FUZZY LOGIC MODEL FOR OBSTACLES AVOIDANCE MOBILE ROBOT IN STATIC UNKNOWN ENVIRONMENT

AGGREY SHITSUKANE SHISIALI

MASTER OF SCIENCE

(Computer Systems)

JOMO KENYATTA UNIVERSITY OF

AGRICULTURE & TECHNOLOGY

2022

Fuzzy Logic Model for Obstacles Avoidance Mobile Robot in Static Unknown Environment

Aggrey Shisiali Shitsukane

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Systems of the Jomo Kenyatta University of Agriculture and Technology

2022

DECLARATION

This research is my original work and has not been presented for a degree in any other university

Signature……..Date…….

Aggrey Shisiali Shitsukane

This thesis has been submitted for examination with our approval as the University Supervisors

Signature……..Date…….

Prof. Wilson Cheruiyot, PhD

JKUAT, Kenya

Signature……..Date…….

Dr. Otieno Calvin, PhD

M.U, Kenya

Signature……..Date…….

Dr. Mgala Mvurya, PhD

TUM, Kenya

DEDICATION

Thank you, Almighty God.

I dedicate this thesis to my family, Catherine, Dainty, Daisy, Deyla, and Baris Nzing' for nursing me with affections and love and their dedicated partnership for success in my life. I sincerely thank you for your support and encouragement.

ACKNOWLEDGMENT

I would like to thank Prof. Cheruiyot, Dr. Calvin Otieno and Dr. Mvurya Mgala for their valuable suggestions, discussions, useful comments, remarks, and engagement throughout the learning process, as well as their guidance in writing this thesis. Special thanks to Jomo Kenyatta University of Agriculture and Technology for providing the necessary resources and environment for the research to be completed. Thanks to my mother for providing financial support and enduring this process with me, always offering guidance and love. I'd also like to express my gratitude to Mr. Rajab Omar for his professional assistance with all of our experiments. Mr. Wituka, who generously contributed both financial and moral support to the report, made this research possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

ABBREVIATIONS AND ACRONYMS

ANFIS	Adaptive Neural-Fuzzy Inference System
ANN	Artificial Neural Network
ANOVA	One-way analysis of variance
API	Application Programming Interface
CDS	Constant Degree of Similarity
CN-Fuzzy	Cascade Neuro-Fuzzy
CoG	Center of Gravity
DS	Degree of Similarity
FLC	Fuzzy Logic Controller.
FLD	Fuzzy Logic Designer
FOU	Foot Print of Uncertainty.
GA	Genetic Algorithm
GNRON	Goals are Non-Reachable with Obstacle Nearby
GUI	Graphical user interface
HSD	Honestly Significant Difference
IT2 FLS	Interval Type-2 Fuzzy Logic Systems.
LMF	Lower Membership Function

LSD	Least Square Difference
MATLAB	Matrix Laboratory
MF	Membership Function.
MFs	Membership Functions
MTWIP	Mobile Two-Wheeled Inverted Pendulum
PWM	Pulse Width Modulation.
ROS-	Robot Operating System
SRP	Similarity of Rule Premise
T1-FIS	Type-1 Fuzzy Inference System.
T1FLC	Type-1 fuzzy logic controller
T1FLS	Type-1 fuzzy logic systems
T1MFLC	Type-1 Mamdani Fuzzy Logic Controller
T2-FIS	Type-2 Fuzzy Inference System
T2FLC	Type-2 Fuzzy Logic controller
T2MFLC	Type-2 Mamdani Fuzzy Logic Controller.
TR	Type-Reduction
TSK	Takagi Sugeno Kang
UMF	Upper Membership Function

V-REP Virtual Robot Experimentation Platform

ABSTRACT

Autonomous mobile robot navigation has received a lot of attention from researchers. Traditionally, sensors are mounted on robots to detect the surroundings. However, they are sometimes inaccurate due to the pertinent problem of dealing with uncertainty in the environment. Fuzzy logic has long been regarded as a useful method for dealing with ambiguity that arises from imprecise knowledge. For wheeled mobile robots, several researchers have proposed three input proximity sensors and fuzzy logic controllers with 27 rules. Although this approach is interesting, it fails to account for uncertainty, resulting in difficulties avoiding obstacles. This research aimed to improve a Mamdani fuzzy controller by increasing the number of sensors, reducing the number of fuzzy rules, and revising membership functions. The developed type-1 fuzzy controller (M) was then compared to its corresponding type-2 (K). Simulation research method was adopted using commercially available V-REP and MATLAB software. A purposive sampling technique was chosen, and all simulations were run fourteen times. The results presented a new Mamdani fuzzy logic model with nine inputs, two outputs, and eighteen rules. ANOVA test revealed a significant effect of membership functions at $p<.05$ level for the three conditions [$F(2, 39) = 9.17$, $p = 0.001$]. Model comparison was done using an independent samples t-test. Model K had a higher score ($M = 219.79$, $SD = 4.509$) than model M ($M = 223.79$, $SD = 3.886$), indicating a disparity in the models $t (26) = 2-514$, $p = 0.018$) with a Cohen's d effect size of 0.924 meaning a large effect. Triangular membership functions constitute an immediate solution to the optimization problems in fuzzy logic modeling. Type-2 overcomes the limitations of type-1 fuzzy controllers presenting a way forward to fuzzy controllers in highly uncertain environments and real-world applications. It is envisaged to see a widespread use of type-2 fuzzy logic controllers in the next decade. The prospect of the thesis catalyzes additional research into a hardware implementation. This will aid in the development of robots for use in hazardous and crowded environments.

CHAPTER ONE

INTRODUCTION

1.1 Research background

Autonomous robot navigation establishes one of the main trends in robotics study. This is inspired by an existing gap between current technology and new application demands (Paulius & Sun, 2019). Many existing robots have little flexibility and self-sufficiency. Normally, the robots implement a pre-programmed sequence of procedures in an extremely controlled environment and are unable to function in new situations or tackle unpredicted states.

Studies and applications utilizing non-analytical methods such as Fuzzy logic, neural network and evolutionary computation paradigms are examples of computing methodologies that have demonstrated efficacy and potential smart control for multifaceted systems. Several studies suggest fuzzy logic as an appropriate tool for dealing with uncertainty and knowledge representation (Faisal et al., 2013; Ontiveros-Robles et al., 2018; Spolaor, 2019; R. Zhao et al., 2015).

Potential uses for autonomous robots include service robots that are intelligent for offices, factory floors, and hospitals; robots navigating in dangerous areas that cannot be accessed easily; home robots for housework or entertainment and semi-autonomous wheelchairs to support disabled people in society the (Abdulkareem et al., 2019; Alatise & Hancke, 2020). However, Pathfinding and motion control, on the other hand, have received far too little attention. Throughout the research Path planning in a traveling robot refers to methods for defining how a mobile robot will reach its destination safely while avoiding obstacles (Indri et al., 2019; Kanezaki et al., 2018; Plunk et al., 2015).

Path planning robots can be grouped as global or local methods. In the global planning technique, the mobile robot prerequisite is that the setting is fully known and stationary. Local path planning, on the other hand, necessitates a partial or unknown knowledge of the surroundings (Indri et al., 2019; Marin-Plaza et al.,

2018). In the latter autonomous robots use received sensory information for their local navigation (Signifredi et al., 2015). A set of actions were activated to make the desired performance. Although this approach was applauded there were empirical investigations on the uncertainty of sensors making robots unable to complete missions (Axelrod et al., 2018; Raulcezar & Lopes, 2016).

When a robot is steering using sensors, one of the problems that it faces is uncertainty in the environment. This is due to the large number of inaccuracies found in real-world readings. (Handayani et al., 2017).

The design of a robot control system needs to handle uncertainties by accepting vague and noise inputs without noteworthy effect on robot performance. In robotic navigation apparent uncertainties when using type-1 FLC include errors due to changing environments, sensor measurements, approximation of sensors and actuators features (Axelrod et al., 2018).

Fuzzy logic model reacts swiftly and is good when dealing with sensor measurement errors (Batti et al., 2019). Robot response in fuzzy logic is obtained from the quality of controller and inference type. The fuzzy logic methodology is applauded as a suitable tool for handling imprecision that arises from vague knowledge including process, model measurement, and implementation uncertainty (Berisha et al., 2016).

There are three types of Fuzzy inference systems namely Tsukamoto, Takagi Sugeno Kang (TSK) and Mamdani fuzzy (Selvachandran et al., 2019). A performance comparison of the TSK and Mamdani model for obstacle avoidance robot navigation was done (Farooq et al., 2011). MATLAB Fuzzy Logic Toolbox was used for the controller simulation. Output focused on the smooth movement of controllers and application memory. Farooq et al results presented Mamdani as a better choice which we aim to improve its performance in our thesis.

Type-1 fuzzy logic controller (T1FLC) has remained useful in various practical applications with great accomplishment. The main disadvantage is that ambiguity related to input and outputs membership functions cannot be handled successfully. It is normally grounded on experts' skills, membership functions (MFs) and knowledge

of fuzzy rules. However, uncertainties in real degree cannot be modeled once the membership functions have been well-defined (Handayani et al., 2019).

To overcome uncertainties, several studies still use Type-1 Fuzzy Inference System (T1-FIS) as a flexible and steadfast methodology capable of dealing with real-time steering (Gupta, 2014; D. Wu, 2013). Unfortunately, it has been criticized as having the inadequate capability to sustain uncertainty directly (Enyinna et al., 2015; D. Wu & Mendel, 2019). We optimized T1-FIS by sensor fusion, fuzzy rules reduction and tunning of membership functions. An alternative solution, Type-2 Fuzzy Inference System (T2-FIS) has been proposed providing leeway to prototype and uphold such uncertainty (Ontiveros-Robles et al., 2018). Type-1 fuzzy logic MF by crisp value can not apprehend proficiently the perceived vagueness in some applications. However, T2-FIS can give an effective image of variables (Baklouti et al., 2020; Ontiveros-Robles et al., 2018).

In the robotics system, performance enhancements can be introduced by T2FLS (Ajeil et al., 2020).The use of the extra dimension of MF gives systems better decision-making flexibilities and good representation of uncertainty than Type-1 specifically in the robotic field (Chao et al., 2020; T. Zhao et al., 2020).

Several studies focus on three input proximity sensors fuzzy controllers for obstacle avoidance robots (Berisha et al., 2016; Faisal et al., 2013; Najmurrokhman et al., 2019; Silva et al., 2014). This usually resulted in a dead zone and difficulties in avoiding obstacles. Although many studies have been conducted still the problem is insufficiently explored.

This research work addressed the development and optimization of Mamdani fuzzy logic controller model to autonomously navigate a wheeled robot. Using a reactive strategy, the robot interacts with the unknown environment by sensors mounted on the robot chassis to generate ultimate control commands (Faisal et al., 2013) however in some instances the robot did not complete its mission due to uncertainty in the environment.

The aim of the study was to develop an overarching fuzzy logic model for an obstacle avoidance robot in an unknown static environment. This thesis made some significant contributions to the field of autonomous wheeled robot navigation. The focus was on sensor fusion, rules reduction and selection of the most effective MFs in avoiding unknown static obstacles. The developed Type-1 Mamdani Fuzzy Logic Controller (T1MFLC) was converted to a corresponding Type-1 Mamdani Fuzzy Logic Controller (T2MFLC) and the performance compared. Potential application of the model includes the construction of wheeled robots for hazardous areas unsafe for human beings.

1.2 Statement of the problem

Mobile robots are used in various applications, including search and rescue missions, manufacturing, military, mining, and transportation (Alatise & Hancke, 2020). The most important task in robotics steering is to get to the destination while avoiding obstacles along the way (Indri et al., 2019). However, dealing with environmental uncertainties is a much more important requirement (Miller & Papanikolopoulos, 2020). Some research has been done on the potential of fuzzy logic controllers (FLCs) equipped with three sensors for obstacle detection and avoidance (Berisha et al., 2016). However, sensory deprivation, limited spatial coverage and imprecision are some gaps that cause dead zones and make maneuvering difficult (Najmurrokhman et al., 2019). In the traditional approach, researchers focused on a type-1 fuzzy logic controller (T1FLC) with obstacle distance measurements and input and velocity crisp value for output. This could not effectively treat perceived uncertainties in some applications (Baklouti et al., 2020; Elleithy et al., 2016; Shamsfakhr & Sadeghibigham, 2017; R. Zhao et al., 2015). Sensor fusion, fuzzy rule reduction, and the selection of appropriate MFs have received far too little attention when using T1FLC. Many researchers have compared the performance of type-1 and type-2 fuzzy systems in a variety of applications (Baklouti et al., 2020; Mendel et al., 2020; Naik & Gupta, 2017). They proposed that with more uncertainties, Type-2 Fuzzy Logic controller T2FLC can be used to improve efficiency, which has received less attention for obstacle avoidance robot navigation.

1.3 Justification of the research

There is a big market for autonomous robots, including office Robots, factory floors, semi-autonomous vehicles to assist disabled people in society, and robots operating in hazardous or areas that cannot be accessed easily. This implies great demand for optimum obstacle avoidance controllers. Several studies have proposed fuzzy logic as a useful tool for dealing with uncertainty and representing knowledge (Abdessemed et al., 2014; Ontiveros-Robles et al., 2018; Spolaor, 2019; R. Zhao et al., 2015). Being the case, many researchers are currently developing autonomous robots inspired by existing gaps between current technology and new application demands.

In robotics, pathfinding is the most important task i.e. reaching the destination while avoiding obstacles along the route (Indri et al., 2019). Though, a much bigger requirement is dealing with uncertainties in the environment (Miller & Papanikolopoulos, 2020). A lot of work on the potential of FLC with three input proximity sensors for detection and avoidance robots has been carried out. In the classical approach researchers usually concentrated on T1FLC with crisp values that did not treat proficiently the perceived uncertainties in some applications (Almasri et al., 2015; Baklouti et al., 2020; Ontiveros-Robles & Melin, 2020; Shamsfakhr & Sadeghibigham, 2017).

Many existing robots lack flexibility and self-sufficiency. Normally, robots adopt a pre-programmed sequence of procedures in a highly controlled environment and are unable to work in novel circumstances or cope with unpredictable outcomes. (Paulius & Sun, 2019). However, when using T1FLC, far too little attention has been paid to sensor fusion, fuzzy rule reduction, and the selection of appropriate MFs.Numerous authors have compared type-1 and type2 fuzzy logic controllers in several applications (Mendel et al., 2020; Naik & Gupta, 2017; Ontiveros-Robles & Melin, 2020).

They proposed that in case of increased uncertainties, T2FLC can be introduced to improve efficiency which has been given less attention.

A novel approach is therefore needed to guide robots to find direction in an indeterminate environment. Which combines sensory data for a fuzzy logic-controlled obstacle avoidance robot in a static unknown environment outside the controller to have a wider scanning area with reduced rules. This makes the model less computational for practical applications. Using fuzzy similarity measure the conventional twenty-seven if-then rules were effectively reduced to eighteen giving optimum results. We alternately revised Membership Functions for an ideal obstacle avoidance mobile robot. The three most commonly used MFs i.e., triangular, trapezoidal and gaussian were tested in a similar environment to select the most effective and efficient. The developed T1MFLC was compared with its corresponding T2MFLC in a simulated surrounding and evaluated the results using an independent samples t-test and Cohen's d effect size.

The purpose of this study was to improve on the traditional model by efficiently increasing the number of sensors, reducing fuzzy rules and tuning membership functions for T1MFLC. Subsequently to compare the performance of the optimized model with a corresponding T2MFLC using independent samples t-test analysis based on the time taken for the robot to reach the target. The study's contributed to the body of knowledge by presenting an optimized Mamdani fuzzy logic model for obstacle avoidance robots.

1.4 Objectives of the study

1.4.1 General objective

The general objective of this research was to develop a fuzzy logic model for obstacle avoidance mobile robots in static unknown environments.

1.4.2 Specific objectives

i. To demonstrate sensor fusion for a type-1 Mamdani fuzzy logic-controlled obstacle avoidance robot in a static unknown environment.

ii. To examine how to reduce fuzzy logic rules for an effective type-1 Mamdani fuzzy logic-controlled obstacle avoidance robot in a static unknown environment.
iii. To develop a model with appropriate Membership Functions for an ideal type-1 Mamdani fuzzy logic-controlled obstacle avoidance robot in a static unknown environment.
iv. To evaluate the optimized type-1 with its corresponding type-2 Mamdani fuzzy logic controller using an obstacle avoidance robot in a static unknown environment.

1.5 Research questions

This thesis was motivated by the following research questions.

i. How can sensors be used effectively to improve the performance of a fuzzy logic-controlled obstacle avoidance robot?
ii. How can fuzzy logic rules be effectively reduced to aid robots navigate in a static unknown environment?
iii. How can the Membership Functions be tuned for an ideal type-1Mamdani fuzzy logic-controlled obstacle avoidance robot?
iv. What is the difference between the optimized type-1 model with its corresponding type-2 Mamdani fuzzy logic controller?

1.6 Research scope

This research aimed to improve a Mamdani fuzzy logic controller for navigating an autonomous obstacle avoidance robot in a static environment. The research mainly focused on current approaches to robot navigation, designs of fuzzy logic control models, fuzzy logic rules reduction methods for effective performance, effects of membership function on the fuzzy logic controller and evaluating the developed Type-1 and corresponding Type-2 Fuzzy Logic sets for Robot path planning. The simulated robot consists of two wheels each driven independently by a pulse width modulated (PWM) DC motors and one castor. V-REP simulation software was used to model the environment of obstacles and robot modules. Mamdani fuzzy logic

inference technique consists of nine sensor models designed and two wheels powered using a PWM controller. Fuzzy rules were reduced by sensor fusion and Similarity-based simplification method. Environments created had scattered obstacles. Data was collected by experimental simulation with transducers mounted on the robot chassis to sense obstacles in the front, right and left for collision evasion and navigation. Fuzzy logic Toolbox in MATLAB was used for tuning and revision of membership functions. The robot was intended to avoid objects on a horizontal surface from the starting position to reach the target.

1.7 Limitations

The approach utilized suffers from several limitations.

- Fuzzy logic control systems rely heavily on expert knowledge, so rule bases must be updated regularly.
- Extensive testing was required for system validation and verification, which took a long time and was not completed conclusively.
- The profile of the robotic simulation field included 12 cylindrical obstacles, which are not always present in real-world situations.
- The robot moved forward only, with no reverse motion, and the environment is static, with no dynamic obstacles.
- Simulations on computers with varying processor speeds and computational power give varying ranges of times to reach the target.
- There is a time lag during data collection due to the client-server connection method used by V-REP and MATLAB.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

In the literature review, there are several methods for dealing with the problem of autonomous mobile robot approaches to navigating in an unknown environment. This section presents a review of recent literature on obstacle avoidance robots in a static unknown environment. The review is conceptualized under the objectives of the study and focuses mainly on current approaches to robot navigation, designs of fuzzy logic control models, fuzzy logic rules reduction methods for effective performance, effects of membership function on fuzzy logic controller and comparison between Type-1 and Type-2 Fuzzy Logic sets for Robotic path planning. Finally, some gaps and shortcomings are identified.

2.2 Approaches to obstacle avoidance robot navigation

A lot of efforts are devoted to studying mobile robots to resolve navigational problems in numerous studies reviewed (Mittal et al., 2020) . However, this continues to be a disputed topic among researchers as reviewed (Miller & Papanikolopoulos, 2020). Table 2.1 summarizes approaches used for obstacle avoidance robots. Researchers are keen to develop novel techniques that can offer a smooth and ideal collision-free path. Fuzzy logic stands out as the best method to navigate robots to their destination. Although numerous studies have been conducted, (Berisha et al., 2016; Faisal et al., 2013; Najmurrokhman et al., 2019; Silva et al., 2014) the problem of safely navigating autonomous robots are still insufficiently explored in dealing with uncertainty. Additional studies to understand completely the key tenets of fuzzy logic-controlled obstacle avoidance robots are required.

Table 2.1: Advantages and limitations of obstacle avoidance approaches

	Approach	Advantages	Limitations
1	Potential fields (A. A. Ahmed et al., 2015)(B. Li et al., 2017)(Rasekhipour et al., 2017)	-superior in contrast to global path planning -Simplicity and mathematical elegance.	-The potential force of the robot is zero when attractive force and repulsive is equal or almost equal then in the opposite direction thus trapped in local minima or oscillation. - Robot move close to obstacles When attractive force becomes very great Therefore risk of collision to obstacles -"Goals are non-reachable with obstacle nearby" (GNRON) -Do not perform well in complex scenarios having many obstacles.
2	Neural networks (Wong et al., 2017),(K. Wu et al., 2019)	-Minimization of time required for training the network -Able to absorb and model non-linear and intricate relationships - can infer unseen relationships After learning from initial inputs and their relationships.	-No clue how results are generated, black box, so if you want to know what causes the output you can't with a neural network. -Need a large data --Hardware requirements are large -Suitable for complex problems else costs overweighs benefit
3	Genetic algorithm (Herrera Ortiz et al., 2013) (Tuncer et al., 2012)	-Easy to understand -The solution gets better with time. -Optimization is good for noisy environments.	-No assurance of getting global maxima but chances of getting trapped in a local maxima - good results are found on a sized population and a lot of generations. - Parameters fine tuning for GA, similar to mutation rate, elitism percentage, crossover parameters, fitness normalization/selection parameters, etc., is often trial and error.

			-communication to the system is through the fitness function. The result could be crazy, inefficient or incomprehensible
4	Vision-based navigation (English et al., 2014). *(Tsai et al., 2013).*	-Accurate -Less noisy.	-locating obstacle fail when the mark cannot be seen -The main drawback of this method are in terms of cost and challenges associated to installing
5	Fuzzy logic (X. Li & Choi, 2013) *(Chaudhari & Patil, 2014).* (R. Zhao et al., 2015)	-Easy to model the reasoning. -Can deal with ambiguity and nonlinearity. - use of linguistic variables and easy to implement. -Imitate human control logic. -Use vague language and is characteristically stable. -Fuzzy controllers are flexible and can be changed without difficulty. -It can be combined easily with other control techniques.	-logic requires experimentation and experience. -Finding suitable function can be by trial and can take quite some time.

2.1.1 Potential fields-based approach

Potential fields' method has been appreciated for many years among researchers as one of the capable techniques for controlling a mobile autonomous robot. Many attempts have been made (A. A. Ahmed et al., 2015; Rasekhipour et al., 2017; Wahid et al., 2017) proposing potential field methods for the navigation of mobile robots putting forward essential limitations of the technique. A more comprehensive description can be found in the aforementioned authors. They explained that potential field techniques can be used elegantly and easily for navigation of autonomous robots. Comparisons between theoretical and experimental results of

their proposed methods are presented for evaluation. It shows that potential field is positively adopted with some deficiencies in dealing with environmental ambiguity.

A seminal article on “A Potential Field-Based Model Predictive Path-Planning Controller for Autonomous Road Vehicle” was presented (Rasekhipour et al., 2017). Diverse obstacles and road erections were placed in the path planning system while planning an optimum path with vehicle dynamics. In some complex test situations, the path planner was modelled and simulated on a CarSim model. The results show that the vehicle avoided obstacles with this path plan controller and observed road rules with little consistency.

Investigations into advanced numerical potential field technique, and route planning in robotics. The authors affirm that the method is superior contrast to the global path planning technique since a global path planner needs additional computational time. This algorithm solved a variety of path planning problems (Barraquand, 1992).

Artificial potential field integrating fuzzy logic was proposed as an efficient approach for mobile robot navigation (Melingui et al., 2014). In this study, both advantages and weaknesses were highlighted. The incorporation of the two techniques into a common control scheme improved the performance of the resultant hybrid controller. An omnidirectional mobile robot was designed to validate the efficacy of the proposed control system. However, in this work and related references it was observed that the approach does not perform well in an environment with many obstacles.

In this study (Wahid et al., 2017) some possible field functions were used to track vehicle desired movements to plan and control collision avoidance driver assistance systems. The parameter value was determined based on the human driver's perception of risk in the design of the potential field function to adapt the driving behavior of people to avoid collision. The potential function of the street borders kept the car on its track. In future, they recommended using a real experimental vehicle.

A multi-objective optimization problem was modeled for autonomous robot navigation based on three objectives, minimizing the distance to the goal, maximizing the distance between the robot and the obstacles, and minimizing the distance traveled. The concept was simulated by three altered obstacles alignments and ten routes of diverse nature to demonstrate robustness and capability (Herrera Ortiz et al., 2013).

Sumo, a direction-finding robot in dynamic environments with moving targets and static obstacles was presented (Carlos Erlan Olival Lima et al., 2013). Potential field method was used aimed at planning velocity and robot direction to reach the target within the shortest possible time. They designed a hybrid controller making use of the potential field with a Mamdani fuzzy logic controller to acquire essential variables for defining velocity and direction. In validating performance, a simulation was done using MATLAB. Results exposed that hybrid technique overcome local minima in static or dynamic environment.

In a similar manner potential functions for path planning of mobile robots were used (A. A. Ahmed et al., 2015; B. Li et al., 2017) in various environments,. The major drawback was the existence of local minima. The technique only considered the immediate best course of action, the robot did not move towards the global but got stuck in a local minimum of the potential field function. When the robot was far away, the attractive force became very strong, causing it to move very close to obstacles, risking collision.

2.1.2 Neural networks-based approach

Many studies have been published on artificial neural network (ANN) method for path finding in mobile robot navigation. In the study (Kanezaki et al., 2018) employed ANN and prototyped a complex relationships involving inputs and outputs of a controller. They proposed a common method to infer data from several types of two-dimensional distance sensors and a neural network set of rules to carry out the navigation task. The approach was applied to different types of range sensors and robot platforms. The method contributed significantly to minimizing the time required for training the network.

An intelligent path planning robot avoiding obstacles in difficult and unknown environments, a feed-forward neural network based on feature approximation using back propagation algorithms (Shamsfakhr & Sadeghibigham, 2017). The proposed method was successfully applied to real-world data. The algorithm developed a logical behavior for a mobile robot, considering the kinematic constraint of the autonomous robot. Navigation time tests revealed that achieving highly accurate steering angle values causes the robot to move more steadily when corners occur, reducing both the path length and navigation time in the presence of uncertain environments.

In order to solve navigation difficulties in a known environment, a combination of fuzzy logic and spiking neural networks was presented (LAOUICI et al., 2014). Simulation results demonstrated challenges in implementing hybrid design for robots working together.

A radial basis function neural compensator for an autonomous mobile robot was implemented (Rossomando et al., 2011). They used a kinematic inverse dynamic controller for the proposed study. They claimed that an adaptive controller was efficient enough in terms of tracking. Demonstrations validated results through various navigational exercises.

In this study, a neural network learned from human driving data (K. Wu et al., 2019) was implemented to model the prevention of barriers utilizing dense areas. The trained neural network was then evaluated in various scenarios and compared to the ideal network structure using cross-validation (number of nodes and layers). The findings were also contrasted with other obstacle reduction methods that served as the basis for comparison.

Researchers proposed velocity measurement of the leading robot using formation control. They presented a reference trajectory that the leader generated and the follower robot tracked (Ghommam & Saad, 2014). Authors (Eraqi et al., 2016) researched a reactive anti-collision scheme of a surface vehicle using a neural network. This study described the architecture of a neural autonomous surface vehicle for tracking while avoiding a collision as reliable for a known environment.

A static and dynamic obstacle avoidance Q-learning and neural network algorithm inspired by the computational time of conventional methods was proposed (Wong et al., 2017), their work allowed efficient mobile robot navigation, at adjustable speeds and avoiding local minimum requirements. They presented a travelling robot steering in both static and dynamic environment. The results showed a mobile robot collision-free trajectory with uncertain workspace using neural networks. This method had the disadvantage of being computationally intensive and required a very powerful processor for practical applications.

2.1.3 Genetic algorithm-based approach

Genetic algorithm (GA) is an evolutionary method, grounded on the standard of survival of the fittest and natural selection; it has been broadly put to use in various fields of science and research (Lamini et al., 2018; López-González et al., 2020; Oleiwi et al., 2015; T. Zhao et al., 2020). GA is a search-based algorithm that attempts to find a suitable individual by the evolution of the original population. The greatness of GA lies in the capability to handle nonlinear and complex multi-objective problems (Herrera Ortiz et al., 2013). The major disadvantages of this algorithm are fast convergence and the need for a large number of evolutions to reach a global solution.

In research on "knowledge-based genetic algorithm for path planning of mobile robot" wheeled to avoid obstacle was presented (Yanrong Hu & Yang, 2004). This view was supported by the study "Path Planning for Robot Navigation Based on Cooperative Genetic Optimization" (Hsu, Chen-chien, 2014). They proposed a straight path for obstacle avoidance with computation effectiveness using a cooperative genetics algorithm to mitigate collision problems. The robot successfully avoided obstacles, the major defect in their experiments was that they entailed tedious training sessions.

Path planning in a static environment algorithm was investigated in research work (Ismail et al., 2008). They experimented with GA in handling different types of tasks in static environments. Results report the proposed method is efficient but lacking consistency.

This study suggested an improved crossover operator (Lamini et al., 2018) to solve problems of route planning in static environments with genetic algorithms (GA). GA was commonly used in road optimization problems to find a viable route between two positions while avoiding obstacles and optimizing criteria like distance, protection and other criteria. The proposed crossover operator prevented premature convergence and provided viable paths that were stronger in fitness than their parents, thereby converging the algorithm more quickly. A new fitness feature was also suggested that took account of distance, protection and energy. It was applied in several ways and compared to literature research to show the validity of the proposed technique.

A Motion Planning System for Mobile Robots using *GA* to find an achievable path for a mobile robot in surroundings with obstacles was proposed (Tuncer et al., 2012). They implemented a grid-based environment prototype used in indoor applications. The major shortcoming was the need for a large number of evolutions to get a solution.

As an alternative to the traditional genetic algorithm for the global planning route of a moving robot, a study developed an enhanced genetic algorithm. The benefit of the model was the ability to efficiently direct navigating robots from the initial stage to the final target without any collisions (F. Ahmed & Deb, 2012).

A technique for robot route planning built on a modified genetic algorithm to discover a viable route for a mobile robot working in a dynamic environment was explained (Yan-ping & Bing, 2010). Simulated results demonstrated that the proposed system achieved considerable improvements in comparison to basic *GA.*

2.1.4 Vision-based navigation

A novel vision-based tracking method to guide autonomous vehicles in agricultural fields was presented to perform various tasks on the farm. Evaluated results applauded this as a novel model (English et al., 2014) though it was expensive to install cameras.

This work (Yu et al., 2018) presented a quadrotor-equipped with a visual sensor using an obstacles avoidance approach that allowed an untrained vehicle to estimate the distance between the obstacles and move to the target point. The efficiency of the system is checked on real experimental ground.

A framework for avoiding dynamic obstacles in the course of visual navigation using a mobile robot with wheels was designed and validated (Cherubini et al., 2014). Steering entailed following a pathway, characterized as an ordered set of significant pictures captured by an onboard camera in a training stage. The robot was capable to avoid stationary and dynamic obstacles even those that were absent in the teaching phase. The approach took explicitly into account the velocity of obstacles projected using a Kalman-based observer.

