




A Distributed Algorithm for Exploration of Unknown Environments with Multiple Robots

Gabriel Aguilar¹ · Luis Bravo¹ · Ubaldo Ruiz²  · Rafael Murrieta-Cid¹ · Edgar Chavez³

Received: 27 February 2018 / Accepted: 24 September 2018 / Published online: 25 October 2018
© Springer Nature B.V. 2018

Abstract

In this paper, we present a complete algorithm for exploration of unknown environments containing disjoint obstacles with multiple robots. We propose a distributed approach considering several variants. The robots are modeled as points or discs, the obstacles are distinguishable or they are not distinguishable, the point robots only communicate at rendezvous, the disc-shaped robots can communicate if they are visible to each other, finally the free subset of the configuration space has one or several connected components. Two possible applications of our algorithms are: 1) Search of a static object in an unknown environment. 2) Damage verification in unknown environments composed by multiple elements (e.g. buildings). The main contributions of this work are the following: 1) The algorithms guarantee exploring the whole environment in finite time even though the robots are not capable of building an exact map of the environment, they cannot estimate their positions and each robot does not have full information about the part of the environment explored by other robots. 2) The method only requires limited communication between the robots. 3) We combine and extend the velocity obstacle method with our proposed approach to explore the environment using disc-shaped robots that are able to avoid collisions with both moving and static obstacles. 4) We propose an exploration strategy such that even if the configuration space has several connected components this strategy guarantees covering the largest possible portion of the environment with an omnidirectional sensor detecting the visibility regions. 5) The algorithm scales well to hundreds of robots and obstacles. We tested in several simulations the performance of our algorithms using different performance metrics.

Keywords Multi-robot · Exploration · Distributed algorithm · Collision avoidance

This work was partially funded by CONACYT projects 220796 and 264896. The authors would also like to acknowledge the financial support of Intel Corporation for the development of this work.

Electronic supplementary material The online version of this article (<https://doi.org/10.1007/s10846-018-0939-9>) contains supplementary material, which is available to authorized users.

✉ Ubaldo Ruiz
uruiz@cicese.mx

Gabriel Aguilar
gabriel.aguilar@cimat.mx

Luis Bravo
luixbravo@cimat.mx

Rafael Murrieta-Cid
murrieta@cimat.mx

Edgar Chavez
elchavez@cicese.mx

1 Introduction

This work addresses the problem of exploring an unknown environment containing a finite number of polygonal obstacles with bounded perimeter using a team of disc-shaped holonomic robots equipped with omnidirectional depth sensors. We assume the robots have the following limitations: 1) They are not capable of building an exact map of the environment. 2) They cannot estimate their positions in the environment. 3) Each robot does not have full information about the part of the environment explored by other robots. We propose a distributed approach that

¹ Centro de Investigación en Matemáticas (CIMAT), Guanajuato, Mexico

² CONACYT Research Fellow, Centro de Investigación Científica y de Educación Superior de Ensenada (CICESE), Baja California, México

³ Centro de Investigación Científica y de Educación Superior de Ensenada (CICESE), Baja California, México

guarantees exploring the whole environment or the largest possible portion of it in finite time even though the robots have the previous constrains.

The environment (workspace) may or may not be bounded by a closed polygonal curve. For the case of an unbounded environment, we assume that the robots are equipped with unbounded range sensors. Without obstacles, we assume that an unbounded environment can be covered by an unbounded range sensor. If the environment is bounded, we assume that the robots are provided with a sensor range larger than the longest line segment contained in the free space. Note that even if the robots are equipped with omnidirectional unbounded range sensors, the obstacles in the environment create occlusions and a motion strategy is needed in order to cover the environment with the foot-print of the robots' sensors. To perform the exploration, the robots navigate the boundary of all obstacles in a distributed way. If the environment has a single connected component, we prove that the proposed strategy covers the whole environment with the foot-print of the robots' sensors. In the case of an environment with several connected components, we prove that the same strategy covers the largest possible region of it.

To support the previous guarantees, first we require to determine whether or not there exists a solution to the next problem: Given several two-dimensional circular agents and a region bounded by a collection of walls find a trajectory connecting two given configurations of the agents such that the agents do not collide with each other and the walls. A complete algorithm that solves that problem has been found in [1]. It is important to note that the solution to that problem is only a necessary condition to explore or cover the whole environment in finite time. Thus, our work builds over the work in [1], and it solves the exploration problem using a novel complete strategy provided that collision-free trajectories for the team of robots exist and that the sensor range is boundless or at least greater than the longest line segment contained in the free space. Furthermore, since unfortunately the approach in [1] is exponential on the number of robots for finding collision-free trajectories, in our work, we adapt the method proposed in [2, 3] to compute collision-free trajectories for the moving agents in a practical way. Besides, based on previous results presented in [4], we can guarantee that for a collision-free subset of the configuration space with several connected components, following the boundary of the obstacle is the motion strategy that makes the robots to cover with the field of view of their sensors the largest possible region of the environment.

1.1 Related Work

The problems of exploration and mapping [5], coverage [6, 7], object search [8–10] and robot rendezvous [11]

are related to this work. In [8], the problem of exploring an unknown environment for searching one or more recognizable targets is considered. A method with limited sensing capabilities of the robot is presented and the environment is represented in the so-called boundary place graph, which records the set of landmarks. Similar to our work, the algorithm described in [8] is based on following the boundary of the obstacles and creating an abstract representation of the environment. However, some important differences between both works are the following. In our work, a robot can explore the whole environment using only a range depth sensor while in [8] it has to be equipped with a vision based system. We propose a distributed multi-robot strategy to explore the environment while in [8] only the case of one robot is considered. In [9], the authors addressed the problem of continuous sensing for finding an object, whose unknown location is characterized by a probability density function, using one robot. As in [9], our strategy can be used to find a static target in an unknown environment, however, in our case several robots working in a distributed fashion can perform the task.

In [10], the Gap Navigation Tree (GNT) is proposed for a point robot navigating without using the robot coordinates. The GNT can be considered as a topological map. The GNT is different from previous approaches in that it is a local representation, defined for the current position of the robot, rather than a global one. In [10], the authors also presented a method to explore a simply connected environment to find and encode a landmark or object in the GNT (to later come back to it). The method proposed in [4] extends the work in [10] to a disc-shaped differential drive robot, that method guarantees exploring the whole environment or the largest possible region of it. The disc-shaped robot is able to find a landmark and encode it in the GNT or declare that an exploration strategy for this objective does not exist. In this work we take inspiration from [10] and [4]. As in [10], the environment is represented by a tree data structure and a robot uses pebbles to detect when a robot has totally circumnavigated an obstacle. However, while in [10] the tree is build locally respect to the robot location, in this work the tree represents the environment globally with respect to the initial location of the robots. Besides, in [10] the obstacles are considered distinguishable, while in this work we also consider indistinguishable obstacles. The term distinguishable is used to establish that every obstacle can be identified as different from the others. Furthermore, the work in [10] only deals with a single robot while in this work we deal with multiple robots. As in [4], we consider a disc-shaped robot, however, while in [4] we consider *workspaces* with a single bounded connected component, in this work we deal with maps having unbounded *workspaces* with multiple connected components. Furthermore, the most important difference between the work presented in [4] and

this work is that [4] only considers a single robot while in this work the main interest is to propose distributed algorithms for multiple robots.

Some exploration strategies are based on exploring the frontier of the environment. This idea was originally proposed by Yamauchi in [12]. In frontier-based exploration methods, the robot goes to the imaginary line that divides the known and unknown portions of the environment. An approach to multi-robot exploration of large environments is presented in [13]. The method uses a vision system that sweeps the free space and generates a graph-based description of the environment. The graph is used to guide the exploration process and can also be used for tasks such as place recognition or path planning. An important difference between our work and the one in [13] is that in our case an arbitrary number of robots can be used to perform the exploration, while in [13] only two robots can perform the task, where one is used as landmark for localizing the second robot. Multi-robot exploration and mapping have been proposed in some works [5, 14]. In [5], the authors proposed an exploration strategy where the map is represented using an occupancy grid and the possible locations for the next exploration step are defined over cells lying on the boundary between the known and unknown space. In [14], a multi-robot exploration strategy is presented where a segmentation of the environment is used to determine exploration targets for the individual robots. This segmentation improves the distribution of the robots over the environment. In [15], the authors propose a method for multi-robot exploration based on Decentralized Markov Decision Processes (Dec-MDP). In [16], several exploration strategies are experimentally compared in different environments in order to get a comprehensive assessment of the strengths and weakness of the approaches. In [17], distributed algorithms for the construction of a triangulation using a multi-robot system are proposed. The authors apply their approach to exploration, coverage and surveillance using a swarm of robots with limited individual capabilities. Some important differences between this paper and the one in [17] are the following: In [17] a triangulation to cover the environment is done over the free space while in this work a triangulation is done over the obstacles. In [17] the robots are used as nodes of the triangulation to cover the free space while in this work the triangulation is required only to prove that all obstacles shall be visited by the robots. In this work, we consider multiple connected environments while in [17] the authors considered simply connected environments. In this paper, the robots follow the boundaries of the obstacles, which makes this work related somehow to Bug algorithms [18]. In [18], the performance of several bug algorithms is compared in terms of different criteria, for instance the distance traveled to reach a goal. However, in [18] the

authors did not analyze the performance of multi-robot systems for the task of covering an environment. The work in [18] neither studied the effect of limited communication among the robots and how this affect the distance that the robots need to travel in order to cover the environment. In this paper, we study those issues.

The so-called velocity obstacle is a family of methods [2, 3, 19–21] that generates a region of collision within the velocity space considering the dimension of the robot, moving obstacles (or other robots) and their velocities. Several techniques improve the general method proposing strategies to avoid undesired oscillations of the robot [2, 3]. The work proposed in [21] truncates the collision cone region to consider only collisions that will occur within a finite window of time. Some approaches consider probabilistic reasoning (e.g [20]). In this work, to implement our proposed exploration strategy we adapt the approach called reciprocal velocity obstacle [2, 3] to avoid collisions among the disc-shaped agents.

1.2 Main Contributions

The main contributions of this work are the following. We propose distributed algorithms that guarantee a complete exploration of the environment in finite time by a team of robots under two general constraints:

1. The robots do not have full information about the regions being explored by other members of the team.
2. The communication between robots is limited.

We study the problem under different variants:

- The robots are modeled as points or discs.
- The obstacles are distinguishable or they are indistinguishable.
- The point robots can only communicate at rendezvous and the disc-shaped robots can communicate if there is a clear line of sight between them.
- The collision-free subset of the configuration space has only one connected component or it has several connected components.

