Mobile Robot Map Learning from Range Data in Dynamic Environments

Wolfram Burgard[1] Cyrill Stachniss[1,2] Dirk Hähnel[1,3]

[1]University of Freiburg, Dept. of Computer Science, 79110 Freiburg, Germany
[2]Eidgenössische Technische Hochschule Zürich (ETH), 8092 Zürich, Switzerland
[3]Intel Research Seattle, 1100 NE 45th Street, Seattle, WA 98105, USA

Summary. The problem of generating maps with mobile robots has received considerable attention over the past years. Most of the techniques developed so far have been designed for situations in which the environment is static during the mapping process. Dynamic objects, however, can lead to serious errors in the resulting maps such as spurious objects or misalignments due to localization errors. In this chapter, we consider the problem of creating maps with mobile robots in dynamic environments. We present two approaches to deal with non-static objects. The first approach interleaves mapping and localization with a probabilistic technique to identify spurious measurements. Measurements corresponding to dynamic objects are then filtered out during the registration process. Additionally, we present an approach that learns typical configurations of dynamic areas in the environment of a mobile robot. Our approach clusters local grid maps to identify the typical configurations. This knowledge is then used to improve the localization capabilities of a mobile vehicle acting in dynamic environments. In practical experiments carried out with a mobile robot in a typical office environment, we demonstrate the advantages of our approaches.

1.1 Introduction

Learning maps with mobile robots is one of the fundamental problems in mobile robotics. In the literature, the mobile robot mapping problem is often referred to as the *simultaneous localization and mapping problem (SLAM)* [10, 13, 16, 20, 21, 31, 33]. This is because mapping includes both, estimating the position of the robot relative to the map and generating a map using the sensory input and the estimates about the robot's pose.

Whereas most of todays mapping systems are able to deal with noise in the odometry and noise in the sensor data, they assume that the environment is static during mapping. However, if a person walks through the sensor range of the robot during mapping, the resulting map will contain evidence about an object at the corresponding location. Moreover, if the robot scans the same area a second time and registers the two scans, the resulting pose estimates will be less accurate if the person has moved in between. Thus, dynamic objects can lead to spurious objects in the resulting maps and at the same time can make localization harder.

Throughout this chapter, we consider the problem of learning grid maps with mobile robots in dynamic environments. In principle, there are different ways to dealing with the dynamic aspects in an environment. The most naïve way of dealing with dynamic objects is to apply the standard map updating equations. Typical techniques in this context are the occupancy grid algorithm [25] or the counting model used throughout this chapter. Under both algorithms, areas assumed as occupied will certainly be regarded as free after the robot has seen them unoccupied for a long enough period of time (and vice versa). This is due to the fact that the map updating operations are additive as can be seen, for example, from the log-odds representation of occupancy grid maps [25] or directly from the counting model. Accordingly, the robot needs to see an area as free more often as it has seen it occupied to make it belief that the corresponding area is free. An alternative way is to identify whether or not a beam is reflected by a dynamic object. One popular way to achieve this is to use features corresponding to dynamic objects and to track such objects while the robot moves through its environment [17, 34]. Before the robot updates its map, it can then simply filter all measurements that correspond to dynamic objects. Whereas such approaches have been demonstrated to be quite robust, their disadvantage lies in the fact that the features need to be known a priori. Throughout this chapter, we will describe an alternative technique that applies the Expectation-Maximization (EM) algorithm. In the expectation step, we compute a probabilistic estimate about which measurements might correspond to static objects. In the maximization step, we use these estimates to determine the position of the robot and the map. This process is iterated until no further improvement can be achieved.

Whereas techniques for filtering dynamic aspects have been proven to be quite robust, their major disadvantage lies in the fact that the resulting maps only contain the static aspects of the environment. Throughout this chapter, we therefore will also describe an approach to explicitely model certain dynamic aspects of environments, namely the so-called low-dynamic or quasi-static states. Our approach is motivated by the fact, that many dynamic objects appear only in a limited number of possible configurations. As an example, consider the doors in an office environment, which are typically either open or closed. In such a situation, techniques to filter out dynamic objects produce maps which do not contain a single door. This can be problematic since in many corridor environments doors are important features for localization. The knowledge about the different possible configurations can explicitly improve the localization capabilities of a mobile robot. Therefore, it is important to integrate such information into the map of the environment.

The contribution of this chapter is novel approach to generating grid maps in dynamic environments from range data. Our algorithm first estimates for each individual beam whether or not it has been reflected by a dynamic object. It then uses this information during the range registration process to estimate get better pose estimates. It also learns the quasi-static states of areas by identifying sub-maps which have typical configurations. This is achieved by clustering local grid maps. We present experiments illustrating the accuracy of the resulting maps and also an extended Monte-Carlo localization algorithm, which uses the clusters of the local maps to more accurately localize the robot.

1.2 EM-based Filtering of Beams Reflected by Dynamic Objects

As described above, one of the key problems in the context of mapping in dynamic environments is to determine whether or not a measurement is reflected by a dynamic object. Our approach to discover such measurements is strictly statistical. We use the popular EM-algorithm to identify data items that cannot be explained by the rest of the data set. The input to our routine is a sequence of data items $z = \{z_1, \ldots, z_T\}$. The output is a model m obtained from these data items after incorporating the estimates about spurious measurements. In essence, our approach seeks to identify a model m that maximizes the likelihood of the data. Throughout this chapter, we assume that each measurement z_t consists of multiple data $z_{t,1}, \ldots, z_{t,N}$ as it is the case, for example, for laser-range scans. Throughout this chapter, we assume that the data $z_{t,n}$ are beams obtained with a laser-range scanner.

To accurately map a dynamic environment, we need to know which measurements are caused by dynamic objects and therefore can safely be ignored in the alignment and map updating phase. To characterize spurious measurements in the data, we introduce additional variables $c_{t,n}$ that tell us for each t and each n whether the data item $z_{t,n}$ is caused by a static object or not. Each such variable $c_{t,n}$ is a binary variable that is either 0 or 1. It is 1 if and only if the $z_{t,n}$ is caused by a static object. The vector of all these variables will be denoted by c.

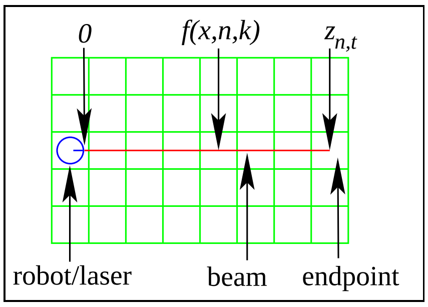

Fig. 1.1. Beam covering $z_{t,n}$ cells of a map.

For the sake of simplicity, we give the derivation for beams that are parallel to the x-axis of the map. In this case, the length $z_{t,n}$ directly corresponds to the number of cells covered by this beam. We will later describe how to deal with beams that are not parallel to the x-axis. Let f be a function that returns for each position x_t of the robot, each beam number n, and each $k \leq z_{t,n}$ the index $f(x_t, n, k)$ of k-th field covered by that beam in the map (see Figure 1.1). To determine whether or not a beam is reflected by a dynamic object, we need to define the likelihood of a measurement given the current map m of the environment, the pose x of the robot, and the information about whether $z_{t,n}$ is reflected by a maximum range reading. Typically, maximum-range readings have to be treated differently, since those measurements generally are not reflected by any object. Throughout this chapter, we introduce indicator variables $\zeta_{t,n}$ which are 1 if and only if $z_{t,n}$ is a maximum range reading and 0, otherwise. The likelihood of a measurement $z_{t,n}$ given the value of $c_{t,n}$ and the map m can thus be computed as

$$p(z_{t,n} \mid c_{t,n}, x_t, m) = \left[\prod_{k=0}^{z_{t,n}-1} (1 - m_{f(x_t,n,k)}))\right]^{\zeta_{t,n}} \cdot \left[[m_{f(x_t,n,z_{t,n})}]^{c_{t,n}} \cdot [1 - m_{f(x_t,n,z_{t,n})}]^{(1-c_{t,n})} \cdot \prod_{k=0}^{z_{t,n}-1} (1 - m_{f(x_t,n,k)}) \right]^{(1-\zeta_{t,n})} \tag{1.1}$$

The first term of this equation specifies the likelihood of the beam given it is a maximum range scan. In such a situation, we compute the likelihood as the product of the probabilities that the beam has covered the cells 0 to $z_{t,n-1}$. Please note that the cell in which the beam ends does not provide any information since we do not know, whether there is an object or not given the beam is a maximum range reading. Thereby, the probability that a beam covers a cell $k < z_{t,n}$ is equal to $1 - m_{f(x_t,n,k)}$. The second row of this equation specifies how to deal with the case that a cell that reflects a non-maximum range beam. If $z_{t,n}$ is not reflected by a dynamic object, i.e. $c_{t,n} = 1$, then the likelihood equals $m_{f(x_t,n,z_{t,n})}$. If, in contrast, $z_{t,n}$ is reflected by a dynamic object, the likelihood is $1 - m_{f(x_t,n,z_{t,n})}$. As well as for the maximum range measurements, we have to consider in both cases that the beam has covered $z_{t,n} - 1$ cells before reaching cell $f(x_t, n, z_{t,n})$.

