Local modeling of dynamic environments for navigation purposes

Luis Montesano Javier Minguez Luis Montano
Dpto. de Informática e Ingeniería de Sistemas, Universidad de Zaragoza (Spain).
E-mail: {montesano,jminguez,montano}@unizar.es

Abstract

This paper addresses the modeling of the static and dynamic parts of the scenario and how to use this information within a real sensor-based navigation system. The contribution in the modeling aspect is a formulation of the detection and tracking of mobile objects and the mapping of the static structure in such a way that the nature (static/dynamic) of the observations is included in the estimation process. This is achieved by a set of filters tracking the moving objects and a map of the static structure constructed on line. In addition, this paper discusses how this modeling module is integrated in a real sensor-based navigation system taking advantage selectively of the dynamic and static information. The experimental results confirm that the complete navigation system is able to move a vehicle in unknown and dynamic scenarios. Furthermore, the system overcomes many of the limitations of previous systems associated to the ability to distinguish the nature of the parts of the scenario.

I. Introduction

Current autonomous motion systems have to deal with unknown and dynamic environments. The presence of people or other moving objects turn the scenario into a evolving and unpredictable one. The vehicle motion in this type of environments is computed by hybrid architectures that combine aspects of modeling, tactical planning and obstacle avoidance. The skill to model the environment distinguishing the dynamic and static parts opens a new dimension in these systems, since it allows a selective treatment of this information to improve the performance of all the modules of the architecture. This greatly ameliorates the overall behavior of the sensor-based navigation system. In this paper, we present a modeling module that includes the detection and tracking of moving objects, and its integration within the navigation architecture that currently works on our wheelchair vehicle.

The modeling of the static parts of the environment from a mobile robot has been widely studied in the last years. There exist different approaches ranging from local techniques like incremental maximum likelihood mapping [15], [14], [20], batch mapping algorithms [17] or online Simultaneous Localization and Map Building (SLAM) [8], [7], [40]. On the other hand, the Tracking of Moving Objects (TMO) is also a well-studied problem [3], [5]. However, a robust modeling of dynamic environments requires to perform both tasks at the same time.

Most of the systems that address this problem use the sensor information to model the static parts of the environment with a map and apply some filtering techniques to track the moving objects. For example [36], [16] use a feature based approach to detect the moving objects in the range profile of the laser scans. Next, they use Joint Probabilistic Data Association particle filters to track the moving objects and a probabilistic SLAM technique to build the map. In [43] a rigorous formulation of the TMO-SLAM problem is provided assuming a known classification of the observations into static and dynamic. The detection algorithm is based on the violation of the free space and the tracking on extended Kalman filters (EKF). So as to model the free space, the method builds a local dense grid map. Another technique detects and filters dynamic measurements while solving the mapping problem using the Expectation Maximization (EM) algorithm [17]. It classifies the observations during the expectation step and correct the robot pose in the maximization step. However, these methods do not take into account the uncertainty of the robot motion in the classification step. Thus, if many observations are classified incorrectly, difficulties may arise in the presence of large odometry errors, since these errors affect the precision and the convergence of the previous algorithms. However, incorporating the classification process within the estimation process is hard due to the combination of discrete and continuous hidden variables and results in intensive computing algorithms.

Another approach for range sensors uses scan mathching techniques to compute the robot displacement between pairs of consecutive scans and incrementally build a local map. Unfortunately, scan matching techniques are not robust in dynamic environments. Some authors have proposed extensions to minimize the effect of moving objects

observations features. For instance, [4] uses a sectorized version of the Iterative Dual Correspondence algorithm [21] and discard those sectors with big correspondence errors. In [18], the filtering is performed using a EM to detect spurious measurements. However, these extensions do not explicitly track the moving objects, but discard their observations.

The second key aspect is how the use the information about the static and dynamic parts of the environment within the motion system. Most of the hybrid motion systems that work in unknown and dynamic scenarios assume that all the sensory information available is instantaneously static (there is no dynamic obstacle modeling) [1], [25], [41], [6], [34], [38]. These systems perform very well in most surroundings due to a combination of environment modeling (short term memory), tactical planning (that allows to avoid the potential minima and the trap situations) and the obstacle avoidance (that computes the motion using the information of the previous modules). Nevertheless, there exist situations in which the behavior of these systems is degraded due to the necessity to model the dynamic part of the environment. Significant examples are an obstacle moving in a direction where it will collide with the robot, or people that temporarily obstruct zones of passage like doors or corridors.

On the other hand there are systems that model both, the static and dynamic parts of the environment. This information is used next to improve the vehicle location and the avoidance of obstacles [19], [35]. Nevertheless, the limitation of these systems lies in the lack of an integration architecture that combines planning and collision avoidance techniques to solve the problems associated with navigation. For example they fall in trap situations or cyclic behaviors, or they exhibit difficulties to compute safe motion in troublesome scenarios (very dense, cluttered and complex). A reliable solution must address both: a module able to model the static and dynamic parts of the scenario, and the integration within an architecture able to deal with the typical navigation issues.

In this paper, we present present a local mapping maximum likelihood technique to model dynamic environments. It models the static structure with a local grid map and tracks the moving objects using a set of filters. The classification of the observations is implicitly included in the estimation process using the EM algorithm. As a result, the algorithm is able to improve the measurement classification as the pose error decreases. The second contribution of this paper is the integration of the local mapping algorithm in our navigation architecture. The usage of the static and dynamic information selectively by the planning and obstacle avoidance modules allows to avoid the undesirable situations outlined previously, while fully exploiting the advantages of an hybrid sensor-based navigation system. We present experimental results with a wheelchair vehicle working in a realistic scenario.

The rest of the paper is organized as follows. Next section briefly introduces the navigation architecture. Section III describes our method to model dynamic environments and Section IV shows how this information is used in our system. Experimental results are presented in Section V.

II. Navigation System

The capabilities required for the navigation of an autonomous robot are tied up with the specific application and vehicle. For instance, the a priori knowledge, the information provided by the on-board sensors, the motions constraints of the vehicle or the computational power. This usually leads to the development of specific navigation systems that accommodate the requirements of each application.

In all the systems, an important issue is to bound the scope of the mobility system, which is related to the differences between global and local navigation systems (Figure 1). In fact, the concerns of these systems are different. For instance, for global systems, the construction of accurate models and the tracking of the position of the vehicle are important to create global plans and to guarantee motion convergence, while the specific vehicle constraints or real-time execution are not. However, for local systems, simpler local models and rough planning are enough, while motion constraints related to real-time or to the vehicle such as shape, kinematics and dynamics are important to guarantee robust obstacle avoidance. In this Section, we give an overview of the local navigation system architecture [25]. It integrates three modules with the following functionalities: model construction, tactical motion planning and obstacle avoidance,

- **Model Builder Module**: construction of a model of the static and dynamic parts of the environment. We describe the static parts with a probabilistic occupancy grid map (with limited size and that travels centered with the robot), and track the moving objects with a set of filters. This model increases the spatial domain of the planning and is used as local memory for the obstacle avoidance. Section III provides a detailed description of this module.

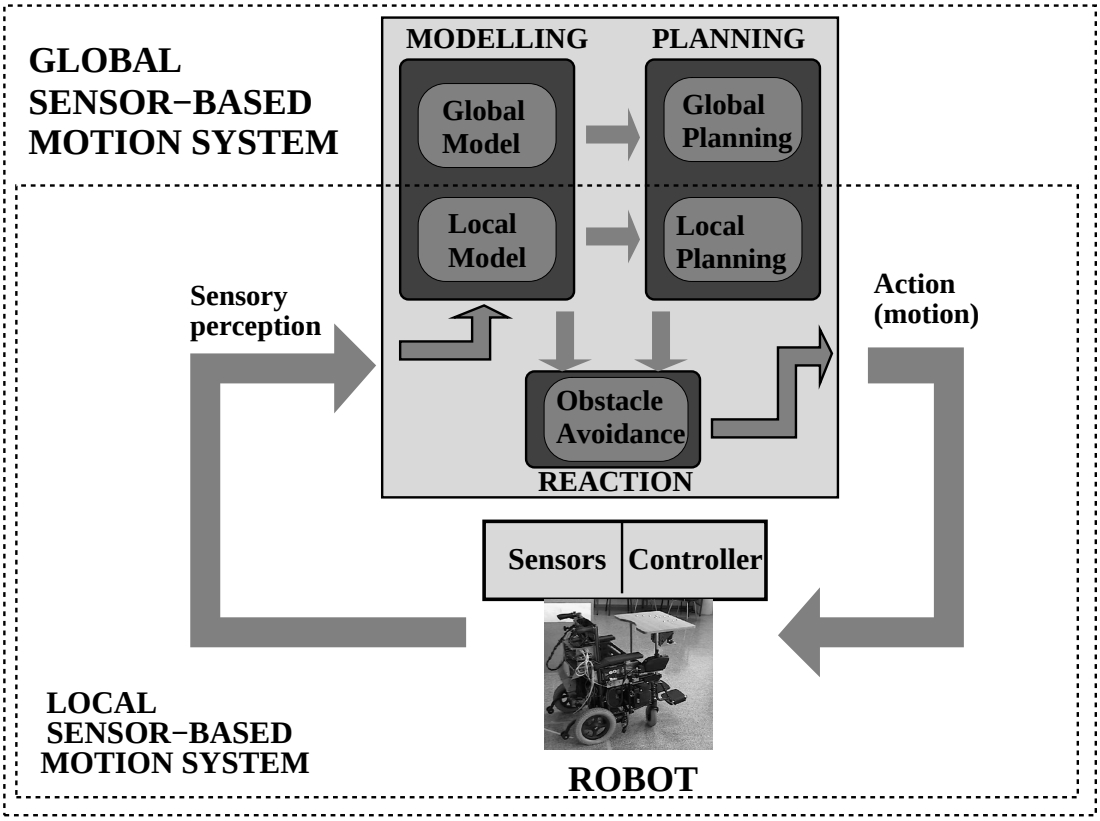


Fig. 1. Overview of the sensor-based navigation system.

- **Planner Module**: extraction of the connectivity of the free space (used to avoid the cyclical motions and trap situations). We developed a planner that computes the existence of a path that joins the robot and goal locations [25]. The planner constructs iteratively a graph whose nodes are locations in the space and the arcs represent sets of free of collision homotopic paths between the nodes. Once the goal is included in the graph, it means that it can be reached through the paths defined by the arcs. This planner avoids the local minima and is very efficient so that it can be executed in real time.
- **Obstacle Avoidance Module**: computation of the collision-free motion. We chose the Nearness Diagram Navigation (ND method in short) [24], which is based on selecting at each time a navigational situation and to apply a motion law adapted to each one. In other words it employs a "divide and conquer" strategy based on situations to simplify the difficulty of the navigation. This method has demonstrated to perform in scenarios that remain troublesome for many existing methods.

Globally the system works as follows (Figure 1): given a laser scan and the odometry of the vehicle, the model builder incorporates this information into the existing model. Next, the static and dynamic information of obstacles in the model is selectively used by the planner module to compute the course to follow to reach the goal (tactical information). Finally, the obstacle avoidance module uses the planner tactical information together with the information of the obstacles (static and dynamic) to generate the target-oriented collision-free motion. The motion is executed by the vehicle controller and the process restarts with a new sensorial measurement. It is important to stress that the three modules work synchronously within the perception - action cycle.