Machine vision centered obstacle avoidance method for a robot using a camera was presented; it accomplished obstacle escaping as well as path scheduling. This system was put together using a camera and two laser projectors that were secured to the base. When the robot encountered an unfamiliar environment, it came to a halt and took photographs. To distinguish obstacles, the system used simple image processing steps. (Tsai et al., 2013).

The work (McDowell et al., 2012) illustrated the representation and implementation of the genuine methodology in the scheme of planning path for a traveling robot using the vision devices. They gave various exercises that demonstrated the usefulness of the technique.

Studies (Ohnishi & Atsushi, 2013) proposed a hybrid paradigm using visual potential field method for robot direction finding. In comparison with simple vision-based algorithm, performance demonstrated the capability of robots to avoid an obstacle in a complex environment. The key shortcoming was the need to keep obstacles within the vision sensors region. To improve coverage and reliability, multiple cameras were to be installed. The cost of equipping the mobile robot with high-performance navigation hardware was high.

2.1.5 Fuzzy logic-based approach

The fundamental characteristic of fuzzy logic is using extraordinary reasoning capability like a human being for making a decision. The next section gives a review of prior work that relates to mobile robot navigation in previous decades. Fuzzy Logic is an instrument for modeling ambiguous systems decision making by allowing common sense reasoning in the deficiency of detailed and precise information (Ren, n.d.). It allows coming up with a confident deduction using input data that is ambiguous, indeterminate, vague, and noisy (Baker & Ghadi, 2020).

2.2 Fuzzy logic controller

Key features of Fuzzy logic-based control system include the construction of dynamic and smooth controller mechanisms starting with heuristic knowledge and qualitative models. Using unreliable, inaccurate, vague information and amalgamation of symbolic reasoning with numeric processing (Spolaor, 2019).

An inference engine and a set of linguistic IF-THEN rules that code the mobile robot's actions are the main components of a fuzzy logic controller. The major problem in developing fuzzy logic control (FLC) is the effective creation of fuzzy IF-THEN rules. The antecedent of a fuzzy rule base is simple to create but difficult to produce without expert knowledge of the rule base. (Prokopowicz et al., 2017). A generalized fuzzy logic robot controller is depicted in Figure 2.1.

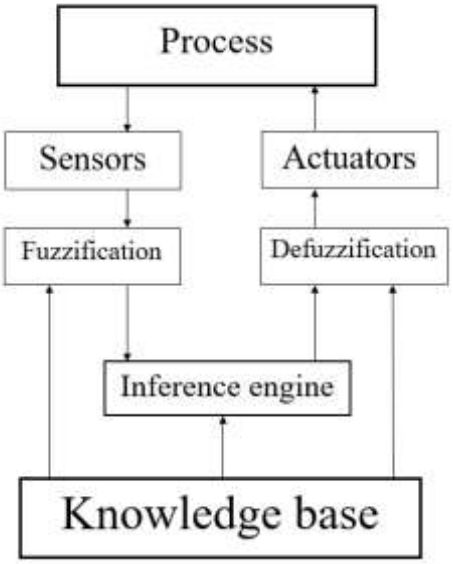


Figure 2.1: Fuzzy decision-making controller

There are two connections between the fuzzy logic controller and the process under control, i.e., input and output links. The inputs of FLC are interconnected to the process through sensors and the outputs are linked to the process through actuators. Fuzzification, inference mechanism, and defuzzification are the main components of the fuzzy logic controller. Navigation is achieved using information stored in the knowledge base (Prokopowicz et al., 2017).

2.2.1 Types of fuzzy inference systems

Based on the foundation of the fuzzy rules required for the inference procedure, fuzzy inference engines are classified into three categories. They include Takagi-Sugeno-Kang (TSK), Mamdani and Tsukamoto fuzzy inference systems (Melek, 2018).

2.2.2 Mamdani fuzzy inference system

The Mamdani inference technique was created to track the performance of a boiler and a steam engine using a variety of linguistic variables. The Fuzzy rules of the system were acquired by a professional human operator (Hessburg & Tomizuka, 1993).

Mamdani fuzzy inference system rule is of the following structure (Najmurrokhman et al., 2019)

If x is large, then y is medium.

If x is small, then y is big.

Both linguistic variables and fuzzy are implications and antecedents of the laws. The system produces fuzzy output that is required to be converted into a crisp. This is why different defuzzification procedures are used for transforming the fuzzy output into a crisp one (Selvachandran et al., 2019).

2.2.3 Takagi-Sugeno-Kang fuzzy (TSK)

The TSK model was presented by researchers Takagi, Sugeno, and Kang for making an organized method to generate fuzzy rules from known input and output data sets (Enyinna et al., 2015).

Typical rule methodology follows as,

$$If\ x\ is\ B\ and\ y\ is\ C\ the\ Z = f(x,y) \dots\dots\dots\dots\dots\dots\dots .2.0$$

The antecedent is defined by a function x and y in its fuzzy form and consequent to the rule.

Fuzzy input. $z = f(x,y)$ (crisp function).

$$Fuzzy\ input\ Z = f(x,y)(crips\ function) \dots\dots\dots\dots\dots\dots .2.1$$

If $f(x,y)$ is a first-order polynomial, the inference scheme is referred to as the first-

order TSK fuzzy method. It is known as zero-order Sugeno fuzzy inference model If F is constant which is a unique instance of Mamdani fuzzy model (Deng et al.,

2013). The order of the TSK fuzzy model is determined by a polynomial (Couceiro & Marichal, 2010).

The TSK Fuzzy system's global output is achieved through the weighted average. The Weighted mean method is often replaced by a weighted sum operation to prevent computation problems(Enyinna et al., 2015).

2.2.4 Tsukamoto fuzzy inference system

The consequences of fuzzy if-then rules are represented by a fuzzy set with monotonic membership functions in the Tsukamoto fuzzy inference engine. As a result, all rule outputs are specified as a narrow value. The total output is a weighted average of each rule's output. (Chaudhari & Patil, 2014). However, it's not usually used to build systems since Tsukamoto's fuzzy inference method isn't explicit compared to TSK and Mamdani's fuzzy model (Kansal & Kaur, 2013). The following are the principles of this inference system;

If x is small then y is f1.

If x is medium then y is f2.

Results are usually fuzzy sets irrespective of the data, and the performance of the Tsukamoto fuzzy inference method is crisp (Prokopowicz et al., 2017; Singh & Mishra, 2015).

2.3 Fuzzy logic systems

Zadeh proposed fuzzy sets with fuzzy membership functions in 1975 as an extension of the fuzzy set, which formed fuzzy sets of type, n =2, 3. Membership function ranges over fuzzy sets of type n-1 (Zadeh, 1965). The fuzzy logic was founded on the fuzzy set theory where a particular object or variable had a degree of membership in a given set that range between 0 to 1. The basic set operations of Boolean logic, such as intersection (AND), complement (NOT), and union (OR), are useful for fuzzy logic. (Prokopowicz et al., 2017).

2.3.1 Membership functions

Fuzzy logic membership functions (MFs) are curves that outline how each point on input is plotted to a membership worth or degree of belonging between zero and one. Input space is normally called the "universe of discourse". MFs are always used during fuzzification and defuzzification stages in the Fuzzy controller System. They plot input strengths that are not fuzzy to fuzzy linguistic terminologies and vice versa (Thaker & Nagori, 2018). To quantify linguistic terms, membership functions are used. Numerical figures do not have to be fuzzified by a single membership function. A value can fit multiple sets at the same time. (Prokopowicz et al., 2017). This gives fuzzy an important useful feature.

Using straight lines, simple membership functions are formed. The widely used are Triangular MFs and trapezoidal MFs because of their simple mathematical formula and performance competence, particularly in real-time applications (Kreinovich et al., 2020). However, at the turning points specified by parameters, they are not smooth because they are made of straight-line sectors (Najmurrokhman et al., 2019).

2.3.2 Types of membership functions

The most common types of MF are: (Kreinovich et al., 2020).

i. Triangular
ii. Trapezoidal
iii. Generalized bell
iv. Gaussian
v. π- Shaped
vi. S-Shaped

The Aforementioned list of MFs is by no means exhaustive; For definite applications if necessary other specific MFs can be formed.

There are three key features used in characterizing the membership function which is shown in figure 2.2. (Prokopowicz et al., 2017)

The *core* is signified by whole membership function within the set. Core of a fuzzy set A is well illustrated as the section of workspace with elements x fulfilled by expression $\mu A(x) = 1$.

For fuzzy set *A* the region of space that is categorized with non-membership function is defined as *support* of a membership function. "The support includes elements x of workspace where $\mu A(x) > 0$".

The Boundary of MFs of a fuzzy set *A* is described as part of the workspace that has no zeros nonetheless not a whole MFs in that set. "The boundary is made of elements x in the universe where $0 < \mu A(x) < 1$".

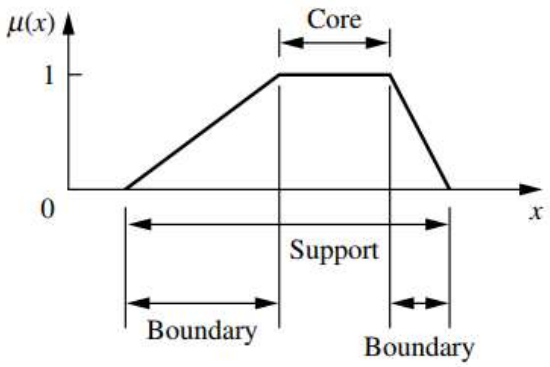


Figure 2.2: MFs Features

2.3.3 Triangular membership functions

Triangular membership function is among the most basic normally used linear function (Al-mayyahi & Wang, 2014; Kreinovich et al., 2020). Figure 2.3 illustrates MF points a, b and c denote the x coordinates with three vertices of μA(x) of set A. The coordinate a is defined as the lower boundary of set A with a membership degree

of zero. The upper boundary coordinate c is shown with a degree of membership of zero. Lastly, coordinate b has degree of membership as one and is the apex of triangle. Triangular membership functions are well thought out as suitable for apprehending the imprecision of linguistic assessments (Barua et al., 2014; Kreinovich et al., 2020).

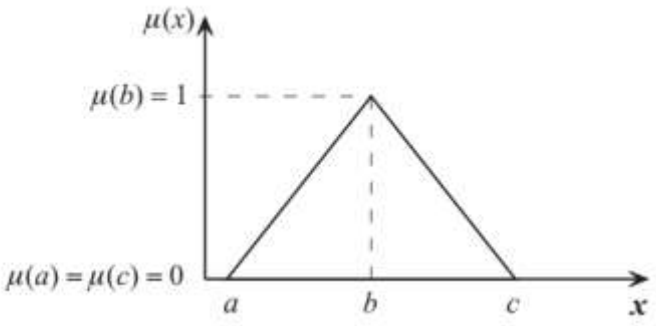


Figure 2.3: Triangular MF (Barua et al., 2014).

Eqn.2.2 represents mathematical formula for calculating degree of membership of elements.

$$f(x,a,b,c) = f(x) = \begin{cases} 0, & x \le a \\ x-a)/(b-a), & a \le x \le b \\ \dfrac{c-x}{c-b}, & b \le x \le c \\ 0, & c \le x \end{cases} \quad \ldots\ldots\ldots\ldots\ldots 2.2$$

Using max and min we can have an alternative expression as equation 2.3

$$triangle(x;a,b,c) = \max\left(min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \ldots\ldots\ldots\ldots\ldots 2.3$$

The elements (a, b, c) having a<b <c regulates fundamental triangular MF x coordinates of the three corners.

2.3.4 Gaussian membership function

Gaussian membership function is another very importantly used in fuzzy logic. A Gaussian MF is defined by two parameters, which are represented by Equation 2.4.

$$(x) = \left[\left(-\frac{b}{\sigma}\right)2\right] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.4$$

With x denoting input variable, b membership function center and σ as a constant which signifies the membership function width. Gaussian MFs are commonly used with regards to fuzzy systems (H. M., & Panigrahi, B. K. 2015). Figure 2.4, shows an ideal Gaussian membership function.

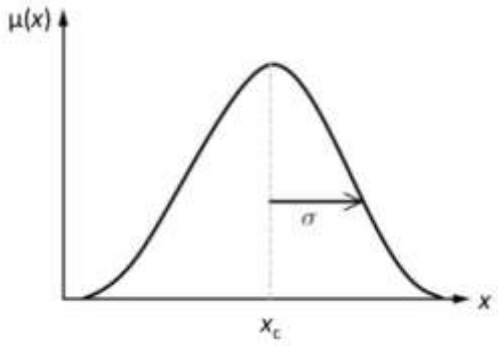


Figure 2. 4: Gaussian MF (Barua et al., 2014)

2.3.5 Trapezoidal membership functions

Four parameters are used to quantify a trapezoidal MF (a, b, c, d) as illustrated in figure 2.5 supported by equation 2.5.

$$trapezoidal(x;a,b,c,d) = \begin{cases} 0, & x \leq a \\ x-a/b-a, & a \leq x \leq b \\ 1, & b \leq x \leq c \\ d-x/d-c, & c \leq c \leq d \\ 0, & d \leq x \end{cases} \ldots\ldots\ldots 2.5$$

An alternative expression by means of min and max is shown in equation 2.6.

$$trapezoid(x;a,b,c,d) = \max\left(min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \ldots\ldots\ldots 2.6$$

The parameter (a, b, c, d) having a < b < c < d Limit the x coordinates of the basic trapezoidal MF's four corners.

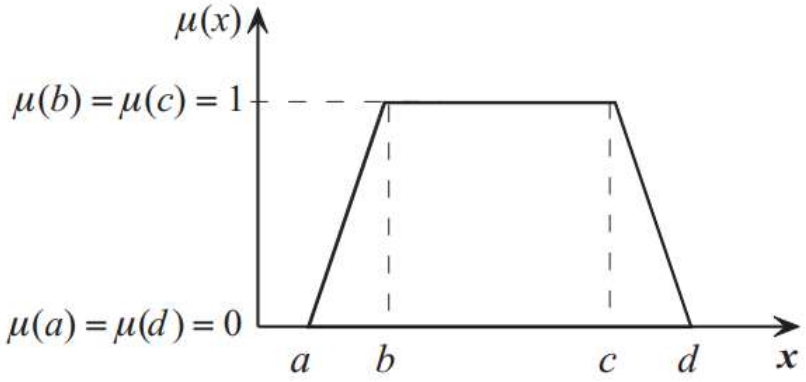


Figure 2.5: Trapezoidal MF (Barua et al., 2014)

Triangular and trapezoidal Membership Functions are widely used, particularly in real-time applications, due to their simple formulas and computational effectiveness. (Kreinovich et al., 2020). However, for the reason that the membership Functions are made of straight-line parts, smoothness at the corner points is limited by their parameters.

2.3.6 Defuzzification

The overall result is a fuzzy value after the inference stage. The output should be defuzzied to get a crisp output (Prokopowicz et al., 2017) which is the function of a Fuzzy Logic System's defuzzier section. Defuzzification is accomplished based on the output variable's membership function. As shown in figure 2.6, the shaded areas are all part of the fuzzy result. The goal is to extract a crisp value which is represented by a dot in the figure.

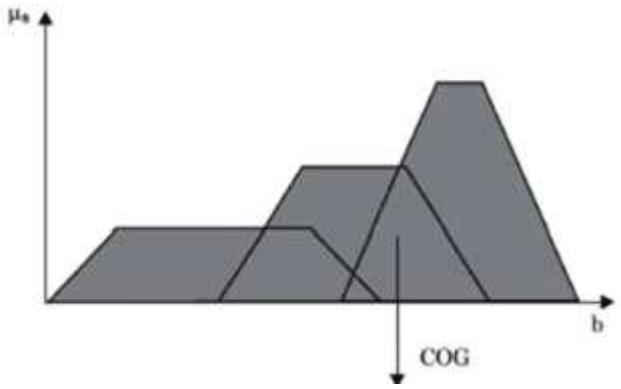


Figure 2.6: Defuzzification steps (Omrane et al., 2016)

If no defuzzification is performed, the final output from the inference stage remains a fuzzy set. However, a crisp control signal is required in many applications. A fuzzy set is reduced to a single numbered output in this step. (Leottau & Melgarejo-Rey, 2010). There are several defuzzification methods available, some of which are discussed next:

2.3.7 Centre of gravity / Weighted average method

The Center of gravity method (COG) is the most popular defuzzification technique and is widely utilized in practical applications. It is most prevalent and physically appealing of all the defuzzification methods (Baker & Ghadi, 2020). Center of gravity method is one of the most accurate methods for estimating crisp value from fuzzy output (Ruiz-Garcia et al., 2019). It is a method for obtaining a crisp value (u) from fuzzy set output's middle point using a weighted average of membership grades. (Melek, 2018). Assume there is a fuzzy set in a discrete universe with membership value in the membership function. The weighted average of the elements in the support set is described by expression 2.7.

$$COG = \frac{\int_z \mu_A(z) z dz}{\int_z \mu_A(z) dz} \pi r^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.7$$

The aggregated output of $\mu_A(z)$ is MF and z denotes the output number. Centre of

Gravity is most commonly adopted defuzzification technique. The above may be set as equation 2.8 for discrete values

$$COG = \frac{\sum_{k=1}^{n} \mu_A(z_k) z_k dz}{\sum_{k=1}^{n} \mu_A(z_k) dz} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.8$$

Where $\mu_A(Z_K)$ is k=1,2…,n sample values of the combined output membership

function.

2.3.8 Mean of maximum (MoM) method

The Mean of Maximum method determines the average z when the membership of the fuzzy set is at its maximum. (Ross, 2004). Numerous maximum points exists so, taking the mean of all maximum values is the common practice. This method is computationally simpler than others and produces relatively good results, but it ignores the shape of the fuzzy set entirely.

As previously stated, fuzzy systems make use of fuzzy reasoning and fuzzy rules to decide in a given situation. Fuzzy reasoning is based on the extension principle, which enables a systems developer to map a function sandwiched between two fuzzy sets. (Prokopowicz et al., 2017) there exists a fuzzy set A in a universe Z for any given set-up.

$$Z = \frac{(Z1 + Z2)}{2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.9$$

The average output value is achieved with this defuzzification technique in which the first value is Z1, while the last is Z2, in which the total output $\mu_A(z)$ function is the

limit using equation 2.9.

2.3.9 Linguistic variables and fuzzy if-then rules

Professor Zadeh (Zadeh, 1965) suggested the concept of linguistic variables that are otherwise "fuzzy." Which are linguistic words or objects instead of numbers. A sensor's input is a noun, such as distance, velocity, temperature, flow, and so on. Subsequently the error is the respective difference thought of the in similar manner. The fuzzy variables are modifiers for variables (e.g., positive, small positive, negative, large negative). Additional ranges, such as very very large and very very small, can be added to spread sensitivity to special or nonlinear conditions. However they are not essential for a basic system (Melek, 2018).

The rules of the fuzzy inference system are verbalized upon the linguistic variables and values being defined. fuzzy inputs are mapped to fuzzy outputs using the rules. Mapping occurs via a compositional rule of inference built by Zadeh's extension of modus ponens, traditionally the if-then conditional form. A fuzzy logic if-then rule, also called a fuzzy rule, has the general structure.

When x is A then y is B,

Where A and B are linguistic values that are described by fuzzy sets within the discourse universes X and Y. The antecedent is "x is A," while the consequent is "y is B." The rule is commonly abbreviated $A \rightarrow B$. The antecedent is typically made up

of input variables, while the consequent is contains output variables. (Ren, n.d.).

2.3.10 Development of type-1 fuzzy controller.

Fuzzy logic system generally has four components: inference engine, fuzzifier, defuzzifier and rule base as illustrated in Fig.2.7. The input variables are the input grades known as fuzzy variables, which are fuzzified to determine the degree of crispness. Rule base is produced by gathering information from an expert or through trial and error. Inference is the process of modeling to show how an expert determines the amount of input. Fuzzy logic variables are then transformed by

defuzzification stage to a precise output variable that is crisp (Handayani et al., 2019).

A type-1 fuzzy set is made up of pairs of $(x, \mu_A(x))$, that for each member of domain $x \in X$, a membership value $\mu_A(x) \in [0,1]$ is expressed as eqn.2.9 (Handayani et al., 2019).

$$A = \{(x, \mu_A(x)) I \forall x \in X, \mu_A(x) \in [0,1]\} = \sum_{x \subset X} (x, \mu_A(x)) \ldots\ldots\ldots 2.9$$

At this point, X is the universe of discourse and $\sum is$ the summation of elements within the set. All elements in type-1 fuzzy sets have a degree of membership, however it can not quantify the level of uncertainty in membership degree (Handayani et al., 2019) . Researchers (Baklouti et al., 2020) proposed type-2 fuzzy sets to quantify uncertainty in the degree of membership.

Researchers investigated fuzzy controller for a toy car that was constructed using a sonar sensor and a microcontroller to implement fuzzy rules. In regulating steering angles the wheeled toy competently negotiated sharp corners (Leottau & Melgarejo-Rey, 2010).

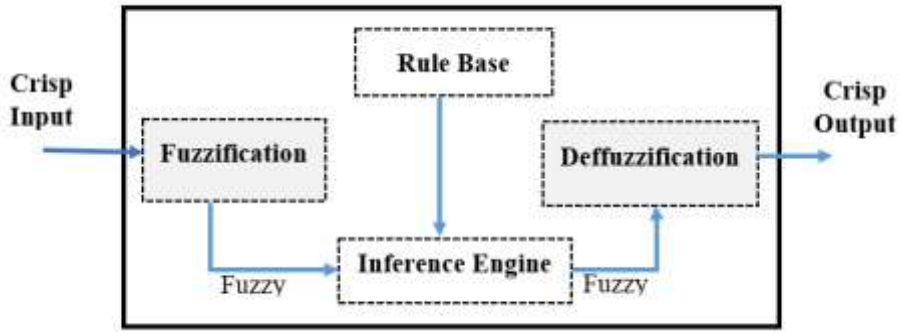


Figure 2.7: T1-FIS architecture

In recent years there is a considerable trend in using genetic algorithms and neural network to improve fuzzy sets. A few research work done include “Adaptive neuro-fuzzy based hybrid force position control for an industrial robot manipulator” (Chaudhary et al., 2016). “Neuro-fuzzy control method for balancing a two-wheel mobile robot” (View & Pandey, 2017) research work, “Fuzzy-Simulated Annealing (Fuzzy-SA) algorithm” “Hybrid Fuzzy (H-Fuzzy) architecture”, “Cascade Neuro-Fuzzy (CN-Fuzzy) architecture”, , “Wind Driven Optimization (WDO) algorithm”, and “Fuzzy-Wind Driven Optimization (Fuzzy-WDO) algorithm” , developed d and realized to resolve navigation difficulties of a mobile robot in numerous static and dynamic environments.

A lot of studies abound to literature of fuzzy and neural-fuzzy controller of path planning robots. Most works is grounded on type-1 fuzzy logic systems. Type-1 sets don’t account for ambiguity in input measurements because it is a two-dimensional fuzzy. Sensors are notorious for noise inherent; typically twisting the accuracy of measurements. The crisp outputs do not completely account for vagueness in the actuator control. Uncertainties may also arise in describing linguistic variables. Though type-1 fuzzy controllers have shown excellent results in controlling mobile robots they normally do not wholly account for all uncertainties occurring in the antecedents, consequents and inputs inherent in the external surroundings of a mobile robot (Cherroun et al., 2019).Several researchers revealed these difficulties while comparing type-1 and the type-2 fuzzy logic controllers for autonomous robots (Handayani et al., 2019; Naik & Gupta, 2017).

2.3.11 Type-2 fuzzy logic systems overview

Fuzzy logic controllers have been developed as a real-world substitute for traditional control techniques because they aid in decision making and allow the developer to use expert knowledge in the control mechanism. Classical fuzzy logic systems (T1FLC) cannot completely take care of measurement, linguistic and parameter uncertainties (Handayani et al., 2019). A new class of fuzzy logic structures, Type-2 fuzzy logic system was applauded for minimizing the effects of uncertainties (Mittal et al., 2020). Zadeh first proposed the concept of type-2 fuzzy sets as an extension of

ordinary type-1 fuzzy sets. When determining a precise membership function is difficult,Type-2 fuzzy sets are useful for dealing with uncertainty. (Ruiz-Garcia et al., 2019)

A three-dimensional membership function defined by a type-2 fuzzy set is known as a type-2 membership function. If we consider blurring the type-1 membership function as shown in Figure 2.8(a) by shifting points on the membership function to the right or left by different amounts, as shown in Figure 2.8(b). Then, at a value of x, say x_1, the degree of membership is no longer a clear value, but rather takes on

values where the vertical line crosses the blur. Because those values are not all equally weighted, an amplitude distribution can be assigned to all of them. If this is done for all x ∈X (X denotes the universe of discourse), then, a three-dimensional

membership function known as a type-2 membership function is formed, which is characterized by a type-2 fuzzy set. (Ontiveros-Robles et al., 2018).

A type-2 fuzzy set, denoted by $\tilde{A}$, is distinguished by a type-2 membership function.

$\mu_{\tilde{A}}(x,u)$,with $x \in X$ and $u \in J_x \subseteq [0,1]$, articulated in eqn.2.10.

$$\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) | \forall x \in X, \forall u \in J_x \subseteq [0,1]\} \dots\dots\dots\dots\dots\dots 2.10$$

In that $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$ $\tilde{A}$ can similarly be expressed as equation 2.11

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \frac{\mu_{\tilde{A}}(x,u)}{(x,u)} \quad J_x \subseteq [0,1] \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots 2.11$$

where $\int\int$ represent the union of all allowable x and u

At each value of x say $x = x_1$, the 2D plane with axes u and $\mu_{\tilde{A}}(x_1, u)$ is referred to as a vertical slice of $\mu_{\tilde{A}}(x, u)$. It is $\mu_{\tilde{A}}(x = x_1, u)$ for $x_1 \in X$ and $\forall u \subseteq J_{x_1} \subseteq [0,1]$

eqn.2.12,

$$\mu_{\tilde{A}}(x = x_1, u) \equiv \mu_{\tilde{A}}(x_1) = \int_{u \in J_{x_1}} \frac{f_{x_1}(u)}{u} \quad J_{x_1} \subseteq [0,1] \ldots\ldots\ldots\ldots 2.12$$

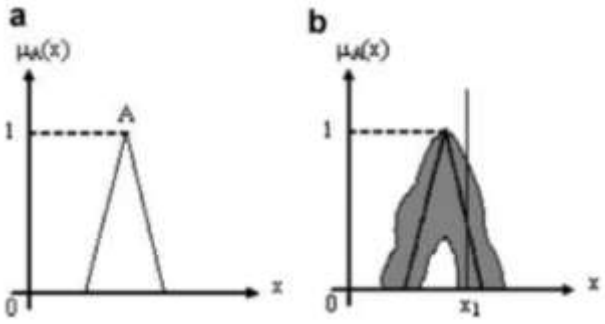


Figure 2. 8 (a) type-1 MF. (b) Blurred type-1 MF.

Where $0 \leq f_{x_1}(u) \leq 1$. Since $x_1 \in X$,1 is dropped on $\mu_{\tilde{A}}(x_1)$ and it is referred to as a secondary membership function denoted by $\mu_{\tilde{A}}(x)$; it can also be referred to as a secondary set because it is a type-1 fuzzy set.

Using eqn.2.12, a type-2 fuzzy set could be represented as the union of all secondary sets and $\tilde{A}$, can be restated in a vertical-slice manner as eqn.2.13.

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | \forall x \in X\} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.13$$

Or

$$\tilde{A} = \int_{x \in X} \mu_{\tilde{A}}(x)/x$$

Which is the same as $\int_{x \in X} \frac{\int_{u \in J_x} f_x(u)/u}{x}$ $J_x \in [0,1]$ 2.14

The primary membership of the x domain is known as the secondary membership function.

In eqn 2.14 , J_x is the primary membership of x, with $J_x \subseteq [0,1]$.

The secondary grade is the amplitude of a secondary membership function. In eqn.2.10, $\mu_{\tilde{A}}(x_1, u_1)(x_1 \in X, u_1 \in J_{x_1})$ is a secondary grade; in eqn.2.14, $f_x(u)$ is a secondary.

When $f_x(u) = 1 , \forall u \in J_x \subseteq [0,1]$, secondary membership functions are interval sets, an interval type-2 membership function is obtained if $\forall x \in X$,. An interval secondary membership function reflects constant uncertainty at x's primary memberships.

Type-2 fuzzy concept set was first presented by Zadeh. Later (Karnik & Mendel, 1998) designed and deliberated on some vital properties of type-2 fuzzy sets but very little work was presented to further implement it into a suitable and practical methodology. Mendel stated that “Type-1 Fuzzy Logic Systems (FLSs) are unable to directly handle rule uncertainties” this is due to using type-1 fuzzy sets that are not certain (Mendel, 2001. A modest and forthright treatment of type-2 fuzzy sets was

explained by (Enyinna et al., 2015) (D. Wu, 2013). The three-dimensional structure of type-2 fuzzy sets proposes that uncertainties would be accommodated fully in comparison type-1 that is a two-dimensional fuzzy sets.Type-2 fuzzy logic controllers have been put in practice in various application summarizes in the study (Mittal et al., 2020)

Type-2 fuzzy set are made of $((x,u),\mu_{\tilde{A}}(x,u))$ in that for every member of domain $x \in X$ there exixts a primary membership value, $u \in J_x$ (J_X = range of primary membership for a given x) and the secondary membership,$\mu_{\tilde{A}}(x,u)$. Symbolically a type-2 set can be well illustrated as eqn.2.15.

$$\tilde{A} = \{((x,u),\mu_{\tilde{A}}(x,u))I\ \forall x \in X, \forall u \in J_x \subseteq [0,1), \mu_{\tilde{A}}(x,u) \in [0,1]\}$$

$$= \sum_{u \in J_x} \sum_{x \in X} ((x,u),\mu_{\tilde{A}}(x,u)) \dots\dots\dots\dots\dots\dots 2.15$$

Using type-2 FLC is typically computationally demanding and thus costly. To alleviate this burden, most researchers employ interval type-2 FLC. It significantly reduces computation costs while retaining the major benefits of type-2 FLSs (Baklouti et al., 2020).

In eqn.2.16, an interval type-2 set can be defined as a fuzzy set whose secondary membership values are always unity.

$$\tilde{A} = \{((x,u),1|\forall x \in X, \forall u \in J_x \subseteq [0,1]\} = \sum_{u \in J_x} \sum_{x \in X} ((x,u),1) \dots\dots 2.16$$

Type-2 general schema has five components, namely type-reducer, fuzzifier,defuzzifier, rule base and inference engine,. The process begins with a fuzzifier plotting crisp values to type-2 fuzzy sets. Inference engine then calculates rules executing logical combinations between the inputs (antecedents) and outputs (consequences) of the type-2 fuzzy set which results to a type-2 fuzzy output. Subsequent step put together all type-2 fuzzy outputs and converts them to type-1 fuzzy, this unit is known as the type reducer (Birkin & Garibaldi, 2009). Lastly, the concentrated set is input to a defuzzifier for generation of crisp output that can be used to drive an actuator in a control system. Defuzzification stage and type reduction is represented as the output processing unit. Finally, the interval reduced by the defuzzifier block gives a crisp real number (Ruiz-Garcia et al., 2019) as shown in figure 2.10.

