Chapter 4
Topological Map Extraction with Semantic Information

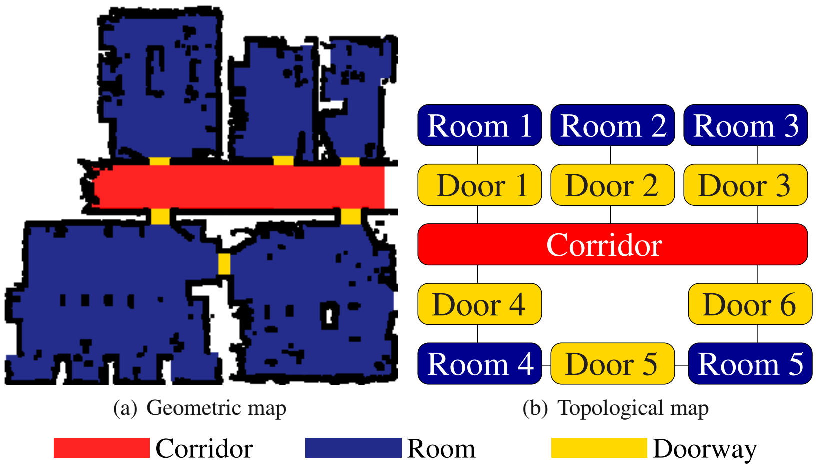
4.1 Introduction

In the previous chapter we saw 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 such as 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 also be communicated more easily to users of a mobile robot. Many researchers have considered the problem of building topological maps of the environment from the data gathered with a mobile robot. However, few techniques exist that permit semantic information to be added to these maps.

In this chapter, we consider the problem of learning topological maps with semantic information from occupancy grid 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 Fig. 4.1(a), 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 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. Figure 4.1(b) depicts a possible topological representation for the map in Fig. 4.1(a)

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 using local and neighboring information. By local information we mean the set of geometrical features the robot obtains from a laser observation at a concrete location (cf. Chap. 3). By neighboring information we refer to the semantic information from the neighboring locations.



(a) Geometric map (b) Topological map

Fig. 4.1 (**a**) Geometric map of a typical indoor environment with rooms, doorways, and a corridor, depicted in colors/gray levels. (**b**) Corresponding semantic-topological map.

Two different methods are presented which use both local and neighboring information for the final classification. The first approach determines the semantic class of each unoccupied cell of a grid map. 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 Chap. 3 (cf. Fig. 3.2). The classification is then done using a sequence of classifiers learned with the AdaBoost algorithm arranged in a probabilistic decision list in a similar way as introduced in Sect. 3.3. However, in this chapter we present some modifications in the learning and classification process which permit the use of probability values for the different classifications. Finally, 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 semantic classification at one position inside the map is done using simultaneously the local information and the relation between semantic labels from neighboring positions. We apply a variant of AMNs called instance-based associative Markov networks (iAMNs). This concrete approach combines AMNs with nearest-neighbor techniques.

Experimental results shown in this chapter illustrate that these methods can determine the semantic-topological map of an environment with high recognition rates. We also present results that illustrate that this approach can even construct a topological map of an environment from which no training data was available. Finally, we extend the set of simple features used in Chap. 3 with new ones. As the experimental results illustrate, the newly created set provide better classification results.

The rest of the chapter is organized as follows. In the next section we introduce the application of the generalized AdaBoost for the concrete task of place recognition. A probabilistic version of a decision list is presented in Sect. 4.3. The extended set of geometric features is described in Sect. 4.4. The probabilistic relaxation approach is presented in Sect. 4.5. Instance-based associative Markov networks are introduced in Sect. 4.6. Section 4.7 describes the method used to extract semantic regions and to create the final topological map. In Sect. 4.8, experimental results are presented. We discuss related work in Sect. 4.9. Finally, we conclude in Sect. 4.10.

4.2 Generalized AdaBoost

In this chapter we use the variant of the AdaBoost algorithm known as generalized AdaBoost [19]. As introduced in Sect. 2.2.2, this version has several advantages over the original AdaBoost algorithm. In addition, its output can be easily converted into a confidence value.

The input to the generalized AdaBoost is also composed of a set of labeled training examples $(x_n, y_n), n = 1, \ldots, N$. However, the label for the examples is in this case $y_n = +1$ when x_n is positive, and $y_n = -1$ when x_n is negative. Similar to the original AdaBoost, during the different iterations $t = 1, \ldots, T$ the algorithm selects a weak classifier with small error in the weighted training examples. The weight distribution D_t is changed on each iteration to give more importance to the most difficult examples. The final strong classifier is composed of a weighted majority sum of the selected weak hypotheses.

Following the approach presented in Sect. 3.2, each weak classifier is based on single-valued features f_j and has the form

$$h_j(x) = \begin{cases} +1 \text{ if } p_j f_j(x) < p_j \theta_j \\ -1 \text{ otherwise} . \end{cases} \tag{4.1}$$

Equation (4.1) differs from (3.1) in the output for a negative classification, which in this case is -1. The final generalized AdaBoost algorithm modified for the concrete task of place labeling is given in Fig. 4.2.

