République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Chadli Bendjedid
Faculté des Sciences et de la Technologie
Département d'Informatique

الجمهـــوريـــة الجزائريـــة الديمقراطيـــة الشـعبيـــة
وزارة التعليـــم العـــالي والبحث العلمـــي
جامعـــة الشـــاذلي بن جـديـــد
كليـــة العـلـوم و التكـنـولوجيـــا
قســـم الاعـــلام الالـــي

جامعة الشاذلي بن جديد
UNIVERSITE CHADLI BENDJEDID

Master Thesis in Computer Science

Presented by

Khaoula BOUMAHNI

Specialty: Intelligent Computer Systems

Theme

Trajectory planning of a mobile robot based on the Bacterial Foraging Optimization Algorithm

Presented on :

Mr Touahri D	**MAA**	**Univ.of el tarf**	**President**
Mr Benmachiche A	**MCA**	**Univ.of el tarf**	**Supervisor**
Mr Betouil AA	**MCB**	**Univ.of el tarf**	**Examiner**

University Year: 2021/2022

Thanks

My thanks go first to God Almighty for the will, health and patience he gave me during all these years of study.

In particular, I would like to thank **Mr. Benmachiche A** , Senior Lecturer in the Department of Computer at Chadli Ben Djedid University, who has mentored me throughout this thesis, for his valuable advice and support. I am grateful for his availability and the trust he has placed in me. My gratitude also to the team of teachers of the Option, Artificial Intelligence.

I sincerely thank:

- **Mr DjamelEddine Touahri**, for having honored me to chair the jury
- **Dr Ali Abdelatif Betouil**, for agreeing to be an examiner.
- Thank **Dr Anguel F** I wish she was with us to thank her, because of her I loved the specialty.

A big thanks to all my family for their concern and concern for me, their encouragement and followed them with patience during the course of my studies. My thanks also go to my colleagues, for the good times we passed together. Finally, I thank all those who helped me from afar or from the meadows

Thank you all, from the bottom of the heart

Dedications

I dedicate this modest work

To the one who represents my world and my happiness

To the one who represents the day and the light of my life

To the one who always wakes up to see me the best

To the one who opened the portals for me and gave me tenderness and courage.

To the one who mourns to see me happy to the one who warmly awaits this day

"My Dear Mother"

To the one who made great efforts for my happiness.

To the one who dreamed of seeing this day.

To the one who guided me and taught me the secrets of life

"My Father"

To my **Brothers**, and my **Sister** whom I love too much

To my family in general and To my best friend **Kaouther**

To my friends who have always been next to me and supported me

To all those who have helped me from near or far .

Abstract

The design of artificial systems has undergone an epistemic revolution throughout the past fifty years. It entailed transcending the traditional dichotomy between the organic and inorganic worlds by attempting to imbue machines with the adaptability and autonomy found in living systems. Evolutionary robotics strives to create devices that can learn new skills on their own in an ever-changing, uncontrollable environment. This technology allowed for the construction of genuine robots with complicated reactive behaviors. Based on this observation, our work presents a non-exhaustive view of the research themes related to the field of mobile robotics, as well as the scientific hurdles that must be overcome before an autonomous robot may be developed. The latter's autonomy necessitates the accomplishment of tasks of control and perception of the environment in a coordinated manner.

Navigation is one of these, and it is critical to the robot's interaction with its evolutionary environment. It entails determining the robot's trajectories to follow a pre-determined path while avoiding mobile or immovable impediments.

Our solution is based on Bacterial Foraging Optimization Algorithm to find the shortest path. After then, the navigation problem is described as a constraint optimization problem, with a fitness function that quantifies the difference between the best robot path and the other random paths.

The impediments are implemented as limitations that penalize the robots' movement, to allow them to change positions while avoiding obstructions.

Our strategy has been put into action, and various situations have been put to the test. The collected results indicate the method's robustness as well as its performance.

Résumé

La conception des systèmes artificiels a connu une révolution épistémique au cours des cinquante dernières années. Il s'agissait de transcender la dichotomie traditionnelle entre les mondes organique et inorganique en tentant d'imprégner les machines de l'adaptabilité et de l'autonomie que l'on retrouve dans les systèmes vivants. La robotique évolutive s'efforce de créer des appareils capables d'apprendre de nouvelles compétences par eux-mêmes dans un environnement en constante évolution et incontrôlable. cette technologie a permis la construction de véritables robots aux comportements réactifs compliqués. Partant de ce constat, notre travail présente une vue non exhaustive des thématiques de recherche liées au domaine de la robotique mobile, ainsi que des verrous scientifiques qui doivent être surmontés avant qu'un robot autonome puisse être développé. L'autonomie de ce dernier nécessite l'accomplissement de tâches de contrôle et de perception de l'environnement de manière coordonnée.

La navigation est l'une d'entre elles, et elle est essentielle à l'interaction du robot avec son environnement évolutif. Il s'agit de déterminer les trajectoires du robot pour suivre un chemin prédéterminé tout en évitant les obstacles mobiles ou immobiles.

Notre solution est basée sur l'algorithme d'optimisation de l'alimentation bactérienne du plus court chemin. Ensuite, le problème de navigation est décrit comme un problème d'optimisation de contraintes, avec une fonction de fitness qui quantifie la différence entre la meilleure trajectoire du robot et les autres trajectoires aléatoires.

Les entraves sont implémentées comme des limitations qui pénalisent le mouvement des robots, pour leur permettre de changer de position tout en évitant les obstructions.

Notre stratégie a été mise en œuvre et diverses situations ont été mises à l'épreuve. Les résultats recueillis indiquent la robustesse de la méthode ainsi que ses performances.

Abstract

ملخص

شهد تصميم الأنظمة الاصطناعية ثورة معرفية على مدار الخمسين عامًا الماضية. لقد استلزم تجاوز الانقسام التقليدي بين العالمين العضوي وغير العضوي من خلال محاولة إضفاء القدرة على التكيف والاستقلالية الموجودة في الأنظمة الحية على الآلات.

تسعى الروبوتات التطورية جاهدة لإنشاء أجهزة يمكنها تعلم مهارات جديدة بمفردها في بيئة دائمة التغير ولا يمكن التحكم فيها. سمحت هذه التكنولوجيا ببناء روبوتات أصلية بسلوكيات تفاعلية معقدة. بناءً على هذه الملاحظة، يقدم عملنا وجهة نظر غير شاملة لموضوعات البحث المتعلقة بمجال الروبوتات المتنقلة، فضلاً عن العقبات العلمية التي يجب التغلب عليها قبل تطوير الروبوت مستقل . يستلزم استقلالية الأخير إنجاز مهام التحكم وإدراك البيئة بطريقة منسقة.

يعد التنقل أحد هذه العناصر ، وهو أمر بالغ الأهمية لتفاعل الروبوت مع بيئته التطورية. يستلزم تحديد مسارات الروبوت لإتباع مسار محدد مسبقًا مع تجنب العوائق المتحركة أو الثابتة.

يعتمد حلنا على خوارزمية تحسين التغذية البكتيرية لأقصر مسارً. بعد ذلك، يتم وصف مشكلة التنقل على أنها مشكلة تحسين القيد، مع وظيفة اللياقة التي تحدد الفرق بين أفضل مسار للروبوت والمسارات العشوائية الأخرى.

يتم تنفيذ العوائق كقيود تعاقب حركة الروبوتات، للسماح لها بتغيير المواضع مع تجنب العوائق.

لقد تم وضع استراتيجيتنا موضع التنفيذ ، وتم اختبار المواقف المختلفة. تشير النتائج التي تم جمعها إلى متانة الطريقة بالإضافة إلى أدائها.

.

Contents

Table of figures

List of tables

Abbreviation list

General introduction…….... ..1

Chapter 01 : navigation systems

1 . 1- Introduction:.....5

1.2-Navigation Strategies:.....5

1.2.1 -Artificial Potential Fields (APF):.....5

1.2.2 –Artificial neural network :.....9

1.2.3 –Fuzzy Logic :.....9

1.2.4 -Ant Colony:.....11

1.2.5 -Bee Colony:.....13

1.2.5.1 -Bee Colony Optimization.....13

1.2.5.2 -Dance Bee Optimization.....14

1.2.5.3-Artificial Bee Colony.....15

1.2.5.4- Virtual Bee Algorithm:.....16

1.2.6- Genetic Algorithm:.....16

1.2.6- Bacterial forging optimization:.....18

1.3-Comparison between Methods.....21

Conclusion.....22

Chapitre 2 : System Design

2.1- Introduction:.....24

2.2-Our Problem:.....24

2.3-The proposed solution: 25

2.3.1- Problem definition.....25

2.3.2- The Activity Diagram:.....25

2.3.3- **The Sequence Diagram**……27

Contents

2.3.4- The Class Diagram……28

2.3.4- The Use case Diagram……29

2.4- Conclusion: 33

Chapter 3: Realization and Implementation

3.1- Introduction:..... 34

3.2-Tools and working environments: 34

3.2.1- Software environment: 34

3.3-Description of the different modules: 35

3.3.1- The environment: 35

3.3.2- Initialization of Matrix : 37

3.3.3.-Robot move: 39

3.3.4.-The paths: 39

3.3.4.-The paths after entering the chemotaxis region: 40

A- **Simulation..... 41**

3.4.Discussion……..43

3.4.1-Comparison between different methods : 43

3.5.**Conclusion:..... 44**

General Conclusion

References

Table of Figures

Figure 1.1: Local and global minima in a system. 8
Figure 1.2: Artificial Neural network. 9
Figure 1.3: The fuzzy logic& Boolean logic 10
Figure 1.4: Membership functions 11
Figure 1.5: Ant Colony 12
Figure 1.6: Genetic Algorithm.. 17
Figure 2.1: Activity Diagramm 26
Figure 2.2: Sequence Diagramm 27
Figure 2.3: Class Diagramm 28
Figure 2.4: Robot use case diagram 30
Figure 2.5: User use case diagram 31
Figure 3.1: Matlab logo 35
Figure 3.2: Diagram representing the passage from a RGB matrixto a navigation environment. 36
Figure 3.3: Interface of Our Application 37
Figure 3.4: final matrix 38
Figure 3.5: The Four Directions of robots (bacterial) 39
Figure 3.6: Matlab Code of Direction 39
Figure 3.7: Matlab Code of distance 40
Figure 3.8: Matlab Code of Paths that bacteria can take when they enter the trophic zone. …..40
Figure3.9: Matlab Code The path of the first bacteria…….. ……..40
Figure3.10: Bacteria move in the ocean and avoid obstacles to find the shortest path 41
Figure3.11: The Shortest Path Found by the First Bacterial 42

List of tables

Table1.1: Comparison of the Methods…….……21

Table 3.1: Diagram representing a path found between 2 points ……...39

Table 3.2 : Data that was used to get the results……...42

Table 3.3: A comparison between the four algorithms about time and path length…….43

Abbreviations list

- **AI** : Artificial intelligence
- **APF**: Artificial Potential Fields
- **GA** : Genetic Algorithm
- **VBA** : Virtual Bee Algorithm
- **BCO** : Bee Colony Optimization
- **BOD** : Dance Bee Optimization
- **ABC** : Artificial Bee Colony
- **ANN** : Artificial Neural network.
- **BFO** : Bacterial forging optimization

General Introduction

Artificial Intelligence (AI) is a discipline of computer science in which machines are taught and constructed to emulate human decision-making and reaction functions without the need for human interaction.

The application of several approaches aimed at allowing robots to emulate a kind of actual intelligence is known as artificial intelligence (AI). Artificial intelligence is becoming increasingly important in all fields.

Alan Turing, a mathematician, created a test to measure artificial intelligence in 1950. The Turing Test checks if the answer can be determined by the machine in this manner through a sequence of questions.

Artificial intelligence is defined as a computer with reactions that are indistinguishable from those of humans. The goal of artificial intelligence is to create systems that can mimic human thinking behavior. The modeling of intelligence as a phenomenon is the goal of AI (as well as physics, chemistry, or biology which aim to model other phenomena).

Artificial intelligence is a scientific subject that focuses on studying, inventing, and implementing "smart machines." It's important to note that the term "machine" refers to an autonomous system capable of processing data rather than a physical item.

With the advent of computer science and technology, a larger area of inquiry in the field of robotics, which is concerned with devices that move around in space, has opened up. As a result, when we use the term "robot" in artificial intelligence, we're referring to a computer program that exhibits intelligence.

Robotics is a sub-domain of AI that can be thought of as an intelligent linkage of perception, action, and robot functionality. It helps robots acquire the ability to converse in normal language by allowing them to maintain dynamic representations of their environment.

Robotics is a great example of a multidisciplinary field that includes mechanics, mechatronics, electronics, automation, computer science, and artificial intelligence, to name a few.

There are numerous definitions of the term robot depending on the writers' region of origin, but they all revolve around this one: A robot is a machine with perception, decision, and action capabilities that enable it to behave independently in its surroundings based on his view of it.

General Introduction

The navigation systems that allow a mobile robot to move to achieve a goal, as well as the classifications that can be applied to them, are exceedingly diverse.

The problem of calculating the motions required for a robot to complete a job is referred to as movement planning in this framework.

In its most basic form, movement planning is defined as follows: given a model of the robotic system and its surroundings, planning a movement entails calculating the movement that the system must make in order to achieve an a priori determined goal.

The term "movement planning," often known as "autonomous navigation," refers to movement planning for robots or mobile agents.

"Autonomous navigation in a dynamic environment" is the problem addressed in this document.

Exact solutions for solving this problem ([1], [2], [3]) are inapplicable to complicated systems with numerous degrees of freedom.

The most successful motion planning systems use probabilistic algorithms to explore the space of configurations at random in order to characterize their connection[4] , or more current references [5], [6] provide a summary of various strategies.

One of his main achievements was the creation of a unique algorithmic framework that allows us to solve difficulties: bacterial methods for solving combinatorial problems.

These mechanisms are used by c-bacterial algorithms to address issues of combinatorial optimization.

The goal is to create an adaptive group for each group with the objective function to be improved. Bacterial algorithms are evolutionary approaches that rely heavily on biological factors associated with selection and principles of natural evolution.

Bacterial algorithms were first created by Passino [7]. It is worth noting that the concepts underlying bacterial algorithms are fairly basic. The document is divided into three chapters:

The presentation of mobile robots takes up the first chapter; the typology of mobile robots is examined after a brief survey of the topic of mobile robotics. We present the current state of navigation in general, as well as cooperative and autonomous navigation in dynamic situations with obstacle avoidance.

General Introduction

We propose a conceptual study to determine the shortest path of an autonomous robot using bacterial algorithms, contributing to the research challenge. The approach is used to simulate the environment, while the BFO method is presented separately. Simulated environments are briefly explored in the final chapter. To test the performance of our BFO algorithm, several experiments are conducted in the MATLAB development environment. The results of the presented methods for finding the shortest path for an autonomous robot via simulations are presented.

Finally, we conclude and review the entire thesis project, describing how we arrived at our conclusions through simulated experiments. The potential scope for future work of this thesis is discussed.

Chapter 1

Navigation Systems In Dynamic Environments

Chapter 1: Navigation Systems

1 – 1 Introduction:

An autonomous mobile robot is a mechanical, electronic, and computer system that interacts with its surroundings to achieve a certain goal. Even if he finds himself in unexpected situations without the need for human involvement. The ability of a mobile robot to adapt or make a decision to execute a task in an unfamiliar or unknown environment is known as autonomy.

