ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

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

Semantic Labeling of Places with Mobile Robots

Óscar Martínez Mozos

Betreuer: Prof. Dr. Wolfram Burgard

Dekan der Fakultät für Angewandte Wissenschaften:
Prof. Dr. Bernhard Nebel

Gutachter:

1. Prof. Dr. Wolfram Burgard
   Albert-Ludwigs-Universität Freiburg, Freiburg, Deutschland.

2. Prof. Dr. Aleš Leonardis
   Univerza v Ljubljani, Ljubljana, Slowenien.

Tag der Disputation: 04.07.2008

Zusammenfassung

Innenumgebungen können normalerweise in Orte verschiedener Funktionalität unterteilt werden, wie etwa Korridore, Räume oder Türdurchgänge. Die Fähigkeit solch semantische Kategorien aus Sensordaten zu lernen, ermöglicht einem Roboter die Umgebungsrepräsentation zu erweitern und seine Fertigkeiten zu verbessern. So können beispielsweise natürlichsprachliche Ausdrücke, wie etwa *Korridor* oder *Raum*, dazu verwendet werden um die Position des Roboters in einer intuitiveren Weise mitzuteilen. Andere Aufgaben, wie Exploration oder Lokalisierung, können durch den Roboter ebenfalls besser bewältigt werden, wenn dabei semantische Informationen berücksichtigt werden.

In dieser Doktorarbeit wird ein Verfahren vorgestellt, das einem mobilen Roboter ermöglicht, verschiedenen Orten in Innenumgebungen semantische Kategorien zuzuordnen und die Umgebungsrepräsentation durch diese Information zu erweitern. Die Kernidee dabei ist, die Position des Roboters durch die aktuellen Beobachtungen des Roboters zu klassifizieren. Die Beobachtungen, die wir in der vorliegenden Arbeit nutzen werden, sind die Abstandsmessungen eines Laserscanners. Aus jedem Scan wird eine Menge von Merkmalen extrahiert, welche Informationen über die geometrischen Eigenschaften des Ortes an dieser Position tragen. Der Scan wird dann aufgrund dieser Merkmale in die entsprechende semantische Kategorie eingeteilt. Das Ergebnis der Klassifikation ist eine Wahrscheinlichkeitsverteilung über der Menge möglicher semantischer Klassen. Diese wahrscheinlichkeitstheoretische Darstellung erlaubt es uns weitere probabilistische Techniken einzusetzen, um die Klassifikation zu verbessern, und somit die Anzahl an Fehlern zu verringern. Wir werden auch eine Erweiterung des Verfahrens vorstellen, die es dem Roboter ermöglicht andere Beobachtungsmodalitäten, wie etwa Kamerabilder, in den Klassifikationsprozess zu integrieren.

Zusätzlich stellen wir in dieser Arbeit verschiedene Anwendungen des obigen Verfahrens im Bereich der Robotik vor. Zunächst werden wir zeigen, wie wir semantische Informationen dazu verwenden können, topologische Karten von Innenumgebungen zu erstellen. In einer zweiten Anwendung stellen wir eine Methode vor, welche die übergänge zwischen verschiedenen Orten in Betracht zieht, um die Trajektorie eines mobilen Roboters zu klassifizieren. Zudem kann durch Berücksichtigung semantischer Information die von einem Roboter benötigte Zeit für Explorations- und Lokalisierungsaufgaben verringert werden. Die vorliegende Arbeit wird solche Verbesserungen aufzeigen. Schließlich stellen wir die seman-

tische Klassifikation als Teil eines Robotersystems vor, das dazu ausgelegt ist, mit Menschen in natürlicher Sprache zu kommunizieren.

Summary

Indoor environments can typically be divided into places with different functionalities like corridors, rooms or doorways. The ability to learn such semantic categories from sensor data enables a mobile robot to extend the representation of the environment, and to improve its capabilites. As an example, natural language terms like *corridor* or *room* can be used to communicate the position of the robot in a more intuitive way. Other tasks, like exploration or localization, can also be carried out by the robot in a better way when semantic information is taken into account.

In this thesis, we present a method that enables a mobile robot to classify the different places of indoor environments into semantic classes, and then use this information to extend its representations of the environments. The main idea is to classify the position of the robot based on the current observations taken by the robot. In this work, we use as main observations the scans obtained from a laser range sensor. Each scan is represented by a set of features that encode the geometrical properties of the current position. These features are then used to classify the scan into the corresponding semantic class. The output of the classification is represented by a probability distribution over the set of possible semantic classes. This probabilistic representation permits us to apply further probabilistic techniques to improve the final classification, reducing the number of errors. We also present an extension which enables the robot to include other types of observations in the classification, like camera images.

This work additionally introduces several applications of the previous approach in different robotic tasks. First, we will show how the semantic information can be used to extract topological maps from indoor environments. In a second application, we present a method that incorporates transitions between different places when classifying a trajectory taken by a mobile robot. It will also be shown that the semantic information can reduce the time needed by the robot in exploration and localization tasks. Finally, we present the semantic classification of places as part of an integrated robotic system designed for interacting with humans using natural language.

Acknowledgments

I would like to thank all the people who made this thesis possible.

First of all, I would like to thank Prof. Wolfram Burgard for giving me the opportunity to work in his research group. I must admit that this was one of the most useful experiences I have ever had.

Many thanks to Prof. Aleš Leonardis for accepting being my co-supervisor. I really appreciate his interest in my work.

The research presented in this work was carried out under the support of the EU project CoSy. Many thanks to all the researches involved in this project. Special mention to Hendrik Zender and Patric Jensfelt for some very long days preparing demos.

During my stay at the Autonomous Intelligent Systems research lab I had the opportunity to work with many people. Thanks to Cyrill Stachniss for his help during my first steps in the lab, and for his contributions to this thesis. To Axel Rottmann for being also a very good contributor (and for lending me his mixer). Thanks to Rudolph Triebel who contributed not only to my thesis, but also to my German (and sometimes to my Spanish as well). Christian Plagemann, Jürgen Sturm and Daniel Meyer-Delius: thanks for the good atmosphere while sharing the very bright office room 1015 with me. And thanks to Kai Arras for being so friendly when collaborating with me.

Many other students and colleagues joint the lab during these last years: Patrick, Barbara, Kai, Boris, Basti, Reiner, Hauke, Dominik, Malu, Giorgio, Slawomir. Some others left: Maren, Dirk. I think all of them deserve my gratitude.

It is not easy for a Spaniard like me to live six years in Germany...and stay alive. Many thanks to all the friends I met in Freiburg.

And finally, although a little bit far away, many many thanks to my friends in Spain who always give me a very warm welcome when I visit them.

Mama, papa, hermano y resto de familia, por supuesto que vosotros sois los más importantes.

To my family and friends
A mi familia y amigos

Contents

Chapter 1

Introduction

1.1 Semantic Labeling of Places in Indoor Environments

Building accurate maps of indoor environments is one of the typical problems in mobile robotics. In this task, a mobile robot moves along a trajectory while gathering information with sensors. This information is then used to construct the map. Typical maps represent the parts in the environment which are occupied by objects, as for example occupancy grid maps [Elfes, 1989; Moravec, 1988]. The maps are then used for localization and navigation tasks.

The problem of how to augment such maps by adding semantic information has been of increasing interest during the last years. For a lot of applications, robots can improve their service if they are able to recognize places and distinguish them. A robot that possesses semantic information about the type of places can enrich its human-robot communication capabilities and easily be instructed, for example, to "open the door to the corridor." The semantic information can additionally augment the abilities of a robot in other tasks such as localization or exploration.

In this dissertation, we consider the problem of semantically classifying the different locations of indoor environments using a mobile robot. An example is given in Figure 1.1. The top image shows the occupancy map corresponding to the ground floor of the building 52 at the University of Freiburg. In this map, only information about occupied and free space is given. However, some natural divisions can be extracted from this environment, as for example rooms, doorways and a corridor. These divisions are depicted in the bottom image of Figure 1.1. Using this information, the robot can communicate its position in a more natural way saying, for instance, that it is placed "in the corridor," or "in the doorway leading to the corridor." It seems clear that communication becomes more human-friendly.

The process of applying a term to some division of the map is also known as a *semantic labeling* of the map. In this labeling process, the terms refer to semantic expressions that relate each division of the map to some place or objective situation. For example, a corridor implies a place which communicates different

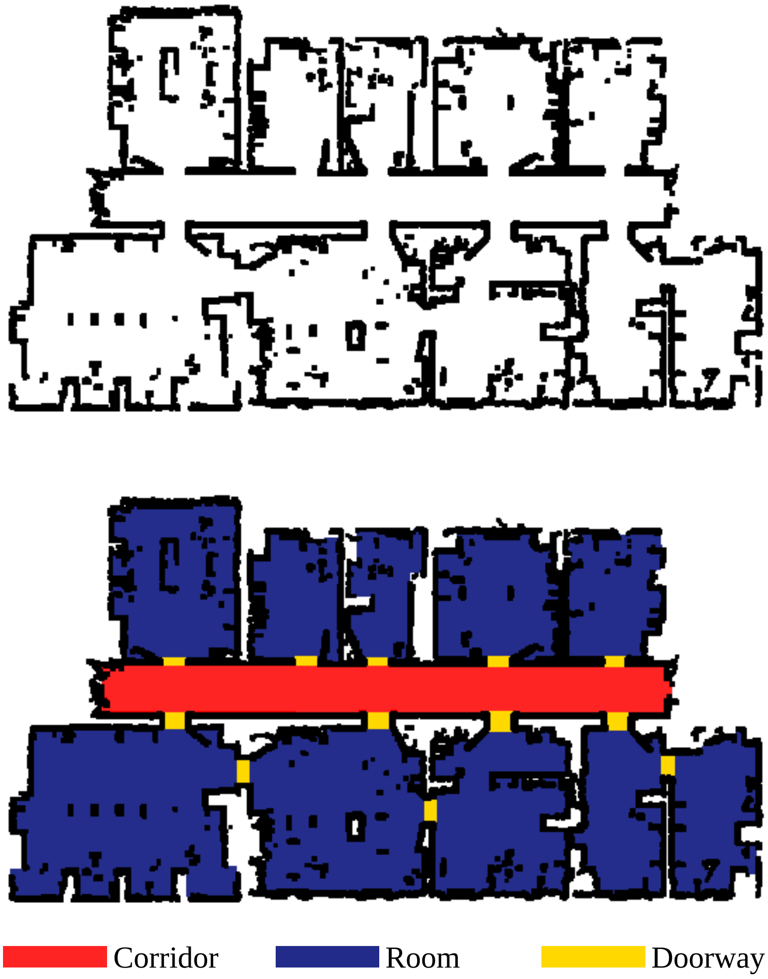


Figure 1.1: The top image shows the occupancy grid map corresponding to the ground floor of the building 52 at the University of Freiburg. Some natural divisions can be extracted from this environment, as for example rooms, doorways, and a corridor. This is shown in the bottom image.

rooms. Furthermore, doorways indicate a transition between two different rooms, or between a room and a corridor. Therefore, semantic labels not only represent places but also possible affordances.

One of the main questions that arrises when doing semantic labeling is how a mobile robot can recognize the different places of an indoor environment. This question, at the same time, leads us to the problem of finding the adequate characteristics or features of each place that can be used to recognize it.

The work presented in this thesis presents different solutions to the previous problems. We introduce novel methods for the semantic labeling of places in indoor environments using autonomous mobile robots. Some of these techniques allow us, at the same time, to select the features that can be used to best characterize each place.

The first contribution of this work is a supervised method based on AdaBoost to assign semantic labels to the different places in indoor environments using a mobile robot. The main idea is to classify each pose of a mobile robot into one of the semantic classes according to the laser range observation the robot gathered at that pose. The classification is carried out using geometrical features extracted from the laser beams. Additionally, the boosting approach allow us to determine which are the most informative features used to recognize each place.

The above method is additionally extended for the extraction of topological maps from indoor environments. The key idea is to apply the semantic classification to all possible poses of the robot in a map, obtaining a complete classification of the free spaces. Neighboring poses with similar classification are then grouped into regions, which form the different nodes in the final a topological map. Previous to the grouping, a filtered method is applied that takes into account spatial dependencies between different labels.

The previous two methods cover the semantic classification of the different poses of a mobile robot, but they do not take into account the movement of the robot along a trajectory. When operating in indoor environments, the robots usually have a moderate velocity and a relatively continuous movement. That means, that observations obtained by a mobile robot at nearby poses are typically very similar. Based on this assumptions, we present a method that takes into account previous classifications when classifying a new pose of a mobile robot along a trajectory using hidden Markov models. The approach also includes information from other sensors rather than lasers. In particular, we use information of objects extracted from images to extend the classification of places. In this way, we are able to increase the number of labels to places such as kitchen, laboratory, offices, and seminar room.

The semantic labeling can be applied not only to improve the human-robot communication, but also to better carry out some other specific tasks for autonomous mobile robots. A further contribution of this work presents the exploration of environments with a team of robots using place information. We will show how the semantic labeling of places can improve the distribution of the robots during the exploration. The main idea here is that places as corridors are better exploration targets as they lead to other rooms. On a second application, we will see how to accelerate the localization process of a single robot using the classification of the different rooms.

The above mentioned techniques are used to augment the maps in the environments with semantic information. However, the AdaBoost-based classifiers can also be applied to include semantic information in sensor data. In this work, we will show an approach to label each of the beams of a laser scan. The main idea is to assign each beam the class of the object it hits. In this work, we restrict the classification to the labels *person* and *non-person*, although the method can be easily extended to use additional labels.

Finally, in this thesis we present the sematic labeling of poses as part of a high level conceptual representation of indoor environments, which we call *multi-layer conceptual map*. This representation extends the semantic classification of places adding upper layers which include more complex conceptual terms as, for example, living rooms. The terms not only represent places but also objects, as TV sets or couches, and are used to create a human-friendly dialogue while interacting with people.

The rest of this thesis is organized as follows. Chapter 2 gives an introduction to supervised learning, and presents some of the methods we will use in the posterior chapters. Chapter 3 presents the approach for semantic labeling of places using range data. The extraction of topological maps from indoor environments is described in Chapter 4. In Chapter 5, we present the classification of trajectories using hidden Markov models. The speeding-up of multi-robot exploration, and single robot localization using semantic place information is described in Chapter 6. In Chapter 7, we explain how add semantic information to proximity sensor data. Chapter 8 gives an overview of the multi-layer conceptual model for indoor environments. Finally, in Chapter 9 we present conclusions about the complete work presented in this dissertation.

1.2 Publications

The following is a list of publications derived from the work presented in this thesis:

Journals

- Andrzej Pronobis, Óscar Martínez Mozos, and Barbara Caputo. Semantic Place Classification Integrating Multiple Sensory Modalities. *Robotics and Autonomous Systems*. 2008. Under review.

- Hendrik Zender, Óscar Martínez Mozos, Patric Jensfelt, Geert-Jan M. Kruijff and Wolfram Burgard. Conceptual Spatial Representations for Indoor Mobile Robots. *Robotics and Autonomous Systems*. 2008. Accepted for publication.

- Cyrill Stachniss, Óscar Martínez Mozos, and Wolfram Burgard. Efficient Exploration of Unknown Indoor Environments using a Team of Mobile Robots. *Annals of Mathematics and Artificial Intelligence*. 2008. Accepted for publication.

- Óscar Martínez Mozos, Rudolph Triebel, Patric Jensfelt, Axel Rottmann, and Wolfram Burgard. Supervised semantic labeling of places using information extracted from sensor data. *Robotics and Autonomous Systems*. Volume 55, issue 5, pp. 391-402. May, 2007.

- Cyrill Stachniss, Giorgio Grisetti, Óscar Martínez Mozos, and Wolfram Burgard. Efficiently Learning Metric and Topological Maps with Autonomous Service Robots. *it - Information Technology*. Volume 49, issue 4, pp. 232-237. 2007.

Book Chapters

- Rudolph Triebel, Óscar Martínez Mozos, and Wolfram Burgard. Relational Learning in Mobile Robotics: An Application to Semantic Labeling of Objects in 2D and 3D Environment Maps. *Studies in Classification, Data Analysis, and Knowledge Organization*. Springer, 2007. Accepted for publication.

- Óscar Martínez Mozos, Cyrill Stachniss, Axel Rottmann and Wolfram Burgard. Using AdaBoost for Place Labeling and Topological Map Building. *Robotics Research: Results of the 12th International Symposium ISRR*. S. Thrun, R. Brooks, H. Durrant-Whyte (Eds.), Springer Tracts in Advanced Robotics, Vol 28, pp 453-472, 2007.

Conferences

- Andrzej Pronobis, Óscar Martínez Mozos, and Barbara Caputo. SVM-based Discriminative Accumulation Scheme for Place Recognition. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena, California, USA. May, 2008. Accepted.

- Hendrik Zender, Patric Jensfeld, Óscar Martínez Mozos, Geert-Jan M. Kruijff, and Wolfram Burgard. An Integrated Robotic Systen for Spatial Understanding and Situated Interaction in Indoor Environments. *In Proceedings of the Conference on Artificial Intelligence (AAAI)*, pp 1584-1589. Vancouver, British Columbia. July, 2007.

- Kai O. Arras, Óscar Martínez Mozos, and Wolfram Burgard. Using Boosted Features for the Detection of People in 2D Range Data. *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. pp. 3402-3407. Rome, Italy. April, 2007.

- Rudolph Triebel, Richard Schmidt, Óscar Martínez Mozos, and Wolfram Burgard. Instace-based AMN Classification for Improved Object Recognition in 2D and 3D Laser Range Data. *In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. pp. 2225-2230. Hyderabad, India. January, 2007.

- Óscar Martínez Mozos and Wolfram Burgard. Supervised Learning of Topological Maps using Semantic Information Extracted from Range Data. *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2772-2777. Beijing, China. October, 2006.

- Cyrill Stachniss, Óscar Martínez Mozos, and Wolfram Burgard. Speeding-Up Multi-Robot Exploration by Considering Semantic Place Information. *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1692-1697. Orlando, FL, USA, May, 2006.

- Axel Rottmann, Óscar Martínez Mozos, Cyrill Stachniss, and Wolfram Burgard. Semantic Place Classification of Indoor Environments with Mobile Robots using Boosting. *In Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 1306-1311. Pittsburgh, PA, USA. July, 2005.

- Óscar Martínez Mozos, Cyrill Stachniss, and Wolfram Burgard. Supervised Learning of Places from Range Data using AdaBoost. *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1742-1747. Barcelona, Spain. April, 2005. **Finalist best student paper**.

Workshops and Symposia

- Óscar Martínez Mozos, Patric Jensfeld, Hendrik Zender, Geert-Jan M. Kruijff, and Wolfram Burgard. An Integrated System for Conceptual Spatial Representations of Indoor Environments for Mobile Robots. *Proceedings of the IEEE/RSJ IROS 2007 Workshop: From Sensors to Human Spatial Concepts (IROS)*. pp 25-32. San Diego, CA, USA. November, 2007.

- Óscar Martínez Mozos, Axel Rottmann, Rudolph Triebel, Patric Jensfelt, and Wolfram Burgard. Semantic Labeling of Places using Information Extracted from Laser and Vision Sensor Data. *In Proceedings of the IEEE/RSJ IROS 2006 Workshop: From Sensors to Human Spatial Concepts (IROS)*, pp. 1742-1747. Beijing, China. October, 2006.

- Cyrill Stachniss, Óscar Martínez Mozos, Axel Rottmann, and Wolfram Burgard. Semantic Labeling of Places. *In Proceedings of the International Symposium of Robotics Research (ISRR)*. San Francisco, CA, USA, 2005.

1.3 Collaborations

Parts of this thesis have been done in collaboration with other people. I would like to thank them for the excellent work done.

The simulation of rear beams to cover a complete field of view in the classification of lasers was done together with Patric Jensfelt.

The application of instance-based associative Markov networks to the classification of maps was a collaboration with Rudolph Triebel.

The approach for the classification of trajectories using additional vision information was developed in collaboration with Axel Rottmann while supervising his master's thesis.

The approaches for speeding-up multirobot exploration and for localization using semantic information about places was a collaboration with Cyrill Stachniss.

The method for assigning semantic information to range data was a join work with Kai O. Arras.

Finally, the multi-layer conceptual map was originated from the research work on the Explorer Scenario in the CoSy project [CoSy, 2004]. In particular, the work here presented was done together with Hendrik Zender, Patric Jensfelt, and Geert-Jan M. Kruijff.

Chapter 2

Supervised Learning

In this chapter we give a short introduction to supervised learning and to the probably approximately correct (PAC) framework. We then present two methods for classification. The first method, called AdaBoost, is based on boosting. The second one is an implementation of a decision list. Both approaches will be often used in this thesis.

2.1 Introduction

In supervised learning we are interested in finding a function that best classifies a set of given examples into a set of classes. This function will be used later to classify new examples, which in general are different from the given ones. This process is also called a *learning task* [Mitchell, 1997] and is described as follows.

There is some space of possible instances X over which different target concepts or classes C can be defined. Each target concept $c \in C$ corresponds to some subset of X. Alternatively, a function can be used which maps each instance x to some concept c. An example of such function is the boolean-valued function defined as $c : X \rightarrow \{0, 1\}$. Here, if one instance $x \in X$ is a positive example of c, then we will write $c(x) = 1$; and $c(x) = 0$ if x is a negative example. Additionally, each instance $x \in X$ is described by a set of attributes or features. We will call the set of features the *feature vector* $f \in F$, where F is the set of all possible feature vectors.

We additionally assume that different instances in X may be encountered with different frequencies. A convenient way to model this is to assume that there is some unknown probability distribution $\mathcal{D}$ that defines the probability of encountering each instance in X. Notice that $\mathcal{D}$ says nothing about whether x is a positive or negative example of some concept. It only determines the probability that x will be encountered.

The set X of all possible instances may be large or even infinite, and obtaining a training set with all possible instances may be impossible for the majority set of problems. For this reason, a subset $S \subset X$ is usually used during the learning process. The set S must be sufficiently representative of the whole set of instances

X. Some guidelines for obtaining an adequate training set are given in [Witten and Frank, 2000].

Given a set S of training examples corresponding to the target concept c, we need to apply a learning algorithm L, which will select the hypothesis h which best approximates the target concept c. Let H be set of all possible hypotheses that the learner L may consider regarding the identity of the target concept c. Similarly to the case of target concepts, we can represent each hypothesis h in H as a boolean-valued function defined over X: $h : X \to \{0, 1\}$. The goal of the learner is then to find a hypothesis h such that $h(x) = c(x)$ for all $x \in X$. On one hand, the learner L must learn from a training set S, which is a subset of X. But on the other hand, the target concept c is defined over the whole set X. That means, that the output $h(x)$ generated by the learner must always be considered as an approximation of the target concept $c(x)$. It may be possible that $h(x) = c(x)\ \forall x \in X$, but we can only be sure of that when $S = X$. Finally, the hypotheses output by the learner will be somehow used to create a classifier to classify new instances in X.

In summary, the goal of the learner L is to find a hypothesis h that best approximates the target concept c using a set of training examples S. The question here is, wether the hypothesis h will be also a good approximation when using a set of unseen examples. An answer to this question is given by the *inductive learning* assumption. Informally, any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples.

Alternatively, we can face this question using the error in the classification for the output hypothesis h. For that, we can try to estimate the error in the classification of the selected hypothesis h with respect to new examples providing the error of h in the training set. These two errors are formally defined as follows [Mitchell, 1997]

True error The true error $\mathrm{error}_{\mathcal{D}}(h)$ of hypothesis $h \in H$ with respect to target concept $c \in C$ and distribution $\mathcal{D}$ is the probability that h will misclassify an instance $x \in X$ drawn at random according to $\mathcal{D}$

$$\mathrm{error}_{\mathcal{D}}(h) = \Pr_{x \in \mathcal{D}} [c(x) \neq h(x)] \, . \tag{2.1}$$

Training error The training error $\mathrm{error}_S(h)$, also called sample error, of hypothesis h with respect to target concept $c \in C$ and training set S is the fraction of misclassified instances $x \in S$

$$\mathrm{error}_S(x) = \frac{1}{|S|} \sum_{x \in S} \delta(c(x), h(x)) \, , \tag{2.2}$$

where

$$\delta(c(x), h(x)) = \begin{cases} 1 & \text{if } c(x) \neq h(x) \\ 0 & \text{otherwise} \, . \end{cases} \tag{2.3}$$

Different statistical approximations can be used to calculate the true error provided the training error as shown in [Mitchell, 1997] and [Witten and Frank, 2000].

Another problem when selecting a hypothesis h is its *overfitting*. A hypothesis $h \in H$ overfits the training examples S if some other hypothesis h' that fits the training examples less well actually performs better over the entire set X, including instances beyond the training set. In his book, Mitchell [1997] defines overfitting as following

Overfitting Given a hypotheses space H, a hypothesis h is said to overfit the training set $S \subset X$ if there exists some alternative hypothesis $h' \in H$, such that h has smaller error than h' on S, but h' has a smaller error than h over the entire set X of instances.

Different methods can be used to avoid overfitting as, for example, use a separate validation set of examples to modify the learned hypothesis. For a further discussion on this topic we refer the reader to [Mitchell, 1997] and [Witten and Frank, 2000].

2.2 PAC Learning

Some other important questions related to some specific learning task are:

- What is the number of training examples needed to assure that the hypothesis output by the learner has a low true error?
- Can this number be bounded in some way?
- Can we bound the time the learner needs to output a hypothesis?

These questions can be answered for certain sets of problems using the probably approximately correct (PAC) framework [Valiant, 1984]. The idea of the PAC learnability is to characterize classes of target concepts that can be reliably learned from a polynomial number of randomly drawn training examples, and a polynomial amount of computation. PAC learnability is defined formally as follows

PAC learnability Consider a concept class C over a set of instances X of length n, and a learner L using hypothesis space H. C is *PAC learnable* by L using H, if for all $c \in C$, distribution D over X, ϵ such that $0 < \epsilon < 1/2$, and δ such that $0 < \delta < 1/2$, the learner L will, with probability at least $(1 - \delta)$, output a hypothesis $h \in H$ such as that $error_D(h) \leq \epsilon$, in time polynomial in $1/\epsilon$, $1/\delta$, n, and $size(c)$.

Here n is the size of the feature vector f representing each instance $x \in X$. The second space parameter, $size(c)$, is the encoding length of the target c.

The previous definition of PAC learning may appear at first to be concerned only with the computational time, whereas we are usually interested in the number of training examples. However they are closely related. In fact, a typical approach to show that some class C of target concepts is PAC learnable, is to first demostrate that each target concept $c \in C$ can be learned from a polynomial number of training examples and then show that the processing time per example is also polynomial bounded.

A learner L following the previous definition is also known as a *strong PAC learning algorithm*. On the other hand, a *weak PAC learning algorithm* is defined analogously except that it is only required to satisfy the condition $\epsilon \geq 1/2 - \gamma$, where $\gamma > 0$ is either a constant, or decreases as $1/p$, where p is a polynomial in the relevant parameters. A hypothesis learned from a weak PAC learning algorithm is called a *weak hypothesis*. This term is also used for a hypothesis which performs just slightly better than a random guessing, as we will se in the next section.

Other various extensions and generalizations of the basic PAC concept are given by by Anthony and Bartlett [1999], Haussler [1992], and Kearns and Vazirani [1994].

2.3 Boosting

Boosting is a general method which attempts to improve the accuracy of a given learning algorithm [Freund and Schapire, 1999; Meir and Rätsch, 2003; Schapire, 2001]. This approach has its roots in the PAC framework (see previous Section 2.2).

Kearns and Valiant [1988; 1994] were the first to pose the question of whether a weak learning algorithm, which performs just slightly better than a random guessing in the PAC model, can be combined into an arbitrarily accurate strong learning algorithm. Later, Schapire [1990] demonstrated that any weak learning algorithm can be efficiently transformed or boosted into a strong learning algorithm.

The underlying idea of boosting is to combine a set of weak hypotheses $\{h_1, h_2, \ldots, h_T\}$ to form a strong hypothesis h_S such that the performance of the strong hypothesis is better than the peformance of each of the single weak hypothesis h_t. Formally

$$h_S(x) = \sum_{t=1}^{T} w_t h_t(x) \,, \tag{2.4}$$

here w_t denotes the weight of hypothesis h_t. Both w_t and the hypothesis h_t are to be learned within the boosting procedure. The resulting strong hypothesis h_S has the form of a weighted majority vote classifier.

Formally, boosting proceeds as follows: the boosting algorithm is provided with a set of labeled training examples $(x_1, y_1), ..., (x_N, y_N)$, where y_i is the label associated with instance x_i. On each round $t = 1, \ldots, T$, the boosting algorithm devises a distribution D_t over the set of examples, and requests (from an unspecified oracle) a weak hypothesis h_t with low error ϵ_t with respect to D_t, where

$\epsilon_t = \Pr[h_t(x_i) \neq y_i]$. Thus, the distribution D_t specifies the relative importance of each example for the current round. After T rounds, the booster must combine the weak hypotheses into a strong one. The intuitive idea is to alter the distribution over the training examples in a way that increases the probability of the harder elements, thus forcing the weak learner to generate new hypotheses that make less mistakes on these elements.

An important aspect related to boosting is overffiting. Large part of the early literature explain that boosting would not overfit even when using a large number of rounds. However, simulations by Grove and Schuurmans [1998] and Rätsch *et al.* [2001] show that data sets with higher noise content could clearly show overfitting effects.

2.3.1 AdaBoost

The AdaBoost algorithm, introduced by Freund and Schapire [1995], is one of the most popular boosting algorithms. Following the general idea of boosting, the AdaBoost algorithm takes as an input a training set of examples $(x_1, y_1), ..., (x_N, y_N)$, where each x_i belongs to some domain space X, and each label y_i pertains to the label set Y. In the case of a binary classification we have $\mid Y \mid = 2$. On each round $t = 1, \ldots, T$, AdaBoost calls a weak learning algorithm (weak learner) repeatedly to select a weak hypothesis. This weak learner takes as input the training set of examples and outputs a weak hypothesis for their classification.

Unlike previous boosting algorithms [Freund, 1993; 1990] and [Schapire, 1990], the AdaBoost algorithm needs no prior knowledge of the accuracies of the weak hypotheses. Rather, it adapts to these accuracies and generates a weighted majority hypothesis in which the weight of each weak hypothesis is a function of its accuracy.

The complete algorithm, presented by Freund and Schapire [1995] is described in Figure 2.1. In this algorithm, the distribution D indicates the importance of the examples at the begining of the training process and later it is controlled by the learner. This distribution can be set initially as the uniform distribution so that $D_1(i) = 1/N$, meaning that all examples have the same importance at the begining. On each round t, the algorithm maintains a set of weights $D_t(i), \ldots, D_t(N)$ over the training examples and computes a distribution p_t by normalizing these weights. The distribution p_t is fed to the weak learner which generates a hypothesis h_t that, hopefully, has a small error with respect to the distribution p_t. The accuray of the weak hypothesis h_t is measured by its error as

$$\epsilon = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right] = \sum_{i:h_t(x_i) \neq y_i} p_t(i)\,. \tag{2.5}$$

Notice that the error is measured with respect to the distribution p_t on which the weak learner was trained. In practice, the weak learning algorithm may be able to use the weights p_t on the training examples. Alternatively, when this is not possible, a subset of the training examples can be sampled according to p_t, and these resampled examples can be used to train the weak learner.

Using the new hypothesis h_t, the boosting algorithm generates the next weight vector D_{t+1}, and the process is repeated. After T iterations, the final strong hypothesis h_S is generated. The hypothesis h_S combines the outputs of the T weak hypotheses using a weighted majority vote.

Freund and Schapire [1995] proved that, for binary classification problems, the training error of the final hypothesis h_S generated by the AdaBoost algorithm is bounded by

$$\epsilon \leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)} \leq \exp\left(-2\sum_{t=1}^{T}\gamma^2\right), \tag{2.6}$$

where $\epsilon_t = 1/2 - \gamma_t$ is the error of the tth weak hypothesis. Since a hypothesis that makes an entirely random guess has error 1/2, γ_t measures the accuracy of the weak hypothesis h_t relative to random guessing. This bound shows that the final training error drops exponentially if each of the weak hypotheses is better than a random guess.

2.3.2 Generalized AdaBoost

An alernative version to the original AdaBoost algorithm was introduced by Schapire and Singer [1999]. This version, called generalized AdaBoost, presents several improvements with respect the previous one. First, the weak hypothesis can have a range over all $\mathbb{R}$ rather than only two values. Second, in this version of the algorithm the different α_t, which correspond to the weights of the final weak hypotheses, are left unspecified. The complete algorithm is shown in Figure 2.2.

Schapire and Singer [1999] also give a possible choice for the diferent weights α_t as follows

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1+r_t}{1-r_t}\right), \tag{2.7}$$

where r_t is chosen at each iteration so that its absolute value $|r_t|$ is maximum according to

$$r_t = \sum_{i=1}^{N} D_t(i) y_i h_t(x_i)\,. \tag{2.8}$$

In their work, Schapire and Singer [1999] present different versions of the generalized AdaBoost algorithm together with some experimental comparison.

```
• Input:
    set of $N$ labeled examples $(x_1, y_1), \ldots, (x_N, y_N)$,
      where $y_i = 1$ if the example $x_i$ is positive,
      and $y_i = 0$ if the example $x_i$ is negative;
    distribution $D$ over the $N$ examples;
    weak learner;
    integer $T$ specifying number of iterations.

• Initialize the weight vector $w_1(i) = D_1(i)$ for $i = 1, \ldots, N$.

• For $t = 1, \ldots, T$

    1. Set
         $$p_t(i) = \frac{w_t(i)}{\sum_{j=1}^{N} w_t(j)}$$

    2. Train weak learner, providing it with the distribution $p_t$ and get back a
       hypothesis $h_t : X \to [0, 1]$.

    3. Calculate the error of $h_t$
         $$\epsilon_t = \sum_{i=1}^{N} p_t(i) \, |h_t(x_i) - y_i|$$

    4. Set
         $$\beta_t = \frac{\epsilon_t}{(1 - \epsilon_t)}$$

    5. Set the new weights:
         $$w_{t+1}(i) = w_t(i) \beta_t^{1-|h_t(x_i)-y_i|}$$

• The final strong hypothesis is given by:
    $$h_S(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \log \frac{1}{\beta_t} \\ 0 & \text{otherwise} \end{cases}$$
```

Figure 2.1: The AdaBoost algorithm.

- Input:
  set of N labeled examples $(x_1, y_1), \ldots, (x_N, y_N)$,
    where $y_i = 1$ if the example x_i is positive,
    and $y_i = -1$ if the example x_i is negative;
  weak learner;
  integer T specifying number of iterations.

- Initialize the weights $D_1(i) = 1/N$ for $i = 1, \ldots, N$.

- For $t = 1, \ldots, T$

  1. Train the weak learner using distribution D_t and get back a hypothesis $h_t : X \to \mathbb{R}$.
  2. Choose $\alpha_t \in \mathbb{R}$.
  3. Update:
  $$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$
     where Z_t is a normalization factor so that D_{t+1} is a distribution

- The final strong hypothesis is given by:
$$h_S(x) = \operatorname{sign}\left(\sum_{i=1}^{T} \alpha_t h_t(x) \right).$$

Figure 2.2: A generalized version of the AdaBoost algorithm.

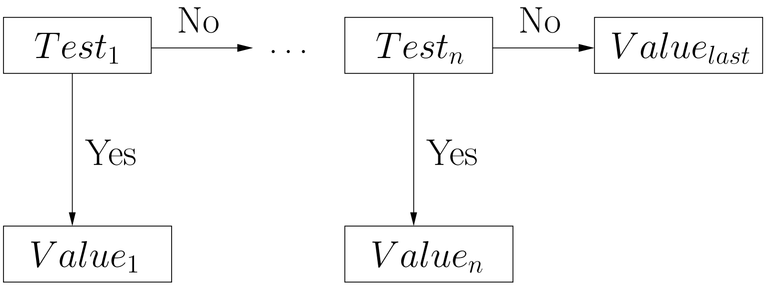


Figure 2.3: Structure of a decision list.

2.4 Decision Lists

A decision list is an alternative way of constructing a classifier. It is based on logical expressions of a restricted form, and consists of a series of tests, each of which is a conjunction of literals [Russell and Norvig, 2003]. If a test succeeds when applied to an example description, then the decision list specifies the value to be returned. If the test fails, processing continues with the next test in the list. Figure 2.4 shows the structure of a decision list.

In this learning task we want to find a consistent decision list with the training data. This can be done using the algorithm of Figure 2.4. This procedure repeatedly finds a test that will agree exactly with some subset of the training set S. Once such a test is found, it adds it to the decision list outputting the value corresponding to the class assigned to the subset. At the same time, the examples that correspond to the test are removed from the training set. This process is repeated until there are no more examples. The process shown in Figure 2.4 does not specify the method to select the next test to add to the list. This selection will depend on the specific task and on the implementation of the different tests. Some examples of such selection for some concrete tasks are shown by Russell and Norvig [2003], Viola and Jones [2001], and Rottmann *et al.* [2005].

```
• Input:
    set $S$ of training labeled examples,
• Do until $S$ is empty
    1. Select a test $Test$ that matches a nonempty subset $S_{Test}$ of $S$ such that
       the examples of the subset are positive with respect to $Test$.
    2. If the there is no such $Test$ then return $failure$.
    3. Output the value $Value$ corresponding to the label of the examples in
       $S_{Test}$.
    4. Add $Test$ and $Value$ to the decision list.
    5. Update $S = S - S_{Test}$.
• Return the output value $Value_{last}$.
```

Figure 2.4: Algorithm to learn the different tests and their positions inside the list.

Chapter 3

Semantic Learning of Places from Range Data

3.1 Introduction

As we explained in the introductory section, building accurate maps of indoor environments is one of the typical problems in mobile robotics. In this task, a mobile robot moves along a trajectory while gathering information with sensors. This information is then used to construct the map, and for navigation and localization tasks. However, little work have been done in the area of semantic mapping.

In this chapter, we address the problem of assigning semantic labels to locations of the environment using a mobile robot. Indoor environments, like the one depicted in the top image of Figure 1.1, can typically be decomposed into areas with different functionalities, such as office rooms, corridors, hallways, or doorways. Generally, each of these places has a different structure. For example, the bounding box of a corridor is usually longer than that of rooms and hallways. Furthermore, rooms are typically smaller than hallways, and also are more cluttered than corridors or hallways. As an example, Figure 1.1 (bottom) showed a typical hand-labeled division of the environment into three possible categories of places.

The key idea presented in this chapter is to classify the position of the robot based on the current scan obtained from the range sensor. Figure 3.1 shows an example of a range scan taken by a mobile robot in a corridor. Other examples for typical range scans obtained in an office environment are shown in Figure 3.2.

