To appear in *Advanced Robotics*
Vol. 00, No. 00, January 2013, 1–24

Full paper

Motion Strategies for Exploration and Map-Building under Uncertainty with Multiple Heterogeneous Robots

Luis Valentin†, Rafael Murrieta-Cid†, Lourdes Muñoz-Gómez‡,
Rigoberto López-Padilla†, Moises Alencastre-Miranda‡
† *Centro de Investigación en Matemáticas CIMAT, México*
{*luismvc,murrieta,rigolpz*}*@cimat.mx*
‡ *Tecnológico de Monterrey Campus Santa Fé, México*
{*lmunoz,malencastre*}*@itesm.mx*

(*Received 00 Month 201X; accepted 00 Month 201X*)

In this paper[2], we present a multi-robot exploration strategy for map-building. We consider an indoor structured environment and a team of robots with different sensing and motion capabilities. We combine geometric and probabilistic reasoning to propose a solution to our problem. We formalize the proposed solution using Stochastic Dynamic Programming (SDP) in states with imperfect information. Our modeling can be considered as a Partially-Observable Markov Decision Process (POMDP), which is optimized using SDP. We apply the dynamic programming technique in a reduced search space that allows us to incrementally explore the environment. We propose realistic sensor models and provide a method to compute the probability of the next observation given the current state of the team of robots based on a Bayesian approach. We also propose a probabilistic motion model, which allows us to take into account errors (noise) on the velocities applied to each robot. This modeling also allows us to simulate imperfect robot motions, and to estimate the probability of reaching the next state given the current state. We have implemented all our algorithms and simulations results are presented.

keywords: Exploration, Uncertainty, Map-building, Multiple robots.

1. Introduction

Automatic environment exploration and map building are important problems in mobile robotics. Autonomous robots must possess the ability to explore their environments, build representations of those environments (maps), and then use those representations to navigate effectively. Maps built upon exploration can be used later by the robot to perform other tasks such as object finding [2].

A strategy for exploring an unknown environment and building its representation with a mobile robot can be performed as follows: (i) the robot builds a local map with the sensor readings (ii) a global map is updated merging the information between the current global map and the new local map (iii) the robot moves to an intermediary goal, which is defined based on suitable properties.

In the last two decades several approaches have been proposed for map-building, for instance [3], [4], [5] and [6] just to name a few. Most previous research has focused on developing techniques to extract relevant information from raw data and to integrate the collected data into a single model. However, a robot motion strategy to explore the environment has been less studied. In this work, we deal mainly with this latter problem. In this paper an exploration strategy is proposed. Our exploration strategy considers sensing and motions capabilities of each robot. Our algorithm outputs the sensing configurations to be visited and the trajectories to reach them. The sensing configurations are associated to the borders between known and unknown space.

[2]A preliminary version of some portions of this work is reported in [1].

Our method assigns a robot of the team for visiting a selected sensing configuration, according to its capabilities and without considering predefined roles.

The remainder of this paper is organized as follows. In Section 2, we briefly describe related works about exploration strategies, we also present the main contributions of our work with respect to previous work from other authors, and with respect to our previous work. Then in Section 3, we present the problem definition and the main elements of our modeling. In Section 4, we present the proposed observation model, which is a classifier. In Section 5, we describe the probabilistic motion model used in this work. In Section 6, a solution to the problem addressed in this work, is proposed using dynamic programming. In Section 7, we present our simulation results. Finally, in Section 8, we present the conclusions and future work.

2. Related Work

Several robot's exploration strategies for map building have been proposed. It is possible to classify those exploration strategies into two main types: (i) systematic exploration and (ii) strategies in which sensed information is taken into account to define the next sensing location. In systematic explorations (exploration type (i)), the robots follow a predefined motion pattern, for instance following walls, moving in concentric circles [7], and so forth. In non-systematic exploration (type (ii)), information taken by the sensor is frequently used to select an appropriate sensing location. Some exploration strategies of type (ii) use frontier-based exploration, originally proposed by Yamauchi in [8]. In frontier-based exploration, the robot goes to the imaginary line that divides the known and unknown parts of the environment. In [9–11], the proposed exploration strategies lead the robot to locations in which maximal information gain is expected; a utility function is defined to maximize the new information that will be obtained in the next sensing location. Several works have proposed to generate random sensing locations for exploration (e.g. [6, 12]). The work reported in [12] presents sensor-based exploration techniques. The generated sensing locations are evaluated to select one that maximizes a utility function [6]. The exploration strategy presented in [6] is based on the computation of the next best view and the use of randomized motion planning. In [6] the proposed motion strategy only considered the case of a single robot.

In [13], an interesting experimental evaluation of several exploration strategies is presented. However, in that work the case of multi-robot exploration is not studied. In [14], the authors have presented a mapping system that uses an information-based exploration strategy and that allows a mobile robot to efficiently build the map of an unknown environment. In [15], a theoretically-grounded approach is proposed. This approach is based on Multi-Criteria Decision Making (MCDM). A result in [15] is that making more informed local decisions results in a better global performance. The authors apply the method in search and rescue tasks and they evaluated its performance in simulated environments. In [16], a comparison between different techniques for exploration and mapping is presented. The techniques were compared in simulations using different criteria as exploration time or map quality.

Some works have proposed multi-robot exploration and mapping [17, 18]. In both of these works, the information gain and the exploration cost are considered simultaneously to define the next locations for each robot of the team. In [18], the map is represented using an occupancy grid and the possible locations for the next exploration step are defined over cells lying on the border between the known and unknown space. In [19], the authors propose a multi-robot exploration strategy in which instead of frontiers, the authors use a segmentation of the environment to determine exploration targets for the individual robots. This segmentation improves the distribution of the robots over the environment.

We have already proposed motion strategies for exploration and mapping. In [20], we have presented techniques that allow one or multiple mobile robots to efficiently explore and model their environment. We developed a utility function that measures the quality of proposed sensing configurations, and a randomized algorithm for selecting the next sensing configuration.

In this work, we use some of the techniques proposed in [20]. We use a frontier-based exploration; the frontiers between the known and unknown parts of the environment are found based on visibility computations. We use sampling to generate candidate sensing configurations for the robots. Visibility computation is also used to bias the sampling generation. The samples are generated near to the border between the explored an unexplored environment. Finally, the Hausdorff distance is used to find the best alignment between the global map and the new local one. Nevertheless, we propose new techniques that have not been presented in [20]. We model the exploration problem, as a *resource allocation task*. We consider *heterogeneous robots* with different motion and sensing capabilities. We use dynamic programming as a tool *to assign a robot* (which is the resource) to explore a part of the environment (which is the task), according to the robot's capabilities. We propose a probabilistic observation model, which is a classifier, and we use a probabilistic motion model to move the robots from a state to another (more details about our contributions are given in Section 2.1).

In [21], the authors propose a method for multi-robot exploration based on Decentralized Markov Decision Process (Dec-MDP). Some important differences between the work in [21] and the approach proposed in the paper are the following. Our work has as an advantage compared with the work in [21], that in this work we consider heterogeneous robots, while in [21], the authors consider homogeneous robots. Besides, in [21], the robots' states are observable, while in our work the robots' states are partially observable. However, the work in [21] has the advantage over this work that the approach is decentralized which might allow to handle a larger number of robots.

Some previous works have already proposed exploration with multiple robots having different capabilities. The work presented in [22] considers a team composed by robots of different sizes. During the exploration, if a robot is too big to navigate among obstacles and reach a sensing location, then it asks to a smaller robot to perform the task.

In [23], a formal study of multi-robot task allocation (MRTA) is presented. In that work an interesting domain independent taxonomy of MRTA problems is provided. However, in [23] is pointed out that the problem of assigning target locations to a team of robots for exploring an unknown environment, *one of the goals of this paper*, is not captured by the proposed taxonomy. A difficulty in the problem of assigning robots to target locations is that the cost for a robot to target C depends on whether that robot first visit target A or target B.

In [24], a survey and analysis of market-based multi-robot coordination is presented. In that work is stated that market-based techniques are proving to be versatile and powerful coordination schemes for multi-robot systems. However, in [24] is also indicated that in domains where centralized approaches are feasible, market-based approaches can be more complex to implement and produce poorer solutions. In this work we propose several strategies to make feasible a centralized approach (see Subsection 2.1 and Section 6). However, market based approaches might be implemented in decentralized schemes, which will scale better for a larger number of robot compared with a centralized approach.

In [25], robots have one of two roles: navigator or cartographer, navigators randomly move in the environment until they find a target location for a cartographer, after that, the cartographer moves to the target location. In the works mentioned above each robot follows a specific and predefined role according to its type. In this work, we propose motion strategies for exploration and map building considering the robots' capabilities, but we do not assign pre-defined roles to the robots of the team, instead, we model the robots' capabilities probabilistically.

2.1 *Main contributions and originality of the approach*

Our modeling can be considered as a Partially-Observable Markov Decision Process (POMDP). The team of robots does not have directly access to the state, but it is possible to get some information about it through an observation model $z_k = h(x_k)$. Furthermore, the next state of the system ($x_{k+1} \in \mathcal{X}$) (and consequently the next observation $z_{k+1} = h(x_{k+1})$) will only depend on the current state, it does not matter how the current state was obtained (Markov assumption).

It is well known, that finding optimal solution for POMDPs (for instance using dynamic programming as in our approach) has very high computational cost. Indeed, it has been shown that the problem of finding an exact optimal solution is intractable [26]. Consider 4 robots on a 2-D environment divided in cells with a grid. Assuming that every robot can only reach its 8 neighboring cells, for optimizing one step-ahead greedy optimal policy, 2^{12} possible controls must be evaluated. Furthermore, while the problem is already expensive for one step-ahead evaluation, when long term planning is done, an exponential branching of possibilities must be considered due to the exponential growing number of possible sequences of observations [27].

