1

Using AdaBoost for Place Labeling and Topological Map Building

Óscar Martínez Mozos Cyrill Stachniss Axel Rottmann Wolfram Burgard

University of Freiburg, Department of Computer Science, D-79110 Freiburg, Germany
omartine|stachnis|rottmann|burgard@informatik.uni-freiburg.de

Summary. Indoor environments can typically be divided into places with different functionalities like corridors, kitchens, offices, or seminar rooms. We believe that the ability to learn such semantic categories from sensor data or in maps enables a mobile robot to more efficiently accomplish a variety of tasks such as human-robot interaction, path-planning, exploration, or localization. In this work, we first propose an approach based on supervised learning to classify the pose of a mobile robot into semantic classes. Our method uses AdaBoost to boost simple features extracted from vision and laser range data into a strong classifier. We furthermore present two main applications of this approach. Firstly, we show how our approach can be utilized by a moving robot for robust online classification of the poses traversed along its path using a hidden Markov model. Secondly, we introduce a new approach to learn topological maps from geometric maps by applying our semantic classification procedure in combination with probabilistic labeling. Experimental results obtained in simulation and with real robots demonstrate the effectiveness of our approach in various environments.

1.1 Introduction

In the past, many researchers have considered the problem of building accurate maps of the environment from the data gathered with a mobile robot. The question of how to augment such maps by semantic information, however, is virtually unexplored. Whenever robots are designed to interact with their users, semantic information about places can be important. It can furthermore be used to intuitively segment an environment into different places and learn accurate topological models.

In this work, we address the problem of classifying places of the environment of a mobile robot using range finder and vision data as well as building topological maps based on that knowledge. Indoor environments, like the one depicted in Figure 1.1, can typically be divided into areas with different functionalities such as laboratories, office rooms, corridors, or kitchens. Whereas some of these places have special geometric structures and can therefore be distinguished merely based on laser range data, other places can only be identified according to the objects found there like, for example, coffee machines in kitchens. To detect such objects, we use vision data acquired by a camera system.

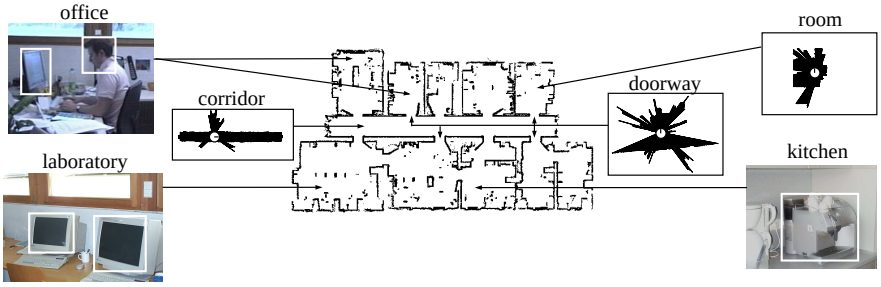


Fig. 1.1. An environment with offices, doorways, a corridor, a kitchen, and a laboratory. Additionally, the figure shows typical observations obtained by a mobile robot at different places.

In the approach described here, we apply the AdaBoost algorithm [7] to boost simple features, which on their own are insufficient for a reliable categorization of places, to a strong classifier for the semantic labeling of the poses of a robot in an indoor environment. Since the original version of AdaBoost provides only binary decisions, we determine the decision list with the best sequence of binary strong classifiers. We then use this semantic classifier in two main applications. Firstly, we show how to classify the different poses of a mobile robot along its trajectory by applying a hidden Markov model (HMM) which estimates the label of the current pose based on the current and the previous outputs of the semantic classifier. Secondly, we introduce a new approach to learn topological maps from occupancy grids. This is achieved by simulating the laser scans of a mobile robot at the corresponding locations and applying our semantic classification algorithm. We then apply a probabilistic relaxation algorithm to smooth the classification output, followed by a region extraction. Experimental results presented in this paper illustrate that our classification system yields recognition rates of more than 88% or 98% (depending on the number of classes to distinguish between). We also present experiments illustrating that the resulting classifier can even be used in environments from which no training data were available. This offers the opportunity to label places and to learn accurate topological maps from unknown environments.

In the past, several authors considered the problem of adding semantic information to places. Buschka and Saffiotti [4] describe a virtual sensor that is able to identify rooms from range data. Also Koenig and Simmons [11] apply a pre-programmed routine to detect doorways from range data. Althaus and Christensen [1] 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.* [20] train a neural network to estimate the location of a mobile robot in its environment using the odometry information and ultrasound data.

Learning algorithms have additionally been used to identify objects. For example, Anguelov *et al.* [2, 3] apply the EM algorithm to cluster different types of objects from sequences of range data and to learn the state of doors. Limketkai *et al.* [16] use relational Markov networks to detect objects like doorways based on laser range data. Furthermore, they employ Markov Chain Monte Carlo to learn the parameters

of the models. Treptow *et al.* [29] utilize the AdaBoost algorithm to track a soccer ball without color information. Finally, Torralba and colleagues [28] use hidden Markov models for learning places from image data.

Compared to these approaches, our algorithm is able to combine arbitrary features extracted from different sensors to form a sequence of binary strong classifiers to label places. Our approach is also supervised, which has the advantage that the resulting labels correspond to user-defined classes.

