T.C.

İSTANBUL KÜLTÜR UNIVERSITY

INSTITUTE OF GRADUATE STUDES

MULTI – OBJECTIVE TRAJECTORY TRACKING FOR AN AUTONOMOUS MOBILE ROBOT IN DYNAMIC ENVIRONMENTS USING EVOLUTIONARY ALGORITHMS

Masters of Applied Science Thesis

Mahyar TEYMOURNEZHAD

1700005194

Department: Computer Engineering

Program: Computer Engineering

Supervisor: Prof. Dr. Özgür Koray ŞAHİNGÖZ

JANUARY 2023

T.C.

İSTANBUL KÜLTÜR UNIVERSITY

INSTITUTE OF GRADUATE STUDES

MULTI – OBJECTIVE TRAJECTORY TRACKING FOR AN AUTONOMOUS MOBILE ROBOT IN DYNAMIC ENVIRONMENTS USING EVOLUTIONARY ALGORITHMS

Masters of Applied Science Thesis

Mahyar TEYMOURNEZHAD

1700005194

Department: Computer Engineering

Program: Computer Engineering

Supervisor: Prof. Dr. Özgür Koray ŞAHİNGÖZ

Members of Examining Committee:

Prof. Dr. Özgür Koray ŞAHİNGÖZ

Associate Prof. Dr. Akhan Akbulut

Prof. Dr. Murat Ermiş

JANUARY 2023

FOREWORD

I have always been thankful of meeting great people in my life. The deep and lasting influence of these people in my life is undeniable, from whom I have learned wisdom, knowledge and life lessons.

Prof. Dr. Özgür Koray Şahingöz has undoubtedly been one of the most influential professors in my academic life. His patience and great energy and motivating ability during the project is exemplary. Along with scientific discussions, I always benefited greatly from his effective instructions in my educations and career. During my researches for this thesis and publishing the articles I was always benefited from his instructions and he never stopped supporting me.

I would like to express my special thanks and appreciation to my dear mother, who has always been my inspiration, and has always devoted herself to my life, education and development, and who has always supported and encouraged me with her good prayers and advice.

I sincerely thank my dear wife for all her support and patience while I was busy with my research. I appreciate her for providing suitable conditions for my study and research while I was studying and conducting research for the preparation of this thesis. I am grateful to my family for their support and encouragement all this time.

CONTENTS

FIGURE LIST

TABLE LIST

ABBREVIATIONS

ACO	: Ant Colony Optimization
FL	: Fuzzy Logic
UAV	: Unmanned Aerial Vehicle
AUV	: Autonomous Underwater Vehicle
CDM	: Cell Decomposition Method
GA	: Genetic Algorithm
PSO	: Particle Swarm Optimization
SI	: Swarm Intelligence
AI	: Artificial Intelligence
TSP	: Travelling Salesman Problem

Üniversite	**:**	**T.C. İstanbul Kültür Üniversitesi**
Enstitüsü	**:**	**Lisansüstü Eğitim Enstitüsü**
Anabilim Dalı	**:**	**Bilgisayar Mühendisliği**
Program	**:**	**Bilgisayar Mühendisliği**
Tez Danışmanı	**:**	**Prof. Dr. Özgür Koray ŞAHİNGÖZ**
Tez Türü ve Tarihi	**:**	**Yüksek Lisans – Ocak 2023**

ÖZET

DİNAMİK ORTAMLARDA OTONOM HAREKETLİ BİR ROBOT İÇİN EVRİMSEL ALGORİTMALAR KULLANARAK ÇOK AMAÇLI YÜZEY TAKİBİ

Mobil robotlar, günümüz modern yaşamında insan yaşamının neredeyse tüm yönlerini etkileyen çok önemli bir role sahiptir; askeri ve tarımdan eğitime, sağlığa, sosyal hizmetlere ve turizme. Mobil robot, bir insansız hava aracı (İHA), insansız bir denizaltı, restoranda çalışan bir garson robot veya dağıtım sisteminin bir temsilcisi olabilir. Mobil robotlardan bahsederken ilk akla gelen, farklı arazilerde otonom olarak hareket etmelerini sağlayan navigasyon sistemi ve onlar için yörünge planlamasıdır. Mobil robotların otonom bir şekilde çalışmasına izin vermek, ortamdaki engellere çarpmaktan kaçınmanın yanı sıra, tıpkı bir insan operatörün yapacağı gibi en kısa ve en sorunsuz olan ve en düşük maliyetli olan optimum rotayı bulmak için üzerlerine yapay bir sistemin yerleştirilmesini gerektirir. . Buna olası çözümler arasından en iyi yörüngeyi seçmek için mobil robotun davranışlarını iyileştirmek için hareketlerinin izlenmesi ve gözetlenmesi denir.

Mobil robotik alanındaki en önemli araştırma alanlarından biri, her tür hareketli insansız robotik sistem için yörüngeleri izlemek ve yönlendirmek için en iyi yöntemlerin geliştirilmesidir. Çevresel engeller ve engellerle karşılaşmadan bir rota bulmak için yörünge takibi yapılır. Yol planlaması için pek çok çalışma yapılmış olmasına rağmen, bunların çoğu gerçek dünyadaki duruma daha az benzeyen statik ortamlarda yapılmıştır. Gerçek hayat senaryosunda, yönlendirme ortamında birden fazla engelimiz olabilir ve engellerin şekil ve boyutları birbirinden farklı olabilir. Ayrıca, farklı hız ve özelliklerde farklı yönlerde hareket eden hareketli bazı engeller olabilir. Bu, insansız robotik sistemler için yörünge takibini zorlaştırır.

Yol planlama problemlerinin çözüm alanı çok geniş olduğu için NP-Zor problemler olarak sınıflandırılır. Problemi çözmek için buluşsal bir yaklaşıma sahip olan evrimsel algoritmalar, şimdiye kadar NP-zor problemlerin çözümünde çok önemli bir yol kaydetmiştir ve hala daha iyi çözümler elde edebilmek için bunların geliştirilmesine büyük bir ilgi vardır.

Problemi çözmek için kesikli bir yaklaşım mı yoksa sürekli bir yaklaşım mı benimsememize neden olan yol planlama problemi tanımına bağlı olarak, en uygun çözümleri bulmak için evrimsel algoritmalardan uygun olanı seçebiliriz.

Bu tezde, sabit ve hareketli engellerin bulunduğu dinamik ve bilinmeyen uzaylarda hareket eden robotların rotalama probleminin çözümü için iki farklı bakış açısının incelenmesi ve karşılaştırılması önerilmiştir. Amacımız, başlangıç noktasından hedef noktaya en uygun yolun elde edildiği ve bu yol boyunca robot tarafından statik ve dinamik engellerle çarpışmadan kaçınan bir algoritma tasarlamak ve uygulamaktır. İlk olarak karınca kolonisi optimizasyon algoritması ile yönlendirme problemi çözülmekte daha sonra bulanık mantık yöntemi ile aynı ortamda yönlendirme işlemi yapılmaktadır. Yapılan araştırmalar, bulanık yöntemin daha uygun etkinliğini göstermektedir; zaman alıcı süreçlerde uygulamadaki basitlik açısından ve dinamik ve statik ortamlarda hareket yolunun zaman ve uzunluğunun araştırılmasının sonuçları, bu yöntemin, özellikle dinamik bir ortamda, diğer evrimsel algoritmalara kıyasla gücünü göstermektedir.

Anahtar Kelimeler: Çok Amaçlı Yol Planlama, Otonom Mobil Robot, Karınca Kolonisi Optimizasyonu, Bulanık Mantık, Evrimsel Algoritmalar.

University : **İstanbul Kültür University**
Institute : **Institute of Graduate Studies**
Department : **Computer Engineering**
Program : **Computer Engineering**
Thesis Advisor : **Prof. DR. Özgür Koray ŞAHİNGÖZ**
Degree Awarded and Date : **MS – January 2023**

ABSTRACT

MULTI – OBJECTIVE TRAJECTORY TRACKING FOR AN AUTONOMOUS MOBILE ROBOT IN DYNAMIC ENVIRONMENTS USING EVOLUTIONARY ALGORITHMS

Mobile robots have got very significant role in today's modern life affecting nearly all aspects of human beings' lives; from military and agriculture to education, healthcare, social services, and tourism. Mobile robot can be an unmanned aerial vehicle (UAV), an unmanned submarine, or a waiter robot working at the restaurant, or an agent of delivery system. While talking about mobile robots, the first think sparks to think about is the navigation system and trajectory planning for them so that enables them to move in different terrains autonomously. Letting mobile robots run in an autonomous way needs an artificial system to be embedded on them to find the optimal route which is the shortest and the smoothest one with the lowest cost, exactly as a human operator would act, besides avoiding colliding the obstacles in the environment. This is called monitoring and surveillance of mobile robot's motions to improve their behavior in order to choose the best trajectories among the possible solutions.

One of the most crucial areas of research in the field of mobile robotics is the development of the best methods for monitoring and routing trajectories for all types of moving unmanned robotic systems. Trajectory tracking is done in order to find a route without encountering environmental obstacles and hurdles. Although, lots of studies have been done for path planning, but most of them were done in static environments which is less like the real-world situation. In a real-life scenario, we may have multiple obstacles in the routing environment and the shapes and sizes of the obstacles may vary from each other. Furthermore, there may be some moving obstacle moving in different directions with different speed and characteristics. This makes the trajectory tracking more difficult for unmanned robotic systems.

As the solution space for path planning problems is very wide, it is categorized as NP-Hard problems. The evolutionary algorithms having a heuristic approach to solve the problem

have recorded a very significant trail in solving NP-hard problems till now and still there is a great appeal on improving them to be able to get better solutions.

Depending on the path planning problem definition which induces us whether to take a discrete approach or a continuous approach to solve the problem, we can choose the appropriate one among the evolutionary algorithms to find the most appropriate solutions.

In this Thesis, the investigation and comparison of two different points of view for solving the routing problem of moving robots in dynamic and unknown spaces with the presence of fixed and moving obstacles have been proposed. Our goal is to design and implement an algorithm according to which an optimal path is obtained from the starting point to the target point and also avoids any collision with static and dynamic obstacles during this path by the robot. First, the routing problem is solved with the ant colony optimization algorithm, and then routing is done in the same environment with the fuzzy logic method. The conducted investigations show the more appropriate efficiency of the fuzzy method in terms of simplicity in implementation in time-consuming processes and the results of the investigation of the time and length of the movement path in dynamic and static environments indicate the strength of this method compared to other evolutionary algorithms, especially in a dynamic environment.

Keywords— Multi-Objective Path planning, Autonomous Mobile Robot, Ant Colony Optimization, Fuzzy Logic, Evolutionary Algorithms.

1. INTRODUCTION

Today, robots have affected our lives by entering directly or indirectly in many areas. It is important that the working principles of robots, which have many different areas of use and become more and more needed to make life easier, can be determined in different environments where they are used. The word robot entered the language of humanity by Karel Čapek in the 1900s and was defined as workers who can work on their own [1].

Robots are electro-mechanical devices that understand and interpret their surroundings with the help of sensors or remote computer control and make decisions according to the result, which leads to perform or stop movements. While some types of robots consist of arms controlled by computers or processors, there are also types of these robots that move on legs, rails, wheels or pallets [2].

Industrial robots and robot manipulators were the first application areas of robots, but current applications are focused on autonomous robots that can work. Especially in industry, robot manipulators are preferred in jobs that cannot be overcome with human power or in conditions that are not suitable for humans. Robot manipulators with fixed or mobile structures, which are used in fields such as assembly, transportation, loading and unloading in the industry, do not have any movement capability other than the given commands and programmed actions. For this reason, robotics science has turned to robots that can make their own motion decision. This group, which acts autonomously and aims to fulfill the given task, is called mobile robots. With the rapid progress of technology, these mobile robots, which are used in many fields, have been used in fields such as military, disaster and search studies, space research, medicine and health sciences, hobby, entertainment and education in recent years [2].

Most mobile robot applications involve human control and management. In the actions of robots that perform some critical functions, all control may be in humans. However, current studies have shifted to minimizing or zeroing human influence and interference in the movement of robots.

Being able to recognize obstacles, determine a path for itself, and perceive the environment is essential for a robot to move autonomously. By means of sensors, the robot senses its surroundings and processes the data to create an environment map or try to reach the target according to a predefined map. If the environment is obstructed, the robot must also avoid

obstacles. Under all these conditions, in some applications, robots are expected to make a certain movement, while in some applications, robots are aimed to reach the desired target from a starting point. For the efficient use of resources such as time, money, energy and the robot, it may be desirable to reach the target in the shortest way, the safest, the most accurate and according to different performance indicators [3].

1.1. Definition and Importance of the Thesis Subject

Path planning problems can be expressed as operationally very complex problems. In Artificial Intelligence these problems are mentioned as NP-hard problems. The reason for this is that there are many alternative route options in the terrain where the trajectory tracking will be processed and it is aimed to find the optimum pathway among these routes without colliding the obstacles. The first goal in path planning is solving the collision problem for the picked route in the terrain. For this, there are different methods and techniques in the literature to solve the problems of avoiding colliding many obstacles. The obstacles in the other hand can be fixed road blocks or some dynamic and moving obstacles such as human bodies, vehicles, or other robots besides the static and fixed robots. These methods are also called path planning methods by some researchers. These path planning methods and techniques can be expressed as classical path planning, probabilistic path planning and heuristic or meta heuristic path planning methods. Especially in recent years, it has been seen that heuristic and meta heuristic routing approaches are being used more in solving path planning problems. The reason for this is to avoid complex processes and to obtain better results by making time more efficient.

1.2. Objective of the Thesis

The aim of this thesis is to search the literature about the solution of the path planning problems of mobile robots, to provide knowledge about path planning and to plan the path from the starting point to a targeted point according to the optimum criteria with the help of the Ant Colony Optimization Algorithm, which is a heuristic approach and employing Fuzzy Logic for real – time trajectory tracking in unknown dynamic environments. In parallel with this, a simulation environment has been developed to monitor the results according to the performance of the Ant Colony Optimization Algorithm vs Fuzzy Logic. Thanks to this simulation program, it is aimed to find the safest and the shortest trajectory by entering the required values and making evaluations.

1.3. The Content of the Thesis

In the second part of the thesis, a literature summary of the studies obtained by comprehensive literature review on path planning for mobile robots using the evolutionary algorithms is given.

In the third part of the thesis, firstly the path planning problem is introduced, then the path planning approaches and the planned terrain are discussed. In addition, detailed information is given about the different techniques proposed and developed for the path planning problems of mobile robots.

In the fourth chapter of the thesis, the development of the ant colony algorithm and Fuzzy Logic based approach, their mathematical formulation, some ant colony systems and their application to different problems are explained, starting with the explanation of the real ant behaviors and then discussing Fuzzy System and fuzzy-based inference to gain the optimized solution regarding the ant colony algorithm and Fuzzy Logic are used for the path planning problem.

In the fifth chapter of the thesis, firstly, basic information about MATLAB and the simulation environment is given, and the targeted path planning algorithms are explained by introducing the simulation program we have developed. Again, Fuzzy Logic with a the real – time obstacle avoidance method in dynamic environments and the ant colony algorithm we developed, which are the methods we used to solve the path planning problem, are explained in detail.

In the sixth chapter of the thesis, the findings obtained by entering different parameter values using the simulation program we have designed are given. Comparing the results over the findings obtained, the results obtained are shown numerically and graphically.

In the seventh chapter of the thesis, evaluations are made with the results obtained and suggestions for the studies to be done are given.

2. LITERATURE REVIEW

Studies on path or motion planning problems for mobile robots started in the mid-1960s and gained momentum in the 1980s and have become an important research topic since then. In this section, a literature review and summary of the path planning studies made for mobile robots using evolutionary algorithms, especially the ant colony algorithm and Fuzzy Logic, from previous years until now is presented. The first study based on the natural life of ants was made by Deneubourg et al. [4]. This study was adapted for transportation systems. In their study, they took the positive feedback behavior of ants' natural life as an example, and thus they aimed to create transportation networks.

Sauter et al. [5] used ACO and genetic algorithm in the same algorithm for the path planning of mobile robots by avoiding obstacles. They used the genetic algorithm to automatically adjust the pheromone material used in the ant colony algorithm to find shortcuts in dynamic environments, and they named it synthetic pheromone. As a result of the simulation studies, it was seen that the appropriate pheromone value was found shortly after the operation of this algorithm and gave good results.

Jin et al. [6] developed an ant colony algorithm for the path planning of the mobile robot in free flight fields. They showed the efficiency of the results by adding different obstacles (mountain, hill, etc.) to these fields with the simulation programs they developed.

Fan et al. [7] developed a condensed ACO for a continuous optimization problem involving constraint conditions to be used in path planning of mobile robots for the first time in their study. By making comparisons between important coefficients with this algorithm, by dynamically creating possible paths, they selected the transition probabilities for the ants. They showed the efficiency of the algorithm with simulation results.

Hsiao et al. [8] used the ant colony algorithm to plan the optimal path in an area without obstacles. They created their algorithm as a simulation program in C++ programming language. Thanks to this program, the routes were randomly created and optimal path planning was made among these created routes.

Fan et al. [9] developed ACO to overcome some of the existing difficulties of the techniques used for path planning of mobile robots. They adapted this algorithm to solve a continuous

optimization problem with some constraint conditions. They applied the algorithm with a simulation program in different complex environments and demonstrated the efficiency of the results.

Garro et al. [10] proposed a variant of the ant colony to optimize a path for the mobile robot to reach its destination. In this algorithm, some parameters of the ant colony algorithm are adjusted by using genetic algorithm. In this study, they saw that they obtained better results by comparing the genetic algorithm they created with the ant colony and the stagnation in real environments with the classical ant colony.

Wen et al. [11], using the ant colony algorithm, developed a path planning algorithm for the aircraft according to height or altitude above sea level. They tried to find a better way of convergence by manipulating some parameters of the ant colony and evaluated the efficiency of the algorithm by creating a simulation program.

Xu et al. [12] developed a new ACO for path planning problems and simulated this method with a Java program. They tested the results of the proposed method in different environments and compared it with the Dijkstra algorithm and saw the efficiency and effectiveness of the algorithm.

Mohamad et al. [13] proposed a new approach to the path planning problem by using probabilistic roadmap and ant colony algorithm. The ant colony algorithm was used to find the optimal path in this area by creating an area to be planned with a probabilistic road map. As a result of experiments with this proposed algorithm, it has been seen that it is more capable and effective in finding what is desired between the starting point and the target point at an appropriate runtime.

Liu et al. [14] proposed a new algorithm using the ant colony algorithm to plan the path of mobile robots in two-dimensional environments. With the proposed algorithm, it was aimed to solve the drawbacks of local optimization. As a result of their work, they found that they accelerated the search for the optimum value in these environments.

Mei et al. [15], in their study, obtained a new algorithm by combining the algorithms of the ACO for global path planning and the potential fields approach for local path planning. In the study, the pheromone parameter used in the ant colony algorithm was used to avoid being stuck at the local minima in the artificial potential area. According to the results of their simulation,

they saw that this new algorithm was efficient to meet the real-time demands and compared the results of this algorithm with the work done with the genetic algorithm.

Liu et al. [16] presented an ant colony algorithm and a local search algorithm with obstacle avoidance in complex and static environments for path planning of mobile robots. In the proposed algorithm, they added some special tasks to the classical ant colony algorithm. When they encountered a collision while searching for paths, they created a table for these places and tried to neutralize those areas by applying a penalty function so that the mobile robot avoids these obstacles. By adapting the algorithm with the simulation program, they showed that the algorithm is effective and efficient according to the results obtained.

Lv and Feng [17] presented a new method using a numerical potential field technique and the ant colony algorithm together. In this new method, the digital potential field is used to create the environment in which the mobile robot operates. In the study, the movement of obstacles is taken into account by changing the local potential values. Then, the ant colony algorithm was used to search for the path and find the optimal path. The results of the trials on the implementation of the algorithm showed that the algorithm was effective.

Mohamad et al. [18] applied the ant colony algorithm to 3D robot path planning. In this study, they created a path planning area by reusing the probability-based roadmap technique that they used in their previous studies. Again, they found that this study was more effective and intelligent.

Ye et al. [19] proposed a self-adaptive CCS for path planning by mobile robots to avoid obstacles. By applying the algorithm in MATLAB-Toolbox, they made some adaptations on the classical ant algorithm and found better ways. As a result of the studies, it has been seen that the algorithm is more effective and suitable than the classical ant algorithm.

Guan-Zheng et al. [20] proposed the ant colony algorithm for the formation of the global optimal path with real-time mobile robots in their study. This study consists of three steps. In the first step, they used MAKLINK graph theory to construct the free space model of the mobile robot. In the second step, the Dijkstra algorithm was used to find a sub-optimal path without collisions. In the third step, the ant colony algorithm was used to create the globally optimal path instead of the sub-optimal path. The simulation program made with this algorithm was compared with the work done with the genetic algorithm before, and it was seen that this

algorithm gave better results in terms of dynamically joining at one point, results and working speed.

Shi et al. [21] developed a new algorithm using the particle swarm algorithm with ACO to solve the path planning problem. The particle swarm algorithm was applied to use the parameters of the ant colony algorithm in the most efficient way. They prepared a simulation program in MATLAB and applied the parameter values that they thought were the best in different areas with obstacles and showed the efficiency of the results.

Chen et al. [22] proposed a ACO that finds the optimal path at an appropriate time on an intermittent-sparse graph. They considered this algorithm to plan the path of real drones. By applying the algorithm as a simulation in the VC++ language, they produced quality solutions by minimizing the risks in test problems with various complexity, and thus they showed that the algorithm was efficient.

Vien et al. [23] used Ant-Q algorithm for obstacle avoidance path planning for mobile robots in their study. In their work, they investigated the reinforcement sensitivity of the update and applied them in the simulation program they created experimentally. As a result of this application, they compared the results they obtained with the genetic algorithm and the classical ant colony algorithm, and it was seen that the algorithm provided a better convergence.

Lee et al. [24] developed a new ACO for path planning using the potential fields technique. With the simulation program they made in MATLAB, they tried to reach optimal solutions by manipulating the control parameters in different environments and geometric shapes and applying them to complex maps. They compared the results with the results of a classical ant colony algorithm they created and saw that the new algorithm they created gave better results.

Yu et al. [25] developed a ACO for path planning of mobile robots in dynamic environments and simulated this algorithm in C++. Accordingly, they carried out path planning by moving a robot through different dynamic environments they had created. According to the results, they stated that this algorithm gives very good results for path planning of the robot in dynamic environments.

Chen and Yuan [26] presented an unusual ACO for the path planning problem. In their study, they developed a method by manipulating the pheromone limitation and pheromone evaporation coefficients. In this way, they tried to reveal the stagnation of the algorithm and the global search ability of the ant colony algorithm in the application process. With the created

simulation program, grid-shaped maps were created and this algorithm was applied. The simulation results showed that the algorithm is efficient and fast.

Viet et al. [27] proposed a multi-ant colony system formed by the use of several cooperative ant colonies that found good results with good information exchange in an obstacle avoidance and path planning problem. With the simulation program they created in MATLAB, they showed the behavior of the multi ant colony formed with different information between the colonies. They showed that the algorithm is efficient by comparing different behaviors with this program.

Gao et al. [28] proposed a new ant colony algorithm for global path planning of mobile robots to overcome some of the flaws of path planning problems. Mutation and crossover operators of the genetic algorithm are used to improve the quality of the ant colony algorithm and the heuristic probability function is added to the initial population formation process. This algorithm has been applied in different environments as a simulation in the VC++ language, and as a result of the tests, it has been seen that it has a higher capacity than the classical ant colony algorithm and improves the ant colony algorithm.

Zhang et al. [29] used the ant colony algorithm for global path planning of a rescue robot. In this study, a function was proposed to push the robot to the target in order to improve the search quality of the ant colony algorithm in complex and dynamic environments. The effectiveness of this algorithm has been tested on the RoboCup Rescue simulation system.

Lee and Jung Kim [30] developed a new ACO to solve some path planning problems. They have four different aspects from their previous work by Lee et al. [31]. First, in this algorithm, potential field is used instead of distance for the ant movement rule existing in the ant colony algorithm, and this potential field is used to create the regional pheromone field. Second, when updating the pheromone, the paths were classified by value. Third, certain control parameters such as α, β were used with modifications. Finally, they did not use the classical ant colony algorithm to reduce the time required to solve the problem. According to the results of the algorithm they applied in different environments as a simulation in MATLAB, they saw that the optimal path for physical robots was found faster than other algorithms.

Garcia et al. [32], developed a new algorithm using ACO together with a fuzzy logic system for the path planning of autonomous mobile robots avoiding static and dynamic obstacles, and this method is named SACOdm. Here d is for distances and m is for memory. With this

algorithm, different features have been added to the decision-making process and around 10 case studies have been made. The algorithm has two different ways of working, the first is for virtual environments and the other is created by establishing wireless communication. According to their studies, they have shown that this algorithm gives very good results for robots in static and dynamic environments.

Wang and Xiong [33] proposed an ACO for autonomous underwater vehicles routing. In their proposed study, they created the work space using the visibility graph technique and used the ant colony algorithm to find optimal paths on this space. By adding and applying the algorithm as a simulation, they obtained a shorter, safer and smoother path of the underwater vehicles.

Zotos [34], in his study, used ACO for humane robots to plan paths without hitting obstacles on a grid map in the form of a chessboard. In this study, he tested the performance of the algorithm by giving α and β values between 0-4, simulating and making the results graphically using the WeBost program.

Sariff and Buniyamin [35] proposed an ACO for path planning for mobile robots in global static environments. The heuristic equation of the transition rules in their proposed algorithm was to further optimize the ant colony for the robot path planning solution that finds the optimal path. This algorithm was applied on a global static map covering possible free space points. The algorithm was written in MATLAB, and the results were evaluated in terms of time and lap performance. The accuracy of this algorithm was evaluated by comparing it with another study made with a genetic algorithm.

Yang et al. [36] proposed a new algorithm for path planning of mobile robots. This algorithm was for real-time globally optimal path planning of mobile robots. In the algorithm, Maklink graph technique was used to create the space in which the mobile robot moves, and the ant colony algorithm was employed to find the optimal path between the starting point and the end point in this created space graph. The algorithm was adapted by simulation and applied in necessary environments, and the results were compared with the Dijkstra algorithm. According to the results, it has been seen that the proposed algorithm is more effective and quickly finds shorter paths than the original algorithm.

Luo and Wu [37] proposed an algorithm for path planning of a mobile robot in an unknown environment using the ant colony algorithm and potential fields technique, and they named it

ACOPF. In this algorithm, potential fields technique was used for obstacle avoidance and robot movement. In addition, the ant colony algorithm was also used to find the optimal path between the starting point and the ending point. With the simulation program written in C++, they have seen that this algorithm gives more efficient and effective results than the classical ant colony algorithm.

Wu et al. [38], in their study, proposed a bidirectional ACO to eliminate certain difficulties related to path planning and also to find the optimal path. This pair of ant colonies was run in different strategies and searching two paths at the same time made this algorithm global. Then, with these strategies, a faster and more efficient algorithm was obtained by increasing the positive feedback, and finally, this algorithm locally optimizes each path. The experimental results showed that the algorithm found the optimal paths in a short time after numerous studies in which it was applied simply.

Shaogang and Ming [39] proposed a method based on the basic ant colony algorithm for path planning of an inspection robot moving between several checkpoints. In the proposed algorithm, they realized the path planning problem on the map they created by applying the grid method and the visibility graph technique. Then, they used the ant colony algorithm to optimize the control mechanisms on this map. The algorithm was simulated in MATLAB, and they showed that the results were effective in solving such problems.

Zhao et al. [40] used ACO for path planning of mobile robots. In this study, firstly, two fuzzy controllers were designed to make efficient use of α, β, ρ values. Then a dynamic search window for ants was created and finally a new evaluation criterion was proposed to distinguish where the routes were perfect or not. The results obtained by adapting the algorithm in the simulation program have shown that this algorithm can quickly do the routing job in 3-dimensional complex environments.

Lee et al. [41] proposed a new ant algorithm called heterogeneous ant colony optimization for the solution of the global path planning problem and named it HACO. They proposed this algorithm to further improve the ant algorithm they had developed before. This algorithm they have developed is different from the classical ant colony algorithm in three aspects. First, they changed the transition probability function and the pheromone update rule. Second, they proposed a new path crossing within the pheromone rule. Finally, they used heterogeneous ants in the ant colony algorithm. They demonstrated the efficiency and accuracy of this algorithm in different environments with the simulation program they created in MATLAB.

Brand et al. [42], in their study, used ACO to find the shortest path of mobile robots without collisions within a gyre map. In their study, they found the results by making 2 different updates, local and global pheromone update, in terms of comparing the results, and it was aimed that the robot automatically goes to its original position and searches for the optimum path again when an obstacle is added to the environment. They wrote their studies in Python and conducted simulations in different dynamic environments with various obstacles, and the simulation results showed that the algorithm gave very successful and good results.

3. TRAJECTORY TRACKING PROBLEM FOR MOBILE ROBOTS

3.1 Introduction

From the day they were discovered, mobile robots have been in great development when both their features and usage areas are examined. While these usage areas of mobile robots were limited in the past, today they have started to be used in health services, military and security applications, energy resources, cleaning industry, agricultural services, as well as activities such as assembly, packaging, storage, transportation, especially in the industrial field. With its use in these wide areas, many problems have emerged over time for mobile robots, and although many solutions have been produced for them, the improvement of solutions is still continuing today. One of the most important research topics on mobile robots is the path planning problem.

As a general definition, path planning problem can be defined as a problem type that aims to find the optimal length of path from a desired point to another targeted point in a certain area without hitting obstacles in the area.

Trajectory Tracking Problem for Mobile Robots is the transportation of a robot from a starting point to a target point in a short time, under certain constraints, in line with a specific purpose or purposes. If trajectory planning is for a single purpose, it is expressed as Single-Objective Trajectory Tracking, and if it is for more than one purpose, it is expressed as Multi-Objective Trajectory Tracking. For example, it is considered as SOTTMR if it is desired for the robot to reach the target point by the shortest path from a starting point or without hitting obstacles in a static environment. On the other hand, if it is desired to reach the target both in the shortest way and safely, it is considered as MOTTMR.

As it can be understood from the above definition of the path planning problem, some criteria must be met while performing the routing process. The most important of these are that the planned path shouldn’t collide with obstacles, while it is smooth and the shortest route as well. A simple example of planning a route within an area is shown in Figure 3.1. By giving different alternative routes and evaluating them, we can more easily see whether the path planning is suitable for the criteria.

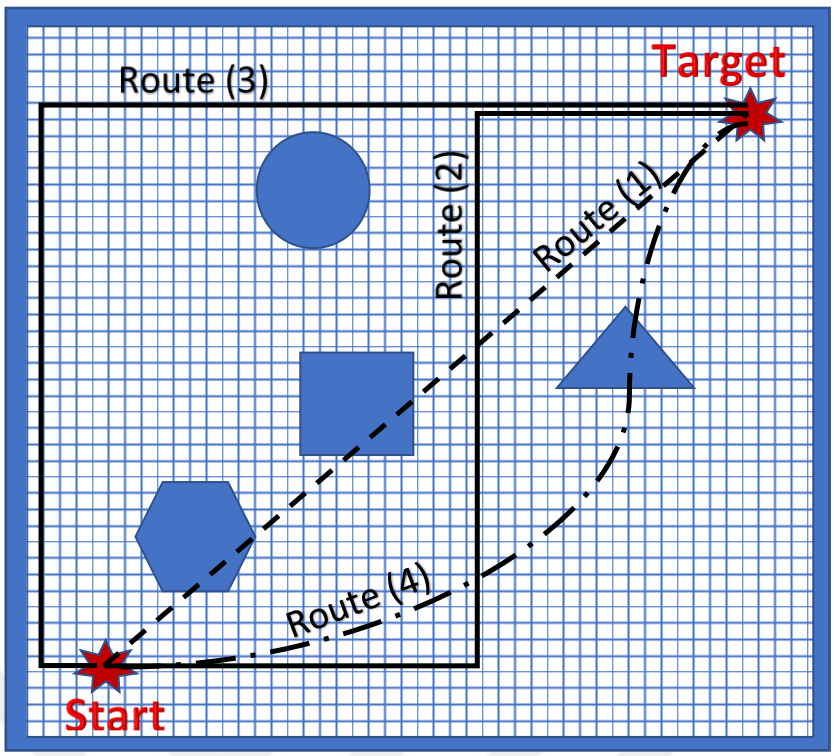

Figure 3.1- Criteria For A Simple Path Planning Example

Looking at Figure 3.1; It is seen that although the route (1) is the shortest path from the starting point to the target but it cannot be a valid acceptable one as it is not collision free. Route (2) is avoiding colliding with obstacles while it is a suitable pathway in terms of optimum route length. It is inferred that the route (3) is a safer trajectory regarding the distance from the obstacles comparing with the route (2) keeping further distance with the obstacles in order to avoid colliding them but it is a long way. Route (4) has optimal length and smoothness but it is not a collision free rout again. The point to be noted here is that an acceptable path should be the shortest possible while avoiding collision. As it can be inferred from this figure, for route (2) besides being collision free, the optimal path criterion is also provided for the complete solution of the path planning problem. As a result, paths 2 and 3 can be picked as possible solutions but then due to its fitness and length we pick route (2) as the optimal solution for this path planning problem.

Looking at the steps above, it is seen that providing only one criterion is not enough for the solution of the trajectory tracking problem. All the criteria determined for the solution of the trajectory tracking problem must be met at the same time. Apart from these basic criteria, in some path planning problems, different criteria (time, energy, etc.) can be added to the routing problems. When all these criteria are met, it is seen that the trajectory tracking problem will give the most accurate result [43].