Using the generalized version of the AdaBoost algorithm shown in Fig. 4.2, and following the method suggested in [5], we can additionally compute a confidence value $C^+ \in [0, 1]$ for a positive binary classification of a new example as

$$C^+ = P(y = +1 \mid x) = \frac{e^{F(x)}}{e^{-F(x)} + e^{F(x)}} , \tag{4.2}$$

where $F(x)$ is the output of the algorithm according to Fig. 4.2. If the example is classified as negative, the positive confidence value can be calculated as

- Input:
  - Set of N labeled examples $(x_1, y_1), \ldots, (x_N, y_N)$ with $y_n = +1$ if the example x_n is positive, and $y_n = -1$ if the example x_n is negative
  - Integer T specifying the 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
  $$r_j = \sum_{n=1}^{N} D_t(n) y_n h_j(x_n) ,$$
  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
$$H(x) = \operatorname{sign}(F(x)) ,$$
where
$$F(x) = \sum_{t=1}^{T} \alpha_t h_t(x) .$$

Fig. 4.2 The generalized version of the AdaBoost algorithm for place labeling using laser-based features.

$$C^+ = P(y = +1 \mid x) = 1 - C^- , \tag{4.3}$$

with

$$C^- = P(y = -1 \mid x) = \frac{e^{-F(x)}}{e^{-F(x)} + e^{F(x)}} . \tag{4.4}$$

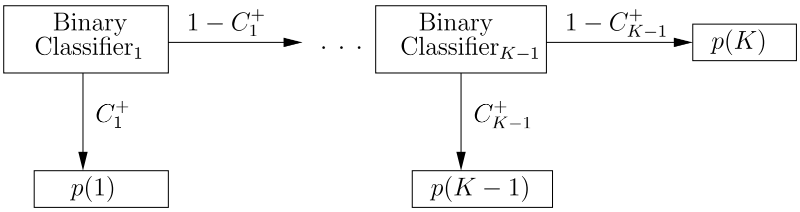


Fig. 4.3 A decision list classifier for K classes using binary classifiers. The output of each binary classifier $p(k)$ contains the probability that the classified example belongs to the k-th class.

4.3 Probabilistic Decision List

Extending the ideas introduced in Sect. 3.3, we use a probabilistic decision list to create a classifier for multiple classes. Each element of such a list represents one binary classifier which determines if an example belongs to one specific class. In addition, each binary classifier outputs a confidence value C_k^+ for a positive classification of its class k. Figure 4.3 illustrates the structure of a probabilistic decision list.

In this decision list, each test example is fed into the first binary classifier, which outputs a confidence value C^+ for a positive classification. The example is also passed to the next binary classifier, but with a negative confidence value $1 - C^+$. This process is repeated until the last element in the list. The complete output of the decision list is represented by a histogram P. In this histogram, the bin $p(k)$ stores the probability that the classified location belongs to the k-th class according to the sequence of classifiers in the decision list. Let C_k^+ refer to the positive confidence value of the k-th binary classifier in our decision list. Then, the probability that the example to be classified belongs to the k-th class is given by the bin $p(k)$ of the histogram P computed as

$$p(k) = C_k^+ \prod_{j=1}^{k-1} (1 - C_j^+), \tag{4.5}$$

whereas for the confidence value C_K^+ of the last bin holds $C_K^+ = 1$ according to the structure of the decision list in Fig. 4.3. An example of a histogram for six classes is illustrated in Fig. 4.4.

To select the order of the different binary classifiers we try all possible combinations and choose the one with best classification rates. Each binary classifier is trained in a one-against-all fashion, selecting one class as positive examples and the rest of the classes as negative examples. This is similar to the approach introduced in Sect. 3.3.

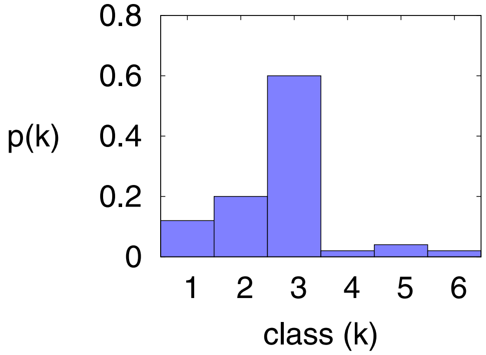


Fig. 4.4 An example of a classification output for the decision list of Fig. 4.3 using six classes.

4.4 New Geometrical Features from Sensor Range Data

As explained in Chap. 3, in order to classify each free cell in the occupancy grid map we simulate a range scan in its position ray-casting in the map. The simulated scans correspond to a robot equipped with a 360^o degree field of view laser sensor. Each simulated laser observation then consists of 360 beams. Each training example for the AdaBoost algorithm consists of the features extracted from the laser observation, together with its classification. Moreover, we assume that the classification of the training examples is given in advance. The single-valued features used in the AdaBoost algorithm are geometrical features used for shape analysis [7, 9, 13, 15, 18]. The features are selected to be rotationally invariant to make the classification of a pose dependent only on the (x,y)-position of the robot and not on its orientation. A feature f is defined as a function that takes as argument one observation $z \in Z$ and returns a real value: $f(Z) \to \mathbb{R}$, with Z being the set of all observations. In this chapter we apply an extended set of the features introduced in Sect. 3.4. The following list extends the original set A, which contains features calculated from the raw beams of z:

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 set B, which corresponds to the geometrical features extracted from the polygonal approximation $\mathrm{Pol}(z)$ of the observation z, is also extended with the following new features:

14. The circularity of $\mathrm{Pol}(z)$.
15. The normalized circularity of $\mathrm{Pol}(z)$.

16. The average normalized distance between the centroid and the shape boundary of $\text{Pol}(z)$.
17. The standard deviation of the normalized distance between the centroid and the shape boundary of $\text{Pol}(z)$.