A preliminary version of a portion of this work appeared in [22]. The main distinguishing features of our current work compared with our previous research in [22] are:

- In [22], we have shown that the point robots will travel the boundary of all obstacles, in this work we show that this condition is sufficient to collectively cover the whole environment with the field of view of their sensors.
- In this work, we present a larger set of simulation experiments and an analysis of results including the execution time as a performance metric.
- In this work, we extend the approach to disc-shaped robots instead of point robots. In particular, if a

solution exists for the problem of finding a trajectory connecting two given configurations of the disc-shaped agents such that the agents do not collide with each other and the walls [1], then we show that under the proposed approach the agents are able to collectively navigate the boundary of all obstacles and cover the whole environment with the foot-print of their sensors, assuming a line of sight communication capability between the agents and that the sensor range is boundless for unbounded environments or at least greater than the longest segment contained in the free space of a bounded environment.

- In [1], it has been shown that finding a collision-free trajectory connecting two given configurations of disc-shaped robots is exponential on the number of moving agents in the environment. Consequently, in this work, we adapt the approach proposed in [2, 3] to compute collision-free trajectories for the moving agents in a practical way.
- Based on previous results presented in [4], we can guarantee that for a collision-free subset of the configuration space with several connected components, following the boundary of the obstacle is the motion strategy that makes the robots to cover with the field of view of their sensors the largest possible region of the environment.

2 Problem Formulation

A team of disc-shaped holonomic robots with omnidirectional sensors is moving in an unknown environment E with a finite number of polygonal obstacles. It is assumed that all obstacles have bounded perimeter. The environment (workspace) may or may not be bounded by a closed polygonal curve. The configuration space may have a single or several connected components. For the case of an unbounded environment, we assume that the robots are equipped with unbounded range sensors. Without obstacles, we assume that an unbounded environment can be covered by an unbounded range sensor. If the environment is bounded, we assume that the robots are provided with a sensor range larger than the longest line segment contained in the free space. The task of the robots is to navigate the boundary of all obstacles in a distributed way. If the environment has a single connected component, the goal is to cover the whole environment with the foot-print of the robots' sensors. In the case of an environment with several connected components, the goal is to cover the largest possible region of it.

First, we model the robots as points, later we extend the approach for disc-shaped robots. In both cases, the robots have no initial knowledge of E and they are not capable of building an exact map of the environment. They also lack

of sensors that might be used to estimate their positions in E . However, each robot has an abstract sensor that is able to detect and track discontinuities in *depth information* (an instance of this abstract sensor can be a laser-range finder measuring depth-distance discontinuities), and we assume that each robot has an ordered unique identifier. Let ∂E denote the boundary of obstacles. To explore the environment, each robot moves in contact with ∂E or it moves over bitangents of the environment. For the case of disc shaped robots, they also must avoid collisions with other robots.

We consider two types of environments, in the first case, the obstacles are distinguishable, i.e., each obstacle is uniquely identifiable, which can be imagined as each obstacle having a different color, and in the second case, the obstacles are not distinguishable. In the case of distinguishable obstacles, we suppose that the robots are equipped with visual sensors that are able to differentiate the obstacles using their colors or any other visual characteristic. We also assume that the robots are equipped with sensors that are able to detect bitangents. For the indistinguishable case, we only assume that the robots have sensors which are able to detect bitangents. Regarding the communication capabilities of the robots, the point robots can only communicate at rendezvous and the disc-shaped robots can communicate if there is a clear line of sight between them.

3 Distinguishable Obstacles

In this section, we describe our first approach to solve the problem. First, we make the assumption that each obstacle is uniquely identifiable, which can be imagined as each component having a different color.

3.1 General Setup

Suppose there are n robots tagged with unique numeric identifiers and m distinguishable obstacles in E . We assume that the robots can only communicate to each other at rendezvous. Each robot is denoted as r_i where $i = 1, \dots, n$ and each obstacle as O_j where $j = 1, \dots, m$. Recall that all robots are initially located at the vertex of one obstacle.

3.2 Obstacle Exploration

The robot with the smallest id in the team is selected as a *scout robot* and its task is to explore the current obstacle. From the initial location, the scout robot follows the boundary of the obstacle storing at each vertex the sequence of bitangents detected by its sensor sorted by angular value, please refer to Fig. 2. It may be possible to find more than one bitangent between two obstacles. In this case, the robot considers only the first occurrence in

angular value. The scout keeps track of the obstacles that it has detected using a stack D_i . Once the scout robot has completely circumnavigated the boundary of the obstacle it shares D_i with the rest of the robots located at the same obstacle.

Since the scout robot has no knowledge of its location in the environment, it makes use of a *unique distinguishable marker* to keep track of the first contact with an obstacle. If by following the boundary the marker is found again, this indicates that the obstacle has been circumnavigated completely. Before visiting a new obstacle the scout robot always picks up its marker.

3.3 Obstacle Assignment

To visit the unexplored obstacles in D_i the robots implement the following strategy. Let $|D_i^{ne}|$ be the number of unexplored obstacles in D_i . If $|D_i^{ne}| < n$ then $k \in \mathbb{N}$ robots are assigned to each unexplored obstacle in D_i , where $n = k|D_i^{ne}| + l$ and $l < |D_i^{ne}|$. Each one of the l remaining robots is assigned to a different obstacle until all robots in the team have an assignment. Note that each unexplored obstacle in D_i is visited by a team of at most $k + 1$ robots. If $|D_i^{ne}| > n$, then each robot is assigned to a different unexplored obstacle in D_i . In this case, the robots start a somewhat independent exploration of E .

Before departing from the current obstacle, all robots mark the obstacles in their stacks that are going to be visited by each one of the teams. This helps to avoid that one team explores again an obstacle already assigned to other team. Once the teams reached the assigned obstacles, the procedure described before is applied again.

3.4 Representation of the Strategy

A graphical description of the exploration strategy is shown in Fig. 1. The proposed strategy can be represented using a tree. Each node in the tree contains information about the visited obstacle, the assigned team members, and the stack of those members after exploring the obstacle. The children of the node represent the unexplored obstacles that are visited from that node. An example of the strategy and its tree-based representation is presented in Subsection 3.6.

3.5 Backtracking

To reduce the possibility of visiting the same obstacle several times, we propose the following strategy. The robots are constrained to gather at the root node once the tree has reached at most a given height h . After all robots have arrived to the root node, they share their information about the obstacles. Later, the robots continue visiting the unexplored obstacles using the strategy described above

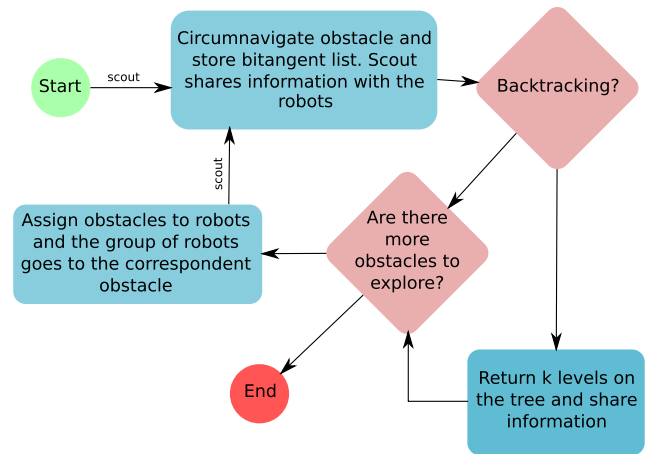


Fig. 1 Graphical description of the exploration strategy for distinguishable obstacles

to create teams (see Fig. 1). Experiments varying h and showing the behavior of the strategy are presented in Section 6.

3.6 Example

Figure 2 shows an example where the robots are located at the boundary of O_1 (black circle). From the initial location, the robots have found $D_i = [O_1, O_2, O_4]$. Note that at this point, all robots share the same information. In Fig. 2, the scout robot r_1 places its marker and circumnavigates O_1 in counter-clockwise direction, adding each new obstacle detected during the trip. In this case, r_1 finds two new obstacles O_5 and O_7 , thus $D_1 = [O_1^e, O_2, O_4, O_5, O_7]$. The obstacle O_1 has been completely explored thus it is identified

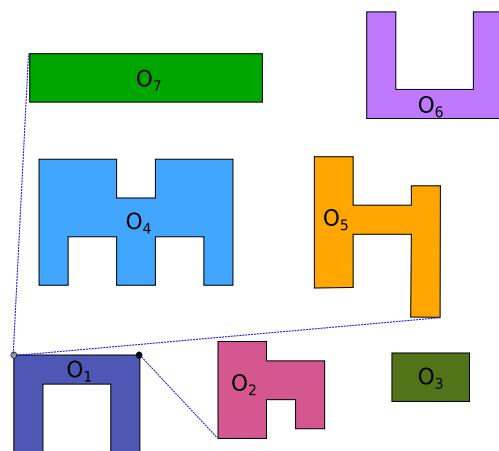


Fig. 2 An environment with distinguishable obstacles. The robots compute the obstacles that are visible from their initial location (black circle). One robot circumnavigates O_1 in counter-clockwise direction detecting new obstacles in E . The gray circle indicates the locations where a new obstacle is detected using the bitangents generated by the obstacles

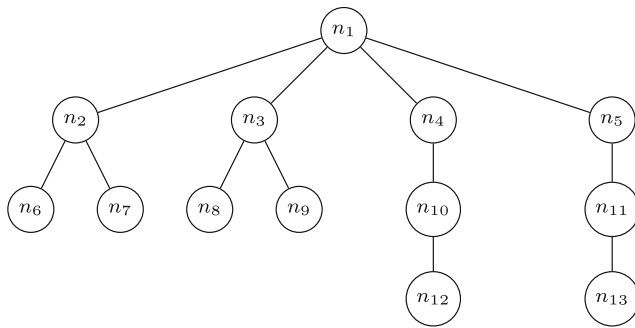


Fig. 3 An example of the tree representation of the strategy to explore environments with distinguishable obstacles. The node information is described in Table 1

as O_1^e . Following the boundary of O_1 , r_1 reaches again the location of the team, it picks its marker and shares D_1 with the rest of the members, thus $D_i = D_1$.

Figure 3 shows a tree representation of the strategy using 6 robots to explore the environment in Fig. 2. The description of the nodes is shown in Table 1. The rows in Table 1 correspond to the nodes in the tree representing the robots' assignment to the obstacles, and the columns the different features of each node. In Table 1, we assume that once the strategy starts to assign one robot per obstacle, the robots do not meet again. In this case, it is very likely that the robot explores obstacles already visited by other teams. In our example, this can be observed in obstacles O_3 , and O_6 , since each one is visited by three different teams. The previous strategy implies that in worst case, each robot visits all obstacles in the environment, i.e., any pair of robots never visits an obstacle at the same time during their motion in E , thus, they cannot share their information.

