2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 24–28, 2017, Vancouver, BC, Canada

A distributed exploration algorithm for unknown environments with multiple obstacles by multiple robots

Luis Bravo, Ubaldo Ruiz, Rafael Murrieta-Cid, Gabriel Aguilar and Edgar Chavez

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 two main cases. In the first one, 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. 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 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. In both cases distinguishable and indistinguishable obstacles, the robots communicate only at rendezvous. 3) The algorithm scales well to hundreds of robots and obstacles. We tested in several simulations the performance of our algorithms for both cases, in terms of the distance traveled by the robots.**

I. INTRODUCTION

This work deals with the problem of exploring an unknown environment composed by disjoint obstacles with multiple robots. We propose a distributed method for robots with limited communication capabilities, a robot does not have full information about the part of the environment explored by other robots.

A. Previous work

This work is related to the problems of exploration and mapping [1], coverage [2], [3], object search [4], [5], [6] and robot rendezvous [7]. The problem of exploring an unknown environment for searching one or more recognizable targets is considered in [4]. That method assumes limited sensing capabilities of the robot and the environment is represented in the so-called boundary place graph, which records the set of landmarks. The work presented in [6] proposed the Gap Navigation Tree (GNT) for navigation of a point robot without using the robot coordinates. The GNT can be considered as a topological map. The GNT differs from previous approaches in that it is a local representation, defined for the current position of the robot, rather than a global one. The work in [6] 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 [8] extends the work in [6] to a disc shaped differential drive robot, that method guarantees exploring the whole environment or the largest possible region of it. The disc 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. The work in [5] addresses the problem of continuous sensing for finding an object, whose unknown location is characterized by a probability density function.

Some exploration strategies use frontier-based exploration, originally proposed by Yamauchi in [9]. In frontier-based exploration, the robot goes to the imaginary line that divides the known and unknown parts of the environment. Some works have proposed multi-robot exploration and mapping [10], [1]. In [1], 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 [10], the authors proposed 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. In [11], the authors propose a method for multi-robot exploration based on Decentralized Markov Decision Process (Dec-MDP).

In [12], the authors proposed distributed algorithms for the construction of a triangulation using a multi-robot system. The authors apply the approach to exploration, coverage and surveillance by a swarm of robots with limited individual capabilities. Some important differences between this paper and the one in [12] are the following: In [12] a triangulation to cover the environment is done over the free space, it is done over the obstacles. In [12] the robots are used as nodes of the triangulation to cover the free space while in this work the triangulation is required to prove that all obstacles shall be visited by the robots. In this work, we consider multiple connected environments while in [12] the authors considered simply connected environments.

In this paper, the robots follow the obstacles' boundaries, this makes this work related somehow to bug algorithms. In [13], the authors compared the performance of several bug algorithms in terms of different criteria, for instance the distance traveled to reach a goal. However, in [13] the authors did not analyze the performance of multi-robot systems for the task of covering an environment. The work in [13] neither studied the effect of limited communication among the robots

Luis Bravo, Rafael Murrieta-Cid and Gabriel Aguilar are with CIMAT, Guanajuato, Mexico. {luixbravo, murrieta, gabriel.aguilar}@cimat.mx

Ubaldo Ruiz and Edgar Chavez are with CICESE, Baja California, Mexico. {uruiz, elchavez}@cicese.mx

This research was partially funded by CONACYT grants no. 220796 and no. 264896. Ubaldo Ruiz is a CONACYT Research Fellow. He was financially supported by Consejo Nacional de Ciencia y Tecnología (CONACYT), México.

978-1-5386-2682-5/17/$31.00 ©2017 IEEE

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.

B. 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 two different settings:

- The obstacles are *distinguishable*. In this case, the robots can only communicate at rendezvous. We made experiments to find tradeoffs in terms of the distance traveled by all robots when they follow two strategies: 1) The robots gather during the exploration to share information about the remaining unexplored obstacles avoiding redundant assignations. 2) The robots explore the environment without sharing information.
- The obstacles are *indistinguishable*. In this case, the robots also can only communicate at rendezvous.