On the other hand, different algorithms for creating topological maps have been proposed. Kuipers and Byun [14] extract distinctive points in the map. These points are defined as local maxima using a measure of distinctiveness between locations. Kortenkamp and Weymouth [12] fuse the information obtained with vision and ultrasound sensors to determine topologically relevant places. Shatkey and Kaelbling [26] apply a HMM learning approach to learn topological maps in which the nodes represent points in the plane. Thrun [27] uses the Voronoi diagram to find critical points, which minimize the clearance locally. These points are then used as nodes in a topological graph. Choset [5] encodes metric and topological information in a generalized Voronoi graph to solve the simultaneous localization and mapping problem. Additionally, Kuipers and Beeson [13] apply different learning algorithms to calculate topological maps of environments of a mobile robot.

In contrast to these previous approaches, the technique described in this paper 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. The knowledge about semantic labels of places is used to build accurate topological maps with a mobile robot.

The rest of the chapter is organized as follows. In Section 1.2, we describe the sequential AdaBoost classifier. In Section 1.3, we present the application of a hidden Markov model to the online place classification with a moving mobile robot. Section 1.4 contains our approach for topological map building. Finally, Section 1.5 presents experimental results obtained using our methods.

1.2 Semantic Place Labeling using AdaBoost

One of the key problems to be solved is to define a classifier that allows us to categorize places in the environment according to a set of given categories. Rather than hand-coding such a classification system, our approach is to apply the AdaBoost algorithm to learn a strong classifier from a large set of simple features. In this section, we first present the AdaBoost algorithm and our approach to deal with multiple classes. We then describe the different features extracted from laser and vision data used in our current system.

1.2.1 The AdaBoost Algorithm

Boosting is a general method for creating an accurate strong classifier by combining a set of weak classifiers. The requirement for each weak classifier is that its accuracy is better than a random guessing. In this work we apply the boosting algorithm

AdaBoost in its generalized form presented by Schapire and Singer [25]. The input to this algorithm is a set of labeled training examples. The algorithm repeatedly selects a weak classifier $h_j(x)$ using a distribution D over the training examples. The selected weak classifier is expected to have a small classification error on the training data. The idea of the algorithm is to modify the distribution D by increasing the weights of the most difficult training examples in each round. The final strong classifier H is a weighted majority vote of the best T weak classifiers.

Throughout this work, we use the approach presented by Viola and Jones [30] in which the weak classifiers depend on single-valued features $f_j \in \mathbb{R}$. Two kinds of weak classifiers are created in our current system. In addition to the classifier defined by Viola and Jones, which has the form

$$h_j(x) = \begin{cases} +1 \text{ if } p_j f_j(x) < p_j \theta_j \\ -1 \text{ otherwise,} \end{cases} \tag{1.1}$$

where θ_j is a threshold and p_j is either -1 or $+1$ and thus represents the direction of the inequality, we designed a second type

$$h_j(x) = \begin{cases} p_j \quad \text{if } \theta_j^1 < f_j(x) < \theta_j^2 \\ -p_j \text{ otherwise,} \end{cases} \tag{1.2}$$

where θ_j^1 and θ_j^2 define an interval and p_j is either $+1$ or -1 indicating whether examples inside the interval are positive or negative. For both types of weak classifiers, the output is $+1$ or -1 indicating whether the classification is positive or negative. The AdaBoost algorithm determines for each weak classifier $h_j(x)$ the optimal parameters, such that the number of misclassified training examples is minimized. The final AdaBoost algorithm place categorization is shown in Algorithm 1.1.

The AdaBoost algorithm has been designed for binary classification problems. To classify places in the environment, we need the ability to handle multiple classes.

Algorithm 1.1 Generalized version of AdaBoost for place categorization.

Input: Set of N labeled examples $(x_1, y_1), \ldots, (x_N, y_N)$, where $y_n = +1$ for positive examples and $y_n = -1$ for negative examples.

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

for $t = 1, \ldots, T$ **do**

1. Normalize the weights $D_t(n)$ so that $\sum_{n=1}^{N} D_t(n) = 1$.
2. For each feature f_j train a weak classifier h_j using D_t.
3. For each classifier h_j calculate $r_j = \sum_n D_t(n) y_n h_j(x_n)$, with $h_j(x_n) \in \{-1, +1\}$.
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 $\alpha_t = \frac{1}{2} \ln(\frac{1+r_t}{1-r_t})$.

end for

Output: The final strong hypothesis $H(x) = \text{sign}(F(x))$, where $F(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$.

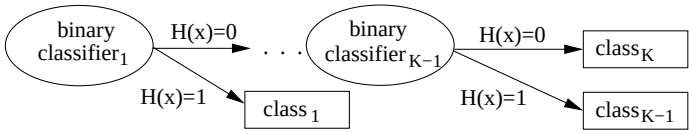


Fig. 1.2. A decision list classifier for K classes using binary classifiers.

To achieve this, we use a sequence of binary classifiers, where each element of such a sequence determines if an example belongs to one specific class. If the binary classifier returns a positive result, the example is assumed to be correctly classified. Otherwise, it is recursively passed to the next element in this list. Figure 1.2 illustrates the structure of such a decision list classifier.

In our current system, we typically consider a small number of classes which makes it feasible to evaluate all potential sequences and choose the best order of 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 of classes. In practice, we found out that the heuristic which sorts the classifiers in decreasing order according to their classification rate 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 six different classes performed in average only 1.3% worse as shown by Rottmann *et al.* [23].

To evaluate the performance of the decision list, we compared it to the AdaBoost.M2 [7] algorithm, which is a multi-class variant of AdaBoost. In our experiments, the sequential AdaBoost classifier yields better results than the AdaBoost.M2 algorithm. A more detailed comparison between both algorithms can be found in the work by Martínez Mozos [17].