In order to deal with those problems and propose a practicable approach we do the following:

1) For the exploration task, we test one-step ahead greedy optimization and compare it versus long term plans. In other words, for a given discrete time or step k, we compare the result of planning an order for visiting *all* the known frontiers between the know and unknown environment (long term plan), against the result of generating a plan, in which one robot is allowed to move only once (short term plan) (see Section 7). We compare the results of these two planning strategies, in terms of the traveled distance by the robots and the number of sensing locations needed to explore the whole environment. 2) The sub-goals that the robots may reach are generated using sampling. Several previous works have already proposed a sampling based approach for POMDPs [28–30] to obtain anytime solutions. However, notice that in our approach the generation of sub-goals is biased using visibility computations. This approach improves existing POMDPs sampling methods, in which the samples are generated without a suitable bias. 3) Our probabilistic observation model $p(z|x)$ is a classifier, the Bayes' rule is used to estimate this probability. In our modeling, the sensing of each robot is probabilistically dividing both the observation and state spaces (see Fig. 1). An observation corresponds to a class, and there is a relation between a type of local map (class) and a part of the state space, which is also split. Note that, one observation might map to more than one state, two states x and x' might have equivalent observations $h(x) = h(x')$. Thus, two different geometric states might be indistinguishable for the sensor. To see this, a very simple example is shown in Fig. 1. That figure shows a corner in a polygonal map, there are three regions in the map $R1$, $R2$ and $R3$. Robot configurations q_1 in region $R1$ and q_3 in region $R3$ sense two sets of points, which are classified as a wall, while robot configuration q_2 inside region $R2$ senses a set of points, which is classified as a corner. The main idea of this work is to transform a geometric state x (typically, the state depends on a robot configuration q and the local map e where the robot lies) to a label or type tp, and to associate a probability to each label. Analogously, an observation z (which is a sensor reading, e.g. points obtained with a laser range-finder) is also transformed to a symbol (a class or type) and a probability is also associated to each class. This allows us to choose the most appropriate robot according to its sensing and motion capabilities to explore that part of state space. This equivalent relation also reduces both the number of samples to generate and the sensibility of the solution respect to the sampling process. Note that a type of local map will not change significantly in a neighborhood of a sampled configuration. This third point represents the main originality and contribution of the proposed approach.

A preliminary version of portions of this work appeared in [1]. The main distinguishing features of our current work compared with our previous research in [1] are: 1) We include a probabilistic robot motion model, see Section 5. 2) We test one-step ahead greedy optimization and compare it versus long term plans, see Section 7. 3) We include simulation results in larger environments with a bigger number of robots, see Section 7. 4) We present a clearer and more detailed description of the proposed approach.

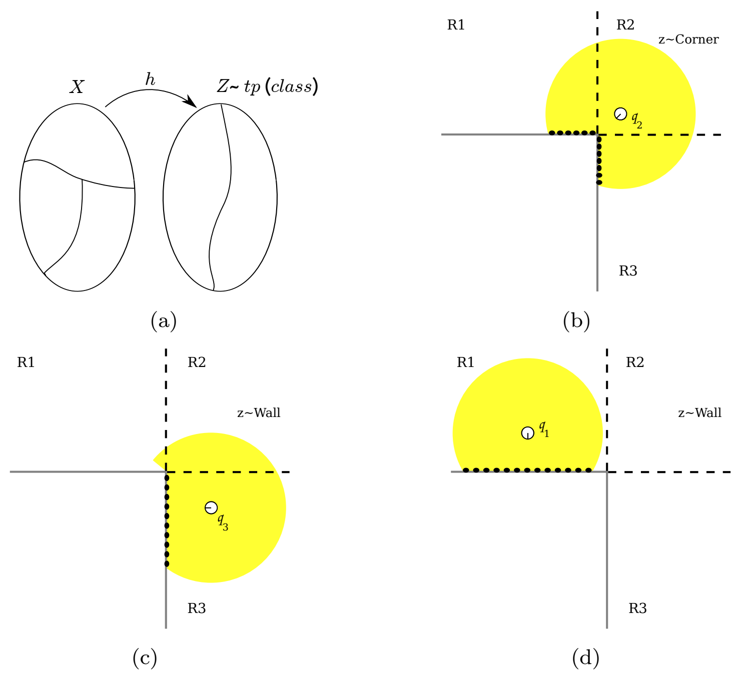


Figure 1. (a) Equivalent relation between observation and state space, (b), (c) and (d) a corner in a polygonal environment, the light gray (yellow) region represents the area sensed, and the dots over the obstacles indicates laser reading.

3. Problem definition and system elements

The problem addressed in this paper consists in exploring an unknown environment and incrementally building a model of that environment, with a team of mobile robots having different sensing and motion capabilities. We assume that the sensors and motors of each robot are imprecise. Our goal is to propose a robot motion strategy, which generates a fast and reliable map building. We also want to profit the most of the different capabilities of each robot according to the exploration and map building tasks.

We assume that the environment is unknown, in the sense that robots have not sensed it previously, however, it is known that the environment corresponds to an indoor structured space. We also assume that our robots are equipped with sensors having *different range, resolution, and different precision to determine the distance from the robot to the obstacles.* Robots also have different motion capabilities, in the sense that some of them have more precise motors compared with other robots.

3.1 *System Elements*

Since we assume *uncertainty* in both sensing and motion, we model our problem as a dynamic system with imperfect information state; the dynamic system evolves between discrete times k and $k+1$. Let $\mathcal{C}$, $\mathcal{U}$ and $\mathcal{Z}$ respectively denote the configuration, control and observation spaces for the robot. The geometrical state space is $\mathcal{X} \subset \mathcal{C} \times E$, E is the set of all possible environments in which the robots might be. A robot configuration is denoted $q_i \in \mathcal{C}$, a robot state $x_i \in \mathcal{X}$, a robot observation $z_i \in \mathcal{Z}$, a robot control $u_i \in \mathcal{U}$ and i denotes the i-th robot. Our dynamic system is defined as follows:

Robots: We have a system with n robots. All the robots are modeled as discs of the same radius, but they have different sensing and motion capabilities. We consider robots that can rotate in place (e.g. differential drive robots).

Observations: $z_i \in \mathcal{Z}$ denotes the i-th robot's observation. Each robot is equipped with a laser range-finder, the lasers mounted on each robot might be different (e.g. they might have different resolution, range, or precision of measurements), more details are given in Section 7. Points belonging to the obstacles (discrete sensor reading) are obtained from the laser sensor. Every point is denoted $s_j \in \mathbb{R}^2$. The set of points sensed from configuration q_i is denoted $S(q_i) = \bigcup_j s_j$. From this set of points, a class or type denoted tp is associated to each robot observation z_i (more details are given in Section 4).

States: Each robot state is denoted by x_i, and it is defined by $q_i \times e_i$, q_i denotes the i-th robot's configuration and $e_i \in E$ denotes the local map in which the robot i lies. Based on the set of points $S(q_i)$, a local map e_i is modeled with poly-lines applying a line fitting technique [20]. Thus, a local map is a list of segments. For n robots a state is defined by $x = q_1 \times e_1 \times \cdots \times q_i \times e_i \times \cdots \times q_n \times e_n$, in which q_1, q_i and q_n respectively denote the configuration of robot 1, robot i and robot n. We assume mobile robots with 3 degrees of freedom, so the configuration of each robot i is a vector $q_i = (p_x, p_y, \theta)^T$, p_x and p_y denote the robot's position in a two dimensions global reference frame, θ denotes the robot orientation with respect to the abscissa axis in the global reference frame. Visibility computation over the local map e_i yields a visibility polygon $V(x_i)$ associated to robot i at state x_i; $V(x_i)$ is used to define the borders between the known and unknown environment and send robots to explore the unknown environment.

Controls: The set of possible controls is denoted $\mathcal{U}$, each element of the set $u_i \in \mathcal{U}$ is a vector having the controls for each robot. Each robot is controlled using an angular velocity w_i and a linear velocity v_i. Thus, at the *execution level* $u_i = [v_i, w_i]^T$. We model these controls as random variables. This modeling allows us to take into account errors (noise) on the velocities applied to each robot. This modeling also allows us to simulate imperfect robots' motions. For exploration at the *planning level*, a control for a given robot i has the form $u_{(i,k)} = x_{(i,k)} \rightarrow \text{free-edge}_j$, where $x_{(i,k)}$ represents the current state of robot i and free-edge$_j$ a frontier or free edge j between known and unknown space. U_k denotes a set having the controls of all robots. The set having the controls for the complete team of robots has the form $U_k = \{x_{(i,k)} \rightarrow \text{free-edge}_j, x_{(i,k)} \rightarrow x_{(i,k+1)} = x_{(i,k)}\}$, under the restriction that two robots cannot visit the same free edge, but one or more robots at a given time k are allowed to remain motionless, hence the control $x_{(i,k)} \rightarrow x_{(i,k+1)} = x_{(i,k)}$ is allowed. For more detail see Section 5.

Observation Model: We use a probabilistic observation model, $p(z_{k+1}|u_k, x_{k+1})$ is the probability of obtaining a type of observation z_{k+1} given that the system is at state x_{k+1}, having applied control u_k, recall that k denotes a discrete time. We assume that the observation z_{k+1} is independent of the control u_k. It means that robots sense the environment, when they are motionless, after reaching a given sensing configuration. Hence, $p(z_{k+1}|x_{k+1}, u_k) = p(z_{k+1}|x_{k+1})$. Furthermore, we also assume these probabilities independent for each robot, thus:

$$p(z_{k+1}|x_{k+1}) = p(z_{(1,k+1)}|x_{(1,k+1)})p(z_{(2,k+1)}|x_{(2,k+1)}) \dots p(z_{(n,k+1)}|x_{(n,k+1)})$$

In Section 4 we propose a Bayesian approach to estimate $p(z_{k+1}|x_{k+1})$.