The approach presented in this chapter uses the AdaBoost algorithm to boost simple geometrical features from the scans into a strong classifier. Each of this features alone is insufficient for a reliable categorization of places. The features are represented by a numerical value computed from the beams of a laser range scan as well as from a polygon representation of the covered area. Since AdaBoost provides only binary decisions, we determine the decision list with the best sequence of binary classifiers. Experimental results shown in this chapter illustrate that the resulting classification system can determine the type of the place with

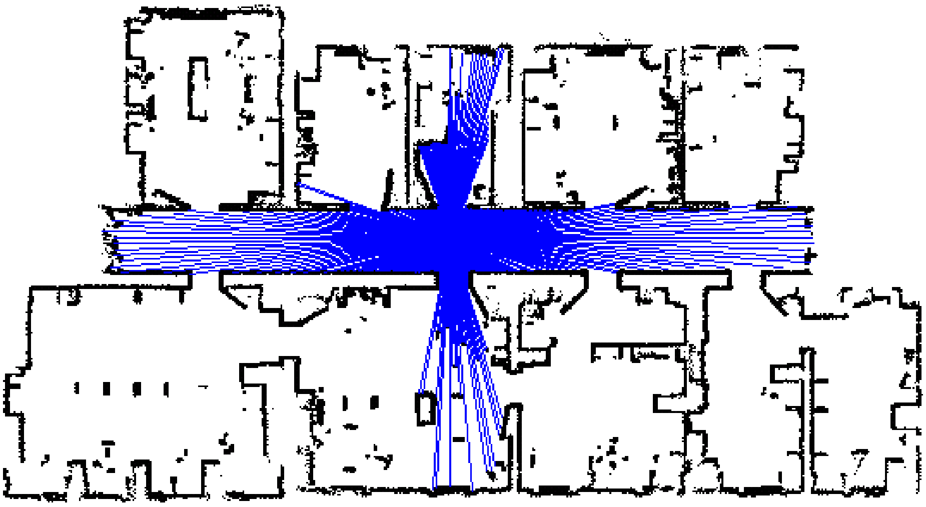


Figure 3.1: Range scan obtained by a mobile robot in a corridor. The image also shows the complete map of the environment where the scan was taken. The scan covers the complete 360° field of view of the robot.

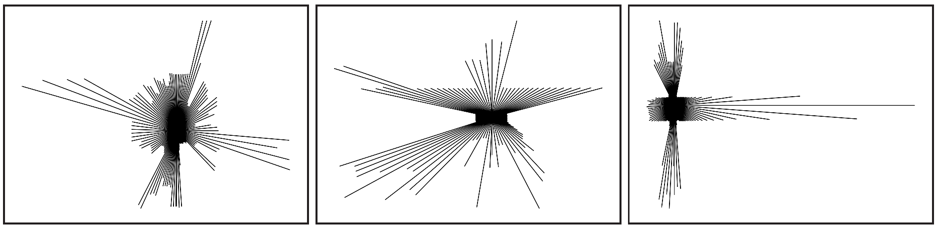


Figure 3.2: Example scans recorded in a room, a doorway, and a corridor.

high classification rates. Moreover, results are presented illustrating that the resulting classifier can even be used in environments from which no training data was available.

Throughout this chapter we assume that the robot is equipped with a laser range scanner that covers 360° field of view around the robot. However, common configurations on real mobile robots have only a laser covering 180° in front of the robot. We also present a solution for these cases.

The rest of the chapter is organized as follows. In the next section we describe the particular implementation of the AdaBoost algorithm for place labeling. Section 3.3 presents its extension to multiple classes. Section 3.4 introduces the features extracted from laser range scans. A solution to the problem of restricted field of view is given in Section 3.5. In Section 3.6, experimental results are presented. We discuss related work in Section 3.7. Finally, we conclude in Section 3.8.

3.2 Binary Classification Using AdaBoost

The AdaBoost algorithm, introduced by Freund and Schapire [1995], is one of the most popular boosting algorithms. Following the general idea of boosting, the AdaBoost algorithm takes as an input a training set of examples $(x_1, y_1), ..., (x_N, y_N)$, where each x_i belongs to some domain space X, and each label y_i pertains to the label set Y. In the case of binary classification we have $\mid Y \mid = 2$. On each round $t = 1, \ldots, T$, AdaBoost calls a weak learning algorithm, called *weak learner*, repeatedly to select a weak hypothesis. This weak learner takes as input the training set of examples and outputs a weak hypothesis for their classification.

The key idea of the algorithm is to maintain a weight distribution D over the training examples. The distribution D indicates the importance of the examples at the beginning of the training process and later it is controlled by the learner. This distribution can be set initially as the uniform distribution so that $D_1(i) = 1/N$, meaning that all examples have the same importance at the beginning. Some other initializations have provided good results, as the one presented by Viola and Jones [2001].

Throughout this work we apply the generalized version of AdaBoost [Schapire and Singer, 1999], which have several advantages and gives us de possibility of calculating a confidence value for the final classification (see below). More details about this algorithm are given in Section 2.3.

The representation of the weak classifiers is done following the ideas presented by Viola and Jones [2001]. This implementation restricts the weak classifiers to depend only on single-valued features f_j. Thus, each weak classifier has the form

$$h_j(x) = \begin{cases} +1 & \text{if } p_j f_j(x) < p_j \theta_j \\ -1 & \text{otherwise}\,, \end{cases} \tag{3.1}$$

where θ_j is a threshold and p_j is either -1 or 1 and represents the direction of the inequality. The algorithm determines for each weak classifier $h_j(x)$ the optimal values for θ_j and p_j, such that the number of misclassified training examples is minimized

$$(p_j, \theta_j) \quad = \quad \underset{(p_i, \theta_i)}{\operatorname{argmin}} \sum_{n=1}^{N} |h_i(x_n) - y_n| \;. \tag{3.2}$$

The final generalized AdaBoost algorithm modified for the concrete task of place labeling given in Figure 3.3.

Using the generalized version of the AdaBoost algorithm shown in Figure 3.3, and following the method suggested by Friedman *et al.* [2000], we can additionally compute a confidence value $C^+ \in [0, 1]$ for a positive binary classification of a new example as

$$C^+ = P(y = +1 \mid x) = \frac{e^{F(x)}}{e^{-F(x)} + e^{F(x)}}\,, \tag{3.3}$$

- Input:
  set of N labeled examples $(x_1, y_1), \ldots, (x_N, y_N)$,
  where $y_i = 1$ if the example x_i is positive,
  and $y_i = -1$ if the example x_i is negative;
  integer T specifying number of iterations.

- Initialize weights $D_1(n) = \frac{1}{2l}$ for positive examples, and $D_1(n) = \frac{1}{2m}$ for negative examples, where l is the number of positive examples and m the number of negative ones.

- For $t = 1, \ldots, T$

  1. Normalize the weights $D_t(n)$
     $$D_t(n) = \frac{D_t(n)}{\sum_{i=1}^{N} D_t(i)}.$$
  2. For each feature f_j train a weak classifier h_j using the distribution D_t.
  3. For each classifier h_j calculate
     $$r_j = \sum_i D_t(i) y_i h_j(x_i),$$
     where $h_j(x_n) \in \{-1, +1\}$.
  4. Choose the classifier h_j that maximizes $|r_j|$ and set $(h_t, r_t) = (h_j, r_j)$.
  5. Update the weights
     $$D_{t+1}(n) = D_t(n) \exp(-\alpha_t y_n h_t(x_n)),$$
     where $\alpha_t = \frac{1}{2} \log(\frac{1+r_t}{1-r_t})$.

- The final strong classifier is given by
  $$h_S(x) = \operatorname{sign}(F(x)),$$
  where
  $$F(x) = \sum_{t=1}^{T} \alpha_t h_t(x).$$

Figure 3.3: The generalized version of the AdaBoost algorithm for place labeling using laser-based features.

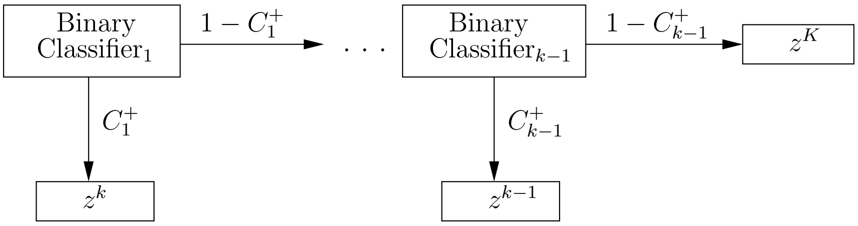


Figure 3.4: A decision list classifier for k classes using binary classifiers. The output of each binary classifier z^k contains the probability that the classified example belongs to the k-th class

where $F(x)$ is the output of the algorithm according to Figure 3.3. If the example is classified as negative, a positive confidence value can be calculated as

$$C^+ = P(y = -1 \mid x) = 1.0 - C^- \tag{3.4}$$

with

$$C^- = P(y = -1 \mid x) = \frac{e^{-F(x)}}{e^{-F(x)} + e^{F(x)}} . \tag{3.5}$$

3.3 Multi-class Classification Using AdaBoost

The previous AdaBoost algorithm was designed for binary classification problems. To label places in the environment, however, we need the ability to handle multiple classes. A way to construct a multi-class classifier is to arrange several binary classifiers into a decision list (Section 2.4). Each element of such a list represents one binary classifier which determines if an example belongs to one specific class. In addition, ech binary classifier outputs a confidence value C_k^+ for a positive classification of its class k. Figure 3.4 illustrates the structure of the probabilistic decision list.

In the decision list, each test example is fed into the first binary classifier, which outputs a confidence value C^+ for a positive classification. Then the example is passed to the next binary classifier. This process is repeated until the last element in the list. The complete output of the decision list is represented by an histogram z. In this histogram, the k-th bin stores the probability that the classified location belongs to the k-th class according to the sequence of classifiers in the decision list (Figure 3.4). Let C_k refer to the confidence value of the k-th binary classifier in our decision list. Then, the probability that the location to be classified belongs to the k-th class is given by the k-th bin of the histogram z computed as

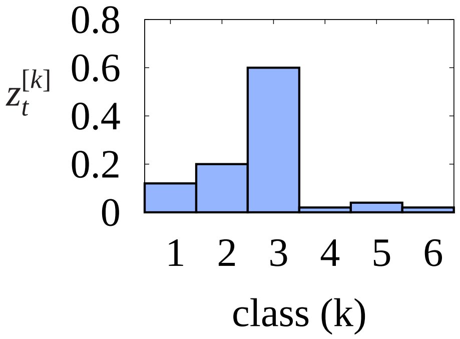


Figure 3.5: An example of a classification output for the decision list of Figure 3.4 using six classes.

$$z^{(k)} = C_k^+ \prod_{j=1}^{k-1} (1 - C_j^+), \tag{3.6}$$

whereas for the confidence value C_K of the last bin (the K-th bin) holds $C_K = 1$ according to the structure of the decision list in Figure 3.4. An example of a histogram for six classes is illustrated in Figure 3.5

One important question in the context of a sequential classifier is the order in which the individual binary classifiers are arranged. This order can have a major influence on the overall classification performance, because the individual classifiers typically are not error-free and classify with different accuracies.

In general, the problem of finding the optimal order of binary classifiers that minimizes the classification error is NP-hard. In this chapter, however, we typically are confronted with a small number of classes and we can easily enumerate all potential permutations to determine the optimal sequence. Since the first element of such a sequential classifier processes more data than subsequent elements, it is typically a good strategy to order the classifiers in increasing order according to their training error rate. Compared to the optimal order, the classifier generated by this heuristic for an application with several classes performed on average only 1.3% worse as demonstrated by Rottmann [2005]. In several situations, the sequence generated by this heuristic turned out to be the optimal one.

3.4 Simple Features from Sensor Range Data

In the previous section we described the key principles of the AdaBoost algorithm for boosting simple features to strong classifiers. It remains to describe the features of the range scans used in the system. We assume that the mobile robot is equipped with a 360^{o} field of view range sensor. Each observation $z = \{b_0, ..., b_{M-1}\}$ contains a set of beams b_i. Each beam b_i consists of a tuple (α_i, d_i) where α_i is the angle of the beam relative to the robot and d_i is the length of the beam.

Each training example for the AdaBoost algorithm consists of one observation z and its classification y. Thus, the set of training examples is given by

$$E = \{(z_i, y_i) \mid y_i \in Y = \{\text{Room}, \text{Corridor}, \ldots\}\} , \tag{3.7}$$

where Y is the set of classes corresponding to the places we want to recognize. Throughout this chapter we assume that the classification of the training examples is given in advance. In practice this can be achieved by manually labeling places in the map or by instructing the robot while it is exploring its environment. The goal is to learn a classifier that is able to generalize from these training examples and that can classify so far unseen places in this environment or even new ones.

As already mentioned, the method for place classification is based on simple geometrical features extracted from the range scans. We call them *simple* because they are single-valued features. All features are rotational invariant to make the classification of a pose dependent only on the (x, y)-position of the robot and not of its orientation. Most of the features are standard geometrical features often used in shape analysis [Gonzalez and Wintz, 1987; Haralick and Shapiro, 1992; Loncaric, 1998; O'Rourke, 1998; Russ, 1992].

We define a feature f as a function that takes as argument one observation and returns a real value: $f : Z \rightarrow \mathbb{R}$, where Z is the set of all possible observations.

Two sets of simple features are calculated for each observation. The first set $\mathcal{B}$ is calculated using the raw beams in z. The following is a list of the single-valued features pertaining to this set:

1. The average difference between the length of consecutive beams.
2. The standard deviation of the difference between the length of consecutive beams.
3. Same as feature #1, but considering different max-range values.
4. Same as feature #2, but considering different max-range values.
5. The average beam length.
6. The standard deviation of the beam length.
7. Number of gaps in the scan. Two consecutive beams build a gap if their difference is greater than a given threshold. Different features are used for different threshold values.
8. Number of beams lying on lines that are extracted from the range scan [Sack and Burgard, 2004].
9. Euclidean distance between the two points corresponding to two consecutive global minima.

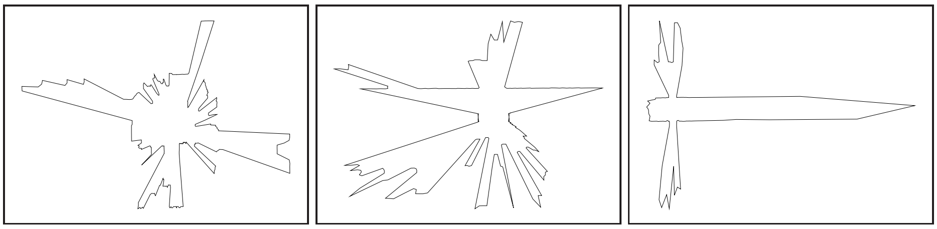


Figure 3.6: Example of polygon approximations of scans recorded in a room, a doorway, and a corridor.

10. The angular distance between the two points corresponding to two consecutive global minima.
11. Average of the relation between the length of two consecutive beams.
12. Standard deviation of the relation between the length of two consecutive beams.
13. Average of normalized beam length.
14. Standard deviation of normalized beam length.
15. Number of relative gaps.
16. Kurtosis.

The second set $\mathcal{P}$ of features is calculated from a polygonal approximation $P(z)$ of the area covered by z. The vertices of the closed polygon $P(z)$ correspond to the coordinates of the end-points of each beam b_i of z relative to the robot:

$$P(z) \quad = \quad \{(d_i \cos \alpha_i, d_i \sin \alpha_i) \mid i = 0, \ldots, M - 1\} \tag{3.8}$$

As an example, the polygonal representations of the laser range scans depicted in Figure 3.2 are shown in Figure 3.6. The list of features corresponding to the set $\mathcal{P}$ is as following:

1. Area of $P(z)$.
2. Perimeter of $P(z)$.
3. Area of $P(z)$ divided by Perimeter of $P(z)$.
4. Mean distance between the centroid and the shape boundary.
5. Standard deviation of the distances between the centroid and the shape boundary.

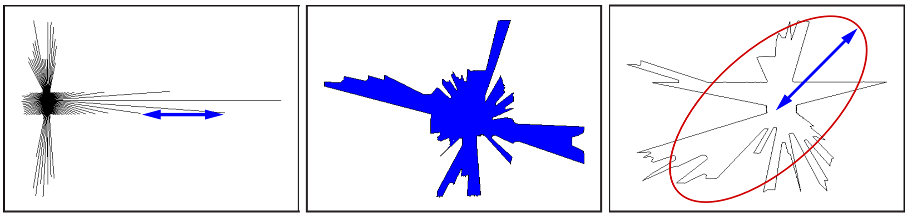


Figure 3.7: Examples of features generated from laser data, namely the average distance between two consecutive beams, the perimeter of the area covered by a scan, and the mayor axis of the ellipse that approximates the polygon described by the scan. The laser beams cover a 360^o field of view.

6. Similarity invariant descriptors based in the Fourier transformation. We use the first 200 descriptors.

7. Major axis Ma of the ellipse that approximates $P(z)$ using the first two Fourier coefficients.

8. Minor axis Mi of the ellipse that approximate $P(z)$ using the first two Fourier coefficients.

9. Ma/Mi.

10. Seven invariants calculated from the central moments of $P(z)$ [Gonzalez and Wintz, 1987].

11. Normalized feature of compactness of $P(z)$.

12. Normalized feature of eccentricity of $P(z)$.

13. Form factor of $P(z)$.

14. Circularity of $P(z)$.

15. Normalized circularity of $P(z)$.

16. Average normalized distance between the centroid and the shape boundary.

17. Standard deviation of the normalized distance between the centroid and the shape boundary.

Figure 3.7 shows graphically some of the features corresponding to the $\mathcal{B}$ and $\mathcal{P}$ sets. In particular, the features are the average distance between two consecutive beams (feature #1 in set $\mathcal{B}$), the area covered by a scan (feature #3 in set $\mathcal{P}$), and the mayor axis of the ellipse that approximates the polygon described by the scan (feature #7 in set $\mathcal{P}$).

The complete list of features, together with their mathematical definition can be found in Appendix A.

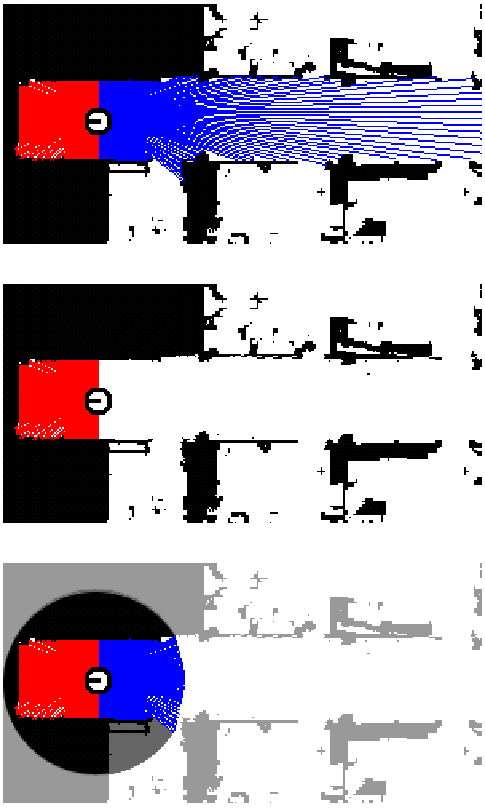


Figure 3.8: The top image shows a laser observation covering 360° field of view. In the middle image the range observation covers only 180° in front of the robot. In the bottom image, the rear beams are simulated using the occupancy information contained inside the shaded area (local map).

3.5 Feature Extraction with Restricted Field of View[1]

As mentioned in the previous section, the simple features are based on laser observations covering 360° field of view (top image in Figure 3.8). However, common configurations on real mobile robots have only a laser covering 180° in front of the robot (middle image in Figure 3.8). In these last cases we propose to maintain a local map around the robot when classifying a pose of the robot during a trajectory. This local map can be updated during the movements of the robot, and then used to simulate the rear laser beams (bottom image in Figure 3.8).

In our case, we maintain a sparse local map around the robot. This map contains the endpoints of the previous laser beams that hit some object around the robot. We simulate the rear beams using this sparse map. For each simulated beam that does not hit any object in the sparse map, we calculate its value using an interpolation between the values of their (known) neighboring beams at both sides.

[1] The work presented in this section originated from a collaboration with Patric Jensfelt.

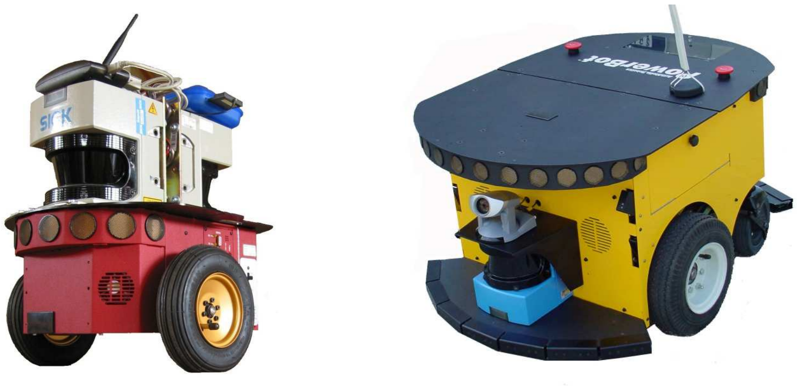


Figure 3.9: On the left, an ActivMedia Pioneer 2-DX8 equipped with two SICK laser range finders. On the right, a PowerBot robot equipped with a front laser.

3.6 Experimental Results

The complete approach described in this chapter has been implemented and tested on a real robot as well as in simulation using the Carnegie Mellon Robot Navigation Toolkit (CARMEN) [Montemerlo *et al.*, 2003]. The robots used to carry out the experiments were an ActivMedia Pioneer 2-DX8 equipped with two SICK laser range finders, and a PowerBot robot equipped only with a front laser. Both robots are shown in Figure 3.9. The goal of the experiments is to demonstrate that the simple features can be boosted to a robust classifier of places. Additionally, we analyze whether the resulting classifier can be used to classify places in environments for which no training data were available. We first describe the results obtained with the sequential version of AdaBoost. In the next experiment we analyze how well a mobile robot can utilize the resulting classifier. Furthermore, we present an experiment illustrating that a classifier can be applied to robustly classify places in a completely new environment. Finally, we present results comparing the method presented in this chapter with previous approaches.

One important parameter of the AdaBoost algorithm is the number of weak classifiers T used to form the final strong classifier. All in all we formulated more than 300 simple features, each of them with one free parameter, which is determined in the learning phase according to Equation (3.2). AdaBoost even uses features multiple times with different parameters. Thus, much more than the initial sets of simple features are available to form the strong classifier. We performed several experiments with different numbers of weak classifiers and analyzed the classification error. Throughout the experiments, we found that 100 weak classifiers provide the best trade-off between the error rate of the classifier and the computational cost of the algorithm. Therefore we used this value in all the experiments presented in this chapter.

3.6.1 Results Using a Decision List

The first experiment was performed using data from the office environment in building 79 at the University of Freiburg. This environment contains three different types of places, namely rooms, doorways, and a corridor. For the sake of clarity we give a result obtained by separating the environment into two parts. The left half of the environment contains the training examples, and the right half of the environment was then used as a test set, as shown in the top image of Figure 3.10.

In this experiment we used a probabilistic sequential classifier as shown in Section 3.3. In this particular case, each binary classifier identifies one place of the environment, i.e. room, door or corridor. Because we only have three classes, we tried all the possible combinations of binary classifiers for the decision list. We used the sequential classifier corridor-room which gives the best results and correctly classifies 97.27% of the test examples. The classification results are also depicted as colored/grey-shaded areas in the lower image of Figure 3.10. This illustrates that the approach presented in this chapter is well-suited to classify places according to a single laser range scan.

A similar experiment was carried out in the office environment of building 52 at the University of Freiburg (see Figure 3.11). In this case, the classification result in the test data was of 97%.

3.6.2 Transferring the Classifiers to New Environments

The next experiment is designed to analyze whether a classifier learned in a particular environment can be used to successfully classify the places of a new environment. To carry out the experiment we trained a decision list classifier using the training examples of the maps corresponding to buildings 79 and 52 at the University of Freiburg (Figure 3.10 and Figure 3.11), with different scales. In this way, we obtained a classifier with a better generalization. The resulting classifier was then evaluated on scans simulated in the map denoted as *SDR site B* in the Radish repository [Howard and Roy, 2003] (top image in Figure 4.5). This map represents an empty building in Virginia, USA. We use the sequence corridor-doorway which results in a classification rate of 92.36%, as can be seen in the bootom image of Figure 4.5.

3.6.3 Classification of Trajectories Using Sensors with Restricted Field of View

In this experiment we show the results of applying the previous classification methods when the laser range scan has a restricted field of view. We first steered a PowerBot robot equipped with only a front laser along the 6th floor of the CAS building at KTH (right to left). The trajectory is shown in the top image of Fig. 3.13. The data recorded in this floor was used to train the AdaBoost classifier. We then classified a trajectory on the 7th floor in the same building. We started the trajectory in an opposite direction (left to right). The rear beams were simulated using

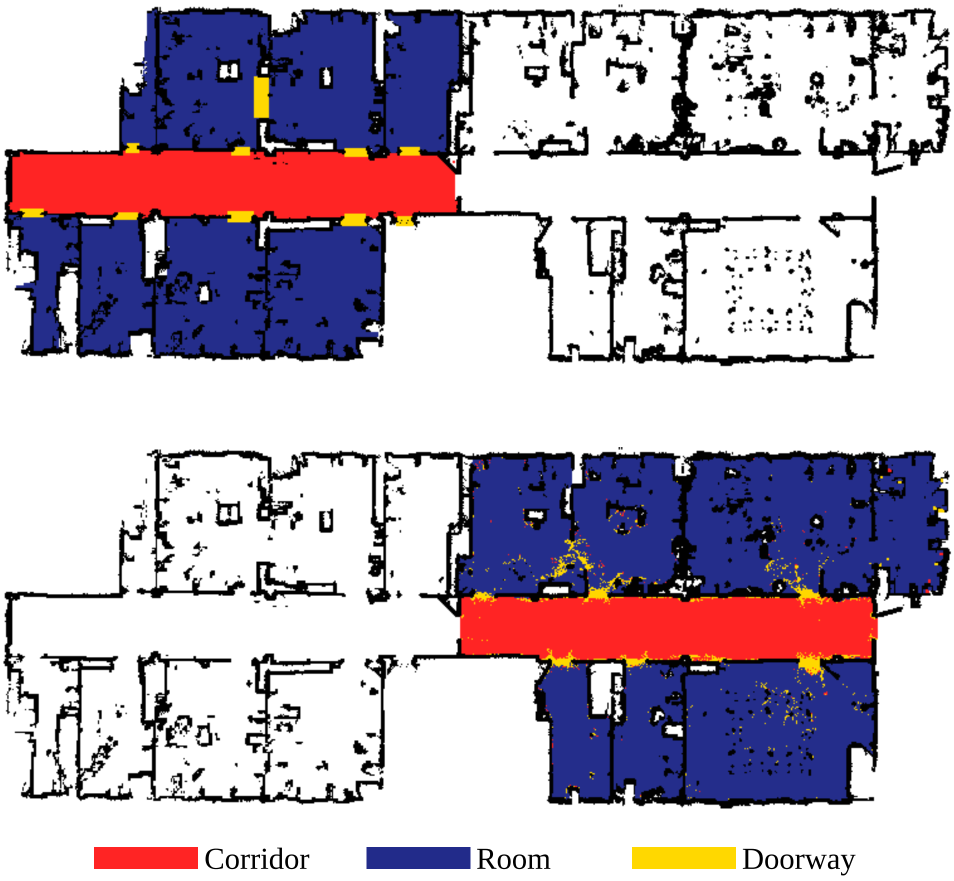


Figure 3.10: The top image shows the training of the building 79 used to train the classifiers. The lower image shows the classified test data using the sequential AdaBoost algorithm.

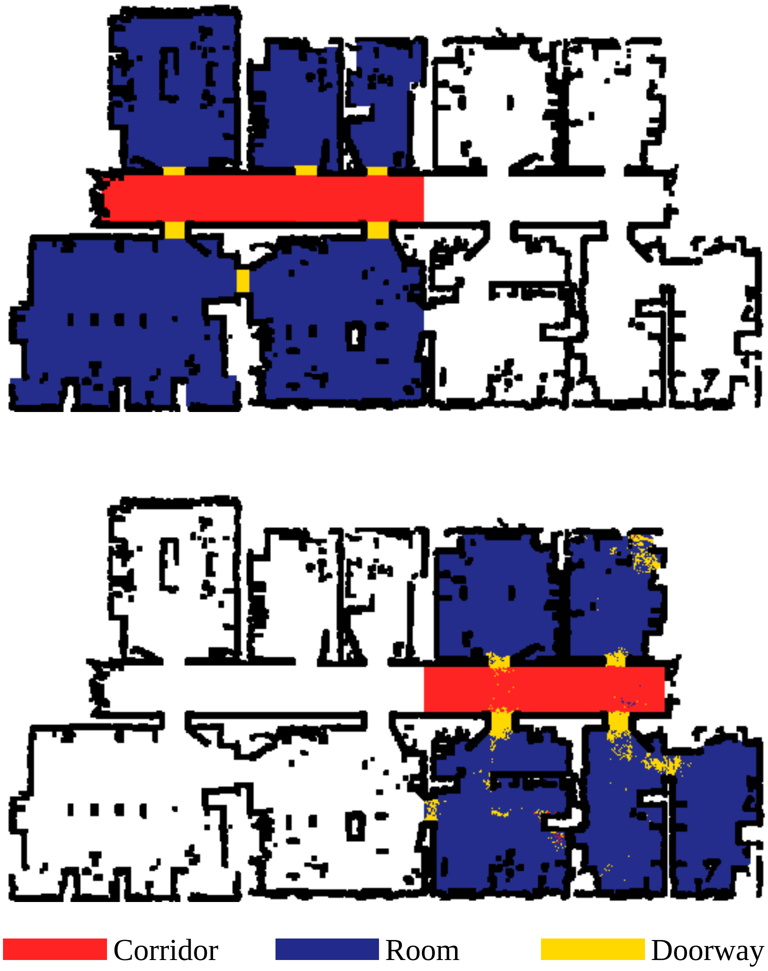


Figure 3.11: The top image shows the training of the building 52 used to train the classifiers. The lower image shows the classified test data using the sequential AdaBoost algorithm.

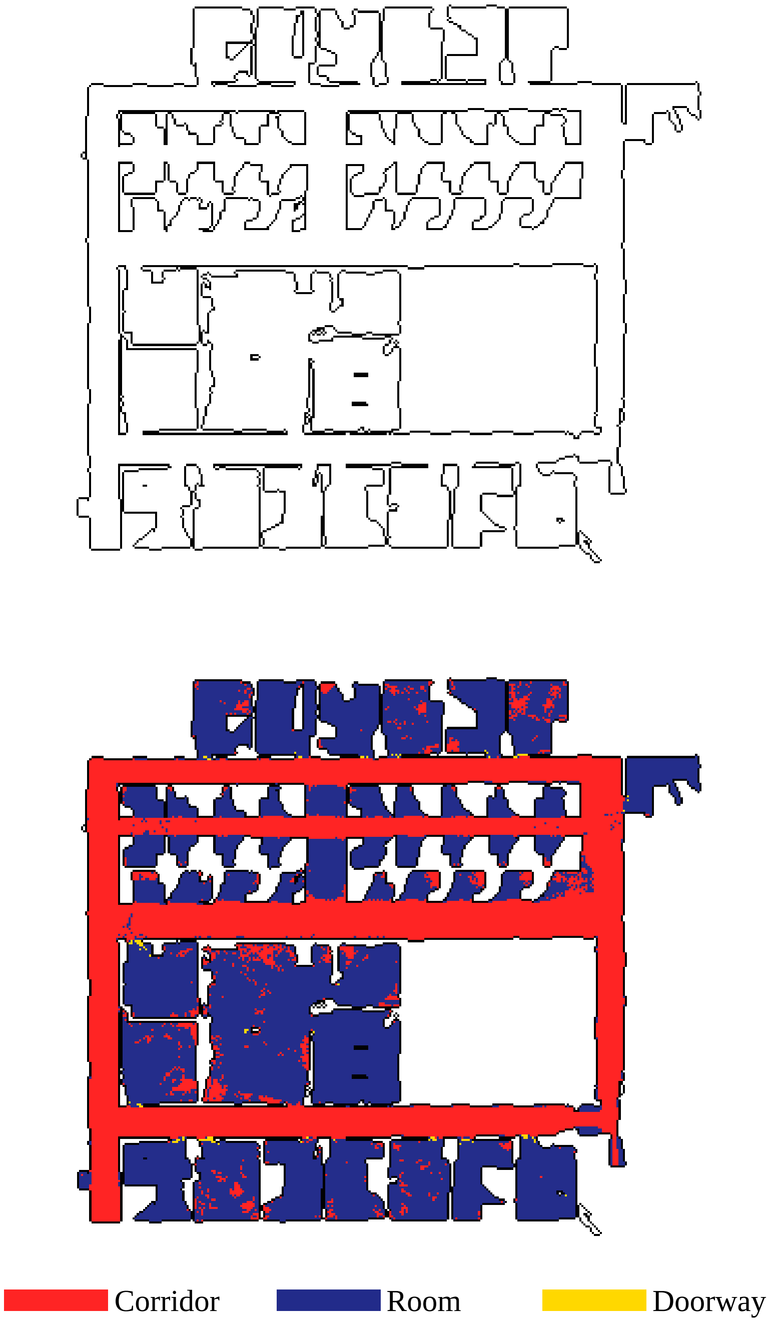


Figure 3.12: The top image shows the map of the building denoted as *SDR site B* in the Radish repository. The lower image shows the resulting classification.

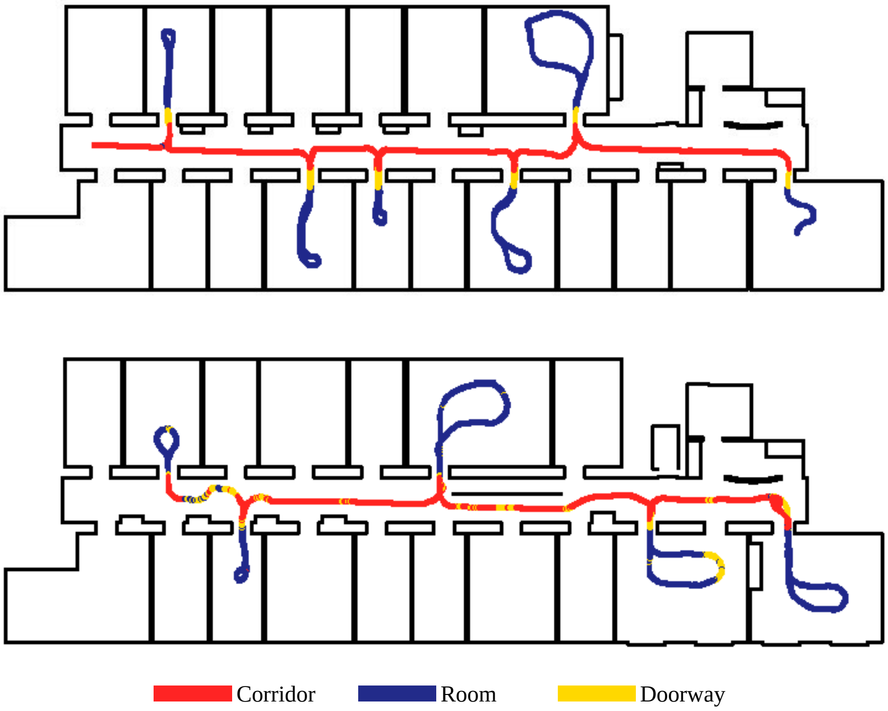


Figure 3.13: The top image shows the training trajectory on the 6th floor of the CAS building at KTH. The bottom image depicts the labeling of the trajectory of the 7th floor using only a front laser with a classification rate of 84.4%. The map shown is for informative purposes only and does not represent exactly the environment in which the experiments were carried out, as furniture is missing.

a local map as explained in Section 3.5. The resulting classification rate of 84.4% is depicted in the bottom image of Fig. 3.13. As the results indicate, restricting the field of view decreases the classification rate. However, the classification maintains at acceptable levels.

3.6.4 Comparison of Feature Sets

We compare now the feature set described in Section 3.4 with the one proposed in my previous master's thesis [Mozos, 2004]. For this purpose, we trained a AdaBoost-based decision list for each of the feature sets using the training set shown in Figure 3.10. The different sequential classifiers were then applied to the test set depicted in Figure 3.10. The obtained classification results are shown in Table 3.1. As can be seen, the new extended feature-set provides better results in all of the experiments.

Decision List	New feature set [%]	Original feature set [%]
room-door	96.94	93.94
room-corridor	97.26	93.31
corridor-room	97.27	93.16
corridor-door	87.73	80.10
door-corridor	87.21	80.10
door-room	86.60	80.49

Table 3.1: Classification results of the new improved feature set compared to the one in [Mozos, 2004].

binary classifier	seven best features
corridor	$\mathcal{P}$.7, $\mathcal{B}$.15, $\mathcal{P}$.6, $\mathcal{B}$.2, $\mathcal{B}$.7, $\mathcal{B}$.3
room	$\mathcal{P}$.5, $\mathcal{P}$.12, $\mathcal{B}$.7, $\mathcal{P}$.7, $\mathcal{B}$.1, $\mathcal{B}$.12
doorway	$\mathcal{B}$.15, $\mathcal{P}$.17, $\mathcal{B}$.11, $\mathcal{P}$.6, $\mathcal{B}$.3, $\mathcal{B}$.9

Table 3.2: The best six features for each binary classifier.

3.6.5 Selected Weak Features

Finally, we analyzed the importance of the individual weak features in the final strong classifier. To carry out the experiment, we trained a decision list classifier using the training examples of the maps shown in Figure 3.10 and Figure 3.11 with different scales. This classifier was transferred to classify the map of Figure 4.5 (Section 3.6.2).

Table 3.2 lists the six best features for each binary classifier with the leftmost feature the most important. In this table an entry $\mathcal{B}.i$ represents the i-th feature for raw beams in z, whereas an entry $\mathcal{P}.j$ represents the j-th feature of the polygon $P(z)$. Both sets of features were described in Section 3.4.

Analyzing the table we can see that feature $\mathcal{P}$.7 is the most important to recognize the corridor. This feature represents the major axis of an ellipsis approximation of the laser observation. This indicates that more elongated ellipses corresponds to corridors. For recognizing room places, we can see that the feature $\mathcal{P}$. 5 is the most critical. This feature represents the standard deviation of the distance from the centroid to the shape boundary of the observation. The feature has a smaller value for rooms because they have a more regular perimeter. The second feature for rooms, $\mathcal{P}$.12, confirms this fact.

Doorways are more complicated to describe. For these places the most important feature is the number of relative gaps, $\mathcal{B}$.15, which indicates the number of jumps from one beam to its neighbor. This is another way of detecting big changes in the beam lengths corresponding to the doorframes (middle image in Figure 3.6). Another feature designed to detect doorframes is $\mathcal{B}$.9, which is also selected as one of the six more importants for door detection.