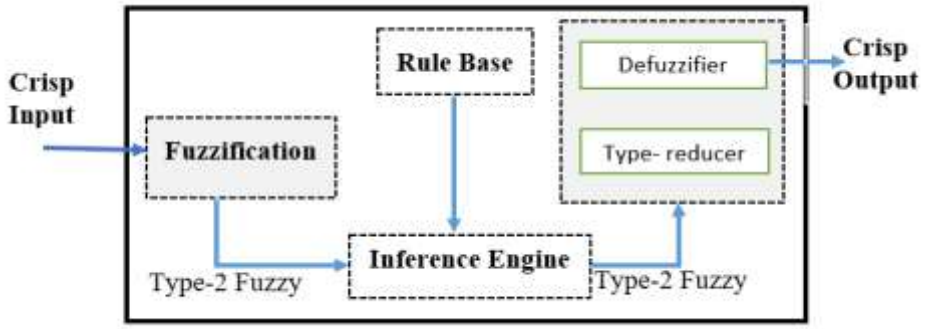


Figure 2. 9 IT2-FIS architecture.

Handayani et al., explained in details steering problems of robot when using sensors as the uncertainty in the environments. This is usually due to a large number of inaccuracies in the readings obtained in a real-world environment that must be addressed (Handayani et al., 2017).The design of the robot control system needs to handle uncertainties, because a significant imprecision values and uncertainty have a noteworthy affect on the application of robotic systems. In robotic apparent uncertainties include: errors due to changing environments, sensor measurements, approximation of sensors and actuators features (Potena et al., 2019).

Type-1 fuzzy logic can not treat proficiently perceived vagueness in autonomous mobile robot navigation. However, type-2 fuzzy MF can give an effective image of

variables as echoed by numerous researchers (Baklouti et al., 2020; Ontiveros-Robles et al., 2018).

A Robotic system can be developed to navigate in unknown environment with a lot of uncertainty using Type-2 Fuzzy Logic Systems (T2FLS) which is currently gaining recognition in literature (Naik & Gupta, 2017) . Interval Type-2 Fuzzy Logic Systems (IT2 FLS) has frequently been used to reduced complexity representation of T2 FLS and becoming very handy in most applications (Baklouti et al., 2020).

The current fuzzy logic controllers are Type-2 fuzzy logic systems (T2FLS).The use of extra dimension of MF gives a better decision-making flexibilities and good representation of uncertainty than T1MF particularly in robotic field as proposed (Baklouti et al., 2020; Chao et al., 2020; T. Zhao et al., 2020).

Researchers in this study (Cherroun et al., 2019) used Sugeno type-2 fuzzy logic inference to develop an obstacle-avoidance, goal seeking and wall-following robot. They proposed an intelligent control grounded on type-2 fuzzy logic technique for autonomous robot navigation.

Numerous Authors have compared effectiveness of type-1 and type2 fuzzy systems in several applications (Baklouti et al., 2020; Mendel et al., 2020; Naik & Gupta, 2017; Ontiveros-Robles & Melin, 2020).They propose that in case of increased uncertainties, T2FLC can be introduced to improve efficiency.

Type-1 fuzzy sets with membership degree in the interval are shown in Figure 2.11. [0,1][1,0], From fuzzy theory type-2 fuzzy sets are well-defined by two dimensional MFs (Functions & Raj, 2018).

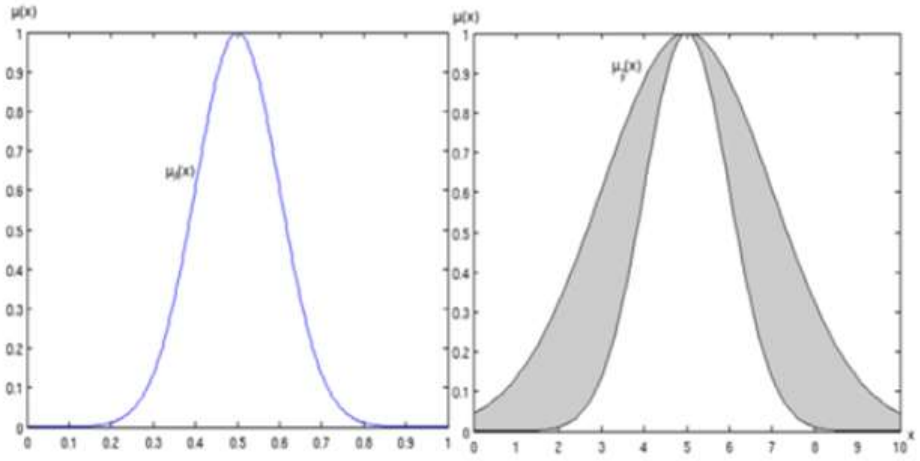


Figure 2. 10 Type-1 and type-2 MFs.

2.4 Empirical evidence of fuzzy logic-controlled robots.

Several publications have appeared in recent years documenting navigational approaches for autonomous robots using various artificial intelligence techniques. However, researchers have applauded fuzzy logic algorithm as a novel technique.

Research work (Singh, R. & Bera, T.K. 2020), “Jansen Walking Models for a Quadruped robotic mechanism” made use of fuzzy logic model to develop a fuzzy autonomous for difficult task of obstacle avoidance. The proposed controller had two inputs and one output. The input was the left and right sensors distance from obstacles, while the rotation angle was the output. The specification of the fuzzy logic controllers with graphical user interface was provided with a total of 36 rule to accomplished its goals.

In an unknown environment, the authors (Alatise & Hancke, 2020; C. Chiu, 2015) proposed an intelligent fuzzy obstacle avoidance technique for mobile wheel robots. In comparison to the conventional model-based approach, the fuzzy logic control system proposed imitated the driving intellect of man and had the benefit of easy implementation. The synchronized sensors minimized the complexity of the fuzzy logic controller. The simulation results indicated that the technique could be used to prevent robot collision with the obstacle. In the future work they proposed a

control method which can prevent dead zones and improve control precision using ultrasonic sensors.

In this study, an integrated IT2 FLS that models and controls simultaneously a mobile two-wheeled inverted pendulum (MTWIP) was presented, its effectiveness was tested practically by real experiments. Results exposed IT2 FLS as able to handle modelling ambiguity than type-1 counterpart (Baklouti et al., 2020).

An arduino microcontroller board was used in the investigation of a fuzzy logic approach for obstacle avoidance. (Yerubandi et al., 2015). The authors Utilized mobile platform and ultrasonic sensors mounted on a robot chassis, actuation was done by DC motors. The robot was able to react to the environment accordingly aided by gathered distance information. The same methodology was demonstrated (Abdulkareem et al., 2019), however in their study there was still considerable ambiguity where ultrasonic sensors failed to detect some objects with spherical and cylindrical shapes. The sonar sensor beam struck the surface with oblique incidence and reflected away echo instead of reverberation thus no detection. Subsequently, similar work was done using weighted average method with the assumption that the position of the target and obstacles was known throughout the traverse labyrinth environment (Batti et al., 2019)

"Generation near-optimal path" was proposed (Al-dahhan & Al-dahhan, n.d.). To pursue the anticipated target a 28- fuzzy rules controller was developed. Ultrasonic sensors were installed to sense obstacles for the robot to perform the right action. Three infra-red sensors were mounted to detect small nearby obstacles. Analysis showed algorithm working competently in real environment. They deliberate in future to concentrate on improving the proposed algorithm by including camera to help in noticing all kinds of obstacles.

Research in this study (Handayani et al., 2019) shows that IT2FLS and T1FLS are not the same in term of performance. T1FLS simpler mathematical representation to make the robot process data quicker than it counterpart IT2FLS that has a type-reducer to make decisions. The complexity and number of obstacles in the environment influence robot behaviour when avoiding obstacles. It avoids obstacles that block its way to reaches the target. The more obstacles we have in the route the slower the robot will find the right path. In experimenting with IT2FLS results shows

robot moving more slowly due to inclusion type-reducer than T1FLS. Nevertheless, they concluded that IT2FLS can move smoothly and avoid obstacles with a lot of efficacy.

In research on an autonomous mobile robot in a partially known environment, uncertainty caused by inaccuracy of sensors were reflected in the design. The rules of the controller ought to be adjusted correctly to get better performance (R. Zhao et al., 2015) these studies provide important insights. Al-Dahhan & Ali, (2016) used fuzzy logic to make a controller for safe robot steering. The controller required information about the robot features and performance to form its rule base that was stimulated by humanoid capability in such a scenario. A 153-fuzzy rule controller was used for path following problems, while the additional fuzzy controller aided the robot to avoid obstacles. Experimenting with MATLAB as a tool for simulation the proposed fuzzy controllers were evaluated. Results revealed the idea as noble although the system required a large memory to implement the proposed rules.

In a major advance in 2020, Rendyansyah et al. presented omnidirectional wheeled Robot able to move competently in all directions (Rendyansyah, Nurmaini, Sari, & Ramarta, 2020). The robot moved freely to the target point by using Omni wheels. The direction of motion used kinematic methodology aided by a fuzzy logic. The controller accepted five inputs from proximity sensors. The output resulted to the effective movement of the wheeled robot. The controller of the robot was made to work in two ways, to avoid obstacles and approach the target. When the sensors detected objects, the obstacle avoiding behavior was triggered, whereas the target approaching behavior was based on the error value of the actual position and reference. The experimental results demonstrated that the mechanical device navigated efficiently, avoiding obstacles on its way to the specified target.

In the development of line following and efficient obstacle avoidance robot in a static and dynamic environment, a system was described. Various sensor and actuators were mounted on the robot for detecting the surroundings and making verdicts consequently, heterogeneity of sensors provided not the same accuracy and features. They suggested using multiple sensors mounted to optimize the design. Complementary, redundant and cooperative data fusion of the sensor was proposed

to improve efficiency and robustness. E-puck robot model was used with several sensors for monitoring the surrounding environment. WEBOT simulator realized the model design and program environment of the robot (Elleithy et al., 2016)

Path planning module based on fuzzy logic for autonomous motion control was presented for reactive navigation (Abdessemed et al., 2014). The authors used stereo vision-based modules to be able to maneuver complex unknown environments. They used behavioral control, and each local navigational task was studied in terms of primitive performance. Using systematic style some fuzzy rules were omitted in the critical situation in which the vision camera could intercede to unlock the mobile robot. The design control laws were experimentally tested with success but had problems with inertia effect and they were focused on making outdoor navigation in unstructured and unknown terrain.

A study to develop a pioneer p3dx robot simulated by V-rep and MATLAB using fuzzy Logic controller for was proposed (Batti et al., 2019). The controller was used to navigate a robot with very few sensors applying fuzzy rules. TSK inference engine was selected over Mandani-type because of the computational effectiveness and stability. They claimed that Mamdani controller was not flexible compared to Sugeno-type. The effectiveness of the developed algorithm was verified through simulation. The robot was able to reach the target with three sensors and 27 rules but lacked consistency.

Research in guiding an autonomous robot in space of obstacles an improved path planning method that involved image processing, path scheduling, cluster reduction, and smoothing was employed. The fuzzy inference was adopted to get collision-free shortest distance conditions. Robot performances and roles were made to complement one another for cooperative behavior and provided various smoothing techniques to manipulate the rough planned path. MATLAB simulation demonstrated the technique as successful and novel for obstacle avoidance robot (Su & Phan, 2014).

An obstacle avoidance robot was presented that utilized a fuzzy logic controller. In the study, FLC was constructed using sensorial data to control speed and turning

angle. The result showed that the fuzzy logic method was used to find the safest route to the targeted point by the mobile robot. The workspace environment was simulated by WEBOT Pro software and surroundings consisted of static obstacles (Almasri et al., 2015).

A fuzzy logic-based motion planner was presented (Khaksar et al., 2019) which was used to control a nonholonomic mobile robot in unknown environments. The fuzzy planner used the readings of the robot's sensory system and calculated three fuzzy variables hoping to move the robot closer to the final position and away from the surrounding obstacles. The outputs of this fuzzy system were the angular and linear velocities used to adjust the motion, speed and direction. Several experiments were carried out in order to evaluate the proposed planner's performance in various typical navigation problems. The planner generates safe and stable results with low computational. Implementing the proposed work on a TurtleBot shows the applicability of the planner in real implementations. The work can be further improved by implementing more deterministic tools for designing the elements of the fuzzy controller. They planned to improve the fuzzy membership functions and fuzzy rules in the future.

Takagi-Sugeno Type-1 and Type-2 fuzzy logic for autonomous robot control were proposedThey proposed fuzzy model was based on users intellect and knowledge about the robot. The developed systems provided the robot with intellect and capability of navigating freely. Simulated outcomes validated competence of the controllers for steering the robot. The controllers performed equally most cases however increasing uncertainties, T2FLC was better than T1FLC in terms of precision. In their future work they wish to adjust fuzzy parameters to improve both controllers (Cherroun et al., 2019)

The authors (Batti et al., 2019) presented "Autonomous smart robot for path predicting and finding in maze based on neuro-Fuzzy". They carefully investigated advanced types of models in controlling a non-holonomic mobile robot to navigate in areas set disorderly with static objects. In their research, trapezoidal MFs were used to create a fuzzy logic system. To improve the results, adaptive neuro fuzzy inference

system controller was used. V-REP and MATLAB software were used to simulate feasibility and effectiveness of the proposed model. A comparative evaluation was done based on the speed of the robot. Simulated results exposed the mobile robot capable to navigate positively in maze with but ANFIS gave better results in scene having many obstacles.

A study of a hybrid approach based on multisensory information by fuzzy technique for navigation of autonomous mobile robot to operate in real-time with imprecise knowledge was presented. The mobile robot sends sensing data via the Arduino board to control a computer over wireless communication unit. The navigation improvements by tuning fuzzy knowledge especially in extreme obstacle scenarios was demonstrated (Lin et al., 2015).

For real-time control methodology of autonomous robot employing fuzzy logic in static and dynamic obstacles was discussed. They critiqued mathematical methods that require a complex algorithm and a large memory (Omrane et al., 2016).

Researchers emphasized on implementation of human perception in the mobile robot using fuzzy logic for collision avoidance (Oleiwi et al., 2015; Wahid et al., 2017). In the industrial application, it was projected to address the problem of path motion planning in a highly dynamic environment. On these grounds of the human mind and its intelligent introducing perceptual judgment, decision making to robots and giving them intelligence to overcome complex hurdles. They explained the need for establishing a high-efficiency rule set to avoid collision.

2.5 Rule base reduction methods

Rule base is basically a matrix that govern controller output based on the inputs thus holds the input/output relationships. It provides a structure to translate the human information into fuzzy “IF…THEN rules. Readability and comprehensibility should be considered when considering interpretability of a fuzzy logic model as reported by (Bartczuk et al., 2016)

Rule base reduction is a method to try simplify a fuzzy system. In this study (Tan et al., 2019) classification methods of rule base simplification are presents in five categories.

2.5.1 Feature reduction

The group comprise of techniques which depend on feature reduction using selection or transformation (feature extraction). Transformation entails creating more features from the given set, or adding different features in the place of old one. This method makes interpretation a bit hard, eventually giving rise to semantics loss. The Feature transformation does not really change the underlying features meaning this shortcoming doesn't affect our controller since the selection of subset is done on dominant features, and reject features that are disturbed by noise or don't add meaning to accuracy of FIS (Angelov & Gu, 2017).

2.5.2 Similarity-based simplification

In this Methods merging of similar rules of the class is performed and redundancy eliminated in FIS. The similarity merging methods accomplish integration of fuzzy sets that represents similar concepts by employing some form of similarity measure (Tsekouras, 2016). When there is high redundancy in the model, this process may lead to some rules looking alike thus marrying them hence lessening the size of rule base. "Compatible cluster merging algorithms" merges similar groups of rules into one so that FIS rule base is reduced. To finish a means of consistency and inactivity inspection is used to reduce the rules further (D. Wu & Mendel, 2019) Consistency checking particularly reduces regulation base by removing inconsistency whereas inactivity eliminates rules having little firing strength based on a determined threshold.

2.5.3 Orthogonal transformation

This technique reduces rules by using matrix computations. This is achieved either by considering firing strength matrix and implementing a system of measurement to evaluate the effect of the rule on FIS application or by putting in mind matrix

decompositions and eradicating rules that corresponds to less significant features and updating membership functions consequently (Seines & Babuška, 2001)

2.5.4 Interpolative reasoning

Normally, current fuzzy reasoning techniques require the rule base to cover the entire universe of discourse of antecedents. This method doesn't do well if the input data is within region not covered by universe of discourse. Hence there is no triggering of rules thus no outputs are defined in the rule base. In this study (F. Li et al., 2018) first fuzzy interpolative reasoning methodology was presented. By approximation the method generated conclusions on FIS with light a rule base. In situation of FIS generalization, interpolation can be used to minimize rule base. This can be accomplished by removing rules which can be estimated by interpolation of adjacent rules (Long et al., 2019).

2.5.5 Hierarchical reasoning

To achieve a hierarchical fuzzy system, approaches implementing a hierarchical reasoning technique rearrange fuzzy rule base structure. Numerous low-dimensional FIS are linked in a well-defined hierarchy to form hierarchical systems. (Mutlu et al., 2018) This approach was used by (Liang et al., 2019) ,authors recommended a hierarchical fuzzy system "automatic control of Unmanned Quadrotor Transportation Systems" that worked very well with minute rule base.

2.5.5 Ultrasonic sensor

The ultrasonic rangefinders work on the principle of measuring the time it takes for a signal to travel from the transmitter to the receiver. The sensor is made up of a transmitter that generates ultrasonic waves, a receiver that detects the echo, and auxiliary nodes that allow the module to function normally.

Sound waves with a frequency of 40 kHz are produced by the rangefinder. The sound waves reflect from the object and return to the receiver, and the sensor provides information about the time required for sound waves to propagate from the sensor to the object and back as illustrated in figure 2.11.

Practical measurements (Batti et al., 2019) show that the ultrasonic sensor measures distance with high precision. When measuring distances up to 3m, the accuracy class was 0.5%, and when measuring distances up to 6m, the accuracy class was 0.7%. The main idea is to calculate the time it takes an ultrasonic sound wave to travel from the sensor to the detected object. An ultrasonic transmitter sends a sound frequency of more than 18 kHz through the air at 344 meters per second (at 20°C), and the receiver receives the reflected sound from the object.

The distance between the transmitter and the object can be calculated by calculating the time it takes the ultrasonic wave to travel from the transmitter to the receiver and reflected.

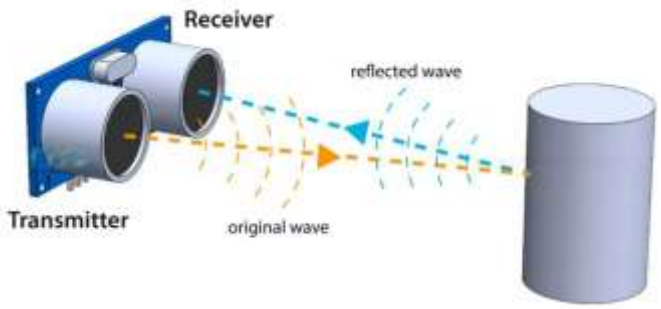


Figure 2. 11 Ultrasonic transceivers

Ultrasonic sensors are a good choice for many purposes since almost all materials reflect sound waves. These sensors stand out from their photoelectric counterparts when it comes to detecting and measuring films, transparent objects, and liquids. Ultrasonic sensors are also unaffected by target color or rapid color shifts. Ultrasonic sensors work well in dusty, unclean situations because they work with sound waves. Small targets on big background do not work effectively.

Objects deflect the wave back before the receiver is operational, therefore ultrasonic sensors have a "dead zone" directly in front of them where they cannot be detected. This is because the transmitter's reverberations compel the receiver to hesitate for a moment before starting to listen for the echo.

2.5.6 Sensory fusion

Sensor fusion is the process of combining data from various sensors to provide a more accurate representation of the scene being studied. The concept is that each sensor has both strengths and weaknesses; the goal is to use the strengths of each and remove any uncertainty in order to generate a precise model of the environment under study. Sensor fusion can be classified into three types. Complementary, Competitive/Redundant, and Collaborative.

The following issues commonly occur when using a physical sensor: Sensor Deprivation occurs when a sensor element fails, resulting in a loss of perception of the target item. A single sensor can only cover a small area due to its limited spatial coverage. Some sensors require a specific amount of setup time before they may perform and transmit a measurement, restricting the maximum measurement frequency. Imprecision: Measurements from individual sensors are limited to the precision of the employed sensing element. Uncertainty, in contrast to imprecision, depends on the object being observed rather than the observing device. Uncertainty arises when features are missing (e. g., occlusions), when the sensor cannot measure all relevant attributes of the percept, or when the observation is ambiguous.

Aguileta et al made effort reorganizing fusion methods into three main families i.e. decision level, feature and data level (Aguileta et al., 2019). Sensor fusion is frequently used to improve navigation, object position, and location precision. Some information may be untrustworthy when using the sensor data sources individually. By fusing the data from several independent sensors, each of which contains unique information improves perception. Sensor fusion can also improve single sensor measurements by reducing sensory deprivation, limited spatial coverage, limited temporal coverage, imprecision, and uncertainties. Sensor fusion can be designed in competitive, complementary, or cooperative modes.

The handbook of Multisensory Data Fusion (Liggins et al., 2009) give more details of works done in this field. Multi-Sensor Image Fusion are organized in combinations of the main groups. They established a systematic comparison of works done by various researchers. After analysis they identified and related

performance of techniques that used one fusion method with those that used two. They identified and compared approaches for manual selection of features and with those for automatically selection. Finally, they highlighted applicable directions and future steps.

In this article (Almasri et al., 2015), “a multisensory fusion collision avoidance mobile robot” was proposed. Eight range sensors were used to detect and avoid obstacles. Three ground sensors were installed with line follower and GPS to attain the position of the robot. Sensors fusion was grounded on fuzzy logic inference system giving eight inputs, two outputs with a total of 24 rules. The fusion model had distance traveled reduced which saved on computational power and time.

2.6 Research gap

Search and rescue missions, manufacturing, military, mining, and transportation are just a few of the applications for mobile robots. In robotics steering, the most important task is to get to the destination while avoiding obstacles along the way. The potential of FLCs equipped with three sensors for obstacle detection and avoidance has been studied. However, some gaps that cause dead zones and make maneuvering difficult. The traditional approach focused on T1FLC with crisp value, which cannot effectively treat perceived uncertainties in some applications. Sensor fusion, fuzzy rule reduction, and the selection of appropriate MFs have received far too little attention, despite the fact that they form a solid foundation upon which to build. Many researchers have compared the performance of type 1 and type 2 fuzzy systems in a wide range of applications. They proposed that when there are more uncertainties, T2FLC, which has received less attention in obstacle avoidance robot navigation, can be used to improve efficiency.

2.7 Summary

A review of several studies conducted on the works of fuzzy logic controllers revealed several previously developed approaches for path planning in mobile robots with a paradigm shift towards development of reactive approaches. Due to robustness and ability to handle uncertainties, fuzzy logic controllers have emerged

as a suitable technique for mobile robots. Mostly Fuzzy Logic models for Obstacle Detection Robots results in dead zone and difficulties in detecting some obstacles for optimum performance. Fuzzy rule base reduction, sensor fusion and Tuning of the fuzzy logic controller for better performance have been given less attention hence identified as gaps to be addressed. This research develops an optimized T1MFLC by sensor fusion data, rules reduction and evaluated effects of controller's Membership Functions for ideal obstacle avoidance robot.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

3.2 Background

Research method is described in this chapter, as well as how we deal with issues that arise during the execution of the experiments. A description of the research design, data collection methods, and the sampling aspect of the study is also included in this section. Initially we developed the environment, constructed the walls, created the obstacles, and modeled the robot to create the simulation field using MATLAB and V-REP software. The input and output variables, as well as the robot's navigation procedure in its environment, were then determined. The methods for combining sensors and reducing controller rules are described. The procedure for changing the membership functions of a type-1 Mamdani fuzzy logic-controller is described, as well as a methodology for comparing the developed and optimized T1MFLC to the T2MFLC. Methods developed are to help answer the research questions. The research process includes all of the steps required to carry out the study successfully.

Figure 3.1 illustrates the methodological guide during the research process: Formulating the research problem, conducting an extensive literature review, developing the hypothesis, preparing the research design, determining sample design, collecting data, analyzing data, testing the hypothesis, generalizations, interpretating, and finally presenting the results.

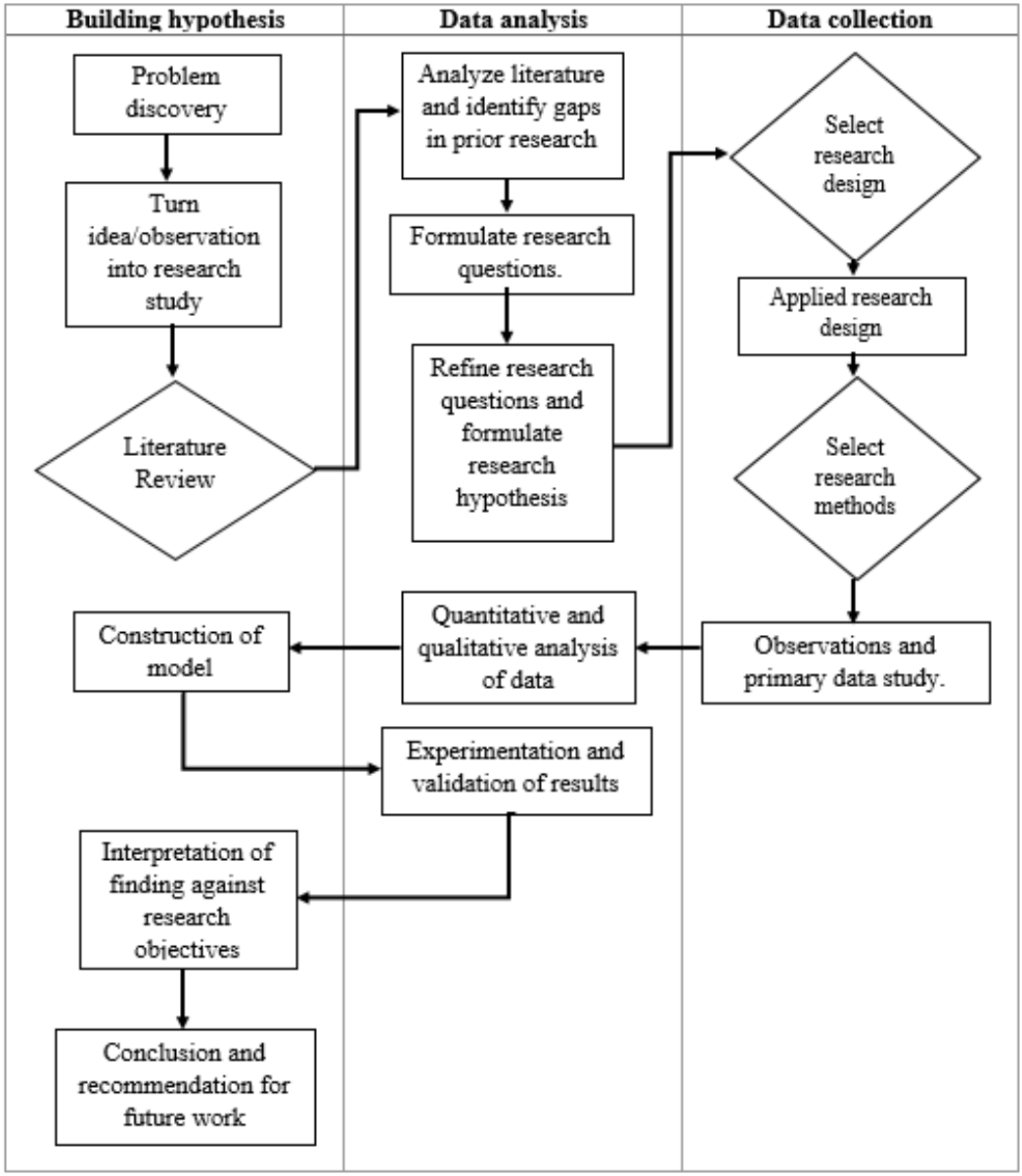


Figure 3.1: Research method.

3.3 Research method

The navigation of a two-wheeled obstacle avoidance mobile robot in a complex setting was modeled. Using various tuning techniques, the fuzzy inference system was investigated for the possibility of reducing uncertainty. A fuzzy logic model (T1MFLC) with fewer rules was created. The fuzzy controller's membership functions were changed to improve performance. The model was an improvement over the traditional 27-rule model (Batti et al., 2019; Bayar et al., 2014; C. S. Chiu et al., 2016; Pandey et al., 2014). Finally, the developed T1MFLC was replaced with

a corresponding T2MFLC, and the results were compared. We analyzed data collected from proximity sensors as input variables providing distance information for object detection and path planning. Two output variables (actuators) were used to power motors. An experimental research approach was used to validate the efficacy of the proposed technique through simulation (V-REP and MATLAB). The time taken by the robot to move from the start point to the target was the data collected in this study. Figure 3.2 depicts the activities involved in modeling the world and robots, as well as the development of fuzzy logic rules and data collection.

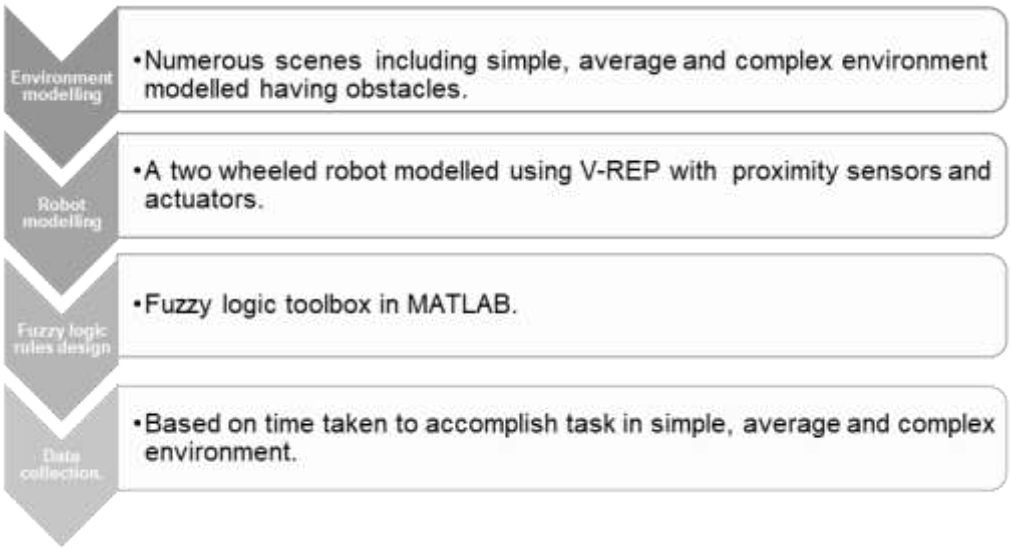


Figure 3. 2: Development methodology

3.4 Research design

An applied research design was adopted due to the practical nature of this study. It entailed collecting data on an obstacle avoidance mobile robot in a simulated environment. Using the information to determine techniques that work and do not optimize the Mamdani fuzzy logic obstacle avoidance robot model. The main goal of the experiments was to demonstrate the reliability of using fuzzy logic to navigate a mobile obstacles avoidance robot, simulation was ideal. Using a simulation research approach, the applied research design method enabled the collection of the necessary information, analysis, and development of a novel robot navigation model.

