

***"SAARTHI- A self-learning system for autonomous navigation
using Symbolic Computing"***

**A thesis submitted to the
*University of Petroleum and Energy Studies***

**For the award of
Doctor of Philosophy
in
Computer Science Engineering**

**BY
NIHARIKA**

January 2021

**SUPERVISOR (s)
Dr. Manish Prateek
Dr. Piyush Chauhan**



**School of Computer Science & Engineering
University of Petroleum and Energy Studies
Dehradun-248007; Uttarakhand**

***"SAARTHI- A self-learning system for autonomous navigation
using Symbolic Computing"***

**A thesis submitted to the
University of Petroleum and Energy Studies**

**For the award of
Doctor of Philosophy
In
Computer Science Engineering**

**BY
NIHARIKA
(SAP ID 500041827)**

January 2021

SUPERVISOR (s)

**Dr. Manish Prateek
Professor
SoCS, UPES, Dehradun**

**Dr. Piyush Chauhan
Assistant Professor-SG
SoCS, UPES, Dehradun**



**School of Computer Science and Engineering
University of Petroleum and Energy Studies
Dehradun-248007; Uttarakhand**

I dedicate my Ph.D. Thesis to

My loving Parents, husband, son and

my Guide Professor

Dr. Manish Prateek and Dr. Piyush Chauhan

for their endless support, blessings, and guidance.

**SCHOOL OF COMPUTER SCIENCE AND
ENGINEERING**

**UNIVERSITY OF PETROLEUM AND ENERGY
STUDIES**

DEHRADUN-248007



DECLARATION

I declare that this thesis, which I submit to the University of Petroleum and Energy Studies, Dehradun, for examination in consideration of the award of a higher degree Doctor of Philosophy in Computer Science and Engineering, is my effort. Where any of the content presented results from the input or data from a related collaborative research program, this is duly acknowledged in the text that it is possible to ascertain how much of the work is my own. Furthermore, I took reasonable care to ensure that the work is original. To the best of my knowledge, it does not breach copyright law and has not been taken from other sources except where such work has been cited and acknowledged within the text.


Signature of the candidate

SAP ID: 500041827

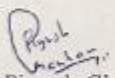
Date: 25-01-2021

THESIS COMPLETION CERTIFICATE

This is to certify that the thesis on "**SAARTHI**". *A self-learning system for autonomous navigation using Symbolic Computing* submitted by Niharika to School of Computer Science and Engineering, UPES, Dehradun in Partial completion of the requirements for the award of the Degree of Doctor of Philosophy (Engineering- Computer Science) is an original work carried out by her under our joint supervision and guidance. It is further certified that the work has not been submitted anywhere else for the award of any other diploma or degree of this or any other University.



Dr. Manish Prateek
(Supervisor)



Dr. Piyush Chauhan
(Co-Supervisor)

ABSTRACT

For numerous years, beginning in the mid-1970s, the path-planning problem for moveable machines has been a hot topic of research. There were various approaches proposed for this research domain. It is vital to find an appropriate technique to achieve both the quality and efficiency of a search. Because of the expected course, the robot does not waste time on superfluous steps or moving in local minimum locations. It is also preferable to avoid all of the identified barriers in the region. The mobile robot interacts with barriers in a heuristic manner. Mobile robots' main goal is to land at predetermined target places without colliding with impediments. The applications claims an intelligent selection of the optimal path. It could be achieved by discovering and learning the environment. Simultaneously, by removing the impediments, the machines should be able to reach the goal points in a discrete amount of period.

Using improved optimization algorithms, the work presents an effective route-planning approach for mobile robots. To elucidate the challenge of mobile robot path planning, a new solution uses a cuckoo optimization algorithm. This program accurately identifies objects and assesses the impact of various design decisions by creating a model to determine its performance. The proposed technique outperforms state-of-the-art understanding in various benchmarks, as well as providing categorization in real-time applications.

The work's key contribution was to efficiently deployment of cuckoo-type robots for terrain assessment and disclosure. Q-learning, Cellular Automata (CA), and Cuckoo Search Optimization are the three methods used in this research (CSO). During the automation, the goal of this system becomes more sophisticated as the best sets of actions are identified. The different interactive components necessitate a parallel reinforcement learning process. Finally, the system's output was collision-free navigation in an unfamiliar environment. Different statistical parameters are also utilized to compare an algorithm's performance.

Keywords: Q-learning, Cuckoo Search Optimization, Cellular Automata, Unknown Environment.

ACKNOWLEDGEMENT

I bow my head humbly to pay sincere regards to Almighty God to give me the strengths and blessing in completing this work. Quite a few people have helped me in one way to the completion of this work. It was a great pleasure, and I would like to thank all of you from very deep inside.

Foremost, I would like to express my sincere gratitude to my Ph.D. advisors Prof. Manish Prateek and Dr. Piyush Chauhan, for selecting me as a student and for the continuous support of my Ph.D. study and research for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and completing this work.

It is not easy to find and develop an idea without the help of a specialist in the domain. I found in my advisors the source of beautiful views to create and the support that a Ph.D. student needs. I could not have imagined having a better advisor and mentor for my Ph.D. study.

Besides my mentor, I would like to thank Chancellor Dr. S.J. Chopra, Vice-Chancellor Dr. Sunil Rai at the University of Petroleum and Energy Studies for encouraging, suggest, and valuable support for my research work.

I want to express my special thanks to Dr. Kiran, Dr. Rakhi Ruhai for their assistance during my research work. I am grateful to the University of Petroleum and Energy Studies to pursue my research and provide all the School of Computer Science and Engineering facilities.

I want to thank all the heads of the School of Computer Science and Engineering Departments, doctoral students for their feedback, cooperation, and of course, friendship. In addition, I would like to express my gratitude to all colleagues in the university.

I cannot begin to express my gratitude to my family for all the love, support, encouragement, and prayers they have sent my way along this journey. I am

eternally indebted to my loving parents and in-laws for all the sacrifices they have made on my behalf. I want to express sincere gratitude to my beloved husband Ajay, who believed in me and provide encouragement during challenging times. Your unconditional love and support in the moments when there was no one to answer my queries have helped me immensely. To my son Aayansh, he is the inspiration for me to complete this journey and the sacrifices made along the way.

Table of Contents

Declaration.....	ii
Thesis Completion Certificate.....	iii
Abstract.....	iv
Acknowledgement.....	v
Table of Contents.....	vi
List of Abbreviations.....	x
List of Figures.....	xi
List of Tables.....	xiv
Chapter 1: Introduction.....	1
1.1 Introduction.....	1
1.2 Introduction to Artificial Intelligence.....	4
1.2.1 Classification of Artificial Intelligence.....	5
1.2.2 Applications and Concerns of Artificial Intelligence.....	6
1.2.3 Applications of Artificial Intelligence in Autonomous Navigation.....	7
1.3 Problem Statement.....	7
1.4 Motivation.....	8
1.5 Objective.....	9
1.6 Contribution of the Thesis.....	10
1.7 Organization of the Thesis.....	10
Chapter 2: Literature Review.....	12

2.1 Background.....	12
2.2 Algorithms for Autonomous Navigation in the proposed work.....	17
2.2.1 Q-Learning Algorithm.....	17
2.2.2 Cellular Automata.....	22
2.2.3 Cuckoo Search Optimization.....	27
2.3 The work focus for autonomous navigation in an unknown environment.....	30
2.4 Summary.....	36
Chapter 3: Model Development for Autonomous Navigation in an unknown environment.....	37
3.1 Introduction.....	37
3.2 The proposed framework.....	39
3.2.1 Model and assumptions.....	42
3.3 Environment Learning.....	43
3.4 Summary.....	45
Chapter 4: Implementation of the self-learning system.....	46
4.1 Exploration of the terrain and the coverage of the robotic environment.....	46
4.2 The proposed scheme implementation.....	47
4.3 Implementation of the test cases.....	59
4.4 Summary.....	61
Chapter 5: Results and discussions.....	62
5.1 Role of MATLAB in robotic environment.....	63

5.2 Results and discussions of the robot in a dynamic environment.....	65
Chapter 6: Conclusion and Future Scope.....	103
6.1 Conclusion.....	103
6.2 Future Scope.....	103
Bibliography.....	104
Appendix A.....	116

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
RL	Reinforcement Learning
CA	Cellular Automata
PSO	Particle Swarm Optimization
CSO	Cuckoo Search Optimization
ANN	Artificial Neural Network
GANs	Generative Adversarial Networks

LIST OF FIGURES

Figure 2-1 Types of Robotic Systems.....	13
Figure 2-2 Example of Programmable robot.....	14
Figure 2-3 Example of Non-Programmable Robot.....	15
Figure 2-4 Example of Autonomous Robot.....	16
Figure 2-5 Illustration of the learning process in RL.....	18
Figure 2-6 Spatial Neighborhood.....	26
Figure 2-7 Movements of Robots.....	26
Figure 3-1 The workflow of the SAARTHI Algorithm.....	39
Figure 3-2 Sensing Directions of the Cells.....	42
Figure 4-1 Flow Diagram of the Q-learning Algorithm.....	49
Figure 4-2 Sample of Q-table.....	50
Figure 4-3 Updated Q-table.....	51
Figure 4-4 Final Q-table.....	52
Figure 4-5 Flowchart of Q-learning for navigation.....	53
Figure 4-6 Scenario of the robotic environment.....	58

Figure 4-7 Illustration of Simulating Environment for the robot.....	60
Figure 5-1 Construction of the map.....	66
Figure 5-2 Illustration for the MAP 8 at different target positions.....	82-84
Figure 5-3 Illustration for the MAP 10 at different target positions.....	85-87
Figure 5-4 Illustration for the MAP 12 at different target positions.....	88-90
Figure 5-5 Illustration for the MAP 14 at different target positions.....	91-93
Figure 5-6 Illustration for the MAP 20 at different target positions.....	94-96
Figure 5-7 Target Points Vs. Error in Path Selection for Map 8.....	98
Figure 5-8 Target Points Vs. Error in Path Selection for Map 10.....	98
Figure 5-9 Target Points Vs. Error in Path Selection for Map 12.....	99

Figure 5-10 Target Points Vs. Error in Path Selection for Map
14.....99

Figure 5-11 Target Points Vs. Error in Path Selection for Map
20.....100

LIST OF TABLES

Table 3-1 Steps of SAARTHI Algorithm.....	40
Table 3-2 SAARTHI Algorithm.....	41
Table 4-1 Steps of Cellular Automata for Path Selection.....	56
Table 4-2 Steps of Cuckoo Search for Robot Movement.....	58-59
Table-5-1 Sample input data of the working environment.....	66
Table 5-2 State Vs. Actions for Map 8.....	68
Table 5-3 State Vs. Actions for Map 10.....	69
Table 5-4 State Vs. Actions for Map 12.....	70
Table 5-5 State Vs. Actions for Map 14.....	71
Table 5-6 State Vs. Actions for Map 20.....	72-73
Table 5-7 Obstacles combination on generating the paths.....	75
Table 5-8 Distance traveled by the robots.....	77
Table 5-9 Success Matrix for Q-learning between existing and proposed model.....	78
Table 5-10 Confusion Matrix between existing and proposed model.....	79

Table 5-11 Comparative Analysis of Q-learning and SAARTHI

Algorithm.....97

Table 5-12 Time Consumption for Q-learning and SAARTHI

Algorithm.....101

CHAPTER 1

INTRODUCTION

1.1 Introduction

In robotics and artificial intelligence, autonomous navigation is a critical subject. The technique of directing movement from source to destination is known as navigation. Real-time navigation is a simple task for humans and animals, but it is extremely challenging for robots, particularly in an unfamiliar and changing environment. [Anmin Zhu et al. (2007)]

Learning about the environment is an important step in autonomous navigation. Robot navigation may be classified into two types based on the surroundings: [Bashan Zuo et al. (2014)]

(a) Navigation in a known environment: The mobile device has access to the global path and the location of the obstacles ahead of time.

(b) Navigation in an unknown environment: The mobile device has no knowledge of the path or the location of the obstacles. This category can be further divided into the following subcategories:

(a) Behavior-based approach: Within the sensor range, information about the surroundings can be acquired, and the pathway can be determined.

(b) Learning-based approach: The robot learns the navigation method on its own, and as it gains experience, its performance improves.

A learning-based methodology is necessary to implement the self-learning capability (reinforcement learning) within an autonomous system. Intelligent decision-making is enabled by a system's ability to self-learn. The system evaluates the environment before making a decision.

As a result, Navigation with Autonomy could be defined as the development and execution of a path from one point in space to different point in order to complete a task while identifying and avoiding collisions with obstacles or undesirable behavior.

To optimize cost and energy measures, the plan should make the best use of existing resources. The navigation module for autonomous vehicles is a critical component to consider.

Robots are utilized in practically every industry where a repetitive and complex task is required, as well as those that are hazardous or impossible to complete manually, such as:

- (a) In the aerospace and automobile industries, painting the car, welding various specimens or machines, and surface finishing
- (b) Applications in submarines and space.
- (c) Destructive fritter away remediation in administrative, nuclear, and medical labs.
- (d) Parts examination.
- (e) Assembling electronic and consumer products.
- (f) Inspection and distribution of parts in a diversity of productions.

The idea of mobile robotic machine sovereignty encompasses numerous fields of technology in industrial engineering and modern technology. These approaches are intended for trajectory management, obstacle avoidance, mobile robot localization, path planning, and so on. The convenience of employing an explicit display of the navigation environment almost entirely determines the sensation of a map planning, obstacle avoidance, and navigation work performed by an autonomous mobile robot.

The paradigms [Alempijevic et al. (2004)] regressed within the robotics community for self-navigation system are:

(a) Hierarchical (Plan Based Approach)

(b) Reactive

(c) Hybrid Deliberative

The aforementioned concepts point to a significant inconsistency in Autonomous Navigation. A global map is used in the plan-based approach, which relies on global self-localization and is defined by sense-plan-act. The reactive system is defined by sense-act and employs local control rules in relation to local features. It also relies on precise local feature recognition. If the hierarchical approach utilizes sense-plan-act and the reactive approach uses sense-act, then Plan, Sense-Act describes the hybrid-deliberative process. As a result, this technique combines the best of both worlds, in that it includes a planner and the deliberate part of the planner. Following an overview of the paradigms, the following section examines the key operational aspects of an autonomous vehicle.

The following are the primary operating characteristics of an autonomous vehicle: (a) Perception

(b) Intelligence

(c) Action

Artificial intelligence, as an expression of the aforementioned attributes, is a flawless component. A great number of tools and algorithms are available in the field of artificial intelligence. Hill Climbing, A*, D*, Fuzzy logic, Artificial Neural Network (ANN), Neuro-Fuzzy, Genetic Algorithm, Swarm Intelligence, Particle Swarm Optimization (PSO), and so on are some of the AI methods available. However, if the environment changes even slightly in the above-mentioned techniques, reprogramming is required. Self-learning (Reinforcement learning) is required for any autonomous mobile system to avoid this.

As a result, the research reveals that artificial intelligence is becoming the brain of a self-driving car. This chapter focuses on an introduction to artificial

intelligence (AI), followed by a classification of AI and its applications. In addition, the chapter concluded with a research problem statement, motivation, and study goal. The contribution of the thesis and the organization of the thesis are discussed in the chapter's later sections.

1.2 Introduction to Artificial Intelligence

Artificial Intelligence (AI) is an upcoming paradigm shift to the field of computer science, which provides computers the discretion and ability to perform and execute tasks, which require manual or human intervention. Artificial Intelligence musters the prowess to perform highly complex tasks with ease and give time-bound results. With the development of fields like machine learning and deep learning under the umbrella of artificial intelligence, machines now can perform actions that took years of experience and undivided attention with just a few lines of code written to provide the machine instructions based on the previous data.

Artificial Intelligence relies hugely on complex algorithms. Algorithms are a chain of specific commands, which the machine follows to obtain a particular output. AI algorithms mostly rely on a steady stream of data, which helps the model learn and adapt to the data's pattern. These algorithms are powerful enough to recognize patterns, which are incomprehensible for a normal human brain. Sometimes the information is so enormous that it has to be represented in tens and hundreds of dimensions in vectors. This approach is undoubtedly not very recognizable for human brains as it cannot simply process such vast chunks of data at a given point in time.

Artificial Intelligence relies so much on data that it has become the oil to every industry. The data, which was thought to be useless until 70 years ago, now had a significant role in providing companies and enterprises new business. These corporations could quickly learn how, when, and where the customer is most likely to use a service and make it available conveniently, which immediately made these companies such as uber successful.

Until the past decade, only major corporations, which had the resources, could use Artificial Intelligence, however with improved public technology, artificial intelligence was democratized for the public. Now everyone has the power of artificial intelligence built into their smart mobile devices. Applications like photo editing to file management are using artificial intelligence to provide ordinary people a better experience with increased productivity right at the convenience of their fingertips.

With increasingly powerful machines at disposal for the typical person and rapid research in artificial intelligence, the lives of familiar people will be nothing but easier and convenient.

1.2.1. Classification of Artificial Intelligence

Unlike many other computer science branches, artificial intelligence spans out almost all possible applications through myriad data and structures. Broadly, AI has two types, but it can further be classified into more subdomains based on its functionalities. One of the broad classifications is weak artificial intelligence or narrow artificial intelligence. This type of AI focuses on a singleton task and directs all its efforts towards achieving one goal: acquiring the best set of performance on that one task. The other type of AI is Strong AI or an artificially intelligent machine that can think and process information like humans. Every weak AI is helping to achieve the goal of building someday a Strong AI model.

The other type of classification for AI is based on its functionality. For example, a machine can be a reactive machine where the device does not have the capabilities to remember the past data and makes decisions on and as the data comes through. These machines were good back in the day, but with the development of better hardware, there are limited memory machines that could store some data to make valuable decisions in the not-so-distant future. Autonomous vehicles use this learning in many areas like observing lane changes and so on. Then there are two more proposed models: theory of mind, which can understand emotions, beliefs, thought process, etc., and the last

type of machines are the one self-aware ones. These machines have their consciousness and have their own set of beliefs and perceptions.

1.2.2. Applications and Concerns of Artificial Intelligence

Artificial Intelligence has been enormously pervasive in most industries, products, and services that surround us today. Applications range from producing information-altering e-systems like deep fakes, which work on Generative Adversarial Networks (GANs), to autonomous vehicles and places like creating original music and movies.

While AI applications have since proved to be a massive boon for the entire community, it has many challenges and hurdles. The first and foremost is the problem of ethics. AI models are capable of learning about anything from the data they are being fed. Since these models do not have discretionary power, they cannot decide what is ethically correct and wrong. Applications like deep fakes and GPT-3 have since been brought to light due to their uncanny abilities to alter accurate data into some truth-altering information, which is seemingly indistinguishable for the human eye. Although they have the power to solve many problems, these applications seem to have started creating new ones. Ethics is not the only challenge that AI faces. Cognitive abilities, knowledge representation, resource planning, social intelligence, perception, general intelligence are just some of the other difficulties AI faces. However, with increasingly successful research outputs and the continuing efforts towards making AI possible for everyone AI is becoming better and better and is coming closer to becoming one single pool of information for all the needs.

1.2.3 Applications of AI in Autonomous Navigation

Autonomous vehicles rely heavily on substantial data dumps by the onboard sensors to make decisions in real-time. However, this data requires almost instant processing. The algorithms must make the insights and understand the data as humans. This cognitive touch to these algorithms is provided by artificial intelligence. Almost every AV manufacturer and researcher have to build models using branches of AI like machine learning and deep

learning to make sense of this data and provide impromptu results and feedbacks to the system for path planning operations.

While complete autonomy over human interactions on vehicles is a far-fetched dream, AI proves helpful in driving assistance in emergency cases, taking over as co-pilot on the car in urban planned settings, traffic light coordination, cross-traffic relays, and simulates blind spot detection.

With a massive amount of pictorial data, the AI models have started learning to adapt to complex environments with exponential numbers of stakeholders and objects in consideration. A large area where AI is also helping autonomous vehicles keeps the vehicle's health in check. With real-time data on vehicle's physical conditions and breakdowns, AI is reaching a new application area.

1.3 Problem Statement

In the age of unscrewed cars, autonomous navigation in an unfamiliar area is still a developing subject of study. Self-learning routing allows vehicles to travel from point A to point B with minimum human intervention. Any vehicle's capacity to self-learn allows it to explore the terrain in an unfamiliar location without supervision. 3D vision is a crucial field that allows autonomous vehicles to identify impediments with ease using depth perception. 3D vision may be used to identify the dimensions of the obstructions in the route. A system's reliability is important in addition to its capacity to identify impediments. The dependability of an autonomous system verifies that the service is delivered and the objective is met with the least amount of mistake possible. As a result, dependability may be defined as an autonomous vehicle's robustness and fault tolerance. If vehicles are modified to self-learn in an unfamiliar environment, robots can undertake tasks such as housework and space exploration with little or no human interaction. Individuals' lives will become less difficult and more comfortable as a result. The following research gaps were found:

- (a) The ability to make decisions in an unfamiliar environment.
- (b) Travel planning over a long period.

- (c) The algorithms that are now in use are computational.
- (d) The ability to self-learn how to recognize impediments and retrace one's steps.
- (e) 3D vision for symbolizing the environment's objects.
- (f) Reliability, which includes Robustness and Fault Tolerance.
- (g) Symbolic Computing is more efficient than existing image processing techniques.

Hence, a need to frame a research problem on efficient, trustworthy, self-learning algorithms for autonomous navigation utilizing Symbolic Computing for 3D vision by noting the significance and obstacles in the literature on autonomous navigation.

1.4 Motivation

In the epoch of advanced research, particularly in robotic automation, development and deployment are vast. The variety of mechanical applications range is tiny. Assume unmanned robots to rovers in deep space where the availability of the human interface may result in catastrophic life endangerment. The complete operation of these autonomous vehicles solely depends on the algorithm designed for its effective execution. The primary function of an autonomous vehicle is navigation. Thus, appropriate navigation techniques and decision-making capability aided by the self-learning experiences of robots are essential. These experiences are based upon the algorithm designing for effective and efficient operation. Given that time is another crucial constraint that limits the efforts, to be thorough. Let us consider a case where a given autonomous robotic vehicle gets trapped in a confined, limited access space, and then, this may lead to a failure to reach the destination.

Given the complexity of the above-stated operation, a need for an effective algorithm specially designed for obstacle detection, avoidance, and restoration of the original path in a real-time scenario. Additionally, it would

be advantageous to step up learning and decision-making capabilities to complement the vehicles intelligently. All the above-stated characteristics are essential for autonomous vehicles. Despite the above field, dependability is required to provide a smooth service, i.e., the system should be robust and fault-tolerant.

Thus, the idea of designing an efficient self-learning intelligent algorithm for obstacle detection, avoidance, and restoration of the original path for an autonomous vehicle is worth enduring research.

1.5 Objective

The objective of the research is "**To develop Symbolic computing-based 3D machine vision and self-learning algorithm for autonomous mobile system.**"

At the onset, the following sub-objectives were laid down, outlining the significant aspects of the undertaken research work. The design of artificial intelligence algorithm involves:

1. To study the various existing algorithms for autonomous navigation.
2. Identify the issues in existing independent navigation algorithms.
3. Understanding the concepts of Symbolic Computing.
4. Explore the various possibilities of the environment to prioritize the Autonomous Navigation algorithm according to the need of the situation.
5. Developing the complementary model of the environment using the 3D vision.
6. Devising the intelligent, self-learning algorithm for efficient autonomous navigation.
7. Implementing the newly designed algorithm.
8. Performance testing of the newly implemented algorithm and comparing the existing algorithms to show the efficiency of the newly implemented algorithm.

1.6 Contribution of the Thesis

The approach given here demonstrated a proven cooperative autonomous robot without the use of prior terrain knowledge. The work's key contribution is to fully and efficiently deploy cuckoo-type robots to explore and resolve terrain exploration and coverage. Another prominent part of the thesis describes symbolic computing. The effective path planning results with the incorporation of cellular automata during map-building.

1.7 Organization of the Thesis

The research is organized into six chapters. Each chapter is divided into sections and subsections.

Chapter1: Introduction

It introduces the autonomous vehicle, artificial intelligence, and various applications of an autonomous vehicle.

Chapter2: Literature Review

It covers the various techniques used for the navigation of the autonomous vehicle and a comparison of the past studies.

Chapter 3: Model development for autonomous navigation in an unknown environment.

It covers the detailed description of the design of the system for autonomous navigation for an unknown environment.

Chapter 4: Implementation of the self-learning system for autonomous navigation

It covers the performance of the system for the environment provided to the mobile device.

Chapter 5: Result and Analysis

It covers the results and analysis of the complete research work.

Chapter 6: Conclusion and Future Scope

This chapter focuses on the conclusion of the complete research work and the future scope of the work.

Bibliography and Appendix A

This section will be covering the references used for writing a thesis, followed by a list of publications in Appendix A.

CHAPTER 2

LITERATURE REVIEW

2.1 Background

The autonomous mobile device should accomplish the goal of navigating in an unknown environment without human interaction. The mobile robots should have proper path planning by avoiding collision with the obstacles and achieve the target. There were wide applications for the design, development, and operation of navigation in obscure environs in the later past. The study shows that the robots can be divided into three categories: manually controlled, remote-controlled, and autonomous controlled robotic systems, as illustrated in figure 2-1. Micromanagement of automated systems that call for extensive human involvement for its functioning was termed a manual robot. These automatic systems need exceptional human authority for controlling various processes incorporated with the machines. Manual Robots embrace a series of robotic systems, from elementary to highly advanced, all dealing with a specific control system according to its diligence. On the other side, a remote-controlled robot is defined as any vehicle, i.e., teleoperated by a means that does not restrict its motion with an origin external to the device. In addition, the last and foremost category of robots is autonomous. Human-like robots that have the power to get to their conclusions and accomplish an action consequently called Autonomous robots. The primary working mechanism of these robotic systems was to perceive the environment, make conclusions on the perception, and trigger a movement for the environment.

