UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

RAFAEL MARQUES RACHE

Coordinated navigation of multiple robots in formation

Work presented in partial fulfillment
of the requirements for the degree of
Bachelor in Computer Engineering

Advisor: Prof. Dr. Renan de Queiroz Maffei

Porto Alegre
February 2024

ABSTRACT

This work addresses the problem of coordinated navigation of multiple robots in formation in confined spaces. With the advancement of robotics and the increasingly frequent use of autonomous robots in applications, it becomes necessary to efficiently plan paths and coordinate robot actions so that they act robustly and safely during the cooperation process. Potential field techniques that solve a boundary value problem (BVP) have proven to be an interesting way of dealing with the planning problem, but little has been explored in the context of navigation in multiple robot scenarios. This work proposes a system of multi-robot coordination using BVP and elastic formations that allow breaking the formation of a group of robots in confined spaces while being able to reunite the formation in areas along the goal path where a minimal formation size fits in the environment.

Keywords: Potential Fields. Multiple Robots. Robots in Formation. Navigation. Breaking Formation.

Navegação Coordenada de Múltiplos Robôs em formação

RESUMO

Este trabalho aborda o problema da navegação coordenada de múltiplos robôs em formação em espaços confinados. Com o avanço da robótica e o uso cada vez mais frequente de robôs autônomos em aplicações, torna-se necessário planejar caminhos de forma eficiente e coordenar as ações dos robôs para que atuem de forma robusta e segura durante o processo de cooperação. Técnicas de campo potencial que resolvem um problema de valor de contorno (BVP) têm se mostrado uma maneira interessante de lidar com o problema de planejamento, mas pouco foi explorado no contexto da navegação em cenários de múltiplos robôs. Este trabalho propõe um sistema de coordenação de múltiplos robôs usando BVP e formações elásticas que permitem quebrar a formação de um grupo de robôs em espaços confinados, sendo capaz de reunir a formação em áreas ao longo do caminho objetivo onde um tamanho mínimo de formação cabe no ambiente.

Palavras-chave: Campos Potenciais. Múltiplos Robôs. Robôs em Formação. Navegação. Quebra de Formação.

LIST OF ABBREVIATIONS AND ACRONYMS

C-SPACE	Configuration Space
BVP	Boundary Value Problem
MRS	Multi-Robot System
HIMM	Histogramic In-Motion Mapping
PDE	Partial Differential Equation
ROS	Robot Operating System
VS	Virtual Structures
VB	Virtual Bodies

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

1.1 Motivation

Mobile robots have been used in many applications in a wide variety of industries over the last two decades and the idea of robots cooperating has become popular. Numerous practical applications of autonomous robots such as natural resource monitoring, manufacturing, emergency response, and rescue, necessitate the involvement of multiple robots providing solutions to tasks that are inherently distributed in terms of time, space, or functionality.

The creation of multi-robot applications deals with the design, construction, and deployment of a group of robots that cooperate to solve a problem or perform a task in a coordinated way. The creation of these applications aims to build systems that are adaptable to changes in the environment, fault-tolerant, and robust (DORIGO; THERAULAZ; TRIANNI, 2021).

One crucial challenge in these mobile robot groups is coordinating the movements of multiple robots within the same workspace. Irrespective of the mission of the robot, it must be capable of effectively sharing the workspace to avoid any interference among members. Solutions to coordinated navigation problems are approached in diverse ways, depending on the underlying objectives of the robot team. (PARKER, 2009)

Mobile robot research led to the development of many algorithms that solve issues surrounding robot navigation, such as path planning and obstacle avoidance. To solve these problems, Occupancy grids (MORAVEC; ELFES, 1985) became a widely used paradigm for environmental modeling. It consists of creating a 2D grid where each cell stores quantitative information about the area in which the robot is operating, indicating the probability of it being occupied or free.

Using this paradigm, a traditional method for resolving a path planning task is creating Artificial Potential Fields. Originally introduced by (BORENSTEIN; KOREN, 1989), this algorithm is an elegant and efficient solution for robot navigation problems and has been one of the most used schemes, especially for collision avoidance, since its proposition. The concept entails on treating the robot as a particle within a field formed by the combination of an attractive field generated by the robot's desired final position and a repulsive field generated by obstacles surrounding the robot.

To achieve the goal, which is defined as the global minimum of the potential field, the robot simply needs to follow the gradient descent of the field. But, this solution comes with a problem, where robots can get stuck in a local minima due to the sum of separate components of the potential field. This issue can be overcome with the computation of harmonic potential fields (CONNOLLY; BURNS; WEISS, 1990) by solving the boundary value problem (BVP) using the Laplace equation, generating paths free from local minima.

While potential fields such as BVP-based approaches have demonstrated efficacy in guiding single robots, extending their application to multiple robot scenarios such as the ones investigated in this work necessitates strategies to mitigate issues such as inter-robot collisions and formation maintenance. Thus, exploring the ideal holds promise for advancing the coordinated navigation of multiple robots.

1.2 Objectives

This work aims to analyze strategies for coordinated navigation of multiple robots in formation using potential fields. For that, we need a reliable method of accurately mapping the environment and providing a path for the robot to reach its goal, then coordinating multiple robots in formation along this path by defining their movement. In the past, BVP was used for path planning of multiple agents in crowd simulations, and such approach can be exploited for multiple robots (DAPPER; PRESTES; NEDEL, 2007).

For that, three types of occupancy grid maps are used, one small *local map* for every agent, where its center is the robot itself. A *group map* to coordinate all the agent's movements in formation, where the focal point is regarded as an agent and a path planning algorithm is employed to generate a collision-free route towards the intended destination. And finally, a *global map* to represent the environment the agent is immersed in. (DAPPER; PRESTES; NEDEL, 2007)

Sometimes, these environments can have a limited space for the formation to go through, being smaller than the original formation plan required. For the group of robots to pass these confined locations and to guarantee the formation geometry, an elastic formation approach is an effective alternative for this problem, compressing and relaxing the formation to pass smaller spaces and controlling for formation velocity to avoid robots being left behind. (SANTOS, 2022). However, in some configuration spaces, it is better to break the formation in confined spaces than to move as a group.

We propose a system to break the formation of a group in confined spaces, allowing the robot to move freely in the direction of the nearest available formation-free space. Then, when all the robots are reunited, they start to move in formation again, repeating this process until the final goal is reached.

1.3 Organization

The organization of this work is as follows. First, we will present a bibliographical review of multi-robot coordination classification and related work done in the field. Next, we present a theoretical background of potential field algorithms for path planning using occupancy grids, choose the data structure that will represent the robot environment, and define the calculation of the robots and the formation movements. Following, we propose our strategy for multi-robot coordination. Then an overview of the technologies and libraries used in the simulations. Finally, implement our method in a virtual environment using Robot Operating System (ROS) and show the results of the experiments made.

2 BIBLIOGRAPHICAL REVIEW

2.1 Multi-Robot Coordination Classification

In order to effectively control the cooperative actions of multiple robots within a system, it is crucial to establish a mechanism for coordination, which plays a vital role in enabling cooperation within a Multi-Robot System (MRS). Cooperation in MRS is defined as the manifestation of collaborative behavior among the robots, resulting in an overall increase in the system's total utility. This cooperative behavior is achieved through the underlying mechanism of coordination, which ensures the successful completion of a specified task, where the robot not only focuses on its own tasks but also requires awareness of any pressing assignments from its partner.

2.1.1 Static and Dynamic Coordination

Static coordination (also known as deliberative coordination or offline coordination) typically involves the establishment of a convention before starting a task. It can handle complex tasks, but its real-time control might be poor.

Dynamic coordination (also known as reactive coordination or online coordination) occurs during the execution of a task and is based on the analysis and synthesis of information obtained via means of communication. It can be achieved using both implicit communication or explicit communication. (YAN; JOUANDEAU; CHERIF, 2013). The dynamic method meets the expectation while completing real-time tasks, but it has difficulty in dealing with more complex tasks.

2.1.2 Implicit and Explicit Coordination

Implicit coordination approaches employ the dynamics of interactions between robots and the environment. This type of coordination relies on implicit communication, which is primarily manifested through emergent behaviors that are devised by the robots. To perceive changes in the environment, various sensors or devices are used.

Explicit coordination approaches utilize intentional communication, also known as explicit communication, and cooperation methods that are similar to those employed in Multi-Agent Systems (MAS).

Explicit coordination approaches are capable of handling more sophisticated robots compared to implicit coordination approaches. Although there are some similarities in the methods used for coordination in MAS and Multi-Robot Systems (MRS) when explicit coordination is employed among robots, they are not fundamentally equivalent.

2.1.3 Weak and Strong Coordination

A weak or strong coordination is a form of coordination that relies or not on a coordination protocol.

As mentioned by (IOCCHI; NARDI; SALERNO, 2001) "strongly coordinated systems are based on a system of signals by which a robot exerts its influence on the behavior of another. In other words, strong coordination is based on the application of predefined or learned rules concerning the way two or more robots have to interact."

On the other hand, a weakly coordinated MRS does not rely on a specific protocol, making it more resilient to communication failures. However, it lacks several organizational capabilities that a coordination protocol provides. Comparing both, a strong coordinated MRS is more effective in accomplishing its objective when the environment is highly dynamic and the goal is complex than a weakly coordinated MRS.

There is another option where an MRS can not be coordinated, those systems rely on its less complex design which involves a lower risk of fault, however, there may be a significant waste of resources, either due to the robots performing conflicting tasks or due to the emergence of interference.

2.1.4 Centralized Coordination

Centralized coordination refers to the arrangement of an MRS wherein a robotic agent, known as the leader (be it a single robot or a server), assumes responsibility for coordinating the tasks of the other robots. The leader actively participates in the decision-making process for the entire team, while the remaining members carry out their actions based on the leader's instructions.

Furthermore, centralized coordination can be sub-categorized as strongly centralized or weakly centralized. Where a weakly centralized coordination can permit one or more robots/systems to be a leader for the duration of the task. Strongly centralized coordination uses a fixed leader during the entire task.

Strongly centralized methods are more vulnerable to failure as a result of both leader malfunction and communication breakdown. In these approaches, the failure of communication can result in the collapse of the entire coordination process. Additionally, if the leader becomes incapacitated, a strongly centralized technique may fail to achieve any form of coordination. On the other hand, weakly centralized techniques are more resilient compared to strongly centralized ones as they have the ability to designate a new leader in the event of leader failure.

If there exists a scenario where there are multiple leaders in a system, and these leaders are ultimately controlled by one leader, such approaches are also classified as centralized. Conversely, if multiple leaders operate independently without being controlled by a single leader, is referred to as hierarchical.

2.1.5 Decentralized Coordination

On the other hand, decentralized coordination does not need such a leader, in fact, the agents in the formation possess complete independence in the decision-making process, operating without any central authority.