3.5 Sampling techniques

Sampling technique, which is an important component of any piece of research, determines the quality of the results and findings. Sampling techniques are classified into two types: probability and non-probability. Probability sampling enables the researcher to draw broad conclusions from a small sample of the target population. Non-probability sampling is a sampling method in which the researcher chooses samples based on subjective judgment rather than random selection.

Purposive sampling is non-probability sampling that does not involve random selection was used; the results were not used to characterize the wider population. The selection was what we regarded as representative sampling units. There was no statistical representative sample. The sample size selected served the purpose of the researcher.

Expert sampling type of purposive sampling technique where we drew a sample from experts in the field of autonomous robot navigation. It was used because the research needs to glean knowledge from individuals that have particular expertise (Kashyap & Parhi, 2022). Expert sampling was particularly useful due to lack of empirical evidence in this area, high levels of uncertainty, and it would take a long time before the findings from the research could be uncovered. In the following studies (Adam et al., 2021; Ahmad Fauzi et al., 2021; Haddi & Kharchaf, 2021; Kashyap & Parhi, 2022) 14 runs were used to validate the performance of the mobile robot which we build on.

3.6 Data collection tools

Data collection is a systematic process of gathering observations or measurements. The collection of data allowed us to gain first-hand knowledge and unique insights into the research problem. Primary data is information gathered from firsthand experience. An experiment is a type of data collection method in which we change some variables and observe how they affect other variables. The variables that we manipulate are independent, whereas the variables that change as a result of the

manipulation are dependent. In a simulated field, we measured the time it takes a robot to travel from start position to destination in seconds.

A simulation was used to replicate the process conditions using a computer model. Because the actual system would be poorly constructed and expensive, investigations were conducted on the model rather than the actual system. We ran the experiment for obstacle avoidance robots by timing the process while purposefully changing the model, and perceptions of robot behavior were recorded. In this sense, simulation served as a framework testing strategy. We developed a model of the phenomenon under investigation. The model was run several times with different variations, and the results were observed.

Running the simulation and determining whether or not the model worked as part of model confirmation. Data was collected by identifying five analysis components: the underlying conditions, the time structure, result estimation, the number of iterations, and any variation in model boundaries or initial conditions. The variations allowed for the testing of different hypotheses to answer the research questions, as well as the model's sensitivity to parameter changes.

3.7 Environment modeling

Simulation was a perfect way to test the proposed technique before performing research with a real robot because it was less expensive and simpler to set up. In this study, the Virtual Robot Experimentation Platform (V-REP) was used. (Spica et al., 2016). It's a distributed control development environment where each object/model can be controlled independently with the help of an embedded script, a plugin, a ROS node, a remote API client, or a custom solution. The V-REP simulator was used to create a navigation environment for robots (Rohmer et al., 2013). To model the area, the scene tree was used to monitor the size of the space, time phase, real-time, lighting, and other settings required for the mobile robot. To construct walls and obstacles, a 'Solid' node and a'shape' node were added to the scene tree. Appearance and Geometry nodes appeared, and corresponding nodes were added to build the outline of the wall, which varied in size and color. The basic layout of the environment consisted of four walls. Several models were included in a scene. Both

scenes and models contained Objects, Collections, Collision objects, Distance objects, Inverse kinematics classes, Geometric constraint solver objects and Custom user interfaces (V-Rep, 2012). A model was recognizable sub-element of a scene.

The steps for creating a new scene were using the menu bar and the default scene included the following elements: Various camera and light objects: The scene was barely visible without a light. Several views were associated with and displayed a camera. Several pages with one or more views. The environment was made up of ambient light, fog, background color, and other properties. The floor was made up of objects arranged in a model. we didn't need to use child scripts because the default main script allowed us to run basic simulations. If a child script was later copied into the scene and linked to a scene entity, it was immediately executed (called by the main script).

The environment specified the properties and parameters of a scene in V-REP. The environment properties and parameters were not saved when a model was saved, but only when a scene was saved. The environment dialog could be accessed via the Menu bar, Tools then Environment. An environment also specified background colors, fog parameters, ambient light, scene creation detail, and other additional settings.

3.7.1 Creating the walls

We Selected a solid node in the scene using on add new button. The shape was made dynamic and respondable and pure shape. In the properties, window color was adjusted and the objects were made respondable. Object special properties were given in the scene object properties that included collidable, measurable, detectable, renderable and selectable (V-Rep, 2012).

3.7.2 Creating the obstacles

We Selected the solid node in the scene and clicked on *Add New* button. Just as the wall obstacles were given special properties and made dynamic and respondable (V-Rep, 2012). In the environment twelve cylindrical obstacles were placed in the field

as shown in figure 3.3. The robot was expected to avoid the obstacle and navigate to the target point.

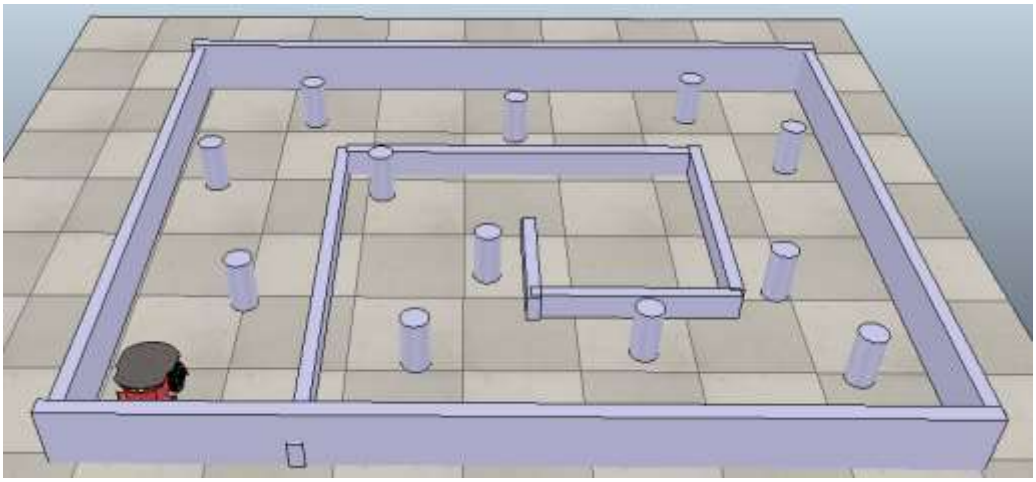


Figure 3.3: Robot navigation field

3.7.3 Robot model

As shown in fig. 3.4, the robot model used in this study was a unicycle-type mobile robot. It consisted of two separate driving wheels attached to the same chassis axis, as well as a third freewheel (castor wheel) for balancing the robot when in motion. Two actuators drove the wheels independently to achieve motion and orientation. (V-Rep, 2012).

Figure 3.4: Pioneer P3dx.

3.7.4 Kinematic model of the nonholonomic robot

Our experimental model was inspired by the study (Handayani et al., 2017; Yekinni

& Dan-Isa, 2019) proposed. The non-holonomic differential drive two-wheeled mobile robot's kinematic and dynamic model as depicted in Figure 3.7. The term "non-holonomic" refers to the robot's inability to move sideways and its reliance on the rolling-wheel principle. The robot had two driving wheels in the front and a caster wheel in the back to carry the chassis. The wheels were driven by two different motors, which controlled the robot's movement and direction.

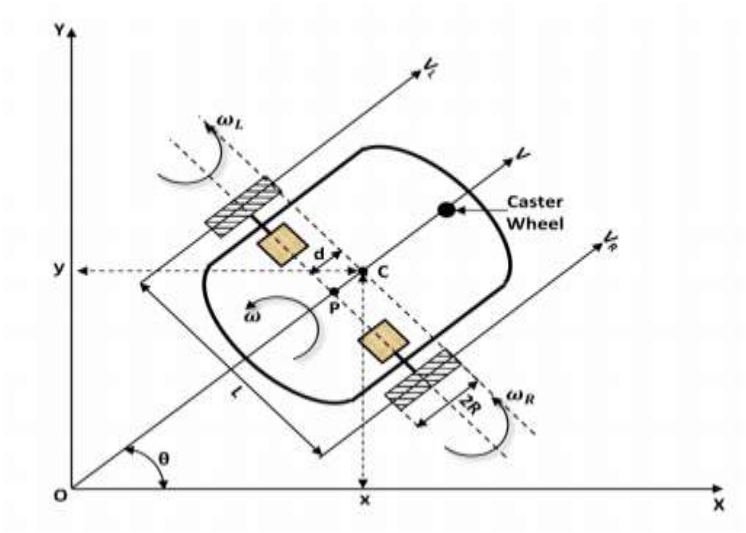


Figure 3.5: Kinematic and dynamic model (Nurmaini & Chusniah, 2017).

$$q = [x, t, \theta]T \qquad 3.1$$

The driving wheels of the mobile robot adhered to a nonholonomic constraint, which meant they rolled without slipping. Equation 3.1 represents the nonholonomic constraint.

$$ycos\theta - xsin\,\theta = 0 \qquad 3.2$$

The relationship between the wheels' linear and angular velocity is described by equations 3.2-3.7.

$$V = \omega . R \qquad 3.3$$

$$V_r = \omega_r . R \qquad 3.4$$

$$V_l = \omega_l . R \qquad 3.5$$

$$\omega = \frac{V_r + V_l}{L} \qquad 3.6$$

$$V = \frac{V_r + V_l}{2} \qquad 3.7$$

Combining equations (3.6) and (3.7) yields the following result.

$$\begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} V_r \\ Vl \end{bmatrix} \qquad 3.8$$

The equation that follows computes the mobile robot's moving axis (x,y) speed and turning angle over time (t).

$$\frac{dx}{dt} = \dot{x} = V . cos\theta \qquad 3.9$$

$$\frac{dy}{dt} = \dot{y} = V . sin\theta \qquad 3.10$$

$$\frac{d}{dt} = \dot{\theta} = \omega \qquad 3.11$$

As a result, the two-wheeled mobile robot's kinematic equations 3.12 can be

obtained.

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \qquad 3.12$$

Using equations (3.6) and (3.7) to update equations (3.9), (3.10), and (3.11) we develop

$$\dot{x} = \frac{R}{2} (\omega r + \omega l).cos\theta \qquad 3.13$$

$$\dot{y} = \frac{R}{2} (\omega r + \omega l).sin\theta \qquad 3.14$$

$$\dot{\theta} = \frac{R}{2} (\omega r + \omega l) \qquad 3.15$$

The mobile robot kinematic model is represented by a combination of equations (3.13), (3.14), and (3.15).

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{R}{2} cos\theta & \frac{R}{2} cos\theta \\ \frac{R}{2} sin\theta & \frac{R}{2} sin\theta \\ \frac{R}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \qquad 3.16$$

The formulated equation 3.17 was used to model the robot's motion. More information can be found in the study (Nurmaini & Chusniah, 2017) that provides the formal solution of carrying out our experiments.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{cos\theta}{2} & \frac{cos\theta}{2} \\ \frac{sin\theta}{2} & \frac{sin\theta}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} V_r \\ V_l \end{bmatrix} \qquad 3.17$$

Here Vr and Vl are the right and left wheels' linear velocities, that are used as motion commands for mobile robot navigation. The right and left wheels' angular velocities are respectively ω_R and ω_L . The V and ω denote the mobile robot's Centre linear velocity and Centre angular velocity, respectively.

3.7.5 Robot navigation

The mobile robot had two wheels powered separately by DC pulse width modulation (PWM) motors and a free wheel for stability. The sensors measured the front, left, and right obstacle distances, abbreviated as **disf, disl,** and **disr**. Relevant fuzzy control rules are activated based on the information acquired by the sensors at the input. Fuzzy logic operations combine the outputs of activated rules to control the velocities and steering angles of the robot's driving wheels. These were denoted by the left and right motors as velocity of the left wheel (Vl) and velocity of the right wheel (Vr) of the robot, respectively. Figure 3.6 describes a flowchart on how the autonomous mobile robot navigated.

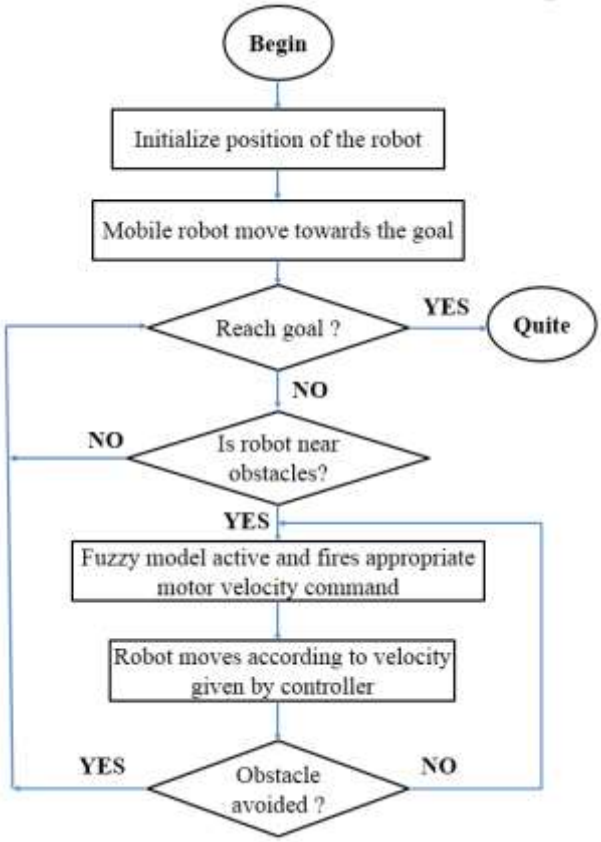


Figure 3.6: Mobile robot navigation flowchart.

During the navigation process, the mobile robot must align obstacle detection with motion planning. It was expected that autonomous obstacle avoidance would be realized if the robot responded appropriately and achieved a collision-free path based on data perceived by sensors in the surrounding environment. This study took into account a fuzzy controller that simulated human driving intelligence. The obstacle avoidance control technique bears a strong resemblance in the study (C. S. Chiu et al., 2016) represented in Fig. 3.7 showing the conceptual framework. During the obstacle avoidance process, a fuzzy controller generated a command signal (i.e., velocity of two drive axle Vl and Vr) for navigation achieved by transducers mounted on the robot chassis. The robot's movement was controlled to avoid collisions by an intelligent fuzzy controller.

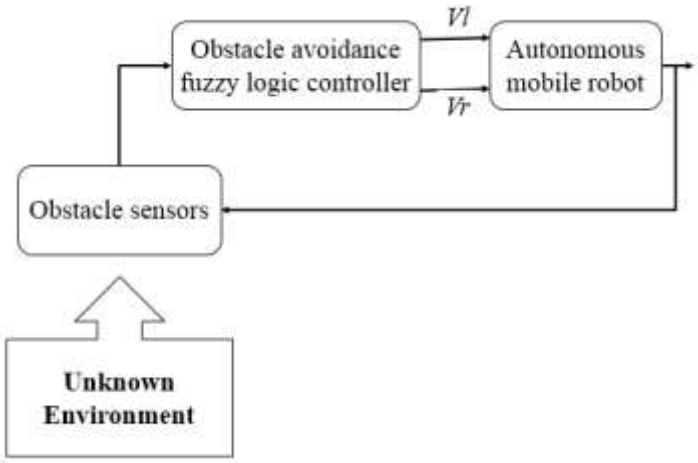


Figure 3.7: Conceptual framework

The following conditions illustrated in figure 3.8-3.10 shows how the mobile robot decided on motion direction in the environment. The robot only moved in straight line when the left and right velocities were equal.

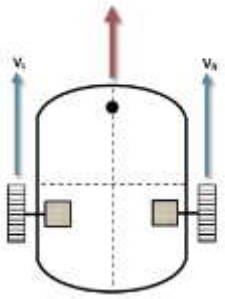


Figure 3.8: Robot moves in a straight line. (VL equals VR)

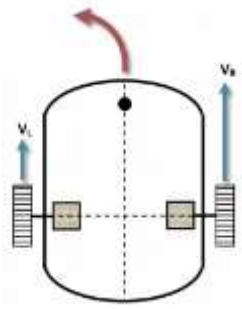


Figure 3.9: Robot turns left side ($V_L < V_R$)

Figure 3.9 shows that the robot turns to the left side when the left velocity is less than the right velocity.

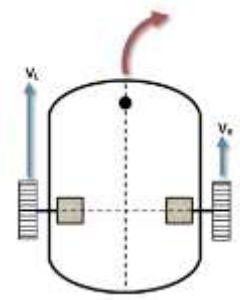


Figure 3.10: Robot turns right side ($V_L > V_R$.)

As shown in fig.3.10, a right turn was achieved when the right velocity was greater than the left velocity.

3.7.6 Fuzzy logic algorithm

The ultrasonic sensor circuit measures the distance crisp values between the robot and the obstacles, which were then used to build the fuzzy membership function. The data from the sensors indicated whether or not there were any obstacles in the robot's path. When a robot was close to an obstacle, it changed its velocity and steering angle to avoid it.

The fuzzy logic algorithm followed the steps outlined below (Spolaor, 2019)

1. Definitions of linguistic variables and terms.

2. Membership functions construction.
3. The development of the rule base.
4. Fuzzification (conversion of crisp input data to fuzzy values using membership functions).
5. Evaluation of rules in the rule base (inference)
6. Combination of each rule's outcomes (inference)
7. Converting the output data to non-fuzzy values (defuzzification)

As shown in Figure 3.11, the Mamdani T1FLC is divided into four sections. The fuzzifier performs input variable (input signal) measurements, scale mapping, and fuzzification. Fuzzification refers to the transformation of calculated signals (crisp input quantities) into fuzzy quantities (also referred to as linguistic variables). MFs are used to complete the conversion. The membership function, which has a value between 0 and 1, determined the magnitude of belonging to a fuzzy set. If it is certain that the quantity belongs to the fuzzy set, its value is 1, but if it is certain that it does not belong to this set, its value is 0.(Melek, 2018).

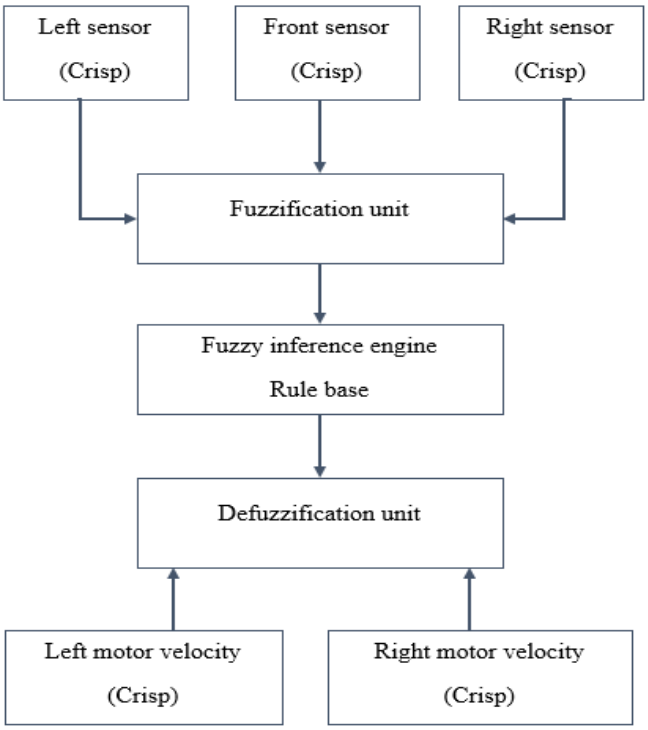


Figure 3.11: Mamdani T1FLC architecture (Melek, 2018).

Rules were developed for application and control based on the experience and expertise of a competent driver. The FLC's primary goals were to eliminate significant errors in the process that had been acquired with the assistance of an expert. Appropriate control performance changes ensured a smooth control action close to the reference value and kept process output within the specified limits.

The inference engine functioned as the brain of the FLC, mimicking human decision-making. This was based on fuzzy principles and inferring fuzzy control behavior utilizing fuzzy consequences and fuzzy logic rules. When controlled input parameters were converted into their corresponding linguistic variables, the inference engine evaluated the set of if-then rules (rule base), and linguistic values of the linguistic variable. The linguistic variables were then converted into a crisp FLC output value, the reason why the second transformation was performed by a defuzzifier that does both scale mapping and defuzzification. The defuzzifier used the rules' consequent membership functions to generate non-fuzzy, crisp control action from the inferred fuzzy control action.

On the basis of fuzzy subsets, the following control rules are described based on the concept of the obstacle ovoidance robot controller:

$$if\ disl\ is\ disli\ and\ disf\ is\ disfj\ and\ disr\ is\ disrk$$

$$then\ Vl\ is\ \ Vlijk\ and\ Vr\ is\ Vrijk \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.21$$

Where i=1-3

j=1-3

k=1-3

Because disl, disf, and disr all have three membership functions.

Two sets of rules can be derived from Equation 3.21.

$$If\ disl\ is\ disli;\ and\ disf\ \ is\ disfj\ and\ disr\ \ is\ disrk\ then\ Vl\ is\ Vlijk$$

$$If\ disl\ is\ disli;\ and\ disf\ \ is\ disfj\ and\ disr\ \ is\ disrk\ then\ Vr\ is\ Vrijk \ldots\ldots 3.22$$

According to the fuzzy logic control method, a factor W*ijk* is defined for the rules as follows(Spolaor, 2019).

$$Wijk\ (vel) = \ \mu disli\ (disi) \wedge\ \ \mu dfj\ (disj)\ \wedge \mu disrk\ (disk) \ldots\ldots\ldots\ldots\ldots 3.23$$

Disl, disj, and disk are the values of the measured left front and right distance, respectively. The composition rule of inference was used to compute the membership value of the left and right wheel velocity *vel Vl* and *vel Vr*.

$$\mu Vlijk\ (vel) = Wijk \wedge\ \ \mu Vlijk(vel\ Vl)\ \ \forall\ vel \in Vl$$

$$\mu Vrijk\ (vel) = Wijk \wedge\ \ \mu Vrijk(vel\ Vl)\ \ \forall\ vel \in Vr \quad \text{……..…..3.24}$$

The final inference derived from combining the fuzzy rule outputs is as per equation 3.25.

$$Vl(vel) = \mu Vl'111(vel\ Vl)\ \ V \ldots\ldots V\ \mu Vl'ijk(vel\ Vl)\ V \ldots\ldots V\ \mu Vl'333(vel\ Vl).$$

$$Vr(vel) =$$
$$\mu vr'111(vel\ vr)\ \ V \ldots\ldots V\ \mu Vr'ijk(vel\ vr)\ V \ldots\ldots V\ \mu vr'333(vel\ vr).$$

……3.25

The center of gravity method was used to compute the crisp value of the left and right velocity, which was similar to the work performed in the research (Spolaor, 2019).

Left velocity.

$$Vl = \frac{\int x\mu Vl(x)dx}{\int x\mu Vl(x)dx} \quad \text{……..3.26}$$

Right velocity.

$$Vr = \frac{\int x\mu Vr(x)dx}{\int x\mu Vr(x)dx} \text{……..3.27}$$

Where x = velocity.

The crisp values $Vr\ and\ Vl$ define the duty cycle variation of the PWM signal controlling the speed of the DC motors driving the wheels.

3.7.7 External client application.

The robot simulation was controlled by writing an external client application that relied on the remote API in MATLAB. This was a convenient and easy way to run the control code from an external application. This allowed control of the model (virtual robot) with the same code as the one that runs the real robot. The remote API functionality relied on the remote API plugin (on the server-side), and the remote API code on the client-side. Both programs/projects. (can be easily extended or translated for support of other languages) and were found in the 'programming' directory of V-REP's installation.

To use the remote API functionality in the MATLAB program, remoteApiProto.m, remApi.m, remoteApi.dll, and remoteApi.dylib or remote API were important based on the target platform. Files were located in V-REP's installation directory, under *programming/remoteApiBindings/Matlab*. The remoteApi shared library using *remoteApiSharedLib.vcproj* or *remoteApiSharedLib_Makefile*) can be built if not already built.

Once the above elements were available in MATLAB current folder, we called *vrep=remApi ('remoteApi')* to build the object and load the library. To enable the remote API on the client-side (i.e. Application), we called *vrep.simxStart*. V-REP remote API functions can easily be recognized from their "simx"-prefix. MATLAB must be the same bit-architecture as the remoteApi library. A 64bit MATLAB with 32bit remoteApi library did not work, and vice-versa.

3.7.8 MATLAB type-1 fuzzy logic toolbox

In this study, the MATLAB Fuzzy Logic Toolbox was used to develop a Fuzzy Inference System (FIS) for an obstacle avoidance mobile robot in a static unknown environment. The toolbox contained five primary graphical user interface (GUI)

tools for building, editing, and observing fuzzy inference systems. Figure 3.12 shows the GUI component i.e., the FIS Editor, Membership Function Editor, Rule Editor, Rule Viewer, and Surface Viewer.Fuzzy Logic Designer handled the system's high-level issues, including input and output variables and their names. The Fuzzy Logic Toolbox does not impose any restrictions on the number of inputs. However, the number of inputs was limited by our robot's existing sensors. If the number of inputs or membership functions is too large, it may be difficult to evaluate the FIS.

Membership Function Editor: The shapes of all the membership functions associated with each variable were defined.

Rule Editor: Edited the list of rules that described the systems.

Rule Viewer: Fuzzy inference diagram was used as a diagnostic method to determine which rules are active and how individual membership function shapes influenced the outcomes.

Surface Viewer: It generated and plotted an output surface map for the system to visualize the reliance of one of the outputs on any one or two of the inputs.

The GUIs were dynamically linked, any changes made to the FIS via one of them affected the other open GUIs. Changing the names of membership roles in the Editor, for example, had an impact on the rules displayed in the Rule Editor. To read and write variables to the MATLAB workspace GUIs was used (the read-only viewers can still exchange plots with the workspace and save them to a file).

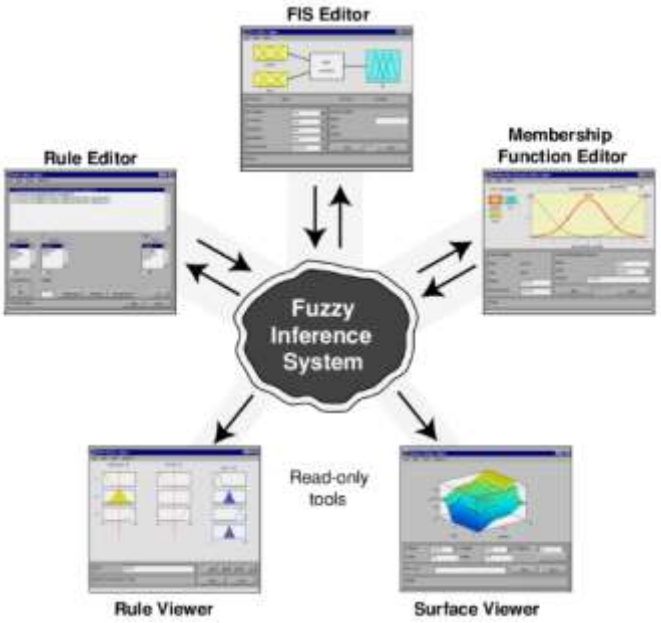


Figure 3.12: T1FLC Graphical tools

The information about a fuzzy inference system was displayed by the Fuzzy Logic Designer (FLD). At the MATLAB prompt, we typed the following command to open FLD: *Fuzzy Logic Designer*. Fuzzy Logic Designer opens and displays an example of the fuzzy inference method with the names of each input variable on the left and the names of each output variable on the right.

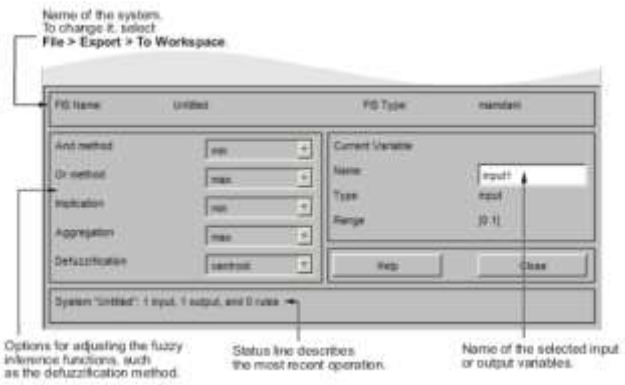


Figure 3.13: Fuzzy logic designer screenshot

The drop-down lists in the Fuzzy Logic Designer in figure 3.14 allows us to change the fuzzy inference functions. The name of an input or output variable, as well as its form and default range, were displayed in the Current Variable field. The most recent activity was shown in a status line at the bottom. Figure 3.15 shows the generic untitled Fuzzy Logic Designer, which has one input and one output.

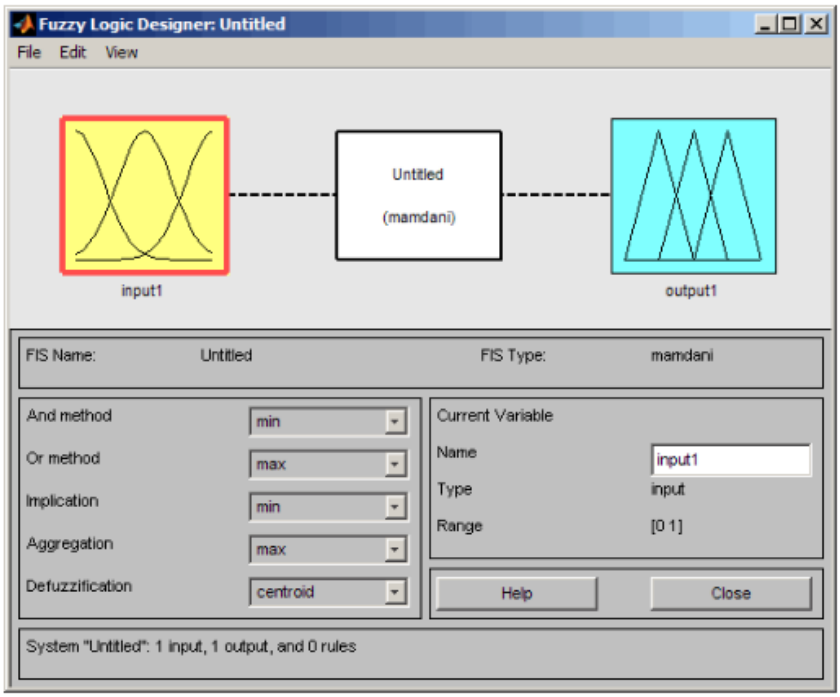


Figure 3.14: Fuzzy logic designer untitled

The Membership Function Editor tool allowed viewing and editing of all membership functions associated with the fuzzy inference system's input and output variables.