The term "navigation" refers to the process of guiding vehicles from one location to another. The word is enlarged in mobile robotics to include all mechanisms that enable the mobile robot to detect and perform the movements required to achieve a certain goal.

One of the issues with a mobile robot system is moving in an unknown environment, and the independence of movement becomes a harder problem to overcome when compared to when the mobile robot grows in a well-known environment. The challenge of localization, which is the determination of the mobile robot's position and orientation ("position") in its surroundings, is another issue with mobile robotics. In a dynamic environment, the robot's navigation consists of moving from the beginning location to the destination position. The creation of an executable trajectory is required for robot motion planning. In robotics, planning trajectories to allow a mobile robot to move from an initial position to an end location is frequently required. Trajectory planning is a topic that is frequently discussed in scientific circles.

As a result, the most advanced autonomous robots' programming makes use of artificial intelligence and algorithms that provide the autonomous robot with some type of learning.

The goal of this project is to construct a presence-detecting robot that walks around a room and avoids obstacles without the assistance of humans.

Several algorithms solve this problem. In this chapter, we will talk about pre-existing works

1- 2. Navigation Strategies:

The navigation strategies allowing a mobile robot to move and arrive at a goal are exorbitant, as well as the classifications that can be made.

1- 2.1 Artificial Potential Fields (APF):

Artificial Potential Fields (APFs) have been used in real-time obstacle avoidance for over thirty years. They were first published in Khatib's seminal study [8], and have subsequently been further developed, beginning with computational methods [9]. There has also been significant.

Work in using APFs to obstacle avoidance [10], including dynamic barriers [11], which is particularly important to our study

Improvements to the behavior of artificial potential fields have also been made, particularly in dealing with undesired oscillation behavior [12]. Finally, the search for viable approaches for route planning using APFs has continued [13], including application to UAV path planning [14].

Control barrier functions (CBFs) were recently introduced [15], and are a mechanism for giving nonlinear system safety guarantees via optimization-based controllers.

Because of their resilience [16] and real-time performance capabilities, they are widely employed in the safety-critical controls sector, even for dynamic robotics [17].

The basic concept behind APF is to surround the robot with an artificial potential field in which obstacles are repelled by repulsive force and the robot is drawn toward the objective by attractive force.

The attracting and repulsive forces are responsible for the potential field. The goal generates an attractive force toward the robot, while barriers generate a repulsive force that is inversely proportional to the distance between the robot and the obstacles and is directed toward the obstacles. In the APF field, the robot moves from high to low potential. As illustrated in equations 1–5, the Potential function can be constructed for both attracting and repulsive forces:

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{1}$$

The most commonly utilized appealing potential is

$$U_{att}(q) = \frac{1}{2}\zeta\rho^2(q.q_{goal})$$

where ζ is gain, and (q, q_{goal}) is the distance between robot and the goal. (2)

As it approaches the target, the attractive force is expressed as the negative gradient of attractive potential, and the attractive force will be deemed zero.

$$F_{att(q)} = -\nabla U_{att}(q) = \zeta(q_{goal} - q) \tag{3}$$

The repulsive potential function can be used in the following way.

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(q.q_{obs})} - \frac{1}{\rho_o}\right)^2 \end{cases} \quad (4)$$

if

$\rho(q.q_{obs}) < \rho_o$

The repulsive potential function's negative gradient:

$$F_{rep}(q) = -\nabla U_{rep}(q)$$

$$= \begin{cases} \zeta\left(\frac{1}{\rho\left((q.q_{obs}) - \frac{1}{\rho_0}\right)}\right)\frac{1}{p(q)^2}\frac{q - q_{obs}}{p(q)} \end{cases} \quad (5)$$

To achieve the goal, a potential field method creates an appealing field. In most cases, the potential field is defined across the entire free space, and the induced force produced by the field is calculated at the robot position in each time step. The robot should then move in accordance with the forces mentioned above. The main issue with APF is that it might cause a local/global minima problem, in which the robot becomes stuck at a position where the effects of both fields/forces cancel out and the robot is unable to move forward or even backward. Figure 2 illustrates this.

1- 2.1.1. Dijkstra :

The simplest technique for identifying a path for a robot is to use the graph

searching method. It is regarded as a well-defined, effective, and efficient strategy for selecting a nonobstructive path that requires less time and computing complexity. The robot's environment is created, and the path is connected by a line so that the robot can easily approach the objective. The process is repeated until the best and most optimal solution is found from one node to the next. The robot is free to move to a new site after it has reached the targeted goal. The Dijkstra algorithm is a graph-searching method that finds the shortest path by solving the optimal path problem with non-negative edge path costs. It's used to calculate the cost of a path from a single point to a single destination [18].

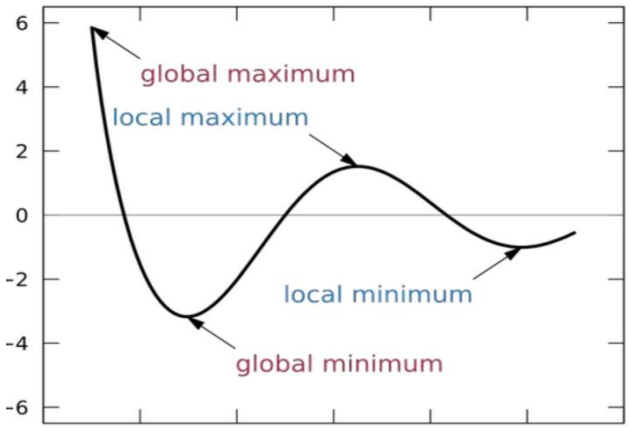


Figure 1.1: Local and global minima in a system.[19]

This technique is critical in traffic information systems where tracking the source and destination is possible.

It is used to find the shortest and least expensive path between the starting node and additional nodes in a graph. The algorithm's main aim is to compute the shortest distance from a starting point to an end point while excluding longer distance paths[20].

1- 2.1.2. C. A star

The A* algorithm is a path-finding algorithm in a graph between two given numbers. Because of its simplicity, it is frequently used as an example of a typical planning algorithm in the artificial intelligence sector[21].

The A* algorithm was created to ensure that the first answer found was one of the best, which is why it is popular in applications such as video games that prioritize calculation speed over accuracy. Peter E. Hart, Nils John Nilsson, and Bertram Raphael proposed this algorithm for the first time in 1968. It's a modification of Dijkstra's method from 1959[22].

Nils Nilsson, an artificial intelligence researcher, was attempting to improve the planning of Shakey the robot, a prototype robot that moves about a room with obstacles. The algorithm for finding a path, dubbed A1 by Nilsson, was a faster version of the most well-known method at the time, the Dijkstra algorithm for finding the shortest paths in a graph. Bertram Raphael suggested certain improvements, resulting in the revised A2 version. After then, Peter E. Hart made minor improvements to A2. Under specific situations, Hart, Nilsson, and Raphael have shown that A2 is the best option for finding the shortest paths[23].

1- 2.2. Artificial Neural network:

A computer model that models the operation of neurons in the human brain is known as an artificial neural network (AN)[24].

When fresh inputs are received, Artificial Neural Networks (ANNs) use learning algorithms that can autonomously make adjustments - or learn, in a sense. As a result, it's a powerful tool for modeling nonlinear statistical data.

Learning at a deeper level Machine learning (ML) and the broader field of artificial intelligence (AI) technology rely heavily on ANNs[25].

A Logistic Regression can be thought of as a single perceptron (or neuron). At each layer of an Artificial Neural Network, or ANN, there are many perceptrons/neurons. Because inputs are exclusively processed in the forward direction[26], an ANN is also known as a Feed-Forward Neural Network:

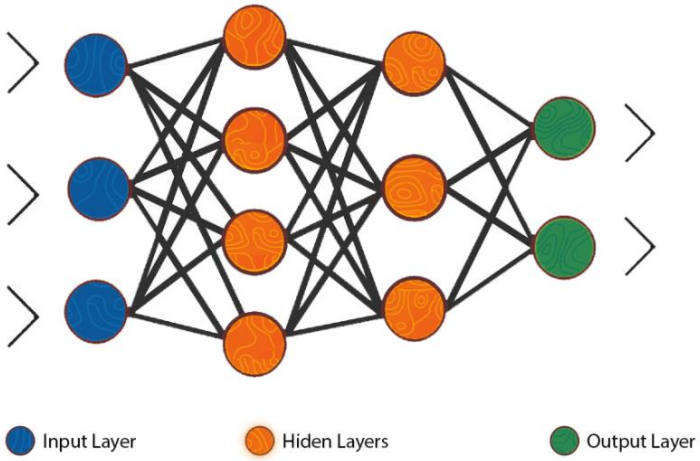


Figure 1.2: Artificial Neural network.[27]

As you can see, ANN is made up of three layers: input, hidden, and output. The input layer receives the data, the hidden layer processes it, and the output layer generates the output. Basically, each layer is attempting to learn specific weights[28].

1- 2.3 Fuzzy Logic:

With Lotfi Zadeh's proposal of fuzzy set theory in 1965, the term fuzzy logic was coined. [29] [30] However, fuzzy logic had been investigated as infinite-valued logic since the 1920s, especially by Ukasiewicz and Tarski. [31].

The phrase "fuzzy" refers to something that is unclear or ambiguous. In the actual world, we frequently encounter situations in which we are unable to tell whether a condition is true or untrue; their fuzzy logic provides extremely significant reasoning flexibility. In this approach, we can account for any situation's inaccuracies and uncertainties. 1and 0 represents the absolute truth value in the boolean system, while 0.0 represents the absolute false value. However, there is no logic for absolute truth and absolute false value in the fuzzy system. However, in fuzzy logic, an intermediate value exists that is both partially true and partially false.

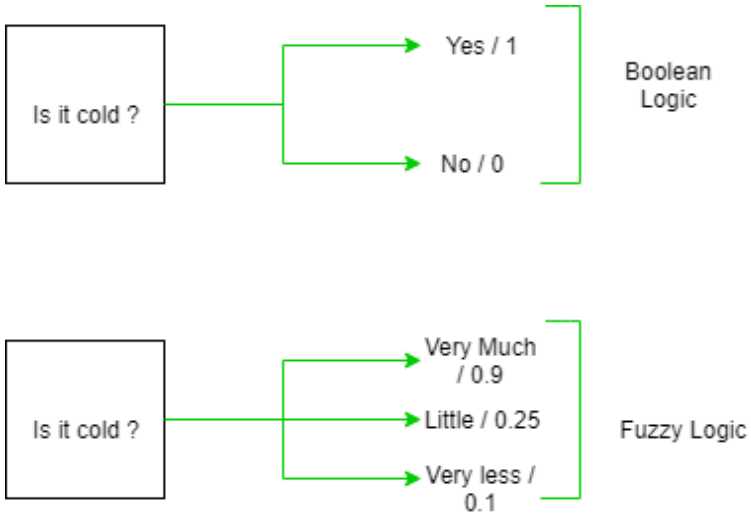


Figure 1.3: The difference between fuzzy logic and boolean logic [32]

The fuzzy logic controller's principle is divided into three stages:

A fuzzification stage, in which the input variables are transformed into fuzzy variables; A stage in which logical rules of the kind "if (condition 1) and/or (condition 2) then (action on the outputs)" are applied to a table of behavior rules; Finally, the defuzzification process converts the action decided by the rules of behavior into a command for the actuators to execute.

Fuzzy intervals are used in the fuzzification stage to divide the space of the input variables into a number of fuzzy subsets (for example, in proximity, we can have very close (contact), quite close, average distance, far enough, and very far); membership functions are then used to define the degree of truth (probability of belonging) of the fuzzy variable as a function of the input quantity.

These functions can be triangles, Gaussian functions, and so on. As a result, the membership rule will tell us that "there is a 95% likelihood that the obstacle is close enough, and a 5% chance of being in contact" for a particular distance measurement.

This viewpoint is based on the idea that sensor measurements and information in general are always subject to uncertainty.

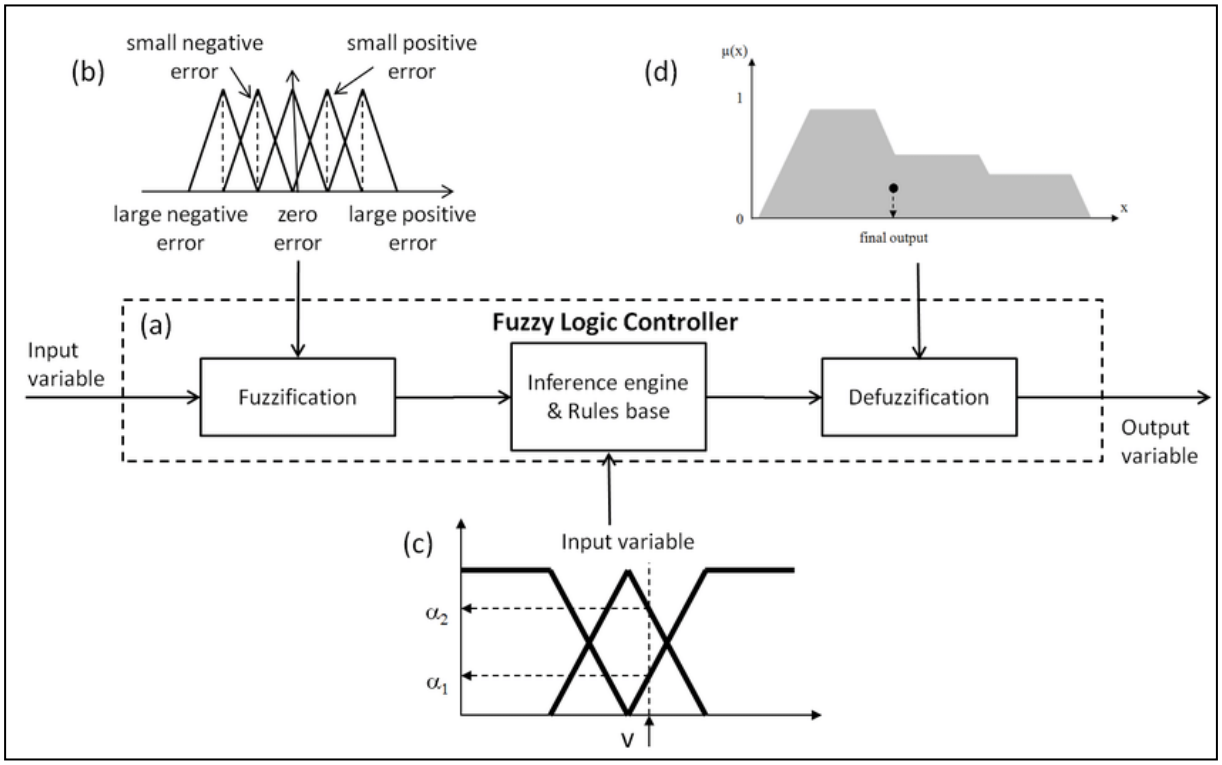


Figure 1.4: Membership functions.[33]

1- 2.4 Ant Colony:

In 1959 Pierre-Paul Grassi [34] invented the stigma theory to explain the nest-building behavior of termites, then Deneubourg and colleagues [35] studied the collective behavior of ants in 1983. The successful integration of the promethee multi-criteria decision-making method into the ACO algorithm (Humant algorithm) was recently in 2017 [36].

The ant colony algorithm is a path-finding algorithm based on the behavior of ants in their search for food. Initially, the ants roam aimlessly. When an ant discovers a food supply, it returns to the colony, leaving "markers" (pheromones) that indicate the trail has food. When other ants come across the markings, they are more likely than not to follow the course. If they succeed, they will mark the road with their own markings as they return with the food. The path becomes stronger as more ants discover it, until there are several streams of ants heading to diverse food sources near the colony. Shorter paths are more likely to be stronger since the ants drop pheromones every time they carry food, thereby maximizing the "solution".