Decentralized approaches can be further classified into two types: all robots have equal responsibility in coordinating the task, and hierarchical approaches which are locally centralized

For these reasons, a decentralized approach is better for coordinating an MRS with a large number of robots due to its high computation requirement and the communication cost among the robots.

2.2 Related work

The study of swarm robotics first started while researchers were testing the concept of stigmergy, a mechanism to coordinate agents through the environment. This research created the Ant System, the first ant colony optimization (ACO) algorithm. ACO takes inspiration from the foraging behavior of specific ant species that deposit pheromones on the ground to mark some favorable path that should be followed by other members of the colony. Ant colony optimization exploits a similar mechanism for solving optimization problems. (DORIGO; BIRATTARI; STUTZLE, 2006).

As the years went by, methods for the navigation of multiple robots in formation used in real-world applications became an important topic in the field. The use of virtual bodies (VB) (OGREN; FIORELLI; LEONARD, 2004) and virtual structures (VS), is a common way of keeping the geometry of the formation along the group path to the final destination.

(LEWIS; TAN, 1997) method applies a virtual force field on a virtual structure, making individual robots within the VS compelled to move in the direction of the force. As a result, the robots adjust their movements to remain within the virtual structure, while the virtual structure adapts to accommodate the robots' current positions. In this method, the formation structure is rigid, and cannot be changed, meaning it is not elastic, to ensure a smooth movement of the formation it relies on changes in the velocity of the formation and the velocity of the robots.

(SHAO; WANG; XIE, 2006) present an algorithm for multi-robot formation system control with obstacle avoidance inspired by the phenomenon of hydro-statics. In their method, a group of robots establish a rigid body-like formation, with a virtual point as its center. For robots to avoid obstacles, the virtual structure changes 'flexibility', for that the contract strategy postulates that every robot is linked to the virtual robot through a virtual spring (or an elastic rod), making the formation change globally, meaning the formation shape change size, and independently, meaning that just one robot can be different to the original formation to avoid an obstacle.

(OLCAY; SCHUHMANN; LOHMANN, 2020) uses a strategy to allow the navigation of a single robot to reach its goal in an unknown environment and extends it to decentralized multi-robots without formation using local communication and information exchange with three types of information packages: orientation information, information about endpoints and information about corners. Additionally, the utilization of potential fields allows for inter-agent collision avoidance, the preservation of proximity, and the enhancement of safety during collision avoidance with static obstacles.

(REN; BEARD, 2008) used a neighbor-to-neighbor information exchange architecture to solve the issue of formation control of multiple robots using virtual leaders. For obstacle avoidance, they integrated potential fields method. The use of potential fields is usually associated with obstacle avoidance, but it can be used for keeping the geometry of a group formation (SILVEIRA; SILVA; NEDEL, 2008) (SCHNEIDER; WILDERMUTH, 2003), where various forces from different robots, obstacles, and the desired formation shape are integrated and utilized to maneuver each robot to its intended position within the formation. As the group moves in unison, they adeptly navigate around obstacles and steadily progress towards a predetermined target.

Table 2.1: Related work classification

Article	Centralized	Robot Leader	Handle Robot Separation	Elastic Formation	Velocity Control	Uses virtual robots/bodies
(OGREN; FIORELLI; LEONARD, 2004)	Yes	No	Yes	No	Yes	Yes
(LEWIS; TAN, 1997)	Yes	No	No	No	Yes	Yes
(SHAO; WANG; XIE, 2006)	Yes	No	No	Yes	No	Yes
(REN; BEARD, 2008)	No	Yes	No	No	No	No
(SCHNEIDER; WILDERMUTH, 2003)	No	No	No	No	No	No
(OLCAY; SCHUHMANN; LOHMANN, 2020)	No	No	No	No	Yes	No

3 THEORETICAL BACKGROUND

This chapter presents a background of methods and techniques used in this work for robot navigation, path planning, collision avoidance behavior, and map data structure, with its main focus on potential fields.

3.1 Path Planning

Path planning is an essential role in the navigation of mobile robots, a task that must ensure the safe movement of the robot towards its goal, avoiding collision with other objects and preventing the robot from getting lost in the workspace area.

The data structure that represents the environment captured by the robot sensors and the algorithm used to find a feasible and minimal path, from the robot's current position and configuration to its destination, are the main components of path planning.

Configuration space (c-space) representations can be based on multiple data structures, such as Voronoi diagrams, occupancy grids, generalized cones, trees, quad-trees, and many more shown by (BURGARD; HEBERT, 2008). Generally, all these structures contain the space regions that the robot can move and forbidden regions that the robots must avoid, being these regions' obstacles or low-preference regions. The selected data structures directly influence the complexity and efficiency of the algorithm implementation.

Many algorithms and data structures can be used to solve a path-planning problem. Potential Fields over occupancy grids have shown to be a great solution for path planning and will be used for this paper.

3.2 Potential Fields

Potential fields' main idea is to unite two types of potential field forces to map the robot's movement, as shown in Figure 3.1. One field repels the robot from objects and another attracts the robot to its final goal, resulting in behavior that avoids obstacles and moves in the direction of the goal.

This solution can be computed in different ways: for example, the potential field can be produced by adding all the potentials of each type (attractive and repellent), it has a

Figure 3.1: On the left, the image represents the attractive potential without obstacle, on the center the repulsive potential sets the highest value to the obstacle and finally on the right, the potentials of the left and the center summed resulting in a final potential field.

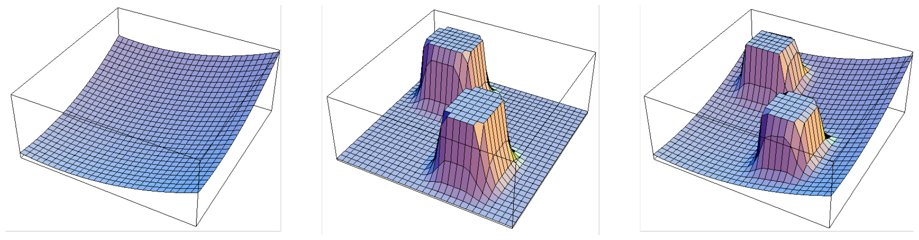


Image from (SABUDIN; OMAR; MELOR, 2016)

low computational cost but suffers from local minima. In contrast, the potential field can be calculated from the solution of a partial differential equation (PDE) from a harmonic function using the boundary value problem (BVP), where both obstacles and goals are treated as boundary conditions. The equation of the potential, $p(r)$ of a given cell r is defined by the Laplace Equation:

$$\Delta p(r) = \nabla^2 p(r) = 0 \tag{3.1}$$

This approach removes the local minima problem and is formally complete, however, it increases the computational cost. (TREVISAN et al., 2006)

The Laplace operator Δ is a second-order differential operator in the n-dimensional Euclidean space, defined as the divergence ($\nabla\cdot$) of the gradient of the potential ($\nabla p(r)$).

$$\Delta p(r) = \nabla^2 p(r) = \nabla \cdot \nabla p(r) \tag{3.2}$$

$$\nabla = \left(\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_n} \right). \tag{3.3}$$

For the BVP solution, we indicate that the surfaces of the obstacles are set to the value of 1, while the goal position is set to 0. Now, we use these values to calculate Eq (3.1), resulting in a potential field whose gradient can be used to guide the robot towards the goal. With this approach, any position in free space where the robot is placed, there is a path known to the objective.

To solve Eq (3.1) efficiently, there are approximation methods, like Jacobis's Method or Gauss-Seidel Method. These methods' solution consists of a finite number of iterations using Eq (3.4) until convergence.

Where r are points in a square lattice, and the sum is over the four next neighbors

Figure 3.2: Different paths using Eq (3.5) : (a) path produced by Laplace's equation, i.e., with $\varepsilon = 0$; (b) with $\varepsilon = 0.8$ and $v = (1, 0)$; (c) with $\varepsilon = 0.8$ and $v = (1, sin(wt))$.

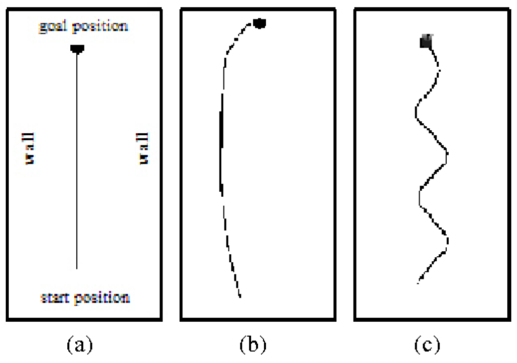


Image from (DAPPER; PRESTES; NEDEL, 2007)

of each point, represented by the set $\aleph(r)$.(TREVISAN et al., 2006)

$$p(r, t + \Delta t) = \sum_{r' \epsilon \aleph(r)} \frac{1}{4} p(r', t) \tag{3.4}$$

By adding a distortion bias to Eq (3.1),

$$\nabla^2 p(r) - \varepsilon v \cdot \nabla p(r) = 0 \tag{3.5}$$

where v is a bias vector and ε is a scalar value, we can manipulate the potential field, as shown in Figure 3.2, to generate specific behavior, such as making the robot path closer to the wall. Just like Eq (3.1), it can be efficiently calculated using relaxation. (SILVEIRA; SILVA; NEDEL, 2008)

$$p(r, t + \Delta t) = \sum_{r' \epsilon \aleph(r)} \frac{1}{4} p(r', t) - \frac{\varepsilon}{8}((p_{i+1,j} - p_{i-1,j})v_x + (p_{i,j+1} - p_{i,j-1})v_y)) \tag{3.6}$$

3.3 Map representation

An important part of path planning is to define a data structure that represents efficiently the environment in which the robot is inserted. A popular representation used to plan a robot navigation route is by making approximations of the c-space in regular cells, called occupancy grids. This work uses occupancy grids at different levels to achieve the results of collision avoidance, path planning, and formation keeping.

Figure 3.3: Environment map on the right and its respective occupancy grid on the left.

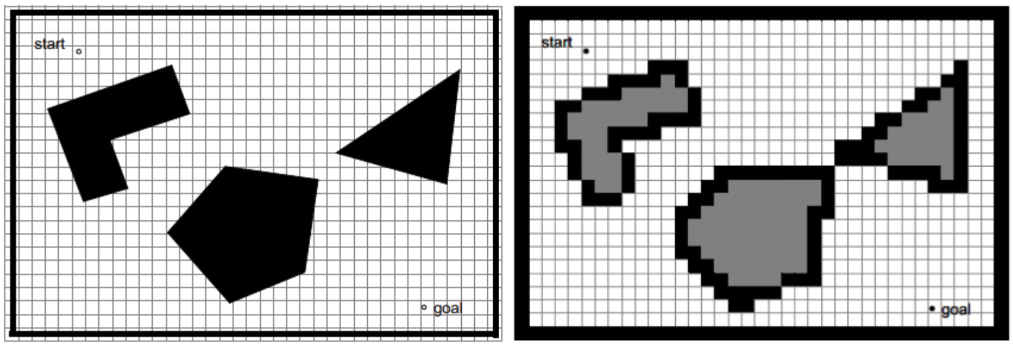


Image from (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011)

3.3.1 Occupancy grids

The occupancy grid framework provides a robust and unified approach to a variety of problems in spatial robot perception and navigation (ELFES, 1989). It involves dividing the space surrounding the robot into a multidimensional grid with a fixed size, typically in 2D or 3D. Each cell in the grid stores a probabilistic estimate of its state, being it free or occupied as shown in Figure 3.3.