The contribution of this paper resides in the modeling module and in the integration within the navigation system of Figure 1. The objective of this module is to provide the other modules with the information necessary to achieve local sensor based navigation capabilities. There are two important characteristics that greatly determine the requirements of this module. First, the scope of the navigation task is local. This means that the system does not intend to guarantee global convergence toward the goal. It works in unknown and dynamic scenarios which prevent to obtain such a solution (it can even not exist such a solution due to permanent changes in the environment). Moreover, the map must contain all the obstacles perceived by the sensors (dense representation) independently of

their shape and size, must be able to accommodate the changes of the environment and must provide the other modules with the information necessary to avoid obstacles and plan the path to the goal. Finally, it also provides the vehicle localization to minimize the errors of dead reckoning.

Second, the system must close the control loop which imposes real time constraints and requires a high frequency sensor rate to react to the changes. We propose to use a local map with a limited size that travels centered with the robot. As discussed in the next Section this allows us to reduce the complexity of the estimation process and allows a very efficient implementation that matches our real time requirements.

Next, we address the two main issues of this paper: the model builder module and how its information is employed by the other modules.

III. Bayesian modeling of dynamic environments

Let $Z_k = \{z_{k,1}, ..., z_{k,N_z}\}$ be the N_z observations obtained by the robot at time k and u_k the motion command executed at time k. The sets $Z_{1:k} = \{Z_1, ..., Z_k\}$ and $u_{0:k} = \{u_0, ..., u_k\}$ represent the observations and motion commands up to time k. We denote the robot location at time k by x_k, the map of the static environment by M. In addition, let $O_k = \{o_{k,1}, ...o_{k,N_O}\}$ be the location of the N_O moving objects at time k. The objective is to estimate the robot pose x, the map M and the position and velocities of the moving objects O,

$$\begin{aligned} &p(O_k, x_k, M \mid Z_{1:k}, u_{0:k-1}) = \\ &\quad \eta p(Z_k \mid O_k, x_k, M, Z_{1:k-1}, u_{0:k-1}) p(O_k, x_k, M \mid Z_{1:k-1}, u_{0:k-1}) = \\ &\quad \eta p(Z_k \mid O_k, x_k, M) \int\int p(O_k, x_k, M \mid O_{k-1}, x_{k-1}, u_{k-1}) p(O_{k-1}, x_{k-1}, M \mid Z_{1:k-1}, u_{0:k-2}) dx_{k-1} dO_{k-1} \end{aligned} \tag{1}$$

which is the classical Bayes recursive estimator. In the previous formulation the Markov assumption allows us to discard previous commands and measurements and simplifies the likelihood term, $p(Z_k \mid O_k, x_k, M, Z_{1:k-1}, u_{0:k-1})$, and the motion model $p(O_k, x_k, M \mid O_{k-1}, x_{k-1}, u_{k-1})$. In addition, the integration affects only to those variables that change over time, x and O and we make the common assumption that the map does not evolve. Another common assumption is to consider that the objects $o_{k,i}$ move independently which leads to the following update equation,

$$\begin{aligned} p(O_k, x_k, M \mid Z_{1:k-1}, u_{0:k-1}) = &\int p(x_k \mid x_{k-1}, u_{k-1}) \\ &\prod_i^{N_z} \int p(o_{k,i} \mid o_{k-1,i}) p(O_{k-1}, x_{k-1}, M \mid Z_{1:k-1}, u_{0:k-1}) do_{k-1,1} ... do_{k-1,n} dx_{k-1} \end{aligned} \tag{2}$$

The update equation measures how well the observations match the prediction done by the motion model. This task is hard because it also requires to solve the data association problem. In our particular case the observations may come from the static parts of the environment or from the moving objects. Figure 2 shows the graphical representation of the problem. The model contains the continuous variables of Equation 2 and the discrete variables c_j representing the data association (or classification) problem. In general, it is very hard to perform exact inference on such models due to the integration over all the possible correspondences . Therefore, in general one has to use approximations. As described in Section I several works have already addressed this problem. Most of them assume that at least the classification into static and dynamic is known. In this case the data association is restricted to each class of features, static or dynamic. In practice the classification is not available and it is usually computed based on the distribution $p(O_k, x_k, M \mid Z_{1:k-1}, u_{0:k-1})$ using patterns and/or free space constraints. A more robust approach requires to incorporate the nature of the observations within the estimation process. Particle filters [10], [31] provide a powerful representation for this kind of problems as they allow to keep several hypotheses at the same time.

Another approach is to use a Maximum Likelihood technique (ML). The main idea is to compute at each point in time the robot pose x_k that maximizes the likelihood term $p(Z_k \mid O_k, x_k, M)$ where the map M and the objects O_k are computed from the set of poses $\hat{x}_{1:k-1}$ up to time $k-1$,

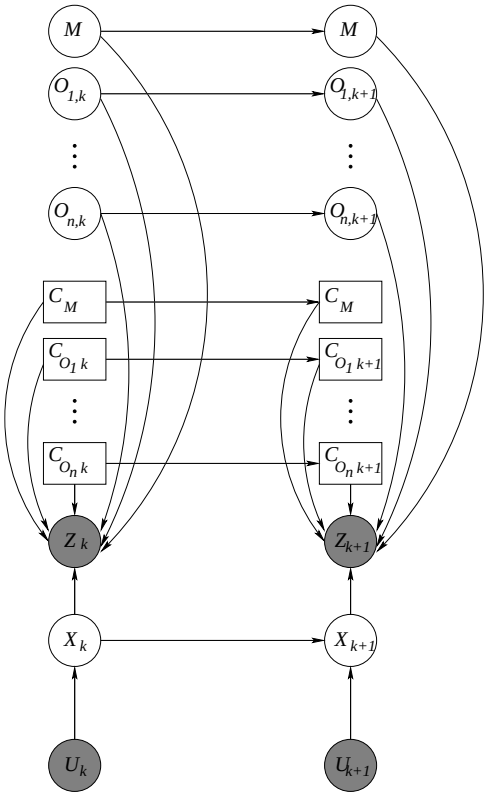


Fig. 2. Graphical representation of the problem including the data associations. The circles represent the continuous variables and the squares discrete ones. The filled nodes are the measurements whereas the empty ones represent hidden variables.

$$\hat{x}_k = arg\max_x p(Z_k \mid O_k, x_k, M)p(x_k \mid \hat{x}_{k-1}, u_{k-1}) \quad (3)$$

$$M_k = f_M(\hat{x}_{1:k-1}, Z_{1:k-1}) \quad (4)$$

$$O_k = f_O(\hat{x}_{1:k-1}, Z_{1:k-1}) \quad (5)$$

where the term $p(x_k \mid \hat{x}_{k-1}, u_{k-1})$ allows introducing the uncertainty of the last robot motion u_{k-1} to constraint the possible solutions of the optimization algorithm.

The advantage of the ML approach is that the classification of the measurements can be included within the maximum likelihood algorithm using the Expectation-Maximization (EM) algorithm. However, the drawback of this type of techniques is that they do not consider the uncertainties and the corresponding correlations of the robot poses, the map and the moving objects. As a consequence, the set of poses is fixed and cannot be modified in subsequent steps. This represents a problem when closing loops if the accumulated error is big. In the case of sensor based navigation, the spatial domain of the problem is small and, thus, we do not need to close this kind of loops. The next Section presents a EM based maximum likelihood algorithm to estimate the robot pose and the nature of the observations that can be integrated in our navigation system.

A. Expectation Maximization Maximum likelihood approach

The Equation 3 requires to maximize the likelihood term $p(Z_k \mid O_k, x_k, M)$ which requires to integrate over all the possible sources (static map or a dynamic object) for each observation. The resulting expression has no close solution and is difficult to maximize due to the unknown classification variables c_j of Figure 2. However, if the source of each observation was known, the resulting likelihood, the complete-data likelihood function, is much simpler and its maximization has a closed form solution (given that we linearized the measurement equation). The EM algorithm allows to solve this type of problems.

1) Expectation Maximization algorithm:
The EM algorithm maximizes the incomplete-data likelihood iteratively using the corresponding complete-data likelihood [9], [22]. As the latter is unobservable, it is replaced by its conditional expectation given the measurements and the current estimate $x^{(t)}$ of vector x to be maximized. Let $x^{(0)}$ be the initial value of x. The iterative process consists in two steps:

1) **E-Step.** Compute $Q(x, x^{(t)}) = E_{x^t}\{logLc(x \mid Z)\}$
2) **M-Step.** Chose x^{t+1} that maximizes $Q(x, x^{(t)})$,

$$Q(x^{(t+1)}, x^{(t)}) \geq Q(x, x^{(t)}), \quad \forall x \tag{6}$$

where $Lc(x \mid Z)$ is the complete-data likelihood using the unobservable variables. Both steps are repeated until the change in likelihood $Lc(x^{(t)} \mid Z) - Lc(x^{t+1} \mid Z)$ is arbitrarily small. The original incomplete likelihood is assured not to decrease after each iteration. Thus, it is only necessary to specify the complete-data likelihood. The variables in the complete state vector are choosen for computational reasons to ease the E- and M-Steps.

2) EM mapping of dynamic environments:
We next show how to apply the EM algorithm to maximize the likelihood term $p(Z_k \mid O_k, x_k, M)$. Let us define for each observation $z_{k,i},\ i \in 1..N_z$ of Z_k a vector of binary variables c_{ij} with $j \in 0..N_o+1$. The possible sources are represented by the index j and accounts for static measurements ($j = 0$), the tracked objects ($j \in 1..N_O$) and unknown sources ($j = N_O + 1$). A variable c_{ij} is set to 1 if observation i was originated by source j and 0 if not. Let denote $c_k = \{c_{ij}\}$ to the set of all binary variables. The complete-data likelihood model is,

$$L_c = p(Z_k, c_k \mid x_k, M_k, O_k) = \prod_{i=1}^{M} \left[p_s(z_{k,i} \mid x_k, M_k)^{c_{i0}} p_u(z_{k,i})^{c_{iN_O+1}} \prod_{j=1}^{N} p_d(z_{k,i} \mid x_k, o_{k,j})^{c_{ij}} \right] \tag{7}$$

where $p_s(.)$, $p_d(.)$ and $p_u(.)$ define the likelihood models for observations originated from the map, each of the tracked moving objects and new discovered areas respectively.

The maximization of Equation 7 includes known and unknown variables and it is performed through the EM algorithm [9], [22]. The complete-data log likelihood function is,

$$\log L_c = \sum_{i=1}^{N_z} \left[c_{i0} \log p_s(z_{k,i} \mid x_k, M_k) + c_{iN_O+1} \log p_u(z_{k,i}) + \sum_{j=1}^{N_O} c_{ij} \log p_d(z_{k,i} \mid x_k, o_{k,j}) \right] \tag{8}$$

Taking the current conditional expectation of the complete-data log likelihood $\log L_c$ we obtain,

$$Q(x_k, x_k^{(t)}) = E_{x_k^{(t)}}\{logL_c \mid Z_k, x_k, M_k, O_k\} \tag{9}$$

As $\log L_c$ can be seen to be linear in the unobservable data c_k, the computation of $Q(x_k, x_k^{(t)})$ on the E-Step at each iteration simply requires to replace each c_{ij} by its conditional expectation given the current measurements Z_k and the current estimates of x_k, M_k and O_k,

$$Q(x_k, x_k^{(t)}) = \sum_{i=1}^{N_z} \left[\hat{c}_{i0} \log p_s(z_{k,i} \mid x_k, M_k) + \hat{c}_{iN_O+1} \log p_u(z_{k,i}) + \sum_{j=1}^{N_O} \hat{c}_{ij} p_d(z_{k,i} \mid x_k, o_{k,j}) \right] \tag{10}$$

where

$$\hat{c}_{i0} = E_{x_k^{(t)}}\{c_{i0} \mid Z_k, x_k, M_k, O_k\}$$
$$\hat{c}_{iN_O+1} = E_{x_k^{(t)}}\{c_{iN_O+1} \mid Z_k, x_k, M_k, O_k\}$$
$$\hat{c}_{ij} = E_{x_k^{(t)}}\{c_{ij} \mid Z_k, x_k, M_k, O_k\}$$

Before going on with the description of the EM algorithm we present next the actual likelihood models used for static, dynamic and unknown observations.