Motion Model: The system changes from a state x_k in time k to other state x_{k+1} after having applied control u_k with probability $p(x_{k+1}|x_k, u_k)$. Since there are n robots, we have a probability $p(x_{(i,k+1)}|x_{(i,k)}, u_{(i,k)})$, in which i denotes the i-th robot. We assume this probability independent for each robot. Therefore, we have:

$$p(x_{k+1}|x_k, u_k) = p(x_{(1,k+1)}|x_{(1,k)}, u_{(1,k)}) \dots p(x_{(n-1,k+1)}|x_{(n-1,k)}, u_{(n-1,k)})p(x_{(n,k+1)}|x_{(n,k)}, u_{(n,k)})$$

We estimate these probabilities using the probabilistic motion model presented in Section 5.

In our motion model, the robots move following only two motion primitives: Rotation in place without translation and straight line motions.

Dynamic programming for exploration: Dynamic programming is used to assign robots to the next sensing configuration for exploring a frontier. This assignation uses the information vector I_k [31]. The information vector is defined as $I_k = (u_1, \ldots, u_{k-1}, z_1, \ldots, z_k)$; I_k is the history of all observations until time k and all controls that have been applied to the system until time $k-1$. More details are given in Section 6.

4. Probabilistic observation model $p(z_{k+1}|x_{k+1})$

Inspired by Bayesian approaches, we propose to use *a priori* information of the environment, which can be incorporated to our models to make better decisions. Our observation model $z_{k+1} = h(x_{k+1})$ is a classifier. Indeed, we define the mapping $z_{k+1} = h(x_{k+1})$ between the observation z_{k+1} and the system state x_{k+1} as an equivalent relation, $z_{k+1} \sim tp$; $tp \in \mathbb{N}$. Our observation model has the following form: $h : x = q \times (T : S(q) \to e^{tp}) \to z \sim tp$, transformation T estimates a type (class) of local map e^{tp} by virtue of the nearest neighbor method using the partial Hausdorff distance as metric, and transformation h estimates the type (class) of observation tp using the Bayes rule. The Hausdorff distance is also used to find the best alignment between the current sensor reading and the new one. We used original data, to align local maps with the global one.

We want to estimate the probability of the next observation z_{k+1}, given that the robots are in the state x_{k+1}. We assume this probability independent of the controls u_k. The Bayes rule is used to estimate $p(z_{k+1}|x_{k+1})$. The robots are provided with a training set of observations. In this work, 5 types or classes are used $\Omega = \{wall, corridor, corner, type\ T, crossroad\}$ (see Fig. 2). In practice, this a priori information might be obtained from blueprints or databases. Each type itself has associated a probability of being sensed $p(z_{k+1})$. In this work, this probability is set equal for all types. The maximum a posteriori probability (MAP) is the one assigned to the observation z_{k+1} given the state x_{k+1}.

$$p^{\max}(z_{k+1}|x_{k+1}) = \max_{tp \in \Omega} \left\{ \frac{p(x_{k+1}|z_{k+1} = tp)p(z_{k+1} = tp)}{\sum_{z_{k+1}} p(x_{k+1}|z_{k+1} = tp)p(z_{k+1} = tp)} \right\} \tag{1}$$

$p(x_{k+1}|z_{k+1})$ is estimated using the nearest neighbor equation $p(x_{k+1}|z_{k+1}) = \frac{\zeta}{n_{tp}\phi_{tp}}$ [32], which is discussed in detail in the next section.

4.1 *Nearest Neighbor Method*

The *k-nearest neighbor* method is a classifier [32]. ζ is the number of elements inside ϕ_{tp}, n_{tp} is the number of elements used in a training set of type tp, and ϕ_{tp} is the size of the surface that assigns a class. In general for a n dimensions features space, and assuming that the features are normalized, the surface that assigns a class is a hyper-sphere. For a two dimensions features space, that surface is a circle. For a training set in a feature space with a single characteristic, ϕ_{tp} is defined by: $\phi_{tp} = 2r_{tp}$, where r_{tp} is the distance to the k-nearest neighbor.

Since we need to measure how similar are two sets of points, we use the partial Hausdorff distance as metric r_{tp}, in the nearest neighbor method to estimate the resemblance between the type of observation tp (related to the observation z_{k+1}) and a hypothesized set $S(q_{k+1})$ (which depends on the system configuration q_{k+1}). Indeed, we estimate $p(z_{k+1}|x_{k+1})$ without actually taking the robot to state x_{k+1}. We compute a hypothesized visibility region $V(x_{k+1})$ based on the global map built until time k (see Fig. 6 c)). Based on $V(x_{k+1})$, a hypothesized set $S(q_{k+1})$

is generated. In fact, *we make a prediction about $S(q_{k+1})$ from state x_k.*

We evaluate the partial Hausdorff distance considering a transformation including *both translation and rotation.* Given two sets of points P and Q, the partial Hausdorff distance is defined as (see [20]):

$$H(P, Q) = \max(h(P, Q), h(Q, P)) \tag{2}$$

where

$$h(P, Q) = M_{p \in P} \min_{q \in Q} \|p - q\| \text{ and } h(Q, P) = M_{q \in Q} \min_{p \in P} \|p - q\| \tag{3}$$

where $\|.\|$ is the Euclidean metric for measuring the distance between two points p and q. $M_{p \in P} f(q)$ denotes the statistical mean value of $f(q)$ over the set P. $f(q) = \min_{q \in Q} \|p - q\|$ denotes the Euclidean distance from the point p to a point $q \in Q$ that is the one closest to p.

Thus, the Hausdorff distance is computed between the set $S(q_{k+1})$ and the training set, which is stored according to the tp classes.

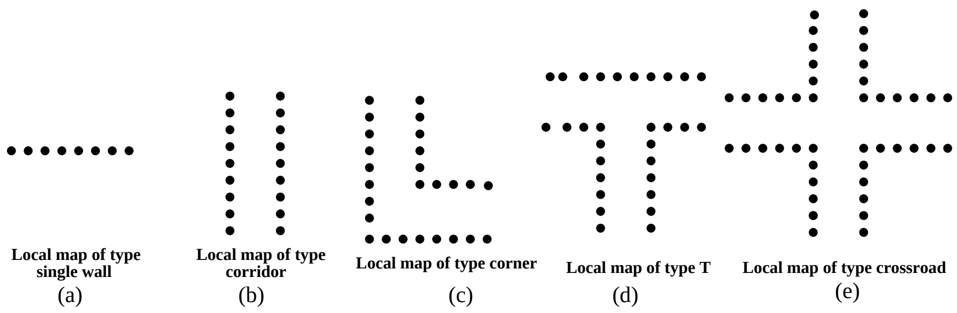

Local map of type single wall
(a)
Local map of type corridor
(b)
Local map of type corner
(c)
Local map of type T
(d)
Local map of type crossroad
(e)

Figure 2. Classes of local maps.

Fig. 2 shows the five types of models that we use. We consider these five models as typical for an indoor structured environment. The dots represent the sensor reading. We use models with different number of points to simulate robots equipped with sensors having different resolutions, and the training data considers some variants of each one of the five main patterns shown in Fig. 2.

Obviously, our approach has the disadvantage that other types of observations can appear. However, the class is only used for selecting the most appropriate robot according to its sensing and motion capabilities. At the end, the local map integrated to the global map corresponds to the actual sensor readings obtained when the system has reached the state x_{k+1} [20].

An example of the Nearest Neighbor Method is presented here. The input data to be classified are the points shown in Fig. 3.

The elements of training set, with the smallest Hausdorff distance to each class are shown in Fig. 4.

z_k	$P(z_k \mid x_k)$
Single wall	0.0942
Corridor	0.2342
Corner	0.1249
Type T	0.1373
Crossroad	0.4091

Table 1. $P(z_k|x_k)$.

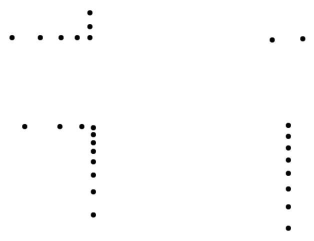


Figure 3. Input data.

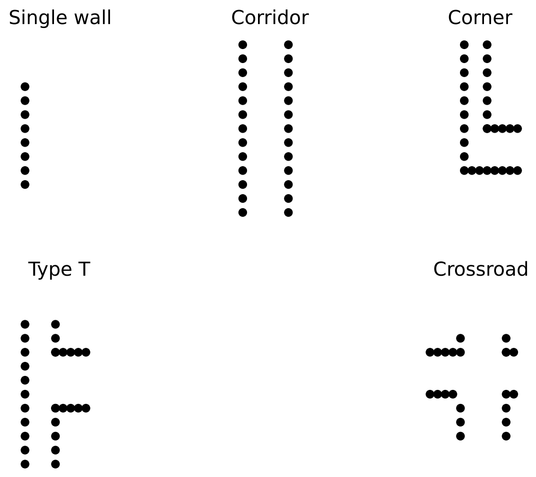


Figure 4. Best matching (smallest Hausdorff distance) per class.

The best matching, which assigns a class, between the input data and all the elements of the training set is shown in Fig. 5. The input data is shown in the figure with dark gray (red) points and the element of the training set is shown with light gray (cyan) points.

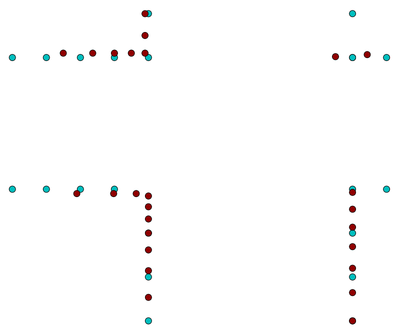


Figure 5. Best matching.

The probability for each class (next observation z_{k+1}), given that a robot is at a state x_{k+1}

(local map and robot configuration) are presented in Table 1. The Bayes' rule is used to calculate those probabilities.