To deal with the uncertainties of the robot's sensors, many algorithms provided by a formal theory of evidence can be used to determine if a cell is occupied by an obstacle or free to move, such as Bayesian or Dempster-Shafer method, or by heuristics, like HIMM (Histogramic In-Motion Mapping).

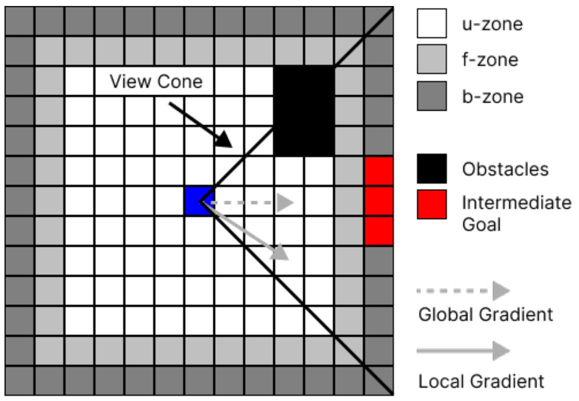
3.3.2 Environment Management for Potential Fields with Multiple Agents

This work is based on (DAPPER; PRESTES; NEDEL, 2007), which divides the environment map into three different levels of abstraction, each one created to facilitate the development and robustness of the system as a whole.

3.3.2.1 Global Map

The environment in which the robots will be placed, called *global map*, will be represented by homogeneous meshes m_k with L_x x L_y cells, denoted by $c^k_{i,j}$, representing a region in the environment with coordinates $r = (r_x, r_y)$, storing a potential value $p^k_{i,j}$ computed by Eq (3.5). This global map is computed before the simulation and is surrounded by obstacles with potential value 1, making the robot stay inside the map.

Figure 3.4: Agent Local Map. White, light gray and dark gray cells comprise the update, free and border zones, respectively. Red, black and blue cells correspond to the intermediate goal, obstacles and the agent position, respectively.



Created by author, based of (DAPPER; PRESTES; NEDEL, 2007)

3.3.2.2 Local Map

Each robot r_k will have an associated map rm_k, called *local map*, with a mesh of l_x x l_y cells, denoted by $rc^k_{i,j}$, containing the information obtained by its sensors in the environment and it is centered on the robot's position. The *local map* will be smaller than the *global map* to limit the number of cells to produce a refined movement for the robot.

In this map, there will be three regions: the update zone (u-zone), the free zone (f-zone), and the border zone (b-zone), as shown in Figure (3.4).

The b-zone is used to generate an intermediate goal for the robot to follow, making cells that are directed to the goal with a potential value of 0 and the rest occupied, with a potential value of 1. This generates the behavior of the robot following the path to the objective and also allows it to produce particular motion when facing nearby obstacles or other robots in the u-zone.

The u-zone cells contain the potential value of the obstacles and even other robots in the *local map*, making them with potential value 1. F-zone cells are always free cells (value 0), without them the robot can have nowhere to go if the b-zone with the intermediate objective is blocked by an obstacle in the F-zone.

After the mapping of the environment by the robot's sensors, and filling the *local map* with its correct values, relaxation is used to generate the potential field of the u-zone in the mesh, generating a path for the robot to follow.

3.3.2.3 Group Map

For a group of robots to move in formation, there should be a specific shape that represents each robot's position. The formation shape will correspond to a list of points $L = (x_i^l, y_i^l)$.

The group of robots in formation will have an associated map, called *group map*, made by Y_x x Y_y cells. It will be projected into the environment, aligned with the center of the robot group.

The *group map* will store a potential value on each cell, similar to section 3.2, with an u-zone, where obstacles have high potential values equal to 1 and free cells values from 0 to 1, a f-zone, with just free cells with potential 0, and a b-zone, where we have an intermediate goal guiding the robot to its final destination. Two forces will influence the movement of the robot: the force of obstacles and players pushing the robot away and the *formation force* that attracts the robot with low potential.

The force of obstacles and players are calculated with Eq (3.5), and the forces z exerted by the *formation points* can be calculated by

$$z_k = (x_k^l, y_k^l) - (x_k, y_k) \tag{3.7}$$

where (x_k^l, y_k^l) is the position of the formation point of a robot k.

3.4 Robot and Formation movement using potential fields

To achieve the behavior of robots moving in formation, while avoiding obstacles and moving towards the group goal, this work is based on (SILVEIRA; SILVA; NEDEL, 2008), where agents' actions are composed of individual and group movement, which will be discussed next.

3.4.1 Individual movement

All robots have two forces influencing their movement, the first one is the force trying to keep the robot in formation and the other one is the force produced from the gradient descent where the global minima is the group goal and obstacles are global maximums, as shown by Figure 3.5. At each iteration of the algorithm, the resulting force

Figure 3.5: Characters inside this map are under the influence of two forces. The formation force (dark arrows) and the grid force (light colored arrow). 0 is the group map border and is an obstacle zone used as the BVP boundary condition. 1 and 2 are the zones where the grid force has higher influence than the formation force.

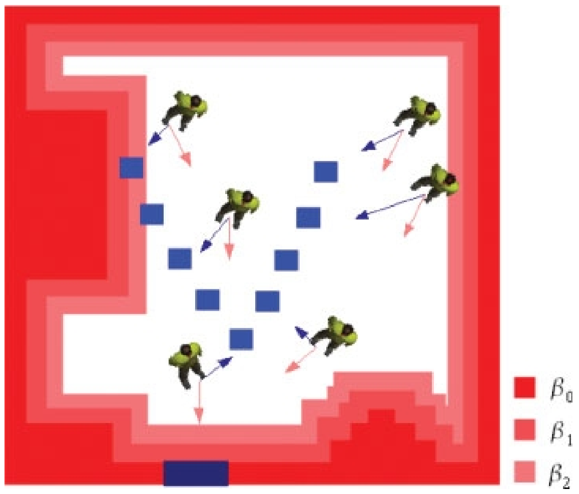


Image and caption from (SILVEIRA; SILVA; NEDEL, 2008)

produces a collision-free path while trying to keep the robot in formation position. The following equation describes the update of position r of character k at time t :

$$\Delta r = V_{max}[\mu_{\beta i}\varepsilon + (1 - \mu_{\beta i})\alpha\mathcal{F}] \tag{3.8}$$

where the environment contribution is ε and the formation contribution is $\mathcal{F}$,

$$\varepsilon = (|\phi^{t-1} - \zeta^t|)(cos(\phi^t), sin(\phi^t)) \tag{3.9}$$

$$\mathcal{F} = (|w^t - w^{t1}|)(cos(w^t), sin(w^t)) \tag{3.10}$$

$$\phi^t = \phi^{t-1} + (1)\zeta^t \tag{3.11}$$

ϕ^t is the character orientation in the instant t, ζ^t is the gradient descent orientation in the current character position in the instant t and w^t is the orientation of the force z, as shown in (SILVEIRA; SILVA; NEDEL, 2008). v_{max} must be greater than the group map velocity to ensure that the character always stays inside the group map. If we reduce the equation above to

$$\Delta r = v_{max}\varepsilon \tag{3.12}$$

the group will move without a defined formation. (SILVEIRA; SILVA; NEDEL, 2008)

3.4.2 Group movement

To move all robots in formation, we will need to handle the motion of the *group map*. For this, the center of the *group map* will be treated as a robot and a path planning algorithm will be used to produce a path free of collisions to the objective. With each step of the algorithm, a new position for its center will be calculated (usually calculated based on the max robot velocity), and the parts of the *global map* that are now contemplated in the *group map* will be projected. Formation points are rotated to align their orientation with the direction of the path planner. Using this approach, the new positions of the formation points will guide the robots to their new desired position, getting each step closer to the goal. (VERMA; RANGA, 2021)

3.4.3 Elastic Formation

Recently, (SANTOS, 2022) developed a system that restricts the split of a swarm of robots into sub-groups when going through complex and cluttered environments using potential fields that compute elastic formations for swarms while coordinating the robots' velocities so that the swarm can uniformly move toward the goal avoiding a strayed robot. With this method, we can compress and expand the formation geometry if needed and ensure that the group moves in unison with no robot being left behind.

4 PROPOSED STRATEGY FOR MULTI-ROBOT COORDINATION

We propose a strategy to break the formation of a group in confined spaces, based on the BVP-based strategy for group formation-keeping from (SILVEIRA; SILVA; NEDEL, 2008), in which the potential fields are used to avoid obstacles and move in the formation towards the goal. Our method is also based on the elastic formation and group control velocity method from (SANTOS, 2022), to avoid robots being left behind the formation and guaranteeing the formation geometry.

As presented in Algorithm 1, the group formation will be broken in regions where the area available for the robots to move is below a certain threshold th relative to the area of the original formation itself. After the formation is broken, it will only be restored in the next position for the formation center following the path to the goal which does not contain the limitation of size mentioned above. In the regions where the formation is disabled, the robots use the original path of the formation center to navigate through the narrow space in the direction of the new formation position.

When all the robots get close to their new formation position, the group is reunited and starts to move as a unity again. This process continues until the formation center reaches the final destination goal.

Algorithm 1 Break Formation Algorithm

```
 1: $isFormationBroken \leftarrow false$
 2: while Formation center is not in final goal and robots are not in their respective
    position in formation do
 3:     if $isFormationBroken == false$ then
 4:         Move formation center
 5:         if Formation center fits group area limit then
 6:             Move robots in formation
 7:         else
 8:             Calculate next formation center passing confined space
 9:             Move robots individually
10:             $isFormationBroken \leftarrow true$
11:     else
12:         if Robots near formation center then
13:             $isFormationBroken \leftarrow false$
14:         else
15:             Move robots individually
16:
```

An example of the strategy can be seen in Figure 4.1. Before the algorithm starts (4.1.a), the formation geometry must be defined, such as the specific position of each robot in the formation and the path from the start to the goal location. With these prerequisites achieved, we can start the algorithm (Figure 4.1.b).

In our method, the formation starts united and the stop condition for the algorithm is if the formation center reaches the goal and all robots are in their specified position in the formation. In the main loop, first we verify if the formation is broken or not, if it is not broken, we move the formation to the next step of the path to the goal (Figure 4.1.c). When we move the formation center, we expand or contract the geometry of the formation if there is or not space available in the environment.