Comparing the Table 3.2 with previous results in [Mozos, 2004], we can see that several of the new features are selected as more important. This fact confirms the usefulness of the new feature sets presented in this chapter.

3.7 Related Work

In the past, several authors considered the problem of adding semantic information to places. Buschka and Saffiotti [2002] describe a virtual sensor that is able to identify rooms from range data. Also Koenig and Simmons [1998] use a pre-programmed routine to detect doorways from range data. Additionally, Althaus and Christensen [2003] use line features to detect corridors and doorways.

Some authors also apply learning techniques to localize the robot or to identify distinctive states in the environment. For example, Oore *et al.* [1997] train a neural network to estimate the location of a mobile robot in its environment using the odometry information and ultrasound data. Torralba *et al.* [2003] use Hidden Markov Models for learning places from image data. Compared to these approaches, the algorithm presented in this chapter does not require any pre-defined routines for extracting high-level features. Instead, it uses the AdaBoost algorithm to boost simple features to strong classifiers for place categorization.

Additionally, Kuipers [2000] detect distinctive states in the map that are used as places in a topological graph. However, these states does not contain semantic information. Finally, Wolf and Sukhatme [2006] present some automated techniques for classifying, modeling and ultimately understanding the usage of space in a typical urban outdoor environments.

Boosting has been used to identify objects using different features. Maybe one of the most famous applications of AdaBoost is the fast recognition of faces in images by Viola and Jones [2001]. Also Treptow *et al.* [2003] use the AdaBoost algorithm to track a ball without color information in the context of RoboCup.

Some of the ideas in this chapter are similar to the work by Viola and Jones [2001]. For example, we use a similar AdaBoost algorithm and also create simple features for the classification. However, the problem to solve is totally different, since we classify locations in indoor environments using 2D range data.

The work from this chapter is an extension of my master's thesis [Mozos, 2004]. However, in [Mozos, 2004] only a discrete classification was possible. In contrast, the work presented in this chapter extends the classification methods adding confidence values to the output of the classifiers. The output of the decision list is now represented by a histogram which allows us to use the labeling in further probabilistic methods. Moreover, the set of geometrical features is extended, improving the classification of the different places.

Some ideas presented in this chapter have been used in several posterior works about semantic place recognition. For instance, Friedman *et al.* [2007] present an approach for the classification of places using Voronoi random fields. This work also uses simple features that are selected using boosting as characteristics for the nodes in a Markov random field. The paper by Pronobis *et al.* [2006] shows an approach to classify the different places of an indoor environment using vision. An extension to this work has been recently introduced by Pronobis *et al.* [2008], in which the classification of places is done using an additional laser sensor together with the set of features presented in Section 3.4. Moreover, Topp and Christensen [2006] use a similar idea of describing regions with simple geometrical features extracted from laser readings. The work by Brunskill *et al.* [2007] presents an online method for generating topological maps from raw sensor information based on spectral clustering. Here, the laser observations are represented by the set of features presented in this chapter, and submap recognition uses the boosting method similar to the one of Section 3.2. Finally, Sousa *et al.* [2007] apply the same set of features for classifying places in indoor environments. Instead of AdaBoost, they use a support vector machines as a classifier. In their work, only laser readings with a 180^o field of view in front of the laser are classified.

Other works use the semantic labeling of places presented in this thesis as base for other high level tasks. For example, the approach by Kersting *et al.* [2007] shows that the semantic classification of places presented in this chapter can be used to learn navigation policies using relational Markov decision processes.

The idea of classifying laser range readings into categories using boosted simple features has also been applied in tasks other than place recognition. Arras *et al.* [2007] apply similar methods as the ones presented in this chapter but using a different set of features for the laser scans. The idea here is to classify each beam as hitting a person or not. This work is presented in detail in Chapter 7.

Finally, boosting simple features is also used in the work by Persson *et al.* [2007] to create a virtual sensor for the semantic classification of images in outdoor.

3.8 Conclusions

In this chapter we presented an approach to classify different places in the environment into semantic classes, like rooms, corridors, and doorways. The described technique uses simple geometric features extracted from a single laser range scans and applies the AdaBoost algorithm to form a strong classifier. To distinguish between more than two classes we use a sequence of binary classifiers arranged in a decision list. Experiments carried out on a real robot as well as in simulation illustrate that our technique is well-suited to classify places in different environments even without training the classifier for each environment.

Chapter 4

Topological Map Extraction with Semantic Information

4.1 Introduction

In the previous chapter we have seen how a robot can classify its pose in an indoor environment into a semantic class. The different semantic classes represented typical divisions of the environment, as for example corridors, rooms or doorways. This chapter will show how a robot can extract a topological map from the environment using the previous semantic labeling.

Topological maps have been quite popular in the robotics community because they are believed to be cognitively more adequate, since they can be stored more compactly than geometric maps, and can be also communicated more easily to users of a mobile robot. In the past, many researchers have considered the problem of building topological maps of the environment from the data gathered with a mobile robot. However, few techniques exit that permit to add semantic information to the maps.

In this chapter, we consider the problem of learning topological maps with semantic information from geometric maps that were obtained with a mobile robot in an indoor environment using range data. The approach is based on the assumption that indoor environments, like the one depicted in the left image of Figure 4.1, can be typically decomposed into areas with different functionalities such as rooms, corridors and doorways, and that these areas build the vertices of a topological graph. The connections of the vertices are then given by the neighborhood of the regions in the occupancy map. For example, a doorway is typically connected to two rooms, two corridors, or to a room and a corridor. The right image in Figure 4.1 depicts a possible topological representation for the map in the left image of the same figure.

Throughout this chapter we assume that the robot is given a map of the environment in the form of an occupancy grid. The main idea is to decide about the semantic label of each free cell in the occupancy grid using local and neighbor-

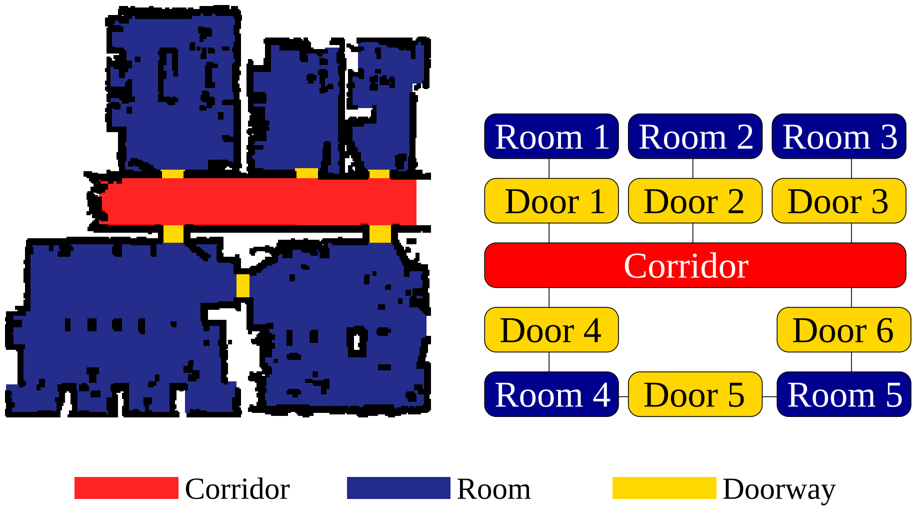


Figure 4.1: The left image shows a geometric map of a typical indoor environment with rooms, doorways, and a corridor, depicted in colors/grey levels. The right image shows the corresponding semantic-topological map.

ing information. By local information we mean the set of geometrical features the robot obtains from a laser observation at a concrete location, as explained in Chapter 3. By neighboring information we refer to the semantic information from the neighboring cells of the location to be classified.

Two different methods are presented which use both local and neighboring information for the final classification. The first approach determines for each unoccupied cell of a grid map its semantic class. This is achieved by simulating a range scan of the robot given it is located in that particular cell, and then classifying this scan into one of the semantic classes. Examples for typical simulated range scans obtained in an office environment were shown in Chapter 3 (Figure 3.2). The classification is then done using a sequence of classifiers learned with the AdaBoost algorithm arranged in a probabilistic decision list as explained in Section 3.3. To remove noise and clutter from the resulting classifications, we apply an approach denoted as probabilistic relaxation labeling. This method corrects the classification at each location taking into account the semantic class of neighboring positions.

The second method for the classification is based on associative Markov networks (AMNs). In this case, the classification of one position is done using simultaneously the local information together with the relation between semantic labels from neighboring positions. We apply an variant of AMNs called instance-based associative Markov networks (iAMNs). This approach combines AMNs with nearest-neighbor techniques.

One we have a final labeling resulting from any of the previous methods, a graph is constructed whose nodes correspond to the regions of identically labeled poses, and whose edges represent the connections between them. Additionally, each node contains geometrical information about the region it represents, like the area, the centroid and the orientation. A typical topological map obtained with this approach is shown in the right image of Figure 4.1. The approaches for topological map building presented in this chapter are off-line methods.

The rest of the chapter is organized as follows. In Section 4.2, the probabilistic relaxation approach is described. Instance-based associative Markov networks are introduced in Section 4.3. Section 4.4 describes the method used to extract semantic regions and to create the final topological map. In Section 4.5, experimental results are presented. We discuss related work in Section 4.6. Finally, we conclude in Section 4.7.

4.2 Probabilistic Relaxation Labeling

The first approach for extracting topological maps determines for each unoccupied cell of the grid its semantic class. This is achieved by simulating a range scan of the robot given it is located at that particular cell, and then labeling this scan into one of the semantic classes using a probabilistic decision list. This approach was already introduced in Section 3.3. This results in an occupancy map with a semantic label in each free cell. However, the final maps usually contain some errors in the classification. To smooth the final classification of each cell, we apply a probabilistic relaxation labeling method introduced by Rosenfeld *et al.* [1976]. This method changes (or maintains) the label of a cell according to the labels of its neighborhood.

The probabilistic relaxation labeling problem is defined as follows. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph consisting of nodes $\mathcal{V} = \{v_1, \ldots, v_N\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Let furthermore $\mathcal{L} = \{l_1, \ldots, l_L\}$ be a set of labels. We assume that every node v_i stores a probability distribution about its label. This distribution is represented by a histogram P_i. Each bin $p_i(l)$ of that histogram stores the probability that the node v_i has the label l. Thus, $\sum_{l=1}^{L} p_i(l) = 1$. For each node v_i, $\mathcal{N}(v_i) \subset \mathcal{V}$ denotes its neighborhood which consists of the nodes $v_j \neq v_i$ that are connected to v_i. Each neighborhood relation is represented by two values. Whereas the first one describes the compatibility between the labels of two nodes, the second one represents the influence between the two nodes. The term $\mathcal{R} = \{r_{ij}(l, l') \mid v_j \in \mathcal{N}(v_i)\}$ defines the compatibility coefficients between the label l of node v_i and the label l' of v_j. Additionally, we define $\mathcal{C} = \{c_{ij} \mid v_j \in \mathcal{N}(v_i)\}$ as the set of weights indicating the influence of node v_j on node v_i.

Given an initial estimation for the probability distribution over labels $p_i^{(0)}(l)$ for the node v_i, the probabilistic relaxation method iteratively computes estimates $p_i^{(r)}(l)$, $r = 1, 2, \ldots$, based on the initial probabilities $p_i^{(0)}(l)$, the compatibility coefficients $\mathcal{R}$, and the weights $\mathcal{C}$, in the form

$$p_i^{(r+1)}(l) = \frac{p_i^{(r)}(l)\left[1 + q_i^{(r)}(l)\right]}{\sum_{l'=1}^{L} p_i^{(r)}(l')\left[1 + q_i^{(r)}(l')\right]}, \tag{4.1}$$

where

$$q_i^{(r)}(l) = \sum_{j=1}^{M} c_{ij} \left[\sum_{l'=1}^{L} r_{ij}(l,l') p_j^{(r)}(l') \right] . \tag{4.2}$$

Note that the compatibility coefficients $r_{ij}(l,l') \in [-1,1]$ do not need to be symmetric. A value $r_{ij}(l,l')$ close to -1 indicates that label l' is unlikely at node v_j when label l occurs at node v_i, whereas values close to 1 indicate the opposite. A value of exactly -1 indicates that the relation is not possible, and a value of exactly 1 means that the relation always occurs.

Probabilistic relaxation provides a framework for smoothing but does not specify how the compatibility coefficients are computed. In this work, we apply the coefficients as defined by Yamamoto [1979]

$$r_{ij}(l,l') = \begin{cases} \frac{1}{1-p_i(l)} \left(1 - \frac{p_i(l)}{p_{ij}(l|l')}\right) & \text{if } p_i(l) < p_{ij}(l \mid l') \\ \frac{p_{ij}(l|l')}{p_i(l)} - 1 & \text{otherwise} , \end{cases} \tag{4.3}$$

where $p_{ij}(l \mid l')$ is the conditional probability that node v_i has label l given that node $v_j \in \mathcal{N}(v_i)$ has label l'. Each of the values $p_i(l)$ and $p_{ij}(l \mid l')$ are pre-calculated only once and remain the same during the iterations of the relaxation process. The coefficients $\mathcal{R}$ remain the same as well.

So far, we have described the general method for relaxation labeling. It remains to describe how we apply this method for spatial smoothing of the classifications obtained by our classifier. To learn a topological map, we assume a given two-dimensional occupancy grid map in which each cell $m_{(x,y)}$ stores the probability that it is occupied. We furthermore consider the eight-connected graph induced by such a grid. Let $v_i = v_{(x,y)}$ be a node corresponding to a cell $m_{(x,y)}$ from the map. Then we define a neighborhood $N_8(v_{(x,y)})$ using the 8-connected cells to $v_{(x,y)}$ as described in [Gonzalez and Wintz, 1987].

For the initial probabilities $p_{(x,y)}^{(0)}(l)$, we use the output z of the classifier as described in Section 3.3. This output is represented by a histogram in which each bin k indicates de probability that the pose belongs to class k. Furthermore, our set of labels $\mathcal{L}$ is composed by the labels corridor, room, doorway, and wall. For each node $v_{(x,y)}$ in the free space of the occupancy grid map, we calculate the expected laser scan by ray-casting in the map. We then classify the observation and obtain a probability distribution z over all the possible places according to Equation (3.6). The classification output z for each pose (x,y) is used to initialize the probability distribution $P_{(x,y)}^{(0)}$ of node $v_{(x,y)}$. For the nodes lying in the free space, the probability $p_{(x,y)}^{(0)}(wall)$ of being a wall is initialized with 0. Accordingly, the nodes corresponding to occupied cells in the map are initialized with $p_{(x,y)}^{(0)}(wall) = 1$.

Each of the weights $c_{ij} \in C$ is initialized with the value $\frac{1}{8}$, indicating that all the eight neighbors v_j of node v_i are equally important. The compatibility coefficients are calculated using Equation (4.3). The values $p_i(l)$ and $p_{ij}(l \mid l')$ are obtained from statistics in the given occupancy grid map corresponding to previously labeled training data.

4.3 Instance-based Associative Markov Networks[1]

The second approach for topological map extraction presented in this chapter is based on associative Markov networks (AMNs). In particular, we use the instance-based associative Markov networks (iAMNs) introduced by [Triebel *et al.*, 2007b]. The idea behind AMNs is to combine the advantage of instance-based nearest-neighbor (NN) classification with the AMN approach to obtain a collective classifier that is not restricted to the linear separability requirement.

4.3.1 Associative Markov Networks

This section gives a short overview on associative Markov networks. A more detailed description about AMNs can be found in Appendix C.

An associative Markov network is an undirected graphical model in which no assumption is made about the direction of the causality between nodes in the graph. We restrict to the case of discrete variables, that is, each variable $Y_i \in Y$ corresponds to a set of K possible labels $Y_i \in \{1, \ldots, K\}$. Thus, we define a Markov random field as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the set of nodes $\mathcal{V}$ represent discrete variables, and the edges $\mathcal{E}$ refer to the relations between them [Taskar, 2004]. An AMN can be divided into a subset of cliques C, where each clique $c \in C$ is associated with a subset $Y_c \in Y$. The nodes in a clique Y_c form a fully connected subgraph.

Each clique is accompanied by a potential $\phi_c(Y_c)$ which associates a non-negative value with each assignment y_c to Y_c. To simplify things, we focus on pairwise associative Markov networks [Taskar, 2004], where all of the cliques involved are either a single node, or a pair of nodes (1-clique or 2-clique). In a pairwise AMN with edges $\mathcal{E} = \{(ij) \,|\, i < j\}$, the nodes and edges are associated with potentials $\phi_i(Y_i)$ and $\phi_{ij}(Y_i, Y_j)$ respectively.

In an AMN, each node Y_i can be assigned a feature vector $x_i \in \mathbb{R}^{d_n}$, which describes the properties of the object represented by that node. Similarly, a feature vector $x_{ij} \in \mathbb{R}^{d_e}$ can be assigned to each edge (ij). The feature vector x_{ij} indicates the properties that describe the relation between the objects represented by the nodes Y_i and Y_j. The node and edges potentials are functions of the feature vectors x_i and x_{ij}. The resulting network defines the distribution

$$\log P_w(y|x) = \sum_{i=1}^{N} \sum_{k=1}^{K} (w_n^k \cdot x_i) y_i^k + \sum_{(ij)\in\mathcal{E}} \sum_{k,l=1}^{K} (w_e^{k,l} \cdot x_{ij}) y_i^k y_j^l - \log Z_w(x) \,. \qquad (4.4)$$

[1] The work presented in this section originated from a collaboration with Rudolph Triebel.

Here the partition function $Z_w(x)$ depends on the parameters w and features x, but no on the labels y.

The main task in an associative Markov network consists on finding the assignment $y \in Y$ that maximizes $\log_w P(y|x)$. This is actually a *maximum a posteriori* (MAP) assignment that can be formulated as an integer linear program (see [Taskar, 2004] for more details).

4.3.2 Feature Vector Transformation

The main drawback of the AMN classifier, which is based on the log-linear model, is that it separates the classes linearly. This assumes that the features are separable by hyper-planes, which is not justified in all applications. This restriction does not hold for instance-based classifiers such as the nearest-neighbor, in which a query data point $\tilde{p}$ is assigned to the label that corresponds to the training data point p whose features x are closest to the features $\tilde{x}$ of $\tilde{p}$. In the learning step, the NN classifier simply stores the entire training data set and does not compute a reduced set of training parameters.

To combine the advantage of instance-based NN classification with the AMN approach, we convert the feature vector $\tilde{x}$ of length L pertaining to query point $\tilde{p}$ using the transform $\tau : \mathbb{R}^L \to \mathbb{R}^K$ given by

$$\tau(\tilde{x}) = (d(\tilde{x}, \hat{x}_1), \ldots, d(\tilde{x}, \hat{x}_K)), \tag{4.5}$$

where K is the number of classes and $\hat{x}_k$ denotes the training example with label k closest to $\tilde{x}$. In this way, the transformed features are more easily separable by hyperplanes. An example is given in Figure 4.2. Here, the top image depicts the training and test data for a two class problem, in which the length of the feature vector $x = (x_1, x_2)$ is two . The classification of the test data (triangles) is shown as lines connecting each training example with the closest example (square) in the ground truth. This nearest neighbor classification results in very few errors. However, it seems difficult to separate the test data into the two classes they pertain using an hyperplane (in this case a line). The bottom image of Figure 4.2 shows the training examples in the transformed space using the transformation given by $\tau(\tilde{x}) = (d(\tilde{x}, \hat{x}_1), d(\tilde{x}, \hat{x}_2))$. In this case, the linear separability is improved.

Additionally, the M nearest neighbors can be used in the transform function. For this, we compute the M nearest distances to each of the classes $k = 1, \ldots, K$. The final transformation $\tau_M : \mathbb{R}^L \to \mathbb{R}^{KM}$ given by

$$\tau_M(\tilde{x}) = (d(\tilde{x}, \hat{x}_1^1), \ldots, d(\tilde{x}, \hat{x}_1^M), \ldots, d(\tilde{x}, \hat{x}_K^1), \ldots, d(\tilde{x}, \hat{x}_K^M)). \tag{4.6}$$

The resulting model, introduced by [Triebel *et al.*, 2007b], is called instance-based associative Markov network (iAMN).

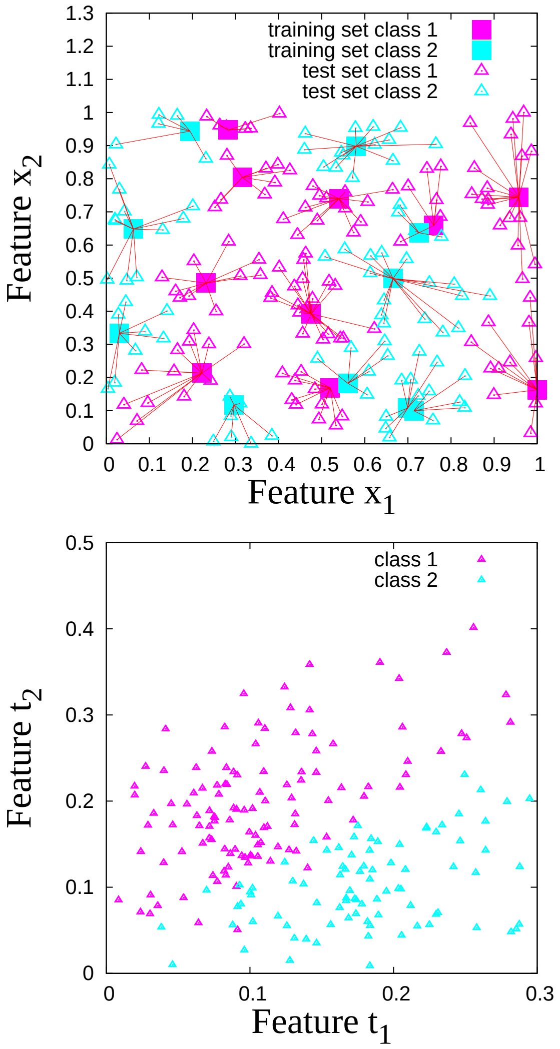


Figure 4.2: Example of the feature transform τ for a two-class problem with two features. The top image shows the training and test data with ground truth labeling. In the bootom image the transformation τ is applied to the test data.

4.3.3 Feature Selection

One of the problems when classifying points represented by range data consists on selecting the size of the feature vectors. As we showed in the experiments of Chapter 3, the number of possible features that can be used to represent each data point is usually very large and can easily be in the order of hundreds. This problem is known as *curse of dimensionality*. There are at least two reasons to try to reduce the size of the feature vector. The most obvious one is the computational complexity, which in our case, is also the most critical, sice we have to learn and inference in networks with thousands of nodes. Another reason is that although some features may carry a good classification when treated separately, maybe there is a little gain when combined together if they have a high mutual correlation [Theodoridis and Koutroumbas, 2006]. The goal thus is to reduce the size of the feature vectors when used with the iAMN and, at the same time, try to maintain their class discriminatory information.

The reduction on the numbers of features used for the classification of places is somehow implicit in the AdaBoost-based classifiers used in the previous Section 4.2. There, the final number of weak classifiers T can be selected. Each selected weak classifier represents a feature together with a threshold (Chapter 3). The problem is that the same feature can appear multiple times with different thresholds and different priorities, which makes it difficult to decide which are the best original features.

In this section we follow an alternative approach. We apply a scalar feature selection procedure which uses a class separability criterion and incorporates correlation information. The selection is independent of the classification algorithm that will use the features (iAMN in our case). This kind of methods are also denoted as *filters*. A filter relies on general characteristics of the data to evaluate and select feature subsets without involving any classification algorithm [Guyon and Elisseeff, 2003].

As separability criterion C, we use the Fisher's discrimination ratio (FDR) extended to the multi-class case [Theodoridis and Koutroumbas, 2006]. For a scalar feature f and K classes $\{w_1, \ldots, w_K\}$, $C(f)$ can be defined as

$$C(f) = FDR_f = \sum_{i}^{K} \sum_{j \neq i}^{K} \frac{(\mu_i - \mu_j)^2}{\sigma_i + \sigma_j}, \tag{4.7}$$

where the subscripts i, j refer to the mean and variance of the classes w_i and w_j respectively. Additionally, the cross-correlation coefficient between any two features f and g given T training examples is defined as

$$\rho_{fg} = \frac{\sum_{t=1}^{T} x_{tf} x_{tg}}{\sqrt{\sum_{t=1}^{T} x_{tf}^2 \sum_{t=1}^{T} x_{tg}^2}}, \tag{4.8}$$

where x_{tf} denotes the value of the feature f in the training example t. Finally, the selection of the best L features involves the following steps

- Select the first feature f_1 as

$$f_1 = \operatorname*{argmax}_{f} C(f).$$

- Select the second feature f_2 as

$$f_2 = \operatorname*{argmax}_{f \neq f_1} \left\{ \alpha_1 C(f) - \alpha_2 |\rho_{f_1 f}| \right\},$$

  where α_1 and α_2 are weighting factors.

- Select f_l, $l = 3, \ldots, L$, such that

$$f_l = \operatorname*{argmax}_{f \neq f_r} \left\{ \alpha_1 C(f) - \frac{\alpha_2}{l-1} \sum_{r=1}^{l} |\rho_{f_r f}| \right\}, \quad r = 1, 2, \ldots, l-1$$

After the scalar feature selection, the learning and inference steps on the instance-based associative Markov network are carried out. More detail about the inference process can be found in Appendix C and in [Triebel *et al.*, 2007a].

4.4 Region Extraction and Topological Mapping

After applying any of the previous approaches for classifying the free cells in an occupancy grid map, we extract the regions from the final classified graph. We define a region λ_l on a adjacency graph $\mathcal{A}$ as a set of 8-connected nodes with the same label l. For each label $l \in \{\text{corridor}, \text{room}, \text{doorway}\}$, regions are extracted from the adjacency graph using the algorithm by Rosenfeld and Pfaltz [1966]. Each region λ_l is assigned a different identifier. The connections between regions are extracted using a similar algorithm [Gonzalez and Wintz, 1987]. Finally, a topological graph $\mathcal{T} = (\mathcal{V}_{\mathcal{T}}, \mathcal{E}_{\mathcal{T}})$ is constructed in which each node $v_i \in \mathcal{V}_{\mathcal{T}}$ represents a region and each edge $e_i \in \mathcal{E}_{\mathcal{T}}$ represents a connection. Additionally, we add to each node v_i information about the properties of the region λ_l which represents: area, centroid, and major and minor axis of the ellipse approximation of λ_l. The major and minor axis are vectors which represent the elongation of the region and its orientation. The topological graph together with the region properties form the final topological map. We finally apply a heuristic region correction to the topological map to increase the classification rate:

1. We mark each region corresponding to a room or a corridor whose size does not exceed a given threshold of 1 m^2 compared to the training set as a classification error and assign the label of one of its connected regions.

2. We mark each region labeled as doorway whose size does not exceed a given threshold of 0.1 m^2 or that is connected to only one region as a false classification and assign the label of one of its connected regions.

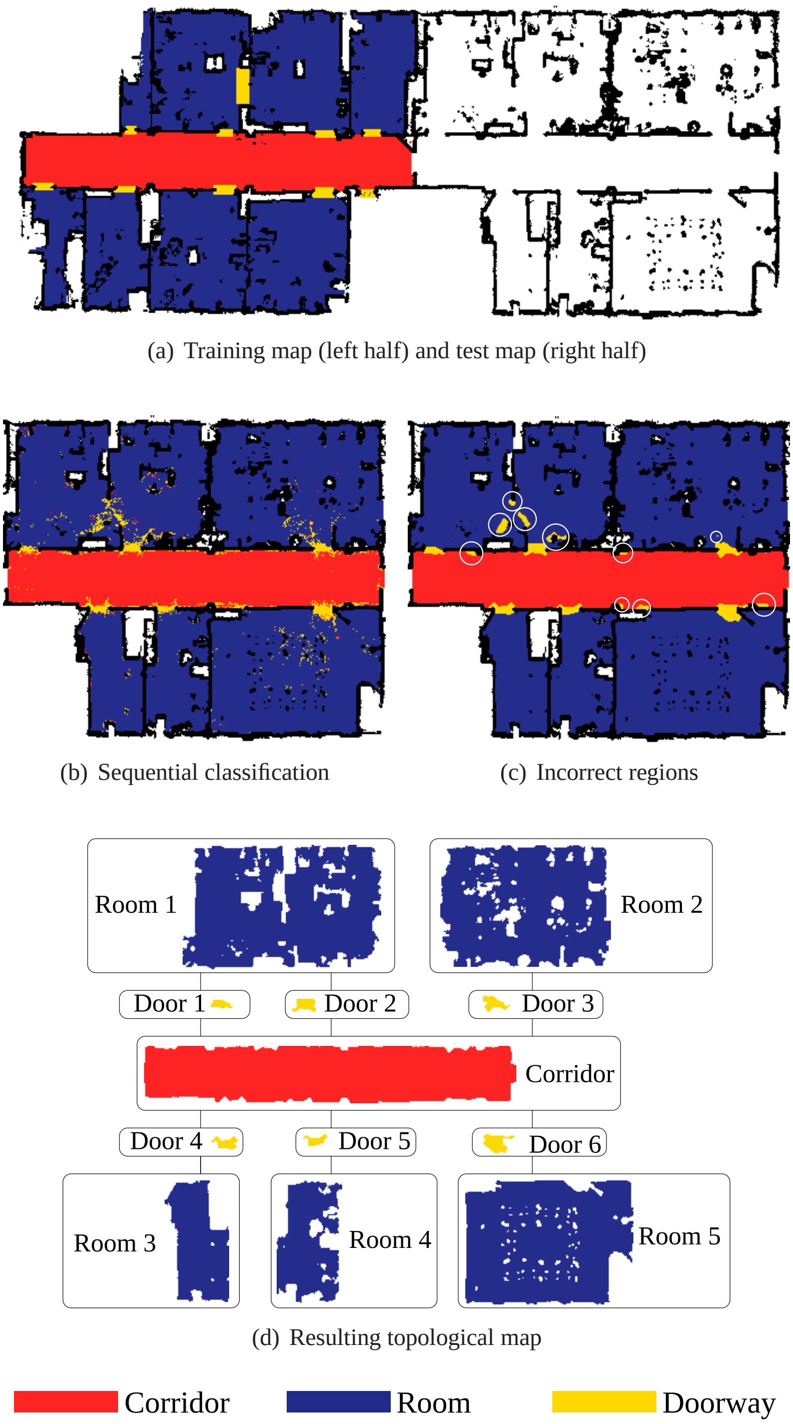
4.5 Experimental Results

The approaches described above have been implemented and tested using occupancy maps obtained from real environments. The laser range data used for the training and classification were simulated using the Carnegie Mellon Robot Navigation Toolkit (CARMEN) [Montemerlo *et al.*, 2003]. The goal of the experiments is to demonstrate that we can construct a semantic-topological map of typical indoor environments using only laser range data. We first apply our method using probabilistic relaxation. Additionally, we analyze whether this method can be used to create a topological map of an environment for which no training data were available. Finally we present one experiment in which iAMNs are use to train and classify an indoor environment.

4.5.1 Results Using Relaxation Labeling

The first experiment was performed using data obtained in the office environment of building 79 at the University of Freiburg. This environment contains rooms, doorways and a corridor, which has a length of approximately 22 meters. For the sake of clarity we give the result of the obtained classification by separating the environment into two parts. The left half of the environment contains the poses used as training examples (see Figure 4.3(a)), and the right half of the environment was used for test classification and for the topological map creation. We used the sequential classifier corridor-room which correctly classifies 97.27% of the test examples. The classification is depicted as colors/grey levels in Figure 4.3(b). After the sequential classification, the probabilistic relaxation method explained in Section 4.2 is applied for 50 iterations. This method generates more compact regions and eliminates noise. The result is illustrated in the Figure 4.3(c). Finally, the topological map is created using the connections between regions. As can be seen in Figure 4.3(c), some regions detected as doorways (marked with circles) do not correspond to real doorways. After applying the heuristics described in Section 4.4 on the corresponding topological map, these false doorways are eliminated. Furthermore, the two left rooms situated above the corridor are detected as only one region. That is due to the fact that the doorway in between was not completely detected. Thus, the two rooms remain connected and are classified as only one region. The final topological map, depicted in Figure 4.3(d), has a final classification rate of 98.95% of the data points.

In a second experiment we created a topological map of the right part of the office environment of building 52 at the University of Freiburg (see Figure 4.4(a)). The length of the corridor in this environment is approximately 20 meters. After applying the decision list classifier room-corridor, the classification of the test set was 97%. Like in the previous experiment, we applied the relaxation process for 50 iterations as well as the operations for region correction. The final result gives a classification rate of 98.66% of the data points. The different steps of the process are illustrated as colors/grey levels in Figure 4.4. As opposed to the previous



(a) Training map (left half) and test map (right half)

(b) Sequential classification

(c) Incorrect regions

(d) Resulting topological map

Figure 4.3: This figure shows (a) the training and test map of the building 79 at the University of Freiburg, (b) the result of applying the decision list with a classification rate of 97.27%, (c) the result of applying relaxation and the detection of incorrect labeled regions (marked with circles), and (d) the final topological map with the corresponding regions.

experiment, the doorway between the two right-most rooms under the corridor is correctly detected (Figure 4.4(c)). Therefore, the rooms are labeled as two different regions in the final topological map.

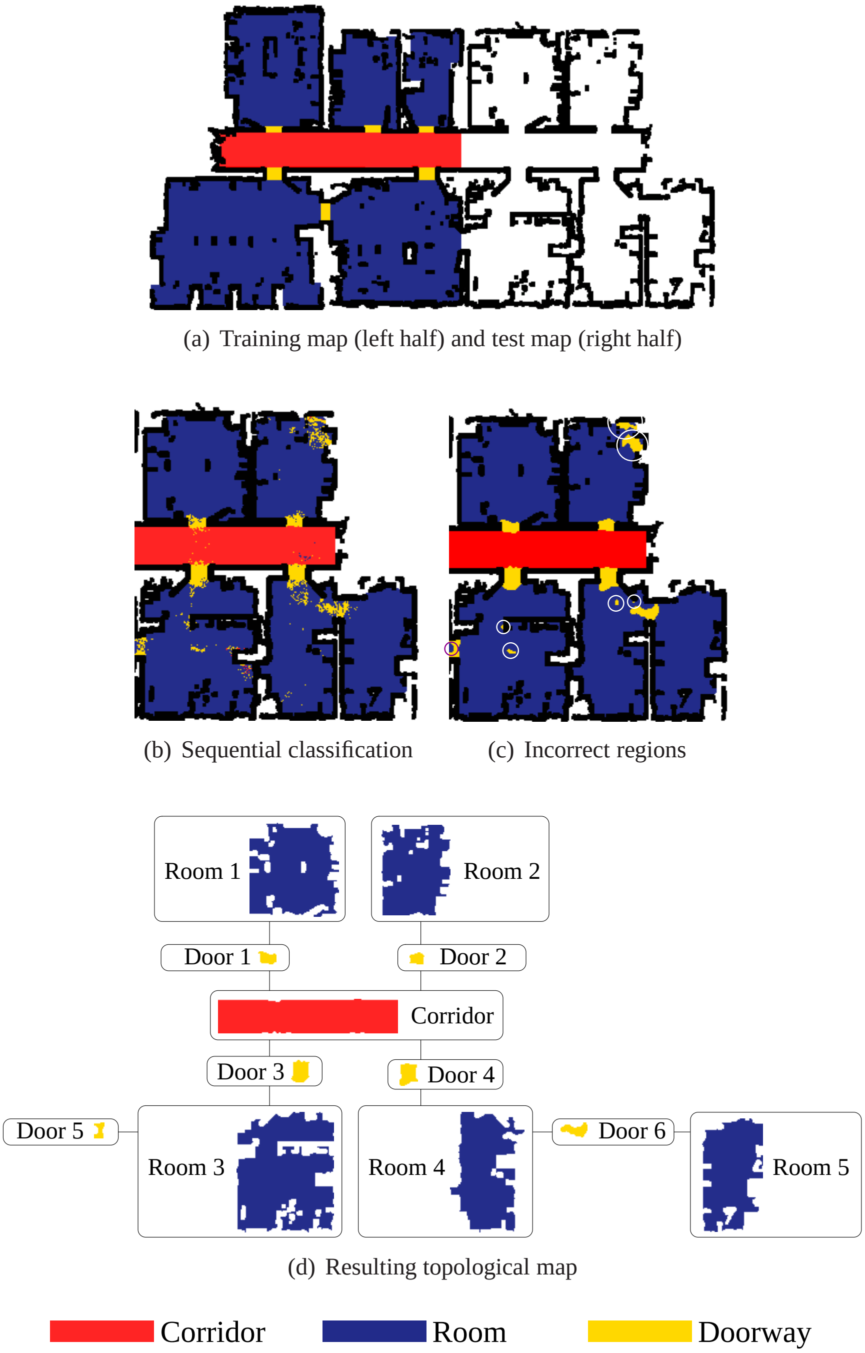
4.5.2 Application to a New and Unknown Indoor Environment

This experiment is designed to analyze whether our approach based on boosting and relaxation labeling can be used to create a topological map of a new environment from which no training data were available. To carry out the experiment we trained a decision list classifier using the training examples of the maps shown in Figure 4.3(a) and Figure 4.4(a) with different scales. In this way, we obtained a classifier with a better generalization. The resulting classifier was then evaluated on scans simulated in the map denoted as *SDR site B* in the Radish repository [Howard and Roy, 2003]. This map represents an empty building in Virginia, USA. The corridor is approximately 26 meters long. The whole process for obtaining the topological map is depicted in Figure 4.5. We use the sequence corridor-doorway which gives a first classification of 92.36%. As can be seen in Figure 4.5(c), rooms number 11 and 30 are originally part of the corridor, and thus falsely classified. Moreover, the corridor is detected as only one region, although humans potentially would prefer to separate it into six different corridors: four horizontal and two vertical ones. Doorways are very difficult to detect by the sequential classifier. The majority of poses detected as doorways disappear after the relaxation process because they are very sparse. The main reason for the problem of doorway detection is that the maps have different sizes and resolutions, and the features are not scale invariant. In the final topological map, 96.94% of the data points are correctly classified.

We also analyzed the results obtained without applying the relaxation process. This had several effects. First, omitting the relaxation procedure reduces the classification rate. Furthermore, the finally obtained regions typically are more sparse and do not represent the original ones as well as with relaxation. Finally, omitting the relaxation procedure increases the number of errors in the resulting topological map. For example, the map for the building in Virginia contained four incorrect nodes without relaxation, whereas there were only two incorrect nodes when we used the probabilistic relaxation.

