localization has to work without human intervention. This means that a mobile robot must be able to handle the above mentioned characteristics of the real world (dynamics with feature extraction, collisions with multi hypotheses tracking, ...). The same point has to hold for automatic mapping where consistent maps are expected to be constructed even if the environment is large, dynamic and contains loops.
- *Practicability*: All the above mentioned goals have to be achieved by means of the computational and sensorial resources of the embedded system itself. Using off-board structures is interesting for research but not acceptable for most application scenarios.

7.2 Limitations

There are two main limitations. They are actually not limitations of the proposed approach but they are instead implementation related.

- The main limitation is by switching from topological to metric where the loss of multimodality can cause inconsistent EKF initialization. However, this can be faced by extending the use of the topological navigation in the metric places too, in parallel to the EKF localization. The only requirement for this is to develop an approach for the perception in rooms.
- Another limitation is in the proposed approach for closing loops: If the environment is highly symmetric in the large and contains, therefore, multiple large parts of the environment which look very similar to the used perception loops could be erroneously closed. This problem could be faced by taking into account rough relative metric information (distance and uncertainty) permitting the rejection of metric inconsistent loop candidates.

7.3 Conclusion

This work has presented a possible solution for mobile robot navigation in indoor (well-structured) environments. This has been performed by taking into account the real characteristics of both the robot and its surroundings.

The presented approach is not the definitive solution to all the problems of mobile robotics. By doing research in this domain we try to let a machine understand the nature but as Antoine de Saint-Exupéry (1900-1944) told us in another context: "*The machine does not isolate man from the great problems of nature but plunges him more deeply into them.*"

Nevertheless, this work shows that by facing the problems with a certain methodology simple and efficient solutions can be found. Analyzing what already exists permits to "*plunge more deeply into*" the nature of the problems (or the problems of nature). Finding a new solution which faces some unresolved problems, permit the continuation in the same direction to see some further problems to solve.

A

The Robots

"As machines become more and more efficient and perfect, so it will become clear that imperfection is the greatness of man."

Ernst Fischer (1899–1972)

A.1 The Operating System

This is essentially a short summary regarding XO/2 which is described in the publications [Brega98], [Brega00] and [Tomatis01c].

XO/2 is an object-oriented, hard real-time system software and framework designed for safety, extensibility and abstraction [Brega98]. It takes care of many common issues faced by programmers of mechatronic products by encapsulating general design patterns into easy-to-understand abstractions. Careful handling of the safety aspects has been the criterion by which the system has been crafted. These mechanisms allow the system to maintain a *deus ex-machina knowledge* concerning the running applications thus providing greeter confidence to the application programmer. The latter, relieved from many computer-science aspects, can better focus his attention to the actual problem to being solved.

A.1.1 Safety

The system sets higher standards for safety through a combination of programming paradigms and modern computer-science solutions. In order to understand this claim, a somewhat more technical definition of safety is required.

Safety, as used in the common speech, can be separated into the more technical terms of *safety*, *progress*, and *security*. These terms can be summarized as fol-

lows: Nothing bad happens, the right things do (eventually) happen, and things happen under proper authorization (or potentially bad things happen under proper supervision). All three interact to make a system safe in a broader sense.

A.1.2 The Role of Programming Languages in Safe Systems

System components are the tools of a software engineer. The safer the tools, the more reliable is the system. It is, therefore, natural to expect programming languages to help improve system safety. It is well-known that languages, or more precisely proper language paradigms and type systems, can do a lot in helping programmers to better realize their solutions. Strangely enough, despite the existence of better alternatives, a lot of safety-critical software is implemented by means of programming languages which do a poor job at ensuring safety.

The commonly used argument against using languages which are type-safe, is the inefficiency of the produced code. This misconception can be easily refuted. In the case of static safety, all restrictions are computed by the compiler (at compile-time). Therefore, there is no overhead in the code to be executed. When static safety cannot be enforced, dynamic checks are needed. The added safety, brought by the validation of the programming invariant at run-time, more than compensates the penalty paid in the execution time. In fact, there is no acceptable trade-off for letting a type-violation be passed.

The programming language chosen for XO/2 is Oberon-2 [Mössenböck93]. Oberon is a successor of Pascal and Modula, featuring strong-typing, compatibility by name and not by structure, object-orientation and modularization.

A.1.3 Handling Untyped Operations

In the section above, it is stated that it is not always possible to ensure (type) safety statically, i.e. at compile-time. Notwithstanding the run-time checks needed for object-oriented polymorphic operations, this also holds for each potentially untyped action. Examples in this direction range from NIL-pointer de-referencing, stack overflows, and dangling references to unloaded modules.

Most of the aforementioned errors can be trapped by run-time checks emitted by the compiler. Anyway, the overhead in the execution time cannot be tolerated. A more aggressive technique, avoiding in-program checks, is by means of memory protection. This scheme, usually found in Unix derivatives, cheats the running programs (also called processes) by allocating to them different, disjunct virtual address spaces.