When the maximum formation area available is established, the next step is verify if this new geometry fits the area threshold of the algorithm. A threshold $th = 0.6$ for example, means that if the distance of an robot to the formation center is $1m$ on its ideal position on the formation, when the formation is contracted and this new distance is smaller than $0.6m$, the formation must be broken. Because, in Figure 4.1.c that's not the case, we can move the robots in formation in direction of the goal (Figure 4.1.d).

Iterating the main loop again, we verify if the formation is broken, which is not the case, move again the formation (Figure 4.1.e), making it expand or contract, and finally check if there new formation area is above the specified area threshold. In this case, because the environment is too narrow, the the formation area is below the limit. In that case, we need to calculate a new position of the formation along the goal path where the minimum formation size can be respected and start moving the robots individually to this new position with an broken formation (Figure 4.1.f).

During this trajectory, we verify if all robots are near the new formation center positioned after the narrow environment. When they are not all near, we continue to move them individually until they reach it. When the condition is reached (Figure 4.1.g), we reunite the formation and restart the algorithm (Figure 4.1.h).

For the formation to arrive at the destination and the algorithm to end, we must move the formation to until its center is in the final destination (Figure 4.1.i) and wait until all the robots arrive at their specified position (Figure 4.1.j).

During the trajectory of the robots, the potential field map for each robot is updated

Figure 4.1: Steps of the proposed system of breaking formations illustrated

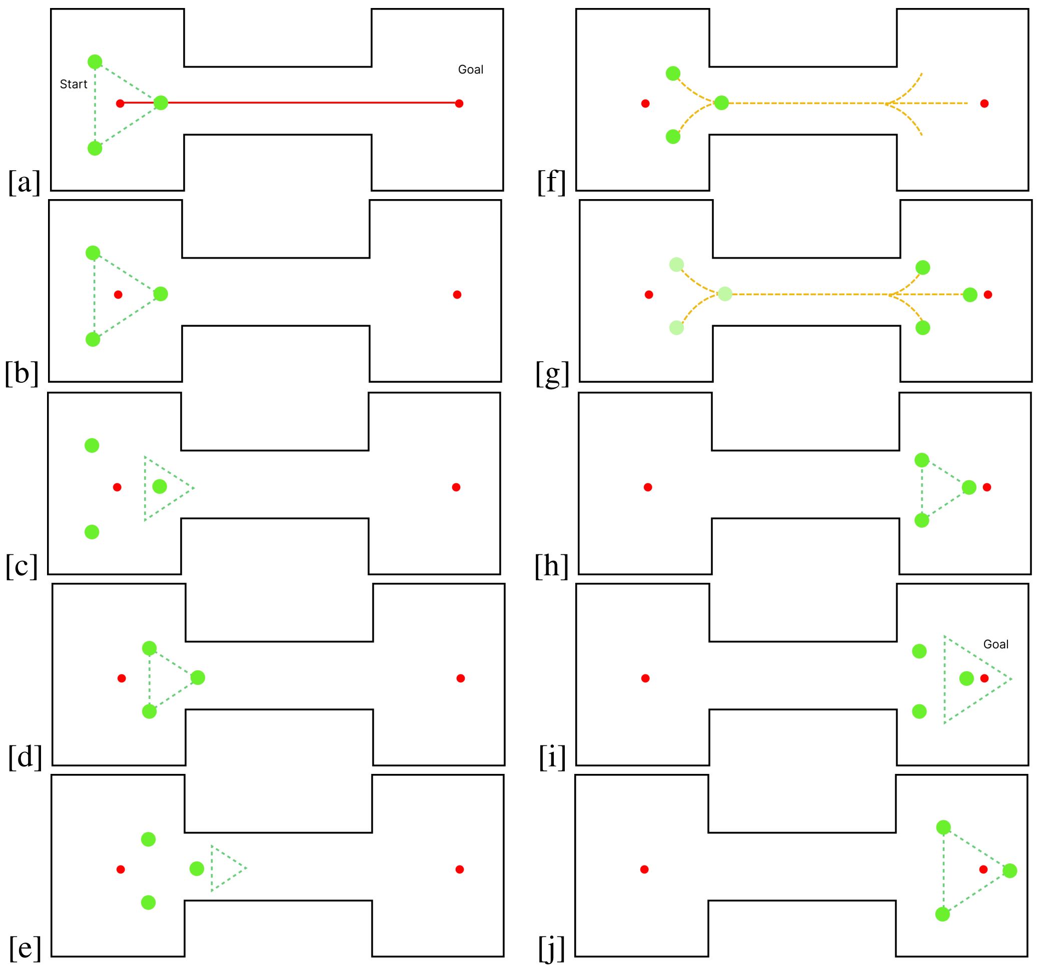


Image from author

with the same equation 3.6. With each state of the algorithm, the bias vector v changes.

$$v^k[final] = \begin{cases} v^k - (r^k_{ideal} - r_c) & \text{if } !isFormationBroken \\ v^k & \text{if } isFormationBroken \text{ and } robotNearFormationCenter \\ 0 & \text{if } isFormationBroken \text{ and } !robotNearFormationCenter \end{cases} \tag{4.1}$$

Where robot k bias v^k, ideal position r^k_{ideal} and formation center position r_c are used to define the $v^k[final]$ of robot k. These cases allow the robots to remain in their specified position in the formation when it is not broken. When the formation is broken the 0 of $v^k[final]$ makes the robot move without any bias. And when the formation is broken and the robots arrive at the next formation center, each one is positioned again at its specific spot at the formation center.

This method being applied to each robot individually produces a behavior in which if a robot arrives first in its new formation position, it awaits for the other robots to arrive. Finally, when all of them are in place to unite the formation, the formation starts to move again.

During the formation and broken formation phase, the map limit where the potential fields will be updated is also defined by the algorithm state. When the group is in formation, the map limits will be the formation size, and when the formation is broken, the limits must be a minimum size in which all robots are inside the update zone.

Figure 4.2 shows a scenario comparing the proposed approach (bottom) with the approaches proposed by (SILVEIRA; SILVA; NEDEL, 2008) (middle) and (SANTOS, 2022) (top). The environment has a narrow passage in which the robot formation cannot pass in its original size. According to the method proposed by (SILVEIRA; SILVA; NEDEL, 2008), the intermediate goals of some robots can be very close to obstacles (since this is not analyzed in the generation of goals) leading the robots to come dangerously close to collision situations. Using the method proposed by (SANTOS, 2022), the formation shrinks within the narrow area to a minimum size but only shrinks enough to fit in that area, so the robots can also pass close to obstacles. In the proposed method, it is also possible to shrink the formation, but if it were to become too small, the formation is momentarily broken and reactivated somewhere later where the space is large enough.

Figure 4.3 shows a second scenario comparing the proposed approach (bottom) with the approach by (SANTOS, 2022) (top). This time the environment is even more narrow. In this case, the method by (SANTOS, 2022) cannot find a path that safely takes the formation to the destination, even in its minimum size. In the case of the proposed method, the formation was broken in the narrow area and resumed later, enabling the successful arrival at the destination.

Figure 4.2: Formation path with (SILVEIRA; SILVA; NEDEL, 2008) method (a), (SANTOS, 2022) method (b) and our method (c). Robots are illustrated as green circles, the start and goal as red circles connected by a red line(representing the path to the goal), robots outside the map in grey circles, formation geometry in green lines, and the individual robot path in a broken formation. In the image, we can see that the first strategy (a) can produce strange behavior if the formation points are outside the calculated path walls, and in the other ones (b and c) all robots follow the formation path.

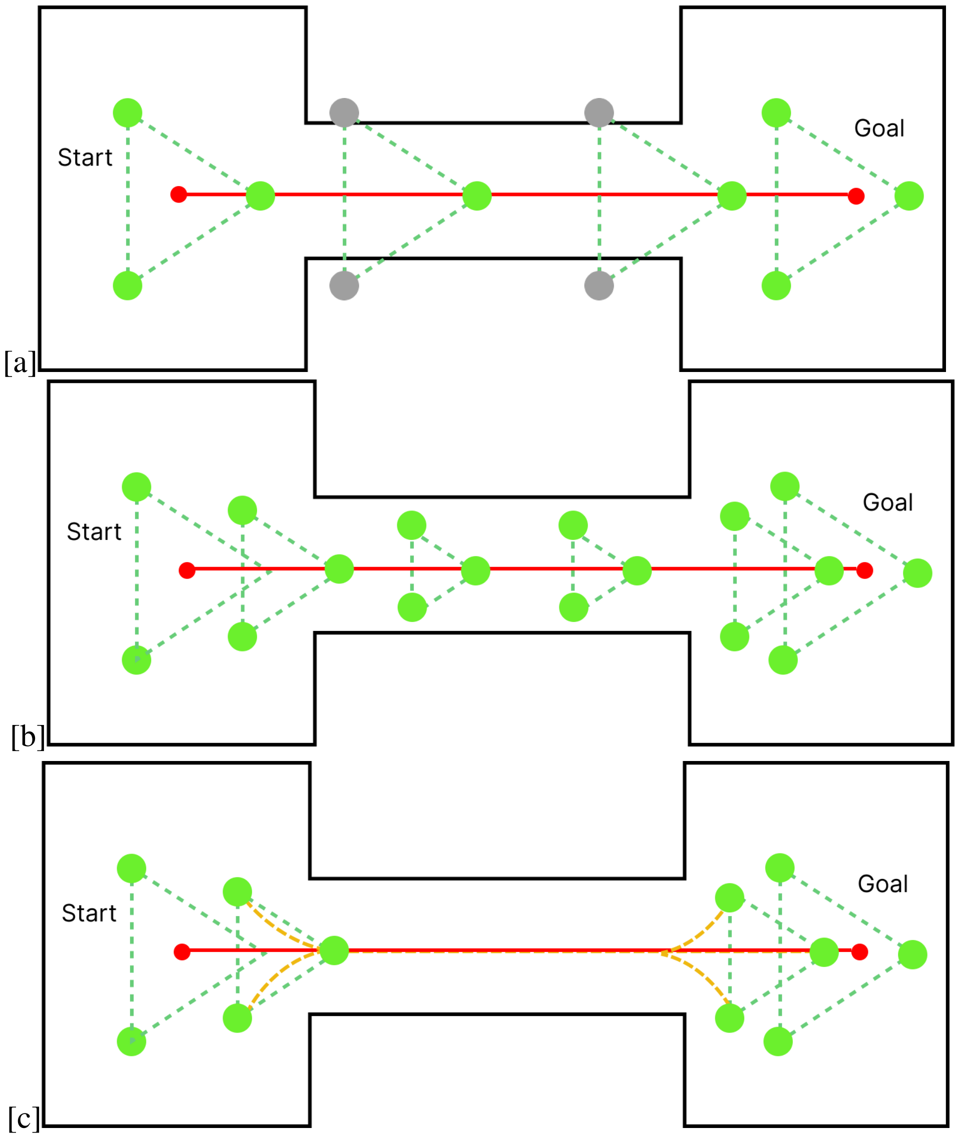


Image from author

Figure 4.3: Formation path with (SANTOS, 2022) method (a) and our method (b). Robots are illustrated as green circles, the start and goal as red circles connected by a red line (representing the path to the goal), formation geometry in green lines, and the individual robot path in a broken formation. In the image, we can see that the first strategy cannot pass small corridors and the second one can.