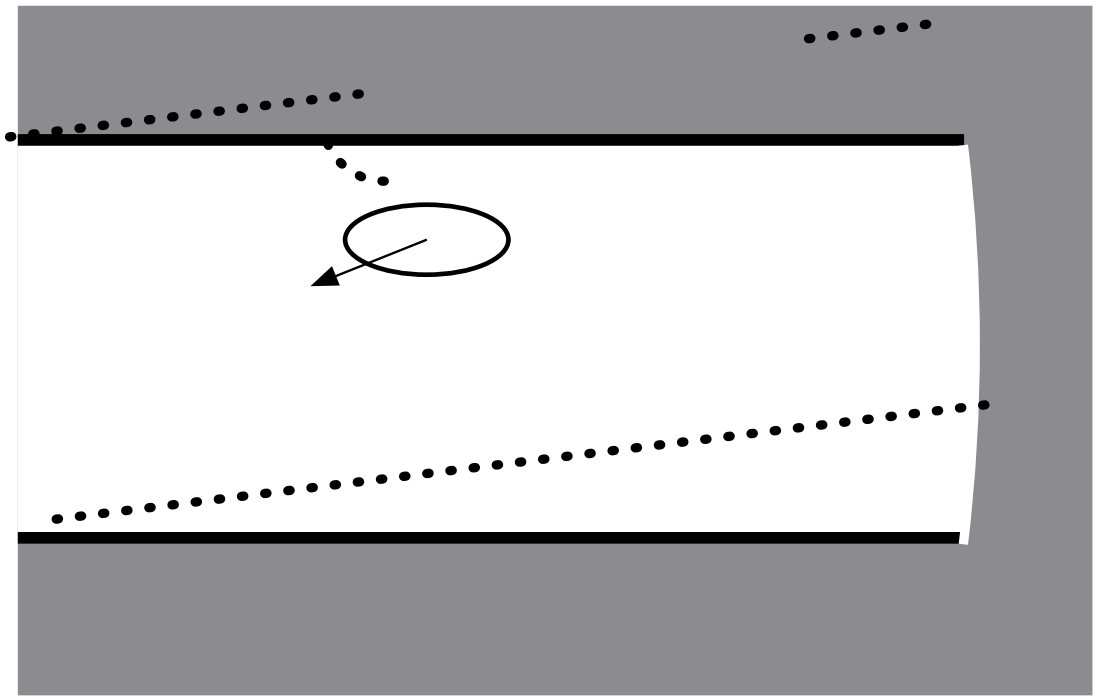


Fig. 3. Errors in the robot position affect the classification of the points.

B. Implementation for range sensors

In this Section we provide the implementation of the previous method for a range sensor , e.g. laser range finder. First, we present the measurement models associated to each type of measurement (static, dynamic, unknown). Based on these models, we derive the equations for the E-Step and M-Step of the EM algorithm. Finally, we present the representations used for both the map and the moving objects and the functions that generate them given the poses, the measurements and the classification provided by the EM algorithm.

1) Models:

The likelihood function of expression 7 represents how well the observation fits our current estimate of the environment. It explicitly reflects the different possible sources for each measurement through the models $p_s(.)$, $p_d(.)$ and$p_u(.)$ and the classification variables c_k.

The specific definition of the measurement models depends on the sensor used. We provide next the models used for a laser range sensor. These models have been widely studied in the literature in the last years specially in the scan matching problem ([17], [40]). There are two additional issues that condition the used model:

- **Robot's position uncertainty**. The error of the estimate of the robot's position has an impact on the classification of the point nature. As the classification process used is usually made based on the current knowledge about the environment, large position errors may produce very bad classifications. As a result the convergence of the method can be slower or even result in wrong pose estimates. Indeed, Figure 3 shows how the effect of the rotation error is bigger for points placed far away from the sensor. Furthermore, the measurements of the top wall are located in an unknown area and those of the bottom one on free space. As the algorithm iterates toward the solution, the classification will improve.
- **Point model**. A complete description of the measurement process must take into account the fact that each laser measurement corresponds to the distance to the first obstacle in the ray direction and, consequently, there must be no obstacles (dynamic or static) between the robot position and the obstacle. For efficiency reasons, it is possible to work with what is called end point models. An end point model ignores the path traversed by the ray and only considers the end point. Note that this is the type of model used in the scan matching techniques presented before. Figure 4 illustrates the differences between both models. In the case of an end point model, both beams have the same probability. On the other hand, a complete model will assign a lower likelihood to the top ray due to the fact that it traverses an obstacle before reaching the final obstacle.

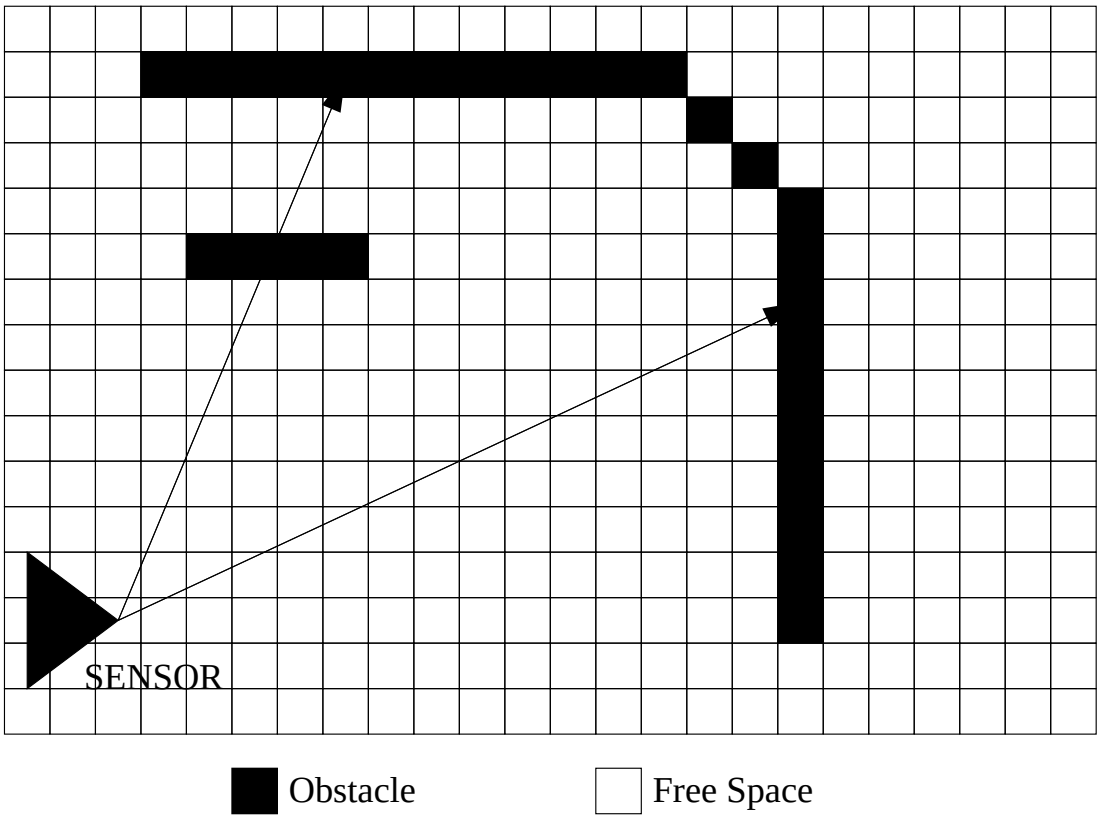


Fig. 4. End point model vs complete model

Our models are correspondence oriented. This means that we need to choose a correspondence for each measurement. For the static case there exist several possible correspondences corresponding to each obstacle included in the map. As in the iterative closest point (ICP) framework, we use the nearest neighbor point to the measurement $z_{k,i}$ as the corresponding point. We use the Mahalanobis distance which implicitly accounts for the uncertainty on the robot pose and the measurement noises ([29], [28]),

$$p_s(z_{k,i} \mid x_k, M_k) \quad = \quad N(p_j; f(x_k, z_{k,i}), Pi0) \tag{11}$$

where p_j is the correspondent point associated to the measurement $z_{k,i}$, $f(x_k, z_{k,i})$ is a transformation between reference systems and $Pi0$ is the covariance matrix computed as in [28],

We use the same model as the likelihood function for each dynamic object,

$$p_d(z_{k,i} \mid x_k, o_{k,j}) \quad = \quad N(o_{k,j}; f(x_k, z_{k,i}), P_{ij}), \forall j \in 1..N_O \tag{12}$$

where the function $f(\cdot, \cdot)$ and the covariance matrix P_{ij} are the same as before but applied to the moving object $o_{k,j}$. Finally, the classification variable c_{iN_0+1} includes spurious observations and those corresponding to unexplored areas and new moving objects. The likelihood of such a measurement is difficult to quantify. It depends on the spurious rate of the sensor and on where the measurement is,

$$p_u(z_{k,i}) = \begin{cases} p_{unexplored} & \text{if } z_{k,i} \notin M_k \\ p_{spurious} & \text{if } z_{k,i} \in M_k \end{cases}$$

The value $p_{spurious}$ is an experimental value measuring the spurious rate of the sensor and $p_{unexplored}$ is typically set to a higher value to avoid those observations placed in unknown areas to influence the optimization process.

2) E-Step:
The E-Step requires the computation of the $\hat{c}_{ij}$ variables,

$$\begin{aligned} \hat{c}_{ij} &= E_{x_k^{(t)}}\{c_{ij} \mid Z_k, x_k, M_k, O_k\} && (13)\\ &= \sum_{c_{ij}} c_{ij} p(c_{ij} \mid Z_k, x_k, M_k, O_k) = p(c_{ij} = 1 \mid Z_k, x_k, M_k, O_k) && (14)\end{aligned}$$

where $i = 1..N_z$ and $j = 0..N_O + 1$ and the expectation is conditioned on the predicted position of the objects O_k, the last map estimate M_k and the current estimate of x_k. Using the Bayes rule, we obtain,

$$\begin{aligned} p(c_{ij} = 1 \mid Z_k, x_k, M_k, O_k) &= \frac{p(z_{k,i} \mid c_{ij} = 1, x_k, M_k, O_k) p(c_{ij} = 1 \mid x_k, M_k, O_k)}{p(z_{k,i} \mid x_k, M_k, O_k)} \\ &= \frac{p(z_{k,i} \mid c_{ij} = 1, x_k, M_k, O_k) p(c_{ij} = 1)}{\sum_l p(z_{k,i} \mid c_{il} = 1, x_k, M_k, O_k)} \end{aligned} \quad (15)$$

The previous derivation assumes a constant value for the prior over the classification variables $p(c_{ij} = 1 \mid x_k, M_k, O_k)$ and computes the probability of the observation as the sum of all its potential sources. The previous equation has been presented in a general way for all the classification variables. Nevertheless, for each classification variable c_{ij} the probability is computed using only the model indicated by the variable c_{ij}.

