Research Article

Distributed Multirobot Exploration Based on Scene Partitioning and Frontier Selection

Jose J. Lopez-Perez 1, Uriel H. Hernandez-Belmonte 2, Juan-Pablo Ramirez-Paredes 1, Marco A. Contreras-Cruz 1, and Victor Ayala-Ramirez 1

1 Department of Electronics Engineering, University of Guanajuato, Campus Irapuato-Salamanca, Carretera Salamanca-Valle de Santiago Km 3.5 + 1.8, Comunidad de Palo Blanco, 36885 Salamanca, GTO, Mexico
2 Department of Art and Enterprise, University of Guanajuato, Campus Irapuato-Salamanca, Carretera Salamanca-Valle de Santiago Km 3.5 + 1.8, Comunidad de Palo Blanco, 36885 Salamanca, GTO, Mexico

Correspondence should be addressed to Uriel H. Hernandez-Belmonte; uh.hernandez@ugto.mx

Received 31 December 2017; Revised 17 March 2018; Accepted 9 May 2018; Published 20 June 2018

Academic Editor: Jaime Valls-Miro

Copyright © 2018 Jose J. Lopez-Perez et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In mobile robotics, the exploration task consists of navigating through an unknown environment and building a representation of it. The mobile robot community has developed many approaches to solve this problem. These methods are mainly based on two key ideas. The first one is the selection of promising regions to explore and the second is the minimization of a cost function involving the distance traveled by the robots, the time it takes for them to finish the exploration, and others. An option to solve the exploration problem is the use of multiple robots to reduce the time needed for the task and to add fault tolerance to the system.

We propose a new method to explore unknown areas, by using a scene partitioning scheme and assigning weights to the frontiers between explored and unknown areas. Energy consumption is always a concern during the exploration, for this reason our method is a distributed algorithm, which helps to reduce the number of communications between robots. By using this approach, we also effectively reduce the time needed to explore unknown regions and the distance traveled by each robot. We performed comparisons of our approach with state-of-the-art methods, obtaining a visible advantage over other works.

1. Introduction

One of the most challenging problems in mobile robotics is the exploration of unknown scenarios. Exploration is the task of gathering information about free space, obstacles, and boundaries in an unknown environment, while building a representation of the discovered area. When the exploration task is finished, a map of the environment is obtained.

Exploration is important in tasks such as search and rescue, planetary missions, and surveillance [1]. In some of these scenarios it is impossible or not reliable to teleoperate a mobile robot to explore the environment. Due to this restriction, the mobile robot should explore the unknown environment autonomously.

There are many approaches focused on single agent exploration in the literature [2–5], which consists of one robot exploring an unknown environment. This robot is capable of performing the exploration based on its perception capabilities. The robot perception is based on sensors such as laser range finders, RGB-D cameras, RGB cameras, or others. While the use of one single robot to explore unknown environments is the best-suited approach for planetary exploration, this is not the case for search and rescue or surveillance, when the time consumed in the exploration is a critical issue.

In applications where the exploration time is an important issue, the use of multiple robots is appropriate. Multi-robot exploration is based on a team of robots that explore an unknown environment and the information collected by each robot is shared among the team members. The coordination of the exploration, the task allocation, and the communication among the robots are some of the challenges that arise from this approach.
Two initial approaches to solve the challenges in multirobot exploration were the seminal works of Yamauchi [2, 6] and Burgard et al. [7, 8]. In his work, Yamauchi states the elemental components to perform a decentralized exploration. The robots need to coordinate their efforts to efficiently explore the unknown environment. The Burgard et al. approach is focused on task allocation, and the evaluation of how good a new target pose for an individual robot is in a global exploration context. Also, this approach covers the fault tolerance feature of multirobot systems. In the following, we review some relevant works in the multirobot exploration community.

Wurm et al. [9] proposed a multirobot exploration method based on a scenario segmentation. To divide the environment, they use Voronoi diagrams. The resulting zones are used to compute a function that evaluates the cost of exploring a specific area for a given robot. The goal area for each robot is assigned using the Hungarian Algorithm. The combination of a centralized approach with the Hungarian Method as task allocator has good results in structured environments such as offices.

Juliá et al. [10] propose an integrated exploration method for a team of robots. Its structure contains a layer of central SLAM that shares the explored map with the robots and also helps the localization of each robot. To control the navigation, each robot contains a deliberative layer and a reactive layer, where the deliberative layer controls the behavior of the reactive layer. The exploration of the scenario is done through the creation of an exploration tree, where each robot builds a tree and where the final leaves are the new positions that the robots must access.

The work in [11] is a centralized area exploration for autonomous agents, with an unknown environment and static obstacles. A reasoning algorithm is proposed which is based on routing priority. This algorithm keeps track of the frontiers, assigning robots to them whenever they fall into a trap situation.

The problem of efficiently allocating navigation goals for multiple robots in exploration tasks is addressed in [12], for unknown environments. Goal candidate locations are repeatedly determined during the exploration. Then, the assignment of the candidate goals to the robots is solved as a task-allocation problem.

One of the challenges in decentralized exploration, apart from task allocation, is information fusion. The work proposed by Rajesh et al. [13] is a good example of information fusion. Their work is focused on the problem of map building by a team of robots. Sensor fusion enables the robot team to sense things that are beyond the capabilities of a single agent. The maps are built by individual robots and these maps are fused to generate a global map. The generated global map is then filtered to remove the redundant data. The memory used for this redundant data is thus freed and can be used to store other values. This map can be used in path planning or similar activities.