Basic path planning problems have been studied frequently when the environment is known in general. From this point of view, we can classify the mobile robot routing with 2 main items while planning the path regarding the environment in which the robot will move besides the obstacles existing in the environment; and each of these items can be separated into two subcategories as follows:

(1) Designing the path of the robot in the static environment, which only includes static obstacles in the road map.

(2) Designing the path of the robot in a dynamic environment that includes static and dynamic obstacles.

Figure 3.2 shows the main approaches in mobile robot routing. Based on how well the robot recognizes its environment, each of these scenarios is split into two sections:

(a) Designing the robot's path in a previously defined and well-known environment with definite positions and sizes of the obstacles will enable the robot to travel with more precision and awareness of its surroundings. Since the whole workspace is known and predetermined, the robot's trajectory would result in the ultimate best outcomes.

(b) Designing the path of the robot in a semi-known or unknown environment. In this design, the robot tries to obtain local information about the position, size and format of obstacles using sensors. Therefore, it uses this information to improve its local route design.

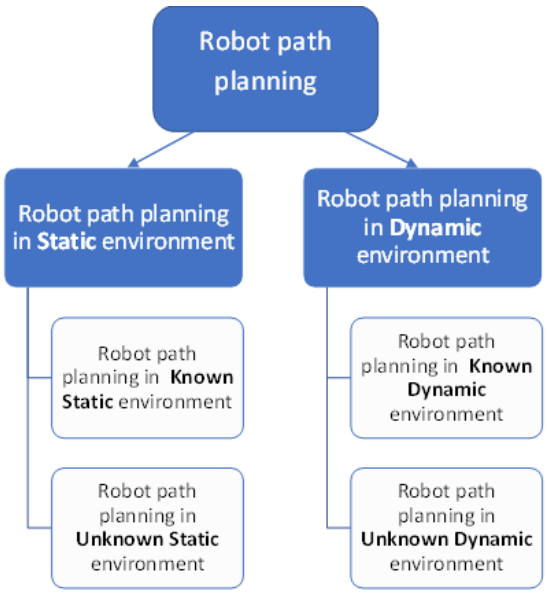

Figure 3.2- Main Approaches in mobile robot's routing

Trajectory tracking in static environments usually give excellent results thanks to being under certain conditions regarding the positions of the obstacles [44]. In practice, however, a problem is often used where information about the current environment is not complete. In most

cases, it is unrealistic to create a detailed map with all the identified obstacles. Especially in outdoor applications, for example, it can obtain the necessary coordinate system with GPS application. However, the information obtained about the environment can be very limited. Under these conditions, too much uncertainty arises for very detailed and logical planning. The most natural way to update plans is to select a path in existing information as a starting point and then moving along that path for a short period of time while new information is being collected. The route is then replanned according to newly obtained information. This method is frequently used in the literature for path planning in unknown areas. If working on a partial map, the aim is to provide more direct oriented behavior towards the desired target. In the algorithms of Lumelsky and Stepanov [45], the robot moves directly to the target by acting as if there are no obstacles on the way to the desired target. When it encounters an obstacle, the robot moves around the obstacle until it finds the closest target to a point on the obstacle, and as soon as it finds the target, it starts to move directly to that target.

3.2Global And Local Trajectory Tracking

There are two different approaches to path planning: global path planning and local path planning. It is assumed to have an excellent knowledge of the environment when making global path planning. With this excellent information, it is ensured that the positions, shape changes, dimensions and coordinates of the obstacles in the work space are known exactly. Knowing the environmental information, the global path planning algorithm works offline and plans the path between the starting point and the destination point without waiting for the mobile robot or similar vehicles to start moving. In other words, in global path planning, both the purpose is defined and the static space of the obstacles is made available. By using this created route, a reference and guide is provided for path planning before starting the movement [46].

In the local trajectory tracking approach, the necessary information for path planning is made in real time with the source of some local information. Local path planning has no information about the environment other than the starting position and target position. Since the process is done online in local planning, trajectory tracking is done during the pathway. So, there is a reflexive process. Therefore, planning is done by reacting to the obstacles encountered while planning and avoiding these obstacles.

The main differences between global and local trajectory tracking approaches are shown in Table 3.1 with their basic elements. When we compare these two approaches, it is seen that global path planning results in slower and more processes than local path planning. The reason

for this is that local path planning is a reactive process and dynamic. It is seen that the global path planning approach works at a higher level in larger obstacles than the local in terms of obstacles. However, since the environment is fully known in global path planning, it has been seen that the global planner improves the quality of the route to a high degree.

Table 3.1- Comparison of Global and Local Trajectory Tracking

GLOBAL TRAJECTORY TRACKING	LOCAL TRAJECTORY TRACKING
➢ Accumulated and a priori path-based information is used. ➢ Plans to get to the goal efficiently and appropriately. ➢ There is a careful and slow process. ➢ Makes plans for long distances and long periods. ➢ It is used for planning simple model vehicles. ➢ It is used in terrain with larger obstacles.	➢ Urgent, reactive sensor-based information is used. ➢ Plans the fastest and most possible trajectory. ➢ There is a reflexive and fast process. ➢ It makes planning for the closest environment and the shortest periods. ➢ It is used for planning complex model vehicles. ➢ It is used in areas with small obstacles.

3.3Routing Space

Path planning problems can be expressed as operationally very complex problems. The reason for this is that there are many route options in the area where routing is made and it is tried to decide to pick a route among these options. For this reason, path planning problems are classified as NP-Hard or P-Space-Hard problems since they increase exponentially depending on the configuration space dimensions and are problems that can be solved by trying all possible solutions.

According to the general definition of path planning problems in the literature, the basic idea in path planning is that mobile robots can automatically move between desired points without the need for humans. While the robot is making path planning automatically, there may be known or unknown obstacles in its area and the robot should be able to move and maneuver around the obstacles without hitting these obstacles. Starting from here, with general definitions, the work space (W-Space) is defined as a space in which the robot carries out its operations and contains obstacles and targets, while the configuration space (C-space) represents all possible configurations of the robot. In addition, it is defined as free space (C-free) in physically suitable regions that allow the robot to maneuver outside of obstacles.

The configuration space can be thought of as a kind of reduced working space of the robot. The work space is explained on Figure 3.3 and the configuration space as a reduced version is explained on Figure 3.4.

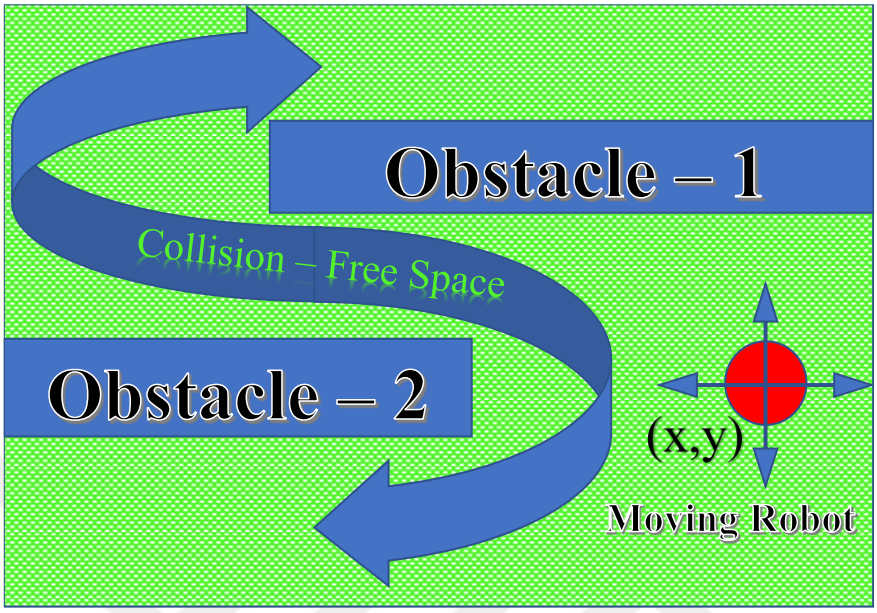

Figure 3.3- Two-Dimensional Workspace

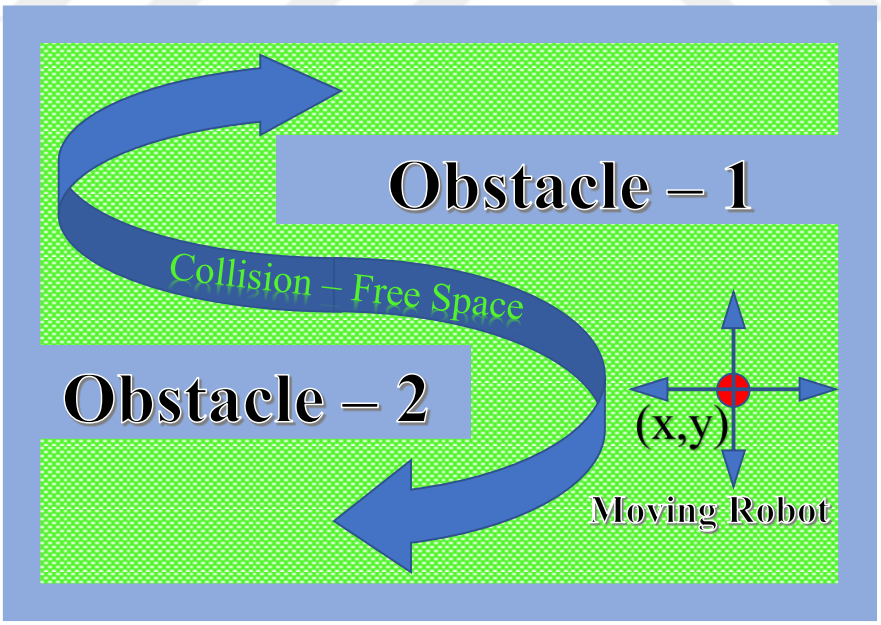

Figure 3.4- Two-Dimensional Configuration Space

Figure 3.3 is the workspace for a simple disk robot. The robot is a 2D round shape. That's why the configurations are (x,y). If the robot were not circular here, an independent motion orientational stage would also have to be specified. In Figure 3.4, the configuration

space given for a simple robot is again. The configuration space is similar to the working space given for the robot in Figure 3.3. The robot is specified as a point in the configuration space. So, each point resembles each configuration of the robot rather than an actual point in space. Let's choose a point on the robot as a reference point as shown in the figure to see how it is formed. It is seen that the curves are radiated outward by the point assumed to be the robot running along the boundaries of the work area and the surfaces of the obstacles. Since this robot comes as a point, the obstacles have been enlarged to meet this situation, and this is an issue that should be taken into account during the inspection of the contact of the point-shaped robot with the obstacles.

The configuring environment's dimensions are determined by the stage at which the robot is willing to roam on its own. That is, if the robot has n independent motion stages, the configuration space is usually n-dimensional. A good configuration space should have a small number of dimensions in order for the planning algorithms to work quickly. For a complicated model, the settings domain expands quickly as independent motion stages are included, and the sheer number of dimensions makes design challenging. As Latombe et al. [47] in 1991 suggested, the planning problem becomes increasingly difficult over time due to how many dimensions there are in the configuring environment. As a result, the size of the configuration space should be taken into account when planning. For this reason, researchers generally used some simplifications in the problems in order for the path planner to reach the target faster. For example, The 2-dimensional coordinate system such as only (x,y) becomes the only configuration option for some path planning issues.

For path planning, either discrete or continuous solution environment may be employed. The universe is modeled as a continuous range of an unending amount of potential situations in scenarios with perpetual spatial structures. Planning is typically more challenging in a continuous situation environment. Instead of using it for overall trajectory tracking, the perpetual spatial structure is employed for responsive collision avoidance. When a discontinuous state space is employed, the searching environment is split into an endless variety of distinct situations in which the robots and similar autonomous vehicles are located. Discrete and continuous state spaces can be applied at many levels. A perpetual domain characterized by arithmetic operations, for instance, can be subject to potential field techniques of trajectory tracking, as well as the operations might have been separated into a grid map that the route designer peruses [48].

3.4 Path Planning Methods and Techniques

Studies on the solution of the path planning problem for mobile robots is a problem that has been going on for many years and continues to be studied more and more every day. A significant number of researchers have made an intense effort to develop effective and efficient techniques for solving the path planning problem. Researchers have considered two general methods while developing these techniques. The first is about reaching a path planning solution using previously known global environment or obstacle information and robot characteristics, and the second is about motion planning using local information and robot characteristics.

Many techniques have been developed to solve the path planning problem. Path planning techniques studies are grouped under 3 main headings in terms of their performance and solutions, as seen in Figure 3.5 below.

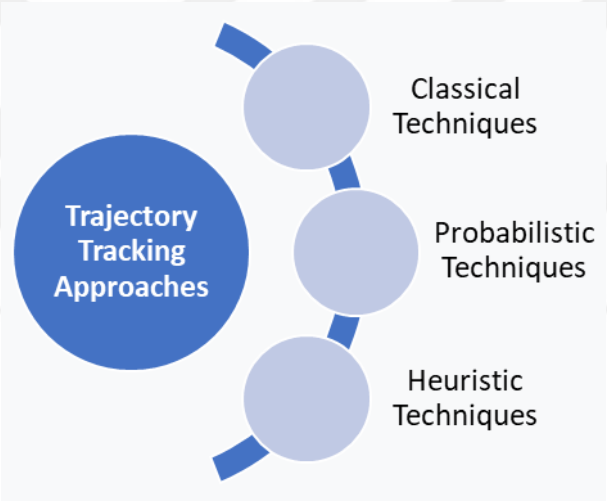

Figure 3.5- Trajectory Tracking Approaches

3.4.1 Classical Path Planning Techniques

Classical path planning techniques can be described as the oldest techniques of the studies for the solution of the path planning problem for mobile robots. Most of the path planning problems can be solved with classical path planning techniques, and therefore, classical techniques have been used in studies conducted for many years. Classical path planning techniques can be expressed in 4 main techniques as shown in Figure 3.6.

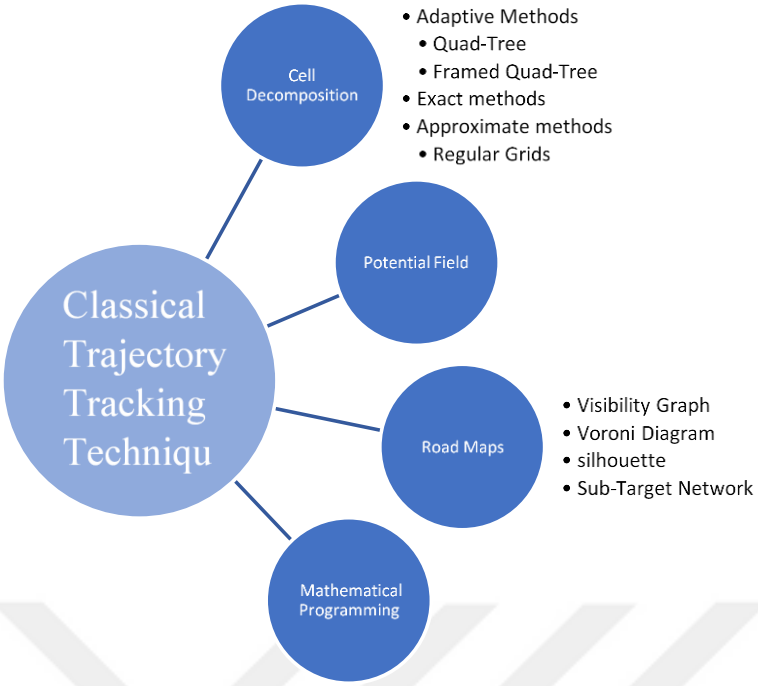

Figure 3.6- Classical Path Planning Techniques

3.4.1.1 Cell Decomposition Technique

Techniques for cell decomposition are among the most investigated and mostly used technique in robotic applications. With this technique, the free configuration space is decomposed into a simple series of cells and cells between very close connections are calculated. A path through which the robot will move without collision between the starting point and the target is first found by 2 cells covering the defined start and target point and then they are connected to a successive series of cells. These techniques use connector graphics for accessibility from one cell to adjacent cells. The robot then tries to go to the target with a successive series of cells, thanks to the connecting graphics and the paths produced accordingly. The value of rendering a cell may vary, and the planner should use good methods to determine the ideal route.

Cell Decomposition techniques are generally employed to create the structural cavity, but in some cases, they can be used for the configuration of the cavity. Cell Decomposition techniques are grouped under three main headings as approximate cell decomposition, adaptive cell decomposition and exact cell decomposition [48].

In approximate cell decomposition, a regular grid spread over the area to be planned is created. Approximate cell separation was first presented by Brooks and Perez [49] in 1983, and was later developed by Zhu and Latombe [50] in 1989. In this technique, every cell's midpoint

turns into a node inside the search network that can be explored to identify a pathway. The reason why this technique is described as approximate cell separation is that the boundaries of physical objects do not match the previously defined cell boundaries. Since grid cells have a predefined shape and size, this technique is extremely easy to apply to world space compared to other cell separation techniques, and they are flexible techniques according to needs. The sizes of the cells can be adapted to reduce computation time, therefore decreasing the size of the cells to search through and get more detail. These cells therefore become attractive to mobile robots. Also, in separation, cells have some values to form lines diagonally. The map only includes the parts with barriers and empty area, which are quite similar to locked robotics if this value is binary. However, despite the ease of implementation of this technique, there are several problems with this technique. For example, the most important of these is the numerical trend. When using approximate cell decomposition, one of many different ways is used to divide the configuration space [48].

Reduced cell density in open areas is achieved using the adaptive cell decomposition technique in order to eliminate the numerical trend of approximate cell decomposition. This technique is an application that can be used efficiently for large natural terrains with large areas with the ability to create a cross line for the robot's trajectory. Adaptive cell decomposition is based on the fact that regular cell sorting makes all the information in free space redundant. The cells are still preserved in their usual form, even though some changes have been made to the space to use the cells more efficiently and provide more detail. Thus, computation time and memory usage are reduced. The Quad-Tree methodology is the most popular sort of adaptive decomposition method. The Quad-Tree technique was first proposed by Samet [51] in 1988 and Noborio et al [52] in 1990. The first step in this method is to cover the whole modelling environment with a big cell. The grid cell is split into four equally sized sub-regions if it is only partly filled, and the obtained sub-regions are then reintroduced toward the trajectory development area. It then gets separated once more. Until all of the cells are completely filled or unfilled, this cycle is repeated. In this case, grid cells of different sizes and densities are formed on the resulting map, but the cell borders are adjacent to very close, similar obstacle borders. For continuously changing and unknown environments where the robot is gathering updated information and revising maps in response to new barriers, adaptive cell sorting presents several challenges. In this case, the map needs to be completely corrected for the entire data structure. There are some difficulties in using maps with continuous values, namely natural terrains, and Quad-Tree technique at the same time. Because it is very difficult to subdivide the

world in occupied or free regions. In addition, Quad-Tree also has some difficulties in generating nearly ideal pathways; these paths frequently have jagged edges. Therefore, as a better solution, Chen et al. [53] proposed the Framed Quad-Tree technique in 1995. The difference between a Quad-Tree and a Framed Quad-Tree can be seen in Figure 3.7 and Figure 3.8. However, in highly rough and mixed environments, the framed quadtree technique may yield less efficient results than regular grids.

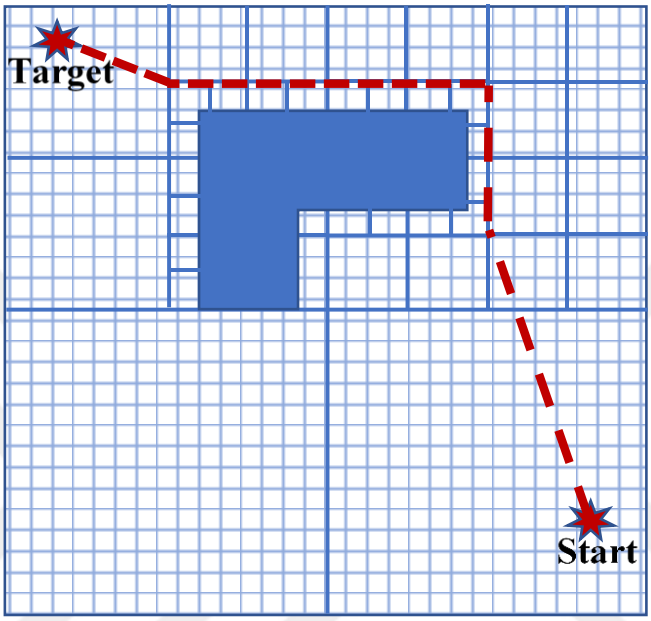

Figure 3.7- Quad-Tree Technique for Path Planning

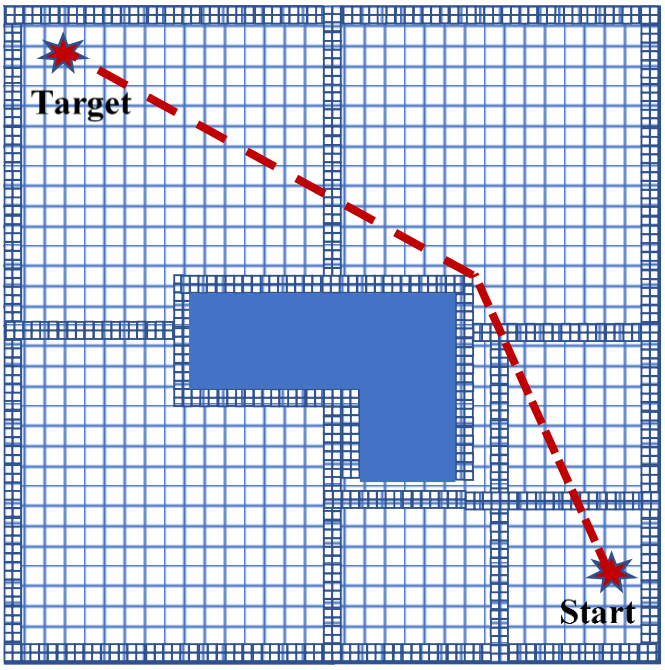

Figure 3.8- Framed Quad-Tree Technique for Path Planning

Full cell decomposition, on the other hand, deals with some solutions for path planning problems with regular grids in a different way. The cells do not have a predefined size or shape, but the cells are determined by the shape of the obstacles within the world map, regarding the world map in Figure 3.9. The cell boundaries exactly match the boundaries in the planning space, and the union of cells is exactly the free space of the world. This makes full cell decomposition complete and they will constantly try to find a way. But these cells will not result in the optimal pathway. This technique is very difficult to apply in environments with irregular and poorly characterized obstacles. Case studies with this technique were made by Schwartz and Sharir [54] in 1983 and later by Avnaim et al. [55] in 1988. However, the application of whole cell sorting technique for path planning of mobile robots is seen in more detail in the study by Sleumer and Gurman [56] in 1999.

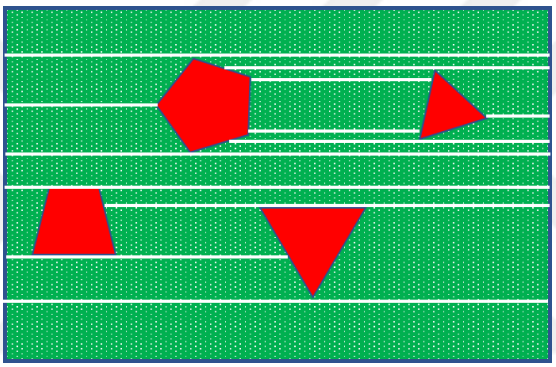

Figure 3.9- Example of Full-Cell Decomposition for Path Planning

3.4.1.2 Potential Fields Technique

The potential fields technique is completely different from other path planning techniques and has been widely used in the past. The potential field technique uses a mathematical function over the entire area the robot will move. This technique treats the robot as a point under the influence of fields produced by targets, and obstacles within the world as an electron in an electric field. Therefore, targets create attractive forces to attract the robot to the target, while obstacles close to the robot create repulsive forces [48]. These attractive and repulsive forces of the potential fields technique are found by general formulas as seen in Equation 3.1 and Equation 3.2.

$$F_{atr} = -k_{atr}(L_r - L_t) \tag{3.1}$$

$$F_{rep} = \begin{cases} k_{rep}\left(\frac{1}{\rho_d} - \frac{1}{\rho_o}\right)\frac{1}{\rho^2}\frac{\partial \rho}{\partial L_r} & if\ \rho_d \leq \rho_o \\ 0 & Otherwise \end{cases} \tag{3.2}$$

Here; L_r is the coordinate of the robot's position, L_t is the coordinate of the target position, ρ_d is the distance from the robot to the obstacle, ρ_o is the distance at which the obstacle is effective, k_{atr} and k_{rep} are positive coefficients as scaling factors.

The potential field technique is actually a technique originally discovered for real-time obstacle avoidance rather than path planning. The idea of using potential functions to avoid obstacles was first introduced by Khatib et al. [57] in 1978 and Hogan [58] in 1985 for power control. However, different theories have been developed later by Miyazaki and Arimoto [59], Pavlov and Voronin [60], and Koditschek [61], and Rimon and Koditschek [62] to make it more efficient. This technique makes use of information they receive dynamically rather than prior knowledge. These methods are relatively easy to implement and work numerically effectively and efficiently. But the biggest problem of the potential field technique is that it has local minimum values. In other words, the robot behaves as if it has reached the target by sticking to a local minimum value while going from the start to the target. Looking at the figures below, an area has been created for the first obstacle in Figure 3.10.a and for the second obstacle in Figure 3.10.b. However, in Figure 3.11.a an area for the attractive forces to the target has been created and in Figure 3.11.b an area that is the sum of all these areas has been created and the total area would be able to manage the robot with low potential field levels, but as it can be seen, it is necessary to pay attention to the minimum local values. Especially because of this problem of potential field, most of the studies on potential fields technique are to eliminate this problem.

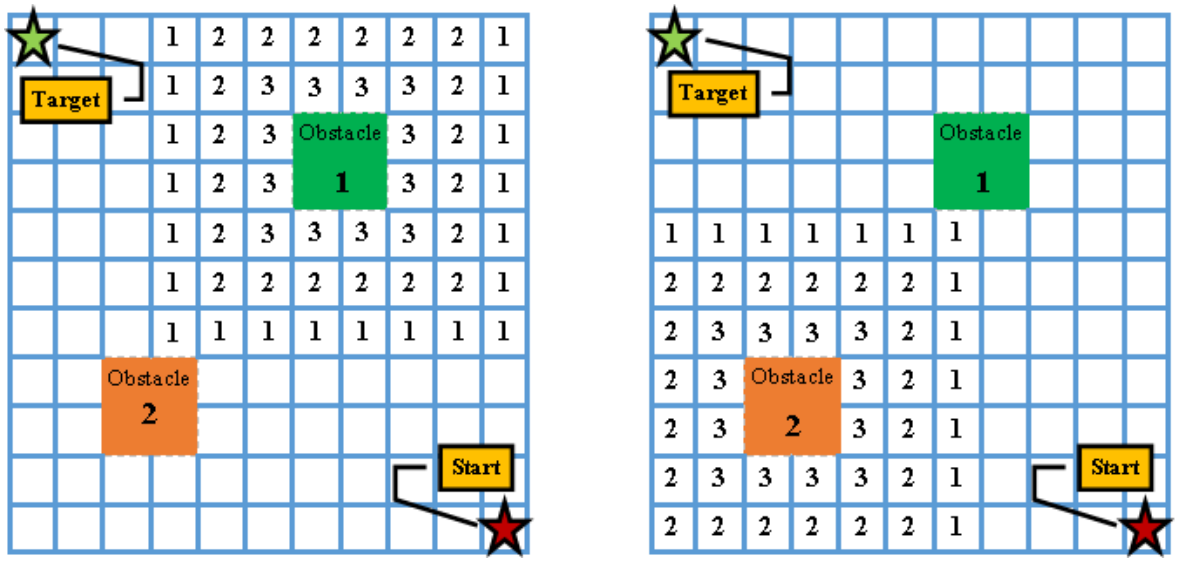

Figure 3.10- a: Potential field for the first obstacle b: Potential field for the second obstacle

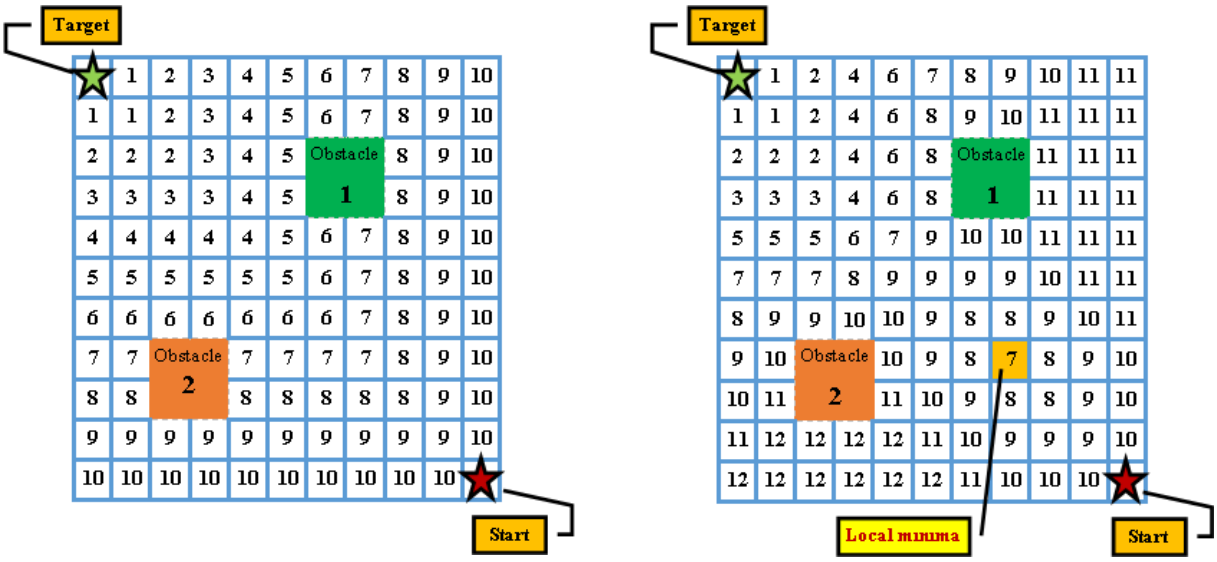

Figure 3.11- a: Potential field generated for the target. b: Total potential field

3.4.1.3 Mathematical Programming Technique

Mathematical programming technique is not used much for solving path planning problems. The mathematical programming technique is used where there is a need for obstacle avoidance with a set of inequalities dependent on configuration parameters. With this technique, path planning for mobile robots is formulated as a mathematical optimization problem that finds a curve between the initial and target configurations that minimizes a given numerical amount. Such an optimization is nonlinear and has inequality limits, and a numerical technique is used to find the optimal solution [63]. Maciejewski and Klein [64] developed a mathematical algorithm to avoid local barrier collisions for unnecessary manipulations in dynamical environments with a mathematical programming. However, Chen and Vidyasagar [65] have tried to solve path planning problems using this technique.

3.4.1.4 Roadmaps Technique

The roadmap technique is one of the most frequently used techniques for solving the path planning problem of mobile robots. Roadmaps can be defined as graphs that represent how to get from one point to another. Roadmap techniques reveal connections between the robot's free space, such as a set of one-dimensional curves. Roadmaps are used as a set of paths from which the planner will seek optimal solutions from the moment they are created. Nodes in the graph are usually auxiliary waypoints for the robot to make a successful move. The roadmap technique has a great advantage over cell separation techniques due to the number of nodes the planner has to search through to find a path. The set of these nodes does not consist of all of the configurations, but consists of a few selected specific parts. Roadmaps are often quick and often

easy to implement. That's why it has many uses. Due to its graphical structure, the path planning technique can be grouped under 4 subheadings: visibility graph, Voronoi diagram, silhouette and sub-target network.

The visibility graph technique is one of the roadmap techniques applied to the 2-dimensional configuration space, and many researchers, especially Nilsson [66], have used this technique. Visibility roadmap connects the nodes of these obstacles with the help of straight lines without passing through them, as seen in Figure 3.12. These straight lines represent the paths that the robot can go through, and with several search techniques, the desired optimal path can be found on the visibility roadmap. The visibility graph technique often seems to be inadequate because the paths identified are, as indicated in Figure 3.12, tangential with obstacles. Another problem is the defined shapes of the obstacles. The problem with the shape of these obstacles is a big problem especially for outdoor robots because the obstacles are almost always round or amorphous ones. As seen in Figure 3.12, the most optimal path of the roads connected by the corners of the obstacles is indicated by dashed lines.