We can improve the performance by using the tree representation of the strategy. Suppose the robots are forced to gather at the root node when the tree's height is 2. In the example in Fig. 3, once the robots have reached the nodes n_6 , n_7 , n_8 , n_9 , n_{10} and n_{11} they are forced to return to node n_1 and share their information. As a consequence, r_3 and r_4 find out that O_6 has already been explored, and they do not need to visit it again. Thus, in this case, the nodes n_{12} and n_{13} are not part of the tree describing the exploration strategy. If after sharing their information, the robots found out that some obstacles need to be explored they can travel to them and start a similar strategy.

4 Indistinguishable Obstacles

In this section, we present a strategy to solve the case of non-uniquely identifiable obstacles. The main idea of the strategy is building a tree representation of the environment

where the nodes correspond to the obstacles and the edges to the bitangents between them. The tree is constructed following a depth-first approach. The robots return to the node (obstacle) in the previous level only when all bitangents of the nodes (obstacles) in the current level have been visited. The obstacle exploration can be seen as a preorder traversal of the tree [23]. The bitangents are completed following a postorder traversal of the tree [23]. More details about the strategy and its representation are given in the following subsections.

4.1 General setup and Markers

Suppose there are n robots tagged with unique numeric identifiers, where each robot is denoted as r_i , $i = 1, \dots, n$. All robots are initially located at the vertex of one obstacle. Since the obstacles are indistinguishable, the only information available for the robot are the bitangents detected by their sensors. As a robot circumnavigates the obstacles it stores at each vertex the detected bitangents sorted by angular value. Note that in this case, it is not possible to distinguish if two or more bitangents are related to the same obstacle. The direction in which the robots circumnavigate the obstacles is fixed at the beginning of the strategy. The robots travel using the bitangents as a path.

To keep track of the progress made by the exploration strategy, the robots make use of two different types of markers which are described in the following list:

1. Each robot has a unique *starting* marker that is used to identify its starting position when it circumnavigates an obstacle. The starting marker has information about the owner's id, and it is available to other robots.
2. The robots have a *generic* unlimited set of markers that are used to label the obstacles as visited. These markers have information about the id of the robot that visited the obstacle and placed them.

In an actual implementation of the approach, we think that it is feasible that a robot left and recover markers, not necessarily with an arm/hand manipulation device but by using other simpler device, for instance, a scoop.

Furthermore, the marker implementation can vary, one option is that a marker is implemented through 1) computer vision, using markers labeled with bars codes or 2) dropping RFID (Radio Frequency Identification) tags as in [24]. 3) Other option is to implement "virtual markers" using a GPS (assuming that the error of the GPS is small in comparison to the size of the environment). Note that, even if the GPS is available, its use is relegated exclusively to the marker implementation. The motion strategy proposed in this paper still is useful to guarantee the exploration of the whole environment. The actual implementation of the markers in

Table 1 Node information for the tree in Fig. 3

Node	Explored	Robots	Queue
n_1	O_1	$r_{1,\dots,6}$	$D_{\{1,\dots,6\}} = \{O_1^e, O_2, O_4, \mathbf{O}_5, \mathbf{O}_7\}$
n_2	O_2	r_1, r_5	$D_{1,5} = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, \mathbf{O}_3, \mathbf{O}_6\}$
n_3	O_4	r_2, r_6	$D_{2,6} = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, \mathbf{O}_3, \mathbf{O}_6\}$
n_4	O_5	r_3	$D_3 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, \mathbf{O}_3, \mathbf{O}_6\}$
n_5	O_7	r_4	$D_4 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, \mathbf{O}_6\}$
n_6	O_3	r_1	$D_1 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_3^e, O_6^e\}$
n_7	O_6	r_5	$D_5 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_3^e, O_6^e\}$
n_8	O_3	r_2	$D_2 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_3^e, O_6^e\}$
n_9	O_6	r_6	$D_6 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_3^e, O_6^e\}$
n_{10}	O_3	r_3	$D_3 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_3^e, \mathbf{O}_6\}$
n_{13}	O_6	r_4	$D_4 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_6^e, \mathbf{O}_3\}$
n_{12}	O_6	r_3	$D_3 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_3^e, O_6^e\}$
n_{13}	O_3	r_4	$D_4 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_6^e, O_3^e\}$

The bold fonts indicate the new obstacles founded after circumnavigating the explored obstacle

real robots is outside of the scope of this paper. A work is needed in which a careful evaluation of the hardness or easiness of the proposed options listed above is carry out. We left it for future work.

4.2 Data Structure for Navigation and Coordination

Each robot has a stack that works as a schedule for visiting bitangents when the number of bitangents is greater than the number of robots. Also, each robot has the ability to store and construct an own tree, whose nodes are the obstacles in the environment and the edges are the bitangents between each pair of obstacles. The robot has knowledge about the index of the vertex and it can distinguish between bitangents in a vertex (based on the angular value). When an obstacle is visited, the information about the arrival vertex and the angular value of the arrival bitangent is encoded in the tree to know how to come back to the previous obstacle.

4.3 Obstacle Exploration

Let N_c be a tree node that contains information about the bitangents in the current obstacle. The robot r_s with the lowest id in the team is selected as a *scout* robot. This robot places its starting marker and circumnavigates the obstacle, recording in N_c the new possible obstacles based on the bitangents detected at each vertex. One of the following two cases occurs during the previous procedure:

1. No marker is already present in the obstacle or the scout robot only found starting markers with a higher id. Once the scout robot has circumnavigated the obstacle, it picks its *starting* marker and places a *generic* marker indicating that all bitangents associated to the obstacle have been detected. The list of vertices of the node and

2. The scout robot found a generic marker or a starting marker with a lower id. The scout robot returns to the initial vertex and picks its starting marker. If the scout robot has elements in its stack it keeps visiting the bitangents, otherwise, the robot has nothing to do and it turns off or returns to the initial obstacle using its own tree.

4.4 Obstacle Assignment

The following rule for distributing the exploration of new obstacles is used by the robots at the initial vertex.

If the number of robots exceeds or equals the number of children of N_c then the robots are evenly assigned to each bitangent creating teams. After that, each team applies the strategy at each new reached obstacle. If the number of children in N_c exceeds the number of robots then each robot is assigned to several bitangents.

The teams are split during exploration but never merged again because backtracking is not useful in this case, since

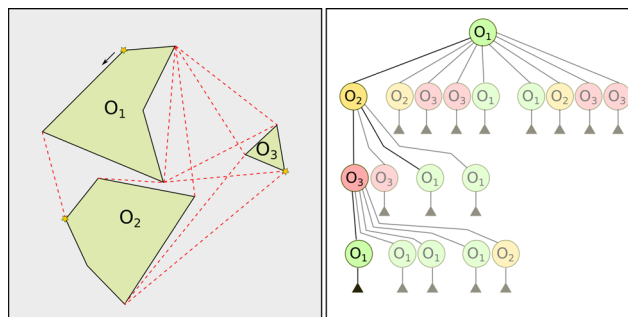


Fig. 4 Example of the representation of the strategy for indistinguishable obstacles using a single robot

a robot only knows which vertex has to visit but it does not know if two vertices belong to the same obstacle. The robots only share information among the members of the team in the current obstacle. A more detailed description of the exploration strategy is presented in Algorithm 1.

Algorithm 1 Environment_exploration_indistinguishable

Input: An unknown environment E , a tree T encoding the obstacles and bitangents detected during the strategy, a team R with n robots initially located at a vertex of an obstacle, the respective bitangent g where the team came from, and the node *parent_node* that contains the list of bitangents of the obstacle in the previous level of the tree.

```

1: Select the robot  $r_s$  with the lowest id in  $R$ .
2: Let  $N_c$  be a tree node that will contain the list of the bitangents
   in the current obstacle.
3:  $r_s$  places its unique starting marker and circumnavigates the
   obstacle recording the detected bitangents in  $N_c$  until the
   unique starting marker is found again.
4: if a generic or a lower unique starting marker is found then
5:   if the robots in  $R$  have unexplored bitangents then
6:     Return to parent_node and select the next unexplored
       bitangent  $g$  in parent_node.
7:     Environment_Exploration_Indistinguishable( $E$ ,  $T$ ,  $R$ ,
        $g$ , parent_node)
8:   else
9:     The robots in  $R$  stop the exploration.
10:  end if
11: else
12:   $r_s$  picks its unique starting marker and shares  $N_c$  with the
     rest of the team members updating their trees (inserting
      $N_c$  in  $T$ ).
13: end if
14: if  $N_c$  is not empty then
15:   The detected bitangents in  $N_c$  are evenly distributed
     among the robots in  $R$  creating subteams.
16:   for each created subteam do
17:     The robots in the subteam travel through the assigned
       bitangent  $g$ .
18:     Environment_Exploration_Indistinguishable( $E$ ,  $T$ ,  $R$ ,
        $g$ ,  $N_c$ )
19:   end for
20: else
21:   The robots in  $R$  stop the exploration.
22: end if

```

4.5 Example

Figure 4 shows an example of the tree representation of the strategy for a single robot. Every node corresponds to an obstacle. Every edge corresponds to a bitangent generated between two obstacles. Note that non-convex obstacles can have bitangents generated between two vertex of the same

obstacle, for example, obstacle O_1 in the figure. The yellow stars denote the arrival vertex of each obstacle. The arrows indicate the direction in which the robot circumnavigates the obstacle. In Fig. 4, the robot starts circumnavigating obstacle O_1 detecting 9 bitangents. After the robot has circumnavigated O_1 , it marks O_1 as visited and proceeds to explore the first detected bitangent which leads to obstacle O_2 . Once the robot has reached O_2 , it circumnavigates the obstacle detecting 3 new bitangents (note that the arrival bitangent is ignored). The robot marks O_2 as visited and proceeds to explore the first detected bitangent which leads to obstacle O_3 . During the exploration of O_3 the robot detects four new bitangents. Traveling each one of those bitangents leads to an obstacle that has already been explored, which is detected using the generic marker leaved by the robot at each visited obstacle. Since all bitangents of obstacle O_3 led to a previously visited obstacle then the robot proceeds to explore the bitangents in the previous level of the tree. In this case, those bitangents also lead to previously visited obstacles. The robot continues applying this strategy until it reaches the root node. It is important to note that some branches of the tree are not completed when an obstacle is visited for second time and the robot found that the obstacle has already visited (see Fig. 4). Recall that each robot has a stack that works as a schedule for visiting bitangents. The bitangents are pushed into the stack according to the order in which they are detected. Since the bitangents are explored as they are popped from the stack then the tree is built and traversed following a depth-first approach. This approach guarantees that each edge in the tree (bitangent) is traveled only two times: when the robot goes down a level in the tree and when it goes up a level in the tree. The obstacle exploration can be seen as a preorder traversal of the tree. The bitangents are completed following a postorder traversal of the tree.