Figure 3.16 shows how the Membership Function Editor shared some features with the FLD.

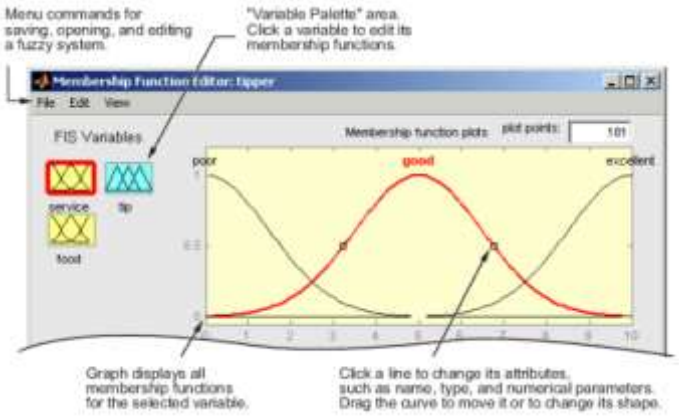


Figure 3.15: Membership function editor

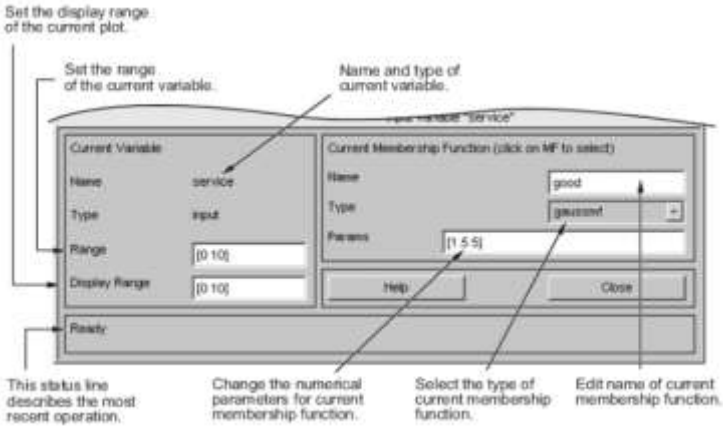


Figure 3.16: MF variable palette

As shown in figure 3.17, the Membership Function Editor has a "Variable Palette" on the upper-left side of the graph that allows you to set the membership functions for a given variable.

We selected a FIS variable in this region by clicking to configure the membership functions associated with an input or output variable for the FIS. Select Add MFs from the Edit pull-down menu. A new window pops up, allowing us to choose the type of membership function as well as the number of membership functions associated with the variable. After selecting the membership function, the controls in the lower-right corner of the window allowed adjusting the name, form, and parameters (shape) of the function.

The main graph displayed the membership functions from the current variable. There were two ways to manipulate these membership functions. Using the mouse, we selected a specific membership function associated with a given variable quality and dragged it from side to side. This behavior affected the statistical definition of the membership function's quality for a given variable. By clicking on the small square drag points on the membership function, and then dragging the function with the mouse toward the outside for dilation, or toward the inside for contraction, the selected membership function could be tagged for dilation or contraction. The parameters associated with that membership function were altered as a result of this action.

The current variable's type and name were displayed below the Variable Palette. In this region, there was a text field that allows us to change the range of the current variable (universe of discourse) and allowed the setting of limits of the current plot.

3.8 Mobile robot operation

The general movement of the robot was created from fuzzy rules. The program read all sensors values attached to the robot. The mean distance value of the obstacle sensor was calculated. The program received range of data from sensors to conduct fuzzy logic operations. Center of area (COA) method of defuzzification methods was used for this purpose as in eqn.3.33.

$$Z_0 = \frac{\sum_{i=1}^{n} Z_{i\mu_{out}(z_i)}}{\sum_{i=1}^{n} \mu_{out(z_i)}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.33$$

Where

$\mu_{out}(Z_i)$ are the *i*=1, 2……...n sampled values of the aggregated output membership function.

Z_0 is the crisp value which defines the duty cycle variation of the PWM signal controlling the speed of the DC motors driving the wheels.

The robot made turns to avoid obstacles depending on the output of the fuzzy logic program. When the turning process was completed, sensors detected and read again for the next action. If there was an obstacle, avoiding process still ran until there were no obstacles then the robot moved forward. After each movement, the sensor was read again for the next path planning.

3.9 Sensor fusion for a type-1 Mamdani fuzzy logic-controlled

Sensor fusion is the method of combining data from various sensors to reduce the amount of uncertainty in robot navigation. It essentially aims to overcome the shortcomings of individual sensors by collecting and fusing data from multiple sensors to provide more accurate and less ambiguous results. This more reliable data can then be used to provide deeper perspectives, allowing for more intelligent and accurate responses.

3.9.1 Determination of input and output variables

The receptive left distances (disl), front distance (disf), and right distance (disr) between the robot and obstacles from the ultrasonic sensors on the chassis of the robot were inputs to the fuzzy controller. The robot's behavior was determined by the fuzzy controller based on the sensed obstacle distance. The controller's outputs were left velocity (Vl) and right velocity (Vr) for driving two wheeled robot. The presumption was that the driving wheels' velocity could be monitored to fire anticipated commands from the fuzzy inferred performance. As shown in Fig.3.15 (a) the inputs disl, disf, and disr were used as antecedents with “three fuzzy linguistic sets labeled NEAR (N), MEDIUM (M), and FAR (F)”. The discussion region of disl, disf, and disr belonged to [0, 1000mm]. As shown in Fig.3.15 (b), “three fuzzy linguistic sets labeled SMALL (S), MEDIUM (M), and LARGE (B)” was used to represent the output variables Vl and Vr. Where the discussion regions were chosen between (0 and 100 cm/s). This method bear little resemblance to the

study done by (C. S. Chiu et al., 2016)

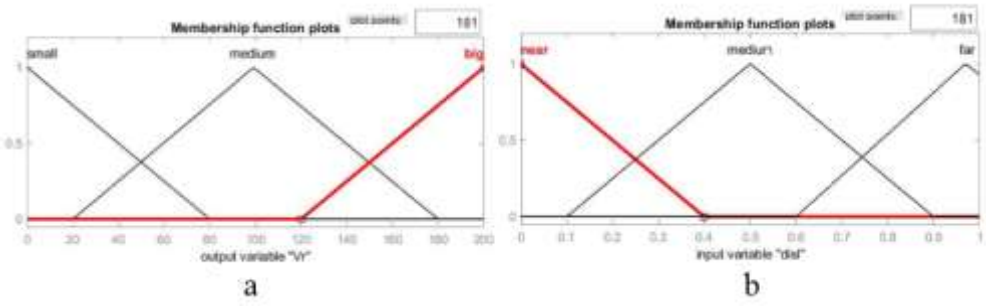


Figure 3.17: (a) Output variables Vl, Vr. (b) Premise variables disl, disf, disr

To avoid obstacles, the mobile robot reacted appropriately based on information detected by ultrasonic sensors (disl, disf, and disr). Obstacle avoidance rules were formulated using an expert human driving experience.

The general rules satisfied the following criteria: “The left and right wheels move quickly if obstacles are not available in the immediate vicinity or if the obstacles are far around the front side of the robot, resulting in the output (BIG)”. When the mobile robot comes face to face with an obstacle, the driving wheels' moving velocities are adjusted to provide adequate reactive motion. The distance between the robot and the obstacle and whether the obstacle is on the left, right, or front, determined the direction.

In various studies, the standard 27 fuzzy rules formulated by an expert driver have been used (Batti et al., 2019; Bayar et al., 2014; Carlos Erlan Olival Lima et al., 2013; C. S. Chiu et al., 2016; Pandey et al., 2014). Based on the controller's three inputs, each of which has three membership functions yields.

$Number\ of\ rules = 3^3 = 27$ listed next.

Rule l: If (disl is N) and (disf is N) and (disr is N), then (Vl is S) and (Vr is S).

Rule 2: If (disl is N) and (disf is N) and (disr is M), then (Vl is M) and (Vr is S).

Rule 3: If (disl is N) and (disf is N) and (disr is F), then (Vl is B) and (Vr is M).

Rule 4: If (disl is N) and (disf is M) and (disr is N), then (Vl is S) and (Vr is S).

Rule 5: If (disl is N) and (disf is M) and (disr is M), then (Vl is M) and (Vr is S).

Rule 6: If (disl is N) and (disf is M) and (disr is F), then (Vl is B) and (Vr is M).

Rule 7: If (disl is N) and (disf is F) and (disr is N), then (Vl is M) and (Vr is M).

Rule 8: If (disl is N) and (disf is F) and (disr is M), then (Vl is B) and (Vr is S).

Rule 9: If (disl is N) and (disf is F) and (disr is F), then (Vl is B) and (Vr is M).

Rule 10: If (disl is M) and (disf is N) and (disr is N), then (Vl is S) and (Vr is B).

Rule 11: If (disl is M) and (disf is N) and (disr is M), then (Vl is S) and (Vr is S).

Rule 12: If (disl is M) and (disf is N) and (disr is F), then (Vl is B) and (Vr is S).

Rule l3: If (disl is M) and (disf is M) and (disr is N), then (Vl is S) and (Vr is B).

Rule 14: If (disl is M) and (disf is M) and (disr is M), then (Vl is M) and (Vr is M).

Rule l5: If (disl is M) and (disf is M) and (disr is F), then (Vl is M) and (Vr is S).

Rule 16: If (disl is M) and (disf is F) and (disr is N), then (Vl is S) and (Vr is B).

Rule l7: If (disl is M) and (disf is F) and (disr is M), then (Vl is M) and (Vr is M).

Rule 18: If (disl is M) and (disf is F) and (disr is F), then (Vl is B) and (Vr is M).

Rule l9: If (disl is F) and (disf is N) and (disr is N), then (Vl is S) and (Vr is B).

Rule 20: If (disl is F) and (disf is N) and (disr is M), then (Vl is S) and (Vr is M).

Rule 2l: If (disl is F) and (disf is N) and (disr is F), then (Vl is S) and (Vr is B).

Rule 22: If (disl is F) and (disf is M) and (disr is N), then (Vl is N) and (Vr is M).

Rule 23: If (disl is F) and (disf is M) and (disr is M), then (Vl is M) and (Vr is B).

Rule 24: If (disl is F) and (disf is M) and (disr is F), then (Vl is M) and (Vr is B).

Rule 25: If (disl is F) and (disf is F) and (disr is N), then (Vl is S) and (Vr is M).

Rule 26: If (disl is F) and (disf is F) and (disr is M), then (Vl is M) and (Vr is B).

Rule 27: If (disl is F) and (disf is F) and (disr is F), then (Vl is B) and (Vr is B).

3.9.2 Sensors arrangement

Unknown obstacles in the environment were detected using ultrasonic proximity sensors labeled **a-i** on the front side of the mobile robot, as shown in Figure3.9. These enabled collision avoidance navigation with object detection. Nine sensors were used, each scanning 20^0 angle thus able to cover 180^0 scan for obstacles. Three distance sensors were mounted on the front side, three on the left and three on the right sides of the mobile robot. Each sensor had a maximum detection range of 1000mm. As a result, the sensors' covered range can be represented by a half-circle with a radius of 1000mm. When an obstacle was detected within any sensor's detection range, the distance between the mobile robot and the obstacles was calculated by coordinating all of the sensory data.

The traditional models normally have three ultrasonic sensors mounted in areas b, e, and h. Our proposed alternative model has nine ultrasonic distance sensors activated, allowing us to scan for obstacles over a larger area.

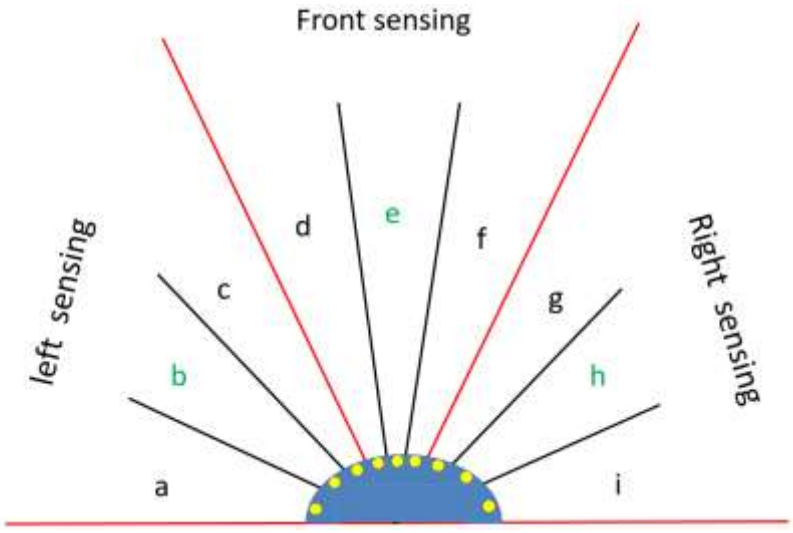


Figure 3.18: Arrangement of sonar sensors

However, the controller would be extremely complex if all nine ultrasonic sensors were used as input data. The control algorithm was developed in such a way that the mobile robot could respond to “unknown environment effectively in real-time”. To simplify the controller, three sensors were combined and treated as a single

input (Aguileta et al., 2019; C. Chiu, 2015). As shown in Figure 3.11, Sensors **a,b,** and **c** were merged for the left scan, sensors **d,e**, and **f** were used together for front scanning, and sensors **g,h,** and **i** were combined for right sensing. To obtain the output arithmetic mean distance was computed by eqn.3.17, to synchronize regions for corresponding coordinated sensors.

$$Mean = \frac{\sum xn}{n} = \frac{x1+x2+\cdots xn}{n} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.17$$

Where

x= mean.

$\sum$ = summation

Xi= Value of the i^{th} sensor data X,

i = 1, 2, …, n

n = total number of sensors

We represented left distance (**disl)** as the averaged distance of sensors "a, b, and c" on the left, front distance (**disf)** as the averaged distance of sensors "d, e, and f" in the front, and right distance (**disr)** as the averaged distance of sensors "g,h, and I" on the right of the mobile robot. The input condition to the fuzzy logic controller was described by equations 3.18-3.20.

$$\text{disl} = \frac{da+db+dc}{3} \qquad 3.18$$

$$disf = \frac{dd+de+df}{3} \qquad 3.19$$

$$disr = \frac{dg+dh+di}{3} \quad 3.20$$

Where **da**, **db** and **dc** were the measured obstacle distances from the sensors "**a**, **b** and **c"** respectively; **dd**, **de df** are the measured obstacle distances from the sensors, respectively "**d,e**, and **f"**;, **dg**, **dh** and **di** are respectively the measured distances from sensors "**g, h** and **I".** As a result, variable **disf** represents the distance of the obstacle in the front of the environment, while variables **disr** and **disl** represent the distance of the obstacle in the right and left sides, respectively.

3.9.3 Sensor fusion

Sensor fusion refers to the ability to combine multiple sensor inputs into a single model or environment image. The resulting model is more accurate because the strengths of different sensors are combined resulting in more accurate model (Aguileta et al., 2019). Our study was based on Competitive/Redundant to provide independent measurements of the same target object to give the highest level of completeness. The output of the sensors required post-processing; the level chosen was Low-Level in which sensor fusion took the raw output of the sensors to ensuring noise was not introduced in the data during transformation. Our new control model after fusion had n = 3 and m = 3 the total number of fuzzy rules was $k = m^n = 3^3 = 27$ rules just like the classical models but with the sensor fused outside the controller instead of having the 729 rules.

3.10 Reducing fuzzy rules for efficient robot navigation

When building a fuzzy controller for a complex system, several observable output and actuating input variables are always involved. Each variable is denoted by m linguistic labels, indicating that the total number of rules is m^n.where n is the number of system variables. Considering m = 9 and n = 3 then the total number of

fuzzy rules will be $k = m^n = 9^3 = 729$. It is obvious from the above example that

using fuzzy control may result in a dimensionality explosion. The size of the fuzzy control system rule base grows exponentially as the number of input variables grows. As a result, in the controller's design, reducing the rule base is critical.

3.10.1 Similarity measure between rule premises

Fuzzy Similarity Measure between fuzzy rules was used to further reduce rules. The traditional rule base consisted of 27 if-then rules as antecedent. There were inconsistencies between rules based on factual values as measured by the Credulity measure and the Incredulity calculation (Kalpana & Kumar, 2013). We examined how similar the rules were. Two rules were chosen in order from the 27 to check similarity. The degree of similarity (DS) between the rules was calculated by dividing the total number of similar parameters between the rules by the total number of input and output parameters. There were two kinds of DS conditions: identical and dissimilar. The Constant Degree of Similarity (CDS) value is normally set to 50% (Kalpana & Kumar, 2013). If the DS value is less than or equal to 50%, we assumed the two rules were identical. We concluded that the two rules were not the same if the DS value is greater than 50%. When the rules were different, we did not change the rule base. We combined rules if they were the same. We calculated the Cvalue, Kcalvalue, and Kbase value. (Kalpana & Kumar, 2013).

Cvalue is the average of the fuzzy triangular numbers for each dissimilar parameter. Allowing a, b, and c to be fuzzy triangular numbers. cvalue can be calculated using eqn.3.28.

$$cvalue = \frac{a+b+c}{3} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.28$$

Kcalvalue is defined in eqn.3.29, compute the difference between the cvalues of the dissimilar parameters 1 and 2.

$$Kcalvalue = dissimila\ parametr2(cvalue) - dissimilar\ parameter1(cvalue) \ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.29$$

Kbase is defined as the discrepancy between a dissimilar parameter 2's first triangular number and a dissimilar parameter 1's first triangular number.

Kbase = first triangular no. of dissimilar parameter2 - first triangular no. of dissimilar parameter1

3.10.2 Inference to reduce rule

If (Kcalvalue ≥ Kbase) Dissimilar parameters 1 and 2 are decreased. Otherwise, the minimum value of a dissimilar parameter is deleted, while the maximum value of a dissimilar parameter is considered in the rule. else If (Kcalvalue < Kbase) then (Not reduced). Table 3.1 shows the fuzzy set and fuzzy numbers of the obstacle avoidance mobile robot during experimentation.

Table 3.1: Fuzzy set and fuzzy numbers of the robot

Fuzzy variables	Representation of fuzzy variables	Linguistic variables	Representation of fuzzy numbers.	Fuzzy triangular numbers (MFs)
Left sensor	disl	Near	D_{11}	[-04 0 0.4]
		Medium	D_{12}	[0.1 0.5 0.9]
		Far	D_{13}	[0.6 0.96 1.4]
Front sensor	disf	Near	D_{21}	[-0.4 0.041 0.4]
		Medium	D_{22}	[0.1 0.5 0.9]
		Far	D_{23}	[0.6 1 1.4]
Right sensor	disr	Near	D_{31}	[-0.4 0 0.4]
		Medium	D_{32}	[0.1 0.5 0.9]
		Far	D_{33}	[0.6 1 1.4]
Left velocity	Vl	Small	V_{11}	[-80 0 80]
		Medium	V_{12}	[20 100 180]
		Big	V_{13}	[120 200 280]

Right velocity	Vr	Small	V_{21}	[-80 0 80]
		Medium	V_{22}	[20 100 180]
		Big	V_{23}	[120 200 280]

Using similarity of rule premise (SRP) used in our study first defined by Authors (Jin et al., 1999).

$$Ri: if\ X_1\ is\ A_{i1}\ and\ \ldots.and\ X_n\ is\ A_{in}\ then\ y\ is\ B_i$$

$$Ri: if\ X_1\ is\ A_{k1}\ and\ \ldots.and\ X_n\ is\ A_{kn}\ then\ y\ is\ B_k$$

The SRP of the two rules can be defined by eqn.3.30

$$SRP\ (i,k) = \min^{n} S(A_{ij}, A_{kj}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.30$$

By testing the SRP of fuzzy rules, it was possible to avoid duplicate and inconsistent rules. Similar fuzzy premise and consequences were combined to form a standard regulation to reduce the number of rules. The AND operator was used to determine the antecedent portion of the rule. Uncertainty was stabilized between the rules using Fact Values, which are divided into two categories: credulity and incredulity. Using Fuzzy Similarity measure for fuzzy rules (Kalpana & Kumar, 2013) reduction the following steps were considered as below.

Begin:

Step 1: For the obstacle avoidance robot, generate an initial fuzzy set, fuzzy numbers, and fuzzy rules.

Step 2: Using eqn.3.31, propose a new similarity measure between three sets A, B, and C.

$$S(A,B,C) = \left(\frac{\max(A,B) - \min(A,B)}{3}, \frac{\max(B,C) - \min(B,C)}{3} \right) \ldots..3.31$$

If the values of sets A, B, and C differ, we assume they are identical. If the two values above are the same, combine them into two sets of A1 and B1.

Step 3: In the rules, apply the combined set.

Step 4: Using eqn.3.32, calculate the degree of similarity between all laws in an orderly manner.

$$\text{Degree of Similarity (DS)} = \frac{Total\ number\ of\ similar\ parameter\ in\ the\ rule}{Total\ number\ of\ input\ and\ output\ parameter} x100\% \ldots.3.32$$

Step 5: set constant degree of similarity (CDS) say 50%.

If (DS >CDS) then

Go to step 6

Else

Stop the algorithm

Step 6: Determine the Cvalues of two dissimilar input parameters (C1 and C2) as well as the output parameter (D1 and D2).

Cvalues for C1 and C2 are computed by taking the average of fuzzy numbers.

Kcalvalue = Cvalue(C2) – Cvalue(C1)

Kbase = First triangular number of C2 – First triangular number of C1

If (Kcalvalue > Kbase) two fuzzy numbers ie, C1 and C2 are reduced. Minimum value C1 is deleted,

Maximum value C2 was considered in the rules.

Else not reduced.

Similarly, the Cvalue, Kcalvalue, and Kbase values of D1 and D2 are computed, and the Rules are combined into a single rule. The following illustrates a working example.

Considering autonomous mobile robot navigation rules.

Rule l: If (disl is N) and (disf is N) and (disr is N), then (Vl is S) and (Vr is S).

Rule 2: If (disl is N) and (disf is N) and (disr is M), then (Vl is M) and (Vr is S).

$$\text{Degree of Similarity (DS)} = \frac{Total\ number\ of\ similar\ parameter\ in\ the\ rules}{Total\ number\ of\ input\ and\ output\ parameter} x100\%$$

$$DS = \frac{3}{5} = 60\%$$

Thus 60% is greater than 50% which means rule 1 and rule 2 are similar. Considering another rule.

Rule 2: If (disl is N) and (disf is N) and (disr is M), then (Vl is M) and (Vr is S).

Rule 3: If (disl is N) and (disf is N) and (disr is F), then (Vl is B) and (Vr is M).

$$\text{Degree of Similarity (DS)} = \frac{Total\ number\ of\ similar\ parameter\ in\ the\ rule}{Total\ number\ of\ input\ and\ output\ parameter} x100\ \%$$

$$DS = \frac{2}{5} x100 = 40\%$$

40% is less than 50% thus rule 2 and rule 3 are dissimilar thus all must be retained for use.

Rule 1 and rule 2 are similar but have dissimilar parameters as (D_{31} and D_{32}) and (V_{21} and V_{22}) their values can be found in Table 3.1 representing fuzzy set and fuzzy numbers of the obstacle avoidance.

$$disr(D_{31}) = \frac{-0.4 + 0 + 0.4}{3} = 0$$

$$disr(D_{32}) = \frac{0.1 + 0.5 + 0.9}{3} = 0.5$$

$$disr\ (kcalvalue) = disr(D_{32}) - disr(D_{31}) = 0.5 - 0 = 0.5$$

$$disr\ (kbase\ ) = first\ triangle\ number(D_{32}) - first\ triangle\ number(D_{31})$$

0.1--0.4= 0.5

$disr\ (kcalvalue) = \ disr\ (kbase\ )$ Thus, we can take either of the rule for representation

However, the minimum value should be removed and the maximum value should be taken into account in the rule base.

Considering V_{21} and V_{22}

$$Vl(V_{21}) = \frac{-80 + 0 + 80}{3} = 0$$

$$Vl(V_{22}) = \frac{20 + 100 + 180}{3} = 66.6$$

$$Vl\ (kcalvalue) = Vl(V_{22}) - Vl(V_{21}) = 66.6 - 0 = 66.6$$

$$Vl\ (kbase\ ) = first\ triangle\ number(V_{22}) - first\ triangle\ number(V_{21})$$

20- - 80$= 100$

$Vl(kcalvalue) < Vl(kbase\ )$ Thus, the maximum value V_{22} to be considered in the rule base.

3.11 Tuning membership functions

Several variables in the controller design need to be adjustment to regulate the final output. Tuning FLC was an overwhelming task as numerous parameters were to be adjusted. Changing the scaling factor of a fuzzy variable altered the definition of each membership function by the same ratio. When the peak value of a membership function was changed only rules affected were those that used the changed fuzzy label. Changing the width value of a membership function affected interpolation between the peak value of the function and its adjacent MF. From this it was observed that any adjustments to be made towards tuning the Controller, should start with Scaling factors of variables then Peak values of membership functions and finally the rules.

Gross adjustment of the system was achieved by iteratively adjusting membership Functions. Experiments were conducted to investigate the effects of the shape of membership function on the mobile robot navigation performance. Once gross tuning was accomplished, the FLC was fine-tuned. This involved slight adjustments of individual membership functions and their ranges for optimum performance.

Tuning was broken down into three main steps which were adjusted to fine-tune the overall output. In the fuzzification step, the range values of each linguistic variable in the input MF could be changed. By changing these values to cover either a shorter or broader range, changed the slope of the shape of the MF. In the second stage, the rule processing referencing the rule base was revised to create rules with substantial differences in the fuzzy outputs. The last section, defuzzification, had two areas that were adjusted. The first was the output MF. It can be modified the same way as the input MF changing the range of the linguistic variables that compose the output MF. Out of all the areas to be fine-tuned, the method of defuzzification was generally the first element to change. If the optimal response was not evoked by changing the method of defuzzification, then we went back and tweak the other steps of the FLC.

3.10.1 One-way analysis of variance

The one-way analysis of variance (ANOVA) was used to determine whether there were any statistically significant differences between the means of three membership functions (gaussian trapezoidal and triangular). The independent variable had three different groups. To identify whether there were statistically significant variations between the means of the three membership functions (triangular trapezoidal and gaussian), a one-way variance analysis (ANOVA) was used. The one-way ANOVA compared the media between the groups and calculated if these means vary statistically. In conducting this experiment, we had two research questions:

Do MFs have a significant effect on the time taken for a robot to reach the target?

How strong is the effect of membership functions?

Null hypothesis: The null hypothesis states that MF has no effect on the time taken for robot to reach the target using eqn.3.38.

$$H_0 = \mu_1 = \mu_2 = \mu_3 \ldots\ldots\ldots\ldots\ldots\ldots 3.38$$

Alternative hypothesis: The alternative hypothesis states MF has an effect on the time taken for the robot to reach the target.

$$H_1 \neq \mu_1 \neq \mu_2 \neq \mu_3.$$

Where:

μ = MFs group mean

k = number of groups.

If one-way ANOVA returns a statistically significant result, we accept the alternative hypothesis (H_1), which means that there are at least two group means that are statistically significantly different from each other. To determine which specific groups differed from each other, we performed a post hoc test.

To test whether different membership functions affect the time for an autonomous robot to complete a task ANOVA was used. In the study we assigned the input MFs (a) triangular MF, (b) trapezoidal MF (c) gaussian MF. The time to complete a task was the outcome (dependent) variable. In this work independent variables were also called attribute independent variables because we were splitting the group based on some attribute possessed (type of input MFs). Each group was then measured on the same dependent variable having undertaken the same task.

The one-way ANOVA test statistics also referred to as the F statistics determine whether the group means vary considerably in an independent variable with k groups. During F statistics computation all statistical components were shown in table 3.4.

Table 3.2: *F* statistic components.

	Sum of squares	df	Mean square	F
Treatment	SSR	df_r	MSR	MSR/MSE
Error	SSE	df_e	MSE	
Total	SST	df_T		

Where

SSR = the regression sum of squares

SSE = the error sum of squares

SST = the total sum of squares (SST = SSR + SSE)

df_r = the model degrees of freedom (equal to $df_r = k - 1$)

df_e = the error degrees of freedom (equal to $df_e = n - k - 1$)

k = the total number of groups (levels of the independent variable)

n = the total number of valid observations

df_T = the total degrees of freedom (equal to $df_T = df_r + df_e = n - 1$)

MSR = SSR/df_r = the regression mean square

MSE = SSE/df_e = the mean square error

Then the F statistic itself is computed using eqn.3.39.

$$F = \frac{MSR}{MSE} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.39$$

3.10.3 Post hoc test

A post hoc test also known as a multiple comparison test was used to determine which groups were distinct from one another. We were able to investigate the

differences between the means of different groups while also adjusting for the family-wise error rate. We used Tukey's Test in making any pairwise comparison.

The ANOVA test indicated the general difference between groups, but it doesn't tell which particular groups differed, so post hoc tests were very helpful. Post hoc tests are used to confirm where discrepancies between groups occurred; however, they are only used when there is a statistically significant difference in group means (i.e., a statistically significant one-way ANOVA result). Attempts to monitor the experimentwise error rate (usually alpha = 0.05) are made with post hoc tests.

When the p-value for the ANOVA is statistically significant, we perform a post hoc test. If the p-value isn't statistically significant, it means that the means for all of the groups are not different, and no post hoc test is needed to determine which groups are different.

3.12 Comparison of type-1 and type-2 fuzzy systems for obstacle avoidance robot

The Type-1 Fuzzy Logic Controllers (FLCs) have been used in a variety of real-world applications with great success. We investigate type-1 and type-2 fuzzy logic controllers as the framework for obstacle avoidance robot control systems in a static unknown environment. According to research, the traditional type-1 FLC, which employs crisp type-1 fuzzy sets, is incapable of handling high levels of uncertainty appropriately. In some applications, it has been demonstrated that using type-2 fuzzy sets can handle uncertainties effectively and yield better results. This has been given little attention in robotic systems which we compare.

3.12.1 Development of type- 2 fuzzy logic system

The type-2 fuzzy logic system is intended to manage uncertainty while improving systems performance. T2FLCs have been demonstrated to outperform T1FLCs. The rules for type-2 fuzzy cases are the similar to type-1 cases (Karnik & Mendel, 1998; Naik & Gupta, 2017). The distinction is due to the nature of membership functions (Ruiz-Garcia et al., 2019). General concepts of type-2 fuzzy systems modelling are

covered in this section. More information on type 2 fuzzy logic applications can be found in a comprehensive review (Mittal et al., 2020)

In T2FLC, the output processing includes an additional stage called the Type-Reduction (TR) algorithm, which converts Type-2 into equivalent Type-1. T2FLC has been argued to have a high potential for producing better performing systems because of the following factors:

- Type-2 fuzzy systems can assist in reducing the difficulties associated with modelling a rule-based system. (Chao et al., 2020).
- Type-2 FS can be used to tune and improve the understanding of a rule-based system. (Ontiveros-Robles & Melin, 2020).
- In a Type-2 FLS complex input/output relationships can be obtained.
- Furthermore, these input/output relationships can be modelled with no difference in the number of rules (e.g., an increase or decrease) (Mendel et al., 2020; D. Wu, 2013).