II. PROBLEM FORMULATION

A team of robots is moving in an unbounded unknown environment E with a finite number of polygonal obstacles. The robots are modeled as points in E. Let ∂E denote the boundary of obstacles. All robots are initially located at some vertex in ∂E. The goal is to navigate the entire environment and distribute this task among the elements of the team. Each robot has an abstract sensor that is able to detect and track discontinuities in *depth information* when it moves in contact with ∂E. First, we assume that the robots are able to group those discontinuities by the obstacle that generates them, and later on, we consider the case when this information is not available. To explore the environment, the robots make use of the bitangents. The robots have no initial knowledge of E and they are no capable of building an exact map of the environment. They also lack of sensors that might be used to estimate their positions in E. We assume that each robot has an ordered unique identifier. 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 presume that the robots are equipped with sensors that are able to detect bitangents, for example, a laser-range finder measuring depth-distance discontinuities. For the indistinguishable case, we only assume that the robots are able to detect bitangents.

III. DISTINGUISHABLE OBSTACLES

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

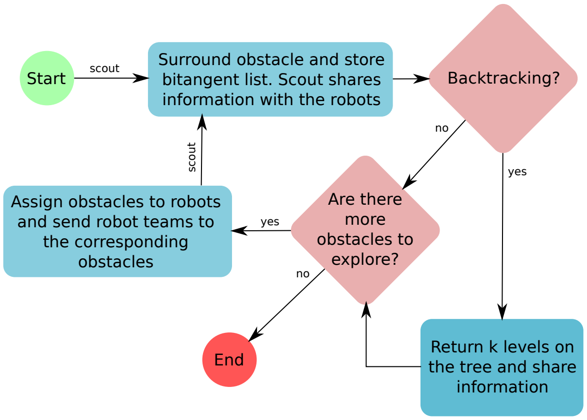

Start
scout
Surround obstacle and store bitangent list. Scout shares information with the robots
Backtracking?
no
yes
scout
Assign obstacles to robots and send robot teams to the corresponding obstacles
yes
Are there more obstacles to explore?
no
End
Return k levels on the tree and share information

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

A. General setup

Suppose there are n robots 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 $i = 1, \ldots, n$ and each obstacle as O_j where $j = 1, \ldots, m$. Recall that all robots are initially located at the vertex of one obstacle.

B. Obstacle exploration

The robot with the lowest 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 the sequence of bitangents detected by its sensors in angular order of appearance. 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 order. 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 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 its marker.

C. Obstacle assignation

To visit the unexplored obstacles in D_i the robots follow the next 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 $l < |D_i^{ne}|$. The remaining l robots are equally distributed among the unexplored obstacles. 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.

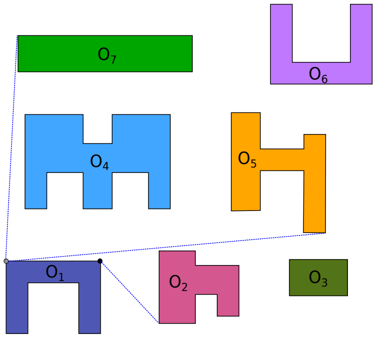


Fig. 2. An environment with distinguishable obstacles. The robots compute the obstacles that are visible from their initial location (black circle). One robot sourrounds O_1 in counter-clockwise direction detecting new obstacles in E. The gray circle indicates the locations where a new obstacle is detected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

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.

D. Representation of the strategy

A graphical description of the exploration strategy is shown in Figure 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 member after exploring the obstacle. The childs 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 III-F.

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

F. 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 completed explored thus it is identified 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 $D_i = D_1$.

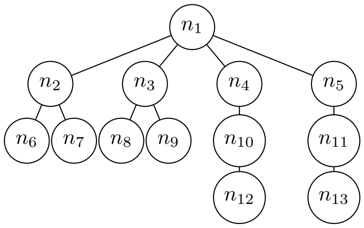


Fig. 3. An example of the tree representation of the strategy to explore environments with distinguishable obstacles. The node information is described in Table I.

Node	Explored	Robots	Queue
n_1	O_1	$r_{1,\dots,6}$	$D_i = \{O_1^e, O_2, O_4, O_5, O_7\}$
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	$D_4 = \{O_1^e, O_2^e, O_4^e, O_5^e, O_7^e, O_6^e, O_3^e\}$

TABLE I

NODE INFORMATION FOR THE GRAPH IN FIGURE 3.