The major drawback of this scheme resides in the overhead which has to be paid for the reloading of the page-table and the memory management unit registers during context-switching. For a real-time system, as with XO/2, which fires the

task scheduler with a time-slice quantum of 100 microseconds, this cannot be tolerated. Supported by the fact that no *explicitly programmed* untyped operation are allowed by the Oberon-2 language, we have favored a more lightweight memory management scheme which helps in catching the possible untyped, unsafe operations emitted by the compiler, without imposing restrictions on what can be shared between programs, nor bringing an unacceptable overhead during context-switching.

The chosen scheme makes pervasive use of the underlying memory management unit of the PowerPC architecture by creating a single virtual address space, where virtual address ranges are allocated to the running processes. Following this method, NIL de-referencing can be mapped to an invalid virtual page. In the same fashion, stacks can be allocated with guard pages between them, thus actively guarding against stack overflows. Additionally, within a virtual address range the stack can be allowed to grow whenever the running task needs it, without asking the programmer to explicitly demand a bigger stack at task creation. Module unloading is handled similarly: When modules are removed from the system their virtual address range is invalidated thus preventing dangling procedure variables to execute upon non-trusted code or data.

A.1.4 Automatic Reclamation of Dynamic Memory

In a highly dynamic, object-oriented, composable system, the central knowledge of all references which exist for a particular object becomes hard to maintain as the dynamic loading of extensions augments. Even worse, it becomes impossible for a programmer to keep track of references in a safe way when the language doesn't impose restrictions on the passing and copying of references. This brings us to the conclusion that in a dynamically extensible system, explicit de-allocation of objects is not feasible. The failure of realizing this introduces a new class of run-time problems such as dangling references and memory leaks: If the object is disposed too soon some stale references could access the object while the same memory block is being referenced by someone else; on the contrary, late or non-existent disposal induces memory leaks, i.e. unused memory is not reclaimed.

The only safe possibility for object reclamation is by means of a system-wide mechanism performing automatic storage reclamation: A so-called garbage collector. A garbage collector decides upon the liveness of heap objects with their reachability, starting from a working set of global and local references. After complete traversal of the heap data structures, objects that have not been visited by the collector's marking are disposed.

XO/2 deploys a very robust, incremental, real-time compatible *mark-and-sweep* garbage collector with object-finalization which combines good collection performance with no memory requirements at execution time. The latter is more

important when the collector is kicked by a low-memory condition, i.e. it can complete the traversal and the collection of the heap-space without demanding memory. Moreover, the proposed solution works very well in a pre-emptive scheduling environment without blocking nor delaying tasks performing access to heap-objects.

A.1.5 Modularization and Separation of Concerns

One of the most important design principles is the separation of concerns. This principle requires a system to be structured into sub-systems, also called modules. Modules should expose an interface by exporting functions and procedures to the clients. The functionality of a module is accessed only by means of its interface; the interface can be generalized enough to hide most of the implementation details thus establishing and guaranteeing invariants for its states and procedures. Disjoint, orthogonal modules implementing this design principle can be exchanged without invalidating clients, therefore, leading to a dynamic composition of the system.

An important pre-condition for the realization of this design principle is the presence of safe dynamic loading and unloading of compilation units. XO/2 provides the required safety by checking at compile time and at linking-loading time the formal interfaces against the actual ones. Only interface-compatible modules may be loaded in the system without threatening the safety of the dynamic composition. A different, non-trivial task resides in the dynamic unloading of modules, i.e. when a module can be safely removed from the system. By means of reference counting, lexical scopes and virtual memory ranges XO/2 can guarantee that a needed module will not be unloaded and that stale references will be trapped before execution.

The presence of safe dynamic loading and unloading in XO/2, along with very short edit-compile-run cycles, has been one of its most appreciated features. In fact, during the development of a complex application different programmers can safely test new code modules without threatening the stability of the system and applications. It is not uncommon that an XO/2 machine will run, uninterrupted, over several days, during which application programmers actively program and test part of a software constellation.

A.1.6 Process Scheduling

The principal responsibility of a real-time operating system can be summarized as that of producing correct results while meeting pre-defined deadlines in doing so. Therefore, the computational correctness of the system depends on both the

logical correctness of the results it produces and the timing correctness, i.e. the ability to meet the deadlines of its computation.

A real-time application can be modeled as a set of cooperating tasks. These tasks can be classified according to their timing requirements as hard real-time, and non real-time. A hard real-time task is a task whose timely execution is labelled as critical to the operation of the whole system. Consequently, it is assumed that the missing of the deadline can result in a system failure. Non real-time tasks are those tasks which exhibit no real-time requirements (e.g. system maintenance tasks running in the background).

XO/2's real-time process manager implements a static, *earliest-deadline-first* scheduling algorithm with *admission testing*. With this algorithm the pool of real-time tasks is statically sorted according to their deadlines. The first one, i.e. the one with the shortest deadline, will be set for execution. This task will remain in the foreground until its normal execution cycle is completed, or when a task characterized by a shorter deadline has been activated by the occurrence of some event, such as the expiration of a waiting period or user intervention. The process manager is also responsible for dispatching non real-time tasks, also called threads. Since their computations can be delivered any time, threads are brought to the foreground only when no other real-time task is pending, waiting for being dispatched. The non–real-time scheduler chooses the thread to be scheduled according to its priority. Threads carrying the same priority are taken in the foreground in a round-robin fashion. *Anti-starvation* mechanisms and *priority inheritance* guarantee fairness and progress.