Based on the definition of the observation likelihood, we now will define the likelihood $p(z, c \mid x, m)$ of the data which we try to maximize in order to find the most likely map of the environment.

$$p(z, c \mid x, m) = \prod_{t=1}^{T} p(z_t, c_t \mid x_t, m) \tag{1.2}$$

$$= \prod_{t=1}^{T} p(z_t, \mid x_t, m) \cdot p(c_t \mid x_t, m) \tag{1.3}$$

$$= \prod_{t=1}^{T} p(z_t, \mid x_t, m) \cdot p(c_t) \tag{1.4}$$

$$= \prod_{t=1}^{T} \prod_{n=1}^{N} p(z_{t,n}, \mid c_{t,n}, x_t, m) \cdot p(c_t) \tag{1.5}$$

We obtain Equation (1.3) from Equation (1.2) by assuming that the z_t and c_t are independent given x_t and m. We furthermore consider c_t as independent from the location x_t and the map m, which leads to Equation (1.4). Finally, Equation (1.5) is derived from Equation (1.4) under the usual assumption, that the neighboring beams of a single scan are independent given the map of the environment.

Maximizing $p(z, c \mid x, m)$ is equivalent to maximizing the corresponding log likelihood, which can be derived from Equation (1.5) and Equation (1.1) by straightforward mathematical transformations.

$$\begin{aligned}
&\ln p(z, c \mid x, m) \\
&= \ln \prod_{t=1}^{T} \prod_{n=1}^{N} p(z_{t,n}, \mid c_{t,n}, x_t, m) \cdot p(c_t) \\
&= N \cdot \sum_{t=1}^{T} \ln p(c_t) + \sum_{t=1}^{T} \sum_{n=1}^{N} \ln p(z_{t,n}, \mid c_{t,n}, x_t, m) \\
&= N \cdot \sum_{t=1}^{T} \ln p(c_t) + \sum_{t=1}^{T} \sum_{n=1}^{N} \Bigg[(1 - \zeta_{t,n}) \cdot \Big[c_{t,n} \cdot \ln m_{f(x_t,n,z_{t,n})} \\
&\quad +(1 - c_{t,n}) \cdot \ln(1 - m_{f(x_t,n,z_{t,n})}) \Big] + \sum_{k=0}^{z_{t,n}-1} \ln(1 - m_{f(x_t,n,k)}) \Bigg] \qquad (1.6)
\end{aligned}$$

Since the correspondence variables c are not observable in the first place, a common approach is to integrate over them, that is, to optimize the expected log likelihood $E_c[\ln p(c, z \mid x, m) \mid x, m, d]$ instead. Since the expectation is a linear operator, we can move it inside the expression. By exploiting the fact that the expectation of $c_{t,n}$ only depends on the corresponding measurement $z_{t,n}$ and the position x_t of the robot at that time. we can derive the following equation:

$$\begin{aligned}
&E_c[\ln p(z, c \mid x, m) \mid z, x, m] = \\
&\gamma + \sum_{t=1}^{T} \sum_{n=1}^{N} \Bigg[e_{t,n} \cdot (1 - \zeta_{t,n}) \cdot \ln m_{f(x_t,n,z_{t,n})} \\
&+(1 - e_{t,n}) \cdot (1 - \zeta_{t,n}) \cdot \ln(1 - m_{f(x_t,n,z_{t,n})}) \\
&+ \sum_{k=0}^{z_{t,n}-1} \ln(1 - m_{f(x,n,k)}) \Bigg] \qquad (1.7)
\end{aligned}$$

For the sake of brevity, we use the term

$$e_{t,n} = E_c[c_{t,n} \mid z_{t,n}, x_t, m] \qquad (1.8)$$

in this equation. The term

$$\gamma = N \cdot \sum_{t=1}^{T} E_c[\ln p(c_t) \mid z, x, m] \qquad (1.9)$$

is computed from the prior $p(c_t)$ of the measurements which is independent of z, x, and m. Accordingly, γ can be regarded as a constant.

Unfortunately, optimizing Equation (1.7) is not an easy endeavor. A typical approach to maximize log likelihoods is the EM algorithm. In the particular problem considered here, this amounts to generating a sequence of maps m of increasing likelihood. In the E-Step, we compute the expectations about the hidden variables c. In the M-step, we then compute the most likely map m using the expectations computed in the E-Step. Both steps are described in detail in the remainder of this section.

In the E-step, we compute the expectations $e_{t,n} = E_c[c_{t,n} \mid z_{t,n}, x_t, m]$ for each $c_{t,n}$ given the measurement $z_{t,n}$, the location x_t of the robot and the current map m. Exploiting the fact that $e_{t,n}$ equals $p(c_{t,n} \mid z_{t,n}, x_t, m)$ and considering the two cases that $z_{t,n}$ is a maximum range reading or not, we obtain:

$$e_{t,n} = \begin{cases} p(c_{t,n}) & \text{, if } \zeta_{t,n} = 1 \\ p(c_{t,n})\epsilon_{t,n} & \text{, otherwise} \end{cases}$$

where

$$\epsilon_{t,n} = \frac{1}{p(c_{t,n}) + (1 - p(c_{t,n}))(\frac{1}{m_{f(x_t,n,z_{t,n})}} - 1)} \tag{1.10}$$

The first equation corresponds to the situation that $z_{t,n}$ is a maximum range reading. Then, $e_{t,n}$ corresponds to the prior probability $p(c_{t,n})$ that a measurement is reflected by a static object. Thus, a maximum range reading does not provide any evidence about whether or not the cell in the map in which the beam ends is covered by a dynamic object.

In the M-Step, we want to determine the values for m and x that maximize Equation (1.7) after computing the expectations $e_{t,n}$ about the hidden variables $c_{t,n}$ in the E-step. Unfortunately, maximizing this equation is also not trivial since it involves a solution to a high-dimensional state estimation problem. To deal with the enormous complexity of the problem, many researchers phrase it as an incremental maximum likelihood process [33, 16]. The key idea of incremental approaches is to calculate the desired sequence of poses and the corresponding maps by maximizing the marginal likelihood of the t-th pose and map relative to the $(t-1)$-th pose and map. In our algorithm, we additionally consider the estimations $e_{t,n}$ that measurement n at time t is caused by a static object of the environment:

$$\hat{x}_t = \underset{x_t}{\operatorname{argmax}} \left\{ p(z_t \mid c_t, x_t, \hat{m}^{[t-1]}) \cdot p(x_t \mid u_{t-1}, \hat{x}_{t-1}) \right\} \tag{1.11}$$

In this equation, the term $p(z_t \mid c_t, x_t, \hat{m}^{[t-1]})$ is the likelihood of the measurement z_t given the pose $\hat{x}_t$ and the map $\hat{m}^{[t-1]}$ constructed so far. The term $p(x_t \mid u_{t-1}, \hat{x}_{t-1})$ represents the probability that the robot is at location x_t given the robot previously was at position $\hat{x}_{t-1}$ and has carried out (or measured) the motion u_{t-1}. The registration procedure is then carried out using the same algorithm as described in our previous work [17].

It remains to describe how the measurement z_t is then used to generate a new map $\hat{m}^{[t]}$ given the resulting pose $\hat{x}_t$ and the expectations $e_{t,n}$. Fortunately, once

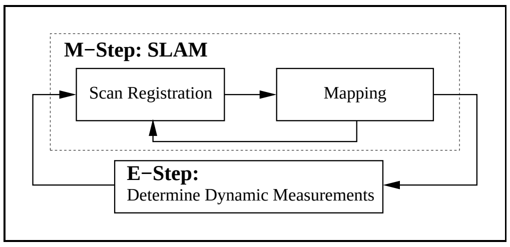

Fig. 1.2. Iteration of SLAM and dynamic beam estimation.