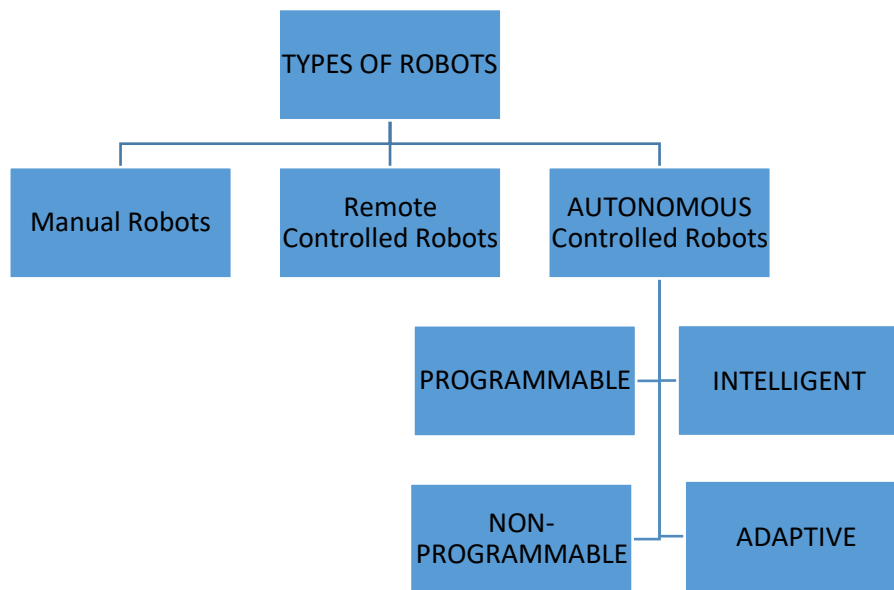


Figure 2-1: Types of Robotic Systems

This category of robots further sub-divided into programmable, non-programmable, adaptive, and intelligent.

A programmable robot is a novel machine with a selector ability. Reprogrammable robots are possible, depending on the type of use that is permitted. After the robot has been adjusted once to play out power in the supplied example and fixed arrangement, the capacity and utilization of the robots can be changed by reinventing them. The main disadvantage of this self-contained robot is that once customized, it cannot be changed. It continues actively, regardless of whether there is a necessity for amendment in the assignment (in case of crisis). They can be utilized in various operations like versatile mechanical autonomy, manufacturing governance, and spacecraft implementations.



Figure 2-2: A Programmable Robot

A non-programmable robot is one of the most basic sorts of robot. This robot is an exploiter without a reprogrammable control device. Mechanical arms employed in enterprises are examples of these types of robots. As demonstrated in Figure 2-3, robots are frequently added to programmable devices used in organizations for large-scale manufacturing.



Figure 2-3: A Non-Programmable Robot

Industrial robots that can adjust to changing ranges in the process on their own are known as adaptive robots. On the other hand, these robots are more sophisticated than programmable robots. These may be modified to a degree, and once evaluated, they can execute the necessary activity in the adapted zone. These robots are frequently equipped with sensors and control systems. Intelligent robots, as their name implies, are the smartest of all the robots with sensors and microprocessors for data storage and processing. Due to their situation-based analysis and task-performing abilities, these robots' performance is extremely efficient. Pain, smell, taste, vision, and hearing are all senses that intelligent robots can detect. Execute actions and expressions such as emotions, thinking, and learning in agreement. These robots have a variety of uses, including medical, military, and household appliance control systems.



Figure 2-4: An Autonomous Robot

In this section, an outline of algorithms specialized in autonomous navigation in an unknown environment is presented. Various procedures like mapping, localization, and path planning are involved in exploration. A map must be made to control the development of a robot in a specific situation by and large. The versatile robot in an obscure domain and gathering helpful information to develop a map for further navigation is called autonomous exploration.

This chapter structure is as follows, Section 2.2 describes algorithms used for autonomous navigation in the proposed work. Further section 2.3 illustrates

the past studies proposed and implemented in various application areas by different researchers. Moreover, the last section, 2.4, delivers a summary of the chapter.

2.2 Algorithm for Autonomous Navigation in the proposed Work

Conventionally, robots have been fully tele operated by human administrators. However, with an increase in applications of the robots and the diverse scenarios that they have to act, the self-learning route can expand the shots of the achievement. The algorithms intended for path planning consume more energy, diminishing the rate of the robot. Traditionally, obstacle avoidance and path planning were performed in 2D using 2D sensors [Simmons R. (1996); Borenstein J. (1989); Thrun S. (2002)]. Since it is not precise to represent the 3D world into the 2D images; hence 3D images can provide better mapping and navigation of the robots. The path-planning algorithm is a NP-hard [B. Chen et al. (2008)], where the complexity is directly proportional to the degrees of freedom of the vehicle.

From the past studies, it reflects that there were algorithms designed for autonomous navigation such as Dijkstra, A*, D*, Genetic Algorithm (GA), Simultaneous Localization and Mapping (SLAM), Simultaneous Positioning and Mapping (SPAM), Reinforcement Learning, and the list continues.

The impending segment of this chapter elaborates reinforcement learning followed by cuckoo search optimization technique, and symbolic computing would be discussed in the latter part of the chapter.

2.2.1 Q-Learning Algorithm

According to [Sutton and Barto (2018)], Reinforcement Learning is a '*goal-oriented machine learning system*.' To begin the task, initially, it interacts with the environment and then takes decisive steps. Generally, an RL control agent is imbibed with every control problem that communicates with the state of the climate iteratively. The actions modify according to the change of signals as well as the duration of the activities. It depends on the behavior policies π , feedback reinforce reward r which estimates from waiting time, delaying time

and the transit time, following states s' and the RL agents optimize its transition probability P . The policies based on the mapping between possible states S and possible actions A with learning factor, discount factor, and the rewards. During learning, the policies are tuned up with the rewards until it reaches the goals. Below fig.2-5 portrays the illustration of the learning process in reinforcement learning.

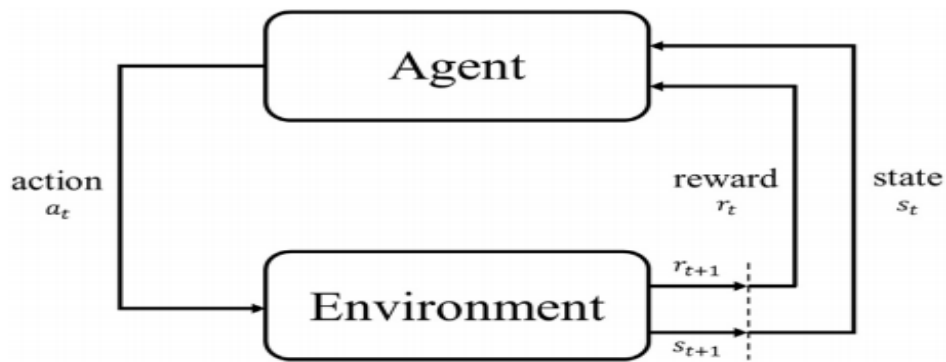


Figure 2-5: Illustration of the learning Process in RL (Sutton and Barto,2018).

A state-action pair recognize control; the reward of each state calculates the value function of the RL. The value of the state is the derivation from the discounted reward in the long term for policy π from state s . It is given as,

$$V_{\pi}(s) = E \left[\frac{R_t}{s_t} = s \right] \quad (1)$$

The above eqn.1 is viewed in the bellman form, and it is given as,

$$V_{\pi}(s) = \sum_a \pi(a \vee s) \sum_{s',r} P(s',r \vee s, a) [r + \gamma V_{\pi}(s')] \quad (2)$$

The value of the action is denoted as 'Q-value,' which consists of a long-term discounted reward on the selected action a and the state s , concerning the policy π is given as,

$$Q_{\pi}(s, a) = E[R_t \vee s_t = s, a_t = a] \quad (3)$$

It is further decomposed into bellman form as,

$$Q_{\pi}(s, a) = \sum_{s',r} P(s',r \vee s, a) \quad (4)$$

About RL algorithms, Q-Learning is one of the well-known RL algorithms that serve as off-policy. Irrespective of the policy, the learning process of the actions is explored independently. In the viewpoint of Q-learning, the RL agent picks the actions 'a' from the possible set of actions A based on the highest Q-value. A greedy searching process computes the Q-values. The computed Q-values are preserved in a matrix referred to as 'Q-table,' which consists of discrete values. It computes the state value s from the set of states S in coordination with action value a from the set of actions A. While computation, the learning policies are updated greedily for every step process. The following equation learns the updated Q-values. (5).

$$Q_k(s, a) = (1 - \alpha)Q_{k-1}(s, a) + \alpha \quad (5)$$

Where,

Q_k - At the learning step k , the Q-value is updated.

Q_{k-1} - Present Q- value stored in the Q-table

r - Present reward value during the k -steps of the learning process.

s - Present state value during the k -steps of the learning process.

a - Present action value during the k -steps of the learning process.

α - An adjustable learning rate.

The detailed process of computing rewards, states, actions, and the learning rate will be discussed. Since the Q-learning algorithm performs at both on-policy and off-policy, it is collectively grouped under the class of Markov Decision Problems (MDP). The MDPs are defined as making decisions in a sequential process based on the selected controlled actions. The stochastic control theory mostly resolves these problems. With the baseline of the work explored by Bellman, the computational complexity of the MDPs is adjudged with the help of Dynamic Programming (DP). It is primarily applicable to large-scale and complex problems that include discrete steps such as Policy Iteration (PI) and the Value Iteration (VI). These are manipulated, preserved, and estimated by the concept of Transition Probability Matrix (TPM). Since the random variables are instantiated, it is quite complex to deal with the probabilities of the transitions. This problem is known as the curse of modeling. The transition elements are more significant in dimension, which impacts the storage and manipulation process. This problem is stated as the curse of dimensionality. Both the problems are challenging in the DPs, which is more reflected in the application areas of the robotic systems.

The system under the MDPs is treated as the chain of action process. In the perspective of the Markov chain, the system navigates randomly from one state to another state at a discrete-time movement. In continuation with the previous states, the current transition will be determined. The current state is determined from the newly updated learning value. Under the subset of states, the system has the responsibility to select the actions from the set of system actions. A policy is established for every state. These states are rewarded with a reward value r , positive, negative, or zero. It helps the system to select the best actions. Therefore, the task of MDPs is to find out the optimal policies

which give an accurate performance measure. Overall, it contributes more to the discrete time taken for the learning process.

It has two famous metrics, discounted reward, and the average reward. The discounted reward is the aggregate sum of the rewards earned on a given discrete time with the pursuit of policy, whereas the average reward is the most awaited reward gained at each step of the learning process. Let $d(i)$ be the chosen action of the state i for the policy d' . Here, the d composes a finite set of states, $|S|$. Let $r(i, a, j)$ represent the expected reward from state i to state j for an action a , and $p(i, a, j)$ represents the probability rate of the same transition.

Definition 1: The policy d' under the discounted reward at the state i is given as:

$$J_{d'}(i) \equiv \varepsilon \quad (6)$$

Where,

x_s is the state prevailed before the s^{th} transition;

γ is the representation of the discount factor.

ε is the expectation operator, which will be updated to derive the optimal solution.

Definition 2: The average reward of the policy d' at state i is given as:

$$\rho_{d'}(i) \equiv \varepsilon \quad (7)$$

The representation of symbols is above the same, in specific to, the average reward is self-determined irrespective of the starting state. By deviating the learning rate, an optimization problem is experienced under the learning process. Henceforth, the findings of the underlying optimization problem become a vital part of the robotic systems.

2.2.2 Cellular Automata

The robotic systems are used for an automation process in various application fields. In manufacturing sectors, some robots are employed for navigation purposes, i.e., randomly moving from one location to another. In that, some cases demand multiple robots to speed up the navigation process. In such a scenario, the different tasks need to be processed. When a robot commits to a task, it moves randomly from its initial position to its destination position under a defined working area. Therefore, navigation control of the robots is known as path planning, which is of prime importance. According to the planned paths, the objective of moving robots has become a fascinating research topic among researchers. The construction of paths as well as navigating the robots to the planned paths are challenging tasks. The cellular automata discover the task of building an obstacle-free optimal path between the source and the destination. In the process of tracking the planned paths, controlling the torque movements becomes critical.

In the course of the path planning process, the design of positions of the robots develops obstacles on the underlying working area. Based on the presence of the information, the planning of the paths may be static (or) dynamic. The knowledge about the obstacles before the navigation control of the robots is known as static path planning. The partial information about the obstacles given to the navigation control of the robots is known as dynamic path planning. The off-line refers to the static process whereas, the on-line refers to the dynamic path planning. Several methods have been developed to find the obstacles at an earlier stage to avoid delays during navigation. Some authors have developed graph-based approaches that connect the source and the destination with the help of intermediates. The edges of the source and the destination points are connected with the possible set of obstacle-free paths. A path, generally, represents a straight line that interlinks the sub-points of the vertices among the source points, destination points, and obstacle points. It is found that the intermediates cause more obstacles before reaching the destination point. A visibility graph is constructed to find out the available

paths, and from these, an optimal path, i.e., the shortest path, is discovered. The shortest path is the one that holds a minimized distance cost, which is the prime motive of the research study.

Once suitable paths are discovered, then the optimal path is selected by different searching algorithms. Along with the graph-based approaches, cell-based decomposition approaches are also established for an optimal path planning process. CA is one of the branches of modern science, representing the high-complex process by its intrinsic discrete nature via digital processors. Cellular Automata (CA) is one of the recent concepts studied in autonomous robot systems. Most of the robotic applications are decentralized by nature. Henceforth, CA establishes new transition rules with the help of neighboring cells and the possible states of the cell. In general definition, it forms a dynamical system by performing local interactions with the components of the navigation systems. Initially, it consists of N cells in a lattice. Each cell possesses a unique identification pattern in its local connections. This typical pattern is used to communicate with other cells. Here, a transition rule for each state of a cell i at time $t+1$ is computed for interacting with the neighborhood cells. Let Z^d be the finite sets of the d -dimensional lattices and Σ be the set of the finite fields. The cells communicated in the lattice are generally represented in map form, and therefore, it is given as,

$$c: Z^d \longrightarrow \Sigma \quad (8)$$

It is further expressed mathematically as,

$s_i^t \longrightarrow$ At time t , the state s of the cell i , which comes under $s_i^t \in \Sigma$.

$\eta_i^t \longrightarrow$ The states of the neighborhood cells and the current cells are stored.

Then, the transition rule of a cell in a lattice is given as,

$$\Phi: \Sigma^n \longrightarrow \Sigma \quad (9)$$

where,

The size of the neighborhood is represented as n .

$\Phi(\eta_i)$ - It gives the information of neighborhood state s_i^{t+1} for the cell i as a function of η^t

These distinctive characteristics in CA have helped for the path-planning process by exploring the goal attraction and local decision making. With the usage of the sensor of robots, the neighborhood's local decision and state at each discrete step are computed by the CA rules, i.e., transition rule. The outcome of the implementation is to navigate the deployed robots to reach a goal point in a straight line. While performing from initial points, an obstacle is faced. To cope with an obstacle, at the same time, reach the goal point as a team has been established through a desirable navigation pattern. The robotic environment is in 2 -dimensional form with a square of cells with side l . In general, the robot's size may be larger than the l . Even so, it is viewed as a single cell. The information about the state of the neighborhood cells is collected using the robot's infrared proximity sensors. It is assessed by two navigation patterns, viz, triangular and linear patterns. Two views employ the transition rules for a robot system:

- a) The first rule: It helps on how to deviate from the obstacles
- b) The second rule: It helps to control the navigations.

These two rules are generalized through master-slave control architecture. The relative position of other robots and their navigation patterns are given by following this architecture. The main robot obtains the position's information of other robots in its team, and then it will decide what rule to apply for further process. Henceforth, this process requires a decentralized control system which is given by CA-based approaches. A navigation pattern is the one, which is eventually deviated from the obstacles. Thereby, the strategies followed in the path planning process are:

- a) Building of neighboring cells
- b) Decision-making process of transition rules
- c) Applying transition rules on actions of robots.

The possible states of the deployed robots are Free (Fr), Robot (Ro), and Obstacle (Ob). Based on the measurement readings of the sensors, the neighborhood cells are formed. The information about the neighborhood cells

are preserved in vector form as, $V = \{v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$ and θ_R stores the current angle of rotation. It is assumed that the five angles of rotation are possible. Consider an instance that a robot is willing to navigate south to the north. Then, the possible angle of rotations is computed as, and shown fig. 2-6.

- a) The initial points v_0 to v_1 is 0°
- b) v_1 to v_2 is 45°
- c) v_2 to v_3 is 90°
- d) v_3 to v_8 is -45°
- e) v_8 to v_7 is -90°

At a given time, step t , two kinds of discrete movements of a robot is possible, the first movement is navigating to its next cell by monitoring the present orientation of cells, and the second movement is rotating the present orientation of cells. During the movement process, the distance is calculated as 1 for neighboring cells and the $\sqrt{1}$ for diagonal cells. Each cell is updated by its current states s , just by defining the next movement of the robots using the transition rule of CA. Overall, a transition rule is the combination of the present state of the neighborhood and the angle of rotation θ_R . With the help of a generated set of rules, i.e., control rule and the deviation rule. Each robot recognizes the neighborhood and attains the target points. If any obstacle is found, then an alternate deviation rule will be used. Else, the control rule will be used.

v_8	v_1	v_2
v_7	v_0	v_3
v_6	v_5	v_4

Figure 2-6: Spatial Neighborhood

According to the deviation rule, it is further encountered with the two sets of rules, and one deals with the movement of the next cell i , and the second deals with the moving of a robot. A pair of rules are being applied over the navigation control of the robots, and the first rule determines the state of the robot within the cell, i.e., the state is free (or) not, and the second rule determines the navigation of the robot if it is free. The below fig. 2-7 explores the two models of robot movements.

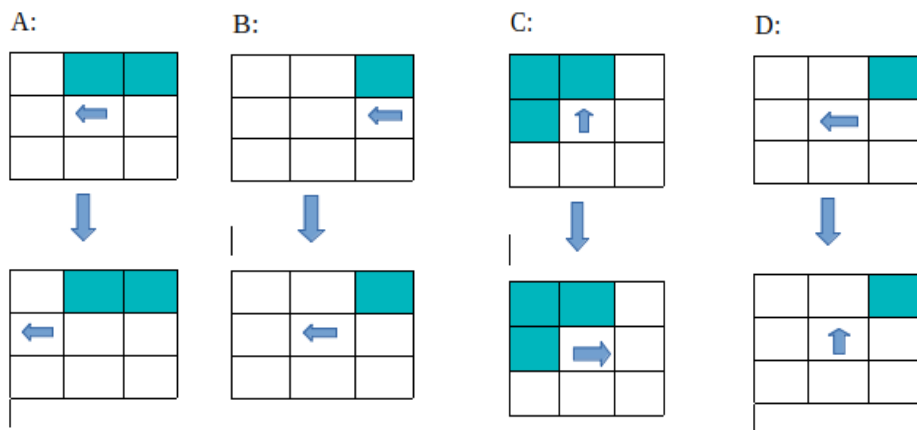


Figure 2-7: Movements of the Robots

The fig. A & B represent the two scenes of robotic movements wherein the pair of rules are applied. The fig. A represents the scene of a neighborhood in which a robot is placed in a central cell. In the second scene, fig. C and D portray the neighborhood with a free central cell. The robots' position and size are denoted using an arrow, green-colored cells are obstacles, and the white cells are free. Let us assume that the robot is willing to be defined from south to the north, then the crossing axis is estimated. The fig. A presents two obstacles i.e v_1 & v_2 and its orientation is -90° , then it moves to the cell v_7 with the same orientation. In the fig. B, central position is the free cells and robot in cell v_3 , then an obstacle at the same orientation is found in v_2 cell. The above two scenes depend on the action of the central cell state, which leads to achieving the goal point. Fig. C & D represents the working of the following transition rule. If the orientation of the robots is different, it is understood that the deviation rule is applied to free itself from obstacles and

then move onto its neighborhood cells. Here, when the robot tries to navigate to the north, three obstacles (v_1 , v_7 and v_8) have to be faced. In such a scene, it rotates the angle to 90° . Similar to that, in fig. D, the robot tries to navigate to the west, an obstacle is identified in its previous state, and thus, it is compelled to rotate towards free cells. As of now, at time step t , cell v_2 acts as the obstacle, which is again compelled to reach its initial orientation.

In this manner, the CA rules are generated for the deployment of multiple robots. Finally, the formation control rule will guide the multiple robots each time step t reaches the goal points. It also helps eliminate the obstacles by taking reverse orientation and making the robot attain its prior paths. It also indicates that the robot will be forwarded if the robot is on-axis; else, it takes diagonal steps to move the robot. However, the main problem is applying deviation rules when an obstacle is found in the planned paths.

2.2.3 Cuckoo Search Optimization

The chief responsibility of deploying cuckoo search optimization is to select optimal paths from the set of available paths. The set of available paths are learned from the Q-Learning and the cellular automata process. Optimization is an essential process for efficient autonomous robots. Since it is an unknown environment, a tremendous amount of variety of obstacles is being populated. Selecting an optimal path in an unknown environment is a complex and challenging task. If the robot needs to attain its goal point without any obstacles, then a different sensor is deployed over the real-time environment. It is a time-consuming as well as a cost-efficient process. The planning of the paths may be local path planning (or) global path planning dependent on the surroundings. The building of global path planning demands a completely known environment, and whereas the local path planning performs even in known and unknown environments. Through various approaches suggested for efficiently finding the paths, it is in under a developmental stage.

The shortest path is the possible path free from obstacles and constraints of the environment, and a smooth straight line is required between the initial point and the goal point. This kind of scenario is projected as an optimization

problem. Here, a recent optimization algorithm known as cuckoo search is explored for the path planning process. Cuckoo Search belongs to the class of metaheuristic search models. Birds' swarming behavior served as inspiration, a searching process is instantiated by the researchers. In specific to, laying eggs on the nest of host birds by its parasitic behavior is explored. It is the most straightforward algorithm compared to the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). It takes fewer simulation parameters than the other optimization models.

Generally, the cuckoo birds follow an intrinsic and aggressive reproductive way. It lays the eggs over the nest of other birds, and that host birds may hatch (or) brood the chicks. In some cases, the host bird destroys the eggs; when the laid eggs are someone else's, (or) it demolishes the nest. A recently created method uses the parasitic behavior of its species to tackle optimization issues. It also uses the levy flight behavior of the birds to build the new nests at each iteration. Levy flight behavior is a behavioral type that deals with the random walk process, wherein each step is scaled up by the distribution of probability. According to the law of power, the probability distribution of levy flight is given as,

$$y = x^{-\beta} \quad (10)$$

Where,

β is an infinite variance that ranges from 1 to 3.

The rules to be followed in deploying cuckoo search are listed as follows:

- a) For a given time t , one egg is yielded by the cuckoo, and, concurrently, it dumps the eggs in another nest in an arbitrary process.
- b) The nest with the quality of eggs is treated as the best nest, and it is forwarded to the next process.

- c) A fixed number of host nests is assumed, and then the egg presented in the nest of the host bird is calculated as a probability $p_a \in [0,1]$.

In some cases, the host birds may build new nests at different locations, and thus, rule c will be maximized (or) minimized based on the random solutions. In the maximization problem, the grade of the fitness value is directly proportional to the objective value. In simple terms, the solution represents the egg in the nest, and the new solution represents the cuckoo eggs. The attainment of a better solution for a nest is the ultimate goal of this system. It can be further extended in the case of resolving multiple eggs in the nest. A novel result is generated by levy flight behavior is given as,

$$X_i^{(t+1)} = X_i^{(t)} + \alpha \oplus \text{levy}(\lambda) \quad (11)$$

Where,

α tends to change its size according to the defined problem area. In most cases, it is generally assumed to be 1.

The above eqn. (11) follows a random walk, which is formulated by the Markov chain process, i.e., its previous location determines the following location of the egg ($X_i^{(t)}$) and its probability of transition ($\text{levy}(\lambda)$). \oplus is the entry-wise multiplication operator, which helps to explore more search space for a longer time.

2.3 The Work Focus on Autonomous Navigation in the past

The literature on autonomous navigation and path planning algorithms illustrates that it is still a domain of research. The researchers are still focusing on the development of an efficient and effective algorithm for navigation. These procedures are regularly required for the arrangement of the local minima issue. Researchers are analyzing distinct efficient ways in which this problem can be resolved. So the recent works on robot navigation are discussed related to various domains in the following section.

The integration of several approaches is necessary for efficient path planning in spatial representation, ensuring a significant and accurate mobile robot

navigation. This paper introduces an optimization algorithm for planning the path of a mobile robot.

Dijkstra's algorithm is used in the algorithm to find the shortest way. In a 3D environment, the robot's size is compared to the size of the obstacle by the proposed algorithm.

[Panda & Choudhury,(2015)]discussed the challenges associated with dynamic motion planning for mobile robots. These challenges are handled through a distinct approach when environments are dynamic by considering the behavior dynamics from a control point of view. The robot interacts dynamically with its local context, which represents the mobile robot existing in motion planning. The process of interaction dynamically models and controls motion planning. The behavior dynamics adjust the motion planning issue of mobile robots powerfully into a reasonable issue where the integrated arranging and control framework is included by bringing a change through an optimization issue in the robot's speeding up space.

[Hosseininejad & Dadkhah,(2019)] uses the cuckoo optimization algorithm to propose a new method that resolves the challenges of planning mobile robot route in a changing scenario. Moreover, currently projected technique uses the feature vector to minimize the computational complexity. Additionally, another technique is proposed which minimizes the feature vector's dimension for minimizing the overall computational complexity entirely.

[Sun, Liu & Leng,(2006)]introduced an efficient algorithm involving shortest path planning which works on planar mobile robots having a time complexity of $O(4 \times n)$ where n is found to denote the geometric complexity of the non-dynamic planar environment. The algorithm employed a limitation as well as the greedy method employed in utilizing the Dijkstra algorithm. The shortest path generated by the method is high-speed and hence can be enhanced.

[Tharwat et al. (2018)], a model based on the Bézier curve is proposed for route mapping. The decision points present in the Bezier curve impact the distance and flatness of the track altogether. A new algorithm named specific

Chaotic Particle Swarm Optimization (CPSO) algorithm is proposed to enhance the control points of the Bézier curve. The proposed calculation gives two variations, specifically CPSO-I and CPSO-II. The selected points enable the ideal even path to limit the absolute distance evaluated between the beginning and ending points. The conventional PSO algorithm is compared and checked with results generated by CPSO-I and CPSO-II algorithms to assess the CPSO calculation.