1.2.2 Features from Vision and Laser Data

In this section, we describe the features used to create the weak classifiers in the AdaBoost algorithm. Our robot is equipped with a 360 degree 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 1.1 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. All features are invariant with respect to rotation to make the classification of a pose dependent only on the position of the robot and not on its orientation. Most of our laser features are standard geometrical features used for shape analysis [9, 24]. Typical examples considered by our system are illustrated in Figure 1.3. A detailed list of laser features is contained in our previous work [18]. In the system described here, we implemented several additional features which are listed in Table 1.1.

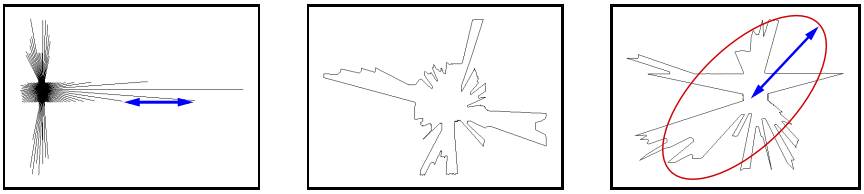


Fig. 1.3. Examples for features generated from laser data, namely the average distance between two consecutive beams, the perimeter of the area covered by a scan, and the mayor axis of the ellipse that approximates the polygon described by the scan.

Table 1.1. New Laser Features

1. Average and standard deviation of the fraction between the length of two consecutive beams.
2. Average and standard deviation of the fraction between the length of two consecutive beams divided by the maximum beam length.
3. Circularity. Let $\mathbf{P}$ be the perimeter of the area covered by the beams and $\mathbf{A}$ be the area covered by the beams. The circularity is defined as $\mathbf{P}^2/\mathbf{A}$.
4. Average and standard deviation of the distance from the centroid of $\mathbf{A}$ to the shape boundary of $\mathbf{A}$, divided by the maximum distance to the shape boundary.
5. Number of gaps. Two consecutive beams form a gap if the fraction between the first and the second is smaller than a threshold.
6. Kurtosis. The kurtosis is defined as

$$\frac{\sum_{i=1}^{N} (\mathrm{length}(\mathrm{beam}_i) - \overline{l})^4}{N \cdot \sigma^4} - 3$$

where $\overline{l}$ is the average beam length and σ the corresponding standard deviation.

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 Eq. (1.1) and Eq. (1.2). 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 1.1. 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.* [15].

In case the observations do not cover a 360 degree field of view, the property of the rotational invariance is lost. In such a situation, we expect that more training data will be necessary and that the classification will be less robust.

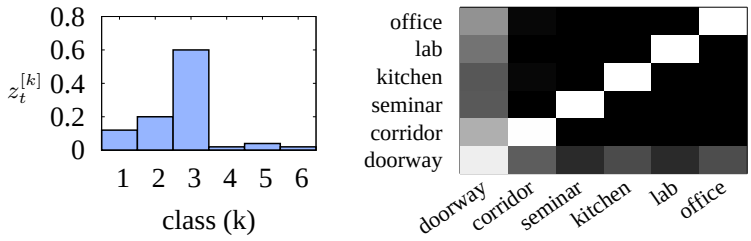


Fig. 1.4. The left image illustrates a classification output z. The right image depicts probabilities of possible transitions between places in the environment. To increase the visibility, we used a logarithmic scale. Dark values indicate low probability.

1.3 Probabilistic Classification of Trajectories

The approach described so far is able to classify single observations only but 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 a hidden Markov model (HMM) and maintain a posterior $Bel(l_t)$ about the type of the place l_t the robot is currently at

$$Bel(l_t) = \alpha P(z_t \mid l_t) \sum_{l_{t-1}} P(l_t \mid l_{t-1}, u_{t-1}) Bel(l_{t-1}). \tag{1.3}$$

In this equation, α is a normalizing constant ensuring that the left-hand side sums up to one over all l_t. To implement this HMM, three components need to be known. First, we need to specify the observation model $P(z_t \mid l_t)$ which is the likelihood that the classification output is z_t given the actual class is l_t. Second, we need to specify the transition model $P(l_t \mid l_{t-1}, u_{t-1})$ which defines the probability that the robot moves from class l_{t-1} to class l_t by executing action u_{t-1}. Finally, we need to specify how the belief $Bel(l_0)$ is initialized.

In our current system, we choose a uniform distribution to initialize $Bel(l_0)$. Furthermore, the classification output z_t is represented by a histogram, as illustrated in the left image of Figure 1.4. 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 our decision list (compare Figure 1.2). To compute the individual values for each bin of that histogram, we use the approach by Friedman *et al.* [8]. It determines a confidence value $C \in [0, 1]$ for a positive binary classification

$$C = P(y = +1 \mid x) = \frac{e^{F(x)}}{e^{-F(x)} + e^{F(x)}}, \tag{1.4}$$

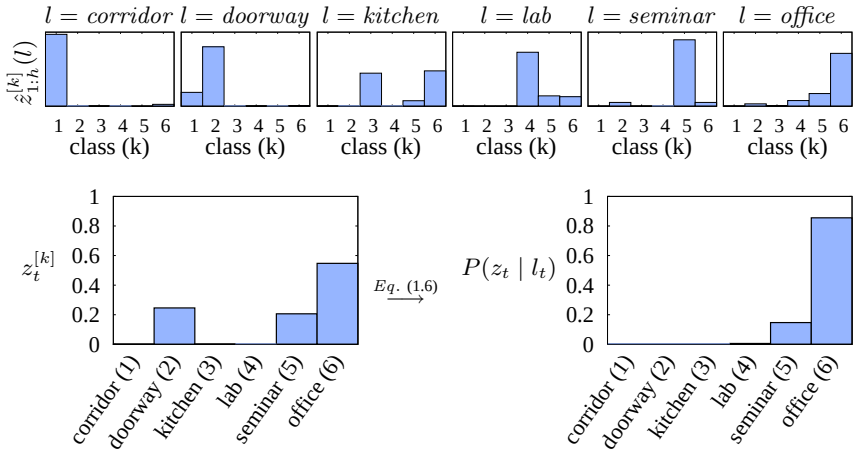


Fig. 1.5. The distributions depicted in the first row show the learned histograms $\hat{z}_{1:h}(l)$ for the individual classes (here corridor (1), doorway (2), kitchen (3), lab (4), seminar room (5), and office (6)). The left image in the second row depicts a possible classification output z_t. In the right image, each bar represents the corresponding likelihood $P(z_t \mid l_t)$ for the different estimates of l_t.