5. Probabilistic motion model $p(x_{k+1}|x_k, u_k)$

We assume robots that can rotate in place (e.g. differential drive robots). The robots move executing only two motion primitives: a rotation in place and a straight line motion. The robots are controlled using a linear velocity v and an angular velocity w. These velocities are not perfect, there is noise modifying them. To model the imperfection over the velocities we use random variables with 0 mean and variance σ^2. Thus, the linear and angular velocities are given by Equations 4 and 5, where ϵ_{σ^2} is an error variable with variance σ^2.

$$\hat{v} = v + \epsilon_{\sigma_v^2} \tag{4}$$

$$\hat{w} = w + \epsilon_{\sigma_w^2} \tag{5}$$

To obtain instances of this error, we use the following simple model. Function sample(.) generates a random sample of the error, with zero mean and variance σ^2. The sample is taken from a normal distribution. Thus, the imperfection over the robot motions depends on the values of both σ_v^2 and σ_w^2. Larger values correspond to larger errors.

$$\hat{v} = v + \text{sample}(\sigma_v^2)$$
$$\hat{w} = w + \text{sample}(\sigma_w^2)$$

To simulate uncertainty over the robots motions, we use an Euler integration method over the robot configuration variables (x, y, θ). Thus, we have:

$$x_{t+1} = x_t + \hat{v}\cos(\theta_t)\Delta t$$
$$y_{t+1} = y_t + \hat{v}\sin(\theta_t)\Delta t$$
$$\theta_{t+1} = \theta_t + \hat{w}\Delta t$$

These equations are iterated to move a robot close to a final desired configuration.

Note that these motions are not perfect and a goal configuration is never reached precisely, because of the noise. Thus, the probability of reaching the next state or equivalent the next configuration is given by Equation 6.

$$p(x_{k+1}|x_k, u_k) = p(\epsilon_{\sigma_w^2})p(\epsilon_{\sigma_v^2})p(\epsilon_{\sigma_w^2}) \tag{6}$$

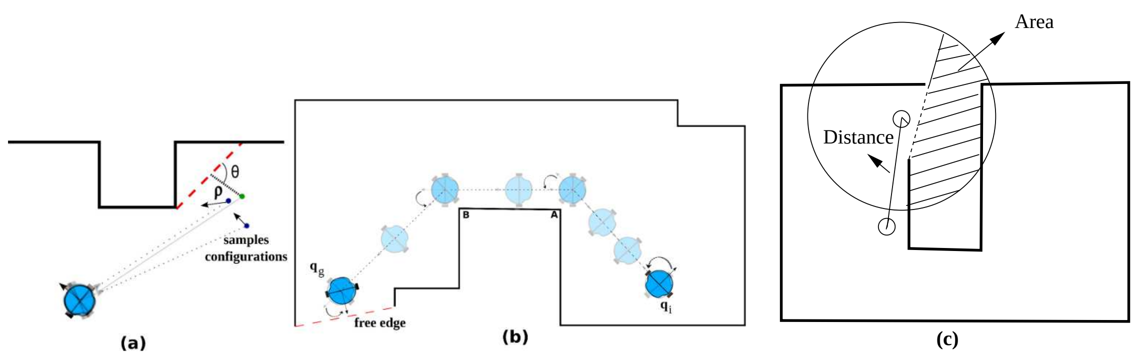
Recall that, a control for a given robot i has the form $u_{(i,k)} = x_{(i,k)} \rightarrow \text{free-edge}_j$, $x_{(i,k)}$ represents the current state of robot i and free-edge$_j$ a free edge to be sensed. To determine one sample per free edge, the sample with the maximal probability per free edge is selected. The probability of each sample is given by Equation 6. To compute an exploration plan, dynamic programming with imperfect information is used (see Section 6).

5.1 *Visibility and Frontiers based Exploration*

We use a frontier-based exploration. To explore the environment robots move to visit free edges. A free edge (frontier) is defined as the border between the visibility region $V(x_i)$ and the unseen environment. Exploring the unknown space is equivalent to sending a robot to visit a free edge.

Each free edge is visited *only once* to explore the space beyond it, and two different robots do not visit the same free edge.

A robot is free to move in the interior of its visibility polygon, as long as it does not collide with an obstacle. We denote by $F(x_i)$ the visibility polygon reduced by the robot radius, i.e., $F(x_i)$ is a safe region for navigation that is visible from state x_i. We define V_{tot} as the total visibility region for the team of robots, and F_{tot} as the total visible region i.e., $V_{tot} = \bigcup_i V(x_i)$ and $F_{tot} = \bigcup_i F(x_i)$.



Motion model Area and traveled distance

Figure 6. (a) and (b) Motion model (c) Expected discovered area and traveled distance.

Based on sampling, we generate candidate sensing configurations over the configuration space $\mathcal{C}$. The samples are only generated close to the free edges using an angle ϕ and a distance ρ as in [20]. See Fig. 6 (a). Besides, only the samples inside F_{tot} are considered. Thus, visibility computations are used to bias the sampling. Since selected samples are close to the border between the explored an unexplored environment, they have a good chance of seeing unknown environment. The exploration starts assuming that F_{tot} has a single connected component. We also assume that the robots will always be inside F_{tot}. Looking for that assumption to hold, only the sample candidate configurations that are further than a minimal distance from a given free edge are evaluated. We only evaluated these sample configurations to avoid that a robot crosses the free edge, due to the noise in the robot velocity, which might result in a collision of the robot. Additionally, to move the robots, we combine the probabilistic model described above, with a feedback motion model based on corners' locations.

We make the robots move over the reduced visibility graph (RGV) [1] computed over $\mathcal{C}$ to follow the shortest path between the two configurations. The global map built until time k is used to compute this RGV [20].

In the feedback motion model, a robot corrects its heading based on the relative position of corners with respect to the robot's configuration. Since the robot is using the location of a corner while it moves, then at each motion's step the robot determines a new heading to reach the corner, and hence the error in the heading does not monotonically increases. For more details see [34].

To illustrate the combination of these two motion models, Fig. 6 (b) is used. While robot moves between its initial configuration q_i to the first corner A and then from corner A to corner B, robot corrects its heading based on the location of the corners, then from corner B to attempt to reach the final configuration q_g, the probabilistic motion model described above is used to simulate the robot motion. While the robot corrects its heading the motion is assumed deterministic, given that the feedback over corners corrected the motion's error. For reaching the configuration

[1]The reduced visibility graph of a polygonal environment, also known as the shortest-path roadmap, is built in the next way: first of all, the vertices in the RVG are the reflex vertices (vertices with an internal angle larger than π) of the polygonal environment. Second, the edge between two vertices in the RVG is generated if the two vertices are endpoints of the same edge of an obstacle, or if a bitangent line can be drawn between such vertices [33].

to explore the free edge (last motion in the sequence) the probabilistic model is used, since no feedback can be used over the configuration sample to be reached. In this case a robot moves as follows: 1) a robot first rotates in place until its heading is pointing to the goal configuration then 2) it moves in straight line until it reaches a goal configuration and finally 3) it rotates again to reach the desired final orientation. These motions are estimated using the probabilistic model. The case when the robot does not move between corners also occurs when a clear line of sight between the initial and goal configuration does not intersect the obstacles in $\mathcal{C}$.

When the global map does not contain free edges the exploration is finished. Since we are updating the global map, we are able to determine whether or not there is a gap (free space) in the boundary of the global map that corresponds to a free edge.

6. Stochastic dynamic programming for selecting an action

We select the action to be executed at time k based on $I_k = \{u_0, u_1, \cdots, u_{k-1}, z_0, z_1, \cdots, z_k\}$, which is the history of all controls applied until time $k-1$ and all the sensed observations until time k. The Stochastic Dynamic Programming (SDP) considering I_k is given by Equation 7 [31]. Equation 7 is evaluated backwards, starting with the last step.

$$J_{N-1}(I_{N-1}) = \max_{u_{N-1}\in U} \tilde{g}_{N-1}(I_{N-1}, u_{N-1}) \text{ and}$$
$$J_k(I_k) = \max_{u_k\in U} \left[\tilde{g}(I_k, u_k) + J_{k+1}(I_k, z_{k+1}, u_k) \sum_{x_{k+1}} p^{\max}(z_{k+1}|x_{k+1}) \sum_{x_k} p(x_{k+1}|x_k, u_k)\, p(x_k|I_k) \right] \tag{7}$$

In Equation 7, J_{k+1} is the utility function, which depends on I_k, z_{k+1} and u_k. N is the planning horizon. The probability of the next state $p(x_{k+1}|x_k, u_k)$ is obtained with Equation 6 (see Section 5). The probability of the state given the information vector $p(x_k|I_k)$ is computed using Equation 11 (see Subsection 6.2). In general, in SDP, there is an expected value computed over all possible observations $E_{z_{k+1}}$, given an information vector I_k and a control u_k. However, in our approach we only use the observation with maximal probability $p^{\max}(z_{k+1}|x_{k+1})$ (given by Equation 1), we select only this observation z_{k+1} to avoid an exponential branching of possibilities due to the exponential growing number of possible sequences of observations.

In Equation 7, $\tilde{g}(I_k, u_k)$ represents the utility (gain over cost) of applying an action. It indicates how useful is to apply a control u_k for an information vector I_k; $\tilde{g}(I_k, u_k)$ can be calculated in terms of $g(x_k, u_k)$ and $p(x_k|I_k)$ as follows [31].

$$\tilde{g}(I_k, u_k) = \sum_{x_k} p(x_k|I_k) g(x_k, u_k) \tag{8}$$

Thus, $\tilde{g}(I_k, u_k)$ depends on both $p(x_k|I_k)$ and $g(x_k, u_k)$ given that uncertainty in both sensing and control is taken into account; $p(x_k|I_k)$ represents the likelihood that process resides in state x at time k. Indeed, in [35], it has been shown that this is sufficient statistic of the information state for a POMDP.