5 Completeness of the Exploration Strategy

In this section, we prove that a team of point robots equipped with unbounded range sensors will sense the entire environment by following the strategy proposed in this work.

Proposition 1 *Given two non-overlapping triangles in general position in the free space, there are at least two bitangents between them.*

Proof Let \mathcal{CH} be the convex hull of the vertices in both triangles (see Fig. 5). A pair of segments of \mathcal{CH} correspond to bitangents generated from their vertices. \square

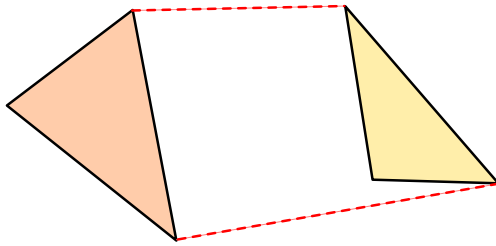


Fig. 5 Bitangents between two triangles

Proposition 2 Given two non-overlapping convex or non-convex polygons A and B in general position in the free space, there are at least two bitangents between them.

Proof Suppose A and B are two non-overlapping polygons with m and n segments, respectively. To prove this proposition, the triangulations of A and B are used. Let $\{A_i | i = 1, \dots, m - 2\}$ be the set of triangles in the triangulation of A . Analogous, $\{B_i | i = 1, \dots, n - 2\}$ denote the set of triangles in the triangulation of B . Select two triangles A_i and B_j , one from each triangulation, and construct their convex hull. From Proposition 1, there are at least two bitangents in the convex hull connecting A_i and B_j . Select a triangle B_k adjacent to B_j . If the triangle B_k intersects one of the previous bitangents then another bitangent can be created with the vertex in B_k not shared by B_j and B_k (see Fig. 6). Thus, triangles A_i , B_j and B_k are connected by at least two bitangents. The remaining triangles in A and B can be added following a similar approach guaranteeing that the number of bitangents never decreases (see Fig. 6). □

In the following, all obstacles are assumed to be polygons with non-zero area.

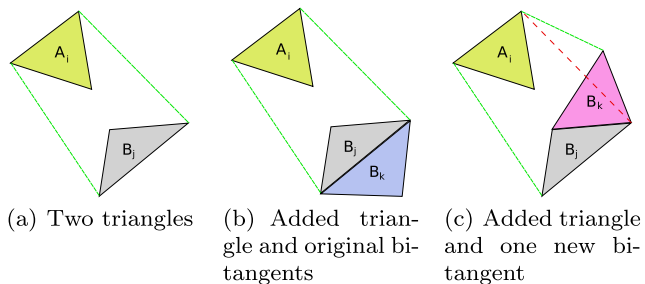


Fig. 6 The left figure shows the bitangents between the two triangles A_i and B_j . Two cases can occur when a triangle adjacent to B_j is added. If the triangle B_k in the middle figure is considered, then the previous bitangents continue to be valid. If the triangle B_k in the right figure is considered, then the vertex in A_i of the intersected bitangent can be connected with the vertex in B_k not shared by B_j and B_k to generate a new bitangent

Lemma 1 There is a path between any pair of obstacles in an unbounded environment with a finite number of obstacles.

Proof Let A and B two obstacles in the environment. From Proposition 2, between two non-overlapping polygons A and B in the free space there are at least two bitangents between them. If the remaining obstacles in the environment do not intersect one of the bitangents between A and B then a path has been found. If all bitangents between A and B are intersected then we proceed to connect A and B with the obstacles intersecting them. If one of those obstacles can be connected directly to A and B by bitangents that are not intersected by any other obstacle in the environment then a path has been found. Otherwise, the process has to be recursively applied to the set of obstacles intersecting the bitangents between A and B . From the previous set two obstacles are selected, one denoted as C that can be directly connected to A and one denoted as D that can be directly connected to B . Note that this is always possible because the environment is not bounded –single connected component–. Then we proceed to connect C and D following a similar approach to the one described above. This process can be completed in finite time since the number of obstacles (intersections) is finite. □

Theorem 1 The boundaries of all obstacles are visited in finite time.

Proof From Lemma 1, we have that for any pair of obstacles there is a path composed of bitangents and the boundary of the obstacles. Since the algorithm collectively explores all paths composed of bitangents in the environment then all obstacles are visited and circumnavigated. Since every obstacle is marked as visited when all its bitangents are traveled and the number of obstacles is finite then the algorithm completes the task in finite time. □

Lemma 2 Any point in the free space of an unbounded environment with a finite set of obstacles is seen by at least some vertex of an obstacle.

Proof Let x be a point in the free space of the environment, i.e. x is not in the interior of an obstacle. Let A be an obstacle in the environment and construct the set of segments S from the vertices of A to x . If one of those segments is not intersected by any other obstacle then x is visible by A . Otherwise, this procedure can be repeated with the vertices of the set of obstacles intersecting S until x is visible from a vertex of an obstacle in that set. This always holds because the environment is unbounded and it has a single connected component. Note that the number of obstacles (intersections) is finite. □

Theorem 2 *All points in the environment are sensed in finite time.*

Proof Since from Theorem 1 the algorithm collectively visits all vertices of the obstacles in finite time, and from Lemma 2 any point in the environment is seen by at least some vertex of an obstacle then all points in the environment are sensed in finite time. \square

5.1 The Case of Disc-shaped Robots

In this section, we show that it is possible to extend the results considering point robots to the case of disc-shaped robots. In the case of disc-shaped robots, we assume that *a robot can communicate with any other if there is a clear line of sight between them*. This assumption along with the results in [1] about robots' motion coordination without collision allow us to establish the following theorem.

Theorem 3 *If a solution exists to the problem of finding a trajectory to visit each visible reflex vertex¹ with at least one two-dimensional circular agent such that the agents do not collide with each other and the walls, and given the ability of the robots to communicate by a clear line of sight, then the strategy of traveling bitangents to move from one obstacle to another and moving the agents following the boundary of the obstacles guarantees covering the whole environment with the field of view of the robots.*

Proof Whether or not there is a trajectory without collision to visit each reflex vertex can be determined by the results in [1]. By Lemma 2 any point in the environment is seen by at least some reflex vertex, the only way that a point in the environment is not seen by an agent is that the circular body of another agent occludes the visibility of the first agent. But given that if an agent occludes the view of another then the agent causing the occlusion is able to detect the information missed by the other and communicate it to that agent, hence the whole environment is covered with the field of view of the robots. \square

Another interesting observation related to moving the agents following the boundary of the obstacles, it is that in [4], it has been proved that if the configuration space has several connected components then that motion strategy guarantees to see the largest possible region of the environment. In [4], a single agent explores the environment, the agent is a Differential Drive Robot, a nonholonomic system shaped as a disc, but the same result applies to holonomic disc-shaped robots as the ones used in this paper. In [4], it is assumed that the robot's sensor is located at the periphery of the robot. This sensor location

is the one that maximizes the area of the environment perceived by an omnidirectional sensor. However, for any other sensor location, for instance the center of the robot, the strategy of moving the circular agents in contact with the obstacles is the strategy that covers the largest possible area. This result is proved in Lemma 3.

Before we proceed to prove it, let us present the following remark about the accessibility of some regions in the environment.

Remark 1 An inaccessible region means that the robot itself cannot reach that portion of the environment because the robot is blocked, there is a bi-contact or multi-contact between the robot and the boundary of the environment. We say that there is an inaccessible region, when that region is not accessible at all and not only locally, meaning that the configuration space has more than one connected component.

Lemma 3 *If the configuration space has several connected components then moving the robots in contact with the obstacles is the motion strategy that covers the largest possible portion of the environment with the visibility regions of the omnidirectional sensors.*

Proof The free space in the environment can be divided into two regions, named accessible region R_a and inaccessible region R_{na} , such that $R_a \cap R_{na} = \emptyset$. An arc of circle determined by the robot's radius is the boundary between both regions. Refer to Fig. 7. It is clear that every ray of light emerging from any source $s \in R_a$ which touches R_{na} must cross the regions' boundary. For any given sensor location on the robot, the strategy of moving the robots in contact with the environment boundary makes the omnidirectional sensor to penetrate as deep as possible in the reachable region and hence covering the largest possible area of the inaccessible region. The result follows. \square

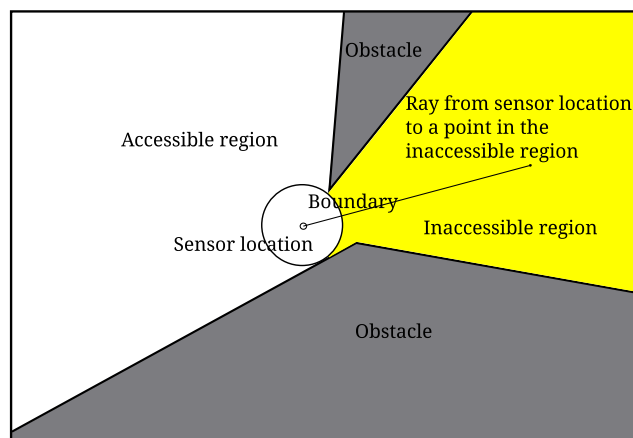


Fig. 7 Accessible and inaccessible regions

¹those with internal angle larger than π

To provide a complete algorithm to explore an unknown environment with disc-shaped robots requires to determine whether or not a trajectory connecting two given configurations of the agents exists such that the agents do not collide with each other and the walls. This last problem has been solved in [1]. If that solution is used then the exploration strategy proposed in this paper is complete. That is, it is able to cover in finite time the whole environment with the field of view of the robots' sensors or declare also in finite time that a solution does not exist.

Unfortunately, the algorithm proposed in [1] is exponential in the number of robots, making it practical only for a small number of robots. In order to deal with a large number of robots, and find collision-free trajectories to move the disc-shaped agents, we have implemented an adapted version of the method proposed in [2, 3]. The method presented in [2, 3] does not guarantee to find collision-free paths, however in many instances, it is able to move the robots without collision and it is able to deal with hundreds of agents moving simultaneously. In the next section, we present simulation results of the exploration task with both point and disc-shaped robots.

Our approach can be extended to explore environments with an external boundary. In the proposed methodology, the robots travel to the obstacles following bitangents between their vertices. When the environment has an external boundary and contains interior obstacles, the following complication might appear. If the robots start exploring the environment lying on the external boundary and no bitangent between the boundary and any internal object exists, then the robots are not able to travel to the internal obstacles. To deal with this type of environments the robots need to be able to detect convex corners. If the robots are able to do so, then they can travel from the external boundary to an internal obstacle merely by reaching a convex corner. Furthermore, if the robots do not start the exploration over the external boundary, then the proposed method holds as it is. For environments only containing an external boundary and no internal obstacles, that is simply connected, in [4], it has been proved that following the external boundary is enough to explore the entire environment provided that the sensor's range is larger than the largest distance between two visible points on the boundary.