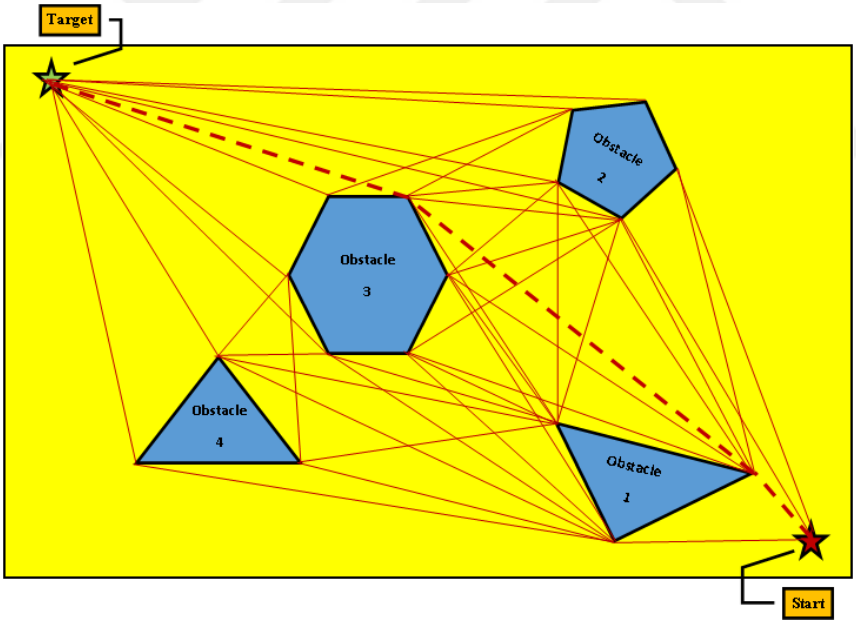

Figure 3.12- Example of Visibility Graph for Path Planning

The Voronoi diagram technique is one of the most popular techniques used to construct a roadmap from a configuration space. Voronoi diagram expresses the distances between independent set elements. Although the Voronoi diagram is named after Georgy Voronoi, the Voronoi diagram has been used in many studies much earlier. Voronoi diagram is also called Dirichlet, Thiessen or Wigner-Seithz diagram in the literature. This diagram can be created as the robot enters a new environment. The roadmap or Voronoi edges of routes are equidistant

from all points in the obstacle zone. An example Voronoi diagram is shown in Figure 3.13. Here, the robot moves between the starting point and the target point, which are indicated by dashed lines, over the points where the Voronoi edges meet. Compared to the visibility graph technique, it is a more suitable technique in terms of distance from obstacles.

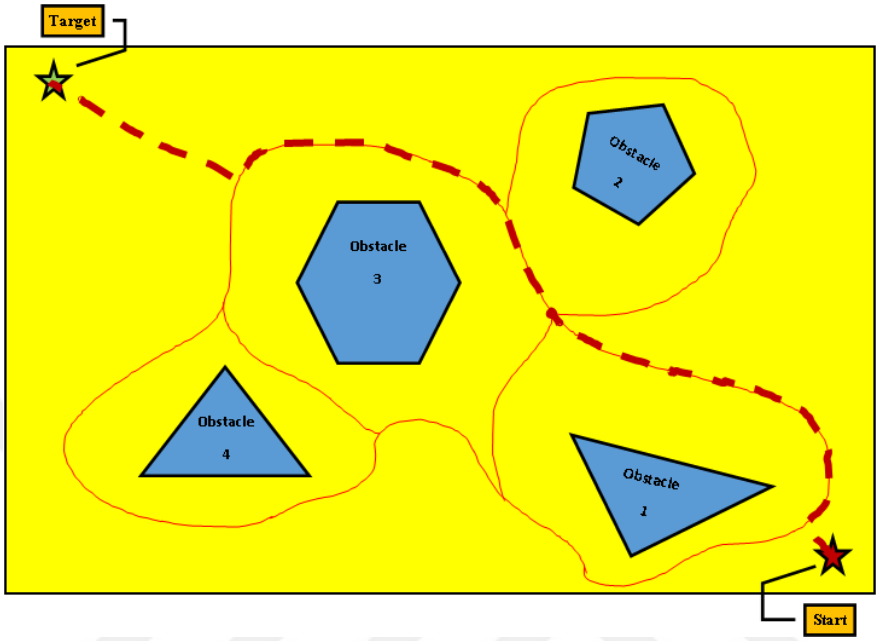

Figure 3.13- Example of Voronoi Diagram for Path Planning

The silhouette technique is formed by the production of the silhouettes of the work cell, and by connecting these silhouette curves with each other, the roadmap grows and forms. The silhouette technique was introduced by Canny [67] in 1987, by creating a roadmap of random or arbitrary dimensions. He designed an object in a smaller dimensional space and then created a sketch of projection curves called a silhouette. This is designed so that the silhouette curves are repeated until one-dimensional lines form into the lower dimensional space. The curves are then connected using the coupled curves where new silhouette curves occur or disappear. This technique was developed to find a way through a graph of one-dimensional curves of less complexity than an original space containing objects. This technique is mostly used in theoretical algorithms that analyze complexity rather than develop practical algorithms [63]. A path from the silhouette curves allows the robot to move along the boundaries of the obstacle and workspace as shown in Figure 3.14. A different version of the algorithm was developed by Canny and Lin [68] in 1993. In this study, they developed a new algorithm by using the potential areas technique together with the silhouette technique.

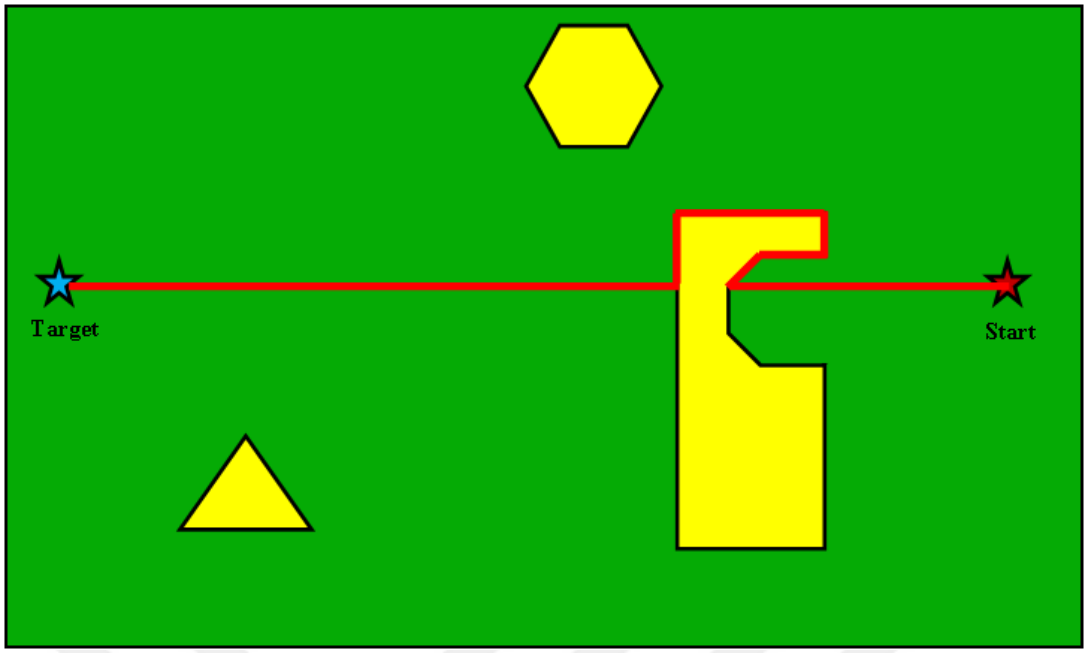

Figure 3.14- Example of a Silhouette Curve for Path Planning

The sub-target technique does not produce a clear and unambiguous description of the configuration barriers. Instead, a list of available configurations is generated from the initial configuration. As soon as the target configuration is reached, the path planning problem is solved. A local operator, assumed to be a fairly simple motion planning algorithm, decides the reachability from one configuration to another as if it is moving the robot in a straight line between configurations. Initially, this technique finds a candidate series (a succession) of in-between configurations that are assumed to be sub-targets and uses the local operator to move the robot through the sub-targets in sequence. A heuristic is usually used to search for sequence, but a sequence of random configurations can also be used. If the robot cannot reach the target configuration, the sub-targets reached are stored in the list and another candidate sequence is created between the target and any of the sub-targets reached. The local operator is reused to control the motion that can occur through cascade, and this process is repeated over and over. The most important advantage of this algorithm is its small memory requirement [63]. An example of path planning related to the sub-target technique is shown in Figure 3.15.

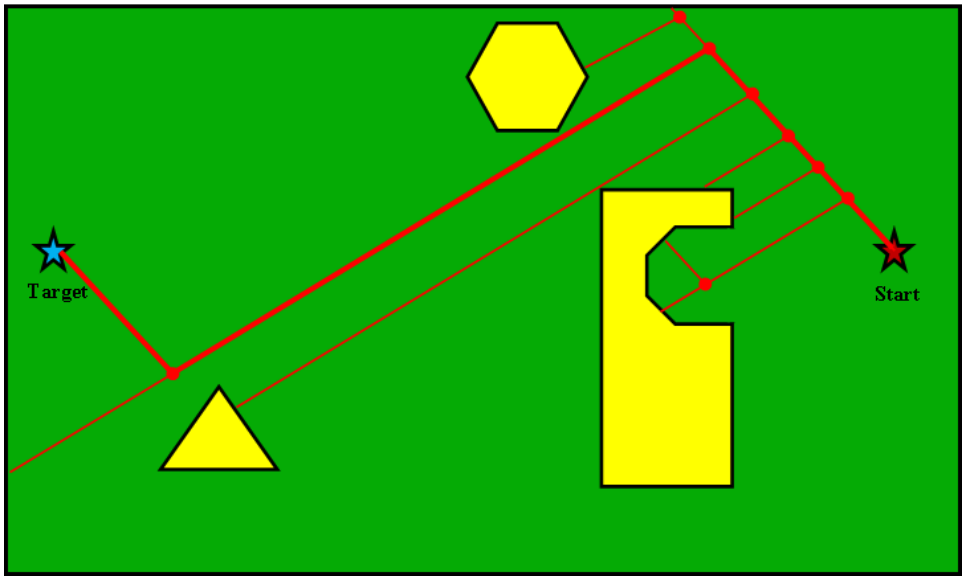

Figure 3.15- Example of Sub-Target Technique for Path Planning

3.4.2 Probabilistic Path Planning Techniques

Probabilistic path planning techniques consist of choosing a sample to represent the entire search space, in which the paths created in the search space have an equal probability of being selected in the first step. Then the sampled points in this space are connected with each other. A map is created for the solution of the routing problem by considering the obstacles in the connections between these points. Probabilistic path planning techniques are also included in the class of classical path planning techniques, but they are expressed as probability-based routing techniques in terms of their functions. For example; Probabilistic roadmaps and fast discovery of random connections, which will be explained below, fall into the category of roadmaps in the classical techniques mentioned above. However, probabilistic path planning techniques have been developed to improve the efficiency of classical techniques. Probabilistic techniques can be expressed in 4 main techniques as seen in Figure 3.16.

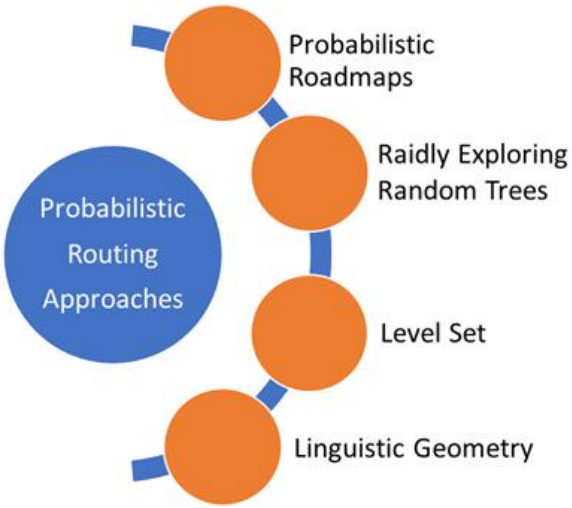

Figure 3.16- Probabilistic Path Planning Techniques

3.4.2.1 Probabilistic Roadmap Approach

Probabilistic roadmap technique can be generally defined as a variant of roadmap technique, as the name suggests. Nevertheless, it is among the probabilistic path planning techniques in terms of the stages of creating the map on which path planning will be made. Probabilistic roadmap technique has made great progress in solving the problem of path planning in large scaled spaces recently. This technique consists of randomly sampling a configuration space, and then the points therein are connected in a roadmap. The technique of probabilistic roadmaps has made significant progress in solving path planning because many other planning techniques, especially cell decomposition methods, solve the planning problem in the entire search space. In addition, the probabilistic roadmap technique tries to solve the path planning problem with a roadmap formed by a randomly selected subset of the search space. The probability-based roadmap technique relies on a relatively small number of points and milestones, and the paths are usually sufficient to achieve connectivity of free space. This assumption can also speed up the process for most solutions to planning problems. Probabilistic roadmap technique consists of two phases: the roadmap creation phase and the query phase. Points are randomly selected in the configuration space to create the map and added to a list of special points. If a point is connected to the obstacle zone, that point is cancelled. The mapping algorithm tries to connect this point to a subset of other existing configurations in the previously randomly selected list of special points. This binding work is done with the help of a straight line if there is no obstacle between two points. This random selection process and binding work is evaluated and repeated many times. A list of available connections between randomly selected points is generated. The list of these links creates the roadmap, showing a series of trajectory points around obstacles in the configuration space. In the query phase, if a path

planning is to be made between two configurations, the map created in the first step is used to search through the pathway point nodes to find the optimal path between the start and end configurations. Although it is very costly and expensive to create the map in the first stage, after creating this map, the search is very efficient. An example of a simple probability-based roadmap technique is shown in Figure 3.17, and the dashed line is determined according to randomly selected points. With all these explanations, the biggest problem in probabilistic path planning technique is that it is insufficient and inefficient for narrow-bounded spaces. Because the points that make the roadmap are chosen randomly, the chance of catching a randomly generated point in a narrow-bounded area is very low and the desired connections may not be established between some parts of the map. As a solution to this, it can be said that with more points, the points will cover more area, but the increase in the number of points makes the path planning problem more complex. To eliminate this problem, Boor et al. [69] conducted studies. Another problem in the technique of probability-based roadmaps arises when obstacles are removed or added from the map, and the whole roadmap has to be reconstructed in this case. Because roadmap creation is slow and cannot be done in real time. Regarding this problem, many studies have been carried out by researchers Hsu et al. [70], Leven and Hutchinson [71] to overcome dynamic barrier limitations.

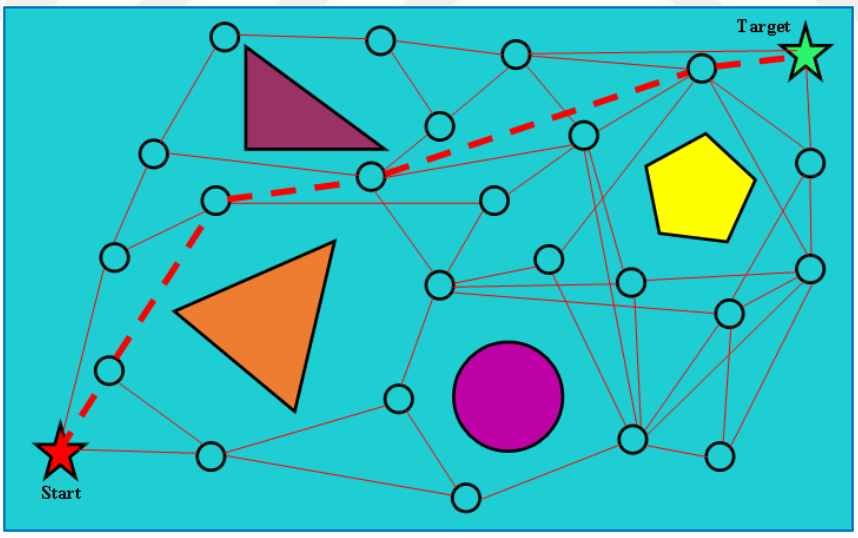

Figure 3.17- Example of a Probabilistic Roadmap for Path Planning

3.4.2.2 Rapidly Exploring Random Trees Technique

The rapidly exploring random trees technique can also be defined as a variation of the previous technique, probabilistic roadmaps, as it contains many features of it. In this technique, as in the probability-based maps technique, it is designed with less search and optional parameters. This allows us to achieve better performance and more consistent results. The

biggest advantage of this technique over the probabilistic roadmap method is that it can be directly applied to Kino dynamic routing. As with probability-based roadmaps, the solution requires many connections, whereas in this technique there is no need to make any connections between configuration pairs. In other words, it starts from a randomly chosen point and moves to the closest point to this point, then moves to another point close to that point and branches until it reaches the target, forming a road map. This branching continues with the expansion of the connected roads towards the areas that are not yet filled. Therefore, these techniques are based on tree-based planning. An example of this technique is shown in Figure 3.18. Studies on this technique have been carried out under the name of random Kino dynamic planning by Lavalle and Kuffner [72], Kuffner and Lavalle [73], Cheng et al. [74], and Frazzoli [75].

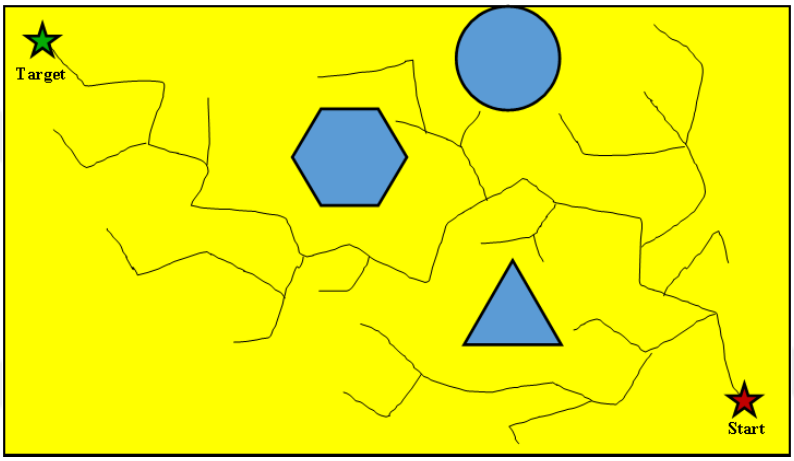

Figure 3.18- Example of Rapidly Exploring Random Trees for Path Planning

3.4.2.3 Level Sets Technique

The level sets technique is an efficient technique used to solve the Hamilton-Jacobi equations used to monitor the motion of propagating interfaces under complex velocity rules and to keep the topological changes under control, as proposed by Osher and Sethian [76] in 1988. In this technique, the goal is the next movement of interfaces under a speed range for computation and analysis. This quickness may depend on location, time, geometric shape of interfaces, and other physical factors. Especially in recent years, Hassouna et al. [77] have developed several algorithms for solving route planning problems using this technique. These techniques calculate optimal paths around obstacles for path planning within an area for mobile robots. The study by Sethian [78] on finding optimal routes with the help of this technique was carried out on a parking lot example. Here, a straight line is drawn as seen in Figure 3.19.a to go from one point to the place where our car is. Here, point A is where we are, while point B is where our car is. Here but can be thought of as creating a surface that extends in a circle from

point A to all directions. Here it is allowed to spread the surface at 1 speed in all directions, since it is not our burden to walk the directions over the others. This surface spread starts from point A and grows in the form of a circle and continues until it touches point B as can be seen in Figure 3.19. b. After this surface contacts point B, an optimal path is created by drawing backwards a line that is always perpendicular to the surface spread, starting from point B as is shown in Figure 3.19.c.

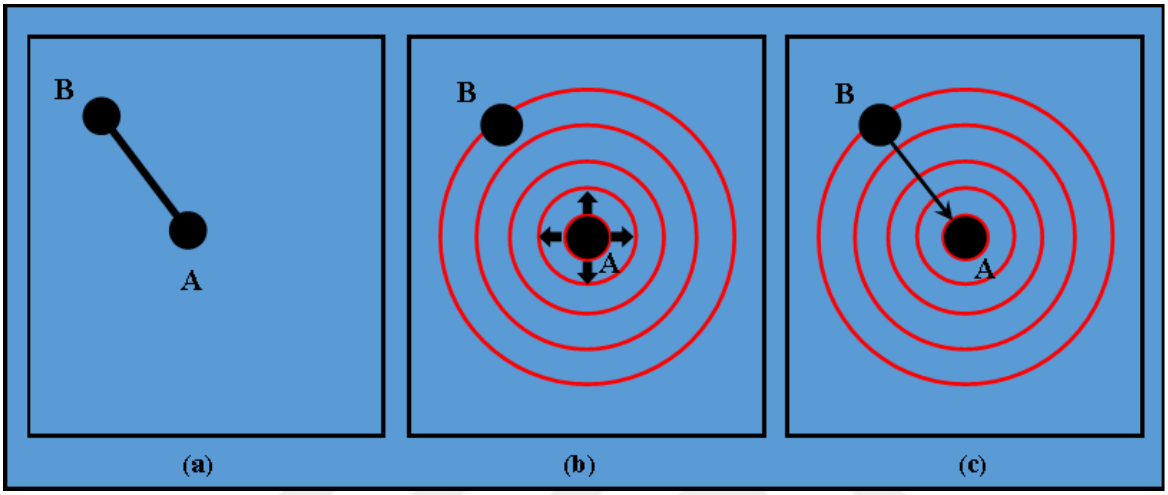

Figure 3.19- (a) Showing a Line Between Points A and B. (b) Surface Spreading Around Point A. (c) Tracking the Return to Find the Way from Point A to Point B **[78]**

In another problem, he has complicated the problem by assuming that some part of it is snowy and some other part is dry, and by putting a few cars in the parking lot that he thinks will be an obstacle while going to his own car. Again, as seen in Figure 3.20, there was a spread in all directions starting from point A, but the spreading surfaces differed in snowy and dry places (Figure 3.20.a). When considered logically, the spread here was different because it was difficult to walk on the snowy ground and was slow in terms of time (Figure 3.20.b). He defined cars having put here as obstacles to be places of infinite value, because he thought it was impossible for a car to pass over or would take forever. In this case, the surface will form sharp corners as seen in Figure 3.20.c. As a result, considering these corners, the return was followed for the solution of the surface spreading problem and the optimal way was found by means of lines.

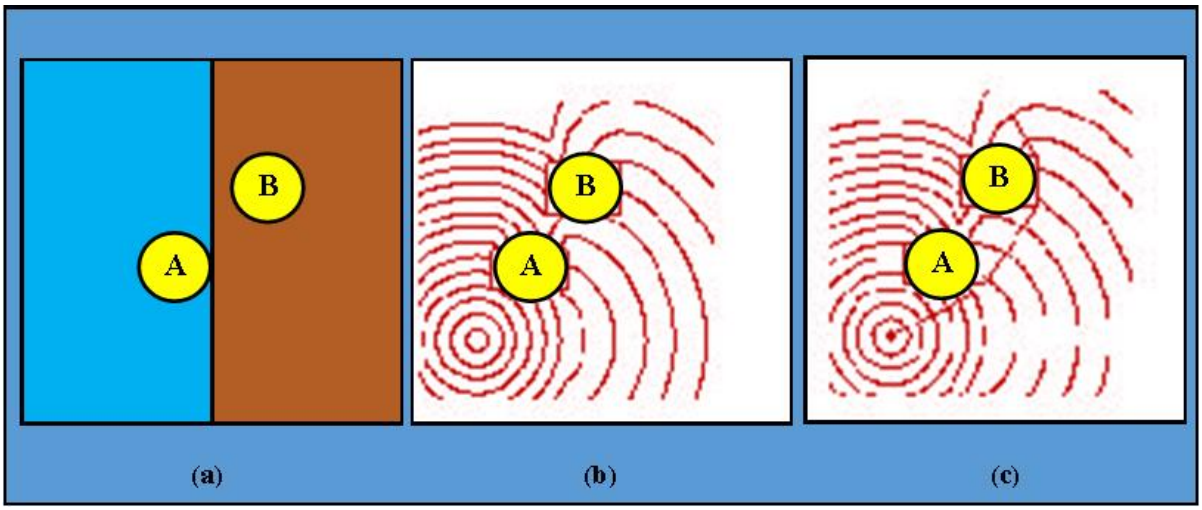

Figure 3.20- (a) Showing that Region A is Snowy and Region B is Dry.

(b) Surface Spreading Around Point A.

(c) Tracking the Return to Find the Way from Point A to Point B **[78]**

3.4.2.4 Linguistic Geometry Technique

Linguistic geometry was developed as a comprehensive technique for a particular class of complex systems. This technique is actually a game theorem and provides us with convenience for solving practical or complex games. From this point of view, it has become a viable technique in military decision mechanisms, robotics applications, software engineering, and traditional entertainment games. There are many studies on linguistic geometry, especially by Stilman [79]; and he applied them to complex systems, including path planning. The linguistic geometry technique includes syntactic tools that can be used to represent information about complex multi-agent systems as well as reasoning. This technique provides us with powerful tools to reduce searching in these complex problems by separating complex systems into a hierarchy of subsystems in a dynamic interaction. These tools give us a perspective for evaluating complexity and quality of solutions. The purpose of linguistic geometry is to provide solutions to various problems with very large state spaces. These problems include issues such as the cooperation or competition of robot teams such as aircraft, spacecrafts, land and sea vehicles acting with intelligent or human guidance, planning, programming and competition of safe-critical control systems for remote fully automated objects moving such as rovers, reconnaissance vehicles and chess game. In general, in order to find the optimal or near-optimal behavior of single-independent objects in the above systems, it is necessary to search the appropriate branches in the giant search trees. Such searches are often confusing and beyond the capability of even modern computers. The linguistic geometry technique drastically reduces

the size of these search trees, making the problems numerically solvable. One of the important features of this technique is its formatting, and the use of search discoveries being developed by highly skilled specialists (chess specialist). These experts have developed successful strategies in their field that have dramatically reduced searches. Based on what is written above, the areas to be planned for the trajectory are considered like a game board in slice geometry. The Linguistic Geometry technique refers to the geometry of the game board in addition to the abstract relationships or connections that describe movements. The abstract relations here are planes, missiles, ships, tanks, firefighters, etc. written above. It expresses the movement of war or similar vehicles and their communication among themselves. As seen in Figure 3.21, the route planning of a kind of aircraft was made according to the desired strategies [80]. Linguistic geometry, used where strategy is required, works by moving a system from one state to another. While this movement is provided by the player in the game of chess, for example, it is the action taken by the senior people for the path planning of a warplane. This move is the displacement of an element of the player from one point to another point, causing the system to make a transition from one state to another.

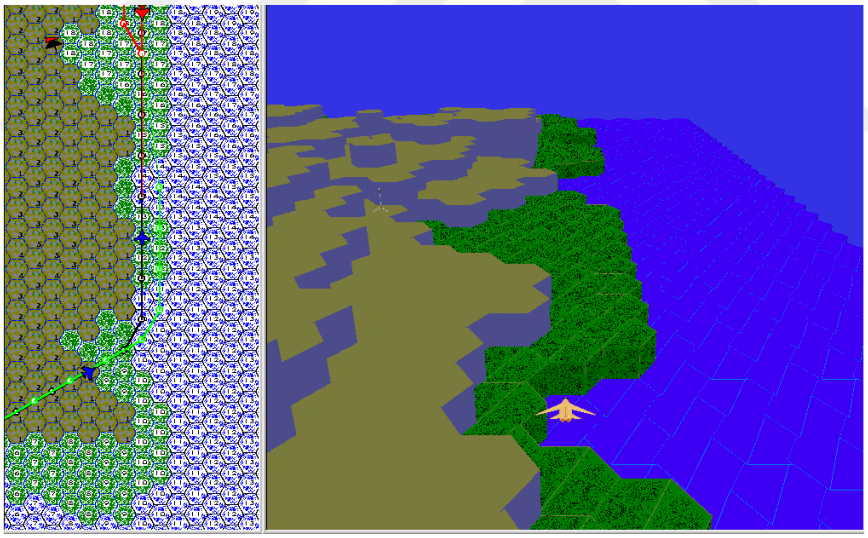

Figure 3.21- Path planning of an Airplane Using Linguistic Geometry Technique **[80]**

3.4.3 Heuristic Path Planning Techniques

Heuristic techniques are methods that provide effective and fast solutions to problems that cannot be solved by classical methods in line with general usage purposes, and which are complex and the time to reach the result increases exponentially, such as NP-Hard. Heuristic techniques have been used for problem solving in almost every field, especially in engineering. Especially in recent years, there are many studies conducted with heuristic techniques in the field of robotics. One of these studies is path planning or motion planning problems for mobile

robots. The reasons why heuristics are used in solving problems of path planning for mobile robots are the problems that arise in these techniques, especially in classical techniques, which have been explained above. Considering all these, researchers have used many heuristic techniques to solve path planning or motion planning problems for mobile robots, as seen in Figure 3.22.

Figure 3.22- Heuristic Route Planning Techniques

With the use of heuristic techniques, it is aimed to eliminate the inadequacy due to erroneous and high calculation values in studies using classical and some probability-based path planning techniques. However, it has been observed that they directly interrupt the planning in local values, especially in the potential fields technique. Again, these heuristic techniques were used to get rid of these problems [42]. Masehian and Sedighizadeh [81] in 2007 examined 1381 studies on this subject from 1973 to 2007 in their study on the techniques used for path and motion planning by robots. As a result of these examinations, it was seen that classical techniques were used in all of the studies carried out in about 10 years from 1973 to 1982. Heuristic techniques started to be used in 1983 and later, and until 2007, they were used more and increased at certain rates. In Figure 3.23, while classical techniques are seen in blue color, heuristic techniques are expressed in orange color. From here, it has been seen that heuristic approaches have started to be used more than classical techniques within 15 years.

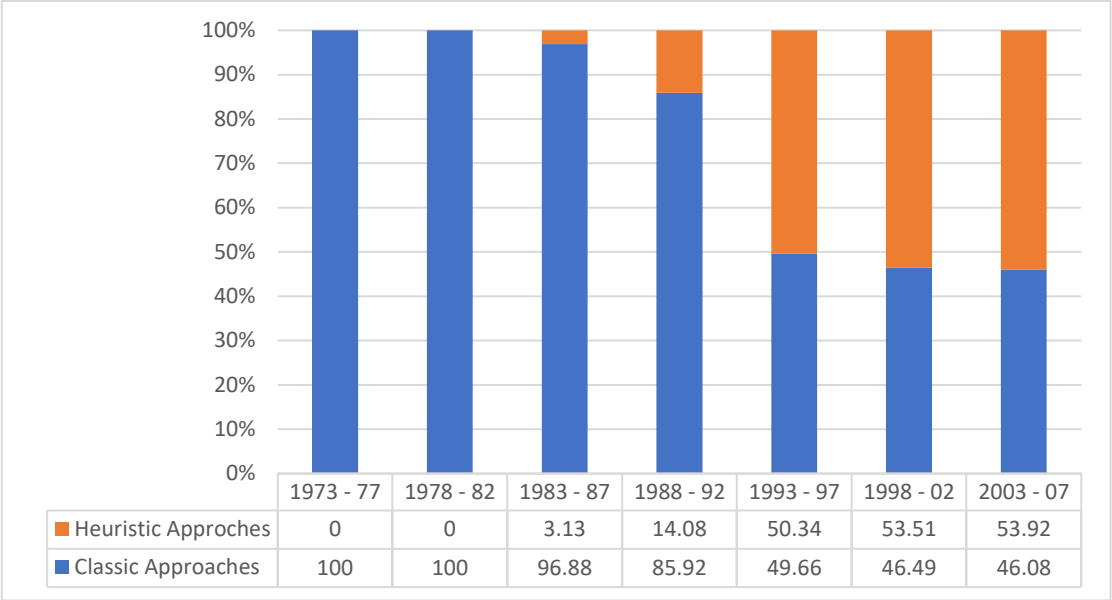

Figure 3.23- Distribution of Path Planning Techniques by Years **[81]**

Again, one of the data obtained as a result of the examination of these 1381 studies is the usage rates of heuristic techniques employed for path planning or motion planning solution for mobile robots, as seen in Table 3.2.

Table 3.2- Usage rates of heuristic techniques used for solving path planning problems of mobile robots by years **[81]**

Approach	83 – 87	88 – 92	93 – 97	98 – 02	03 – 07	Total
Artificial Neural Networks	0	60	36	44.26	25.53	**34.58**
Genetic Algorithm	0	20	34.67	32.24	37.23	**34.54**
Simulated Annealing	0	5	4	1.09	2.13	**2.32**
Particle Swarm	0	0	0	0.55	4.26	**2.04**
Ant Colony	0	0	1.33	3.83	10.64	**6.12**
Stigmergy	0	0	1.33	1.64	3.55	**2.35**
Wavelet Theory	0	5	0.67	1.64	1.77	**1.57**
Fuzzy Logic	100	10	22	14.75	14.89	**16.64**
Total	100	100	100	100	100	**100**

Regarding to the information provided at Table 3.2, the studies on the ant colony algorithm that we used in the thesis for path planning have been increasing at certain rates since 1993. This rate seems to be 6.12% among the studies done with heuristic techniques. In addition, studies on the ant colony algorithm have a rate of 2.83% among all techniques (classical, probability-based and heuristic). As it can be understood from here, the use of the ant colony algorithm is increasing day by day in the field of robotics, as it is in every field.

Fuzzy Logic, on the other hand as another approach for real time trajectory tracking in this thesis, is a very useful approach which is really efficient to solve optimization problems with contiguous response space. All information about the ant colony algorithm and Fuzzy Logic approach used for path planning of mobile robots will be explained in detail in the next section.

4. HEURISTIC APPROACHES IN TRAJECTORY TRACKING

The meta-heuristic approach served as inspiration for the heuristic method, being one of the most used autonomous methods for navigation in engineering and science. In this section, two methods—the Ant Colony Optimization Algorithm (ACO) and fuzzy logic (FL)—that were inspired by evolutionary algorithms and human decision-making are applied to the problem of guiding a mobile robot across an unknown and unpredictable environments. To create innovative mobile robot movement in the presence of stationary and moving obstacles, a novel strategy relying on ACO and FL is proposed.

4.1 Ant Colony Optimization

In recent years, researchers have tried to solve many real problems, especially NP Hard problems, that classical techniques cannot solve, with the help of heuristic techniques. Live creatures in nature and their behaviors have inspired researchers and scientists in the creation of these heuristic approaches. Ants, bees, fish and birds generally come on top of the list of these creatures. While the behavior of them in nature, which normally seem simple, is investigated and examined, it is mostly focused on the behavior of these living creatures together based on cooperation and how they communicate with each other. Behaviors of living creatures based on this cooperation are defined by researchers and scientists as swarm intelligence. Swarm intelligence is based on the solution that all living creatures have come up with based on cooperation beyond the solutions they find as a single living creature, and it has brought a different perspective to artificial intelligence. Over the years, the behaviors of living creatures based on swarm intelligence have been modeled and formulated into algorithms. Especially in recent years, these algorithms have been seen to be successful in producing solutions close to the best solution in robotics, engineering, computers and many fields under the name of heuristic techniques.