In SDP, the planning horizon N might be set to different values. In the exploration problem addressed in this paper, short and long term plans can be made. In a short term plan $N = 1$, yielding that *each robot moves only once*. But, differently to the work presented in [1], in our current implementation, while executing a plan more than one robot can move at the same time k. In contrast, in a long term plan we calculate N such that, all the free edges present at time k must be visited, and *robots are free to move more than once.*

While executing a plan to avoid collision between two moving robots, the following simple motion coordination scheme is executed. The robots' paths are tested for intersection, if they do not intersect all robots move simultaneously. Otherwise the robot with shortest path to travel moves first. Consider that our work aims mainly to the generation of a plan, assigning the most appropriate robot to explore a part of the environment, recent approaches (e.g. [36]) can be used to coordinate the execution of the robot paths for a large number of robots moving simultaneously.

6.1 *Objective function $g(x_k, u_k)$*

$g(x_k, u_k)$ indicates how useful is to apply a control u_k given that the system is at state x_k. For each robot we define $g(x_{i,k}, u_{i,k})$ as a function depending only on two factors: (1) the new area $A(q_{(i,k+1)})$ that the robot can perceive at a next configuration and (2) the distance a robot would travel to reach that configuration $d(q_{(i,k)}, q_{(i,k+1)})$.

Fig. 6 (c) shows a robot at a configuration near to the border between the known and unknown environment. The circle around the robot depicts the area within the sensor range (the radius of the circle represents the sensor range). The shaded area is the maximum new area that the robot can discover at that configuration. As in [6], we use the global map at time k to eliminate portions of this area, since it is known that there are obstacles already sensed.

We define $g(x_k, u_k)$ (Equation 9) as an objective function that relates the utility (ratio of gain over cost) of the action with the control u_k given the current state x_k.

$$g(x_k, u_k) = \sum_i g(x_{(i,k)}, u_{(i,k)}) = \sum_i \frac{A(q_{(i,k+1)})}{d(q_{(i,k)}, q_{(i,k+1)}) + 1} \tag{9}$$

Equation 10 defines the distance between configuration $q_{(i,k)}$ and configuration $q_{(i,k+1)}$, where $\alpha = \min\{|\theta_{(i,k)} - \theta_{(i,k+1)}|, 2\pi - |\theta_{(i,k)} - \theta_{(i,k+1)}|\}$

$$d(q_{(i,k)}, q_{(i,k+1)}) = \sqrt{(p_{x(i,k)} - p_{x(i,k+1)})^2 + (p_{y(i,k)} - p_{y(i,k+1)})^2 + \alpha^2} \tag{10}$$

The rationale in summing distances and angles is that we want the robots not only translate a small distance, but also that they rotate a small angle. This is because an error in the rotated angle induces an important error in the final robots' locations. Reducing the rotation of the robots has as a consequence that the time to explore the environment is reduced as well. In fact Equation 10 defines a metric commonly used in motion planning (see [33] for more details).

6.2 *Computing $p(x_k|I_k)$*

$p(x_k|I_k)$ is given by Equation 11, it is computed forward starting with the first step.

$$p(x_k|I_k) = \frac{\sum_{x_{k-1}} p(x_{k-1}|I_{k-1})p(x_k|x_{k-1}, u_{k-1})p(z_k|x_k, u_{k-1})}{\sum_{x_k}\sum_{x_{k-1}} p(x_{k-1}|I_{k-1})p(x_k|x_{k-1}, u_{k-1})p(z_k|x_k, u_{k-1})} \tag{11}$$

Equation 11 is used in [37] to calculate the *belief* in Bayes Filter and in [33] for representing a mapping on probabilistic information spaces. Equation 11 can be derived from the Bayes' rule, applying joint probability methods and marginalization; $p(x_k|I_k)$ is expressed in terms of $p(z_k|x_k, u_{k-1})$, $p(x_k|x_{k-1}, u_{k-1})$ and $p(x_{k-1}|I_{k-1})$. To compute $p(x_k|I_k)$, $p(x_0|I_0)$ is needed at the first step. At the beginning, no control has been applied and the only observation is z_0, therefore

Algorithm 1: EXPLORATION($\mathcal{Q}_k$,$\mathcal{W}$)

```
   Input  : $\mathcal{Q}_k$: Robots Configurations at time $k$
            $\mathcal{W}$: Environment
   Output: $\mathcal{M}$: Global Map
1  begin
2      repeat
3          $S \longleftarrow$ SENSORS_READING($\mathcal{Q}_k$, $\mathcal{W}$);
4          $\mathcal{M}$.add($S$);
5          $\mathcal{F} \longleftarrow$ GET_FREE_EDGES($\mathcal{M}$);
6          if $\mathcal{F} \neq \emptyset$ then
7              $\mathcal{Q}_{k+1}$= SAMPLES($\mathcal{F}$, $\mathcal{W}$);
8              $\Pi(k, I_k) = \arg\max_{u_k \in U_k} \Big[ \tilde{g}(I_k, u_k) + J_{k+1}(I_k, z_{k+1}, u_k) \sum_{x_{k+1}} p^{\max}(z_{k+1}|x_{k+1}) \sum_{x_k} p(x_{k+1}|x_k, u_k)\ p(x_k|I_k) \Big]$;
9              $X_{k+1} \longleftarrow$ MOVE_NEW_STATES($\mathcal{F}$, $\mathcal{Q}_k$, $\mathcal{Q}_{k+1}$, $\mathcal{M}$, $\Pi(k, I_k)$);
10         end
11     until $\mathcal{F} = \emptyset$ ;
12     Return $\mathcal{M}$;
13 end
```

$I_0 = z_0$ and $p(x_0|I_0) = p(x_0|z_0)$. This probability is computed using Equation 11, which for $k = 0$ is simply equal to: $p(x_0|z_0) = \frac{p(z_0|x_0)p(x_0)}{\sum_{x_0} p(z_0|x_0)p(x_0)}$.

6.3 *Assigning robots to new states related to the free edges (frontiers)*

Algorithm 1 represents the general method to assign the robots to states in the exploration task. In line 7 in this algorithm, candidates samples configurations are generated close to the free edges. Then, only one sample per free edge is kept, this sample is the one with maximal probability $p(x_{k+1}|x_k, u_k)$ per free edge. Recall that probability $p(x_{k+1}|x_k, u_k)$ is given by Equation 6. In line 8 and 9, dynamic programming is used to find the exploration strategy $\Pi(k, I_k)$, which provides the robots' controls that when they are applied move the robots to the selected locations. Hence, $\Pi(k, I_k)$ also assigns robots to states according to their motion and sensing capabilities.

7. Simulation results

In this section, we present simulation results. All our simulation experiments were run on a quad-core processor PC, equipped with 3 GB of RAM, running Linux. Our software is written in C++. For distinguishing the robots, in all the following figures, robots with better sensing capabilities are shown with a square and robots with better motion capabilities are shown with a circle. However for finding collision free paths both robots are modeled as discs with the same radius. The visibility region of robots with better sensing capabilities are shown in medium gray (green) and the visibility region of robots with better control capabilities are shown in light gray (yellow).

Seven different simulation experiments were performed in four different environments. The motion capabilities are defined using two parameters σ_v^2 and σ_w^2, which correspond respectively to a variance of an error on the linear and angular velocities. Larger values correspond to more imprecise motions. The parameters defining robots' motion capabilities are shown by experiment in Table 2. The sensing capabilities are defined with three parameters: 1) sensing range, 2) sensor's resolution and 3) maximal error on the sensed distance to the obstacles; these parameters are shown by experiment in Table 3. For each experiment, we have run 20 simulations with the same initial robots' configurations.

In all the experiments, we report, the mean and standard deviation of the following performance metrics: (i) total number of sensing location needed for a team to explore the environment, (ii) total angle rotated by the robots, (iii) total distance traveled by the robots, and (iv) percentage of area perceived. The average and standard deviation of the metrics of each experiment have been computed over the number of simulations. We also present the computational running time needed to generate a plan to explore the whole environment.

In Experiments 1, 2, 3 and 4, only two robots are used, each of them has diferent sensing and motion capabilities with respect to the other. In Experiments 5, 6 and 7 the robots are grouped in teams. Each team is composed by elements having better or worse motion or sensing capabilities with respect to the other team. (BS) and (WS) denotes better sensing capabilities and worse sensing capabilities. Analogously, (BM) and (WM) denotes better motion capabilities and worse motion capabilities.

Experiments 1, 2, 3, 4, 5 and 6			
(BM) σ_v^2	(BM) σ_w^2	(WM) σ_v^2	(WM) σ_w^2
0.06	0.04	0.7	0.5
Experiment 7			
(BM) σ_v^2	(BM) σ_w^2	(WM) σ_v^2	(WM) σ_w^2
0.006	0.004	0.7	0.5

Table 2. Motion capabilities.

Experiment 1 and 2, Fig. 7					
(BS) range	(BS) maximal error in sensed distance	(BS) sensor resolution	(WS) range	(WS) maximal error in sensed distance	(WS) sensor resolution
1200 units	2 units	2 degrees	800 units	4 units	4 degrees
Experiments 3 and 4, Fig. 8					
(BS) range	(BS) maximal error in sensed distance	(BS) sensor resolution	(WS) range	(WS) maximal error in sensed distance	(WS) sensor resolution
1250 units	2 units	2 degrees	800 units	2 units	3 degrees
Experiments 5 and 6, Fig. 9					
(BS) range	(BS) maximal error in sensed distance	(BS) sensor resolution	(WS) range	(WS) maximal error in sensed distance	(WS) sensor resolution
1300 units	2 units	2 degrees	900 units	5 units	3 degrees
Experiment 7, Fig. 10					
(BS) range	(BS) maximal error in sensed distance	(BS) sensor resolution	(WS) range	(WS) maximal error in sensed distance	(WS) sensor resolution
1200 units	2 units	2 degrees	700 units	4 units	4 degrees

Table 3. Sensing capabilities.