A.1.7 Other OS Functionality

XO/2 provides a wide range of non-core functionality such as multiple-filesystem support, streams-based I/O, TCP/IP networking software, internet standard servers and clients and object-oriented databases of periphery hardware.

The robust, threaded TCP/IP stack allows a wide array of network protocols to be implemented. A standard, full-blown release of XO/2 comes with a Telnet server, an FTP server, a HTTP server, TFTP, SNTP, SMTP and POP3 clients. Applications looking for computer-based man-machine interaction are usually realized as a set of web-pages. A client agent, like a web browser, can present on-robot information without the need of post-processing, or call CGI-commands which are actually normal application entry-points exposed as commands. Object linearization through XML allows browsers, java-applets or custom solutions to access remote objects on the HTTP transport. Facilities like the in-system creation of GIF and JPG images are used for streaming visual information to the client.

A.2 Software Structure

The software structure of the ASL robots has already been briefly presented in [Brega00] and [Tomatis01c]. The structure has been developed by making extensive use of the facilities provided by XO/2. Every *critical*, periodic task is implemented as a *hard real-time* task. The operating system and programming language guarantee both safety and security of the executed code while ensuring progress and timing correctness through the deadline driven scheduler.

When unexpected events occur the operating system is responsible for supervising a proper counter-measure.

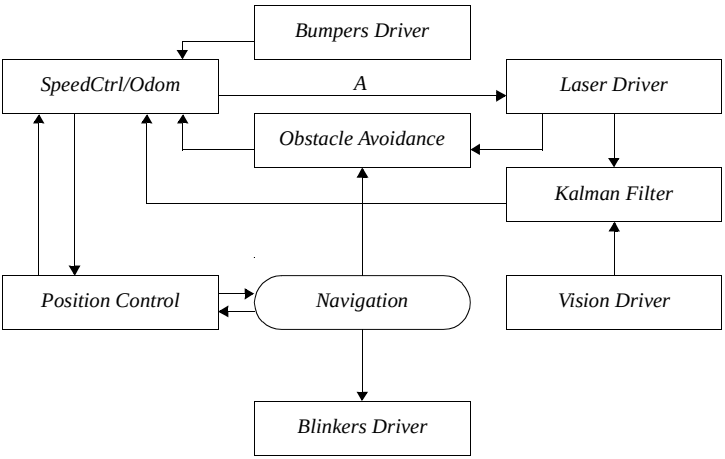

Figure A.1: The tasks constellation deployed on Pygmalion for the experiments of Section 4.1. Rectangles indicate hard real-time tasks and arrows stand for information flow. The former are the only means to guarantee calculation time for vital functions under the typical conditions of unpredictable processor load.

The task constellation running on Pygmalion for the experiments of Section 4.1 is depicted in Fig. A.1. The bumper driver and the speed controller which calls also odometry, run at 1 KHz. The position controller is scheduled with a frequency of 50 Hz. The Acuity laser scanner takes 360 ms for a complete mirror revolution and compensates on-the-fly the distortion imposed by the vehicle movement during acquisition for each arriving range reading (this explains arrow *A*). Obstacle avoidance and localization are vital functions and are also installed as RT task with an appropriate worst case period. The multi-sensor localization cycle is trivially limited by the slow period of the laser scanner. Temporal coherence of acqui-

sition and localization results which are in the past, is guaranteed by means of time stamps provided by odometry to the vision and the laser driver. Forward simulation of the odometry model finally yields the actually valid position update. The navigation module, implemented as a non real-time task, is used for global planning and mission control.

A.3 Obstacle Avoidance

The obstacle avoidance approach has been developed during a diploma work [Persson00].

Obstacle avoidance in mobile robotics is the problem of moving from a starting point to a given goal with an arbitrary constellation of obstacles. For this, the approach has to take into account the kinematic and sensing capabilities of the robots.

The algorithm we devised has been split into two independent modules: The first one implementing local obstacle avoidance and the second one implementing a path planner which generates a path of points from the start to the goal. The local obstacle avoidance is based on the *Dynamic Window* approach. The safe robot behavior is ensured by selecting only motion commands which allow the robot to come to a halt before collision. Above the dynamic window there is a path planner which gives intermediate goal points to the local obstacle avoidance. The path planner makes use of the distance transform which is a grid-based wave propagation technique (*NF1 function*): The raw laser scan points are put into a grid and a wave propagates from the goal point until all points in the grid have been reached. A path from the start to the goal is eventually found by following the negative gradient from the start.

A.4 Web Interfacing

In the framework of a research project on mobile robot localization a graphical web interface for our indoor robots has been developed [Moreau00], [Tomatis01a]. The purpose of this interface is twofold: It serves as a tool for task supervision for the researcher and task specification for the end-user.