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 I. In Table I, 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.

IV. INDISTINGUISHABLE OBSTACLES

In this section, we present a strategy for solve the case of non-uniquely identifiable obstacles.

A. General setup and markers

Suppose there are n robots, 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 the bitangents in the angular order in which they appear at each vertex. Note that in this case, it is not possible to distinguish if two or more bitangents are related to the same obstacle, thus the robots can only communicate to each other at rendezvous. 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 id of its owner, 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 robots that visited the obstacle and place them.

B. 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 bitangent 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 order).

Each node of the tree, has a list of bitangents that can be seen from each vertex in the obstacle. The bitangents play the role of edges in each node of the tree. When the robots are not placed at the initial obstacle, the information about the arrival vertex and the angular order of the arrival bitangent is encoded in the tree to know how to come back to the previous obstacles.

C. Obstacle exploration

The robot 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. This procedure is denoted as the *obstacle characterization*. 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. N_c will have as childs the elements that denote the index of the vertex and the index of the bitangent (see Algorithm 1). 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 has been detected. The list of vertices of the node and bitangents for each vertex in N_c is shared with other robots at the initial vertex, updating the tree in each robot.
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 his starting marker. If the scout robot is the only element in the team it keeps visiting the bitangents in his stack, otherwise, the robots have nothing to do and they turn off or return to the initial obstacle using their own tree.

D. Obstacle assignation

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 the number of childs of N_c then the robots are assigned to each bitangent according to the rule shown in Algorithm 2, creating teams. After that, each team applies Algorithm 1 at each new reached obstacle if there is no a mark at the initial vertex.

If the number of childs in N_c exceeds or match the number of robots then each robot is assigned to several bitangents (see Algorithm 2).

In both cases, the non-explored bitangents are stored in the stack of each robot thus the last added bitangents are the first to be explored.

The tree representing the obstacles is visited in deep-order and the bitangents are traveled top-down when a new obstacle is visited or bottom-up when previous bitangents in the tree have to be explored. Thus, a bitangent can be traveled in both direction but only once.

The teams are splitted during exploration but never merged again because backtracking is not useful in this case. 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 Algorithms 1, 2, and 3. The function Set_Childs adds the information of the obstacle that was discovered through the bitangent and shares the information to the robots of the team.

V. COMPLETENESS OF THE EXPLORATION STRATEGY

In this section, we prove that the proposed algorithms guarantee a complete exploration of the environment (all boundaries of the obstacles) in finite time.

Proposition 1: Let $\{O_i\}_{i=1,\ldots,n}$ a set of 2D polygonal obstacles in the Euclidean plane. For any pair of obstacles in the set there is a path, that can be constructed using the bitangents and the boundary of the elements in the set, which allows to travel between both obstacles. Also, any point in the environment can be seen from some vertex in the boundary of the obstacles.

Proof: Let O_1 and O_2 two obstacles and compute their triangulations. Select two triangles, one from each triangulation, and construct their convex hull. There are at least two bitangents in the convex hull connecting both triangles. The remaining triangles in the triangulations can be added incrementally in such a way that the number of bitangents never decreases. To add an additional obstacle, first its triangulation is computed and the resulting triangles are added in an analogous way guaranteeing that the obstacle is connected to the previous obstacles. If we consider a point obstacle in the environment this can also be connected by bitangents to the obstacles in the plane. Since the point obstacle is seen by at least one vertex of an obstacle in the set, and all obstacles are connected by bitangents then any point in the environment is connected to any obstacle in the environment. Note that the number of bitangents is finite therefore the exploration time of both obstacles and bitangents is finite.