In this section, detailed information has been brought regarding Ant Colony Optimization (ACO), which is one of the most important heuristic approaches and was also used in this thesis study, the developmental stage depending on real ants, the basic features of the algorithm, other ant colony algorithms and their use in different areas, and ACO's application for mobile robot path planning.

4.2 Real Ant Behaviors

In natural life, ants are very simple creatures that live in colonies and produce solutions on their own. They are creatures that have the ability to find the shortest path within a certain period of time between their own nests and the food they find as a result of research. Experiments with real ants were performed by Goss et al. [82] in 1989 on Argentine ant colonies raised in a laboratory environment. According to the results of this study, they saw that ants have no eyesight, and that when they go from nest to food or from food to nest between the nest and the food they find, they secrete a chemical substance called pheromone and thus the ants create a pheromone trace on the path they pass. It has been observed that when ants need to choose a route for food, selection is made according to the amount of pheromone traces in all paths between the nest and the food, and the ants do not provide a central control over their behavior.

The higher the pheromone substance in the trail, the more likely the ant is to follow that trail. This chemical message is the key factor in the success of the ants. This structure is useful because the ants that make the best decisions will be the first to return to the nest, leaving twice as much pheromone on their path as other ants still busy foraging for food. Since ants use a probability distribution to choose a path, there is still a small chance that an ant will choose an undiscovered path. This is an element of trust versus a local optimum solution. The possibility of not finding better solutions should be reduced by locking onto a solution that can occur locally [83].

Ants are creatures that cannot see what is going on around them because they do not have the ability to see. Despite this, they are creatures that adapt to changes in the environment in a short time and adjust their behavior according to environmental conditions, depending on the amount of pheromone material mentioned above. For example, when ants move on the shortest path between the nest and the food as a colony, or when they search for this path, as a result of the emergence of an obstacle on that path, the ants change their behavior according to these obstacles in a short time and find the shortest path between the nest and the food. As seen in Figure 4.1, let's review the situation in which the shortest path was found by all the ants in the colony between the nest and the food in a certain period of time and the ants made their movements on this route.

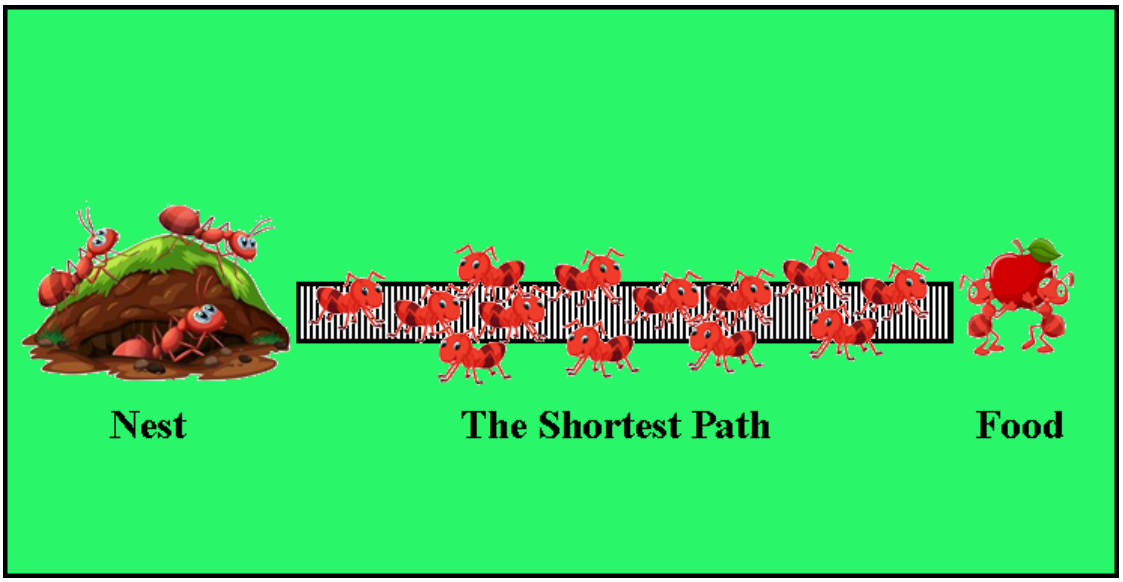

Figure 4.1- Shortest Path Between Nest and Food

Now let's assume that while the ants are on a straight route, an obstacle is placed on their pathway as shown in Figure 4.2, and this way their route has been disrupted.

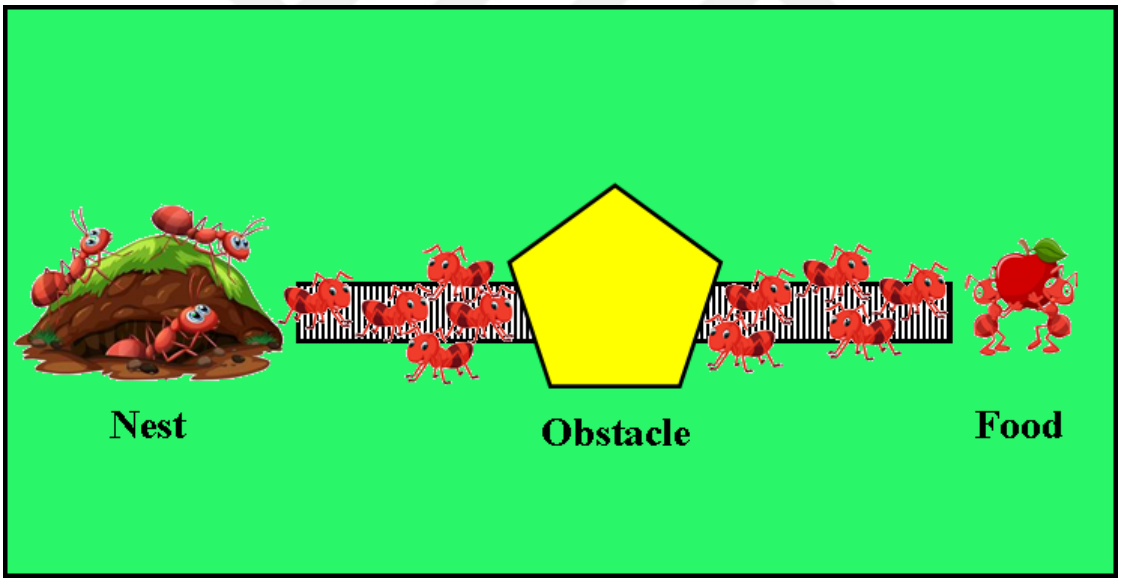

Figure 4.2- Disruption of the Shortest Path When Ants Encounter an Obstacle

In this case, the ants in front of and between the obstacle encounter directions that do not have pheromone, since the new rotes over this obstacle have not been used before. Therefore, they will have to choose a new direction to go to food. In this case, the probability of the ants going to the right and to the left is equal. In the first place, the ants choose randomly and move to one side. As seen in Figure 4.3, the number of ants moving to the right and left is very close to each other in this case.

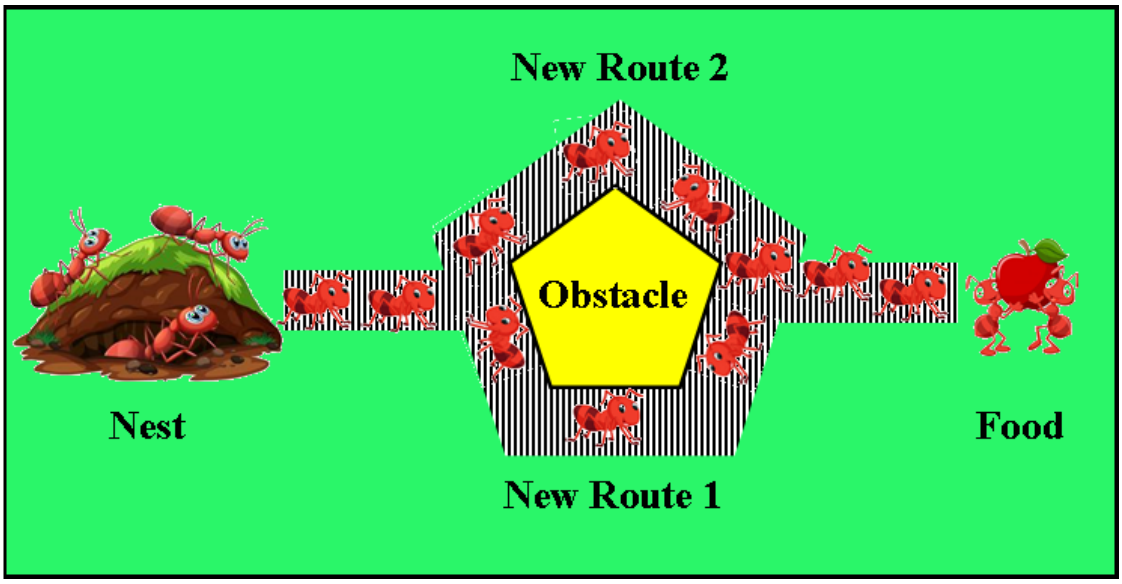

Figure 4.3- Situation of Ants' Behavior After Encountering an Obstacle

Here, if the path chosen by the ants with equal probability is not the shortest path to food, the ants quickly change their direction again. If we want to explain the reason for this we can say, assuming that all ants in the colony secrete the same amount of pheromone material at the same speed and on average to the path they have passed, the number of ants passing over the short path will be higher, and therefore the amount of pheromone material on this path will increase in unit time depending on the number of ants. With the increase in the pheromone material in this pathway, it will become attractive for the ants that come later and the probability of being selected will increase. Thus, all the ants in the colony will start to go to food on this route after a short time, and thus they will adapt to the changing environmental conditions with the placement of the barrier. Here, a positive feedback can be mentioned depending on the pheromone material. In other words, as the number of ants passing by a route increases, the pheromone material on the route will increase, and as the pheromone material increases, the number of those who prefer this route among the later ants will increase. Therefore, after a certain period of time, all of the ants in the colony will find the shortest pathway to food, as shown in Figure 4.4.

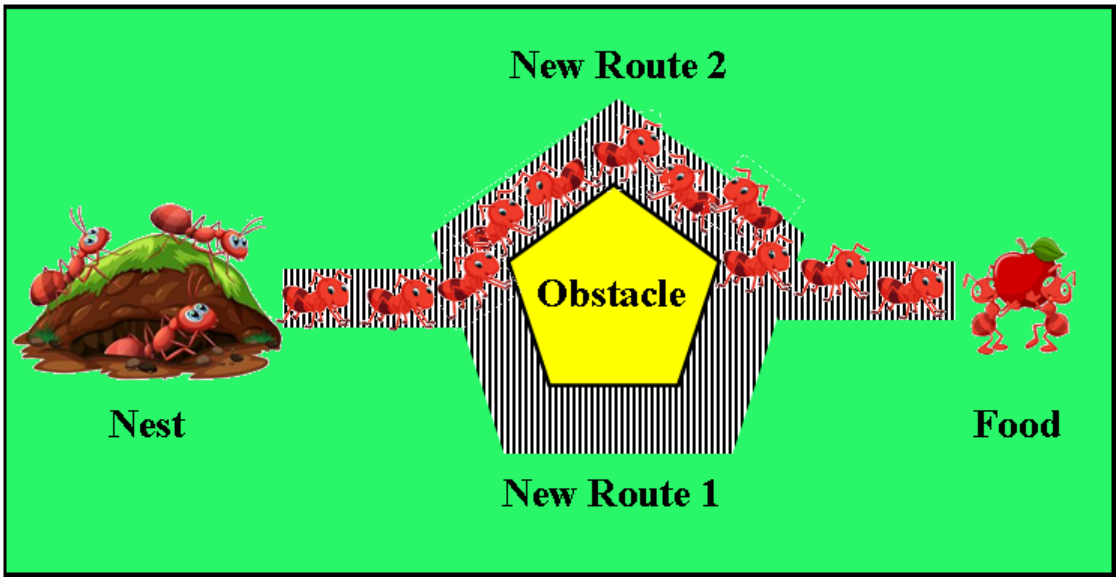

Figure 4.4- The Ants Finding the Shortest Path After a Certain Time

4.3 Ant Colony Algorithm

The first study on the ant colony algorithm (ACO) was started by Dorigo et al. [84] in 1991 and was carried out and published by Dorigo himself in 1992 as a doctoral thesis [85]. In this study, Dorigo made a modeling on mathematical formulas based on the results of his studies on the behavior of real ants by Goss et al. [82]. He named this system as Ant System and the resulting algorithm as Ant System Algorithm. However, in order to apply this algorithm to real life problems and to get more efficient results, some features listed below have been added to the artificial ants to be used in the algorithm, as well as the basic features of real ants.

Features exactly taken from real ants:

- Communication between ants via pheromone.
- Preferring the routes with higher amount of pheromone first.
- Faster increase in the amount of pheromone on shorter paths.

Added features:

- They live in an environment where time is calculated discretely, and their movements involve transitions from discrete-to-discrete states.
- They are not completely blind and can access details about the problem.
- With a certain amount of memory, they can keep the information they have created to solve the problem.

The basic idea in ant-based algorithms is that artificial intelligent agents using simple communication mechanisms can produce solutions for many complex problems [86].

Dorigo first applied the ant system algorithm he developed to traveling salesman problems (TSP) of different sizes. TSP is a problem that aims at returning a seller to the point where he had started, beginning from any point and using the optimum paths, provided that he visits each point only once. The TSP can be fully defined by weighted **G=(N,A)**. Here **N** is the set of cities (points) and **A** is the arcs connecting the points. Each arc (i,j) can be defined as (i,j)∈A. Here **(i,j)** denotes the length between two points, that is, $\mathbf{d_{ij}}$. TSP, on the other hand, is a problem that finds a minimum length on a Hamiltonian tour that reaches the starting point by passing through each point on the graph G only once [87].

4.3.1 Ant System

In the literature, 3 different versions have been proposed for the ant system: ant-density, ant-amount and ant-cycle. The difference between them lies in the pheromone storage. In the ant-cycle algorithm, only at the end of each ant round every edge on that ant's turn is stored with pheromone inversely proportional to their length. However, every virtual ant has a memory to remember the paths it has passed, contrary to the real one. In the other two algorithms, the pheromone path is updated as the ant passes from one city (point) to another. At the end of the tests, it was revealed that the ant-cycle algorithm gave better results and gained a general structure and this algorithm became what is meant when the ant system algorithm is mentioned [88].

The ant system algorithm works with its general steps as seen in Figure 4.5.

Step 1: Initial values are determined for the required parameters.

Step 2: All ants are randomly placed in cities.

Step 3: Select the ants to move from their city to another city according to the probability formula.

Step 4: After completing each ant tour, the length of the paths is calculated and the pheromone is updated according to the best value.

Step 5: Empty the tabu lists until you meet the number of iterations or other criteria and return to Step 2.

Figure 4.5- How the ant system algorithm works

4.3.1.1 Starting the Algorithm and Assigning Values

The Ant System algorithm starts with the determination of the initial parameter values as seen in Figure 4.5. These parameter values affect the performance for many reasons during the solution of the problem, so these parameter values should be selected and determined well. Although some of the parameter values have taken certain values as a result of the studies carried out by the researchers, these values may change in some problems.

The first thing that comes to mind from these parameter values is the number of ants. The higher the number of ants, the higher the running time of the algorithm, although we get better results. Therefore, as a result of the experiments conducted by Dorigo et al. [84] for the solution of traveling salesman problems, it was seen that it would be appropriate to choose the number of ants equal to the number of cities. If the total number of ants in the colony is expressed as m and the number of cities as n, m=n equality should be provided in determining the parameters. Due to this relationship between these two quantities, the Equation 4.1 comes out.

$$m = \sum_{i=1}^{n} b_i \qquad (4.1)$$

Here, if b_i denotes the number of ants in the i^{th} city, the total number of ants can be obtained based on the number of cities.

Another important parameter in the ant system is the first pheromone value. As it can be understood from the definitions explained above, the pheromone material value is one of the important parameters that provide parallelism between the artificial ants in the algorithm. Therefore, a pheromone matrix is created at the beginning of the algorithm to generate the probability values required for the ants to move from one point to another. The size of this pheromone matrix depends on the number of ants and cities. The matrix shows the amount of pheromone stored between points i and j and is defined as $\boldsymbol{\tau_{ij}}$. The initial value suggested by Dorigo and Stützle [87] for the beginning of the pheromone matrix is given in Equation 4.2 for ∀(i,j). Here, it is seen that all paths in the pheromone matrix take the same value, that is, the initial value of $\boldsymbol{\tau_0}$.

$$\tau_{ij} = \tau_0 = m/C^{nn} \qquad (4.2)$$

Here, $\boldsymbol{m}$ represents the total number of ants, while the $\boldsymbol{C^{nn}}$ value represents the tour length created by the nearest neighbor heuristic. If the initial pheromone value, $\boldsymbol{\tau_0}$, was extremely low, the search would quickly become biased in the ants' first rounds. On the other hand, if the initial pheromone value was excessively high, a loss would occur as the same rounds of different paths would be selected until the pheromone evaporated after a few rounds. From this point of view, another parameter value that should be taken into consideration in the beginning of the ant system algorithm is pheromone evaporation.

Without pheromone evaporation, the ant system algorithm does not give accurate results in solving problems. Because the first discoveries in the search space are completely random, the pheromones left at this stage do not contain any information. Therefore, the evaporation mechanism is necessary for other ants to easily forget these useless values and to find a way through good solutions [88]. The evaporation coefficient is denoted by $\boldsymbol{\rho}$ and this value is determined by values between 0 and 1. Dorigo [85] determined as a result of his studies that the most appropriate value for this parameter is 0.5.

One of the most important parameters of the algorithm that should be determined at the beginning is the selectability parameter. The selectability parameter expresses the heuristic value of moving from city $\boldsymbol{i}$ to city $\boldsymbol{j}$. This selectability parameter is also considered in the distance between all paths in the problem, instead of making a selection based on the pheromone value only in the probability value equation. The selectability parameter is constant throughout the algorithm since the cities in the problem are static, and a selectability parameter matrix is created at the beginning of the algorithm. This eligibility parameter, which is created depending on the cities, is found as shown in Equation 4.3.

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{4.3}$$

Here, $\boldsymbol{d_{ij}}$ is the distance between cities $\boldsymbol{i}$ and $\boldsymbol{j}$. These distances are calculated according to the Euclidean equation as in Equation 4.4 at the beginning of the algorithm.

$$d_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \tag{4.4}$$

The most important parameter values for the probability equation that determines the ants going from one city to another city are actually alpha (α) and beta (β) values. While the α value determines the importance of the amount of pheromone between the i and j cities, the beta β value determines the importance of the path length between the $\boldsymbol{i}$ and $\boldsymbol{j}$ points. Dorigo in [85] determined as a result of his studies that the most appropriate value for this parameter is 1 for α and 5 for β. Again, Dorigo et al. in [89] obtained the parameter selection graph for α and β values as seen in Figure 4.6 in their study.

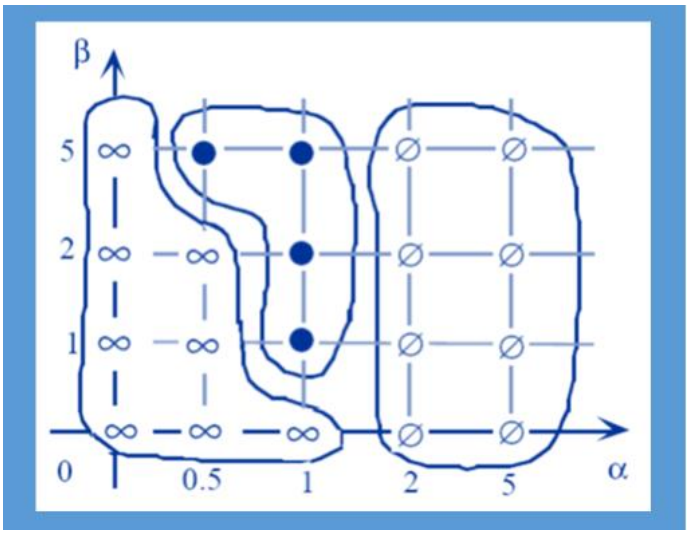

Figure 4.6- Graph of Selection Intervals for α and β Values **[89]**

Here, in the part indicated by (●), the algorithm finds the best-known result without going into stagnation. In the part indicated by (∞), the algorithm cannot find good solutions without entering the stagnation behavior. In the part indicated by (ø), the algorithm cannot find good solutions and reaches a state of stagnation behavior [89].

Another parameter that should be specified at the beginning of the algorithm is the number of turns (iterations) of the ant. The number of iterations is determined according to the creation of the same round of all ants in the algorithm and a determined maximum number of rounds NCmax value.

4.3.1.2 Creation of Ants' Tour (Movement)

After all the parameters mentioned above are determined for the algorithm at first, m artificial ants in the algorithm are randomly placed in n cities and the algorithm starts to run. Artificial ants find out which point they will move to in the next step from a point in each of their rounds with a proportional formula based on probability. Accordingly, the probability formula for the movement of the k[th] ant from city i to city j is as in Equation 4.5.

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k} [\tau_{ij}]^{\alpha} \cdot [\eta_{ij}]^{\beta}} & if\ j \in N_i^k \\ 0 & otherwise \end{cases} \qquad (4.5)$$

The definition of the parameters in equation 4.4 is shown in the table 4.1 below.

Table 4.1- Ant Colony Algorithm's Parameters' Definition

Parameter	Definition
τ_{ij}	The amount of pheromone substance between points i and j.
η_{ij}	The heuristic value of moving from point i to point j.
α	Parameter that controls the amount of pheromone trace.
β	Parameter that controls the selectability (heuristic) parameter.
N	The set or number of cities (nodes).
N_i^k	Set of cities that the kth Ant has not visited yet.

As can be seen from the probability formula in Equation 4.5, the importance of the selection of the determined parameters, especially the α and β values, is evident.

If α = 0, the nearest cities are more likely to be selected, corresponding to a classical stochastic greedy algorithm. If β=0, only the pheromone is used without any heuristic effect and the pheromone rise continues [87]. On the other hand, a high α value increases the probability of choosing the routes with high pheromone concentration, while decreasing the coincidence. Likewise, as the β value increases, coincidence increases in the selection of the next path [89].

However, each $\boldsymbol{k}$ ant has a $\boldsymbol{M^k}$ memory that contains, in order, the cities they have visited before. This memory is used to define possible $\boldsymbol{N_i^k}$ neighbors according to the rule structure defined in equation 4.5. In addition, the $\boldsymbol{M^k}$ memory allows calculating the length of the tour created by the $\boldsymbol{k}^{th}$ ant and reconsidering the path to introduce pheromones [87].

4.3.1.3 Updating Pheromone Traces

After all the ants have made their rounds, the pheromone traces are updated. This is done first by reducing the pheromone value on all routes with a fixed factor, and then by adding pheromones to the routes they pass in their tours [87].

Different pheromone renewal rules are used in different versions of ACO. In the ant colony algorithm, pheromone renewal occurs at two levels, local and global, and the total pheromone level in a route consists of the sum of the local and global pheromone levels [90]. For local pheromone updating, after all ants have completed their tours, old pheromone amounts are evaporated at a certain rate, and a certain amount of pheromone increase is provided on the

paths that each ant has passed through during their tour [91]. Local pheromone update is done as seen in Equation 4.6.

$$\tau_{ij} = (1-\rho)\,\tau_{ij}(t) + \Delta\tau_{ij} \qquad (4.6)$$

Evaporation Equation's parameters are figured out in table 4.2 below.

Table 4.2- The Definition of Evaporation Equation Parameters

Parameters	Definition
ρ	pheromone evaporation coefficient
$(1-\rho)$	The evaporation rate of the pheromone substance between the (t) and (t+n) periods.
$\Delta\tau_{ij}$	The amount of pheromone material left on the (i,j) line per unit time

The amount of pheromone material left on the (i,j) line per unit time which was mentioned as $\Delta\tau_{ij}$ is obtained as it is expressed in Equation 4.7.

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (4.7)$$

Here $\Delta\tau_{ij}^{k}$ is the amount of pheromone material left on the (i,j) line by the kth ant and is expressed as seen in Equation 4.8.

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L^{k}} & if\ the\ ant\ has\ used\ the\ (i,j)path \\ 0 & Otherwise \end{cases} \qquad (4.8)$$

The definition of all the parameters related to pheromone changes used in equations 4.7 and 4.8, are explained in table 4.3 below.

Table 4.3- The Definition of Parameters Used to Calculate Pheromone Changes

Parameters	Definition
m	Total number of ants
$\Delta\tau_{ij}^{k}$	The amount of pheromone substance left on the (i,j) line by the kth ant
Q	a constant value
L^{k}	total tour length of kth ant

The global pheromone update is done after all ants have completed their rounds. After calculating the total path lengths of each of the ants, the ant taking the shortest path is found [90]. Global pheromone update is done as seen in Equation 4.9.

$$\tau_{ij} = (1-\rho)\,\tau_{ij}(t) + \Delta\tau_{ij}^{k} \qquad (4.9)$$

Here, ρ and $(1-\rho)$ have the same definitions as were explained above in table4.2. The parameter $\Delta\tau_{ij}^{k}$ indicates on the amount of pheromone material left on the (i, j) line by the k[th] ant who found the best path, and is expressed as shown in Equation 4.10.

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_{(best)}} & if\ (i,j)\ belongs\ to\ the\ best\ path \\ 0 & Otherwise \end{cases} \qquad (4.10)$$

Here $L_{(best)}$ is the length of the best round found in the current lap [92]

4.4 Ant Colony Algorithm Versions

In the studies carried out for the solution of traveling salesman problems with Ant Colony Algorithm, different algorithms have been developed after the ant system algorithm. At the top of these are the Elitist Ant System, the Rank Based Ant System, and the Maximum Minimum Ant System. In these proposed studies, changes were made based on pheromone update and the best result was tried to be reached.

4.4.1 Elitist Ant System

The first development regarding the ant system was proposed and developed by Dorigo [85] in 1992 and then a few years later by Dorigo et al. [89] in 1996, which is called the elitist strategy for the ant system. The main idea is to ensure that extra pheromones are added to the paths of the best lap found since the start of the algorithm. In this algorithm, the pheromone update is performed as seen in Equation 4.11.

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs} \qquad (4.11)$$

In this equation $\boldsymbol{e}$ is a parameter that defines the importance given to the best round found from the start of the algorithm so far, and $\boldsymbol{\Delta\tau_{ij}^{bs}}$ is the amount of pheromone substance on

the best round from the beginning of the algorithm up to that time and is found as shown in Equation 4.12.

$$\Delta\tau_{ij}^{bs} = \begin{cases} \frac{1}{L^{bs}} & if\ the\ path(i,j) belongs\ to\ \tau^{bs}\ lap \\ 0 & Otherwise \end{cases} \quad (4.12)$$

Here L^{bs} is the length of the best lap.

4.4.2 Rank Based Ant System

The rank-based ant system was proposed by Bullnheimer et al. [93] in 1997; and as a result of the experiments, it was seen that it performed slightly better than the ant system. In this system, before the pheromone tracks are updated, the ants are sorted according to their reproductive length and the amount of pheromone an ant will leave gains weight according to the rth ant's grade. In each round, only the best ranked ants and the ant that has produced the best lap so far are allowed to deposit pheromones. The best lap so far provides strong feedback with a weight parameter w. In other words, the pheromone contribution to be provided by the best rth ant of the current round is according to the result obtained by multiplying a value among the $max(0, w-r)$ values with the $\frac{1}{L^r}$ expression [87]. Accordingly, the pheromone update of this system is done as seen in Equation 4.13.

$$\tau_{ij} = \tau_{ij} + \sum_{r-1}^{w-1} (w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{bs} \quad (4.13)$$

Here $\Delta\tau_{ij}^{r}$ is the amount of pheromone substance left on the i and j pathway by the r -ranked ant and is found with $\frac{1}{L^r}$ (L^r= r-ranked ant's tour length), while $\Delta\tau_{ij}^{bs}$ is the amount of pheromone substance on the best round from the beginning of the algorithm to that point and is found by $\frac{1}{L^{bs}}$.

4.4.3 Maximum-Minimum Ant System

The maximum-minimum ant system was first proposed by Stützle and Hoos [94] in 1997. In order to prevent stagnation, especially during the search, the recommended maximum-minimum ant system brings 4 fundamental changes compared to the ant system. First, the best ant in the current round or rounds so far is allowed to add pheromone material, but this will cause a stagnation where all ants follow the same round. Because, although there is not enough tour, the pheromone trail on a good pathway will increase unlimitedly. Therefore, in order to

avoid this stagnation, secondly, possible pheromone trace values are limited between values as [τ_{min} , τ_{max}]. Third, pheromone traces are initiated relative to the upper pheromone trace boundary with a small evaporation coefficient to increase the discoveries of the search head tours. Finally, in the maximum-minimum ant system, pheromone traces are returned to the baseline each time the system approaches stagnation or does not find a well-rounded tour for a given number of successive tours [87]. According to all these, in the maximum-minimum ant system [τ_{min} , τ_{max}] pheromone boundaries are found as seen in Equation 4.14.

$$\tau_{max} = \frac{1}{\rho . L^{best}} \quad , \quad \tau_{min} = \frac{\tau_{max}}{a} \qquad (4.14)$$

Here τ_{max} and τ_{min} respectively indicate on the upper and lower pheromone boundaries, while L^{best} shows the length of best lap found, and a is a constant value.

4.5 Ant Algorithms for Numerical Problems

Since the day the ant colony was proposed, many studies have been carried out in the field of engineering, especially to solve continuous problems. Ant algorithms, as explained above, were first proposed by Dorigo et al. [95] in 1991, and Dorigo [85] in 1992 as a multi-agent approach for difficult combinatorial problems such as the traveling salesman problem and the quadratic assignment problem. There are many ongoing studies in the scientific world to develop ant-based algorithms for different discrete optimization problems [96]. Examples of this are intermediate routing problems, graph painting, network routing, regular and common sequencing problems. Some of the applications related to these studies so far are given in Table 4.4.

Table 4.4- Studies on ACO for Discrete Optimization Problems **[97]**

Case	Author	Period	The Algorithm
sequential ordering problem	Gambardella & Dorigo	1997 – 2000	HAS-SOP
Common Supersequence Problem	Michel & Middendorf	1998 – 1999	AS – SCS
Quadratic Assignment	Maniezzo, Colorni& Dorigo Gambardella, Taillard &Dorigo Stützle & Hoos Maniezzo & Colorni Maniezzo Colorni,Dorigo&Maniezzo	1994 1997 – 1999 1998 1998 1998 1994	AS – QAP HAS – QAP MMAS – QAP AS – QAP ANTS – QAP AS – JSP
Graph Painting	Costa & Hertz	1997	ANTCOL
Connection Based Network Routing	Schoonderwoerd, Holland, Bruten&Rothkrantz White, Pagurek & Oppacher Di Caro & Dorigo	1996, 1997 1998 1998 1998	ABC ASGA AntNet-FS ABC-smart ants

	Bonabeau, Henaux, Guerin Snyers, Kuntz & Theraulaz		
Connectionless Network Routing	Di Caro & Dorigo Subramanian, Druschel & Chen Heusse, Guerin, Snyers & Kauntz Van der Put & Rothkrantz	1997 – 1998 1997 1998 1998	AntNet&Ant NetFA Regular ants CAF ABC-backward
Vehicle Routing	Bullnheimer, Hartl & Strauss Gambardella, Taillard &Agazzi	1996, 1997, 1998 1999	AS-VRP HAS-VRP
Traveling Salesman problem	Dorigo, Maniezzo & Colorni Gambardella & Dorigo Dorigo & Gambardella Stützle & Hoos Bullnheimer, Hartl &Strauss	1991, 1992, 1996 1995 1996, 1997a,b 1997a,b 1997	Ant-Q ACS & 3 opt MMAS ASrank

Although ant algorithms have been used successfully in the field of discrete optimization problems as above, this is not the same for continuous optimization problems. An ant algorithm was first proposed by Bilchev et al. [98] in the year 1995 for continuous problems, but the ant colony algorithm mechanism was used only for local search. Later, Wodrich [99] in 1997 proposed an efficient two-level search operation using ant logic and this algorithm was named CACO. The algorithm was later expanded by Mathur et al. [100] in 2000, and its performance was significantly increased.

CACO consists of two stages: local and global search levels. The local search is done by the ant colony algorithm while the global search is done by the genetic algorithm. In fact, the global ants perform a simple evaluation in order to update the fitness of certain regions determined in the search space. The creation of some new regions is achieved by a method similar to the genetic algorithm that uses common functions assimilated by researchers for real ant colonies with behavior such as random wandering behavior (manipulating mutation and crossover parts of the Genetic Algorithm). The local level is created by ants that explore more systematically with a simple condensation behavior for orderly movement to regions close to the optimal value. The algorithm sends some local ants to the regions, when these ants find an improvement in the objective function, they deposit pheromone material at those points and these points become attractive to all ants in the colony. Although CACO is very close to classical ACO, the use of two different applications requires a fine adjustment of parameter values [101].

Other proposed algorithms for continuous optimization problems are GACO, which is a grid ant colony optimization algorithm, and TACO, which is a touring ant colony optimization algorithm. These algorithms were suggested by Hiroyasu in 2000 [102].

In TACO, each solution is represented by a string of binary bits. Therefore, artificial ants try to decide on the values of the bits in the bit string. The basic idea of this approach is illustrated in Figure 4.7.