where $F(x)$ is the output of the AdaBoost algorithm according to Algorithm 1.1. Let C_k refer to the confidence value of the k-th binary classifier in our decision list. The probability that the location belongs to the k-th class is given by the k-th bin of the histogram z computed as

$$z^{[k]} = C_k \prod_{j=1}^{k-1} (1 - C_j). \tag{1.5}$$

Note that the confidence value C_K which is used to compute the last bin $z^{[K]}$ of the histogram holds $C_K = 1$ according to the structure of the decision list (compare Figure 1.2).

To determine $P(z_t \mid l_t)$, we use the KL-divergence [6] between two distributions. The first distribution is the current classification output z_t. The second one is learned from a statistics: for each class l, we compute a histogram $\hat{z}_{1:h}(l)$ using h observations recorded within a place belonging to class l (here $h = 50$). This histogram $\hat{z}_{1:h}(l)$ is obtained by averaging over the individual histograms $\hat{z}_1, \ldots, \hat{z}_h$, which are computed according to Eq. (1.5). To determine $P(z_t \mid l_t)$, we use the KL-divergence $kld(\cdot \parallel \cdot)$ which provides a measure about the similarity of two distributions

$$P(z_t \mid l_t) = e^{-kld(z_t \parallel \hat{z}_{1:h}(l_t))}. \tag{1.6}$$

To illustrate the computation of the observation likelihood $P(z_t \mid l_t)$ consider Figure 1.5. The first row depicts examples for the histograms $\hat{z}_{1:h}(l)$. 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 $\hat{z}_{1:h}(l_t)$ is than used to compute $P(z_t \mid l_t)$ according to Eq. (1.6). The result for all classes is depicted in the right image in the second row. In this image, each bin represents the likelihood $P(z_t \mid l_t)$ for the individual classes l_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 $P(l_t \mid l_{t-1}, u_{t-1})$, 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 20cm and 50cm. These values correspond to typical distances traveled by the robot between two consecutive updates of the HMM. The finally obtained transition probability matrix $P(l_t \mid l_{t-1}, u_{t-1})$ for the action $Move$ is depicted in the right image of Figure 1.4. 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 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.

1.4 Topological Map Building

The second application of our classification system is learning topological maps from occupancy grids. To take into account spatial dependencies between neighboring places, we apply a probabilistic relaxation labeling. Additionally, we describe how to perform the region extraction and the final creation of a graph representing the topological structure of the environment.

1.4.1 Probabilistic Relaxation Labeling

One of the key problems that need to be solved in order to learn accurate topological maps, in which the nodes correspond to the individual rooms an the environment, is to eliminate classification errors. In this section, we apply the probabilistic relaxation labeling, which has been introduced by Rosenfeld *et al.* [21], to smooth the classifications based on neighborhood relations.

Probabilistic relaxation labeling 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 which 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. Finally, $\mathcal{C} = \{c_{ij} \mid v_j \in \mathcal{N}(v_i)\}$ is 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

$$p_i^{(r+1)}(l) = \frac{p_i^{(r)}(l)\left[1 + q_i^{(r)}(l)\right]}{\sum_{l'=1}^{L} p_i^{(r)}(l')\left[1 + q_i^{(r)}(l')\right]}, \tag{1.7}$$

where

$$q_i^{(r)}(l) = \sum_{j=1}^{M} c_{ij}\left[\sum_{l'=1}^{L} r_{ij}(l, l') p_j(l')\right]. \tag{1.8}$$

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 [31]

$$r_{ij}(l, l') = \begin{cases} \frac{1}{1 - p_i(l)}\left(1 - \frac{p_i(l)}{p_{ij}(l|l')}\right) & \text{if } p_i(l) < p_{ij}(l \mid l') \\ \frac{p_{ij}(l|l')}{p_i(l)} - 1 & \text{otherwise,} \end{cases} \tag{1.9}$$

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'.

So far we 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 AdaBoost classifier. To learn a topological map, we assume a given two-dimensional occupancy grid map [19] 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, this node is connected to all immediate neighbors of that cell

$$\begin{aligned} \mathcal{N}_8(v_{(x,y)}) = \{\ & v_{(x-1,y-1)}, v_{(x-1,y)}, v_{(x-1,y+1)}, v_{(x,y-1)}, \\ & v_{(x,y+1)}, v_{(x+1,y-1)}, v_{(x+1,y)}, v_{(x+1,y+1)}\ \}. \end{aligned} \tag{1.10}$$

For the initial probabilities $p^{(0)}_{(x,y)}(l)$, we use the output of the classifier described in Section 1.2.1. Our set of labels $\mathcal{L}$ is composed of the labels *corridor*, *doorway*, *room*, 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 (1.5). The classification output z for each pose (x, y) is used to initialize the probability distribution $P^{(0)}_{(x,y)}$ of node $v_{(x,y)}$.