A Mamdani fuzzy system's rule base consists of l rules (l = 1, … , m), as seen in the following model with two inputs and one output:

$$R^l: if\ x_1\ is\ F_1^l\ and\ x_2\ is\ F_2^l\ then\ y^l = G^l \dots\dots\dots\dots\dots\dots 3.40$$

Where x_1 and x_2 are input linguistic variable and y is the output linguistic variable in the domain X_1, X_2 and Y respectively. $F_1^l \in \{F_{1,1} \dots., F_{1,n1}\}, F_2^l \in \{F_{1,1} \dots., F_{1,n2}\}, and\ G^l \in \{G_{1,1} \dots., G_{1,my}\}$, are fuzzy sets, also known as linguistic values, and y^1 is the rule′s consequence.

Type-1 Fuzzy sets are used in Type-1 Fuzzy systems, whereas Type-2 Fuzzy sets are used in at least one Type-2 Fuzzy system. As a result of this simple distinction, a

mathematical extension (fuzzy set operations) as well as inference processing in Fuzzy systems are required. (D. Wu, 2013).

A type-2 fuzzy set $\tilde{A}$ is defined by the following tuple.

$$\tilde{A} = \left\{ \begin{matrix} ((x,u), \mu_{\tilde{A}}(x,u) \text{ I } \forall x \in X, \\ \forall u \in J_x \in [0,1], \mu_{\tilde{A}}(x,u) \in [0,1] \end{matrix} \right\} \ldots\ldots\ldots\ldots .3.41$$

where, X denotes the fuzzy variable's domain and, and u belongs to the interval called the primary membership, that is, $u \in J_x \subseteq [0,1], \mu_{\tilde{A}}(x,u)$ is a 2-dimensional membership function, with $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$ defining secondary membership (Karnik & Mendel, 1998; Ruiz-Garcia et al., 2019).

When all $\mu_{\tilde{A}}(x,u) = 1$, a type-2 fuzzy set is known as an Interval Type-2 fuzzy set (IT2FS). As a result, an Interval Type-2 fuzzy set can be described as:

$$\tilde{A} = \{((x,u), 1 \text{ I } \forall x \in X, \forall u \in J_x \subseteq [0,1]\} \ldots\ldots\ldots\ldots\ldots 3.42$$

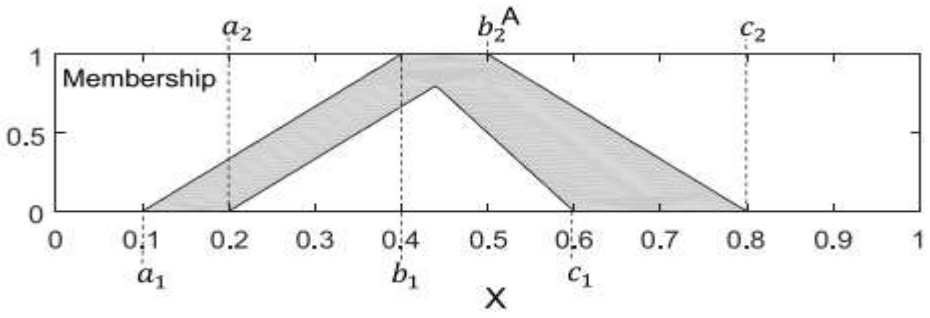


Figure 3.19: Example of a triangular T2FS.

Type-1 Fuzzy Systems can be modeled using T2FSs. The third dimension is unnecessary, as shown by Eq. (3.41), and an IT2FS can be represented by a two-

dimensional tuple. An IT2FS, like a T1FS, can be described by a variety of parametric membership functions, but there are numerous ways to describe the form or class of a fuzzy set in the last approach. (Yekinni & Dan-Isa, 2019) (Castillo & Melin, 2012) (Karnik et al., 1999). According to Castello and Melin , a set of six parameters can be used to describe an interval type-2 triangular MF in this work, as shown in the equation 3.43.

$$\mu_{\tilde{A}}(x) = t2trimf\,(x; a_1, b_1, c_1, a_2, b_2, c_2 \ldots\ldots\ldots\ldots .3.43$$

The first three parameters define the embedded T1-Triangular fuzzy set on the left, while the remaining parameters define the embedded T1-Triangular fuzzy set on the right. A triangular IT2FS is represented by $t2trimf$ (x; .1, .4, .6, .2, .5, .8).

The footprint of uncertainty (FOU) is the bounded area defined in the definition of an IT2FS (the shaded region in Fig. 3.14). It is the sum of all primary memberships Jx found within the shaded area after solving equation 3.44.

$$FOU(\tilde{A}) = \bigcup_{x \in X} J_x \ldots\ldots\ldots\ldots\ldots\ldots 3.44$$

The $FOU(\tilde{A})$ is bounded by the upper membership function (UMF) and lower membership function (LMF), as shown in Eqs. (3.45a) and (3.45b) (LMF).

$$\bar{\mu}_{\tilde{A}}(x) = \overline{FOU(\tilde{A})} \ldots\ldots\ldots 3.45a$$

$$\underline{\mu}_{\tilde{A}}(x) = \underline{FOU(\tilde{A})} \ldots\ldots\ldots .3.45b$$

As a result, Eqs. (3.45) can also be used to express an IT2FS as follows.

$$\tilde{A} = 1/\left[\underline{\mu}_{\tilde{A}}(x), \bar{\mu}_{\tilde{A}}(x)\right] \ldots\ldots\ldots\ldots\ldots\ldots 3.46$$

To obtain the lower and upper membership functions of the triangular MF from Eq (3.43) use the min and max operations

$$\bar{\mu}_{\tilde{A}}(x) = \min(t2trimf(x))$$

$$\underline{\mu}_{\tilde{A}}(x) = \max\left(t2trimf(x)\right) \ldots\ldots\ldots\ldots\ldots\ldots 3.47$$

When the input is a singleton fuzzy set (numeric value) with a core $x_i = x'$, the steps for processing a Mamdani type-2 fuzzy scheme, according to Eq. (3.40) and the extension principle (Prokopowicz et al., 2017) (Karnik & Mendel, 1998) are as follows:

a) Based on the input, calculate the degree of compatibility for each rule, which is an interval. $x'_i, 1/\left[\underline{\mu}_{F_1^l}(x'_1), \bar{\mu}_{F_2^l}(x'_2)\right]$ and $l = 1,2 \ldots\ldots, M$

b) Using t-norm functions, calculate the firing interval of each each l-rule $F^l(x'_1)$ by

$$F^l(x') = 1/\left[\underline{\mu}_{F_1^l}(x'_1) \times \underline{\mu}_{F_2^l}(x'_2), \bar{\mu}_{F_1^l}(x'_1) \times \bar{\mu}_{F_2^l}(x'_2)\right]$$

$$F^l(x') = 1/\left[\underline{f}^l, \bar{f}^l\right] \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.48$$

c) Apply the composition operation to each rule to get an induced consequent IT2FS equation 3.49.

$$B^l(yIx') = 1/\left[\underline{f}^l \star \underline{\mu}_{G^l}(y), \bar{f}^l \star \bar{\mu}_{G^l}(y)\right]$$

$$B^l(yIx') = 1/\left[\underline{\mu}_{B^l}(yIx'), \bar{\mu}_{B^l}(yIx')\right] \text{.....3.49}$$

d) To generate the overall interval IT2FS B, use an s-norm operation to combine all induced IT2FSs B^l.

$$B^l(yIx') = 1/\left[S_{l=1}^{M} \underline{\mu}_{B^l}(yIx'), S_{l=1}^{M} \bar{\mu}_{B^l}(yIx')\right]$$

$$B^l(yIx') = 1/\left[\underline{\mu}_B(yIx'), \bar{\mu}_B(yIx')\right] \text{.....3.50}$$

Using the centroid type-reducer process, construct a type-1 fuzzy set from an interval type-2 fuzzy set. The following are the equations for determining the most left and right centroids:

$$c_l(B) = \frac{\sum_{i=1}^{L} y_i \, \bar{\mu}_B(yIx') + \sum_{i=L+1}^{M} y_i \, \underline{\mu}_B(yIx')}{\sum_{i=1}^{L} \bar{\mu}_B(yIx') + \sum_{i=L+1}^{M} \underline{\mu}_B(yIx')} \text{ ..... 3.51}a$$

$$c_r(B) = \frac{\sum_{i=1}^{R} y_i \, \underline{\mu}_B(yIx') + \sum_{i=R+1}^{M} y_i \, \bar{\mu}_B(yIx')}{\sum_{i=1}^{R} \underline{\mu}_B(yIx') + \sum_{i=R+1}^{M} \bar{\mu}_B(yIx')} \text{ ..... 3.51}b$$

The Enhanced Karnik–Mendel Algorithm can be used to calculate the switch points L and R. (Karnik & Mendel, 1998). The type-1 fuzzy set has membership $\mu_B(y) = 1$ when $y \in [c_l(B), c_r(B)]$ and $\mu_B(y) = 0$ otherwise.

Defuzzification by centroid method yields a crisp output value $y_c(x')$ from the type-1 fuzzy set B, which is equivalent to finding the mean of c_l (B) and c_r (B) as follows in eqn 3.52.

$$y_c(\hat{x}) = \frac{c_l(B) + c_r(B)}{2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.52$$

This is a description of interval type-2 fuzzy systems; however, if the secondary membership function is not equal to 1, the complete type-2 model, known as a general type-2 fuzzy system, can be obtained. The operations are similar in this case, but the complexity rises since we now have to deal with a large number of interval type-2 fuzzy systems to approximate a general type-2 fuzzy system. As a result, general type-2 fuzzy systems have fewer works than interval type-2 fuzzy systems.

3.12.2 MATLAB type-2 fuzzy logic toolbox.

The new Release 2021a of the MATLAB and Simulink product families offers hundreds of new and updated features and functions. New capabilities in MATLAB include dynamic controls in live scripts as well as a new task for adding plots to live scripts without writing any code with capabilities of modelling type-2 fuzzy logic.

We converted type-1 inference system into type-2 inference system with matlab new release 2021a installed in the computer.in the matlab command window we typed

fisT2

Syntax:

fisT2 = convertToType2(fisT1)

ConvertToType2(fisT1) converts the type-1 fuzzy inference system fisT1 into a type-2 fuzzy inference system fisT2.

First created a type-1 fuzzy inference system (optimized traditional model). For this example, load the roboti.fis file.

Type-2 fuzzy inference system, returned as one of the following:

- *mamfistype2* object when fisT1 is a mamfis object
- *sugfistype2* object when fisT1 is a sugfis object

Except for Sugeno output membership functions, fisT2 had the same properties as fisT1, with the exception that each type-1 membership function was transformed to a type-2 membership function. Each type-2 membership function in fisT2 had upper membership function parameters that match those of the corresponding type-1 membership function in fisT1. fisT2 had default "karnikmendel" type reduction method.

The Type-2 Fuzzy Logic Toolbox features an intuitive point-and-click interface that guided us through the steps of developing FLS. We were guided through the steps of designing a type-2 fuzzy inference framework using graphical user interfaces (GUIs). Using basic logic rules, the toolbox allowed the user to create complex type-2 FLS. We could “inspect algorithms, modify source code, and add membership functions, defuzzification techniques, aggregation, implication, AND, OR, and type reduction methods”.

The Fuzzy Logic Toolbox's GUI editor and viewer allowed creating of rules, describing membership functions, and analysing the behaviour of a fuzzy inference system.

The File menu allowed us to create an Untitled type-2 fuzzy system with no variables and no rules, which was a singleton type-2 Mamdani FIS, a non-singleton type-2 Mamdani FIS, a type-2 Sugeno FIS, or a type-2 Sugeno FIS. We could also choose from the following options under the file menu:

- Open from disk... a system can be loaded from a.fis file on disk.
- Save to disk... saving the current system to a disk as a.fis file.

- Save to disk as…to save current system to disk, with the option of renaming or moving the file
- Open from workspace... to load a system from the workspace's FIS structure variable.
- Save to workspace... to save the system in the workspace to the currently named FIS structure variable
- Save to workspace as... to save the system to a specified FIS structure variable in the workspace.

- Close the window to close the GUI.

The Edit menu enabled adding a new input or output to the currently active device. Under the edit menu, the user may also remove a selected variable or undo the most recent update. The choices under the View menu were as follows:

- View surface... to invoke the Surface Viewer
- Edit rules... to open up the Rule Editor
- View rules... to launch the Rule Viewer
- Edit MFs... to invoke the Membership Function Editor

On the FIS Editor, there were six pop-up menus for changing the functionality of the six basic steps in the fuzzy implication process, as shown in Figure 3.22:

- And method: For a custom operation, select min, prod, or Custom.
- Implication method: For a custom process, choose one of the following methods: min, prod, or Custom. For Sugeno-style fuzzy inference, this option is not open.
- Or method: For a custom operation, select max or Custom.
- Aggregation method For a custom procedure, choose max, sum, or Custom as the aggregation tool. For Sugeno-style fuzzy inference, this option is not open.
- Type reduction method: For a custom process, choose one of the following reduction methods: center of sets, center of sums, centroid, height, adjusted

height, or Custom. For Sugeno-style fuzzy inference, this option is not open.

- Defuzzification method: Choose centroid or Custom for a custom operation for Mamdani-style inference. Choose wtaver (weighted average) or wtsum (weighted sum) for Sugeno-style inference (weighted sum).

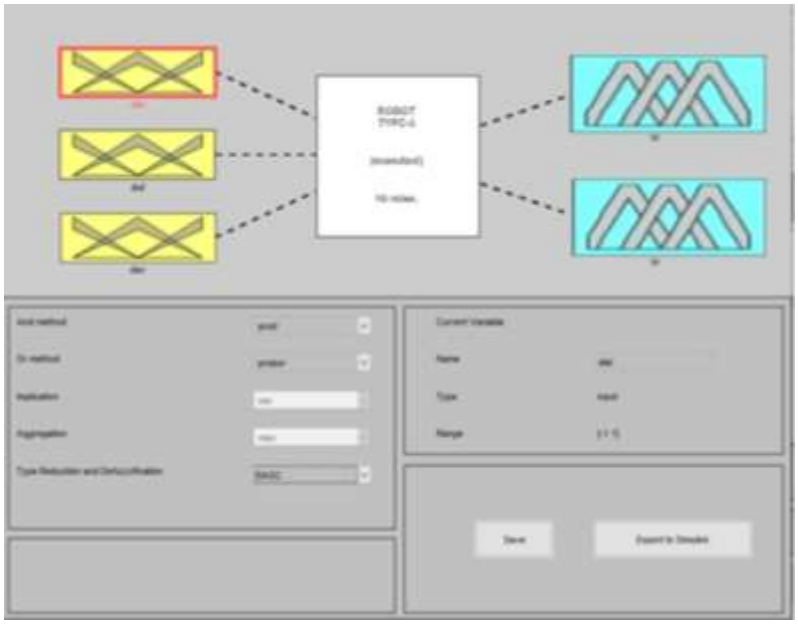


Figure 3.20: T2FLC editor.

The membership function editor enabled viewing and modifying the membership functions correlated with the FIS's input and output variables. There is a menu bar on the Membership Function Editor that allowed the user to open similar GUI tools, open and save systems. The Membership Function Editor's File menu was identical to the FIS Editor's File menu. The Edit menu contained options such as Add MF, Add custom MF, Remove current MF, Remove all MFs, and Undo. Under the View menu, you could edit FIS properties, edit rules, view rules, and view surface.

The rule editor, shown in Figure 3.23, allowed us to display and modify fuzzy rules. The Rule Editor had a menu bar that allowed the user to open similar GUI tools, open and save systems, and change the rule style. The menus in the Rule Editor were identical to those in the other editors.

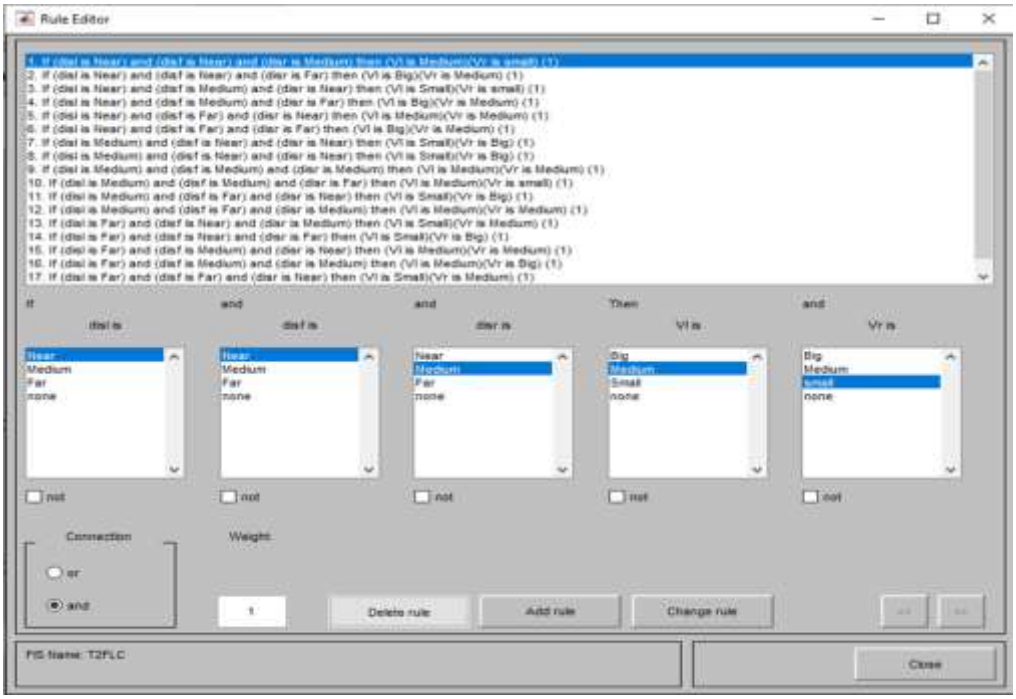


Figure 3.21: T2FLC rules editor.

The Rule Viewer helped the user to examine the behavior of a FIS to diagnose rule behavior or investigate the impact of changing input variables. The Rule Viewer's menu bar enabled the users to access similar GUI resources, open and save systems.

Surface Viewer for creating a 3-D surface from two input variables and a FIS output can also be displayed. The Surface Viewer's menu bar allowed us to access similar GUI resources, open and save systems. The Surface Viewer's menus were identical to those of the other editors.

Similarly we also accomplished simulation using older versions of MATLAB by downloading and Extracting IT2-FLS Matlab/Simulink Toolbox by Taskin and Kumbasar (Taskin & Kumbasar, 2015) RAR file into a convenient directory that was not part of the MATLAB distribution. Then used MATLAB pathtool command. Navigated to the directory we extracted into, down below the IT2FLSv1.0 directory to the IT2-FLS_Toolbox-v1.0 directory and added it to our MATLAB path. Then at the MATLAB command prompt, command we typed.

fuzzyt2

The general workflow of the IT2-FLSs toolbox is given below. The IT2-FLSs toolbox UI starts with fuzzyt2.m function and is used to create a text file with 'it2fis' extension. The created *.it2fis text file includes all information of the IT2FLSs designed by using the toolbox UI.

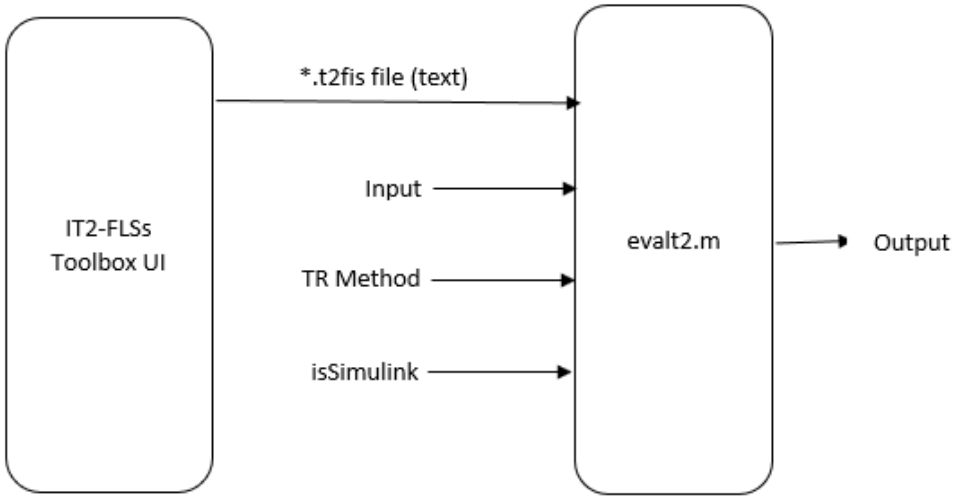


Figure 3.22: IT2FLSs model (Taskin & Kumbasar, 2015).

It was also possible to create or modify this *.it2fis text file manually. But, the text file had to be created or modified according to standard form. The standard structure of *.it2fis text file is given below in the appendix.4

Evaluating the output of an IT2-FLS from MATLAB

Firstly, the generated t2fis file was exported to the workspace as follows:

Matlab Syntax:

T2FIS=readt2fis(fileName);

Inputs

fileName - *.t2fis file created by UI or manually.

Then, the evalt2.m function was used to calculate the output for a given t2fis file

and current input.

Matlab Syntax:

y=evalt2(T2FIS,input, TRType,issimulink)

Inputs

T2FIS - *.t2fis file created by UI or manually.

input – Current input values of the IT2FLSs to calculate output.

TRmethod – Type Reduction/ Defuzzification method:

TRmethod=1-> KM

TRmethod=2-> EKM

TRmethod=3-> IASC

TRmethod=4-> EIASC

TRmethod=5-> EODS

TRmethod=6-> WM

TRmethod=7-> NT

issimulink– A variable for the Simulink library, must be set as “false” while operating in Matlab

Outputs:

y – Double – the defuzzified output value.

To understand the operation of the fuzzy logic toolbox refer to the study (Taskin & Kumbasar, 2015) and related literature.

3.12.3 Independent sample t-test

Independent Samples t-test was used to compare the means of two independent groups in order to determine whether there is statistical evidence that the associated population means are significantly different.

The variables used in this test were the dependent variable, or test variable and the independent variable, or grouping variable. The Independent Sample t-test is commonly used to test the statistical differences between the means of two groups, two interventions or two change scores. The null hypothesis (H_0) and alternative hypothesis (H_1) of the Independent Sample t-test can be expressed as:

Null hypothesis: $H_0 = \mu_1 = \mu_2$ .The means for the two models are equal.

Alternative hypothesis: $H_1 \neq \mu_1 \neq \mu_2$ .The means for the two models are not equal.

where the population means for groups 1 and 2 are μ1 and μ2 respectively. By subtracting μ2 from both sides of the equation, the second set of hypotheses can be derived from the first. An Independent Sample t-test statistic is denoted by the letter t. There are two forms of the test statistic for this test, depending on whether or not equal variances are assumed. SPSS produces both forms of the test and was used for this study.

The procedure to get the independent samples t-test was as follows:

- Calculate the value of Mean (M) = $\bar{x}$ and Variance (S^2) values from each group
- We'll also require the sample size (n)
- The calculated t-value was obtained using the t-value formula in equation 2.
- The critical t-value was determined using the t table.
- Determine whether the calculated t-value was greater than the critical t-value.

- If larger then we reject the null hypothesis otherwise smaller we do not reject the null hypothesis.

3.12.4 Equal variances assumed

When two independent samples are assumed to be drawn from populations with identical population variances (i.e., $\sigma_1^2 = \sigma_2^2$) , the test statistic *t* was computed using eqn.3.34.with pooled standard deviation by eqn.3.35.

$$t = \frac{\overline{x1} - \overline{x2}}{Sp\sqrt{\frac{1}{n1} + \frac{1}{n2}}} \dots\dots\dots\dots\dots\dots\dots\dots\dots 3.34$$

With

$$Sp = \sqrt{\frac{(n1-1)S_1^2 + (n2-1)S_2^2}{n1+n2-2}} \dots\dots\dots\dots\dots\dots 3.35$$

Where

$\overline{x1} = mean\ of\ first\ sample$

$\overline{x2} = mean\ of\ second\ sample$

$n1 = sample\ size\ of\ first\ sample$

$n2 = sample\ size\ of\ second\ sample$

$S1 = standard\ deviation\ of\ first\ sample$

$S2 = standard\ deviation\ of\ second\ sample$

$Sp = pooled\ standard\ deviation$

The calculated *t* value is then compared to the critical *t* value from the *t* distribution table with degrees of freedom $df = n_1 + n_2 - 2$ and chosen confidence level. The null hypothesis is rejected if the calculated t value is greater than the critical t value.

3.12.5 Equal variances not assumed

When two independent samples from populations with unequal variances are assumed (i.e., $\sigma_1^2 \neq \sigma_2^2$), the test statistic *t* is computed as eqn.3.36.

$$t = \frac{\overline{x1} - \overline{x2}}{\sqrt{\frac{S_1^2}{n1} + \frac{S_2^2}{n2}}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3.36$$

Where

$\overline{x1} = mean\ of\ first\ sample$

$\overline{x2} = mean\ of\ second\ sample$

$n1 = sample\ size\ of\ first\ sample$

$$n2 = sample\ size\ of\ second\ sample$$

$$S1 = standard\ deviation\ of\ first\ sample$$

$$S2 = standard\ deviation\ of\ second\ sample$$

The calculated t value was then compared to the critical t value from the t distribution table with degree of freedom calculated using eqn.3.37.

$$df = \frac{\left(\frac{S_1^2}{n1} + \frac{S_2^2}{n2}\right)^2}{\frac{1}{n1-1}\left(\frac{S_1^2}{n1}\right)^2 + \frac{1}{n2-1}\left(\frac{S_2^2}{n2}\right)^2} \ldots\ldots\ldots\ldots 3.37$$

If the calculated t value is greater than the critical t value, the null hypothesis is rejected.

3.12.6 Effect size (Cohen's d)

Cohen's d is a standardized measure of effect size. Cohen argued that the standard deviation of either group could be used when the variances of the two groups are homogeneous Cohen (1988) defined d as the difference between the means, M1 - M2, divided by the standard deviation, of either group. Cohen's *d* can be calculated as the difference between the means divided by the pooled SD:

$$Cohen's\ d \frac{mean\ difference}{standard\ deviation}$$

Or

$$= \frac{M_1 - M_2}{pooled\ standard\ deviation.}$$

$$= \frac{\mu_1 - \mu_2}{\sigma}$$

$$= \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{(n_1-1)S_1^2 + (n_2-1)S_2^2}{n_1+n_2-2}}} \dots\dots\dots\dots 9 \dots$$

3.13 Conclusion

This chapter describes the research methods and techniques used to gather and analyze data. It also gives an overview of how the research was conducted and how the information gathered was used to develop the proposed strategy. The tools that were used to create the model are also discussed. In addition, a virtual environment was developed. Nine sensors mounted on the robot's chassis were used and combined three of them to create three zones. The type-1 Mamdani fuzzy logic-controlled obstacle avoidance robot received the calculated mean value as input. Nine input sensors yield 9^3 =729 rules, but fusion reduced the input to 3^3=27 rules. Fuzzy Similarity Measure between fuzzy rules technique was further used to reduced rules. The most commonly used MFs, which were tested in a similar environment, are triangular trapezoidal and gaussian. An ANOVA hypothesis test was used with a completely randomized design to see if the mean difference between shapes was significant. A post hoc test was used to determine which groups were distinct from one another. Independent Samples t-test was used to compare the developed type-1 Mamdani fuzzy logic controller with its corresponding type-2 to see if there was statistical evidence that the associated population means were significantly different.

Cohen's d was calculated as a standardized measure of effect size by dividing the difference between the means by the pooled standard deviation for effect size. Cohen 1992 guidelines for effect of size d = **0.20** indicates a **small** effect, d = **0.50** indicates a **medium** effect and d = **0.80** indicates a **large** effect were used.

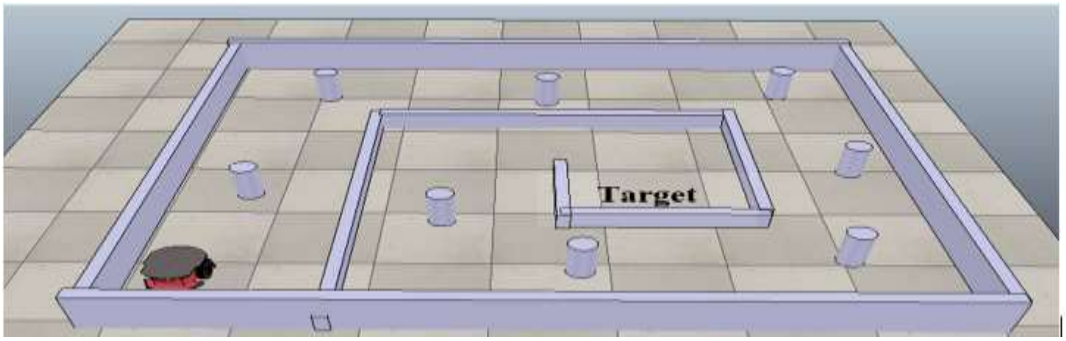
CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Simulation

Simulation of an autonomous mobile robot navigating in an environment with obstacles was performed using V-REP and MATLAB software. A scene mimicking complex environments with twelve obstacles was modeled with a two-wheeled robot selected in V-REP. A type-1 Mamdani fuzzy logic controller (T1MFLC) was developed to optimize traditional models by sensor fusion and Fuzzy rules reduction to effectively aid robots to navigate in a static unknown environment. Triangular, trapezoidal and gaussian input MFs of the controller was tested in a similar navigational environment to analyze their effects. The developed T1MFLC model was transformed to its corresponding T2MFLC and performance was compared. The time in seconds for the robot to reach the target from the starting point was used in evaluating the results.

Several experimental runs were carried out to validate the model's effectiveness. Figure 4.1 shows the robotic simulation field. The robot avoided obstacles and walls to reach the target. https://www.youtube.com/watch?v=8Em1bVukQU4. Shows a



scene in VREP during the data collection exercise.

Figure 4.1: Robot navigation field

The walls and obstacles were made dynamic and respondable by using the toolbar to make them interactive with a mobile robot. The robot successfully avoided obstacles and navigated to the target point in all trials, which was quite encouraging.