3) M-Step:
The M-Step computes a new robot pose $x_k^{(t+1)}$ such that

$$Q(x_k^{(t+1)}, x_k^{(t)}) \geq Q(x_k, x_k^{(t)})$$

Given the models introduced in Section III-B the criteria to minimize is,

$$\begin{aligned} Q(x_k, x_k^{(t)}) &= \sum_{i=1}^{N_z} \Bigg[\hat{c}_{i0} \log(\sqrt{2\pi}|P_{i0}| + \hat{c}_{i0} (f(x_k^{(t)}, z_{k,i}) - p_i)^T P_{i0}^{-1} (f(x_k^{(t)}, z_{k,i}) - p_i)) \\ &+ \hat{c}_{iN_O+1} \log p_u(z_{k,i}) + \sum_{j=1}^{N_O} \Big[\hat{c}_{ij} \log(\sqrt{2\pi}|P_{i0} \\ &+ \hat{c}_{ij} (x_k^{(t)}, f(z_{k,i}) - o_{k,j})^T P_{ij}^{-1} (f(x_k^{(t)}, z_{k,i}) - o_{k,j}) \Big] \Bigg] \end{aligned}$$

where grouping all the terms that do not depend on x_k we get,

$$\begin{aligned} Q(x_k, x_k^{(t)}) &= A + \sum_{i=1}^{N_z} \Bigg[\hat{c}_{i0} (f(x_k^{(t)}, z_{k,i}) - p_i)^T P_{i0}^{-1} (f(x_k^{(t)}, z_{k,i}) - p_i)) \\ &+ \sum_{j=1}^{N_O} \hat{c}_{ij} (f(x_k^{(t)}, z_{k,i}) - o_{k,j})^T P_{ij}^{-1} (f(x_k^{(t)}, z_{k,i}) - o_{k,j}) \Bigg] \end{aligned}$$

which can be solved using a weighted least squares method.

Please note that the moving objects also influence the computation of x_k. Intuitively, only the static parts should be taken into account to improve the current robot pose estimate. In practice the location of the moving objects is much more uncertain than the location of the static objects. This minimizes the effect of the moving objects. However, one could also remove the terms corresponding to the moving objects during the minimization. On the other hand, if the motion model of the moving objects is accurately know, this information can be used in the optimization to compute the pose of the robot.

4) Measurement classification:
After convergence, the EM algorithm provides us with the maximum likelihood pose for the robot according to the current map and moving objects. In addition to this, it also provide us with the correspondence probability of each point of the laser scan. The next step is to compute the new map estimate and moving objects for the next step.

We use the weights to distinguish between static, dynamic and unknown measurements. Except in those situations where there exist ambiguities, once the robot position is corrected all the weight is assigned to a single source. A simple threshold on the different correspondences allow us to classify the measurements,

$$z_i \in \begin{cases} Z^{static}, & \text{if } c_{i0} > \alpha \\ Z^{dynamic}, & \text{if } \sum_{j=1}^{N_z} c_{ij} > \alpha \\ Z^{unknown}, & \text{otherwise} \end{cases} \tag{16}$$

where the value of the threshold $\alpha > 0.5$ to ensure that a measurment only belongs to one of the sets Z_{static}, Z_{moving} and $Z_{unknown}$. The threshold allows to measure how conservative we want to be in the classification procedure. The higher the value of α, the most clear must be the classification into static, dynamic or unknown. The main idea is to avoid introducing within the map or the filters those observations may belong to several sets. In other words, we rather not use a measurement if it is not clear its source.

5) Map estimate:
We now address the implementation of the function $f_M(\cdot)$. We use a two dimensional grid [30] to represent the static parts of the environment. The workspace is divided in cells. Each cell has associated a binary variable m_i whose value is one if its occupied and zero if its free space. This representation has some important advantages within our navigation context. The first one is that this type of maps explicitly contain information about the free space areas required during the E-Step described previously. The second one is that they allow to represent any shape obstacles and, consequently, are well suited for unstructured environments. As a result, this type of representations fits well with the planning and obstacle avoidance algorithms implemented in our sensor-based navigation system.

There exist several methods to compute the probability of the grid cells [13], [32], [40], [17]. We use a Bayesian approach proposed in [11]. The grid is updated with each scan laser. The update process is different for each type of observation (static/dynamic/unkonwn). Intuitively, all the measurements of a scan contribute to update the free space traversed by their corresponding laser beams (see Figure 4). On the other hand, only those measurements classify as static provide information about the static parts. The detailed update function is described in [28].

6) Tracking of moving objects:
The function $f_O(\cdot)$ is implemented using an independent extended Kalman filter to track each of the moving objects. Thus, the function can also be computed recursively at each step using the a posteriori estimate of the previous step, the measurements classified as dynamic $Z_k^{dynamic}$ and the path computed by the EM algorithm.

Our current implementation uses a constant velocity model to predict the positions of the moving objects. So as to solve the data association problem between the moving objects and the dynamic measurements, we could use the final weights provided by the EM algorithm. However, this strategy is prone to loose track of the moving objects in the presence of ambiguities [28]. Therefore, we rather use a Joint Probabilistic Data Association [2] scheme to increase the robustness of the tracking. This is important since the information of these filters is used by the navigation system to avoid collisions with the moving objects.

IV. Using the Model Information for Tactical Planning and Obstacle Avoidance

The model builder presented in the previous section provides a map of the static parts of the environment and a set of filters tracking the moving objects. In this section we describe how this information is used by the rest of the modules (Figure 1). Further details about the architecture are described in [25].

The role of the tactical planner is to determine at each cycle the main cruise to direct the vehicle. A cruise depends on the permanent structure of the scenario (e.g. walls and doors) and not on the dynamic objects that move around (e.g. people). Thus, we use the static map as model for the planner. Figure 5 shows an example. The planner computes the cruise (main direction of the path) that points toward the end of the corridor. This is because the dynamic obstacles do not affect the planner. Notice that for a system that does not model the dynamic parts, *Door* 2 and the corridor would appear closed. If no other passages are available, the planner would fail since it would not find any path to the goal. Furthermore, in the example, if one of the dynamic obstacles stops for a

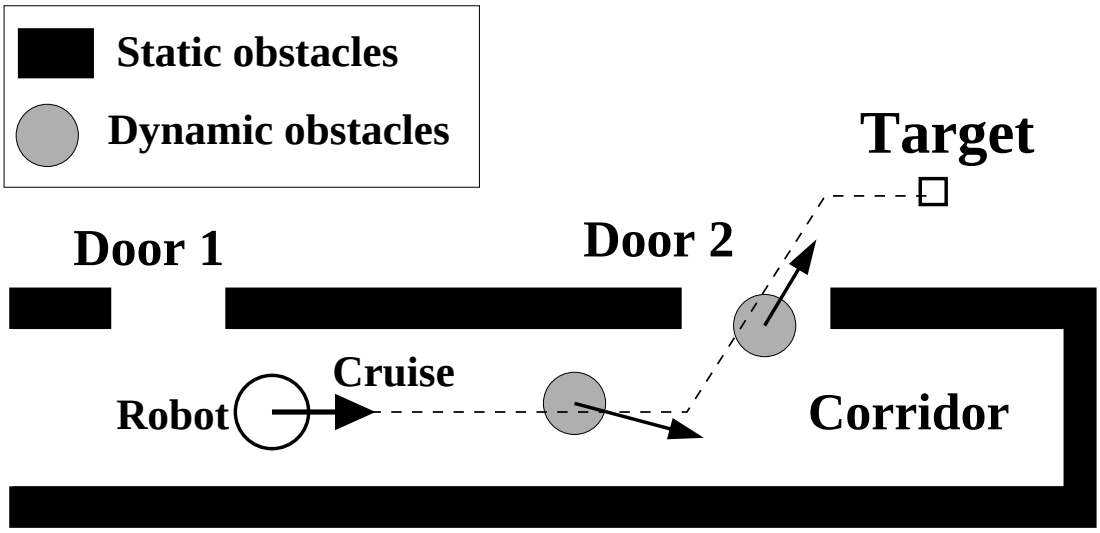


Fig. 5. Moving obstacles are not taken into account to compute the path.