6 Simulation Experiments

In this section, we present simulation experiments for both cases: distinguishable and indistinguishable obstacles. We analyze the exploration task in terms of the distance traveled by the robots and the time needed to explore the whole environment. In the simulations, we have varied the number

of robot and the number of obstacles. Also, for the case of distinguishable obstacles, we have varied the backtracking level h .

6.1 Exploring Distinguishable Obstacles with Point Robots

In the case of distinguishable obstacles, we assume that each obstacle in the environment has a different color. Figure 8 shows the maps used to perform the first set of experiments. For this case, we consider a map with many bitangents (see Fig. 8a) and another with few bitangents (see Fig. 8b).

Figure 9 shows the results of the first set of experiments assuming distinguishable obstacles. The graphs show the cumulative distance traveled by all robots at the end of the simulation. The results are clustered by the number of robots used to perform the exploration. Each color indicates the level of backtracking during the execution when the robots travel using the bitangents.

It is interesting to note, that in the map with many bitangents (see Fig. 8a) the distance traveled by all robots is smaller than the distance traveled in the map with few bitangents (see Fig. 8b). That is because the perimeters of the obstacles in the case of few bitangents is much larger than in the map with many bitangents.

In Fig. 9, we can observe that using a backtracking level of 1 produces the smallest cumulative distance. This behavior appears because most of the obstacles in our maps can be reached from any other obstacle visiting only one additional obstacle. Thus, increasing the level of backtracking also increases the number of visited obstacles by more than one team.

For the map shown in Fig. 10, the backtracking does not help to reduce the traveled distance. In this map, each obstacle in the environment is connected with at most two other obstacles using bitangents. Furthermore, the number of levels in the tree-based representation of the environment is the same than the number of obstacles. Thus, each time that an obstacle is circumnavigated, only a new obstacle

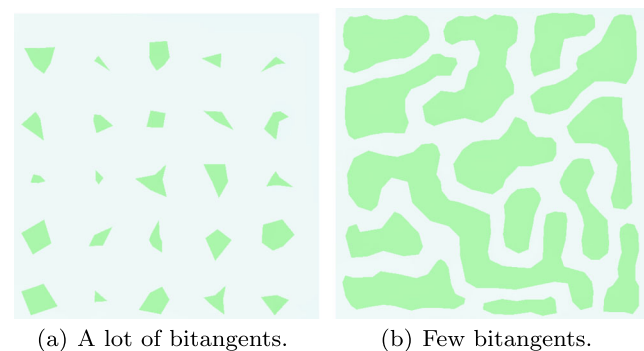
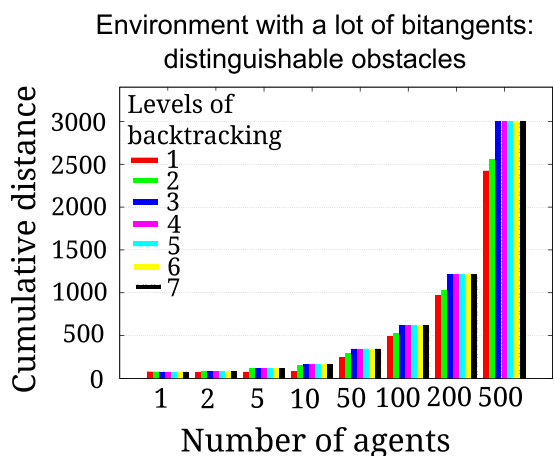
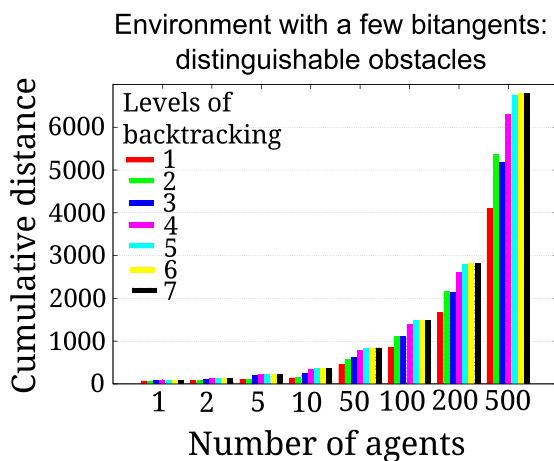


Fig. 8 Environments



(a) Cumulative distance for the map shown in Fig 8(a).



(b) Cumulative distance for the map shown in Fig 8(b).

Fig. 9 Cumulative distances assuming distinguishable obstacles in the maps of Fig. 8

is discovered, hence the backtracking does not help to distribute the robots among the obstacles. In Fig. 11, the goal of the robots is to explore the map of the world, starting in America. At the beginning of the exploration, some obstacles are occluded by other big obstacles. In this case, a backtracking level of 2 or 3 produces the smallest distance traveled by all agents.

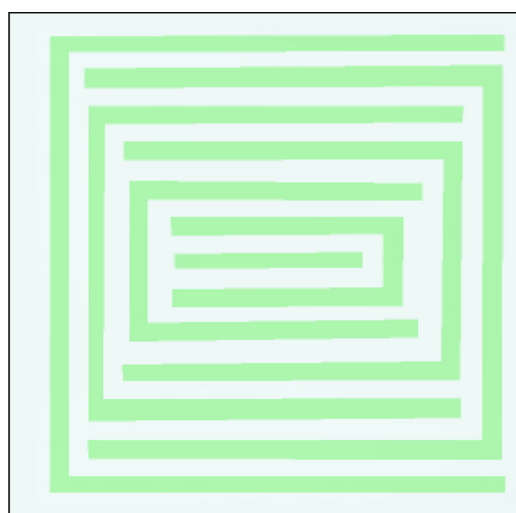
The execution times for the previous maps are presented in Fig. 12. The execution time corresponds to the time from the beginning of the exploration to the time the last robot ends its task.

Figure 12a shows the execution time, according to the number of robots and the level of backtracking for the map with many bitangents in Fig. 8a. Figure 12b shows the same quantities for the map with few bitangents in Fig. 8b. Similarly, Fig. 12c presents the execution time for the map

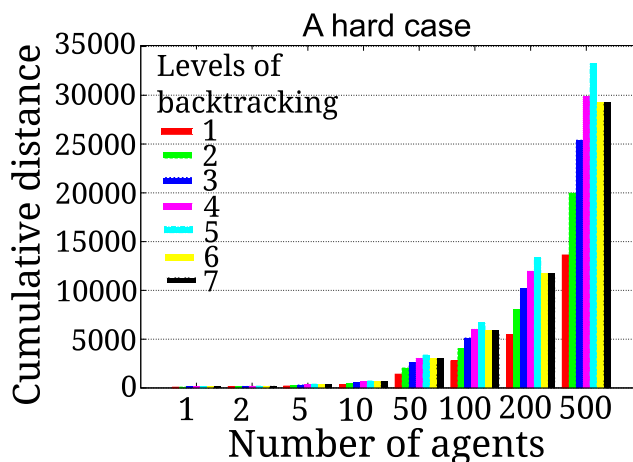
in Figs. 10a and 12d presents the execution time for the map in Fig. 11a.

In general, the execution time decreases as the number of robots to perform the task increases. An exception is the map shown in Fig. 10a, in which increasing the number of robots does not help to reduce the execution time. This happens because the number of levels in the tree representing the environment is the same than the number of obstacles. Thus, each time that an obstacle is circumnavigated, only a new obstacle is discovered and it is not possible to distribute robots among the obstacles.

Now we discuss the impact of the level of backtracking in the performance of the exploration strategy. For the maps shown in Fig. 8a and b, the best execution time is obtained

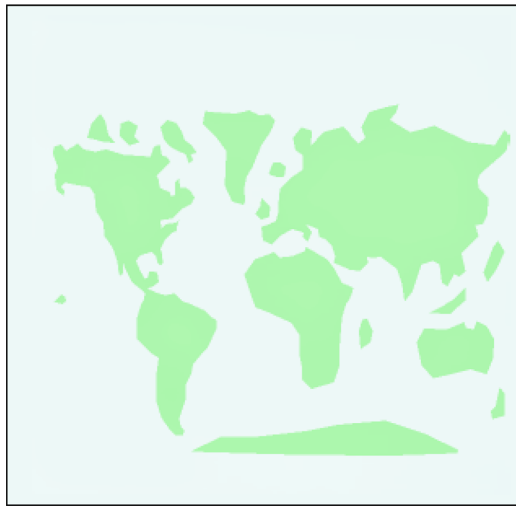


(a) Map.

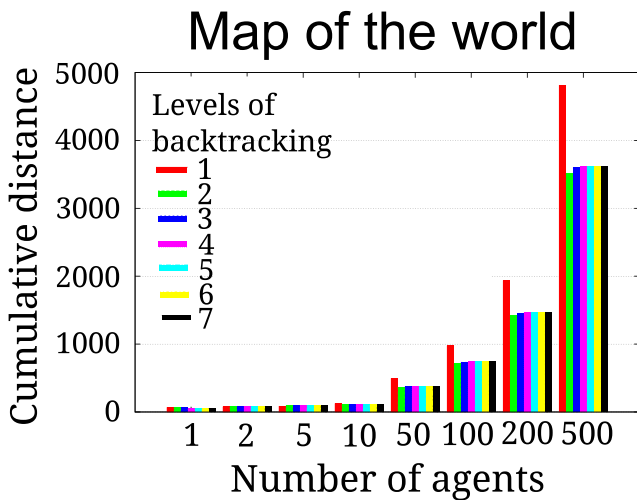


(b) Cumulative distance.

Fig. 10 An environment where backtracking does not help. Fig. 10a shows the map and Fig. 10b the cumulative distance needed to explore that map



(a) Map.



(b) Cumulative distance.

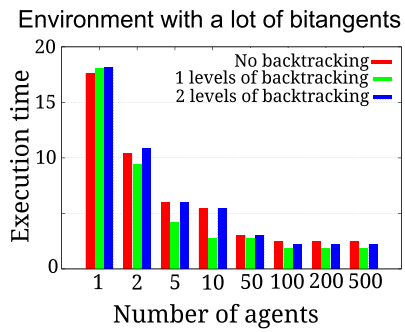
Fig. 11 An environment where backtracking is useful. Figure 10a shows the map and Fig. 10b the cumulative distance needed to explore that map

with a backtracking level of 1 or 2 depending on the number of robots. For the maps shown in Figs. 11a and 10a, the level of backtracking with the best execution time is 0 (no backtracking).