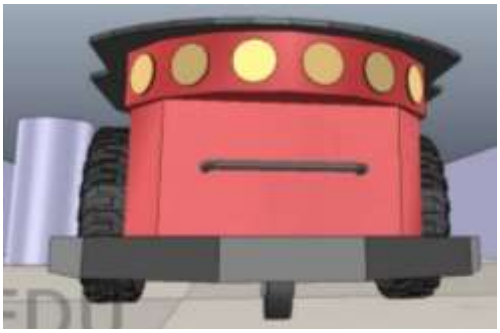


Figure 4.2: Unicycle robot (pioneer P3dx)

V-REP Robot model pioneer P3dx a unicycle type was used with ultrasonic sensors mounted on the chassis as shown in Figure 4.2. The wheels were effectively driven independently by Pulse width modulation (PWM) dc motors to accomplish the motion and alignment aided by rules from the fuzzy model.

To avoid obstacles, the mobile robot responded appropriately according to the obstacle distance detected by transducers. An expert human driver formulated rules and a fuzzy control methodology transformed informal rules into precise control strategy. The results include the developed model, 18 fuzzy logics rules using triangular membership functions for fuzzification, and a type-2 fuzzy logic controller dealing with the limitations of type-1 controllers.

4.1 Fuzzy logic model

The developed model has three inputs as presented in figure 4.3. The input disl

represents the left sensing region disf for the front section and the left region scanned by disr. The output variables are the velocities Vl and Vr for left velocity and right velocity respectively.

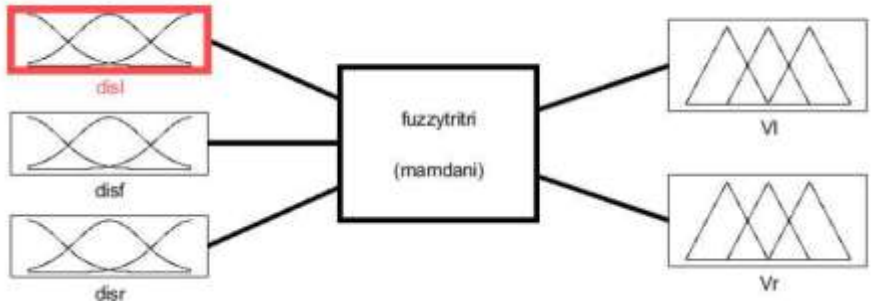


Figure 4.3: Mamdani T1MFLC for an autonomous mobile robot.

The model would become more complex if all nine ultrasonic sensors were used as input data to the fuzzy logic controller. Sensor fusion provided the advantage of quick response and allowing the mobile robot to react to an unknown environment effectively in real-time. This method made the model less computationally intensive and thus can use less powerful processors for practical applications. The developed model is shown in Figure 4.4 illustrating the overall implementation strategy.

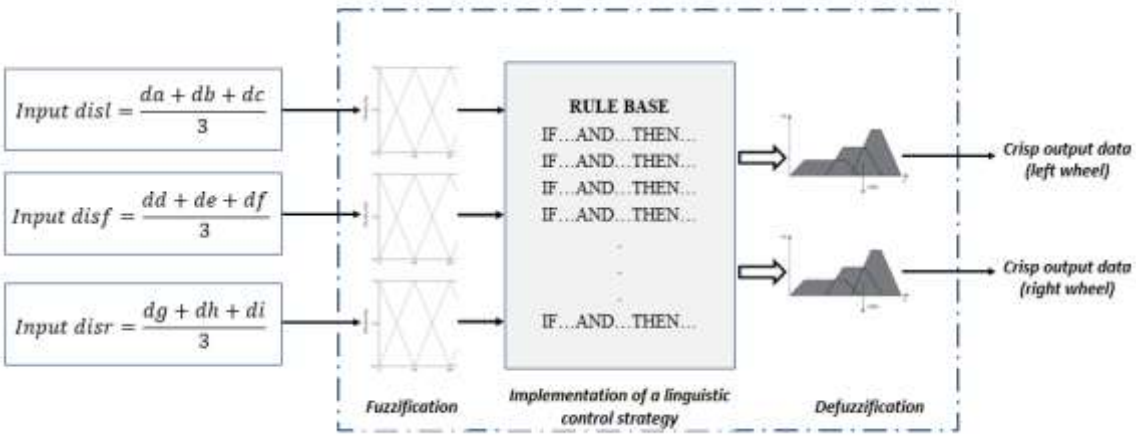


Figure 4.4: Developed T1MFLC model.

4.2 Sensory data fusion

In this thesis, we put together inputs from multiple transducers to form a single image of the environment around. It is anticipated that the results of this method make the model more effective due to the balanced strengths of the different sensors than previous studies. The traditional studies used controller with n = 3 and m = 3 thus the total number of fuzzy rules were $k = m^n = 3^3 = 27$. Figure 4.5 shows an

implementation of sensory data fusion used in our study. Figure 4.5 illustrates the merging of the sensory data. Sensor fusion was used to improve single sensor measurements by reducing sensory deprivation, limited spatial coverage, limited temporal coverage, imprecision, and uncertainties

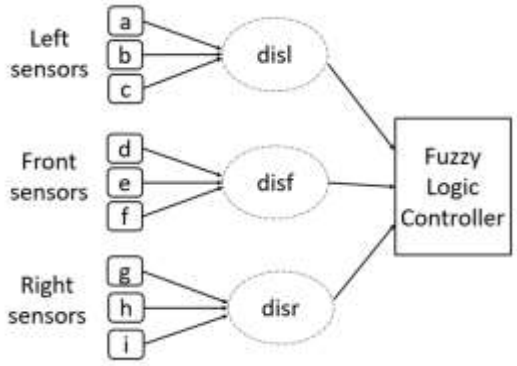


Figure 4.5: Sensor fusion.

4.3 Mobile robot fuzzy rules reduction

When designing a fuzzy controller for a complex system, several measurable outputs and actuating input variables are used. Each variable is represented by a finite number m of linguistic sets, with the total number of rules being indicated by the total number of rules. With nine inputs (sensors) m = 9 and three linguistic sets n = 3

then k = m^n = 9^3 = 729 would have been the total number of fuzzy rules. The number of input variables increases exponentially the size of the rule base in complex fuzzy control systems. The range for input parameters range NEAR=0 to 0.4 meters, MEDIUM =0.1-0.9M and FAR=0.6-1M, the speed for the left and right wheels represented by SMALL=0-80cm/s, MEDIUM= 20-180cm/s and BIG=120-200cm/s.

Fuzzy similarity measure played an important role in further reducing twenty-seven if-then rules to eighteen rules using similarity measure for optimum results and enhancement of computational time. Table 4.1 rules extracted from an expert driver.

Table 4.1: Expert rules subjected to reduction.

Case	disl	disf	disr	vl	vr	DS	>50% similar < 50% dissimilar
1	N	N	N	S	S	3/5	Similar
2	N	N	M	M	S	60%	
3	N	N	F	B	M	1/5	dissimilar
4	N	M	N	S	S	20%	
5	N	M	M	M	S	2/5	dissimilar
6	N	M	F	B	M	40%	
7	N	F	N	M	M	2/5	dissimilar
8	N	F	M	B	S	40%	
9	N	F	F	B	M	0/5	dissimilar
10	M	N	N	S	B	0%	
11	M	N	M	S	S	3/5	Similar
12	M	N	F	B	S	60%	
13	M	M	N	S	B	2/5	dissimilar
14	M	M	M	M	M	40%	
15	M	M	F	M	S	1/5	dissimilar
16	M	F	N	S	B	20%	
17	M	F	M	M	M	3/5	Similar
18	M	F	F	B	M	60%	
19	F	N	N	S	B	3/5	Similar
20	F	N	M	S	M	60%	
21	F	N	F	S	B	1/5	dissimilar
22	F	M	N	N	M	20%	
23	F	M	M	M	B	4/5	Similar
24	F	M	F	M	B	80%	

25	F	F	N	S	M	2/5	dissimilar
26	F	F	M	M	B	40%	
27	F	F	F	B	B		

Using Similarity Measures in Fuzzy Rule Base Simplification twenty-seven were effectively reduced to eighteen. Further reduction done is shown in table 4.2

Table 4.2: Similarity measure rules reduction

Case	disl	disf	disr	vl	vr	DS	>50% similar < 50% dissimilar
1	N	N	M	M	S	2/5	dissimilar
2	N	N	F	B	M	40%	
3	N	M	N	S	S	3/5	similar
4	N	M	M	M	S	60%	
5	N	M	F	B	M	2/5	dissimilar
6	N	F	N	M	M	40%	
7	N	F	M	B	S	3/5	similar
8	N	F	F	B	M	60%	
9	M	N	N	S	B	3/5	similar
10	M	N	M	S	S	60%	
11	M	M	N	S	B	2/5	dissimilar
12	M	M	M	M	M	40%	
13	M	M	F	M	S	1/5	dissimilar
14	M	F	N	S	B	20%	
15	M	F	M	M	M	2/5	dissimilar
16	F	N	M	S	M	40%	
17	F	N	F	S	B	1/5	dissimilar
18	F	M	N	N	M	20%	
19	F	M	M	M	B	1/5	dissimilar
20	F	F	N	S	M	20%	
21	F	F	M	M	B	3/5	similar
22	F	F	F	B	B	60%	

On this basis, we presented the respective eighteen if- then rules adopted for obstacle avoidance navigation as below:

Rule 1: If (disl is N) and (disf is N) and (disr is M), then (vl is M) and (vr is S).

Rule 2: If (disl is N) and (disf is N) and (disr is F), then (vl is B) and (vr is M).

Rule 3: If (disl is N) and (disf is M) and (disr is N), then (vl is S) and (vr is S).

Rule 4: If (disl is N) and (disf is M) and (disr is F), then (vl is B) and (vr is M).

Rule 5: If (disl is N) and (disf is F) and (disr is N), then (vl is M) and (vr is M).

Rule 6: If (disl is N) and (disf is F) and (disr is F), then (vl is B) and (vr is M).

Rule 7: If (disl is M) and (disf is N) and (disr is N), then (vl is S) and (vr is B).

Rule 8: If (disl is M) and (disf is M) and (disr is N), then (vl is S) and (vr is B).

Rule 9: If (disl is M) and (disf is M) and (disr is M), then (vl is M) and (vr is M).

Rule l0: If (disl is M) and (disf is M) and (disr is F), then (vl is M) and (vr is S).

Rule 11: If (disl is M) and (disf is F) and (disr is N), then (vl is S) and (vr is B).

Rule l2: If (disl is M) and (disf is F) and (disr is M), then (vl is M) and (vr is M).

Rule 13: If (disl is F) and (disf is N) and (disr is M), then (vl is S) and (vr is M).

Rule 14: If (disl is F) and (disf is N) and (disr is F), then (vl is S) and (vr is B).

Rule 15: If (disl is F) and (disf is M) and (disr is N), then (vl is M) and (vr is M).

Rule 16: If (disl is F) and (disf is M) and (disr is M), then (vl is M) and (vr is B).

Rule 17: If (disl is F) and (disf is F) and (disr is N), then (vl is S) and (vr is M).

Rule 18: If (disl is F) and (disf is F) and (disr is F), then (vl is B) and (vr is B).

Rule Viewer’s roadmap of the fuzzy inference process is displayed in Figure 4.7. The first rule's antecedent and consequent were represented by the five small plots across the top of the figure. Each rule is a row of plots, and each column is a variable. The membership functions referred to by the antecedent are shown in the first three columns of the plots or the "if" as well as "and" part of each rule. The membership functions referenced by the consequent or then-part of each rule are shown in the fourth and fifth columns of plots. This was created in order to obtain an approximate interpretation of the entire fuzzy inference process all at once. By moving the line indices corresponding to the inputs around, the readjustment and computing of new outputs could be observed.

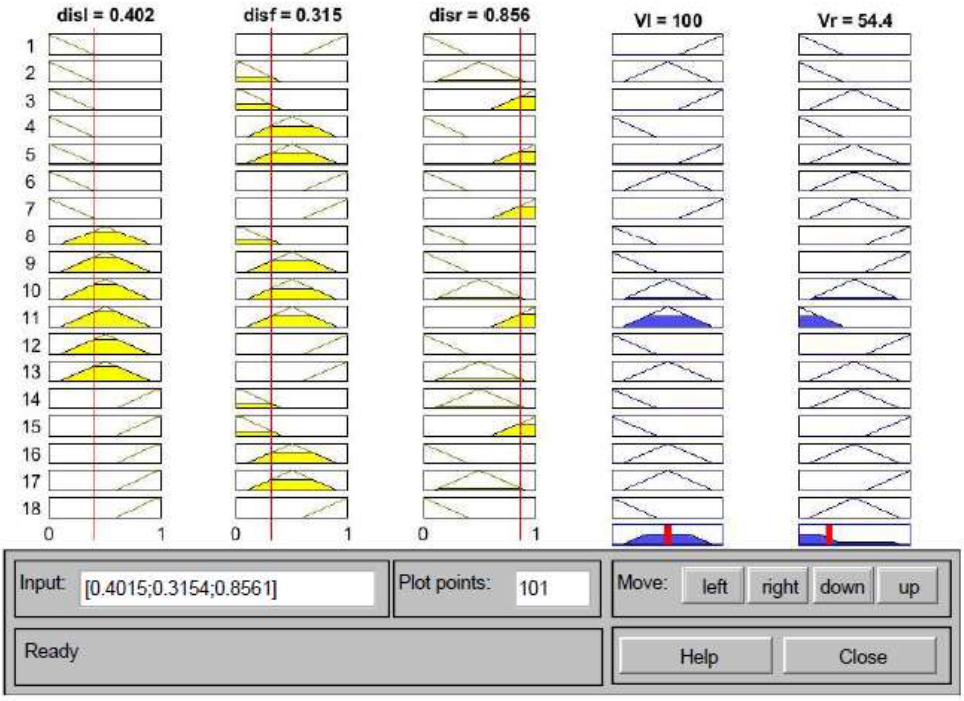


Figure 4.6: T1MFLC Fuzzy inference system

The surface plot viewer was very useful during the experiment. It was beneficial in determining desired response values and operating conditions. Figure 4.7 depicts the wheel response concerning to two predictor variables, front, left and right sensors. The z-axis represents the response values. The peaks and valleys correspond to x and y combinations, resulting in local maxima and minima.

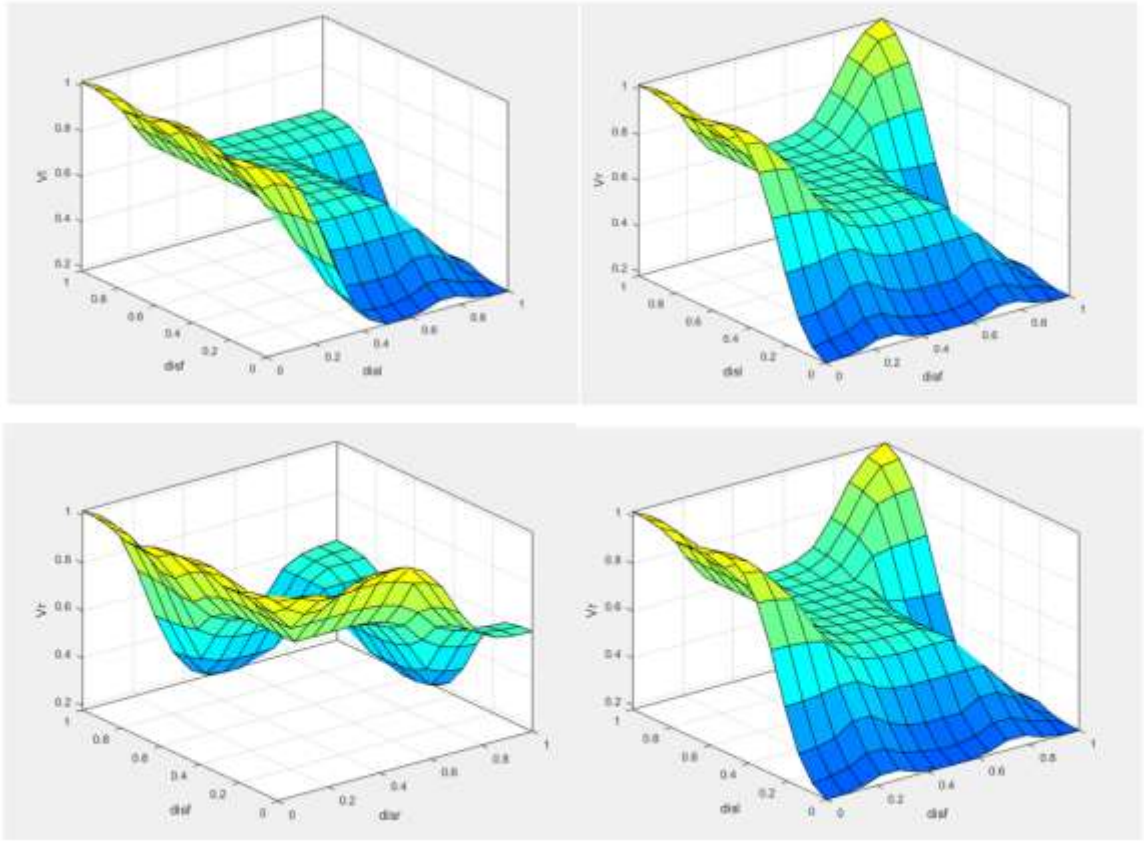


Figure 4.7: Surface plot viewer

4.4 Effects of MFs on robot performance

The main focus here was to evaluate the effect of changing the shape of input MFs with triangular membership function as the output and COG defuzzification method on the obstacle avoidance T1MFLC. The testbed scene had twelve cylindrical objects as obstacle in the environment to simulate a complicated setting. Table 4.3 shows the results of fourteen repeated runs of experiments with T1MFLCand figure 4.8 displays the bar graph of the same.

Table 4.3: Time taken to reach the target in a simulated environment

Test	Fuzzy logic input MFs. (time in sec)			Remarks
	Gaussian	Trapezoidal	Triangular	
1	230	225	221	Reached target
2	223	224	222	Reached target
3	230	226	220	Reached target
4	232	223	225	Reached target
5	225	222	221	Reached target
6	225	223	224	Reached target
7	228	224	221	Reached target
8	225	234	222	Reached target
9	224	226	225	Reached target
10	223	223	224	Reached target
11	228	222	223	Reached target
12	228	223	221	Reached target
13	225	226	224	Reached target
14	234	227	223	Reached target

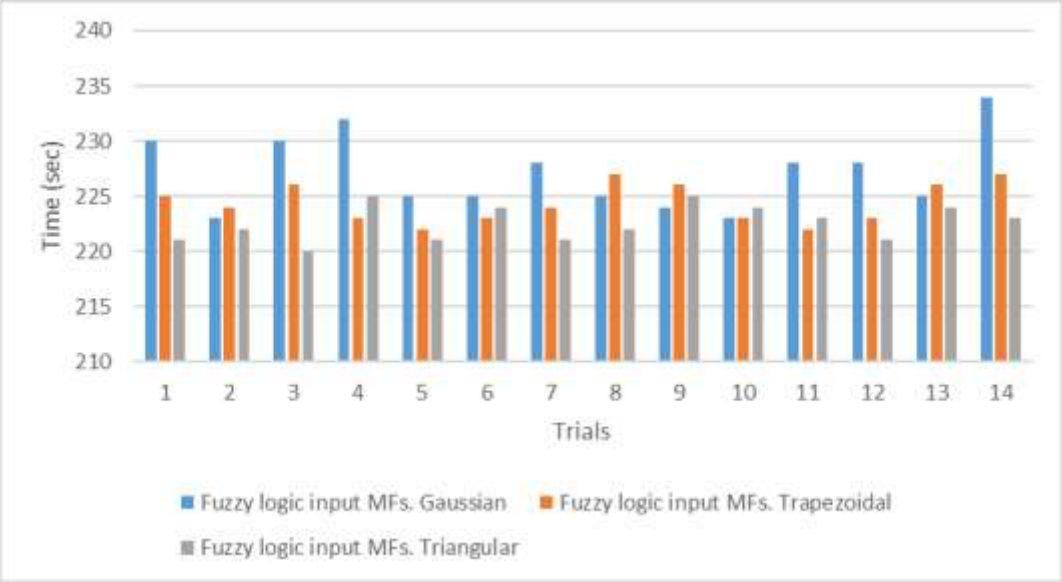


Figure 4.8: Navigation bar graph.

Table 4.4 shows the results Mean time taken to reach the target, standard deviation and standard error of mean. Figure 4.8 is a bar graph to help display the findings. Our results shows that the triangular shape took the shortest time of 222.57 seconds on

average. Gaussian shape took the longest time and the variation is biggest in Gaussian and smallest in triangular.

Table 4.4: Mean time taken to reach the target

logic	Mean	Std. Deviation	Std. Error of Mean
Gaussian	227.14	3.416	.913
Trapezoidal	224.86	3.085	.824
Triangular	222.57	1.651	.441

To determine whether the mean difference across shapes was significant, an ANOVA hypothesis test was performed using a completely randomized design.

Table 4.5: ANOVA computation.

	Gaus x_1	Trap x_2	Tria x_3	$(x_1-\bar{x}_1)^2$	$(x_2-\bar{x}_2)^2$	$(x_3-\bar{x}_3)^2$
	230	225	221	8.18	0.10	7.46
	223	224	222	17.14	8.74	0.32
	230	226	220	8.18	1.30	6.60
	232	223	225	23.62	3.46	5.90
	225	222	221	4.58	8.18	23.46
	225	223	224	4.58	6.46	2.04
	228	224	221	0.74	0.74	2.46
	225	234	222	4.58	83.54	12.32
	224	226	225	9.86	1.30	5.90
	223	223	224	17.14	3.46	2.04
	228	222	223	0.74	8.18	17.1
	228	223	221	0.74	4.46	2.46
	225	226	224	4.58	10.30	32.02
	234	227	223	47.06	7.58	4.18
ng	14	14	14	Σ=151.72	Σ=147.8	Σ=124.26
G	$14+14+14=42$					
$\bar{x}_g$	227.14	224.86	222.57			
$\bar{x}_G$	224.86					
	K=3					

$H_0 = \mu_1 = \mu_2 = \mu_3$ The null hypothesis states input membership functions have no effect on time taken by robot to move from start to the target.

$H_1 = \mu_1 \neq \mu_2 \neq \mu_3$ is an alternative hypothesis implying that there is a difference in the time required for the robot to reach the target. Corresponding to alpha = $\alpha = 0.05$.

F statistics was calculated using Equation 4.1 below.

$$F = \frac{M_{SSB}}{M_{SSW}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.1$$

Where:

M_{SSB} = between groups mean sum of squares.

M_{SSW} = within groups mean sum of squares.

Within groups sum of squares equation 4.2 is used as the formula.

$$M_{ssw} = \frac{\sum g \epsilon G (x - \bar{x}_g)^2}{n-k} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 4.2$$

Where:

G=the total number all variables

$\bar{x}$ = mean of group

n= number in each group.

k= number of groups.

$$M_{ssw} = \frac{151.72 + 147.8 + 124.26}{42 - 3}$$

$$= \frac{423.789}{39}$$

$$= 10.87$$

Sum of squares equation 4.3 was used to calculate between groups.

$$M_{SSB} = \frac{\sum g \in G\, n_g (x - \bar{x}_g)^2}{k-1} \ldots\ldots\ldots\ldots\ldots\ldots 4.3$$

$$= \frac{n_1(\bar{x}_1 - \bar{x}_G)^2 + n_2(\bar{x}_2 - \bar{x}_G)^2 + n_3(\bar{x}_3 - \bar{x}_G)^2}{k - 1}$$

Substituting with the values

$$M_{SSB}$$
$$= \frac{14(227.86 - 224.86)^2 + 14(224.86 - 224.86)^2 + 14(222.57 - 224.86)^2}{3 - 1}$$

$$= \frac{126 + 0 + 73.42}{2}$$

$$= \frac{199.42}{2} = 99.71$$

One way ANOVA is a mathematical generalization of the two-sample t test. The F statistic compares the variability between groups to the variability within the equation of the group. F- statistics can be obtained by substituting equation 4.1.

$$F = \frac{99.71}{10.87} = 9.1729 \ \ (statictics)$$

Referring to the F distribution in the appendix table for $\alpha = 0.05$.

$$df_b = k - 1 = 3 - 1 = 2 \ \ (column)$$

$$df_W = n - k = 42 - 3 = 39 \ \ (raw)$$

The critical Values of the F-Distribution: α = 0.05 from the table is $F_{critical} = 3.238$

$$F_{statistics} > F_{critical}$$

The hypothesis is rejected because $9.1729 > 3.238$.

To validate using SPSS statistics ANOVA test results are given in Table 4.6.

Table 4.6: ANOVA test

	Sum of Squares	df	Mean Square	F	P value
Between Groups	146.286	2	99.71	9.173	.001
Within Groups	310.857	39	10.87		
Total	457.143	41			

The results in table 4.6 show that there was a significant difference in the mean because the p-value was less than 0.05. Our findings show that MFs had a significant impact on robot navigation at the p<.05 level for the three conditions.

$$[F(2,39) = 9.173, p = 0.001]$$

The ANOVA test reveals an overall difference between the groups, but it did not reveal which specific groups differed, so we conduct post-hoc tests. Post hoc tests attempts to control the experimentwise error rate (usually alpha = 0.05) in the same manner that the one-way ANOVA is used instead of multiple t-tests. Post hoc tests are termed *posteriori* tests; that is, performed after the event.

Tukey's post-hoc test first determines the differences in the means of all the groups. Then compares the difference score to a critical value to confirm if the difference is significant. The critical value, in this case, is the HSD (honestly significant difference) . It is the point when a mean difference becomes honestly significantly different.

Tukey's honestly significant difference (HSD) post hoc test was performed using equation 4.4.

$$HSD = q\sqrt{\frac{MS_{within}}{n}} \quad \ldots\ldots\ldots\ldots 4.4$$

n = number of values we are dealing with in each group =14

To find q we use the studentized range statistic (q) table corresponding to alpha = .05

$df_W = n - k = 42 - 3 = 39$ (Degree of freedom within)

K= 3 (number of treatments-groups)

To find "q" or the studentized range statistic. On the table 'k' or the number of groups is found along the top, and degrees of freedom within is down the side

The value of q in the table appendix 2 =3.44

We then computed all possible differences between group means:

1. $\bar{x}_1 - \bar{x}_2 = 227.14 - 224.86 = 2.28$
2. $\bar{x}_1 - \bar{x}_3 = 227.14 - 222.57 = 4.57$
3. $\bar{x}_2 - \bar{x}_3 = 224.86 - 222.57 = 2.29$

We are only concerned with the absolute difference, so we ignore any negative signs.

We then compute the HSD using equation 4.4.

$$HSD = 3.44\sqrt{\frac{7.95}{14}} = 3.03$$

Now we compare the difference scores we computed with the HSD value. If the difference is larger than the HSD, then we say the difference is significant.

Groups 1 and 2 do not differ

Groups 1 and 3 differ

Groups 2 and 3 do not differ

To verify that the difference was significant across all the groups was experimentally investigated by post ANOVA test using least square difference (LSD) test and results by SPSS statistics are presented in Table 4.6.

Table 4.7: Multiple Comparisons

Dependent Variable: time		LSD				
(I) logic	**(J) logic**	**Mean Difference (I-J)**	**Std. Error**	**Sig.**	**95% Confidence Interval**	
					Lower Bound	**Upper Bound**
Gaussian	trapezoidal	2.286*	1.067	.038	.13	4.44
	triangular	4.571*	1.067	.000	2.41	6.73
Trapezoidal	gaussian	-2.286*	1.067	.038	-4.44	-.13
	triangular	2.286*	1.067	.038	.13	4.44
Triangular	gaussian	-4.571*	1.067	.000	-6.73	-2.41
	trapezoidal	-2.286*	1.067	.038	-4.44	-.13

*. The mean difference is significant at the 0.05 level.

A post hoc test revealed that the time to reach the target was statistically significant between triangular and Gaussian MFs (p = 0.000), between triangular and Trapezoidal MFs (p = 0.038) and between Gaussian and trapezoidal MFs. (p = 0.038). From the results, it can be seen that the difference is significant across all the groups since the p value is less than 0.05 between any two groups. However, when comparing all the three MFs it must be pointed out that the triangular shape is the most efficient.

4.5 T1MFLC and T2MFLC comparison

In this experiment, the developed Mamdani T1MFLC (model M) and T2MFLC (model K) were compared based on the mean time for the robot to reach the target in a simulated environment. The testbed was Mamdani inference engine with triangular membership functions. Fourteen rans were performed for each model in the same environment, the experimental results are tabulated in table 4.8.

Table 4.8: Model comparison.

Trial	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Model- K	215	223	219	213	223	226	218	227	221	225	220	216	213	219
Model- M	227	221	221	219	219	224	227	226	230	219	221	230	224	225

The plot in figure 4.12 represents the line graph performance of the models. At some instances it can be seen either model performs better than the other further analysis was done to test the performance hypothesis.

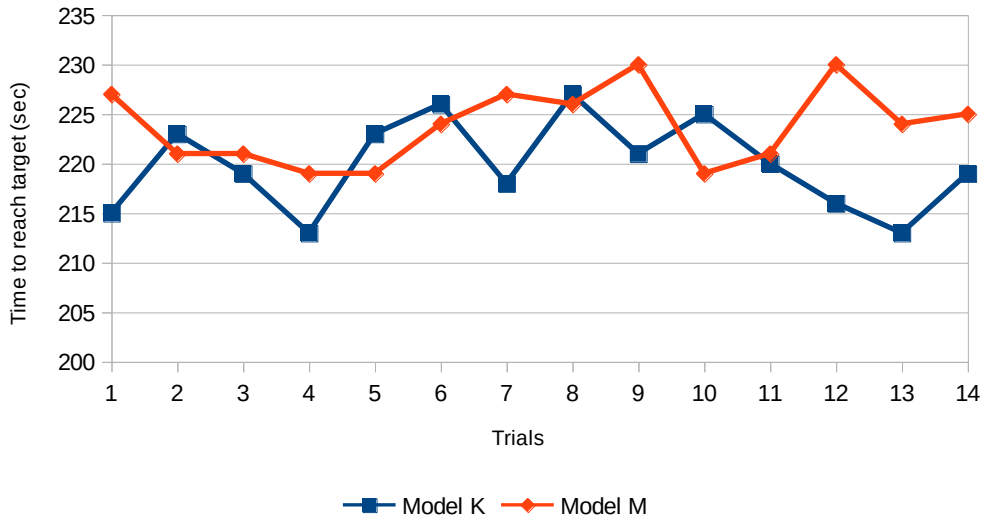


Figure 4.9: Model comparison.

An independent-samples t-test was used to compare performance based on the time taken to reach the destination between the developed models (K) and (M). Model K had a mean time of 219.79 seconds, while Model M has a mean time of 223.79. It is a statistical inference test that determines whether there is a statistically significant difference between the means of two groups. An independent t-test is used to determine whether T1MFLC (M) and T2MFLC (K) take the same amount of time to navigate obstacles and reach their destination in the same scene. In the overall procedure, two t values were used: calculated t and critical t. If the calculated t-value is greater than the critical t-value, the null hypothesis is rejected. The null hypothesis is that the population means are all the same.

$$H_0 : \mu_1 = \mu_2$$

Where:

$\mu = population$

$\mu_1 = T1MFLC\ mean$

$\mu_2 = T2MFLC\ mean$

Alternative hypothesis

The population μs are unequal.