For the nodes lying in the free space, the probability $p^{(0)}_{(x,y)}(wall)$ of being a wall is initialized with 0. Accordingly, the nodes corresponding to occupied cells in the map are initialized with $p^{(0)}_{(x,y)}(wall) = 1$.

Each of the weights $c_{ij} \in \mathcal{C}$ 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 (1.9). The values $p_i(l)$ and $p_{ij}(l \mid l')$ are obtained from statistics in the given (occupancy grid) map corresponding to the training data as will be described in Section 1.5.

1.4.2 Region Extraction and Topological Mapping

We define a region λ_l on an adjacency graph $\mathcal{A}$ as a set of eight-connected nodes with the same label l. For example, the region λ_{room} represents a room in the corresponding occupancy grid map. If there is a different region with the label *room*, this will represent a different room in the map. For each label $l \in \{corridor, room, doorway\}$, regions are extracted from the adjacency graph using the algorithm by Rosenfeld and Pfaltz [22]. In an analog way, we extract the connections between regions.

Finally, a topological graph $\mathcal{T} = (\mathcal{V}_\mathcal{T}, \mathcal{E}_\mathcal{T})$ is constructed so that each node $v_i \in \mathcal{V}_\mathcal{T}$ represents a region and each edge $e_s \in \mathcal{E}_\mathcal{T}$ represents a connection. The topological graph forms the resulting topological map. We finally apply a heuristic region correction step 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 $1\mathrm{m}^2$ compared to the training set as classification error and assign the label of one of its connected regions to it.
2. We mark each region labeled as doorway whose size does not exceed a given threshold of $0.1\mathrm{m}^2$ square meters or that is connected to only one region as false classification and assign the label of one of its connected regions to it.

1.5 Experiments

The approach described above has been implemented and tested on real robots as well as in simulation. The robots used to carry out the experiments were an ActivMedia Pioneer 2-DX8 equipped with two SICK laser range finders as well as an iRobot B21r robot which is additionally equipped with a camera system.

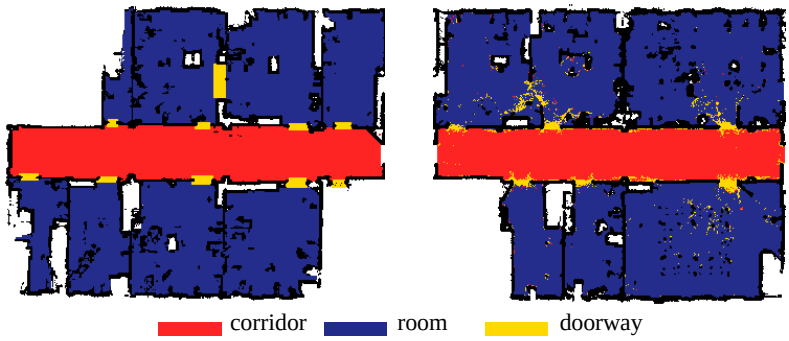


Fig. 1.6. Whereas the left image depicts the training data, the right image shows the classification result on the test set. The training and test data were obtained by simulating laser range scans in the map.

The goal of the experiments is to demonstrate that our 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 was available. Furthermore, we demonstrate the advantages of utilizing the vision information to distinguish between different rooms like, e.g., kitchens, offices, or seminar rooms. Additionally, we illustrate the advantages of the HMM filtering for classifying places with a moving mobile robot. Throughout these experiments, the term classification result refers to the most likely class reported by the HMM or respectively by the sequence of binary classifiers. Furthermore, we present results applying our method for semantic topological maps. We first show the results for a typical office environment. Then, we present an experiment illustrating that our approach is able to construct a topological map of a completely new environment

1.5.1 Results with the Sequential Classifier using Laser Data

The first experiment was performed using simulated data from our office environment in building 79 at the University of Freiburg. The task was to distinguish between three different types of places, namely rooms, doorways, and a corridor based on laser range data only. In this experiment, we solely applied the sequential classifier without the HMM filtering. For the sake of clarity, we separated the test from the training data by dividing the overall environment into two areas. Whereas the left part of the map contains the training examples, the right part includes only test data (see Figure 1.6). The optimal decision list for this classification problem, in which the robot had to distinguish between three classes, is room-doorway. This decision list correctly classifies 93.9% of all test examples (see right image of Figure 1.6). For alternative training and test sets we obtained similar success rates. The worst configurations of the decision list are those in which the doorway classifier is in the first place. This is probably due to the fact, that doorways are hard to detect because typically most parts of a range scan obtained in a doorway cover the adjacent room

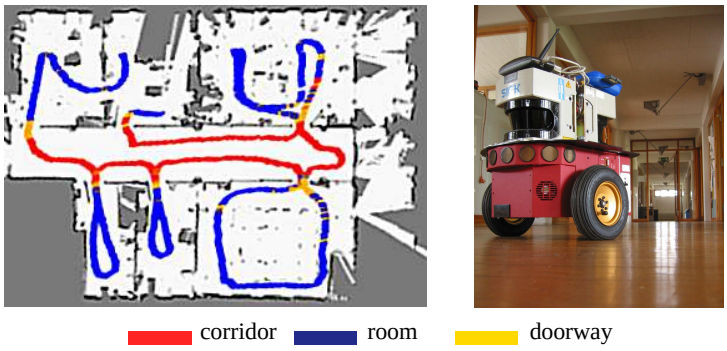


Fig. 1.7. The left image depicts a trajectory of a robot and the corresponding classifications based on real laser data. The robot used in this experiment is depicted in the right image.

and the corridor. The high error in the first element of the decision list then leads to a high overall classification error.