Meanwhile, other ants are still looking for nearby food sources at random. When the food source is depleted, the pathway becomes devoid of pheromones and begins to disintegrate. The ant colony technique works effectively in graphs with shifting topologies because it is based on a very dynamic system, computer networks and workforce simulations using artificial intelligence are examples of such systems.

Ant colony algorithms aim to optimize a path for mobile robots. These algorithms require a group to be realized, they are used for a task that requires a collaborative work of several mobile robots. The aim of this research approach was to model and develop on a simulator, then a real robot, the way ants [37] move using complex combinations of information, as a vector of integration and visual cues. The goal was concretely to solve the problem of navigation. As a first step, developed a simulator to evaluate directly inspired navigation strategies ant behavior, the main interest of the simulation is to quickly allow an evaluation of biological hypotheses before moving to a physical implementation. Then, as the simulation does not allow taking into account all the An Evolutionary Approach To Searching For The Shortest Route Of A Self-Contained Mobile Robot aspects of the study, by simplifying for example the environment or the physical behavior of the robot, the researchers have developed a robot able to really test the strategies of displacement.

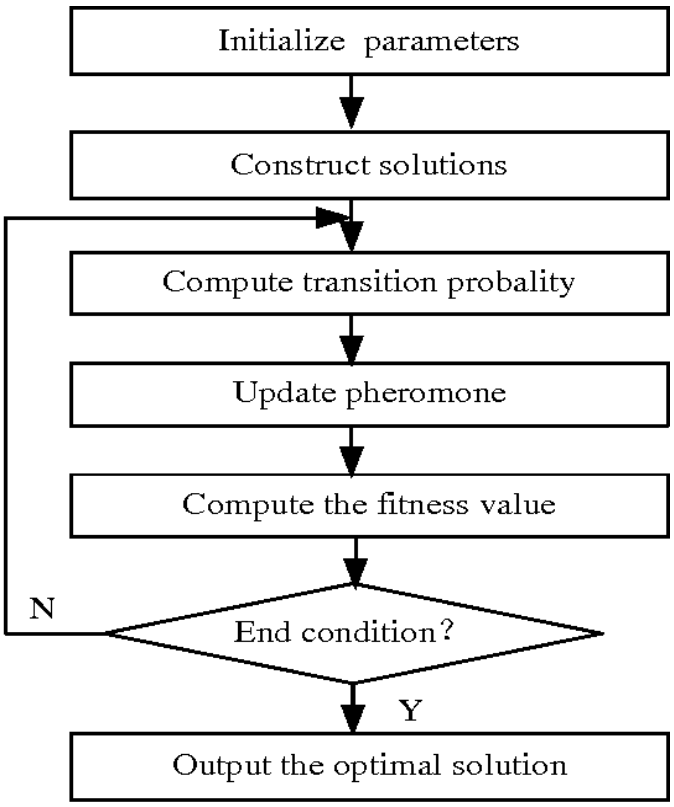


Figure 1.4: Ant Colony [38]

In an environment with obstacles, the robot must first detect and then bypass them. If the robot has already passed a given point, it will be to anticipate the direction to be taken without being disturbed by the large amount of information resulting from the presence of a large number of obstacles. If the robot ant encounters a very long obstacle, it should not be considered as impassable even if no issue appears on the rangefinder that we use to prevent the robot to hit the obstacles present in its environment. It must go along and around it by taking the direction provided by its integration vector being updated.

They developed first modeling based on the estimation of the vector of integration (direction of the nearest, distance) that the ants realize spontaneously from their first outings out of the nest:

- If the robot-ant encounters no obstacle, it will follow its integration vector pointing in a straight line on its starting point.

- If he encounters a close obstacle but occupying a weak angular sector (Figure 2), the robot-ant will follow the free direction "closest" to the integration vector allowing it to bypass the obstacle.

1- 2.5 Bee Colony:

The artificial bee colony algorithm (ABC) is a computer science and operations research optimization algorithm based on the intelligent foraging behavior of honey bee swarms, proposed by Derviş Karaboa (Erciyes University) in 2005 [39], by using only parts of nature like the behavior of bees we can develop a class of new algorithms, based on the behavior of bees while searching for food, we find in the following sections some of the algorithms (the famous).

1- 2.5.1 Bee Colony Optimization

The Bee Colony Optimization (BCO) metaheuristic has been well proposed recently by Lučić and Teodorović [40] as another direction in the field of swarm intelligence. Teodorović and his co-authors [41] effectively applied BCO to different design and execution problems. The BCO method is a "founding" method for dealing with the location of false experts who present unique types by their similarity with bees. Fake bee solvers that collaborate to solve complex combinatorial improvement problems.The bees gather in the hive and move back first. When all solutions are complete, the best one is selected and used to update the overall best solution, thus completing the BCO iteration.

All solutions are removed at this point and new iterations are generated. Let "B" be the number of bees in the hive and "NC" the number of positive steps taken. All the bees are in the hive when the search begins. The BCO algorithm pseudo code can be summarized as follows:

- Initialization: assign an empty solution to each bee;

- For each bee: in. k = 1

a) Count the constructive moves forward
b) Evaluate all possible steps;
c) Choose a step;
d) k = k + 1;

If k ≤ NC, Go to b . Return of all the bees to the hive;

- Each bee evaluates the value of the target work.
- Each bee chooses at random to carry out its own investigation.
- Recruiter, or become a collector bee. Choose for each supporter.
- Another arrangement for the selection of representatives.
- If planning is not complete, proceed to step (b).
- Evaluate each arrangement and find the best one.
- If the stop pattern is not selected, go to step (b), however, go to the next step.
- Display the best arrangement found.

1- 2.5.2 Dance Bee Optimization

Laga and Nouioua in 2009 [42] created the BOD calculation (Optimization of Dancing Bee) to solve the problem of T-shading graphs. This calculation is determined by the behavior of the bees when they forage. The calculation begins by randomly placing n bees in the hunting space. After evaluating the health highlights of these bees, the healthiest bees (Level 1 bees) are selected for community building. In the next step, the computation controls the query around the best destination m found by the first level bees. Of course, it is the bees that are enlisted to look around the best destinations, ie. This registration is an important activity for the calculation of the BOD. For each registered bee (permutation), we work with a neighborhood meta-heuristic to look at this permutation [43]. In the end, there, the best of the m bees (permutations) are then retained to build the next population.

Note that in nature there is no limit of comparison; such a limit is familiar in computer science. The number of survey responses. To complete the colony of bees, spawn the remaining bees at will. At the end of each cycle, the colony will include the description of the m bees per area from a point of view (to level the hunt) and again, the bees are distributed at will (to enlarge the hunt). Dance upwards, dust towards the sun. The dance is vertical and coordinated downwards, the dust opposing the sun. The rising of the sun is thus approached by the vertical, seen from bottom to top; and the point of the course of the crown jewels with that of the sun is recreated comparable to the azimuth.

1- 2.5.3 Artificial Bee Colony:

In 2007, Karaboga and Basturk [44] developed the ABC (Artificial Bee Colony) algorithm, which looked at how real bees find nectar and exchange that information with other bees in the nest. This algorithm divides artificial bees into three groups: The task of discovering new meals is carried out by bees (food-seeking bees), observers (watching bees), and scouts (scout bees) (the new nectar source). For each food source, only one kind of bee is used. In other words, the amount of food sources matches the number of worker bees. If a worker bee at a location cannot find the food source, she must operate as a Scout, exploring for new food sources at random.

Using bees to share information with visitors to the hive so that visitors can choose a food source to study. The following are some of the benefits of using a bee colony:

-Excellent at finding the best solutions.

-Solves the issue of the local optimum

-Easy to put into practice.

-The usage of a number of programmable parameters.

-Sensitive to challenges that are really challenging.

However, there are several disadvantages, such as most optimization algorithms having an evolution and diversification mechanism, incrementing a counter for solutions that do not improve and do not reach a threshold limit, and adjusting this parameter is a challenge in and of itself, and small values can eliminate a solution before exploring its neighborhood incomplete, while large values can trap the algorithm in minima premises for several cycles.

1- 2.5.4 Virtual Bee Algorithm:

Although only functions with two parameters have been given as examples, this approach was invented by XinShe Yang in 2005 [45] for solving numerical optimization issues.

It can optimize functions and discrete problems. The VBA method starts with a virtual bee troop, where each bee wanders randomly into the search space, which in most situations is only a 1-D or 2-space -D. The virtual functions for optimizing functions are the primary steps of the bee algorithm:

- The creation of a multi-agent or virtual bee population.
- Each bee has a solution vector with a number of parameters to optimize.
- Optimization functions (objective functions) coding and virtual food conversion (Virtual Food).
- Definition of a criterion for communicating direction and distance in a way that is comparable to the physical abilities of bees (the dance of the bees).
- Update a population of bee in new positions for virtual food study by performing a virtual dance to specify distance and direction; "the virtual dance of waggle."
- The maximum mode, in terms of the number of virtual bees or the intensity / frequency of the bees that make the visit, correlates to the best evaluation after a set period of evolution.
- Decoding the results in order to find the problem's solution.

1- 2.6. Genetic Algorithm:

Genetic Algorithms are part of the Evolutionary Algorithms family of algorithms [46]. These calculations are stochastic streamlining strategies enlivened by Darwin's hypothesis of development, whose objective is to get a surmised arrangement, at the right time.

Hereditary Algorithms use strategies got from hereditary qualities and common development: crosses, changes, determinations, and so forth ..., they address a stochastic improvement techniques for request "0", which implies neither congruity nor differentiability is fundamental for the smooth running of the strategy, just the information on the worth where the nearness of the capacity to be enhanced is adequate. Thus, the viability of a hereditary calculation relies upon the great information on the issue to be dealt with.

Hereditary calculations are the aftereffect of examination by John Holland and his partners and understudies at the University of Michigan on 1960 [47].

The novelty introduced by taking into account the crossover operator in addition to the mutations in this operator which allows us to get closer to the optimum of a function by combining the genes contained in the different individuals of the population.

Hereditary calculations depend on the thought of characteristic choice and apply it to a populace of solutions to a given issue. The phases of execution of a hereditary calculation:

- **Initial population:** We must choose a random population of n chromosomes, each chromosome here presents an upcoming location of the robot, and we can also change it so that each gene represents a next direction of the robot.
- **Fitness function:** Measure the fitness of each chromosome in the population.
- **Selection:** Create a new population with repetition next steps until the population is complete.
- **Crossover and mutation:** Each pair generates two children, in these operation two chromosomes exchanging one or more parts for data of the new chromosomes. If there is no mixing, the result is an exact copy of the parents.
- **Mutation** means that a gene in a chromosome can substitute for another in a random way.
- **Stop test**: If this criterion is not checked then go to step (2).

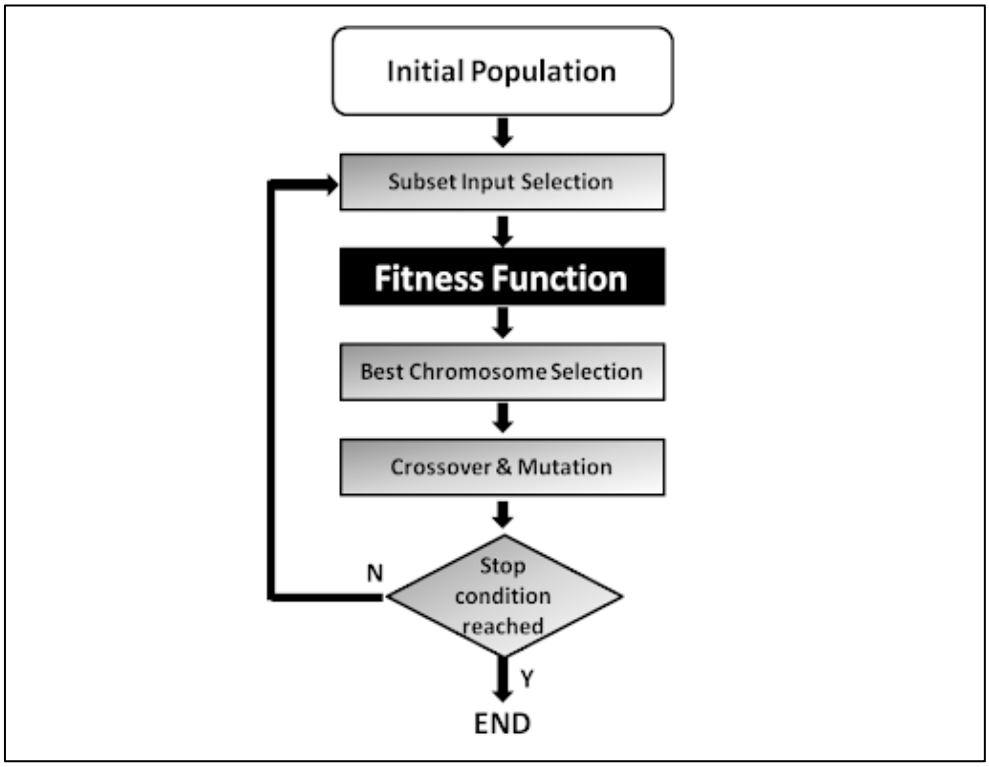


Figure 1.5: Genetic Algorithm [48]

1- 2.7. Bacterial foraging optimization :

Bacterial foraging optimization (BFO) [49], a novel algorithm proposed by Passino, mainly simulates the behaviors of Escherichia coli in the process of searching for nutrients. In chemotaxis, reproduction, elimination dispersal. Besides that, as for function optimization,

During the foraging context, bacteria generally take four significant actions involved the position of one bacterium can be regarded as a feasible solution in the search region. Bacteria adjust their own positions by tumbling based on random directions and swimming with certain step sizes to constantly find out the optimal location. Because of many advantages like the strong robustness and good performance in local search, the BFO algorithm has gradually become popular and until now, it has been applied to sorts of practical fields such as facility layout [50], feature selection [51], training kernel extreme learning machine [52]. At present, relevant theoretical researches about the BFO algorithm are still in the initial stage. In other words, compared with the other traditional swarm intelligent algorithms such as particle swarm optimization (PSO) [53] and genetic algorithm (GA) [54], the development of the BFO algorithm is not enough mature.

Professor Passino created the Bacterial Foraging Optimization (BFO) method in 2002 [41], which is a revolutionary swarm intelligence optimization technique based on E. Coli foraging behavior. Even though it is still in the early stages of study, the BFO algorithm outperforms other optimization algorithms in terms of fast convergence and global search due to its simple bacterial individual structure and behavior, diverse group kinds and characteristics, and efficient life cycle [55]. A bacteria can do one of two things when foraging: tumbling or swimming [56].

The bacterium's orientation is changed by the tumbling movement. The bacterium will migrate in its current direction while swimming, which is the chemotaxis step.

Chemotaxis is continued until a bacterium moves in the positive-nutrient gradient direction. After a given number of complete swims, the best half of the population reproduces and the rest of the population is eliminated.

To avoid local optima, an elimination dispersion event is performed, in which certain bacteria are liquidated at random with a very small probability, and new replacements are initialized at random positions throughout the search space. BFO is made up of three main components:

The chemotaxis process is the first; the reproduction step is the second; and the elimination-dispersal step is the third.

a. Chemotaxis :

This phase is critical in the explanation of bacterial movement, as there are two alternative paths. This bacterium can swim in a straight line without tumbling or swimming in a variety of directions. However, it's plausible that this bacterium can switch between the two types of locomotion. As a result, the bacteria is considered to be swimming if it goes in a specific direction and tumbling if it moves in an unexpected direction.