In the experimental section we will see that these additional features improve the robustness of the resulting classifier. The complete lists of features, together with their mathematical definition can be found in Appx. A and Appx. B.

4.5 Probabilistic Relaxation Labeling

The first approach that we use in this chapter to extract topological maps determines the semantic class of each unoccupied cell of the grid. 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 as presented in Sect. 4.3. This process results in an occupancy map with a semantic label for each free cell. However, the final maps usually contain some errors in the classification. To smooth the final classification of each cell, we apply the probabilistic relaxation labeling method introduced in [16]. 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 $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ be a graph consisting of nodes $\mathrm{V} = \{v_1, \ldots, v_N\}$ and edges $\mathrm{E} \subseteq \mathrm{V} \times \mathrm{V}$. Let furthermore $\mathrm{Y} = \{y_1, \ldots, y_K\}$ 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 $p_i(k)$ of that histogram stores the probability that the node v_i has the label k. Thus, $\sum_{k=1}^{K} p_i(k) = 1$. For each node v_i, $\mathrm{Ne}(v_i) \subset \mathrm{V}$ 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 $\mathrm{R} = \{r_{ij}(k, k') \mid v_j \in \mathrm{Ne}(v_i)\}$ defines the compatibility coefficients between the label k of node v_i and the label k' of v_j. Additionally, we define $\mathrm{O} = \{o_{ij} \mid v_j \in \mathrm{Ne}(v_i)\}$ 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 $P_i^{(0)}$ for the node v_i, the probabilistic relaxation method iteratively computes estimates $P_i^{(r)}$, $r = 1, 2, \ldots$, based on the initial probabilities $p_i^{(0)}(k)$, the compatibility coefficients R, and the weights O, in the form

$$p_i^{(r+1)}(k) = \frac{p_i^{(r)}(k)\left[1 + q_i^{(r)}(k)\right]}{\sum_{k'=1}^{K} p_i^{(r)}(k')\left[1 + q_i^{(r)}(k')\right]}, \tag{4.6}$$

where

$$q_i^{(r)}(k) = \sum_{j=1}^{N} o_{ij} \left[\sum_{k'=1}^{K} r_{ij}(k,k') p_j^{(r)}(k') \right] . \tag{4.7}$$

Note that the compatibility coefficients $r_{ij}(k,k') \in [-1,1]$ do not need to be symmetric. A value $r_{ij}(k,k')$ close to -1 indicates that label k' is unlikely at node v_j when label k 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 in [26] as

$$r_{ij}(k,k') = \begin{cases} \frac{1}{1-p_i(k)} \left(1 - \frac{p_i(k)}{p_{ij}(k|k')}\right) & \text{if } p_i(k) < p_{ij}(k \mid k') \\ \frac{p_{ij}(k|k')}{p_i(k)} - 1 & \text{otherwise} , \end{cases} \tag{4.8}$$

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

Now we describe how to apply this method for the 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 the cell is occupied. We furthermore consider the 8-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. We then define a neighborhood $\mathrm{Ne}(v_{(x,y)})$ using the 8-connected cells to $v_{(x,y)}$ as described in [7].

For the initial probabilities $p_{(x,y)}^{(0)}(k)$, we use the output P of the probabilistic decision list as described in Sect. 4.3. This output is represented by a histogram in which each bin $p(k)$ indicates the probability that the pose belongs to class k. Furthermore, our set of labels Y is composed by four labels

$$\mathrm{Y} = \{\text{corridor}, \text{room}, \text{doorway}, \text{wall}\} .$$

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 (4.5). The classification output P 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 $o_{ij} \in \mathrm{O}$ is initialized with the value $\frac{1}{8}$, indicating that the eight neighbors v_j of node v_i are equally important. The compatibility coefficients are calculated using (4.8). The values $p_i(k)$ and $p_{ij}(k \mid k')$ are obtained from statistics in the given occupancy grid map corresponding to previously labeled training data.

4.6 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 in [25]. The idea behind iAMNs 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.

4.6.1 Associative Markov Networks

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 ourselves to the case of discrete variables, that is, each variable $y_i \in \mathrm{Y}$ 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 $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ where the set of nodes V represents discrete variables, and the edges E refer to the relations between them [22]. An AMN can be divided into a subset of cliques Q, where each clique $q \in \mathrm{Q}$ is associated with a subset $\mathrm{Y}_q \in \mathrm{Y}$. The nodes in a clique Y_q form a fully connected subgraph.

Each clique q is accompanied by a potential $\phi_c(y_q)$ which associates a non-negative value to the variable assignment y_q. We work with pairwise associative Markov networks [22], 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 $\mathrm{E} = \{(ij) \mid i < j\}$, 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 $x_i \in \mathbb{R}^L$, which describes the properties of the object represented by that node. Similarly, a feature vector $x_{ij} \in \mathbb{R}^{L'}$ can be assigned to each edge $(ij) \in \mathrm{E}$. 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 potentials are functions of the node feature vectors x_i, similarly the edge potentials are functions of the edge feature vectors x_{ij}. The resulting network defines the distribution

$$\log P_w(y|x) = \sum_{i=1}^{N} \sum_{k=1}^{K} (w_n^k \cdot x_i) y_i^k + \sum_{e=(ij)\in \mathrm{E}} \sum_{k,k'=1}^{K} (w_e^{k,k'} \cdot x_{ij}) y_i^k y_j^{k'} - \log Z_w(x)\,, \quad (4.9)$$

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

where N is the total number of nodes in the graph, and $Z_w(x)$ is a partition function that depends on the parameters w and features x, but not on the labels y.