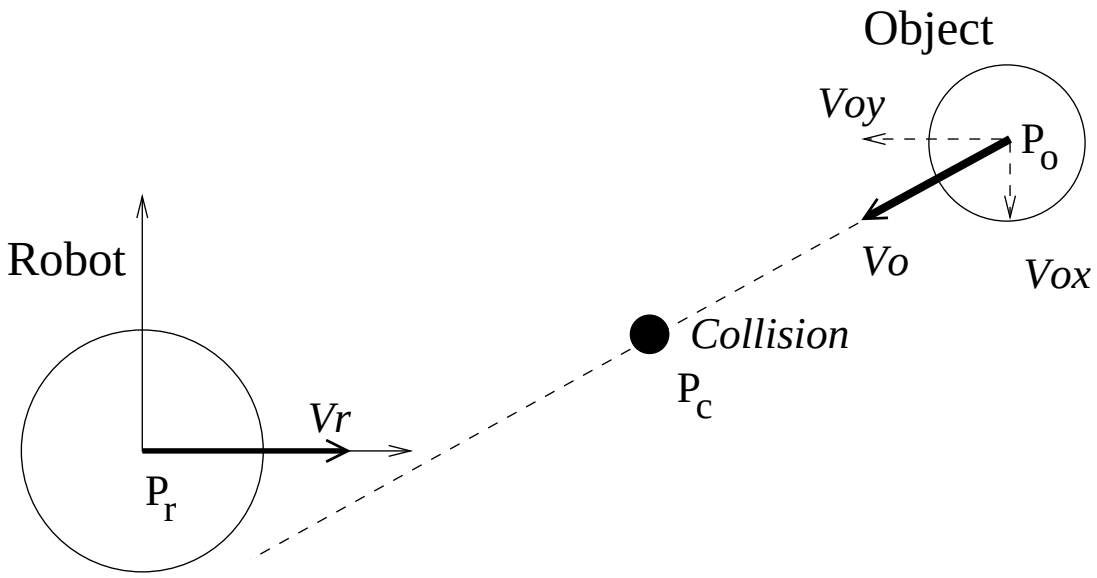


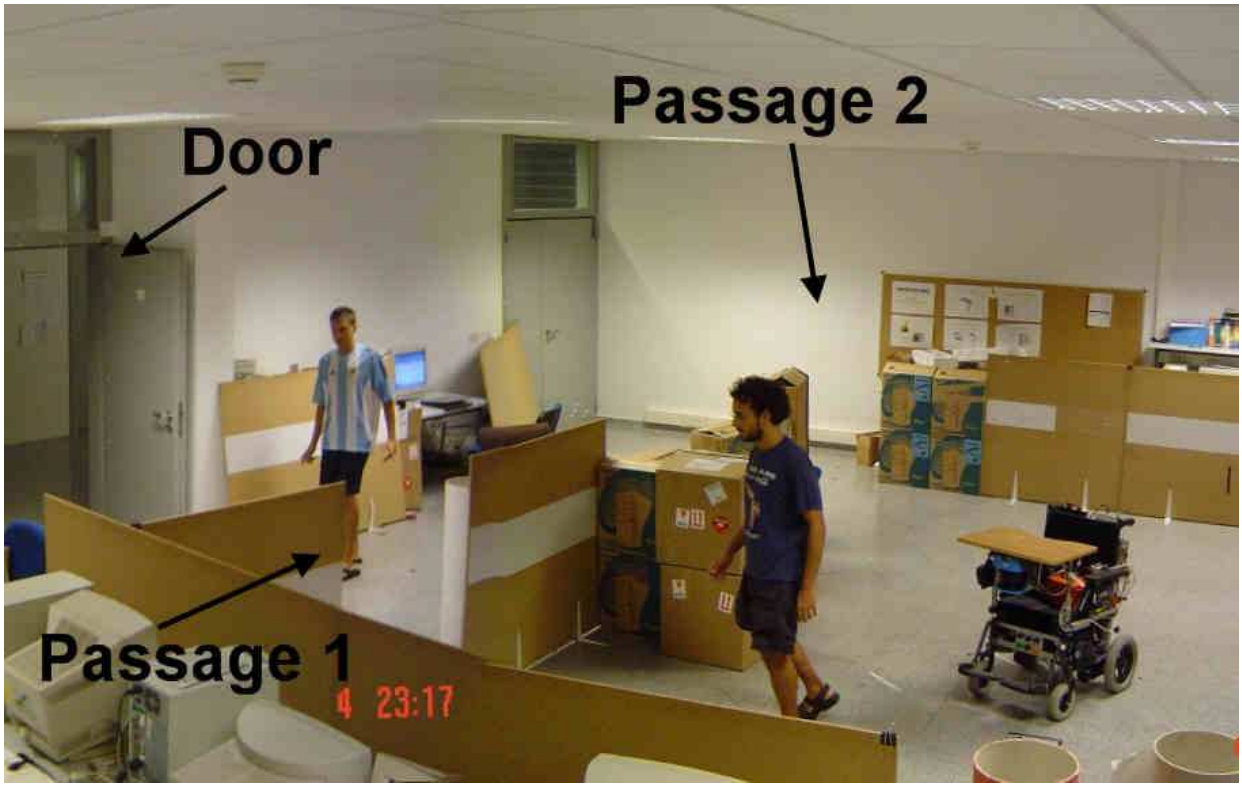
Fig. 6. Computation of the new object location for obstacle avoidance.

given time, the modeling module will include it in the information passed to the planner. Thus, it will be taken into account by the planner and the new cruise would direct the vehicle toward *Door* 1 (avoiding the trap situation).

The obstacle avoidance module generates the collision-free motion to align the vehicle toward the cruise (computed by the planner). Here we use the map of static obstacles, since all the obstacles included in the map must be avoided. So as to consider the dynamic objects, several obstacle avoidance methods have been proposed [12], [33] Here, we use an approximation to the full obstacle avoidance problem with dynamic obstacles which is described next. For each dynamic obstacle, we compute the time t_c at which the collision occurs in the direction of motion of the robot using the current vehicle velocity $v_r = (v_{r_x},\ v_{r_y})$ and obstacle velocities $v_o = (v_{o_x},\ v_{o_y})$ respectively). We assume that both the object and the robot move with constant lineal velocities. The collision time is,

$$t_c = \frac{p_{o_x} - (R_r + R_o)}{v_{r_x} - v_{o_x}} \tag{17}$$

where R_r is the radius of the robot, R_o the radius of the object and p_o is the current location of the obstacle. Then, we place the obstacle at the location p_c that will be in time t_c (Figure 6). The new location of the obstacle p_c is:



(a)

(b)

(c)

Fig. 7. (a) Snapshot of the Experiment 1 and (b) two snapshots of Experiment 2 in a classroom like scenario and in a long corridor.

$$p_c = (p_{o_x} + v_{o_x} t_c, p_{o_y} + v_{o_y}.t_c) \tag{18}$$

In fact, we want to avoid that the robot moves toward p_c since the collision will occur there. Let us remark that the new location of the obstacle depends on the current obstacle location but also on both the vehicle and obstacle relative velocities. Furthermore, if the obstacle moves further away from the robot ($t_c < 0$), it is not taken into account. This approach to avoid the moving obstacle implies: (i) if there is a potential collision, the obstacle avoidance method starts the avoidance maneuver before than if the obstacle was considered static; and (ii) if there is no potential collision, the obstacle is not taken into account. This strategy heavily relies on the information provided by the modeling module. Since our model assumes a constant lineal velocity model, the predicted collision could not be correct when this assumption does not hold. However, this effect is mitigated since the system works at a high rate rapidly reacting to the moving obstacle velocity changes.

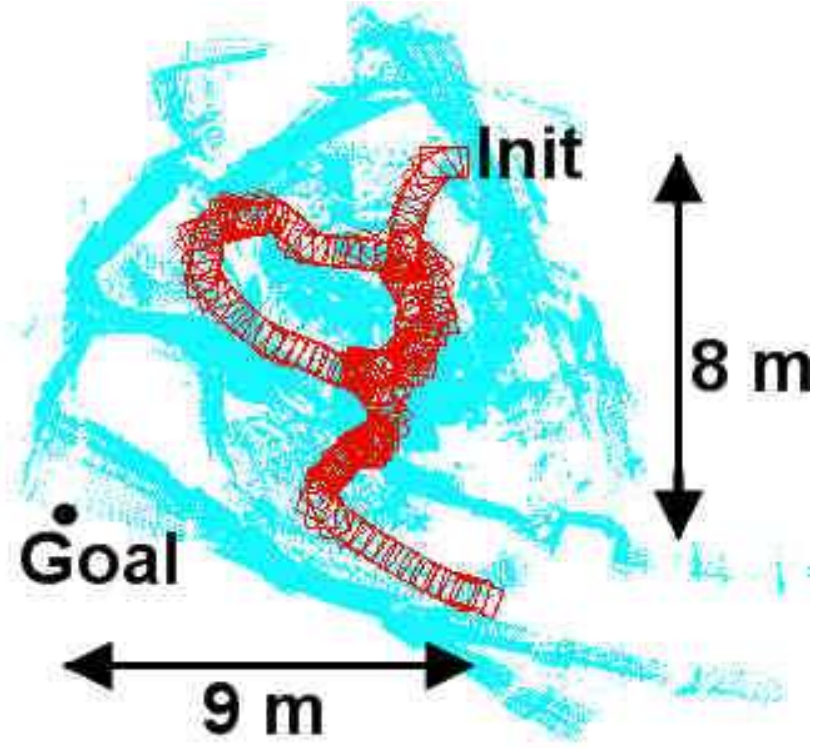


Fig. 8. Real laser data and trajectory using the raw non corrected odometry.

In summary, we have seen how the performance of the tactical planning and obstacle avoidance method are greatly improved by selectively using the static and dynamic information. Next, we describe the experimental results.

V. Experimental results

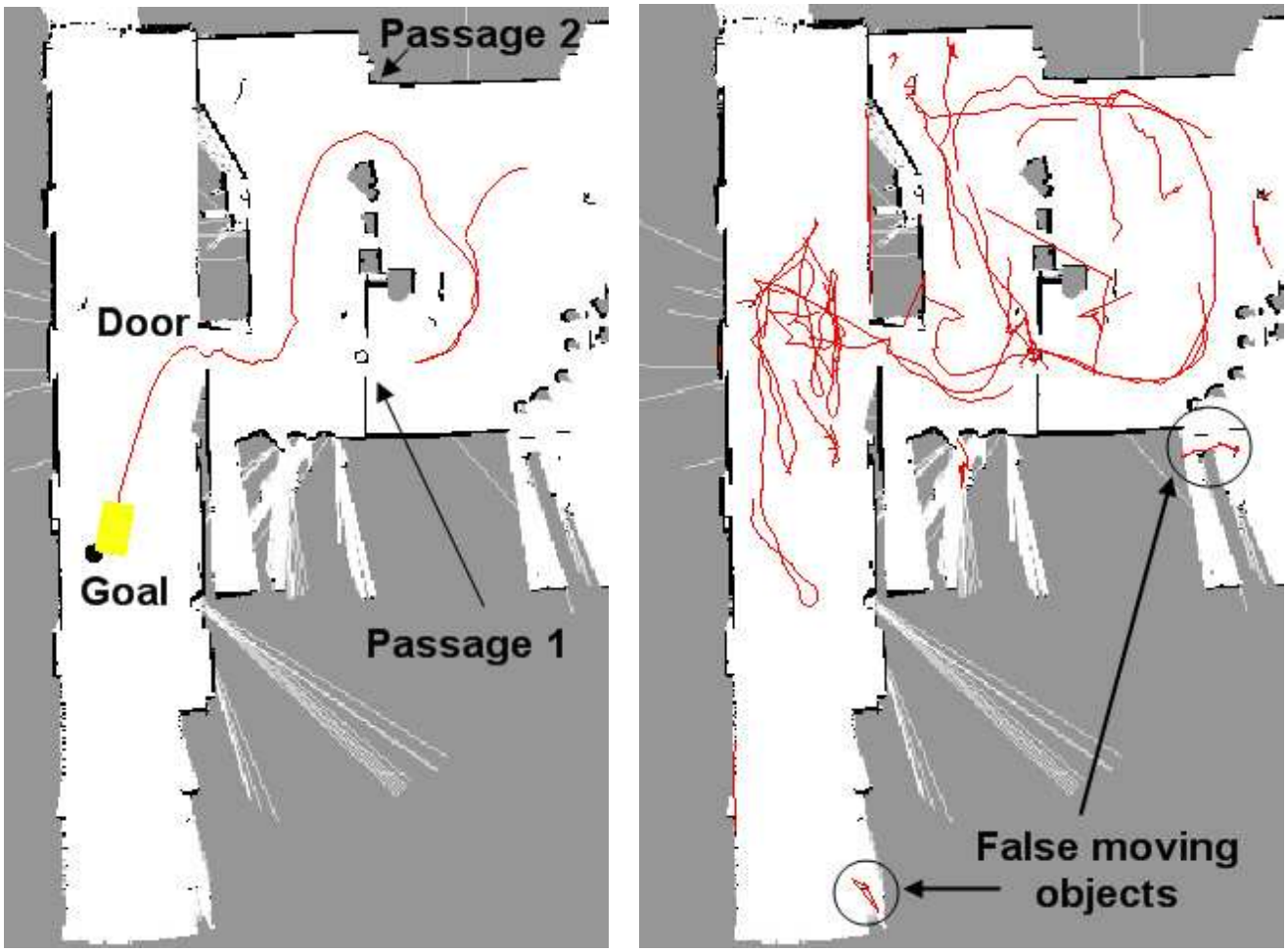
We present in this section the experimental results obtained with real robots. The validation consists in two different parts. In the first one, we examine the performance of the modeling algorithm presented in Section III-B. Then, we evaluate the integration within the navigation system using the information of moving objects.

For experimentation, we used a commercial wheelchair that we have equipped with two on-board computers and a SICK laser. The vehicle is rectangular $(1.2 \times 0.7 meters)$ with two tractor wheels that work in differential-driven mode. We set the maximum operational velocities to $(v_{max}, w_{max}) = (0.3\frac{m}{sec}, 0.7\frac{rd}{sec})$ due to the application context (human transportation). All the modules work synchronously on the on-board $PentiumIII850Mhz$ at the frequency of the laser 5Hz.

The wheelchair is a non holonomic robot with a rectangular geometry (it cannot move in any direction it sweeps an ample area when turns). The laser sensor is placed in the front part of the robot (0.72m) and has a 180° field of view. Thus, sometimes some of the static obstacles to avoid are out of the field of view and dynamic obstacles appear and disappear constantly. Furthermore, the ground is polished and the vehicle slides constantly with an adverse effect on the odometry. On the other hand, the scenario is unknown and not prepared to move a wheelchair and there are many places where there is little room to maneuver. In addition, people turn the scenario in a dynamic and unpredictable place and sometimes modify the structure of the environment creating global trap situations.