7.1 *Experiments 1 and 2, short vs long term plans*

Experiments 1 and 2 were performed in the same environment with the same initial robots' configurations. In Fig. 7 corresponding to Experiments 1 and 2, the environment is 3000 units long and 2300 wide, and both robots have a radius of 50 units. In Experiment 1 short term plans are generated, while in Experiment 2 long term plans are computed.

Fig. 7 (a) shows the initial robots' configurations, Fig. 7 (b) shows a snapshots of the exploration; 15 over 20 times the following happened: the robot with better sensing capabilities, was selected to explore the richer visual local maps (left part of the environment) and the robot

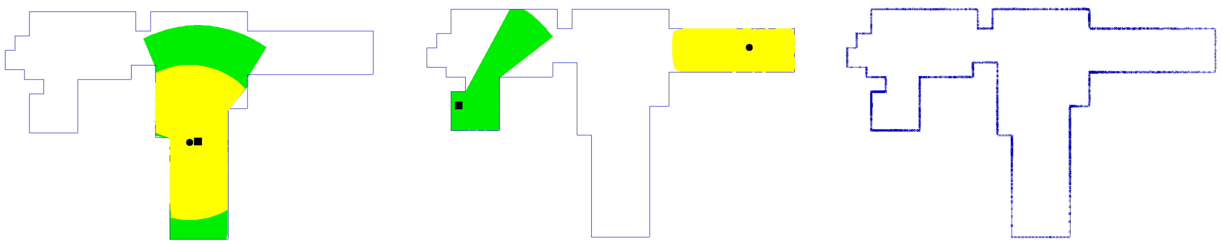

(a) Initial robots' configuration
(b) Snapshot of exploration
(c) Sensed points

Figure 7. Experiment 1.

with better motion capabilities was selected to explore local maps composed only by two parallel lines, the corridor in the right part of the environment (See Fig. 7 (b)), which are hard to be matched with the global map. Notice that in corridors bounded with parallel featureless walls, a matching procedure only finds an alignment between the local and global maps, in the direction perpendicular to the walls. Fig. 7 (c) shows the sensed points collectively obtained by both robots.

Comparing the resulting performance metrics for short term (Experiment 1) vs. long term plan (Experiment 2), one can see that when long term plans are generated, robots travel and rotate more. In long term plans the robots rotate almost three times more than in short term plans, and they travel around 60 % more compared with short term plans. while the other performance metrics remains almost the same. Furthermore, the computational running time needed to generate long term plans are about 3 times larger with respect to short term plans.

Performance metric	Robot 1 (BS) and (WM)	Robot 2 (WS) and (BM)
Mean: number of sensing locations	4.55	4.8
Std. Dev: number of sensing locations	0.73	0.67
Mean: total angle (radians)	6.46	7.48
Std. Dev: total angle (radians)	2.10	1.94
Mean: total distance traveled	3019.41	5031.54
Std. Dev: total distance traveled	528.81	907.63
Mean: area perceived	86.96%	74.23%
Std. Dev: area perceived	2.28%	5.36%
Planning time		
Mean 12.95 sec	Std. Dev. 1.71 sec	

Table 4. Statistics of Experiment 1 shown in Fig. 7, short term plan.

Performance metric	Robot 1 (BS) and (WM)	Robot 2 (WS) and (BM)
Mean: number of sensing locations	4.5	4.9
Std. Dev: number of sensing locations	0.59	0.53
Mean: total angle (radians)	16.16	17.44
Std. Dev: total angle (radians)	6.46	5.43
Mean: total distance traveled	5282.71	9064.66
Std. Dev: total distance traveled	2202.32	2894.15
Mean: area perceived	87.2825%	75.3732%
Std. Dev: area perceived	3.06 %	6.40%
Planning time		
Mean 38.05 sec	Std. Dev. 6.87 sec	

Table 5. Statistics of Experiment 2, long term plan.

7.2 *Experiments 3 and 4, different inital configurations*

In Experiments 3 and 4 the environment is 4000 units long and 4000 wide, and both robots have a radius of 30 units. In both experiments short term plans are computed.

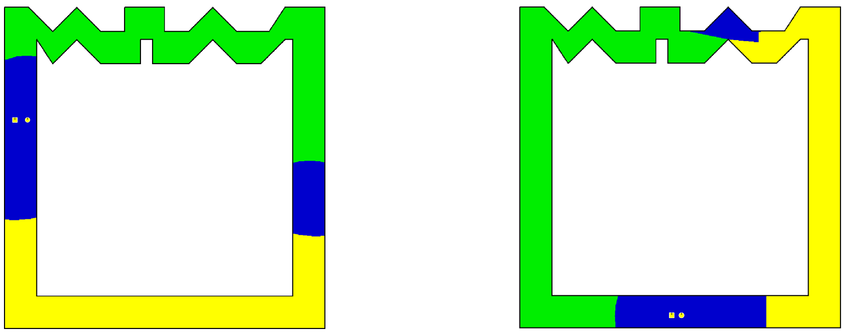

(a) First initial robots' configurations (b) Other initial robots' configurations

Figure 8. Experiments 3 and 4.

In these two experiments, the initial configurations of the two robots are modified. Fig. 8 (a) shows the environment where Experiment 3 is performed, and the first initial configurations of the two robots. In the same figure, the area perceived by the robot with better sensing but worse motion capabilities is shown in medium gray (green), in light gray (yellow) is shown the area perceived by the other robot, and with dark gray (dark blue) the area explored by both robots.

Fig. 8 (b) shows Experiment 4 with the others initial configurations of the two robots. Again, the area perceived by the robot with better sensing but worse motion capabilities is shown in medium gray (green), the area perceived by the other robot is shown in light gray (yellow), and the area explored by both robots is shown in dark gray (dark blue).

For this environment, it is possible to see that independently of the initial robots' configurations, the robot with good sensing capabilities has the tendency to explore local maps with corners, and the robot with good motion capabilities explores local maps composed by corridors. In Fig. 8 (a) this behavior is very clear. Since both robots start perceiving at the same time both a local map with corners and a corridor composed only by two straight line segments. The robot with good motion explores the corridor and the other robot the local map with corners. In Fig. 8 (b), both robots start perceiving only a corridor, a robot moves to the left while the other moves to the right. Once, they arrive to the part of map containing more corners, the robot with good sensing capabilities explores a larger portion of this type of map. In Tables 6 and 7 statistics of Experiments 3 and 4 are presented.

Performance metric	Robot 1 (BS) and (WM)	Robot 2 (WS) and (BM)
Mean: number of sensing locations	9.1	8.9
Std. Dev: number of sensing locations	0.7	0.83066
Mean: total angle (radians)	10.2328	6.95555
Std. Dev: total angle (radians)	1.6258	1.78402
Mean: total distance traveled	7082.54	7193.68
Std. Dev: total distance traveled	2607.13	1409.26
Mean: area perceived	61.9506 %	59.5007 %
Std. Dev: area perceived	7.59017 %	4.02811 %
Planning time		
Mean 15.82 sec	Std. Dev. 1.787 sec	

Table 6. Statistics of Experiment 3 shown in Fig. 8 (a).

In Experiments 3 and 4, the team with (BS) and (WM) capabilities discovers a bit more area, but both teams travels almost the same distance.

Performance metric	Robot 1 (BS) and (WM)	Robot 2 (WS) and (BM)
Mean: number of sensing locations	7.8	9.05
Std. Dev: number of sensing locations	1.36382	1.24399
Mean: total angle (radians)	8.55254	7.04063
Std. Dev: total angle (radians)	2.49364	1.51604
Mean: total distance traveled	9317.03	6673.82
Std. Dev: total distance traveled	4206.27	676.41
Mean: area perceived	62.0572 %	56.4073 %
Std. Dev: area perceived	5.38649 %	5.67474 %
Planning time		
Mean 14.84 sec	Std. Dev. 1.2571 sec	

Table 7. Statistics of Experiment 4 shown in Fig. 8 (b).

7.3 *Experiments 5 and 6, short vs long term plans, with teams of robots*

Experiments 5 and 6 are performed on the same environment shown in Fig. 9. The environment is 8500 units long and 8700 wide, and all robots have a radius of 75 units. 6 robots are used in each experiment. In Experiment 5 the robots make short term plans, while in Experiment 6 the robots make long term plans. In a short term plan more than one robot can move at the same time k but, *each robot moves only once.* In contrast in a long term plan, *robots are free to move more than once, such that, all the free edges present at time k must be visited.*