The interface has been developed for our fully autonomous indoor robots. They have three main sensors: The wheel encoders, a 360° laser range finder and a CCD camera. The used feature based localization approach uses an Extended Kalman Filter to integrate measurements from the encoders, the laser scanner and the CCD camera as presented in Section 4.1.

A.4.1 Supervision

Testing algorithms such as localization and obstacle avoidance on an autonomous self-contained robot [Brega00] requires a means for the researcher to check the algorithmic reactions to the machine's perception of the world. However, the perceived data and the processing results remain embedded in the mobile vehicle until they are explicitly transferred to a local PC for analysis. This can be performed by tracing the robot position (odometry), by saving all the raw data from the sensors, the extracted features and the results of each algorithm and then by transferring this information when an experiment is finished. The analysis can then be performed off-board. Nevertheless, this procedure has several disadvantages:

- The correspondence between the behavior of the robot and the data which caused this behavior, is difficult to identify.
- Critical states of algorithms which may cause a failure cannot be detected immediately before and are, therefore, difficult to isolate and correct.
- Crashes of one, or more processes may cause a loss of information which would be important for the off-line analysis.

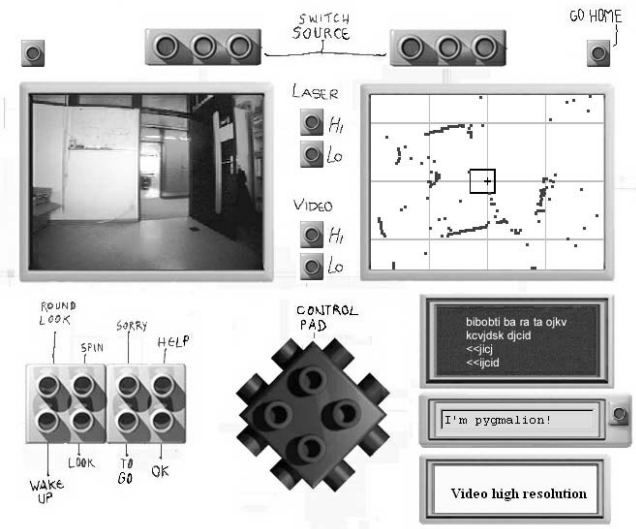

Figure A.2: The first web interface which has been developed for task supervision only. It shows the raw data from each sensors, and each predicted, observed and matched features, as well.

On-line supervision (Fig. A.3) is, therefore, not an option. It is instead an important tool for speeding up the advances in applications such as mobile robotics research by allowing on-line detection of characteristics of the tested approaches. Having access to the machine perception permits the identification of the correspondence between the perception and behavior of the robot. This is performed by visualizing sensory information on several levels of abstraction using state-of-the-art web technology yielding a plug-in-free interface which can be viewed with a standard browser. It provides multimodal information in several representations: Off- and on-board vision, laser and odometry (Fig. A.2).

This tool proved to be indispensable in the development phase of navigation algorithms for localization, obstacle avoidance and path planning ([Arras00], [Arras01a], [Tomatis01b], [Tomatis01d] and [Tomatis01e]).

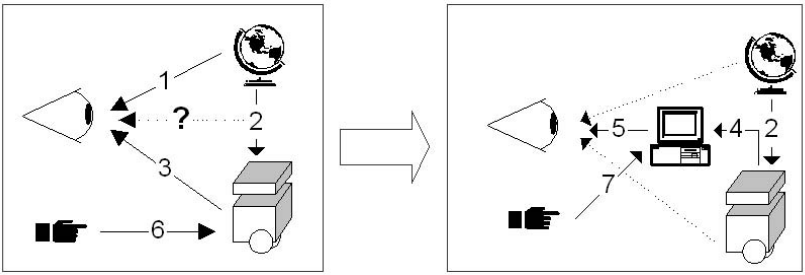

Figure A.3: For research and development in mobile robotics the correspondence between robot behavior (3) and robot perception (2) is very important. This correspondence is easier to understand by using on-line supervision: The robot perception is visualized in a graphical interface (4-5). 1: Human perception of the real world. 2: Machine perception of the real world. 3: Human perception of the robot behavior. 4: On-line transfer of the machine perception and machine states. 5: On-line visualization of the machine perception and machine states. 6: Human commands via in-line text commands. 7: Visual task specification.

A.4.2 Specification

By performing public presentations and common experiments with distant labs some limitations of our system become evident. Obscure inline commands were used to control the robot and the only feedback was the robot behavior and some text output. This was not satisfactory for people who were not familiar with this development and operating system. Including task specification in a graphical

feedback interface for making the results of the research in robotics accessible to potential end-users became a major aspect of the development.

For this purpose a means for controlling the robot which was not a basic aim of the project, has been developed. This has been achieved by introducing modern guidelines for ergonomic interface design such as context-sensitive popup menus, or clickable goal specifications. Defining a navigation goal on a graphical interface by clicking on an image showing the known environment makes the interface very intuitive for the end-user. In the same way local goals near the current robot position can be defined on the image showing the neighborhood of the robot and the raw data from the laser scanner. Furthermore, the robot behavior can even be seen by distant users by means of external cameras (fig 2).