■

Algorithm 1 Environment_Exploration_Undistinguisable

```
Input: An unknown environment $E$, 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 information about the last
    visted obstacule.
Select the robot $r_s$ with the lowest id from $R$.
if there is a mark in the vertex then
    Set_Child($parent\_node$, $g$, $NIL$).
    if $n = 1$ and $r_s$ has a non-empty stack then
        $r_s$ pops an element $q$ from his stack.
        $r_s$ travels to $q.parent\_node.level$ using the tree structure
            of $parent\_node$.
        $r_s$ travels to $q.vertex$ circumnavigating the obstacle.
        $r_s$ travels through the bitangent of $q.bitangent$ until
            reach the next obstacle.
        Let $b$ the bitangent that $r_s$ used to reach the new obstacle.
        Environment_Exploration_Undistinguisable($E$, $\{r_s\}$, $b$,
            $q.parent\_node$)
    else
        Turn off all the robots in $R$.
        return
    end if
end if
Let $N_c$ be an obstacle node that contains information about
    the bitangents in the current obstacle.
$N_c \leftarrow$Obstacle_Characterization($E$, $r_s$, $g$, $parent\_node$).
Set_Child($parent\_node$, $g$, $N_c$).
if $N_c \neq NIL$ then
    Obstacle_Navigation($E$, $R$, $N_c$).
end if
if $n = 1$ and $r_s$ has a non-empty stack then
    $r_s$ pops an element $q$ from his stack.
    $r_s$ travels to $q.parent\_node.level$ using the tree structure
        of $parent\_node$.
    $r_s$ travels to $q.vertex$ surronding the obstacle.
    $r_s$ travels through the bitangent of $q.bitangent$ until reach
        the next obstacle.
    Let $b$ the bitangent that $r_s$ used to reach the new obstacle.
    Environment_Exploration_Undistinguisable($E$, $\{r_s\}$, $b$,
        $q.parent\_node$)
end if
```

VI. 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

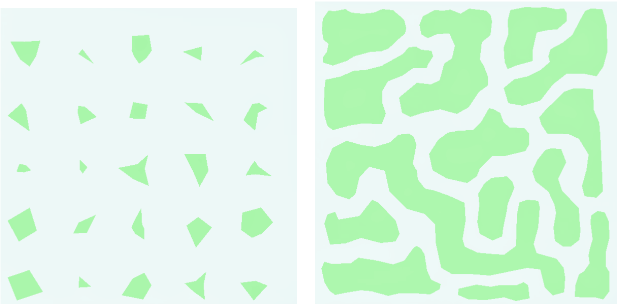
Algorithm 2 Obstacle_Navigation

```
Input: An unknown environment $E$, a team $R$ with $n$ robots,
    and a node $N_c$ of non-visited node obstacles.
Let $|N_c|$ be the number of child nodes in $N_c$.
if $|N_c| < n$ then
    Let $k, l \in \mathbb{Z}^+$ such that $n = |N_c|k + l$ where $k > 0$ and
        $0 \leq l < n$.
    Let $\{R_i\}_{i=1}^{|N_c|}$ be a partition of $R$ where $k \leq |R_i| \leq k + 1$.
    for $i = 0$ to $|N_c|$ do
        The $i$-th child node in $N_c$ is assigned to the team $R_i$.
        Let $g$ the respective bitangent to the $i$-th node in $N_c$.
        The robots in $R_i$ travels to the respective vertex and
            follow the respective bitangent $g$.
        Environment_Exploration_Undistinguisable(E, $R_i$, g,
            $N_c$).
    end for
else
    Let $k, l \in \mathbb{Z}^+$ such that $|N_c| = nk + l$ where $k > 0$ and
        $0 \leq l < |N_c|$.
    Let $\{N_i\}_{i=1}^{n}$ be a partition of child nodes of $N_c$ where
        $k \leq |N_i| \leq k + 1$.
    for each $r_i \in R'$ do
        $N_i$ is pushed in the stack of $r_i$.
        $r_i$ pops an element in his stack and goes to the respective
            vertex and then follows the respective bitangent $g$ until
            it reach a new obstacle.
        Environment_Exploration_Undistinguisable($E$, $\{r_i\}$, $g$,
            $N_c$)
    end for
end if
```