Chemotaxis is a biological process in which bacteria travel in order to obtain a nutrient. One of the characteristics of the E. coli bacterium is that it can move in two different ways: swimming and tumbling. When swimming, the bacterium swims in the same direction in quest of nutrients, whereas when tumbling, the bacterium changes direction. Assume Pi(j, k , I) shows the present position of I^{TH} bacterium, j^{TH} chemotaxis step, k^{TH} reproduction step, and elimination and dispersal step, then tumbling will show the bacterium's position in the next chemotaxis step as:

$$\theta_i^{j+1} = \theta_i^j + C(i).\emptyset(i) \qquad (6)$$

$$\emptyset(i) = \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \qquad (7)$$

where C (I) denotes the number of steps, ϕ(i) denotes the bacterial chemotaxis direction denoted by the tumbling of the ith bacterium, and Δ denotes a vector of random population size directions with continuous values between [-1, 1].

$$J_{he}^i = \sum_{j=1}^{N_c+1} J(i,j,k,l) \qquad (8)$$

b. Reproduction :

Those bacteria that have enough nutrients will reproduce, while others will be destroyed during the reproduction phase. To accomplish so, each bacterium's health condition is calculated. The sum of a person's step fitness across their lifetime can be defined as:

$$J_{he}^i = \sum_{j=1}^{N_c+1} J(i,j,k,l) \qquad (9)$$

Where, N_c is total number of chemotaxis steps, and J_{health} is calculated for i^{th} bacterium , j^{th} chemotaxis step , the K^{th} reproduction step, and the l^{th} elimination and dispersal phase are all steps in the chemotaxis process.

Eventually, all bacteria's health status will be sorted in ascending order, and the first half of the bacteria will multiply and surrogate into the second half according on their top health status. In other words, lower J_{health} levels corresponded to bacteria that were healthier. As a result, bacteria with lower health ratings have a better chance of surviving. This procedure not only maintains the population stable, but it also ensures that the healthier bacteria are passed down to the next generation.

$$J_{i,\text{health}} = \sum_{j=1}^{Nc} J_i(j,k,l) \tag{10}$$

b. Elimination and Dispersal :

Each bacterium is scattered with the chance of P_{ed} after reproduction, but the total number of bacteria remains the same. Once a bacterium has been eradicated, it will be distributed at random to a new place.

$$r = \text{random}[0,1];$$

$$P_i(j,k,l) = \begin{cases} P_i(j,k,l), & r > P_{ed}, \\ \left(m'_1, m'_2, \ldots, m'_p\right), & r < P_{ed}. \end{cases} \tag{11}$$

Elimination happens when ri < P_{ed} is present, as demonstrated in (11). The i^{th} bacteria Pi's original position was substituted by p=(m'1,m'2,....,md'). As a result, the optimal parameter m is replaced by a random value m′ in the optimization space.

1- 3-Comparison between Methods

Method	Advantages	Disadvantages
Fuzzy logic	• Create a set of rules based on human experience. • The most human-like reasoning. • It is light in terms of computation, time savings, and memory space.	• Because of the approximation reasoning process, the navigation path is not optimal. • Requires the presence of an expert. • These rules restrict robot operation.
Neural networks	• Create a straight forward solution • The knowledge representation method	• The ability to calculate a precise travel path
Ant colony	• Extremely adaptable. • Ideal for graph-based issues.	• It's possible to get stuck in a blocking state. • Execution time can be lengthy at times. • It isn't applicable to all types of issues.
Bees Colony	• Effective in identifying the best solutions and simple to execute • Eliminate the problem of the local optimal • Responsive to difficult issues	• It has an evolution and diversification mechanism • Large values can cage the algorithm for numerous cycles and destroy a solution before it can be exploited.
Genetic Algorithm	• AGs work with parameter encoding • AGs work with a population of points • The application of probabilistic transition rules to prevent local optimums. • Aerodynamic form, structure, and composite material synthesis	• The issue of industrial network scheduling • Form recognition and learning through reduction • Bioinformatics pattern identification • Electronic circuit synthesis
Bacterial foraging optimization	• powerful parallel search capability, and it's simple to escape the local minimum	• The bacteria's poor connection, which puts them at risk of reaching the local optimum rather than the global optimum.

Table1.1: Comparison of the Methods.

1- 4. Conclusion

In this chapter, we reviewed the state of the art in mobile robot navigation and the many algorithms that may be used to manage it, such as finding the shortest paths and distances in a given area. We've also seen the various steps; we've also provided the benefits and drawbacks of each strategy, and we've discovered that no navigation methodology has been observed that provides a solution that solves all difficulties. As a result, we can see that navigation is a very dynamic subject of study, with new approaches appearing on a regular basis. The necessity of applying heuristic approaches to treat the motion computation problem as an optimization problem in order to meet several constraints at the same time is revealed by this overview of the literature.

In the next chapter, we will be viewing how our system is designed, a detailed explanation of our chosen methods along with the Conceptual models that describe the system flow, making it more effortless to comprehend.

Chapter 2

System Design

Chapter 2: System Design

2.1. Introduction:

In the previous chapter, we talked about several pre-existing algorithms that solve this problem. Robots are generally machines that include computers; nevertheless, the robot controller is unique to each robot and hence cannot be transferred to another. However, there is a need for a generic model of a robot controller in both industry and research.

In terms of research, the development of applications that require the robot to share its environment with its environment leads to the search for scalable and reliable architectures that are far removed from the fixed architectures of traditional robots whose immediate environment is closed to the human operator.

AI (bacterial algorithm) methodologies for the definition and design of real-time systems should aid in the modeling of the complex system that is a robot controller in this context.

In recent years, a large amount of study has been focused on automatically planning the trajectory of collision-free robots in environments.

Several planning strategies have been proposed over the last two decades, but only a few of them help handle the planning problem satisfactorily.

Our trajectory planning method provides a safe and effective solution to the fundamental problem of planning, as well as the optimization challenge. The path in our work is determined by the bacterial algorithm's modeling of the robot's environment.

In this chapter, we will provide several step-by-step diagrams to see how our proposed solution worked

2. Study of Project

2.1. Our Problem:

Our challenge is to identify the shortest path for an autonomous mobile robot and to find the most cost-effective approaches to reach the desired end state. We examine how bacterial algorithms function to find optimal solutions to large-scale planning issues and avoid impediments.The goal of our research is to find an answer to the following research question:

How can the robots go from their beginning position to their stopping location in the most efficient way possible while avoiding obstacles?

2.3. The proposed solution:

The research focuses on adapting the existing bfo and building a new competency to deal with the challenges of large-scale optimization of tasks applied to mobile robots to address the topic of searching for the shortest path.

2.3.1. Problem definition

We will concentrate on the following scenario: In the initial state, the environment is represented by a square matrix with obstacles arranged at random. A robot must go from a start point to an arrival point; the robot has a start location (Xs, Ys) and an arrival position (Xa, Ya), but it is unaware of its surroundings. The following difficulties piqued our interest: determining the best trajectory (shortest path) and avoiding obstacles.

2.3.2. The Activity Diagram:

An activity diagram is a visual representation of a system or a process. It displays the decisions, activities, and supporting processes that go into a more significant activity. Such a diagram is frequently used to organize, implement, and optimize processes in the management and information technology fields.

At first glance, an activity diagram and a flow diagram can look very similar. The difference is in the application and perspective. A flow diagram shows people (and robots) what they need to do at each stage of activity by illustrating the flow of objects. It is about describing the tools needed to do a job.

An activity diagram, however, gives you a high-level view of the processes and their supporting processes. It is an instrument from which system designers and managers can get the most benefit. He or she is frequently used by project managers, team leaders, business users, etc.

Users can benefit from activity diagrams in several ways. Take into account drawing an activity diagram to:

- Display the reasoning behind an algorithm.
- Explain the actions taken in a UML use case.
- A business process or workflow between users and the system should be demonstrated.
- Clarify difficult use cases to streamline and enhance any workflow.

- Model software architecture components like usage, function, and methods an activity diagram in UML depicts the behavior of a system by describing the process's sequence of actions.

Figure. 2.1 A flowchart showing the dynamic features of the system. The graph starts with the user, who is responsible for initiating the process by configuring the starting and target points, the obstacles, determining the number of bacteria, death points, and the size of the matrix.

Bacteria search for solutions surrounding them, and this process continues until reaching the target, if one of them falls into the poisonous points, it dies automatically, when the first bacteria reach the end site, it draws a path that is considered the shortest path. From here, the rest of the bacteria passing on that path move inside it, so that the rest of the place becomes obstacles for them, this process continues until the specific life of the bacteria ends, and upon completion, the optimal path is drawn in the end, without forgetting that it multiplies when it reaches the feeding points.

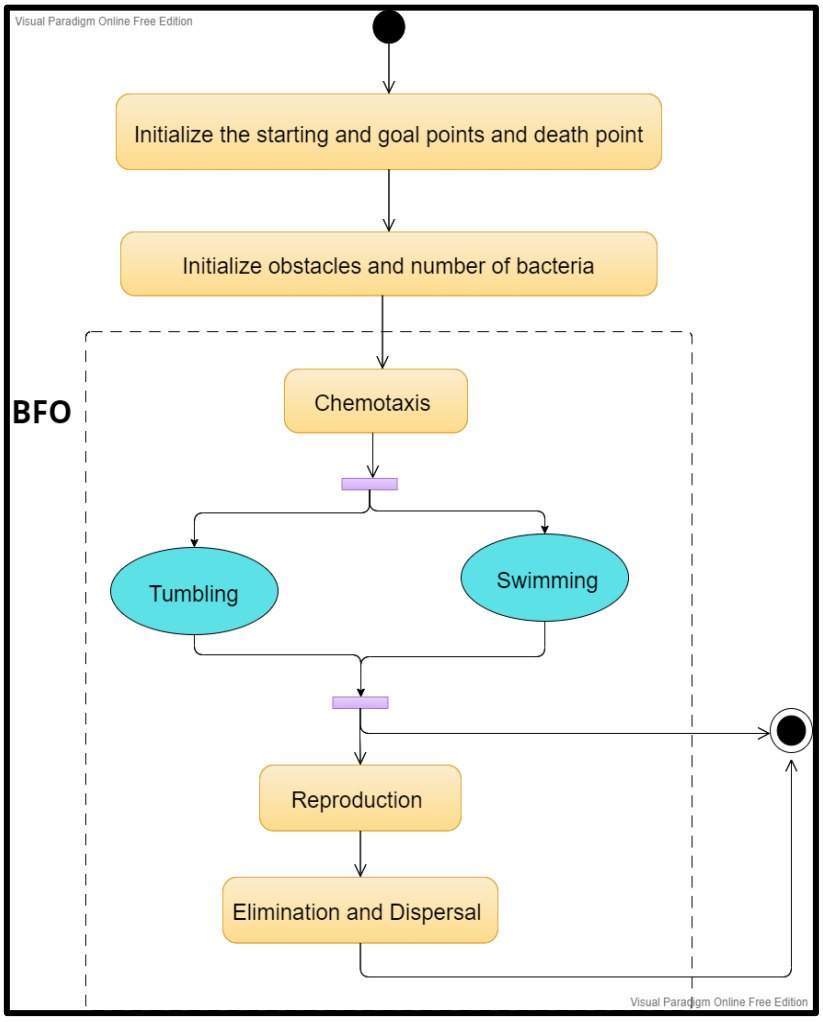


Figure 2.1: Activity Diagram

2.3.3. Sequence diagram:

As they concentrate more intently on lifelines, processes, and objects that exist concurrently, as well as the messages they exchange with one another to execute a function before the end of the lifeline, sequence diagrams are a common dynamic modeling approach in UML. Use this manual to learn everything there is to know about UML sequence diagrams, together with our UML diagramming tool. For businesses and other organizations, sequence diagrams can be a useful resource. Consider creating a flowchart to:

- Represent a UML use case's specifics.
- Model a complicated procedure, function, or operation's logical flow
- Observe how elements and things interact to carry out a process.
- Map out and comprehend the precise workings of a present or hypothetical situation.

The sequence diagrams in Figure 2.2 model communication between these objects.

The robot obtains the directions from the direction computation class using the transmit arrangements strategy, and then sends them using the choice technique at each position update. (At a pre-determined repetition of the new arrangements for testing).

It is possible to derive a worked-out class outline from the collaboration illustrated in these two graphs.

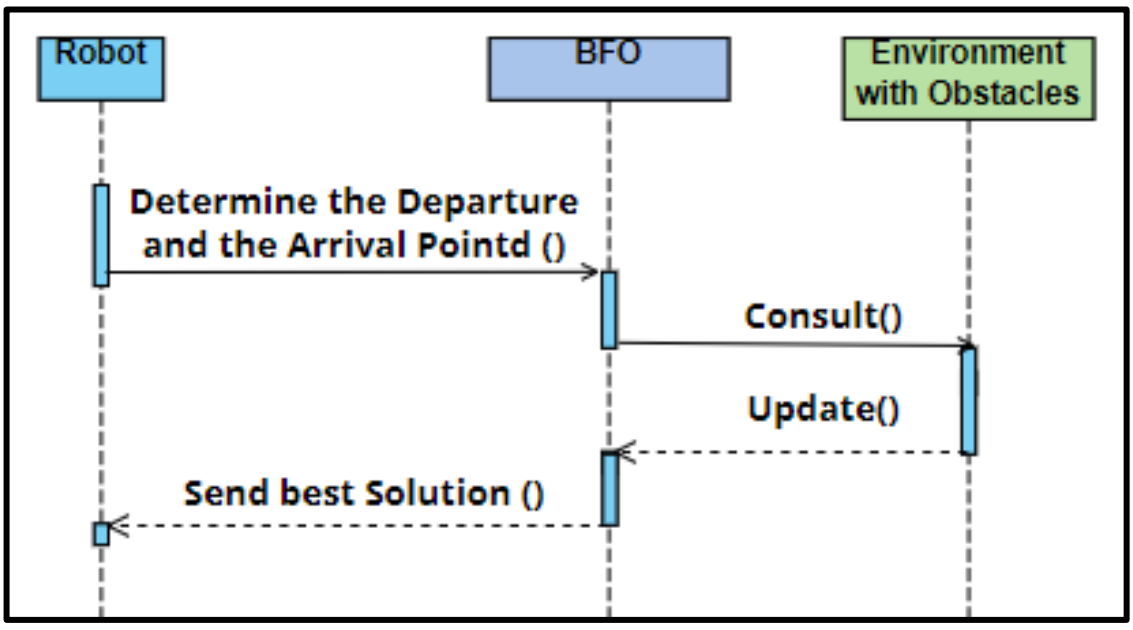


Figure 2.2: Sequence diagram

2.3.4. The Class Diagram:

Because they specify what must be present in the system being modeled, class diagrams are a type of structure diagram. UML was created as a standardized model to depict an object-oriented programming style. Class diagrams are the foundation of UML since classes are the building block of objects. The many components of a class diagram can be used to describe the major objects, the classes that will be programmed, or the relationships between them. The advantages of UML class diagrams include:

- Showcase data models for information systems, regardless of how simple or complicated they are.
- Have a better understanding of the application's schematics' overall overview.
- Visually convey any distinctive system requirements and spread that knowledge.