Liu and Lyons [14] explain a multirobot exploration method for unknown scenarios. The method is based on creating a potential field that guides robots to unexplored areas and away from explored areas or other robots. It also proposes a decentralized method for the exchange of information between robots. They use three metrics to compare the performance of the proposal by varying some parameters, such as the number of robots, the starting point of the exploration of each robot, or the start delay of the exploration for each robot.

Some recent works have focused on the use of cost functions related to information theory, casting the exploration problem as a minimization of map entropy. Bhattacharya et al. [15, 16] use this approach, combining a grid-based map decomposition with an entropy minimization which results in complete coverage of a known map and full exploration if the scenario is unknown.

Coordinating the robots in an exploration team is another challenge. In [17], the authors address the problem of decentralized exploration and mapping of an unknown environment by a multiple robot team. The exploration methodology relies on individual decision rules and communication of topological maps to achieve efficient and fast mapping, minimizing overlap of explored regions. This distributed solution allows scalability of the proposed methods. Each robot broadcasts a graph representing the topological map, with information of the exploration status of each region. Therefore, this kind of information can be transmitted to robots that are not within the communication range, through other robots in a multihop network.

Another example of coordination between multiple robots is [18]. When multiple robots are able to coordinate themselves to explore different areas of the environment, the exploration efficiency can be greatly improved. In this article, the authors present a decentralized approach for multirobot exploration that leverages the classical frontier based methods, combined with a utility function that takes into consideration the information gain and the distance costs of the frontiers to guide the exploration. The robots are able to coordinate and avoid the exploration of redundant areas by exchanging information and merging maps.

Benkrid and Achour [19] developed a method for exploring unknown scenarios using a team of robots. Their algorithm proposes the use of frontier cells by adding a weight, which depends on the energy of the battery of each robot. The method is compared against other methods of the state of the art with respect to the time of exploration before two simulated test scenarios.

In the reviewed works, the authors mainly focused on improving the selection of the best next action for the exploration. This selection results in a reduction in the distance traveled by the robots and shorter exploration time. The next best action could be the choice of the best frontier or the best zone. Also, the selection could be performed using a function to evaluate the environment such as the map entropy or potential fields. In addition to the selection of the best next move, the communication among the robots has a critical role in the exploration. In the method proposed by Yamauchi, the robots only communicate when they arrive to an unknown zone. As a result, the method by Yamauchi is unable to explore some environments successfully. In contrast, in the method proposed by Bhattacharya et al., the communication among the robots is performed periodically, which guarantees the
full exploration of the environment. Hence, an avenue for research is to improve the methodology to select the next best move and reduce the number of the communications among the robots.

The method proposed in this work addresses this problem. Our contribution is a distributed algorithm that causes the robots to explore nearby zones to reduce the traversed distance, while efficiently using the resources to communicate with the other robots, resulting in reduced energy consumption. The results show that it can outperform information-based approaches that the robots do not get into configurations that block their movement and that they will in general provide full exploration or coverage of the environment.

While important, the time to complete the exploration is not the only metric to measure the performance of an exploration algorithm. It is also necessary to record the distance that the explorer robots traverse during the execution of this task. This distance metric directly influences the energy consumption of the robot, since the distance traveled is proportional to the battery depletion, reducing the autonomy.

This document is organized as follows. In Section 2 the proposed methodology of the exploration system developed is described. The results of the simulations carried out with the proposed system as well as the comparison of the performance against other developed methods are found in Section 3. Finally, Section 4 shows the conclusions obtained during the development of this work together with future work and acknowledgments.

2. Materials and Methods

2.1. Problem Formulation. We focus on the multirobot exploration problem, also known as online multirobot coverage problem [21]. In the multirobot exploration problem, the objective is to discover all information about an unknown environment by using a team of robots and optimizing some criteria, e.g., exploration time and distance traveled by the robots.

The exploration task is performed by a team of \( r \) robots \( \mathcal{R} = \{ R_1, R_2, \ldots, R_r \} \), with the mission to explore an unknown workspace \( \mathcal{W} \in \mathbb{R}^n \). The robots do not have prior knowledge about the environment, i.e., the exact boundaries and the positions of any obstacles in the workspace are unknown. The workspace is decomposed in a collection of \( m \) cells \( \mathcal{C} = \{ c_1, \ldots, c_m \} \), usually forming a grid of squares or cubes. At the beginning of the exploration task, each robot \( R_i \) knows only its initial configuration \( q_i^{\text{init}} \) that describes the position and orientation of the robot. This configuration is used to define the set of points occupied by the robot in the workspace. Each robot has its own exploration map \( M_i \in \mathcal{C} \times \mathcal{L} \), where \( \mathcal{L} \) are the different types of labels for a cell. In addition, each robot has a sensor \( S_i \) to explore the unknown workspace.

While the description given so far is generic we shall focus, as other works in the literature, on the case where \( \mathcal{W} \in \mathbb{R}^2 \) and \( q_i \in \mathbb{R}^2 \times [-\pi, \pi] \). Hence the space of robot configurations is \( SE(2) \). The state of robot \( R_i \) can be decomposed as \( q_i = (C_i, \theta_i) \), where \( C_i \) is the position component and \( \theta_i \) the orientation.

In our treatment of the problem, we make a few reasonable assumptions about the conditions that the robotic exploration team will face.

**Assumption 1.** The maximum bounding box of \( \mathcal{W} \) can be established, even if approximately, before the exploration starts.

**Assumption 2.** There is a surjective mapping \( q_i : SE(2) \mapsto \mathcal{C} \).

**Assumption 3.** The communication between robots is not limited by the extension of the map or the presence of obstacles.