Algorithm 3 Obstacle_Characterization

```
Input: An unknown environment $E$, a robot $r_s$, $g$ the visited
    bitangent, and $parent\_node$ the node of the last visited
    obstacle.
Let $N_c$ the obstacle node of the current obstacle
Let $\partial O_s$ denote the boundary of the current obstacle.
if $g \neq NIL$ then
    The robot marks the bitangent $g_p$ that leads to the previous
        visited boundary.
end if
$r_s$ places its starting marker $p_s$ at the vertex where is located.
repeat
    $r_s$ follows $\partial O_s$ adding a child to $N_c$ for every bitangent
        detected in every vertex on it.
until a generic marker, a starting marker with a lower id, or
    $p_s$ is found
if $p_s$ was found then
    $r_s$ picks $p_s$ and it places a generic marker.
else
    $N_c \leftarrow NIL$
    $r_s$ goes to the departure bitangent picking $p_s$.
end if
return $N_c$
```

distinguishable obstacles, we have varied the backtracking level k. Figure 4 shows the maps used to perform the experiments in this section. In the case of distinguishable obstacles, we assume that each obstacle in the environment has a different color.

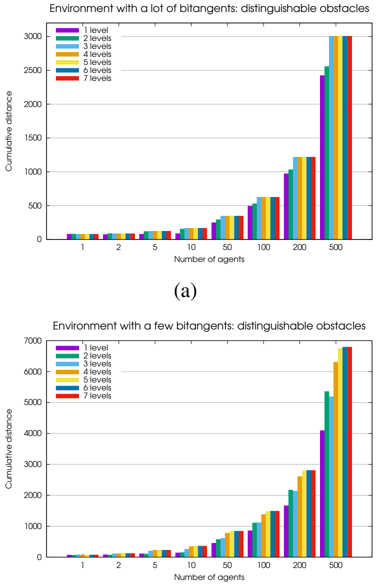


(a) A lot of bitangents

(b) Few bitangents

Fig. 4. Environments

Figure 5 shows the results of the 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.



(a)

(b)

Fig. 5. Results for distinguisable obstacles

In Fig. 5, 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. Note that the execution time was not included in the graphs, but we have observed that in general *the execution time decreases as the number of robots to perform the task increases*.

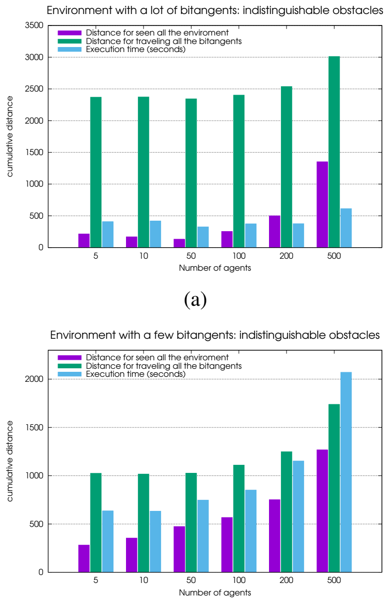
Figure 6 shows the results of the experiments assuming indistinguishable obstacles. For this case, the graphs show the cumulative distance traveled by all robots when all obstacles have been visited for the first time, the cumulative distance after all bitangents have been traveled and the execution 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. The robots do not know that the task is finished until all bitangents are traveled, because of the distributed nature of the algorithm.

From Fig. 6, 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. 6(a), where more bitangents are present with respect to the number of obstacles in the environment, than in Fig. 6(b). Since most of the obstacles in the map of Fig. 4(a) can be reached from any other obstacle visiting only one additional obstacle then the execution time remains low compare to the map in Fig. 4(b). Regarding the distance traveled by all robots to transit all bitangents, in the case of few obstacles, it increases as the number of robots increases. However, for many obstacles it almost remains constant. Another important variable to take into consideration in the algorithm's performance is the perimeter of the obstacles, since each obstacle is visited by a team as many times as the number of bitangents that it generates then a larger perimeter will have a bigger contribution to the cumulative distances.

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 all 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.

Based on the results shown in Fig. 6(a), we can observe that the time taken for traveling all bitangents might increase or decrease as the number of robots increases depending on the environment and the initial configuration of the robots. A more detailed study still remains to be done in order to characterize in which types of environments is convenient to use more robots to reduce the execution time.