4.5.3 Results using Instance-based Associative Markov Networks

In this experiment we apply our classification approach using iAMNs to the indoor environment corresponding to the building 79 at the University of Freiburg. For efficiency reasons we used a grid resolution of 20 cm, which lead us to a graph with 8088 nodes. Smaller resolutions result in much bigger networks difficult to treat. As in the first experiment, the map was divided into two parts, the left one used for learning, and the right one used for classification purposes (Figure 4.6). For each cell we calculate 203 geometrical features. This number was reduced to



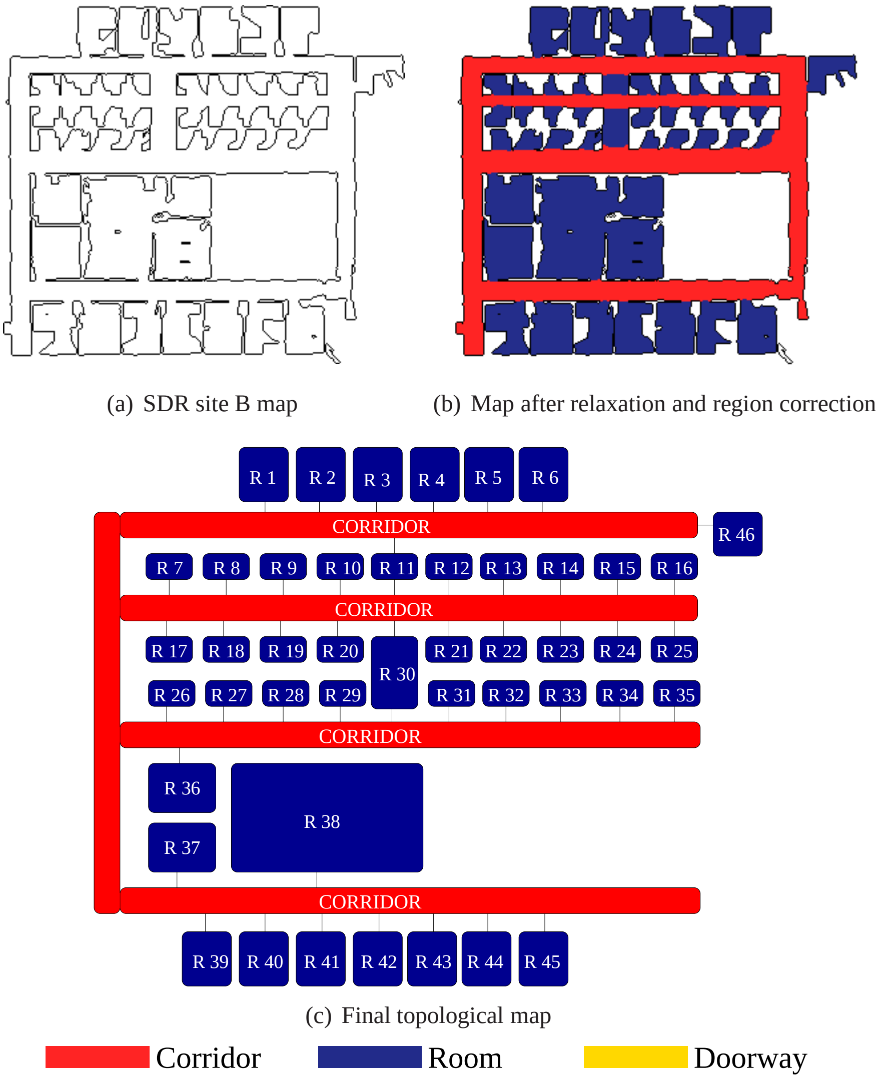
(a) Training map (left half) and test map (right half)

(b) Sequential classification

(c) Incorrect regions

(d) Resulting topological map

Figure 4.4: This figure shows: (a) the training and test map of the building 52 at the University of Freiburg; (b) the result of applying the decision list with a classification rate of 97%, (c) the result of applying relaxation and the detection of incorrect labeled regions (marked with circles), and (d) the final topological map with the corresponding regions.



(a) SDR site B map

(b) Map after relaxation and region correction

(c) Final topological map

Figure 4.5: This figure shows: (a) the original map of the building, (b) the resulting classification after the relaxation an region correction, and (c) the final topological map with semantic information. The regions are omitted in each node. The rooms are numbered left to right and top to bottom with respect the map in (a). For the sake of clarity, the corridor-node is drawn maintaining part of its region structure.

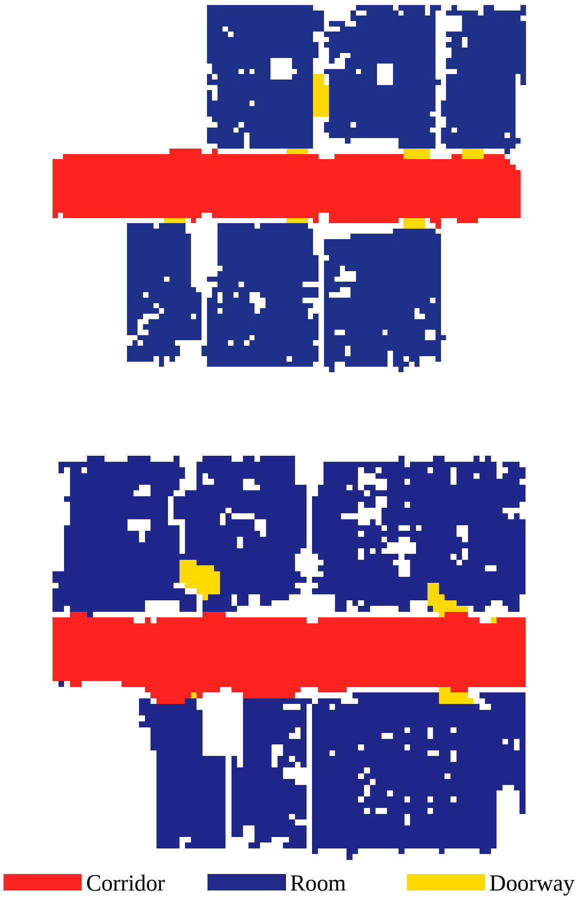


Figure 4.6: The left image depicts the training map of building 79 at the University of Freiburg. The right image shows the resulting classified map using an iAMN with 30 selected features.

30 applying the feature selection of Section 4.3.3. The right image of Figure 4.6 shows the resulting classification with a success rate of 97.6%, which is similar to the classification obtained using relaxation labeling. We can also see in the results that some doorways are lost in the final classification. The reason for this can be the low resolution of the map (20 cm) in comparison with the original resolution (5 cm). However, maintaining the original resolution would lead us to a huge Markov network almost impractical to use.

4.6 Related Work

Different algorithms for extracting topological maps in indoor environments have been proposed. Kuipers and Byun [1991] extract distinctive points in the map, which are defined as the local maximum of some measure. These points are used as nodes in a toplogical map. In their work, Kortenkamp and Weymouth [1994] fuse vision and ultrasound information to determine topologically relevant places. Additionally, Shatkey and Kaelbling [1997] apply a based learning approach based on hidden Markov models to learn topological maps in which the nodes represent points in the plane. Critical points are also found by Thrun [1998], in this case using Voronoi diagrams. The critical points minimize the clearance locally, and are then used as nodes in a topological map. Also Beeson *et al.* [2005] detect topological places with an extension of the Voronoi graph. Furthermore, Choset [2001] encodes metric and topological information in a generalized Voronoi graph to solve the SLAM problem.

In comparison to these previous approaches, the technique described in this chapter applies a supervised learning method to identify complete regions in the map like corridors, rooms or doorways that have a direct relation with a human understanding of the environment.

In addition, mathematical morphology is used in the work by Fabrizi and Saffiotti [2000]. This method uses a disc as structuring element for the dilation and erosion operations. This approach extract large open spaces from the map, but is quite sensitive to irregularities in the map.

Other works use vision sensors to distinguish places in an indoor environment. Tapus and Siegwart [2005] use fingerprints extracted from images to create topological maps. In their work, Zivkovic *et al.* [2005] create a higher level conceptual map with visual landmarks and geometric constraints. These approaches used features extracted from images that are quite specific to the environment the robot is located at, which makes it difficult to generalize to new environments. In contrast to these works, the methods presented in this chapter have better generalization, since they used the geometrical properties of the different places.

In a recent work, Friedman *et al.* [2007] use Voronoi Random Fields for extracting the topologies of occupancy grid maps. This work also uses simple features that are selected using boosting as characteristics for the nodes in a Markov random field. This approach is similar to the one in Section 4.3. However, in [Friedman *et al.*, 2007], only the points lying in the Voronoi diagram are used in the MRF, whereas we used all the free positions in the map.

For related work about semantic place classification we refer the reader to Section 3.7.

4.7 Conclusions

In this chapter, we presented several approaches to create topological maps from indoor environments. The first one uses AdaBoost to learn a strong classifier for categorizing places into semantic classes such as rooms, doorways, and corridors. A probabilistic relaxation process is applied on the resulting classifications to reduce classification errors. The second approach is based on iAMNs together with scalar feature selection. Finally, we extract regions and their connections. The advantage of both approaches is that the nodes of the resulting graph correspond to the individual semantic regions.

Both methods has been implemented and evaluated on various maps from real-world environments. Experiments demonstrate that they are well-suited to create topological maps from indoor environments even without training the classifier for each environment.

Chapter 5

Probabilistic Semantic Classification of Trajectories[1]

5.1 Introduction

The approaches described in the previous chapters are able to classify static observations of a mobile robot. However, mobile robots are dynamic agents that move along different trajectories. When operating in indoor environments, the robots usually have a moderate velocity and a relatively continuous movement. That means, that observations obtained by a mobile robot at nearby poses are typically very similar. Furthermore, certain transitions between classes in a trajectory are rather unlikely. For example, if the classification of the current pose is *kitchen*, then it is rather unlikely that the classification of the next pose is *office* given the robot moved a short distance only. To get from the kitchen to the office, the robot first has to move through a doorway.

In this chapter, we present an approach that takes into account the dependencies between the classification of the poses along a trajectory. In particular, we use a hidden Markov model (HMM) to filter the output of the current classification based on previous ones. In this way, we reduce the number of outliers during the classification.

Additionally, in this chapter we add new places to be recognized in indoor environments. In particular, we want to recognize corridors, doorways, kitchens, seminar rooms, offices, and laboratories. For this purpose, we include a camera on the robot and extract new features from images that permit us to extend the classification to the additional places. The new vision features are based on the recognition of objects. As an example, Figure 5.1 shows an office environment together with some laser and vision data. The increment on the number of places to be recognized shows clearly the improvement of the final classification when applying an HMM to the different place transitions.

[1] The work presented in this chapter originated from a collaboration with Axel Rottmann.

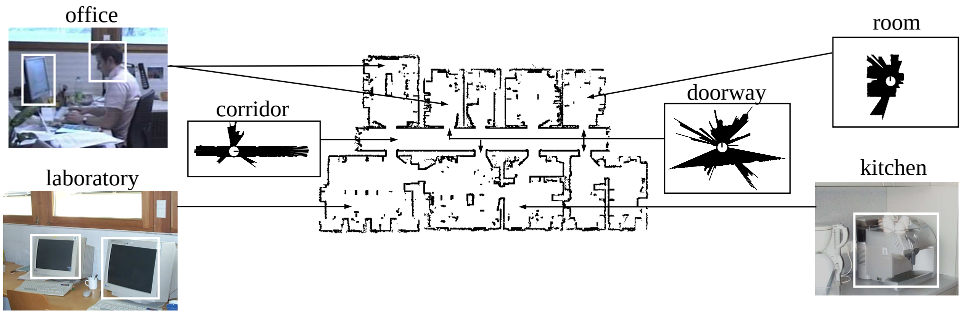


Figure 5.1: An environment with offices, doorways, a corridor, a kitchen, and a laboratory. Additionally, the figure shows typical observations obtained by a mobile robot at different places.

The approach presented in this chapter uses a probabilistic decision list to classify each pose of the robot along a trajectory. Then it applies a Hidden Markov Model to filter the current classification result based on previous ones. As a result the mobile robot is able to classify the different places it traverses with high confidence.

The rest of the chapter is organized as follows. The following section introduces our modification of the AdaBoost algorithm to include the new weak classifiers for vision features. Section 5.3 describes the complete set of simple features extracted from laser and vision data. The models for the HMM are introduced in Section 5.4. In Section 5.5, experimental results obtained with this approach are presented. We discuss related work in Section 5.6. Finally, we conclude in Section 5.7.

5.2 Generalized AdaBoost

As explained in Section 2.3, the generalized AdaBoost algorithm is a supervised learning algorithm designed to find a binary classifier that discriminates between positive and negative examples. AdaBoost boosts the classification performance of a simple learning algorithm by combining a collection of weak classifiers to a stronger classifier. The final strong classifier takes the form of a weighted combination of weak classifiers followed by a threshold. Large weights are assigned to good classification functions whereas poor functions have small weights.

To classify the different places using laser and vision features, two kinds of weak classifiers are created. The first type is used for laser and vision features and has the form

$$h_j(x) = \begin{cases} +1 & \text{if } p_j f_j(x) < p_j \theta_j \\ -1 & \text{otherwise,} \end{cases} \tag{5.1}$$

where θ_j is a threshold and p_j is either -1 or $+1$ and thus representing the direction of the inequality. Note that this form is the same as the one introduced in Section 3.2.

The second type of weak classifiers is designed for our set of vision features and has the form

$$h_j(x) = \begin{cases} p_j & \text{if } \theta_j^1 < f_j(x) < \theta_j^2 \\ -p_j & \text{otherwise}, \end{cases} \tag{5.2}$$

here θ_j^1 and θ_j^2 are thresholds delimiting an interval, and p_j is either $+1$ or -1 indicating if the examples inside the interval are positive or negative. The kind of weak classifier is motivated by that fact that objects appear at different places in different numbers. For example, in an office room we expect more monitors than in the kitchen, but less than in the laboratory. Equation (5.2) was thought to encode this kind of information.

For the multiple class case, we use the same approach as in Section 3.3, and create a probabilistic decision list. Each element k in the list is represented by a classifier for the class k in the environment.

5.3 Simple Features from Laser and Vision Data

In this section, we describe the complete set of features used to create the weak classifiers for each binary AdaBoost classifier in the decision list.

The robot used for the experiments in this chapter is equipped with a 360^o field of view laser sensor and a camera. Each laser observation consists of 360 beams. Each vision observation consists of eight images which form a panoramic view. Figure 5.1 shows typical laser range readings as well as fractions of panoramic images taken in an office environment. Accordingly, each training example for the AdaBoost algorithm consist of one laser observation, one vision observation, and its classification.

Our method for place classification is based on single-valued features extracted from laser and vision data. In the case of laser observations, we extract the set of simple features presented in Section 3.4. These are standard geometrical features used for shape recognition. Furthermore, they are rotational invariant to make the classification of a pose dependent only on the (x, y)-position of the robot and not on its orientation.

In the case of vision, the selection of the features is motivated by the fact that typical objects appear with different probabilities at different places. For example, the probability of detecting a computer monitor is larger in an office than in a kitchen. For each type of object, a vision feature is defined as a function that takes as argument a panoramic vision observation and returns the number of detected objects of this type in it. This number represents the single-valued feature f_j within AdaBoost according to Equations (5.1) and Equation (5.2).

In our case, we consider monitors, coffee machines, soap dispensers, office cupboards, frontal faces, face profiles, full human bodies, and upper human bodies.

An example of such objects is shown in Figure 5.1. The individual objects are detected using classifiers also trained with AdaBoost and based on the set of Haar-like features proposed by Lienhart *et al.* [2003].

5.4 Probabilistic Place Classification

The approach described so far is able to classify single observations, but it does not take into account past classifications when determining the type of place the robot is currently at. However, whenever a mobile robot moves through an environment, the semantic labels of nearby places are typically identical. Furthermore, certain transitions between classes are unlikely. For example, if the robot is currently in a kitchen, then it is rather unlikely that the robot ends up in an office given it moved a short distance only. In many environments, to get from the kitchen to the office, the robot has to move through a doorway first.

To incorporate such spatial dependencies between the individual classes, we apply an HMM and maintain a posterior $P(y_t)$ about the type of the place $y_t \in Y$ the robot is currently at, where Y represents the set of possible semantic labels. The posterior is calculated as

$$P(y_t) = \alpha P(z_t \mid y_t) \sum_{y_{t-1}} P(y_t \mid y_{t-1}, u_{t-1}) P(y_{t-1}). \tag{5.3}$$

In this equation, α is a normalizing constant ensuring that the left-hand side sums up to one over all y_t. To implement this HMM, three components need to be known. First, we need to specify the observation model $P(z_t \mid y_t)$, which is the likelihood that the classification output is z_t given the actual class is y_t. Second, we need to specify the transition model $P(y_t \mid y_{t-1}, u_{t-1})$, which defines the probability that the robot moves from class y_{t-1} to class y_t by executing action u_{t-1}. Finally, we need to specify how the belief $P(y_0)$ is initialized.

In our current system, we choose a uniform distribution to initialize $P(y_0)$. Furthermore, the classification output z_t is represented by a histogram. In this histogram, the k-th bin stores the probability that the classified location belongs to the k-th class, as shown in Section 3.3.

To determine $P(z_t \mid y_t)$, we use the KL-divergence between two histograms [Cover and Thomas, 1991]. The first distribution is the current classification output z_t. The second one is learned from a statistics: for each class y, we compute a histogram $\hat{z}_{1:h}(y)$ using h observations recorded within a place belonging to class y (here $h = 50$). This histogram $\hat{z}_{1:h}(y)$ is obtained by averaging over the individual histograms $\hat{z}_1, \ldots, \hat{z}_h$, which are computed according to Equation (3.6). To determine $P(z_t \mid y_t)$, we use the KL-divergence $kld(\cdot \parallel \cdot)$ which provides a measure about the similarity of two distributions

$$P(z_t \mid y_t) = e^{-kld(z_t \parallel \hat{z}_{1:h}(y_t))}. \tag{5.4}$$

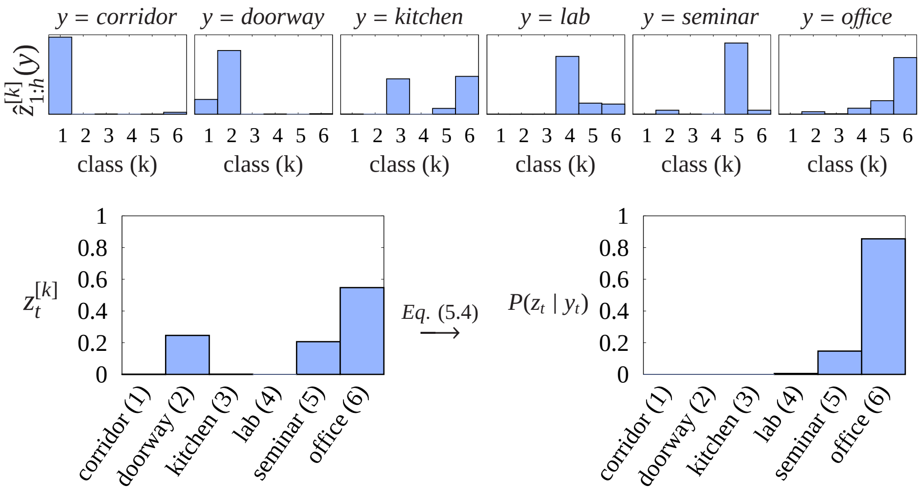


Figure 5.2: The distributions depicted in the first row show the learned histograms $\hat{z}_{1:h}(y)$ for the individual classes (here corridor (1), doorway (2), kitchen (3), lab (4), seminar room (5), and office (6)). The left image in the second row depicts a possible classification output z_t. In the right image, each bar represents the corresponding likelihood $P(z_t \mid y_t)$ for the different estimates of y_t.

To illustrate the computation of the observation likelihood $P(z_t \mid y_t)$ consider Figure 5.2. The first row depicts examples for the histograms $\hat{z}_{1:h}(y)$. The left image in the second row depicts the output z_t of the sequential classifier while the robot was in an office. As can be seen, also the classes doorway and seminar room have a probability significantly larger than zero. This output z_t and the histogram $\hat{z}_{1:h}(y_t)$ is than used to compute $P(z_t \mid y_t)$ according to Equation (5.4). The result for all classes is depicted in the right image in the second row. In this image, each bin represents the likelihood $P(z_t \mid y_t)$ for the individual classes y_t. As can be seen, the observation likelihood given the robot is in a doorway is close to zero, whereas the likelihood given it is in an office is around 90%, which is actually the correct class.

To realize the transition model $P(y_t \mid y_{t-1}, u_{t-1})$, we only consider the two actions $u_{t-1} \in \{Move, Stay\}$. The transition probabilities were learned in a manually labeled environment by running 1000 simulation experiments. In each run, we started the robot at a randomly chosen point and orientation. We then executed a random movement so that the robot traveled between 20 cm and 50 cm forward. These values correspond to typical distances traveled by the robot between two consecutive updates of the HMM. The finally obtained transition probability matrix $P(y_t \mid y_{t-1}, u_{t-1})$ for the action *Move* is depicted in Figure 5.3. As can be seen, the probability of staying in a place with the same classification is higher than the probability of changing the place. Moreover, the probability of moving from a room to a doorway is higher than the probability of moving from a room directly to a corridor. This indicates that the robot typically has to cross a doorway first in

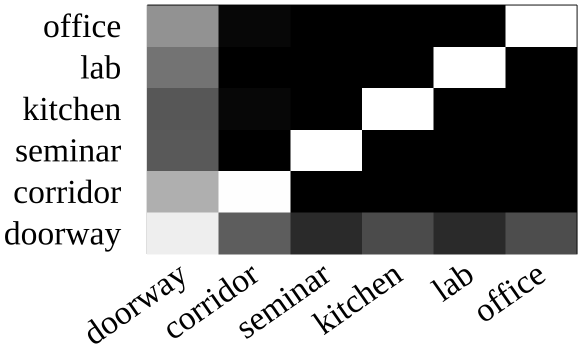


Figure 5.3: The image depicts probabilities of possible transitions between places in the environment. To increase the visibility, we used a logarithmic scale. Dark values indicate low probability.

order to reach a different room. Furthermore, the matrix shows a lower probability of staying in a doorway than staying at the same type of room. This is due to the fact that a doorway is usually a small area in which the robot never rests for a longer period of time.

5.5 Experimental Results

The approach described above has been implemented and tested using simulated and real robot data obtained in our office environment. The goal of the experiments is to demonstrate that our approach provides a robust classification of places in indoor environments into typical categories. We furthermore describe results indicating that the filtering of the classification output using an HMM significantly increases the performance of the overall approach. Additionally, we analyze the benefits of using vision features for the classification.

To train the classifier used throughout the experiments, we used 38,500 training examples. For each training example, we simulated the laser observations given an occupancy grid map of the environment. To generate the features extracted from vision data, we used 350 panoramic views recorded with our B21r robot, which is equipped with a SICK laser range finder and a camera system mounted on a pan/tilt unit as shown in Figure 5.4. Each panoramic view consists of 8 images covering the 360^o field of view around the robot. For each simulated laser scan, we then randomly drew a panoramic view from those corresponding to the type of the current place and used the vision features extracted from this view. Figure 5.5 shows two distributions over the number of coffee machines detected in the database images.

One important parameter of the AdaBoost algorithm is the number T of weak classifiers used to form the final strong binary classifier. For each strong binary classifier, we performed several experiments with up to 500 weak classifiers and analyzed the classification error. The number T of weak classifiers used to carry

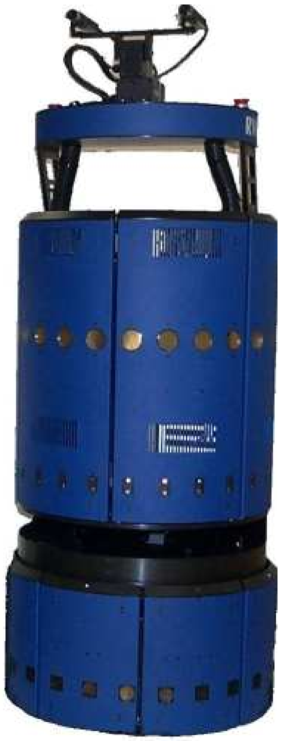


Figure 5.4: The image shows the robot used for the experiments, a B21r robot equipped with a SICK laser range finder and a camera system mounted on a pan/tilt unit.

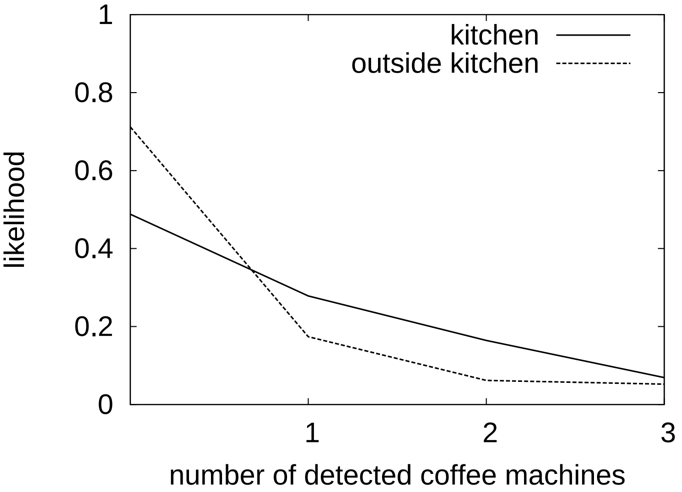


Figure 5.5: Likelihood of detecting *n* coffee machines inside and outside a kitchen using Haar-like classifiers.

Binary Classifier	T	Training error [%]
lab	440	0.99
corridor	165	2.02
doorway	171	2.10
kitchen	68	2.46
seminar	334	2.58
office	288	7.31

Table 5.1: Number T of weak classifiers and training error for the individual binary classifiers.

out the experiments has then been determined as the minimum in the error function. The resulting numbers T of weak classifiers used to form the strong binary classifiers and the classification errors of the finally obtained strong classifiers on the training data are given in Table 5.1.

In our current system, we determine the optimal sequence of strong binary classifiers by considering all possible sequences of strong binary classifiers. Although this approach is exponential in the number of classes, the actual number of permutations considered is limited in our domain due to the small number classes. In practice, we found out that the heuristic which sorts the classifiers in increasing order according to their training classification error also yields good results and at the same time can be computed efficiently. Compared to the optimal order, the classifier generated by this heuristic for an application with six different classes performed on average only 1.3% worse as demonstrated by Rottmann [2005]. In several situations, the sequence generated by this heuristic turned out to be the optimal one.

5.5.1 Classifying Places along Trajectories

The first experiment is designed to demonstrate that the classifier learned from the training data in combination with the HMM can be used to robustly classify observation sequences acquired with a mobile robot in a real office environment. This environment contains six different types of places, namely offices, doorways, a laboratory, a kitchen, a seminar room, and a corridor. The ground truth for the different places in this environment is shown in the top image of Figure 5.6. We steered our robot through the environment and collected laser and image data along its trajectory. We then calculated the classification output without and with the HMM filtering and compared this to the ground truth information.

The classification rate of the sequential classifier without applying the HMM is 74.8%. The generated labels are shown in the middle image of Figure 5.6. If we additionally use the HMM to filter the output of the sequential classifier, the classification rate increases to 83.8%. The labels obtained after applying the HMM are shown in the lower image of Figure 5.6. As we can see in this example, the model for the HMM encodes the possible transitions and discards the ones with low prob-

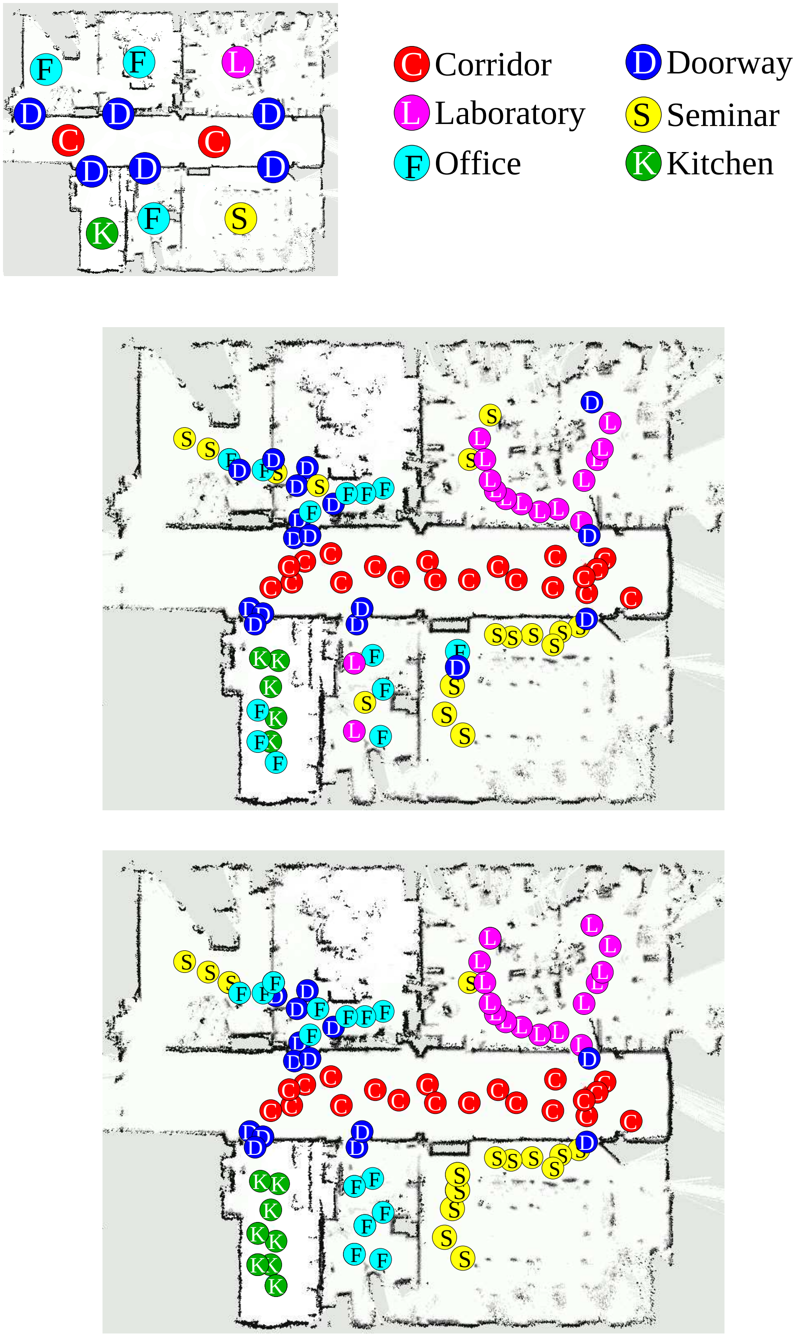


Figure 5.6: Ground truth labeling of the individual areas in the environment (top), and typical classifications obtained for a test set using only the output of the sequential classifier (middle) and in combination with the HMM (bottom).

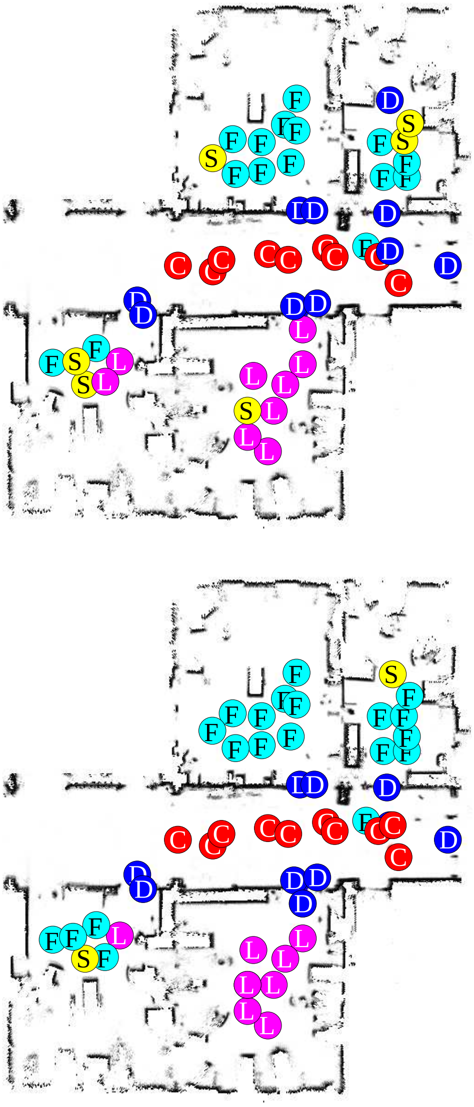


Figure 5.7: Classification obtained without (top) and with HMM filtering (bottom) for a different part of the building.

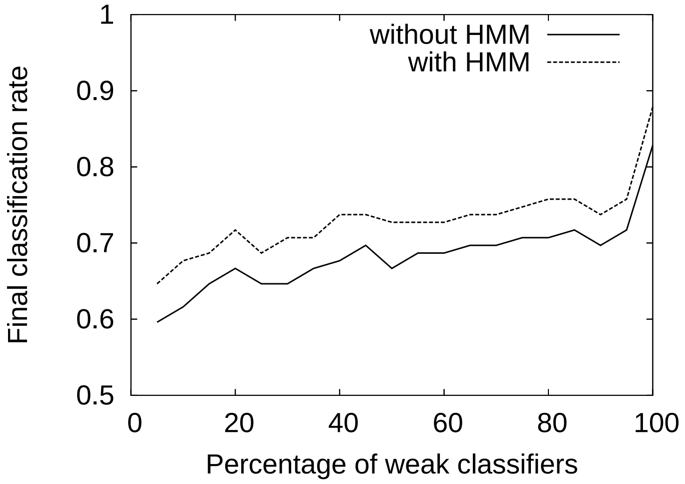


Figure 5.8: Improvement of the HMM according to the percentage of weak classifiers used in each of the binary AdaBoost classifiers.

ability. For example, the wrong office labels that appear in the kitchen (Figure 5.6 middle image) are corrected after the application of the HMM (Figure 5.6 bottom image). The reason is that there is a very low probability of going directly from the kitchen to the office according to the learnt model shown in Figure 5.3. A two-sample *t* test revealed that the improvements of the resulting classification are significant at the $\alpha = 0.01$ level. This illustrates that by using the HMM the overall classification rate can be improved seriously.

A second experiment was carried out using test data from a different part of the same building. We used the same sequential classifier as in the previous experiment. Whereas the sequential classifier yields a classification rate of 77.19%, the HMM generated the correct answer in 87.72% of all cases (see Figure 5.7). This improvement is also significant at the $\alpha = 0.01$ level.

Finally, we studied how the HMM improves the final classification rate according to the output of AdaBoost. For this purpose, we analyzed the improvement of the HMM using different classification rates from AdaBoost. This is achieved by increasing the percentage of weak classifiers used in each binary classifier of the AdaBoost decision list. Here, 100% corresponds to the number of weak classifiers used in the previous experiment (Table 5.1). For example, the classification rate decreases to 60% if only 5% of the weak classifiers are used. The results are shown in Figure 5.8. In average, the HMM improves the classification rate by 5.0%.

5.5.2 Improvement Obtained by Combining Laser and Vision Data

Additionally we analyzed whether the integration of vision and laser data yields any improvements over using only laser. To perform this experiment, we trained AdaBoost only with the three classes office, corridor, and doorway, because the

Sequential Classifier	Error [%] laser	Error [%] laser & vision
corridor-doorway	3.21	1.87
doorway-room	3.74	2.67
doorway-corridor	3.21	2.14
room-corridor	1.60	1.34
corridor-room	1.60	1.34
room-doorway	1.60	1.60
average	2.50	1.83

Table 5.2: Classification error obtained when using only laser data comparing to both laser and vision data.

other classes kitchen, seminar room, and lab can hardly be distinguished from offices using only laser observations. The classification obtained by integrating both modalities is summarized in Table 5.2. As can be seen, the combination of laser and vision data yields better results than the classifier only relying on laser range data.

5.6 Related Work

Classifying the places along a trajectory of a mobile robot is a recent area of interest. At the time of carrying out the work presented in this chapter, very few works considered this problem. Maybe the most known is the work by Torralba *et al.* [2003], which applies a hidden Markov model to distinguish between the places that a mobile robot traverses. Here, the information about the appearance of images is used to discriminate between different places. Compared to this approach, the method presented in this chapter uses an additional laser range finder sensor. Moreover, we use the objects detected in the images instead of calculating visual features based on appearance. We classify the places based on their geometrical 2D structure and the objects found in them. In this way, we enable our robot to generalize better when classifying new environments.

Subsequent works analyze the capabilities for distinguishing places along a trajectory using camera images. Pronobis *et al.* [2006] recognize the different places of an office environment using vision. Their approach is based on two kinds of features extracted from the images: interest points descriptors and appearance features. A similar approach is used by Luo *et al.* [2007], but this time applying incremental learning. Also in [Spexard *et al.*, 2006], rooms are classified according to the appearance of images. In this case the goal of the robot is to recognize already seen rooms. However, these approaches do not take into account past classifications when calculating the current semantic label.

In a very recent work, Pronobis *et al.* [2008] extend their previous work using additionally a laser range finder and the set of geometrical features presented in this thesis. Results show that the laser features improve the generalization of the classifier. In this case no HMM is used to smooth the classification.

For related work about semantic place classification of static poses we refer the reader to Section 3.7.

5.7 Conclusions

In this chapter, we presented a novel approach to classify different places in the environment into semantic classes. This technique uses a combination of simple geometric features extracted from laser range scans as well as features extracted from camera images. It further applies the AdaBoost algorithm to form a strong classifier. To distinguish between more than two classes, we use a sequence of binary classifiers arranged in a probabilistic decision list. To incorporate the spatial dependency between places, we apply a hidden Markov model that is updated upon sensory input and movements of the robot.

Our algorithm has been implemented and tested using a mobile robot equipped with a laser range finder and a camera system. Experiments carried out on a real robot as well as in simulation illustrate that our technique is well-suited to classify places in indoor environments. The experiments furthermore demonstrate that the hidden Markov model significantly improves the classification performance. Additional experiments revealed that the combination of vision and laser data increases the robustness and at the same time allows to distinguish between more classes compared to the approach in which only laser is used.

Chapter 6

Applications of Semantic Information: Localization and Multi-Robot Exploration[1]

6.1 Introduction

The work presented in the previous chapters showed how to augment the representation of indoor environments using semantic and conceptual information about places. These extended representations were mainly used to facilitate the interaction between robots and humans. In this chapter we will see how the semantic information helps to improve other robotic tasks. The main idea is that mobile robots can use the intrinsic information of human-made environments to improve their actions.

In this chapter, we will show the applications of semantic labeling in two robotic tasks: multi-robot exploration, and localization. In both cases, an improvement is obtained when taking into account the classification of the robot location.

The exploration of environments belongs to the fundamental problems in mobile robotics. Exploration is the task of controlling a robot with the goal of maximizing its knowledge about the external world. There exist several applications in which the exploration task is an integral part of the robotic mission (see, for example, [Murphy, 2004], [Thrun *et al.*, 2003], [Huang *et al.*, 1986], and [Jäger and Nebel, 2002]). Additionally, the use of multiple robots is often suggested to have advantages over single robot systems [Cao *et al.*, 1997; Dudek *et al.*, 1996]. In particular, cooperating robots have the potential to accomplish a task faster than a single robot [Guzzoni *et al.*, 1997].