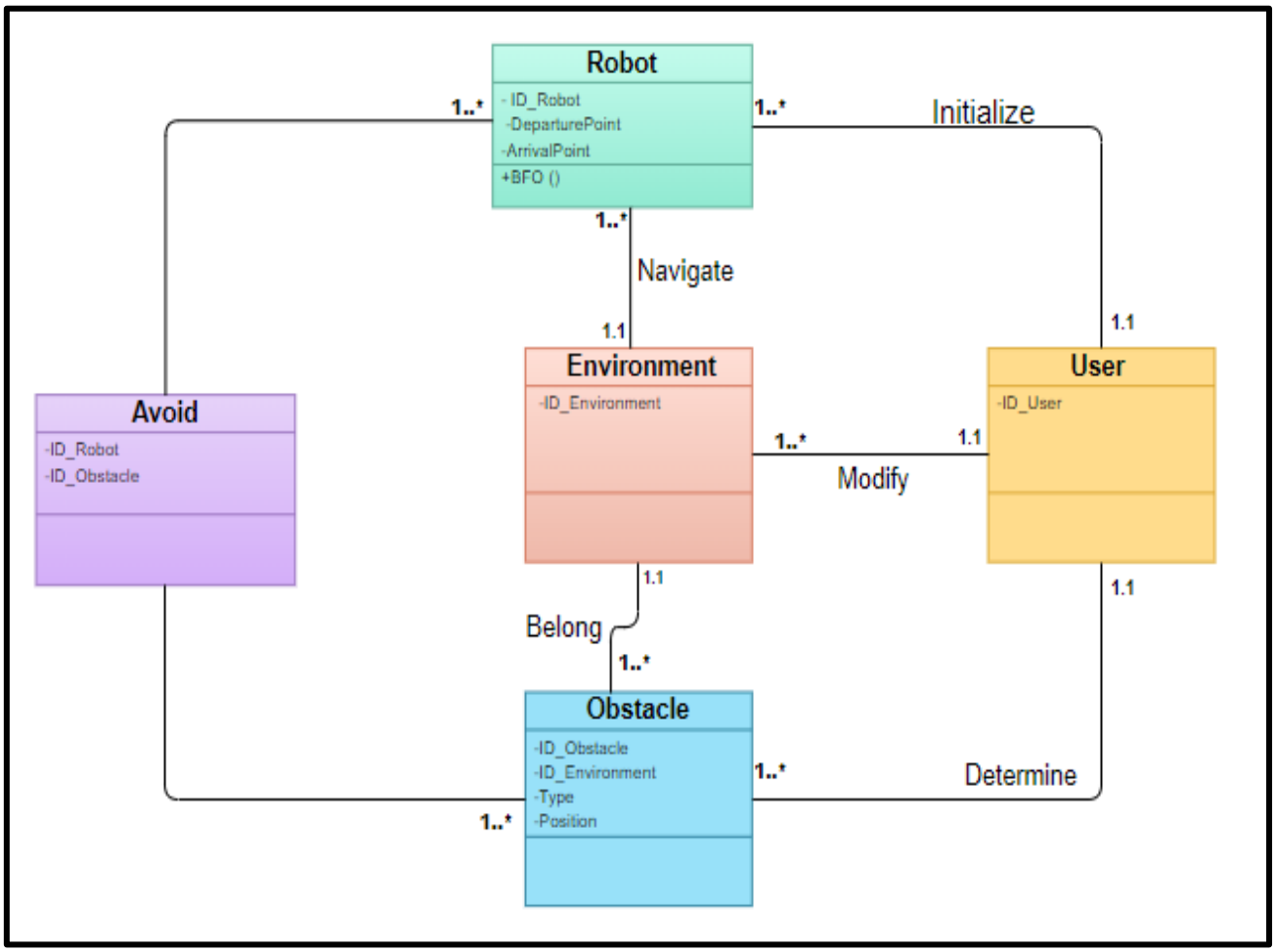


Figure 2.3: Class Diagram

Figure 2.3 in class diagrams, which are used to precisely translate models into computer code and to conceptually depict the static view of an application. The arrangement of classes, properties, methods or processes, and associations between objects is depicted in this graph precisely. The classes of our system are:

- Robot class: possess the qualities of robot ID, departure point, and arrival point. It controls the order of trajectory creation and commands to prevent errors and confusion, such as malfunctioning and frustrating halts, and regulates all bacterial optimization algorithm operations.
- Environment Class: possess as an attribute the environment ID. It is created and altered by the user so that the robot may move around.
- Obstacles Class : contains the obstacle and environment ID as attributes and oversees all operations of the obstacles including positions and type whether they are fixed or moving
- User class : hold the user ID as an attribute Handles all operations done by the user
- Avoid class: includes attributes for the robot and obstacles ID. Given that the robot can avoid one or more obstacles at once, avoided was initially thought to represent a relationship between the robot and obstacle classes. And one or more robots can work together to escape an impediment at once.

2.3.5. Use Case Diagram:

A use case diagram in Unified Modeling Language (UML) can be used to condense details about the users of your system (also known as actors) and their interactions with it. A particular collection of symbols and connectors are needed to create this kind of UML diagram. Use case diagrams, when created effectively, can facilitate teamwork and show:

- Interactions between your system or application and external systems, organizations, or individuals.
- The objectives that your system or program enables actors (or entities) to accomplish.
- The system's coverage area.

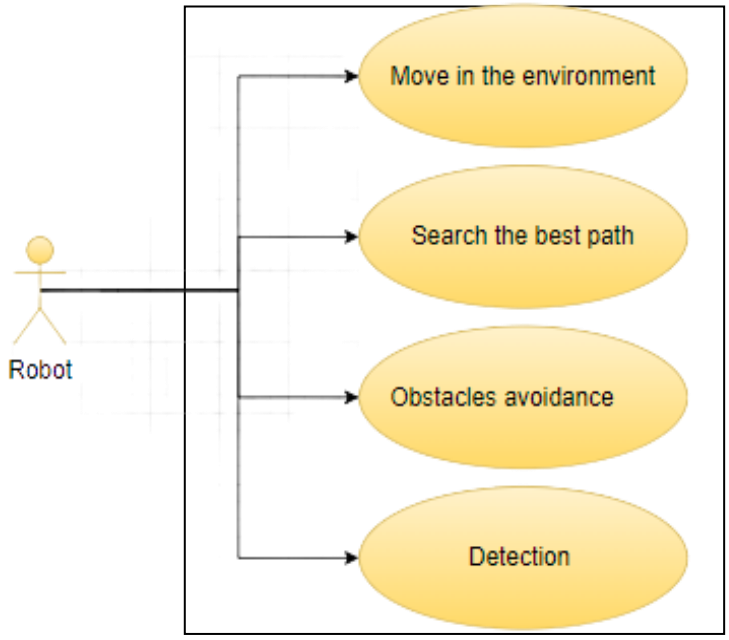


Figure 2.4: Robot use case diagram

The robots have four main functions: search, movement, obstacle avoidance, and detection:

- Research is the process of determining objectives.
- The movement consists of activities that change positions in response to controller commands.
- Obstacle avoidance is also a method of achieving objectives
- Detection is also the process of searching for targets

We have a single entity in our diagram that represents our robot as the main player. The latter has four central points. operations to carry out are:

- The search for the best path: our robots are placed at the beginning of an environment with an endpoint to achieve; it is then up to them to determine the best and shortest way.. It includes if movement and detection
- Move around in the surroundings, bearing the ending point in mind. The robots begin to navigate around the area in the recording.
- Obstacle avoidance: Because obstacles are randomly put in the environment based on the start and end points entered, the robots must avoid them when they cross their path.

There are 7 main types of operations available to the user:

- Determine the starting and ending points.

- Determine Death points.
- Determine The Interval of Chemotaxis.
- Determine Obstacle Percent.
- Determine Bacterial Number.
- Determine Max-age, and Max Arrived Bacteria.

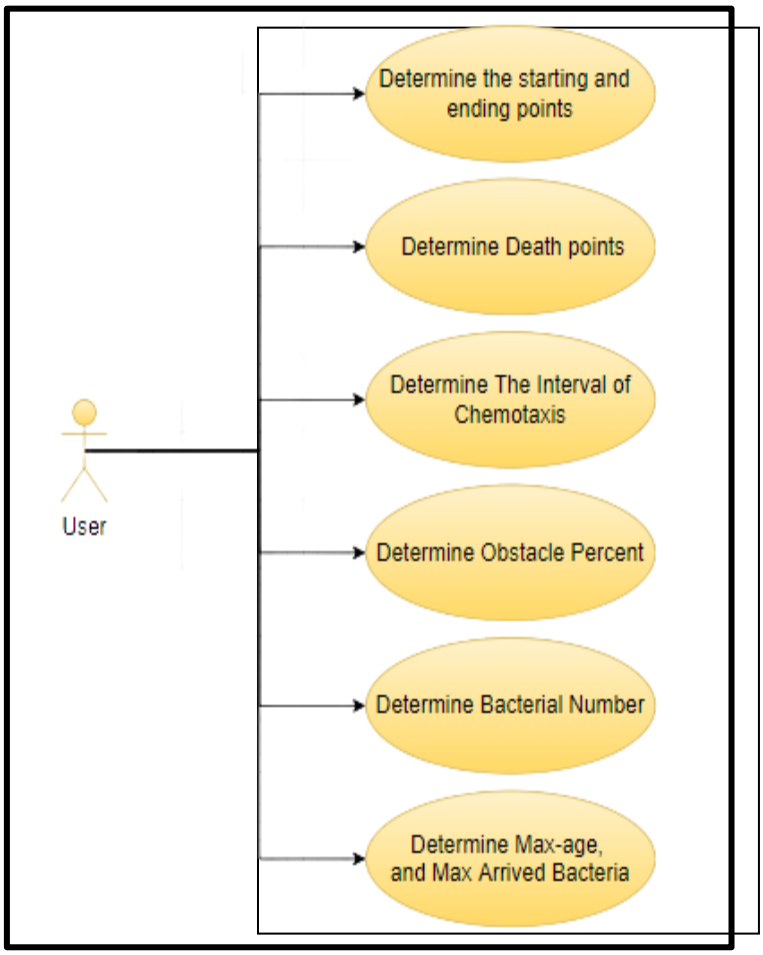


Figure 2.5: User use case diagram

The user is the only actor who has performed a single operation in this use case diagram. When you run the program, a window appears asking you to enter the information it needs.

Conclusion:

We discuss our bacterial method for guiding a mobile, autonomous robot in this chapter. It was decided to create a robustness criterion. Finding a solution to good performance that is sensitive to uncertainty is the goal of this criterion. The goal is to identify the quickest way to go to the destination.

Unknown environments with dynamic obstacles in the path of the mobile robot force the path planning algorithm to solve extremely difficult computational problems that necessitate high-performance computing to calculate new paths online and then calculate the trajectory while meeting the requirements for small-time local controllability. We presented our method to autonomous navigation for a mobile robot in an obstacle-filled environment in this work. To find the shortest path, this method employs a BFO Algorithm. In a dynamically shifting obstacle environment, the robot may move from its initial position to its final position without colliding. The robot is able to reach the destination by avoiding the obstacles in its path, according to simulation data. Aside from navigating mobile robots in complicated situations.

In the next chapter, we will present the obtained results compared to other algorithms in addition to a more detailed explanation of our solution

Chapter 3 :

Realization And

Implementation

Chapter 3: Realization and Implementation

3.1. Introduction:

In the previous chapter, we saw several diagrams to see how the proposed solution works , To demonstrate autonomous control of the mobile robot, we used a Matab application that is an intelligent system with numerous capabilities for simulating the robot's behavior in obstacle avoidance to achieve the destination and evaluating the performance of the strategies used.

The complexity of optimization challenges is increasing, and the rapid advancement of technology has made it more difficult to solve them.

The employment of an evolutionary strategy is becoming more and more important.

Furthermore, the cost-to-performance ratio in parallel IT systems is decreasing. The Meta-heuristics are designed and implemented using an evolutionary strategy to speed up research. To increase the quality of the answers obtained, as well as their robustness and the ability to solve issues on a broad scale .

Our study in this thesis is focused on the topic of determining the shortest path for an autonomous mobility robot. Based on a combination of bacterial algorithms, we have proposed a new evolutionary technique for solving the problem of determining the shortest path for an autonomous mobile robot.

In this chapter we will see realization and implementation

3.2. Tools and working environments:

3.2.1. Software environment:

MATLAB is the most generally utilized open source programming language for IT experts. This language has impelled itself to the highest point of foundation the board, information investigation or programming improvement. In reality, among its characteristics, Matlab outstandingly permits engineers to zero in on what they do instead of in transit they do it [57].

Matlab offers a few valuable highlights for amateurs and specialists the same, from the outset, it is not difficult to learn and use. Its highlights are not many, which permits you to make programs rapidly and with little exertion. What's more, its grammar is intended to be clear and direct. Another benefit of Matlab is its prevalence.

This language chips away at all major working frameworks and PC stages. Furthermore, regardless of whether it is plainly not the quickest language, it makes up for its gradualness with its adaptability.

Chapter 3: Realization and Implementation

At last, regardless of whether it is mostly utilized for prearranging and robotization, this language is likewise used to make proficient quality programming. Regardless of whether it's applications or web administrations, Matlab is utilized by countless engineers to make programming. [58]

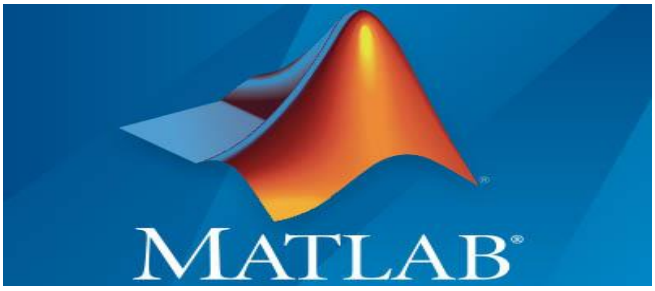


Figure 3.1: Matlab logo [59]

3.3. Description of the different modules

3.3.1. The environment:

A computer representation of the navigation space is required in some way.

Occupational grids and graphs are the two most common types of maps. The environment in the first scenario is a pixel matrix (color pictures : red, green, blue).

The black boxes that represent barriers have the value (0,0,0) and the white boxes that indicate free boxes have the value (1,1,1) and (0.75,0.75,0.75) chemical zone , point of death (1,0,1) and (0,255,0) Start point and (255,0,0) end point , in the initial state of the matrix representing the environment with obstacles placed randomly (Figure 3.2).

