Autonomous Navigation in Dynamic Environments
Editorial Advisory Board

Herman Bruyninckx, KU Leuven, Belgium
Raja Chatila, LAAS, France
Henrik Christensen, Georgia Institute of Technology, USA
Peter Corke, CSIRO, Australia
Paolo Dario, Scuola Superiore Sant’Anna Pisa, Italy
Rüdiger Dillmann, Universität Karlsruhe, Germany
Ken Goldberg, UC Berkeley, USA
John Hollerbach, University of Utah, USA
Makoto Kaneko, Hiroshima University, Japan
Lydia Kavraki, Rice University, USA
Sukhan Lee, Sungkyunkwan University, Korea
Tim Salcudean, University of British Columbia, Canada
Sebastian Thrun, Stanford University, USA
Yangsheng Xu, Chinese University of Hong Kong, PRC
Shin’ichi Yuta, Tsukuba University, Japan

STAR (Springer Tracts in Advanced Robotics) has been promoted under the auspices of EURON (European Robotics Research Network)
At the dawn of the new millennium, robotics is undergoing a major transformation in scope and dimension. From a largely dominant industrial focus, robotics is rapidly expanding into the challenges of unstructured environments. Interacting with, assisting, serving, and exploring with humans, the emerging robots will increasingly touch people and their lives.

The goal of the new series of Springer Tracts in Advanced Robotics (STAR) is to bring, in a timely fashion, the latest advances and developments in robotics on the basis of their significance and quality. It is our hope that the wider dissemination of research developments will stimulate more exchanges and collaborations among the research community and contribute to further advancement of this rapidly growing field.

The collection edited by Christian Laugier and Raja Chatila is the third one in the series on mapping and navigation, and is focused on the problem of autonomous navigation in dynamic environments. The state of the art is surveyed, a number of challenging technical aspects are discussed and upcoming technologies are addressed.

The ambitious goal is to lay down the foundation for a broad class of robot mapping and navigation methodologies for indoors, outdoors, and even exploratory missions. Future service robots and intelligent vehicles are waiting for effective solutions to such kind of problems.

The material is organised in three parts; namely, dynamic world understanding and modeling for safe navigation, obstacle avoidance and motion planning in dynamic environments, and human-robot interaction. Gathering some of the authorities working in the field, this volume constitutes a fine addition to the Series!

Naples, Italy
May 2007

Bruno Siciliano
STAR Editor
Preface

Autonomous navigation in open and dynamic environments is a challenging problem and an increasingly important topic for the near future applications of mobile robotics and intelligent vehicles. It is more and more obvious that future robot systems will have to share a common physical space with humans while intensively interacting with them, for the purpose of assisting peoples in various everyday tasks at work, at home, or during transportation. This means that such robots will clearly have to move safely in an open and unpredictable world, while interacting in various ways with peoples.

Navigation techniques in static environments are well known, but it is not clear how these techniques can cope with dynamic environments including people, other robots, and changing landmarks and environment features. Today robotics technologies have shown their ability to solve various navigation problems mainly in static environments; the success of the last DARPA Grand Challenge in USA is a clear example of such current robotics capabilities. However, moving to more dynamic spaces often populated by human is still an open issue. This is why the new DARPA Grand Challenge is focusing onto city environments and urban car traffic. From the application point of view, future Service Robots and Intelligent Vehicles are waiting for robust solutions to this problem.

The purpose of this book is to address the challenging problem of Autonomous Navigation in Dynamic Environments, and to present new ideas and approaches in this newly emerging technical domain. The book surveys the state-of-the-art, discusses in detail various related challenging technical aspects, and addresses upcoming technologies in this field. The aim of the book is to establish a foundation for a broad class of mobile robot mapping and navigation methodologies for indoor, outdoor, and exploratory missions.

Three main topics located on the cutting edge of the state of the art are addressed, from both the theoretical and technological point of views: Dynamic world understanding and modelling for safe navigation, Obstacle avoidance and motion planning in dynamic environments, and Human-robot physical interactions. Several models and approaches are proposed for solving problems such as Simultaneous Localization and Mapping (SLAM) in dynamic environments, Mobile obstacle detection and tracking, World state estimation and motion prediction, Safe navigation in dynamic environments, Motion planning in dynamic environments, Robust decision making under uncertainty, and Human-Robot physical interactions.