The main task in an associative Markov network consists of finding the assignment $y \in \mathrm{Y}$ that maximizes $\log_w P(y|x)$. This is actually a maximum a posteriori (MAP) assignment that can be formulated as a linear program [22].

4.6.2 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 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 $\tau : \mathbb{R}^L \rightarrow \mathbb{R}^K$ given by

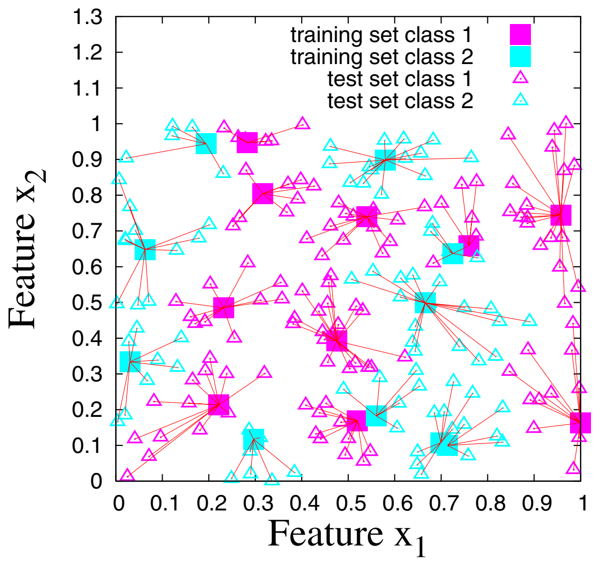
$$\tau(\tilde{x}) = (t_1 = d(\tilde{x},\hat{x}_1),\ldots,t_K = d(\tilde{x},\hat{x}_K)) \ , \tag{4.10}$$

where K is the number of classes, and $\hat{x}_k$ denotes the training example with label k closest to $\tilde{x}$. In addition, the function $d(\cdot,\cdot)$ calculates the distance in feature space. Using this transformation the resulting features are more easily separable by hyperplanes. An example is given in Fig. 4.5. 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 in the ground truth (squares). This nearest neighbor classification results in very few errors. However, it seems difficult to separate the test data into the two classes to which they pertain using a hyperplane (a line in this case). The bottom image of Fig. 4.5 shows the training examples in the transformed space using the transformation given by $\tau(\tilde{x}) = (t_1, t_2)$, with $t_1 = d(\tilde{x},\hat{x}_1)$, and $t_2 = d(\tilde{x},\hat{x}_2)$. The linear separability is improved in the transformed space.

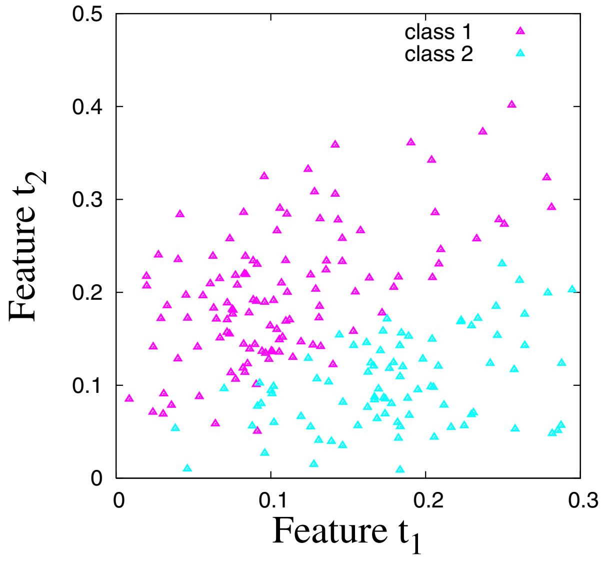
Additionally, the M nearest neighbors can be used in the transform function. For this, we compute the M nearest distances to each of the classes $k = 1,\ldots,K$. The final transformation $\tau_M : \mathbb{R}^L \rightarrow \mathbb{R}^{K \cdot M}$ is given by

$$\tau_M(\tilde{x}) = (d(\tilde{x},\hat{x}_1^1),\ldots,d(\tilde{x},\hat{x}_1^M),\ldots,d(\tilde{x},\hat{x}_K^1),\ldots,d(\tilde{x},\hat{x}_K^M)) \ . \tag{4.11}$$

The resulting model, first introduced in [25], is called instance-based associative Markov network (iAMN).



(a) original feature space



(b) original feature space

Fig. 4.5 Example of the feature transform τ for a two-class problem with two features. (**a**) Training and test data in the original feature space. The classification of the test data (*triangles*) is shown as lines connecting each training example with the closest example in the ground truth (*squares*). (**b**) The transformation τ is applied to the test data (*triangles*). The transformed examples are more easily separable.

4.6.3 Feature Selection

One of the problems when classifying points represented by range data consists of 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 the *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, since we have to learn and infer 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 are very correlated [23]. The goal is thus to reduce the size of the feature vectors when used with the iAMN and, at the same time, to 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 classifiers, since the final number of weak classifiers T can be determined, and each selected weak classifier represents a feature (cf. Sect 4.2). 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 for selecting features. 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). These kinds 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 [8].