1,1,1	1,1,1	1,1,1	0,0,0	1,1,1	1,1,1	0,0,0	0,0,0	1,1,1	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	1,1,1
1,1,1	1,1,1	1,1,1	0,255,0(S)	1,1,1	1,1,1	0,0,0	0,0,0	0,0,0	0,0,0
1,1,1	0,0,0	1,1,1	0,0,0	1,1,1	1,1,1	1,1,1	0,0,0	0,0,0	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	1,1,1	0,0,0	1,1,1	0,0,0	0,0,0	0,0,0
1,1,1	1,0,1	0,0,0	0,0,0	1,1,1	1,1,1	0.75,0.75,0.75	1,1,1	1,1,1	1,1,1
1,1,1	1,0,1	1,1,1	1,1,1	1,1,1	0.75,0.75,0.75	255,0,0 (E)	0.75,0.75,0.75	0,0,0	1,1,1
1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	0.75,0.75,0.75	0,0,0	1,1,1	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	1,1,1	1,1,1	0,0,0	1,1,1	1,1,1	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	0,0,0	1,1,1	0,0,0	1,1,1		1,1,1

Chapter 3: Realization and Implementation

1,1,1	**1,1,1**	**1,1,1**	**0,0,0**	**1,1,1**	**1,1,1**	0,0,0	0,0,0	**1,1,1**	**1,1,1**
1,1,1	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	0,0,0	1,1,1
1,1,1	1,1,1	1,1,1	0,255,0	1,1,1	1,1,1	0,0,0	0,0,0	0,0,0	0,0,0
1,1,1	0,0,0	1,1,1	0,0,0	1,1,1	1,1,1	1,1,1	0,0,0	0,0,0	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	1,1,1	0,0,0	1,1,1	0,0,0	0,0,0	0,0,0
1,1,1	1,0,1	0,0,0	0,0,0	1,1,1	1,1,1	0.75,0.75,0.75	1,1,1	1,1,1	1,1,1
1,1,1	1,0,1	1,1,1	1,1,1	1,1,1	0.75,0.75,0.75	255,0,0	0.75,0.75,0.75	0,0,0	1,1,1
1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	1,1,1	0.75,0.75,0.75	0,0,0	1,1,1	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	1,1,1	1,1,1	0,0,0	1,1,1	1,1,1	1,1,1
1,1,1	0,0,0	0,0,0	0,0,0	0,0,0	1,1,1	0,0,0	1,1,1		1,1,1

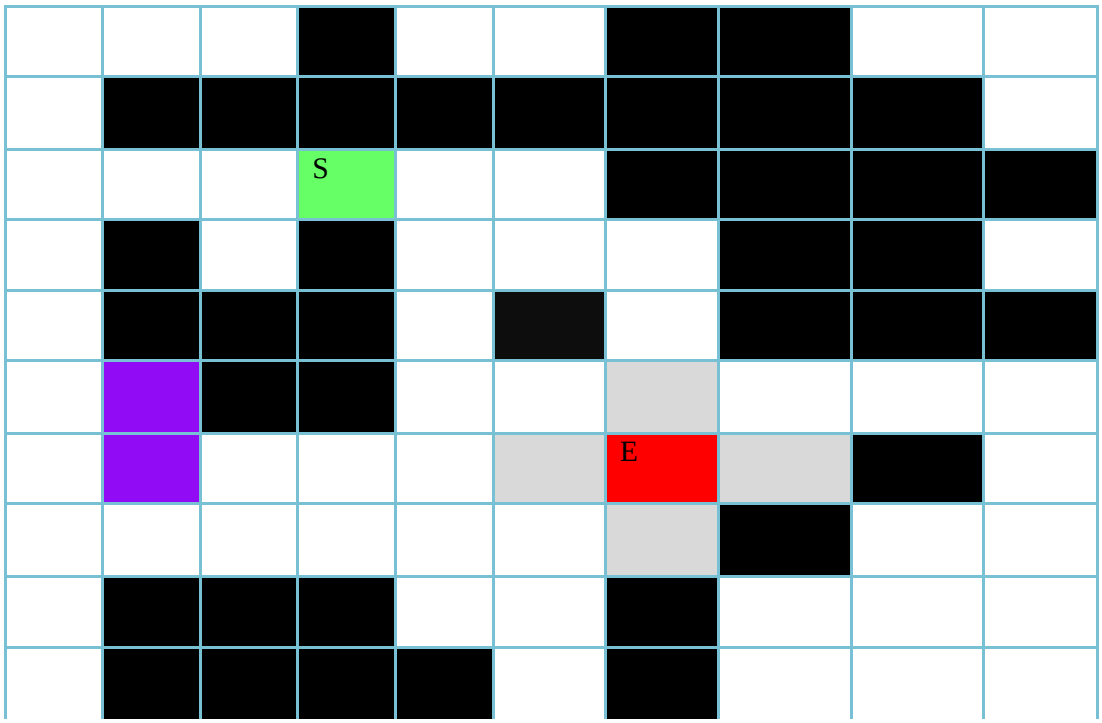


Figure 3.2: The passage from a RGB matrix to a navigation environment.

3.3.2 Initialization of matrix:

A square matrix is presented with a starting point and ending position, as well as random obstacles. We have a button that allows us to enter the starting location, the destination, the map size, the death points, the time interval of chemotaxis, the percentage of obstacles, the number of bacteria, specifying the maximum age and the bacteria that reached the maximum. In our paper, we suggested a 50 * 50 matrix.

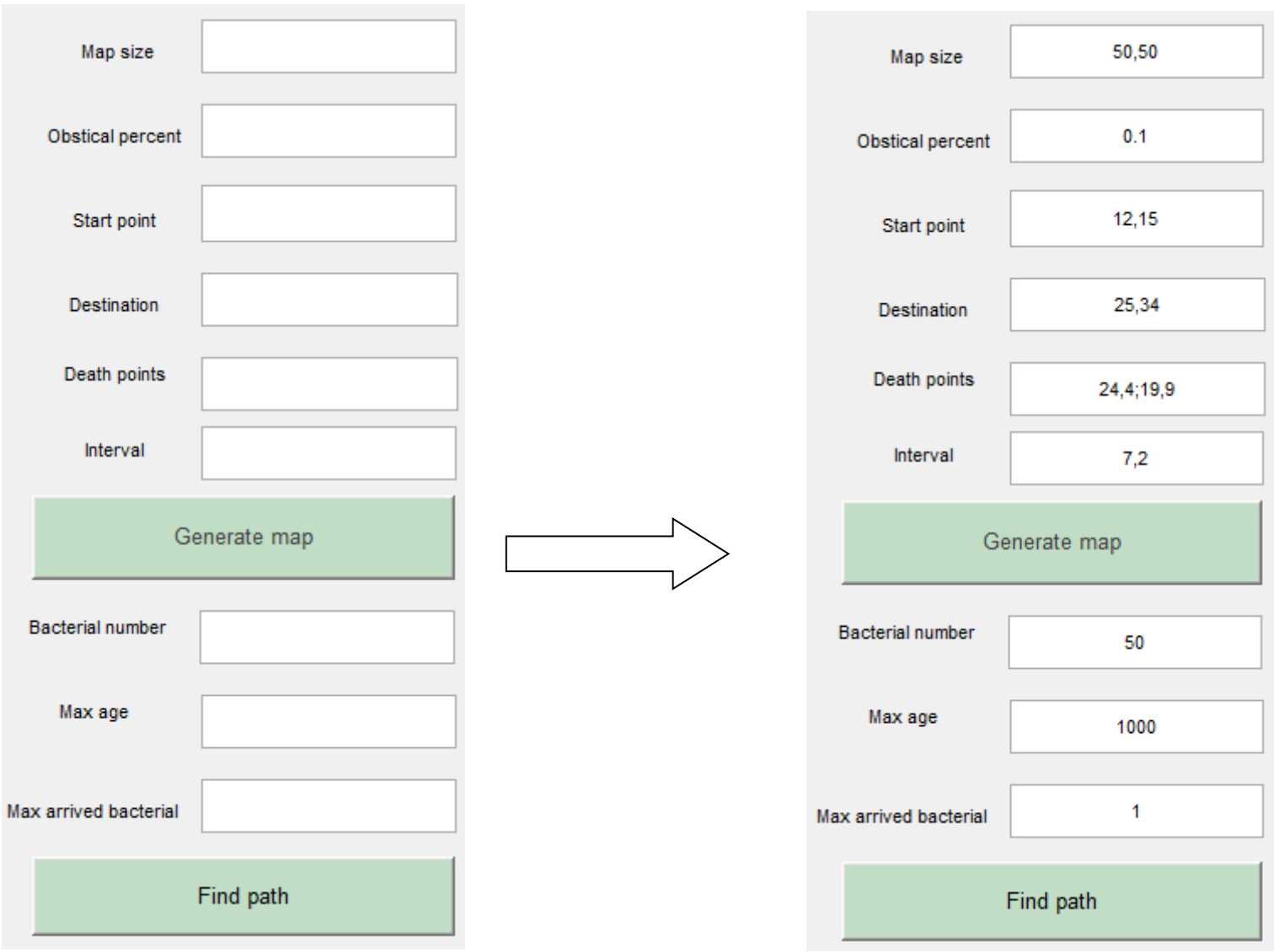


Figure 3.3: Interface of Our Application

Chapter 3: Realization and Implementation

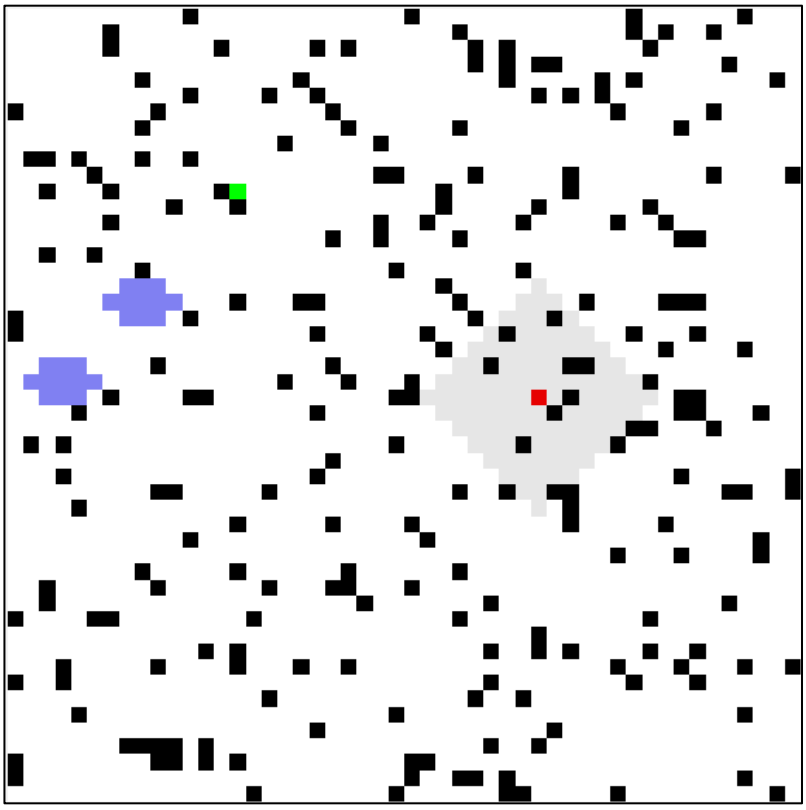


Figure 3.4: final matrix

The green point is the starting point, and the red point is the end point , The gray color surrounding the endpoint is the chemotaxis zone, As for the blue points, they are death points.

We color the pixels using RGB colors using the following mask:

- The starting point :

```
mask(start(1),start(2),[1,3])=0;
```

- Final point:

```
mask(dest(1),dest(2),[2,3])=0;
```

- Death point:

```
for j=1:size(deathPoints,1)
  mask(deathPoints(j,1),deathPoints(j,2),[2,1])=0.5;
  mask(deathPoints(j,1),deathPoints(j,2),3)=0.95;
 end
```

- The chemotaxis zone :

```
mask(max(1,dest(1)-(interval-i)):min(dest(1)+(interval i), mapSize(1)) ,
max(1,dest(2)-i) :min(dest(2)+i,mapSize(2)),:)=0.9;
```

3.3.3. Robot move:

We were driven to choose a modeling of our paths to which we might use these general concepts in order to develop a bacterial algorithm for the shortest path. For this, we've created a path that looks like this:

Path = {X1, X2, X3…, Xn}, where is the robot's number, n is the length of the path and X an integer ϵ {1, 2, 3, 4}.

3.3.4. The paths

A path is made up of a succession of squares, each of which has a pair of coordinates (position in the environment is matrix-style t) and an integer between 1 and 4 that defines the robot's moving directions. The table is a set of moves carried out by the moving robot, beginning at the starting point and ending at the access point (Table 3.1).

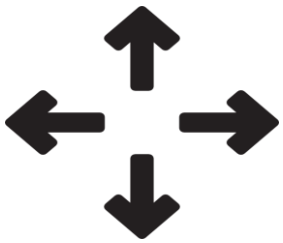


Figure 3.5: The Four Directions of bacterial

Travel cost	1	1	2	1	3	4	….	…	1
The direction	→	→	↓	→	↑	←	….	…..	→

Table 3.1: Diagram representing a path found between 2 points

- **Matlab code (Directions of bacterial) :**

```
if b.position(1)<size(map,1) && map1(b.position(1)+1,b.position(2))==1
  directions = [directions,1];
end
if b.position(2)<size(map,2) && map1(b.position(1),b.position(2)+1)==1
  directions = [directions,2];
end
if b.position(1)>1 && map1(b.position(1)-1,b.position(2))==1
  directions = [directions,3];
end
if b.position(2)>1 && map1(b.position(1),b.position(2)-1)==1
  directions = [directions,4];
end
```

Figure 3.6: MATLAB Code of Direction

3.3.5. The paths after entering the chemotaxis region:

After entering the chemotaxis zone, the path is determined by calculating the distance to become only two directions. The distance is calculated by law the distance from Manhattan :

the separation between two points determined by right-angle axes. In a plane with p1 at (x1, y1) and p2 at (x2, y2), it is |x1 - x2| + |y1 - y2|.

```
function dist = mPointDistance(p1,p2)
    dist = abs(p1(1)-p2(1))+abs(p1(2)-p2(2));
end
```

Figure 3.7: Matlab Code of distance

Every time the bacteria move, the distance is calculated , If the distance decreases, then this path is taken, and if it increases, it is removed from the path. The paths that bacteria can eventually take are : right & top or right & left ; bottom & left ; bottom & top;

```
cDist = mDistance(d,dest);
lastDir = [];
if interval>=cDist
   for i =1:length(directions)
       if cDist>mPointDistance(d.position+dirAddMat(directions(i),:),dest)
        lastDir=[lastDir,directions(i)];
       end
   end
else lastDir = directions;
end
if isempty(lastDir)
   lastDir = directions;
end
dirMove = lastDir(randi(length(lastDir)));
```

Figure 3.8: Matlab Code of Paths that bacteria can take when they enter the chemotaxis zone

After the arrival of the first bacteria , it charts the path it took , Then the path adopted as the shortest path becomes visible only to the bacteria when they enter that area :

```
d.path = [b.path;b.position];
map1 = map;
if path(b.position(1),b.position(2))==1
     for k = 1:3
         map1(:,:,k) = map1(:,:,2).*path;
     end
     map1(dest(1),dest(2))=1;
end
```

Figure 3.9: Matlab Code The path of the first bacteria.

Chapter 3: Realization and Implementation

a. Simulation :

The simulation and experimental findings of a robot's movement as it navigates from a start point to a destination point over numerous obstacles in various positions are shown in the following images.

After the bacteria are randomly moved in the space. When the first bacteria enter the trophic affinity field, they draw the path, the rest of the bacteria follow, and when any bacteria enter the toxic zone, they die automatically. And the bacteria multiply when they reach the end to return to their original number.