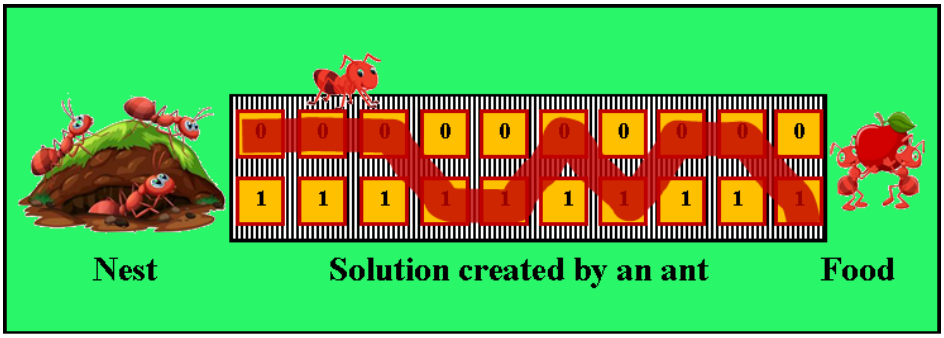

Figure 4.7- Solution Created by an Ant **[103]**

In this structure, ants use only pheromone information to decide the value of bits. After deciding on the values of all the bits in the array, it means that a solution candidate for the problem has been produced. The solution candidate produced is evaluated in the problem, and the quality value of this solution candidate is calculated and the amount of artificial pheromone material to be left on the pathway of the artificial ant is calculated by using the quality value.

For example, the probability of choosing a connection (0→1) between bits 0–1 at any position in a bit string is calculated as seen in Equation 4.15.

$$p_{01} = \frac{\tau_{01}}{\tau_{01} + \tau_{00}} \qquad (4.15)$$

Here τ_{00} indicates on the amount of artificial pheromone over (0 → 0) connections, while τ_{01} denotes the artificial pheromone amount of (0 → 1) connections, and p_{01} is the probability of choosing the (0 → 1) connection.

The amount of artificial pheromone is calculated using the expression in Equation 4.16.

$$\Delta\tau_{01}^{k}(t, t+1) = \begin{cases} \dfrac{Q}{CF_k} & if\ the\ k^{th}\ ant\ crosses\ the\ path\ (0 \rightarrow 1) \\ 0 & Otherwise \end{cases} \qquad (4.16)$$

Here $\Delta\tau_{01}^{k}$ indicates on the amount of artificial pheromone adhered to (0 → 0) by the kth ant, while Q is a positive constant value, and CF_k is the value of the Cost (Purpose) Function.

After the M ants in the colony complete the research process and produce the solutions, the amount of pheromone to be attached to the (0→1) connection in the [t-(t+1)] time interval is found as seen in Equation 4.17.

$$\Delta\tau_{01}(t,t+1) = \sum_{k=1}^{M} \Delta\tau_{01}^{k}(t,t+1) \qquad (4.17)$$

The amount of pheromone in the (0→1) interconnection at the time of (t+1) is found as shown in Equation 4.18.

$$\tau_{01}(t+1) = \rho\tau_{01}(t) + \Delta\tau_{01}(t,t+1) \qquad (4.18)$$

Here ρ denotes the evaporation coefficient and $(1-\rho)$ represent the evaporation amount of the pheromone.

The main disadvantage of the algorithm described above for continuous optimization problems is that it only uses the amount of odor when determining the direction. For this reason, early convergence and local minima problems are encountered in some optimization problems [103]. In order to overcome this disadvantage, a strategy based on the random variation of the odor amount of the sub-pathways was developed by Kalınlı et al. [104] in 2001.

4.6 Ant Algorithms for Mobile Robots' Path Planning Problems

Path planning problems for mobile robots are known as NP-Hard problems. In order to reach a solution in NP-Hard problems, all possible solutions in the problem must be tried one by one. Therefore, in very complex problems, this means that the time increases rapidly. The most important factors that make path planning problems difficult are the increase in the size of the workspace and the configuration space of the robot. As this dimension increases, the routing problem will become more complex and more difficult. In solving this difficult problem, the biggest advantage of heuristic path planning techniques over classical and probabilistic path planning techniques is that it produces an acceptable solution quickly and fast, and therefore heuristics are techniques that seem suitable for solving NP-Hard problems such as path planning problem.

One of the heuristic techniques used in path planning for mobile robots is ACO, and there are many studies in the literature on ant colony for path planning. Compared to the past, especially in recent years, road planning studies with ant colonies are frequently carried out. In path planning research and studies with ant colonies, ACO has often been used to search in a space created by classical and probabilistic path planning techniques. However, some studies have tried to find the desired solution by creating search spaces alone or with different heuristic techniques. In some of these studies, different obstacle avoidance formulas were modeled with ant colony algorithms.

Studies with ACO have generally yielded good results and it has been stated that it is an effective technique in solving these problems. In particular, finding optimal solutions to problems quickly is shown as the main reason for using this technique. However, one of the common features of heuristic techniques, making changes on the algorithm, being understandable and providing suitable features for evaluation has shown the applicability of the ant colony for trajectory tracking problems.

4.7. Trajectory Planning of the Mobile Robot with Fuzzy Logic

Mobile robot trajectory planning is a crucial aspect of robot navigation and control. It involves the creation of algorithms and strategies that enable a robot to move from one location to another while avoiding obstacles, following a predetermined path, and maintaining a desired speed and acceleration. However due to the physical constraints of the application theatre, it is really trivial issue to construct a path due to the missing and uncertain data of the environment. Fuzzy logic is a mathematical system that permits the representation and manipulation of imprecise or uncertain data. It is based on fuzzy sets, which permit continuous rather than discrete values, and fuzzy rules, which permit the representation of complex relationships between variables. Therefore, in this thesis we aimed to develop a fuzzy logic-based trajectory planning algorithm for mobile robots in a dynamic and uncertain environment. Experimental results showed that the proposed approach reaches applicable trajectory paths for complex environments.

4.7.1 Fuzzy Logic Based Trajectory Tracking Introduction

Autonomous systems are frequently employed in many industries to complete difficult tasks in hazardous locations. A major problem in the obstacle avoidance and surveillance of unmanned robotic systems is the design of the robots' pathway between the primary location and the destination spot, which has been the focus of interest in recent decades. There are numerous reasons why mobile robots require trajectory planning. First, it allows robots to operate safely and efficiently in environments that are dynamic and uncertain. By meticulously planning their trajectory, robots are able to avoid collisions with obstacles and other objects, as well as follow the most efficient path to their destination. This is particularly important for robots operating in crowded or dangerous environments, where even a single collision could result in severe damage or injury.

Secondly, trajectory planning enables robots to complete complex missions and tasks. Numerous modern mobile robots are designed to perform a variety of tasks, including inspection, delivery, and rescue operations. In order to complete these tasks, robots must frequently navigate complex environments and interact with a variety of objects and systems. Algorithms for trajectory planning assist robots in determining the optimal method for approaching and manipulating objects, allowing them to complete their tasks efficiently and effectively. Planning trajectories is crucial for the development of autonomous systems. Autonomous systems, such as self-driving cars and drones, rely on accurate and reliable

trajectory planning algorithms to navigate their environments and make action decisions. Without effective trajectory planning, these systems could not operate effectively or safely.

If you have full knowledge of the environment, it is a relatively easy task to construct a trajectory for mobile robots. However, in many real-world scenarios, it is really hard to get full information about the environment. Therefore, an applicable algorithm should be used on imprecise and missing data.

If you have full knowledge of the environment, it is a relatively easy task to construct a trajectory for mobile robots. However, in many real-world scenarios, it is really hard to get full information about the environment. Therefore, an applicable algorithm should be used on imprecise and missing data.

Fuzzy logic is ideally suited for dealing with imprecise or uncertain data because it permits the representation of values that fall between true and false using a range of continuous values between 0 and 1. This permits the representation of uncertainty and the modeling of complex relationships, which may not be possible with Boolean logic, which only permits true or false values. Fuzzy logic also includes the application of fuzzy rules, which permit the representation of intricate relationships between variables. Typically, fuzzy rules are written as "if-then" statements, with the "if" portion representing the antecedent, or input to the rule, and the "then" portion representing the consequence, or output of the rule. The combination of fuzzy rules can create a fuzzy logic system, which can be used to process imprecise or uncertain data and make decisions based on that data.

Bajrami, et al [105]. in 2016, developed a mobile robot that can detect and avoid obstacles in a closed system using only one camera. They used an optimization approach like the genetic algorithm on the steps generated by the fuzzy logic algorithm that the mobile robot needs to achieve its target. For autonomous navigation, trajectory tracking is a crucial task. Such a task is challenging to complete, though, because the best path must be altered as soon as a new roadblock shows up. The result is a path that is less than ideal, and the robot may become stuck in local minima. Tutuko, et al. [106] in 2017, combined the ant colony optimization (ACO) and fuzzy logic approaches to make a globally optimal path. They conducted a real experiment with the Explorer-Follower robot. The results showed that the proposed algorithm enables them to successfully identify the shortest path without colliding.

In decision-making systems based on fuzzy logic, membership functions are essential. It is challenging to determine the value of the membership functions in the fuzzy logic model, though. This method always relies on trials and errors and professional technical input, with some works producing inferior results. Therefore, it is required to choose the best membership functions possible. For fine-tuning the fuzzy membership functions, Siti Nurmaini et al. [107] in 2018, used an optimization technique based on the Particle Swarm Optimization algorithm. The technique was used to regulate the direction and location of the mobile robot. An excellent response is produced by the fuzzy control when utilizing this method, including a quick rise time, a minimal maximal overrun, and a quick best way to reach a stable situation. Therefore, in this paper, we aim to embed the fuzzy logic concept into the trajectory and path planning of the mobile robots to be executed in an uncertain environment.

4.7.2 Different Trajectory Tracking Approaches

The goal of trajectory trailing for an autonomous vehicle is to find a short route connecting the primary location to the destination that avoids crashes while achieving specific requirements, including the route's span or distance. This issue has been resolved in a number of ways. Regarding the features of the environment, many such techniques are typically categorized into two groups as shown in Figure 4.8 below.

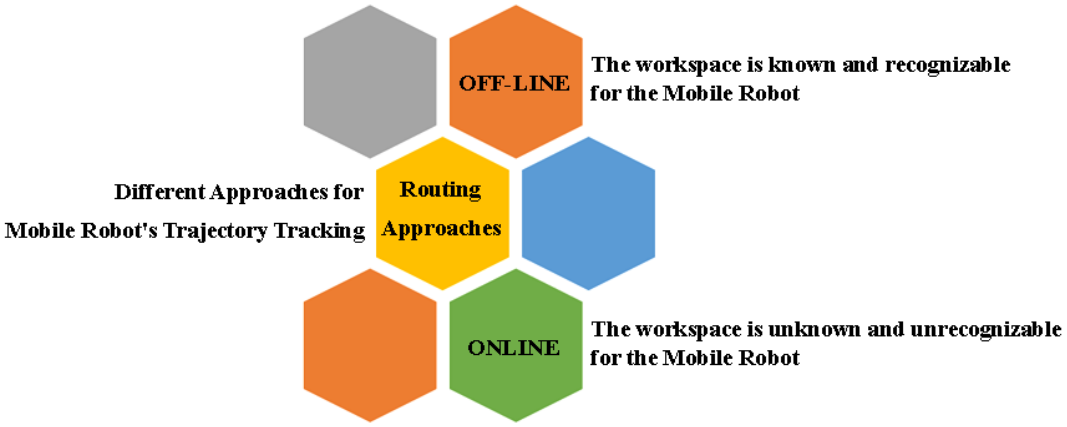

Figure 4.8- Different Approaches for Mobile Robot's Trajectory Tracking

4.7.2.1 Starting the Algorithm and Assigning Values

In this approach the terrain is recognizable by the robot and trajectory occurs base on a predefined map of the settings. The off-line route design method provides the overall optimal route by using complete and accurate information about the environment and obstacles.

4.7.2.2 Routing with an on-line approach:

This approach is preferred when the terrain is not static and predefined, so it cannot be recognized by the robot due to a lack of a predefined map of the settings. Consequently, autonomous vehicles need to use sensor data to plan online and real-time routes. Online routing can create a collision-free path utilizing the information that is already accessible from sensors, but because it only uses partial and inaccurate information, the route may not be the overall best one. The route design methods are divided into static and dynamic categories in relation to the obstacles in the environment. The static environment does not include any moving obstacles in the robot's movement space, while the dynamic environment has moving obstacles (it can be considered as the presence of humans or animals in the environment, moving cars, and any other kind of unmanned moving robots). Mobile robots are expected to perform specific tasks in any kind of environments either known or unknown, so after assigning the task, the mobile robot faces the problems of dynamic trajectory planning besides the static obstacles while moving in the environment, and avoiding moving obstacles is the most important matter to be solved. Many of the classical routing techniques in static environments have been developed for dynamic environments as well, but it is claimed in many studies such as Kyprianou et al. [108], they do not work well in dynamic environments. Various methods have been proposed by researchers to solve the trajectory tracking problem for mobile robots. Barraquand and Latombe [109] in 1991 proposed the cell division method and divided the search space into simple and interconnected areas called cells. In this method, a connection graph is formed by specifying the starting point, target, and unobstructed and adjacent cells, and the path of the robot is determined by using search methods.

Wang et al. [110] in 2000 analyzed the free space by using the geometry of obstacles in the form of a network of curves and lines, which is called a road map, and by connecting the origin and destination of the robot to the road map and using search methods, he determined the path of movement. But cell division and road map methods are not suitable for route design in complex environments due to the high computational cost. Khatib [111] in 1986 used the potential field method to guide the robot. In this method, it is assumed that the robot, as a particle, is in an environment under the influence of a force field. In this environment, the target exerts an attractive force, and the obstacles exert a repulsive force on the robot. At any time, the movement is done in the direction of the virtual forces acting on the particle. Using this method requires a lot of calculations and is considered a time-consuming method. Also, one of the disadvantages of this method is that the robot stops at points with the least potential (local

minima). Of course, several methods, such as turning back and moving randomly, have been proposed to avoid placing the robot in these places such the studies held by Dudek [112] in 2000. Borestein et al. [113] in 1991, developed a rapidly avoidance technique for mobile robots called Vector Field Histogram. The VFH method is another method derived from the potential field method. In this method, the space around the robot is divided into radial parts. Then, according to the obstacles around the robot and their distance, a number is assigned to each piece as a density by the function. This density represents the distribution of obstacles around each part. Next, a set of possible directions for the robot's movement is selected, and finally one of these directions is chosen for the robot's movement. Chang and Yamamoto [114] in 2008, have combined the road map method with the potential field method to prevent the robot from getting trapped in the potential field method.

Rahul Kala [115] in 2014 applied fuzzy logic to get the shortest path in a static environment with different complexities. In this work, there were 5 different predefined static environments containing roadblocks of different shapes and sizes. It was assumed that the robot was equipped with an overhead camera that could be simply calibrated, and the image obtained from that camera could be utilized to generate a road map. This work can be mentioned as a simplified version of real-world situations in which numerous cameras are used to record various angles of the entire workspace, and their outputs are combined to produce a general map that the motion planning algorithms may use. It was assumed that a map of this kind already exists and is provided as input to the map. The robot's location at the beginning of planning and its path could both be recorded using the same camera. This fixes the localization issue. The robot's objective is changed to a region of interest with a nice appearance so that it can be used by motion planning algorithms. Additionally, it was supposed that the source and objective of the robot would be provided unambiguously.

The trajectory tracking for mobile robots in complex settings with a lack of knowledge about the surroundings is studied in this work by giving a strategy that relies on identifying a knowledge base known as a rule-based table, which is a significant and efficient concern in gaining the outcomes of the FL system.

4.7.3 Fuzzy Logic-Based Approach

The idea of fuzzy system, which is a variation of common set, was first forth by Lotfi L.A. Zadeh [116] in 1965. Instead of the characteristic function, which has a value of either zero or one, the membership function is used to specify the fuzzy set. Membership is the

likelihood that each object is a member of a set. The value of membership lies between [0, 1]. When the membership is one, an object is supposed to be a member of the set; if it is zero, it does not.

Assilian and Mamdani [117] in 1974, proposed using Fuzzy Logic to handle complicated processes, particularly when there were no rigid models. The Fuzzy Logic Control is an alternative to equation-based control, employing conditional statements (if then rules). Inference is the process of determining a rule, and it must be defined by a membership function. It is feasible to assess the reliability of each assumption through interference. A basic FL control system's procedure is described in Figure 4.9.

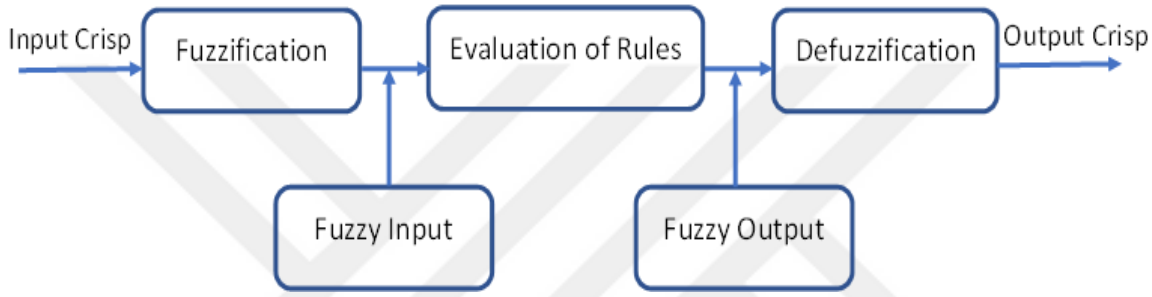

Figure 4.9- A Basic Fuzzy Logic Control System's Procedure

Fuzzy logic is based on fuzzy sets and subsets. In the classical approach, an entity is either a member of the set or it is not. When expressed mathematically, the entity takes the value "1" when it is a member of the set, and "0" when it is not a member of the set. Fuzzy logic is the extension of the classical set notation to an infinite range. Each entity in the fuzzy entity set has a membership degree. The membership degree of entities can be any value in the range (0, 1) and is represented by the membership function M(x).

Let’s take an autonomous car as an example; if we accept the safe distance as 10 meters, according to the classical set theory, we accept the distances above 10 meters as safe and the membership degrees in the safe set of these distances will be "1". Distances below 10 are dangerous and consequently the membership degrees in the safe cluster become "0" for them. These values are reversed when we base the dangerous set. In the fuzzy set approach, membership values take values in the range of [0,1]. For example, the membership degree may be "0" for a distance of 3 meters, and the membership value may be "0.25" for a distance of 10 meters. A fuzzy logic function of the distance safety (to be farther) member function based on fuzzy logic is shown in Figure 4.10 below.

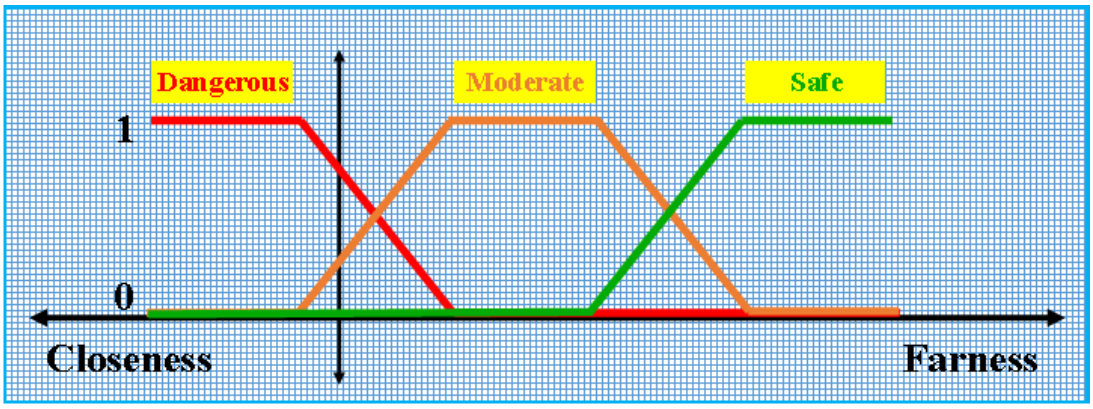

Figure 4.10- A Basic Fuzzy Logic Control System's Procedure

Unlike classical sets, in fuzzy sets, the membership degrees of the elements can change infinitely in the range of [0, 1]. They are a continuous and uninterrupted total cluster of degrees of membership. Binary variables such as cold-warm, fast-slow, light-dark in sharp sets are softened with flexible qualifiers such as a little cold, a little warm, and a little dark in fuzzy logic, and they are likened to the human logic and the real world. The most important difference is that such a framework does not have strictly defined prerequisites for cluster membership at the source of the information, and more problems and random variables are already available.

The main differences between classical logic and fuzzy logic are shown in Table 4.5 below.

Table 4.5- Differences Between Classic Logic vs Fuzzy Logic.

Classical Logic	**Fuzzy Logic**
X or Not X	*X or Not X*
precise	Partial
All or None	Regarding To Certain Degrees
0 or 1	Continuum Between 0 and 1
Binary Units	Fuzzy Units

The fuzzy algorithm controllers have proven to be an effective controller for autonomous mobile robots in planning paths and avoiding obstacles in unknown environments. Martinez et al. [118] in 1994, studied about collision avoidance for a Mobile Robot using Fuzzy Logic. They figured out the rule tables in fuzzy controller systems are extremely complex. The number of elements in this rule table increases exponentially as the input values increase.

In this thesis, mobile robot motion planning using Fuzzy Logic based method is used in simulation environment. Fuzzy Logic based routing algorithm is used to avoid obstacles.

Thanks to the developed fuzzy-based approach, it is aimed for the mobile robot to reach the target successfully by avoiding any collision with the moving and fixed obstacles in the terrain.

4.7.4 Fuzzy Logic Components and Definitions

Fuzzification, rule evaluation, and defuzzification are the three important phases that can be taken to implement the FL control, as we saw in Figure 4.9 above. Here in Figure 4.11, we can see another detailed perspective of Fuzzy Logic Components showing rule evaluation takes place employing fuzzy reasoning mechanism and knowledge base.

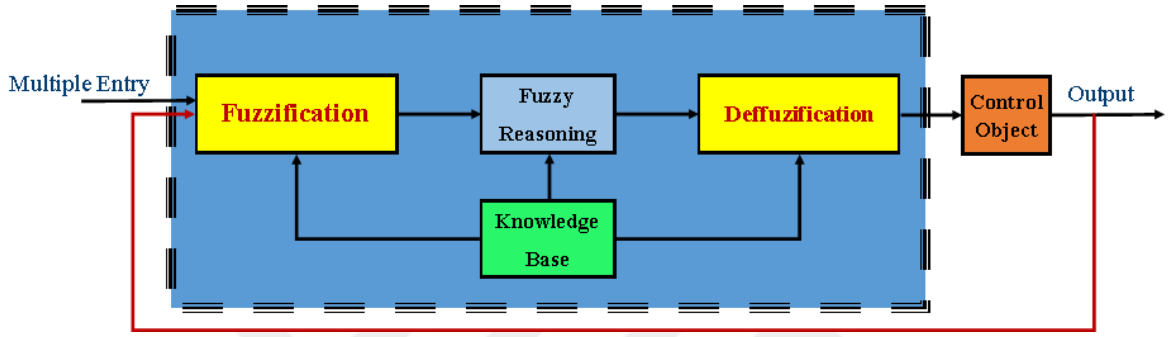

Figure 4.11- Detailed Fuzzy Logic System Architecture Showing the Fuzzy Components

In the following, the definition of each component of the fuzzy system is discussed in addition to providing detailed information for each component.

Here the definition of each component is provided:

- **Fuzzification:**

  This first unit of the fuzzy inspector measures the actual values of the input variables. It compares this measured value with membership functions and gives an appropriate label. Thanks to this label, the input values gain a linguistic feature [119]. In other words, it transforms the correct input variables into input degrees called fuzzy variables.

- **Knowledge base:**

  The linguistic controller provides the expressions necessary to describe its rules and fuzzy information management. The assessment method is determined by rule bases. It is used to store relevant data and Fuzzy Controller's rules.

- **Fuzzy Reasoning:**

  The Fuzzy reasoning is executed on the existing rules in the knowledge base and a connection is provided between the input value and the output value regarding

these rules. The most crucial component of a controller with fuzzy logic is reasoning.

- **Defuzzification:**

  The values converted to linguistic variables in the fuzzification phase, are converted back to the actual digitized values to be used in the controller in the defuzzification.

The values converted to linguistic variables in the fuzzification phase, are converted back to the actual digitized values to be used in the controller in the defuzzification.

T. Jin [120] in 2012, in his researches regarding mobile robot's obstacle avoidance based on behavior hierarchy used Fuzzy Logic. He figured out; when mobile robots move towards the target, there are three main factors that affect steering: the position of obstacles, the avoidance motion and the rotational motion. These factors are the factors that affect the cost. While performing many tasks of steering, the robot must avoid obstacles on the trajectory it moves. Therefore, in this part of the thesis, the implementation of a fuzzy-based controller that provides the obstacle avoidance behavior for the mobile robot is explained. The design of the fuzzy-based controller begins with the selection of language variables, process states, input and output member functions. The next step is the selection of the knowledge base and the reasoning process. After inference, a clarification strategy should be established which is known as defuzzification.

Fuzzy systems are easy to understand as they mimic the human control mechanism. Ergüven [121] in 1999, compared Fuzzy Logic Controller vs Classic PID Controller and figured out due to this imitation of human behavior feature, it is simple to implement and allows a large number of parallel operations. Compared to other systems, their software is simple and their efficiency is high, so the performance cost ratio is also high as Cox [122] in 1992 stated in his publication about Fuzzy Fundamentals.

In this part of the thesis, we want to embed fuzzy logic approach in order to control the motion of the mobile robot to progress from the starting point to the target. It is aimed to avoid the obstacles both dynamic and static while running the robot towards the target taking the shortest possible trajectory. If the mobile robot faces with any obstacles on its pathway, referring the rules provided in its rule base table, by real-time decision-making approach will change the direction and replace other alternative trajectories preventing crashes. In order to get

an optimal output, we should ensure of having declared efficient input(s), knowledge base rules, and assessment criteria.

4.7.5 Defining Input and Output Values

Fuzzy logic proposes to generate a control signal to avoid obstacles in unknown dynamic environments while leading the mobile robot towards the target. This generated control signal should be able to lead the robot to the target avoiding any collisions, so declaring efficient variables as the input and output is very important besides defining qualified rules.

The inputs of the fuzzy logic controller can be composed of the distances with next obstacles or the target, and also the angular variance of the robot's direction with the target.

The linguistic variables such as "near", "middle" and "far" can be used to demonstrate the distance from the obstacle to the robot. Fuzzy logic controllers can be implemented with various number of membership functions for each input.

In this thesis, fuzzy algorithm is used to avoid obstacles. As the input value for the fuzzy logic system, the distances to the nearest obstacles besides the angular variance with the target has been taken into consideration as we will give detailed information about it in the next chapter of this study. Outputs are produced as a result of the rules determined according to the inputs. The distance to the nearest obstacle is measured according to the Euclidean relation. The input values that make up the fuzzy algorithm are selected in the range of [0,1].

Figure 4.12 shows an example of an input variable in MATLAB as the distance to nearest obstacle along with its five membership functions which is used in this study.

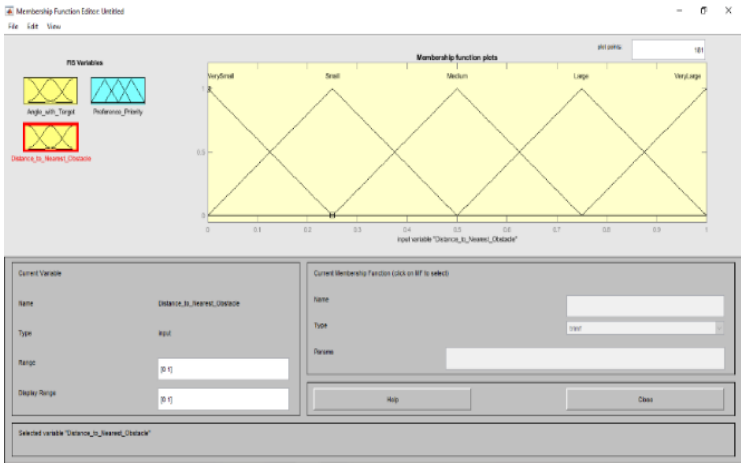

Figure 4.12- Distance_to_Nearest_Obstacle Input Membership Functions

The output values are gained corresponding to the input value being composed of desired number of member functions. The output values of the fuzzy algorithm can be chosen at any intervals depending on the problem and the modeling requirements. The output values can be named as “Reverse fast”, “backward”, “backward slow”, “stop”, “forward slow”, “forward”, “forward fast” for example. Figure 4.13 shows a sample output variable which is used in this thesis to obtain the preference priority to choose the next position for the mobile robot.

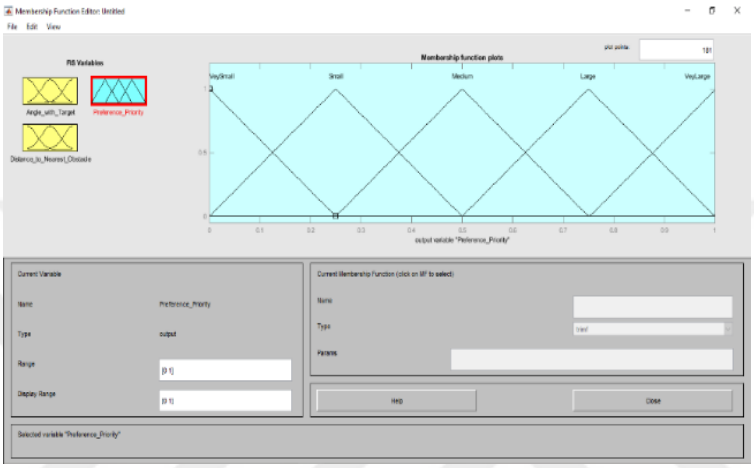

Figure 4.13- Preference_Priority Output Membership Functions

4.7.6 Fuzzification

This first unit of the fuzzy inspector measures the actual values of the input variables. It compares this measured value with membership functions and gives an appropriate label. Thanks to this label, the input values gain a linguistic feature [119]. In other words, it transforms the correct input variables into input degrees called fuzzy variables.

Fuzzing is the process of changing numerical input and output variables to symbolic values. That is each input such as Direction, Position, Angle and Distance fuzzy sets are divided into subsets such as “small”, “very small”, “medium”, “large”, “very large” or any expression like them defining the characteristic features of them. Also a validity interval range should be defined for them such as [0 , 1], [20 , 35], etc. depending on the problem and the solution approach. An example for the assumed Distance input and its subsets has been provided in Table 4.6 below.

Table 4.6- An Example for the Assumed Distance Input and Its Subsets.

SUBSETS	Interval Range
VeryNear	[0 , 10]
Near	[5 , 20]
Middle	[15 , 40]
Far	[35 , 60]
VeryFar	[55 , 80]

4.7.7 Membership Procedures

If $\boldsymbol{X}$ is a set of elements denoted by $\boldsymbol{x}$, then the fuzzy set $\boldsymbol{\tilde{A}}$ in $\boldsymbol{X}$ is the set of ordered pairs as is shown in Equation 4.19.

$$\tilde{A} = \{x, \mu_{\tilde{A}}(x) | x \in X\} \qquad (4.19)$$

Here, $\mu_{\tilde{A}}(x)$ is the membership function or degree of membership of x in $\tilde{A}$. The membership function maps the set X to the space M, as it is illustrated in Figure 4.14 below.

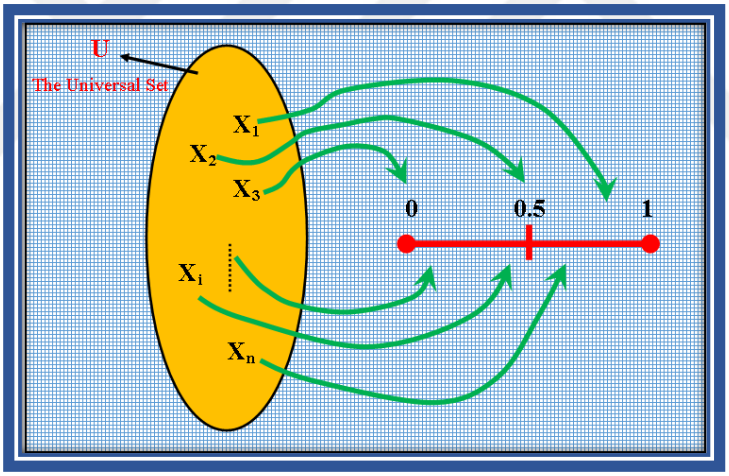

Figure 4.14- Schematic Representation of the Members of the Fuzzy Set with Membership Degrees in the Range of Zero and One

The construction of membership functions in the case of fuzzy words related to a variable includes both the general form of the membership function and its parameters.

There are different methods for determining membership functions, all of which are based on the mind (Subjective) and experience of people, and in each field, it is usually determined by the experts of that field.

Fuzzy numbers are fuzzy sets used to describe numbers imprecisely. In addition, the fuzzy number membership function must have the following conditions:

1. It must be normal, that is, it has 1 point with membership degree 1.
2. It must be convex.
3. It must be continuous or at least partially continuous.

A fuzzy set is determined through the membership functions that the smooth curve of the membership function is more compatible with human thinking and behavior. While in a classical set, the membership function is stepwise and zero or one and cannot explain the real way of human thinking. Regarding the characteristic features' behavior as well as the desires of the system designer, different models like "trimf" (triangular), "trapmf" (trapezoidal), "sigmf" (sigmoid), "gaussmf" (gaussian), can be embedded to illustrate the membership functions. In this thesis we have applied "trimf" model for optimized trajectory planning for the mobile robot in dynamic environment.