Indoor environments constructed by humans often contain certain structures, like corridors with adjacent rooms or offices. This information has been typically ignored when coordinating a team of robots in an exploration task. In typical indoor

[1] The work presented in this chapter originated from a collaboration with Cyrill Stachniss

environments, corridors usually provide more branchings to new unexplored areas like adjacent rooms. Therefore, it makes sense to focus first on corridors in order to obtain a better assignment of targets to robots.

Mobile robot localization is the problem of determining the pose of the robot relative to a given map of the environment. Different approaches have been used to solve this problem, like for example, grid-based Monte Carlo localization [Simmons and Koenig, 1995], multi-hypothesis EKFs [Jensfelt and Kristensen, 2001], and condensation-based algorithms [Dellaert *et al.*, 1999]. In this chapter we will use the last approach, also known as Monte Carlo localization. We will show how global localization can be improved using both odometry and semantic labels in comparison to using only odometry.

In this chapter, we first present an approach to include semantic information about places to better distribute the robots in human-made environments during the exploration tasks. The key idea is to assign higher rewards to robots that first explore corridors. As a result, the overall completion time of an exploration can be significantly reduced.

In a second approach, we use the semantic labeling in the Monte Carlo localization algorithm. The main idea here is to take as observation model the semantic classification of the current pose of the mobile robot.

The rest of the chapter is organized as follows. In Section 6.2, we present our method to assign semantic labels to target locations during the exploration task. The algorithm for target assignment to robots using semantic information is shown in Section 6.3. In Section 6.4, we introduce the Monte Carlo approach for localization using semantic labels. In Section 6.5 experimental results are presented. We discuss related work in Section 6.6. Finally, conclusions are presented in Section 6.7.

6.2 Sematic Classification of Target Locations

During this chapter we assume that the knowledge about the environment is represented by an occupancy grid map. Thus, the exploration problem consist of controlling the robot so that its maximizes the occupancy information in the map. Using this representation, target locations are located at the frontier between known and unknown areas and can be extracted using the approach by Yamauchi *et al.* [1999]. The left image of Figure 6.1 shows an example of a map together with the frontiers detected there (shown as dashed lines). For each of the frontiers, a target location is generated.

The goal now is to classify each potential target location into a semantic class. One possible solution to classify a target location is to simulate an observation at its position, and then classify this observation using the approach presented in Chapter 3. However, the target position is located at a frontier, which means that part of the neighboring areas are not known. This situation is shown in Figure 6.1. Therefore, the laser observations simulated at frontier cells contain a significant

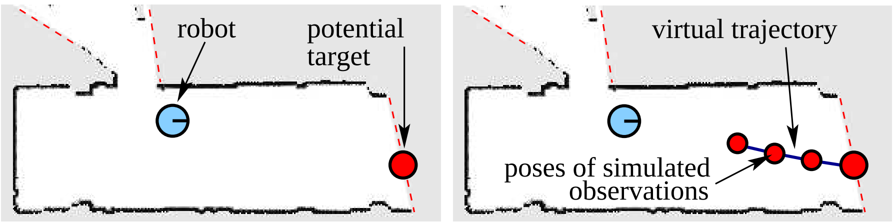


Figure 6.1: The left image shows a situation in which a robot has extracted the frontiers of the occupancy grid map (dashed lines). Additionally, a target location is shown for one of the frontiers. In the right image, a virtual trajectory to the target is generated by the robot.

number of maximum-range readings, which can lead to high missclassification rates. To increase the classification rate in these cases, we generate a short virtual trajectory to the desired goal location. We then simulate laser range observations at different poses along the virtual trajectory using the partially know map. These poses are generated selecting cells in the occupancy grid which are as far away as possible from the unknown locations in the current map. The reason for this selection is that cells having more information about its surroundings will have a lower error in its semantic classification, since their simulated range scans will contain fewer maximum-readings.

To generate the different positions along the trajectory we apply the euclidian distance transformation [Meijster *et al.*, 2000] with respect to unknown and occupied cells in the local area of the frontier. We select the pose in the free space within that local area containing the highest distance to unknown areas. Then an A* planner is used to generate the virtual trajectory to the target location. An illustrating example is depicted in Figure 6.1.

Once we have the virtual trajectory, we follow the approach presented in Chapter 5. We apply a hidden Markov model (HMM) and maintain a posterior $P(y_t)$ about the type y_t of the place the virtual sensor is currently at as

$$P(y_t) = \alpha P(z_t \mid y_t) \sum_{y_{t-1}} P(y_t \mid y_{t-1}, u_{t-1}) P(y_{t-1}). \tag{6.1}$$

The different components of this model are calculated in the same way as explained in Chapter 5. Using Equation (6.1), we classify the target location $P(y_{target})$ using the classification of the positions leading to it.

6.3 Target Assignment using Semantic Place Labeling

We now present the algorithm used to assign target locations to each robot of the team during the exploration. As indicated above, the main idea is to give priority to target locations that are located at corridors, as they lead to a higher number of unknown areas. The algorithm used for target assignment is shown in Figure 6.2.

In a first step, the algorithm calculates the set of frontier cells. This is done using the method explained in Section 6.2. For each robot i in the team, the algorithm then calculates the cost V_t^i of reaching each frontier cell t. This cost is based on the distance the robot has to travel to reach the cell. Additionally, the algorithm estimates the semantic label L_t of the target location t using the HMM-based approach presented in Section 6.2.

Using the label L_t of cell t and the number n of robots in a team, an initial utility function $U_{init}(L_t, n)$ is assigned to each target location t. At this point the semantic information about places is considered in the algorithm. The target locations classified as corridors get an initial utility U_{init} which is γ times higher than other locations of the current indoor environment. After several experiments we select a value of 5 for γ. This value led to the best results in different runs of the algorithm.

An iterative process is then carried out in which the best combination of robot i and target t is selected. This selection is done maximizing the utility function U_t at each step. As several robots can be assigned the same frontier cell, each cell is discounted each time it is assigned to a robot. In this way, only one robot is assigned to each frontier cell. Additionally, target locations which can potentially be observed by other robots already assigned are discounted. This is done by introducing a utility function $U(t)$ given by

$$U(t_n \mid t_1, \ldots, t_{n-1}) \quad = \quad U_{t_n} - \sum_{i=1}^{n-1} P_{vis}(t_n, t_i)\,, \tag{6.2}$$

where $P_{vis}(t_n, t_i)$ describes the probability that the frontier t_n can be observed by a robot moving to t_i. In our approach, this probability density is approximated by a linear function.

The algorithm of Figure 6.2 furthermore reduces the interfence of robots during the exploration taking into account the visibility constraints, which are included in the utility function. Moreover, the inclusion of semantic information about the target locations improves the distribution of robots, giving preference to corridor places when selecting goal position for exploring unknown areas. As a result, the time needed to explore an indoor environment using a team of robots is significantly reduced.

However, the exploration time reduction is not significant when using teams with a small number of robots. This fact can be explained by considering the single-robot exploration scenario. In this case, it makes no sense to focus on exploring the corridors first, since the robot has to cover the overall environment with its sensor. In our experiments, the exploration time doesn't decrease if the team has less than five robots.

- Determine the set of frontier cells.
- Compute for each robot i the cost V_t^i for reaching each frontier cell t.
- Estimate for each frontier cell t the semantic labeling L_t.
- Set the utility U_t of all frontier cells t to $U_{init}(L_t, n)$ according to their semantic labeling L_t and the size n of the team.
- While there is one robot left without a target point

  1. Determine a robot i and a frontier cell t which satisfy

     $$(i, t) = \underset{(i',t')}{\operatorname{argmax}} \left(U_{t'} - V_{t'}^{i'} \right).$$

  2. Reduce the utility of each target point t' in the visibility area according to

     $$U_{t'} \leftarrow U_{t'} - P_{vis}(t, t').$$

Figure 6.2: Algorithm for the assignation of target locations to the different robots in a team.

6.4 Localization Using Place Recognition

In this section we show how to include the semantic information about places in the popular Monte Carlo localization approach introduced by Dellaert *et al.* [1999]. This localization method applies a recursive Bayesian scheme to maintain a posterior about the location of the robot x_t given the map m of the environment, the odometry information $u_{0:t-1}$, and the observations $z_{1:t}$

$$\begin{aligned} p(x_t \mid m, z_{1:t}, u_{0:t-1}) \quad = \quad & \eta \cdot p(z_t \mid m, x_t) \cdot p(x_t \mid m) \cdot \\ & \cdot \int_{x'} p(x_t \mid x', u_{t-1}) \cdot p(x' \mid m, z_{1:t-1}, u_{0:t-2})\, dx'. \end{aligned} \tag{6.3}$$

In our implementation, m is a occupancy grid map, in which each cell additional stores the the semantic label corresponding to its place. The set of possible places to be recognized are corridor, doorway, office, kitchen, seminar room, and laboratory. This set of places corresponds to the one applied in Chapter 5.

As observations $z_{1:t}$, we use the output of the classifier the robot uses for place labeling. This classifier is the same as the one introduced in Section 5.2, and applies a probabilistic decision list in which each element is a binary AdaBoost-based classifier. The quantity $p(z_t \mid m, x_t)$ is then determine as $p(z_t \mid y_t)$, where y_t is the class assigned to x_t in m. To estimate $p(z_t \mid y_t)$, we generated statistics about the output of the sequential multiclass classifier given the robot was at a place corresponding to y_t. Additionally, we weight the particles inversely proportional to the occupancy probability at x_t in m.

6.5 Experimental Results

The approaches presented in this chapter were implemented using real robots as well as simulations. The goal of the experiments is to show that the semantic information about places can improved both, multi-robot exploration in indoor environments, and localization of single robots.

6.5.1 Improving the Exploration Time Using Semantic Information

The first experiments were designed to show how the semantic information about places can improve the distributions of the robots in a team during the explorationtasks. Due to the big numbers of robots used, we evaluated our technique only in simulation experiments.

To prevent a loss of performance compared to approaches which do not consider semantic place information for small robot teams, we trigger the influence of the semantic place information depending on the size of the team. We linearly decrease the influence γ for teams smaller than 10 robots. The linear interpolation of the influence of the semantic labels is encoded in the utility function $U_{init}(L_t, n)$, where n denotes the number of robots in the algorithm of Figure 6.2.

In a first experiment we used the map of the Fort Sam Huston hospital (Figure 6.3, top), which contains several corridors together with rooms adjacent to them. In the experiment we apply our method for coordinating several robots using semantic information about places, and compared it to the case in which no place information is used. The bottom image of Figure 6.3 shows the results when using different robot teams. The number of robots varies from 5 to 50 in each team. For each team size, we repeated the experiments 50 times. In all the experiments the robots started from the same initial position. As the plot shows, the time needed to explore the complete environment is significantly reduced at the confidence level of 0.05 when using semantic place information.

The reason for this time reduction is the increment of target locations when using our approach. As the robots concentrate on exploring the corridor first, more target locations appear coming from the adjacent rooms. Figure 6.4 depicts the effect on the number of frontier cells when using place labeling. As the image shows, a bigger number of target locations appears during the different decision steps of the target assignment algorithm.

A similar experiment was carried out using the map of the Intel Research Lab (top image of Figure 6.5). Again we observed a significant reduction in the exploration time as shown in the bottom image of Figure 6.5.

Additionally, our assignation method reduces the interferences between the path of the robots during the exploration. This reduction was of up to 20% in our experiments.

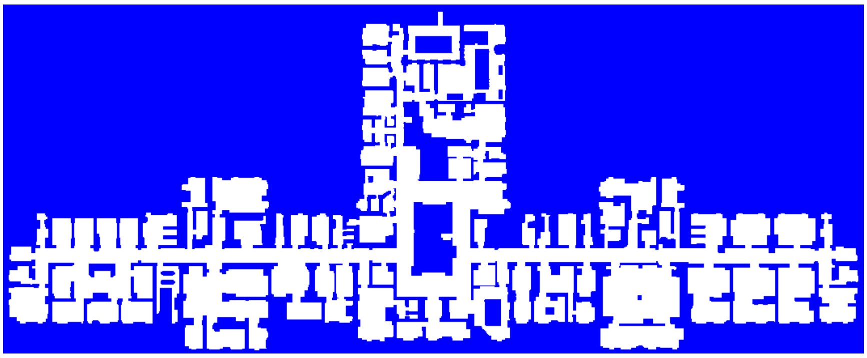
6.5.2 Influence of Noise in Place Labeling

So far we have assumed that the semantic classification of the target locations had no errors. In real situations,however, errors usually appear during the labeling process (see experimental results in Chapter 3 and 5). It is therefore interesting to analyze how the classification errors affect the performance of our method. For this, we carried out an experiment in which we randomly misclassified different percentages of target locations, and measure the exploration time according to them. Figure 6.6 shows the resulting performance using different team sizes. When the error in the classification exceeds 15% the improvement using semantic information is not significant anymore.

6.5.3 Localization Using Place Recognition

The last experiment is designed to illustrate how semantic information about places can be used to improve the localization of a mobile robot in its environment. In this experiment, we used an ActivMedia Pioneer II robot. Note that the laser data is only fed into the semantic classifier and not used for metric localization.

Figure 6.7 illustrates the evolution of two particle sets over time. In the first row, the semantic information was available whereas in the second row only the odometry information was used. Both filters were initialized with a uniform dis-



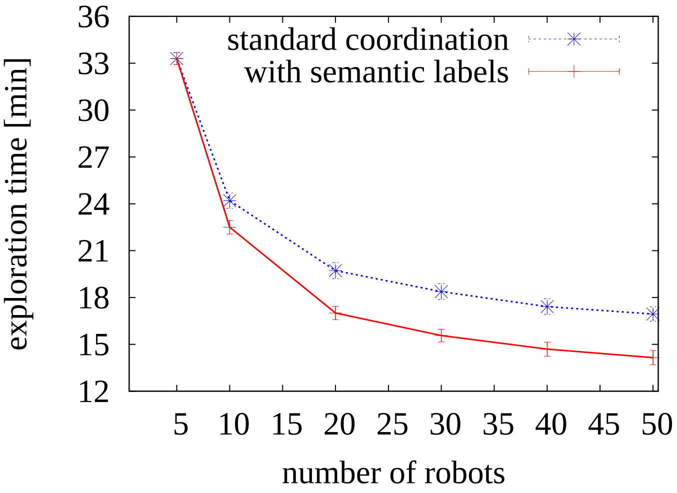


Figure 6.3: The top image shows the map of the Fort Sam Huston hospital. The bottom image depicts the performance when semantic information used in comparison to the case where no label information is used.

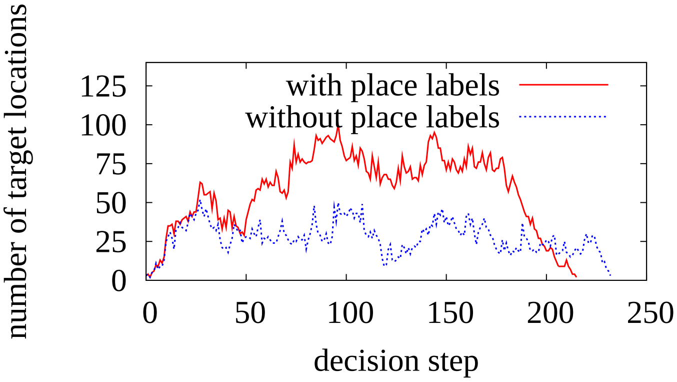
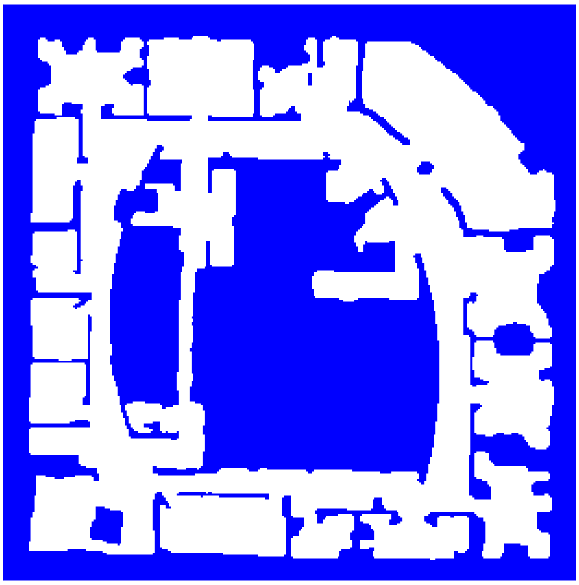


Figure 6.4: Plot showing the number of potential target locations during the exploration.



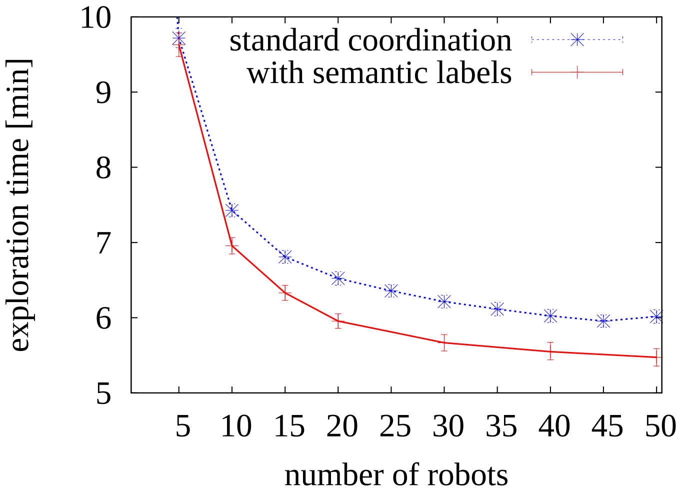


Figure 6.5: The top image shows the map of the Intel Research Lab. The bottom image depicts the exploration time with and without using semantic information.

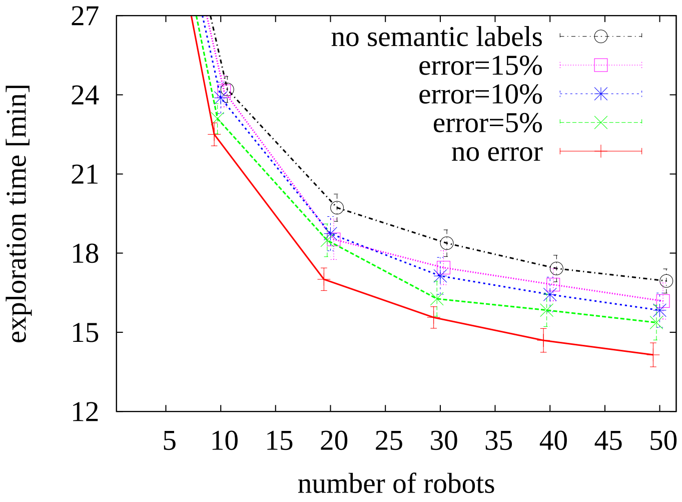


Figure 6.6: Exploration performance with different classification errors.

tribution with 10, 000 particles. The robot initially was located in the second left office, north of the corridor. Therefore, particles located in office received higher importance weights compared to the other samples. Whereas the approach utilizing semantic information converges quickly to the correct solution, the particle filter that relies only on the odometry information $p(x_t \mid m)$ finally diverges.

6.6 Related Work

Different aspects of multi-robot exploration have been studied in the past. For example, Singh and Fujimura [1993] present a method for heterogeneous robot teams. In this approach, if a robot is too big to pass through a narrow passage, it informs other robots to do this task. Howard *et al.* [2002] introduce an incremental deployment approach that explicitly deals with situations in which the path of one robot is blocked by another.

Matarić and Sukhatme [2001] present different strategies for allocating tasks in a robot teams and analyze their performance in different experiments. The work by Parker [2003] studies how a team of heterogeneous robots can jointly solve certain task that can not be accomplished by a robot individually. The Hungarian method to compute the assignments of frontier cells to robots is introduced by Ko *et al.* [2003]. In contrast to our work, Ko *et al.* [2003] mainly focuses on finding a common frame of reference in case the start locations of the robots are not known.

The coordination technique presented is this chapter is an extention of the work by Burgard *et al.* [2005]. We also discount the utility of target locations if they are visible from a goal location already assigned to a robot. In contrast to [Burgard *et al.*, 2005], our approach estimates and incorporates background knowledge about

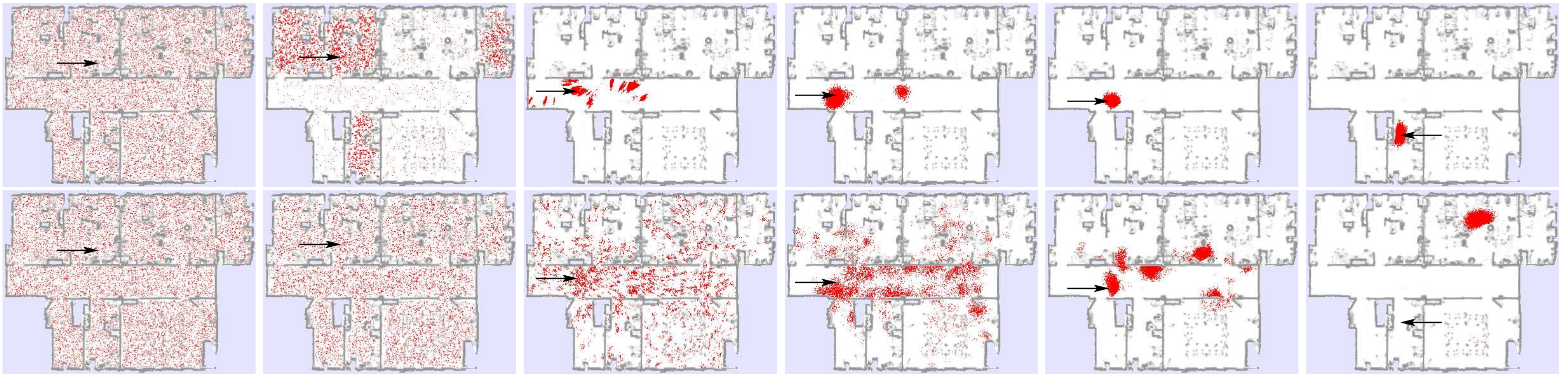


Figure 6.7: Global localization using semantic information and odometry (first row) compared to an approach using only the odometry information (second row). The images in one same column represent the corresponding filter at the same time. The arrow indicates the ground truth position. As the results indicate, semantic information can be used to speed up global localization.

environmental structure into the goal point assignment procedure.

The semantic labels used to improve multi-robot coordination can be seen as background knowledge about spacial structures. Fox *et al.* [2003] presented a technique which aims to learn background knowledge in typical indoor environments and later on use that knowledge for map building. They apply their approach to decide whether the robot is seeing a previously built portion of a map, or is exploring new terrain.

Localization is a typical problem in mobile robotics, and different approaches have been applied to solve this problem. The grid-based grid-based Monte Carlo localization was introduced by Simmons and Koenig [1995]. This approach approximates the posterior of the robot pose using a histogram over the possible discrete poses. Several authors have successfully applied grid-based Monte Carlo localization in their work, as for example Burgard *et al.* [2000a], Hertzberg and Kirchner [1996], and Simmons *et al.* [2000]. Multi-hypothesis extended Kalman filters is another approach for localization used in different works, as for instance in [Jensfelt and Kristensen, 2001], [Roumeliotis and Bekey, 2000], and [Reuter, 2000]. Finally, particle filter approaches were introduced by Dellaert *et al.* [1999] and Fox *et al.* [1999]. In this chapter we will use the last approach, also known as Monte Carlo localization, including the semantic classification of places.

For related work about the method for semantic labeling of places, we refer the reader to Sections 3.7 and 5.6.

6.7 Conclusions

In this chapter, we have shown how the semantic information helps to improve other robotic tasks. In particular, we proposed a novel technique that takes into account semantic information about places in the context of coordinated multi-robot exploration. The main idea is that mobile robots can use the intrinsic information of human-made environments to improve their actions. This improvement is obtained by selecting the best target locations according to their semantic classification. The semantic labeling of the target locations is done using an AdaBoost-based classifier. Additionally, a hidden Markov model is apply to improve the classification in a virtual trajectory to the target position.

Alternatively we have seen how the semantic information about places can be used to localize the robot in an indoor environment using the Monte Carlo localization approach. In this case, the observation model of the robot corresponds to the semantic classification of its position.

Both methods demonstrated that the semantic information can be useful in different tasks using autonomous mobile robots.

Chapter 7

Semantic Information in Sensor Data[1]

7.1 Introduction

So far, we have seen how to augment the maps in the environments with semantic information. This additional information was obtained by classifying the laser range data obtained by a mobile robot into some of the classes that represent the different places in the environment.

Here, we present a different approach. Instead of classifying the pose of the robot according to the corresponding range observation, we classify the observation itself by assigning a semantic label to each of its measurements. The main idea is to classify each laser beam into the class of object it hits. For example, if a beam hits a person, then we assign the label *person* to it. In this way, the data provided by the range sensor contains additionally semantic information about the objects in the environment.

In this chapter we consider the binary case in which two possible labels are assigned to the different range measurements: *person* or *non-person*. The choice of these labels is given by the fact that the world is a dynamic environment in which different agents, such as people, cars, animals and others, move constantly. In the specific case of indoor environments, like offices or houses, we can consider that people are the most common moving agents. It is thus interesting to design methods for the detection of people by mobile robots.

The application of laser sensors for people detection has been popular in the past, as they provide a large field of view and, opposed to vision, are mainly independent from ambient conditions. However, laser range data contain little information about people, especially because they typically consist of two-dimensional range information. Figure 7.1 shows an example scan from a cluttered office environment. While this scan was recorded, several people walked through the office. The scan suggests that in cluttered environments, people detection in 2D is difficult

[1] This chapter originated from a joint work with Kai O. Arras.

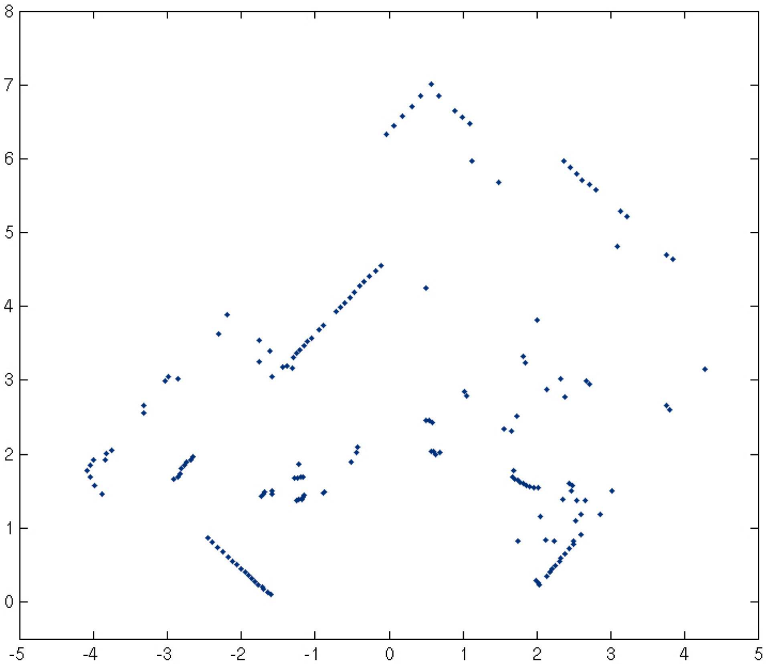


Figure 7.1: Example scan from a typical office. It seems very difficult to detect which beams are hitting people.

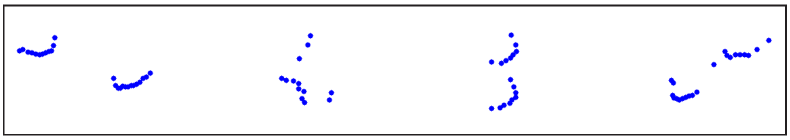


Figure 7.2: Typical range readings from legs of people. As can be seen, the appearance can change drastically, also because the legs cannot always be separated. Accordingly, the proper classification of such pattern is difficult.

even for humans. However, at a closer look, range measurements that correspond to humans have certain geometrical properties such as size, circularity, convexity or compactness (see Figure 7.2).

The key idea presented in this chapter is to divide the range observations into segments, and then extract several scalar features from them that encode their geometrical properties. Finally, we apply a supervised learning algorithm based on AdaBoost to select the best features while at the same time creating a classifier for the laser beams.

The rest of the chapter is organized as follows. Section 7.2 describes the method to classify the beams in a scan. In Section 7.3, the set of geometrical features is described. Experimental results are shown in Section 7.4. We discuss related work in Section 7.5. Finally, we conclude in Section 7.6.

7.2 Classification of Segments Using Boosting

The key idea of this work is to classify the beams corresponding to a laser scan into ones hitting a person or not. For this purpose we segment the beams into subsets using a segmentation algorithm. We then extract some geometrical features from these subsets and apply the generalized AdaBoost algorithm to select the features that best classify the beams into measurements corresponding to people.

As explained in Section 2.3, the generalized AdaBoost algorithm is a supervised learning algorithm designed to find a binary classifier that discriminates between positive and negative examples. AdaBoost boosts the classification performance of a simple learning algorithm by combining a collection of weak classifiers to a stronger classifier. The final strong classifier takes the form of a weighted combination of weak classifiers followed by a threshold. Large weights are assigned to good classification functions whereas poor functions have small weights.

To classify the different segments of a laser observation, we create a weak classifier for each of the geometrical features f extracted from them. The weak hypotheses have the form

$$h_j(x) = \begin{cases} +1 & \text{if } p_j f_j(x) < p_j \theta_j \\ -1 & \text{otherwise,} \end{cases} \tag{7.1}$$

where θ_j is a threshold and p_j is either -1 or $+1$ and thus representing the direction of the inequality. This form is similar to the one used in Section 3.2. Also here the algorithm determines for each weak classifier $h_j(x)$ the optimal values for θ_j and p_j, such that the number of misclassified training examples is minimized

7.3 Feature Extraction

In this section we explain how the geometrical features are extracted from the laser observations. We assume that the robot is equipped with a range sensor that delivers observations $z = \{b_1, ..., b_L\}$ that consist of a set of beams. Each beam b_j corresponds to a tuple (ϕ_j, ρ_j), where ϕ_j is the angle of the beam relative to the robot and ρ_j is the length of the beam.

The beams in the observation scan z are split into subsets using a jump distance condition: If two adjacent beams are farther away than a threshold distance, a new subset is initialized. As we will see in the experiments, this simple method results in good segmentations for the detection of people. For an additional list of segmentation algorithms we refer the reader to the work by Premebida and Nunes [2005].

The output of the partitioning procedure is an ordered sequence $\mathcal{P} = \{S_1, ..., S_M\}$ of segments such that $\bigcup S_i = z$. The elements of each segment $S = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ are represented by Cartesian coordinates $\mathbf{x} = (x, y)$, where $x = \rho\cos(\phi)$ and $y = \rho\sin(\phi)$, and (ϕ, ρ) are the polar coordinates of the corresponding beam.

The final training set for the AdaBoost algorithm is then given by a set of segments together with their corresponding labels

$$E = \{(S_i, l_i) \mid l_i \in \{+1, -1\}\},$$

where $l_i = +1$ indicates that the segment S_i represents a person and $l_i = -1$ indicates that the segment S_i is not a person.

Once the laser observation is divided into the different segments, we can proceed to extract the geometrical features from them. We define a feature f as a function $f : \mathcal{S} \rightarrow \mathbb{R}$ that takes a segment S as an argument and returns a real value. Here, $\mathcal{S}$ is the set of all possible segments. For each segment we determine the following fourteen features:

1. Number of points in the segment.
2. Standard deviation of the beams length.
3. Mean average deviation from median.
4. Jump distance from preceeding segment.
5. Jump distance to succeeding segment.
6. Euclidian distance between the first and last point of a segment.
7. Linearity of the segment.
8. Circularity of the segment.
9. Radius of the circle fit in the segment.
10. Boundary length.
11. Boundary regularity.
12. Mean curvature.
13. Mean angular difference.
14. Mean speed between two consecutive scans.

This collection of features constitutes a profile of each segment (see Figure 7.3). Since certain features are not defined for less than three points (e.g., circularity, radius) only segments with $n > 2$ points are taken into account. Details for the calculation of each feature are given in Appendix B.

Some works [Fod *et al.*, 2002; Cui *et al.*, 2005; Topp and Christensen, 2005] report the use of additional conditions on the distance between blobs, typically to associate two legs to the same person. We deliberately do not consider such conditions. The association of single legs to persons, especially when several people stand close together, is a complex data association problem. In our opinion, this problem can more robustly be solved by integrating multiple observations over time rather than directly on the level of the feature detector.

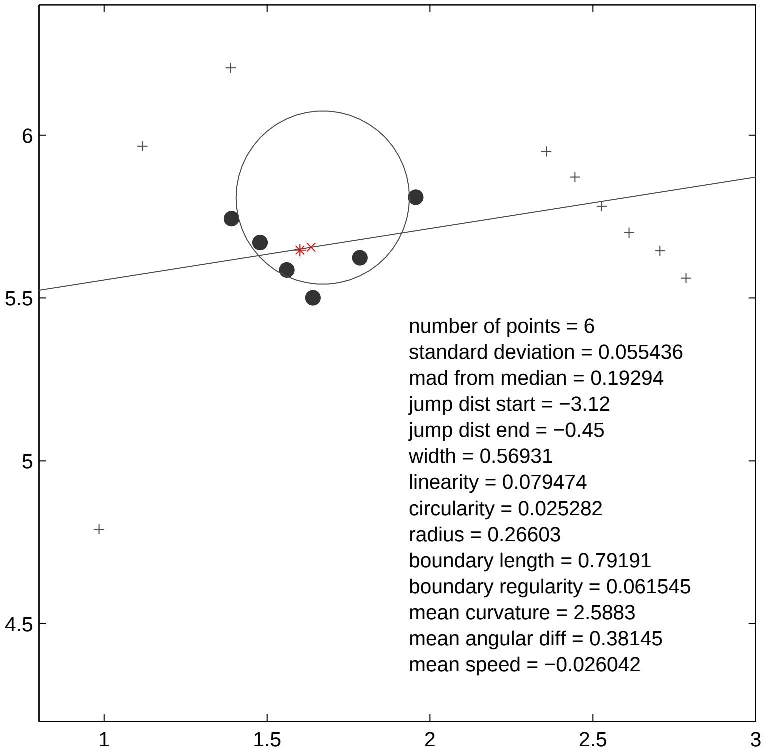


Figure 7.3: Laser segment with its feature profile. The highlighted points correspond to the segment and the crosses depicts other readings in the scan. The circle and line are fitted to the segment for the *linearity* and *circularity* features.

	Detected Label		
True Label	Person	No Person	Total examples
Person	**239** (99.58%)	**1** (0.42%)	**240**
No Person	**27** (1.03%)	**2589** (98.97%)	**2616**

Table 7.1: Confusion matrix for the corridor environment.

7.4 Experimental Results

The approach presented above has been implemented using a 180^o SICK laser range finder. The goal of the experiments is to demonstrate that our simple features can be boosted to a robust classifier for the semantic classification of the beams corresponding to a laser scan. Each beam is semantically labeled as person or non-person.

Throughout the experiments, the sensor was kept stationary and mounted 30 cm above the floor. The corresponding scans where segmented and the geometrical features were calculated for each segment according to Section 7.3. The complete set of labeled segments was then divided randomly into a training and a test set, each containing approximately 50% of the segments. The training sets were employed for learning a strong classifier using the method presented in Section 7.2, whereas the test set was used for the evaluations. The segments in both sets were labeled manually with the help of videos recorded during the experiment.

One important parameter of the AdaBoost algorithm is the number of weak classifiers T used to form the final strong classifier. We need a fast people detector as we want the classifier to work in real time. After several experiments, we found that a value of $T = 10$ weak classifiers was the best trade-off between the error rate and the speed of the classifier.

7.4.1 Corridor and Office Environments

In the first experiment we analyze the performance of our method when used in a corridor. We recorded a total of 540 scans in the corridor of the building 79 at the University of Freiburg. This corridor is approximately 20 meters long. The scans were recorded while a person was both moving and standing still (Figure 7.4 left). Each scan was divided into segments and for each segment the features #1 to #13 were calculated. The total number of segments extracted was 5734. After dividing the segments into a training and a test set, we trained our AdaBoost classifier. The results from the test set are shown in Table 7.1. Only 1 from 240 segments (0.42%) corresponding to the person was misclassified (false negatives), whereas 27 from 2616 segments (1.03%) not corresponding to the person were classified as people (false positives).

In a second experiment, we placed the laser in an office that contained tables, chairs, boxes, round shaped trash bins, and other furniture, creating a cluttered environment (Figure 7.4 right). An example scan taken in this environment can be

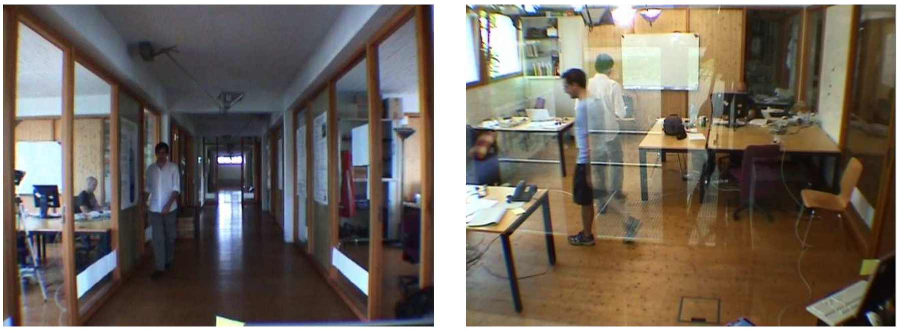


Figure 7.4: The corridor (left) and office (right) environments in which the experiments were carried out.

	Detected Label		
True Label	Person	No Person	Total examples
Person	**497** (97.45%)	**13** (2.55%)	**510**
No Person	**171** (2.73%)	**6073** (96.26%)	**6244**

Table 7.2: Confusion matrix for the office environment

shown in Figure 7.2. In this case two people were in the room during the experiment. Like in the previous experiment, the people were moving and occasionally standing still. A total of 791 scans were recorded from which we extracted 13838 segments. The segments were divided into a training and a test set and a strong classifier was learned. Although the office was cluttered with objects and furniture that strongly resemble features of legs, we still obtained an overall classification rate of 97.25%. The confusion matrix is shown in Table 7.2.

In a third experiment we created a common set of segments containing all the segments from both the corridor and the office environment. Again, the set was divided into a training and a test set. Table 7.3 shows the confusion matrix. Although the error rates slightly increase with respect to Tables 7.1 and 7.2, they still remain under 4%, which in our opinion is a fairly good level. This result demonstrates that a common classifier can be learned using both environments while still obtaining good classification rates.

	Detected Label		
True Label	Person	No Person	Total
Person	**722** (96.27%)	**28** (3.73%)	**750**
No Person	**225** (2.54%)	**8649** (99.88%)	**8860**

Table 7.3: Confusion matrix for both corridor and office environments simultaneously.