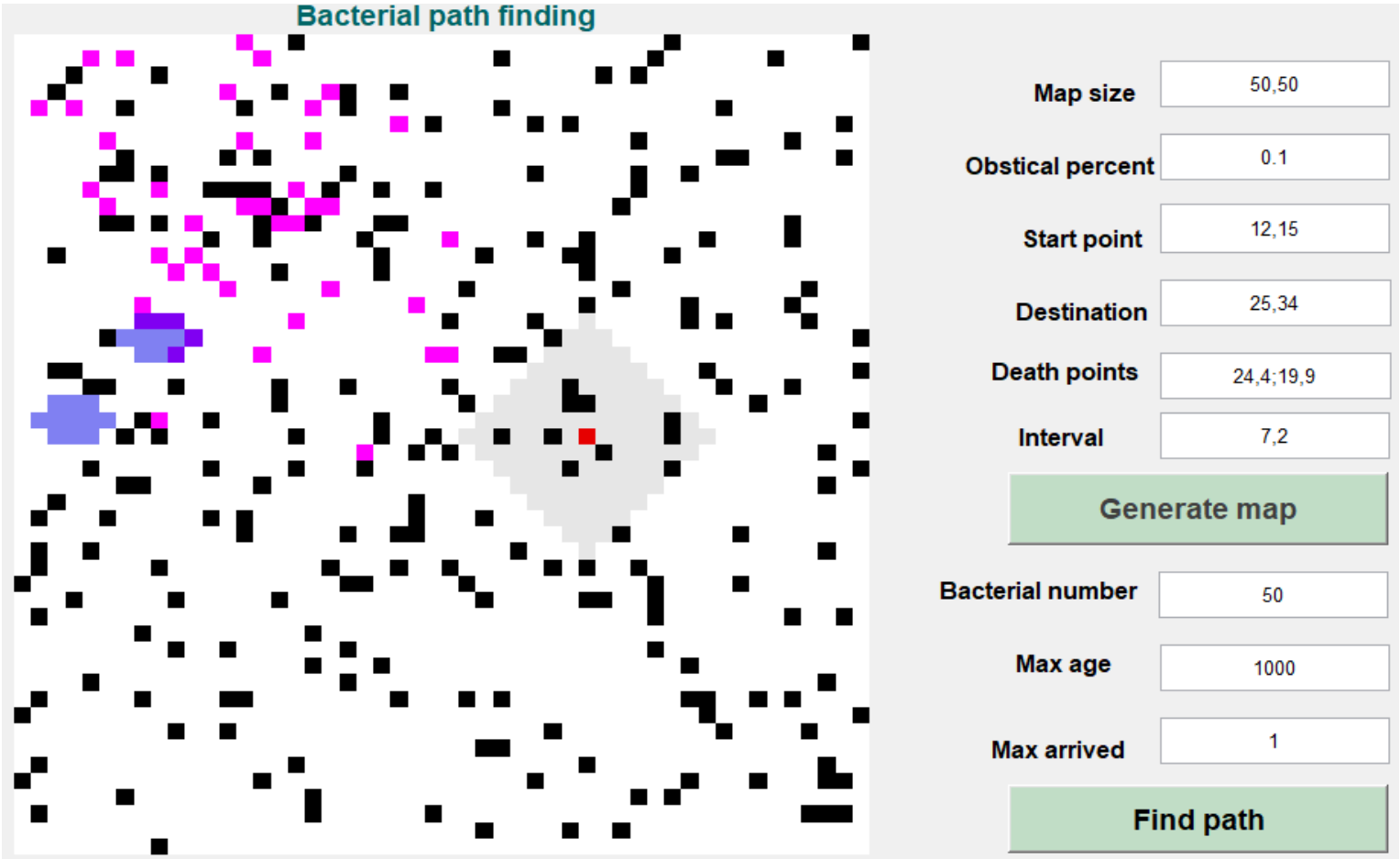


Figure 3.10: Bacteria move in the ocean and avoid obstacles to find the shortest path.

Chapter 3: Realization and Implementation

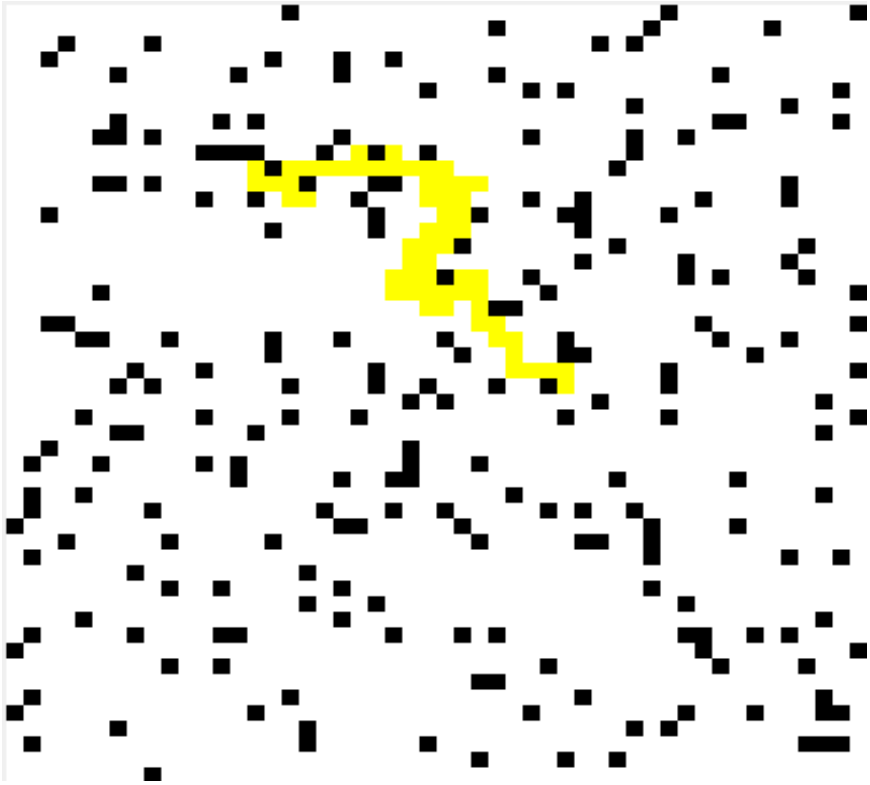


Figure 3.11: The Shortest Path Found by the First Bacteria

Using the Matlab programming language, the following results were obtained. The scripts are executed on a machine with an Intel Core i3 processor and 8GB of RAM.

Bacterial number	150
maximum number of steps that a bacterium can swim	3
elimination-dispersal probability P_{ed}	0.3
number of reproductions N_{re}	5
number of elimination-dispersals Ned	3
number of chemotactic steps	15

Table 3.2: Data that was used to get the results

Chapter 3: Realization and Implementation

3.4. Discussion:

3.4.1. Comparison between different methods:

	Environment	star	goal	obstacles	Euclidean distance	Manhattan Distance	Path Length (with Obstacles)	Convergence rate
BFO	[10X10]	0,0	9,9	10	12,8	18	22	81,81%
ANN							23	78,26%
A*							22	81,81%
A G							22	81,81%
PSO							19,08	94,33%
BFO	[100X100]	0,0	99,99	100	140,00	198	212	93,33%
ANN							235	84,25%
A*							212	93,33%
A G							221	90%
PSO							207	95,65%
BFO	[1000X1000]	0,0	999,999	1000	1412,8	1998	2154	92,71%
ANN							2705	73,86%
A*							00	Stack overflow
A G							2268	88,05%
PSO								96,15%

Table. 3.3 : A comparison between the four algorithms about time and path length.

We have watched algorithms operate; Using a variety of ideas to guide the robots and make sure they reach their destination.

In mobile robotic navigation in unknown environments, we have compared the route planning run times in a common environment under the same conditions for the BFO method along with PSO algorithm[60], ANN [61] ,A* [62] and AG [63], and to demonstrate some real comparison - planning time performance path.

This shows that the algorithm in this thesis can produce a near-perfect global path. We've watched the algorithms in action, using a variety of ideas to guide the bots and make sure they reach their destination without crashing. The initial comparisons between these algorithms are as follows:

Before using bacterial algorithms, we addressed the problem of motion control. The algorithm takes some time to generate the raw data, as we have seen. So far it produces effective answers. On each iteration, these results were better. As a result, the algorithm is effective in guiding the robot through avoiding obstacles.

The goal of this proposal is to create a learning system that helps the robot to move in an unfamiliar area. This method produced good solutions in a constrained environment for predefined tasks effectively, even in very turbulent times even when the complexity increases if the starting and entry points are far apart (on a very wide map). Observing several runs of different algorithms makes performing comparative analysis simple. Although we understand that the performance of different algorithms may vary depending on the size of the map, the following points - Steady, Straight and Backtrack - are another indication of how well the three methods are working. The artificial neural network algorithm failed compared to it. When you use genetic algorithms with few inputs, the A* algorithm performs better in terms of time and efficiency (speed). The problem arises when the size of the map exceeds the constraints and is too large.

As for the results, they were very close each time. When there were few or no obstacles, all of these algorithms performed very well. However, the strength of the bacterial algorithm becomes apparent with increasing complexity. Automated navigation and route planning in dynamic situations have already made significant progress in being able to approach challenges from a variety of perspectives. Some of them are trying to tackle the issue of tracking dynamic targets

3.5. Conclusion:

We tried to present many of the processes we went through in this chapter in a straightforward way. The shortest path of an autonomous moving robot has been discovered using a new evolutionary approach. This method can help the robot to move on its own. The proposed method is more effective, although it is new. In general, solutions based on this methodology enable us to solve our problem.

General Conclusion

In recent years, roboticists have become increasingly interested in the subject of autonomous navigation of mobile robots in natural environments. They hope to steadily raise the degree of autonomy of their robots until they approach complete autonomy and are capable of long missions. Fully autonomous mobile robot mobility in dynamic situations is still a tough subject to tackle. It necessitates the integration of functionalities that allow the perception, decision, and action cycle to be completed. To be able to deal with the vast range of scenarios that the robot may encounter while navigating.

We reviewed in the first chapter the most recent developments in navigation for mobile robots and the various algorithms that can be employed to control them. We've also discussed the numerous processes, as well as the advantages and disadvantages of each technique, and we've concluded that there isn't a single navigation methodology that can be used to address every problem.

In the second chapter, we demonstrated our approach to autonomous navigation for a mobile robot in a hostile environment. This approach uses a BFO Algorithm to determine the shortest path. The robot can move from its starting location to its ending position without clashing in an area with dynamically moving obstacles. According to simulation data, the robot can get there by avoiding the things in its way. In addition to controlling mobile robots in challenging, dynamic environments.

And in the last chapter, Many of the procedures we went through in this chapter were presented as simply as possible. A novel evolutionary method has been used to identify the autonomous moving robot's shortest path.

In general, we used Matlab in this work to illustrate the importance of intelligent systems not just for autonomous control of mobile robots. The Matlab tools enable the modeling of events that are similar to human reasoning in some ways.

The bacterial algorithm's implementation demonstrates that the goal was met by avoiding impediments. We discovered that bacterial algorithms are not always the best; they can be tweaked to solve similar issues.

Following our work, a variety of perspectives are possible. As a result, it would be intriguing to incorporate "dynamics" into our research into autonomous navigation, taking into consideration movable barriers, which would present particularly challenging issues.

This will enable us to detect a greater number and variety of barriers, allowing us to avoid any problems. Navigation is disrupted due to an unanticipated change in the surroundings.

The bacterial algorithm gave good results compared to other algorithms, and it can be developed in the future by integrating it with other algorithms that give better results, especially with the algorithm of birds swarm, ant colony, or bee colony to minimize the path .

References

[1] Weifu W & Ping L. “Planning through Workspace Constraint Satisfaction and Optimization” . Cornell University , 16 Jun 2022 .

[2]Schwartz J, Schwartz T & Sharir M. “On the Piano Movers’ Problem : II. General Techniques for Computing Topological Properties of Algebraic Manifolds” . Advanced applied Mathematics, vol 4, pages 298–351, 1983b.

[3]Schwartz J, T Schwartz & M Sharir. “On the Piano Movers’ Problem : III. Coordinating the Motion of Several Independent Bodies”. International Journal of Robotics Research, vol. 2, no 3, pages 97–140, 1983.

[4] Zhirui S, Jiankun W & Max Q H M. “Multi-Tree Guided Efficient Robot Motion Planning ” . Cornell University , Submitted on 10 May 2022 (v1), last revised 18 May 2022 (this version, v2)

[5]Javier M, Blanca L, Fernando Q , Ramón B , Santiago G & Luis M. “geometrically constrained path planning for robotic grasping with Differential Evolution and Fast Marching Square”. Published online by Cambridge University Press: 04 March 2022.

[6] Jinwook H , Daniel D L & Volkan I. “Neural Cost-to-Go Function Representation for High Dimensional Motion Planning”. International Conference on Robotics and Automation (ICRA), IEEE, 2020.

[7] Xin X & Hui-ling C . “Adaptive computational chemotaxis based on field in bacterial foraging optimization“.Soft Comput 18, 797–807 (2014). https://doi.org/10.1007/s00500-013-1089-4 ,21 August 2014 , articl

[8] Nicole G, Francisco P & Hasan K. “Integration of whole-genome DNA methylation data with RNA sequencing data to identify markers for bull fertility” , Wiley Online Library ,23 April 2020, articl .

[9] Gregory S, Sandeepkumar K, Jens M & Edward W . “Computational Methods in Drug Discovery”, Pharmacological Reviews January 2014, 66 (1) 334-395; DOI: https://doi.org/10.1124/pr.112.007336.

[10] Andrew S, Karl K, Joseph B , Andrew B ,Phil T & Aaron A . “Comparative Analysis of Control Barrier Functions and Artificial Potential Fields for Obstacle Avoidance” , 2013 II International Congress of Engineering Mechatronics and Automation (CIIMA) . DOI: 10.1109/CIIMA32745.2013. 23-25 Oct. 2013.

[11] Daniel C M , Ross B , Gabriel C & Sanya C . “A review of barriers in implementing dynamic electricity pricing to achieve cost-causality”, 25 August 2020, Published by IOP Publishing Ltd, Environmental Research Letters, Volume 15, Number 9 , articl .

References

[12] Gbrant F . “A numerical method for studying liquid drop behavior: Simple oscillation”, Journal of Computational Physics, Scientific Reports ,1973

[13] Nick M, Hao-Tien C, Kendra L, Meeko O & Lydia T . “Hybrid Dynamic Moving Obstacle Avoidance Using a Stochastic Reachable Set-Based Potential Field” ,Journals IEEE , Volume: 33 Issue: 5, 2013.

[14] Muhammad T R K , Malik M S ,Yang R ,Junho S & Dongkyun K . “Aspects of unmanned aerial vehicles path planning: Overview and applications”. International Journal of Communication SystemsVolume 34 ,21 April 2021.

[15] Aaron D A, Samuel C, Magnus E, Gennaro N, Koushil S & Paulo T. “Control Barrier Functions: Theory and Applications” . Conferences 2019 18th European Control

[16] Maria K, Antonio L & Tom L . “Aerospace performance factor and its potential advances”. MATEC Web of Conferences 236, 01007 (2018).

[17] Yizhi L , Mahmoud H & Houtan J . “Brain-computer interface for hands-free teleoperation of construction robots” . Automation in Construction Volume 123, March 2021

[18] Yong D , Yuxin C, Yajuan Z & Sankaran M . “Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment”. Volume 12, Issue 3, March 2012.

[19] Allison B S . “Local and Global Minima in Visual Completion: Effects of Symmetry and Orientation” , May 1, 1994 , articl.

[20] Ulrik B . “On variants of shortest-path betweenness centrality and their generic computation” ,Social Networks Volume 30, Issue 2, May 2008 .

[21] Guansong C. “Intelligent Financial Data Analysis and Decision Management Based on Edge Computing”. Journal of Sensors , 2022 , Article.

[22] Omar S, Rabie B , David D, AbedlHakim A, Nicolas B & Pierre F. “Path planning: A 2013 survey” , Conferences of 2013 International articl, Publisher IEEE .