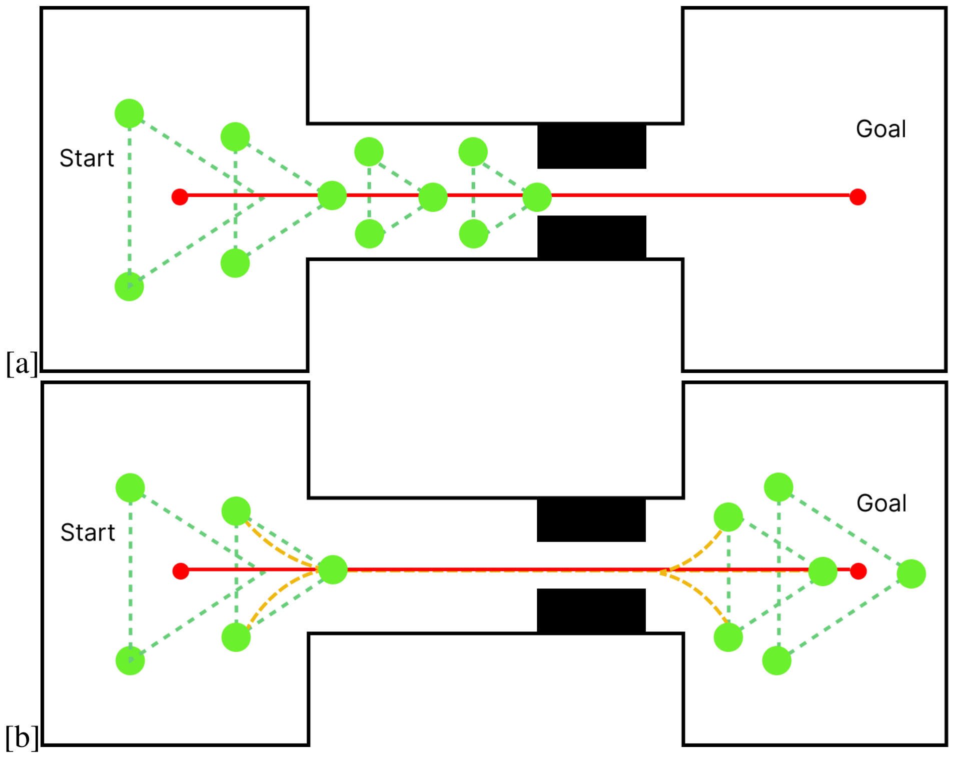


Image from author

5 TECHNOLOGIES AND LIBRARIES

This chapter presents an overview of the technologies used to develop the simulations and experiments made in this work, focusing on the Robot Operating System (ROS) as the main tool, and other software attached to ROS like the Gazebo Simulator. All of them simulate a real robot platform called TurtleBot3.

5.1 ROS

Since the creation of the first robots, several frameworks have been developed to help build more complex and dynamic applications, but Robot Operating System (ROS) has proved to be a reliable and popular framework amongst research communities and has become the standard for robotics research and development. (SONI; HU, 2019)

Developed by the Stanford Artificial Intelligence Laboratory in support of the Stanford AI Robot project in 2007, ROS is a set of open-source software libraries and tools that help build robot applications. It is similar to an operating system on a personal computer, but in the case of ROS however, these programs allow a user to control the mobile operations of a robot rather than applications on a computer (KERR; NICKELS, 2012)

The objective of ROS is not centered around being a framework that boasts the most extensive range of features. Instead, the primary aim of ROS is to facilitate the reuse of code in the field of robotics research and development.

For that, ROS has three levels of base concepts: the Filesystem level, the Computation Graph level, and the Community level.

Filesystem level concepts mainly cover ROS resources that you encounter on disk, such as packages, metapackages, package manifests, Repositories and message types

"The Computation Graph is the peer-to-peer network of ROS processes that are processing data together" (Stanford Artificial Intelligence Laboratory et al.,). In this graph, one of the base concepts used are the nodes.

A node is a process that performs computation and a robot control system (RCS) can contain many of those working simultaneously. These processes can be organized into Packages and Stacks, which can be effortlessly shared and distributed. Nodes communicate via messages, which contain data structures with standard types such as integer, floating point, boolean, etc.

Figure 5.1: ROS node communication interface

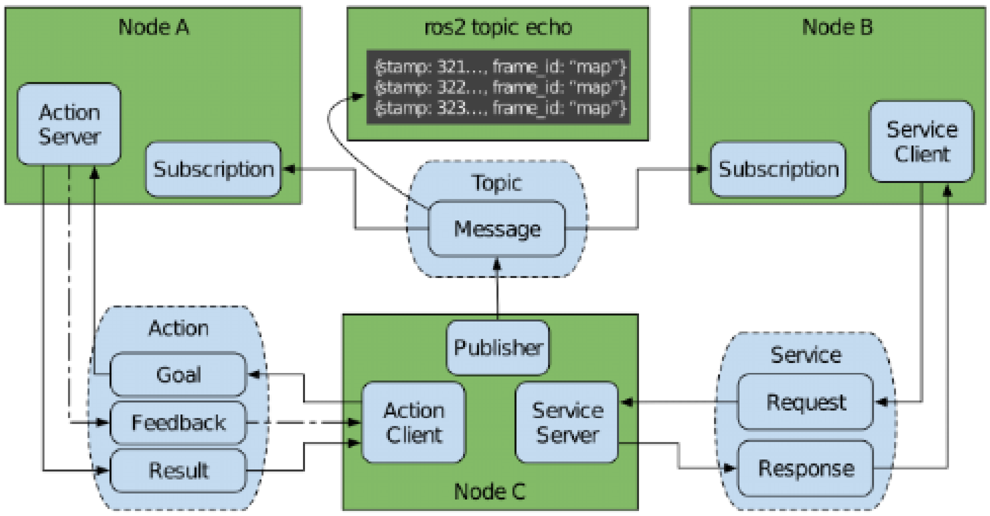


Image from (MACENSKI et al., 2022)

Those messages are routed via publish / subscribe semantics, where a node can publish data into a given topic, where the topic is a name that is used to identify the content of the message, and another node that is interested in a certain kind of data can subscribe to the topic. In one topic, there can be multiple publishers a subscribers, and a node can publish and/or subscribe to multiple topics, configuring a many-to-many relationship.

For a request and replay kind of communication of nodes, a service is used. Services are executed through the use of two message structures, a request structure and a reply structure. A provider node offers a service with a designated name, while a client accesses the service by sending a request message and patiently waiting for the corresponding reply.

For periodic communication of nodes, action is a distinctive communication pattern in ROS. It serves as a goal-oriented and asynchronous communication interface, encompassing a request, response, periodic feedback, and the ability to be canceled. All types of communication cannot be completed without a ROS Master, it acts as a nameservice in the ROS Computation Graph storing topics and services registration information for ROS nodes.

Additionally, ROS supports a system of code repositories that enables collaboration to be distributed, as well as mailing lists for ros-users to know about new updates and a ROS community Wiki for documenting information. This design, empowers developers to make independent decisions, while still being able to integrate their work seamlessly using ROS infrastructure tools. (Stanford Artificial Intelligence Laboratory et al.,)

Figure 5.2: General Structure of Gazebo components

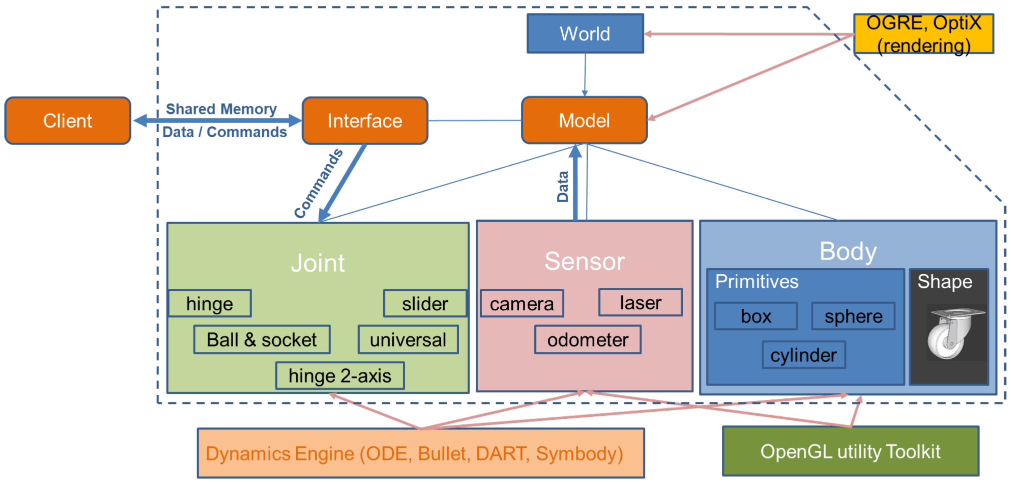


Image from (RIVERA; SIMONE; GUIDA, 2019)

5.2 Gazebo Simulator

Started in 2002, Gazebo is a collection of open-source software libraries that enable developers to design high-performance applications, usually within the robotics field of research, for physics simulation, rendering, user interface, communication, and sensor generation. By utilizing Gazebo, it becomes possible to simulate robots within a three-dimensional environment with realistic sensor feedback and physics interactions between objects by incorporating a precise simulation of rigid-body physics.

In the most recent version of ROS, Gazebo has been established as an independent system that operates separately from ROS. However, it is also feasible to configure Gazebo as a node, enabling it to simulate robot motion and facilitate data exchange with other nodes. Like ROS, Gazebo has its core principles based on modularity, extensibility, flexibility, stability and etc. (QIAN et al., 2014)

Its architecture can be described in Figure 5.2. The World encompasses all models and environmental elements, including gravity and lighting. Each model consists of a minimum of one body, along with multiple joints and sensors. The third-party libraries interact with Gazebo at the lowest level, ensuring that models do not rely on particular tools that might undergo changes in the future. Lastly, client commands are received and data is exchanged through a shared memory interface. A model can possess multiple interfaces to facilitate various functions, such as joint control and transmission of camera images. (KOENIG; HOWARD, 2004)

Gazebo uses an open-source physics engine called The Open Dynamics Engine (LEGER, 2000) as well as OpenGL and GLUT (OpenGL Utility Toolkit) for visualization

Figure 5.3: A TurtleBot3 robot

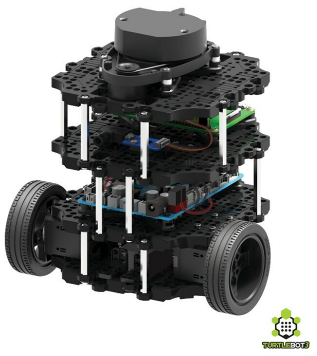


Image from ROBOTIS e-Manual

tools, which is a standard library for the creation of 2D and 3D interactive applications.