**Assumption 4.** Each robot has access to its own localization and orientation with respect to a common inertial frame.

Since \( \mathcal{C} \) forms a square grid over \( \mathbb{R}^2 \), it will be useful for our exposition to introduce two different distance concepts. The first is the usual Euclidean distance. For some \( c_i \in \mathcal{C} \), we can denote its \( x \) and \( y \) components as \( c_i = (c_i^x, c_i^y) \). The Euclidean distance between two cells is then

\[
  d(c_i, c_j) = \sqrt{(c_i^x - c_j^x)^2 + (c_i^y - c_j^y)^2}.
\]

The second concept of distance is related to the fact that the set of cells \( \mathcal{C} \) forms a space over which a connectivity between neighbors can be defined in different ways. A common assumption in the literature that we use in this work is 8-connectivity, which means that an agent positioned over a cell can travel to one of the surrounding 8 cells. We define a geodesic distance between cells, \( d_g(c_i, c_j) \), which is the shortest path between cells \( c_i \) and \( c_j \) that respects 8-connectivity.

2.2. Methodology. In this section we present our methodology to solve the problem of the coordination of a robotic team for the exploration of unknown scenarios.

2.3. System Description. The proposed system has a modular structure, with the purpose of allowing some of its parts to be further improved in future works or to permit the addition of new modules.

Figure 1 shows the flowchart diagram of the proposed multirobot coordination method. In this diagram, we can distinguish four essential modules:

(i) **Information sharing module**: using this module the robot shares information about its position and its exploration map with the other robots.

(ii) **Zone assignment module**: this module is responsible for separating the scene into zones and for selecting a zone for the robot to explore.

(iii) **Data acquisition module**: this module is in charge of obtaining data from an exteroceptive sensor, to represent the scenario in which the mobile robot is immersed.

While the description given so far is generic we shall focus, as other works in the literature, on the case where \( \mathcal{W} \in \mathbb{R}^2 \) and \( q_i \in \mathbb{R}^2 \times [-\pi, \pi] \). Hence the space of robot configurations is \( SE(2) \). The state of robot \( R_i \) can be decomposed as \( q_i = (C_i, \theta_i) \), where \( C_i \) is the position component and \( \theta_i \) the orientation.

In our treatment of the problem, we make a few reasonable assumptions about the conditions that the robotic exploration team will face.

**Assumption 1.** The maximum bounding box of \( \mathcal{W} \) can be established, even if approximately, before the exploration starts.

**Assumption 2.** There is a surjective mapping \( q_i : SE(2) \mapsto \mathcal{C} \).

**Assumption 3.** The communication between robots is not limited by the extension of the map or the presence of obstacles.

**Assumption 4.** Each robot has access to its own localization and orientation with respect to a common inertial frame.

Since \( \mathcal{C} \) forms a square grid over \( \mathbb{R}^2 \), it will be useful for our exposition to introduce two different distance concepts. The first is the usual Euclidean distance. For some \( c_i \in \mathcal{C} \), we can denote its \( x \) and \( y \) components as \( c_i = (c_i^x, c_i^y) \). The Euclidean distance between two cells is then

\[
  d(c_i, c_j) = \sqrt{(c_i^x - c_j^x)^2 + (c_i^y - c_j^y)^2}.
\]

The second concept of distance is related to the fact that the set of cells \( \mathcal{C} \) forms a space over which a connectivity between neighbors can be defined in different ways. A common assumption in the literature that we use in this work is 8-connectivity, which means that an agent positioned over a cell can travel to one of the surrounding 8 cells. We define a geodesic distance between cells, \( d_g(c_i, c_j) \), which is the shortest path between cells \( c_i \) and \( c_j \) that respects 8-connectivity.

2.2. Methodology. In this section we present our methodology to solve the problem of the coordination of a robotic team for the exploration of unknown scenarios.

2.3. System Description. The proposed system has a modular structure, with the purpose of allowing some of its parts to be further improved in future works or to permit the addition of new modules.

Figure 1 shows the flowchart diagram of the proposed multirobot coordination method. In this diagram, we can distinguish four essential modules:

(i) **Information sharing module**: using this module the robot shares information about its position and its exploration map with the other robots.

(ii) **Zone assignment module**: this module is responsible for separating the scene into zones and for selecting a zone for the robot to explore.

(iii) **Data acquisition module**: this module is in charge of obtaining data from an exteroceptive sensor, to represent the scenario in which the mobile robot is immersed.
(iv) Exploration module: this module is in charge of controlling the movement of the mobile robot so that it navigates through the area it has to explore.

2.4. Information Sharing Module. The information gathered by each robot needs to be shared with the team members to complete the exploration. The information sharing module is in charge of performing the exchange of information between two robots. A robot $R_i$ has a list of positions with the locations of all other robots $V_i = \{C_1, C_2, \ldots, C_r\}$ and the exploration map $M_i$. Each robot $R_i$ shares its current list $V_i$ and exploration map $M_i$ with another robot.

During this exchange, each robot $R_i$ fuses its own map $M_i$ and list of positions $V_i$ with the $M_j$ and $V_j$ transmitted by another robot $R_j$. The transmission network topology used in the communication is a ring, which reduces the number of messages used in the exploration.

2.5. Zone Assignment Module. Once the position information is shared by each robot, it is necessary to separate the scenario into different zones, so that each robot can explore one. Making the robots explore different zones reduces any potential blocking problems caused by interference, while letting each robot focus on a different part of the map.