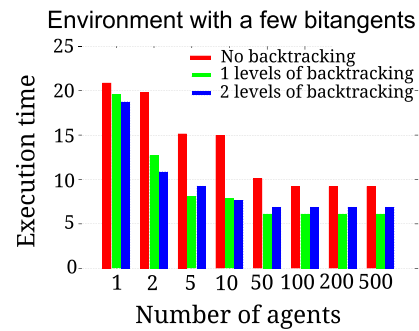
A snapshot of the exploration task in an environment with 144 distinguishable obstacles and 576 robots in shown in Fig. 13. In the figure the robots are represented by blue discs.

6.2 Exploring Indistinguishable Obstacles with Point Robots

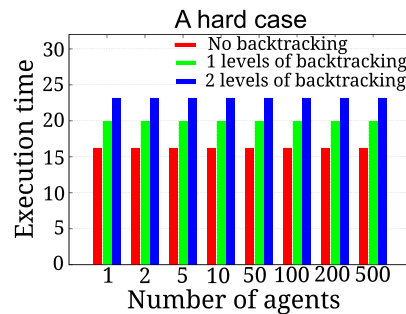
Figures 14 and 15 show the results of the experiments assuming indistinguishable obstacles for the maps in Fig. 8a



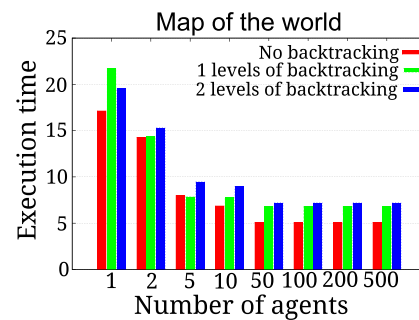
(a) Execution time for map in Fig. 8(a).



(b) Execution time for map in Fig. 8(b).



(c) Execution time for map in Fig. 10(a).



(d) Execution time for map in Fig. 11(a).

Fig. 12 Execution times for the maps in Figs. 8a, b, 10a and 11a

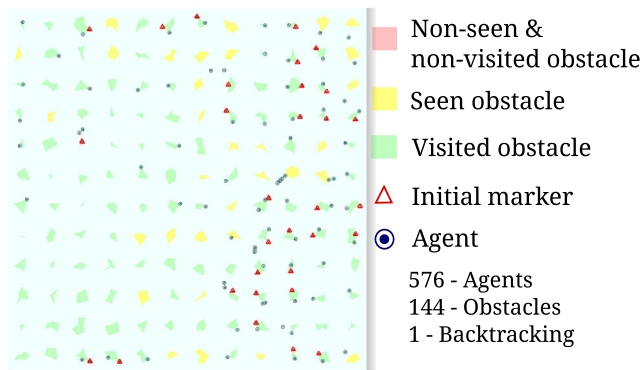
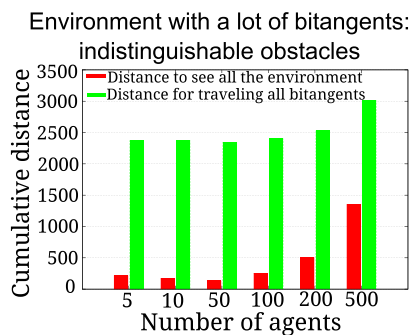
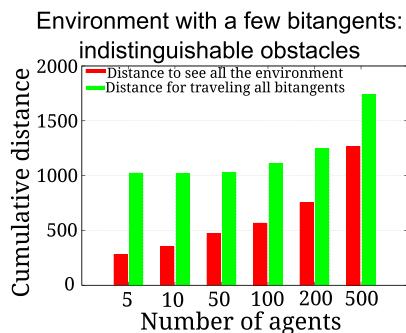


Fig. 13 Example of the exploration strategy for distinguishable obstacles

and b. For this case, the plots in Fig. 14 show the cumulative distance traveled by all robots when all obstacles have been visited for the first time and the cumulative distance after all bitangents have been traveled. Figure 15 shows the execution time of the algorithm for completing the exploration. Note that the distance and time needed to travel all bitangents in order to be sure that the whole environment has been explored – every robot has finished its task – is different to the time and distance needed to actually explore the obstacles by circumnavigating them. A robot



(a) Cumulative distance needed to explore the map shown in Fig 8(a).



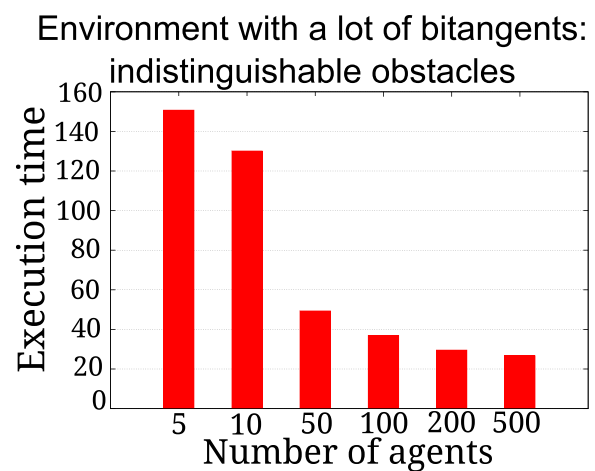
(b) Cumulative distance needed to explore the map shown in Fig 8(b).

Fig. 14 Cumulative distance assuming indistinguishable obstacles for the maps in Fig. 8a and b

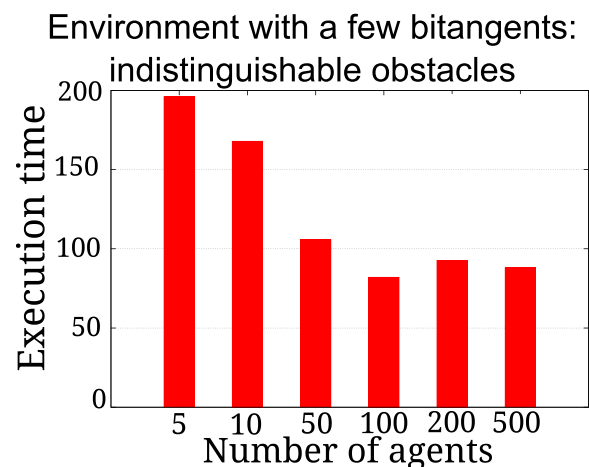
does not know that the task is finished until all of its assigned bitangents are traveled. When there are no robots with assigned bitangents to be visited, the task is finished.

From Fig. 14, we can observe that the number of bitangents in the environment has a strong influence in the algorithm’s performance. The cumulative distance for visiting all obstacles for the first time is significantly higher in Fig. 14a, where more bitangents are present with respect to the number of obstacles in the environment than in Fig. 14b. Since most of the obstacles in the map of Fig. 8a can be reached from any other obstacle visiting only one additional obstacle then the execution time shown in Fig. 15 is lower compare to the one of the map in Fig. 8b.

Regarding the distance traveled by all robots to transit all bitangents, in both cases, it increases as the number of



(a) Execution time needed to explore the map shown in Fig 8(a).



(b) Execution time needed to explore the map shown in Fig 8(b).

Fig. 15 Execution time assuming indistinguishable obstacles for the maps in Fig. 8a and b

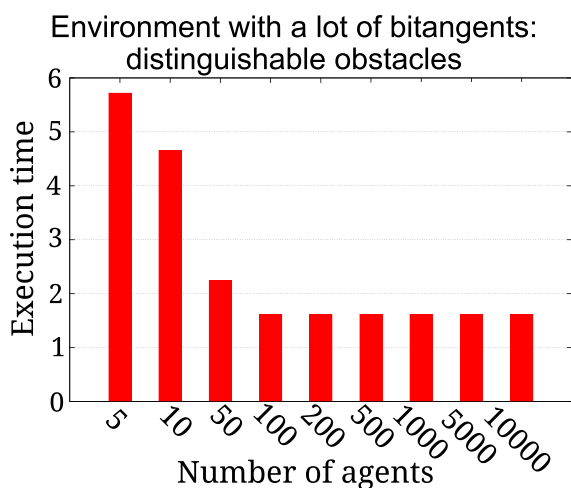
robots increases. However, based on the results shown in Fig. 15, we can observe that the time taken for traveling all bitangents in general decreases as the number of robots increases.

The experiments have also shown that in the case of indistinguishable obstacles, the obstacles are circumnavigated faster than visiting all bitangents in the environment. Recall that the distance and time needed to travel all bitangents to be sure that every robot has finished its task is different to actually circumnavigate every obstacle and explore the whole environment. Thus, for search applications in most cases it would not be necessary to visit the entire set of bitangents in the environment in order to find the target.

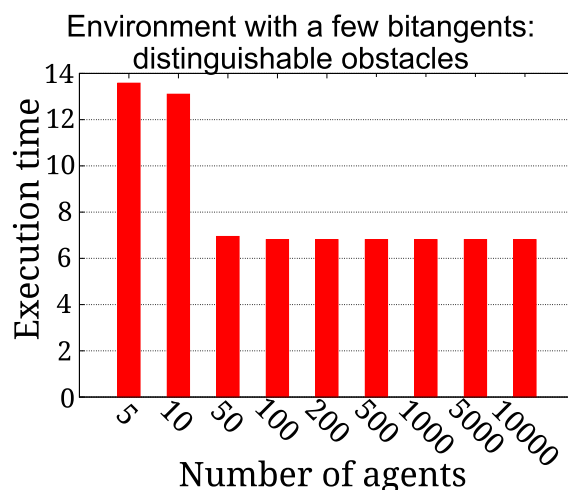
Figure 16 shows a comparison of the execution time between distinguishable and indistinguishable obstacles for the environment shown in Fig. 8a. Figure 17 shows the same comparison for the environment shown in Fig. 8b. In general it takes longer to explore an environment with indistinguishable obstacles.