Figure 4.15, illustrates psigmf style convex fuzzy set against a nonconvex one as an example.

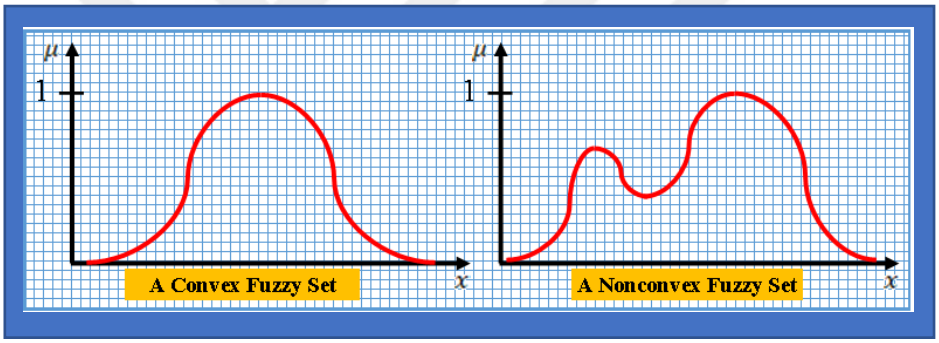

Figure 4.15- A Convex Fuzzy Set vs A Nonconvex One

4.7.8 Fuzzy-based Reasoning

An FLC-based system's the most crucial component is the reasoning block; because it is here that the mobile robot's base of knowledge and capacity to make choices are developed. There are numerous ways to develop fuzzy-based cognition and inferring.

Every rule is applied with a specific amount of weight for any inputs. Mamdani fuzzy inference model, as Min-Max, has been chosen to perform the trajectory tracking job in this thesis. In this method, the fuzzy set at the output emerges as a result of the logical "and" operation of the sets at the input. There are lots of studies in the literature discussing and comparing different approaches for determination of the output numerical value; and [123] done by Islam and Hossain in 2022 is the most recent one. Regarding their findings, centroid and weighted average methods are two of the most used defuzzification techniques. The weight average approach is also used in this thesis to determine the output numerical value.

4.7.9 Fuzzy Rules

After defining input and output variables and creating appropriate subsets and membership functions for them in the fuzzification step, now it's time to specify proper fuzzy rules for each one. Fuzzy rules are some "If – Then" conditional sentences containing logical "AND", "OR", and "NOT" expressions in the conditional part which leads to a result in the following part, and can be used to create any FL control system, like Equation 4.20:

$$\begin{aligned} &Rule\ 1: If\ S\ is\ x_1\ \&\ B\ is\ y_1, then\ C\ is\ z_1 \\ &Rule\ 1: If\ S\ is\ x_2\ \&\ B\ is\ y_2, then\ C\ is\ z_2 \end{aligned} \tag{4.20}$$

Variable C resembles the consequence for each of the rules; S and B are conditional variables indicating state and behavior; and the fuzzy parameters x_n, y_n, and z_n are outlined via membership functions.

Mamdani offered the fuzzy reasoning procedure for said two principles as well as any other membership function since the dependent parameters are frequently evaluated as 1. According to this procedure the memberships of the dependent parameters for Rules 1 and 2 mentioned in 4.20, are evaluated using I_1 and I_2 as interval ranges, resulting in $\mu_{x_1}(S)$ and $\mu_{y_1}(B)$ for Rule 1, as well as $\mu_{x_2}(S)$ and $\mu_{y_2}(B)$ for Rule 2, respectively. Following that, the associated fuzzy variables were paired with the measurements.

The controlling instructions matching $S = s$ as well as $B = b$ could be integrated into the following principles from the Fuzzy rule base as rules assessment; as it is figured out in Equation 4.21.

$$\begin{aligned} &Principle\ 1:\ \mu_1 = \mu_{x_1}(s) \wedge \mu_{y_1}(b) \\ &Principle\ 2:\ \mu_2 = \mu_{x_2}(s) \wedge \mu_{y_2}(b) \end{aligned} \tag{4.21}$$

Here, the minimum function—a Mamdani-type rule evaluation operator (∧) is used, and the mentioned principles' validity qualifications are represented by 1 and 2, accordingly.

Figures 4.16 and 4.17 illustrate the graphical view Mamdani-type assessment of the rules 1 and 2 accordingly, while $\mu_{z1}(C)$ and $\mu_{z2}(C)$ are indeed the memberships of s besides b with the corresponding fuzzy subgroups $z_1(C)$ and $z_2(C)$.

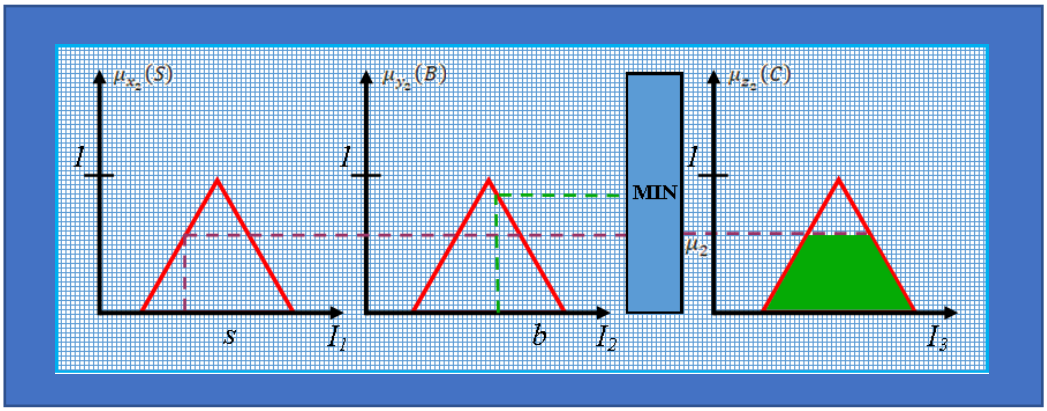

Figure 4.16- The Graphical View of Assessment for Rule 1 In Mamdani-Type Illustration

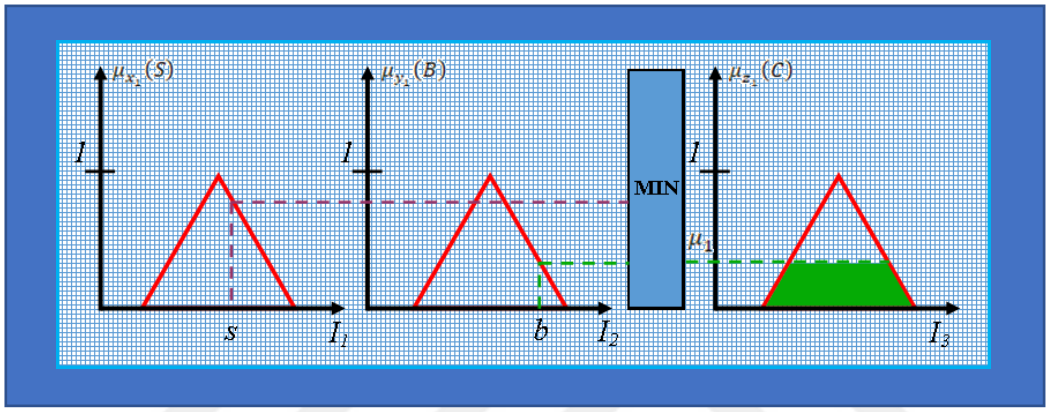

Figure 4.17- The Graphical View of Assessment for Rule 2 In Mamdani-Type illustration

The outcome $z(C)$ of the FL control model is defined as the set of all fuzzy subsets $z_n(C)$, considering interval range I_3 for the output variable. The total number of memberships makes up the output's membership, which is indicated as $\mu_{z_n}(C)$ below in Equation 4.22, while ∨ is the maximum function used to assess Mamdani-type regulations.:

$$\mu_z(C) = \mu_{z_1}(C) \vee \mu_{z_2}(C) \qquad (4.22)$$

Figure 4.18 indicates the $\mu_z(C)$ as the outcome for the both rules' assessment.

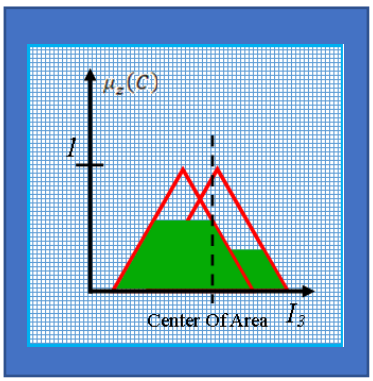

Figure 4.18- The Graphical View of Assessment for Rules1 and 2 in Mamdani-type Illustration

In table 4.7 an example of the fuzzy rules table has been provided. The rules' assessment is done based on the information provided in a table like this, and the fuzzy output for each point is based on this table, in which the rules can be assessed based on AND, OR, or NOT depending on the problem definition and the experts' idea.

Table 4.7- An Example of Fuzzy Rule-Based Table.

Input 1	**Input 2**	**Output**
Distance_to_Nearest_Obstacle	***Angle_with_Target***	***Preference_Priority***
VerySmall	VerySmall	Medium
VerySmall	Small	Small
VerySmall	Medium	VerySmall
VerySmall	Large	VerySmall
VerySmall	VeryLarge	VerySmall

Here, the rules are assumed to be assessed employing AND operator. For the first row in this table, if input 1 identified as "*Distance_to_Nearest_Obstacle*" is "very small", and input 2 identified as "*Angle_with_Target*" is "very small", then the output which is identified as "*Preference_Priority*" will take the "Medium" value. It means if the mobile robot is very close to the next obstacle, and it has a very small angle with the target the preference weight to choose the next point should be medium. Consequently, we can refer to this table showing that the intended point has a stronger preference for being picked as the next position in the route being farther from the nearest obstacle and keeping smaller angle with the final destination; so, we can add other fields to this table as is shown in Table 4.8, and we can add them to the previous table.

Table 4.8- Adding Some New Fields to the Fuzzy Rule-Based Table.

Input 1	**Input 2**	**Output**
Distance_to_Nearest_Obstacle	***Angle_with_Target***	***Preference_Priority***
Large	VerySmall	VeryLarge
VeryLarge	VerySmall	VeryLarge

This table should be field for all the possible situations for each membership function of inputs and outputs. The experts' idea plays the most important role in defining accurate and precise rules to get the best performance in the output.

4.7.10 Defuzzification

The information received from the output of the fuzzy logic controller unit of the mobile robot is fuzzy information, and needs to be rinsed (fuzzified) in order to convert it to an exact value.

First, the cluster of the output universe is utilized to determine each set of the Fuzzy outcomes, which consists of membership grades for every criterion employed known as the Fuzzy rules. The conceptual combo set produced by these sets is then subjected to one of the defuzzification procedures, and the defuzzifying procedure is carried out by locating a singular final output. The system's fuzzy logic controller will use the acquired result as its output.

As stated before, regarding different studies in the literature centroid and weighted average approaches have been used very widely in the defuzzification section to be embedded to the data gained from fuzzy inference of the mobile robot.

Employing the centroid method, the created fuzzy inference set shape will have a centroid and the value corresponding to this center is taken as the exact value as was shown in Figure 4.18 previously. In other words, a real value can be created from the fuzzy output, or the inferred consequence, to use as the entry for the Fuzzy Logic controller. The numerical quantity regarding the outcome of the Fuzzy Logic controlling model is defined from $\mu_{z_n}(C)$ that's because the expected outcome is not a fuzzy one. For defuzzification, the Centroid of Area (COA) is obtained as in Equation 4.23 below.

$$Z_{COA} = \frac{\int_z \mu_A(Z).Zdz}{\int_z \mu_A(Z).dz} \tag{4.23}$$

Here, Z resembles the contiguous and unbroken range of the Area on which the affiliation of all the resulting Fuzzy Logic subsets' members summation is represented by $\int_z$.

In the next chapter we will talk about our proposed method applied for real time trajectory tracking of a mobile robot in dynamic and unknown environment.

5. SIMULATION ENVIRONMENT AND PROPOSED APPROACH

In this chapter, first the simulation environment in being introduced, and then we will talk about our proposed approaches for the mobile robot's path planning using Ant Colony Optimization and Fuzzy Logic. The detailed procedure is explained as following.

5.1 Simulation Environment

In this section, the tools and techniques used for the simulation of the application of the thesis study are mentioned. In this section, firstly, general information is given about MATLAB, which we have used in the simulation of the algorithms developed for mobile robot's trajectory planning, and accordingly the workspace environment developed to implement the path planning algorithms on, that we provide the visualized proof for the trajectory tracking job this way. Then, the algorithm we have created for the thesis is explained with the main lines. In this part of the thesis, we will discuss the suggested Ant Colony Optimization and Fuzzy Logic to find the shortest path as a moving robot trajectory in an environment with both static and dynamic obstacles. As the environment is dynamic and there are some moving obstacles, so we need to use a real time approach to solve the problem.

5.1.1 Fuzzy Logic Based Trajectory Tracking Introduction

MATLAB has got a wide area of use, particularly in engineering and computer fields, especially in recent years. The main reasons for this are being easy and understandable, user-interactive, and suitable for developing programming and creating different algorithms. MATLAB is a package program developed by the MathWorks company and its name is an abbreviation of the words "Matrix Laboratory". MATLAB is a matrix-based program that performs simple and technical calculations, solves and analyzes complex mathematical problems very quickly, as well as has the ability to create images and graphically data and produce solutions. Although the MATLAB program was originally written in Fortran, the final versions are created in the C programming language. Because MATLAB is matrix-based, all variables and operations that are not in matrix format are converted to matrix format. Therefore, there are many ready-made functions in the MATLAB library to perform these operations. In addition, MATLAB has a very large graphics library. In this graphics library, all kinds of data or operational graphics can be created and stored in 2 or 3 dimensions depending on different variables. All desired adjustments can be made on the created graphics. One of the reasons why

MATLAB is used in different areas is Simulink and App Development plugins. With Simulink, it provides the opportunity to build many experimental studies that we have established in real life virtually, and includes control systems, generators, mathematical functions, measuring instruments, etc. It is a virtual lab with many different libraries. Although Simulink is very easy to use, it has features that provide great convenience to the user during testing of the installed systems.

5.1.2 Workspace and Implementation Environment of Thesis Study

All simulations were run on a laptop computer with an Intel(R) Core (TM) i7-4700MQ processor running at 2.40 GHz and 8GB RAM, running Windows 10, and the MATLAB R2019b software.

The same environment as shown in Figure 5.1 with the same starting and target points was considered as the moving robot trajectory environment for both ACO and Fuzzy Logic approaches. As it is shown in Figure 5.1, the starting point and the destination for the moving robot is indicated with red cross sign at (60,40) and (400,465) and also there are six static and fixed obstacles in different shapes along with 3 moving obstacles which are shown with green circles all in different sizes and moving in different directions with different speed covering most of the workspace with dynamic and static obstacles. This environment is assumed as a square workspace each side in 500 units. The step size for each movement of the mobile robot has been declared to be 5 units so there are 101 steps along each axis of x and y.

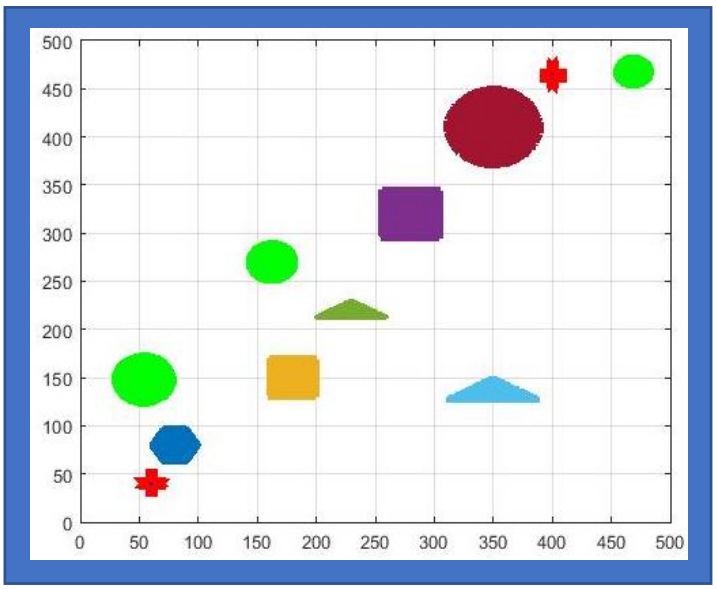

Figure 5.1- Simulation Environment for Mobile Robot Trajectory Planning

As seen in Figure 5.1, the designed workspace consists of six fixed obstacles as well as three dynamic ones. First, the coordinates of the start and end points where the route planning will be made, the sizes and shapes of the obstacles to be assigned at the workspace will be determined besides the number of obstacles in the settings. In addition, the size of the environment besides the start and end points for the mobile robot's trajectory, and obstacles are drawn as was shown in Figure 5.1, and eventually by saving this workspace consisting all the mentioned data we will be able to embed it in the algorithm we will provide later. It is the part where all the environment parameters of the developed Ant Colony and Fuzzy Logic approaches are entered and saved. After applying any desired changes to the environment such as the coordinates or the shapes or sizes of the obstacles, or also changing the starting and ending points of the trajectory, we will change the primary data settings in this part and by just running the workspace once all the new data settings will be saved and we would be able to apply it for our developed algorithms as we wish.

5.2 Proposed Approach

In this thesis study we will apply our developed Ant Colony Optimization and Fuzzy Logic approaches to find the shortest pathway as the mobile robot's optimized trajectory with the lowest cost. So, since there are some moving obstacles besides the predefined fixed ones and the environment is unknown for the robot, we should take an online and real time approach to avoid the obstacles. Besides applying the proposed ant colony and fuzzy logic, which are the main methods used for route planning, an obstacle avoidance method is used as a sub-method.

5.2.1 Obstacle Avoidance Method

The studies about multi-objective trajectory planning for mobile robots, it is aimed not to hit the determined obstacles besides finding the optimal path for road planning while going from the starting point to the target point. Therefore, as explained in detail in the previous sections, there are many methods and techniques to avoid hitting these obstacles. In the thesis, a mathematical obstacle avoidance method was used. Due to the dynamic characteristic of the environment, we need a real time obstacle avoidance approach according to the current position of the obstacles and the mobile robot.

In this thesis study, there are two general groups of obstacles divided into two categories as static and dynamic. The static obstacles category consists of six fixed obstacles in different sizes and shapes as two squares, two triangles, one circle, and one hexagon. The dynamic obstacles have been assumed as circles of various sizes moving with unique speed in different

directions from each other. First all fixed obstacles, by calculating their area according to the radius size and the shape, are added into the black list very easily. Then for the moving obstacles, considering their direction based on the starting point and the ending point of each obstacle, and also taking their velocity and size into account we can determine the black list related to them. This is done by calculating their position on the workplace as moving over the time line while the robot is planning its trajectory. As all the moving obstacle are assumed as circles of various sizes moving with different speed on different directions, by calculating their area and moving them on the workspace we will be able to determine their position in a real time approach. It is figure out better in Figure 5.2 by illustrating the moving direction and terrain of each dynamic obstacle.

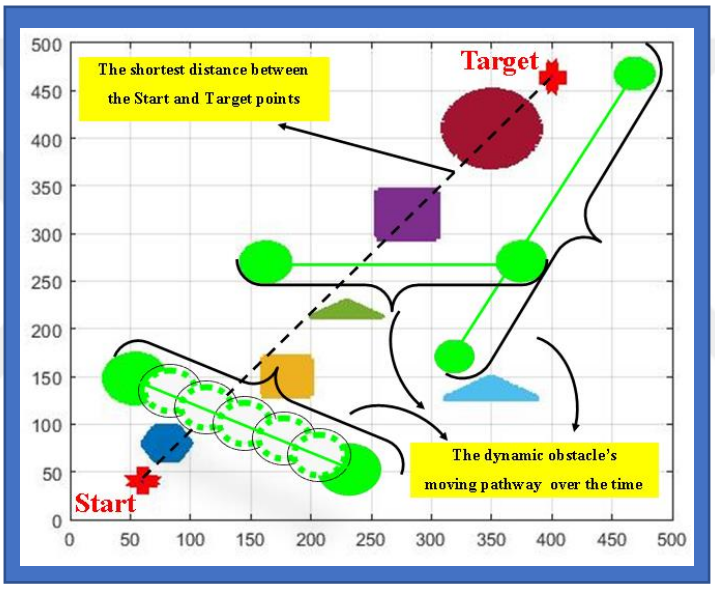

Figure 5.2- Simulation Environment for Mobile Robot Trajectory Planning

As we can understand from Figure 5.2, dynamic obstacles position change over the time, so the black list needs to be updated for each moment. As each moving obstacle is assumed to be circle, taking the center coordinates of the beginning and ending points for each obstacle's pathway into consideration, as well as velocity value we can determine the position of obstacles at each moment and this way it will be possible to update the blacklist for online path planning. For this purpose, we need to calculate total number of positions gained by each obstacle using Equation 5.1.

$$Num_{MP} = \frac{\sqrt{(x_s - x_d)^2 + (y_s - y_d)^2}}{V_o} \qquad (5.1)$$

Here, (x_s, y_s), and (x_d, y_d) indicate the coordinates of the starting and destination for each obstacle, while V_o refers to the velocity of the obstacle.

Then, it is needed to obtain the velocity value over each x and y axis according to Equations 5.2 and 5.3.

$$V_x = \frac{x_d - x_s}{Num_{MP}} \tag{5.2}$$

$$V_y = \frac{y_d - y_s}{Num_{MP}} \tag{5.3}$$

Now that we know the velocity and position of the obstacles at each moment, we can locate the obstacles over the time and know their position for real time path planning. We will use this information to update our black list by adding the moving obstacles' current positions to the black list which was containing only the data about the fixed and static obstacles before.

5.2.2 Proposed Ant Colony Algorithm Method

In this part, the proposed ant colony algorithm developed for path planning for mobile robots in dynamic and uncertain settings is included in the thesis study.

The developed ant colony algorithm consists of three basic parts. In the first part, the terrain containing the workspace information such as the positions and sizes of the obstacles as well as the speed and trajectory for the dynamic obstacles has been settled along with the algorithm's basic parameters such as the basic pheromone amount, evaporation ratio, secretion, etc. are initialized in order to determine the locations of the starting and target points for the ants, and then starting and target points and all dynamic and static obstacles are drawn according to these points. In the second part, operations are performed according to the local update of the ant colony algorithm. In the third part, although everything is similar to the second part, operations are carried out according to the global update of the ant colony algorithm depending on the pheromone update. Accordingly, the flow chart of our proposed algorithm employing Ant Colony Optimization is shown in Figure 5.3 below.

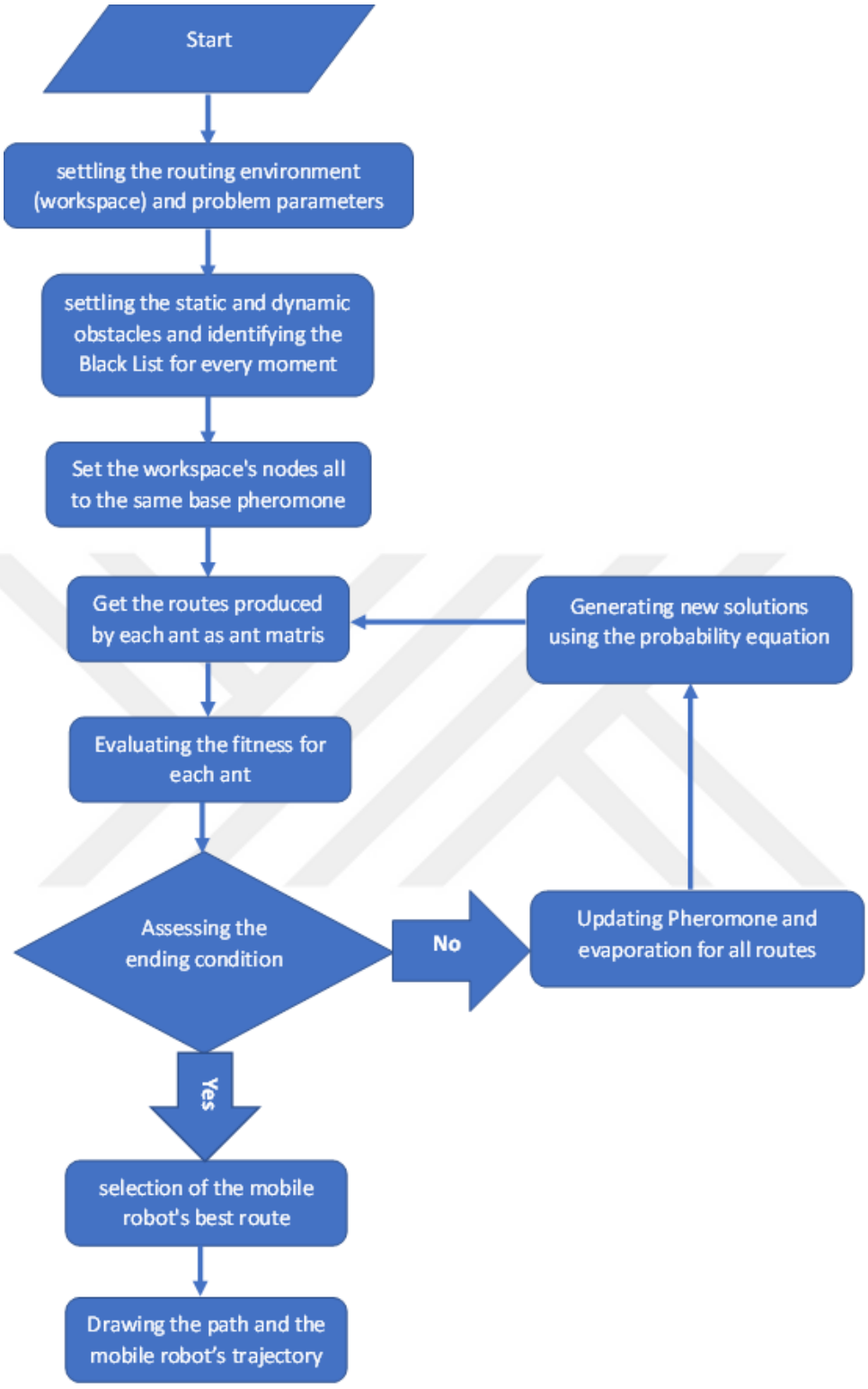

Figure 5.3- Simulation Environment for Mobile Robot Trajectory Planning

5.2.2.1 How the Proposed ACO Algorithm Works

A. Initialization – Identifying the Suitable Solutions by Ants

As stated before, in order to find a feasible route for a mobile robot in dynamic environment with static and dynamic obstacles unlike the static environment we need the path to be planned real time and according to the current position of the obstacles. To do this, in a dynamic and complex environment, a variety of starting and target points have been taken into consideration. Each ant in the ant colony algorithm will choose a path in each iteration depending on the probability equation, and eventually a solution will be generated if the path is feasible and leads to the target. Routing occurs for all sets of starting and target points that are taken into consideration. Then it's time to evaluate the fitness and cost as the length of the moving robot's path between each pair of points for assessing the ant-generated solution. The general procedure of this approach is provided in the rest of this section.

As shown in Figure 5.4 we take the eight points surrounding the ant, each one having the same distance from each other, as was mentioned as the step size, into account. Now if we imagine the ant ***k*** to be in the location ***i*** at time ***t***, it should plan a movement to the next location ***j*** which hasn't been visited yet and also is in the neighborhood of the ant ***k***.

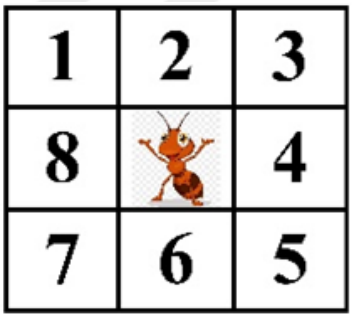

Figure 5.4- Possible Neighborhood Positions to Be Chosen for Each Ant

So, the decision to choose the next step ***j*** will be made due to the previous pheromone amount having been left before by the other ants and for the unvisited locations the probability of choosing the next position is calculated using Equation 5.4.

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t) \times \eta_{ij}^{\beta}(t)}{\sum_{s \in \text{feasible}_k} \tau_{ij}^{\alpha}(t) \times \eta_{ij}^{\beta}(t)} & j \in feasible_k \\ 0 & otherwise \end{cases} \qquad (5.4)$$

In this equation α is used to fix the importance of pheromone level in route choosing; and β is used to fix the heuristic information (availability) in the route choosing; and the list of potential nodes in the neighborhood from which ant ***k*** may select is known as $\boldsymbol{feasible_k}$ which

doesn't belong to the ***Black List***; while $\boldsymbol{\eta_{ij}}$ is the heuristic information in the $\boldsymbol{i^{th}}$ point which shows the availability of ***ij*** line and is obtained using Equation 5.5.

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{5.5}$$

In Equation 5.5 above, $\boldsymbol{d_{ij}}$ is the length of the edge ***ij*** which is calculated with the Euclidean distance between two neighbor points as the Equation 5.6.

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{5.6}$$

B. Fitness – Evaluating the Produced Solutions

For all the population size in each main loop, once each ant has created their own solutions, and then the routes are all assessed using Equation 5.7, and the best ant is chosen.

$$Cost_k^t = \frac{\sum_{i=1}^{N} L_i^k}{N} \tag{5.7}$$

In this equation ***N*** indicates on the number of routes in the routing environment; $\boldsymbol{L_i^k}$ shows the length of the obtained route for the $\boldsymbol{i^{th}}$ pair of starting and target points for the $\boldsymbol{k^{th}}$ ant.

C. Updating the pheromone

After completing the initialization loop for all the population ants to search their solution searches, now it's time to update the pheromone amount. This will happen in accordance with the evaporation ratio of the released pheromone and new generated pheromone by each ant according to Equation 5.8 below.

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \Delta\tau_{ij} \tag{5.8}$$

In equation 5.8 above, $\boldsymbol{\tau_{ij}(t)}$ is used to show the pheromone amount on the path ***ij***; and $\boldsymbol{\tau_{ij}(t+1)}$ indicates on the pheromone amount on the track ***ij*** released in the most recent iteration; and $\boldsymbol{\Delta\tau_{ij}}$ indicates on the pheromone quantity changes in the most recent loops and is obtained using Equation 5.9.

$$\Delta\tau_{ij} = -\lambda\tau_{ij}(t) + \begin{cases} \frac{Q}{Pnum} & line\ (i,j)\ is\ chosen\ by\ the\ best\ ant \\ 0 & otherwise \end{cases} \tag{5.9}$$

In Equation 5.9 above, $\boldsymbol{\lambda}$ is the pheromone evaporation ratio; $\boldsymbol{Pnum}$ shows the Priority Number of ants among candidates for releasing pheromones on the route – which gives us the ants' priority to choose a route – chosen by the best ant; and $\boldsymbol{Q}$ is the quantity of pheromones left on the path in accordance with the best ants' route.

The suggested algorithm's general pseudocode is shown in Algorithm 1.

Algorithm 1. The proposed ACO

```
Upload workspace
Set the initial values
Backlist ← The position and size of static obstacles
Parameters ← initial values
For each population
   Locate ants
   Black List ← Fixed Obstacles + Current Position
   While (X_Current, Y_Current) ≠ (X_goal, Y_goal)
      Black List ← Black List + Moving Obstacles
      Locates ants on the paths
   End While
End For
For each iteration
   For each population
      If path is valid
         Evaluate the Fitness
         implement probabilities
         Update Black List
    End For
    Selected ants ← Best Fitness ants
    Ph ← Ph – (1 – damp_rate)
    For selected ants
       Update Pherpmone (Ph)
     End For
    Ensure: Ph_low ≤ Ph≤ Ph_high
    Generate new ants
    While (X_Current, Y_Current) ≠ (X_goal, Y_goal)
    For each ant
        If ant is active
           Update Local Pheromone
           implement probabilities
           Update Black List
      End For
     Update the best solution
     Draw the trajectory
End For
```

Based on the information mentioned earlier, the moving robots' routing problem has been solved employing the step-by-step process to utilize the ACO approach as indicated in Algorithm 1. Regarding the termination criteria, provides the evidence of being convergent for the ACO approach for this dynamic routing problem, however, these results fail to obtain the requirements of being time convergent. Setting a time limit during algorithm implementation

is necessary to apply the termination condition and find the optimal answer. In this study, the Ant Colony Optimization algorithm was tested numerous times with various parameters to reach optimal precision. In the end, the best parameter values for the algorithm were chosen as it is in Table 5.1.

Table 5.1- Ant Colony Optimization Parameters Setting.

ACO Algorithm Parameters	Values
Iteration's Loop	30
Population Size	30
Candidate Ants' Num	5
Base Pheromone Amount	0.5
Evaporation Ratio	0.1
Deposition Rate	0.3
Coefficient for pheromone Trailing (α)	1
Heuristic Coefficient to show the Availabality (β)	0.5

5.2.3 Proposed Fuzzy Logic Approach

As was stated in the previous sections, Fuzzification, rule evaluation, and defuzzification are the three phases that can be taken to implement the FL control. In this section of the study, we want to apply Fuzzy Logic to an unidentified and complex environment to pick the shortest route as a moving robot's trajectory in a workspace with both static and dynamic roadblocks. Accordingly, the flow chart of our proposed algorithm using Fuzzy Logic Approach is shown in Figure 5.5 below.