As separability criterion S, we use the Fisher's discrimination ratio (FDR) extended to the multi-class case [23]. For a scalar feature f and K classes $\{y_1,\ldots,y_K\}$, $S(f)$ can be defined as

$$S(f) = FDR_f = \sum_{i=1}^{K} \sum_{j \neq i}^{K} \frac{(\mu_i - \mu_j)^2}{\sigma_i + \sigma_j}, \tag{4.12}$$

where the μ_i and σ_i refer respectively to the mean and variance of the class i. Additionally, the cross-correlation coefficient between any two features f and f' given N training examples is defined as

$$\rho_{ff'} = \frac{\sum_{t=1}^{N} f_n f'_n}{\sqrt{\sum_{n=1}^{N} f_n^2 \sum_{n=1}^{N} f'^2_n}}, \tag{4.13}$$

where f_n denotes the value of the feature f in the training example n. Finally, the selection of the best $L^* \subset L$ features involves the steps shown in Fig. 4.6.

After the scalar feature selection, the learning and inference steps on the instance-based associative Markov network are carried out. Further details on the inference process can be found in [25].

- Select the first feature f_1 as

$$f_1 = \underset{f}{\operatorname{argmax}}\, S(f).$$

- Select the second feature f_2 as

$$f_2 = \underset{f \neq f_1}{\operatorname{argmax}} \left\{ w_1 S(f) - w_2 |\rho_{f_1 f}| \right\},$$

  where w_1 and w_2 are weighting factors.

- Select f_{l^*}, $l^* = 3, \ldots, L^*$, such that

$$f_{l^*} = \underset{f \neq f_r}{\operatorname{argmax}} \left\{ w_1 S(f) - \frac{w_2}{l^* - 1} \sum_{r=1}^{l^*} |\rho_{f_r f}| \right\}, \quad r = 1, 2, \ldots, l^* - 1$$

Fig. 4.6 Feature selection algorithm according to [23].

4.7 Region Extraction and Topological Mapping

After applying any of the previous two approaches, we obtain an occupancy grid map in which each free cell contains a distribution over the set of its possible labels. From this map we extract complete regions that correspond to places in the environment.

We define a region on a adjacency graph G as a set of 8-connected nodes with the same class y. For each label $y \in \{\text{corridor}, \text{room}, \text{doorway}\}$, regions are extracted from the adjacency graph using an algorithm for extracting connected regions [17]. Each extracted region is assigned a different identifier. The connections between regions are extracted using a similar algorithm [7]. Finally, a new topological graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ is constructed in which each node $v_i \in \mathrm{V}$ represents a region and each edge $e_i \in \mathrm{E}$ represents a connection. Additionally, we add to each node v_i information about the properties of the region which represents: area, centroid, and major and minor axes of the ellipse approximation of the region. The major and minor axes 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 as follows:

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 it the label of one of its connected regions.
2. We mark each region previously labeled as doorway, and 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. Then we assign these marked regions the label of one of their connected regions.

The different thresholds used in the heuristics are obtained from statistics in the training set.

4.8 Experimental Results

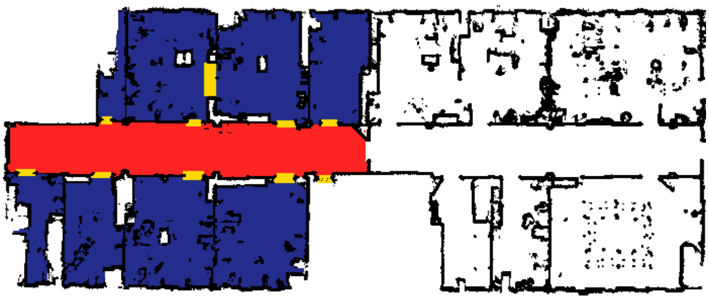
The approaches described above have been implemented and tested using occupancy maps obtained from real environments. The laser range data used for the training and classification steps were simulated using the Carnegie Mellon Robot Navigation Toolkit (CARMEN) [2, 14]. The goal of the experiments is 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. Furthermore, we analyze the improvement of the AdaBoost-based decision list classifier using the new set of features. Finally we present one experiment in which iAMNs are used to train and classify an indoor environment.

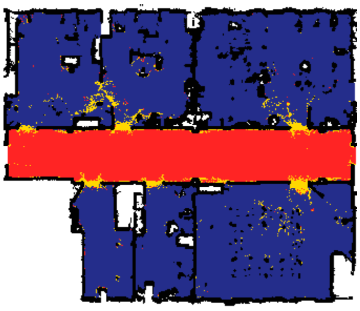
4.8.1 Results Using Relaxation Labeling

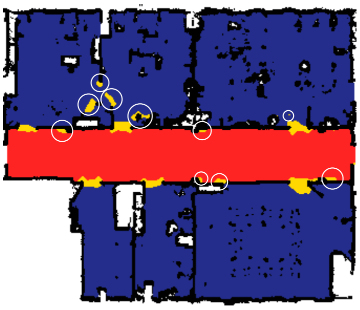
The first experiment was performed using data obtained in the office environment of building 79 at the University of Freiburg. This environment contains rooms, doorways and a corridor, which has a length of approximately 22 meters. For the sake of clarity we give the result of the obtained classification by separating the environment into two parts. The left half of the environment contains the poses used as training examples, and the right half of the environment was used for test classification and for the topological map creation as shown in Fig 4.7(a).