6.3 Collision Avoidance for Disc-shaped Robots and Local Trajectory Deformation using the Velocity Obstacle Approach for Exploring the Environment

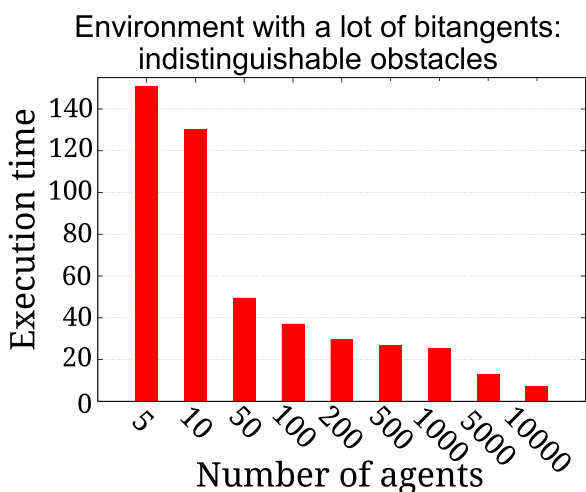
In the case of disc-shaped robots to avoid collisions among the robots, we use the velocity obstacle approach [2, 3]. The robots still explore the environment following the



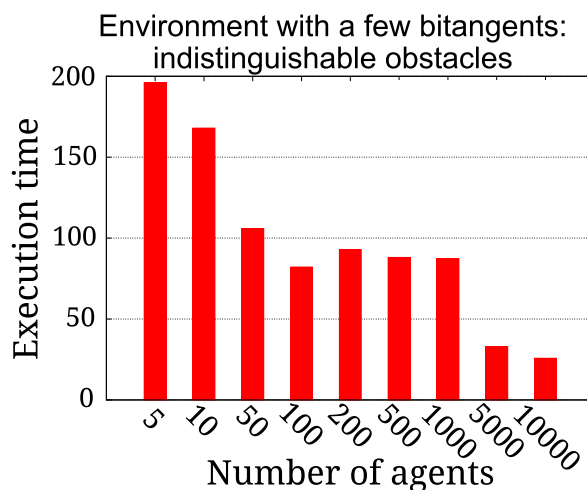
(a) Execution time considering distinguishable obstacles for the map shown in Fig 8(a).



(a) Execution time considering distinguishable obstacles for the map shown in Fig 8(b).



(b) Execution time considering indistinguishable obstacles for the map shown in Fig 8(a).



(b) Execution time considering indistinguishable obstacles for the map shown in Fig 8(b).

Fig. 16 Comparison of execution times between distinguishable and indistinguishable obstacles for the map shown in Fig. 8a

Fig. 17 Comparison of execution times between distinguishable and indistinguishable obstacles for the map shown in Fig. 8b

strategies presented in Sections 3 and 4, but the following modifications are included.

1. The velocity obstacle method is extended to locally avoid collision with static obstacles. To do so, for each visible reflex vertex (those with internal angle larger than π), a circle is calculated centered in the reflex vertex. The velocity cone is computed using tangent lines to the extremal circles (see Fig. 18). The velocity vector applied to a robot and computed using the velocity obstacle method produces a collision-free motion and must lead the robot to see the destination vertex at all time. Note that since the velocity obstacle method is only activated in robots moving without contact with obstacles (following bitangents), most of the times these trajectories are collision-free. However, when a robot locally deforms its trajectory to avoid collision with other moving robot, the robot might also need to locally avoid collision with a static obstacle.
2. Robots activate the reciprocal velocity obstacle when they do not make contact with the obstacles in the environment to avoid collisions with other robots. A robot that does not move in contact with an obstacle uses a *no reciprocal* velocity obstacle to avoid collision with a robot circumnavigating an obstacle moving in contact with it.
3. In the case when the robots backtrack and gather at some vertex of an obstacle in order to share information about the obstacles to be visited; first, they arrive to the vertex and then the robots move to another vertex,

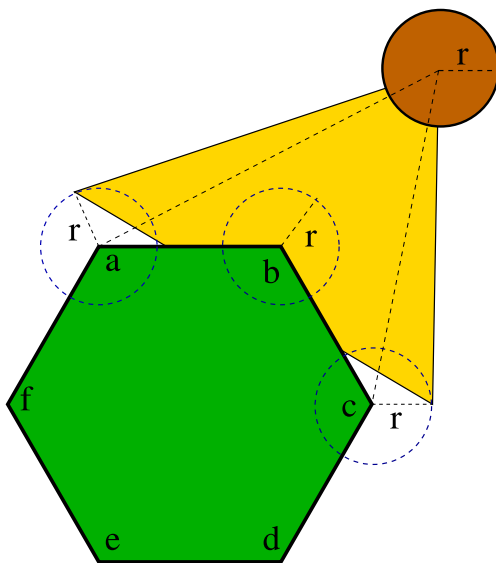


Fig. 18 Velocity cone (yellow region) for a robot of radius R . The disc-shaped robot senses vertices a , b and c of the hexagonal obstacle with its omnidirectional sensor. The tangent lines to the dashed circles at vertices a and c are used to construct the velocity cone

where they make a line formation, to avoid blocking the departed vertex.

4. When the robots move in contact with the obstacles following the boundary, they always move in the same counterclockwise direction and at the same velocity to avoid collisions among them.

Figure 19 shows three snapshots of a simulation experiment using the reciprocal velocity obstacle approach. The robots are represented with the yellow discs and the obstacles with the black polygons. The local deformations of the trajectories to avoid collisions among the disc-shaped robots and also with the static obstacles in the environment are shown in blue.

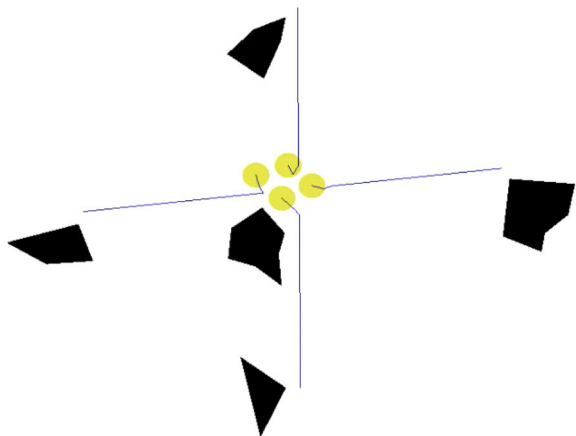
Figure 20 shows a snapshot of an exploration task with disc-shaped robots, this experiment has an environment with 81 distinguishable obstacles and 50 robots. The method presented in Section 3 together with the adaptations presented in the list above and the reciprocal velocity obstacle method [2, 3] are used to obtain this result.

Figure 21 shows an environment with an external boundary and 5 internal obstacles. As we have mentioned before, to deal with this type of environments the robots need to be able to detect convex corners. If the robots are able to do so, then they can travel from the external boundary to an internal obstacle merely by reaching a convex corner.

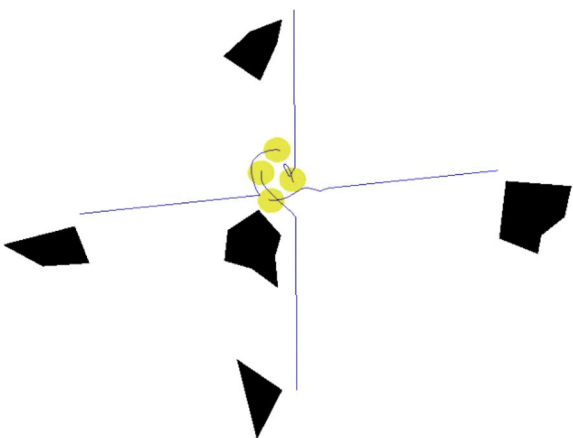
In the case of distinguishable obstacles, the exploration terminates when all obstacles have been visited and circumnavigated. In the case of indistinguishable obstacles, the exploration finishes when all robots have traveled all bitangents that they have assigned. Note that in both cases, it is possible that more than one robot visits the same obstacles and some redundancy appears. Thus, it may happen that for a given number of obstacles, first the exploration time decreases as the number of robots increases, and later on, once a saturation value is reached, the exploration time increases as the number of robot increases since it takes more time to move more robots, which may visit areas already explored by other agents. This behavior usually appears in the case of disc-shaped robots since they are also coordinated to avoid collisions between them. For point robots, it may happen that as the number of robots increases the exploration time decreases until a constant value is reached. That occurs because point robots do not require to avoid collisions between them.

6.4 Toward Finite Sensor Range and Multimedia Material

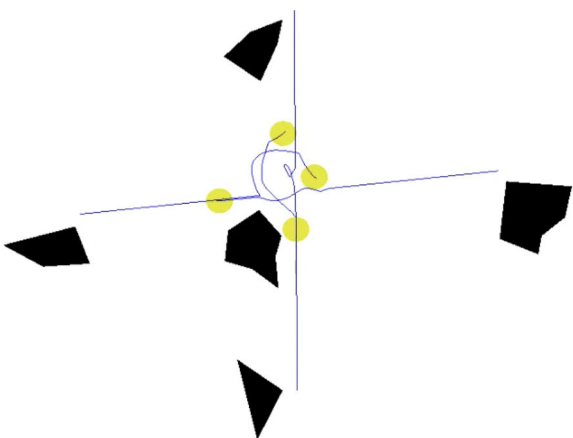
For dealing with the case of limited sensor range, if there is a path to visit all obstacles in the environment in which, as the robot travels, the current visited obstacle and the next obstacle to be visited are closer than the maximum sensor's range, then our proposed algorithm can be used to



(a) Robots start deforming their trajectories to avoid collision.



(b) The robots keep avoiding collision with other robots.



(c) The robots start coming back to the bitangents between the obstacles.

Fig. 19 Trajectories are locally deformed to avoid collisions among disc-shaped robots



Fig. 20 Exploration of an environment with disc-shaped robots. This experiment is done with 81 distinguishable obstacles and 50 robots

circumnavigate all obstacles in environments larger than the maximum sensor’s range of a bounded sensor. A snapshot of a simulation of that case is shown in Fig. 22. In that figure the sensor’s range is depicted with a dashed circle.

In that case, however, it is not guaranteed to cover all free space with the field of view of the sensors. In the

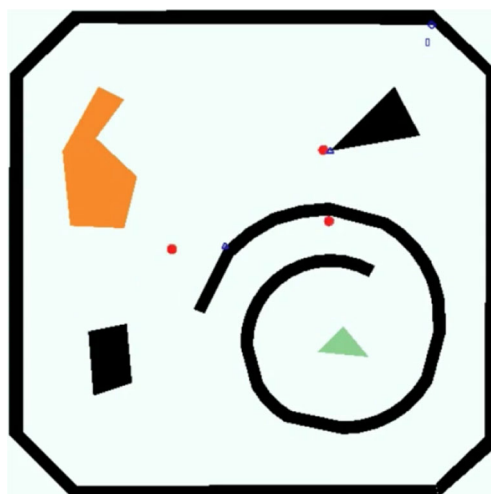


Fig. 21 Exploration of an environment with an external boundary. This experiment is done with 5 internal distinguishable obstacles and 3 disc-shaped robots

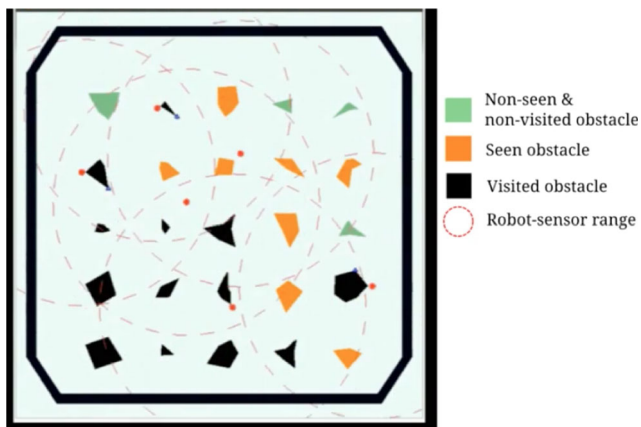


Fig. 22 Exploration of an environment with robots equipped with bounded range sensors

multimedia material we have added a video showing a simulation experiment of this case.