The assignment of zones to be explored by each robot is done as follows. Each robot $R_i$ has a list $E_i = \{c_1, c_2, \ldots, c_n\}$ with all the $n$ cells that it must explore. The assignment of each one of the cells $c_j$ to the list $E_i$ is done according to

$$E_i = \{c_j : d_g(C_i, c_j) \leq d_g(V_i( k), c_j) \forall i \neq k, c_j \in U_i\} \quad (2)$$

where $U_i \subset M_i$ are the unexplored cells from the map associated with robot $R_i$. In Figure 2 one can observe how the zones are assigned to each robot.

Once robot $R_i$ has explored all cells contained in the list $E_i$ or if the robot can no longer explore the remaining cells of the list, a zone reassignment is performed, using the updated information about the robot positions and the updated map with the information from the other robots. If $E_i$ is empty, $R_i$ will remain inactive until another robot activates the information sharing and zone assignment modules, so that $R_i$ can add cells to $E_i$ again.

2.6. Data Acquisition Module. This module is tasked with obtaining and updating a representation of the scenario in which a robot is immersed. The mobile robot is provided with proprioceptive and exteroceptive sensors. The proprioceptive sensors are used to compute the current position of the robot. The exteroceptive sensors let the robot acquire information about its surrounding area. In this exploration task, the robot
uses a laser range finder (LRF) to get the measures from the obstacles that are near it.

The LRF provides distance readings at fixed angles in the form of an array $S_i = \{l_1, l_2, \ldots, l_N\}$ with $N$ being the maximum number of readings taken by the LRF. After processing the current readings of the LRF sensor, an updated map $M_i$ is computed. This enables the robot to handle the mobile and static obstacles present in the environment.

We use the method proposed by Lopez-Perez et al. [20] to label each cell, where the cells in the map $M_i$ can be labeled using four possible values: $\mathcal{L} = \{\text{unexplored cell}, \text{explored cell}, \text{static obstacle cell}, \text{mobile obstacle cell}\}$. The labeling procedure depends on the computation of two functions:

1. Function $f_1$ serves to determine which cells are occupied by an obstacle, either static or mobile obstacle. This function uses the collision points of $S_i$.
2. Function $f_2$ serves to update the label of all the cells inside the polygon formed by the laser measures to the explored cell status.

For function $f_1$, we obtain the collision points $l_k$ for each of the measurements of the LRF. The distance from these points $l_k$ to each of the center positions $c_j$ of the map grid is computed. If the distance $d(l_k, c_j)$ is smaller than the safety radius of the mobile robot $r_s$, the function returns 1; otherwise it returns 0 (see Figure 3).

$$
    f_1(c_j) = \begin{cases} 
      1 & \text{if } \min d(l_k, c_j) \leq r_s \\
      0 & \text{otherwise}
    \end{cases}
$$

For the function $f_2$, we generate a polygon using the laser points $l_k$ as vertices. If a cell is inside the polygon, $f_2$ takes the value of 1, otherwise it returns a value of 0. The computation to know if the point $c_j$ is inside the polygon is made by tracing a vertical line and counting the number of intersections with the laser measurement polygon. If there is an odd number of intersections, the function $f_2$ returns 1; otherwise it returns 0 (see Figure 4).

The transition between the states that a cell can have is described via the finite-state machine diagram in Figure 5. Note that the combination $(f_1, f_2) = (1, 1)$ is not represented there, because it can not occur in any state. When $f_1$ is 1, $f_2$ automatically takes the value of zero. All the cells are initially labeled as unexplored cells.

When a cell $c_j$ is labeled with a value different from unexplored cell, or if $d_g(C_i, c_j) > d_g(V_k(k), c_j)$ for any $k \neq i$, due to the presence of undetected obstacles during the execution of the zone assignment module, $c_j$ is removed from $E_i$.

2.7. Exploration Module. For a mobile robot to be able to explore an unknown scenario, it must navigate towards unmapped areas. The purpose of the exploration module is to guide the mobile robot towards unknown zones of the scenario. Figure 6 shows the flow diagram for the exploration module. In this diagram we can distinguish three essential submodules:

(i) Goal assignment: this submodule is in charge of assigning a goal to each robot. The candidate goals belong to the zone that the robot is tasked with exploring. The goal assignment depends on a weight function.

(ii) Path planning: this submodule uses the A* algorithm to provide a collision-free route between the robot and the goal assigned.
(iii) Trajectory execution: this submodule computes the required set of \((v, \omega, t)\) so that the mobile robot displaces over the scenario, following the planned trajectory.

During the execution of this module and the data acquisition module, the robot continuously deletes some cells \(c_j\) from the list \(E_i\). If \(E_i\) is empty, robot \(R_i\) will request a new zone assignment.

2.7.1. Goal Assignment. Once robot \(R_i\) has an exploration zone assigned, a goal \(G_i\) is selected, which is one of the cells contained in the list \(E_i\). The assignment of this goal depends on a weight function \(f_p\) which is computed as

\[
f_p(c_j) = k_d d_p(C_i, c_j) + k_a \phi_i(C_i) \tag{4}
\]

where \(\phi_i(c_j)\) is the angle between the current orientation \(\theta_i\) of the robot and the orientation of the vector with origin on \(C_i\) that ends at \(c_j\). The positive constants \(k_d, k_a\) determine the influence of the geodesic distance compared to the orientation the cell.

The cell \(c_j\) assigned as the goal \(G_i\) for a robot has to be a cell labeled as unexplored which has the minimum value in the weight function \(f_p(c_j)\) of all cells belonging to list \(E_i\).

\[
G_i = \arg \min_{c_j \in E_i} f_p(c_j) \tag{5}
\]

Once cell \(c_j\) has been selected as the goal \(G_i\) for the robot, it is removed from the list \(E_i\). When this cell is no longer labeled as unexplored by the data acquisition module, it is removed as a goal and replaced with a different one, by repeating the goal assignment procedure. Using this continuous goal assignment, the robot can displace in a safe manner and explore the whole zone that it was assigned to.