Christian Laugier
Research Director at INRIA (France)
Raja Chatila
Research Director at LAAS-CNRS (France)
Contents

Part I: Dynamic World Understanding and Modelling for Safe Navigation

1 Mobile Robot Map Learning from Range Data in Dynamic Environments
   Wolfram Burgard, Cyrill Stachniss, Dirk Hähnel .............................. 3

2 Optical Flow Approaches for Self-supervised Learning in Autonomous Mobile Robot Navigation
   Andrew Lookingbill, David Lieb, Sebastian Thrun ......................... 29

3 Steps Towards Safe Navigation in Open and Dynamic Environments
   Christian Laugier, Stéphane Petti, Dizan Vasquez, Manuel Yguel,
   Thierry Fraichard, Olivier Aycard............................................ 45

Part II: Obstacle Avoidance and Motion Planning in Dynamic Environments

4 Provably Safe Motions Strategies for Mobile Robots in Dynamic Domains
   Rachid Alami, K. Madhava Krishna, Thierry Siméon....................... 85

5 Motion Planning in Dynamic Environments
   Zvi Shiller, Frederic Large, Sepanta Sekhavat, Christian Laugier ........ 107

6 Recursive Agent Modeling with Probabilistic Velocity Obstacles for Mobile Robot Navigation Among Humans
   Boris Kluge, Erwin Prassler.................................................... 121
7  Towards Real-Time Sensor-Based Path Planning in Highly Dynamic Environments
Roland Philippsen, Björn Jensen, Roland Siegwart ....................... 135

Part III: Human-Robot Physical Interactions

8  Tasking Everyday Interaction
Elena Pacchierotti, Patric Jensfelt, Henrik I. Christensen ............... 151

Index .......................................................................................... 169
Part I
Dynamic World Understanding and Modelling for Safe Navigation
Mobile Robot Map Learning from Range Data in Dynamic Environments

Wolfram Burgard\textsuperscript{1}, Cyrill Stachniss\textsuperscript{1,2}, and Dirk Hähnel\textsuperscript{1,3}

\textsuperscript{1} University of Freiburg, Dept. of Computer Science, 79110 Freiburg, Germany
\textsuperscript{2} Eidgenössische Technische Hochschule Zürich (ETH), 8092 Zürich, Switzerland
\textsuperscript{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 \textit{simultaneous localization and mapping problem (SLAM)} \cite{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 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 explicitly 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 \( c \).

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

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

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.

\[
\ln p(z, c \mid x, m) = \ln \prod_{t=1}^{T} \prod_{n=1}^{N} p(z_{t,n} \mid c_{t,n}, x_{t}, m) \cdot p(c_t)
\]

\[
= N \cdot \sum_{t=1}^{T} \ln p(c_t) + \sum_{t=1}^{T} \sum_{n=1}^{N} \ln p(z_{t,n} \mid c_{t,n}, x_{t}, m)
\]

\[
= N \cdot \sum_{t=1}^{T} \ln p(c_t) + \sum_{t=1}^{T} \sum_{n=1}^{N} \left[ (1 - \zeta_{t,n}) \cdot \left[ c_{t,n} \cdot \ln m_{f(x_{t},n,z_{t,n})} 
\right.ight.
\]

\[
+ (1 - c_{t,n}) \cdot \ln(1 - m_{f(x_{t,n},z_{t,n})}) \right] + \sum_{k=0}^{z_{t,n}-1} \ln(1 - m_{f(x_{t},n,k)}) \right]
\]

(1.6)

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 \( E_c[\ln p(c, z \mid x, m) \mid z, x, m, d] \) 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:

\[
E_c[\ln p(z, c \mid x, m) \mid z, x, m] = 
\gamma + \sum_{t=1}^{T} \sum_{n=1}^{N} \left[ e_{t,n} \cdot (1 - \zeta_{t,n}) \cdot \ln m_{f(x_{t},n,z_{t,n})} 
\right.
\]

\[
+ (1 - e_{t,n}) \cdot (1 - \zeta_{t,n}) \cdot \ln(1 - m_{f(x_{t,n},z_{t,n})}) 
\]

\[
+ \sum_{k=0}^{z_{t,n}-1} \ln(1 - m_{f(x_{t},n,k)}) \right] \]