A model is any object that maintains a physical representation with at least one rigid body. Bodies serve as the fundamental components of a model, constituting its essential building blocks. Their physical representation takes the form of geometric shapes chosen from boxes, spheres, cylinders, planes, and lines. Each body possesses specific attributes such as mass, friction, bounce factor, as well as rendering properties like color, texture, transparency, and so forth.

Joints serve as the means to establish connections between bodies, enabling the formation of kinematic and dynamic relationships. There exists a range of joints, such as hinge, slider joints, ball and socket joints, and universal joints. When a force is applied to a joint, the friction between the connected body and other bodies causes motion. (KOENIG; HOWARD, 2004)

A Gazebo sensor is a conceptual device that does not have a tangible form or representation. This characteristic enables the sensors to be utilized in multiple models, resulting in a reduction of code and minimizing any potential confusion. The main sensors used are for odometer, range finders, and camera vision.

5.3 TurtleBot3

TurtleBot is a ROS standard robotic platform and TurtleBot3 [Figure 5.3] is the third version of the TurtleBot model, which is an affordable, programmable, ROS-based mobile robot for education and research use. It also provides a group of packages for SLAM, Navigation and Manipulation which facilitate the the development and simulation of new algorithms.

Its TurtleBot3 Burger model has an 360 LiDAR for SLAM and Navigation, attached to a single board computer, being it a Rasberry Pi, an OpenCR (Open-source Control Module), which is an open source robot controller embedded with a powerful MCU from the ARM Cortex-M7 line-up and a Li-Po battery.

A 360 Lidar (Light detection and ranging) utilizes laser beams that are safe for the eyes to perceive the surrounding world in three dimensions. This enables machines and computers to obtain a precise representation of the environment they are immersed in.

6 EXPERIMENTS

The following experiments were conducted in an Ubuntu 20.04 with Intel Core i7-4510U, 8GB of RAM and Intel Haswell-ULT Integrated Graphics, using ROS Noetic and simulated with the Turtlebot3 robotic platform environment.

The tests were developed using the Gazebo integration with ROS. The algorithms used were Gmapping technique, generating our occupancy grid map, and Timed-Elastic-Band local planner.

Two different configurations of scenarios were made with different formations to test our proposed strategy, (SILVEIRA; SILVA; NEDEL, 2008) and (SANTOS, 2022) methods. For each experiment, the algorithm ran five times for the data collection.

6.1 Tunnel map

The first map used called "Tunnel", as shown in Figure 6.1, is composed of two large rooms separated by a straight corridor connection, the corridor is $4m$ long and $2m$ large. Three types of formation configurations were used, the first one is a triangle with three robots, a square with four robots and lastly, an "X" formation with five robots, as shown in Figure 6.2. To compare our method and (SANTOS, 2022) method, we used $\varepsilon = -0,4$, max formation velocity $v_{max} = 0.3m/s$ and $th = 0,6$. Also, were tested the same configuration for (SILVEIRA; SILVA; NEDEL, 2008) method, using $\varepsilon = -0,4$ and max formation velocity $v_{max} = 0.05m/s$.

In (SILVEIRA; SILVA; NEDEL, 2008) method, the time remained the same throughout all experiments, but with more robots, there was a smaller distance they had to travel to reach the goal [Table 6.1]. Comparing all methods, this one had the biggest distance between robots during the navigation, because the distance of the formation center and the robot's designated formation point did not change during the algorithm.

Figure 6.1: "Tunnel" map used for testing

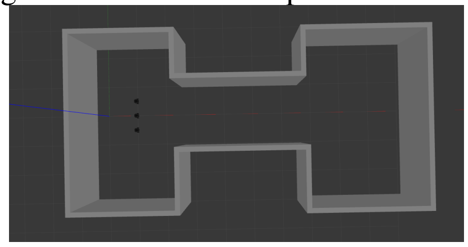


Image from author

Figure 6.2: Formation with 3 robots in the shape of a triangle (top left), 4 robots in the shape of a square (top right) and 5 robots in the shape of an "X"(bottom)

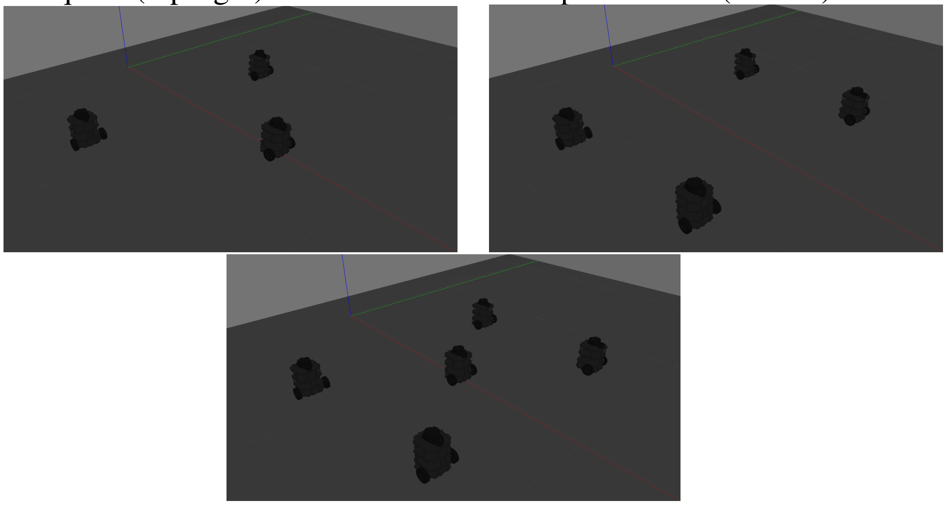


Image from author

Also, we can see a very similar time average for both our and (SANTOS, 2022) method [Table 6.1], in (SANTOS, 2022) we can see that the robot's distance to one another during the navigation was bigger, but our method had a decrease in the distance traveled by the robots to reach the goal [Table 6.2].

During all simulations, there was no collision between robots, but because of the small distance between robots, in some of the experiments we can see in the maps showing the individual movement of the group members using our method 6.8 that they did not have smooth navigation along the path when more robots were added to the formation, we can see some rapid direction changes when using four and five robots, something less present in (SANTOS, 2022) and (SILVEIRA; SILVA; NEDEL, 2008) method. On the other hand, in (SILVEIRA; SILVA; NEDEL, 2008) we can notice that with more robots, the navigation path became more constant, proven by the difference in distance traveled by each robot.

6.2 Small Tunnel map

In the second experiment, we used a map called "Small Tunnel", as shown in Figure 6.3, which is composed of two large rooms separated by a straight corridor connecting them, the corridor is $2m$ long, $1.2m$ large in the first half and $0,6m$ large in the second half. The group geometry used was the triangle formation of three robots [Figure 6.2 (top left)]. The methods tested were ours and (SANTOS, 2022) method, using $\varepsilon = -0,4$, max formation velocity $v_{max} = 0.3m/s$ and $th = 0,6$.

Figure 6.3: "Small Tunnel" map used for testing

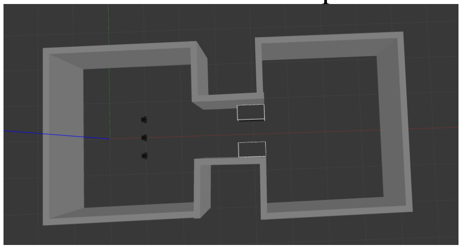


Image from author

In this map, during the experiences our method would finish the simulations reaching the goal without collisions, but in (SANTOS, 2022) strategy, sometimes the robots would get stuck in the middle of the smaller gap of the corridor as shown in Figure 6.9, showing an advantage of our strategy that did not stall.

This happens because in the narrow area this method (SANTOS, 2022) cannot maintain a formation with safe distances between the robots. In case the formation does not reach the goal, with our method, there is no such problem.

Table 6.1: Table with the time average results in seconds from our method (SILVEIRA; SILVA; NEDEL, 2008), (SANTOS, 2022) and our proposal in map Tunnel

Num Robots	SILVEIRA	SANTOS	Proposal
3	136,39 $\pm 0,46$	59,03 $\pm 2,90$	61,26 $\pm 7,58$
4	135,74 $\pm 2,24$	91,32 $\pm 12,30$	90,78 $\pm 9,66$
*5	137,50 $\pm 0,35$	126,11 $\pm 20,44$	108,48 $\pm 2,81$

Table 6.2: Table with minimum and maximum distance average between robots in meters, speed average in meters per second, and distance average in meters, results from our method, (SANTOS, 2022) and (SILVEIRA; SILVA; NEDEL, 2008) in map Tunnel

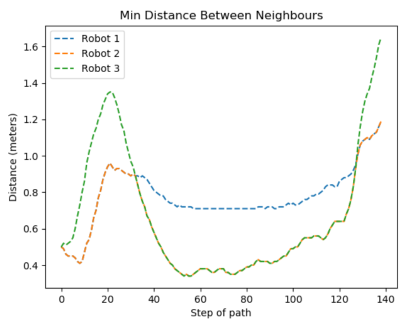
Method	Num Robots	Min Dist Avg	Max Dist Avg	Speed Avg	Distance Avg
Proposal	3	0,71 $\pm 0,07$	1,08 $\pm 0,15$	0,12 $\pm 0,01$	7,30 $\pm 0,17$
	4	0,70 $\pm 0,06$	1,36 $\pm 0,05$	0,09 $\pm 0,01$	7,46 $\pm 0,11$
	*5	0,58 $\pm 0,02$	1,31 $\pm 0,06$	0,06 $\pm 0,00$	6,80 $\pm 0,30$
SANTOS	3	1,18 $\pm 0,05$	1,70 $\pm 0,06$	0,13 $\pm 0,00$	7,95 $\pm 0,14$
	4	0,86 $\pm 0,06$	1,80 $\pm 0,08$	0,08 $\pm 0,01$	7,82 $\pm 0,10$
	*5	0,75 $\pm 0,03$	1,73 $\pm 0,04$	0,05 $\pm 0,00$	7,13 $\pm 0,13$
SILVEIRA	3	1,29 $\pm 0,11$	2,01 $\pm 0,19$	0,06 $\pm 0,00$	9,11 $\pm 0,45$
	4	0,87 $\pm 0,03$	1,94 $\pm 0,26$	0,06 $\pm 0,00$	8,32 $\pm 0,48$
	*5	0,81 $\pm 0,03$	2,02 $\pm 0,10$	0,05 $\pm 0,00$	7,23 $\pm 0,17$