2.7.2. Path Planning. Using the occupancy grid approach, the mobile robot computes a path from the current position \(C_i\) to
the goal position $G_i$ by using the $A^*$ planning algorithm [22–24]. Initially, the robot does not know any information about the environment, so it considers that the whole map is free space and the robot computes an initial path $P_i$. If, throughout navigation, the planned path $P_i$ is obstructed by some static or mobile obstacle, a replanning is performed on the updated occupancy grid map $M_i$, as shown in Figure 7. In this figure, we observe in each frame the planned path $P_i$, the mobile robot in the current position $C_i$, and its goal $G_i$, the real static or mobile obstacles in the scenario, and the different labels of the cells.

The global planning of a path is necessary so that the robot is endowed with sufficient knowledge to be able to escape the problems of local planning. For example, in the scenario of Figure 7 a typical local planning problem is observed, which is a local minimum.

**Figure 5:** State transition diagram for each cell of the occupancy grid [20].

**Figure 6:** Flow diagram for the exploration module. The robot uses this module to compute and assign the next goal, obtain a path to the assigned goal, and execute the motion commands to follow the trajectory.
Figure 7: Path planning for a mobile robot. The robot performs replanning when its planned path is obstructed by some obstacle either (a) static obstacle or (b) mobile obstacle. We observe the planned path $P_i$ (set of red line segments), the mobile robot in the current position $C_i$ and its goal $G_i$, the static obstacles in the scenario in gray color, the other mobile robots in color gray with blue, and the different labels of the cells (in black color for the static obstacle cells, in yellow color for the explored cells, in green color for the mobile obstacle cells, and in white color for the unexplored cells).
An initial path $P$ will only consist of unexplored cells and explored cells (see Figure 8(a)). If the mobile robot does not find a path, the submodule considers the mobile obstacle cells too and tries to find a path (Figure 8(b)). If an obstacle-free path is not found with this process, another goal is selected in the goal assignment submodule.

2.7.3. Trajectory Execution. The trajectory execution submodule is responsible for controlling the linear velocity $v$ and the angular velocity $\omega$ of the mobile robot so that it moves from its current position $C_i$ to the position of a local goal $G_l$, following the planned path $P_i$. The local goal $G_l$ is obtained from the selection of a cell belonging to the planned path $P_i$ and located at some distance from the current position $C_i$.

Finding the set of translational and rotational velocities, in order to move the robot from its current position to a local goal position, is a problem of inverse kinematics. For this particular problem there exist many possible solutions, some of which can cause the robot to not follow the path. The analytic solution to this problem requires the use of some constraints in order to simplify the problem. On the other hand, there are techniques than can generate solutions automatically without solving the analytic expression. One such technique is evolutionary computation, which has been used to solve complex design problems [25]. An advantage of evolutionary computation for solving engineering design problems is that it can generate interesting designs that are competitive or in some cases better than designs by experts [25].

If the problem of inverse kinematics is represented as a parametric problem, where the parameters to find are the set of velocities and the time interval during which they have to be applied, we can use an evolutionary technique in order to generate a solution to this problem. In this work, we propose a method based on a Genetic Algorithm, given that it has been successfully applied for solving different mobile robotics problems [26–28].

The Genetic Algorithm finds the set of $v$, $\omega$, and $t$ required for the robot to move from its current position $C_i$ to the position of a local goal $G_l$. Each individual of the Genetic Algorithm represents a triplet of variables $v$, $\omega$, and $t$. The ranges of the variables to be optimized are:

\begin{align}
v & \rightarrow \left[0.0 \frac{m}{s} , 0.1 \frac{m}{s}\right] \\
\omega & \rightarrow \left[-0.1 \frac{rad}{s} , 0.1 \frac{rad}{s}\right] \\
t & \rightarrow \left[0.0s , \frac{d(C_i,G_l)}{0.1s}\right]
\end{align}

An individual is better fitted if it represents a motion command suited to move the mobile robot in less time and less distance. The individual is evaluated by simulating the displacement of the mobile robot, obtaining the final position $C_i'$ and the final orientation $\theta_i'$.

The fitness function to be minimized is

\[ F \left(C_i', \theta_i', G_l \right) = k_d' d \left(C_i', G_l \right) + k_d' \phi_i' \left(G_l \right) \]

where $\phi_i' \left(G_l \right)$ is the angle between the orientation of the robot $\theta_i'$ and the vector from the location $C_i$ to $G_l$.

We used two weighting terms $k_d'$ and $k_d'$ that were adjusted during experimentation by trial and error. As can be
observed, the fitness function rewards the closeness of the final position $C'_i$ with respect to the local goal $G_i$ and also the similarity of the final orientation $\theta'_i$ with the orientation of the vector from the point $C'_i$ to the $G_i$.

The discretization of the planned path $P_i$ in local goals $G_i$ contributes to the following: (i) reducing the search space of the Genetic Algorithm, which leads to obtain the best individual in a faster way; (ii) helping the mobile robot to develop smoother paths between its current position $C_i$ and its local goal $G_i$; and (iii) ensuring that the mobile robot follows the planned path $P_i$ avoiding collision with any obstacles.