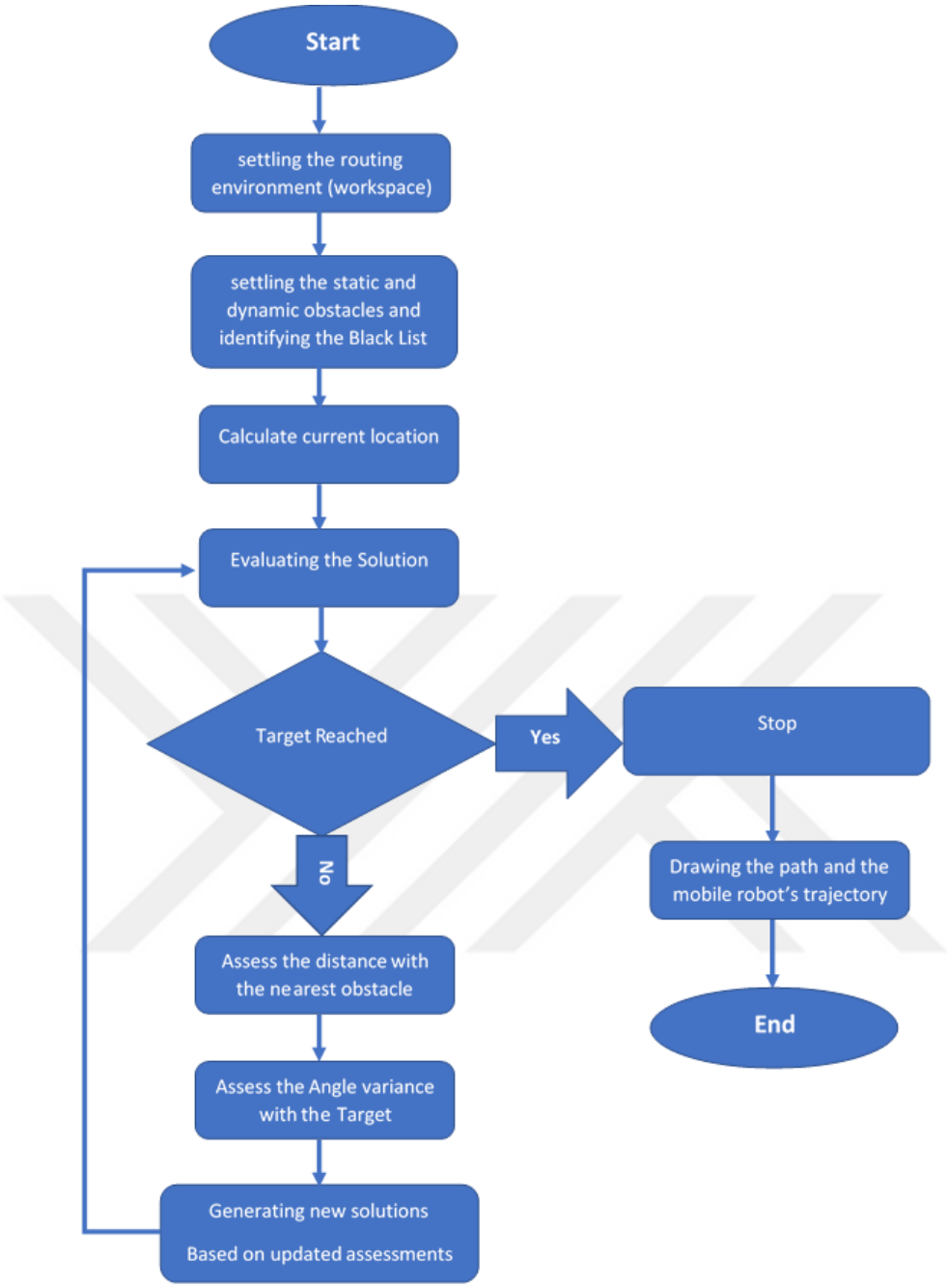

Figure 5.5- The Proposed Fuzzy Logic Algorithm's Overall Step by Step Process Flow

As the environment is dynamic and there are some moving obstacles, we need to use a real time approach to solve the problem. In other words, here we don't have a predefined map to make the autonomous robot know about the placement and locations of the obstacles and also as the location of moving obstacles are changing over the time so we need to have an online decision-making approach special for the current position of the robot and this can be possible only by a node-to-node process of routing. The next position is always selected in accordance with the eight possible positions, located around the current position of the robot, in order to

have optimality criteria for the dynamic unknown environment in the workspace with both static and dynamic obstacles as indicated in Figure 5.6.

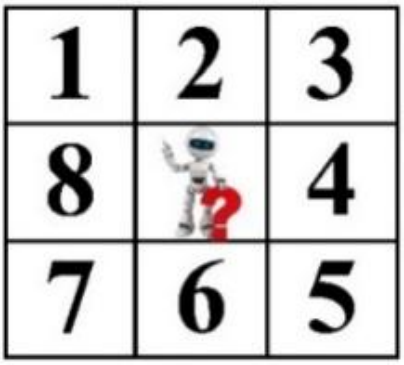

Figure 5.6- Possible Neighborhood Positions to Be Chosen as the Next Position

Among the surrounding neighbor positions, each of the eight nodes is selected based on two factors: The distance to the closest obstacle and the difference in angle with the target. For each of the eight nodes, the objective is to obtain a priority index for the following position. Following the use of fuzzy logic, a non-fuzzy or final output is produced for each node's inputs. Thus, the next position to choose will be the one having the highest non-fuzzy priority index. Figure 5.7 illustrates our proposed Fuzzy Logic system in this paper as follows.

Figure 5.7- The Illustration of the Suggested Fuzzy Logic for Mobile Robot Path Planning

Input criteria 1 indicates the difference in angle with the target while input criteria 2 shows the distance to the closest obstacle in the workspace.

5.2.3.1 How the Proposed Fuzzy Logic Approach Works

A. Fuzzy Logic Optimization Inputs

The embedded Fuzzy logic uses point-to-point path selection and considers the number of various points along the way. In this study, as it was illustrated in Figure 5.6, the rotation variance (between the angles) with the final goal and the closest hurdle’s remoteness were employed as the two fuzzy inputs to identify the path. Naturally, along with the trajectories (moving path), the size, and the angle are also considered in each of the five input membership functions that have been fuzzified. By calculating the route’s size and its trajectory heading

regarding the destination, it connects the current position to the ending point. To optimize and shorten the lateral process, the angle difference has been chosen as the most ideal choice. Both inputs defined as Angle and Distance have membership functions as shown in Figure 5.8 and Figure 5.9 below. In this study first we normalize the non-fuzzy inputs to apply them to the eight neighborhood points and then the fuzzification process occurs. As is shown in Figure 5.10, we also have employed five membership functions for the output as very small, small, medium, large, and very large. In other words, the non-fuzzy inputs as angle difference with the destination and the nearest hurdle's remoteness, at first get adjusted in the range between zero and one which is shown as [0, 1] in each iteration, and from there, the membership of various fuzzy functions is generated. At this section of this study selecting the next point was accomplished by employing Fuzzy Logic. Thus, in each position of our environment lattice, the smaller the angle between a point and the goal, and the further distant is the point from the nearest obstacle, the current position is more likely to be selected as the next position by fuzzy possibility in the routing process, gaining higher-priority coefficient rather than the other points.

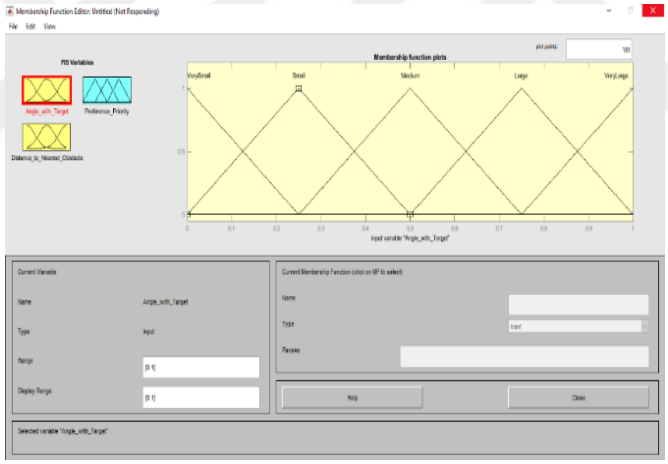

Figure 5.8- Angle_with_Target Input Membership Functions

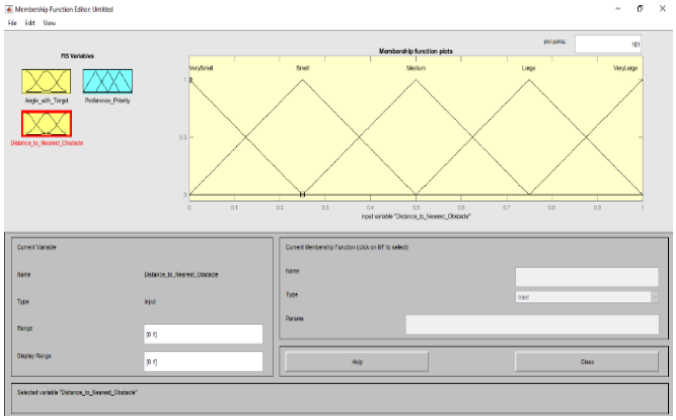

Figure 5.9- Distance_to_Nearest_Obstacle Input Membership Functions

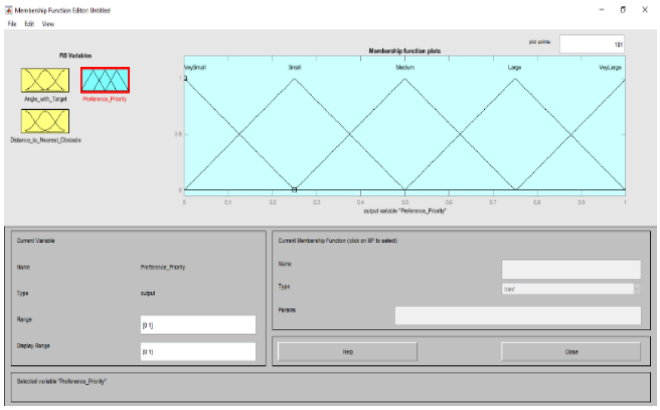

Figure 5.10- Preference_Priority Output Membership Functions

B. Fuzzy Rule-Base Table

As previously stated, the next point on the route is chosen employing Fuzzy Logic. Table 5.2 indicates the Fuzzy Rules for the preference implying the next node to be chosen real time based on the position of the moving robot according to the next point and the target. In other words, we can refer to this table showing that the intended point has a stronger preference for being picked as the next position in the route being farther from the nearest obstacle, comparing with the other neighborhood nodes, and keeping smaller angle with the final destination. As previously explained, the table of fuzzy rules for assessing the fuzzy output for each point is based on the Table 5.2, in which all rules have been assumed as AND. The Mamdani method is applied in the following step to acquire the output fuzzy shape. As a result, every node now runs a Fuzzy Rule-Base table at this phase, and then one or more rules from Table 5.2 below are executed based on the fuzzy input membership functions related to the points.

Table 5.2- Fuzzy Rule-Base Table for online motion planning.

Input 1	**Input 2**	**Output**
Distance_to_Nearest_Obstacle	***Angle_with_Target***	***Preference_Priority***
VerySmall	VerySmall	Medium
VerySmall	Small	Small
VerySmall	Medium	VerySmall
VerySmall	Large	VerySmall
VerySmall	VeryLarge	VerySmall
Small	VerySmall	Medium
Small	Small	Medium
Small	Medium	Small

Input 1	Input 2	Output
Distance_to_Nearest_Obstacle	***Angle_with_Target***	***Preference_Priority***
Small	Large	Small
Small	VeryLarge	VerySmall
Medium	VerySmall	Large
Medium	Small	Medium
Medium	Medium	Medium
Medium	Large	Small
Medium	VeryLarge	VerySmall
Large	VerySmall	VeryLarge
Large	Small	VeryLarge
Large	Medium	Large
Large	Large	Medium
Large	VeryLarge	Small
VeryLarge	VerySmall	VeryLarge
VeryLarge	Small	VeryLarge
VeryLarge	Medium	Large
VeryLarge	Large	Medium
VeryLarge	VeryLarge	Medium

C. C. Defuzzification

As previously stated, Centroid and weighted average methods are two of the most used defuzzification techniques regarding the studies held in the literature. In this thesis, the weighted average method has been applied for overall defuzzification; eventually, the choice with the highest non-fuzzy result is chosen as the next location in the route once the non-fuzzy results for all eight goal alternatives have been evaluated. Till the mobile robot plans the trajectory and achieves its destination, this operation goes on.

6. EXPERIMENTAL RESULTS

The solution of path planning problems for mobile robots is generally done in static environments. In other words, there are only fixed obstacles whose geometry is known in the workspace to be planned. The fact that the obstacles are fixed gives better results for the solution of the road planning problem, and it also provides convenience in terms of implementation. The parameters such as the geometric shape, radius, length and coordinate system of the factors such as obstacle, start-end points in static circles must be determined before starting the problem and these parameters remain constant throughout the problem. Though there seems to be a great demand on developing online approaches to do the trajectory tracking for dynamic environments as well.

In the thesis study we have done, different obstacles in various shapes and features such as static and dynamic, each one having unique features as speed and moving direction, have been created for the area where the routing task will be done. All the features such as the radius, size, shape, speed, and the number of obstacles of each shape assigned to the created terrain are different. After the obstacles were created, both the ant colony algorithm and Fuzzy Logic Optimization were run by changing the parameters and evaluations were made on the path planning results obtained.

As the parameter values of the proposed ant colony algorithm, the number of ants is given differently for each test. 10 different values were determined for pheromone Trailing (α) and Heuristic Coefficient to show the Availability (β) parameter values. In order to determine the evaporation coefficient value, tests and trials were carried out for evaporation coefficient values ranging from 0 to 1 in 10 different values determined for α and β in different experiments. As a result of these tests, the best values for α and β were decided to be 1, and 0.5 respectively, while the evaporation ratio was determined as 0.1 for dynamic path planning problem, and the base pheromone amount value was determined as 0.5. However, regarding the tests, the value of the deposition rate was determined as value of 0.3. Table 6.1 indicates the most appropriate values gained to be assigned as the algorithm's parameters.

Table 6.1- Ant Colony Optimization Parameters Setting.

ACO Algorithm Parameters	Values
Iteration's Loop	30
Population Size	30
Candidate Ants' Num	5
Base Pheromone Amount	0.5
Evaporation Ratio	0.1
Deposition Rate	0.3
Coefficient for pheromone Trailing (α)	1
Heuristic Coefficient to show the Availability (β)	0.5

All simulations were run on a laptop computer with an Intel(R) Core (TM) i7-4700MQ processor running at 2.40 GHz and 8GB RAM, running Windows 10, and the MATLAB R2019b software. The same environment as shown in Figure 6.1 with the same starting and target points was considered as the moving robot trajectory environment and consequently, obtained paths by the proposed algorithms in this workspace are mentioned briefly in the related tables in the following sections. As it is shown in Figure 6.1, the starting point and the destination for the moving robot is indicated with red cross sign at (60,40) and (400,465) and also there are six static and fixed obstacles in different shapes along with 3 moving obstacles which are shown with green circles all in different sizes and moving in different directions with different speed covering most of the workspace with dynamic and static obstacles. This environment is assumed as a square workspace each side in 500 units. The step size for each movement of the mobile robot has been declared to be 5 units so there are 101 steps along each axis of x and y.

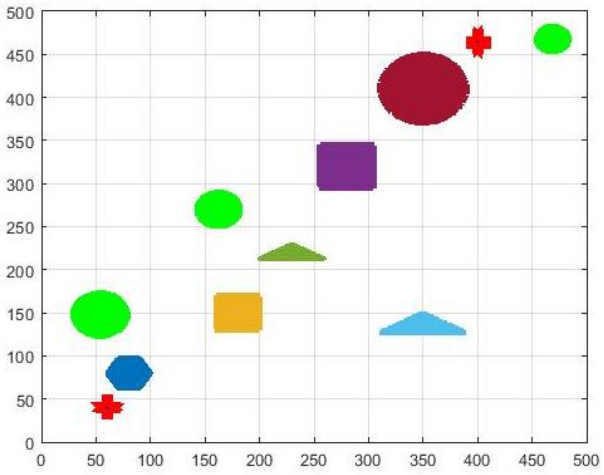

Figure 6.1- Simulation Environment for Mobile Robot Trajectory Planning

6.1 Test Experiments and Results

In order to evaluate the performance of the proposed ACO and Fuzzy Logic approaches for this thesis study, the same environment was used for both of them as the workspace which was shown in 6.1. The coordinates of the points determined as the positions of the obstacles all are between 0 and 500 limit values for the x and y axis. The coordinates of the obstacles of the terrain and the starting and target points were determined manually. The coordinates of the obstacles and trajectory starting and ending points within the created terrain do not exceed these limit values. The specialties of the environments are as seen in Tables 6.2 and 6.3 below.

Table 6.2- Static Obstacles of the Trajectory Terrain.

Shape	number	Position	Radius Size
Square	2	(180,150)	40
		(280,320)	50
Triangle	2	(230,230)	60
		(350,150)	80
Circle	1	(350,410)	40
Hexagon	1	(80,80)	20

Table 6.3- Dynamic Obstacles of the Trajectory Terrain.

Shape	number	Starting Position	Ending Position	Radius Size	Velocity
Circle	1	(160,270)	(370,270)	20	2.5
Circle	1	(470,470)	(320,170)	15	2.5
Circle	1	(50,150)	(230,60)	25	4.5

6.1.1 Ant Colony Optimization Algorithm Performance and Results

Figure 6.2 shows some screen shots of some of the iterations of the proposed ACO algorithm to plan the shortest and optimized path for the moving robot in the mentioned environment containing static and dynamic obstacles. As it is shown at first the ants draw a non-optimized route and the optimization happens in the next iterations. As Ant Colony assessment is more suitable for offline path planning without any dynamic obstacles it takes more time to calculate and evaluate the feasible solutions to find an optimized and smooth route for the moving robot trajectory. The algorithm parameters were set for 30 iteration, and 30 population size, and τ=0.5 as the initial pheromone value which gets updated later by 0.05 damping rate and 0.3 new pheromone release for the best ants of each iteration.

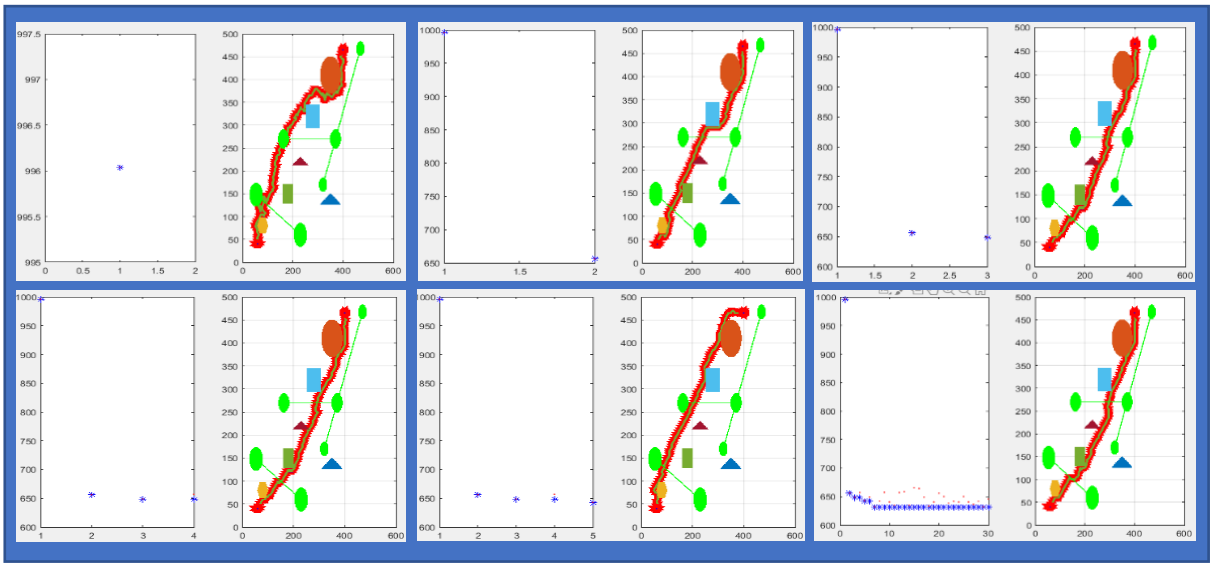

Figure 6.2- Path Planning Results Employing Ant Colony Optimization in Some of the Iterations

As a summary, we can infer to Figure 6.3, as the final evaluation and trajectory of the mobile robot after all the iterations. As it is shown in Figure 6.3, there is a nearly forty percent optimization in the second iteration and most of the optimization is done by the 7th iteration. The Best cost we gained through this proposed algorithm was between 612.0950 and 640.2598 in 30 times running the algorithm. Also, the final cost we got for this running was 632.1930 for the same parameters values as mentioned before. In this figure the best cost of each iteration is mentioned with red dots on the left panel, while the global best cost is indicated by blue stars. On the right panel, the final optimized global path is drawn along with static obstacles besides the moving obstacles and their related trajectories in the routing environment.

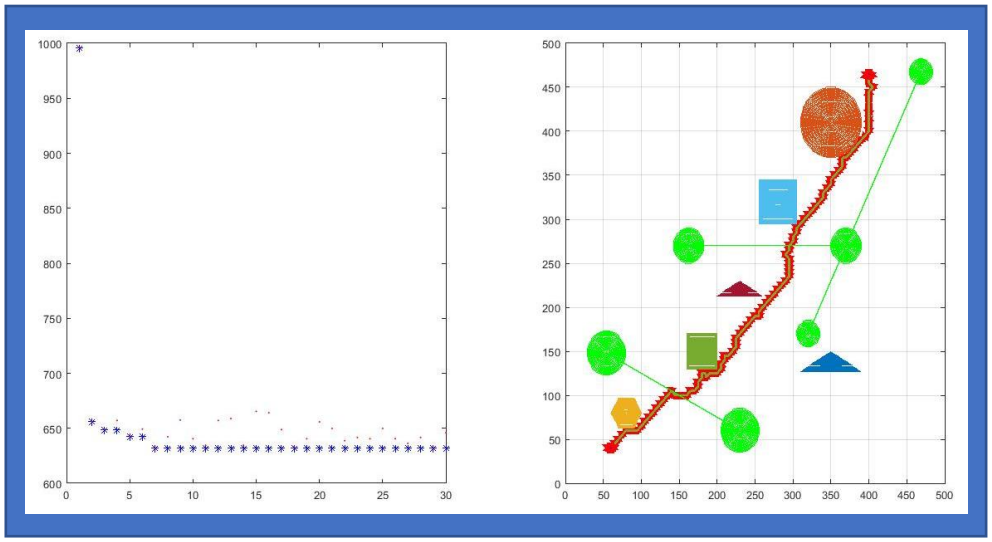

Figure 6.3- Final Evaluation and Trajectory for Mobile Robot Using ACO After 30 Iterations

After the final assessment and drawing the final trajectory based on offline map of static obstacles and also considering the real time changes of the dynamic obstacles now it's time to run the mobile from the starting point to the destination following the optimized route found by ACO as is shown in the Figure 6.4 below.

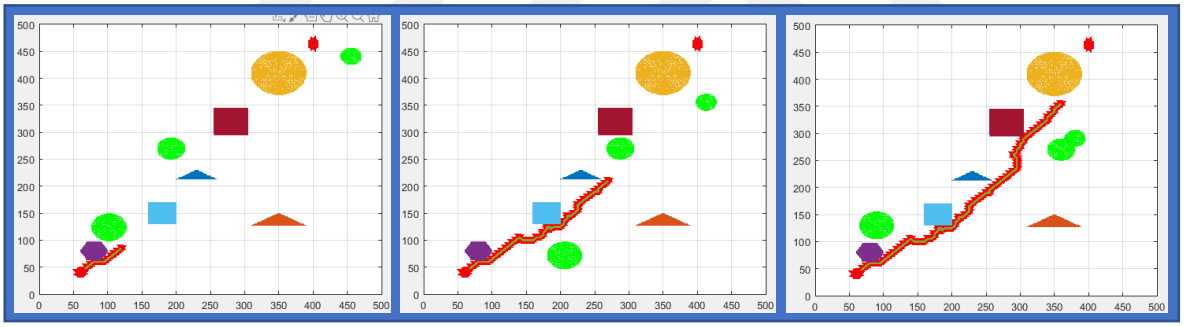

Figure 6.4- Real-Time Collision free Mobile Robot Trajectory Using ACO

Figure 6.5 shows the complete form of the achieved route by the proposed ACO which shows that the final route is optimized, smooth, and also collision free. Figure 6.6 shows the proposed ACO algorithm's convergence which indicates on the best cost optimization over the 30 iterations.

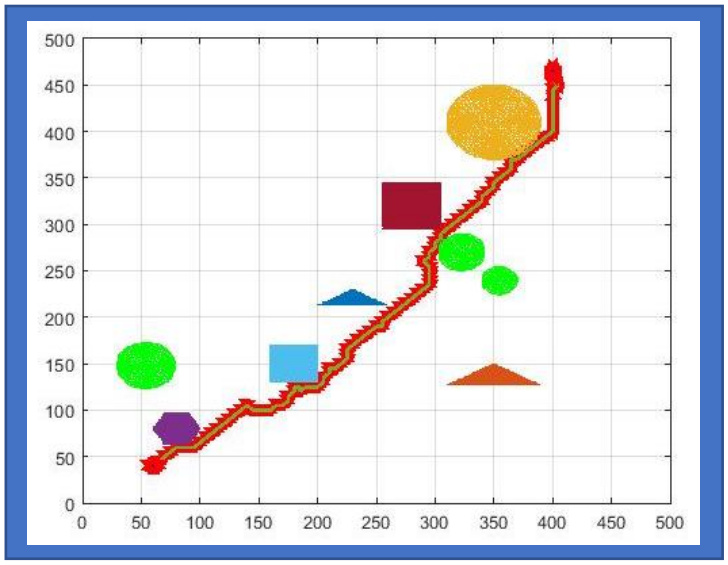

Figure 6.5- Final Optimized Collision Free Trajectory Using ACO

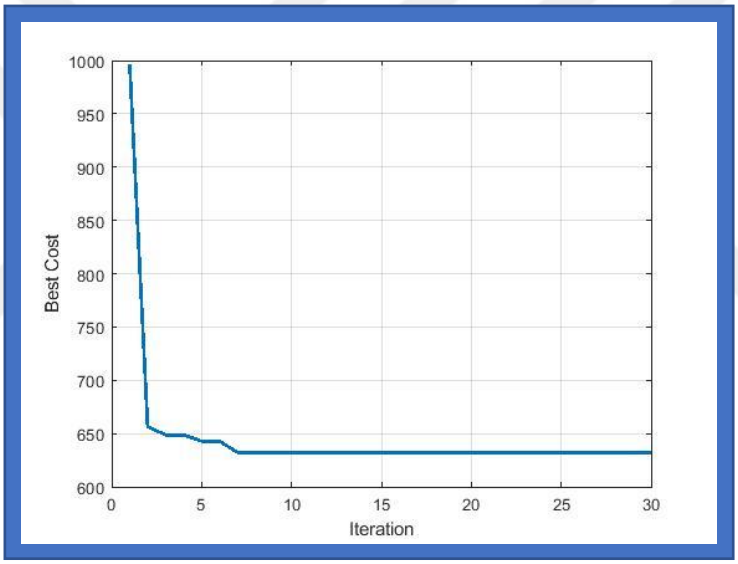

Figure 6.6- Optimization Convergence Employing Proposed ACO Algorithm

The average results gained employing ACO for online trajectory tracking is briefly mentioned in Table 6.4

Table 6.4- Trajectory Achieved by Ant Colony Optimization.

Implementation Environment	**Trajectory Results employing ACO**				
	Start	***Target***	***Average Best Cost***	***Self Time***	***Total Time***
500 x 500 Unknown Dynamic	(60,40)	(400,465)	632.1930	437.896 s	472.769 s

Note: The amount of time spent performing a function that is not included in the time spent performing its child functions, is called Self-Time. Self-Time additionally accounts for overhead associated with profiling.

6.1.2 Fuzzy Logic Performance and Results

The simulation for mobile robot trajectory employing Fuzzy Logic Approach has been done on the same laptop computer and also the same environment map as workspace. For path planning employing Fuzzy Logic approach, the robot will travel through its route from starting point to the finishing point without colliding any static or dynamic obstacles. When getting closer to each dynamic or static obstacle, the trajectory, besides smoothing and optimization for each position, chooses the shortest route. When the dynamic obstacle is moving in a special direction getting closer to the robot, by changing its direction regarding the Fuzzy input parameters at that special time the robot decides to choose a new position to go to and continue its journey to the goal. To prevent collisions, the moving robot alters the previous route and, doing a route seek, chooses the route employing Fuzzy Logic approach and the trajectory among the fixed and moving obstacles goes on for the mobile robot to achieve the desired goal; optimizing its trajectory to gain the shortest path available. Figure 6.7 shows some screen shots indicating a detailed real time trajectory of the moving robot showing different motions while planning its trajectory dealing with static and dynamic obstacles.

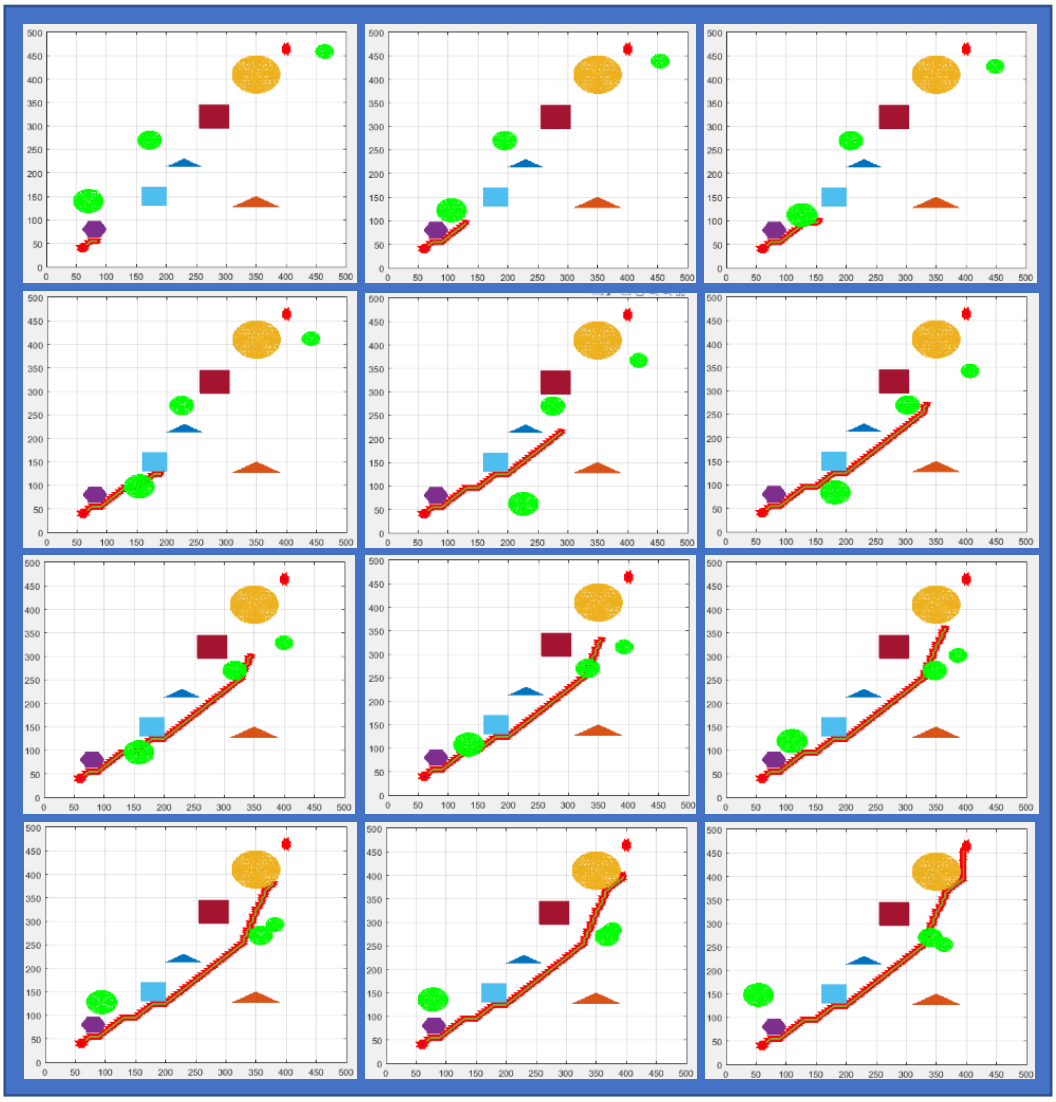

Figure 6.7- A Detailed Real Time Trajectory of the Mobile Robot Based on Fuzzy Logic

We can see the final trajectory of the mobile robot employing Fuzzy Logic approach in Figure 6.8. The final cost in this real time trajectory was 598.0509 which is lower than the final cost gained by ACO algorithm. The running time was significantly shorter and also the obtained path was smoother and better optimized.