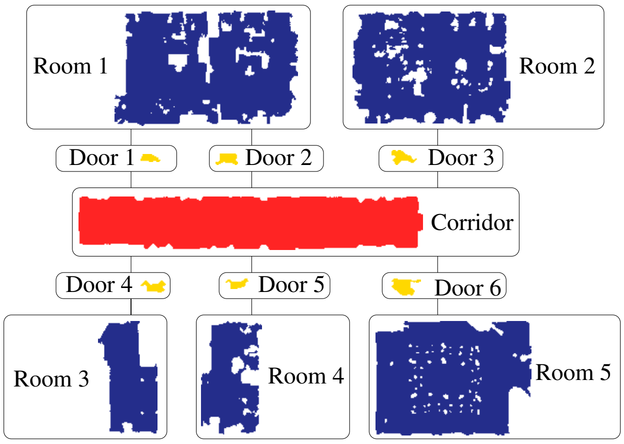
We first applied a probabilistic decision list (cf. Sec. 4.3) to classify the free cells in the occupancy grid map. The list was formed by the best combination of binary classifiers, which in this case was corridor-room. This sequence correctly classified 97.27% of the test examples. The classification is depicted as colors/gray levels in Fig. 4.7(b).

After the sequential classification, the probabilistic relaxation method explained in Sect. 4.5 was applied for 50 iterations. This method generates more compact regions and eliminates noise. The result is illustrated in the Fig. 4.7(c). Finally, the topological map is created using the connections between regions. As can be seen in Fig. 4.7(c), some regions detected as doorways (marked with circles) do not correspond to real doorways. After applying the heuristics described in Sect. 4.7 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 Fig. 4.7(d), has a final classification rate of 98.95% in the free cells.


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


(b) Sequential classification

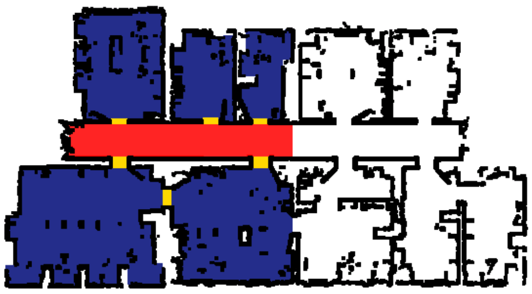

(c) Incorrect regions



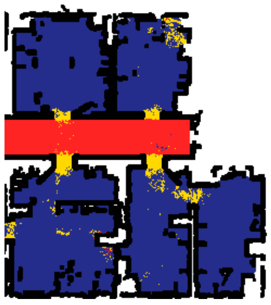
(d) Resulting topological map

🟥 Corridor 🟦 Room 🟨 Doorway

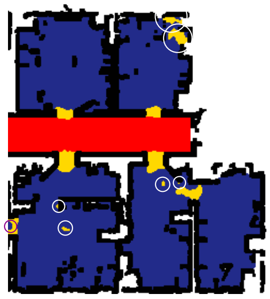
Fig. 4.7 (**a**) Training and test map of building 79 at the University of Freiburg. (**b**) Result of applying the decision list with a classification rate of 97.27%. (**c**) Result of applying relaxation and the detection of incorrect labeled regions (*white circles*). (**d**) Final topological map with the corresponding regions.



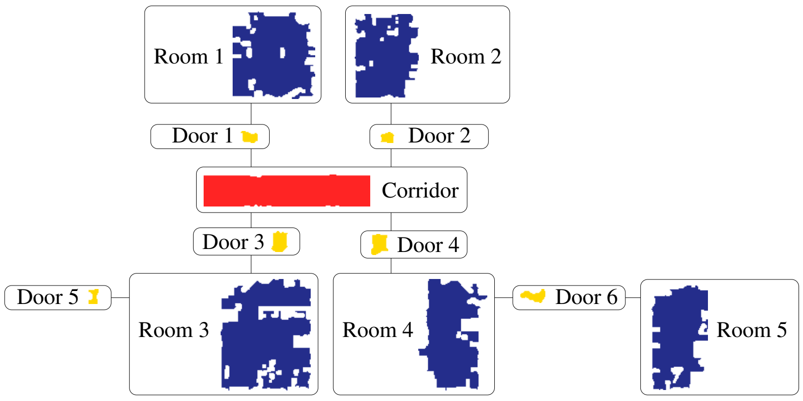
(a) Training map (left half) and test map (right half)



(b) Sequential classification



(c) Incorrect regions



(d) Resulting topological map

🟥 Corridor 🟦 Room 🟨 Doorway

Fig. 4.8 (**a**) Training and test map of the building 52 at the University of Freiburg. (**b**) Result of applying the decision list with a classification rate of 97%. (**c**) Result of applying relaxation and the detection of incorrect labeled regions (*white circles*). (**d**) Final topological map with the corresponding regions.