### 3. Results and Discussion

We present the results obtained by the proposed system in a series of comparisons with other methods, which involved extensive simulations. Afterwards, we present the results of an implementation of our method in an accurate simulation using the Gazebo system. In the proposed system, the GA uses a population size of 40 individuals, a crossover probability of 0.8 with a single crossover point, a mutation probability of 0.01 (multibit mutation), and the roulette wheel selection. In addition, the GA uses a steady-state policy with two elite members. These parameters were determined experimentally. In these tests, we discretized the environment as a grid of cells of dimensions 0.5 m $\times$ 0.5 m.

#### 3.1. Performance Comparison with the State-of-the-Art Methods.

The performance of the proposed method was characterized using 6 test scenarios shown in Figure 9 and in Table 1 we present the identifier of each map with its size in meters.

They are available from the motion planning repository [29]. These maps can be related to real problems such as traps for the robot, narrow corridors, areas with a large density of obstacles, labyrinths, and rooms.

For each of the test scenarios, 9 different configurations were generated. That is, the number of robots comprising the exploration team was varied from 2 to 10. One-hundred executions per configuration were made for each test scenario.

There exist several different methods to initialize the positions of the exploration team. Some related works initialize the positions of the team with a predefined formation at some selected zone of the scenario [14, 30]. Other authors define the initial positions of the robots arbitrarily within a zone that depends on the scenario [6, 15]. Finally, in other works the positions of the exploration team are initialized at random within the free space [8, 19]. We use this last initialization strategy, since we consider it to be the best way to show the performance of different methods for comparison. We believe all exploration algorithms should achieve full exploration of the reachable free space, so this initialization can also help to demonstrate this for different techniques.

The metrics to be evaluated to compare the performance of the proposed system against other methods established in the literature are as follows: the average distance traveled by each robot, the total distance of the exploration team, the average time of exploration, and the average number of communication exchanges between the robot team members, divided by the average time of explore.
Figure 10: Average distance traveled by each robot to complete the exploration task, for each test scenario.

The results presented in this article were obtained simulating the proposed system on differential drive robots with a LRF with 361 measurements, a vision field of 360° and 5 m of range. The distance that a robot travels is the sum of the displacement of its wheels. The sum of the distance that each robot of a team travels represents the total exploration distance, and the total exploration time is measured from the start of the simulation until the team has labeled each and every cell of the map with a value different from unexplored.

In order to compare the performance of our method (here named Scene Split into Zones, SSZ), we implemented three different strategies. The first of them is the method by Yamauchi, which is a decentralized algorithm based on the exploration of the nearest frontiers. Many other techniques have been compared to this method [8–10, 19].

The second method we implemented is a variant of the first, where instead of using the nearest frontier, the frontier to use is chosen at random. Finally, the third method is by Bhattacharya et al. [15], which is based on entropy minimization and the exploration is guided towards the centroid of a zone to explore by each robot.

In the interest of completeness, we now describe the information-based method by Bhattacharya et al. In this method, each robot maintains three grid-based maps containing information about occupancy probability, obstacles, and entropy. The robots are assumed to know their locations and to relay them to their peers. To perform the exploration, each robot computes a Voronoi tessellation of its own maps, which is weighted using the entropy measure. Since the map is constantly updated, the entropy is a time-varying quantity, so for a given map cell \( c \), it is computed as

\[
e(c, t) = p^i(c) \ln p^i(c) + (1 - p^i(c)) \ln (1 - p^i(c))
\]

where \( p^i(c) \) is the probability of occupancy for cell \( c \) at time \( t \). Then, each robot takes a step along the shortest-path route towards the nearest entropy-weighted Voronoi region centroid. We refer the reader to the article for details about the handling of edge cases, such as nonconvex Voronoi regions with a centroid not inside them.

In Figure 10 one can observe the average distance traveled by each robot for each test scenario. On this figure the distance that each robot travels diminishes as the number of
exploring robots increases. It can also be seen that the proposed method SSZ outperforms the others on all test cases.

All of the compared methods converge to some value as the number of robots increases. In Figure 10 the Random Frontier method is the worst performing, with the highest performance being achieved by our method, with the method by Bhattacharya et al. following as second best and matching ours for teams of 6 robots or more.

Figure 11 shows the total distance traveled by the robot team for each test scenario. The distance that the team travels diminishes as the number of robots increases for the SSZ method and the method by Bhattacharya et al. However, our method has significantly better results for scenarios with less than 6 robots and remains as the best for more than 6 robots. The nearest frontier algorithm and the Random Frontier algorithm do not reflect such behavior, with the traveled distance increasing with the number of robots.

In Figure 12, where the average exploration time is shown for each test scenario, one can observe the same behavior as in Figure 10, where the exploration time diminishes as the number of robots increases and the results obtained by the SSZ method are better in most cases.

As evidenced by the simulation results, the method that we propose outperforms others, both classic and from the state of the art. The last comparison from these simulations is the average number of communication exchanges between the robot team members, divided by the average time to explore the scenario, shown in Figure 13. In this Figure there is a notable difference between the two methods with the best performance from the prior tests, which are SSZ and that by Bhattacharya et al.

It can be noticed that our method has less communication exchanges per time unit, for most test cases. As we mentioned earlier, the number of communication exchanges is an important factor, since the battery life of the robots is related to it. In a few cases, the Random Frontier method presents better results than SSZ in this comparison, since the metric is inversely proportional to the exploration time. Given that the exploration time is much longer for the Random Frontier algorithm, it is expected to show less communication per time unit.