(1.7)

For the sake of brevity, we use the term

\[
e_{t,n} = E_c[c_{t,n} \mid z_{t,n}, x_{t}, m] \quad (1.8)
\]

in this equation. The term

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

is computed from the prior \( p(c_t) \) of the measurements which is independent of \( z, x, \) and \( m \). Accordingly, \( \gamma \) 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:

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

where

\[ e_{t,n} = \frac{1}{p(c_{t,n}) + (1 - p(c_{t,n})) (\frac{1}{m_f(x_{t,n}, z_{t,n})} - 1)} \]  (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 al-
gorithm, we additionally consider the estimations $e_{t,n}$ that measurement $n$ at time $t$ is
caused by a static object of the environment:

\[ \hat{x}_t = \arg \max_{x_t} \{ p(z_t \mid c_t, x_t, \hat{m}_{[t-1]}) \cdot p(x_t \mid u_{t-1}, \hat{x}_{t-1}) \} \]  (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 $x_1, \ldots, x_t$, have been computed, we can derive a closed-form solution for $m_{[t]}$. We want to deter-
mine 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.

\[
\hat{m}^{[t]} = \arg\max_m \left( \sum_{j=1}^{J} \sum_{t=1}^{T} \sum_{n=1}^{N} \left[ I(f(x_t, n, z_{t,n}) = j) \cdot (1 - \zeta_{t,n}) \cdot (e_{t,n} \ln m_j + (1 - e_{t,n}) \ln(1 - m_j)) + \sum_{k=0}^{z_{t,n}-1} I(f(x_t, n, k) = j) \cdot \ln(1 - m_j) \right] \right) \tag{1.12}
\]

Now suppose, we define

\[
\bar{I}(x, n, k, j) := I(f(x, n, k) = j)
\]

and

\[
\alpha_j := \sum_{t=1}^{T} \sum_{n=1}^{N} \bar{I}(x_t, n, z_{t,n}, j) \cdot (1 - \zeta_{t,n}) \cdot e_{t,n}
\]

\[
\beta_j := \sum_{t=1}^{T} \sum_{n=1}^{N} \left( \bar{I}(x_t, n, z_{t,n}, j) \cdot (1 - \zeta_{t,n}) \cdot (1 - e_{t,n}) + \sum_{k=0}^{z_{t,n}-1} I(f(x_t, n, k) = j) \right)
\]

The quantity \( \alpha_j \) 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 \( \beta_j \), 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 \( \alpha_j \) corresponds to the number of times a beam that does not have the maximum length ends in \( j \). In contrast to that, \( \beta_j \) is the number of times a beam covers a cell.
Using the definitions of $\alpha_j$ and $\beta_j$, Equation (1.12) turns into

$$m^{[t]} = \arg\max_{m} \left( \sum_{j=1}^{J} \alpha_j \ln m_j + \beta_j \ln(1 - m_j) \right)$$  \hfill (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:

$$\frac{\partial m}{\partial m_j} = \frac{\alpha_j}{m_j} - \frac{\beta_j}{1 - m_j} = 0$$  \hfill (1.14)

By straightforward mathematical transformations we obtain

$$m_j = \frac{\alpha_j}{\alpha_j + \beta_j}. \hfill (1.15)$$

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 $\ln(1 - m_j)$. 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 explicitly 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 $\Phi$ of observation sequences for each sub-map

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

(1.16)

where each

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

(1.17)

Here $z_t$ describes an observation obtained at time $t$. For each sequence $\phi_i$ 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 $\xi$ 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

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

(1.18)

where $\epsilon$ 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 $\theta$ using the Bayesian Information Criterion (BIC) [30]:

$$BIC = \log P(d | \theta) - \frac{|\theta|}{2} \log n$$  \hspace{1cm} (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 $\theta$ 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 $t$

$$p(x_t, m^{[t]} \mid z_{1:t}, u_{0:t-1}) = \eta \cdot p(z_t \mid x_t, m^{[t]}, z_{1:t-1}, u_{0:t-1}) \cdot p(x_t, m^{[t]} \mid z_{1:t-1}, u_{0:t-1}).$$  \hspace{1cm} (1.20)