The next experiment has been carried out with a real mobile robot that we manually steered through the environment. We used the same classifier as in the previous experiment. The trajectory including the corresponding classification results as well as the mobile robot are depicted in Figure 1.7. As can be seen from this figure, the learned classifier yields a robust labeling also for real robot data.

Additionally, we performed an experiment using a map of the entrance hall at the University of Freiburg which contained four different classes, namely rooms, corridors, doorways, and hallways. The optimal decision list is corridor-hallway-doorway with a success rate of 89.5%.

1.5.2 Transferring the Classifiers to New Environments

The second 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 this experiment, we trained our sequential classifier in the left half of the map shown in Figure 1.1. In the right half of this environment, our approach was able to correctly classify 97% of all places. The resulting classifier was then evaluated on scans simulated given the map of the Intel Research Lab in Seattle depicted in Figure 1.8. Although the classification rate decreased to 86.0%, the result indicates that our algorithm yields good generalizations which can also be applied to correctly label places of so far unknown environments. Note that a success rate of 86.0% is quite high for this environment, since even humans typically cannot consistently classify the different places.

1.5.3 Classification of Trajectories using HMM Filtering

The third experiment was performed using real laser and vision data obtained in an office environment, which contains six different types of places, namely offices,

Fig. 1.8. The left map depicts the occupancy grid map of the Intel Research Lab and the right image depicts the classification results obtained by applying the classifier learned from the environment depicted in Figure 1.1 to this environment. The fact that 86.0% of all places could be correctly classified illustrates that the resulting classifiers can be applied to so far unknown environments.

doorways, a laboratory, a kitchen, a seminar room, and a corridor. The true classification of the different places in this environments is shown in Figure 1.9.

The classification performance of the classifier along a sample trajectory taken by a real robot is shown in left image of Figure 1.10. The classification rate in this experiment is 82.8%. If we additionally apply the HMM for temporal filtering, the classification rate increases up to 87.9%. The labeling obtained with the HMM is shown in the right image of Figure 1.10.

A further experiment was carried out using test data obtained in a different part of the same building. We applied the same classifier as in the previous experiment. Whereas the sequential classifier yields a classification rate of 86.0%, the combination with the HMM generated the correct answer in 94.7% of all cases. A two-sample t-test applied to the classification results obtained along the trajectories for both experiments showed that the improvements introduced by the HMM are significant on the $\alpha = 0.05$ level. Furthermore, we classified the same data based solely on the laser features and ignoring the vision information. In this case, only 67.7% could be classified correctly without the HMM. The application of the HMM increases the classification performance to 71.7%. These three experiments illustrate that the HMM seriously improves the overall rate of correctly classified places. Moreover, the third experiment shows that only the laser information is not sufficient to distinguish robustly between places with similar structure (see *office* and *kitchen* in Figure 1.10).

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

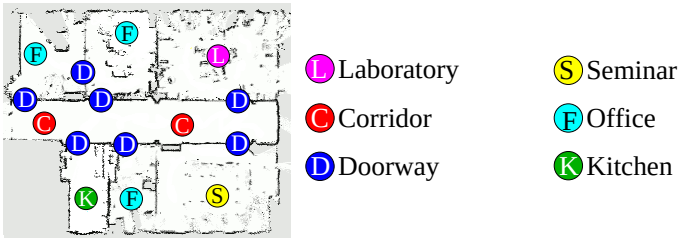


Fig. 1.9. Ground truth labeling of the individual areas in the environment.

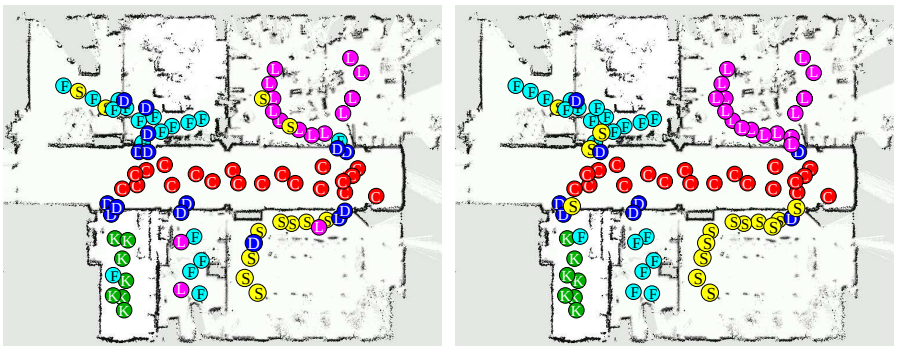


Fig. 1.10. The left image depicts a typical classification result for a test set obtained using only the output of the sequence of classifiers. The right image shows the resulting classification in case a HMM is additionally applied to filter the output of the sequential classifier.

Fig. 1.11. Improvement of the HMM according to the percentage of weak classifiers used in each of the binary AdaBoost classifiers.

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 experiments (Figure 1.10). For example, the classification rate decreases to 60% if only 5% of the weak classifiers are used. The results are shown in Figure 1.11. In average, the HMM improves the classification rate by 5.0%.