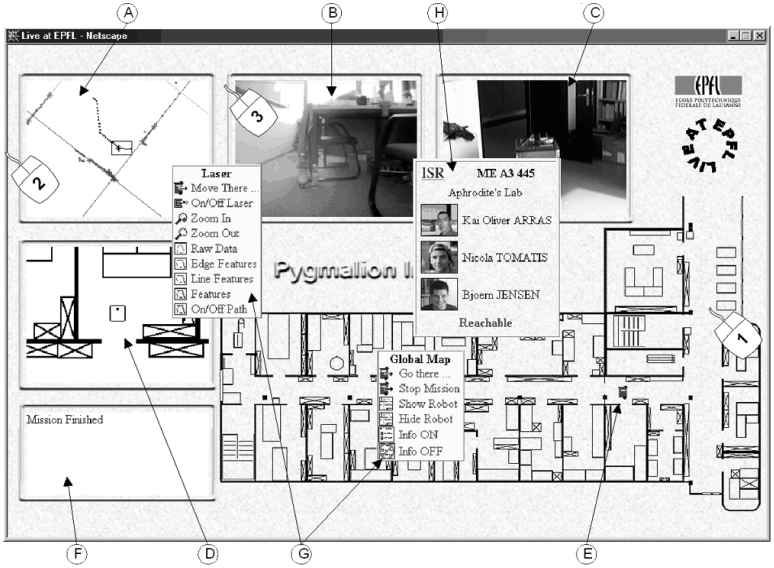

Figure A.4: The web interface. A: Multi-sensor localization monitor. B: On board video. C: External video. D: Local robot position (x, y, θ). E: Global robot position (x, y). F: Message window. G: Popup menu on each window to access corresponding functionality. H: Office information popup menu on global map. Numbered mice are a possible way to control the robot. Mouse1: Set a room as the new global goal. Mouse2: Set (x, y) as the new local goal. Mouse3: assign new orientation θ.

Its practicability has been extensively demonstrated in the “Computer2000” exhibition event (Section 4.1.6 and Appendix B.3) where over a period of 4 days the robot was remote controlled in a fully autonomous mode by visitors of the tradeshow using this interface. The visitors defined 724 missions for the robot which had to travel a total of more than 5 Km in order to fulfill them.

B

Events

"One of the extraordinary things about human events is that the unthinkable becomes thinkable."

Salman Rushdie (born 1948)

B.1 Eurobot '99

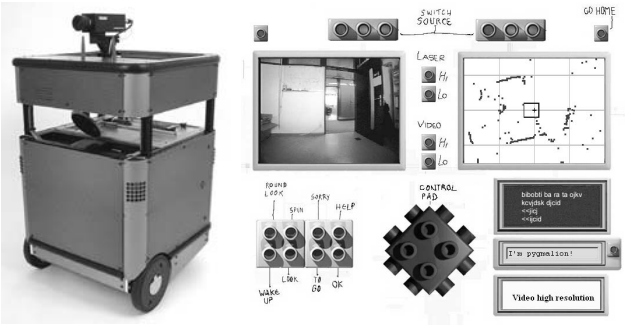

Figure B.1: Pygmalion and the web interface which has been exhibited at the Eurobot '99 demonstration in Zurich.

After having submitted the paper for the Eurobot '99 conference [Arras99a] we thought that bringing Pygmalion to Zurich where the conference had to take place, for a demonstration during the event could be an interesting experience. The goal was to show our strength in putting robotic theory into practice. For this purpose we defined some new goals which had to be attained with the robot:

- On-the-fly navigation (Section 4.1.5): The robot was localizing in a step-by-step way until the experiments made for the Eurobot paper. For the demonstration we were interested in localization during motion for aesthetic purposes and to gain experience in facing the problems which would then arise.
- Web interfacing (Appendix A.4): We also wanted to show the internal state of the robot which permits better understanding how it works.

Pygmalion did a short 50 meters tour of the Institute of Robotics, ETH Zurich, showing the performance of the EFK on-the-fly localization approach even by using the old Acuity AccuRange 4000 laser sensor.

B.2 ICRA 2000

In April 2000 at the IEEE International Conference on Robotics and Automation (ICRA) in San Francisco, USA, there was a workshop on "Internet Robotics". This workshop has been co-organized by Prof. Roland Siegwart and represented the first intercontinental testbed for our web interface (Appendix A.4). The web interface we were developing for the Computer 2000 event has been successfully tested during this workshop on the 24th April. Furthermore, both Pygmalion (controlled from San Francisco) and Donald Duck were moving together under supervision (web cam on the interface) of ICRA.

B.3 Computer 2000

The "Computer" trade show is an annual fair for computer hard- and software at the *Palais de Beaulieu* Exposition Centre in Lausanne, Switzerland. Our laboratory was present throughout the four days –from May 2nd to May 5th 2000– giving visitors the opportunity to control Pygmalion by means of web-interfacing (Fig. A.4) and to interact with Daffy Duck on site (Fig. B.2).

Daffy Duck used for physical interaction the CCD camera as an input device locating human beings in the environment. The robot detects human motion using statistical change detection algorithms. Output devices are the pan-tilt head being directed towards the chosen user which is addressed by the robot with synthesized speech. Simple tracking algorithms allow the robot to interact with the person even when he is moving.