$x_1, \ldots, x_t$, have been computed, we can derive a closed-form solution for $m^{[t]}$. We want to determine the value of each field j of the map $m^{[t]}$ such that the overall likelihood of $m^{[t]}$ is maximized. To achieve this, we sum over individual fields $j \in [1, \ldots, J]$ of the map. Thereby, we use an indicator function $I(y)$ which is 1, if y is true and 0, otherwise.

$$\begin{aligned}\hat{m}^{[t]} = \underset{m}{\operatorname{argmax}} \Bigg(\sum_{j=1}^{J} \sum_{t=1}^{T} \sum_{n=1}^{N} \Bigg[& I(f(x_t, n, z_{t,n}) = j) \\ & \cdot (1 - \zeta_{t,n}) \cdot (e_{t,n} \ln m_j + (1 - e_{t,n}) \ln(1 - m_j)) \\ & + \sum_{k=0}^{z_{t,n}-1} I(f(x_t, n, k) = j) \cdot \ln(1 - m_j) \Bigg] \Bigg) \end{aligned} \tag{1.12}$$

Now suppose, we define

$$\tilde{I}(x, n, k, j) := I(f(x, n, k) = j)$$

and

$$\begin{aligned} \alpha_j &:= \sum_{t=1}^{T} \sum_{n=1}^{N} \tilde{I}(x_t, n, z_{t,n}, j) \cdot (1 - \zeta_{t,n}) \cdot e_{t,n} \\ \beta_j &:= \sum_{t=1}^{T} \sum_{n=1}^{N} \Bigg(\tilde{I}(x_t, n, z_{t,n}, j) \cdot (1 - \zeta_{t,n}) \\ & \qquad \cdot (1 - e_{t,n}) + \sum_{k=0}^{z_{t,n}-1} I(f(x_t, n, k) = j) \Bigg) \end{aligned}$$

The quantity α_j corresponds to the sum of the expectations $e_{t,n}$ that beam n of scan t is reflected by a static object of all beams that are not maximum-range beams and that end in cell j. The term β_j, on the other hand, is the sum of two terms. The first term is the sum of the expectations $1 - e_{t,n}$ that beam n of scan t is reflected by a dynamic object of all beams that are not maximum-range beams and that end in cell j. The second value of the sum simply is the number of times a beam covers j but does not end in j. Please note that this value is independent from whether or not the corresponding beam is reflected by a dynamic object or not. Please furthermore note that in a static world with $e_{t,n} = 1$ for all t and n the term α_t corresponds to the number of times a beam that does not have the maximum length ends in j. In contrast to that, β_j is the number of times a beam covers a cell.

Using the definitions of α_j and β_j, Equation (1.12) turns into

$$m^{[t]} = \underset{m}{\operatorname{argmax}} \left(\sum_{j=1}^{J} \alpha_j \ln m_j + \beta_j \ln(1 - m_j) \right) \tag{1.13}$$

Since all m_j are independent, we maximize the overall sum by maximizing each m_j. A necessary condition to ensure that m_j is a maximum is that the first derivative equals zero:

$$\frac{\partial m}{\partial m_j} = \frac{\alpha_j}{m_j} - \frac{\beta_j}{1 - m_j} = 0 \tag{1.14}$$

By straightforward mathematical transformations we obtain

$$m_j = \frac{\alpha_j}{\alpha_j + \beta_j}. \tag{1.15}$$

Note that given the sensor model specified in Equation (1.1), this closed-form solution for the most likely map m for given positions x and static environments corresponds to the naïve counting technique. In this approach, one determines the probability that a map cell reflects a beam by counting how often a beam has ended in that cell and how often a beam has covered it without ending in it. This differs from the occupancy mapping approach in which one seeks to determine whether or not a particular area in the environment is occupied or not. To understand the difference between the two approaches, consider an object that reflects a beam in 70% of all cases. Whereas the counting model yields a value of 0.7 for this cell, the value of the same cell will typically converge to 1 in the context of occupancy grids.

The overall approach can be summarized as follows (see also Figure 1.2). We start with an initial map $\hat{m}$ obtained by the incremental mapping approach. Thereby, the expectations $e_{t,n}$ are initialized with the prior probability $p(c_{t,n})$ that a measurement is caused by a static object. Given the resulting map $\hat{m}$ and the corresponding positions $\hat{x}$, we compute new expectations $e_{t,n}$ for each beam according to Equation (1.8). These expectations are then used to compute a new map. The overall process is iterated until no improvement of the overall likelihood (Equation (1.6)) can be achieved or a certain number of iterations has been exceeded.

At the end of this section, we would like to discuss how to deal with beams that are not parallel to the x-axis. In this case, we no longer can compute the likelihood that a beam covers a cell j of m as $(1 - m_j)$. Otherwise, transversal beams covering more cells would accumulate a lower likelihood. The solution to this is to weigh the beams according to the length by which they cover a cell. Suppose B is the set of cells in m covered by a beam. Furthermore, suppose l_j is the length by which the beam covers field $j \in B$. Then, the likelihood of covering all cells in B is computed as $\prod_{j \in B} (1 - m_j)^{l_j}$.

This concludes the description of our algorithm for filtering measurements reflected from dynamic objects. As we will see in the experiments, this approach drastically improves the accuracy of the pose estimates and the quality of the resulting map.

1.3 Learning Maps of Quasi-Static Environments

In certain situations, it can be advantageous to explicitely model dynamic aspects rather than simply filtering them out. As a motivating example consider the individual local maps depicted in Figure 1.3. These maps correspond to typical configurations of the same place and have been learned by a mobile robot operating in an office environment. They show a part of a corridor including two doors and their typical states. The approach described in this section learns such local configurations and to uses this information to improve the localization accuracy of a mobile robot.

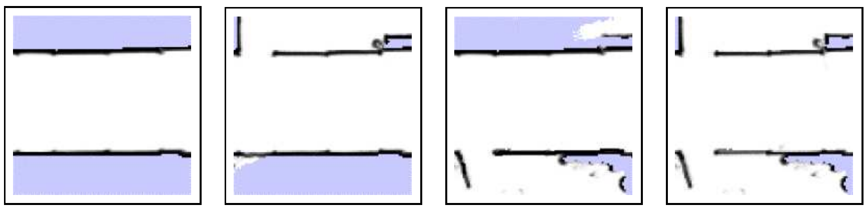

Fig. 1.3. Possible states of a local area. The different configurations correspond to open and closed doors.

The key idea of our approach is to use the information about changes in the environment during data acquisition to estimate possible spatial configurations and store them in the map model. To achieve this, we construct a sub-map for each area in which dynamic aspects have been observed. We then learn clusters of sub-maps that represent possible environmental states in the corresponding areas.

1.3.1 Map Segmentation

In general, the problem of learning maps in dynamic environments is a high-dimensional state estimation problem. A naïve approach could be to store an individual map of the whole environment for each potential state. Obviously, using this approach, one would have to store a number of maps that is exponential in the number of dynamic objects. In real world situations, the states of the objects in one room are typically independent of the states of the objects in another room. Therefore, it is reasonable to marginalize the local configurations of the individual objects.

Our algorithm segments the environment into local areas, called sub-maps. In this chapter, we use rectangular areas which inclose locally detected dynamic aspects to segment the environment into sub-maps. For each sub-map, the dynamic aspects are then modeled independently.

Note that in general the size of these local maps can vary from the size of the overall environment to the size of each grid cell. In the first case, we would have to deal with the exponential complexity mentioned above. In the second case, one heavily relies on the assumption that neighboring cells are independent, which is not justified in the context of dynamic objects. In our current system, we first identify positions in which the robot perceives contradictory observations which are typically caused by dynamic elements. Based on a region growing technique, areas which inclose dynamic aspects are determined. By taking into account visibility constraints between regions, they are merged until they do not exceed a maximum sub-map size (currently set to $20\,m^2$). This limits the number of dynamic objects per local map and in this way leads to a tractable complexity. An example for three sub-maps constructed in such a way is depicted in Figure 1.11. Note that each sub-map has an individual size and different sub-maps can (slightly) overlap.

1.3.2 Learning Environmental Configurations

To enable a robot to learn different states of the environment, we assume that the robot observes the same areas at different points in time. We cluster the local maps built from the different observations in order to extract possible configurations of the environment. To achieve this, we first segment the sensor data perceived by the robot into observation sequences. Whenever the robot leaves a sub-map, the current sequence ends and accordingly a new observation sequence starts as soon as the robot enters a new sub-map. Additionally, we start a new sequence whenever the robot moves through the same area for more than a certain amount of time ($30\,s$). This results in a set Φ of observation sequences for each sub-map

$$\Phi = \{\phi_1, \ldots, \phi_n\}, \tag{1.16}$$

where each

$$\phi_i = z_{start(i)}, \ldots, z_{end(i)}. \tag{1.17}$$

Here z_t describes an observation obtained at time t. For each sequence ϕ_i of observations, we build an individual grid map for the corresponding local area. Thereby, we use the algorithm proposed in Section 1.2. Note that this approach eliminates highly dynamic aspects such as people walking by. Quasi-static aspects like doors, typically do not change their state frequently, so that the robot can observe them as static for the short time window. The different states are usually observed when the robot returns to a location at a later point in time.