We analyze next two of the tests we performed to validate the system. The first one is a more controlled experiment that will be used to validate both the modeling algorithm and its integration in the navigation system (see Figure 7(a)). This allowed to create the appropriate navigation situations such as global trap situations. The second one is a more realistic test where a cognitive handicapped child drove the vehicle using a voice interface during rush hour in the Computer Science department of the University of Zaragoza (see Figure 7(b-c)).

We analyze next the results obtained by the EM mapping algorithm and then evaluate the performance of the whole architecture.



(a) (b)

Fig. 9. (a) The map built during the experiment and the vehicle trajectory. The map shows the occupancy probability of each cell. White corresponds to a probability of zero (free cell) and black to a probability of one (occupied cell). (b) The trajectories of the detected moving objects.

A. Modeling module results

The laser data collected during the Experiment 1 and the vehicle trajectory provided by the odometry are illustrated in Figure 8. We used a $20m \times 20m$ grid map with a $5cm$ resolution cell. The map is centered around the current robot location, always contains the goal location and its resolution its enough for planning and obstacle avoidance. Figure 9(a) depicts the map of the static parts and the vehicle trajectory computed by the module. The map also includes some obstacles that stopped after moving for a certain period. The model was used for tactical planning purposes and as short-time memory for obstacle avoidance.

The trajectories of the people tracked during the experiment are shown in Figure 9(b). Most of them correspond to the people walking in the free space of the office. All the moving objects were detected by the modeling module. There were also some false positives due to misclassifications. They occurred mainly in two situations: when the laser beams were almost parallel to the reflected surface or when the beams missed an object because its height was similar to the height of the laser scan (around 70cm). Figure 9(b) depicts two of them which were included in the map when the system realized they were static.

Finally, Figure 10(a) illustrates a moment of the experiment where the robot was tracking two moving objects. The rectangles represent the estimated location of the moving objects being tracked containing the observations associated to each of them. The estimated velocity vector is represented by a red straight line. Another two snapshots are shown in the evaluation of the navigation system (Figures 14(a) and 15(a)).

An important characteristic of the proposed algorithm is to provide a fast solution to the local modeling problem so as to have enough time to execute the other modules of the architecture. Figures 10(b-c) show the number of

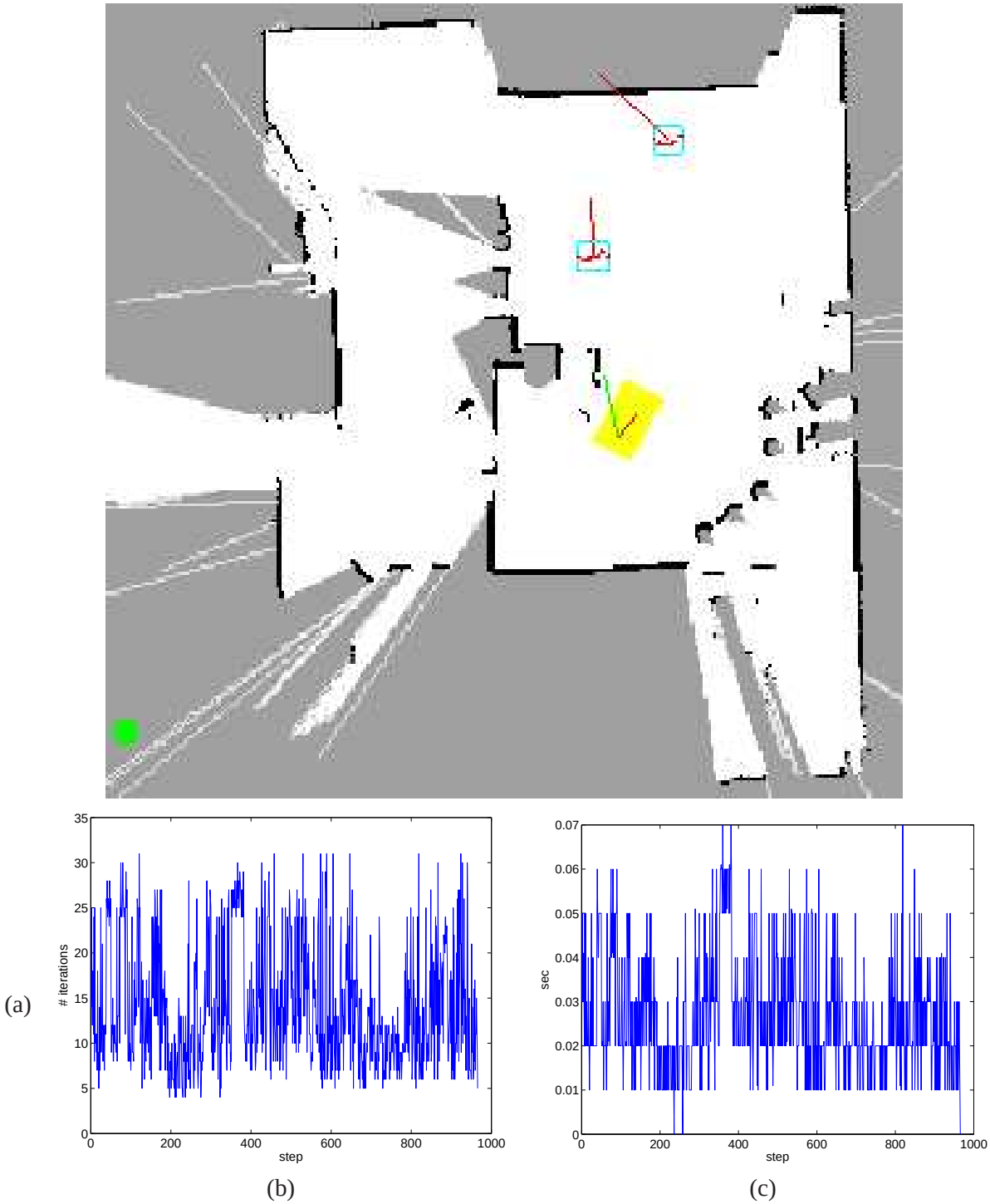


Fig. 10. (a) Snapshot of the map of static objects and filters tracking the moving object at a given time. (b) Shows the number of iterations until convergence for each step of Experiment 1 and (c) the corresponding computation time.

iterations and the computation time at each step. Despite the clock resolution of the computer (10 milliseconds), the figures reflect the that the cost per iteration is constant. It depends on the number of points of the scan, the grid resolution (number of static points) and the number of moving objects. The mean values for the whole process are 14.1 iterations and 0.021 seconds. The time spent in the update of the map and in the prediction and update of the filters is negligible and its below the clock resolution.

We next analyze the modeling module results obtained in Experiment 2. Figure 11 shows the raw laser data and the vehicle trajectory according to the odometry readings. The wheelchair was driven by the child using a voice

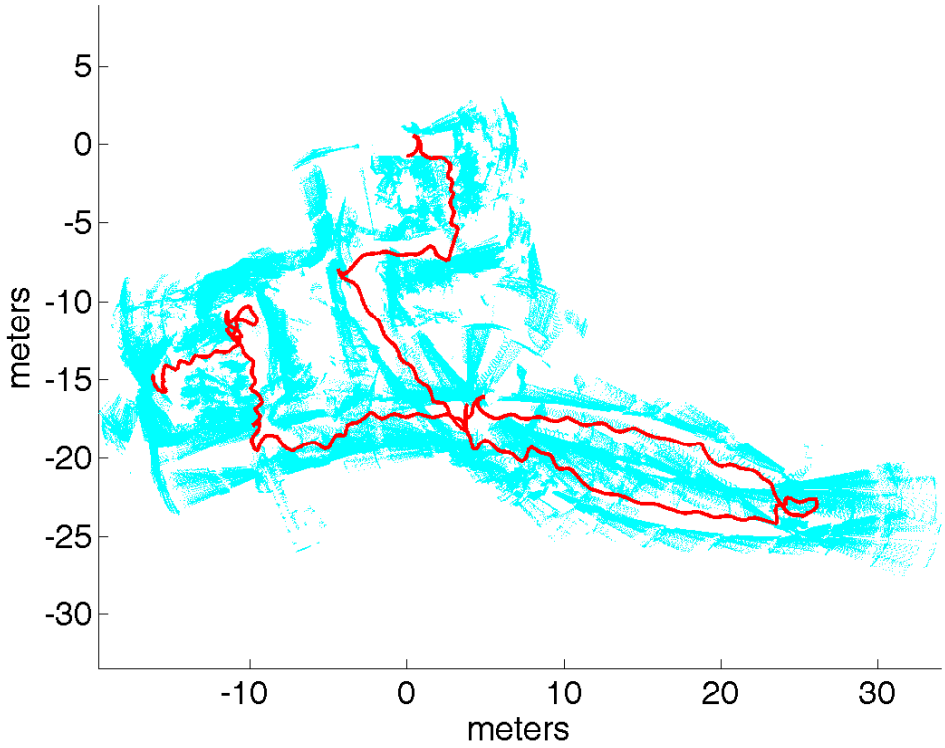


Fig. 11. Raw laser data and trajectory using the raw non corrected odometry.