Figure 14 shows how the proposed system explores an unknown scenario (test case 6 from Figure 9). At time step 1 the explorer robots are shown, along with the cells assigned
Table 2: Summary of results of our method and others in the literature.

| Metric     | Nearest Frontier | Random Frontier | Bhattacharya | SSZ    |
|------------|------------------|-----------------|--------------|--------|
| AD (m)     | 161.04           | 551.88          | 151.88       | 91.3   |
| TD (m)     | 795.44           | 2835.2          | 563.61       | 398.54 |
| AT (steps)| 907.8            | 3120.2          | 1237.6       | 733.86 |
| NC/T       | 0.6784           | 0.5539          | 1.0617       | 0.2636 |

FIGURE 12: Average exploration time for each test scenario.

3.2. Accurate Simulation of Multirobot Exploration. The first step to achieve real mobile robot implementation is the validation of our approach in a realistic simulation environment. Many frameworks provide the facilities to simulate multirobot systems; nonetheless, the Robot Operating System (ROS) [31] using Gazebo [32] as a simulator engine is a reliable choice for testing new algorithms. The usage of ROS and the Gazebo simulator as the development environment offers the possibility of using the same implementation for the simulated and real mobile robots.

We chose the Husky A200 all-terrain mobile robot for the simulation. This robot is equipped with a LRF that provides 640 measurements, a vision field of 180 degrees and 5 m of range. The LRF is mounted at the front of the robot. We...
implemented the proposed SSZ approach on ROS using a team of homogeneous robots utilizing this robot and the Gazebo simulator.

The most common method to share information on ROS is using topics. A robot publishes data utilizing a topic (publisher) then another robot consumes this data (subscriber) and performs some task. This model is used to share information among the mobile robot team. In our implementation, we use this communication model to share the robot positions $V_i$ and the exploration map $M_i$.

The SSZ approach is presented in this simulation environment, and the data visualization is shown in Figure 15. We present the Gazebo display output on the left side of the figure. The Gazebo output serves to render the current state of the virtual world. In our case, it renders the Husky A200 robots and the test scenario 1 (Figure 9). We use the RViz tool to visualize the current state of the exploration map for each robot. This can be seen on the right side of Figure 15. We can see that the robots have already divided the environment and have assigned a region to explore. We can observe that the map is labeled in the same way as in Figure 14. In the supplementary material, we provide a video (available here) with the full simulation for this environment. From the qualitative results obtained from this simulation, we conclude that it is possible to implement the SSZ approach on a physical platform.

4. Conclusion

In this work we present a method to coordinate a robot team for the exploration of unknown scenarios, which may include moving obstacles. The method we propose is based on the separation of the scenario into zones, so that each robot explores a different one. By continuously assigning goals and based on the concept of frontier cells, the mobile robot navigates inside its exploration zone to acquire a representation of the scenario. This representation is fused with those of the other robots to obtain a full map.
The algorithm is implemented in a modular way, where each module is designed to deal with dynamic scenarios. This augments the robustness of the system since any obstruction by mobile obstacles, such as people, objects, or other robots is minimized.

The method is decentralized, which reduces the communication cost between the robots, when compared to other centralized methods. This also helps to make our method fault-tolerant, since even if a robot is unable to complete its own exploration task it can be aided by the others. The
decentralized and modular nature of our system allows the use of heterogeneous teams of robots.

We tested the system with several test scenarios, in extensive simulations. The results obtained demonstrate that our method outperforms the well-established nearest frontier method, a variant using Random Frontiers, and an information-based method, when comparing exploration time and the traveled distance per robot. From this we conclude that our system is efficient in its coordination of a robotic exploration team.

Our method can be further improved. One aspect that can be enhanced is that the current implementation requires the robots to know the dimensions, i.e., width and height of the scenario to explore, and they will not venture outside these limits. Another point for improvement is the resilience of the method under localization uncertainty, since the current algorithm assumes that there is no sensor noise.

As for future work, we plan to compare our approach to more methods from the state of the art and to deploy our algorithm on a hardware platform by using the proposed implementation for the Robot Operating System (ROS).

**Data Availability**

Any data related to the simulations described in the article are available from the authors upon request.

**Conflicts of Interest**

The authors declare no conflicts of interest.

**Acknowledgments**

Jose J. Lopez-Perez and Marco A. Contreras-Cruz would like to thank CONACYT for the financial support through Scholarship Grants 291047/736591 and 568675/302121, respectively.

**Supplementary Materials**

We include a video recording containing an execution of the proposed system running on Linux and implemented using ROS and the Gazebo simulator as supplementary material. This demonstrative video shows three robots performing the exploration of an unknown environment using the SSZ algorithm. The Gazebo simulator is presented at the left side of the video. We use a visualization tool called RViz to show the exploration progress. This information is presented on the right side of the screen. The visualization tool shows how the grid values are changing during the execution of the exploration task. The exploration process finalizes when all the environment has been explored. *(Supplementary Materials)*

**References**

[1] A. Marjovi, J. G. Nunes, L. Marques, and A. de Almeida, “Multi-robot exploration and fire searching,” in *Proceedings of*
the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS ’09), pp. 1929–1934, IEEE, October 2009.

[2] B. Yamauchi, “Frontier-based exploration using multiple robots,” in Proceedings of the 5th symposium on robotics and autonomous systems, pp. 47–53, Minneapolis, MN, USA, May 1998.

[3] D. Santosh, S. Achar, and C. V. Jawahar, “Autonomous image-based exploration for mobile robot navigation,” in Proceedings of the 2008 IEEE International Conference on Robotics and Automation, ICRA 2008, pp. 2717–2722, May 2008.

[4] M. Juliá, A. Gil, and O. Reinoso, “A comparison of path planning strategies for autonomous exploration and mapping of unknown environments,” Autonomous Robots, vol. 33, no. 4, pp. 427–444, 2012.