Each grid computed for a local region is then transformed into a vector of probability values ranging from 0 to 1 and one additional value ξ to represent an unknown (unobserved) cell. All vectors which correspond to the same local area are clustered using the fuzzy k-means algorithm [14]. During clustering, we treat unknown cells in an slightly different way, since we do not want to get an extra cluster in case the sensor did not covered all parts of the local area. In our experiment, we obtained the best behavior using the following distance function for two vectors a and b during clustering

$$d(a,b) = \sum_i \begin{cases} (a_i - b_i) & a_i \neq \xi \wedge b_i \neq \xi \\ 0 & a_i = \xi \wedge b_i = \xi \\ \epsilon & \text{otherwise,} \end{cases} \tag{1.18}$$

where ϵ is a constant close to zero.

When comparing two values representing unknown cells, one in general should use the average distance computed over all known cells to estimate this quantity. In our experiments, we experienced that this leads to additional clusters in case a big part of a sub-map contains unknown cells even if the known areas of the maps were nearly identical. Therefore, we use the distance function given in Equation (1.18) which sets this distance value to zero.

Unfortunately, the number of different environmental states is not known in advance. Therefore, we iterate over the number of clusters and compute in each step a model using the fuzzy k-means algorithm. In each iteration, we create a new cluster initialized using the input vector which has the lowest likelihood under the current model. We evaluate each model θ using the Bayesian Information Criterion (BIC) [30]:

$$BIC = \log P(d \mid \theta) - \frac{|\theta|}{2} \log n \tag{1.19}$$

The BIC is a popular approach to score a model during clustering. It trades off the number $|\theta|$ of clusters in the model θ multiplied by the logarithm of the number of input vectors n and the quality of the model with respect to the given data d. The model with the highest BIC is chosen as the set of possible configurations, in the following also called patches, for that sub-map. This process is repeated for all sub-maps.

Note that our approach is an extension of the classical occupancy grid map [25] or counting model, in which the environment is not supposed to be static anymore. In situations without moving objects, the overall map reduces to a standard grid map.

The complexity of our mapping approach depends linearly on the number T of observations multiplied by the number s of sub-maps. Furthermore, the region growing applied to build up local maps introduces in the worst case a complexity of $p^2 \log p$, where p is the number of grid cells considered as dynamic. This leads to an overall complexity of $O(T \cdot s + p^2 \log p)$. Using a standard PC, our current implementation requires around 10% of the time needed to record the log file.

1.4 Monte-Carlo Localization Using Patch-Maps

It remains to describe how our patch-map representation can be used to estimate the pose of a mobile robot moving through its environment. Throughout this chapter, we apply an extension of Monte-Carlo localization (MCL), which has originally been developed for mobile robot localization in static environment [12]. MCL uses a set of weighted particles to represent possible poses of the robot. Typically, the state vector consists of the robot's position as well as its orientation. The sensor readings are used to compute the weight of each particle by estimating the likelihood of the observation given the pose of the particle and the map.

Besides the pose of the robot, we want to estimate the configuration of the environment in our approach. Since we do not use a static map like in standard MCL, we need to estimate the map $m^{[t]}$ as well as the pose x_t of the robot at time t

$$\begin{aligned} & p(x_t, m^{[t]} \mid z_{1:t}, u_{0:t-1}) = \\ & \quad \eta \cdot p(z_t \mid x_t, m^{[t]}, z_{1:t-1}, u_{0:t-1}) \cdot p(x_t, m^{[t]} \mid z_{1:t-1}, u_{0:t-1}). \end{aligned} \tag{1.20}$$

Here η is a normalization constant and u_{t-1} refers to the motion command which guides the robot from x_{t-1} to x_t. The main difference to approaches on simultaneous localization and mapping (SLAM) is that we do not reason about all possible map configurations like SLAM approaches do. Our patch-map restricts the possible states according to the clustering of patches and therefore only a small number of configurations are possible.

Under the Markov assumption, the second line of Equation (1.20) can be transformed to

$$\begin{aligned} & p(x_t, m^{[t]} \mid z_{1:t-1}, u_{0:t-1}) \\ & = \int_{x_{t-1}} \int_{m^{[t-1]}} p(x_t, m^{[t]} \mid x_{t-1}, m^{[t-1]}, z_{1:t-1}, u_{t-1}) \\ & \quad \cdot p(x_{t-1}, m^{[t-1]} \mid z_{1:t-1}, u_{0:t-2})\, dx_{t-1}\, dm^{[t-1]} \qquad (1.21) \\ & = \int_{x_{t-1}} \int_{m^{[t-1]}} p(x_t \mid x_{t-1}, m^{[t-1]}, z_{1:t-1}, u_{t-1}) \\ & \quad \cdot p(m^{[t]} \mid x_t, x_{t-1}, m^{[t-1]}, z_{1:t-1}, u_{t-1}) \\ & \quad \cdot p(x_{t-1}, m^{[t-1]} \mid z_{1:t-1}, u_{0:t-2})\, dx_{t-1}\, dm^{[t-1]} \qquad (1.22) \\ & = \int_{x_{t-1}} \int_{m^{[t-1]}} p(x_t \mid x_{t-1}, u_{t-1}) p(m^{[t]} \mid x_t, m^{[t-1]}) \\ & \quad \cdot p(x_{t-1}, m^{[t-1]} \mid z_{1:t-1}, u_{0:t-2})\, dx_{t-1}\, dm^{[t-1]}. \qquad (1.23) \end{aligned}$$

Equation (1.23) is obtained from Equation (1.22) by assuming that $m^{[t]}$ is independent from $x_{t-1}, z_{1:t-1}, u_{t-1}$ given we know x_t and $m^{[t-1]}$ as well as assuming that x_t is independent from $m^{[t-1]}, z_{1:t-1}$ given we know x_{t-1} and u_{t-1}. Combining Equation (1.20) and Equation (1.23) leads to

$$\begin{aligned} & p(x_t, m^{[t]} \mid z_{1:t}, u_{0:t-1}) \\ & = \eta \cdot p(z_t \mid x_t, m^{[t]}, z_{1:t-1}, u_{0:t-1}) \\ & \quad \int_{x_{t-1}} \int_{m^{[t-1]}} p(x_t \mid x_{t-1}, u_{t-1}) p(m^{[t]} \mid x_t, m^{[t-1]}) \\ & \quad \cdot p(x_{t-1}, m^{[t-1]} \mid z_{1:t-1}, u_{0:t-2})\, dx_{t-1}\, dm^{[t-1]}. \end{aligned} \tag{1.24}$$

Equation (1.24) describes how to extend the standard MCL approach so that it can deal with different environmental configurations. Besides the motion model $p(x_t \mid x_{t-1}, u_{t-1})$ of the robot, we need to specify a map transition model $p(m^{[t]} \mid x_t, m^{[t-1]})$, which describes the change in the environment over time.

In our current implementation, we do not reason about the state of the whole map, since each sub-map would introduce a new dimension in the state vector of each particle, which leads to a state estimation problem, that is exponential in the number of local sub-maps. Furthermore, the observations obtained with a mobile robot provide information only about the local environment of the robot. Therefore, we only estimate the state of the current patch the robot is in, which leads to one additional dimension in the state vector of the particles compared to standard MCL.

In principle, the map transition model $p(m^{[t]} \mid x_t, m^{[t-1]})$ can be learned while the robot moves through the environment. In our current system, we use a fixed density for all patches. We assume, that with probability α the current state of the environment does not change between time $t-1$ and t. Accordingly, the state changes to another configuration with probability $1-\alpha$. Whenever a particle stays in the same sub-map between $t-1$ and t, we draw a new local map configuration for that sample with probability $1-\alpha$. If a particle moves to a new sub-map, we draw the new map state from a uniform distribution over the possible patches in that sub-map. To improve the map transition model during localization, one in principle can update the values for α for each patch according to the observations of the robot. However, adapting these densities can also be problematic in case of a diverged filter or a multi-modal distribution about the pose of the robot. Therefore, we currently do not adapt the values of α while the robot acts in the environment.

Note that our representation bears resemblance with approaches using Rao-Blackwellized particle filters to solve the simultaneous localization and mapping problem [26, 23], as it separates the estimate about the pose of the robot from the estimate about the map. It computes the localization of the vehicle and uses this knowledge to identify the current state of the (local) map. The difference is that we aim to estimate the current state of the sub-map based on the possible configurations represented in our enhanced environmental model.