[23] Peter E, Hart; Nils J, Nilsson & Bertram R. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths” . Journals ,Volume: 4 Issue: 2, 1968

[24] SB Maind & P Wankar . “Research Paper on Basic of Artificial Neural Network”. International Journal IJRITCC ,2014

[25] Ioannis A , Valentin R , Benoit C , Desen K ,Sonam N ,Aristides K ,David F , Sergio E & Steve W . “Artificial intelligence and machine learning approaches to energy demand-side response: A systematic review” . Volume 130, September 2020, 109899.

References

[26] Nivetha S . “Recognition and Digitization of Handwritten Text using Histogram of Gradients and Artificial Neural Network”. Vol. 12 No. 6 (2021) , Turkish Journal of Computer and Mathematics Education (TURCOMAT).

[27] Fengrui Y, Xueliang F , Honghui L & Gaifang D .“Improved Roulette Wheel Selection-Based Genetic Algorithm for TSP”. International Conference on Network and Information Systems for Computers (ICNISC) , 2016 .

[28] Keiron O & Ryan N . “An Introduction to Convolutional Neural Networks”. 2 Dec 2015 , Computer Science , Neural and Evolutionary Computing.

[29] Lotfi A Z. “Fuzzy logic—a personal perspectiv”. Fuzzy Sets and Systems ,Volume 281, 15 December 2015, Pages 4-20

[30] Lotfi A Z. “Fermatean fuzzy sets”. fuzzy logic and fuzzy systems , Journal of Ambient Intelligence and Humanized Computing volume 11, pages663–674 (2020)

[31] Bahman Z & Masoud M . “Business Resilience System (BRS): Driven Through Boolean, Fuzzy Logics and Cloud Computation” . March 2017, book .

[32] Ardakani A H H , Shojaei S , Shahvaran A R, Kalantari Z, Cerdà A & Tiefenbacher J ."Selecting potential locations for groundwater recharge by means of remote sensing and GIS and weighting based on Boolean logic and analytic hierarchy process". SpringerLink , Published: 06 December 2021.

[33] Timothy J R . “Membership Functions, Fuzzification and Defuzzification”. 2000 , book.

[34] Teng F, Xinxin W, Liyi Z ,Yong Z & Lei C ; "Research on improved ant colony optimization for traveling salesman problem". Mathematical Biosciences and engineering , 06 June 2022

[35] Stamatios C N, Deneubourga J L . "The effect of idiosyncrasy on aggregation in group-living organisms". Journal of Theoretical Biology Volume 542, 7 June 2022.

[36] Guangshi J . "Simulation Research on the Palm Mechanism of Volleyball Robot Based on Artificial Intelligence and Ant Colony Optimization Algorithm ". Volume 2022 . Article ID 6030545.Security and Communication Networks .10 Mar 2022

[37] Annu L, Kunal G& Kriti C. “Genetic Algorithm- A Literature Review”. Publisher: IEEE, 2019 International Conference.

[38] Esteva A. Kuprel B, Novoa R A, Ko J, Swetter S M & Blau H M. “Dermatologist-level classification of skincancer with deep neural networks” . Nature2017,542, 115–118

References

[39] Spring B, Quentin L, Mahmut S S, Vijay K & Stephen C. “Pratt, Experimental Study and Modeling of Group Retrieval in Ants as an Approach to Collective Transport in Swarm Robotic Systems”. Volume: 99 Issue: 9,2011

[40] Jagdish C B ,Harish S & Shimpi S J . “Artificial bee colony algorithm: a survey”. 2013 articl. International Journal of Advanced Intelligence ParadigmsVol. 5, No. 1/2.

[41] Lučić T . “The Bee Colony Optimization (BCO)”. metaheuristic has been well proposed recently ,2009, articl.

[42] Miloš N ,Milica Š , Dragana M & Jovana Ć . “Bee Colony Optimization metaheuristic for fuzzy membership functions tuning”. Volume 158, 15 November 2020, articl .

[43] Elbanhawi M & Simic M . “Sampling-Based Robot Motion Planning: A Review”. Journals & Magazines , IEEE Access, 2014.

[44] Khatib O. “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots” , IEEE International Conference on Robotics and Automation, St. Louis, Missouri, pp.500-505, March 25-28, 1990.

[45] Li P & Wen S. “Realization of the Path Planning Algorithm Based on Fuzzy Control”. Journal of Hangzhou Dianzi University 27(6), 82–86 (2007)

[46] Goldberg D E. “Genetic algorithms in search, optimization, and machine learning”, Addison Wesley 1987.

[47] Benmachiche A., Bouhadada T, Laskri M T & Zendi A. “A dynamic navigation for autonomous mobiles robots, Intelligent Decision Technologies”, ISSN 1875-8843 (E), 10 (1). 2016.

[48] Haupt R & Werner D , “genetic algorithms in electromagnetics” , 2007 , book

[49] Kevin M Passino. “Bacterial Foraging Optimization”. International Journal of Swarm Intelligence Research …, 2010 - igi-global.com

[50] Xiaobing G & Baoyu X ,”Improved Bacterial Foraging Optimization Algorithm with Comprehensive Swarm Learning Strategies”, First Online: 13 July 2020, articl .

[51] ChenY-P ,Li Y ,Wang G ,Zheng Y-F ,Xu Q , Fan J-H & Cui X-T . “A novel bacterial foraging optimization algorithm for feature selection” .15 October 2017, articl .

[52] Huiling C, Qian Z ,Jie L , Yueting X & Xiaoqin Z . “An enhanced Bacterial Foraging Optimization and its application for training kernel extreme learning machine”. Volume 86, January 2020, articl .

References

[53] Jun T, Gang L & Qingtao P . “A Review on Representative Swarm Intelligence Algorithms for Solving Optimization Problems: Applications and Trends”. 2021 , articl .

[54] Kousik D , Brototi M ,Paramartha D , Jyotsna K M & Santanu D . “A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing”. Volume 10, 2013, articl .

[55] De-gan Z , Si L , Xiao-huan L ,Ting Z & Yu-ya C . “Novel dynamic source routing protocol (DSR) based on genetic algorithm-bacterial foraging optimization (GA-BFO)”.16 October 2018, articl .

[56] M. Tripathy, S. Mishra, L. L. Lai & Q. P. Zhang ; “Transmission Loss Reduction Based on FACTS and Bacteria Foraging Algorithm”;articl , 2006.

[57] Gábor P & Árpád T , “A novel safety assessment method through the example of a reduced complexity binary integer autonomous transport model” , Volume 217, January 2022, articl.

[58] Albert R, Jeremy K, William A, David B, Bill B, Chansup B ,Matthew H ,Peter M ,Julie M , Andrew P & Antonio R “ LLSuperCloud: Sharing HPC systems for diverse rapid prototyping”. Conferences , 2013 IEEE

[59] Cadieux J & Ariba Y. “Manuel Matlab “ Départements GEL et Mécanique version,0.1 page 6. A book

[60] Boutabia I. “ A Particle swarm optimization approach for finding paths for a robot in an environment with obstacles” .Master Académique SII, université chadli Bendejdid el Tarf. .2022

[61] Merdaci A. “A New Approach To Finding The Shortest Path Of a Mobile Robot Based On Artificial Neural Networks” .Master Académique SII, université chadli Bendejdid el Tarf. .2021

[62] Tridi S .“Trajectory planning of a mobile robot based on Artificial Potential Fields”.Master Académique SII, université chadli Bendejdid el Tarf. .2021

[63] Mellouk A .“AN Evolutionary Approach To Searching The Shortest Of A Self-Contained Mobile Robot”. Master Académique SII, université chadli Bendejdid el Tarf. .2020

- Springer - ICISAT'2022 will be a Hybrid event, comprising both an in-person conference and a virtual conference.
- ICISAT'2022 will be a virtual / hybrid conference depending on the CoVID-19 situation...
- Due COVID'19 a virtual track will be available presentations for author who have travel restrictions...

ABOUT ICISAT'2022

12th ICISAT International Conference

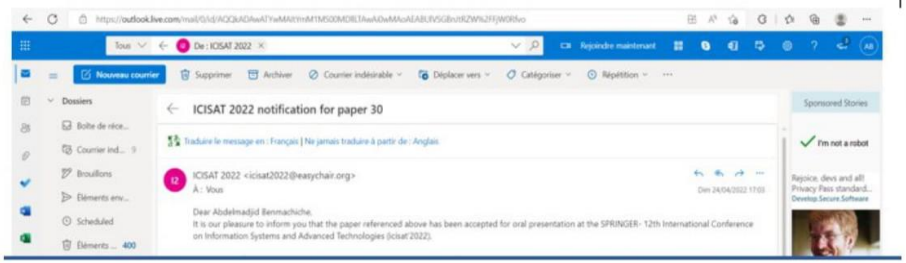
Indexing:

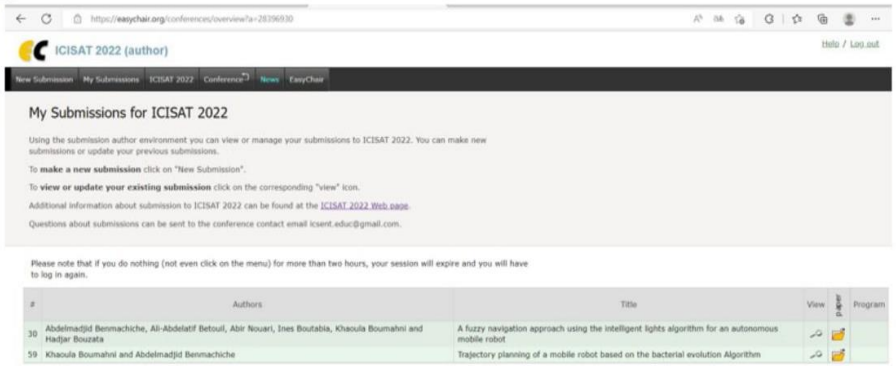
The books of this series are submitted to ISI proceedings, SCOPUS, Google Scholar and Springerlink**

ICISAT'2022 is a forum for researchers, developers and Industrials in the information systems field and the continuance of the following event: ICIST'11 (Tebessa, Algeria), ICIST'12 (Sousse, Tunisia), ICIST'13 (Tangier, Morocco), ICIST'14 (Valancia, Spain), ICIST'15 (Istanbul, Turkey), ICIST'16 (Barcelona, Spain), ICIST'2017 (Dubai, UAE), ICIST'2018 (Istanbul, Turkey), ICIST'2019 (Cairo, Egypt), ICIST'2020 (Lecce, Italy) and ICISAT'2021 (Tunisia).
Since 2021 edition, the new acronym of the event is ICISAT (International Conference on Information Systems and Advanced Technologies). It reports progress and development of methodologies, technologies, planning and implementation, tools and standards in information systems. The conference looks also at socio-economic aspects, impacts and success factors of information systems. ICISAT'2021 aims at addressing issues related to the design, development and use of information systems in organizations from a multidisciplinary perspective, and to discuss, the research, teaching and professional practice in the field. ICISAT 2022 brings together leading academics and professionals in information systems from around the world. It aims at providing a platform for discussions on issues that take into consideration the social and technological aspects of information systems. The conference program includes paper presentation, keynote talks from prominent academics, Posters and Panels.

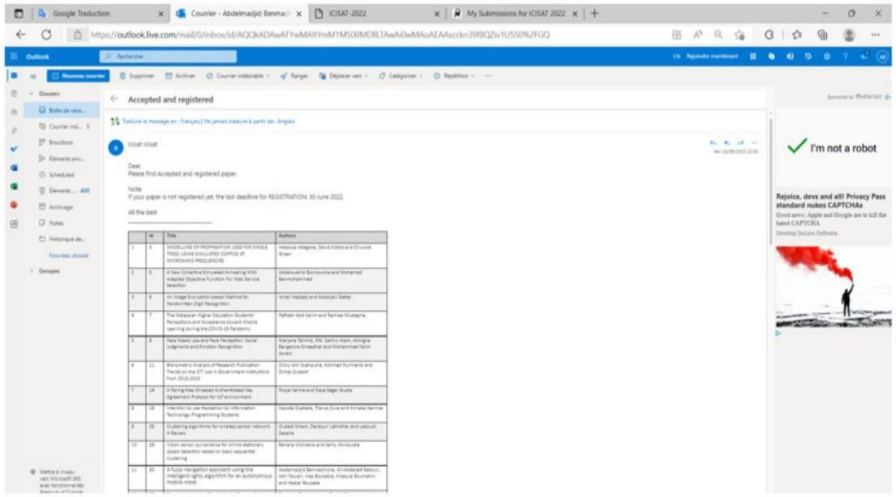
ICISAT'2022 KEYNOTE SPEAKERS AND GUEST EDITORS

Sean EOM	Valentina Emilia Balas	Jean-Paul Van Belle	Mirjana Ivanovic	Amirrudin KAMSIN
Southeast Missouri State University, USA	University of Arad, Romania	University of Cape Town, South Africa	University of Novi Sad, Serbia	University of Malaya, Kuala Lumpur, Malaysia





#	Authors	Title	View	Program
30	Abdelmadjid Benmachiche, Ali-Abdelatif Betouil, Abir Nouari, Ines Boutabia, Khaoula Boumahni and Hadjar Bouzata	A fuzzy navigation approach using the intelligent lights algorithm for an autonomous mobile robot		
59	Khaoula Boumahni and Abdelmadjid Benmachiche	Trajectory planning of a mobile robot based on the bacterial evolution Algorithm		



11	30	A fuzzy navigation approach using the intelligent lights algorithm for an autonomous mobile robot	Abdelmadjid Benmachiche, Ali-Abdelatif Betouil, Abir Nouari, Ines Boutabia, Khaoula Boumahni and Hadjar Bouzata

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
CHADLI BENDJEDID EL-TARF UNIVERSITY
SCIENCES AND TECHNOLOGY FACULTY
COMPUTER SCIENCE DEPARTMENT

CERTIFICATE OF PARTICIPATION

Awarded to:

Khaoula Boumahni

For presenting the paper entitled:

Trajectory planning of a mobile robot based on the BFOA Algorithm

At the 1st International Conference on Autonomous Systems and their Applications (ICASA'22).

Author(s): Khaoula Boumahni Abdelmadjid Benmachiche, Amina Makhlouf and Aziza Merdaci.

Dr Abdelmadjid Benmachiche

International Conference On Autonomous Systems and their Applications • First International
ICASA'2022
General Chair

UCBET / FST / INF - P.O. Box 73, El-Tarf 36000, Algeria - : http://univ-eltarf.dz - : cniati.info@gmail.com

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
ChadliBendjedid El-Tarf University
Sciences and Technology Faculty
Computer Science Department

CERTIFICATE OF PARTICIPATION

Awarded To:

BOUMAHNI Khaoula

For presenting the **Poster** entitled:

« communication in multi-robot systems: A survey »

At the First **N**ational **C**onference on **A**rtificial **I**ntelligence and **I**nformation **T**echnologies (**CNIATI'20**).

Co-Author(s): Benmachiche Abdelmadjid, Bouguerne Imen, Makhlouf Amina

El-Tarf on May 24th, 2021
Conference Chair

جامعة الشاذلي بن جديد - الطارف
كلية العلوم والتكنولوجيا
الملتقى الوطني الأول حول "الذكاء الإصطناعي
وتكنولوجيات المعلومانية" "CNIATI 20"
رئيس الملتقى الدكتور
عبد المجيد بن مشيش

UCBET / FST/ INF- P.O. Box 73, El-Tarf 36000, Algeria
: http://univ-eltarf.dz
: +213 (0) 38 88 09 87