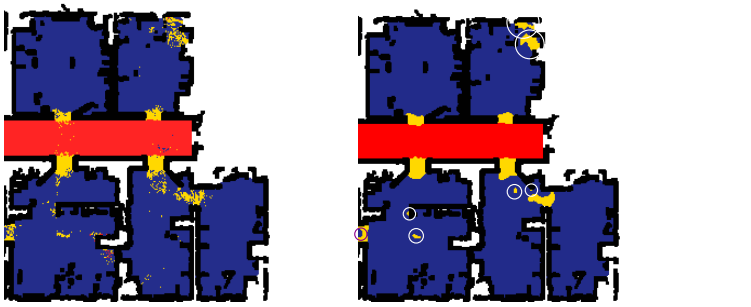
1.5.4 Building Topological Maps

The next experiment is designed to analyze our approach to building topological maps. It was carried out in the office environment depicted in the motivating example shown in Figure 1.1. The length of the corridor in this environment is approx. 20 m. After applying the sequential AdaBoost classifier (see Figure 1.12(a)), the classification of the test set was 97%. Then, we applied the probabilistic relaxation method for 50 iterations. As can be seen from Figure 1.12(b), this method generates more compact regions and eliminates noise. Finally, the topological map is created using the connections between regions. Some regions detected as doorways do not correspond to real doorways and are marked with circles. After applying the steps described in Section 1.4.2 on the corresponding topological map, these false doorways are eliminated. The final result gives a classification rate of 98.7% for all data points. The different steps of the process are illustrated as colors/grey levels in Figure 1.12. The doorway between the two right-most rooms under the corridor is correctly detected (Figure 1.12(c)). Therefore, the rooms are labeled as two different regions in the final topological map.

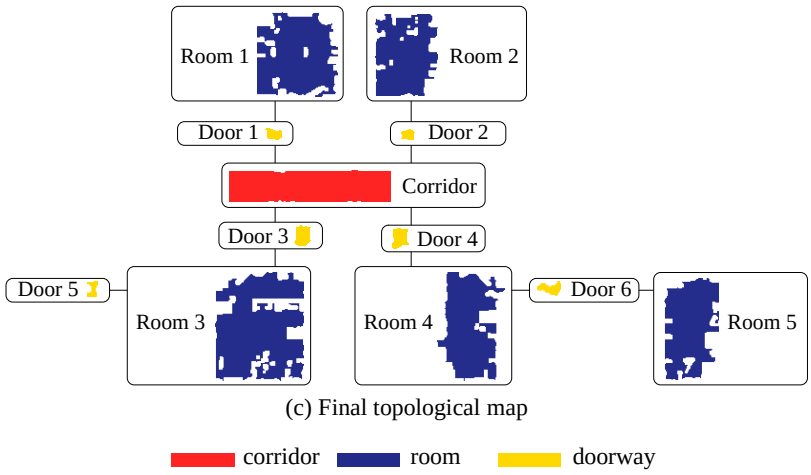
1.5.5 Learning Topological Maps of Unknown Environments

This experiment is designed to analyze whether our approach can be used to create a topological map of a new unseen environment. To carry out the experiment we trained a sequential AdaBoost classifier using the training examples of the maps shown in Figure 1.6 and Figure 1.12 with different scales. In this case only the classes *room* and *corridor* were used in the training process. The resulting classifier was then evaluated on scans simulated in the map denoted as "SDR site B" in Radish [10]. This map represents an empty building in Virginia, USA. The corridor is approx. 26 meters long. The whole process for obtaining the topological map is depicted in Figure 1.13. The Adaboost classifier gives a first classification of 92.4%. As can be seen in Figure 1.13(d), 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. In the final topological map, 96.9% of the data points are correctly classified.

We also analyzed the results obtained without applying the relaxation process. Not using relaxation had several effects. Firstly, omitting the relaxation procedure reduces the classification rate. Secondly, the finally obtained regions are typically 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 of the SDR building contained four incorrect nodes without relaxation, whereas there were only two incorrect nodes when we used the probabilistic relaxation.



(a) Sequential classification (b) Relaxation (circles identify incorrect regions)



(c) Final topological map

Fig. 1.12. This figure shows in image (a) the result of applying the sequential AdaBoost with a classification rate of 97%. (b) the result after applying relaxation including some incorrectly labeled regions (marked with circles), and finally in image (c) the final tropological map with the corresponding regions.

1.6 Conclusion

In this paper, we presented a novel approach to classify different places in the environment of a mobile robot into semantic classes, like rooms, hallways, corridors, offices, kitchens, or doorways. Our algorithm uses simple geometric features extracted from a single laser range scan and information extracted from camera data and applies the AdaBoost algorithm to form a binary strong classifier. To distinguish between more than two classes, we use a sequence of strong binary classifiers arranged in a decision list.

We presented two applications of our approach. Firstly, we perform an online classification of the positions along the trajectories of a mobile robot by filtering the classification output using a hidden Markov model. Secondly, we present a new ap-