[Mo & Zu,(2015)] consolidated another hybrid optimization algorithm BPSO with an accurate Voronoi boundary network to propose another robot path planning approach. The method for condition modeling is shown. Following this, the Modified BPSO decides the best method dependent on AVBN. Compared to other algorithms, the proposed method comprises a much faster convergence speed and minimum failure rate. Further, intelligent optimization proposes a new approach to solve RPP.

[Ghosh et al. (2017)] employ an autonomous mobile robot to achieve optimal path by proposing two efficient, intelligent optimal controllers consisting of bat algorithm (BA) and flower pollination algorithm (FPA) in an unfamiliar condition. FPA is designed by taking into consideration the process of pollinating flowering plants where various pollinators transport pollen. BA solves different kinds of optimization problems in engineering through echolocation and frequency tuning. A fitness function is considered autonomously for achieving the path-planning task of a mobile robot by considering the distance between the robot and the obstacle or between the robot and the goal satisfying the criteria of obstacle avoidance. The robot shows the goal achiever's behavior. The mobile robot considers the values provided by the objective function to avoid obstacles in an unfamiliar environment and arrives at its goal.

[Das et al. (2015)] employed hybridization of improved particle swarm optimization (IPSO) in combination with an improved gravitational search algorithm (IGSA) for multi-robot to determine the optimal trajectory of the determined path in a cluttered environment. IPSO possesses social

characteristics which the proposed approach incorporates into the movement of IGSA. The developed hybridization IPSO-IGSA maintains the suitable equilibrium between searching and overuse due to the adoption of co-evolutionary techniques for enhancing the expedition of IGSA and particle positions combined with IPSO velocity together. The algorithm diminishes the maximum path length. It also reduces the time of arrival of each robot to its distinct destination in the environment. The robot generates individualistic decisions by employing the proposed hybrid IPSO-IGSA to understand and communicate to identify the following positions from their current location in the world map.

[Zhang et al. (2016)]proposes a modified ant colony algorithm to arrange the path of a mobile robot in a perceived stable condition. The modified ant colony algorithm expands the searching through a range that debilitates the local minima issue, enabling the algorithm to focus quickly. The turning element is likewise considered in the optimal path searching process.

[Das et al., (2016)] a new methodology to solve optimization problems is proposed along with several evolutionary algorithms, including Genetic Algorithms, Differential evolution algorithm gravitational search algorithm, PSO, and Bee Colony Optimization for application in the problem of planning of multi-robot path. There are two parts in the fitness function of the GSA to avoid the impact which occurs when robots collide with static obstacles. They are the fitness function depicting the way toward choosing the following position on an optimal trajectory through the estimation of speed, and different incorporates the restrictions on acceleration. Newtonian's law of gravity and movement helped to make the design of the heuristic algorithm PSO. There are various changes in GSA, and these changes have several applications. Currently, the original version is improved by the different variants of GSA. The algorithm has also found application in many areas. There are apparent targets of the different robots, and these robots have PSO where an overall fitness function is created in a multi-robot way arranging the issue. This fitness function decides the accompanying position of the robots, which are remaining in ideal directions and moving towards the individual

objectives. The path-planning problem emerges in the circumstance when an iterative algorithm gets a request to decide the accompanying position of the considerable number of robots by settling every one of the imperatives existing in the multi-target work. The algorithm continues repeating until each robot arrives at its destination. In the molecule swarm streamlining algorithm, various new highlights are added to improve it to decide the way direction for different robots, which uniquely characterizes beginning positions to explicitly picked objective views in the earth to limit the way length of the considerable number of robots.

The outcome uncovers how the algorithm has improved the arrangement quality inside a reasonable timeframe. The particle swarm optimization algorithm (IPSO) gets enhanced to make arrangement with the way arranging issue of the multi-robots, leaving all around by expanding the combination rate. Ultimately, the recreation has demonstrated the productivity of the IPSO with the Khepera condition, and the outcome is contrasted and different algorithms, including a PSO and DE. IPSO method offers vigorous execution, self-deterministic interaction and deals with a problematic domain in the multi-robot framework dependent on a dynamic structure. IPSO proposes a path-planning plan to ideally get the accompanying states of the robots from the present position in the intended condition.

[Zheng et al. (2016)] proposed an improved ant colony algorithm where the infinite step length exists to estimate the optimal path. It focuses mainly on the drawbacks of the standard ant colony algorithms, such as determining the single step length to recognize the optimal path inclined towards the local optima and weak convergence. There exists an increased chance of selecting a path of ant along with optimizing the results. The heuristic information adopts multiple priorities and utilizes choose/ grid mode to adopt a modified update mode of local information.

[Sudhakara & Ganapathy, (2016)] determines the best path for moving from a start state to a destination state through a new optimization technique for a robot with no collision with obstacles. Modifications are carried out on the

existing A-star algorithm so that the robot can travel in an unfamiliar environment that comprises static obstacles. The robot is assumed to move to the destination position without colliding with any of the obstacles present. The Enhanced A* algorithm follows an optimal path to help the robot to reach the target.

[Parhi,(2018)] performed a review analysis on navigational methodologies of robots through different artificial intelligence techniques, including Neural Network, Particle Swarm Optimization (PSO), Fuzzy Logic, Genetic Algorithm, and additional Artificial Intelligence techniques. During the review analysis, the review was carried out systematically, and the role of several artificial intelligence techniques was utilized to control and navigate different kinds of robots facing different environmental conditions.

[Song et al., (2017)] uses η^3 -splines along with a modified particle swarm optimization (MPSO) to propose a new approach. At first, η^3 - splines are utilized for including an arbitrary arrangement of points where the kinematic parameters are chosen to relate with the movement and the control of mobile robots. The MPSO algorithm comprises adaptive random fluctuations (ARFs) and is, for the most part, created to control the frequently oversaw local convergence. The issue is the system of arranging the smooth worldwide way. The MPSO algorithm has shown the evolutionary state incorporating classification averagely at each iteration provided by the evaluated evolutionary factor. There exists a switch in the velocity improving dynamics for varying modes as per the evolutionary state with the ARFs. These are imposed on the global/local best particles in a way suggested by the current iteration.

[Naz, et al.,(2018)]describes the networks of few modular robots which are lattice-based and only use neighbor-to-neighbor interactions. These networks develop into sparse and huge breadth networks. Furthermore, tight limit to the diameter and the breadth of these networks. There exists a crucial design issue through complex distributed networks due to the large diameter and the vast average distance of massive-scale lattice-based networks.

[Huang et al. (2019)] reviews various techniques associated with mobile robots in WSNs. It helps scholars understand the flow within every category, the relationship among different solutions, accurate information, and in-depth analysis. The distinctions and likenesses between the accessible approaches are compared past various classes regarding scientific formulation, application, and so forth.

[Chen & Chiu, (2015)] developed a map, planned optimal paths, and designed mobile robots through an optimal robot path planning system. The system designs a grid-based map by collecting the information from several foundation and still hindrances. The system estimates the optimal flight by utilizing a basic neural network model and developing a mobile robot. The mobile robot determines the ambient conditions for dynamic obstacles and avoids possible collisions.

[Yuan et al. (2019)] develop a mobile robot collision avoidance algorithm by utilizing the characteristics of improved ACO and APF and figuring a novel GRU-RNN system model to finish the changing route development of moveable robots in a new domain. The GRU-RNN system has recognized the principal framework's arranging approach, and there is an all-out presentation in light of the use of the improved ACO and APF calculation.

[Thai Duong et al. (2020)] focusses on real-time occupancy mapping and collision testing for an autonomous machine navigating in an obscure environment. A novel mapping technique suggested for engaged and unrestricted space. A distinct decision boundary is available for the spaces available.

2.4 Summary

This chapter focuses on past research in autonomous navigation. Various algorithms were used for navigation, and the advantages and limitations were discussed. In addition, it dealt with algorithms used for the design and development of the self-learning algorithm. In the next chapter, the design and development of the model for autonomous navigation.

CHAPTER 3

MODEL DEVELOPMENT FOR AUTONOMOUS NAVIGATION IN AN UNKNOWN ENVIRONMENT

The chapter covers the design and development of a new optimization technique using the Q-Learning and Cellular automata algorithms.

3.1 Introduction

In the modern era, the scope of robotic applications has been increasing gradually with the possibility of technology developments. In the current scenario, robotic applications are least deployed due to certain complexities faced by the deployer during deployment actions, such as setting motion variables, planning of paths, and box pushing. It also consumes heavy computational time. The applications of robots are widely adopted from engineering fields to the agricultural fields. Planning the path is a critical task in robotic applications, and it proves to be challenging to create a setup environment. Thus, the researchers are more fascinated with the study of path planning in robots by avoiding the obstacles. Many techniques are invented to control the movements of autonomous vehicles. In order to identify the obstacles, the touch sensors and infra-red are used in its working area. Radio Frequency (RF) sensors observe the position of the target area. Along with these lines, machine vision approaches are deployed for the accurate position of the robots

In the course of autonomous robot applications, the finding of accurate positions becomes a challenging task. It belongs to the class of heuristic approaches. The concept of the heuristic approach is to resolve the sub-problems by learning the past experiences to cope with the solution of new problems. Henceforth, different learning models are established, such as reinforcement learning and deep learning. It is observed that the minimized fabrication procedure and the localized navigation models are of prime importance. The wide use of robots is noticed in the industrial sectors. The key objective of the robotic arrangement is to operate collaboratively without

any obstacles. When the robots are performing collaboratively, the automation tasks become relatively simpler and more manageable. The advantages of collaborative learning based robots are presented as follows:

- a) It offers a distributed solution, i.e., reduced time, space, and automatic functions.
- b) The parallel performance of robots has ignited the different tasks of different applications.
- c) Duplicating potentialities has increased the robustness and the reliability of the other robots.
- d) It supports different kinds of operations with more expandability.

Despite its numerous benefits, the most common issue that prevails is route development for circumventing the barriers (or) planning the paths according to the industrial application requirements. In this context, 'robot navigation' becomes an active and fascinating research area. The prominent emphasis of this work is to discover the unswerving track among the existing paths. The significant metric of the robot path planning is estimating the time taken to start from the target and reaching the targets. With the association of different sensors, efficient environments have been created by the robotic system. In the implementation, cellular automata and Q-learning concepts are some of the reinforcement learning models that have been explored. In addition to that, one of the well-known optimization techniques, the Cuckoo Search algorithm, has also been explored here.

3.2 Proposed Framework

The section portrays the implementation of the proposed design of a robot. Fig. 1 presents the workflow of the proposed navigation function of a robot.

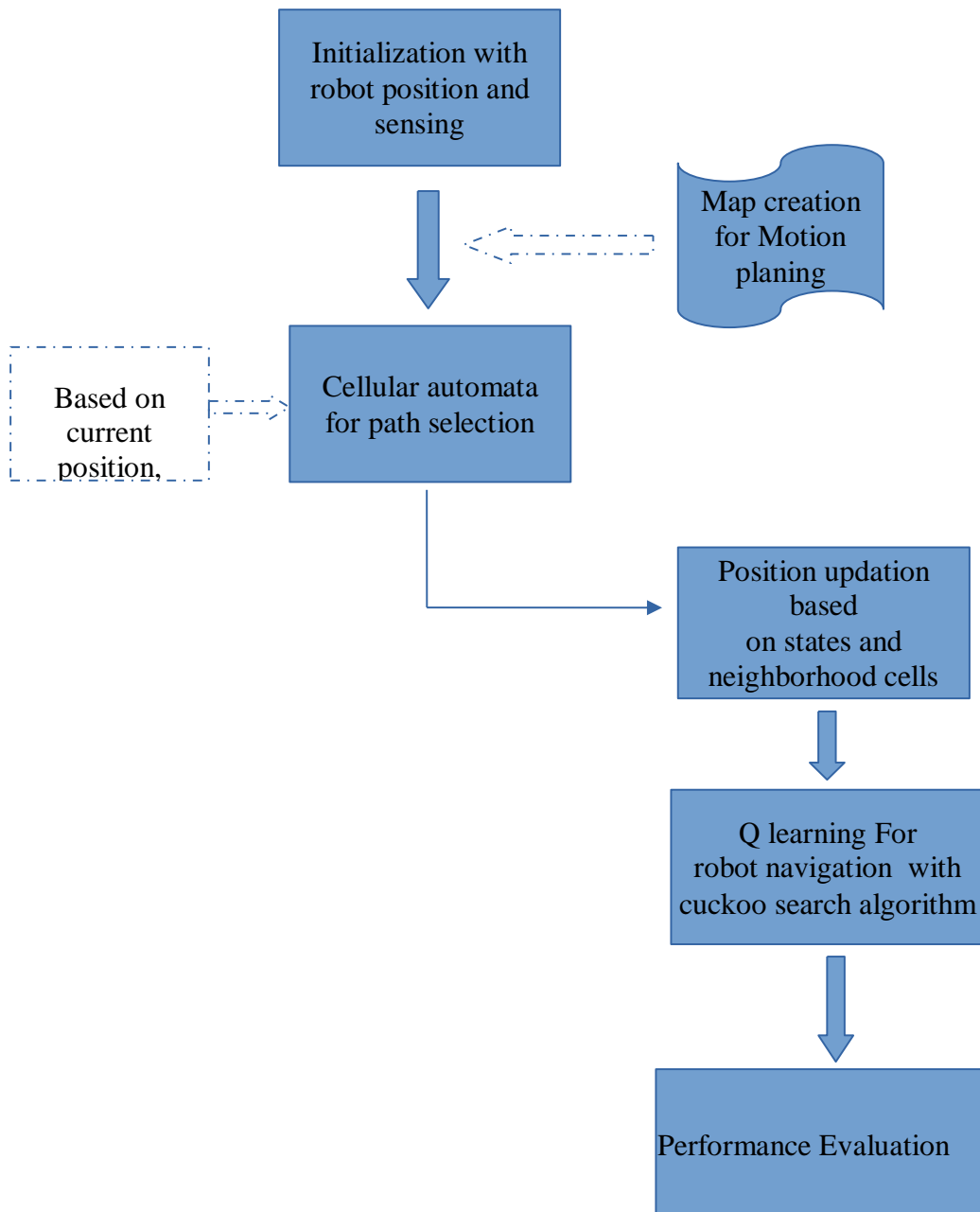


Figure 3-1: Workflow of the proposed algorithm

Steps of the proposed framework

Step 1: Initiating the position of the robots by their built-in sensing devices.

Step 2: Generating the 3D maps for the planning of motion.

Step 3: Applying cellular automata to build the paths.

Step 4: Dividing the dynamic environment into cellular decomposition.

Step 5: Each cell is anticipated as an interactive component that consists of states and actions.

Step 6: Constructing the set of available paths with the help of the Q-learning system.

Step 7: Performing the CSO parameters to select the optimal shortest path.

Step 8: Finally, the performance measures are taken by introducing the obstacles and without obstacles.

Table 3-1: Steps of SAARTHI Algorithm

Algorithm: The novel approach for autonomous navigation in an unknown environment.

Input: A= No. of actions, $S < R$: No. of actions to be selected,
 $(\lambda(j))_{j \in N}$: Rate of the learning, $(\eta(j))_{j \in N}$: Rate of the averaging
factor, $(\gamma(j))_{j \in N}$: Rate of the discount factor.

Initialize the population: n host nest

Output:

While end condition is not satisfied,

1. for each cell [i], synchronously with other cells, do:

(i) Perform an exploratory selection with probability $\alpha(k)$,

(a) According to CLA, choose a random super action as [i]

(b) Using greedy selection procedure, obtain a maximum

expected reward for the action $\alpha_i(k)$

8. Finding the set of paths by observing the cells.

9. for Robot Cell (C) \neq Target Cell (T)

10. Find the best cell

11. Evaluate with the free cell

12. Finding the paths $P = \sum_{i=0}^p S$

13. Initiating the set of paths as Cuckoos

14. For ($F_i == \text{Best}(f(N))$), $F_i ++$)

16. Obtain the best solutions, i.e., best path (P)

17.end for

18.end for

19.end for

19.end While

Table 3-2: SAARTHI Algorithm

3.2.1 Model and Assumptions

The robots have knowledge on only about their original position. The sensors deployed in it have to detect the obstacles within its environment automatically. The unknown environment is divided into grid cells with a size of 20 cells. The built-in sensors of the robot sense these cells during the central placement of the robots, the cells of an entire area are covered. During the experiment, it is also assumed that the range of the robot's sensor can recognize the obstacles in the three-dimensional orthogonal view of the neighboring cells. During robots in a cell, an individual spanning tree is administered for each exploration process of a cell. The present location of the cell is occupied as input to the algorithm, and the output is the spanning tree of the robot. Since it scans by orthogonal views, all positions, i.e., up/down/ top/ bottom, are visited. The below fig. 3-2 illustrates the sensing directions of the cells, where it depicts the position of the robots (RP), Obstacles (O), and the intermediate cells (a,b, and c).

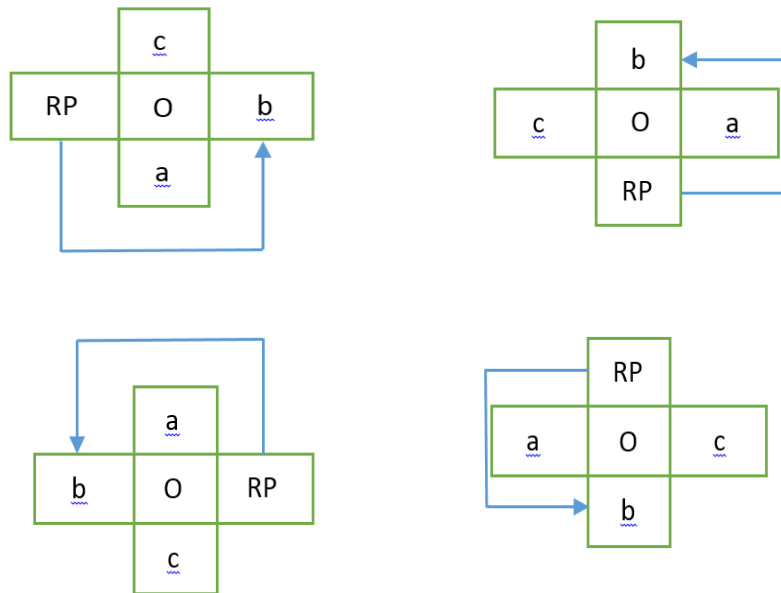


Figure 3-2: Sensing the direction of the cells

Here, the robots can label the visited cells, enabling the differentiation of the covered and uncovered robot cells. With the help of intermediate cells, the

sub-cells are also covered by the robots in a defined terrain. The covered cells are frequently updated in the Q-table. This table will permit us to easily cover the uncovered cells by constantly pushing the cells into stack S. Later on, according to the experiments, the covered cells are taken up for the other analytic process, viz, path planning. The deployed robots are homogeneous by nature, in which the cells are accessible from any position. It is also assumed that the robots implement independently due to the design of decentralized algorithms. The following are the constraints of the proposed algorithm for an unknown environment.

A planar structure of terrain is considered, which is continuous and encircled with an outer boundary. It is partitioned into 2D square cells, which composes of 4- subcells. Robots are moving continuously between two adjacent subcells. It follows a complete path-coverage, i.e., the starting and ending point of the sub-cells remains unique. The robot takes only one move at a particular period.

3.3 Environment Learning

The living creatures are emerging and developing sustainable ecosystems by abiding with the known and unknown environment. Along with this growth, the robots are designed based on adapting to the dynamic environment. The intelligence of the robots performs coordination with the processing ability of perception, decision-making, and the action over the environment. The entire structure of the intelligent capability of the robots has been achieved by the skill and knowledge acquisition capabilities of the robots within a dynamic environment. Therefore, the scope of defining an intelligent and interactive learning module about the robotic environment plays a crucial role in the manufacturing sectors.

Learning can be ensured by discovering, interacting, and observing the environment. In human beings, the knowledge is inherited from their parents and children exploring via teaching. It takes the inheritance of knowledge by offspring. Therefore, symbolic information processing is employed to percept intelligent robotic systems. In some cases, logical information processing [R.

Bormann, (2016)] is done because of the inference, which might reduce the state of the data such as vagueness, fuzziness, and the paradox. Likewise, intuitive information processing is also done by dealing with incomplete data, fuzzy and feeble interference, and information synthesis. The information analysis of logical and intuitive information processing takes a little higher computational effort and time. Henceforth, it can be declared that the organization of the information processing is also essential. Terrain Exploitation and the Coverage analysis are the two prominent sectors that determine the efficiency of industrial robots. It gives an overview of the behavior postulated by the robots presented in the alignment of developing digital technologies. The intelligence of the robots is acquired from their learning and experiences of the past conducts.

On the learning and adaptation process, the established intelligent techniques (E. Castello, 2013) are explained as follows:

Science deals with the brain system: It portrays the process of biochemical and physical models of the human brain. The connection among the nerves dictates the function of the human brain. Hence, it is named neural networks.

Soft Computing: Zadeh establishes that it deals with the information processing units. Some objective functions are created and analyzed over the collected information to enumerate the functionalities of the application requirements.

Artificial Life: It is explored by three approaches, namely, molecular level, cellular level, and organism levels. Each level is exposed using software and hardware mechanisms.

Computational Intelligence (CI): It depends on symbolic data rather than knowledge acquisition. It is operated from three modes, artificial, biological, and computational.

Based on the above technologies, numerous techniques are explored, which are summarized in brief. Mainly CIs are employed to develop an intelligent

robotic system from self-adaptability, evolution, and intelligence. It explores the description of the internal and the externals of the intelligence approaches. It stimulates the functions of the human brain, and thus, information accuracy is ensured. In recent times, Neural Networks(NN) and Fuzzy Systems (FS) are the widely used mechanism to simulate the human brain. The cognitive patterns of the brain are used for training and the pattern recognition process.

In some cases, it also identifies incomplete patterns. The domain analyzes the role of the dynamic system, ‘neurodynamics,’ which develops a non-linear mapping [P. Chand, (2013)]. Likewise, neural networks are composed of different interconnected layers among input, output, and activation units. It helps for applications dealing with the recognition, controlling, and pattern matching models. The psychological features of the brain will be simulated for modeling purposes.

3.4 Summary

This chapter summarizes the design and development of the system. Further, it discusses the importance of environment learning. Communication between the environment and the robot is also discussed. The intelligence of the robots is acquired from their learning and experiences of the past conducts. It is understood from the discussion; Map Generation will have some effects on resolving the terrain exploitation. The implementation will be discussed in the next chapter.

CHAPTER 4

IMPLEMENTATION OF THE SELF-LEARNING SYSTEM

This chapter covers the implementation of the autonomous machine system in a dynamic environment. The preceding chapters have discussed the fundamentals design of the Q-learning, Cellular Automata (CA), and the Cuckoo Search Optimization (CSO) straightforwardly. Here, the implementation of the algorithms mentioned above under a robotic environment has been portrayed with an implementation scenario.

4.1 Exploration of terrain and the coverage of the robotic environment

Most robotic applications nowadays require robots to intelligently choose the best course by exploring and understanding their surroundings. At the same time, the robots must eliminate the barriers in order to reach the objective spots in a specific period. In some cases, with the help of past learning and experiences, it should respond to dynamic requests. By adopting technological developments, multi-robots are being deployed in an unknown environment, which offers a higher level of scalability, robustness, and redundancy reduction. Along with that, path exploration and coverage are handled effectively to achieve a collaborative environment. Depending upon the area covered by the deployed sensors, the robot can learn the environment, and it is also not facilitated with any prior information. While sensing the environment, robots share many aspects of the exploration process. Despite that, exploration covers the boundaries of the environment, even remote sensing, whereas coverage senses the working area of the environment. It is a need that the designing of the proposed algorithms should meet the requirements of performance enhancement of multi-robot instead of optimizing the sequence number of actions.

Coverage analysis will help out to build an efficient path planning process. The geometrical properties of the coverage area should be identified and performed with the cellular decomposition of the environment. Once the cells are decomposed, then searching-based coverage algorithms are deployed. Due

to the positioning of furniture and the dynamic change of human actions, the coverage algorithm produces a high level of complexity in an unstructured (or unfamiliar) setting. Therefore, the work on assuring an obstacles-free environment with the capabilities such as communication, location-knowledge, planning of paths, and collision avoidance is prime important. The objectives of this work is to find the best shortest path for a mobile device in a new (or changing) environment.

4.2 The proposed scheme implementation

The proposed scheme, a “novel cuckoo search optimization using Q-learning,” tries to develop an intelligent selection of an optimal shortest path for the robots in a changing environment. The proposed scheme is a kind of self-learning process. The scheme performs incrementally by constructing multiple Q-table for multiple robots at concurrent time steps. It is assumed that each of the robots is presumed from its starting positions ($S_1 \dots S_k$) of the terrain. The challenge of this work is to discover the optimal express route from the k-sub paths of the k-robots. On the other side, it is cautiously monitored that the robots cover the entire terrain. Along with these lines, the robot covers each of its sub cells only once.

The behavioral properties of the robots are explained in a detailed manner for the proposed scheme. During the navigation process of the robots, some controlled actions are performed, which are known as behavioral properties. Here, four behaviors are recognized: choice of the cells, identification of the boundary, avoidance of the obstacles, and movements. These behavioral properties are explained as follows:

(a) Choice of the cells: The selection of the target cell is the goal of the work. The present position and the information about the neighboring cells are obtained from this behavior. The search action is performed heuristically, in which the uncovered cells are covered by continuous exploration of the robots under the terrain.

(b) Identification of the boundary: Here, the obstacles are assumed to be static. It combines with the obstacle avoidance properties to update the robots about their previous and current boundaries.

(c) Avoidance of the obstacles: The choice of the cells does this. If any obstacles (or) boundary are identified, the robot aims to eliminate the collision among the working area by moving to the unoccupied space of adjacent cells.

(d) Movement: With the help of the Q-table, the robot intends to reach its target points via subcells.