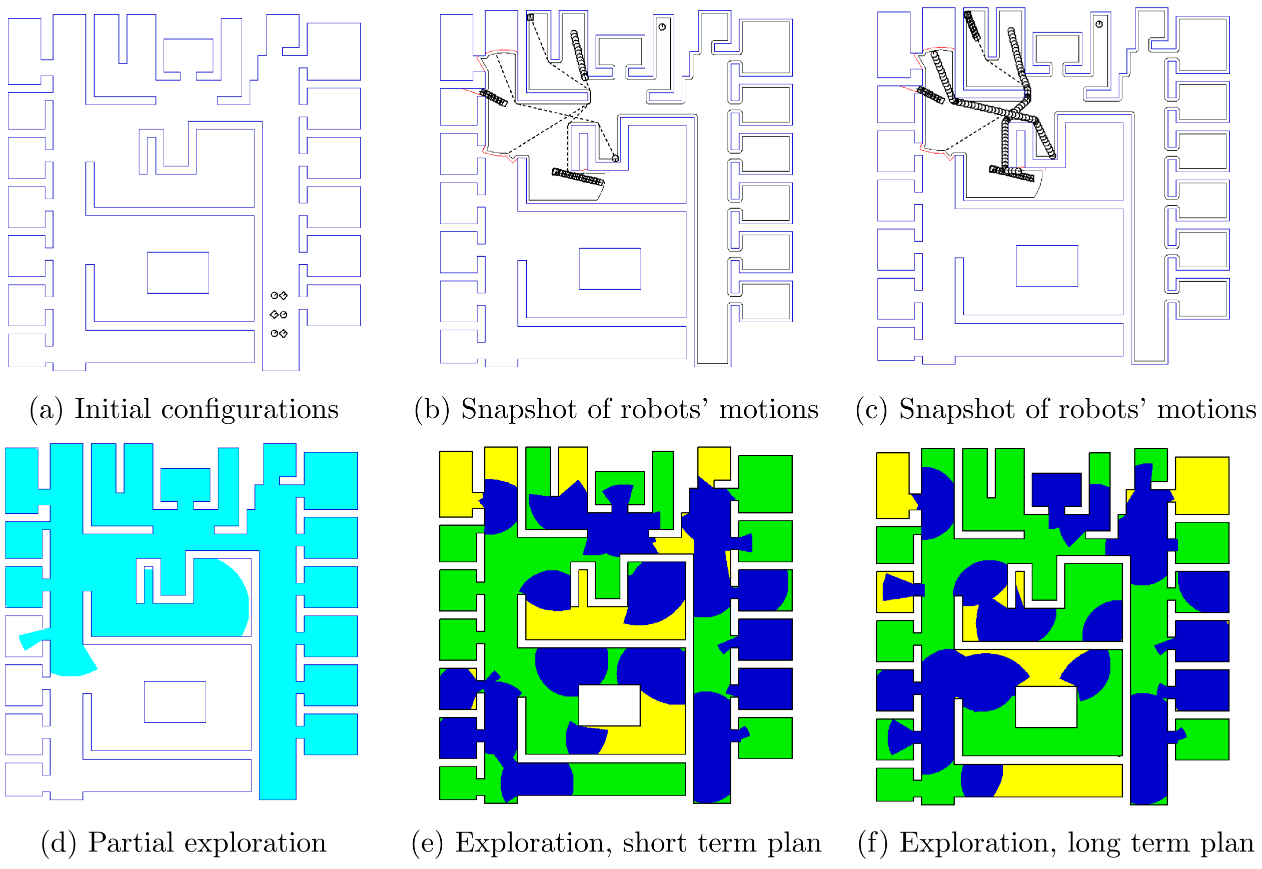

(a) Initial configurations (b) Snapshot of robots' motions (c) Snapshot of robots' motions
(d) Partial exploration (e) Exploration, short term plan (f) Exploration, long term plan

Figure 9. Experiment 5 and 6.

Fig. 9 shows the result of Experiments 5 and 6. The results of Experiment 5 are presented in Figs. 9 (a) (b) (c) (d) and (e). Fig. 9 (a) shows the initial configurations of the 6 robots, Figs. 9 (b) and (c) show snapshots of the robots' motions; 5 of the 6 robots move simultaneously, while one robot is not selected to explore a free edge. Fig. 9 (d) shows in light gray (cyan) the area

Performance metric	Team 1 (BS) and (WM)	Team 2 (WS) and (BM)
Mean: number of sensing locations	43	33.85
Std. Dev: number of sensing locations	3.57	5.17
Mean: total angle (radians)	131.76	100.88
Std. Dev: total angle (radians)	15.88	22.40
Mean: total distance traveled	110257	95255.4
Std. Dev: total distance traveled	11187.8	15276.5
Mean: area perceived	87.73 %	54.34 %
Std. Dev: area perceived	3.57 %	5.46 %
Planning time		
Mean 7 min 3 sec	Std. Dev. 41.4 sec	

Table 8. Statistics of Experiment 5, shown in Fig. 9 (a), (b), (c), (d) and (e).

Performance metric	Team 1 (BS) and (WM)	Team 2 (WS) and (BM)
Mean: number of sensing locations	42.35	37.2
Std. Dev: number of sensing locations	2.35	4.78
Mean: total angle (radians)	166.96	134.12
Std. Dev: total angle (radians)	33.97	30.86
Mean: total distance traveled	134656	119774
Std. Dev: total distance traveled	26666.4	27262.6
Mean: area perceived	86.53 %	59.11 %
Std. Dev: area perceived	3.65 %	7.59 %
Planning time		
Mean 38 min 41 sec	Std. Dev. 21 min 31 sec	

Table 9. Statistics of Experiment 6, long term plans, shown in Figs. 9 (a) and (f).

collectively perceived for all robots until a given time before finishing the exploration. Fig. 9 (e) shows the final exploration result, the area perceived by the team of robots with better sensing but worse motion capabilities is shown in medium gray (green), in light gray (yellow) is shown the area perceived by the other team, and with dark gray (dark blue) the area explored by both teams. Performance metrics related to the exploration task are presented in Table 8. The performance metrics of Experiment 6 are presented in Table 9. The final exploration is presented in Fig. 9 f). Comparing the resulting performance metrics of Experiments 5 and 6, we can see that when long term plans are generated, robots travel and rotate more (for these experiments about 25%), while the other performance metrics remain almost the same. Furthermore, the computational running time needed to generate long term plans is about 5 times larger with respect to the one for short term plans.

7.4 *Experiment 7, grouping the best robots in the same team*

In Experiment 7, the environment is 4000 units long and 3500 wide, and all robots have a radius of 50 units. There are two teams, each team consists of 2 robots, both teams compute short term plans. In this experiment, the robots with better motion (BM) and better sensing (BS) capabilities (with respect to the other team) are grouped in the same team. The objective is to evaluate the performance of each team, when a team is composed by "better" robots, while the other team is composed by robots with worse capabilities. In Fig. 10, the initial locations of the robots and the total area discovered by each team are shown. The area discovered by the team with better robots is shown in medium gray (green), The area discovered by the team composed by "worse" robots is shown in light gray (yellow), and the area perceived by both teams is shown in dark gray (blue).

Table 10 presents performance metrics measuring the exploration task in Experiment 7. The robots with better sensing and motion capabilities discover more than the double of the area discovered by the other team. The number of sensing locations is also almost the double for the team with better robots. However, the total distance traveled by both teams is almost the same. As an interpretation of the results of this experiment, we conclude that while both teams move almost the same, the team composed by better robots is able to discover a significant larger portion of the environment.

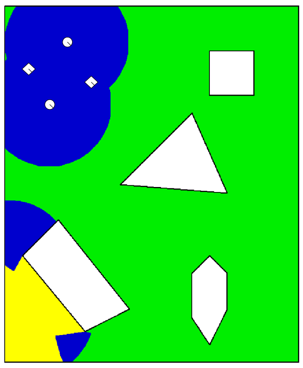


Figure 10. Experiment 7.

Performance metric	Team 1 (BS) and (BM)	Team 2 (WS) and (WM)
Mean: number of sensing locations	8.15	4.85
Std. Dev: number of sensing locations	0.57	1.27
Mean: total angle (radians)	8.42	8.24
Std. Dev: total angle (radians)	3.20	4.93
Mean: total distance traveled	10141.8	8763.92
Std. Dev: total distance traveled	2586.42	3752.88
Mean: area perceived	95.6372 %	42.3139 %
Std. Dev: area perceived	2.58 %	6.28 %
Planning time		
Mean 17.66 sec	Std. Dev. 1.52 sec	

Table 10. Statistics of Experiment 7 shown in Fig. 10.

Based on the results of all the experiments, we conclude that making long term plans with partial and dynamic information will often produce unnecessary long trajectories. Notice that as soon as new frontiers (free edges) appear (which have not been considered at time k), the robot might need to come back to some configurations near to other configurations already visited, thus traveling longer than required. We also conclude that for some environments, our approach generates robot behaviors, that is, robots with good sensing capabilities are selected to explore visually rich local maps, and robots with good motion capabilities are used to explore local maps, which are hard to be matched with the global map. Hence, as in [15], one can notice that making more informed local decisions results in a better global performance.

8. Conclusion and Future Work

In this paper, we have proposed a multi-robot exploration strategy for map-building. We have modeled the exploration problem, as a resource allocation task. We consider heterogeneous robots with different motion and sensing capabilities. We use dynamic programming in states with imperfect information (also called stochastic dynamic programming) as a tool to assign a robot (which is the resource) to explore a part of the environment (which is the task), according to the robots capabilities. We have proposed a Bayesian method to estimate the probability of the next observation given the state of the team of robots. Thus, an observation corresponds to a class, and there is a relation between a type of local map (class) and a part of the state space. This allows us to choose the most appropriate robot according to its sensing and motion capabilities to explore that part of state space. We have also proposed a sampling based one-step ahead optimization, which improves both the quality of the plan and the time to generate it. Our greedy approach avoids unnecessary robots' motions that could arise from a lack of knowledge about the unexplored part of the environment.

A disadvantage of our method is that our planner is centralized and assumes that all information obtained for each robot at a given time k is available to make a decision. As future work, we shall investigate methods to distribute the planning process. Finally, we also want to test our approach in real robots.

References

[1] L. Muñoz Gómez, M. Alencastre-Miranda, R. Lopez-Padilla, R. Murrieta-Cid, Exploration and map-building under uncertainty with multiple heterogeneous robots, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 2011, Shanghai, China, 2011, pp. 2295–2301.

[2] A. Sarmiento, R. Murrieta-Cid, S. Hutchinson, An efficient motion strategy to compute expected-time locally optimal continuous search paths in known environments, Advanced Robotics 23 (12-13) (2009) 1533–1560.

[3] R. Chatila, J. P. Laumond, Position referencing and consistent world modeling for mobile robots, in: IEEE Int. Conf. on Robotics and Automation, ICRA 1985, St. Louis, Missouri, USA, 1985, pp. 135–145.

[4] A. Elfes, Sonar-based real world mapping and navigation, IEEE Transactions on Robotics 3 (3) (1987) 249–264.