1.5 Experiments

The approaches described above has been implemented and tested on different robotic platforms, in different environments, and with 2d and 3d data acquired with SICK laser range finders. In all experiments, we figured out, that our approach can robustly filter out high-dynamic aspects. We present results demonstrating that the obtained maps contain fewer registration errors and less spurious objects. Additional experiments indicate that our approach can reliably model the quasi-static aspects of environments.

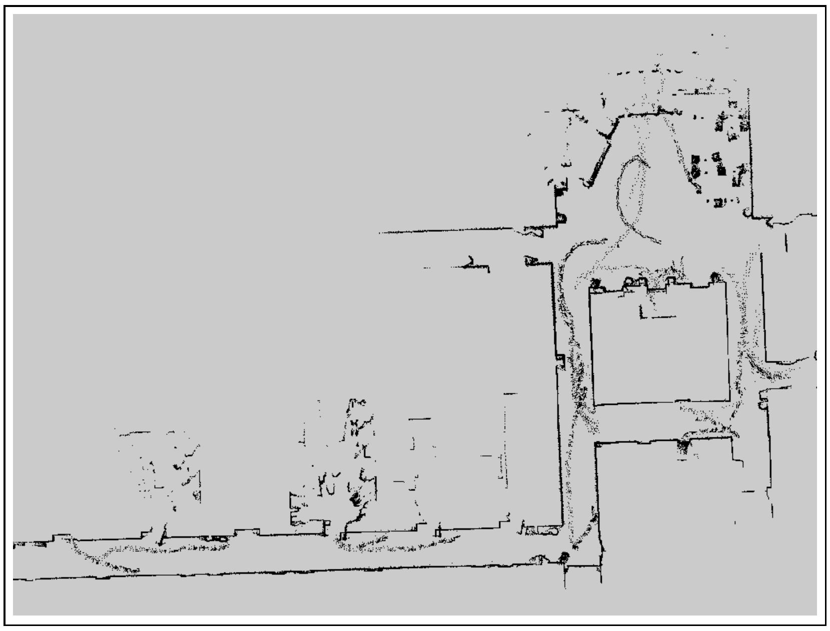
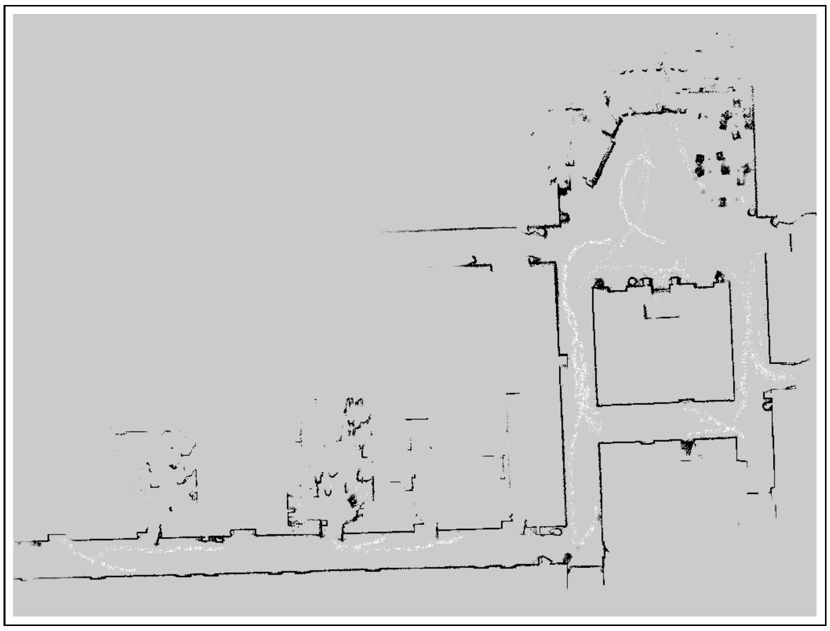

Fig. 1.4. Maps of Wean Hall at Carnegie Mellon University obtained without (top image) and with filtering measurements corrupted by dynamic objects (bottom image). The beams identified as reflected by dynamic objects are indicated by white dots.

1.5.1 Filtering Dynamic aspects

In the first set of experiments, we illustrate that our filtering algorithm can be used to reliably eliminate spurious measurements from range scans and at the same time reduces the registration errors.

Filtering People

The first experiments were carried out using a Pioneer I robot in Wean Hall of Carnegie Mellon University. There were several people walking through the environment while the robot was mapping it. The top image of Figure 1.4 shows the map obtained by a standard scan-matching procedure. As can be seen from the figure, the map contains many spurious objects and a huge number of registration errors. The most likely map resulting from the application of our approach is shown in the bottom image of Figure 1.4. The beams labeled as dynamic are drawn white in this figure. This demonstrates that our approach can reliably identify dynamic aspects and is able to learn maps that include the static aspects only.

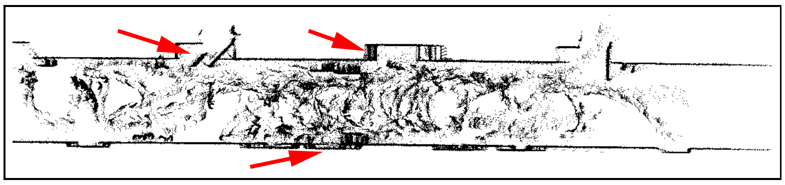

Fig. 1.5. Map obtained in a populated corridor of the Wean Hall at Carnegie Mellon University using the raw input data.

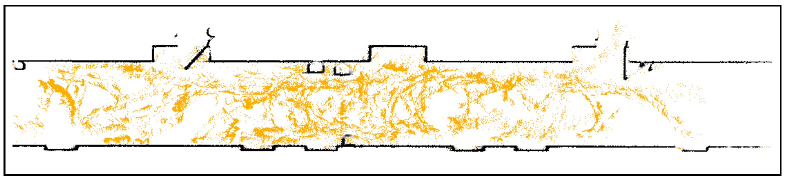

Fig. 1.6. Map generated by our algorithm.

Improved Registration Accuracy by Filtering Dynamic Objects

Besides the fact that the resulting maps contain less spurious objects, our approach also increases the localization accuracy. If dynamic objects are not handled appropriately during localization, matching errors become more likely. Figure 1.5 shows

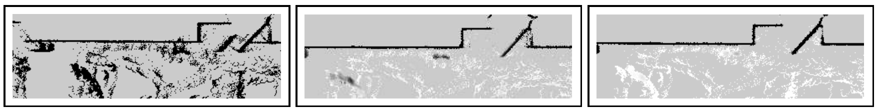

Fig. 1.7. Evolution of the map during EM. The images corresponds to iteration 1, 2, and 6.

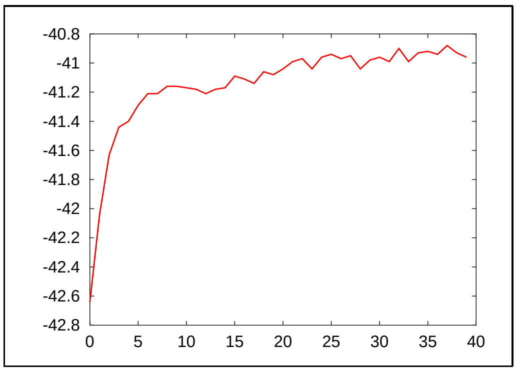

Fig. 1.8. Typical evolution of the log likelihood (Equation (1.6)) during the individual iterations of EM.

a typical map we obtained when mapping a densely populated environment. In this case, we mapped a part of the Wean Hall Corridor at Carnegie Mellon University during peak office hours when many persons were around. Some of them were trying to block the robot, so that the robot had to make detours around them. Therefore, the robot traveled $74\,\mathrm{m}$ with an average speed of $0.15\,\mathrm{m/s}$ ($0.35\,\mathrm{m/s}$ maximum). Despite the fact, that the huge amount of spurious objects make the map virtually useless for navigation tasks, the map also shows serious errors in the alignment. Some of the errors are indicated by arrows in the corresponding figure.

Figure 1.6 shows the map generated by our algorithm. As the figure illustrates, the spurious measurements (indicated by grey/orange dots) have been filtered out completely. Additionally, the alignment of the scans is more accurate.

Figure 1.7 depicts the evolution of a part of the map in the different rounds of the EM. It shows how the beams corresponding to dynamic objects slowly fade out and how the improved estimates about these beams improve the localization accuracy.

Figure 1.8 plots a typical evolution of $E_c[\ln p(c, z \mid x, m) \mid x, m, d]$ over the different iterations of our algorithm. It illustrates that our algorithm in fact maximizes the overall log likelihood. Please note that this curve generally is not monotonic because of the incremental maximum-likelihood solution to the SLAM problem. Slight variations in the pose can have negative effects in future steps, so that the map likelihood can decrease. However, we never observed significant decrease of the log likelihood.