The proposed schemes are explained as follows:

(A) Q-learning algorithm:

This algorithm aims to learn about the behavior of the cells and their subcells during the navigation process of robots. This algorithm takes trial and error on the defined parameters. The deployed robots initiate the working process by just learning the environment via the reward and penalty process. Here, the selected action is accurate; then, the learned data is represented as ‘reward.’ Else it is labeled as ‘penalty.’ The Markov Process does this learning process, and thus, decisions made from this process are known as ‘Markov Decision Process.’ The term ‘Markov’ is coined because it portrays the future states depending on the data provided by the present states. Let $S = \{S_1, \dots, S_k\}$ be the set of states and t be the present time of the states. Then, the probability state of the Markov process is given as,

$$P[S_{t+1}|S_t] = P[S_{t+1}|S_1, S_2, \dots, S_t] \quad (12)$$

Since the robots are homogeneous by nature, the markov chain under the transition probability is independent towards time t , which is given as,

$$P[S_{t+1} = s'|S_t = s] = P[S_t = s'|S_{t-1} = s] \quad (13)$$

Let the tuple (S, P) be the Markov chain process, in which S represents the finite set of the states map and the P be the transition probability matrix of the state. The proposed Q-learning algorithm maximizes the rewards of a robot

by taking a consequence series of actions in the dynamic robotic environment. The rewards are maximized by learning the environment and presenting it in the form of transition maps. Fig. 4-1 represents the block diagram of the proposed Q-learning algorithm.

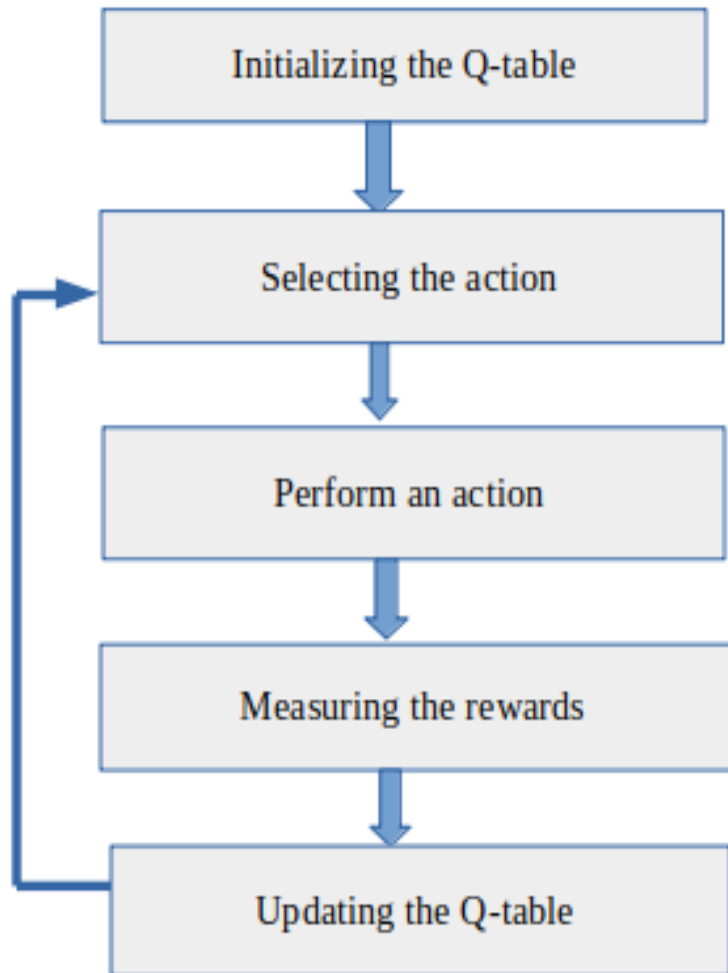


Figure 4-1: Block diagram of the proposed Q-learning algorithm

Step 1: Initializing the Q-table

The learning process is started by creating the Q-table. It is projected as $m * n$ matrix, where m represents the number of states and n represents the number of actions. The sample Q-table is illustrated in fig.4-2.

Action	Up ↑	Down ↓	Left ←	Right →
Start	0	0	0	0
Idle	0	0	0	0
Correct path	0	0	0	0
Wrong path	0	0	0	0
End	0	0	0	0

Figure 4-2: Sample Q-table

Figure 4-2 portrays the sample Q-table. It consists of four actions, viz, up, down, left, and right, and five states, viz, start, idle, correct path, wrong path, and the end.

Step 2: Selecting an action

Depending on the robotic environment, the learning process begins by selecting any one action at a time t .

Step 3: Performing an action

Here, the actions are performed based on time steps. The robots learn the environment by collecting the previous and past states of the cells by taking a series of actions. Initially, the Q-value in the Q-table is represented to be 0. Consider an instance, (action a) right (→) performed to start (state s) the process, then the Q-table is updated by using the Bellman equation. The exploration process begins by performing a greedy strategy on the dynamic

environment. Therefore, the sample updated Q-table according to the Bellman equation is shown in fig. 4-5.

Action	Up ↑	Down ↓	Left ←	Right →
Start	0	0	0	1
Idle	0	0	0	0
Correct path	0	0	0	0
Wrong path	0	0	0	0
End	0	0	0	0

Figure 4-3: Updated Q-table

Step 4: Measuring the rewards

A crucial step that helps to build an efficient path planning. It is evaluated by the function $Q(s, a)$. The learning rate is determined at the initial step. The function $Q(s, a)$ will continuously learn the environment until the criteria of learning rate terminates. The maximized Q-value of an action at that state is rewarded. The bellman equation is given as:

$$Q(s, a) = Q(s, a) + \alpha [R(s, a) + \gamma \max_{a'} Q'(s, a) - Q(s, a)] \quad (14)$$

Where,

$Q(s, a)$ ----> Updated (new) Q-value for the state and action

$Q(s, a)$ ----> Present Q- values

α ----> Rate of learning

$R(s, a)$ ----> Rewards taken for action at a state

γ ----> Rate of discount

$maxQ'(s, a)$ ---> Maximum Expected future rewards

Here, the rewards are symbolized as follows,

(+ 1) ---> When the current step is closer to the goal

(-1) ----> When the current step hits the obstacles

(0) ----> When the current step is in idle state

Action	Up ↑	Down ↓	Left ←	Right →
Start	0	0	0	1
Idle	0	0	0	0
Correct path	0	43	12	0
Wrong path	12	0	23	0
End	0	0	1	0

Figure 4-4: Final Q-table

Figure 4-6 represents the sample final Q- table. In this way, the Q-learning algorithm has helped to find (or) learn about the unknown environment. The work is known as collecting the data for the training process using the Q-learning algorithm. The estimation of reward value portrays the robust learning model that enhances the performance of the system. Likewise, the discount factor helps estimate the data by an agent, which is greatly helpful to deal with the shortest path planning. It reduces the effects of data overfitting. Fig. 4-7 presents the workflow of the Q-learning algorithm.

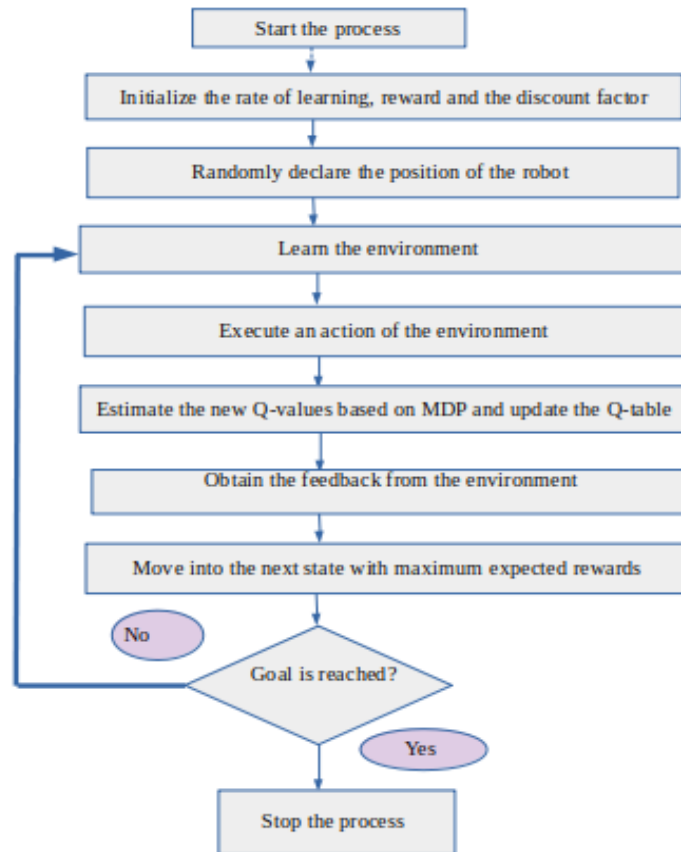


Figure 4-5: Flowchart of the Q-learning algorithm for navigation

(B) Cellular Automata & Cuckoo Search Optimization:

Once the training data process is created, then the position of the robots has been explored by the concepts of cellular automata. Here, the objective becomes more complex because of finding the optimal actions from the set of actions. Since the environment is projected into a grid of cells, each cell contributes as a learning component. During the learning process, the best actions are learned to recommend the neighboring components for further actions. For the experiment, a parallel reinforcement learning process makes an optimal action selection from a set of actions. By doing so, it can manage all kinds of actions and preserve the time and space complexity of the environment. Once the optimal action is selected, the optimal shortest path is analyzed using Cuckoo Search Optimization (CSO).

Initially, the position of the robots by acting in a state is explored using cellular automata. It is a hybrid learning model that combines the learning

process and the automation process of a robot's cell in a dynamic environment. With the help of Q-learning, the automation process by determining the local rules. The research objective is to explore a total capability of maximizing the expected reward function for each learning automation process. Let the aggregate number of actions at cell i be a_i and the action to be chosen be ω_i . Then, the set of actions of cell i is denoted as $A_{-i} = \{a_{-i1}, \dots, a_{-iz}\}$. Let Δ_i be the policy strategies associated with it and n_i be the number of neighbors of cell i . At step k , the probability of selecting A_{-i} is given as $p_{ij}(k)$ and β be the policy strategy over A_{-i} . And the average actions of reinforcement learning at time step k is given as Q_i . Let $N_i(1) \dots N_i(n)$ be the index value of the neighboring cells.

The learning system operates on the probability of the set of actions. Initially, the joint distribution of the actions was estimated by the maximum empirical rewards. It conjoins with other learning automata, even on neighborhood cells. The obtained LA combines with the probability of joint distribution to form a single optimal action (or) super-action. The super-action increases gradually by the selected probability of super-actions. During the exploration process, the probability of the super-actions is acquired. Each selected action in the neighborhood is continuously observed, and their empirical values are updated frequently. Every performed action sent feedback from the environment to its learning component. Depending upon the obtained feedbacks and the probability distributions, the estimated Q-table is updated.

Based on the selection super-action, a mutated greedy selection process is performed. Initially, the maximum expected reward action must be selected. Here, let us assume $\alpha_i(k)$ be the present action. Then, the new action from the selected action $\alpha_i(k)$ is computed as

$$p_l = \frac{1 - q_l(k)}{\sum_{j \in L} 1 - q_j(k)} \quad \forall l \in L \quad (15)$$

Where,

$$q_h = \frac{Er(a_{ih}, \underline{p}_{n_i}(k))}{\sum_{a_{is} \notin \alpha_i(k)} Er(a_{is}, \underline{p}_{n_i}(k))} \quad \forall h \in \left\{ \frac{l}{a_{il}} \in A_i \wedge a_{il} \notin \alpha_i(k) \right\} \quad (16)$$

The above eqn. (15), is continuously operated for all maximum expected reward of super-action. Then, the reinforcement learning vector $\beta_i(k)$ is calculated for the chosen super-action=1, as below.,

$$p_{ij}^{k+1} = \left\{ \begin{array}{l} (1 - \gamma(k))p_{ij}^k + \gamma(k) \text{ if } \alpha_i(k) = a_{ij} \\ (1 - \gamma(k))p_{ij}^k \text{ otherwise} \end{array} \right\} \quad (17)$$

$$Q_i(k+1, :) = Q_i \quad (18)$$

$$Q_i(k+1, a_{ij}, \alpha_{(j)}(k)) = Q_i(k, a_{ij}, \alpha_{(1)}(k), \dots, \alpha_{(j)}(k)) + \eta(k) \left(\beta_{ij}(k) - Q_i(k, a_{ij}, \alpha_{(1)}(k), \dots, \alpha_{(j)}(k)) \right) \quad \forall j \in \left\{ \frac{l}{a_{ij}} \in \alpha_i(k) \right\} \quad (19)$$

The above process is continued until the stopping criterion meets. During step k, the selected super-action is learned via the learning component, i.e., neighborhood cells. Since each action is performed separately, the learning process becomes more viable. The probability of empirical distribution of all super-action is then given as input to the cuckoo search optimization technique. The constructed maps have assisted in tracking their position and planning time of the navigation. In order to achieve the main goal of the research, this optimization technique has been beneficial in an unknown environment. Before moving into the working of an optimization technique, the proposed learning automata are given in Algorithm II.

Steps of cellular automata for path selection

Step 1: Declaring the initial cell as the current cell with a state s and the action a . Each cell is a learning component.

Step 2: Initially, all cells are marked to be 'not visited.'

Step 3: Finding the unvisited neighboring cell with the maximum expected reward points.

Step 4: If the neighboring cell is not found, then labeled it as 'visited' and stopped the learning process.

Step 5: If the neighboring cell with maximum expected reward points \leq Current cell with maximum expected reward points, then labeled it as 'visited' and stop the learning process.

Step 6: Re-declare the current cell as a neighboring cell.

Step 7: Terminates the process until all cells are visited.

Step 8: Estimate the set of planned paths.

Table 4-1: Steps of cellular automata for path selection

Path planning is carried out on some barrier constraints, which are known as obstacles. With the help of discrete steps of time, the robots can navigate on the planned paths. Likewise, the learning rate is defined to be less than the sensing range of the sensor associated with the robot. At each step, robots look for the availability of obstacles in the navigation process. Along with that, the position of the obstacles is also estimated. If the robot finds any obstacles with its distance step, then the cuckoo search model suggests a new obstacles free path. Here, the point of the obstacle and the existing position of the robot is declared as the initial point. The path is planned between the new starting point and the target point connected with new intermediate points. Then, an optimal distance value is estimated if the robot does not face any obstacles

during the navigation process. Here, the path exploration process is done by two processes, namely, local navigation and global navigation process.

(a) Local search navigation process:

Here, a conventional line that contains the preliminary point and the destination point assists the robotic machine. The robots follow this route until it recognizes the obstacles. The detected obstacles will be resolved by deviating from the present position of the robot, which is frequently updated by the learning component. Then, a new position and new distance will be rationalized for assisting the upcoming robots. In this process, the robots already know the position of the target point.

(b) Global search navigation process:

It is a planning process where the robot moves randomly with the prior information known to them. It is the procedure for choosing the optimized mode to reach its target points by information loaded on it. Henceforth, the judgement for the movement of the mobile device from the start and the end is predetermined and directed into the environment. The fig. 4-8 presents the sample scenario of the need for optimization technique. It is clearly understood from the figure, in order to resolve (or) eliminate the obstacles presented in the robotic environment, an optimization algorithm is required. Here, different dimensions of test cases are explored to find an optimal shortest path.

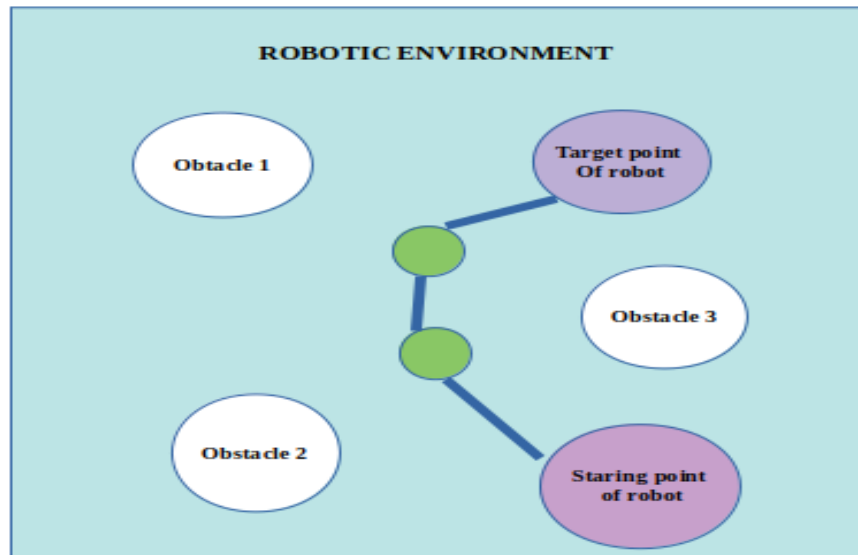


Figure 4-6: Sample scenario of the robotic environment

Steps of Cuckoo Search for Robot Movement

- a) Initializing the paths with known obstacles, set of super-actions, and the robots.
- b) Initializing the parameters, maximum iteration (I_{max} Which is always less than the sensing range of the robot).
- c) For each time step k , the robots are moving randomly in association with the planned paths. While doing so, the robot senses the obstacles within its sensing range.
- d) If the obstacle is not recognized on the planned path, it moves to the last step.
- e) The robot is traversing through its planned paths. If the obstacle is found, then the middle coordinate of the device is treated as the starting point. The initial location of the automaton is considered to be the local search value. The local search value is estimated for all robots under the learning component. Finally, the number of

intermediate paths with obstacles-free is found, then step (c) is described as the optimum path.

- f) The robot randomly moves onto the next step. If the straight line of the path (between two cuckoos) followed by the robot is always lesser than the (I_{max}) . Each cell will be covered at each time step to range the next intermediate point.
- g) The machine moves until the destination point is attached (or) proceeds with step (c).

Table 4-2: Steps of Cuckoo Search for robot movement

4.3 Implementation of the test cases

During the execution, different dimensions of the map are being constructed. Each map was tested under the robotic environment. The proposed map consists of 10 sets of states with ten sets of actions. Assume that the speed of the obstacles is greater than the speed of the robot navigation. The obstacles are projected as static obstacles, which are captured by the shapes like rectangles. Here, it contains three grid points, i.e., starting, target, and intermediate points. If the obstacles were found, then the intermediate point is taken as the starting point. Initially, the motion of the obstacles is arbitrary, and thus, the exact position of the robot is not measured accurately. Therefore, the robot finds the place of obstacles by learning the cells. In order to avoid a collision, the cells with minimum distance are considered. If the robot does not find the obstacles, it moves into following neighboring cells. The process continues until it reaches the target points.

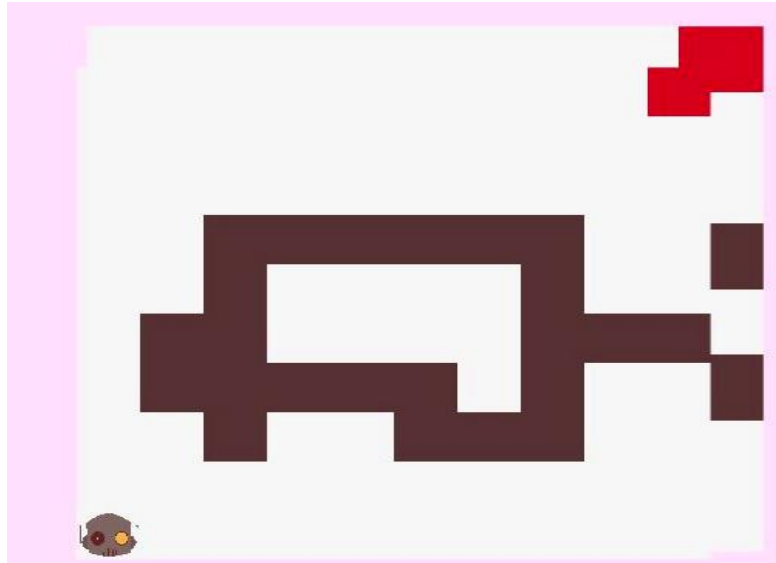


Figure 4-7: Illustration of the simulating environment

Here,



----> It represents the robots



----> It represents the obstacles



----> It represents the targets

The environment simulation infers the following benefits. The robot can attain its target point, even in the presence of obstacles. In some cases, dynamic obstacles are also created. Thus, the formation of the rectangular shape of the grid points has helped the robot find its paths to attain the targets.

The deployed robots can also eliminate (or) resolve the obstacles from the opposite directions, i.e., obstacles presented in the inner boundary and an outer boundary. It ensures that the robot takes the obstacles free- movement. It also guarantees that the quickest way is chosen; ensuring that the least amount of money and time is spent. It is beneficial for all sorts of dynamic environments. The proposed scheme takes discrete control over the robotic

learning agents, which is helpful for an efficient dynamic path planning process.

Each mapping point on the map has been estimated from its local association with the neighboring cells. The preserved data in each cell point is the present computation between the target point and the neighboring points. Its distance determines it. The information being updated by distance as well as grid point has helped the neighboring points. Thereby, it is concluded that the computation process is independent by the nature of arrangements of the grid points. The current order of the updated grid points gives the global knowledge of the current and previous grid points. Likewise, the path of the robots is estimated from its real-time information, and thus, the sensor mapping value may be subject to the changes in another dynamic environment. The computing work required to update each point is small, allowing for fast propagation of distance information from the target sites outward along the grid.

4.4 SUMMARY

This chapter has discussed the implementation of the proposed scheme in the defined robotic environment. The pseudo-steps of the proposed scheme with an illustration are described. The next chapter will be discussing the simulation parameters, software used, and the performance measures in continuation.

CHAPTER 5

RESULTS AND DISCUSSION

The chapter portrays the outcomes and discussion of the suggested hybrid algorithm. In association with the previous chapters, the evaluation of the proposed algorithm is explained in a detailed manner.

The Autonomy for a robotic navigation is a good research area because of the benefits offered to real-time applications. In general, perception, the navigation of the robot comprises four primary requirements, namely, sensitivity, positioning, perception, and route planning and controller of movements according to path planning. In work, path planning is the leading research area triggered by the use of heuristic algorithms. The conventional methods have leveraged the decomposition of cells, strength fields of a robot, sub-goals, and the road map. Though the approaches were simple, the challenges pertaining to this field still existed due to the expensive computational models and uncertainty.

Path planning in a robot is continuous movements with the translation and rotation points from origin points to the target points without any obstacles. The paths were exploited using local and global search analysis. Generally, global path planning deals with the construction of high-level paths for the developed environment maps. It helps to achieve an optimized path; however, it is more prone to unknown (or) dynamic obstacles. In contrast, local path planning works on track construction without any prior information. Though it resolves the complexities of a dynamic environment, the long-distance target point is not effectively reached. In the area of robotic movement control, global path planning is mainly used for accomplishing tasks. Different path planning algorithms were available such as the configuration space approach, artificial potential approach, cell decomposition approach, intelligent control approach, and the sampling approach. In specific, an intelligent control approach is employed to attain the target points. Along with the coordination of 3D map generation systems, the proposed hybrid algorithm is designed and

tested in the dynamic robot environment. The environment simulation is processed out using a software platform, Matlab 18.

5.1 ROLE OF MATLAB IN ROBOTIC ENVIRONMENT

With the rapid advancement of the Industrial Revolution 4.0, there have been more efforts to maximize manufacturing processes with cost minimization constraints. Industrial robots are proliferating among manufacturers in different regions. The recent developments made in computers have also enhanced the solutions to the problems related to automating industrial applications. The efficient algorithms were developed to overcome the challenges in robot development for different purposes. It could be viewed from the other aspects, i.e., the demanded tasks on engineering problems were resolved by the efficiency of the computers and the numerical steps processed out in the developed algorithm. The conventional systems use kinematics and the concept of the dynamic of robotic systems by abiding by the principles of analytical processes. Owing to the current technologies, the requirements of the uses take more complex approaches that deal with the mechanical problems of Coupled Mechanical System (CMS) [Alzbeta Sapetova et al. (2018)]. It includes variants of topology synthesis, planar mechanism, and other features. The examination of the numerical models with some algorithmic procedures is simulated and modulated according to the robotic constraints. Matlab, a high-level programming language, is mainly explored by the researchers to simulate the robotic environment in a creative approach, thus defining how the fittest solutions are obtained.

It includes software packages based on the analysis of robot manipulators such as robot arms, vehicles, and mobile robots. It helps to resolve most of the kinematic and dynamic examinations over the manipulators. Nowadays, researchers have been investigating the intensive development of the robotic fields' parallel and hybrid kinematic structures. With the assistance of the technologies, the areas in the robotic environment are explored efficiently. The algorithms of intelligent management systems trigger control elements.

From the perspective of structural optimization, the mechanisms involved in the robotic systems are performed as follows:

- i) To percept the conditions and the expediency factor of the robotic environment. The conditions are important in the CMS, and, on the other hand, the expediency factors depend on the agents that validate the purpose of the mechanisms.
- ii) To estimate the criteria for optimum settings. The quality of the agents is analyzed in order to gain the optimum constraints.
- iii) To illustrate the geometry and the variables of the dependent and independent systems. It has helped to retrieve the constraint functions.
- iv) Here, every point has some draft space which is also known as draft variables. It helps to find out the deviation of the path trajectory.
- v) To develop a mathematical model for describing the optimization tasks, which is represented as,

$$\text{minimize } F(x)$$

$$\text{Subject } g_j(x) \leq 0, j = 1 \dots m$$

- vi) To choose an accurate mathematical optimization method is a complex task. Depending on the time parameters, the optimum constraints tend to be dynamic variables. In some cases, arbitrary values are not permitted to describe the mathematical models. Thus, a different local search algorithm was formulated to construct the optimal parameters.
- vii) In addition, some technical interpretations are done and formulated into the language of engineers. It is employed as a real-time application.

5.2 RESULTS AND DISCUSSION OF THE ROBOTS IN A DYNAMIC ENVIRONMENT

This section describes the simulation of the shortest path finding using 3D map generation concepts. Here, model-based reinforcement learning and the Heuristic algorithms are formulated, implemented, and tested using Matlab

programming. The sample result images are presented to describe the simulated output paths.

As discussed in previous chapters, the robot can move freely underneath the obstacles (or) it simply takes steps to understand the perceptual capabilities of the information presented in its surroundings. The acquisition of a reliable and flexible information process is a difficult job for a dynamic environment. Therefore, 3D environment maps were generated from the collected data, which has helped for an accurate range of data into the segments of planar coordinates. The segmented data has been used to estimate the height of the floor, which is further integrated with the 3D grid developed from different measurements. This sort of representation splits the environment of the robot into grids. Grid is the combination of cells presented in a row and column manner. Each cell possesses data about the type of environment. It comprises several regions like walking areas, stepping up, stepping down using stairs, obstacles regions include safe and unsafe objects. Therefore, the application of this navigation map is used for route development strategies and the removal of hurdles.