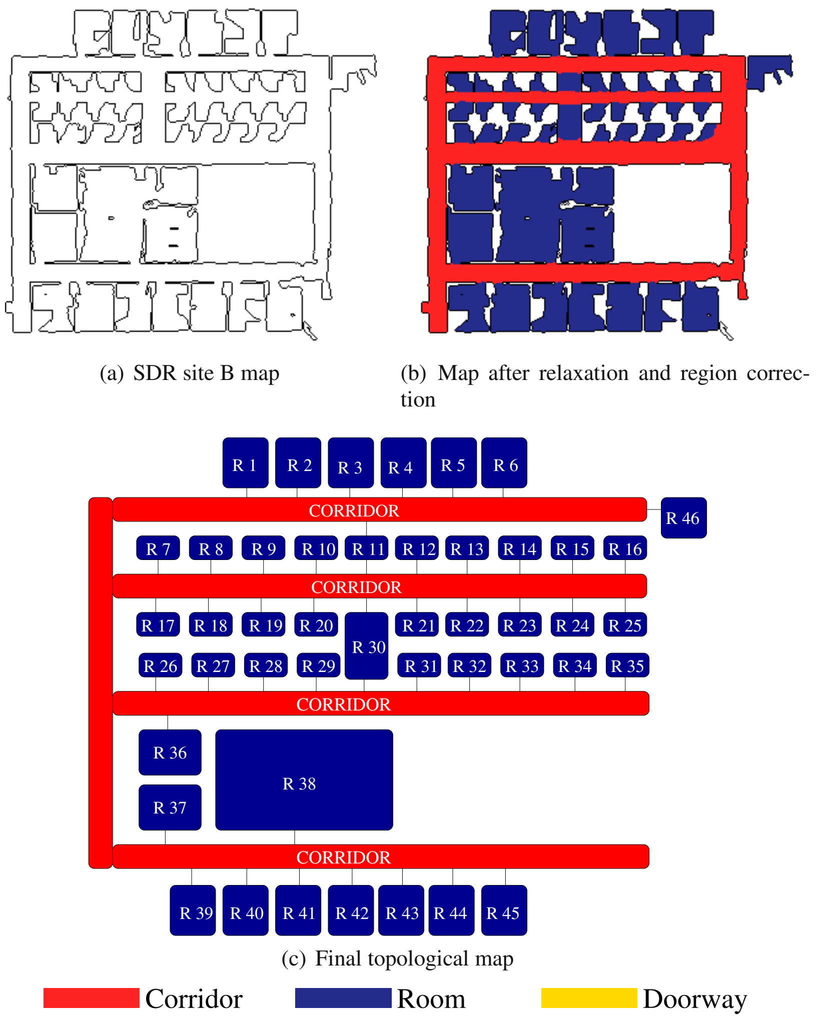
In a second experiment we created a topological map of the right part of the office environment of building 52 at the University of Freiburg. The complete occupancy grid map of this environment is shown in Fig. 4.8(a). The length of the corridor in this environment is approximately 20 meters. After applying the decision list classifier room-corridor, the classification of the test set was 97%. As in the previous experiment, we applied the relaxation process for 50 iterations as well as the heuristics for region correction of Sect. 4.7. The final result gives a classification rate of 98.66% in the free cells. The different steps of the process are illustrated as colors/gray levels in Fig. 4.8. Opposite to the previous experiment, the doorway between the two right-most rooms under the corridor is correctly detected, as can be shown in Fig. 4.8(c). Therefore, the rooms are labeled as two different regions in the final topological map as shown in Fig. 4.8(d).

4.8.2 Application to New Indoor Environments

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 Fig. 4.7(a) and Fig. 4.8(a) at 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 [10]. 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 Fig. 4.9. We use the sequence corridor-doorway which gives a first classification of 92.36%. As can be seen in Fig. 4.9(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. Doorways are very difficult to detect with 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 training maps have different sizes and resolutions, and not all features are scale invariant. In the final topological map, 96.94% of the data points are correctly classified.

4.8.3 Results with Instance-Based Associative Markov Networks

In this experiment we applied our classification approach using iAMNs to the indoor environment corresponding to building 79 at the University of Freiburg. For efficiency reasons we used a grid resolution of 20 cm, which lead us to a graph containing 8088 nodes. Smaller resolutions result in much bigger networks, which are 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 as



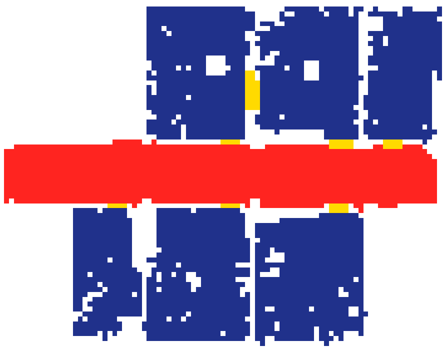
(a) SDR site B map

(b) Map after relaxation and region correction

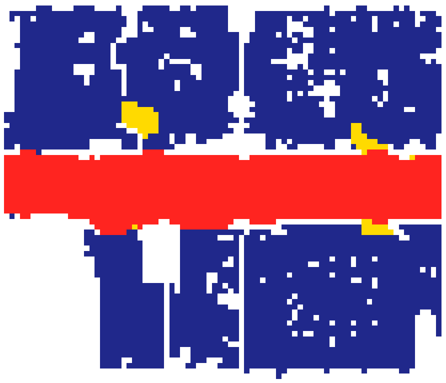
(c) Final topological map

Fig. 4.9 (**a**) Original map of the building. (**b**) Resulting classification after the relaxation an region correction. (**c**) 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 original structure.

shown in Fig. 4.10. For each cell we calculate 203 geometrical features. This number was reduced to 30 applying the feature selection of Sect. 4.6.3. Figure 4.10(b) 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 could be the low resolution of the map (20 cm) in comparison with the original



(a) Training map



(b) Classification

🟥 Corridor 🟦 Room 🟨 Doorway

Fig. 4.10 (**a**) Training map of building 79 at the University of Freiburg. (**b**) Resulting classification using an iAMN with 30 selected features.

resolution (5 cm). However, maintaining the original resolution would lead us to a huge Markov network almost impractical to use.

4.8.4 Comparison of Feature Sets

In this final experiment, we compare the new extended feature set described in Sect. 4.4 with the one proposed Sect. 3.4. For this purpose, we trained an AdaBoost-based decision list for each of the feature sets using the training set shown in Fig. 4.7(a). The different sequential classifiers were then applied to the test set depicted in Fig. 4.7(b). The obtained classification results are shown in Table 4.1. As can be seen, the new extended feature-set provides better results in all of the experiments. This result indicates that the feature set has a major influence in the final classification. However, the advantage of AdaBoost is that we can keep adding

Table 4.1 Classification results of the new improved feature set compared to the one in Section 3.4.