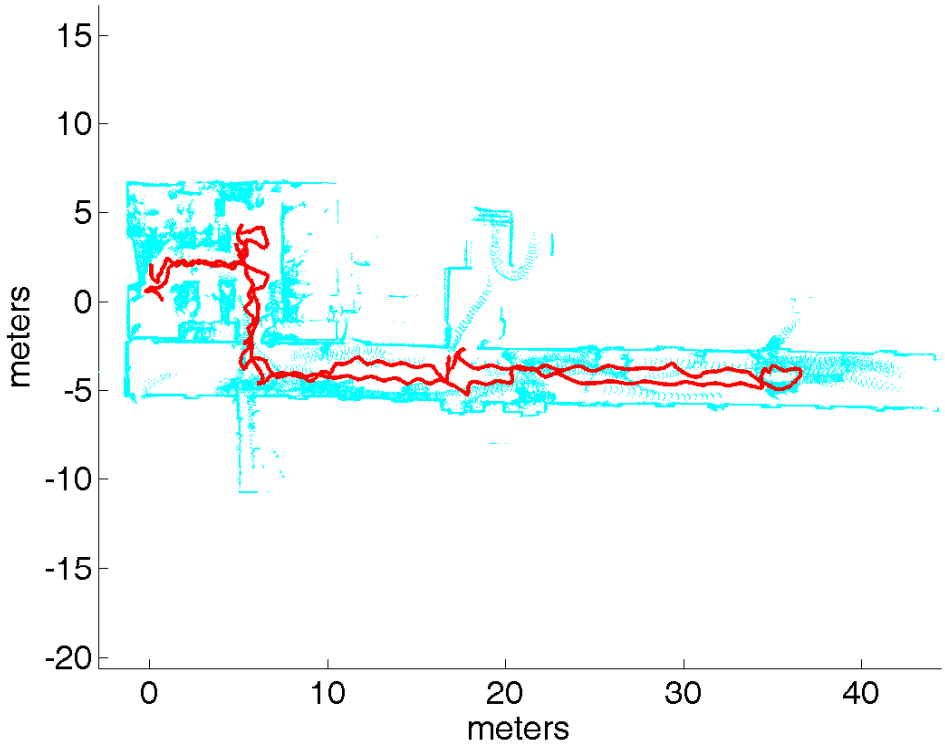
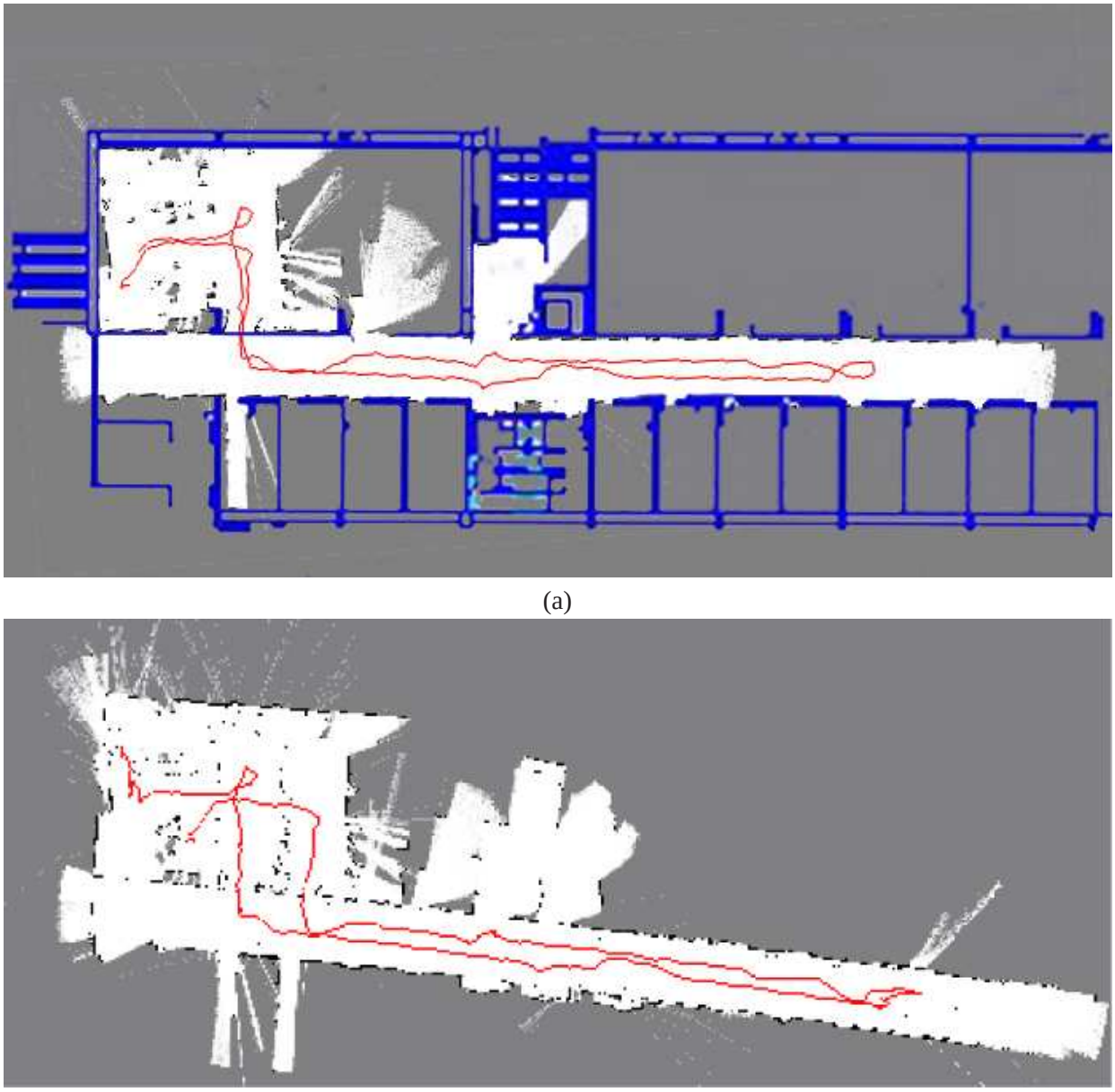


Fig. 12. Raw laser data and trajectory using the maximum likelihood estimated poses. The figure shows also the measurements corresponding to the dynamic objects.

interface to place the final goal within the map. He first drove the vehicle out of a class like scenario (Figure 7(b)). Next, he followed the corridor (Figure 7(c)) and then came back to the start location. The wheelchair autonomously computed the path to reach the goals placed by the child and negotiated all the obstacles. From the modeling



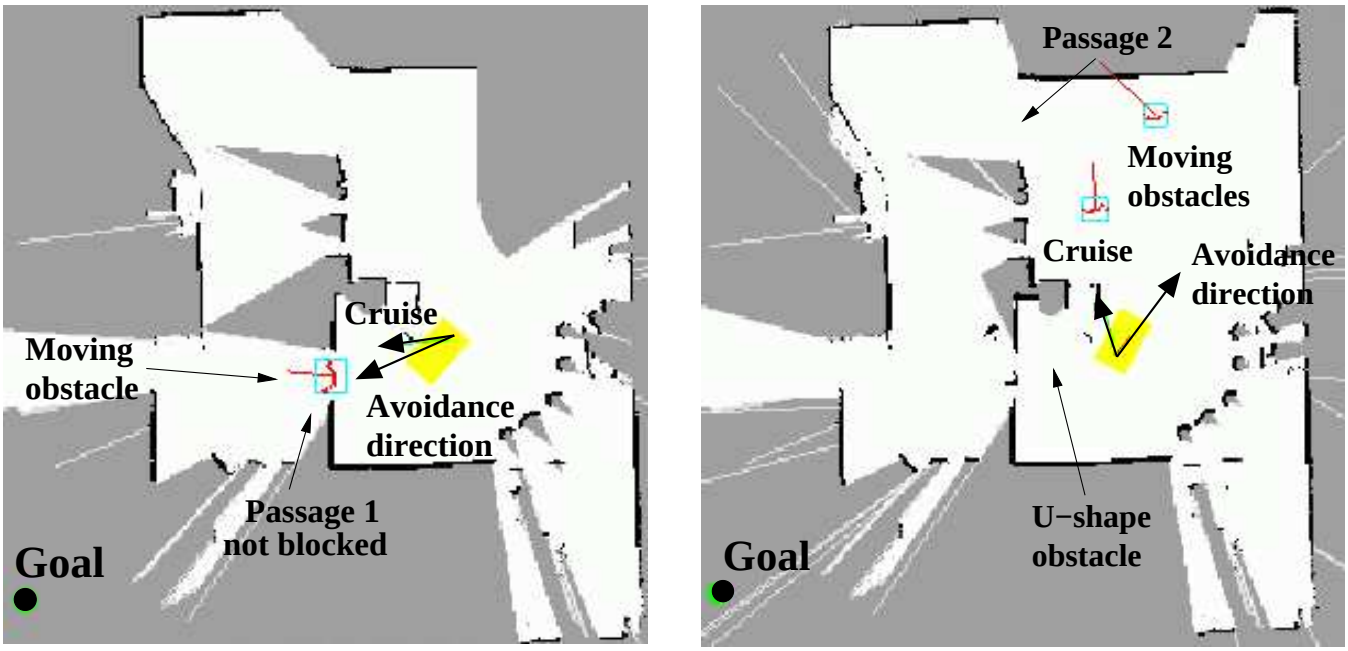
(a)

(b)

Fig. 13. (a) Map obtained using the modeling algorithm of Section III-B. The blue structure corresponds to the blue print of the building. (b) Map obtained without taking into account the moving objects. Note how the dynamic objects corrupt the displacement estimation along the corridor.

module point of view, the experiment has two different parts. First, the class like scenario is unstructured due to the presence of chairs, tables and other people. Then, the wheelchair traveled along a long corridor. The static structure of the environment contains very few information about the robot displacement in the direction of the corridor. This makes difficult to correct the robot displacement, specially in the presence of moving moving people. Actually, it is mandatory filter the moving objects to obtain a correct estimation of the robot's displacement. If not, the robot while use the observations of the moving object to correct the position along the corridor resulting in wrong poses.

Figure 12 shows the raw data using the robot poses computed by the modeling module. The figure includes the observations corresponding to the moving objects. Note how there were constantly moving people around the wheelchair. So as to test the precision of the method, we increased the size of the grid map ($80m \times 80m$) to represent the whole covered area. Despite the presence of moving objects, the modeling module was able to detect them and, consequently, they did not affect the vehicle pose. The obtained map is shown in Figure 13(a). Figure 13(b) shows the same map without taking into account the moving objects (i.e. considering all the measurements as



(a) (b)

Fig. 14. (a) A moving obstacle placed in the area of passag and (b) robot avoiding a trap situation. The figures shows the tracked moving objects (rectangles), the dynamic observations associated to them and the estimated velocities. The two arrows on the vehicle show the cruise computed by the planner module and the direction of motion computed by the obstacle avoidance one.

a static). The observations of the moving objects affect the resulting map not only in the motion along the corridor, but in certain cases also in the other robot coordinates.

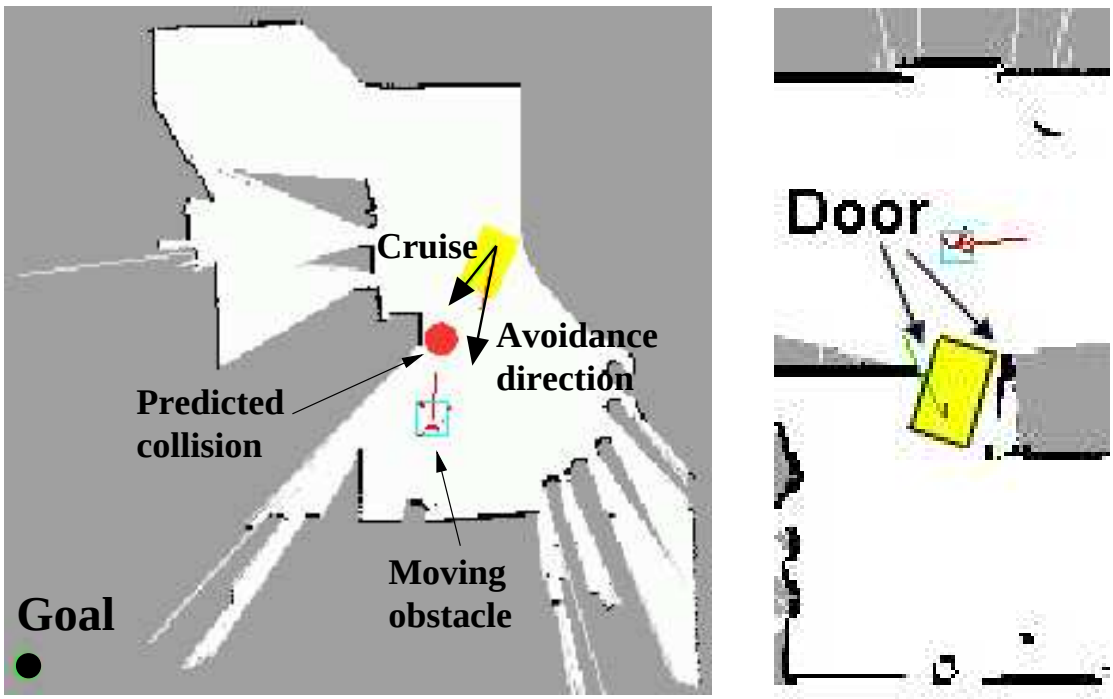
B. Sensor based navigation system performance

So as to evaluate the performance of the whole navigation system and the selective use of the static and dynamic information, we used the Experiment 1 (Figure 7(a)).