To formulate the problem of robotic mapping, the basic assumptions to be followed. The robot has some in-built sensors that capture the data like position and orientation and analyzes the environment. Next, to develop the systems, accurate knowledge about the position and the robot's orientation are fed into its other coordinate systems. The below fig. 5-1 presents the construction of maps. Here, x denotes the position and direction of the current pose of a mobile device, z denotes the sensed data over a particular period, and m denotes the iterative mapping process for a while.

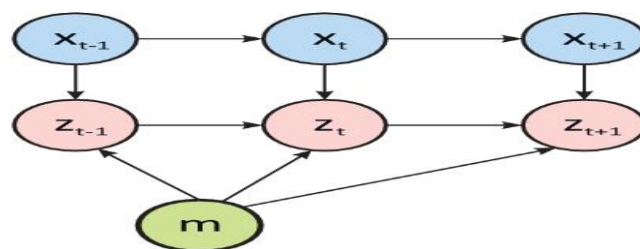


Figure 5-1: Construction of the map

During the experiment, a 3D-based occupancy grid is employed to examine the robotic cells. Each element in the matrix denotes the present status of the cells, i.e., filled, void (or) uncovered. The below table 5-1 below presents the sample input data of the working environment.

<i>Points</i>	<i>Sample coordinates of data</i>	<i>Denotation</i>
C_0	(4, 8)	Origin position
C_1	(9,17)	Obstacle 1
C_2	(14, 3)	Obstacle 2
C_3	(16,5)	Obstacle 3
C_4	(22,9)	Target Position

Table 5-1: Sample Input Data of the working environment

A code has been written using Matlab programming to achieve the research objective, i.e., to find the shortest paths. The robotic environment is segmented into several grids. Some points are randomly assumed as input data points, obstacles points, and output data points. With the help of these points, the resultant paths are created by learning the movements of the cells using the Q- Learning algorithm.

The paths were generated using the Bellman equation and the Markov Decision process of the Q-learning algorithms. The obtained paths were not optimal due to the invasion of obstacles. These obstacles were explored in sequential order. Thus, a possible combination of the data points was estimated for the obtained data points. Here, the origin and the target points were set to be static. The obstacles are set to be dynamic by nature. The main goal of this experimentation is to discover the fittest path from the set of

possible combinations of data points (or) in response to the formulated obstacles.

Before moving into path generation analysis, let us discuss the assumptions made in the dynamic robotic environment. The followings are the rules of the robots while incorporating them into the dynamic environment.

- a) Dynamic environment: The sensing area is unknown (or) unstructured.
- b) Restricted communication: Direct communication between robots is not possible.
- c) Restricted sensing: Low-cost sensing devices are equipped with robots.

In order to make an efficient sequence of decisions, this algorithm takes trial and error on the defined parameters. The deployed robots initiate the working process by just learning the environment via the reward and penalty process. Here, the selected action is accurate, and then the learned data is represented as 'reward.' Else it is labeled as 'penalty.' The Markov Process does this learning process, and thus, decisions made from this process is known as 'Markov Decision Process.' Once the cells are examined, it is marked as visited. Based on the cells' status, the cell's action is labeled with reward (or) penalty.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
S1	0	0	0	0	0	0	x	0	100	0
S2	0	0	0	0	0	0	x	0	1	0
S3	0	0	0	0	0	0	x	1	1	0
S4	0	0	0	0	1	1	x	1	1	0
S5	0	0	0	1	1	1	1	1	1	0
S6	0	0	0	1	x	1	1	1	1	0
S7	0	0	0	1	x	0	0	0	0	0
S8	0	0	0	1	x	0	0	0	0	0
S9	0	x	x	1	x	0	0	0	0	0
S10	0	1	1	1	x	0	0	0	0	0

Table 5-2 Reward Table for MAP 8

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11
S1	1	1	1	x	0	x	0	x	x	1	1
S2	1	x	1	x	0	x	0	x	x	1	1
S3	1	x	1	x	0	x	0	x	x	1	x
S4	1	x	1	x	0	x	0	x	x	1	x
S5	1	x	1	x	0	x	0	x	x	1	x
S6	1	x	1	x	1	x	1	x	1	x	x
S7	1	x	1	x	1	1	1	x	1	x	x
S8	1	x	1	x	1	1	x	x	1	x	x
S9	1	x	1	1	1	1	1	x	1	x	x
S10	1	x	x	1	1	x	x	1	x	x	x

Table 5-3: Reward Table for MAP 10

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
S1	0	0	0	0	0	0	0	1	1	100
S2	1	1	1	1	1	1	1	1	1	1
S3	1	1	1	1	1	1	1	1	1	1
S4	1	1	x	x	x	x	x	x	0	0
S5	1	1	x	0	0	0	0	x	0	0
S6	1	x	x	0	0	0	0	x	x	x
S7	1	x	x	x	x	x	0	x	0	0
S8	1	1	x	0	0	x	x	x	0	0
S9	1	1	1	0	0	0	0	0	0	0
S10	1	1	1	0	0	0	0	0	0	0

Table 5-4: Reward Table for MAP 12

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
S1	0	0	0	0	0	0	0	1	1	100
S2	1	1	1	x	1	1	1	1	1	1
S3	1	1	1	1	1	1	1	1	1	1
S4	1	1	x	x	x	x	x	x	0	0
S5	1	x	x	x	x	x	x	x	0	0
S6	1	x	x	x	x	x	x	x	x	x
S7	1	1	x	0	x	x	x	x	0	0
S8	1	x	x	x	0	x	x	x	x	x
S9	1	1	1	x	x	0	0	0	0	0
S10	1	x	0	0	0	0	0	0	0	0

Table 5-5: Reward Table for MAP 14

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20
S1	0	1	1	1	1	1	x	1	1	1	1	x	1	1	1	1	x	1	1	100
S2	1	x	x	1	x	1	1	1	1	x	1	1	1	1	x	1	x	x	1	x
S3	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	x	1	1
S4	1	x	0	x	0	x	1	1	1	1	1	1	1	1	x	x	1	x	0	x
S5	1	x	0	x	0	x	1	x	x	x	x	x	x	1	1	x	1	1	1	0
S6	1	1	1	1	1	1	1	x	0	0	0	0	x	x	1	x	0	0	0	0
S7	x	x	x	x	x	x	x	0	1	1	1	0	0	x	1	x	x	0	0	0
S8	0	1	1	1	1	1	1	x	1	x	1	x	0	x	1	x	0	x	0	x
S9	0	1	1	1	1	1	1	x	1	x	1	x	0	x	1	x	x	x	x	x
S10	x	1	x	x	x	x	1	x	1	x	1	x	0	x	1	x	0	0	0	x

S11	0	1	x	1	1	x	1	x	1	x	1	x	0	x	1	x	0	0	0	x
S12	0	1	1	1	1	x	1	0	1	x	1	x	0	x	1	x	0	0	0	0
S13	0	1	1	1	1	x	1	1	1	x	1	0	0	x	1	0	0	x	0	0
S14	0	1	1	x	1	x	x	x	x	x	1	x	x	x	1	x	0	x	0	0
S15	x	x	x	x	1	x	0	1	1	1	x	0	1	1	x	x	x	0	0	0
S16	0	0	1	1	1	x	0	1	x	x	x	x	1	x	0	x	0	0	0	x
S17	0	0	1	1	1	x	1	1	x	1	1	1	1	x	0	0	0	0	0	0
S18	0	1	1	x	x	x	1	x	x	1	0	0	x	x	0	x	0	x	0	x
S19	1	1	1	x	0	0	1	x	x	1	x	x	x	0	0	0	0	0	0	0
S20	1	1	1	x	0	0	1	1	1	1	x	0	0	0	0	x	0	0	0	x

Table 5-6: Reward Table for MAP 20

The above tables 5-2-5-6 present the rewards for the given state and the actions for maps 8,10,12,14, and 20. Initially, a default map shows the current state and the robot's actions at the stipulated period t . The reward table is also known as the 'Q-table.' In this approach, the Q-learning algorithm has helped to have insights into the unknown robotic environment. This information about the state and action of the cells will be viewed as collecting the data for the training by means of the Q-learning algorithm. The estimation of reward value portrays the robust learning model that enhances the performance of the system. Likewise, the discount factor helps estimate the data by an agent, which is greatly helpful to deal with the shortest path planning. It reduces the effects of data overfitting.

After finding the robot's current position via cellular automata and Q-learning systems, a training database is created. In order to ease the success rate of the defined objectives, a set of paths is generated based on the reward table. Since the setting is projected interested in a lattice of cells, each cell contributes as a learning component. During the learning process, the best actions are learned to recommend the neighboring components for further actions. This type of analysis is known as the reinforcement learning model. It can manage all kinds of actions and preserve the time and space complexity of the environment. Now, the possible set of actions, the shortest paths are generated. Here, 20 maps were generated using 3D map generation systems. Out of 20 maps, some maps hold the optimal set of actions, i.e., Map 8, Map 10, Map 12, Map 14, and Map 20. For these maps, the evaluation of the fittest shortest paths and their time consumption rate is explored.

Path planning is carried out on some barrier constraints, which are known as obstacles. With the help of discrete steps of time, the robots can navigate on the planned paths. Likewise, the learning rate is defined to be less than the sensing range of the sensor associated with the robot. At each step, robots look for the availability of obstacles in the navigation process. Along with that, the position of the obstacles is also estimated and updated. If the robot finds any obstacles with its distance step, then the cuckoo search model suggests a new obstacles free path.

Here, the point of the obstacle and the present location of the machine is declared as the preliminary point. The track planned is sandwiched between the new initial point and the destination point in connection with new intermediate points. Then, an optimal distance value is estimated if the robot does not face any obstacles during the navigation process. Below table 8.7 presents the various combinations of the obstacles on the generated paths using Q learning with Cellular Automata.

<i>Data points</i>	<i>Combinations</i>					
	1	2	3	4	5	6
	C0	C0	C0	C0	C0	C0
	C1	C2	C1	C1	C2	C2
	C2	C3	C1	C1	C3	C2
	C3	C2	C3	C1	C2	C1
C0,C1, C2, C3 &C4	C4	C4	C4	C4	C4	C4

Table 5-7 Obstacles Combination on generating the paths

With the above table 5-7, the total length from the origin to the target points, i.e., $C[u]$, is estimated using Euclidean distance value. The resultant paths are generated according to the Bellman equation in Markov Decision Process (MDP). The mapping point on the map has been obtained from the neighboring cells. The current data in each cell point is the estimation between the target point and the coordinating points determined by the Euclidean distance values. Based on the updated distance value from the obtained information, it has helped the coordinating points. Thereby, it is concluded

that the computation process is independent by the nature of arrangements of the grid points. The current order of the updated grid points gives the global knowledge of the current and previous grid points. Likewise, the path of the robots is estimated from its real-time information, and thus, the sensor mapping value may be subject to the changes in another dynamic environment. A combination of 20 Maps was generated, and the sample map values are explored. The robots are observed under the Markov Decision Process. Here, ten states and ten actions are commonly employed to observe the robots' motion to reach the target points. Each mapping point on the map has been estimated from its local association with the neighboring cells. The preserved data in each cell point is the present computation between the target point and the neighboring points. Its distance determines it. The information being updated by distance as well as grid point has helped the neighboring points. Thereby, it is concluded that the computation process is independent by the nature of arrangements of the grid points. The current order of the updated grid points gives the global knowledge of the current and previous grid points. Likewise, the path of the robots is estimated from its real-time information, and thus, the sensor mapping value may be subject to the changes in another dynamic environment. The computational work required to update each point is small, allowing for quick propagation of distance information from the goal position outward along the grid.

Map No.	Distance traveled by the robots per cell
8	15
10	17
12	38
14	31
20	129

Table 5-8: Distance Traveled by the robots

From table 5-8, it is inferred that the estimated euclidean distance of the generated paths ranges between 14 to 130m. The generated map takes the least distance values to perform the navigation task since these values are estimated by the sum of the distance taken from an origin point to reach the target points despite all obstacles using Cellular Automata-based Q-learning systems.

The confusion matrix derived for the selected maps, which proves the accomplishment degree of the proposed procedure. It is the size of $n * n$ matrix that defines the decision to take an exact path and the observed path, when the target's position is altered. Here, n is the count of successively taken routes. Initially, the success rate of the estimated routes engaged by the robots to reach the destination point is given in the table below 5-9 and 5-10.

	<i>Q-learning with CA(Existing)</i>					<i>Q-learning with CA & cuckoo (proposed)</i>				
	Map 8	Map 10	Map 12	Map 14	Map 20	Map 8	Map 10	Map 12	Map 14	Map 20
Target 1	42	1	2	2	3	50	0	0	0	0
Target 2	2	40	3	2	3	1	47	1	0	1
Target 3	2	2	39	3	4	0	1	47	0	2
Target 4	2	2	2	41	3	1	0	0	47	2
Target 5	2	3	3	5	37	1	1	1	1	46

Table 5-9: Success Matrix for Q-learning between the existing and the proposed model

Table 5-9 represents the success matrix. Here, it defines the shortest route taken by the deployed robots when the position of the targets is altered. There are five target points assumed in the experiment. When target one is taken as input, the shortest path in Map 8 has assisted the robots to reach it. Likewise, target two as input makes use of Map 10 than Map 8, 12, 14, and 20. Here, the values in the table represent the success count of the robots to accomplish the tasks.

	<i>Q-learning with CA(Existing)</i>					<i>Q-learning with CA & cuckoo (proposed)</i>				
	Target 1	Target 2	Target 3	Target 4	Target 5	Target 1	Target 2	Target 3	Target 4	Target 5
Map 8	0.84	0.02	0.04	0.04	0.06	1	0	0	0	0
Map 10	0.04	0.8	0.06	0.04	0.06	0.02	0.94	0.02	0	0.02
Map 12	0.04	0.04	0.78	0.06	0.08	0	0.02	0.94	0	0.04
Map 14	0.04	0.04	0.04	0.82	0.06	0.02	0	0	0.94	0.04
Map 20	0.04	0.06	6	0.1	0.74	0.02	0.02	0.02	0.02	0.92

Table 5-10: Confusion Matrix between the existing and the proposed model

Table 5-10 represents the confusion matrix for the above-explored success matrix. Here, the value 1 dictates the equalized actual and observed shortest paths; value 0 represents the maps that do not use the varied target & floating values representing the error values, i.e., deviated paths taken by the robots. It is measured in terms of True Positive (TP) and False Negative (FN), estimated from the success matrix. True Positive (TP) measures the success rate of correctly reaching the targets for the given Maps; False Positive (FP) measures the success rate of incorrectly reaching the targets that were incorrectly labeled with the wrong set of maps. True Negative (TP) measures the success rate of incorrectly reaching the targets for the correct set of Maps & False Negative (FN) measures to reach the target points erroneously labeled as wrong maps correctly. Here, there are five maps and five target points in which each map consists of 10 states and ten actions discussed in the earlier section.

In the work, the robots are deployed in an unknown environment by giving prior information of their origin point. With the help of built-in sensors, the task of the deployed robot is to select the paths to attain the target position intelligently, irrespective of its environment, i.e., known (or) unknown environment. Initially, the cells are divided into grid forms. Each grid consists of 20 cells. The built-in sensors of the robot sense these cells. During the central placement of the robots, the cells of an entire area are covered.

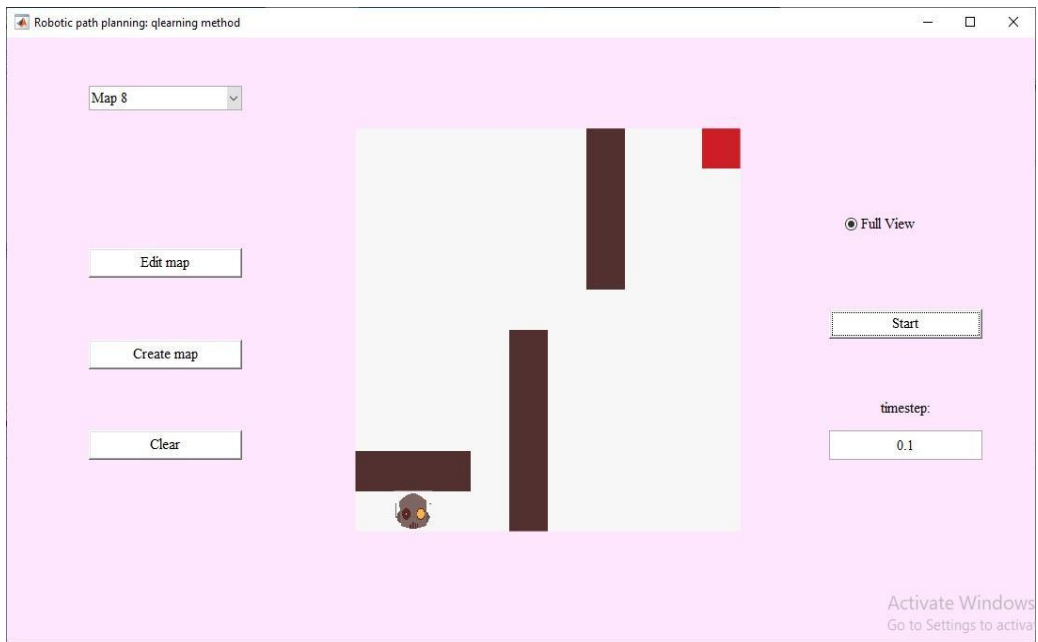
Along with that, it is also assumed that the range of the robot's sensor is capable of recognizing the obstacles in the three-dimensional orthogonal view of the neighboring cells. During robots in a cell, an individual spanning tree is administered for each exploration process of a cell. The present location of the cell is fed as input to the process, and the output is the spanning tree of that robot. Since it scans by orthogonal views, all the robots, i.e., walking style, step up, step down, and idle movements, are keenly observed.

Time consumption analysis is the performance measure explored in work. Several studies have explored optimizing robot path planning strategies. However, the operation time is not mainly concentrated by the researchers. The reduced time consumption has shed light on providing a better energy utilization rate, which is relied upon by the actions taken by the robots. Practical action on the unknown robotic environment has significantly reduced the time taken for building the paths. The multiple robots were deployed in an unknown environment, and therefore, the efficient analysis of discovering the fittest shortest paths is determined by the time consuming, which is implemented here. Generally, the deployment of a single robot will not bring any challenges to the environment, whereas deploying a multi-robot, the challenges like collisions, computation overheads, and the heavy time is taken to reach the target position. There are several motivations for on-time analysis of multi-robot systems.

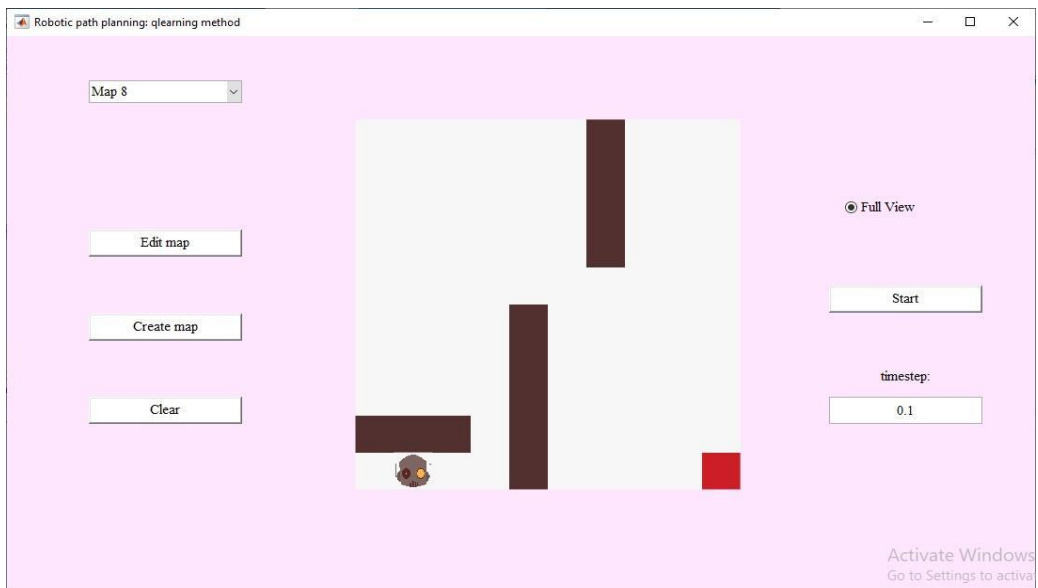
The navigation of mechanical device is complex by nature, even for solitary robotic systems. For multi-robot systems, robots' movements become more and more complex, which demands the parallel execution of transportation

loads. Even though some tasks are monotonous, the execution of parallel processing has exposed irrelevant consumption of resources. To improve the robustness and flexibility of the robotic systems, the need for optimal resources and optimal time have a prominent part in the accomplishment level of the robotic applications. The thesis covers the issues of geometric pattern formation, terrain coverage, and task decomposition and allocation. With the help of cellular automata, the coverage issue has been resolved. In continuation with this, the Q-learning algorithms explore issues of geometric pattern formation and the optimal actions of the deployed. The above tables have shown the results and analysis of the cellular automata-based Q-learning algorithm. The shortest paths are computed and explored by the distance taken from origin and target position from the given set of planned paths. Here, optimality refers to the time consumed by the robots. On proceeding to this, the upcoming screenshots present the working of the cuckoo search algorithm.

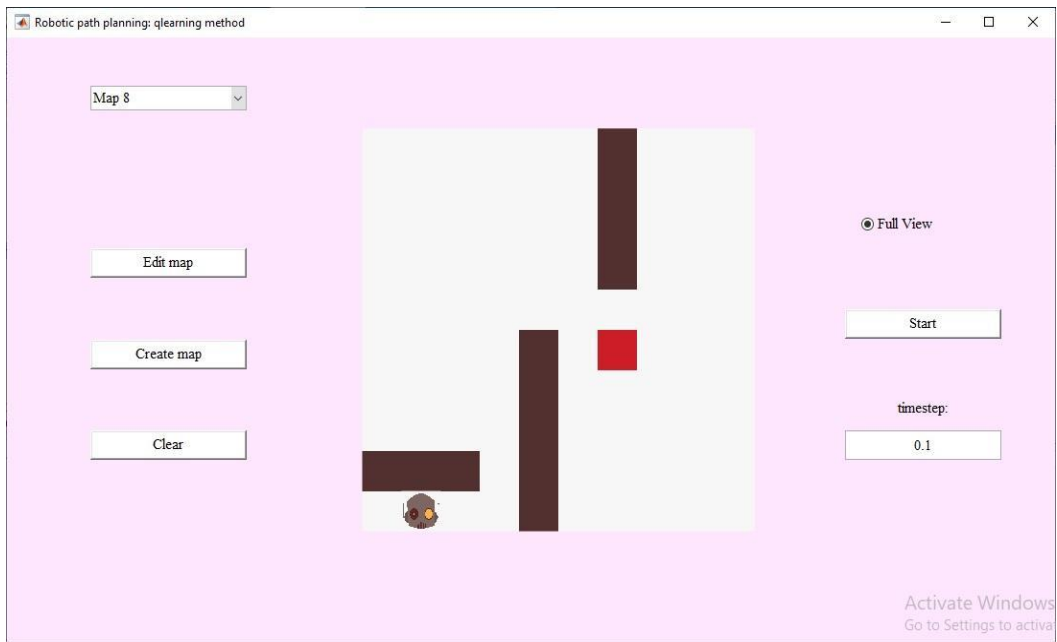
The simulation constraints of cuckoo search algorithms are fed into the Matlab environment. The population of the cuckoos taken here is 10. Each cuckoo operates by knowing its map, current position, and target position. Along with the initial iteration starts, the time step is set to 0.1 sec. The variable, *sbest*, preserves the best solution of that particular iteration and its time step. The cuckoos are exploited in the robotic environment with their input variables by implementing the levy flight behavior. It performs random walks on the environment until the termination criteria are reached. The below figures portrays the working of each map.