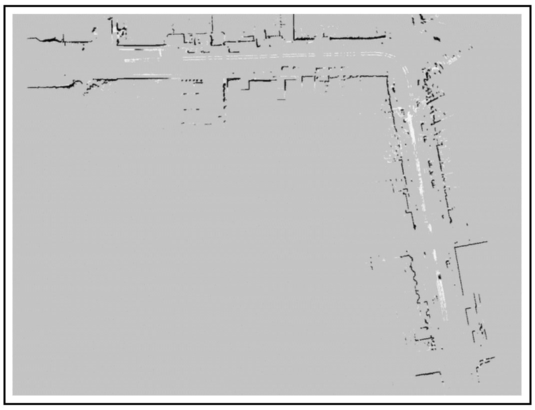

Fig. 1.9. Map of an outdoor scene after filtering dynamic objects.

Generating Large-Scale Outdoor Maps

To evaluate the capability of our technique to deal with arbitrary features, we mounted a laser-range scanner on a car and drove approximately $1\ \mathrm{km}$ through Pittsburgh, PA, USA (Corner between Craig Street and Forbes Avenue). The maximum speed of the car was $35\ \mathrm{MPH}$ in this experiment. We then applied our approach to the recorded data. The map generated by our algorithm is shown in Figure 1.9. Whereas the black dots correspond to the static objects in the scene, the white dots are those which are filtered out using our approach. Again, most of the dynamics of the scene could be removed. Only a few cars could not be identified as dynamic objects. This is mainly because we quickly passed cars waiting for turns and because we drove along the path only once. Please also note that due to the lack of a GPS, the map had to be computed without any odometry information.

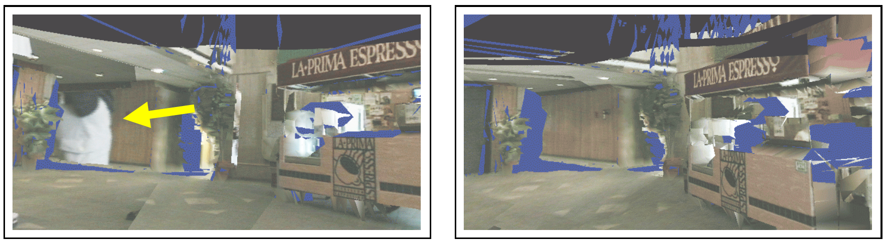

Fig. 1.10. The images show textured 3d models of the Wean Hall lobby obtained without (left image) and with filtering (right image) dynamic aspects.

Generating Textured 3D Maps

To demonstrate that our approach is not limited to 2d range data, we carried out several experiments with the mobile robot Robin which is equipped with a laser-scanner mounted on an AMTEC pan/tilt unit. On top of this scanner, we installed a camera which allows us to obtain textured 3d maps of an environment. Additionally, this robot contains a horizontally scanning laser range finder which we used in our experiments to determine dynamic objects. To label the beams in the 3d data as dynamic, we use a bounding box around the dynamic 2d points. To filter dynamic objects in the textures recorded with Robin's cameras, we choose for every polygon that image which has the highest likelihood of containing static aspects only. The left image of Figure 1.10 shows one particular view of a model obtained without filtering of dynamic objects. The arrow indicates a polygon whose texture contains fractions of an image of a person which walked through the scene while the robot was scanning it. After applying our approach, the corresponding beams and parts of the pictures were filtered out. The resulting model shown in the right image of Figure 1.10 therefore only contains textures showing static objects.

1.5.2 Learning Configurations of Environments

The second set of experiments is designed to illustrate that our approach to learning environmental configurations yields accurate models and at the same time improves the localization capabilities of a robot.

Application in an Office Environment

The first experiment on learning typical environmental configurations has been carried out in a typical office environment. The data was recorded by steering the robot through the environment while the states of the doors changed. To obtain a more accurate pose estimation than the raw odometry information, we apply an incremental scan-matching technique. Figure 1.11 depicts the resulting patch-map. For the three sub-maps that contain the doors whose states were changed during the experiment our algorithm was able to learn all configurations that occurred. The sub-maps and their corresponding patches are shown in the same figure.

[tbh]

The second experiment is designed to illustrate the advantages of our map representation for mobile robot localization in quasi-static environments compared to standard MCL. The data used for this experiment was obtained in the same office environment as above. We placed a box at three different locations in the corridor. The resulting map including all patches obtained via clustering is depicted in Figure 1.12. Note that the tiles in the map illustrate the average over all patches. To evaluate the localization accuracy obtained with our map representation, we compare the pose estimates to that of a standard MCL using an occupancy grid map as well as a grid map obtained by filtering out dynamic objects [18].

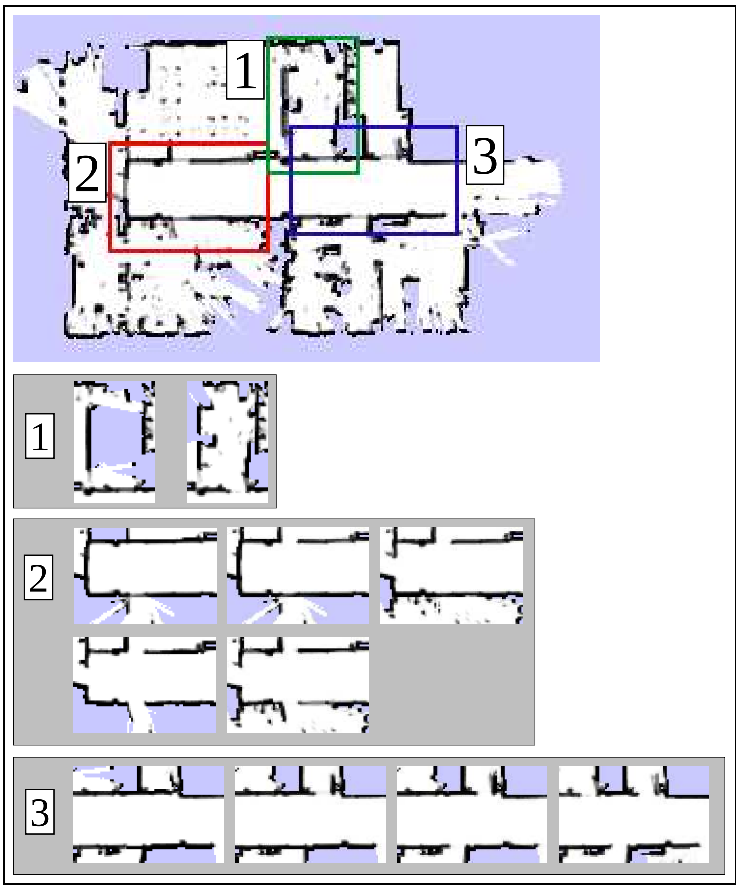

Fig. 1.11. A patch-map representing the different configurations learned for the individual sub-maps in a typical office environment.

Figure 1.13 plots the localization error over time for the three different representations. The error was determined as the weighted average distance from the poses of the particles to the ground truth, where each weight is given by the importance factor of the corresponding particle. In the beginning of this experiment, the robot traveled through static areas so that all localization methods performed equally well. Close to the end, the robot traveled through the dynamic areas, which results in high pose errors for both alternative approaches. In contrast to that, our technique constantly yields a high localization accuracy and correctly tracks the robot.

To further illustrate, how our extended MCL is able to estimate the current state of the environment, Figure 1.14 shows the path of the robot through a non-static area. Figure 1.15 plots the corresponding posterior probabilities for two different patches belonging to one sub-map. At time step 15, the robot entered the corresponding sub-map. At this point in time, the robot correctly identified, that the particles, which

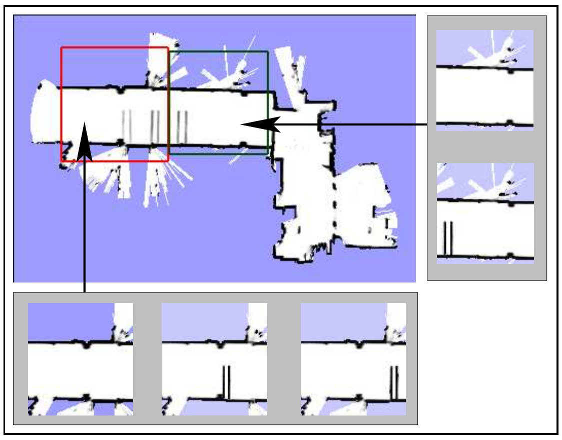

Fig. 1.12. A patch-map with the different configurations for the individual patches.