Decision List	Feature set of Sect. 4.4 [%]	Feature set of Sect. 3.4 [%]
corridor-room	97.27	93.16
room-corridor	97.26	93.31
room-door	96.94	93.94
corridor-door	87.73	80.10
door-corridor	87.21	80.10
door-room	86.60	80.49

features to the set without worrying about the curse of dimensionality, since the boosting process will select only the necessary features.

4.9 Related Work

Different algorithms for extracting topological maps in indoor environments have been proposed in the past. For example, Kuipers and Byun [12] extract distinctive points in the map, which are defined as the local maximum of some measure. These points are used as nodes in a topological map. In their work, Kortenkamp and Weymouth [11] fuse vision and ultrasound information to determine topologically relevant places. Additionally, Shatkey and Kaelbling [20] 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 [24], in this case using Voronoi diagrams. These critical points minimize the clearance locally, and are then used as nodes in a topological map. Also Beeson et al. [1] detect topological places with an extension of the Voronoi graph. Furthermore, Choset [3] encodes metric and topological information in a generalized Voronoi graph to solve the SLAM problem. The distinctive places extracted with the previous methods do not represent concrete spaces such as rooms or corridors although they can have some relation to them. In comparison, 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.

Mathematical morphology is used in the work by Fabrizi and Saffiotti [4]. This method uses a disc as structuring element for the dilation and erosion operations. This approach extracts 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 [21] use fingerprints extracted from images to create topological maps. In their work, Zivkovic et al. [27] 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 in which the robot is located, which makes it difficult to generalize with 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. [6] 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. The approach is similar to the one in Sect. 4.6. However, in [6], only the points lying in the Voronoi diagram are used in the MRF, whereas we used all free locations in the map.

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

4.10 Conclusion

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 to the resulting classifications to reduce classification errors. The second approach is based on iAMNs together with scalar feature selection. After applying any of the previous methods the different regions and their connections are extracted. Each region corresponds to a place in the map such a corridor, room, or doorway.

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

References

1. Beeson, P., Jong, N.K., Kuipers, B.: Towards autonomous topological place detection using the extended voronoi graph. In: Proceedings of the IEEE International Conference on Robotics and Automation (2005)
2. CARMEN. Carnegie Mellon Robot Navigation Toolkit, `http://carmen.sourceforge.net/`
3. Choset, H.: Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. IEEE Transactions on Robotics and Automation (2001)
4. Fabrizi, E., Saffiotti, A.: Extracting topology-based maps from gridmaps. In: Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, pp. 2972–2978 (2000)
5. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Annals of Statistics 28(2), 337–407 (2000)
6. Friedman, S., Pasula, H., Fox, D.: 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)

7. Gonzalez, R.C., Wintz, P.A.: Digital Image Processing. Addison-Wesley Publishing Inc., Reading (1987)
8. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)
9. Haralick, R.M., Shapiro, L.G.: Computer and Robot Vision. Addison-Wesley Publishing Inc., Reading (1992)
10. Howard, A., Roy, N.: The robotics data set repository, Radish (2003), `http://radish.sourceforge.net/`
11. Kortenkamp, D., Weymouth, T.: Topological mapping for mobile robots using a combination of sonar and vision sensing. In: Proceedings of the National Conference on Artificial Intelligence, pp. 979–984 (1994)
12. Kuipers, B., Byun, Y.T.: A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. Journal of Robotics and Autonomous Systems 8, 47–63 (1991)
13. Loncaric, S.: A survey of shape analysis techniques. Pattern Recognition 31(8), 983–1001 (1998)
14. Montemerlo, M., Roy, N., Thrun, S.: Perspectives on standardization in mobile robot programming. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2003)
15. O'Rourke, J.: Computational Geometry in C, 2nd edn. Cambridge University Press, Cambridge (1998)
16. Rosenfeld, A., Hummel, R.A., Zucker, S.W.: Scene labeling by relaxation operations. IEEE Transactions on Systems Man and Cybernetics 6(6), 420–433 (1976)
17. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. Journal of the Association for Computing Machinery 13(4), 471–494 (1966)
18. Russ, J.C.: The Image Processing Handbook. CRC Press, Boca Raton (1992)
19. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learning 37(3), 297–336 (1999)
20. Shatkey, H., Kaelbling, L.P.: Learning topological maps with weak local odometric information. In: Proceedings of the International Conference on Artificial Intelligence (1997)
21. Tapus, A., Siegwart, R.: Incremental robot mapping with fingerprints of places. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, August 2005, pp. 2429–2434 (2005)
22. Taskar, B.: Learning associative markov networks. In: Proceedings of the International Conference on Machine Learning (2004)
23. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 3rd edn. Academic Press, London (2006)
24. Thrun, S.: Learning metric-topological maps for indoor mobile robot navigation. Artificial Intelligence 99(1), 21–71 (1998)
25. Triebel, R., Schmidt, R., Mozos, O.M., Burgard, W.: 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, Hyderabad, India, pp. 2225–2230 (2007)
26. Yamamoto, H.: A method of deriving compatibility coefficients for relaxation operators. Computer Graphics and Image Processing 10, 256–271 (1979)
27. Zivkovic, Z., Bakker, B., Kröse, B.: Hierarchical map building using visual landmarks and geometric constraints. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2005)