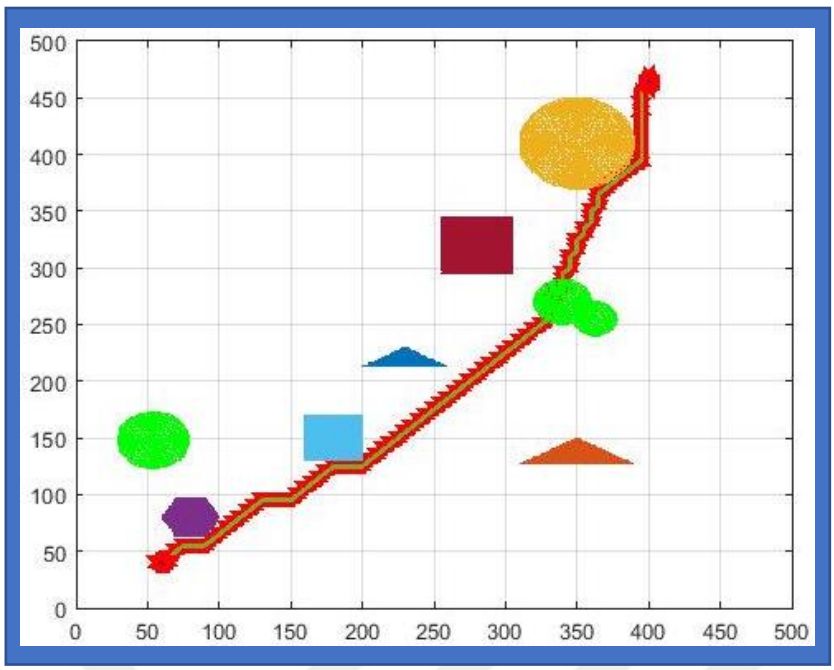

Figure 6.8- Final Collision Free Trajectory for Mobile Robot Based on Fuzzy Logic Approach

A priority index for preference is acquired for every one of the eight choices available around a node to determine the next position in trajectory tracking with Fuzzy Logic, as explained before. The algorithm should then be applied to the series of various nodes on the route. Consequently, a real time routing table is generated after several run-sorted for motion planning as is shown by Table 6.5. In other words, this table was manually modified to enhance the routing. After running 20 times, the customized Fuzzy Rule-Based table has been reformed, and the optimized path planning in Figure 6.7 is achieved for mobile robot fixing the parameters deliberately 20 times as shown in Table 6.5. Figure 6.8 also shows an overall perspective of the final collision free trajectory of the mobile robot based on this method.

Table 6.5- Fuzzy Rule-Base Table for online motion planning regarding to the node-to-node preference priority.

Input 1	**Input 2**	**Output**
Distance_to_Nearest_Obstacle	***Angle_with_Target***	***Preference_Priority***
VerySmall	VerySmall	Medium
VerySmall	Small	Small
VerySmall	Medium	VerySmall
VerySmall	Large	VerySmall
VerySmall	VeryLarge	VerySmall
Small	VerySmall	Medium
Small	Small	Medium

Input 1	Input 2	Output
Distance_to_Nearest_Obstacle	***Angle_with_Target***	***Preference_Priority***
Small	Medium	Small
Small	Large	Small
Small	VeryLarge	VerySmall
Medium	VerySmall	Large
Medium	Small	Medium
Medium	Medium	Medium
Medium	Large	Small
Medium	VeryLarge	VerySmall
Large	VerySmall	VeryLarge
Large	Small	VeryLarge
Large	Medium	Large
Large	Large	Medium
Large	VeryLarge	Small
VeryLarge	VerySmall	VeryLarge
VeryLarge	Small	VeryLarge
VeryLarge	Medium	Large
VeryLarge	Large	Medium
VeryLarge	VeryLarge	Medium

Below in Table 6.6, we can see the results in the same environment with the same starting and ending point, so that when the robot encounters static obstacles, the proposed algorithm with Fuzzy Logic Optimization implements the priority preference for relocating from the current position and picking the smallest straight transition that will take the minimum amount of time to get to the target point in an unknown environment. As shown below, manually adjusting the rules of the Fuzzy table after 20 times affects the distances to be shorter, and consequently altering the Rule-Base table results in routes that are short, smooth, and optimized thanks to the Fuzzy Logic. In other words, changes in the parameters of the Rule-Base table and Fuzzy Optimization, produce minimal distance differences. As a result, the total trajectory is optimal (a short route) at a dynamic unknown environment.

Table 6.6- Trajectory Achieved by Fuzzy Optimization.

Implementation Environment	**Trajectory Results**				
	Start	***Target***	***Fuzzy Optimization***	***Self Time***	***Total Time***
500 x 500 Unknown Dynamic	(60,40)	(400,465)	598.0509	6.452 s	15.568 s

Note: Self time is the time spent in a function excluding the time spent in its child functions. Self time also includes overhead resulting from the process of profiling.

Employing a real time routing approach in our Fuzzy Logic – based method, the processing time has been gained; as shown in Table 6.5 above. Although the moving robot lacks sufficient and precise information about the dynamic environment, it is able to find the shortest and also the smoothest route very quickly and consequently moves along the route spending very short time avoiding any crash.

Looking at the trajectory figures, proposed ACO approach doing a local planning and taking a node to node approach regarding its eight neighbor positions first plans the possible pathways avoiding any obstacles, and then by global planning improves the solutions found and makes the route between the starting and target points optimized and smoother. As seen in Figure 6.2, although the rout obtained in the first iteration is a possible solution but it is not optimized yet. In local planning, the routes colliding the obstacle get an inf value in cost calculation. In other words, there are some situations where there is contact with the obstacle in the optimal result obtained for local planning. By eliminating them in global planning, on the other hand, we got better results in 30 iterations by obtaining the optimal value without collision with the any obstacles in the terrain and obtained nearly optimal solutions.

Fuzzy Logic on the other hand, taking an approach similar to human behavior tries to find the optimized trajectory real time. It is very important to define efficient inputs and related membership functions should be assigned and manipulated very carefully to get the optimized result.

Regarding the results mentioned in the tables and trajectory visualization for each of the proposed approaches, and based on the experiments till now in this study comparing Ant Colony Optimization with the Fuzzy Logic approach which has been briefly brought in the table 6.7, it is inferred that Fuzzy Logic approach acts better than ACO considering the process time, trajectory length, path smoothness, and real time decision making. Best Cost and spent time amount are the average of 30 times running the Algorithm.

Table 6.7- Trajectory Achieved by Ant Colony Optimization and Fuzzy Logic Optimization.

	Ant Colony Optimization		Fuzzy Logic Optimization	
Average Best Cost	612.0295		598.0509	
Time for assessment and trajectory Evaluation	Self Time	437.896 s	Self Time	6.452
	Total Time	472.769 s	Total Time	15.568 s
Elapsed Time for trajectory trailing		19 s		12 s

7. DISCUSSION AND CONCLUSION

In this thesis study, it was aimed to solve the trajectory tracking problem for mobile robots in dynamic environment with both static and moving obstacles, which are in the NP-Hard problem class. An ACO and FLO are proposed to solve this path planning problem. A node to node approach was taken into consideration in order to choose a collision free pathway. A mathematical obstacle avoidance method is used in the ACO algorithm to avoid Collision with obstacles as well. According to this method, it is aimed to increase the value of road lengths by changing the cost value to infinite when there is contact with obstacles on the pathway and thus to eliminate the probability of choosing these roads.

The ACO proposed by us for the thesis study has been tested along with Fuzzy Logic for the same environment. Roadblocks are created as fixed and moving ones by changing the radii and number of obstacles and the starting target points for the moving obstacles and the mobile robot. The results were obtained by using various parameter values of the recommended ACO in numerous runs, and the Ant Colony Optimization algorithm was tested numerous times with various parameters to reach optimal precision. In the end, the best parameter values for the algorithm were chosen. For the Fuzzy Logic, on the other hand, manually adjusting the rules of the Fuzzy table after numerous times affects the distances to be shorter, and consequently altering the Rule-Base table results in routes that are very near to the Fuzzy Optimization outcomes. In other words, changes in the parameters of the Rule-Base table and Fuzzy Optimization produce minimal distance differences. As a result, the total trajectory is optimal (a short route) at a dynamic unknown environment regarding the obtained solutions. According to the results, the relative errors were calculated and the running times were calculated for all parameter values in the generated environment. The obtained results were tabulated and these tables were used to evaluate the performance of the algorithm.

Comparing the results of the trajectory planning solutions made embedding proposed approaches to the created settings, it is seen that Fuzzy Logic achieves better results than ACO due to the run time and the obtained solution's quality. Although our ACO was able to optimize the solutions found, and plan a smooth and short path in a good quality, but FLO on the other hand was able to obtain better results in a very little time. The results regarding the quality of each algorithm's implementation were tabulated and according to those results we saw there is a huge difference in run time between the two approaches. The self time for ACO, which is

defined as the time spent in a function that does not include time spent in its child functions, was 67.86 times more than FLO. In the same way the total time for ACO was 30.36 times more than FLO.

As a result, path planning, one of the crucial objectives of mobile robots, has been studied in this research paper. Here we compared the performance of an improved evolutionary algorithm known as Ant Colony Optimization algorithm versus Fuzzy Logic approach in real time path planning for a mobile robot in unknown dynamic environment. The process of choosing an optimal path with the lowest cost for a moving robot to follow through different environmental challenges – which are mentioned here as fixed and moving obstacles – is known as path planning. Fuzzy Logic approach having the advantage of its real time node to node trajectory and surveillance efficiency can be a very excellent choice for dealing with path planning issues to get the shortest route meanwhile avoiding colliding, which can be claimed to be the best optimized trajectory solution in such environments considering the presence of fixed and moving obstacles all in various shapes and forms moving in different directions with different speed; or in scenarios in which the robot has only a little or no information beforehand to understand the trajectory environment, which seems to be one of the great concerns of the real-world trajectory problems. Regarding The theoretical foundation of the Fuzzy Logic, the initial rules are obtained from an expert and then altering a number of its criteria the final rules get extracted and the final Fuzzy Logic gets outlined. Allowing the mobile robots to find the fastest and most efficient path in unknown dynamic environments containing any kinds of static or dynamic obstacles in any size and shape makes this part very important and pivotal.

As future studies related to this thesis, developing a 3-D trajectory terrain can be thought of to implement the proposed approaches in this study. However, it is possible to add image processing to this project to improve the studies regarding this thesis, and also by creating new environment having more complexities the proposed approaches can be examined for path planning with more obstacles besides assuming dynamic obstacles having different geometric shapes. Combining these two approaches on the other hand can be a novel idea to get better solutions in different cases.

REFERENCES

[1] D. Çamoğlu, Bilgisayar kontrollü Robotik, Ankara, Turkey: Dikey Eksen, 2011.

[2] O. Okutkan, "Yapay Zeka ile Mobil Robot Kontrolü (Doctoral dissertation, Fen Bilimleri Enstitüsü)," Isatnbul technical University, Istanbul, 2006.

[3] S. Geetha, G. M. Chitra and V. Jayalakshmi, "Multi objective mobile robot path planning based on hybrid algorithm," *2011 3rd International Conference,* pp. 251-255, 2011.

[4] J. Deneubourg, P. Clip and S. S. Camazine, "Ants, Buses and Robots-Self-Organization of Transportation Systems," *From Perception to Action Conference,* pp. 12-23, 7-9 September 1994.

[5] J. A. Sauter, R. Matthews, H. V. D. Parunak and S. Brueckner, "Evolving Adaptive Pheromone Path Planning Mechanisms," *Proceedings of the First International Conference on Autonomous Agents and Multi-Agent Systems,* pp. 434-440, 2002.

[6] F. H. Jin, B. R. Hong and Q. J. Gao, "Path planning for free-flying space robot using ant algorithm," *Robot,* vol. 24, no. 6, p. 526–529, 2002.

[7] X. Fan, X. Luo, S. Yi, S. Yang and H. Zhang, "Optimal Path Planning for Mobile Robots Based on Intensified Ant Colony Optimization Algorithm," in *IEEE Intemational Conference on Robotics Intelligent Systems and Signal Processing*, China, 2003.

[8] Y.-T. Hasio, C.-L. Chuang and C.-C. Chien, "Ant colony optimization for best path planning,," *IEEE International Symposium on Communications and Information Technology,* p. 109–113, 26–29 October 2004.

[9] X. P. Fan, . X. Luo, S. Yi and H. Zhang, "Path Planning For Robots Based On Ant Colony Optimization Algorithm Under Complex Environment," *Control and Decision,* vol. 19, no. 2, p. 166–170, 2004.

[10] B. A. Garro, H. Sossa and R. A. Vazquez, "Evolving ant colony system for optimizing path planning in mobile robots," *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007),* pp. 444-449, 2007.

[11] Y. E. Wen and H. D. Fan, "Algorithm for low altitude penetration aircraft path planning with improved ant colony algorithm," *Chinese Journal of Aeronautics,* vol. 18, no. 4, p. 304–309, 2005.

[12] W. Xu, C. U. I. Pingyuan and C. Yangzhou, "A new method and simulation for path planning problem based on ant colony algorithm," *Computer Simulation,* vol. 22, no. 7, p. 60–62, 2005.

[13] M. M. Mohamad, N. K. Taylor and M. W. Dunnigan, "Articulated robot motion planning using ant colony optimisation," in *2006 3rd International IEEE Conference Intelligent Systems*, London, UK, 2006.

[14] G. Liu, T. Li, Y. Peng and X. Hou, "The ant algorithm for solving robot path planning problem," in *Third International Conference on Information Technology and Applications (ICITA'05)*, Sydney, Australia, 2005.

[15] H. Mei, Y. Tian and L. Zu, "A hybrid ant colony optimization algorithm for path planning of robot in dynamic environment," *International Journal of Information Technology,* vol. 12, no. 3, p. 78–88, 2006.

[16] S. Liu, L. Mao and J. Yu, "Path planning based on ant colony algorithm and distributed local navigation for multi-robot systems," in *2006 International Conference on Mechatronics and Automation*, Luoyang, 2006.

[17] N. Lv and Z. Feng, "Numerical potential field and ant colony optimization based path planning in dynamic environment," in *2006 6th World Congress on Intelligent Control and Automation*, Dalian, China, 2006.

[18] M. M. Mohamad, . N. K. Taylor and M. W. Dunnigan, "Articulated robot motion planning using ant colony optimisation," in *2006 3rd International IEEE Conference Intelligent Systems*, London, UK, 2006.

[19] W. Ye, D. Ma and H. Fan, "Path planning for space robot based on the self-adaptive ant colony algorithm," in *2006 1st International Symposium on Systems and Control in Aerospace and Astronautics*, Harbin, China, 2006.

[20] T. A. N. Guan-Zheng, H. E. Huan and A. Sloman, "Ant colony system algorithm for real-time globally optimal path planning of mobile robots," *Acta automatica sinica,* vol. 13, no. 3, p. 279–285, 2007.

[21] C.-X. Shi, B. Ying-yong, L. Zi-guang and T. Jun, "Solving path planning problem by an ACO-PSO hybrid algorithm," in *Proceedings on Intelligent Systems and Knowledge Engineering (ISKE2007)*, Southwest Jiaotong University, Chengdu, P.R. China,, 2007.

[22] X. S. Chen, M. H. Lim and Y. S. Ong, "An Ant Colony System algorithm for path planning in sparse graphs," in *2007 International Conference on Intelligent and Advanced Systems*, Kuala Lumpur, 2007.

[23] N. A. Vien, N. H. Viet, S. Lee and T. Chung, "Obstacle avoidance path planning for mobile robot based on ant-Q reinforcement learning algorithm," in *Advances in Neural Networks – ISNN 2007*, Berlin Heidelberg,, 2007.

[24] J.-W. Lee, J.-J. Kim, B.-S. Choi and J.-J. Lee, "Improved Ant Colony Optimization algorithm by potential field concept for optimal path planning," in *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*, Daejeon, 2008.

[25] Y. Yu, H. Gao and D. Wang, "Research on path planning for mobile robot based on ant colony algorithm in dynamic environment," in *2008 3rd IEEE Conference on Industrial Electronics and Applications*, Singapore, 2008.

[26] X. Chen and Y. Yuan, "Novel ant colony optimization algorithm for robot path planning," *Systems Engineering and Electronics*, vol. 30, p. 952–955, 2008.

[27] N. H. Viet, N. A. Vien, S. Lee and T. Chung, "Obstacle avoidance path planning for mobile robot based on multi colony ant algorithm," in *First International Conference on Advances in Computer-Human Interaction*, Sainte Luce, Martinique, 2008.

[28] M. Gao, J. Xu and J. Tian, "Mobile robot global path planning based on improved augment ant colony algorithm," in *2008 Second International Conference on Genetic and Evolutionary Computing*, Jinzhou, China, 2008.

[29] X. Zhang, M. Wu, J. Peng and F. Jiang, "A rescue robot path planning based on ant colony optimization algorithm," in *2009 International Conference on Information Technology and Computer Science*, Kiev, Ukraine, 2009.

[30] J.-W. Lee, J.-J. Kim and J.-J. Lee, "Improved Ant Colony Optimization algorithm by path crossover for optimal path planning," in *2009 IEEE International Symposium on Industrial Electronics*, Seoul, South Korea, 2009.

[31] J.-W. Lee, J.-J. Kim, B.-S. Choi and J.-J. Lee, "Improved Ant Colony Optimization algorithm by potential field concept for optimal path planning," in *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*, Daejeon, 2008.

[32] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Appl. Soft Comput.*, vol. 9, no. 3, p. 1102–1110, 2009.

[33] H.-J. Wang and W. Xiong, "Research on global path planning based on ant colony optimization for AUV," *J. Mar. Sci. Appl.*, vol. 8, no. 1, p. 58–64, Mar. 2009.

[34] P. Zotos, A. J. Ijspeert and S. Degallier, "Path Planning with the humanoid robot iCub," 2009.

[35] N. B. Sariff and N. Buniyamin, "Ant colony system for robot path planning in global static environment," *9th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE'10),* vol. 192, pp. 192-197, 4-6 October 2010.

[36] A. Yang, L. Gao and Y. Luo, "An improved ant colony system algorithm for optimal path planning problem of mobile robots," in *2010 Second International Conference on Computer Modeling and Simulation*, Sanya, China, 2010.

[37] D. L. Luo and S. X. Wu, "Ant Colony Optimization With Potential Field Heuristic For Robot Path Planning," *Systems Engineering and Electronics*, vol. 6, 2010.

[38] Y. Wu, L. Shi, Q. Li and Y. Chen, "Novel path planning of robots based on bidirectional ant colony algorithm," in *2010 Sixth International Conference on Natural Computation*, Yantai, China, 2010.

[39] Z. Shaogang and L. Ming, "Path planning of inspection robot based on ant colony optimization algorithm," in *2010 International Conference on Electrical and Control Engineering*, Wuhan, China, 2010.

[40] J.-P. Zhao, X.-W. Gao, J.-G. Liu and Y.-Q. Chen, "Research of path planning for mobile robot based on improved ant colony optimization algorithm," in *2010 2nd International Conference on Advanced Computer Control*, Shenyang, China, 2010.

[41] J.-W. Lee, Y.-I. Choy, M. Sugisakaz and J.-J. Lee, "Study of novel heterogeneous ant colony optimization algorithm for global path planning," in *2010 IEEE International Symposium on Industrial Electronics*, Bari, Italy, 2010.

[42] M. Brand, M. Masuda, N. Wehner and X.-H. Yu, "Ant Colony Optimization algorithm for robot path planning," in *2010 International Conference On Computer Design and Applications*, Qinhuangdao, China, 2010.

[43] G. Nagib and W. Gharieb, "Path planning for a mobile robot using genetic algorithms," in *International Conference on Electrical, Electronic and Computer Engineering, 2004. ICEEC '04*, Cairo, Egypt, 2005.

[44] H.-Z. Zhuang, S.-X. Du and T.-J. Wu, "On-line real-time path planning of mobile robots in dynamic uncertain environment," *J. Zhejiang Univ. - Sci. A*, vol. 7, no. 4, p. 516–524, 2006.

[45] V. Lumelsky and A. Stepanov, "Dynamic path planning for a mobile automaton with limited information on the environment," *EEE Trans. Automat. Contr.*, vol. 31, no. 11, p. 1058–1063, 1986.

[46] N. Buniyamin, N. Sariff, W. A. J. Wan Ngah and Z. Mohamad, "Robot global path planning overview and a variation of ant colony system algorithm," *International journal of mathematics and computers in simulation*, vol. 5, no. 1, p. 9–16, 2011.

[47] J.-C. Latombe, A. Lazanas and S. Shekhar, "Robot motion planning with uncertainty in control and sensing," *Artif. Intell.*, vol. 52, no. 1, p. 1–47, Nov. 1991.

[48] J. Giesbrecht, "Global path planning for unmanned ground vehicles," DEFENCE RESEARCH AND DEVELOPMENT SUFFIELD (ALBERTA), 2004.

[49] R. A. Brooks and T. Lozano-Pérez, "A subdivision algorithm in configuration space for findpath with rotation," in *International Joint Conference on Artificial Intelligence*, Karlsruhe, Germany, 1983.

[50] D. J. Zhu and J. Latombe, "Constraint reformulation and graph searching techniques in hierarchical path planning," Stanford University, 1989.

[51] H. Samet, "An overview of quadtrees, octrees, and related hierarchical data structures," *Theoretical Foundations of Computer Graphics and CAD*, p. 51–68, 1988.

[52] H. Noborio, T. Naniwa and S. Arimoto, "A quadtree-based path-planning algorithm for a mobile robot," *J. Robot. Syst.*, vol. 7, no. 4, p. 555–574, Aug. 1990.

[53] D. Z. Chen, R. J. Szczerba and J. J. Uhran, "Planning conditional shortest paths through an unknown environment: a framed-quadtree approach," *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 3, p. 33–38, 1995.

[54] J. T. Schwartz and M. Sharir, "On the "piano movers'" problem I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers," *Communications on pure and applied mathematics*, vol. 36, no. 3, p. 345–398, 1983.

[55] F. Avnaim, J.-D. Boissonnat and B. Faverjon, "A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles," in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, IEEE, 1988, p. 1656–1661.

[56] N. Sleumer and N. Tschichold-Gürmann, "Exact cell decomposition of arrangements used for path planning in robotics," *Technical Report/ETH Zurich, Department of Computer Science*, vol. 329, 1999.

[57] O. Khatib and L. M. Mampey, "Fonction decision-commande d'un robot manipulateur, Rep. 2/7156," *DERA/CERT, Toulouse, France*, 1978.

[58] N. Hogan, "Impedance control: An approach to manipulation: Part II—Implementation," *J. Dyn. Syst. Meas. Control*, vol. 107, no. 1, p. 8–16, 1985.

[59] F. Miyazaki and S. Arimoto, "Sensory feedback for robot manipulators," *Journal of Robotic Systems*, vol. 2, no. 1, p. 53–71, 1985.

[60] V. V. Pavlov and A. N. Voronin, "The method of potential functions for coding constraints of the external space in an intelligent mobile robot," *Soviet Automatic Control*, vol. 17, no. 6, p. 45–51, 1984.

[61] D. Koditschek, "Exact robot navigation by means of potential functions: Some topological considerations," in *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, IEEE, 1987, p. 1–6.

[62] E. Rimon and D. E. Koditschek, "Exact Robot Navigation Using Artificial Potential Fields," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, p. 501–518, 1992.

[63] Y. K. Hwang and N. Ahuja, "Gross motion planning—a survey," *ACM Computing Surveys (CSUR)*, vol. 24, no. 3, p. 219–291, 1992.

[64] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The international journal of robotics research*, vol. 4, no. 3, p. 109–117, 1985.

[65] Y.-C. Chen and M. Vidyasagar, "Optimal trajectory planning for planar n-link revolute manipulators in the presence of obstacles," in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, PA, USA, 1988.

[66] N. J. Nilsson, "Shakey the robot Technical note 323," *Artificial Intelligence Center, SRI International, Menlo Park, CA*, 1984.

[67] J. Canny, "A new algebraic method for robot motion planning and real geometry," in *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, Los Angeles, CA, USA, 1987.

[68] J. F. Canny and M. C. Lin, "An opportunistic global path planner," *Algorithmica*, vol. 10, no. 2, p. 102–120, 1993.

[69] V. Boor, M. H. Overmars and A. F. Van Der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, 1999.

[70] D. Hsu, J.-C. Latombe and R. Motwani, "Path planning in expansive configuration spaces," in *Proceedings of International Conference on Robotics and Automation*, 1997.

[71] P. Leven and S. Hutchinson, "Robust, compact representations for real-time path planning in changing environments," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, 2001.

[72] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, p. 378–400, 2001.

[73] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2, IEEE, 2000, p. 995–1001.

[74] P. Cheng, Z. Shen and S. M. LaValle, "RRT-based trajectory design for autonomous automobiles and spacecraft," *Archives of control science*, vol. 11, no. 3/4, p. 167–194, 2001.

[75] E. Frazzoli, "Robust hybrid control for autonomous vehicle motion planning," Massachusetts Institute of Technology, 2001.

[76] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations," *Journal of computational physics*, vol. 79, no. 1, p. 12–49, 1988.

[77] M. S. Hassouna, A. E. Abdel-Hakim and A. A. Farag, "PDE-based robust robotic navigation," in *he 2nd Canadian Conference on Computer and Robot Vision (CRV'05)*, IEEE, 2005, p. 176–183.

[78] J. A. Sethian, "Tracking Interfaces with Level Sets: An" act of violence" helps solve evolving interface problems in geometry, fluid mechanics, robotic navigation and materials sciences," *American Scientist*, vol. 85, no. 3, p. 254–263, 1997.

[79] B. Stilman, "Syntactic hierarchy for robotic systems," *Integrated Computer-Aided Engineering*, vol. 1, no. 1, p. 57–81, 1993.

[80] B. Stilman, "Stilman Advanced Strategies," [Online]. Available: http://www.stilman-strategies.com. [Accessed 12 10 2022].

[81] E. Masehian and D. Sedighizadeh, "Classic and heuristic approaches in robot motion planning-a chronological review," *World Academy of Science, Engineering and Technology*, vol. 23, no. 5, p. 101–106, 2007.

[82] S. Goss, S. Aron, J.-L. Deneubourg and J. M. Pasteels, "Self-organized shortcuts in the Argentine ant," *Naturwissenschaften*, vol. 76, no. 12, p. 579–581, 1989.

[83] C. Öztürk, Karınca Ve Sürü Optimizasyon Yöntemlerinin Incelenmesi Ve Yazılım Uygulamlarının Oluşturulması (Doctoral dissertation), Marmara Universitesi (Turkey), 2006.

[84] M. Dorigo and V. Maniezzo, "The Ant System: An Autocatalytic Optimizing Process," Politecnico di Milano, Milan, 1991.

[85] M. Dorigo, "Optimization, Learning and Natural Algorithms," in *Thesis(PHD), Dipartimento di Elettronica e Informazione*, Milan, Dipartimento di Elettronica e Informazione, Politecnico di Milano, 1992.

[86] G. Dalkiliç and F. Türkmen, "Karınca kolonisi optimizasyonu," *YPBS2002–Yüksek Performanslı Bilişim Sempozyumu, Kocaeli, Ekim,* 2002.

[87] M. Dorigo and T. Stutzle, "Ant Colony Optimization," *MIT Press, Cambridge, MA,* 2004.

[88] A. Uğur and D. Aydın, "Ant system algoritmasının jawa ile görselleştirilmesi," *Akademik Bilişim,* p. 572–576, 2006.

[89] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 26, no. 1, p. 29–41, 1996.

[90] T. Keskintürk and H. Söyler, "Global karınca kolonisi optimizasyonu," *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi,* vol. 21, no. 4, p. 689–698, 2006.

[91] H. Söyler and T. Keskintürk, "Karinca Kolonisi Algoritmasi ile Gezen Satici Probleminin Çözümü," *8. Türkiye Ekonometri ve İstatistik Kongresi,* p. 24–25, 2007.

[92] K.-C. Ying and C.-J. Liao, "An ant colony system for permutation flow-shop sequencing," *Computers & Operations Research,* vol. 31, no. 5, p. 791–801, 2004.

[93] B. Bullnheimer, R. F. Hartl and C. Strauß, "A new rank based version of the Ant System. A computational study," *SFB Adaptive Information Systems and Modelling in Economics and Management Science,* 1997.

[94] T. Stutzle and H. Hoos, "MAX-MIN ant system and local search for the traveling salesman problem," *Proceedings of 1997 IEEE international conference on evolutionary computation (ICEC'97),* p. 309–314, 1997.

[95] M. Dorigo, V. Maniezzo and A. Colorni, "The ant system: An autocatalytic optimizing process," Technical report,, Milan, 1991.

[96] D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli and K. Price, New ideas in optimization, McGraw-Hill Ltd., UK, 1999.

[97] M. Dorigo, G. Di Caro and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial life,* vol. 5, no. 2, p. 137–172, 1999.

[98] G. Bilchev and I. C. Parmee, "The ant colony metaphor for searching continuous design spaces," in *AISB workshop on evolutionary computing*, 1995, p. 25–39.

[99] M. Wodrich, "Cooperative distributed search: The antss way," *Control and Cybernetics,* vol. 26, no. 3, p. 413–446, 1997.

[100] M. Mathur, S. B. Karale, S. Priye, V. K. Jayaraman and B. D. Kulkarni, "Ant colony approach to continuous function optimization," *Industrial & engineering chemistry research,* vol. 39, no. 10, p. 3814–3822, 2000.

[101] M. H. Afshar, H. Ketabchi and E. Rasa, "Elitist continuous ant colony optimization algorithm: application to reservoir operation problems," *International Journal of Civil Engineering,* vol. 4, no. 4, p. 274–285, 2000.

[102] D. Karaboğa, "Yapay Zeka Optimizasyon Algoritmaları," *Atlas Yayın Dağıtım, 1. Basım,* 2004.

[103] N. Karaboğa, K. Güney, A. Akdağli and D. Karaboğa, "DOĞRUSAL ANTEN DİZİLERİNDE OPTİMUM DEMET ŞEKİLLENDİRME AMACIYLA KARINCA KOLONİ OPTİMİZASYON ALGORİTMASININ KULLANILMASI," *Erciyes Üniversitesi Fen Bilimleri Enstitüsü Fen Bilimleri Dergisi*, vol. 22, no. 1, p. 66–74, 2006.

[104] A. Kalınlı, D. Karaboğa and N. Karaboğa, "A modified touring ant colony optimisation algorithm for continuous functions," 2001.

[105] X. Bajrami, A. Dermaku, N. Demaku, S. Maloku, A. Kikaj and A. Kokaj, "Genetic and Fuzzy logic algorithms for robot path finding," *2016 5th Mediterranean Conference on Embedded Computing (MECO),* p. 195–199, 2016.

[106] B. Tutuko, S. Nurmaini and G. Ogi, "Fuzzy Logic-Ant Colony Optimization for Explorer-Follower Robot with Global Optimal Path Planning," *Computer Engineering and Applications Journal,* vol. 7, no. 1, p. 61–74, 2018.

[107] S. Nurmaini and F. Setianingsih, "Enhancement of the fuzzy control response with particle swarm optimization in mobile robot system," *2018 International Conference on Electrical Engineering and Computer Science (ICECOS),* p. 73–78, 2018.

[108] G. Kyprianou, L. Doitsidis and S. A. Chatzichristofis, "Towards the achievement of path planning with multi-robot systems in dynamic environments," *J. Intell. Robot. Syst.,* vol. 104, no. 1, 2022.

[109] J. Barraquand and J.-C. Latombe, "Robot motion planning: A distributed representation approach," *The International Journal of Robotics Research,* vol. 10, no. 6, p. 628–649, 1991.

[110] Y. Wang, D. M. Lane and G. J. Falconer, "Two novel approaches for unmanned underwater vehicle path planning: constrained optimisation and semi-infinite constrained optimisation," *Robotica,* vol. 18, no. 2, p. 123–142, 2000.

[111] O. Khatib, "Real-time Obstacle Avoidance For Manipulators and Mobile Robots," *International Journal of Robotics,* vol. 2, no. 7, p. 90–98, 1986.

[112] G. Dudek, Computational Principles of Mobile Robotics, Cambridge University Press, 2000.

[113] J. Borestein and Y. Koren, "The Vector Field Histogram Fast Obstacle Avoidance for Mobile Robots," *IEEE Journal of Robotics and Automation,* vol. 7, no. 3, pp. 278-288, 1991.

[114] Y. Chang and Y. Yamamoto, "Online Path Planning Strategy Integrated with Collision and Dead-lock Avoidance Schemes for Wheeled Mobile Robot in Indoor Environment," *Industrial Robot-An International Journal,* vol. 35, no. 5, pp. 421-434, 2008.

[115] R. Kala, "Code for Robot Path Planning using Fuzzy Logic, Indian Institute of Information Technology Allahabad," 2014. [Online]. Available: http://rkala.in/codes.html. [Accessed 15 11 2022].

[116] L. A. Zadeh, "Fuzzy sets," *Information and control,* vol. 8, no. 3, p. 338–353, 1965.

[117] S. Assilian and E. H. Mamdani, "Learning control algorithms in real dynamic systems," in *4th IFAC/IFIP International conference on digital computer applications to process control*, Springer, 1974, p. 13–24.

[118] A. Martinez, E. Tunstel and M. Jamshidi, "Fuzzy logic based collision avoidance for a mobile robot," *Robotica,* vol. 12, no. 6, p. 521–527, 1994.

[119] S. Şaka, "Bulanık kontrol ve uygulamaları," in *PhD Thesis*, İTÜ, Fen Bilimleri Enstitüsü, 1999.

[120] T.-S. Jin, "Obstacle avoidance of mobile robot based on behavior hierarchy by fuzzy logic," *International Journal of Fuzzy Logic and Intelligent Systems,* vol. 12, no. 3, p. 245–249, 2012.

[121] Ç. Ergüven, "Bulanık mantık kontrolör ile klasik PID kontrolör algoritmalarının karşılaştırılması," in *Doctoral dissertation*, İTÜ, Fen Bilimleri Enstitüsü, 1999.

[122] E. Cox, "Fuzzy fundamentals," *IEEE spectrum,* vol. 29, no. 10, p. 58–61, 1992.

[123] M. A. Islam and M. S. Hossain, "Mathematical comparison of defuzzification of fuzzy logic controller for smart washing machine," *J. Bangladesh Acad. Sci.,* vol. 46, no. 1, p. 1–8, 2022.