From a motion point of view, the objective of the experiment was to drive the vehicle to a location that was out of the office (Figure 7(a)). Initially, the vehicle proceeded toward the *Passage* 1 avoiding collisions with the people that move around. Then, we blocked the passage creating a global trap situation that was detected by the system (Figure 9(a)). The vehicle moved backwards through *Passage* 2 and then traversed the *Door* exiting the room and reaching the goal location without collisions. The time of the experiment was 200sec and the distance traveled around 18m. As mentioned before, the wheelchair is an squared non-holonomic robot with the sensor located in the front part. Figure 8 shows the trajectory followed by the vehicle according to the odometry readings. Although it is possible to navigate using this information, the resulting model is in general not good enough, since it is not possible to reach the goal with an acceptable precision. Furthermore, due to the location of the sensor on the robot, it is necessary to use the map so as to avoid the obstacles behind the laser. In narrow passages the odometry error can accumulate blocking the passage even if there is enough space to maneuver. The modeling module provides an improved laser based odometry and allows to build models

We next describe several issues regarding the navigation performance. The improvement of the individual module behavior due to the selective usage of the static and dynamic information, and the improvement of the whole behavior of the system due to the integration architecture.

The planner computed at any moment the tactical information needed to guide the vehicle out of the trap situations (the cruise) but only using the static information. The most representative situations happened in the *Passage* 1. While the vehicle was heading this passage, people was crossing it. However, since the humans were tracked and labeled dynamic they were not used by the planner and thus the cruise points toward this passage (Figure 14(a)).



(a) (b)

Fig. 15. (a) Moving obstacle going toward the robot. The figure shows the tracked moving objects (rectangles), the dynamic observations associated to them and the estimated velocities. The two arrows on the vehicle show the cruise computed by the planner module and the direction of motion computed by the obstacle avoidance one. (b) Detail of the robot maneuver to cross the *Door*.

Then, the vehicle aligned with this direction. Notice that systems that do not model the dynamic obstacle would consider the human static and the vehicle trapped within a U-shape obstacle.

Next, while the vehicle was about reaching the passage, a human placed an obstacle in the way. The vehicle was trapped in a large U-shape obstacle (Figure 14(b)). After a given period, the modeling module realized that the object was not moving anymore and included it in the information provided to the planner. Immediately, the planner computed a cruise that pointed toward the *Passage* 2. The vehicle was driven toward this passage avoiding the trap situation.

The obstacle avoidance module computed the motion taking into account the geometric, kinematic and dynamic constraints of the vehicle [23]. The method used the static information included in the map, but also the predicted locations of the objects computed using the obstacle velocities. Figure 15(a) depicts an object moving toward the robot, and how the predicted collision creates an avoidance maneuver before than it would be obtained if the obstacle is consider static (case of the navigation systems that do not model the dynamic). Furthermore, obstacles that move further away from the robot are not considered. In Figure 14(b) the two dynamic obstacles were not included in the avoidance step (systems that do not model the dynamic would consider them).

The performance of obstacle avoidance module was determinant in some circumstances, specially when the vehicle was driven among very narrow zones. For example, when it crossed the door (Figure 15(b)), there were less than 10cm at both sides of the robot. In addition, during the passage, the obstacle avoidance method computed motion between very near obstacles, and this movement was free of oscillations, and at the same time was directed toward zones with great density of obstacles or far away form the final position. That is, the method achieves robust navigation in difficult and realistic scenarios avoiding the technical limitations of many other existing techniques.

In summary, the modeling module is able to model the static and dynamic parts of the environment. The selective use of this information allows the planning and obstacle avoidance modules to avoid the undesirable situations that arise from false trap situations and improve the obstacle avoidance task. Furthermore, the integration within the architecture allows to fully exploit the advantages of an hybrid sensor-based navigation system that performs in difficult scenarios avoiding typical problems as the trap situations.

VI. Discussion and conclusions

In this paper we have addressed two issues of great relevance in sensor-based navigation: how to model the static and dynamic parts of the scenario and how to make use of this information within a real sensor-based navigation system.

Our contribution in the modeling aspect is to incorporate the information about the moving objects within a maximum likelihood formulation of the scan matching process. In this way, the nature of the observation is included in the estimation process. The result is an improved classification of the observations that increases the robustness of the algorithm improving the resulting model. Most of previous works [37], [42], [43], [16] assume a known classification within the optimization process. This means that the classification is done prior to the estimation of the vehicle location. They focus on the reliable tracking of the moving objects or on the construction of accurate maps.

The algorithm proposed in [17] iteratively improves the classification via an EM algorithm. This is a batch technique that focus on the detection of spurious measurements to improve the quality of the map. The approach presented in [26], [27] applies learning techniques but do not improve the vehicle location and do not use probabilistic techniques to track the moving objects.

In any case, all the methods assume a hard classification between static and dynamic objects. This is clearly a simplification of the real world since there are objects that can act as static or dynamic; for instance, doors, chairs, tables, cars,... Although there exist some preliminary work on the estimation of the state of some of this objects [39], it is our belief that having a prior on the behavior of the objects will greatly simplify the problem. This can be done using other type of sensors as cameras or 3D range sensors.

The second issue is the integration of the modeling module in a real system taking advantage of the dynamic and static information. The sensor-based navigation uses selectively this information in the planning and obstacle avoidance modules. As a result, many problems of existing techniques (that only address static information) are avoided without sacrificing the advantages of the full hybrid sensor-based navigation schemes. The experimental results confirm that the system is able to drive the vehicle in difficult, unknown and dynamic scenarios.

References

[1] K. Arras, J. Persson, N. Tomatis, and R. Siegwart. Real-time Obstacle Avoidance for Polygonal Robots with a Reduced Dynamic Window. In *IEEE Int. Conf. on Robotics and Automation*, pages 3050–3055, Washington, USA, 2002.

[2] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Mathematics in Science and Engineering. Academic Press., 1988.

[3] Y. Bar-Shalom, XR Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. J. Wiley and Sons, 2001.

[4] O. Bengtsson and A-J. Baerveldt. Localization in changing environments by matching laser range scans. In *EURobot*, pages 169–176, 1999.

[5] S. Blackman and R. Popoli. *Design and analysis of modern tracking systems*. Artech House, Norwood, MA, 1999.

[6] O. Brock and O. Khatib. High-Speed Navigation Using the Global Dynamic Window Approach. In *IEEE Int. Conf. on Robotics and Automation*, pages 341–346, Detroit, MI, 1999.

[7] J. A. Castellanos and J. D. Tardós. *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers, Boston, 1999.

[8] P. Cheeseman and P.Smith. On the representation and estimation of spatial uncertainty. *International Journal of Robotics*, 5:56–68, 1986.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. 34:1–38, 1977.

[10] A. Doucet, S.J. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

[11] A. Elfes. Occupancy grids: A probabilistic framework for robot perception. *PhD thesis*, 1989.

[12] P. Fiorini and Z. Shiller. Motion planning in dynamic environments using velocity obstacles. *Int. Journal of Robotic Research*, 17(7):760–772, 1998.

[13] F. Gambino, G. Ulivi, and M. Vendittelli. The transferable belief model in ultrasonic map building. In *In Proceedings of the Sixth IEEE International Conference on Fuzzy Systems*, 1997.

[14] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Conference on Intelligent Robots and Applications (CIRA)*, Monterey, CA, 1999.

[15] D. Hähnel. Mapping with mobile robots. *PhD thesis, University of Freiburg*, 2004.

[16] D. Hähnel, D. Schulz, and W. Burgard. Map building with mobile robots in populated environments. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[17] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[18] B. Jensen and R. Siegwart. Scan alignment with probabilistic distance metric. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[19] H. Koyasu, J. Miura, and Y. Shirai. Recognizing moving obstacles for robot navigation using real-time omnidirectional stereo vision. *Int. Journal of Robotics and Mechatronics*, 14:to appear, 2002.
[20] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.
[21] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Intelligent and Robotic Systems*, 18:249–275, 1997.
[22] G. McLachlan and T. Krishnan. *The EM algorithm and extensions*. Wiley series in probability and statistics. J. Wiley and Sons, 1997.
[23] J. Minguez and L. Montano. Robot navigation in very dense and cluttered indoor/outdoor environments. In *Int.Federation of Automatic Control IFAC 15th World Congress*, Barcelona, Spain, 2002.
[24] J. Minguez and L. Montano. Nearness diagram navigation (nd): Collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, 20(1), 2004.
[25] J. Minguez, L. Montesano, and L. Montano. An architecture for sensor-based navigation in realistic dynamic and troublesome scenarios. In *IEEE Int. Conf. on Intelligent Robot and Systems*, Sendai, Japan, 2004.
[26] J. Modayil and B. Kuipers. Bootstrap learning for object discovery. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-04)*, Sendai, Japan, 2004.
[27] J. Modayil and B. Kuipers. Towards bootstrap learning for object discovery. In *AAAI-2004 Workshop on Anchoring Symbols to Sensor Data*, 2004.
[28] L. Montesano. Detection and tracking of moving objects from a mobile platform. application to navigation and multi-robot localization. *PhD thesis, Universidad de Zaragoza*, 2006.
[29] L. Montesano, J. Minguez, and L. Montano. Probabilistic scan matching for motion estimation in unstructured environments. In *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.
[30] H. P. Moravec and A. Elfes. High Resolution Maps from Wide Angle Sonar. In *IEEE Int. Conf. on Robotics and Automation*, pages 116–121, March 1985.
[31] K. Murphy. Dynamic bayesian networks: Representation, inference and learning. *PhD thesis, UC Berkeley, Computer Science Division*, 2002.
[32] G. Oriolo, G. Ulivi, , and M. Vendittelli. Fuzzy maps: A new tool for mobile robot perception and planning. *Journal of Robotic Systems*, 14(3):179–197, 1997.
[33] S. Petti and Th. Fraichard. Safe motion planning in dynamic environments. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Edmonton, AB (CA), August 2005.
[34] R. Philipsen and R. Siegwart. Smooth and efficient obstacle avoidance for a tour guide robot. In *IEEE Int. Conf. on Robotics and Automation*, Taipei, Taiwan, 2003.
[35] E. Prassler, J. Scholz, and P. Fiorini. Navigating a robotic wheelchair in a railway station during rush hour. *International Journal of Robotics Research*, 18:711–727, 1999.
[36] D. Schulz, W. Burgard, D. Fox, and A. Cremers. Tracking Multiple Moving Targets with a Mobile Robot using Particle Filters and Statistical Data Association. In *IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, 2001.
[37] D. Schulz, W. Burgard, D. Fox, and A.B. Cremers. People tracking with a mobile robot using sample-based joint probabilistic data association filters. *International Journal of Robotics Research (IJRR)*, 22(2):99–116, 2003.
[38] C. Stachniss and W. Burgard. An Integrated Approach to Goal-directed Obstacle Avoidance under Dynamic Constraints for Dynamic Environments. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 508–513, Switzerland, 2002.
[39] C. Stachniss and W. Burgard. Mobile robot mapping and localization in non-static environments. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, Pittsburgh, PA, USA, 2005.
[40] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT press, 2005.
[41] I. Ulrich and J. Borenstein. VFH*: Local Obstacle Avoidance with Look-Ahead Verification. In *IEEE Int. Conf. on Robotics and Automation*, pages 2505–2511, San Francisco, USA, 2000.
[42] C. Wang and C. Thorpe. Simultaneous localization and mapping with detection and tracking of moving objects. In *IEEE Int. Conf. on Robotics and Automation*, Washington, USA, 2002.
[43] C.-C. Wang, C. Thorpe, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.