* In the 5 robots test the goal distance was reduced in 0.5 meters

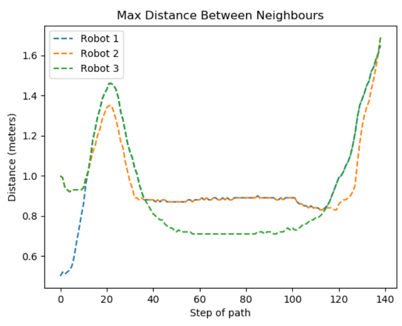
Table 6.3: Table with minimum and maximum distance average between robots in meters, speed average in meters per second, and distance average in meters, time average, results from our method in map Small Tunnel

Num Robots	Min Dist Avg	Max Dist Avg	Speed Avg	Distance Avg	Time Avg
3	0,69 $\pm 0,05$	1,20 $\pm 0,11$	0,10 $\pm 0,01$	5,11 $\pm 0,3$	52,59 $\pm 11,89$

Figure 6.4: Distance in meters of the nearest (a, c, e) and furthest (b, d, f) neighbor robot in map "Tunnel" using 3 robots with our method (a, b), (SANTOS, 2022) (c, d) and (SILVEIRA; SILVA; NEDEL, 2008) (e,f)

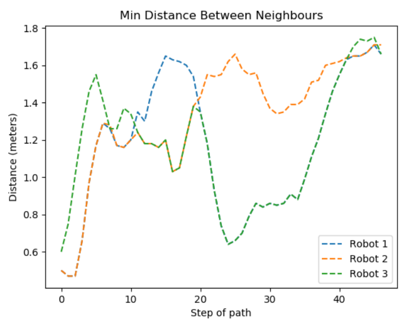


[a]

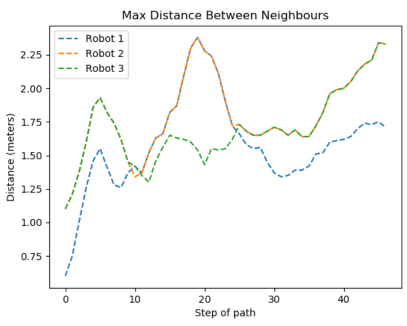


[b]

Proposal

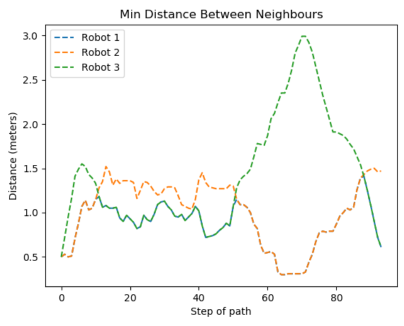


[c]

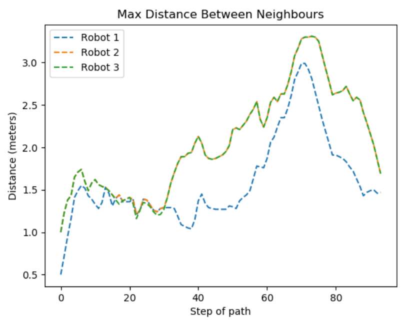


[d]

(SANTOS, 2022)



[e]

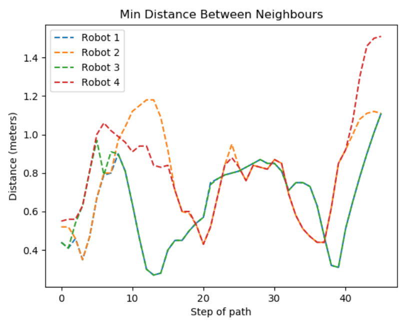


[f]

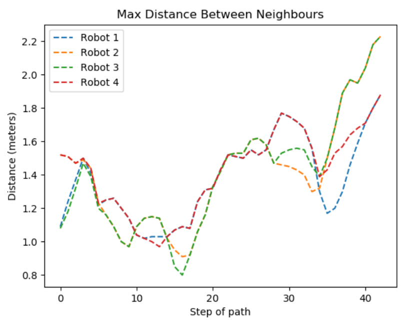
(SILVEIRA; SILVA; NEDEL, 2008)

Image from author

Figure 6.5: Distance in meters of the nearest (a, c, e) and furthest (b, d, f) neighbor robot in map "Tunnel" using 4 robots with our method (a, b), (SANTOS, 2022) (c, d) and (SILVEIRA; SILVA; NEDEL, 2008) (e,f)

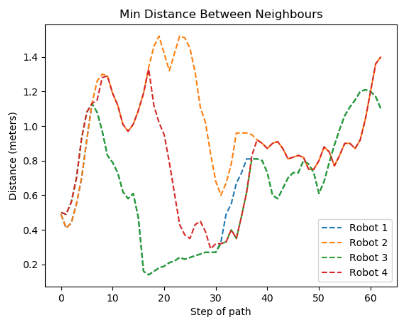


[a]

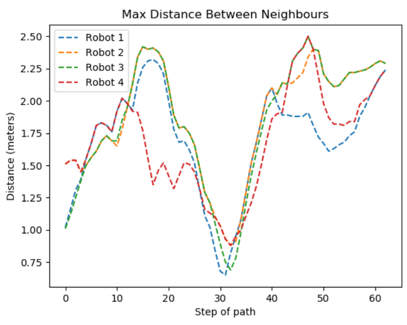


[b]

Proposal

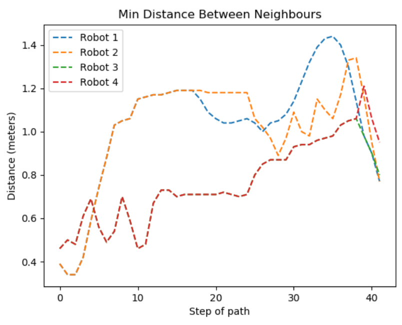


[c]

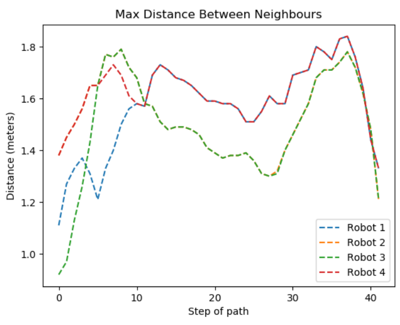


[d]

(SANTOS, 2022)



[f]

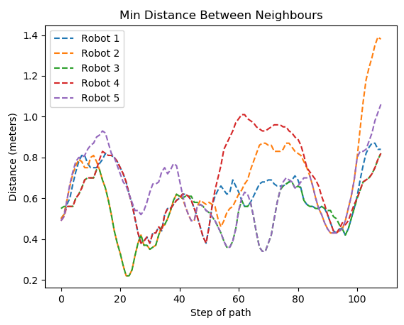


[g]

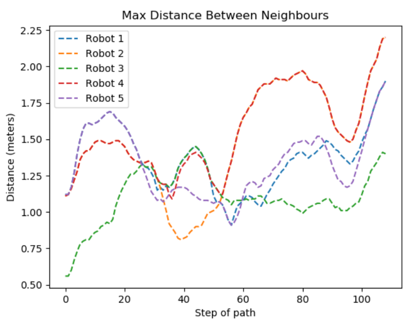
(SILVEIRA; SILVA; NEDEL, 2008)

Image from author

Figure 6.6: Distance in meters of the nearest (a, c, e) and furthest (b, d, f) neighbor robot in map "Tunnel" using 5 robots with our method (a, b), (SANTOS, 2022) (c, d) and (SILVEIRA; SILVA; NEDEL, 2008) (e,f)

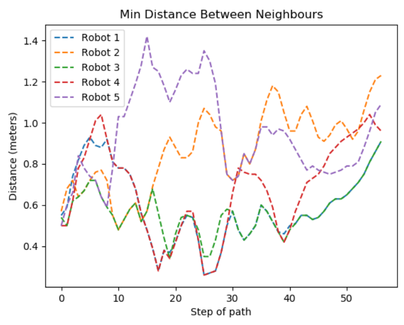


[a]

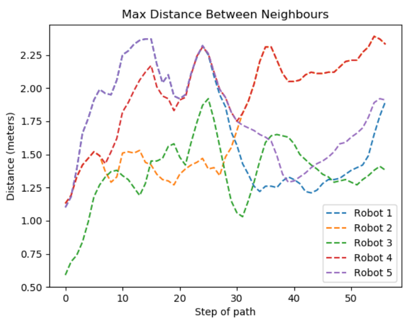


[b]

Proposal

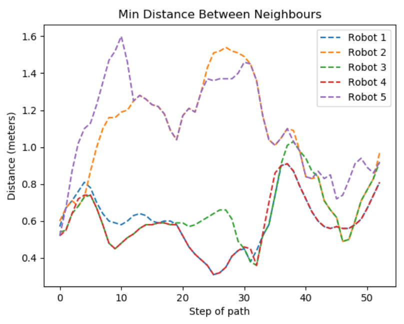


[c]

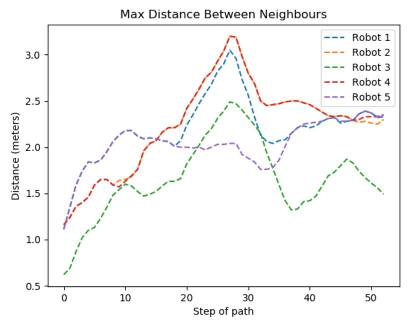


[d]

(SANTOS, 2022)



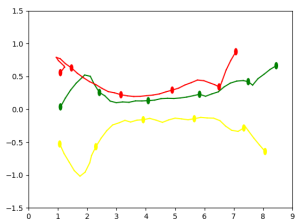
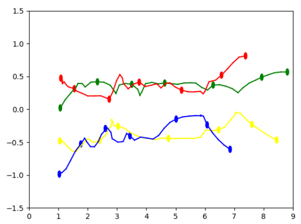
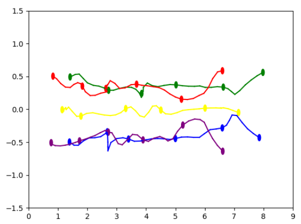
[f]



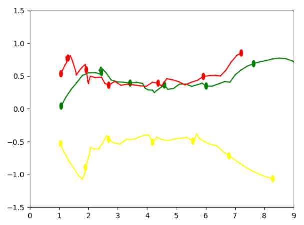
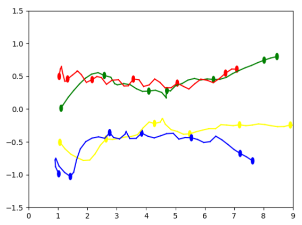
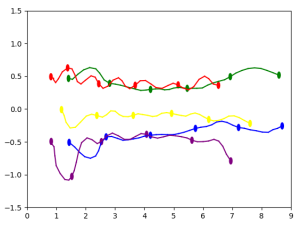
[g]

(SILVEIRA; SILVA; NEDEL, 2008)
Image from author

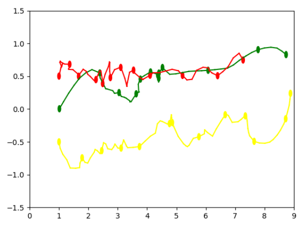
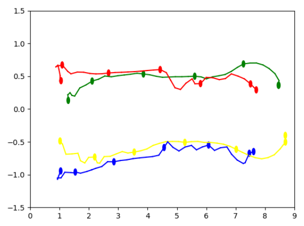
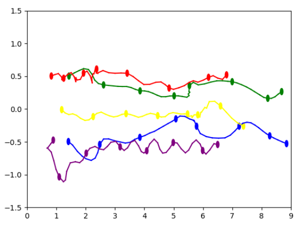
Figure 6.7: Trajectory of the robots using our method (a, b, c), (SANTOS, 2022) (d, e, f) and (SILVEIRA; SILVA; NEDEL, 2008) method (g, h, i) using 3 (a, d, g), 4 (b, e, h) and 5 (c, f, i) robots in map "Tunnel". Each color indicates the path traveled by a different robot, starting on the left and ending on the right.