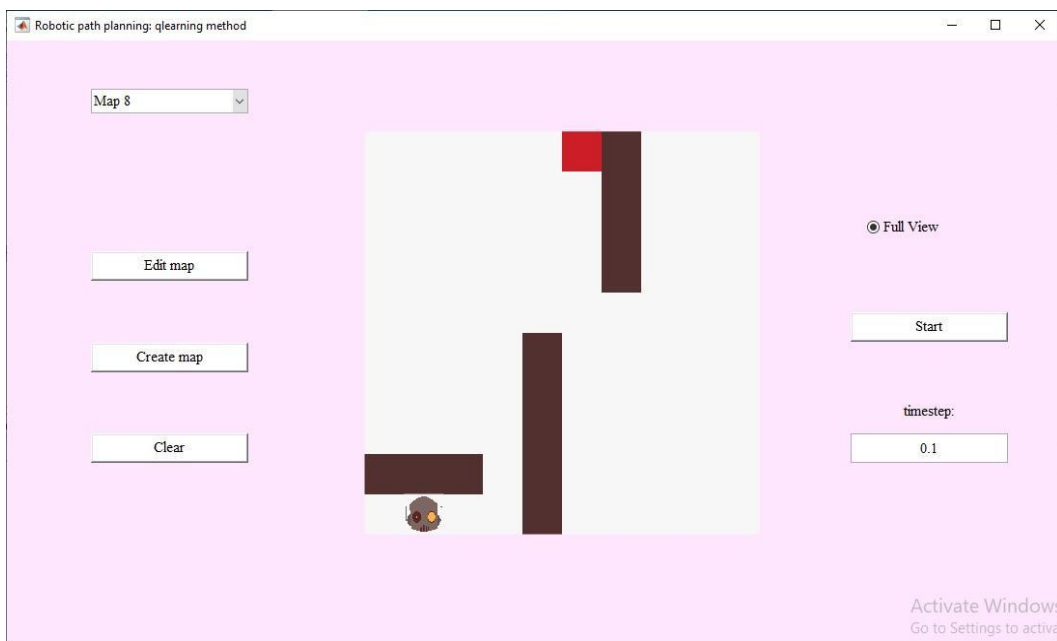
(a) Target 1



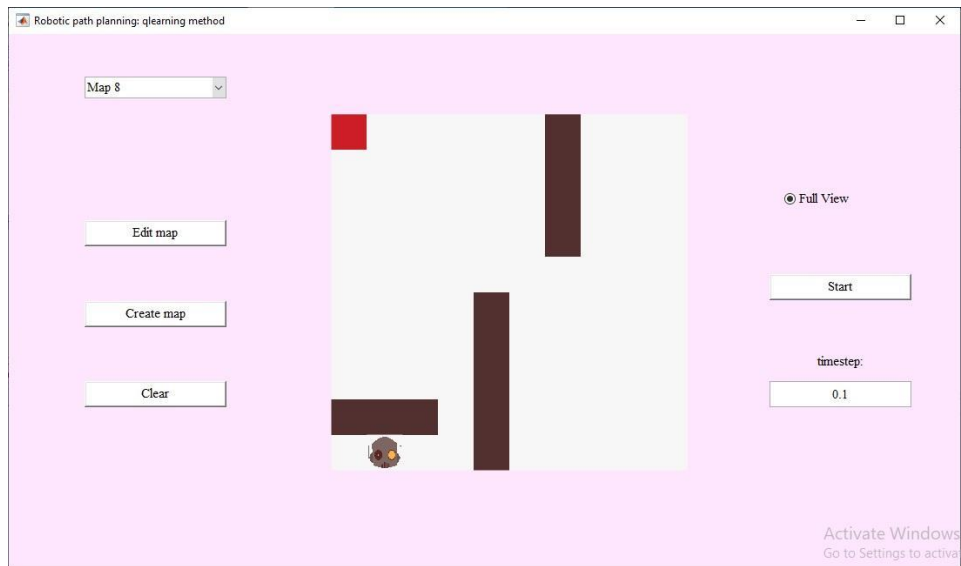
(b) Target 2



(c) Target 3



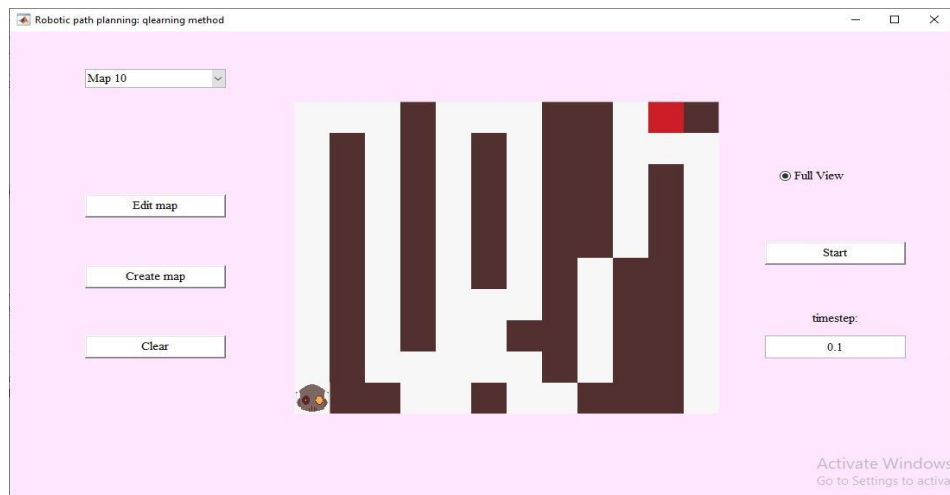
(d) Target 4



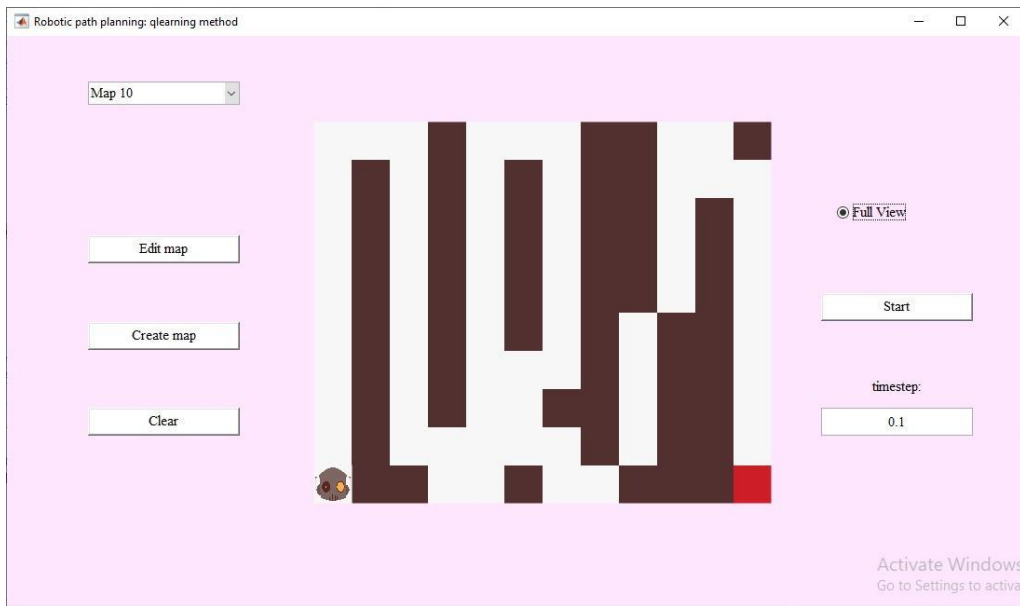
(e) Target 5

Figure 5-2: Working of MAP 8 for different target positions

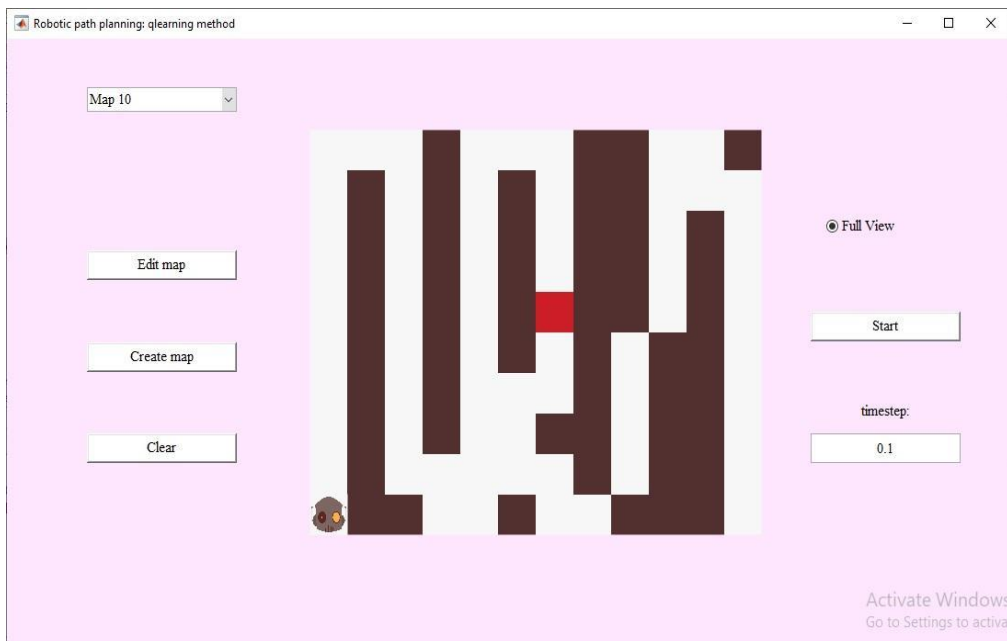
The above fig. 5-2 represents the performance of Map 8. By varying the targets, the performance of the map is also discussed. With the help of a reward table, the robots take 5.66sec to reach target points as training data.



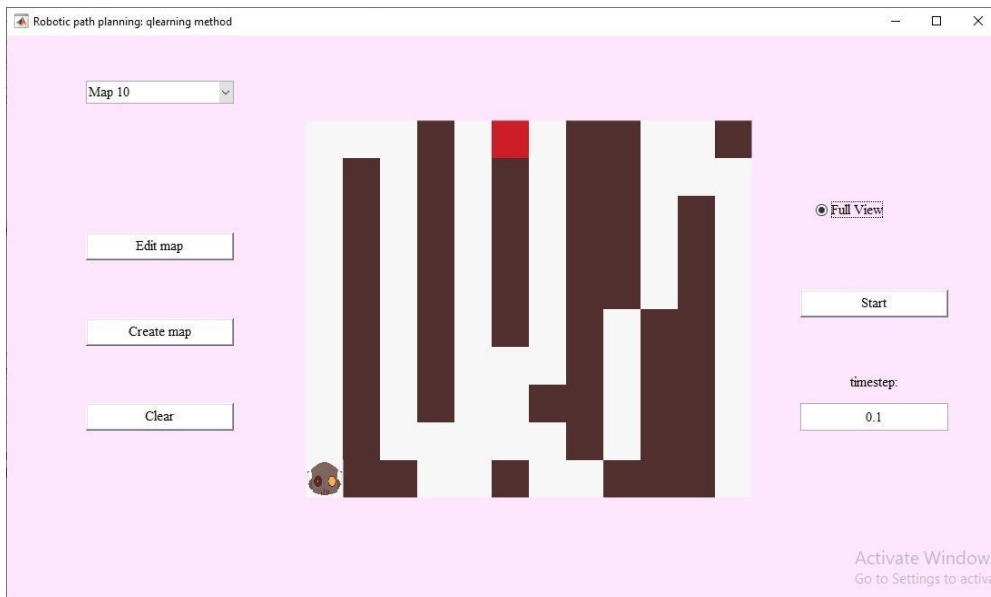
(a) Target 1



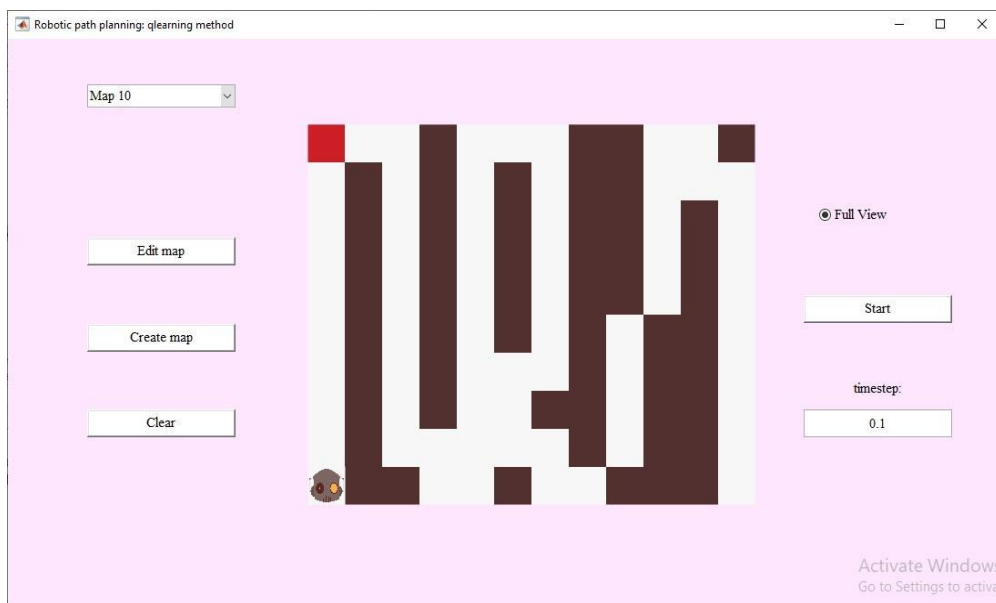
(b) Target 2



(c) Target 3



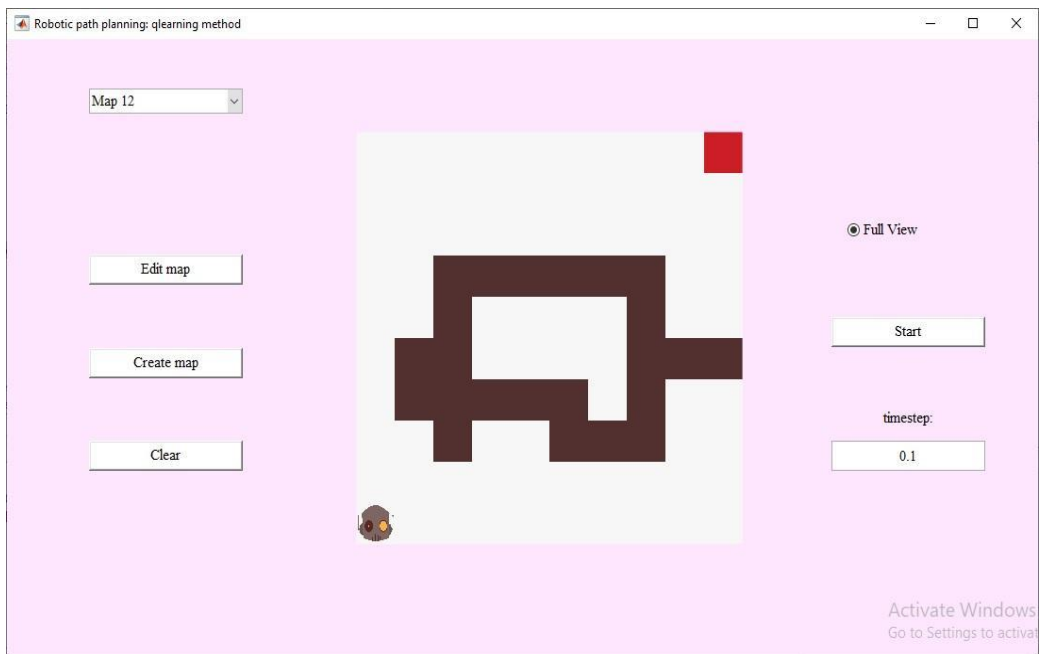
(d) Target 4



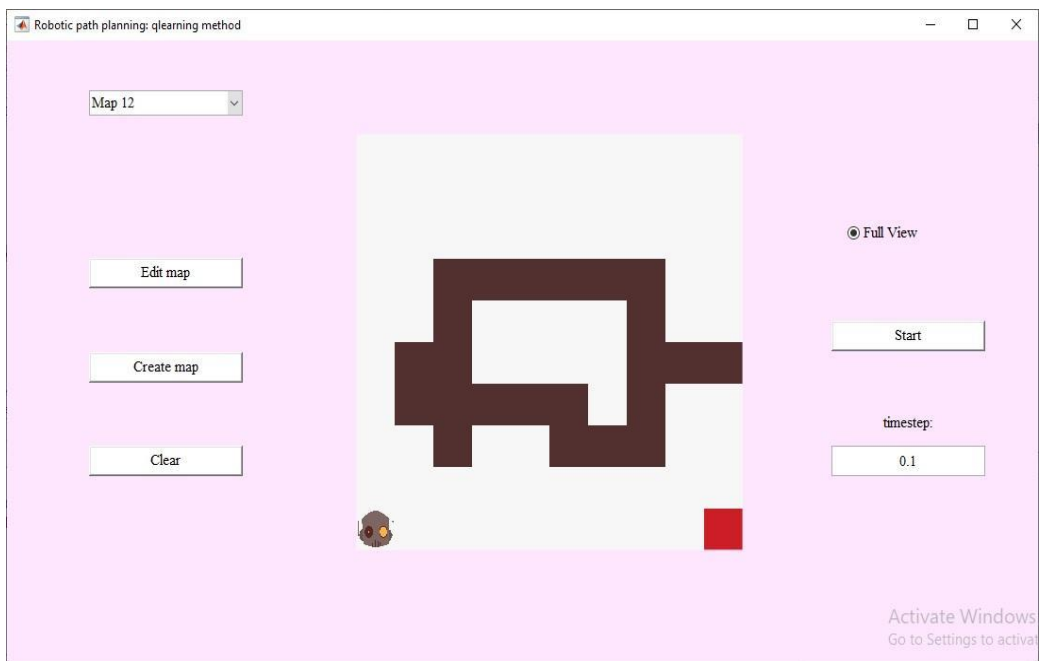
(e) Target 5

Figure 5-3: Working of MAP 10 for different target positions

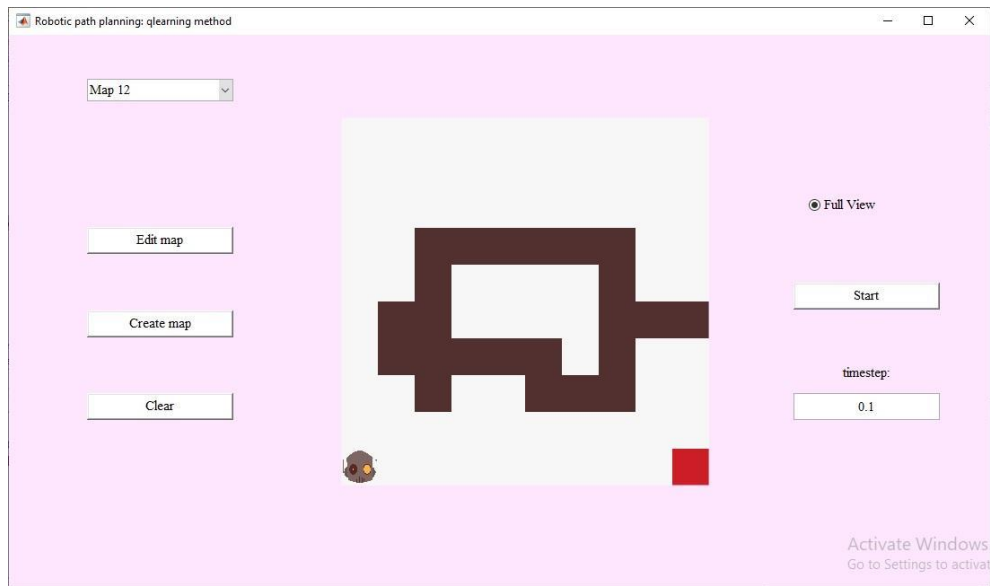
The above fig. 5-3 represents the working of map 10. It is inferred that the map10 takes 7.23sec.



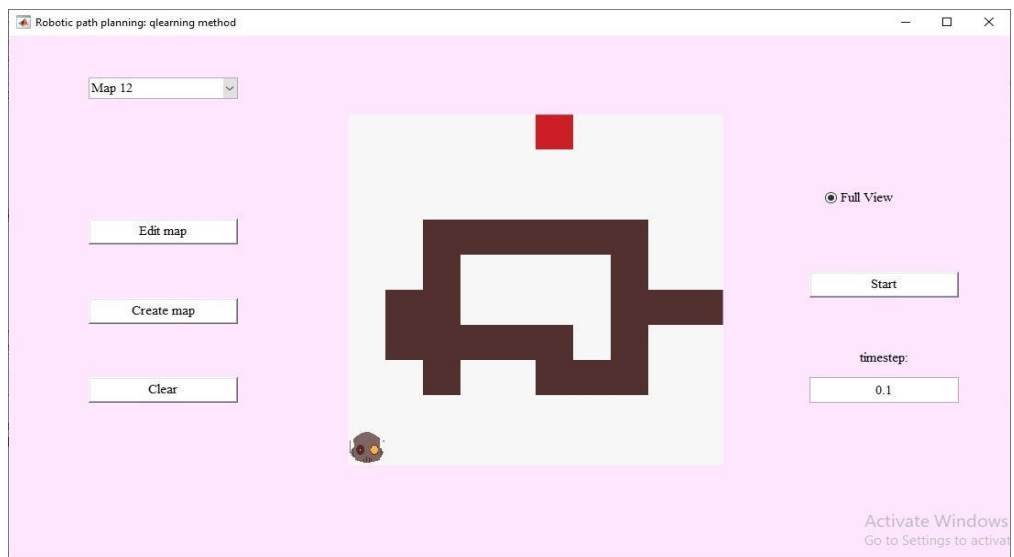
(a) Target 1



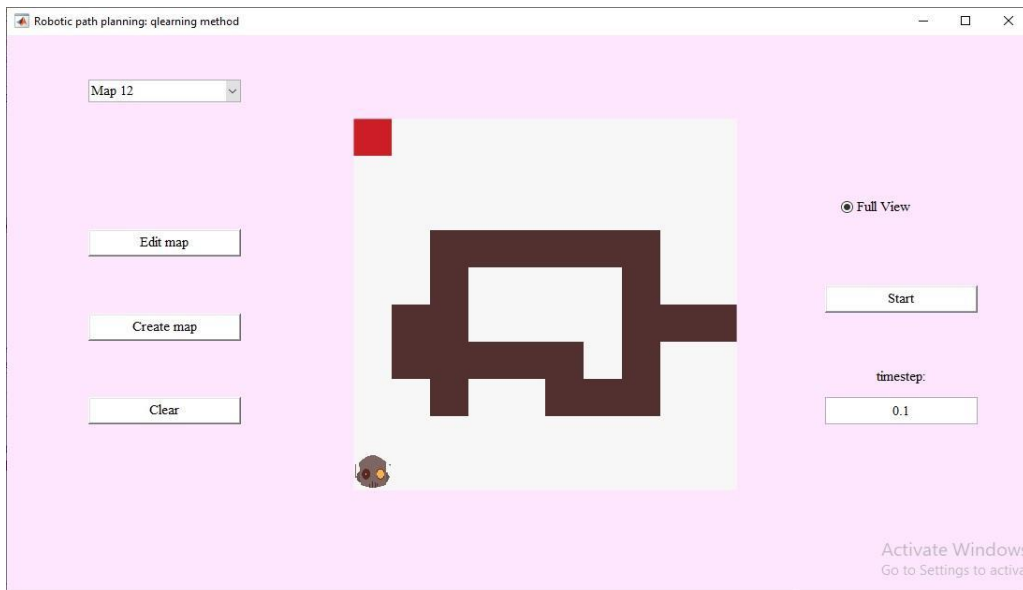
(b) Target 2



(c) Target 3



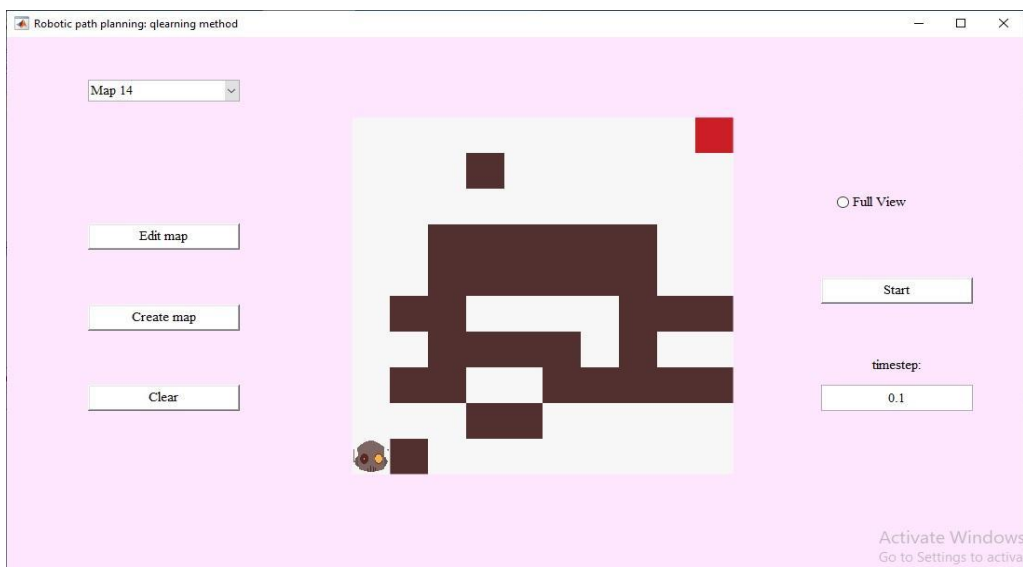
(d) Target 4



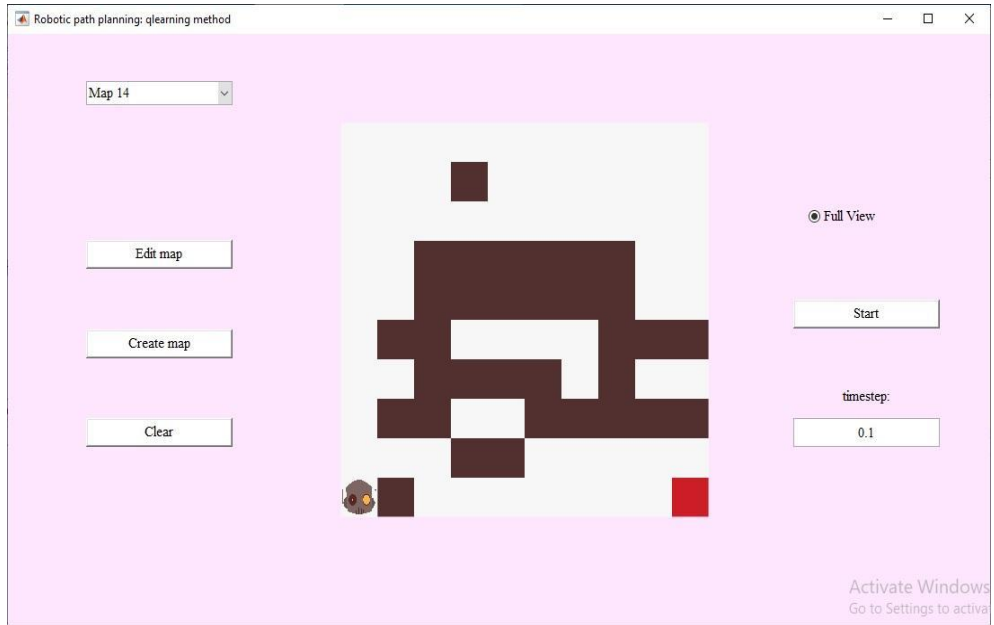
(e) Target 5

Figure 5-4: Working of MAP 12 for different target positions

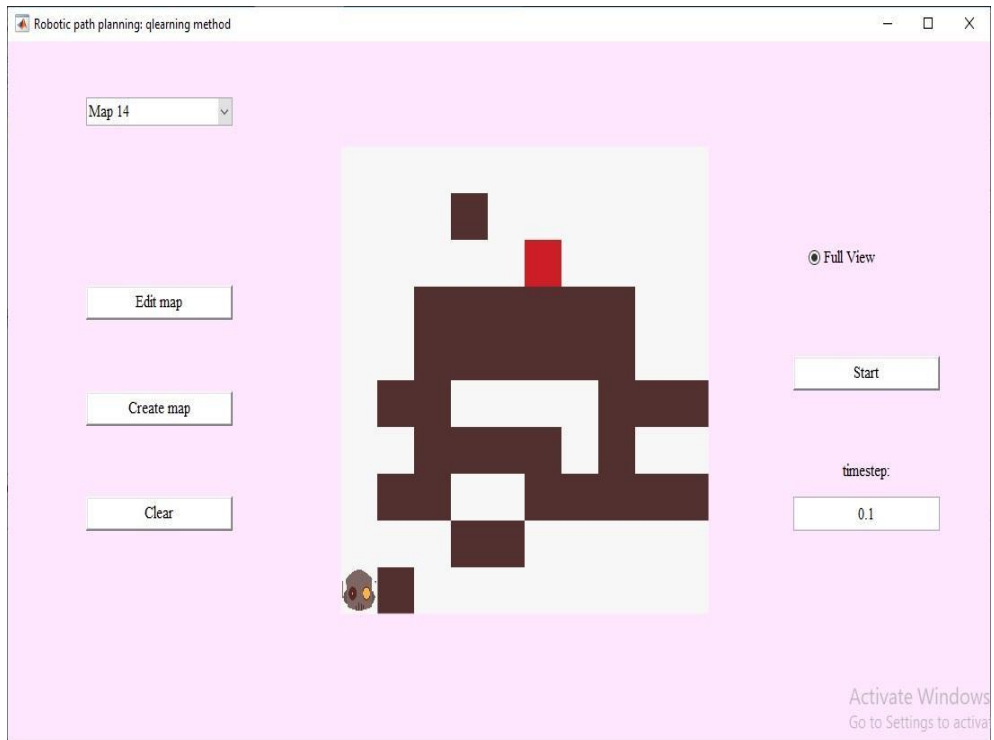
Above figure 5-4 represents the working of Map 12, which consumes 8.45 sec. As time moves on, the decompositions of the cells are modulated in accord with the mapping environment.