We think that our approach can be extended to deal with bounded environments which are larger than the maximum sensor's range and a path like the one described in the previous paragraph does not exist or is required to guarantee covering all free space with the sensor's field of view. In that case, it should be possible to follow an approach similar to the one presented in [17]. In that approach, some of the robots are static and serve themselves as antennas that maintain connectivity in networks of robots to cover the whole environment. Note that in this work, we consider multiple connected environments while in [17] the authors considered simply connected environments. Hence, a combination between wall following, traveling bitangents and static robots will be needed. Consequently, a detailed analysis is required to extend this approach to that case. We left that analysis for future work.

In the multimedia material of the paper, we have added a video showing simulation results. In that video, we have included seven experiments. In the first experiment, we show the case of distinguishable obstacles with backtracking equal to $h = 1$, there are 200 robots and 225 obstacles. In the second experiment, we compare the behavior of our algorithms for the cases of distinguishable and indistinguishable obstacles. In this experiment, there are 144 obstacles and 576 robots. This experiment clearly shows that knowing the identity of the obstacles (distinguishable obstacles) allows the robots to perform a faster exploration of the environment. In the case of indistinguishable obstacles, the robots know when the exploration is finished because all bitangents between the obstacles are visited, each robot has some assigned bitangents which are kept ordered by angular value. When a robot finishes of traveling the bitangents that the robot has assigned it remains

motionless. So, the extra time that the robots remain in the obstacles is because the same obstacle can be visited several times, since several bitangents might lead to the same obstacle.

The third experiment corresponds to the results in Fig. 19 and shows the local deformations of the trajectories to avoid collisions among the disc-shaped robots and also with the static obstacles using the reciprocal velocity obstacle method. The fourth experiment presents the exploration of an environment with disc-shaped robots, for the case of distinguishable obstacles, shown in Fig. 20. In the experiment, there are 81 obstacles and 50 robots. The proposed approach combines the method presented in Section 3, including the adaptations listed in Section 6.3, and the reciprocal velocity obstacle method. The fifth experiment shows the case of non-convex obstacles with 16 distinguishable obstacles and 40 robots. The sixth experiment presents the exploration of an environment with external boundary. The robots can travel from the external boundary to an internal obstacle merely by reaching a convex corner. The seventh experiment shows an example with robots having bounded range sensors and we compare the exploration with a team of robots equipped with unbounded range sensors.

7 Conclusions

In this paper, we have presented a complete algorithm for exploration of unknown environments containing disjoint obstacles with multiple robots. The approach is distributed. We have considered two main cases, one in which the obstacles are distinguishable, i.e., each obstacle is uniquely identifiable, and in the second case the obstacles are not distinguishable.

The main contributions of this work are the following: 1) The algorithms guarantee exploring the whole environment or the largest possible portion of it in finite time even though the robots are not capable of building an exact map of the environment, they cannot estimate their positions and each robot does not have full information about the part of the environment explored by other robots. 2) The method only requires limited communication between the robots. 3) We have combined our proposed approach with the velocity obstacle method to explore the environment with disc-shaped robots that are able to avoid collisions among themselves and also with the static obstacles in the environment, this makes feasible an implementation of our method in physical robots. 4) We have proposed an exploration strategy such that even if the configuration space has several connected components this strategy guarantees covering the largest possible portion of the environment with the visibility regions of omnidirectional sensors.

5) The approach scales well to hundreds of robots and obstacles.

Experiments have shown that the obstacles are circumnavigated faster than visiting all bitangents in the environment. Thus, for search applications in most cases it would not be necessary to visit the entire set of bitangents in the environment in order to find the target.

For future work, we want to implement our method in real robots and extend the approach to find antagonistic moving agents. Finally, as we have mentioned before, we would like to extend the approach presented in this paper to the case of a team of robots equipped with bounded range sensors.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

- Schwartz, J.T., Sharir, M.: On the piano movers' problem: III. Coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers. *Int. J. Robot. Res.* **2**(3), 46–75 (1983)
- van den Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1928–1935 (2008)
- Snape, J., van den Berg, J., Guy, S.J., Manocha, D.: The hybrid reciprocal velocity obstacle. *IEEE Trans. Robot.* **27**(4), 696–706 (2011)
- Laguna, G., Murrieta-Cid, R., Becerra, H.M., Lopez-Padilla, R., LaValle, S.M.: Exploration of an unknown environment with a differential drive disc robot. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 2527–2533 (2014)
- Burgard, W., Moors, M., Stachniss, C., Schneider, F.: Coordinated Multi-Robot exploration. *IEEE Trans. Robot.* **21**(3), 376–386 (2005)
- Howard, A., Mataric, M., Sukhatme, G.: Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem. In: *Distributed Autonomous Robotic Systems*, vol. 5, pp. 299–308 (2002)
- Cortés, J., Martínez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. *IEEE Trans. Robot. Autom.* **20**(2), 243–255 (2004)
- Taylor, C.J., Kriegman, D.: Vision-based motion planning and exploration algorithms for mobile robots. *IEEE Trans. Robot. Autom.* **14**(3), 417–426 (1998)
- Sarmiento, A., Murrieta-Cid, R., Hutchinson, S.: Planning Expected-time Optimal Paths for Searching Known Environments. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 872–878 (2004)
- Tovar, B., Murrieta-Cid, R., LaValle, S.M.: Distance-Optimal Navigation in an unknown environment without sensing distances. *IEEE Trans. Robot.* **23**(4), 506–518 (2007)
- Park, H., Hutchinson, S.: An efficient algorithm for fault-tolerant rendezvous of multi-robot systems with controllable sensing range. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 358–365 (2016)
- Yamauchi, B.: A frontier-based approach for autonomous exploration. In: *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 146–151 (1997)
- Rekletis, I.M., Dudek, G., Milos, E.E.: Graph-based exploration using multiple robots. In: *Distributed Autonomous Robotic Systems*, vol. 4, pp. 241–250 (2000)
- Wurm, K.M., Stachniss, C., Burgard, W.: Coordinated multi-robot exploration using a segmentation of the environment. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1160–1165 (2008)
- Matignon, L., Laurent, J., Abdel-Ilan, M.: Distributed value functions for multi-robot exploration. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1544–1550 (2012)
- Amigoni, F.: Experimental evaluation of some exploration strategies for mobile robots. In: *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 2818–2823 (2008)
- Lee, S.K., Fekete, S., McLurkin, J.: Structured triangulation in multi-robot systems Coverage, patrolling, Voronoi partitions, and geodesic centers. *Int. J. Robot. Res.* **35**(10), 1234–1260 (2016)
- Ng, J.: Thomas bräunl. Performance Comparison of Bug Navigation Algorithms. *J. Intell. Robot. Syst.* **50**, 73–84 (2007)
- Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using velocity obstacles. *Int. J. Robot. Res.* **17**(7), 760–772 (1998)
- Kluge, B.: Recursive agent modeling with probabilistic velocity obstacles for mobile robot navigation among humans. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 376–380 (2003)
- van den Berg, J., Guy, S., Lin, M., Manocha, D.: Reciprocal n-Body Collision Avoidance. *Robotics Research*. In: Pradaliar, C., Siegwart, R., Hirzinger, G. (eds.) *Springer Tracts in Advanced Robotics*, vol. 70, pp. 3–19. Springer (2011)
- Bravo, L., Ruiz, U., Murrieta-Cid, R., Aguilar, L., Chavez, E.: A distributed exploration algorithm for unknown environments with multiple obstacles by multiple robots. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4460–4466 (2017)
- Cormen, T.H., Leiserson, C., Rivest, R.L., Stein, C. *Introduction to Algorithms*, 2nd edn. MIT Press, Cambridge (2002)
- Kleiner, A., Prediger, J., Nebel, B.: RFID Technology-based exploration and SLAM for search and rescue. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4054–4059 (2006)

Gabriel Aguilar received the B.S. degree in electronic engineering from Morelia Institute of Technology, in 1999 and the M.S. in electronics from University of the Americas, Puebla, Mexico, in 2001. He has been test equipment design engineer at Sony Tijuana, instructor at La Piedad Institute of Technology and at the Advance High School and Higher Education System of the State of Guanajuato, Mexico. He is currently (2018) pursuing a Ph.D. degree in Computer Science at Centro de Investigacion en Matematicas, CIMAT, Guanajuato Mexico. His research interests include robotics and robot motion planning.

Luis Bravo received the B.S. in Computer Science from Universidad de Guanajuato, Mexico, in 2018. Currently (2018), he is pursuing a M.S. in Computer Science and Industrial Mathematics at Centro de Investigacion en Matematicas, Guanajuato, Mexico. His research interests include design and analysis of algorithms, swarm robotics, discrete applied mathematics, and data science.

Ubaldo Ruiz received the Ph.D. degree in Computer Science from Centro de Investigacion en Matematicas (CIMAT), Guanajuato, Mexico in 2013. In 2013-2014, he was a Postdoctoral fellow in the Computer Science Department of the University of Minnesota. Since 2014, he is a CONACYT Research Fellow working at Centro de Investigacion Cientifica y de Educacion Superior de Ensenada (CICESE), Ensenada, Mexico. He is a member of Sistema Nacional de Investigadores in Mexico. His research interests are: Robotics, Differential Games, Optimal Control and Motion Planning.

Rafael Murrieta-Cid received the B.S. degree in physics engineering from the Monterrey Institute of Technology and Higher Education, Monterrey, Mexico, 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 Group (RIA) 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 Researcher with the Beckman Institute and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign (UIUC), Urbana, IL, USA. Since 2006, he has been a senior research scientist at Centro de Investigacion en Matematicas (CIMAT), Guanajuato, Mexico, and he is a member of the Mexican National System of Researchers at level 2. In 2016, he was on a sabbatical leave at UIUC. His research interests include robotics, robot motion planning and control theory.

Edgar Chavez is a full professor at Centro de Investigacion Cientifica y de Educacion Superior de Ensenada (CICESE) in northern Mexico. His research interests include data science and algorithms. He is an expert in metric access methods, a member of the Sistema Nacional de Investigadores in Mexico, and have published more than 100 technical papers.