	Detected Label		
True Label	Person	No Person	Total
Person	**217** (90.42%)	**23** (9.58%)	**240**
No Person	**112** (4.28%)	**2504** (95.72%)	**2616**

Table 7.4: Results obtained in the corridor environment using the classifier learned in the office

7.4.2 Transferring the Classifiers to New Environments

In the following experiment we analyze wether a classifier learned in a particular environment can be used to successfully classify the beam observations in a new environment.

For this purpose we trained our AdaBoost-based classifier using the training set corresponding to the office environment in the previous section. We then classified the test set from the corridor scenario. Table 7.4 shows the results of this classification. As expected, the errors increase compared to the situation in which the training and the test data were from the same environment. However, the classification rates remain above 90%, which indicates that our algorithm yields good generalizations and can also be employed for people detection in new environments.

7.4.3 Comparison With a Heuristic Approach

To analyze how much can be gained by our learning approach, we compared the classification results of our AdaBoost-based classifier with the results obtained using a manually designed classifier. This classifier employs features that are typically found in the literature on laser-based people tracking. In particular, we create a classifier using the following list of heuristics:

- Jump distance between adjacent beams for local minima extraction (features #4 and #5). The threshold for both features has been set to 30 cm.

- Segment width (feature #6). In this case, local minima blobs greater than 5 cm and smaller than 50 cm are considered as people.

- Minimum number of points in the segment (feature #1). A segment with four or more points is considered as corresponding to a person.

- Motion of beams (feature #14). Two consecutive scans are aligned and beam-wise subtracted from each other. Segments that contain beams which moved more than a certain distance are classified as people. This minimal distance was set to 2 cm, close above sensor noise.

- Standard deviation as a compactness measure of a segment (feature #2). The threshold was experimentally determined and set to 0.5 meter.

	Heuristic Approach	AdaBoost
False Negatives (%)	34.67	3.73
False Positives (%)	9.06	2.54
Overall Error (%)	11.06	2.63

Table 7.5: Comparison with the heuristic approach

	Without Motion Feature	With Motion Feature
False Negatives (%)	3.73	3.47
False Positives (%)	2.54	3.13
Total Error (%)	2.63	3.15

Table 7.6: Classification errors including the motion feature

To compare the performance of our AdaBoost-based method with the previous set of heuristics, we repeated the experiment of Section 7.4.1, where segments from the corridor and office were used together as examples. We then classified the test set using both approaches. The results of the classification are shown in Table 7.5. As this table indicates, our approach yields much better results than the heuristic approach.

7.4.4 Experiments Including the Motion Feature

In the previous experiments, only the first thirteen geometrical features were used. These features were static and did not take into account changes on the observations during time.

In the experiment of this section, we added the motion feature #14 to the set of features to be fed to the boosting process. All scans from the corridor and the office runs were simultaneously used for training and classification (Section 7.4.1). The results of the classification are shown in Table 7.6. As can be seen, adding the motion feature results only in a marginal improvement over the classifier without the motion feature (Table 7.3). Although the motion feature receives relatively high weight (it is ranked as the third most informative feature), we think that this marginal improvement is simply an expression of the fact that people do not always move.

7.4.5 Best Features for People Detection

As we did in Section 3.6.5, we now look into the set of weak classifiers selected by AdaBoost to find the most important. Since each weak classifier represents a feature, this is somehow equivalent to choose the best set of features. We take into account the importance of the individual feature weights in the final strong classifier. Table 7.7 lists the five best features for the classifier trained in the corridor, office and both environments respectively. Note that sometimes the same features

Environment	Five Best Features
Corridor	9, 4, 5, 2, 4
Office	9, 13, 3, 4, 5
Both	9, 13, 4, 3, 5

Table 7.7: The best five features for each classifier

occurs more than once in a classifier, differing in their threshold or weight values.

Analyzing the Table 7.7, we can see that the most informative feature in all the environments is the the radius of the circle fitted into the segment (feature #9). This feature is an alternative estimation of the size of each segment. The mean angular difference (feature #13) is the second most important feature, quantifying the convexity of the segment. The following features in importance are the two jump distances (features #4 and #5). These two features are typically used in the literature for people detection. Finally, we found the features #2 and #3, which measure the compactness of the segment. Feature #3 seems to be preferred. The reason for this is likely to be the more robust properties of the mean absolute deviation from the median over the simple standard deviation.

7.5 Related Work

In the past, many researchers focused on the problem of tracking people in range scans. One of the most popular approach in this context is to extract legs by the detecting moving blobs that appear as local minima in the range image [Fod *et al.*, 2002; Kleinhagenbrock *et al.*, 2002; Scheutz *et al.*, 2004; Schulz *et al.*, 2003a]. To this end, two types of features have been quite popular: motion and geometry features. Motion in range data is typically identified by subtracting two subsequent scans. If the robot is moving itself, the scans have first to be aligned, e.g., using scan matching. The drawback of motion features is that only *moving* people can be found. Topp and Christensen [2005] extend the method of Schulz *et al.* [2003a] by the ability to track also people standing still, which, for instance, is useful for interaction. They report on good results in typical scenarios but also on problems in cluttered environments. They also conclude that either improved motion models or more advanced pattern detection of people are necessary.

Cui *et al.* [2005] pursue a multi-sensor approach to people tracking using multiple laser scanners at foot height and a monocular camera. After registration of the laser data, they extract moving blobs of 15 cm diameter as feet candidates. Two feet candidates at a distance of less than 50 cm are treated as a step candidate.

Geometric features have also been used by Xavier *et al.* [2005]. With a jump distance condition, they split the range image into clusters and apply a set of geometric rules to each cluster to distinguish between lines, circles and legs. A leg is defined as a circle with an additional diameter condition.

In all approaches mentioned above, neither the selection of features nor their thresholds are learned or determined other than by manual design and hand-tuning. This motivates the application of the learning technique presented in this chapter.

Finally, Haehnel *et al.* [2003] have considered the problem of identifying beams in range scans that are reflected by dynamic objects. They consider the individual beams independently and apply EM to determine, whether or not a beam has been reflected by a dynamic object such as a person. Our method, in contrast, considers groups of beams and classifies the entire groups according to their properties.

Parts of the approach presented in this chapter have been used in recent works on people detection and/or tracking. Zivkovic and Krose [2007] apply our method to detect people in 2D laser range data. The detection of people is also done using vision. Both methods are combined to create a robust people detector. In their work, Premebida *et al.* [2007] use some of the geometrical features presented in this chapter to the detection and tracking of objects in laser readings.

7.6 Conclusions

This chapter addressed the problem of adding semantic information about people in sensor readings. Our approach applies the AdaBoost algorithm to learn a robust classifier from simple features, and it identifies groups of beams that correspond to legs of people. The method has been implemented and applied in cluttered office environments. In practical experiments carried out in different environments we obtained encouraging detection rates of over 90%.

From the features selected by AdaBoost we can conclude that the shape of people in range data is best recognized by a radius feature, a convexity feature, a local minimum feature and a robust compactness feature.

Although in this chapter we concentrate only on the detection of people, we think that the approach here presented can be easily extended to add semantic information from other objects in the environment.

Chapter 8

Conceptual Spatial Representation of Indoor Environments[1]

8.1 Introduction

In this chapter we will show how the semantic classification of places is used as part of a more complex representation system in a mobile robot. In particular, we introduce our integrated system for conceptual spatial representations of indoor environments for service robots.

Recently, there has been an increasing interest in robots whose aim is to assist people in human-like environments, such as domestic or elderly care robots. In such situations, the robots will no longer be operated by trained personnel but instead have to interact with people with little or no formal training in robotics. Communication and interaction between robots and humans become key issues for these systems.

One of the most intuitive and powerful ways for humans to communicate is spoken language. It is therefore interesting to design robots that are able to speak with people and understand their words and expressions. For this, the robot needs to perceive the world similar to a human. However, when comparing the way robots typically perceive and represent the world with the findings from cognitive psychology about how humans do it, it is evident that there is a large discrepancy. Bridging the gap between human and robot spatial representations is thus of paramount importance.

In this chapter we present an integrated approach for creating conceptual representations of human-made environments using mobile robots. The concepts represent spatial and functional properties of typical indoor environments. Our model is composed of layers containing maps at different levels of abstraction as shown

[1] This chapter originated from a joint work with Hendrik Zender, Patric Jensfelt, and Geert-Jan M. Kruijff

in Figure 8.1. The lower layers contain a metric map, a navigation map and a topological map, each of which plays a role in navigation and self-localization of the robot. On the topmost level of abstraction, the conceptual map provides a richer semantic view of the spatial organization. The complete model permits the robot to do spatial categorization rather than only instantiation. Additionally, the multi-layered representation is created in a semi-supervised map acquisition process, in which a tutor communicates with the robot using spoken language.

The rest of the is organized as follows. In Section 8.2, we describe the multi-layered conceptual spatial representation. The map acquisition process is outlined in Section 8.3. In Sections 8.4, we present implementation details of the complete system. In Section 8.5, a demo is presented in which we show the capabilities of the service robot. We discuss related work in Section 8.6. Finally, we conclude in Section 8.7.

8.2 Multi-layered Conceptual Mapping

The aim of our multi-layered conceptual mapping is to generate spatial representations that enable a mobile robot to create a conceptual model of human-made environments similar to the way humans do. These concepts correspond to spatial and functional properties of typical indoor environments. Following findings in cognitive psychology [McNamara, 1986], we assume that topological areas are the basic spatial units suitable for situated interaction between humans and robots. We also proceed from the assumption that the way people refer to a place is determined by the functions people ascribe to that place.

Considering these ideas, our final representation model is divided into layers, each representing a different level of abstraction. Starting from sensory input (laser scanner and odometry), a metric map and a navigation map representing traveled routes are constructed. On the basis of detected doorways, a topological partitioning of the navigation map is maintained. The previous layers play a crucial role for the robot control systems. The conceptual map provides a conceptual abstraction of the lower layers. In the conceptual layer, spatial knowledge, innate conceptual knowledge and knowledge about entities in the world stemming from other modalities, such as vision and dialogue, are combined to allow for symbolic reasoning and situated dialogue. Figure 8.1 depicts the four layers of the conceptual spatial representation.

8.2.1 Metric Map

The first layer of our model (Figure 8.1, bottom) contains a metric representation of the environment in an absolute frame of reference. The geometric primitives of the metric map consist of lines extracted from laser range scans. Such lines typically correspond to walls and other flat structures in the environment. The complete metric map is created by a mobile robot using simultaneous localization and mapping

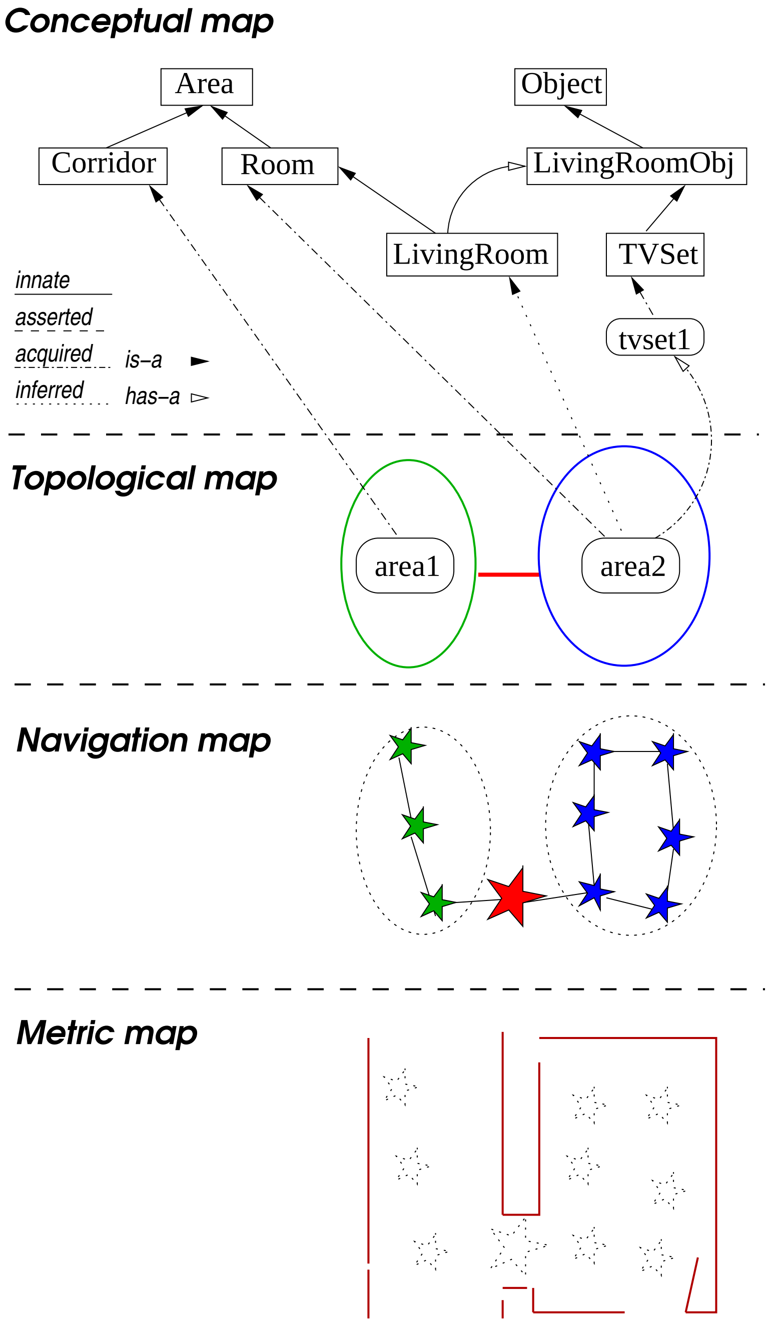


Figure 8.1: An example of a layered spatial representation for an indoor environment.

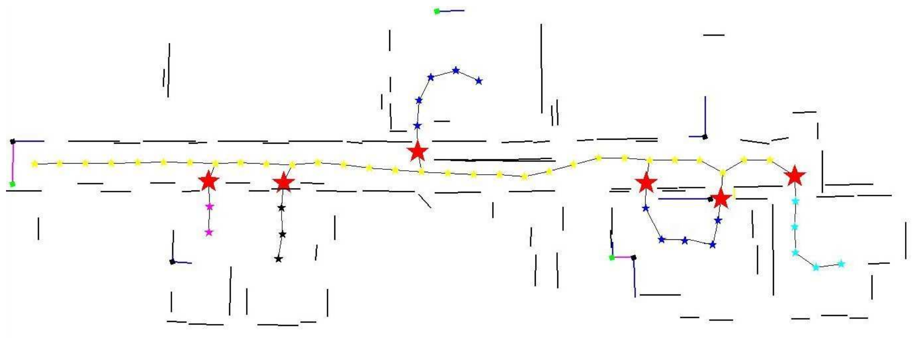


Figure 8.2: The metric map is represented by lines. The navigation map is visually represented by the stars. Different colors represent different areas separated by doors, which are marked by bigger red stars.

(SLAM) techniques. In particular, we apply the same framework as by Folkesson *et al.* [2005], which uses general representations for features that address symmetries and constraints in the feature coordinates . Furthermore, the representation allows for the features to be added to the map with partial initialization. The number of dimensions for a feature can grow with time as more information is acquired. The basis for integrating the feature observations is the extended Kalman filter (EKF) [Thrun *et al.*, 2005]. An example metric map created using this method is shown in Figure 8.2.

8.2.2 Navigation Map

The second layer contains the navigation map represented by a graph. This representation is based on the notion of a roadmap of virtual free-space markers [Latombe, 1991; Newman *et al.*, 2002]. As the robot navigates through the environment, a marker (navigation node) is dropped whenever the robot has traveled a certain distance from the closest existing marker. The graph serves for planning and autonomous navigation in the known part of the environment.

We distinguish between two kinds of navigation nodes: place nodes and doorway nodes. Doorway nodes indicate the transition between different places and represent possible doors. They are detected and added whenever the robot passes through a narrow opening. Later, the status (open/closed) of a known door can be monitored using the laser scanner. Additionally, doorway nodes are assigned information about the door opening such as width and orientation.

Each place node is classified into one of two semantic labels, namely CORRIDOR or ROOM, following the approach presented in Chapter 3. This method for semantic classification assigns a label to the pose which corresponds to the place node we want to classify. To increase the robustness of the method, we additionally classify each place node using the majority vote of the classification of the poses close to it. As explained before, a node is added to the navigation map when the distance to the previous node is greater than a threshold. We use this fact to store the classification of the last N poses of the robot between two place nodes. We then compute the majority vote of these last N poses and assign the final classification to the corresponding destination node.

8.2.3 Topological Map

The topological map divides the set of nodes in the navigation graph into different areas. An area consists of a set of interconnected nodes (Figure 8.2). In this view, the exact shape and boundaries of an area are irrelevant. The set of nodes is partitioned on the basis of the door detection mechanism explained in the previous section. This approach complies with previous studies [McNamara, 1986; Hirtle and Jonides, 1985], which state that humans segment space into regions that correspond to more or less clearly defined spatial areas.

Note that this method for topological map extraction is an alternative to the one presented in Chapter 4. Here, the approach is based mainly on the detection of doorways as the boundaries between different regions. Then the nodes in the different regions are labeled according to its semantic classification. This procedure is more appropriate for an online creation of the topological map. In contrast, the method introduced in Chapter 4 is an offline approach which uses simulated range data for the classification of the free poses in the map.

8.2.4 Conceptual Map

The conceptual map provides the link between the low-level maps and the communication system used for situated human-robot dialogue. It is also in this layer that knowledge about the environment stemming from other modalities, such as vision and dialogue, is anchored to the metric and topological maps.

Based on the work by Zender [2006], our system is endowed with a commonsense OWL ontology [Smith *et al.*, 2004] of an indoor environment. The complete ontology is shown in Figure 8.3. This ontology describes taxonomies (*is-a* relations) of room types and typical objects found therein through *has-a* relations. These conceptual taxonomies have been handcrafted and cannot be changed online. However, instances of the concepts are added to the ontology during runtime. Through fusion of *acquired* and *asserted* knowledge (as will be explained in Section 8.3), and through the use of the *innate conceptual* knowledge, a reasoner [Haarslev and Mölle, 2003] can *infer* information about the world that is neither given verbally nor actively perceived. In this way, linguistic references to spatial areas can be generated.

Acquired Knowledge

While the robot moves around constructing the metric and topological maps, our system derives higher-level knowledge from the information in these layers. Each

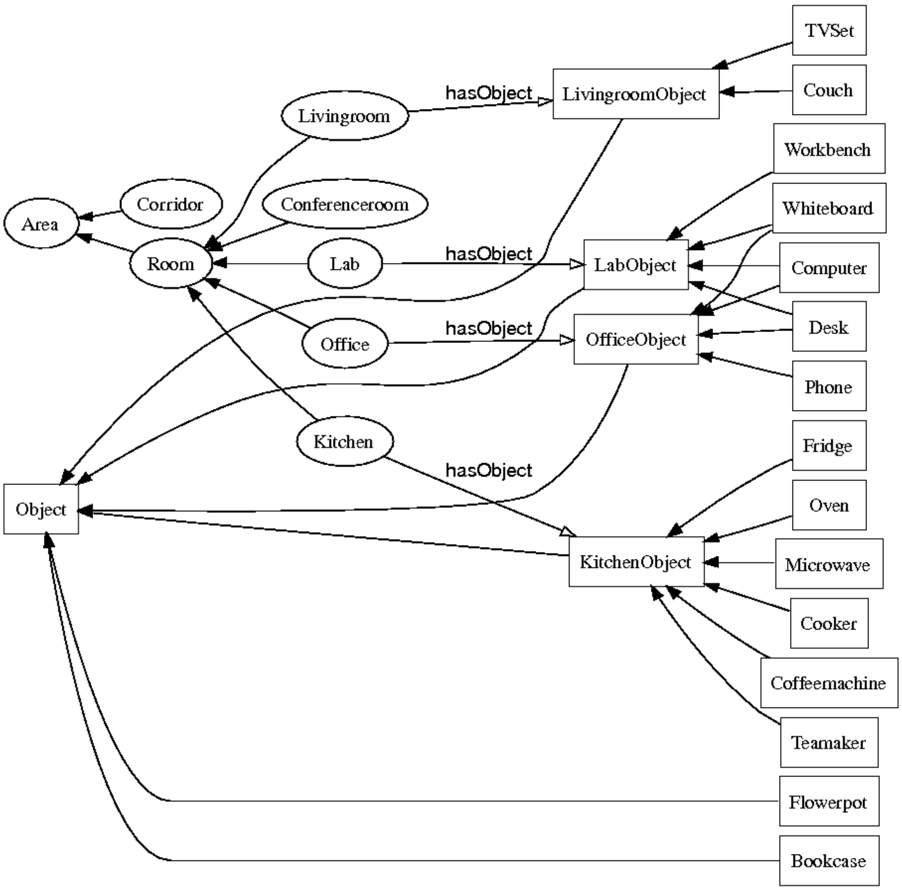


Figure 8.3: Illustration of a part of the commonsense ontology of an indoor office environment. Solid arrows denote the taxonomical `is-a` relation.

topological area, for instance, is represented in the conceptual map as an ontological instance of the type Area. Furthermore, as soon as reliable information about the semantic classification of an area is available, this is reflected in the conceptual map by assigning the area's instance a more specific type of either Room or Corridor. Information about recognized objects stemming from the vision subsystem is also represented in the conceptual map. Whenever a new object in the environment is recognized, a new instance of the object's type, e.g. Couch, is added to the ontology. Moreover, the object's instance and the instance of the area where the object is located are related via the hasObject relation. This process is shown in Fig. 8.1.

Asserted Knowledge

During a guided tour with the robot [Elin A. Topp *et al.*, 2006], the user typically names areas and certain objects that he or she believes to be relevant for the robot. Typical assertions in a guided tour include “You are in the corridor," or “This is the charging station." Any such assertion is stored in the conceptual map, either by specifying the type of the current area or by creating a new object instance of the asserted type and linking it to the area instance with the hasObject relation.

Innate Conceptual Knowledge

We have handcrafted an ontology (Fig. 8.3) that models conceptual commonsense knowledge about an indoor office environment. On the top level of the conceptual taxonomy, there are the two base concepts Area and Object. Area can be further partitioned into Room or Corridor. The basic-level subconcepts of Room are characterized by the instances of Object that are found there, as represented by the hasObject relation.

Inferred Knowledge

Based on the knowledge representation in the ontology, our system uses a description-logics based reasoning software [Haarslev and Mölle, 2003] that allows us to move beyond a pure labeling of areas. Combining and evaluating acquired and asserted knowledge within the context of the innate conceptual ontology, the reasoner can infer more specific categories for known areas. For example, combining the acquired information that a given topological area is classified as a room and contains a couch, together with the innate conceptual knowledge given in our commonsense ontology, it can be inferred that this area can be categorized as being an instance of LivingRoom. Conversely, if an area is classified as a corridor and the user shows the robot a charging station in that area, no further inference can be drawn. The most specific category the area instantiates will still be Corridor.

Our method allows for multiple possible classification of any area because the main purpose of the reasoning mechanisms in our system is to facilitate human-robot interaction. The way people refer to the same room can differ from situation to situation and from speaker to speaker, as reported by Elin A. Topp *et al.* [2006]. For example, what one speaker prefers to call the kitchen might be referred to as the recreation room by another person. Since our aim is to be able to resolve all such possible referring expressions, our method supports ambiguous classifications of areas.

8.3 Interactive Map Acquisition

The multi-layered representation is created using the combination of a user-driven supervised map acquisition process with autonomous exploration discovery by the robot. This process is based on the notion of *Human-Augmented Mapping*, as introduced by Elin A. Topp *et al.* [2006]. We additionally use a linguistic framework that actively supports the map acquisition process and is used for situated dialogue about the environment. More details about the dialog capabilities of the system are given in [Kruijff *et al.*, 2007].

The map can be acquired during a so-called guided tour scenario, in which the user shows the robot around and continuously teaches the robot new places and objects. During such a guided tour, the user can command the robot to follow him or instruct it to perform navigation tasks. Our system does not require an initial complete guided tour. It is also possible to incrementally teach the robot new places and objects at any time the user wishes. With every new piece of information, the robot's internal representations become more complete. Still, the robot can always perform actions and conduct meaningful dialogue about the aspects of its environment that are already known to it.

Whenever the user gives an assertion about areas in the environment or objects found therein, the robot updates the conceptual map with the asserted information. The concurrent constructions of the metrical map and the topological abstraction level propagate the information in a bottom-up manner. Together with the laser-based area classification, these pieces of information lead to an update of the conceptual map with acquired knowledge.

Following the approach by Kruijff *et al.* [2006], the robot can also initiate a clarification dialogue if it detects an inconsistency in its spatial representation, illustrating the mixed-initiative capabilities of the dialogue system.

8.4 System Integration

Figure 8.4 sketches the connections between the different modalities implemented in the CoSy explorer. The robot acquires information about the environment using different sensors, namely a laser range finder and a camera. This information is used for object recognition, place classification, and people tracking. All these perception components are also part of the navigation subsystem, which uses the sensors for SLAM and motion planning.

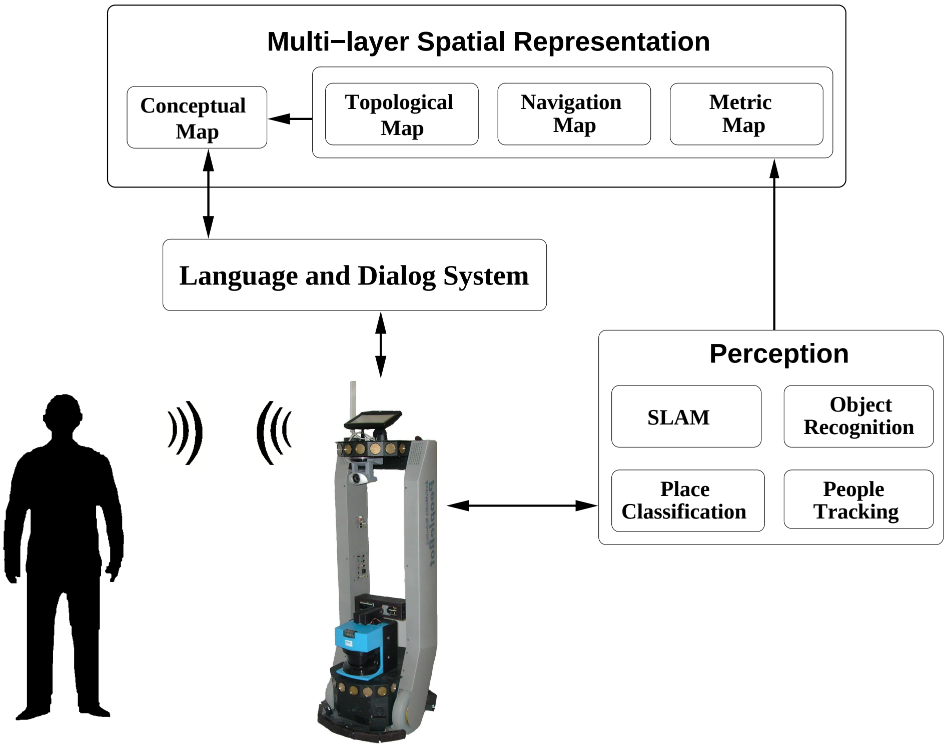


Figure 8.4: The information processing in the integrated CoSy Explorer system.

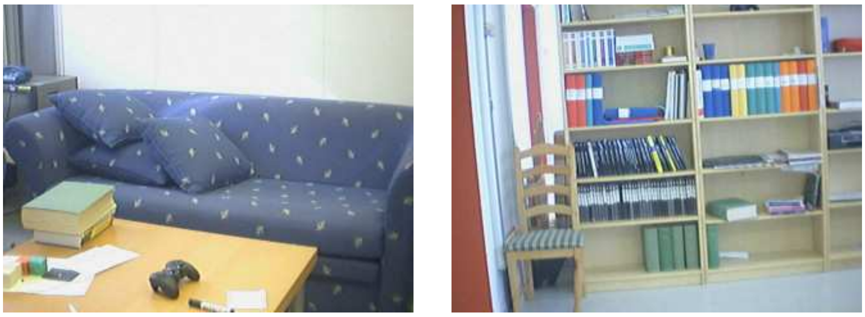


Figure 8.5: Two example objects used in the CoSy Explorer scenario. The left image shows a couch and the right image depicts a bookcase.

The information coming from the sensors is then used to create a multi-layered conceptual and spatial representation of the man-made environment the robot is acting in. Some of the information needed at the conceptual level to complete this representation is given by the user through spoken dialogues. In this case, the communication between the user and the robot supports mixed-initiative: either the user explains some concepts to the robot, or it is the robot that poses questions to the user.

The complete system was implemented and integrated in an ActivMedia PeopleBot mobile platform (robot in Figure 8.4). The robot is equipped with a SICK laser range finder, which is used for the metric map creation, people following, and for the semantic classification of places. The place classification is based on a 360^o field of view. However our robot has only one laser at the front covering a restricted 180^o field of view. To solve this problem we follow the approach described in Section 3.5 and maintain a local map around the robot which permits us to simulate the rest of the beams covering the rear part of the robot. Additionally, a camera is used only for object detection. The detection systems uses SIFT features for finding typical objects like a television set, a couch or a bookcase. We recognize instances of objects and not categories [Lowe, 2004]. The objects must be shown previously to the robot and learned by it. Examples of objects used for recognition are shown in Figure 8.5.

The communication with people was completely done using spoken language (Figure 8.4). The user can talk to the robot using a bluetooth headset and the robot replies using a set of speakers mounted on the mobile platform.

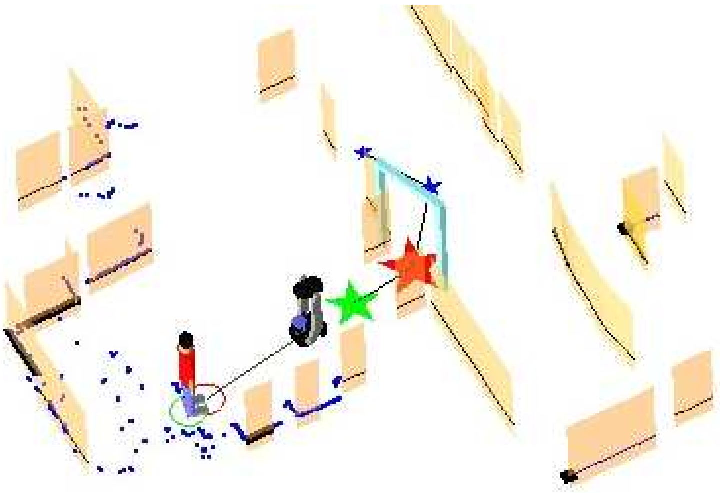
As an additional tool, we use an online viewer for the metric and navigation maps. The output of this program is composed of the lines extracted by our SLAM implementation extended to 3D planes to facilitate the visualization. The viewer shows the different nodes and edges used to construct the navigation map. Nodes corresponding to doorways are drawn bigger and with red color and with an associated doorframe as shown in Figure 8.6. Finally, the robot and the user are constantly shown in the positions where they are localized. The localization of the robot is calculated using SLAM [Folkesson *et al.*, 2005], while the pose of the person is estimated using people tracking methods based only on laser readings [Schulz *et al.*, 2003b].

The robot, being equipped with an onboard computer (850 MHz) connected to two built-in loudspeakers, runs the Player software [Gerkey *et al.*, 2003] for control and access of the hardware, and the Festival speech synthesis software [Clark *et al.*, 2004]. The rest of the system runs on five laptops (1.8 GHz) interconnected using a wireless network. The first laptop is placed aboard the robot platform. It is connected to the onboard computer via an Ethernet crossover cable and to the rest of the system using its wireless adapter. This laptop runs the software for navigation, SLAM and people tracking. A second laptop runs the Windows operating system and is used for the real time speech recognition [Nuance, 1999]. It is also placed on the robot platform in order to ensure a reliable bluetooth connection to the headset that recorded the user's voice commands. The recognized speech strings are sent to a third laptop, which runs the real-time dialogue processing and conceptual mapping subsystems. The fourth computer constantly classifies the current pose of the robot into a semantic class based on laser data. The last computer handles the viewer tool for debugging purposes. The communication between the different processes is established in a mixed environment using TCP/IP sockets and an OAA framework [Cheyer and Martin, 2001]. Fewer computers could have been used, but the setup was convenient as it allowed each subsystem developer to have his own computer.

8.5 Demo

In order to show all the functionalities explained in the previous sections, we carried out a demo at the 7th floor of the CAS building at the Royal Institute of Technology in Stockholm. In this demo the robot, together with a user, goes through different situations (or episodes) along the environment. The complete demo was carried out non-stop, i.e. we did not stop the robot or restart the system at any moment. The duration of the complete experiment was of approximately 6 minutes. Each of the episodes is explained in detail in the next sections and a video is available at the CoSy project website [CoSy, 2004] under the explorer scenario. The demo was thought of as a test, and for this reason we forced some artificial situations to simulate possible real ones (e.g. the false doorway in Section 8.5.2). A similar experiment was carried out in which the robot interacts constantly with the user and the environment for more than 30 minutes during a different demo in the CoSy project [CoSy, 2004]. In this case, the robot was presented to an audience while explaining its actions. Some of the episodes were repeated to clarify some questions. The robot again run with no interruptions or system problems. This led us to think that our implementation is quite robust and maybe can serve as basis for a long term service robot.

The idea of the demo is to show how the robot learns its environment while interacting with a tutor. However, some previous knowledge is needed during this



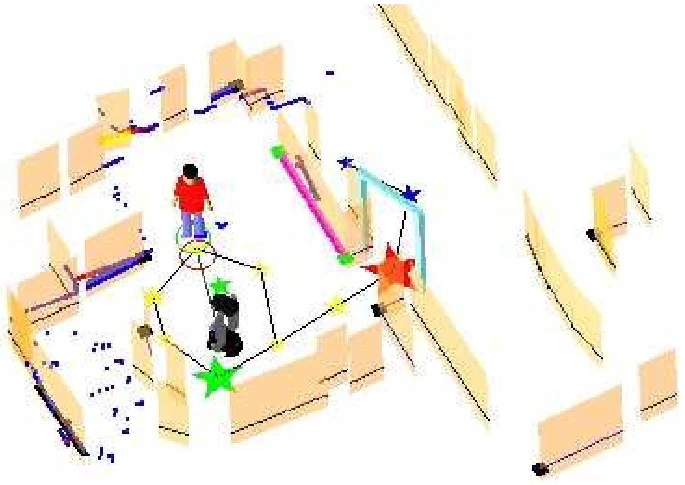


Figure 8.6: Snapshots of the online viewer using during the experiment. The stars indicate the nodes in the navigation map. Small and blue for corridor, small and yellow for room, big and red for doorways and medium and green for the actual position of the robot. Additionally, lines are extended to 3D planes and simulated doorways are drawn for facilitating the visualization. The person is drawn in the position detected by the people following software. In the top image the robot enters room after detecting a doorway. In the bottom image the complete map of the room is created using lines.

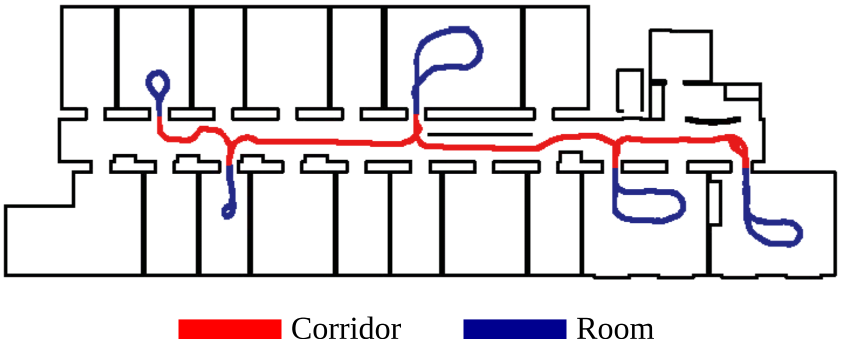


Figure 8.7: Trajectory followed by the robot to train the classifier for distinguishing between corridor and room. The different places are depicted with distinct colors.

process. First, the robot needs an ontology representing the general knowledge about the environment. For this purpose, we use the ontology depicted in Figure 8.3. Furthermore, the classification of places is based on previous general knowledge about the geometry of rooms and corridors, which is encoded in a classifier based on laser readings as explained in Section 8.2.2. The classifier is trained using examples of corridors and rooms from real environments as the one shown in Figure 8.7. These two kinds of knowledge are independent of the environment used for testing, in the sense that the robot does not need to be physically present in the test environment to acquire the information. Finally, the robot has to recognize different objects, such as couches or TV sets, using vision (see Figure 8.5 for some examples). Because we do instance recognition rather than categorization, the objects we want to recognize must be presented to the robot before running the experiment. For this purpose, we position the robot in front of these objects, acquire a training image and label it with the corresponding term, which is added to a small database of objects and also included in the language systems for its posterior use.

We follow explain the different episodes contained in the complete demo.

8.5.1 Episode 1: Waking Up

The demo starts in the corridor, where the robot is positioned close to the charging station. The user activates the robot and tells it that it is located at the charging station (Figure 8.8). The user then asks the robot to follow him. The robot drops markers (navigation nodes), which are classified as corridor. Then the person followed by the robot enters a room through a doorway. The door is recognized and the corresponding node is set. From this point the next nodes will be classified as a new area and correctly labeled as room.

Figure 8.8: The user wakes up the robot and the demos starts.

8.5.2 Episode 2: Clarification Dialogues

In this episode we want to show the utility of the clarification dialogues. As explained in Section 8.2.2, our door detection is simply based on detecting when the robot passes through a narrow opening. However, this alone will still lead to some false doors in cluttered rooms. Assuming that there are few false negatives in the detection of doors, we get great improvements by enforcing that it is not possible to change room without passing through a door. For example, while moving around in a room the robot may detect a narrow passage and falsely assume that a door was passed, putting a door label on that particular node. The robot continues to move around in the room and eventually reaches the nodes from before adding the false door. These nodes will then have different room labels, that is, the room has changed without passing a door. If this happens, an inconsistency is found and a clarification dialogue with the user is triggered.

To test the former situation we put a bucket close to a table in the room creating an illusion of a doorway when using only the laser as sensor. The robot passes through this false doorway and comes back to a previously visited node. At this point the robot infers that there is an inconsistency in the map and initializes a clarification dialogue asking if there was a door previously. The user denies this fact and the map is updated accordingly. A more detailed explanation of the complete process of clarification dialogues for a similar situation is presented by Kruijff *et al.* [2006].