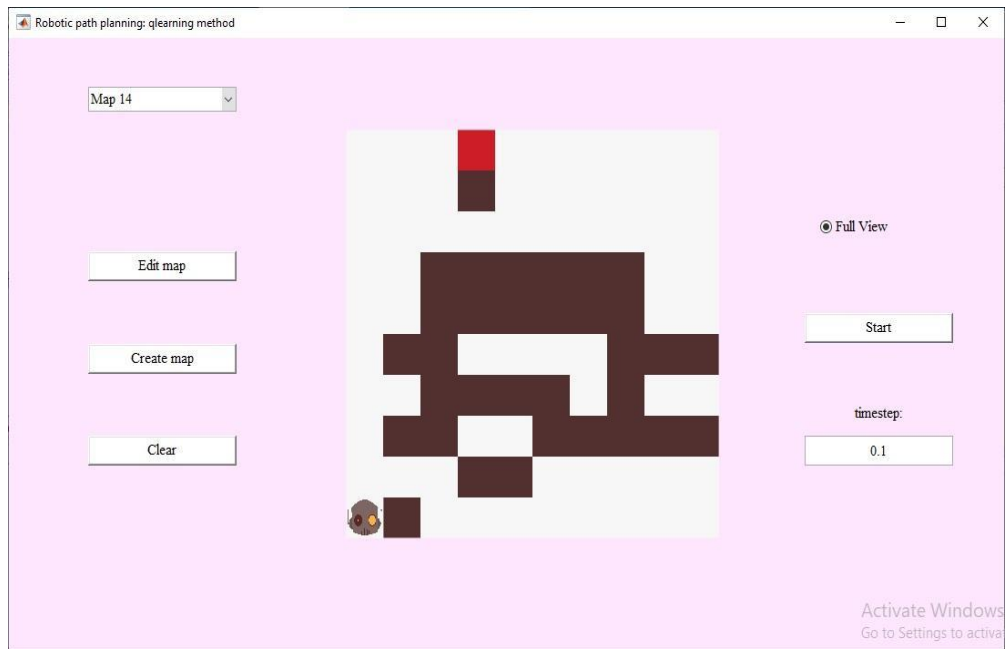
(a) Target 1



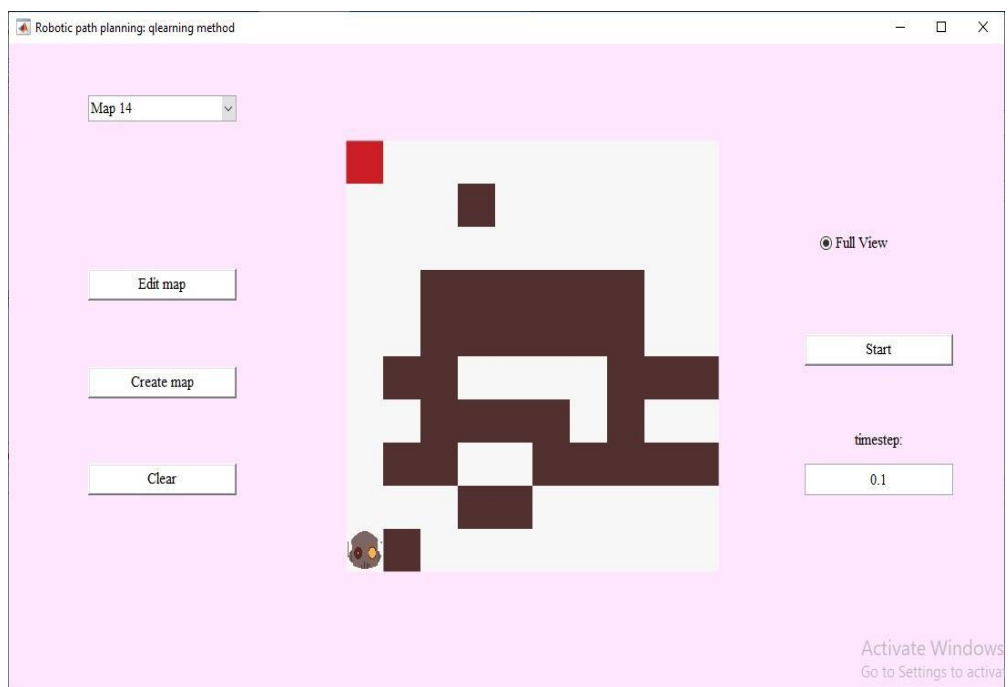
(b) Target 2



(c) Target 3



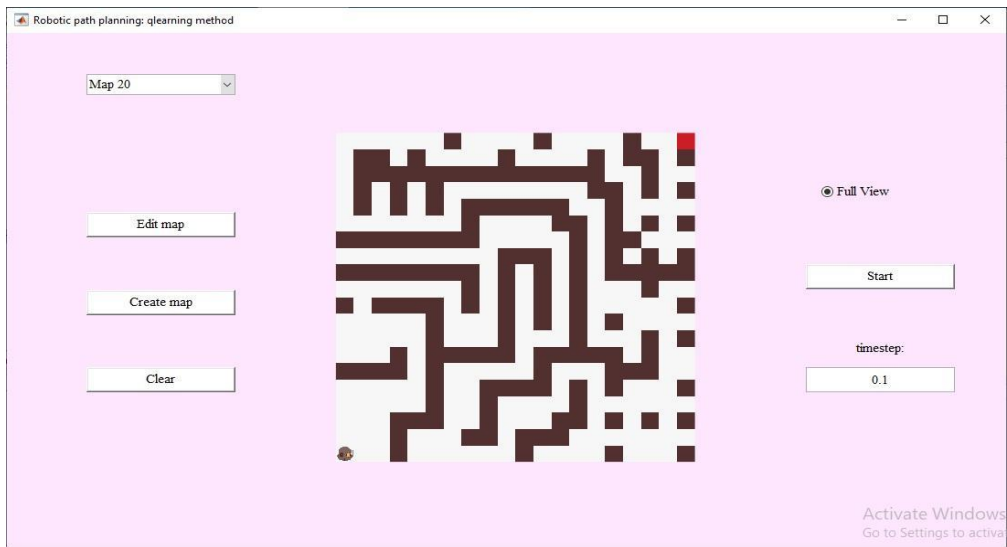
(d) Target 4



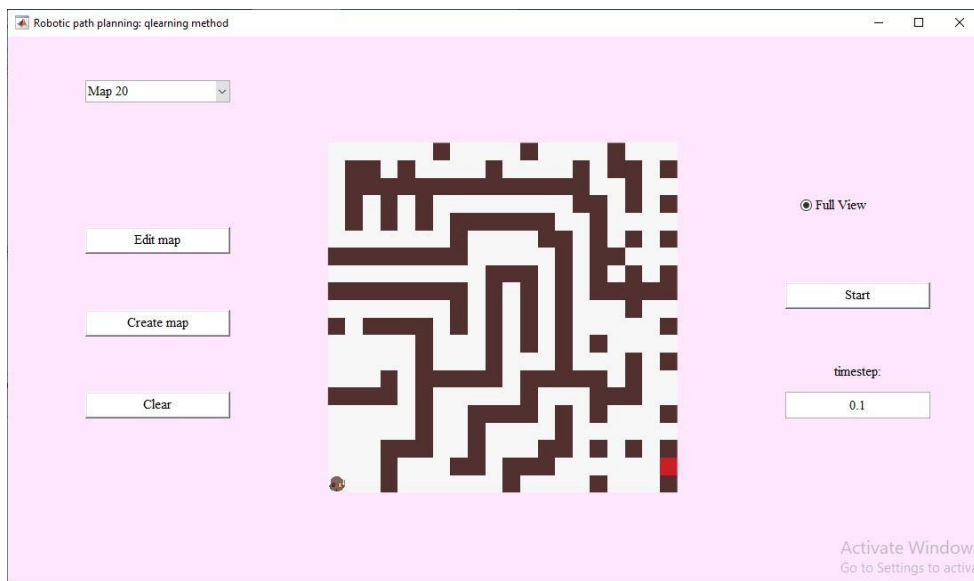
(e) Target 5

Figure 5-5: Working of MAP 14 for different target positions

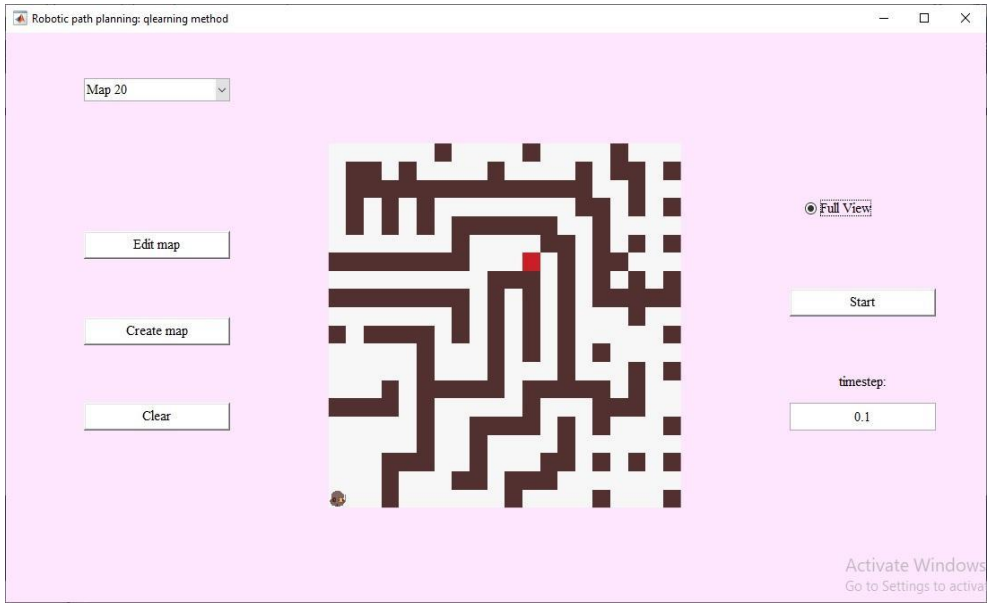
The above fig.5-5 represents the working of map 14, which takes 9.37 secs.



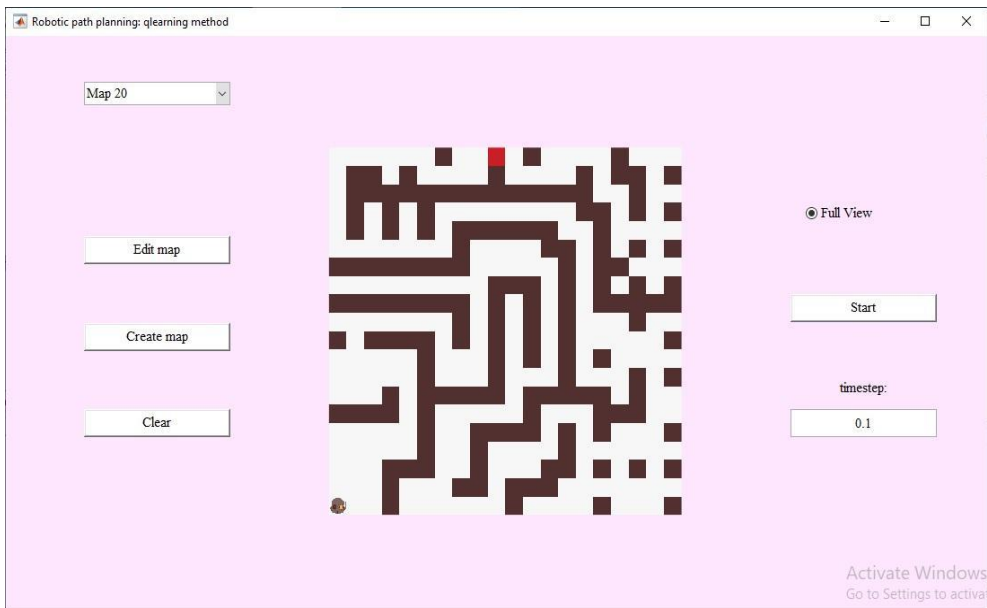
(a) Target 1



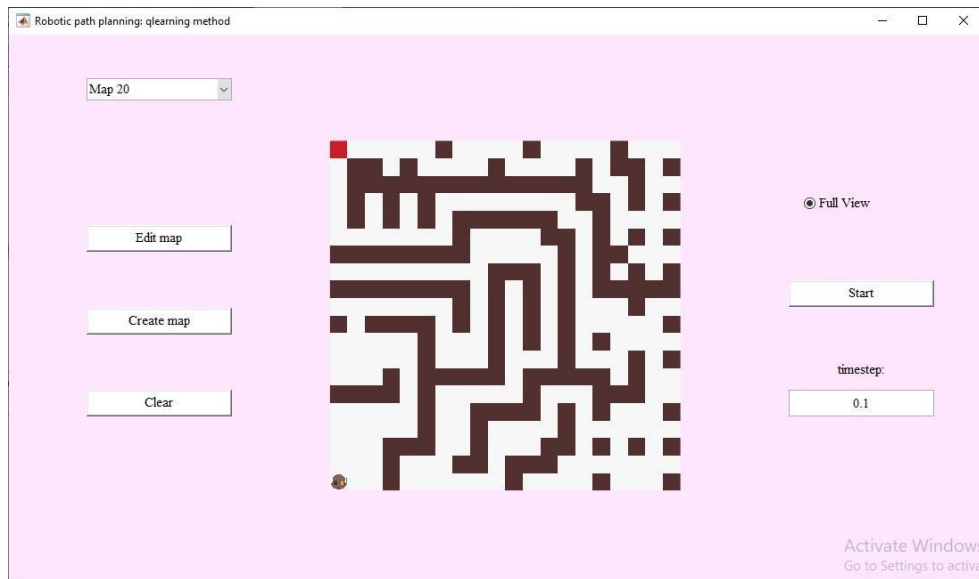
(b) Target 2



(c) Target 3



(d) Target 4



(e) Target 5

Figure 5-6: Working of MAP 20 for different target positions.

The above fig.5-6 represents the working of map 20, which takes 12.13 secs. By following the grid-based decomposition model, the chance of interference (or) collision is eliminated from this system that has improved the efficiency and the robustness of the deployed robots for quick navigation. Once the segmentation of the unknown environment is approximately analyzed and decomposed under a balanced process has paved the way for robots to differentiate the areas between the obstacles and the non-obstacles regions. Thus, the multi-robot can efficiently navigate by its parallel execution of tasks. In the aspects of scalability, finding the fittest shortest paths under reduced time can easily accommodate the data operations of the robots. The use of cuckoo search algorithms has significantly utilized the given resources. Along with that, task allocation and the speed of the robots are also assured. The help of the Q-learning algorithm easily completes the assigned tasks completed by the robots using training data.

Table 5-10 represents the confusion matrix for the above-explored success matrix. Here, the value 1 dictates the equalized actual and observed shortest paths; value 0 represents that the maps do not use the varied target & floating values representing the error values, i.e., deviated paths taken by the robots.

After that, figures 16 to 20 signifies the comparative graphical visualization between the target points and the error obtained during navigation. From the confusion matrix discussed above, different statistical parameters can be calculated to evaluate two algorithms, i.e., Q-learning and SAARTHI. Table 5-11 below illustrates the various parameter values assessed from their respective matrices.

Statistical parameters	Q-learning	SAARTHI
Accuracy	0.907357	0.975541
Error Rate	0.092643	0.024459
Sensitivity/Recall	0.796	0.948
Specificity	0.940071	0.98401
Precision	0.796	0.948
F-measure	0.795854	0.948

Table 5-11: Comparative Analysis of Q-learning and SAARTHI Algorithm

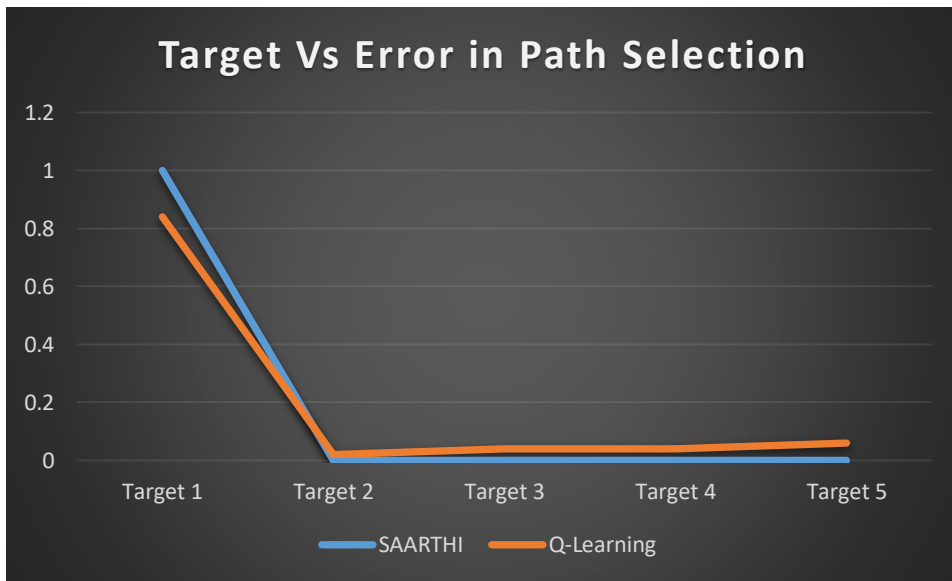


Figure 5-7: Target Points Vs. Error in Path Selection for MAP 8

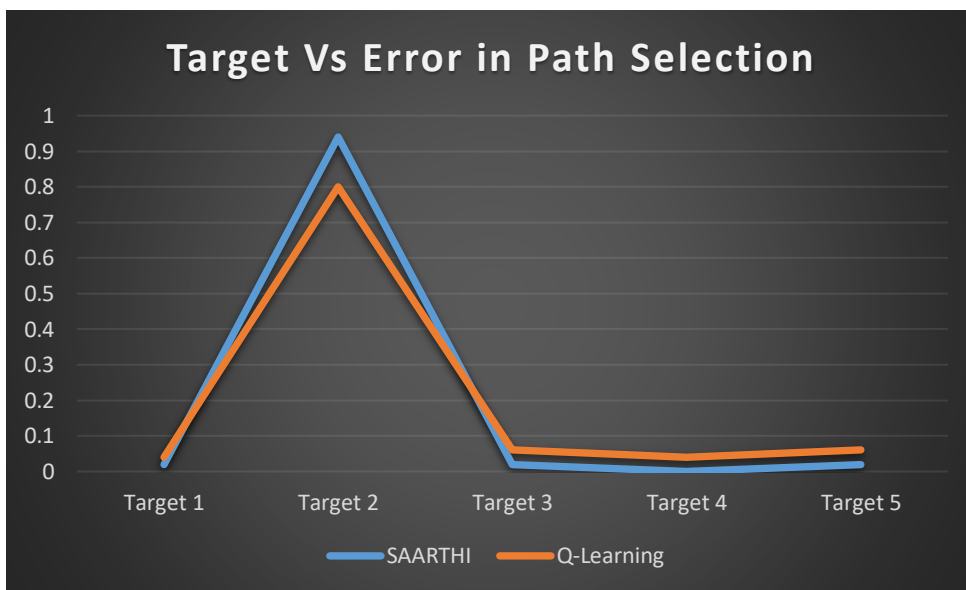


Figure 5-8: Target Points Vs. Error in Path selection for MAP 10

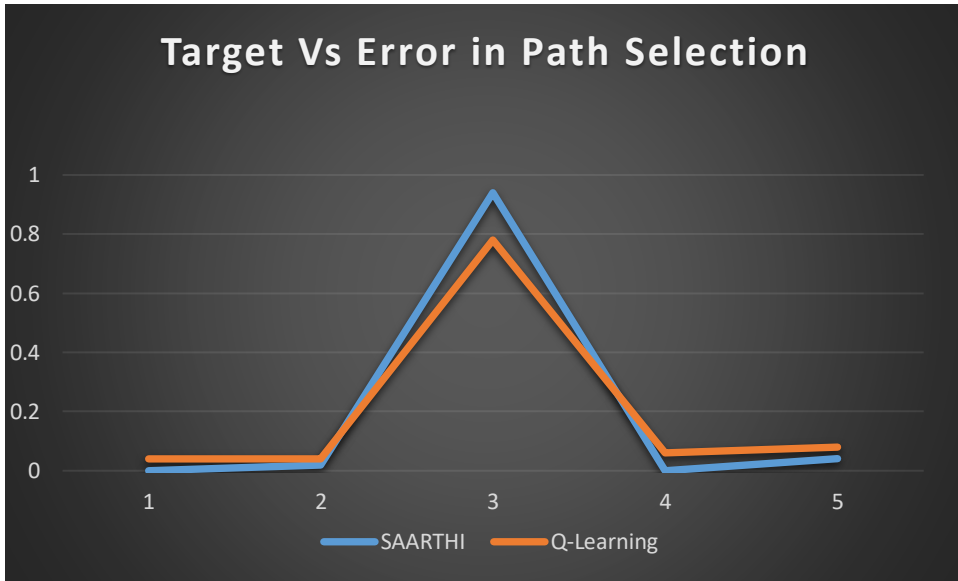


Figure 5-9: Target Points Vs. Error in Path Selection for MAP 12

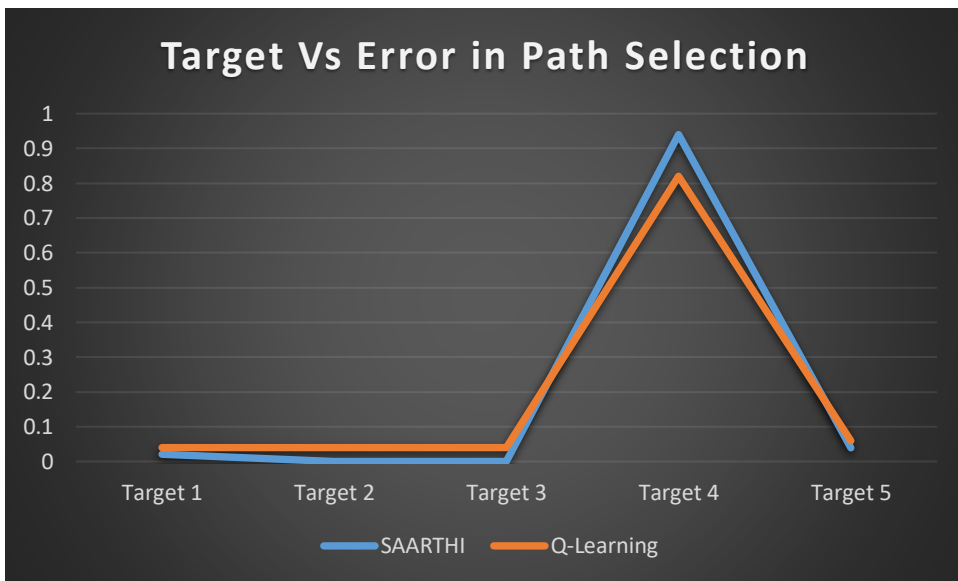


Figure 5-10: Target Points Vs. Error in Path Selection for MAP 14

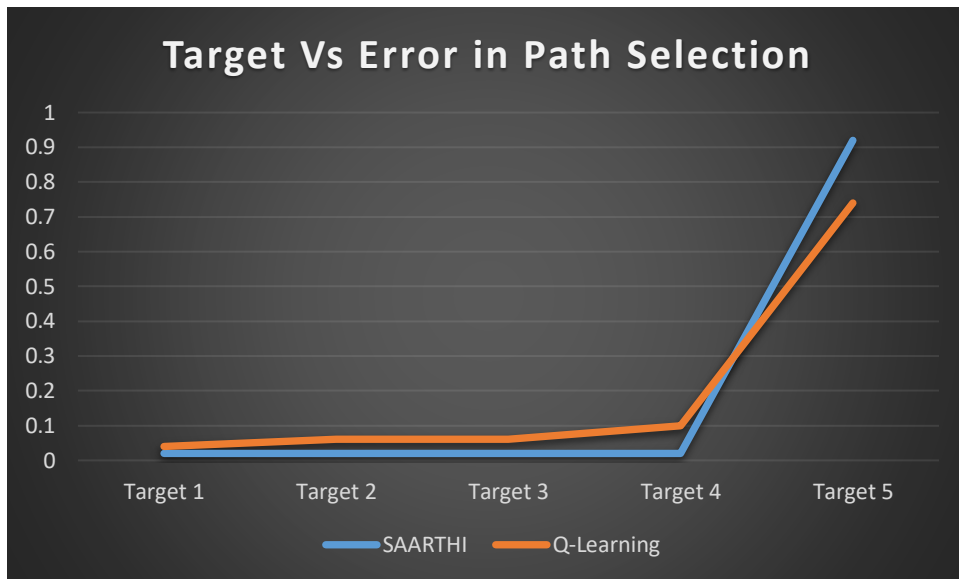


Figure 5-11: Target Point Vs. Error in Path Selection for MAP 20

During the simulation, the robots are deployed in an unknown environment by giving prior information of their origin point. With the help of built-in sensors, the task of the deployed robot is to intelligently select the paths to attain the target position, irrespective of its environment, i.e., known or unknown environment. Initially, the cells are divided into grid forms. Each grid consists of 20 cells. The built-in sensors of the robot sense these cells. During the central placement of the robots, the cells of an entire area are covered.