The Computer 2000 event was the final testbed for the metric localization system of Section 4.1 where we were mainly interested in long-term reliability under application-like conditions. Furthermore, it was a great way to test our approaches in physical man-machine interaction (Fig. B.2) and web-based interfaces (Fig. A.4). The setup was active during normal office hours with an average of approximately 7 hours up-time per day. The environment exhibited typical

dynamics from people, doors, chairs and other robots, as well as daylight illumination. Several doors were open to the corridor thus limiting the space available for robots maneuvers. Travel speed has been limited to 0.4 m/s since the robot had to share its environment with persons, some of them not engaged in robotics. The obstacle avoidance was active during the event. The web-interface (Fig. A.4) allowed navigation commands (e.g. go to office) to be given to the robot.

Figure B.2: Daffy Duck, the youngest of our robots, after detecting a young visitor interacts with him by speaking, tracking him and moving the pan-tilt head for mimicking expressions. It detects human movement by means of the CCD camera and closes the interaction loop with its speech synthesizer and gestures as a combination of robot and pan-tilt movement.

B.4 IROS 2000

At the 2000 edition of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) which took place in Takamatsu, Japan, there were some special events scheduled during the breaks between the technical sessions. One of these was the presentation of a web interface supervising a mobile robot. However, the LAAS-CNRS, Toulouse had problems in making the application work. Therefore, we found out an alternative: From Takamatsu, we called Björn Jensen in Lausanne who was sleeping at home. After half a hour Björn was booting our system and, after some other minutes, Pygmalion was ready. We presented our web interface and also had some help from Lausanne by our Japanese collaborator, Kunitoshi Tazume, who wrote some words in Japanese for the people in Takamazu. It is not difficult to understand that all the people who saw our demonstration were impressed.

B.5 Science & Cité

Science & Cité is an event which has been organized in the whole of Switzerland in order to make the people aware of the role of the universities in our country. For this we presented our web interface during the whole event –from May 5th to May 15th 2000– from the *Espace Arlaud* in Lausanne. In this case Donald Duck was active for 6 to 10 hours per day. It made many km in order to show the visitors at the event what we are doing in the mobile robotics research domain.

Literature

"Copy from one, it's plagiarism; copy from two, it's research."

Wilson Mizner (1876–1933)

[Adams99] Adams, M.D. (1999). Sensor Modelling, Design and Data Processing for Autonomous Navigation. World Scientific Series in Robotics and Intelligent Systems, Singapore.

[Arleo99] Arleo, A. J. R. Millán et al. (1999). "Efficient Learining of Variable-Resolution Cognitive Maps for Autonomous Indoor Navigation." IEEE Transaction on Robotics and Automation. **15**(6): 990-1000.

[Arras97] Arras, K.O. and R.Y. Siegwart (1997). Feature Extraction and Scene Interpretation for Map-Based Navigation and Map Building, Proceedings of SPIE, Mobile Robotics XII, Vol. 3210.

[Arras99a] Arras, K. O. and N. Tomatis (1999). Improving Robustness and Precision in Mobile Robot Localization by Using Laser Range Finding and Monocular Vision. Third European Workshop on Advanced Mobile Robots (Eurobot), Zurich, Switzerland.

[Arras99b] Arras, K. O. and N. Tomatis (1999). Sensordatenfusion zur Robusten und Präzisen EKF Lokalisierung von Mobilen Robotern. 15. Fachgesprächs Autonome Mobile Systeme (AMS 99), München, Deutschland.

[Arras00] Arras, K., N. Tomatis, et al. (2000). Multisensor On-the-Fly Localization Using Laser and Vision. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Takamatsu, Japan.

[Arras01a] Arras, K. O., N. Tomatis, et al. (2001). “Multisensor On-the-Fly Localization: Precision and Reliability for Applications.” Robotics and Autonomous Systems **34**(2-3): 131-143.

[Arras01b] Arras, K. O., J. A. Castellanos, et al. (2001). Towards Feature-Based Multi-Hypotheses Localization and Tracking. Fourth European Workshop on Advanced Mobile Robots (Eurobot), Lund, Sweden.

[Astolfi95] Astolfi A. (1995). Exponential Stabilization of a Mobile Robot. Third European Control Conference, Roma, Italy.

[Brega98] Brega, R. (1998). A real-time operating system designed for predictability and run-time safety. Fourth International Conference on Motion and Vibration Control (MOVIC), Zurich, Switzerland.

[Brega00] Brega, R., N. Tomatis, et al. (2000). The Need for Autonomy and Real-Time in Mobile Robotics: A Case Study of XO/2 and Pygmalion. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Takamatsu, Japan.

[Cassandra96] Cassandra, A. R., L. P. Kaelbling, et al. (1996). Acting under Uncertainty: Discrete Bayesian Models for Mobile-Robot Navigation. IEEE International Conference on Robotics and Automation (ICRA), Osaka, Japan.