[a]

[b]

[c]

Proposal

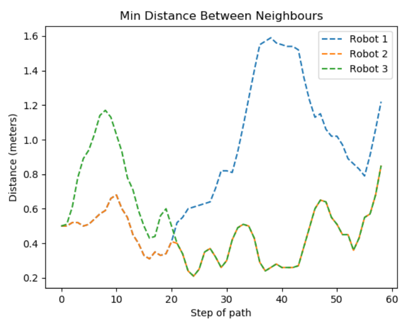

[d]

[e]

[f]

(SANTOS, 2022)

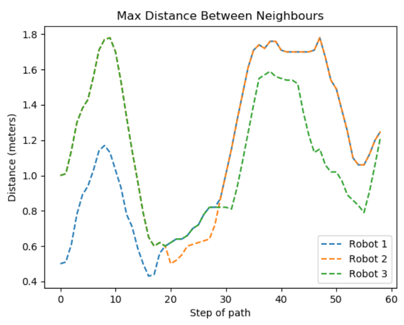

[g]

[h]

[i]

(SILVEIRA; SILVA; NEDEL, 2008)
Image from author

Figure 6.8: Distance in meters of the nearest (a) and furthest (b) neighbor robot in map "Small Tunnel" using 3 robots with our method



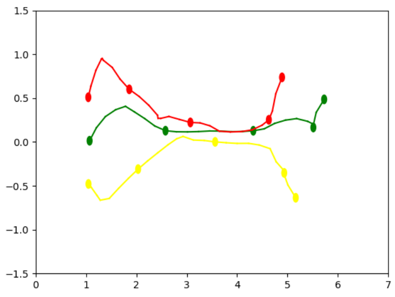
[a]



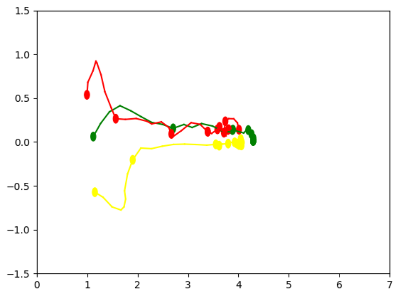
[b]

Proposal
Image from author

Figure 6.9: Trajectory of the robots using our method (a) and (SANTOS, 2022) method (b), where our formation reached the goal and the other one did not in map "Small Tunnel". Each color indicates the path traveled by a different robot, starting on the left and ending on the right.



[a]



[b]

Image from author

7 CONCLUSION

In this paper, we presented a system to break the formation of multiple robots when the environment is too small for the minimum geometry formation size, experimented via ROS simulations. It can be classified as a centralized and dynamic coordination algorithm, with implicit and weak coordination. The results show that our method doesn't compromise the time for the formation to reach its goal and reduces the distance traveled by the robots to reach their destination.

The use of BVP-based potential fields makes it possible to take advantage of the natural features of this type of navigation approach, especially for smooth obstacle avoidance, which includes static obstacles in the environment and other robots, and, as we can see, works both in areas where the formation is active or in narrower areas where the formation is disabled. Compared to the other tested approaches, the proposed strategy is able to pass through narrow areas by momentarily breaking formation and recovering it as soon as possible, unlike the other methods that suffer in such conditions.

For future articles, we pretend to do more experiments in different environment configurations with more robots and reconfigure the potential fields during the broken formation phase to guarantee more space between robots, without compromising time and distance navigated.

REFERENCES

BORENSTEIN, J.; KOREN, Y. Real-time obstacle avoidance for fast mobile robots. **Systems, Man and Cybernetics, IEEE Transactions on**, v. 19, p. 1179 – 1187, 10 1989.

BURGARD, W.; HEBERT, M. World modeling. In: ____. **Springer Handbook of Robotics**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 853–869. ISBN 978-3-540-30301-5. Available from Internet: <https://doi.org/10.1007/978-3-540-30301-5_37>.

CONNOLLY, C.; BURNS, J.; WEISS, R. Path planning using laplace's equation. In: . [S.l.: s.n.], 1990. p. 2102 – 2106 vol.3.

DAPPER, F.; PRESTES, E.; NEDEL, L. P. Generating steering behaviors for virtual humanoids using bvp control. In: . [S.l.: s.n.], 2007.

DORIGO, M.; BIRATTARI, M.; STUTZLE, T. Ant colony optimization. **IEEE Computational Intelligence Magazine**, v. 1, n. 4, p. 28–39, 2006.

DORIGO, M.; THERAULAZ, G.; TRIANNI, V. Swarm robotics: Past, present, and future [point of view]. **Proceedings of the IEEE**, v. 109, n. 7, p. 1152–1165, 2021.

ELFES, A. Using occupancy grids for mobile robot perception and navigation. **Computer**, v. 22, n. 6, p. 46–57, 1989.

IOCCHI, L.; NARDI, D.; SALERNO, M. Reactivity and deliberation: A survey on multi-robot systems. In: **Balancing Reactivity and Social Deliberation in Multi-Agent Systems**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 9–32. ISBN 978-3-540-44568-5.

KERR, J.; NICKELS, K. Robot operating systems: Bridging the gap between human and robot. In: **Proceedings of the 2012 44th Southeastern Symposium on System Theory (SSST)**. [S.l.: s.n.], 2012. p. 99–104.

KOENIG, N.; HOWARD, A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In: **2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)**. [S.l.: s.n.], 2004. v. 3, p. 2149–2154 vol.3.

LEGER, C. Darwin2k: An evolutionary approach to automated design for robotics. In: . [s.n.], 2000. Available from Internet: <https://api.semanticscholar.org/CorpusID:61104196>.

LEWIS, M. A.; TAN, K. H. High precision formation control of mobile robots using virtual structures. **Autonomous Robots**, v. 4, p. 387–403, 1997. Available from Internet: <https://api.semanticscholar.org/CorpusID:17306871>.

MACENSKI, S. et al. **Robot Operating System 2: Design, Architecture, and Uses In The Wild**. 2022.

MORAVEC, H.; ELFES, A. High resolution maps from wide angle sonar. In: **Proceedings. 1985 IEEE International Conference on Robotics and Automation**. [S.l.: s.n.], 1985. v. 2, p. 116–121.

OGREN, P.; FIORELLI, E.; LEONARD, N. Cooperative control of mobile sensor networks:adaptive gradient climbing in a distributed environment. **IEEE Transactions on Automatic Control**, v. 49, n. 8, p. 1292–1302, 2004.

OLCAY, E.; SCHUHMANN, F.; LOHMANN, B. Collective navigation of a multi-robot system in an unknown environment. **Robotics and Autonomous Systems**, v. 132, p. 103604, 2020. ISSN 0921-8890. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0921889020304449>.

PARKER, L. E. Path planning and motion coordination in multiple mobile robot teams. 2009.

QIAN, W. et al. Manipulation task simulation using ros and gazebo. In: **2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)**. [S.l.: s.n.], 2014. p. 2594–2598.

REN, W.; BEARD, R. W. Distributed formation control of multiple wheeled mobile robots with a virtual leader. In: ____. **Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications**. London: Springer London, 2008. p. 193–205. ISBN 978-1-84800-015-5. Available from Internet: <https://doi.org/10.1007/978-1-84800-015-5_10>.

RIVERA, Z. B.; SIMONE, M. C. D.; GUIDA, D. Unmanned ground vehicle modelling in gazebo/ros-based environments. **Machines**, v. 7, n. 2, 2019. ISSN 2075-1702. Available from Internet: <https://www.mdpi.com/2075-1702/7/2/42>.

SABUDIN, E. N.; OMAR, R. B.; MELOR, C. K. N. A. H. C. K. Potential field methods and their inherent approaches for path planning. In: . [s.n.], 2016. Available from Internet: <https://api.semanticscholar.org/CorpusID:34943571>.

SANTOS, L. dos. Autonomous navigation of robot swarms; defining elastic formations as dynamic boundaries for potential fields. 2022.

SCHNEIDER, F.; WILDERMUTH, D. A potential field based approach to multi robot formation navigation. In: **IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003**. [S.l.: s.n.], 2003. v. 1, p. 680–685 vol.1.

SHAO, J.; WANG, L.; XIE, G. Flexible formation control for obstacle avoidance based on numerical flow field. In: **Proceedings of the 45th IEEE Conference on Decision and Control**. [S.l.: s.n.], 2006. p. 5986–5991.

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D. **Introduction to Autonomous Mobile Robots**. 2nd. ed. [S.l.]: The MIT Press, 2011. ISBN 0262015358.

SILVEIRA, R.; SILVA, E. P. e; NEDEL, L. P. Managing coherent groups. **Computer Animation and Virtual Worlds**, v. 19, 2008.

SONI, A.; HU, H. A multi-robot simulator for the evaluation of formation control algorithms. In: **2019 11th Computer Science and Electronic Engineering (CEEC)**. [S.l.: s.n.], 2019. p. 79–84.

Stanford Artificial Intelligence Laboratory et al. **Robotic Operating System**. Available from Internet: <https://www.ros.org>.

TREVISAN, M. et al. Exploratory navigation based on dynamical boundary value problems. **Journal of Intelligent and Robotic Systems**, v. 45, n. 2, p. 101–114, Feb 2006. ISSN 1573-0409. Available from Internet: <https://doi.org/10.1007/s10846-005-9008-2>.

VERMA, J. K.; RANGA, V. Multi-robot coordination analysis, taxonomy, challenges and future scope. **Journal of Intelligent & Robotic Systems**, v. 102, 2021. Available from Internet: <https://api.semanticscholar.org/CorpusID:233256952>.

YAN, Z.; JOUANDEAU, N.; CHERIF, A. A. A survey and analysis of multi-robot coordination. **International Journal of Advanced Robotic Systems**, v. 10, n. 12, p. 399, 2013. Available from Internet: <https://doi.org/10.5772/57313>.