Also, assumption for the range of the robot's sensors capability of recognizing the obstacles in the three-dimensional orthogonal view of the neighboring cells. During robots in a cell, an individual spanning tree is administered for each exploration process of a cell. The present location of the cell is taken as input to algorithm, and output is the spanning tree of that robot. Since it scans by orthogonal views, all positions of the robots, i.e., walking style, step up, step down, and idle movements, are keenly observed.

Time consumption analysis is the performance measure explored during the implementation. The literature shows that the field of optimizing robot path planning strategies. However, the researchers do not mainly concentrate on the operation time. The reduced time consumption has shed light on providing

a better energy utilization rate, which is relied upon by the actions taken by the robots. Practical action on the unknown robotic environment has significantly reduced the time taken for building the paths. The multiple robots are being deployed in an unknown environment, and therefore, the efficient analysis of discovering the fittest shortest paths is determined by the time consuming, which is executed here. Generally, the deployment of a single robot will not bring any challenges to the environment, whereas deploying a multi-robot, the challenges like collisions, computation overheads, and the heavy time is taken to reach the target position. There are several motivations for the time analysis of multi-robot systems. A comparative table between Q-learning and the proposed algorithm is illustrated further.

Maps	Q-learning (Existing)	SAARTHI (Proposed Algorithm)
8	7.11s	5.66s
10	8.07s	7.23s
12	9.43s	8.45s
14	11.03s	9.37s
20	13.54s	12.13s

Table 5-12: Time Consumption for Q-learning and SAARTHI Algorithm

The increment in the number of obstacles and the distance covered increases the time complexity. The total number of obstacles increases from map 8 to map 20. Thus, it is concluded from the above result, with the increase in grid size and an increase in the number of obstacles, time increases in order to approach the destination point. The results discussed above shows that the projected (SAARTHI) algorithm proves to be better than the existing Q-

learning algorithm. The proposed method shows 97.5% accuracy, whereas Q-learning gives 90.7% accuracy for the same stimulating environment.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The following is a deduction of the thesis. A literature work from the past was accomplished. During the survey, a wide variety of algorithms were studied for autonomous navigation. The algorithms were having both limitations as well as advantages for the different simulating environments. In the research work, a hybrid algorithm for autonomous navigation was proposed. The study concludes that by following the grid-based decomposition model, the chance of interference (or) collision is eliminated from the system that has improved the efficiency and robustness of the deployed robots for quick navigation. Once segmentation of unknown environment is approximately analyzed and decomposed under a balanced process, it has paved the way for robots to differentiate between the obstacles and the non-obstacles regions. Thus, the multi-robot can efficiently navigate by its parallel execution of tasks. In the aspects of scalability, finding the fittest shortest paths under reduced time can easily accommodate the data operations of the robots. The use of cuckoo search algorithms has significantly utilized the given resources. The proposed SAARTHI algorithm provides better simulation results compared to the existing Q-learning algorithm.

6.2 FUTURE SCOPE

The presented model is based on the reinforcement concept for direction finding and route mapping in an unidentified environment. The proposed model can be extended further for automation in the industrial revolution. Wide applications such as Mars rovers, driverless cars, automated machines where the environment is not known can be the comprehensive research of the thesis.

BIBLIOGRAPHY

Adrian Agogino, Kagan Tumer, “Efficient Evaluation functions for multi rover systems,” Springer 2004.

Afzal Humaira, Naz Tabbasum, and Sadiq Ayesha, “A Survey on Automatic Mapping of Ontology to Relational Database Schema.” *Research Journal of Recent Sciences*, Vol. 4, no. 4, pp. 66-70, April 2015.

Aguilar, W. G., Sandoval, D. S., Caballeros, J., Alvarez, L. G., Limaico, A., Rodríguez, G. A., & Quisaguano, F. J., "Graph Based RRT Optimization for Autonomous Mobile Robots.", In *International Conference on Intelligent Science and Big Data Engineering* (pp. 12-21). Springer, Cham, August 2018.

Aguilar, W. G., Sandoval, D. S., Caballeros, J., Alvarez, L. G., Limaico, A., Rodríguez, G. A., & Quisaguano, F. J., "Graph Based RRT Optimization for Autonomous Mobile Robots.", In *International Conference on Intelligent Science and Big Data Engineering* (pp. 12-21). Springer, Cham, August 2018.

A. Howard, D. F. Wolf, and G. S. Sukhatme, “Towards 3D Mapping in Large Urban Environments,” in *Proceedings of the IEEE/RSJ International Conference Intelligent Robots and Systems (IROS)*, Sendai, Japan, 2004.

Alempijevic, A., “High-speed feature extraction in sensor coordinates for laser range finders,” *Proceedings of the 2004 Australasian Conference on Robotics and Automation*, 2004.

Alzbeta Sapetova, Milan Saga, Ivan Kuric & Stefan Vaclav, “Application of optimization algorithms for robot systems designing,” *International Journal of Advanced Robotic Systems* January-February 2018: 1–10.

Alzbeta Sapetova, Milan Saga, Ivan Kuric & Stefan Vaclav, “Application of optimization algorithms for robot systems designing,” *International Journal of Advanced Robotic Systems*, 2018.

Anmin Zhu, Simon X. Yang, "Neuro-fuzzy- Based Approach to Mobile Robot Navigation in Unknown Environment," IEEE transactions on systems, man, cybernetics- PART C: Applications and Reviews, Vol.37, No. 4, Page No. 610-621, July 2007.

Anohana-Numeca, Alla & Graudina, Vita, "Diversity of concept mapping tasks: Degree of difficulty, directedness, and task constraints," in A. J. Cañas, J. D. Novak & J. Vanhear (eds), Concept Maps: Theory, Methodology, Technology: Proceedings of the Fifth International Conference on Concept Mapping, Valletta, Malta, Sept 17-20 2012.

Athanasios Ch. Kapoutsis, Savvas A., et al., "Employing Cellular Automata for shaping Accurate Morphology Maps using Scattered Data from Robotics Mission," Springer international publishing, 2015.

Bashan Zuo, Jiabin Chen et al., "A Reinforcement Learning Based Robotic Navigation System," IEEE international conference on systems, man, and cybernetics, 2014.

Benjamin Lussier, Raja Chatila, et al., "On Fault Tolerance and Robustness in Autonomous Systems," Proceedings of the 3rd IARP-IEEE/RAS EURON joint workshop on technical challenges for dependable robots in a human environment, 2004.

Benjamin Lussier, Alexander Lampe, et al., "Fault tolerance in Autonomous Systems: How and How much?", proceedings of the 4th IARP-IEEE/RAS EURON joint workshop technical challenges for dependable robots in a human environment, 2004.

Bormann, R., Jordan, F., Li, W., Hampp, J., & Hagele, M. Room segmentation: Survey, implementation, and analysis. 2016 IEEE International Conference on Robotics and Automation (ICRA). 2016.

Borenstein, J. and Koren, Y, "Real-time obstacle avoidance for fast mobile robots." IEEE Transactions on Systems, Man and Cybernetics, Vol 19, no. 5, 1179–1187, 1989.

Blanco, J.-L., Gonzalez, J., and Fernandez-Madrigal, "Extending obstacle Avoidance methods through multiple parameter-space transformations" *Autonomous Robots*, Vol 24, no. 1, Pages 29–48, 2007.

B. Steder, R. B. Rusu, and K. Konolige, "Point Feature Extraction on 3D Range Scans Taking into Account Object Boundaries," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

Burchfiel, B., & Konidaris, G., "Hybrid Bayesian Eigenobjects: Combining Linear Subspace and Deep Network Methods for 3D Robot Vision.", In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 6843-6850), October 2018.

Charles Richter, John Ware, et al., "High-Speed Autonomous Navigation of unknown environments using learned probabilities of collision," *IEEE conference on robotics and automation*, 2014.

Chen, Y. W., & Chiu, W. Y., "Optimal robot path planning system by using a neural network-based approach.", In *2015 international automatic control conference (CACS)* (pp. 85-90), 2015.

Ching-Chih Tsai, Hsu-Chih Huang, "Parallel Elite Genetic Algorithm and its application to global path planning for autonomous robot navigation," *IEEE transaction on industrial electronics*, Vol 58, Issue 10, Pages 4813-4821, October 2011.

Ch. Raja, et al., "Theory of Markov Normal Algorithm," *International Conference on Computing Sciences*, 2012.

Clark F. Olson, Larry H. Matthies, et al. "Rover Navigation using stereo- ego-motion," Elsevier, *robotics and autonomous system*, 2003.

CONG Yang, PENG Jun-Jian, "V-Disparity based UGV obstacle detection in rough outdoor terrain," *ACTA Automatica Sinica Elsevier*, Vol 36, Issue 5, Pages 667-673, 2010.

Croom JM, Rucker DC, Romano JM, Webster RJ. Visual sensing of continuum robot shape using self-organizing maps, *Robotics and Automation (ICRA)*, 2010 *IEEE International Conference on*, IEEE, 2010.

Daniel Sales, Patrick Shinzato, et al. " Vision-based autonomous navigation system using ANN and FSM control," *Latin American Robotics Symposium*

and intelligent robotics meeting, 2010.

Das, P. K., Sahoo, B. M., Behera, H. S., & Vashisht, S., "An improved particle swarm optimization for multi-robot path planning.", In 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH), (pp. 97-106), February 2016.

Das, P. K., Behera, H. S., & Panigrahi, B. K., "A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning.", *Swarm and Evolutionary Computation*, Vol 28, Pages 14-28, 2016.

Dayal R. Parhi, Jayanta Kumar Pothal, Mukesh Kumar Singh, "Navigation of multiple robots using swarm intelligence," *IEEE World Congress on Nature & Biologically Inspired Computing (NaBIC)*, 2009.

Dugan Um, Dongseok Ryu, "SPAM for a manipulator by best next move in unknown environments." *Hindawi Publishing Corporation*, 2013.

E.G., Rajan, "Symbolic Computing: Signal and Image Processing," *BS Publications*, 2003.

Elmokadem, T., "A 3D Reactive Collision Free Navigation Strategy for Nonholonomic Mobile Robots.", In 2018 37th Chinese Control Conference (CCC) (pp. 4661-4666). July 2018.

Floreano, D., & Mondada, F. "Automatic creation of an autonomous agent: Genetic evolution of a neural network-driven robot. In from Animals to Animats 3." *Proceedings of the Third International Conference on Simulation of Adaptive Behavior* (pp. 421-430). July 1994.

F. Maurelli, D. Droschel, T. Wisspeintner, S. May, and H. Surmann, "A 3D Laser Scanner System for Autonomous Vehicle Navigation," in *Proceedings of the International Conference on Advanced Robotics (ICAR)*, 2009.

Ghosh, S., Panigrahi, P. K., & Parhi, D. R. "Analysis of FPA and BA meta-heuristic controllers for optimal path planning of mobile robot in a cluttered environment.", *IET Science, Measurement & Technology*, Vol 11(7), Pages 817-828, 2017.

Giuseppe Oriolo, "Real-Time Map building and navigation for robots in an unknown environment." *IEEE transaction on systems, men, and cybernetics*, Vol 28, Issue 3, Pages 316-333, June 1998.

G. Ramesh Chandra et al., "3D Edge Detection of Medical Images Using Mathematical Morphological and Cellular Logic Array Processing Techniques", IEEE International Conference on Systems, Man and Cybernetics October 14-17, 2012.

Han, J. "An efficient approach to 3D path planning." Information Sciences, Vol 478, Pages 318-330, 2019.

Hosseininejad, S., & Dadkhah, C." Mobile robot path planning in a dynamic environment based on a cuckoo optimization algorithm," International Journal of Advanced Robotic Systems, Vol 16, Issue 2, 2019.

H. Schafer, A. Hach, et al. "3D obstacle detection and avoidance in vegetated off-road terrain", IEEE international conference on robotics and automation, June 2008.

Huang, H., Savkin, A. V., Ding, M., & Huang, C., "Mobile robots in wireless sensor networks: A survey on tasks." Computer Networks, Vol 148, Pages 1-19, 2019.

Hyansu Bae, Gidong Kim, Jonguk Kim, Dianwei Qian & Sukgyu Lee, "Multi-Robot Path Planning Method Using Reinforcement Learning," Applied Sciences, vol. 9, Issue 15, 2019.

Iadaloharivola Randria, Mohamed Moncef Ben K, "A Comparative study of six basic approaches for path planning towards an autonomous navigation," 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON), Nov 2007.

Jefferson R. Souza, Gustavo Pessin, et al., "Vision-Based Autonomous Navigation Using Neural Networks and Templates in Urban Environment," Ist Brazilian Conference on critical embedded systems, 2011.

Jens Steffen Gutmann, Masaki Fukuchi, et al. "A floor and obstacle height map for 3d navigation of a humanoid robot", IEEE conference on robotics and automation, 2005.

Konstantinos Charalampous, Ioannis Kostavelis, et al., "Autonomous Robot Path Planning techniques using Cellular Automata," Springer international publishing, Pages 175-196, 2015.

Krispin A. Davies, Alejandro Ramirez-Serrano, Graeme N. Wilson, Mahmoud Mustafa, "Rapid Control Selection through Hill-Climbing Methods," Intelligent Robotics and applications, vol.7507,2012.

Lapierre, L., Zapata, R., and Lepinay, P, "Combined Path-following and Obstacle Avoidance Control of a Wheeled Robot," the International Journal of Robotics Research, Vol 26, no. 4, Page No. 361-375, 2007.

Lazaros Nalpantidis, "On the use of Cellular Automata in Vision-based robot exploration," Springer -international publishing, Volume 13, Pages 247-266, 2015.

Levente Tamas* and Lucian Cosmin Goron, "3D map building with mobile robots", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012.

Mandava, R. K., Bondada, S., & Vundavilli, P. R., "An optimized path planning for the mobile robot using potential field method and PSO algorithm.", In Soft Computing for Problem Solving (pp. 139-150). Springer, Singapore, 2019.

Mane, S. B., & Vhanale, S., "Genetic Algorithm Approach for Obstacle Avoidance and Path Optimization of Mobile Robot." In Computing, Communication and Signal Processing, (pp. 649-659). Springer, Singapore, 2019

Marko Mitic, Zoran Miljkovic & Bojan Babic, "Empirical Control System Development for Intelligent Mobile Robot Based on the Elements of the Reinforcement Machine Learning and Axiomatic Design Theory," Faculty of Mechanical Engineering, Belgrade, Vol 39, Issue 1, Pages 1-8, 2011.

Maturana, D., & Scherer, S., "Voxnet: A 3d convolutional neural network for real-time object recognition." In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), (pp. 922-928)." September 2015.

Melo, L. C., Maximo, M. R. O. A., & da Cunha, A. M. "Learning Humanoid Robot Motions Through Deep Neural Networks." arXiv preprint arXiv:1901.00270, 2019.

Meng Wang, James N.K. Liu, "Fuzzy logic based real-time robot navigation in an unknown environment with dead ends," Robotics and Automation systems, Elsevier, 2008.

Md. Arafat Hossain & Israt Ferdous, "Autonomous robot path planning in a dynamic environment using a new optimization technique inspired by bacterial foraging technique," Robotics and Autonomous Systems, Vol. 64, Pages 137-141, 2014.

Mohanarajah, G., Usenko, V., Singh, M., D'Andrea, R., & Waibel, M., "Cloud-based collaborative 3D mapping in real-time with low-cost robots." IEEE Transactions on Automation Science and Engineering, Vol 12, Issue 2, Pages 423-431, 2015.

Mo, H., & Xu, L., "Research of biogeography particle swarm optimization for robot path planning," Neurocomputing, 148, 91-99, 2015.

Naz, A., Piranda, B., Tucci, T., Goldstein, S. C., & Bourgeois, J., "Network Characterization of Lattice-Based Modular Robots with Neighbor-to-Neighbor Communications.", In Distributed Autonomous Robotic Systems (pp. 415-429). Springer, Cham, 2018.

Niharika Singh, Manish Prateek, Piyush Chauhan, "An Efficient Path Planning Algorithm for Networked Robots using Modified Optimization Algorithm," IJITEE, Vol 8, Issue 12, 2019.

Niharika Singh, Manish Prateek, Piyush Chauhan, Tanupriya Choudhury, "An Optimized Self-Learning algorithm for autonomous navigation in an unknown environment." Journal of green engineering, Vol 10, Issue 7, 2020.

Ouarda Hachour, "Towards an approach of Fuzzy control motion for mobile robots in unknown environments," WSEAS Transaction on Systems, ACM, Pages 114-128 Aug 2009.

Panda, R. K., & Choudhury, B. B., "An effective path planning of mobile robot using genetic algorithm.", In 2015 IEEE International Conference on Computational Intelligence & Communication Technology, (pp. 287-291). February 2015.

Parhi, D. R., "Advancement in navigational path planning of robots using various artificial and computing techniques." Int Rob Auto J, Vol 4(2), Pages 133-136, 2018.

Pomerleau, D. A., "Neural network perception for mobile robot guidance." Springer Science & Business Media, Vol 239, 2012.

Qingyang Chen, Zhenping Sun, et al., "Local Path Planning for autonomous land vehicle based on navigation function," Fourth international conference on intelligent computation technology and automation, 2011.

Reza Vafashoar & Mohammad Reza Meybodi, "Reinforcement learning in learning automata and cellular learning automata via multiple reinforcement signals," Knowledge-Based Systems, vol. 169, Pages 1-27, 2019.

Richard S. Sutton, Andrew G. Barto, "Reinforcement Learning: An introduction, Adaptive Computation, and Machine Learning." Francis Bach editor, MIT Press, 2018.

Sanchez-Lopez, J. L., Wang, M., Olivares-Mendez, M. A., Molina, M., & Voos, H. "A Real-Time 3D Path Planning Solution for Collision-Free Navigation of Multirotor Aerial Robots in Dynamic Environments", Journal of Intelligent & Robotic Systems, Vol 93, Pages 33-53, 2019.

Saicharan, B., Tiwari, R., & Roberts, N., "Multi-Objective optimization-based Path Planning in robotics using nature-inspired algorithms: A survey.", In

2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES) (pp. 1-6), July 2016.

Shabbir, J., & Anwer, T., "Artificial Intelligence and its role in the near future." arXiv preprint arXiv:1804.01396.

Shanthi S. et al. "Cellular Automata and their realizations," International Conference on Computing Sciences, 2012.

Simmons, R, "The curvature-velocity method for local obstacle avoidance" Proceedings of IEEE International Conference on Robotics and Automation. April 1996.

Simon Lacroix, Anthony Mallet,etal. "Autonomous Rover navigation on unknown terrains: Functions and Integration." International journal of robotics research, 2002.

S.Russel and P. Norvig, "Artificial intelligence, A Modern approach (2nd edition)", prentice hall,2002.

Sudhakara, P., & Ganapathy, V., "Trajectory Planning of a mobile robot using enhanced A-star algorithm." Indian Journal of Science and Technology, Vol 9, Issue 41, Pages 1-10, 2016.

Sun, L., Liu, X., & Leng, M. "An Effective Algorithm of Shortest Path Planning in a Static Environment.", In International Conference on Programming Languages for Manufacturing (pp. 257-262). Springer, Boston, MA., June 2006.

Song, B., Wang, Z., Zou, L., Xu, L., & Alsaadi, F. E., "A new approach to smooth global path planning of mobile robots with kinematic constraints.", International Journal of Machine Learning and Cybernetics, Vol 10, Issue 1, 107-119, 2019.

Tharwat, A., Elhoseny, M., Hassanien, A. E., Gabel, T., & Kumar, A., "Intelligent Bézier curve-based path planning model using Chaotic Particle Swarm Optimization algorithm." *Cluster Computing*, Vol 22, Issue 4, 2018.

Thai Duong, Michael Yip, Nikolay Atanasov, "Autonomous Navigation in Unknown Environment with Sparse Bayesian Kernel-based Occupancy Mapping," *IEEE Conference on Robotics and Automation*, August 2020.

Thrun, S., "Robotic Mapping: A Survey," Lakemeyer, G. and Nebel, B., editors, "Exploring Artificial Intelligence in the New Millenium." Morgan Kaufmann, 2002.

Venkataraman, A., Griffin, B., & Corso, J. J., "Kinematically-Informed Interactive Perception: Robot-Generated 3D Models for Classification." *arXiv preprint arXiv:1901.05580*, 2019.

Viswanathan, S., Ravichandran, K. S., Tapas, A. M., & Shekhar, S., "An Intelligent Gain based Ant Colony Optimisation Method for Path Planning of Unmanned Ground Vehicles." *Defense Science Journal*, Vol 69, Issue 2, Page No. 167-172, 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Daan Wierstra, Alex Graves, Ioannis Antonoglou & Martin Riedmille, "Playing Atari with Deep Reinforcement Learning," *NIPS Deep Learning Workshop* 2013.

Wang, L., Kan, J., Guo, J., & Wang, C., "3D Path Planning for the Ground Robot with Improved Ant Colony Optimization.", *Sensors*, Vol 19, no. 4, 815, 2019.

Woo, M. H., Lee, S. H., & Cha, H. M., "A study on the optimal route design considering the time of mobile robot using recurrent neural network and reinforcement learning.", *Journal of Mechanical Science and Technology*, Vol 32(10), Pages 4933-4939, 2018.

Yaobang Gong, Mohamed Abdel-Aty, Qing Cai, Md Sharikur Rahman. "Decentralized network-level adaptive signal control by multi-agent deep

reinforcement learning," *Transportation Research Interdisciplinary Perspectives*, Volume 1, 2019.

Yin, H., Wang, Y., Tang, L., Ding, X., & Xiong, R., "LocNet: Global localization in 3D point clouds for mobile robots.", In *Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV)*, Changshu, China (pp. 26-30), June 2018.

Yuan, J., Wang, H., Lin, C., Liu, D., & Yu, D., "A Novel GRU-RNN Network Model for Dynamic Path Planning of Mobile Robot." *IEEE Access*, 7, 15140-15151, 2019.

Zhang, Y., Chen, C., & Liu, Q., "Mobile robot path planning using ant colony algorithm. *International Journal of Control and Automation*, Vol 9(9), Pages 19-28, 2016.

Zhao X, Su Z, Dou L. A path planning method with minimum energy consumption for a multi-joint mobile robot. In: *Control Conference (CCC)*, 2014 33rd Chinese, IEEE, 2014.

Zeng, M. R., Xi, L., & Xiao, A. M., "The free step length ant colony algorithm in mobile robot path planning." *Advanced Robotics*, Vol 30, Issue 23, Pages 1509-1514, 2016.

Zeyad Abd Algfoor, Mohd. Shahrizal Sunar, et al., "A comprehensive study on pathfinding techniques for robotics and video games," *Media and Games Innovative Centre of Excellence (MaGIC-X) UTM-IRDA, Digital Media Centre, Malaysia*, 2015.

<https://www.elprocus.com/different-types-of-autonomous-robots-and-real-time-applications/>

APPENDIX A

LIST OF PUBLICATIONS











Niharika Singh, Manish Prateek, Piyush Chauhan, “An Efficient Path Planning Algorithm for Networked Robots using Modified Optimization Algorithm," IJITEE, Vol 8, Issue 12, 2019.

Niharika Singh, Manish Prateek, Piyush Chauhan, Tanupriya Choudhury, “An Optimized Self-Learning algorithm for autonomous navigation in an unknown environment.” Journal of green engineering, Vol 10, Issue 7, 2020.

Document Information

Analyzed document Niharika_500041827_25-01-2021.pdf (D93474542)
 Submitted 1/25/2021 10:03:00 AM
 Submitted by Kingshuk Srivastava
 Submitter email ksrivastava@ddn.upes.ac.in
 Similarity 3%
 Analysis address ksrivastava.upes@analysis.urkund.com

Sources included in the report

SA	1324479138-TS.pdf Document 1324479138-TS.pdf (D76781632)		4
W	URL: https://www.elprocus.com/different-types-of-autonomous-robots-and-real-time-applic ... Fetched: 1/25/2021 12:06:00 PM		6
SA	regno. 2009740127.pdf Document regno. 2009740127.pdf (D29432962)		3
W	URL: https://arxiv.org/pdf/1902.05177 Fetched: 1/25/2021 12:05:00 PM		4
W	URL: https://archive.org/stream/ost-engineering-cutting-edge-robotics/Cutting%20Edge%20 ... Fetched: 12/10/2019 11:16:53 AM		7
W	URL: https://waman.github.io/papers/waman_icra_14.pdf Fetched: 1/25/2021 12:05:00 PM		1
W	URL: https://people.eecs.berkeley.edu/~sseshia/pubs/b2hd-saha-iros14.html Fetched: 1/25/2021 12:05:00 PM		1
W	URL: https://www.hindawi.com/journals/jr/2011/389158/ Fetched: 1/25/2021 12:05:00 PM		2
W	URL: https://link.springer.com/10.1007%252F978-0-387-30440-3_344 Fetched: 1/25/2021 12:05:00 PM		1
W	URL: https://www.researchgate.net/publication/278652096_Modified_A_Algorithm_for_Mobile ... Fetched: 5/30/2020 3:09:02 AM		1