[5] S. Obwald, M. Bennewitz, W. Burgard, and C. Stachniss, “Speeding-Up Robot Exploration by Exploiting Background Information,” IEEE Robotics and Automation Letters, vol. 1, no. 2, pp. 716–723, 2016.

[6] B. Yamauchi, “Decentralized coordination for multirobot exploration,” Robotics and Autonomous Systems, vol. 29, no. 2, pp. 111–116, 1999.

[7] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, “Collaborative multi-robot exploration,” in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA ’00), vol. 1, pp. 476–481, San Francisco, CA, USA, April 2000.

[8] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, “Coordinated multi-robot exploration,” IEEE Transactions on Robotics, vol. 21, no. 3, pp. 376–386, 2005.

[9] K. M. Wurm, C. Stachniss, and W. Burgard, “Coordinated multi-robot exploration using a segmentation of the environment,” in Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, pp. 1160–1165, September 2008.

[10] M. Juliá, Ó. Reinoso, A. Gil, M. Ballesta, and L. Payá, “A hybrid solution to the multi-robot integrated exploration problem,” Engineering Applications of Artificial Intelligence, vol. 23, no. 4, pp. 473–486, 2010.

[11] K. R. Sharma, D. Honc, F. Dusek, and G. Kumar, “Frontier based multi-robot area exploration using prioritized routing,” in Proceedings of the 30th European Conference on Modelling and Simulation, ECMS 2016, pp. 25–30, June 2016.

[12] J. Faigl, O. Simonin, and F. Charpillet, “Comparison of task- allocation algorithms in frontier-based multi-robot exploration,” in Proceedings of the European Conference on Multi-Agent Systems, pp. 101–110, Springer, 2014.

[13] M. Rajesh, R. Joseph, and T. S. B. Sudarshan, “Fully distributed and decentralized map building for multi-robot exploration,” in Proceedings of the 2014 International Conference on Embedded Systems, ICES 2014, pp. 220–224, July 2014.

[14] T.-M. Liu and D. M. Lyons, “Leveraging area bounds information for autonomous decentralized multi-robot exploration,” Robotics and Autonomous Systems, vol. 74, pp. 66–78, 2015.

[15] S. Bhattacharya, N. Michael, and V. Kumar, “Distributed coverage and exploration in unknown non-convex environments,” in Distributed Autonomous Robotic Systems, pp. 61–75, Springer, 2013.

[16] S. Bhattacharya, R. Ghrist, and V. Kumar, “Multi-robot coverage and exploration on Riemannian manifolds with boundaries,” International Journal of Robotics Research, vol. 33, no. 1, pp. 113–137, 2014.

[17] T. Pereira, A. P. Moreira, and M. Veloso, “Coordination for multi-robot exploration using topological maps,” in Proceedings of the 11th Portuguese Conference on Automatic Control, CONTROL’2014, pp. 515–524, Springer, 2015.

[18] R. G. Colares and L. Chaimowicz, “The next frontier: Combining information gain and distance cost for decentralized multi-robot exploration,” in Proceedings of the 31st Annual Symposium on Applied Computing, SAC 2016, pp. 268–274, April 2016.

[19] A. Benkrid and N. Achour, “A novel approach for coordinated multi-robot exploration,” in Proceedings of the 6th International Conference on Systems and Control, ICSC 2017, pp. 509–513, May 2017.

[20] J. J. Lopez-Perez, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, “Dynamic object detection and representation for mobile robot application,” in Proceedings of the Mexican Conference on Pattern Recognition, pp. 84–93, Springer, 2016.

[21] N. Hazon, F. Mieli, and G. A. Kaminka, “Towards robust online multi-robot coverage,” in Proceedings of the 2006 IEEE International Conference on Robotics and Automation, ICRA 2006, pp. 1710–1715, May 2006.

[22] R. Murphy, Introduction to AI Robotics, MIT press, 2000.

[23] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, Introduction to Autonomous Mobile Robots, MIT press, 2011.

[24] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 4, no. 2, pp. 100–107, 1968.

[25] A. E. Eiben and J. Smith, “From evolutionary computation to the evolution of things,” Nature, vol. 521, no. 7553, pp. 476–482, 2015.

[26] A. Tuncer and M. Yildirim, “Dynamic path planning of mobile robots with improved genetic algorithm,” Computers and Electrical Engineering, vol. 38, no. 6, pp. 1564–1572, 2012.

[27] H. Qu, K. Xing, and T. Alexander, “An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots,” Neurocomputing, vol. 120, pp. 509–517, 2013.

[28] C.-C. Tsai, H.-C. Huang, and C.-K. Chan, “Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation,” IEEE Transactions on Industrial Electronics, vol. 58, no. 10, pp. 4813–4821, 2011.

[29] “Intelligent and Mobile Robotics Group: Motion planning maps,” 2017, http://imr.ciirc.cvut.cz/planning/maps.xml.

[30] C. Nieto-Granda, J. G. Rogers, and H. I. Christensen, “Coordination strategies for multi-robot exploration and mapping,” International Journal of Robotics Research, vol. 33, no. 4, pp. 519–533, 2014.

[31] A. Martinez and E. Fernandez, Learning ROS for robotics programming, Packt Publishing Ltd, 2013.

[32] N. Koenig and A. Howard, “Design and use paradigms for Gazebo, an open-source multi-robot simulator,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS ’04), vol. 3, pp. 2149–2154, IEEE, October 2004.
Submit your manuscripts at
www.hindawi.com