In the multimedia material of the paper, we have added a video showing simulation results. In that video, we have included three experiments. In the first experiment, we show a simple case of 9 distinguishable obstacles and 3 robots without backtracking. The second experiment shows also the case of distinguishable obstacles with backtracking equal to $k = 1$, there are 200 robots and 225 obstacles. In the third experiment, we compare the behavior of our algorithms for



(a)

(b)

Fig. 6. Results for indistinguishable obstacles

the cases of distinguishable and indistinguishable obstacles. In this third 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 explore the environment faster.

A snapshot of the exploration task in an environment with 144 distinguishable obstacles and 576 robots in shown in Figure 7. In the figure the robots are represented by blue discs.

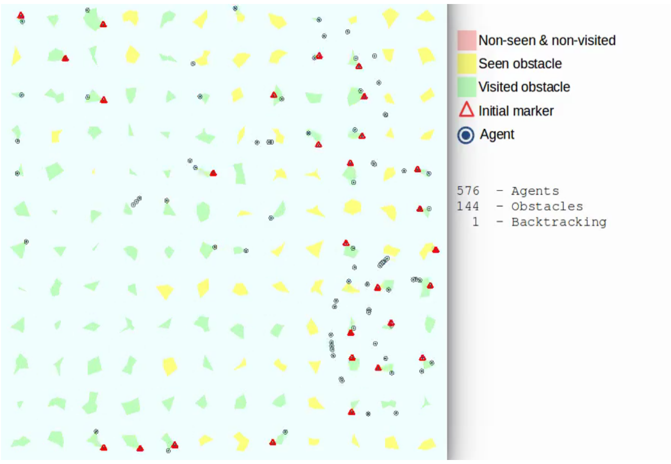


Fig. 7. Example distinguishable obstacles

VII. 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 in finite time even though a given robot does not have full information about the part of the environment explored by other robots. 2) The method only requires very limited communication between the robots, in both cases distinguishable and indistinguishable obstacles, the robots can only communicate at rendezvous. 3) The algorithm 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, in the case of indistinguishable obstacles, we would like to investigate strategies in which more information about the structure of the tree representation is used to improve the performance in terms of the distance travel by the robots.

REFERENCES

[1] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated Multi-Robot Exploration. *IEEE Transactions on Robotics*, 21(3):376–386, 2005.

[2] A. Howard, M. Matarić and G. Sukhatme. Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem. In *Distributed Autonomous Robotic Systems 5*, pages 299–308, 2002.

[3] J. Cortés, S. Martínez, T. Karatas and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[4] C.J. Taylor and D. Kriegman. Vision-based motion planning and exploration algorithms for mobile robots. *IEEE Transactions on Robotics and Automation*, 14(3):417–426, 1998.

[5] A. Sarmiento, R. Murrieta-Cid and S. Hutchinson Planning Expected-time Optimal Paths for Searching Known Environments In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 872–878, 2004.

[6] B. Tovar, R. Murrieta-Cid, and S.M. LaValle. Distance-Optimal Navigation in an Unknown Environment Without Sensing Distances. *IEEE Transactions on Robotics*, 23(4):506–518, 2007.

[7] H. Park and S. Hutchinson. An efficient algorithm for fault-tolerant rendezvous of multi-robot systems with controllable sensing range. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 358–365, 2016.

[8] G. Laguna, R. Murrieta-Cid, H.M. Becerra, R. Lopez-Padilla, and S.M. LaValle. Exploration of an unknown environment with a differential drive disc robot. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 2527–2533, 2014.

[9] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proc of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151, Monterey, CA, USA, 1997.

[10] K.M. Wurm, C. Stachniss and 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*, pages 1160–1165, 2008.

[11] L. Matignon, J. Laurent and M. Abdel-Illan. Distributed value functions for multi-robot exploration. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, pages 1544–1550, 2012.

[12] S. K. Lee, S. Fekete and J. McLurkin. Structured triangulation in multi-robot systems: Coverage, patrolling, Voronoi partitions, and geodesic centers. In *International Journal of Robotics Research*, 35(10):1234-1260, 2016.

[13] James Ng and Thomas Bräunl Performance Comparison of Bug Navigation Algorithms. In *J. Intell Robot Syst* , 50:73-84, 2007.