[Castellanos96] Castellanos, J.A., J.D. Tardós, et al. (1996). Constraint-Based Mobile Robot Localization. International Workshop on Advanced Robotics and Intelligent Machines, Salford, Great Britain.

[Castellanos97] Castellanos, J. A., J. D. Tardós, et al. (1997). Building a Global Map of the Environment of a Mobile Robot: The Importance of Correlations. IEEE International Conference on Robotics and Automation (ICRA), Albuquerque.

[Castellanos00] Castellanos, J. A., M. Devy, et al. (2000). Simultaneous Localization and Map Building for Mobile Robots: A Landmark-based Approach. IEEE International Conference on Robotics and Automation (ICRA), San Francisco.

[Chenavier92] Chenavier, F., J.L. Crowley, (1992). Position Estimation for a Mobile Robot Using Vision and Odometry. IEEE International Conference on Robotics and Automation (ICRA), Nice, France.

[Chong97] Chong, K.S., Kleeman L.(1997). Accurate Odometry and Error Modeling for a Mobile Robot. IEEE International Conference on Robotics and Automation (ICRA), NM, USA.

[Crowley89] Crowley, J.L., (1989). World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging. IEEE International Conference on Robotics and Automation (ICRA), Scottsdale, AZ.

[Duckett99] Duckett, T. and U. Nehmzow (1999). Knowing your place in real world environments. Third European Workshop on Advanced Mobile Robots (Eurobot), Zurich, Switzerland.

[Dellaert99] Dellaert, F., D. Fox, et al. (1999). Monte Carlo Localization for Mobile Robots. IEEE International Conference on Robotics and Automation (ICRA), Detroit, Michigan.

[Fox98] Fox, D. (1998). Markov Localization: A Probabilistic Framework for Mobile Robot Localization and Navigation. Ph.D. Thesis, Institute of Computer Science III. Bonn, Germany, University of Bonn.

[Gutierrez96] Gutierrez-Osuna, R. and R. C. Luo (1996). “LOLA: Probabilistic Navigation for Topological Maps.” AI Magazine **17**(1): 55-62.

[Gutmann96] Gutmann, J.-S., C. Schlegel, et al. (1996). Comparison of Scan Matching Approaches for Self-Localization in Indoor Environments. First European Workshop on Advanced Mobile Robots (Eurobot), Keiserslautern, Germany.

[Gutmann98] Gutmann, J.-S., W. Burgard, et al. (1998). An Experimental Comparison of Localization Methods. IEEE International Conference on Intelligent Robots and Systems (IROS), Victoria, B. C., Canada.

[Hough62] Hough, P.V.C. (1962). Method and Means for recognizing complex patterns. US Patent 3069654.

[Julier01] Julier, S. J., J. K. Uhlmann (2001). A Counter Example to the Theory of Simultaneous Localization and Map Building. IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea.

[Kaelbling95] Kaelbling, L. P., M. L. Littman, et al. (1995). Partially Observable Markov Decision Processes for Artificial Intelligence. International Workshop, Reasoning with Uncertainty in Robotics, RUR '95, Amsterdam, Netherlands, Springer.

[Koenig95] Koenig, S., R. Goodwin, et al. (1995). Robot Navigation with Markov Models: A Framework for Path Planning and Learning with Limited Computational Resources. International Workshop, Reasoning with Uncertainty in Robotics, RUR '95, Amsterdam, Netherlands, Springer.

[Koenig98] Koenig, S. and R. G. Simmons (1998). Xavier: A Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models. Artificial Intelligence and Mobile Robots. D. Kortenkamp, R. P. Bonasso and R. Murphy, The AAAI Press/ The MIT Press: 91-122.

[Kuipers87] Kuipers, B. J. and Y. T. Byun (1987). A qualitative approach to robot exploration and map-learning. Workshop on Spatial Reasoning and Multi-Sensor Fusion, Los Altos, CA, USA, Morgan Kaufmann.

[Kunz99] Kunz, C., T. Willeke, et al. (1999). “Automatic Mapping of Dynamic Office Environments.” Autonomous Robots **7**:131-142.

[Lamon01] Lamon, P., I. Nourbakhsh, et al. (2001). Deriving and Matching Image Fingerprint Sequences for Mobile Robot Localization. IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea.

[Lanser92] Lanser, S. and C. Zierl (1996). MORAL: Ein System zur videobasierten Objekterkennung im Kontext autonomer, mobiler Systeme. Autonome Mobile Systeme 96, Informatik Aktuell, Springer Verlag: 88-98.

[Leonard92] Leonard, J.J. and H.F. Durrant-Whyte, (1992). Directed Sonar Sensing for Mobile Robot Navigation. Kluwer Academic Publishers.

[Lu94] Lu F., E. Milios, (1994). Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA.

[Maybeck90] Maybeck, P. S. (1990). The Kalman Filter: An Introduction to Concepts. Autonomous Robot Vehicles, Springer Verlag: 194-204.

[Matthies88] Matthies, L. and A. Elfes, (1988). Integration of sonar and stereo range data using a grid-based representation. IEEE International Conference on Robotics and Automation (ICRA), Philadelphia, PA.

[Moravec88] Moravec, H. (1988). Mind Children, The Future of Robot and Human Intelligence. Harvard University Press.