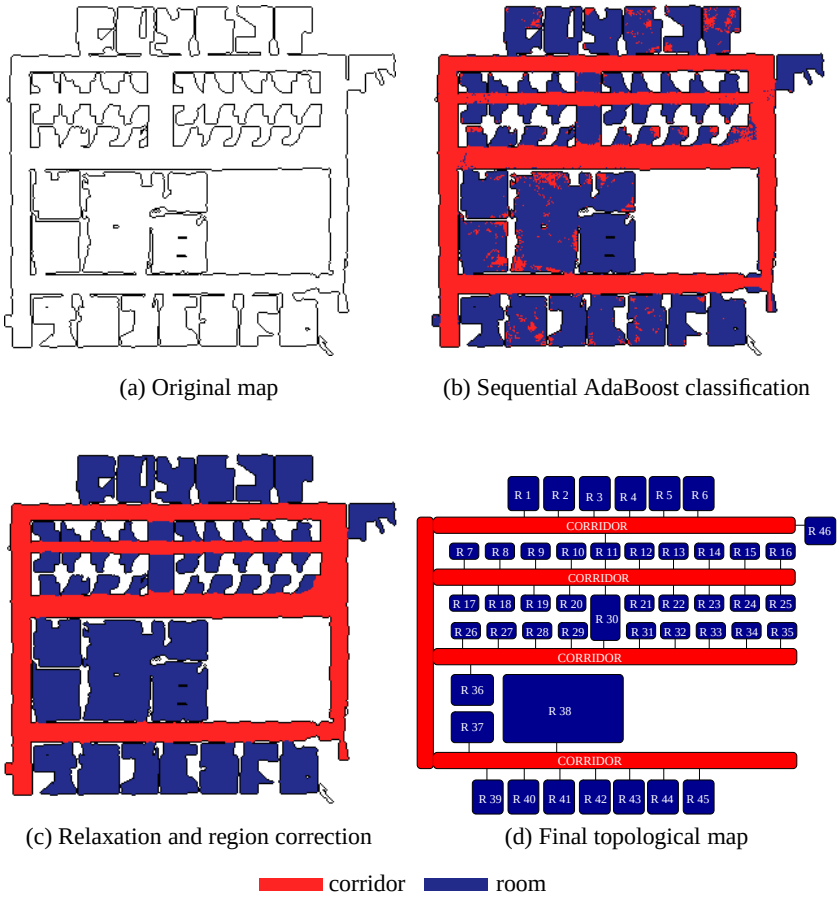


Fig. 1.13. This figure shows (a) the original map of the building, (b) the results of applying the sequential AdaBoost classifier with a classification rate of 92.4%, (c) the resulting classification after the relaxation and region correction, and (d) the final topological map with semantic information. The regions are omitted in each node. The rooms are numbered left to right and top to bottom with respect to the map in (a). For the sake of clarity, the corridor-node is drawn maintaining part of its region structure.

proach to create topological graphs from occupancy grids by applying a probabilistic relaxation labeling to take into account dependencies between neighboring places to improve the classifications.

Experiments carried out using real robots as well as in simulation illustrate that our technique is well-suited to reliably label places in different environments. It allows us to robustly separate different semantic regions and in this way it is able to learn topologies of indoor environments. Further experiments illustrate that a learned classifier can even be applied to so far unknown environments.

Acknowledgment

This work has partly been supported by the German Research Foundation (DFG) under contract number SFB/TR-8 (A3) as well as under contract number GRK 1103/1 and by the EC under contract number FP6-004250-CoSy. Furthermore, we would like to thank Andrew Howard for providing the map of the SDR building.

References

1. P. Althaus and H.I. Christensen. Behaviour coordination in structured environments. *Advanced Robotics*, 17(7):657–674, 2003.
2. D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2002.
3. D. Anguelov, D. Koller, Parker E., and S. Thrun. Detecting and modeling doors with mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2004.
4. P. Buschka and A. Saffiotti. A virtual sensor for room detection. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 637–642, 2002.
5. H. Choset. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 2001.
6. T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley & sons, 1991.
7. Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.
8. J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.
9. R.C. Gonzalez and P. Wintz. *Digital Image Processing*. Addison-Wesley Publishing Inc., 1987.
10. A. Howard and N. Roy. Radish: The robotics data set repository.
11. S. Koenig and R. 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.
12. D. Kortenkamp and T. Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proc. of the Twelfth National Conference on Artificial Intelligence*, pages 979–984, 1994.
13. B. Kuipers and P. Beeson. Bootstrap learning for place recognition. In *Proc. of the Nat. Conf. on Artificial Intelligence (AAAI)*, 2002.
14. B. Kuipers and Y.T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, 8 1981.
15. R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM, 25th Pattern Recognition Symposium*, 2003.
16. B. Limketkai, L. Liao, and D. Fox. Relational object maps for mobile robots. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 1471–1476, Edinburgh, Scotland, 2005.

17. O. Martínez Mozos. Supervised learning of places from range data using adaboost. Master's thesis, University of Freiburg, Department of Computer Science, 2004.
18. O. Martínez Mozos, C. Stachniss, and W. Burgard. Supervised learning of places from range data using adaboost. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 1742–1747, Barcelona, Spain, 2005.
19. H.P. Moravec and A.E. Elfes. High resolution maps from wide angle sonar. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1985.
20. S. Oore, G.E. Hinton, and G. Dudek. A mobile robot that learns its place. *Neural Computation*, 9(3):683–699, 1997.
21. A. Rosenfel, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Trans. Systems. Man. Cybernet*, 6(6):420–433, 1976.
22. A. Rosenfeld and J. L. Pfaltz. Sequential operations in digital picture processing. *Journal of the Association for Computing Machinery*, 13(4):471–494, 1966.
23. A. Rottmann. Bild- und laserbasierte klassifikation von umgebungen mit mobilen robotern. Master's thesis, University of Freiburg, Department of Computer Science, 2005. In German.
24. J.C. Russ. *The Image Processing Handbook*. CRC Press, 1992.
25. R.E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.*, 37(3):297–336, 1999.
26. H. Shatkey and L.P. Kaelbling. Learning topological maps with weak local odometric information. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 1997.
27. S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.
28. A. Torralba, K. Murphy, W. Freeman, and M. Rubin. Context-based vision system for place and object recognition. In *Proc. of the Int. Conf. on Computer Vision (ICCV)*, 2003.
29. A. Treptow, A. Masselli, and A. Zell. Real-time object tracking for soccer-robots without color information. In *Proc. of the Europ. Conf. on Mobile Robots (ECMR)*, 2003.
30. P. Viola and M.J. Jones. Robust real-time object detection. In *Proc. of IEEE Workshop on Statistical and Theories of Computer Vision*, 2001.
31. H. Yamamoto. A method of deriving compatibility coefficents for relaxation operators. *Compt. Graph. Image Processing*, 10:256–271, 1979.