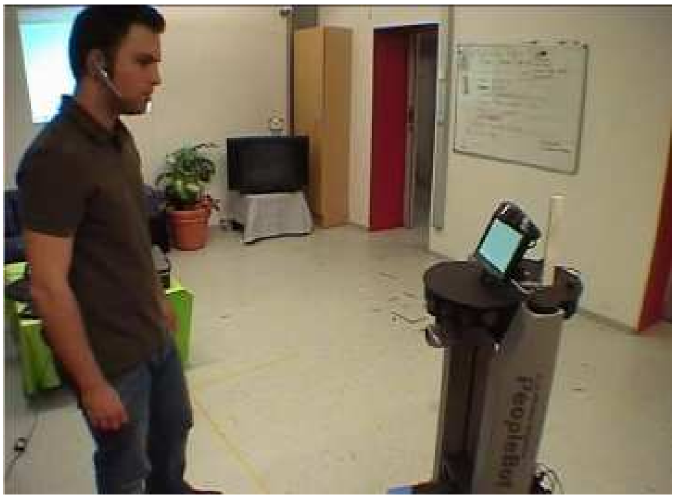


Figure 8.9: The user asks the robot: “Where is the charging station?".

8.5.3 Episode 3: Inferring New Concepts

In this episode we test how the robot infers new categorizations of places when discovering new objects. The goal is to use our SIFT-based object detector together with the laser-based place classification to detect simple objects and places. Then, using the inference on the office ontology as explained in Section 8.2.4, the robot is able to come up with more specific concepts.

While staying in the room, the robot is asked for the current place and it answers with the indefinite description “a room", which is inferred from the navigation nodes in the area. A majority vote among the nodes in the area is used in case the node classification is not unanimous. Then the robot is asked to look around. This command activates the vision-based object detection capabilities of the robot. The robot moves and detects a couch, and then a television set. After that, the user asks the robot for the name of the place. Because of the inference over the detected objects and places, the robot categorizes the place as a `Livingroom`. Note that previous to the detection of objects the same place was categorized as a `Room`. As a further test of the robot’s classification it is asked where the charging station is located and correctly answers “it is in a corridor” (Figure 8.9).

8.5.4 Episode 4: Going to Objects

Finally, we show how the navigation map is used by the robot to come back to previously visited places.

After the door opening situation, the robot is asked to go to the television. The robot then navigates to the node where the television was observe (Figure 8.10) This functionality permits the user to command the robot to places without the need of giving concrete coordinates. It is also more powerful in the sense that the

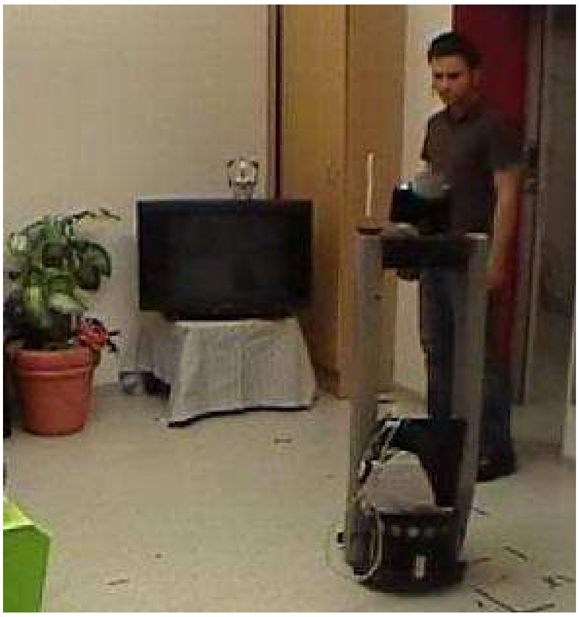


Figure 8.10: Following the order “go to the television", the robots approaches the navigation node from where it saw the television the last time.

user may not know the concrete name of the place, but he can remember it as ‘the room with a television". After that, the robot is commanded to go to the charging station. Again the robot follows the navigation map until it positions itself on the station, thus finishing the experiment.

8.6 Related Work

Several approaches on mobile robotics extend metric maps of indoor environments with semantic information. The work by Diosi *et al.* [2005] creates a metric map through a guided tour. The map is then segmented according to the labels given by the instructor. Friedman *et al.* [2007] use *Voronoi Random Fields* for extracting the topologies.

Research in spatial representations has yielded different multi-layered environment models. Vasudevan *et al.* [2006] suggest a hierarchical probabilistic representation of space based on objects. The work by Galindo *et al.* [2005] presents an approach containing two parallel hierarchies, spatial and conceptual, connected through anchoring. Inference about places is based on objects found in them. Furthermore, the *Hybrid Spatial Semantic Hierarchy* (HSSH) is introduced by Beeson *et al.* [2007]. This representation allows a mobile robot to describe the world using different representations each with its own ontology. Compared to these approaches our implementation uses human augmented mapping for collecting information. The communication with the robot is made entirely using natural language and dialogues. Moreover our conceptual representation comes from the fusion of acquired, asserted, and both inferred and innate knowledge.

There are more cognitively inspired approaches to robot navigation for conveying route descriptions from a technically naive user to a mobile robot. These approaches need not necessarily rely on an exact global self-localization, but rather require the execution of a sequence of strictly local, well-defined behaviors in order to iteratively reach a target position. Kuipers [2000] presents the *Spatial Semantic Hierarchy* (SSH). Alternatively, the *Route Graph* model is introduced by Krieg-Brückner *et al.* [1998]. Both theories propose a cognitively inspired multi-layered representation of the *map in the head*, which is at the same time suitable for robot navigation. Their central layer of abstraction is the topological map. Our approach differs in that it provides an abstraction layer that can be used for reference resolution of topological entities.

A number of systems have been implemented that permit a robot to interact with humans in their environment. Rhino Burgard *et al.* [2000b] and Robox Siegwart *et al.* [2003] are robots that work as tour-guides in museums. Both robots rely on an accurate metric representation of the environment and use limited dialogue to communicate with people. The robot BIRON Spexard *et al.* [2006] is endowed with a system that integrates spoken dialogue and visual localization capabilities on a robotic platform similar to ours. This system differs from ours in the degree to which conceptual spatial knowledge and linguistic meaning are grounded in, and contribute to, situational awareness.

8.7 Conclusions

We presented an integrated approach for creating conceptual representations of human-made environments where the concepts represent spatial and functional properties of typical office indoor environments. Our representation is based on multiple maps at different levels of abstraction. The complete system was integrated and tested in a service robot which includes a linguistic framework with capabilites for situated dialogue and map acquisition. The experiments show that our system is able to provide a high level of human-robot communication and certain degree of social behavior.

Chapter 9

Conclusions

The work presented in this thesis presented different approaches for adding semantic information to the representations of indoor environments. We concentrated on extending the information on the maps created by a mobile robot with labels that represent different places in the environment. Each of these places has a different functionality, such as corridors, offices or kitchens.

Throughout this dissertation we have seen how the semantic information about places can improve the capabilities of mobile robots in different domains including human-robot interaction, localization, and exploration.

We first presented a technique based on supervised learning that enables a mobile robot to recognize the different places in an indoor environment. To carry out this classification the robot must first take observations and then extract some features from them. These features will be used later to recognize the different places. As main observations we have used the range measurements of laser finders, from which several features were extracted that encoded their geometrical properties.

The learning method used for classifying the different places was based on the AdaBoost algorithm. The input for the algorithm were the features extracted from the observations, and as output we obtained a strong classifier which included the more informative features for each place.

The geometrical features are quite good candidates for generalization, since they encode space information. We saw in Chapters 3 to 5 that the strong classifier created with geometrical features could successfully be transferred among different environments. The main reason is that indoor environments usually contain the same type of places, as for instance, corridors, doorways and rooms. Additionally, these places share similar structures between the different indoor environments: corridors are typically elongated, and rooms are usually more compact and cluttered. These common characteristics permit the robot to learn a classifier in one environment and recognize the same places in different ones.

Furthermore, we used vision sensors to increase the number of places to classify. The main problem with vision observations was to select the features that maintained a good generalization in the classifier. We opted for counting the number of specific objects that appear in a panoramic image taken by the robot. The selection of these features was motivated by the fact that typical objects appear at different places with different probabilities. For example, the probability of finding a computer monitor in an office is larger than finding one in a kitchen. Again these features are usually very common in several indoor environments.

The previous approach for semantic classification was used to classify the pose of a mobile robot using the laser and image-based features. However, this method did not take into account the classification of neighboring poses. To include this information, we extended the approach with some probabilistic techniques. We first smoothed the classification of all poses in an environment using probabilistic relaxation and instance-based associative Markov networks. Both approaches improved the final classification using neighboring information, which allowed the robot to extract compact regions of the environment and create a topological map.

Mobile robots are dynamic agents that move along different trajectories. When operating in indoor environments, the robots usually have a moderate velocity and a relatively continuous movement. That means, that observations obtained by a mobile robot at nearby poses are typically very similar. Furthermore, certain transitions between classes in a trajectory are rather unlikely. For example, to go from the kitchen to the office the robot sholud traverse a doorway first. This transitional information was encoded in a hidden Markov model and successfully applied to smooth the classification of the poses of the mobile robot along a trajectory. Some nice examples results using this approach were presented in Chapter 5.

As we stated above, the semantic information about places can improve other typical robotics tasks. The main idea is that mobile robots can use the intrinsic information of human-made environments to improve their actions. In particular, we showed how the information about places could improved the performance of a team of mobile robots during exploration. The results of the experiments in Chapter 6 demonstrated that places as corridors are better exploration targets as they lead to other rooms.

Another typical problem is the localization of mobile robots. In this problem the robot must determine its pose relative to a given map. Recognizing the type of place the robot is located at can be seen as a high level localization. If the robot is in an office, then other places can be discarded, and the robot can concentrate on selecting the poses that belong only to the offices. This idea was presented in Chapter 6 together with experiments that corroborated its usefulness.

The semantic information can also represent other kind of objects in the environment and not only places. In Chapter 7, we presented an approach to include the semantic labels directly to the beams of a laser range finder. In this way much richer information is available from the sensor.

Since one of the main goals of the semantic labeling is to share terms, such as corridor or office, with humans, it seems necessary to develop robotic systems that can communicate these concepts to humans. In Chapter 8, we introduced an integrated system for conceptual representations of indoor environments. This system included a linguistic framework with capabilites for situated dialogue and map acquisition. The different episodes during the demo demonstrated the interaction capabilities of this system, and how a high level conceptual representation could be create based on language communication and semantic information about places.

We think that the work presented in this thesis is one of the first covering so many different aspects relating semantic information about places using mobile robots. As we described in the related literature of several chapters, different posterior works have applied and extended some of the ideas here presented. This indicates that a lot of work can still be done.

It would be interesting, for example, to move from the supervised approach presented in this work to other methods with less supervision. One possibility could be to use semi-supervised techniques, in which the robot can initially create an first classification of the environment autonomously. This classification can be corrected later on by the user. In this way, the user does not need to indicate the label of each place the robot is located at. A second possibility could be to leave the robot create a totally unsupervised classification of the places.

In any case, it seems that information coming from the user is important, since someone has to decide how to name the different places. This last issue bring us to the problem of personalization: people can describe the same place with different terms. For example, what for a person is a living room, can be a sitting room for another. It could be interesting to study approaches able to cope with this flexibility.

The work presented in this thesis concentrated in indoor environments. However, in the last years, outdoor robots have received an increasing interest. Some initial works try to classify the different outdoors environments, but the complexity here is quite high. One of the main problems is that range observations give little information about places in outdoor environments. For this reason, approaches are moving to vision to do this classification. Semantic information can refer not only to places and objects, but also to situations. People crossing a road, or cars overtaking, are examples of situations which can be labeled with semantic terms.

To conclude, we think that the semantic labeling is a research area which can have a high impact in the future of mobile robotics.

Appendix A

Simple Features for Place Classification

In this appendix we give the mathematical formulation of the simple features used for place classification. We also include the ones from my previous work in [Mozos, 2004] so that the reader can have the complete set of definitions.

A.1 Simple Features Extracted from Laser Beams

We use $z = \{b_0, \ldots, b_{M-1}\}$ to define the set of beams taken as one observation. Each beam b_i is represented by a tuple (α_i, d_i), where α_i is the angle of the beam relative to the robot and d_i is the length of the beam.

A.1.1 Average Difference Between the Length of Two Consecutive Beams

The average difference between the length of two consecutive beams f_{average} is defined as

$$f_{\text{average}} = \frac{1}{M} \sum_{i=0}^{M-1} \left| d_i - d_{[(i+1) \mod M]} \right| . \tag{A.1}$$

A.1.2 Standard Deviation of the Difference Between the Length of Two Consecutive Beams

The standard deviation of the difference between the length of consecutive beams f_{std} is defined as

$$f_{\text{std}} = \sqrt{\frac{1}{M-1} \sum_{i=0}^{M-1} \left(\left| d_i - d_{[(i+1) \mod M]} \right| - f_{\text{average}} \right)^2} , \tag{A.2}$$

where f_{average} is the feature defined in Equation (A.1).

A.1.3 Average Difference Between the Length of Consecutive Beams Considering Max-Range

The value max-range is a threshold θ indicating the maximum length d_i for a beam. Using this θ, we define the function $\text{length}_\theta(b_i)$ as follows:

$$\text{length}_\theta(b_i) = \begin{cases} d_i & \text{if } d_i \leq \theta \\ \theta & \text{otherwise}, \end{cases} \tag{A.3}$$

The feature representing the average difference between the length of two consecutive beams using max-range $f_{\text{average},\theta}$ is then defined as

$$f_{\text{average},\theta} = \frac{1}{M} \sum_{i=0}^{M-1} \left| \text{length}_\theta(b_i) - \text{length}_\theta(b_{[(i+1) \mod M]}) \right| . \tag{A.4}$$

A.1.4 Standard Deviation of the Difference Between the Length of Two Consecutive Beams Considering Max-Range

The standard deviation of the difference between the length of two consecutive using max-range $f_{\text{std},\theta}$ is defined as

$$f_{\text{std},\theta} = \sqrt{\frac{1}{M-1} \sum_{i=0}^{M-1} \left(\left| \text{length}_\theta(b_i) - \text{length}_\theta(b_{[(i+1) \mod M]}) \right| - f_{average,\theta} \right)^2}, \tag{A.5}$$

where $f_{\text{average},\theta}$ is the feature defined in A.1.3.

A.1.5 The Average Beam Length

The average beam length $f_{\overline{d}}$ is defined as

$$f_{\overline{d}} = \frac{1}{M} \sum_{i=0}^{M-1} d_i . \tag{A.6}$$

A.1.6 The Standard Deviation of the Beam Length

The standard deviation f_σ of the beam length is defined as

$$f_\sigma = \sqrt{\frac{1}{M-1} \sum_{i=0}^{M-1} \left(d_i - \overline{d} \right)^2}, \tag{A.7}$$

where $f_{\overline{d}}$ is the feature defined in Equation (A.6).

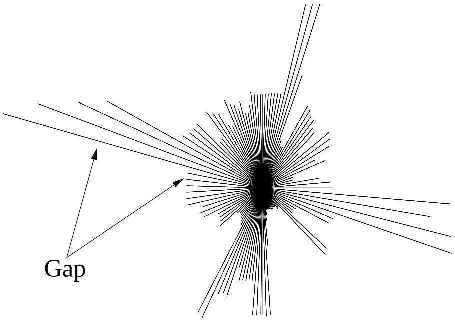


Figure A.1: Example of a gap in a laser scan.

A.1.7 Number of Gaps

Two consecutive beams build a gap if their length difference is greater than a given threshold θ. An example of a gap is shown in Figure A.1. Formally, we define a a gap with threshold θ as

$$\mathrm{gap}_\theta(b_i, b_j) = \begin{cases} 1 & \text{if } \left|d_i - d_j\right| > \theta \\ 0 & \text{otherwise} . \end{cases} \tag{A.8}$$

The feature $f_{\mathrm{gaps},\theta}$ representing the total number of gaps is calculated as

$$f_{\mathrm{gaps},\theta} = \sum_{i=0}^{M-1} \mathrm{gap}_\theta(b_i, b_{[(i+1) \mod M]}) . \tag{A.9}$$

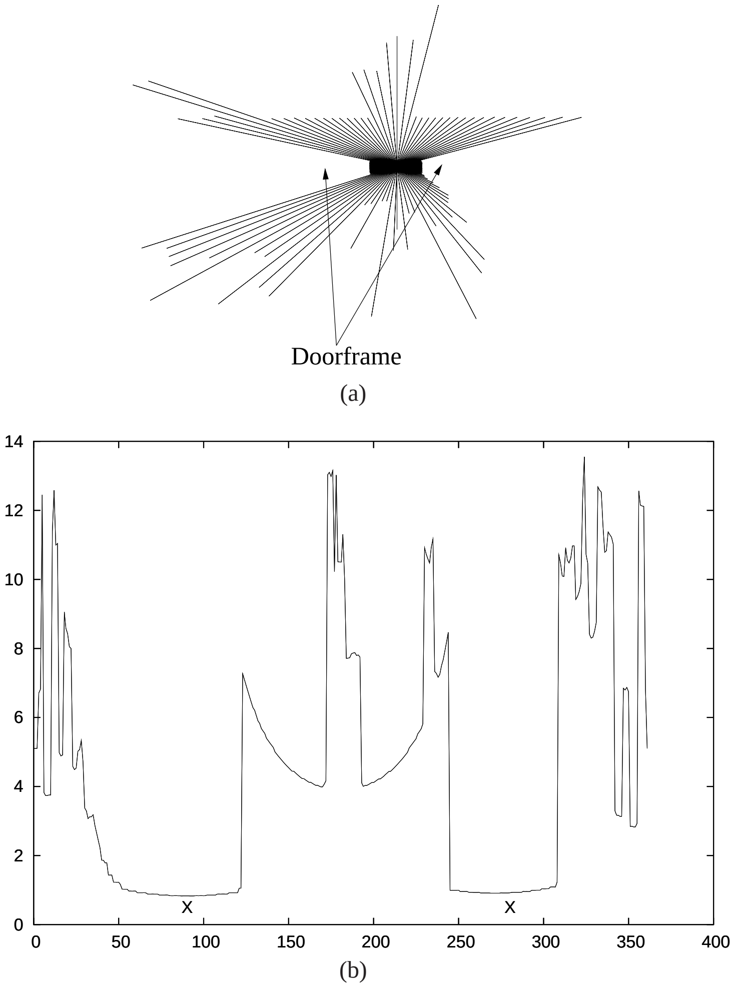
A.1.8 Number of Beams Lying on Lines Extracted from the Range

This feature is calculated using the method by Sack and Burgard [Sack and Burgard, 2004].

A.1.9 Euclidean Distance Between the Two Points Corresponding to Two Consecutive Global Minima

This feature was designed to help in the classification of doors. If we plot the length of the beams of an observation z, we obtain a graph like the one shown in Figure A.2(b). We can look in this graph for two global minima $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$ which can be the representation of a doorframe (Figure A.2(a)). The Euclidean distance between p_1 and p_2 is then calculated as

$$f_{\mathrm{distance-minima}} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} . \tag{A.10}$$



(a)

(b)

Figure A.2: Representation of a set of beams collected where the robot were in a door. In Figure A.2(a) the arrows indicate the two minima which can indicate a doorframe. Figure A.2(b) is a plot of the length of the beams. The two marks **X** indicate the same two minima (p_1 and p_2 in the text).

A.1.10 The Angular Distance Between the Two Points Corresponding to Two Consecutive Global Minima

The two points corresponding to the global minima p_1, p_2 calculated in Section A.1.9 correspond to the end of two beams, i.e., b_1 and b_2. The angular distance between these two beams is used as a feature, in the form:

$$f_{\alpha-\text{minima}} = |\alpha_1 - \alpha_2| \; . \tag{A.11}$$

A.1.11 Average of the Relation Between Two Consecutive Beams

The average relation between the length of two consecutive beams $f_{\text{average-rel}}$ is defined as

$$f_{\text{average-rel}} = \frac{1}{M} \sum_{i=0}^{M-1} \left| \frac{d_i}{d_{[(i+1) \mod M]}} \right| \; . \tag{A.12}$$

A.1.12 Standard Deviation of the Relation Between the Length of Two Consecutive Beams

The standard deviation of the difference between the length of consecutive beams $f_{\text{std-rel}}$ is defined as

$$f_{\text{std-rel}} = \sqrt{\frac{1}{M-1} \sum_{i=0}^{M-1} \left(\left| \frac{d_i}{d_{[(i+1) \mod M]}} \right| - f_{\text{average-rel}} \right)^2} \; , \tag{A.13}$$

where $f_{\text{average-rel}}$ is the feature defined in Equation (A.12).

A.1.13 Average of Normalized Beam Length

The average normalized beam length $f_{\text{average-norm}}$ is defined as

$$f_{\text{average-norm}} = \frac{1}{M} \sum_{i=0}^{M-1} \left| \frac{d_i}{d_{max}} \right| \; , \tag{A.14}$$

where d_{max} corresponds to

$$d_{max} = \max d_i \; . \tag{A.15}$$

A.1.14 Standard Deviation of Normalized Beam Length

The standard deviation of the normalized beam length $f_{\text{std-norm}}$ is defined as

$$f_{\text{std-norm}} = \frac{1}{M} \sum_{i=0}^{M-1} \sqrt{\left(\frac{d_i}{d_{max}} - f_{\text{average-norm}} \right)^2} \; , \tag{A.16}$$

where $f_{\text{average-norm}}$ corresponds to the feature defined is Equation (A.14).

A.1.15 Number of Relative Gaps

Two consecutive beams build a relative gap if the relation between their lengths is greater than a given threshold θ. Formally, we define a relative gap with threshold θ as

$$\text{rgap}_\theta(b_i, b_j) = \begin{cases} 1 & \text{if } \left|\frac{d_i}{d_j}\right| > \theta \\ 0 & \text{otherwise}\,. \end{cases} \tag{A.17}$$

The feature $f_{\text{rgaps},\theta}$ representing the total number of gaps is calculated as

$$f_{\text{rgaps},\theta} = \sum_{i=0}^{M-1} \text{rgap}_\theta(b_i, b_{[(i+1) \mod M]})\,. \tag{A.18}$$

A.1.16 Kurtosis

We define the kurtosis $f_{kurtosis}$ of a scan as

$$f_{kurtosis} = \frac{\sum_{i=0}^{M} \left(d_i - f_{\overline{d}}\right)^4}{M \cdot f_\sigma^4} - 3\,, \tag{A.19}$$

where $f_{\overline{d}}$ and f_σ are the features defined in Equations (A.6) and (A.7) respectively.

A.2 Simple Features Extracted from a Polygon Approximation

This set of features is calculated from a polygonal approximation $P(z)$ of the area covered by the observation $z = \{b_0, \ldots, b_{M-1}\}$. The vertices v_i of the closed polygon $P(z)$ correspond to the coordinates of the end-points of each beam b_i of z relative to the robot

$$P(z) \quad = \quad \{v_0, \ldots, v_{M-1}, v_M = v_0\}\,, \tag{A.20}$$

where $v_i = (x_i, y_i)$ with $x_i = d_i \cos \alpha_i$ and $y_i = d_i \sin \alpha_i$.

A.2.1 Area of $P(z)$

The area of the polygon approximation $P(z)$ is given by

$$f_{\text{Area}} = \frac{1}{2} \sum_{i=0}^{M-1} (x_i y_{i+1} - x_{i+1} y_i)\,. \tag{A.21}$$

A.2.2 Perimeter of $P(z)$

The perimeter of the polygon approximation $P(z)$ is given by

$$f_{\text{Perimeter}} = \sum_{i=0}^{M-1} \text{dist}(v_i, v_{i+1}), \tag{A.22}$$

where

$$\text{dist}(v_i, v_{i+1}) = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}\,. \tag{A.23}$$

A.2.3 Mean Distance Between the Centroid and the Shape Boundary

The centroid $c = (c_x, c_y)$ of $P(z)$ is defined as

$$c_x = \frac{1}{6 \cdot f_{\text{Area}}} \sum_{i=0}^{M-1} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i), \tag{A.24}$$

$$c_y = \frac{1}{6 \cdot f_{\text{Area}}} \sum_{i=0}^{M-1} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i)\,. \tag{A.25}$$

The mean distance between the centroid and the shape boundary of $P(z)$ is calculated as

$$f_{\text{mean-shape}} = \frac{1}{M} \sum_{i=0}^{M-1} \text{dist}(v_i, c), \tag{A.26}$$

where

$$\text{dist}(v_i, c) = \sqrt{(x_i - c_x)^2 + (y_i - c_y)^2} \tag{A.27}$$

A.2.4 Standard Deviation of the Distances Between the Centroid and the Shape Boundary

The standard deviation of the distances between the centroid and the shape boundary of $P(z)$ is given by

$$f_{\text{std-shape}} = \frac{1}{M} \sum_{i=0}^{M-1} \text{dist}(v_i, c) - f_{\text{mean-shape}}\,, \tag{A.28}$$

where $f_{\text{mean-shape}}$ and $dist(v_i, c)$ are defined in Equation (A.26) and Equation (A.27) respectively.

A.2.5 Invariant Descriptors Based on the Fourier Transformation

To calculate the Fourier coefficients we transform each vertex $v_i \in \mathbb{R}^2$ of $P(z)$ into a complex number $\tilde{v}_i \in \mathbb{C}$ in the form [Burkhard, 2004]:

$$v_i = (x_i, y_i), \Longrightarrow \tilde{v}_i = x_i + y_i j, \quad j = \sqrt{-1}\,. \tag{A.29}$$

The Fourier coefficients $\{c_{-n}, \ldots, c_{-1}, c_0, c_1, \ldots, c_n\}$ of $P(z)$ are then calculated as

$$\begin{aligned} c_0 &= \frac{1}{2T}\sum_{k=0}^{M-1}(\tilde{v}_k + \tilde{v}_{k+1}) \mid \triangle\tilde{v}_k \mid \qquad &(A.30)\\ c_n &= \frac{T}{(2\pi n)^2}\sum_{k=0}^{M-1}(\triangle s_{k+1} - \triangle s_k)e^{-j\pi(\frac{2\pi}{T})t_k}\,, \qquad &(A.31)\end{aligned}$$

with

$$\begin{aligned} \triangle\tilde{v}_i &= \tilde{v}_{i+1} - \tilde{v}_i \qquad &(A.32)\\ \triangle s_i &= \triangle\tilde{v}_i / |\triangle\tilde{v}_i| \qquad &(A.33)\\ t_k &= \sum_{i=0}^{k-1}|\triangle\tilde{v}_i| \quad k > 0, t_0 = 0 \qquad &(A.34)\\ T &= \text{perimeter of } P(z)\,. \qquad &(A.35)\end{aligned}$$

The Fourier descriptors $\{\tilde{x}_{-n}, \ldots, \tilde{x}_0, \ldots, \tilde{x}_n\}$, which are invariant to similarity, that is, translation, rotation and scale, are calculated as

$$\begin{aligned} &\left\{\tilde{x}_n := \tfrac{|c_n|}{|c_1|}e^{j(\Phi_n+(1-n)\Phi_2-(2-n)\Phi_1)}\right\} \qquad &(A.36)\\ &\Phi_n = \text{phase of } c_n\,. \qquad &(A.37)\end{aligned}$$

A.2.6 Major Axis Ma of the Ellipse that Approximates $P(z)$

Having the first two Fourier coefficients (c_{-1}, c_1) of $P(z)$ (see Section A.2.5), we can calculate the major axis of an ellipse that approximates the polygon $P(z)$ as

$$f_{\mathrm{Ma}} = |c_1| + |c_{-1}|\,. \tag{A.38}$$

A.2.7 Minor Axis Mi of the Ellipse that Approximates $P(z)$

Having the first two Fourier coefficients (c_{-1}, c_1) of $P(z)$ (see Section A.2.5), we can calculate the minor axis of an ellipse that approximates the polygon $P(z)$ as

$$\mathrm{Mi} = ||c_1| - |c_{-1}||\,. \tag{A.39}$$

A.2.8 Invariant Moments of $P(z)$

The central moments μ_{pq} of $P(z)$ up to three are:

$$\mu_{10} = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} (x_i - \overline{x})^1 (y_j - \overline{y})^0 \,, \tag{A.40}$$

$$\mu_{01} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^0 (y_j - \overline{y})^1 \,, \tag{A.41}$$

$$\mu_{11} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^1 (y_j - \overline{y})^1 \,, \tag{A.42}$$

$$\mu_{20} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^2 (y_j - \overline{y})^0 \,, \tag{A.43}$$

$$\mu_{02} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^0 (y_j - \overline{y})^2 \,, \tag{A.44}$$

$$\mu_{30} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^3 (y_j - \overline{y})^0 \,, \tag{A.45}$$

$$\mu_{03} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^0 (y_j - \overline{y})^3 \,, \tag{A.46}$$

$$\mu_{12} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^1 (y_j - \overline{y})^2 \,, \tag{A.47}$$

$$\mu_{21} = \sum_{i=1}^{M-1} \sum_{j=1}^{M-1} (x_i - \overline{x})^2 (y_j - \overline{y})^1 \,, \tag{A.48}$$

with

$$\overline{x} = \frac{1}{M} \sum_{i=1}^{M-1} x_i \,, \tag{A.49}$$

and

$$\overline{y} = \frac{1}{M} \sum_{j=1}^{M-1} y_j \,, \tag{A.50}$$

The normalized central moments of $P(z)$, denoted η_{pq}, are defined as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \,, \tag{A.51}$$

where

$$\gamma = \frac{p+q}{2} + 1\,, \tag{A.52}$$

for $p + q = 2, 3, \ldots$.

A set of seven invariant moments with respect translation, rotation and scale can be derived from the second and third moments as

$$\phi_1 = \eta_{20} + \eta_{02}\,, \tag{A.53}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2\,, \tag{A.54}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2\,, \tag{A.55}$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2\,, \tag{A.56}$$

$$\begin{aligned}\phi_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] \\ & +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\left[3(\eta_{30} - \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]\,,\end{aligned} \tag{A.57}$$

$$\begin{aligned}\phi_6 = {} & (\eta_{20} - \eta_{02})\left[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right] \\ & +4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})\,,\end{aligned} \tag{A.58}$$

$$\begin{aligned}\phi_7 = {} & (3\eta_{21} - \eta_{03})(\eta_{30} - \eta_{12})\left[(\eta_{30} - \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] \\ & +(3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]\,,\end{aligned} \tag{A.59}$$

A.2.9 Normalized Feature of Compactness of $P(z)$

The normalized feature of compactness $\mathrm{M_{cmp}}$ of $P(z)$ is calculated as

$$\mathrm{M_{cmp}} = \frac{\mathrm{Area}_{\mathbf{P(z)}}}{\mu_{20} + \mu_{02}},\quad 0 \leq \mathrm{M_{cmp}} \leq 1\,, \tag{A.60}$$

where μ_{20}, μ_{02} are the central moments of second order calculated in Section A.2.8.

A.2.10 Normalized Feature of Eccentricity of $P(z)$

The normalized feature of eccentricity $\mathrm{M}_{\mathrm{ect}}$ of $P(z)$ is defined as

$$\mathrm{M}_{\mathrm{ect}} = \frac{\sqrt{(\mu_{20} + \mu_{02})^2 + 4\mu_{11}^2}}{\mu_{20} + \mu_{02}}, \quad 0 \leq \mathrm{M}_{\mathrm{ect}} \leq 1\,, \tag{A.61}$$

where $\mu_{20}, \mu_{02}, \mu_{11}$ are the central moments of second order calculated in Section A.2.8.

A.2.11 Form Factor of $P(z)$

The form factor of the $P(z)$ is given by:

$$f_{\mathrm{f-factor}} = \frac{4\pi f_{\mathrm{Area}}}{\sqrt{f_{\mathrm{Perimeter}}}}\,, \tag{A.62}$$

where f_{Area} and $f_{\mathrm{Perimeter}}$ are the features defined in Equations (A.21) and (A.22) respectively.

A.2.12 Circularity of $P(z)$

The circularity of $P(z)$ is defined as

$$f_{\mathrm{circularity}} = \frac{f_{\mathrm{Perimeter}}^2}{f_{\mathrm{Area}}}\,. \tag{A.63}$$

A.2.13 Normalized Circularity of $P(z)$

The normalized circularity of $P(z)$ is defined as

$$f_{\overline{\mathrm{circularity}}} = \frac{4 \cdot \pi \cdot f_{\mathrm{Area}}}{f_{\mathrm{Perimeter}}^2}\,, \tag{A.64}$$

where f_{Area} and $f_{\mathrm{Perimeter}}$ are the features defined in Equations (A.21) and (A.22) respectively.

A.2.14 Average Normalized Distance Between the Centroid and the Shape Boundary

The centroid $c = (c_x, c_y)$ of $P(z)$ is defined as

$$c_x = \frac{1}{6 \cdot f_{\mathrm{Area}}} \sum_{i=0}^{M-1} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i)\,, \tag{A.65}$$

$$c_y = \frac{1}{6 \cdot f_{\mathrm{Area}}} \sum_{i=0}^{M-1} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i)\,. \tag{A.66}$$

The mean of the normalized distance between the centroid and the shape boundary of $P(z)$ is calculated as

$$f_{\text{mean-norm-shape}} = \frac{1}{M} \sum_{i=0}^{M-1} \overline{\text{dist}}(v_i, c), \tag{A.67}$$

where

$$\overline{\text{dist}}(v_i, c) = \frac{\text{dist}(v_i, c)}{\text{argmax}_j \, \text{dist}(v_j, c)} \tag{A.68}$$

with

$$\text{dist}(v_i, c) = \sqrt{(x_i - c_x)^2 + (y_i - c_y)^2} \tag{A.69}$$

A.2.15 Standard Deviation of the Normalized Distances Between the Centroid and the Shape Boundary

The standard deviation of the normalized distances between the centroid and the shape boundary of $P(z)$ is given by

$$f_{\text{std-shape}} = \frac{1}{M} \sum_{i=0}^{M-1} \overline{\text{dist}}(v_i, c) - f_{\text{mean-norm-shape}}\,, \tag{A.70}$$

where $f_{\text{mean-norm-shape}}$ and $\overline{\text{dist}}(v_i, c)$ are defined in Equation (A.67) and Equation (A.68) respectively.

Appendix B

Simple Features for People Detection

In this appendix we describe the mathematical definition of the features used for segment classification. We define a segment as a set of N consecutive beams $S = \{b_0, \ldots, b_{N-1}\}$. Each beam b_i is represented by a tuple (α_i, d_i), where α_i is the angle of the beam relative to the robot and d_i is the length of the beam.

Additionally, each beam b_i can be represented by its end-point $p_i = (x_i, y_i)$, with $x_i = d_i \cos\alpha_i$ and $y_i = d_i \sin\alpha_i$. In this case, we assume that the origin of coordinates lays on the center of the laser sensor.

B.1 Number of Points in the Segment

This is just the number N of points that composed the segment the segment S.

B.2 Standard Deviation

The standard deviation f_σ of the segment S if defined as

$$f_\sigma = \sqrt{\frac{1}{N-1} \sum_{i=0}^{N-1} ||p_i - \overline{p}||^2}\,, \tag{B.1}$$

where $\overline{p}$ denotes the center of gravity of the segment.

B.3 Mean Average Deviation from Median

This feature measures the segment compactness more robustly than the standard deviation. The median of a distribution $f(x)$ is the value where the cumulative distribution function $F(x) = 1/2$. Given an ordered set of K scalar random samples x_i the median $\tilde{x}$ is defined as

$$\tilde{x} = \begin{cases} x_{(K+1)/2} & \text{if } K \text{ is odd} \\ \frac{1}{2}(x_{K/2} + x_{K/2+1}) & \text{if } K \text{ is even}\,. \end{cases} \tag{B.2}$$

Opposed to the mean, the median is less sensitive to outliers. In our multi-dimensional case, we calculate $\tilde{p}$ using the vector-of-medians approach by Aloupis [2001], i.e. $\tilde{p} = (\tilde{x}, \tilde{y})$. The average deviation from the median is then

$$f_{\varsigma} = \frac{1}{N} \sum_{i=0}^{N-1} ||p_i - \tilde{p}||\,. \tag{B.3}$$

B.4 Jump Distance from Preceeding Segment

This feature corresponds to the Euclidian distance between the first point of S_i and the last point of S_{i-1}

$$f_{jump-prev} = ||p_0^{S_i} - p_{N-1}^{S_{i-1}}||\,. \tag{B.4}$$

B.5 Jump Distance to Succeeding Segment

This feature corresponds to the Euclidian distance between the last point of S_i and the first point of S_{i+1}

$$f_{jump-next} = ||p_{N-1}^{S_i} - p_0^{S_{i+1}}||\,. \tag{B.5}$$

B.6 Euclidian Distance Between the First and Last Point of a Segment

This feature measures the Euclidian distance between the first and last point of a segment as

$$f_{width} = ||p_0 - p_{N-1}||\,. \tag{B.6}$$

B.7 Linearity

This feature measures the straightness of the segment and corresponds to the residual sum of squares to a line fitted into the segment in the least squares sense. Given the segment points in polar coordinates $p_i = (\alpha_i, d_i)$, fitting a line in the Hessian (ϕ, r)-representation that minimizes perpendicular errors from the points onto the line has a closed form solution. We use the (unweighted) expressions from [Arras, 2003]. Once the line parameters (ϕ, r) are found, the residual sum of squares is calculated as

$$f_{linearity} = \sum_{i=0}^{N-1} (x_i cos(\phi) + y_i sin(\phi) - r)^2 \,. \tag{B.7}$$

B.8 Circularity

To calculate the circularity of a segment S, we sum up the squared residuals to a fitted circle. Given a set of points in Cartesian coordinates, a fast way to find the best circle in the least squares sense is to parameterize the problem by the vector of unknowns as $x = (x_c \quad y_c \quad x_c^2 + x_c^2 - r_c^2)^T$ where x_c, y_c and r_c denote the circle center and radius. With this, the overdetermined equation system $A \cdot x = b$ can be established,

$$A = \begin{pmatrix} -2x_1 & -2y_1 & 1 \\ -2x_2 & -2y_2 & 1 \\ \vdots & \vdots & \vdots \\ -2x_n & -2y_n & 1 \end{pmatrix} \quad b = \begin{pmatrix} -x_1^2 - y_1^2 \\ -x_2^2 - y_2^2 \\ \vdots \\ -x_n^2 - y_n^2 \end{pmatrix}, \tag{B.8}$$

and solved using the pseudo-inverse

$$x = (A^T A)^{-1} A^T \cdot b \,. \tag{B.9}$$

The residual sum of squares is then

$$f_{Circularity} = \sum_{i=1}^{n} (r_c - \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2})^2 \,. \tag{B.10}$$

B.9 Radius

This feature represents the radius of the circle fitted to the segment. It corresponds to the value r_c in Equation (B.10).

B.10 Boundary Length

The boundary length $f_{boundary}$ of a segment S is defined as

$$f_{boundary} = \sum_{i=0}^{N-2} \mathrm{dist}(p_i, p_{i+1}) \,. \tag{B.11}$$

It measures the length of the poly-line corresponding to the segment. The function $\mathrm{dist}(p_i, p_j)$ calculates the Euclidean distance between two pints.