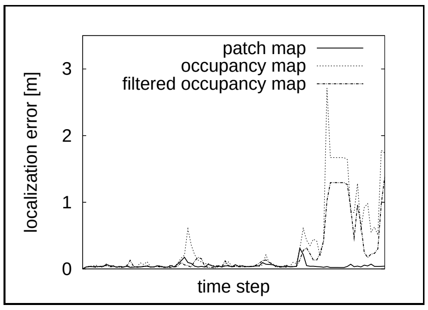

Fig. 1.13. The error in the pose estimate over time. As can be seen, using our approach the quality of the localization is higher compared to approaches using grid maps.

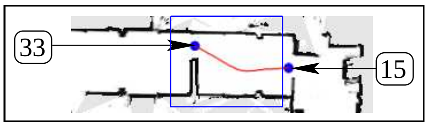

Fig. 1.14. The traveled path of the robot with two time labels. During its motion, the robot correctly identified the current state of the environment (see Figure 1.15).

localize the robot in patch 1, performed much better than the samples using patch 0. Due to the re-samplings in MCL, particles with a low importance weight are more likely to be replaced by particles with a high importance weight. Over a sequence of integrated measurements and re-samplings, this led to an probability close to 1 that the environment looked like the map represented by patch 1 (which exactly corresponded to the ground truth in that situation).

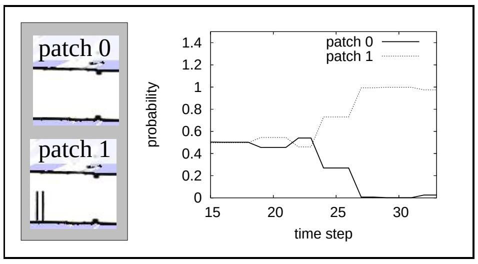

Fig. 1.15. The left image depicts the two possible patches whereas the graph on the right plots the probability of both patches according to the sample set. As can be seen, the robot identified that patch 1 correctly models the configuration of the environment.

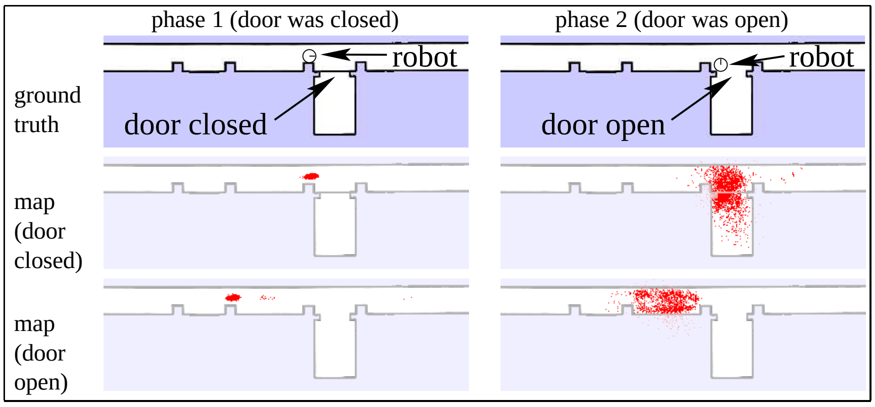

Fig. 1.16. In the beginning the door was closed (left column) but was later on opened (right column). The first row depicts the ground truth, whereas the second row illustrates the particle distributions in case the door is supposed to be closed in the occupancy grid map, whereas no door was mapped in the third row.

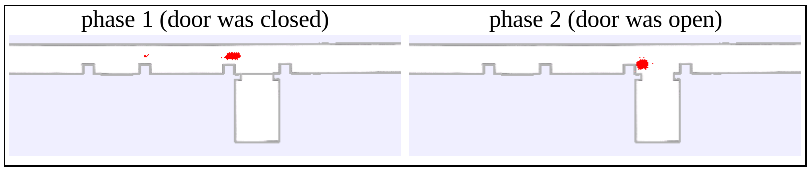

Fig. 1.17. Particle clouds obtained with our algorithm for the same situations as depicted in Figure 1.16.

Global Localization

Additionally, we evaluated all three techniques in a simulated global localization task. We compared our approach using two patches to represent the state of the door with standard MCL using occupancy grid maps (see Figure 1.16 and 1.17). In one experiment, the occupancy grid map contained the closed door and in the second one the open door. During localization, the robot mostly moved in front of the door, which was closed in the beginning and opened in the second phase of the experiment.

As can be seen in left column of Figure 1.16 and 1.17, the MCL approach which uses the occupancy grid that models the closed door as well as our approach lead to a correct pose estimate. In contrast to that, the occupancy grid, which models the open door causes the filter to diverge. In the second phase of the experiment, the door was opened and the robot again moved some meters in front of the door (see right column of the same figure). At this point in time, the MCL technique using the occupancy grid, which models the closed door cannot track the correct pose anymore, whereas our approach is able to correctly estimate the pose of the robot. This simulated experiment again illustrates that the knowledge about possible configurations of the environment is important for mobile robot localization. Without this knowledge, the robot is not able to correctly estimate its pose in non-static environments.

Map Clustering

The last experiment is designed to illustrate the map clustering process. The input to the clustering was a set of 17 local grid maps. The fuzzy k-means clustering algorithm started with a single cluster, which is given by the mean computed over all 17 maps. The result is depicted in the first row of Figure 1.18. The algorithm then increased the number of clusters and re-computed the means in each step. In the fifth iteration the newly created cluster is more or less equal to cluster 3. Therefore, the BIC decreased and the clustering algorithm terminated with the model depicted in the forth row of Figure 1.18.

1.6 Related Work

For mobile robots, there exist several approaches to mapping in dynamic environments. The approaches mostly relevant to the approach to filtering beams reflected by dynamic objects are the methods developed by Wang *et al.* [34] and our previous work described in [17]. Wang *et al.* [34] use a heuristic and feature-based approach to identify dynamic objects in range scans. The corresponding measurements are then filtered out during 2d scan registration. In our pervious work [17], we describe an approach to track persons in range scans and to remove the corresponding data during the registration and mapping process. Recently, Montesano *et al.* [24] describe an algorithm for simultaneously tracking moving objects and estimating the pose of the vehicle and landmarks. They also describe how to utilize the estimates during navigation. Compared to these techniques, our algorithm presented in this chapter does not rely on any pre-defined features or motion models. Rather, it considers every measurement individually and estimates a posterior about whether or not this data item has been generated by a dynamic object.

From a more general perspective, the problem of estimating dynamic aspects in data can be regarded as an outlier detection problem, since the spurious measurements are data items that do not correspond to the static aspects of the environment

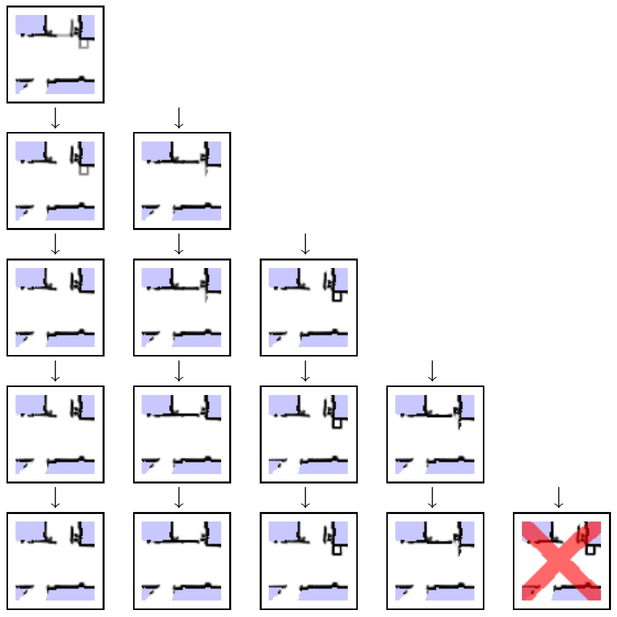

Fig. 1.18. Iterations of the map clustering process. Our approach repeatedly adds new clusters until no improvement is achieved by introducing new clusters (with respect to the BIC). Here, the algorithm ended up with 4 clusters, since cluster 3 and 5 are redundant.

which are to be estimated. The identification of outliers is an important subtask in various application areas such as data mining [8, 19, 27], correspondence establishment [6, 11], clustering [14], or statistics [5]. In all these fields, errors in the data reduce the accuracy of the resulting models and thus can lead to a decreased performance whenever the learned models are used for prediction or robot navigation, for example. The problem considered in this chapter differs from these approaches in the fact that outliers cannot be detected solely based on their distance to the other data items. Rather, the measurements first have to be interpreted and transformed into a global representation (map) before individual measurements can be identified as outliers.