[5] S. Thrun, D. Fox, W. Burgard, Probabilistic mapping of an environment by a mobile robot, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 1998, Leuven, Belgium, 1998, pp. 1546–1551.

[6] H. H. González-Baños, J.-C. Latombe, Navigation strategies for exploring indoor environments, International Journal of Robotics Research 21 (10-11) (2002) 829–848.

[7] R. Sim, G. Dudek, Effective exploration strategies for the construction of visual maps, in: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2003, Las Vegas, Nevada, USA, 2003, pp. 3224–3231.

[8] B. Yamauchi, Frontier-based exploration using multiple robots, in: Proc. of the Second International Conference on Autonomous Agents, AGENTS '98, ACM, Minneapolis, USA, 1998, pp. 47–53.

[9] R. Sim, N. Roy, Global a-optimal robot exploration in slam, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 2005, Barcelona, Spain, 2005, pp. 661–666.

[10] H. Feder, J. Leonard, C. Smith, Adaptive mobile robot navigation and mapping., International Journal of Robotics Research 18 (7) (1999) 650–668.

[11] A. Makarenko, B. Williams, F. Bourgault, H. Durrant-Whyte, An experiment in integrated exploration, in: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2002, Lausanne, Switzerland, 2002, pp. 534–539.

[12] G. Oriolo, M. Vendittelli, L. Freda, G. Troso, The srt method: randomized strategies for exploration, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 2004, New Orleans, LA, USA, 2004, pp. 4688–4694 Vol.5.

[13] F. Amigoni, Experimental evaluation of some exploration strategies for mobile robots, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 2008, Pasadena, California, USA, 2008, pp. 2818–2823.

[14] F. Amigoni, V. Caglioti, An information-based exploration strategy for environment mapping with mobile robots, Robotics and Autonomous Systems 58 (5) (2010) 684–699.

[15] N. Basilico, F. Amigoni, Exploration strategies based on multi-criteria decision making for searching environments in rescue operations, Autonomous Robots 31 (4) (2011) 401–417.

[16] M. Juli, A. Gil, O. Reinoso, A comparison of path planning strategies for autonomous exploration and mapping of unknown environments, Autonomous Robots 33 (4) (2012) 427–444.

[17] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, H. Younes, Coordination for multi-robot exploration and mapping, in: Proc. of the AAAI National Conference on Artificial Intelligence, AAAI, Austin, TX, 2000, pp. 852–858.

[18] W. Burgard, M. Moors, C. Stachniss, F. Schneider, Coordinated multi-robot exploration, IEEE Transactions on Robotics 21 (3) (2005) 376–386.

[19] K. Wurm, C. Stachniss, W. Burgard, Coordinated multi-robot exploration using a segmentation of the environment, in: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2008, Nice, France, 2008, pp. 1160–1165.

[20] B. Tovar, L. Muñoz-Gómez, R. Murrieta-Cid, M. Alencastre-Miranda, R. Monroy, S. Hutchinson, Planning exploration strategies for simultaneous localization and mapping, Robotics and Autonomous Systems 54 (4) (2006) 314 – 331.

[21] L. Matignon, L. Jeanpierre, A.-I. Mouaddib, Distributed value functions for multi-robot exploration, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 2012, Saint Paul, Minnesota, USA, 2012, pp. 1544–1550.

[22] K. Singh, K. Fujimura, Map making by cooperating mobile robots, in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 1993, Atlanta, Georgia, USA, 1993, pp. 254–259.

[23] B. P. Gerkey, M. J. Mataric, A formal analysis and taxonomy of task allocation in multi-robot systems, International Journal of Robotics Research 23 (9) (2004) 939–954.

[24] M. Dias, R. Zlot, N. Kalra, A. Stentz, Market-based multirobot coordination: A survey and analysis, Proc. of the IEEE 94 (7) (2006) 1257–1270.

[25] W. Cohen, Adaptive mapping and navigation by teams of simple robots, Robotics and Autonomous Systems 18 (4) (1996) 411–434.

[26] L. P. Kaelbling, M. L. Littman, A. R. Cassandra, Planning and acting in partially observable stochastic domains, Artificial Intelligence 101 (12) (1998) 99–134.

[27] S. Candido, J. C. Davidson, S. Hutchinson, Exploiting domain knowledge in planning uncertain robot systems modeled as POMDPs., in: Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA 2010, Anchorage, Alaska, USA, 2010, pp. 3596–3603.

[28] S. Thrun, Monte Carlo POMDPs, in: S. Solla, T. Leen, K.-R. Muller (Eds.), Advances in Neural Information Processing Systems 12, MIT Press, 2000, pp. 1064–1070.

[29] T. Smith, R. Simmons, Point-based POMDP algorithms: Improved analysis and implementation, in: Proc. of Uncertainty in Artificial Intelligence, Edinburgh, Scotland, 2005, pp. 542–555.

[30] H. Kurniawati, D. Hsu, W. Lee, Sarsop: Efficient point-based POMDP planning by approximating optimally reachable belief spaces, in: Robotics Science and Systems, 2008.

[31] D. Bertsekas, Dynamic Programming and Optimal Control, Athena Scientific, Belmont, MA, 2000.

[32] R. Duda, P. Hart, Pattern Classification and Scene Analysis, John Wiley And Sons Inc, New York, 1973.

[33] S. M. LaValle, Planning Algorithms, Cambridge University Press, Cambridge, U.K., 2006.

[34] R. Lopez-Padilla, R. Murrieta-Cid, S. M. LaValle, Optimal gap navigation for a disc robot, in: E. Frazzoli, et. al. (Eds.), Algorithmic Foundations of Robotics X, Vol. 86 of Springer Tracts in Advanced Robotics, Springer Berlin Heidelberg, 2013, pp. 123–138.

[35] K. J. Åström, Optimal control of Markov processes with incomplete state information, Journal of Mathematical Analysis and Applications 10 (1965) 174–205.

[36] J. Snape, J. van den Berg, S. Guy, D. Manocha, Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles, in: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2009, St. Louis, MO, USA, 2009, pp. 5917–5922.

[37] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT Press, Cambridge, MA, 2005.

Luis Valentin received the B.S. degree in Electronic Engineering from Tecnologico de Celaya (ITC) Celaya, Mexico in 2005, and the M.S. in Electrical Engineering from Universidad de Guanajuato (UG), Guanajuato, Mexico in 2009. He was in a research stay at the Coordinated Science

Laboratory with Professor Seth Hutchinson in 2012 (University of Illinois Urbana-Champaing). Currently (2014) he is pursuing the Ph.D. degree in Computer Science at the Centro de Investigacion en Matematicas (CIMAT), Guanajuato, Mexico. He is mainly interested in robot motion planning, planning under uncertainty, map building and object search with mobile robots.

Rafael Murrieta-Cid received the B.S degree in physics engineering from the Monterrey Institute of Technology and Higher Education, Monterrey, México, in 1990 and the Ph.D. degree from the Institut National Polytechnique, Toulouse, France, in 1998. His Ph.D. research was done with the Robotics and Artificial Intelligence Group of the LAAS-CNRS. In 1998-1999, he was a Postdoctoral Researcher with the Department of Computer Science, Stanford University Stanford, CA, USA. During 2002-2004, he was a Postdoctoral Research Associate with the Beckman Institute and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA. From August 2004 to January 2006, he was a Professor and Director of the Mechatronics Research Center in Tec de Monterrey, Campus Estado de México. Since March 2006, he has been with the Mathematical Computing Group, Centro de Investigación en Matemáticas, Guanajuato, México. His research interests include robotics and robot motion planning. He has published more than 50 papers in journals and international conferences on these topics.

Lourdes Muñoz-Gómez received the M.Sc. degree in Computer Science (2003) and the Ph.D. also in Computer Science (2007), both from Instituto Tecnolgico y de Estudios Superiores de Monterrey Campus Estado de Mxico (ITESM-CEM). Her Ph.D. research was about planning explorations strategies for SLAM with multiple robots. She was in a research stay at the University of Illinois at Urbana-Champaign with Professor Seth Hutchinson, in 2004. In 2006-2012, she was an assistant professor at Electronics and Information Technologies Department at ITESM Campus Santa Fe. Since August 2012, she is an associate professor in the same Department. In January 2012, she obtained the “Rómulo Garza” prize (ITESM national prize) for her research about exploration strategies. Now she is a member of a Research Group at ITESM (Robotics and Virtual Agents in Dual Reality Environments), and her main interests are in mobile robotics and human computer interaction.

Rigoberto Lopez-Padilla received the B.S. degree in Communications and Electronics Engineering and the M.S. degree in Electrical Engineering from Universidad de Guanajuato, Salamanca, Mexico, in 2004 and 2007, respectively. He was in a research stay at University of Illinois at Urbana-Champaign with Professor Steven M. LaValle, in 2011. He is pursuing the Ph.D. degree in Computer Science at the Centro de Investigacion en Matematicas (CIMAT), Guanajuato, Mexico.

Moises Alencastre-Miranda has a Master in Computer Science with a major in Computer Graphics. He also has a Ph.D. in Computer Science with a major in Intelligent Systems at Tecnologico de Monterrey Campus Estado de Mexico. His Ph.D. thesis won the best thesis award in Computer Science, 1st place, from Asociación Nacional de Instituciones de Educación en Tecnologías de la Información (ANIEI) México in 2008, it was about mobile robot navigation and localization with uncertainty in natural environments. He was in a research stay at the Beckman Institute at UIUC with Professor Seth Hutchinson, in 2004. From 2008 to 2012, he was an Assistant Professor at Electronics and Information Technologies Department at Tecnologico de Monterrey Campus Santa Fe. Currently, he is an Associate Professor since August 2012. In 2012 he won the national "Rómulo Garza" research prize of Science, Technology and Engineering in Tecnologico de Monterrey. He is the leader of the Research Group of Computer Graphics and Mobile Robotics at Tecnologico de Monterrey.