B.11 Boundary Regularity

The boundary regularity indicates the standard deviation of the distances of adjacent points in a segment and is defined as

$$f_{b-reg} = \sqrt{\frac{1}{N-2}\sum_{i=0}^{N-2}\left(\text{dist}(p_i, p_{i+1}) - \bar{d}\right)^2}\,, \tag{B.12}$$

where $\bar{d}$ is the mean distance between consecutive points.

B.12 Mean Curvature

The average curvature $f_{\bar{k}} = \sum \hat{k}_j$ over a segment S is calculated using the following curvature approximation. Given a succession of three points p_A, p_B, and p_C, let A denote the area of the triangle $p_A p_B p_C$ and d_A, d_B, d_C the three distances between the points. Then, an approximation of the discrete curvature of the boundary at p_B is given by

$$f_{\hat{k}} = \frac{4A}{d_A d_B d_C}\,. \tag{B.13}$$

This is an alternative measurement for the radius of the segment, since curvature and radius are inverse proportional.

B.13 Mean Angular Difference

This feature traverses the boundary and calculates the average of the angles β_j between the vectors $\overline{p_{j-1}p_j}$ and $\overline{p_j p_{j+1}}$. The corresponding feature f_{β_j} is then defined as

$$f_{\beta_j} = \angle(\overline{p_{j-1}p_j}, \overline{p_j p_{j+1}})\,. \tag{B.14}$$

Care has to be taken that angle differences are properly unwrapped. This features is a measure of the convexity/concavity of segment S.

B.14 Mean Speed

Given two scans with their associated timestamps T_k, T_{k+1}, this feature determines the speed v_i for each segment point along its beam as

$$v_i = \frac{d_i^{k+1} - d_i^k}{T_{k+1} - T_k}\,, \tag{B.15}$$

where d_j^k and d_j^{k+1} are the range values of beam j at times k and $k+1$. The final feature f_v averages over all beams in the segment

$$f_v = \sum_{i=0}^{N-1} v_i \,. \tag{B.16}$$

Appendix C

Graphical Models

Probabilistic graphical models are graphs in which nodes represent random variables, and arcs represent conditional dependence between them. Hence, they provide a compact representation of joint probability distributions [Murphy, 2001].

There are two main kinds of graphical models: directed and undirected. The first models are represented by graphs whose nodes are connected by directed arcs. They are also also known as Bayesian networks, belief networks, generative models or causal models. On the other hand, the arcs in an undirected graphical model are undirected. Undirected graphical models are also known as Markov networks (MN) or Markov random fields (MRFs). It is also possible to have a model with both directed and undirected arcs, called a chain graph. However, chain graphs are out of the scope of this appendix.

C.1 Directed Models

An example of a simple directed graphical model is given in Figure C.1(a). Here, the arc from A to B can be informally interpreted as indicating that *A causes B*. *A* is also said to be the *parent* of *B*. A well known directed graphical model is the *Bayesian network* (Figure C.1(b)). This representation has two main characteristics. First, each node Y_i has a conditional probability distribution $P(Y_i|Parents(Y_i))$ that quantifies the effect of the parents on the node. And second, the graph has no directed cycles (a directed acyclic graph or DAG).

In a Bayesian network, a node is independent of its ancestors given its parents, where the ancestor/parent relationship is with respect to some fixed topological ordering of the nodes. In Bayesian networks, the expression for the joint probability of all nodes can be simplified using the topology of the network and the conditional independence relationships. As an example, the joint probability of the network of Figure C.1(b) can be expressed using the chain rule as

$$P(A, B, C, D) = P(D|A, B, C)P(C|A, B)P(B|A)P(A). \tag{C.1}$$

By using the conditional independence relationships, we can rewrite it as

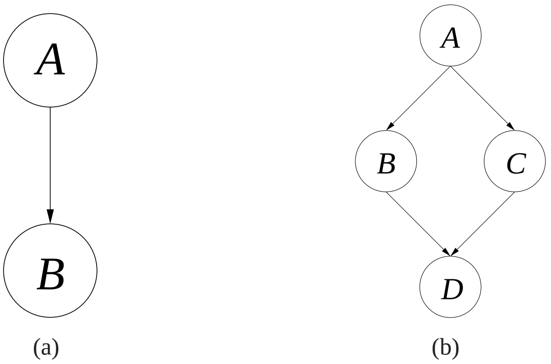


Figure C.1: Examples of directed graphical models.

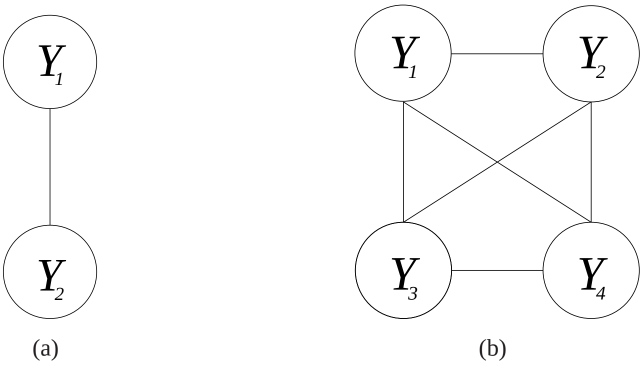


Figure C.2: Examples of Markov random fields.

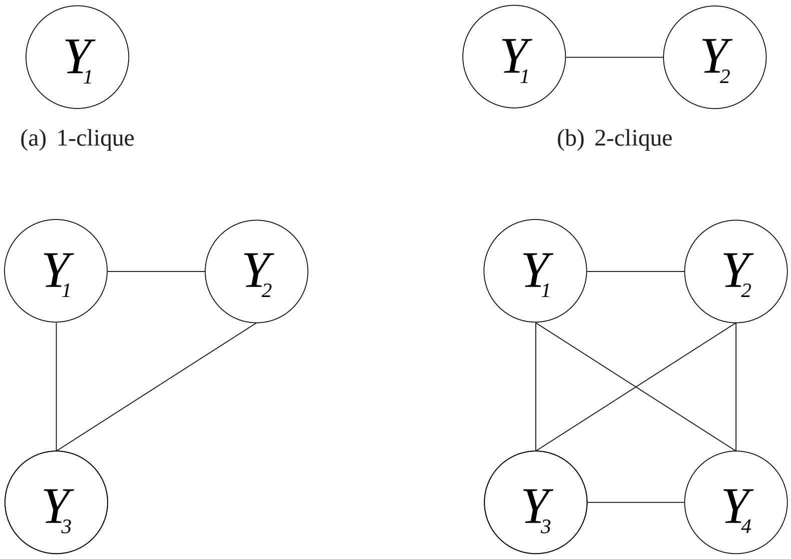
$$P(A, B, C, D) = P(D|B, C)P(C|A)P(B|A)P(A). \tag{C.2}$$

This expression indicates that D is independent of A given B and C, and C is independent of B given A.

C.2 Undirected Graphical Models

An undirected graphical model, also called *Markov random field* (MRF) or *Markov network* (MN), is a graphical model where no assumption is made about the direction of the causality between nodes in the graph. Examples of MRFs are given in Figure C.2.

We restrict ourselves here to the case in which nodes are represented by discrete variables. Each discrete variable $Y_i \in Y$ corresponds to a set of K possible labels $Y_i \in \{1, \ldots, K\}$. We then define a Markov random field as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the set of nodes $\mathcal{V}$ represent discrete variables, and the edges $\mathcal{E}$ refer to the relations between them.



(a) 1-clique

(b) 2-clique

(c) 3-clique

(d) 4-clique

Figure C.3: Possible cliques extracted from the MRF of Figure C.2(b). The 4-clique in (d) corresponds to the complete original network.

A MRF can be divided into a subset of cliques C, where each clique $c \in C$ is associated with a subset $Y_c \in Y$. The nodes in a clique Y_c form a fully connected graph in the MRF. Figure C.3 depicts some example cliques corresponding to the Markov random field of Figure C.2(b).

Each clique is accompanied by a potential $\phi_c(Y_c)$ which associates a non-negative value with each assignment y_c to Y_c. The Markov random field defines the probability distribution

$$P_\phi(y) = \frac{1}{Z} \prod_{c \in C} \phi_c(y_c)\,, \tag{C.3}$$

where y represents an assignment of Y. Additionally, Z is the *partition function* given by

$$Z = \sum_{y'} \prod_{c \in C} \phi_c(y'_c)\,. \tag{C.4}$$

We focus now on pairwise Markov random fields. A pairwise Markov random field is a network where all of the cliques involved are either a single node, or a pair of nodes (1-clique or 2-clique). In a pairwise Markov random field with edges $\mathcal{E} = \{(ij)\,|\,i < j\}$, the nodes and edges are associated with potentials $\phi_i(Y_i)$ and $\phi_{ij}(Y_i, Y_j)$ respectively. This network defines the distribution

$$P_\phi(y) = \frac{1}{Z}\prod_{i=1}^{N}\phi_i(y_i)\prod_{(ij)\in\mathcal{E}}\phi_{ij}(y_i,y_j)\,, \tag{C.5}$$

where N represents the total number of nodes in the network. The partition function Z is now defined as

$$Z = \sum_{y'}\prod_{i=1}^{N}\phi_i(y'_i)\prod_{(ij)\in\mathcal{E}}\phi_{ij}(y'_i,y'_j)\,. \tag{C.6}$$

Additionally, in a Markov ramdom field, each node Y_i can be assigned a feature vector $x_i \in \mathbb{R}^{d_n}$, which describes the properties of the object that the node represents. Similarly, a feature vector $x_{ij} \in \mathbb{R}^{d_e}$ can be assigned to each edge (ij). The feature vector x_{ij} indicates the properties that describe the relation between the objects represented by the nodes Y_i and Y_j.

The node and edge potentials are functions of the feature vectors x_i and x_{ij}. The simplest model of dependence of the potentials on the features is a log-linear combination defined as

$$\log\phi_i(k) = w_n^k \cdot x_i\,, \tag{C.7}$$

$$\log\phi_{ij}(k,l) = w_e^{k,l} \cdot x_{ij}\,, \tag{C.8}$$

where w_n^k and $w_e^{k,l}$ are label-specific row vectors of node and edge parameters, with size d_n and d_e respectively. This formulation assumes that all of the nodes in the network share the same set of weights, and the same for the edges.

The representation of an assignment y is done by a set of $K \cdot N$ indicators $\{y_i^k\}$, where $y_i^k = I(y_i = k)$, with I being the indicator function which returns 1 if its argument is true, and 0 otherwise. Remember that K is the number of possible labels for each node Y_i, and N represents the total number of nodes in the graph.

We can now define the log conditional probability

$$\log P_w(y|x) = \sum_{i=1}^{N}\sum_{k=1}^{K}(w_n^k \cdot x_i)y_i^k + \sum_{(ij)\in\mathcal{E}}\sum_{k,l=1}^{K}(w_e^{k,l} \cdot x_{ij})y_i^k y_j^l - \log Z_w(x)\,. \tag{C.9}$$

Here the partition function $Z_w(x)$ depends on the parameters w and features x, but not on the labels y.

The main task in a Markov random field consists on finding the assignment $y \in Y$ that maximizes $\log P_w(y|x)$. This is actually a *maximum a posteriori* (MAP) assignment that can be formulated as an integer linear program as shown by Taskar [2004].

C.3 Associative Markov Networks

An important subclass of Markov networks are the *associative Markov networks*, introduced by Taskar [2004]. These networks represent problems in which related variables tend to have the same label. AMNs have been used, for example, in text classification domains [Taskar, 2004], or segmentation of 3D scan data [Anguelov *et al.*, 2005]. In this last case, when the labels represent objects in the 3D scene, it is clear that 3D-points which are close to each other tend to have the same label, as they probably pertain to the same object.

A common approach in AMNs in to use a *generalized Potts model* [Potts, 1952], which penalizes assignments that do not have the same label across the edge. In this case the potential functions for edges have the form

$$\phi_{ij}(k,l) = \lambda_{ij}, \quad \lambda_{ij} \leq 1, \quad \forall k \neq l\,, \tag{C.10}$$

$$\phi_{ij}(k,k) = 1\,. \tag{C.11}$$

Additionally, Taskar [2004] extends the Potter model with the goal of allowing different labels to have different attraction strengths. These new potentials are defined as

$$\phi_{ij}(k,l) = 1, \quad \forall k \neq l\,, \tag{C.12}$$

$$\phi_{ij}(k,k) = \lambda_{ij}^k, \quad \lambda_{ij}^k \geq 1\,. \tag{C.13}$$

C.3.1 Training Associative Markov Networks

Training an associative Markov network consists on learning the weights w given an already labeled instance $(x, \hat{y})$ of the network. Here, $\hat{y}$ indicates the labeling given to the nodes, and x represent the features of nodes and edges. The weights w are selected in a way that maximize $\log P_w(\hat{y}|x)$. This is computational expensive, as the partition function Z depends on the weights w, and it has to be calculated for each assignment of w during the search.

An alternative way of finding the weights w consists on maximizing the margin between the optimal labeling $\hat{y}$ and any other possible labeling $y \neq \hat{y}$ as

$$\log P_w(\hat{y}|x) - \log P_w(y|x)\,. \tag{C.14}$$

In this case the partition function Z cancels out and the maximization can be done efficiently. This method is called *maximum margin* optimization. We refer the reader to [Taskar *et al.*, 2003] for more details in the training process.

C.3.2 Inference with Associative Markov Networks

Once we have learnt the weights for the network, we can apply an inference process to label the nodes of a new unlabeled network. This is done by finding the assignment y that maximizes Equation (C.9). Since the partition function Z does not depend on the labels y, we can ignore it in the maximization process. Taking into account some constraints imposed on the variables y_i^k, the inference can be done solving the linear program

$$\begin{aligned} &\sum_{i=1}^{N}\sum_{k=1}^{K}(w_n^k \cdot x_i)y_i^k + \sum_{(ij)\in\mathcal{E}}\sum_{k=1}^{K}(w_e^{k,k} \cdot x_{ij})y_{ij}^k \\ &s.t. \quad y_i^k \geq 0, \quad \forall i,k; \quad \sum_{k=1}^{K} y_i^k = 1, \quad \forall i\,; \\ &\quad y_{ij}^k \leq y_i^k, \quad y_{ij}^k \leq y_j^k, \quad \forall i,j \in \mathcal{E}, k\,. \end{aligned} \tag{C.15}$$

More details about inference in associative Markov networks can be found in [Taskar, 2004].

List of Figures

List of Tables

Bibliography

[Aloupis, 2001] Greg Aloupis. *On Computing Geometric Estimators of Location*. PhD thesis, School of Computer Science, McGill University, 2001.

[Althaus and Christensen, 2003] Philipp Althaus and Henrik I. Christensen. Behaviour coordination in structured environments. *Advanced Robotics*, 17(7):657–674, 2003.

[Anguelov *et al.*, 2005] Dragomir Anguelov, Ben Taskar, Vassil Chatalbashev, Daphne Koller, Dinkar Gupta, Geremy Heitz, and Andrew Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 169–176, 2005.

[Anthony and Bartlett, 1999] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[Arras *et al.*, 2007] Kai O. Arras, Oscar Martinez Mozos, and Wolfram Burgard. Using boosted features for the detection of people in 2d range data. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3402–3407, 2007.

[Arras, 2003] Kai O. Arras. *Feature-Based Robot Navigation in Known and Unknown Environments*. PhD thesis, Swiss Federal Institute of Technology Lausanne (EPFL), Thèse No. 2765, June 2003.

[Beeson *et al.*, 2005] Patrick Beeson, Nicholas K. Jong, and Benjamin Kuipers. Towards autonomous topological place detection using the extended voronoi graph. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005.

[Beeson *et al.*, 2007] Patrick Beeson, Matt MacMahon, Joseph Modayil, Aniket Murarka, Benajmin Kuipers, and Brian Stankiewicz. Integrating multiple representations of spatial knowledge for mapping, navigation, and communication. In *Interaction Challenges for Intelligent Assistants*, AAAI Spring Symposium, Stanford, CA, USA, 2007.

[Brunskill *et al.*, 2007] Emma Brunskill, Thomas Kollar, and Nicholas Roy. Topological mapping using spectral clustering and classification. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems*, San Diego, CA, USA, October 2007.

[Burgard *et al.*, 2000a] W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *ArtificialIntelligence*, 114(1-2), 2000.

[Burgard *et al.*, 2000b] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk HÃd'hnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1–2), 2000.

[Burgard *et al.*, 2005] Wolfram Burgard, Mark Moors, Cyrill Stacniss, and Frank Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–378, 2005.

[Burkhard, 2004] Hans Burkhard. Pattern recognition course. Course slides, 2004.

[Buschka and Saffiotti, 2002] P. Buschka and A. Saffiotti. A virtual sensor for room detection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 637–642, 2002.

[Cao *et al.*, 1997] Y. Uny Cao, Alex S. Fukunaga, and Andrew B. Kahng. Cooperative mobile robotics: Antecedents and directions. *Journal of Autonomous Robots*, 4(1):7–27, 1997.

[Cheyer and Martin, 2001] Adam Cheyer and David Martin. The open agent architecture. *Journal of Autonomous Agents and Multi-Agent Systems*, 4(1):143–148, March 2001. OAA.

[Choset, 2001] Howie Choset. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 2001.

[Clark *et al.*, 2004] Robert A.J. Clark, Korin Richmond, and Simon King. Festival 2 – build your own general purpose unit selection speech synthesiser. In *Proceedings of the ISCA workshop on speech synthesis*, 2004.

[CoSy, 2004] CoSy. Cognitive systems for cognitive assistants. http://www.cognitivesystems.org/, 2004.

[Cover and Thomas, 1991] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & sons, 1991.

[Cui *et al.*, 2005] Jinshi Cui, Hongbin Zha, Huijing Zhao, and Ryosuke Shibasaki. Tracking multiple people using laser and vision. In *Proceedings of the IEEE/RSJ*

International Conference on Intelligent Robots and Systems, Alberta, Canada, 2005.

[Dellaert *et al.*, 1999] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proceedings of the IEEE InternationalConference on Robotics and Automation*, 1999.

[Diosi *et al.*, 2005] Albert Diosi, Geoffrey Taylor, and Lindsay Kleeman. Interactive SLAM using laser and advanced sonar. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[Dudek *et al.*, 1996] Gregory Dudek, Michael R. M. Jenkin, Evangelos Milios, and David Wilkes. A taxonomy for multi-agent robotics. *Journal of Autonomous Robots*, 3(4):375–397, 1996.

[Elfes, 1989] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 6 1989.

[Elin A. Topp *et al.*, 2006] Helge Huettenrauch Elin A. Topp, Henrik I. Christensen, , and Kerstin Severinson Eklundh. Bringing together human and robotic environment representations – a pilot study. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, October 2006.

[Fabrizi and Saffiotti, 2000] E. Fabrizi and A. Saffiotti. Extracting topology-based maps from gridmaps. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2972–2978, San Francisco, CA, 2000.

[Fod *et al.*, 2002] Ajo Fod, Andrew Howard, and Maja J Mataric. Laser-based people tracking. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002.

[Folkesson *et al.*, 2005] John Folkesson, Patric Jensfelt, and Henrik Christensen. Vision SLAM in the measurement subspace. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 30–35, 2005.

[Fox *et al.*, 1999] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proc. of the National Conference on Artificial Intelligence*, 1999.

[Fox *et al.*, 2003] Dieter Fox, Jonathan Ko, Kurt Konolige, and Benjamin Stewart. A hierarchical bayesian approach to the revisiting problem in mobile robot map building. In *Proceedings of the International Symposium of Robotics Research*, Siena, Italy, 2003.

[Freund and Schapire, 1995] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In

Proceedings of the European Conference on Computational Learning Theory, pages 23–37, 1995.

[Freund and Schapire, 1999] Yoav Freund and Robert E. Schapire. A short introduction to boosting. *Journal of Japanase Society for Artificial Intelligence*, 14(5):771–780, 1999.

[Freund, 1990] Yoav Freund. Boosting a weak learning algorithm by majority. In *COLT: Proceedings of the Workshop on Computational Learning Theory*. Morgan Kaufmann, 1990.

[Freund, 1993] Yoav Freund. *Data filtering and distribution modeling algorithms for machine learning*. PhD thesis, University of California at Santa Cruz, 1993.

[Friedman *et al.*, 2000] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

[Friedman *et al.*, 2007] Stephen Friedman, Hanna Pasula, and Dieter Fox. Voronoi random fields: Extracting the topological structure of indoor environments via place labeling. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Hyderabad, India, 2007.

[Galindo *et al.*, 2005] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.A. Fernández-Madrigal, and J. González. Multi-hierarchical semantic maps for mobile robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Alberta, Canada, 2005.

[Gerkey *et al.*, 2003] Brian Gerkey, Richard T. Vaughan, and Andrew Howard. The Player/Stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the International Conference on Advanced Robotics*, pages 317–323, Coimbra, Portugal, June 2003.

[Gonzalez and Wintz, 1987] Rafael C. Gonzalez and Paul A. Wintz. *Digital Image Processing*. Addison-Wesley Publishing Inc., 1987.

[Grove and Schuurmans, 1998] Adam J. Grove and Dale Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. In *Proceedings of the National Conference on Artificial Intelligence*, pages 692–699, 1998.

[Guyon and Elisseeff, 2003] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[Guzzoni *et al.*, 1997] Didier Guzzoni, Adam Cheyer, Luc Julia, and Kurt Konolige. Many robots make short work. *AI Magazine*, 18(1):55–64, 1997.

[Haarslev and Mölle, 2003] Volker Haarslev and Ralf Mölle. Racer: A core inference engine for the semantic web. In *Proceedings of the International Workshop on Evaluation of Ontology-based Tools*, pages 27–36, Florida, USA, October 2003.

[Haehnel *et al.*, 2003] Dirk Haehnel, Rudolph Triebel, Wolfram Burgard, and Sebastian Thrun. Map building with mobile robots in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003.

[Haralick and Shapiro, 1992] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*. Addison-Wesley Publishing Inc., 1992.

[Haussler, 1992] David Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

[Hertzberg and Kirchner, 1996] J. Hertzberg and F. Kirchner. Landmark-based autonomous navigation in sewerage pipes. In *Proceedings of the First Euromicro Workshop on Advanced Mobile Robots.*, 1996.

[Hirtle and Jonides, 1985] Stephen C. Hirtle and John Jonides. Evidence for hierarchies in cognitive maps. *Memory and Cognition*, 13:208–217, 1985.

[Howard and Roy, 2003] Andrew Howard and Nicholas Roy. The robotics data set repository (radish), 2003.

[Howard *et al.*, 2002] Andrew Howard, Maja J. Mataric, and Gaurav S. Sukhatme. An incremental deployment algorithm for mobile robot teams. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2849–2854, Lausanne, Switzerland, 2002.

[Huang *et al.*, 1986] Y. Huang, Z.Cao, S. Oh, E. Kattan, and E. Hall. Automatic operation for a robot lawn mower. In *SPIE Conference on Mobile Robots*, volume 727, pages 344–354, Cambridge, MA, USA, 1986.

[Jäger and Nebel, 2002] Markus Jäger and Bernhard Nebel. Dynamic decentralized area partitioning for cooperating cleaning robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3577–3582, Washington, DC, USA, 2002.

[Jensfelt and Kristensen, 2001] Patric Jensfelt and Steen Kristensen. Active global localisation for a mobile robot using multiple hypothesis tracking. *IEEE Transactions on Robotics and Automation*, 17(5):748–760, October 2001.

[Kearns and Valiant, 1988] Michael J. Kearns and Leslie G. Valiant. Learning boolean formulae or finite automata is as hard as factoring. Technical Report TR-14-88, Harvard University Aiken Computation Laboratory, August 1988.

[Kearns and Valiant, 1994] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, 1994.

[Kearns and Vazirani, 1994] Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994.

[Kersting *et al.*, 2007] K. Kersting, C. Plagemann, A. Cocora, W. Burgard, and L. De Raedt. Learning to transfer optimal navigation policies. *Advanced Robotics. Special Issue on Imitative Robots*, 21(9), September 2007.

[Kleinhagenbrock *et al.*, 2002] M. Kleinhagenbrock, S. Lang, J. Fritsch, F. Lömker, G.A. Fink, and G. Sagerer. Person tracking with a mobile robot based on multi-modal anchoring. In *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication*, Berlin, Germany, 2002.

[Ko *et al.*, 2003] Jonathan Ko, Benjamin Stewart, Dieter Fox, Kurt Konolige, and Benson Limketkai. A practical, decision-theoretic approach to multi-robot mapping and exploration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3232–3238, Las Vegas, NV, USA, 2003.

[Koenig and Simmons, 1998] Sven Koenig and Reid G. Simmons. Xavier: A robot navigation architecture based on partially observable markov decision process models. In D. Kortenkamp, R. Bonasso, and R. Murphy, editors, *Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems*, pages 91–122. MIT Press, 1998.

[Kortenkamp and Weymouth, 1994] David Kortenkamp and Terry Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proceedings of the National Conference on Artificial Intelligence*, pages 979–984, 1994.

[Krieg-Brückner *et al.*, 1998] Bernd Krieg-Brückner, Thomas Röfer, Hans-Otto Carmesin, and Rolf Müller. A taxonomy of spatial knowledge for navigation and its application to the Bremen autonomous wheelchair. In Christian Freksa, Christopher Habel, and Karl F. Wender, editors, *Spatial Cognition*, volume 1404 of *Lecture Notes in Artificial Intelligence*, pages 373–397. Springer Verlag, 1998.

[Kruijff *et al.*, 2006] Geert-Jan M. Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik I. Christensen. Clarification dialogues in human-augmented mapping. In *Proceedings of the 1st ACM Conference on Human-Robot Interaction*, Salt Lake City, UT, USA, 2006.

[Kruijff *et al.*, 2007] Geert-Jan M. Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik I. Christensen. Situated dialogue and spatial organization: What, where. . . and why? *International Journal of Advanced Robotic Systems*, 4(2), March 2007.

[Kuipers and Byun, 1991] Benjamin Kuipers and Yung-Tai Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8:47–63, 1991.

[Kuipers, 2000] Benjamin Kuipers. The Spatial Semantic Hierarchy. *Artificial Intelligence*, 119:191–233, 2000.

[Latombe, 1991] Jean-Claude Latombe. *Robot Motion Planning*. Academic Publishers, Boston, MA, 1991.

[Lienhart *et al.*, 2003] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM, Pattern Recognition Symposium*, 2003.

[Loncaric, 1998] Sven Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, 1998.

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

[Luo *et al.*, 2007] Jie Luo, Andrzej Pronobis, Barbara Caputo, and Patric Jensfelt. Incremental learning for place recognition in dynamic environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, 2007.

[Matarić and Sukhatme, 2001] Maja J. Matarić and Gaurav S. Sukhatme. Task-allocation and coordination of multiple robots for planetary exploration. In *Proceedings of the International Conference on Advanced Robotics*, pages 61–70, Budapest, Hungary, 2001.

[McNamara, 1986] Tim .P. McNamara. Mental representations of spatial relations. *Cognitive Psychology*, 18:87–121, 1986.

[Meijster *et al.*, 2000] A. Meijster, J.B.T.M. Roerdink, and W.H. Hesselink. *Mathematical Morphology and its Applications to Image and Signal Processing*, chapter A General Algorithm for Computing Distance Transforms in Linear Time, pages 331–340. Kluwer Academic Publishers, 2000.

[Meir and Rätsch, 2003] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. In *Advanced lectures on machine learning*, pages 118–183. Springer-Verlag New York, Inc., 2003.

[Mitchell, 1997] Tom M. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[Montemerlo *et al.*, 2003] Michael Montemerlo, Nicholas Roy, and Sebastian Thrun. Perspectives on standardization in mobile robot programming. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[Moravec, 1988] Hans P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9:61–74, 1988.

[Mozos, 2004] Oscar Martinez Mozos. Supervised learning of places from range data using adaboost. Master's thesis, University of Freiburg, December 2004.

[Murphy, 2001] Kevin P. Murphy. An introduction to graphical models. Tutorial, 2001.

[Murphy, 2004] Robin R. Murphy. Human-robot interaction in rescue robotics. *IEEE Systems, Man and Cybernetics Part C: Applications and Reviews*, 34(2):138–153, 2004.

[Newman *et al.*, 2002] Paul Newman, John Leonard, Juan Domingo Tardos, and Jose Neira. Explore and return: Experimental validation of real-time concurrent mapping and localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1802–1809, Washington, D.C., USA, 2002.

[Nuance, 1999] Nuance. Nuance speech recognition system developer's manual version 6.2. http://www.nuance.com/, 1999.

[Oore *et al.*, 1997] Sageev Oore, Geoffrey E. Hinton, and Gregory Dudek. A mobile robot that learns its place. *Neural Computation*, 9(3):683–699, 1997.

[O'Rourke, 1998] Joseph O'Rourke. *Computational Geometry in C*. Cambridge University Press, second edition, 1998.

[Parker, 2003] Lynne E. Parker. The effect of heterogeneity in teams of 100+ mobile robots. In *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*. Kluwer Academic Publishers, Boston, 2003.

[Persson *et al.*, 2007] Martin Persson, Tom Duckett, and Achim J Lilienthal. Virtual sensors for human concepts - building detection by an outdoor mobile robot. *Robotics and Autonomous Systems*, 55(5):383–390, May 31 2007.

[Potts, 1952] Renfrey B. Potts. Some generalized order-disorder transformations. *Proceedings of the Cambridge Philosophical Society*, 48:106–109, 1952.

[Premebida and Nunes, 2005] Cristiano Premebida and Urbano Nunes. Segmentation and geometric primitives extraction from 2D laser range data for mobile robot applications. In *Robótica 2005 - Scientific meeting of the 5th National Robotics Festival*, Coimbra, Portugal, April 2005.

[Premebida *et al.*, 2007] Cristiano Premebida, Goncalo Monteiro, Urbano Nunes, and Paulo Peixoto. Lidar and vision-based approach for pedestrian and vehicle detection and tracking. In *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pages 1044–1049, Seattle,Washington,USA, 2007.

[Pronobis *et al.*, 2006] Andrzej Pronobis, Barbara Caputo, Patric Jensfelt, and Henrik I. Christensen. A discriminative approach to robust visual place recognition. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.

[Pronobis *et al.*, 2008] Andrzej Pronobis, Oscar Martinez Mozos, and Barbara Caputo. SVM-based discriminative accumulation scheme for place recognition. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Pasadena, California, USA, 2008.

[Rätsch *et al.*, 2001] Gunnar Rätsch, Takashi Onoda, and Klaus-Robert Müller. Soft margins for adaboost. *Machine Learning*, 42(3):287–320, 2001.

[Reuter, 2000] J. Reuter. Mobile robot self-localization using pdab. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3512–3518, San Francisco, CA, USA, 2000.

[Rosenfeld and Pfaltz, 1966] Azriel Rosenfeld and John L. Pfaltz. Sequential operations in digital picture processing. *Journal of the Association for Computing Machinery*, 13(4):471–494, 1966.

[Rosenfeld *et al.*, 1976] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems Man and Cybernetics*, 6(6):420–433, 1976.

[Rottmann *et al.*, 2005] Axel Rottmann, Oscar Martinez Mozos, Cyrill Stachniss, and Wolfram Burgard. Place classification of indoor environments with mobile robots using boosting. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1306–1311, Pittsburgh, PA, USA, 2005.

[Rottmann, 2005] Axel Rottmann. Bild- und laserbasierte klassifikation von umgebungen mit mobilen robotern. Master's thesis, University of Freiburg, Department of Computer Science, 2005. In German.

[Roumeliotis and Bekey, 2000] Stergios I. Roumeliotis and George A. Bekey. Bayesian estimation and kalman filtering: a unified framework for mobile robot localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2985–2992, San Francisco, CA, USA, 2000.

[Russ, 1992] John C. Russ. *The Image Processing Handbook*. CRC Press, 1992.

[Russell and Norvig, 2003] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, second edition, 2003.

[Sack and Burgard, 2004] Daniel Sack and Wolfram Burgard. A comparison of methods for line extraction from range data. In *Proceedings of the IFAC Symposium on Intelligent Autonomous Vehicles*, Lisabon, Portugal, 2004.

[Schapire and Singer, 1999] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[Schapire, 1990] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[Schapire, 2001] Robert E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2001.

[Scheutz *et al.*, 2004] M. Scheutz, J. McRaven, and G. Cserey. Fast, reliable, adaptive, bimodal people tracking for indoor environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[Schulz *et al.*, 2003a] Dirk Schulz, Wolfram Burgard, Dieter Fox, and Armin B. Cremers. People tracking with a mobile robot using sample-based joint probabilistic data association filters. *International Journal of Robotics Research*, 22(2):99–116, 2003.

[Schulz *et al.*, 2003b] Dirk Schulz, Wolfram Burgard, Dieter Fox, and Armin B. Cremers. People tracking with a mobile robot using sample-based joint probabilistic data association filters. *International Journal of Robotics Research*, 22(2):99–116, 2003.

[Shatkey and Kaelbling, 1997] Hagit Shatkey and Leslie Pack Kaelbling. Learning topological maps with weak local odometric information. In *Proceedings of the International Conference on Artificial Intelligence*, 1997.

[Siegwart *et al.*, 2003] Roland Siegwart, Kai O. Arras, Samir Bouabdallah, Daniel Burnier, Gilles Froidevaux, Xavier Greppin, BjÃűrn Jensen, Antoine Lorotte, Laetitia Mayor, Mathieu Meisser, Roland Philippsen, Ralph Piguet, Guy Ramel, Gregoire Terrien, and Nicola Tomatis. Robox at expo.02: A large scale installation of personal robots. *Robotics and Autonomous Systems*, 42:203–222, 2003.

[Simmons and Koenig, 1995] Reid Simmons and Sven Koenig. Probabilistic robot navigation in partially observable environments. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1080–1087, 1995.

[Simmons *et al.*, 2000] Reid Simmons, Joaquin Lopez Fernandez, Richard Goodwin, Sven Koenig, and Joseph O'Sullivan. Lessons learned from xavier. *Robotics and Automation Magazine*, pages 733 – 39, 2000.

[Singh and Fujimura, 1993] Karansher Singh and Kikuo Fujimura. Map making by cooperating mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation,* pages 254–259, Atlanta, GA, USA, 1993.

[Smith *et al.*, 2004] Michael K. Smith, Chris Welty, and Deborah L. McGuinness. OWL web ontology language guide, 2004.

[Sousa *et al.*, 2007] Pedro Sousa, Rui Araujo, and Urbano Nunes. Real-time labeling of places using support vector machines. In *Proceedings of the IEEE International symposium on industrial electronics,* Vigo, Spain, June 2007.

[Spexard *et al.*, 2006] Thorsten Spexard, Shuyin Li, Britta Wrede, Jannik Fritsch, Gerhard Sagerer, Olaf Booij, Zoran Zivkovic, Bas Terwijn, and Ben Kröse. BIRON, where are you? - enabling a robot to learn new places in a real home environment by integrating spoken dialog and visual localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[Tapus and Siegwart, 2005] Adriana Tapus and Roland Siegwart. Incremental robot mapping with fingerprints of places. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2429– 2434, August 2005.

[Taskar *et al.*, 2003] Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin markov networks. In *Proceedings of the Neural Information Processing Systems*, 2003.

[Taskar, 2004] Ben Taskar. Learning associative markov networks. In *Proceedings of the International Conference on Machine Learning*, 2004.

[Theodoridis and Koutroumbas, 2006] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Academic Press, third edition, 2006.

[Thrun *et al.*, 2003] Sebastian Thrun, Dirk Hähnel, David Ferguson, Michael Montemerlo, Rudolph Triebel, Wolfram Burgard, Christopher Baker, Zachary Omohundro, Scott Thayer, and William Whittaker. A system for volumetric robotic mapping of abandoned mines. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, 2003.

[Thrun *et al.*, 2005] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[Thrun, 1998] Sebastian Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence,* 99(1):21–71, 1998.

[Topp and Christensen, 2005] Elin A. Topp and Henrik I. Christensen. Tracking for following and passing persons. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems,* Alberta, Canada, 2005.

[Topp and Christensen, 2006] Elin A. Topp and Henrik I. Christensen. Topological modelling for human augmented mapping. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems*, Beijing, China, October 2006.

[Torralba *et al.*, 2003] Antonio Torralba, Kevin P. Murphy, William T. Freeman, and Mark A. Rubin. Context-based vision system for place and object recognition. In *Proceedings of the International Conference on Computer Vision*, 2003.

[Treptow *et al.*, 2003] André Treptow, Andreas Masselli, and Andreas Zell. Real-time object tracking for soccer-robots without color information. In *Proceedings of the European Conference on Mobile Robots*, 2003.

[Triebel *et al.*, 2007a] Rudolph Triebel, Oscar Martinez Mozos, and Wolfram Burgard. Relational learning in mobile robotics: An application to semantic labeling of objects in 2D and 3D environment maps. In *Annual Conference of the German Classification Society on Data Analysis, Machine Learning, and Applications*, Freiburg, Germany, 2007.

[Triebel *et al.*, 2007b] Rudolph Triebel, Richard Schmidt, Oscar Martinez Mozos, and Wolfram Burgard. Instace-based amn classification for improved object recognition in 2d and 3d laser range data. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 2225–2230, Hyderabad, India, 2007.

[Valiant, 1984] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[Vasudevan *et al.*, 2006] Shrihari Vasudevan, Stefan Gächter, Marc Berger, and Roland Siegwart. Cognitive maps for mobile robots âĂŞ an object based approach. In *Proceedings of the IEEE/RSJ IROS 2006 Workshop: From Sensors to Human Spatial Concepts*, Beijing, China, 2006.

[Viola and Jones, 2001] Paul Viola and Michael J. Jones. Robust real-time object detection. In *Proceedings of IEEE Workshop on Statistical and Theories of Computer Vision*, 2001.

[Witten and Frank, 2000] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 2000.

[Wolf and Sukhatme, 2006] Denis F. Wolf and Gaurav S. Sukhatme. Activity-based semantic mapping of an urban environment. In *Proceedings, International Symposium on Experimental Robotics*, 2006.

[Xavier *et al.*, 2005] Joao Xavier, Marco Pacheco, Daniel Castro, Antonio Ruano, and Urbano Nunes. Fast line, arc/circle and leg detection from laser scan data in a player driver. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005.

[Yamamoto, 1979] Hiromichi Yamamoto. A method of deriving compatibility coefficients for relaxation operators. *Computer graphics and image processing*, 10:256–271, 1979.

[Yamauchi *et al.*, 1999] Brian Yamauchi, Alan Schultz, and William Adams. Integrating exploration and localization for mobile robots. *Adaptive Behavior*, 7(2):217–229, 1999.

[Zender, 2006] Hendrik Zender. Learning spatial organization through situated dialogue. Master's thesis, Department of Computational Linguistics, Saarland University, Saarbruecken, Germany, 2006.

[Zivkovic and Krose, 2007] Zoran Zivkovic and Ben Krose. Part based people detection using 2D range data and images. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 214–219, San Diego, CA, USA, 2007.

[Zivkovic *et al.*, 2005] Zoran Zivkovic, Bram Bakker, and Ben Kröse. Hierarchical map building using visual landmarks and geometric constraints. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

Notes