There has been work on updating maps or improving localization in populated environments. For example, in the system described in [9], a given static map is temporarily updated using the most recent sensory input. This allows the robot to consider areas blocked by people in the environment during path planning. Montemerlo *et al.* [22] present an approach to simultaneous localization and people tracking. This approach simultaneously maintains multiple hypotheses about the pose of the robot and people in its vicinity and in this way yields more robust estimates. Siegwart *et al.* [32] present a team of tour-guide robots that operates in a populated exhibition. Their system uses line features for localization and has been reported to successfully filter range-measurements reflected by persons. Fox *et al.* [15] present a probabilistic technique to identify range measurements that do not correspond to the given model of the environment. In contrast to our approach, these methods require a given and fixed map which is used for localization and for the extraction of the features corresponding to the people. Our filtering technique, in contrast, does not require a given map. Rather it learns the map from scratch using the data acquired with the robot's sensors.

Additionally, several authors have proposed techniques for learning dynamic objects of maps of dynamic aspects with mobile robots. For example, Anguelov *et al.* [2] present an approach which aims to learn models of non-stationary objects from proximity data. The object shapes are estimated by applying a hierarchical EM algorithm based on occupancy grids recorded at different points in time. The main difference to our approach to model quasi-static aspects is that we estimate typical configurations of the environment and do not address the problem of learning geometric models for different types of non-stationary obstacles.

Schulz and Burgard [29] proposed a probabilistic algorithm to estimate the state of dynamic objects in an environment. Avots *et al.* [4] apply a Rao-Blackwellized particle filter to estimate the state of doors. Both approaches, however, require a parameterized model of the environment that includes the dynamic objects such as doors. Anguelov *et al.* [3] uses an EM-based approach to detect doors and to estimate their states. Thereby, they take into account shape, color, and motion properties of wall-like objects. In contrast to these works, the approach presented in this chapter is independent of the type of quasi-static object and can learn arbitrary configurations of the environment.

Yamauchi and Beer [35] describe a network of places in which links model a connection between different places. These links may dynamically change their traversability. To deal with these dynamic aspects, they store a confidence value which is updated according to successful or unsuccessful attempts to traverses that link. In the context of landmark-based mapping, the approach presented by Andrade-Cetto and Senafeliu [1] is able to remove landmarks which are not observed anymore from the posterior.

Romero *et al.* [28] describe an approach to globally localize a mobile robot in static environments in which a clustering algorithm is applied to group similar places in the environment. In this way, the robot is able to reduce the number of possible pose hypotheses which speeds up the probabilistic localization process.

In a very recent work, Bieber and Duckett [7] proposed an approach that incorporates changes of the environment into the map representation. Compared to our work, they model temporal changes of local maps whereas we aim to identify the different configurations of the environment.

Our approach to learning typical configurations is designed to explicitely model possible states of the environment, like, e.g., open and closed doors or moved tables. As we have demonstrated in this chapter, it can be used in addition to the filtering techniques. We also demonstrated that the different environmental state hypotheses enable a mobile robot to more reliably localize itself and to also estimate the current configuration of its surroundings.

1.7 Conclusions

In this chapter, we presented probabilistic approaches to mapping in dynamic environments. We first presented an algorithm based on the EM algorithm that interleaves the identification of measurements that correspond to dynamic objects with a mapping and localization algorithm. In this way, it incrementally improves its estimate about spurious measurements and the quality of the map. The finally obtained maps contain less spurious objects and also are more accurate because of the improved range registration. Additionally, we presented a novel approach to model quasi-static environments using a mobile robot. In areas where dynamic aspects are detected, our approach creates local maps and estimates for each sub-map clusters of possible configurations of the corresponding space in the environment. Furthermore, we described how to extend Monte-Carlo localization to utilize the information about the different possible environmental states while localizing a vehicle. Our approach has been implemented and tested on real robots as well as in simulation. The experiments demonstrate, that our technique yields a higher localization accuracy compared to Monte-Carlo localization based on standard grid maps even such obtained after filtering out measurements reflected by dynamic objects.

Our techniques have been implemented and tested on different platforms. In several experiments carried out in indoor and outdoor environments we demonstrated that our approaches yield highly accurate maps. The results illustrate that our approaches can reliably estimate filter beams reflected by dynamic environments and that quasi-static aspects can be modeled accurately.

Acknowledgment

The authors would like to thank Sebastian Thrun and Rudolph Triebel for fruitful discussions. This work has partly been supported by the German Research Foundation (DFG) under contract number SFB/TR-8 (project A3) and by the EC under contract number FP6-004250-CoSy and FP6-IST-027140-BACS.

References

1. J. Andrade-Cetto and A. Sanfeliu. Concurrent map building and localization in indoor dynamic environments. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 16(3):361–274, 2002.
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, E. Parker, and S. Thrun. Detecting and modeling doors with mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 3777–3774, New Orleans, LA, USA, 2004.

4. D. Avots, E. Lim, R. Thibaux, and S. Thrun. A probabilistic technique for simultaneous localization and door state estimation with mobile robots in dynamic environments. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 521–526, Lausanne, Switzerland, 2002.
5. V. Barnett and T. Lewis. *Outliers in Statistical Data*. Wiley, New York, 1994.
6. P. Besl and N. McKay. A method for registration of 3d shapes. *Trans. Patt. Anal. Mach. Intell. 14(2)*, pages 239–256, 1992.
7. P. Biber and T. Duckett. Dynamic maps for long-term operation of mobile service robots. In *Proc. of Robotics: Science and Systems (RSS)*, 2005. To appear.
8. C.E. Brodley and M.A. Friedl. Identifying and eliminating mislabeled training instances. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 1996.
9. W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1-2), 2000.
10. J.A. Castellanos, J.M.M. Montiel, J. Neira, and J.D. Tardós. The SPmap: A probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 15(5):948–953, 1999.
11. I.J. Cox and S.L. Hingorani. An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on PAMI*, 18(2):138–150, February 1996.
12. F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Leuven, Belgium, 1998.
13. G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In *ICRA'2000 Workshop on Mobile Robot Navigation and Mapping*, San Francisco, CA, USA, 2000.
14. R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2001.
15. D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research (JAIR)*, 11:391–427, 1999.
16. J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. of the IEEE Int. Symp. on Computational Intelligence in Robotics and Automation (CIRA)*, 1999.
17. D. Hähnel, D. Schulz, and W. Burgard. Mobile robot mapping in populated environments. *Journal of the Robotics Society of Japan (JRSJ)*, 7(17):579–598, 2003.
18. D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.
19. George H. John. Robust decision trees: Removing outliers from databases. In *First International Conference on Knowledge Discovery and Data Mining*, pages 174–179, 1995".
20. J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In J. Hollerbach and D. Koditschek, editors, *Proceedings of the Ninth International Symposium on Robotics Research*, pages 169–179, 2000.
21. F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.
22. M. Montemerlo and S. Thrun. Conditional particle filters for simultaneous mobile robot localization and people-tracking (slap). In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2002.

23. M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to simultaneous localization and mapping. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, Edmonton, Canada, 2002.
24. L. Montesano, J. Minguez, and L. Montano. Modeling the static and the dynamic parts of the environment to improve sensor-based navigation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.
25. 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)*, pages 116–121, St. Louis, MO, USA, 1985.
26. K. Murphy. Bayesian map learning in dynamic environments. In *Neural Info. Proc. Systems (NIPS)*, Denver, CO, USA, 1999.
27. S. Ramaswamy, R. Rastogi, and S. Kyuseok. Efficient algorithms for mining outliers from large data sets. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, 2000.
28. L. Romero, E. Morales, and E. Sucar. A hybrid approach to solve the global localozation problem for indoor mobile robots considering sensor's perceptual limitations. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2001.
29. D. Schulz and W. Burgard. Probabilistic state estimation of dynamic objects with a moving mobile robot. *Robotics and Autonomous Systems*, 34(2-3):107–115, 2001.
30. G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2), 1978.
31. H. Shatkay. *Learning Models for Robot Navigation*. PhD thesis, Computer Science Department, Brown University, Providence, RI, 1998.
32. R. Siegwart, K.O. Arras, S. Bouabdallah, D. Burnier, G. Froidevaux, X. Greppin, B. Jensen, A. Lorotte, L. Mayor, M. Meisser, R. Philippsen, R. Piguet, G. Ramel, G. Terrien, and N. Tomatis. Robox at Expo.02: A large-scale installation of personal robots. *Journal of Robotics & Autonomous Systems*, 42(3-4), 2003.
33. S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *Int. Journal of Robotics Research*, 20(5):335–363, 2001.
34. C.-C. Wang and C. Thorpe. Simultaneous localization and mapping with detection and tracking of moving objects. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2002.
35. B. Yamauchi and R. Beer. Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man and Cybernetics*, 26(3):496–505, 1996.