[Moreau00] Moreau, B., N. Tomatis, et al. (2000). A Multi Modal Web Interface for Task Supervision and Specification. Proceedings of SPIE, Mobile Robots XV and Telemanipulator and Telepresence Technologies VII, Vol. 4195.

[Mössenböck93] Mössenböck, H. (1993). Object-Oriented Programming in Oberon-2. Springer-Verlag.

[Muñoz98] Muñoz, A.J., J. Gonzales (1998). Two-Dimensional Landmark-based Position Estimation from a Single Image. IEEE International Conference on Robotics and Automation (ICRA), Leuven, Belgium.

[Neira99] Neira, J., J.D. Tardos, et al. (1999). "Fusing Range and Intensity Images for Mobile Robot Localization." IEEE Transactions on Robotics and Automation **15**(1):76-84.

[Nourbakhsh98] Nourbakhsh, I. (1998). Dervish: An Office-Navigating Robot. Artificial Intelligence and Mobile Robots. D. Kortenkamp, R. P. Bonasso and R. Murphy, The AAAI Press/ The MIT Press: 73-90.

[Pérez99] Pérez, J.A., J. A. Castellanos, et al. (1999). Continuous Mobile Robot Localization: Vision vs. Laser. IEEE International Conference on Robotics and Automation (ICRA), Detroit.

[Persson00] Persson, J. (2000). Obstacle Avoidance for Mobile Robotics. Diploma thesis, Department of Electrical Engineering, Linköpings University, Linköpings, Sweden.

[Prescott97] Prescott, B., G.F. McLean. (1997). "Line-Based Correction of Radial Lens Distortion". Graphical Models and Image Processing. **59**(1): 39-47.

[Smith88] Smith, R. C., M. Self, et al. (1988). Estimating uncertain spatial relationships in robotics. Uncertainty in Artificial Intelligence 2. J. F. Lemmer and L. N. Kanal, Elsevier Science Pub.: 435-461.

[Thrun96] Thrun, S. and A. Bücken (1996). Integrating grid-based and topological maps for mobile robot navigation. National Conference on Artificial Intelligence, AAAI, Portland, OR, USA.

[Thrun98] Thrun, S., J.-S. Gutmann, et al. (1998). Integrating Topological and Metric Maps for Mobile Robot Navigation: A Statistical Approach. Tenth Conference on Innovative Applications of Artificial Intelligence, Madison, WI, USA.

[Tomatis01a] Tomatis, N., Moreau B. (2001). "Web Interfacing for Task Supervision and Specification". SPIE's International Technical Group Newsletters: Robotics and Machine Perception, March 2001.

[Tomatis01b] Tomatis, N., I. Nourbakhsh, et al. (2001). A Hybrid Approach to Robust and Precise Mobile Robot Navigation with Compact Environment Modeling. IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea.

[Tomatis01c] Tomatis, N., R. Brega, et al. (2001). A Complex Mechatronic System: From Design to Application. IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM '01), Como, Italy.

[Tomatis01d] Tomatis, N., I. Nourbakhsh, et al. (2001). Simultaneous Localization and Map Building: A Global Topological Model with Local Metric Maps. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Maui, Hawaii.

[Tomatis01e] Tomatis, N., I. Nourbakhsh, et al. (2001). Combining Topological and Metric: A Natural Integration for Simultaneous Localization and Map Building. Fourth European Workshop on Advanced Mobile Robots (Eurobot), Lund, Sweden.

[Ulrich00] Ulrich, I., I. Nourbakhsh (2000). Appearance-Based Place Recognition for Topological Localization. IEEE International Conference on Robotics and Automation (ICRA), San Francisco, CA, USA.

[VanZwyns01] Van Zwynsvoorde, D., T. Simeon, et al. (2001). Building Topological Models for Navigation in Large Environments. IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea.

[Weng91] Weng, J., P. Cohen, et al. (1991). "Camera Calibration with Distortion Models and Accuracy Evaluation." IEEE Transaction on Pattern Analysis and Machine Intelligence. **13**(4): 370-376.

[Wullschleger99] Wullschleger, F. H., K. O. Arras, et al. (1999). A Flexible Exploration Framework for Map Building. Third European Workshop on Advanced Mobile Robots (Eurobot), Zurich, Switzerland.

Curriculum Vitae

Born 29 January 1973, I grew up in Riazzino, TI, Switzerland. However, my place of origin is Arvigo, GR. In 1992, I graduated from the *Liceo Cantonale di Locarno* high school in science (Type C). I entered the *Swiss Federal Institute of Technology Zurich* the same year from which I graduated in 1998 with a M.S. in Computer Science (*Dipl. Informatik-Ing. ETH*). I completed my study with the diploma work "Vision Feedback for Mobile Robots". During the next 6 months I worked as research assistant at the *Institute of Robotics, Swiss Federal Institute of Technology Zurich*, on the development of a frame grabber hardware and software application for embedded systems. As of the end of 1998 I have been at the Autonomous Systems Lab, Institute of Robotics Systems under the supervision of Prof. Roland Siegwart.