$H_1 : \mu_1 \neq \mu_2$

The independent t-test formula is shown in equation 4.5.

$$t = \frac{M_1 + M_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}} \dots\dots\dots\dots\dots 4.5$$

Where :

$M_1 = \bar{x}_1 = mean\ \ group\ 1$

$M_2 = \bar{x}_2 = mean\ group\ 2$

$S_1^2 = variance\ group\ 1$

$S_2^2 = variance \ \ group\ 2$

$n_1 = sample\ size\ group\ 1$

$n_2 = sample\ size\ \ group\ 2$

The denominator in the equation $\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}$ is the standard error of the difference between means.

Table 4.9: Independent samples t-test calculation.

Model K	$x - \bar{x}_1$	$(x - \bar{x}_1)^2$	Model M	$x - \bar{x}_2$	$(x - \bar{x}_2)^2$
215	-4.86	23.62	227	3.21	10.30
223	3.14	9.86	221	-2.79	7.78
219	-0.86	0.74	221	-2.79	7.78
213	-6.86	47.06	219	-4.79	22.94
223	3.14	9.86	219	-4.79	22.94
226	6.14	37.70	224	0.21	0.04
218	-1.86	3.46	227	3.21	10.30
227	7.14	50.98	226	2.21	4.88
221	1.14	1.30	230	6.21	38.56
225	5.14	26.42	219	-4.79	22.94
220	0.14	0.02	221	-2.79	7.78
216	-3.86	14.90	230	6.21	38.56
213	-6.86	47.06	224	0.21	0.04
219	-0.86	0.74	225	1.22	1.49
$\bar{x}_1$=219.86		Σ=273.72	$\bar{x}_2$ =223.79		Σ =196.33
n=14		$S^2 = \frac{273.72}{n-1}$	n=14		$S^2 = \frac{196.33}{n-1}$
		$S^2 = 21.06$			$S^2 = 15.10$

Thus, with means, variance and sample sizes we can then calculate t-value using the equation

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots 4.6$$

$$= \frac{219.86 - 223.79}{\sqrt{\frac{21.06}{14} + \frac{15.1}{14}}}$$

$= 2.445$

To get the t-critical calculate the degree of freedom (df) and specify the alpha level (α) (0.05). we use the df and α to identify the critical t-value in the table.

$$df = n_1 + n_2 - 2$$

$$= 14 + 14 - 2$$

$$df = 26$$

The critical t-value in the t-table is 2.056 when the df (26) and α (0.05) is used.

As a result of the calculated value 2.445 being greater than the critical t-value 2.056, the null hypothesis can be rejected.

Table 4.8 summarizes the results, which show that model K has a lower mean than model M using SPSS statistics.

Table 4.10: Model comparison mean summary

	models	N	Mean	Std. Deviation	Std. Error Mean
time	Model K	14	219.79	4.509	1.205
	Model M	14	223.79	3.886	1.039

Table 4.11: Independent Samples Test.

	t-test for Equality of Means						
	t	**df**	**P value**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
						Lower	**Upper**

time	Equal variances assumed	-2.514	26	.018	-4.000	1.591	-7.270	-.730
	Equal variances not assumed	-2.514	25.446	.019	-4.000	1.591	-7.274	-.726

Where

t = The calculated test statistic is t.

df = is the degrees of freedom

The *p-value* corresponding to the given test statistic and degrees of freedom

The *mean difference* is the difference between the sample means; it is also the numerator of the test statistic.

Std. Error Difference is the standard error; it also corresponds to the denominator of the test statistic.

The scores for Model K (M=219.79, SD=4.509) and Model M (M=223.79, SD=3.886) conditions differed significantly; t(26)=2.514, p = 0.018. Table 4.9 shows that the mean of the two models differs by a significant amount. These findings indicate that when used to navigate an autonomous wheeled mobile robot, the Model K and Model M behave differently. Our findings specifically indicate that Model K is more effective than Model M.

Cohen's d was used to check the effect size of the two means by dividing the difference between the means by the pooled standard deviation:

$$Cohen's\ d \frac{mean\ difference}{standard\ deviation}$$

Or

$$= \frac{M_1 - M_2}{pooled\ standard\ deviation.}$$

$$= \frac{\mu_1 - \mu_2}{\sigma}$$

$$= \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{(n_1-1)S_1^2 + (n_2-1)S_2^2}{n_1+n_2-2}}} \ldots\ldots\ldots\ldots 4.7$$

Cohen 1992 provides some guidelines for the effect of size as below.

- d = 0.20 indicates a minor effect.
- d = 0.50 indicates a medium effect and
- d = 0.80 indicates a large effect.

$$Group\ K\ mean\ (\bar{x}_1) = 219.86 \quad S_1^2 = 21.06 \quad n_1 = 14$$

$$Group\ M\ mean\ (\bar{x}_2) = 223.79 \quad S_2^2 = 15.10 \quad n_2 = 14$$

Using equation 4.7

$$cohen's\ d = \frac{219.86 - 223.79}{\sqrt{\frac{(14-1)21.06 + (14-1)15.10}{14+14-2}} +}$$

$$= \frac{-3.93}{\sqrt{18.08}}$$

$= 0.924 \quad (indicating\ a\ large\ effect.)$

CHAPTER FIVE

SUMMARY, CONCLUSIONS, AND FUTURE WORK

5.1 Introduction

This chapter presents a summary of our findings, conclusions, and plans for future research on a fuzzy logic model for obstacles avoidance robot in a static unknown environment.

5.2 Summary

Autonomous mobile robots have received a lot of attention from researchers, and many robots are currently being developed. Typically, robots are equipped with sensors that allow them to recognize their surroundings. They are, however, generally unreliable due to the issue of dealing with ambiguity in the environment. Fuzzy logic is widely regarded as a useful tool for dealing with uncertainty caused by imprecise knowledge. For wheeled mobile robot navigation, many researchers emphasize the use of a three-input FLC with 27 rules. While this approach is appealing, it fails to account for uncertainties, which makes avoiding obstacles difficult.

The focus of this thesis was to improve the traditional model (T1MFLC) by efficiently increasing the number of sensors, reducing fuzzy rules, and tuning membership functions. The optimized model was then compared to a corresponding T2MFLC. Experiments were carried out to simulate the system using commercially available V-REP and MATLAB software.

As a result of the research, a new Mamdani fuzzy logic controller with nine inputs, two outputs, and eighteen rules was presented. The ANOVA hypothesis was used in this research to test the performance of the MFs, and the results revealed a significant difference between the groups. Tukey's post hoc test, on the other hand, confirmed the location of the differences, exposing triangular MF to be the most efficient. To

compare the performance of T1MFLC and T2FLC, an independent samples t-test was used. Type-2 fuzzy logic controllers significantly outperform, and Cohen's d effect size for t-tests revealed a statistically significant difference in model performance. The results potential catalyzes future research on the hardware implementation and generalization of fuzzy logic. The technique can be used to effectively build robots capable of safely carrying out missions in hazardous and populated environments.

5.3 Conclusions

The goal of this research was to improve on the traditional model by increasing the number of sensors efficiently, reducing fuzzy rules, and revising membership functions for a T1MFLC. The optimized model was then compared to a T2MFLC in a similar environment. To simulate and test the implementation, we used commercially available V-REP and MATLAB software. The research results present a novel Mamdani fuzzy logic controller with nine fused inputs, two outputs, and eighteen fuzzy rules.

First, in a static unknown environment, we combined sensory data for a fuzzy logic-controlled obstacle avoidance robot. This was done outside the controller to have a larger scanning area with fewer rules. Sensor fusion gave the mobile robot the advantage of quick response and allowing it to react to obstacles in real-time. For practical applications, this method can reduce the computational complexity of the model. Using fuzzy similarity measures, the traditional twenty-seven if-then rules were effectively reduced to eighteen, yielding good results for T1MFLC.

Second, we alternately changed Membership Functions to achieve the best obstacle avoidance mobile robot. Three MFs were tested in a similar environment: triangular, trapezoidal, and gaussian. There was a statistically significant difference between groups, according to one-way ANOVA ($F(2,39) = 9.173$, $p = 0.001$). The time required for the robot to complete the task differed statistically significantly between triangular and gaussian MFs ($p = .00$), triangular and trapezoidal MFs ($p = 0.038$), and gausian and trapezoidal MFs ($p=0.038$), as per a Tukey post hoc test. Since $p < 0.05$, the difference was significant across all groups. Tukey's post hoc test, on the

other hand, confirmed where the differences were, revealing triangular MF to be the best.

Finally, a simulated obstacle avoidance robot was used to compare the developed T1MFLC model to its corresponding T2MFLC. Previous research has shown that T1FLC is significantly faster than T2FLS in real-time applications. T1MFLC and T2MFLC were compared using an independent samples t-test. Model K had a higher score (M = 219.79, SD = 4.509) than model M in the control group (M = 223.79, SD = 3.886), indicating a disparity in the models t (26) = 2-514, p = 0.018). A significant impact is indicated by Cohen's d effect size of 0.924. The performance of the Type-2 fuzzy logic controller improved.

Sensor fusion and fuzzy rule reduction allow the robot to respond to obstacles in real-time. This method could reduce the computational complexity in practical applications. T2FLS can be used to model uncertainties and improve accuracy in a variety of applications, but there has been little research on mobile robots. Our method could be used to develop a COVID-19 robotic nurse and/or robots for use in hazardous environments where humans are not permitted. However, the findings should also apply to other fuzzy logic controllers.

5.4 Research contributions

This section focuses on the thesis's contributions. Both theoretical and empirical findings help us understand the fuzzy logic model for obstacle avoidance mobile robots in static unknown environment. This research also contributes to our understanding of how to find a path for an autonomous robot. The study's findings suggest that sensor fusion and the reduction of fuzzy logic rules improve performance and provide a potential way to optimize traditional methods of obstacle avoidance robot navigation.

Knowledge contribution: Adding literature theories on sensor fusion and fuzzy rule reduction as a possible strategy to improve fuzzy logic controller. This study also serves as a guideline for future research on optimizing a fuzzy logic obstacle avoidance controller for mobile robots, thus adding to the literature on existing

research methods. Knowledge gained by using MATLAB as a remote API with V-REP that is useful for the adoption and application of theoretical concepts and development in the context of simulating complex automated systems and other domains.

Methodological contribution: Integrating sensor fusion and fuzzy rule reduction in optimization of a fuzzy logic controller and the development of a new model for testing with VREP and MATLAB. The combination of fuzzy logic sensor fusion and rule reduction techniques was the research's key methodological contribution.

Practical contributions: The detailed insight provided by the prototype's implementation and the framework for the adoption and use of autonomous robots is one of the research's practical contributions. Other researchers will be able to replicate the discovery and use it as a guide for future research.

5.5 Recommendations

The possibility of improving the model acts as a motivator for further research in using it for a holonomic robot in a dynamic environment. We recommend more research to determine the viability of using a real robot to screen and monitor obstacles in an industrial application. This research provides new ideas for practitioners' hardware implementations. The research on dynamic environments will be of particular interest in the future. A highly recommended future research would be the extent to which a holonomic robot that can move in both forward and reverse directions could be investigated. Along with the presented methodology for collision avoidance, incorporating other learning strategies such as Artificial Neural Networks, genetic algorithms, and Adaptive Neuro-Fuzzy Inference System is recommended.

REFERENCES

Abdessemed, F., Faisal, M., Emmadeddine, M., Hedjar, R., Al-Mutib, K., Alsulaiman, M., & Mathkour, H. (2014). A hierarchical fuzzy control design for indoor mobile robot. *International Journal of Advanced Robotic Systems*, *11*(1).

Abdulkareem, A., Ogunlesi, V., Afolalu, A. S., & Onyeakagbu, A. (2019). Development of a smart autonomous mobile robot for cafeteria management. *International Journal of Mechanical Engineering and Technology*, *10*(1), 1672–1685.

Adam, Y. M., Binti Sariff, N., & Algeelani, N. A. (2021). E-puck mobile robot obstacles avoidance controller using the fuzzy logic approach. *2021 2nd International Conference on Smart Computing and Electronic Enterprise: Ubiquitous, Adaptive, and Sustainable Computing Solutions for New Normal, ICSCEE 2021*, 107–112.

Aguileta, A. A., Brena, R. F., Mayora, O., Molino-Minero-re, E., & Trejo, L. A. (2019). Multi-sensor fusion for activity recognition—a survey. *Sensors (Switzerland)*, *19*(17), 1–41.

Ahmad Fauzi, F., Mulyana, E., Mardiati, R., & Eko Setiawan, A. (2021). Fuzzy Logic Control for Avoiding Static Obstacle in Autonomous Vehicle Robot. *Proceeding of 2021 7th International Conference on Wireless and Telematics, ICWT 2021*.

Ahmed, A. A., Abdalla, T. Y., & Abed, A. A. (2015). Path Planning of Mobile Robot by using Modified Optimized Potential Field Method. *International Journal of Computer Applications*, *113*(4), 6–10.

Ahmed, F., & Deb, K. (2012). Multi-objective optimal path planning using elitist non-dominated sorting genetic algorithms. *Soft Computing*, *17*(7), 1283–1299.

Ajeil, F. H., Ibraheem, I. K., Azar, A. T., & Humaidi, A. J. (2020). Autonomous

navigation and obstacle avoidance of an omnidirectional mobile robot using swarm optimization and sensors deployment. *International Journal of Advanced Robotic Systems*, *17*(3), 1–15.

Al-dahhan, R. R., & Al-dahhan, M. R. H. (n.d.). *Target Seeking and Obstacle Avoidance of Omni Robot in an Unknown Environment*.

Al-mayyahi, A., & Wang, W. (2014). *07072684*. *November*, 28–30.

Alatise, M. B., & Hancke, G. P. (2020). A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods. *IEEE Access*, *8*, 39830–39846.

Almasri, M., Elleithy, K., & Alajlan, A. (2015). Sensor fusion based model for collision free mobile robot navigation. *Sensors (Switzerland)*, *16*(1).

Angelov, P., & Gu, X. (2017). A cascade of deep learning fuzzy rule-based image classifier and SVM. *2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017*, *2017-Janua*, 746–751.

Axelrod, B., Kaelbling, L. P., & Lozano-Pérez, T. (2018). Provably safe robot navigation with obstacle uncertainty. *International Journal of Robotics Research*, *37*(13–14), 1760–1774.

Baker, A. A., & Ghadi, Y. Y. (2020). Autonomous system to control a mobile robot. *Bulletin of Electrical Engineering and Informatics*, *9*(4), 1711–1717.

Baklouti, N., Alimi, A. M., Huang, J., Ri, M., Wu, D., Ri, S., Meylani, A., Handayani, A. S., Ciksadan, Rs, C., Husni, N. L., Nurmaini, S., Yani, I., Ruiz-Garcia, G., Hagras, H., Pomares, H., Ruiz, I. R., Cuevas, F., Castillo, O., … Castillo, O. (2020). Comparative study of interval Type-2 and general Type-2 fuzzy systems in medical diagnosis. *Information Sciences*, *95*(2), 103–139.

Barraquand, J. (1992). Numerical potential field techniques for robot path planning. *IEEE Transactions on Systems, Man and Cybernetics*, *22*(2), 224–241. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=148426

Bartczuk, Ł., Przybył, A., & Cpałka, K. (2016). A new approach to nonlinear modelling of dynamic systems based on fuzzy rules. *International Journal of Applied Mathematics and Computer Science*, *26*(3), 603–621.

Barua, A., Mudunuri, L. S., & Kosheleva, O. (2014). Why trapezoidal and triangular membership functions work so well: Towards a theoretical explanation. *Journal of Uncertain Systems*, *8*(3), 164–168.

Batti, H., Jabeur, C. Ben, & Seddik, H. (2019). Mobile Robot Obstacle Avoidance in labyrinth Environment Using Fuzzy Logic Approach. *2019 International Conference on Control, Automation and Diagnosis, ICCAD 2019 - Proceedings*.

Bayar, V., Akar, B., Yayan, U., & Yavuz, H. S. (2014). *Fuzzy Logic Based Design of Classical Behaviors for Mobile Robots in ROS Middleware*.

Berisha, J., Bajrami, X., Shala, A., & Likaj, R. (2016). Application of Fuzzy Logic Controller for obstacle detection and avoidance on real autonomous mobile robot. *2016 5th Mediterranean Conference on Embedded Computing (MECO)*, *v*, 200–205.

Birkin, P. A. S., & Garibaldi, J. M. (2009). A comparison of type-1 and type-2 fuzzy controllers in a micro-robot context. *IEEE International Conference on Fuzzy Systems*, 1857–1862.

Carlos Erlan Olival Lima, de Araujo, F. M. A., da Silva, M. B., Rocha Filho, A. E., Rabelo, R. de A. L., da Silva, T. A. R., & de Oliveira Alves, A. J. (2013). Velocity and direction planning in a sumo robot type using the method of potential field with fuzzy systems. *2013 16th International Conference on Advanced Robotics (ICAR)*, 1–7.

Castillo, O., & Melin, P. (2012). Type-2 fuzzy logic systems. *SpringerBriefs in Applied Sciences and Technology*, *1*(6), 7–12.

Chao, F., Zhou, D., Lin, C. M., Yang, L., Zhou, C., & Shang, C. (2020). Type-2

Fuzzy Hybrid Controller Network for Robotic Systems. *IEEE Transactions on Cybernetics*, *50*(8), 3778–3792.

Chaudhari, S., & Patil, M. (2014). *Study and Review of Fuzzy Inference Systems for Decision Making and Control*. 88–92.

Chaudhary, H., Panwar, V., Prasad, R., & Sukavanam, N. (2016). Adaptive neuro fuzzy based hybrid force/position control for an industrial robot manipulator. *Journal of Intelligent Manufacturing*, *27*(6), 1299–1308.

Cherroun, L., Nadour, M., & Kouzou, A. (2019). Type-1 and Type-2 Fuzzy Logic Controllers for Autonomous Robotic Motion. *Proceedings - 2019 3rd International Conference on Applied Automation and Industrial Diagnostics, ICAAID 2019*, *September*, 25–27.

Cherubini, A., Spindler, F., & Chaumette, F. (2014). Autonomous visual navigation and laser-based moving obstacle avoidance. *IEEE Transactions on Intelligent Transportation Systems*, *15*(5), 2101–2110.

Chiu, C. (2015). *Obstacle Avoidance Control of A Two-Wheeled Mobile Robot*. 4–9.

Chiu, C. S., Chiang, T. S., & Ye, Y. T. (2016). Fuzzy obstacle avoidance control of a two-wheeled mobile robot. *CACS 2015 - 2015 CACS International Automatic Control Conference*, 1–6. https://doi.org/10.1109/CACS.2015.7378356

Couceiro, M., & Marichal, J. L. (2010). Characterizations of discrete Sugeno integrals as polynomial functions over distributive lattices. *Fuzzy Sets and Systems*, *161*(5), 694–707.

Deng, Z., Jiang, Y., Choi, K. S., Chung, F. L., & Wang, S. (2013). Knowledge-leverage-based TSK fuzzy system modeling. *IEEE Transactions on Neural Networks and Learning Systems*, *24*(8), 1200–1212.

Elleithy, K. M., Alajlan, A. M., Lisat, C., Farmingdale, E., Almasri, M. M., Elleithy, K. M., & Alajlan, A. M. (2016). *of efficient obstacle avoidance and line*

following mobile robot with the integration of fuzzy logic system in static and dynamic environments ." In Proceedings of 2016 IEEE Long Island Systems , Applications and Technology This material is posted here wit.

English, A., Ross, P., Ball, D., & Corke, P. (2014). Vision based guidance for robot navigation in agriculture. *Proceedings - IEEE International Conference on Robotics and Automation*, 1693–1698.

Enyinna, N., Karimoddini, A., Opoku, D., Homaifar, A., & Arnold, S. (2015). Developing an Interval Type-2 TSK Fuzzy Logic Controller. *Annual Conference of the North American Fuzzy Information Processing Society - NAFIPS*, *2015-Septe*(1).

Eraqi, H., EmadEldin, Y., & Moustafa, M. (2016). *Reactive Collision Avoidance using Evolutionary Neural Networks*.

Faisal, M., Hedjar, R., Al Sulaiman, M., & Al-Mutib, K. (2013). Fuzzy logic navigation and obstacle avoidance by a mobile robot in an unknown dynamic environment. *International Journal of Advanced Robotic Systems*, *10*.

Functions, T.-F. M., & Raj, D. (2018). Analysis of Data Generated From Multidimensional. *IEEE Transactions on Fuzzy Systems*, *26*(2), 681–693.

Ghommam, J., & Saad, M. (2014). Backstepping-based cooperative and adaptive tracking control design for a group of underactuated AUVs in horizontal plan. *International Journal of Control*, *87*(5), 1076–1093.

Gupta, N. (2014). *Comparative Study of type-1 and Type-2 Fuzzy Systems*. *2*(4), 195–198.

Haddi, Y., & Kharchaf, A. (2021). Obstacle avoidance behavior of an autonomous mobile robot in a radioactive environment based on fuzzy logic. *E3S Web of Conferences*, *234*.

Handayani, A. S., Dewi, T., Husni, N. L., Nurmaini, S., & Yani, I. (2017). Target

tracking in mobile robot under uncertain environment using fuzzy logic controller. *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, *2017-Decem*(September), 19–21.

Handayani, A. S., Husni, N. L., Nurmaini, S., & Yani, I. (2019). Application of type-1 and type-2 fuzzy logic controller for the real swarm robot. *International Journal of Online and Biomedical Engineering*, *15*(6), 83–98.

Herrera Ortiz, J. A., Rodrguez-Vázquez, K., Padilla Castaeda, M. A., & Arámbula Coso, F. (2013). Autonomous robot navigation based on the evolutionary multi-objective optimization of potential fields. *Engineering Optimization*, *45*(1), 19–43.

Hessburg, T., & Tomizuka, M. (1993). Fuzzy Logic Control for Lateral Vehicle Guidance. *IEEE Conference on Control Applications*, *August*, 55–63.

Hsu, Chen-chien, Y. L. (2014). Path Planning for Robot Navigation Based on Cooperative Genetic Optimization. *11th IEEE International Conference on Networking, Sensing and Control*, 316–321.

Indri, M., Possieri, C., Sibona, F., Cheng, P. D. C., & Hoang, V. D. (2019). Supervised global path planning for mobile robots with obstacle avoidance. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, *2019-Septe*, 601–608.

Ismail, A.-T., Sheta, A., & Al-Weshah, M. (2008). A mobile robot path planning using genetic algorithm in static environment. *Journal of Computer Science*, *4*(4), 341–344.

Jin, Y., Seelen, W. Von, & Sendhoff, B. (1999). *On Generating FC Fuzzy Rule Systems from Data Using Evolution Strategies*. *29*(6), 829–845.

Kalpana, M., & Kumar, A. V. S. (2013). *Similarity measure between fuzzy set , fuzzy numbers and fuzzy rules using T Fuzzy Assessment Methodology*. *7*(8), 72–80.

Kanezaki, A., Nitta, J., & Sasaki, Y. (2018). GOSELO: Goal-Directed Obstacle and Self-Location Map for Robot Navigation Using Reactive Neural Networks. *IEEE Robotics and Automation Letters*, *3*(2), 696–703.

Kansal, V., & Kaur, A. (2013). Comparison of Mamdani-type and Sugeno-type FIS for Water Flow Rate Control in a Rawmill. *International Journal of Scientific & Engineering Research*, *4*(6), 2580–2584.

Karnik, N. N., & Mendel, J. M. (1998). Type-2 fuzzy logic systems: type-reduction. *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference On*, *2*, 2046–2051 vol.2.

Kashyap, A. K., & Parhi, D. R. (2022). Obstacle avoidance and path planning of humanoid robot using fuzzy logic controller aided owl search algorithm in complicated workspaces. *Industrial Robot*, *49*(2), 280–288.

Khaksar, W., Uddin, M. Z., & Torresen, J. (2019). Fuzzy motion planning for nonholonomic mobile robot navigation in unknown indoor environments. *International Journal of Mechanical Engineering and Robotics Research*, *8*(1), 6–11.

Kreinovich, V., Kosheleva, O., & Shahbazova, S. N. (2020). Why Triangular and Trapezoid Membership Functions: A Simple Explanation. *Studies in Fuzziness and Soft Computing*, *391*, 25–31.

Lamini, C., Benhlima, S., & Elbekri, A. (2018). Genetic algorithm based approach for autonomous mobile robot path planning. *Procedia Computer Science*, *127*, 180–189.

LAOUICI, Z., Amine, M., & KHELFI, M. (2014). *Hybrid Method for the Navigation of Mobile Robot Using Fuzzy Logic and Spiking Neural Networks*. *November*, 1–9.

Leottau, L., & Melgarejo-Rey, M. A. (2010). A Simple Approach for Designing a Type-2 Fuzzy Controller for a Mobile Robot Application. *Annual Meeting of*

the North American Fuzzy Information Processing Society, 1–6.

Li, B., Du, H., & Li, W. (2017). A Potential Field Approach-Based Trajectory Control for Autonomous Electric Vehicles With In-Wheel Motors. *IEEE Transactions on Intelligent Transportation Systems*, *18*(8), 2044–2055.

Li, F., Shang, C., Li, Y., Yang, J., & Shen, Q. (2018). Fuzzy Rule Based Interpolative Reasoning Supported by Attribute Ranking. *IEEE Transactions on Fuzzy Systems*, *26*(5), 2758–2773.

Li, X., & Choi, B. (2013). Design of Obstacle Avoidance System for Mobile Robot using Fuzzy Logic Systems. *International Journal of Smart Home*, *7*(3), 321–328.

Liang, X., Fang, Y., Sun, N., & Lin, H. (2019). A novel energy-coupling-based hierarchical control approach for unmanned quadrotor transportation systems. *IEEE/ASME Transactions on Mechatronics*, *24*(1), 248–259.

Liggins, M. E., Hall, D. L., & Llinas, J. (2009). Handbook of Multisensor Data Fusion: Theory and Practice. In *CRC Press*.

Lin, J., Tasi, C. Y., & Lin, K. H. (2015). *Hybrid Control Mode Based On Multi-Sensor Information by Fuzzy Approach for Navigation Task of Autonomous Mobile Robot*. *9*(4), 438–445.

Long, Y., Tan, Y., Organisciak, D., Yang, L., & Shao, L. (2019). Towards Light-weight Annotations: Fuzzy Interpolative Reasoning for Zero-shot Image Classification. *British Machine Vision Conference 2018, BMVC 2018*, 1–12.

López-González, A., Meda Campaña, J. A., Hernández Martínez, E. G., & Contro, P. P. (2020). Multi robot distance based formation using Parallel Genetic Algorithm. *Applied Soft Computing Journal*, *86*, 105929.

Marin-Plaza, P., Hussein, A., Martin, D., & De La Escalera, A. (2018). Global and Local Path Planning Study in a ROS-Based Research Platform for Autonomous

Vehicles. *Journal of Advanced Transportation*, *2018*.

McDowell, J., Lindsay, G., McPhail, K., Singh, J. A., Lalonde, L., Goudreau, J., Hudon, E., Lussier, M. T., Duhamel, F., Bélanger, D., Lévesque, L., Martin, E., Rubin, G., De Wit, N., Meineche-Schmidtc, V., Seifertd, B., Halla, N., Hungin, P., Chéron, E., … Curfs, L. M. G. L. M. G. . e f. (2012). ContentServer (5).pdf. In *Family Practice* (Vol. 24, Issue 4, pp. 48–55). https://doi.org/10.1186/ar4524

Melek, W. (2018). *Fuzzy Logic in Intelligent System Design*. *648*(March 2020), 1–8.

Mendel, J. M., Eyoh, I., & John, R. (2020). Comparing Performance Potentials of Classical and Intuitionistic Fuzzy Systems in Terms of Sculpting the State Space. *IEEE Transactions on Fuzzy Systems*, *28*(9), 2244–2254.

Miller, J., & Papanikolopoulos, A. N. (2020). *Obstacle Avoidance - Advanced Motion Planning and Control for Path-Following Robots*. 1–18.

Mittal, K., Jain, A., Vaisla, K. S., Castillo, O., & Kacprzyk, J. (2020). A comprehensive review on type 2 fuzzy logic applications: Past, present and future. *Engineering Applications of Artificial Intelligence*, *95*(August), 103916.

Mutlu, B., Sezer, E. A., & Akcayol, M. A. (2018). End-to-end hierarchical fuzzy inference solution. *IEEE International Conference on Fuzzy Systems*, *2018-July*, 0–8.

Naik, K. A., & Gupta, C. P. (2017). Performance comparison of Type-1 and Type-2 fuzzy logic systems. *4th IEEE International Conference on Signal Processing, Computing and Control, ISPCC 2017*, *2017-Janua*, 72–76.

Najmurrokhman, A., Kusnandar, Komarudin, U., Sunubroto, Sadiyoko, A., & Iskanto, T. Y. (2019). Mamdani based Fuzzy Logic Controller for A Wheeled Mobile Robot with Obstacle Avoidance Capability. *Proceedings of the 2019 International Conference on Mechatronics, Robotics and Systems Engineering, MoRSE 2019*, *December*, 49–53.

Nurmaini, S., & Chusniah, C. (2017). Differential drive mobile robot control using variable fuzzy universe of discourse. *ICECOS 2017 - Proceeding of 2017 International Conference on Electrical Engineering and Computer Science: Sustaining the Cultural Heritage Toward the Smart Environment for Better Future*, 50–55. https://doi.org/10.1109/ICECOS.2017.8167165

Oleiwi, B. K., Mahfuz, A., & Roth, H. (2015). *Implementation of Human Perception in Mobile*. *15*(6).

Omrane, H., Masmoudi, M. S., & Masmoudi, M. (2016). Fuzzy Logic Based Control for Autonomous Mobile. *Computational Intelligence and Neuroscience*, *2016*, 1–10.

Ontiveros-Robles, E., & Melin, P. (2020). Toward a development of general type-2 fuzzy classifiers applied in diagnosis problems through embedded type-1 fuzzy classifiers. *Soft Computing*, *24*(1), 83–99.

Ontiveros-Robles, E., Melin, P., & Castillo, O. (2018). Comparative analysis of noise robustness of type 2 fuzzy logic controllers. *Kybernetika*, *54*(1), 175–201.

Pandey, A., Sonkar, R. K., Pandey, K. K., & Parhi, D. R. (2014). Path Planning Navigation of Mobile Robot With Obstacles Avoidance Using Fuzzy Logic Controller. *International Conference on Intelligent Systems and Control*, 36–41.

Paulius, D., & Sun, Y. (2019). A Survey of Knowledge Representation in Service Robotics. *Robotics and Autonomous Systems*, *118*, 13–30.

Plunk, A., Richard, I., Marks, L., Larsen, E., City, F., & Us, C. A. (2015). *(12) United States Patent FOREIGN PATENT DOCUMENTS*. *2*(12).

Potena, C., Nardi, D., & Pretto, A. (2019). Joint vision-based navigation, control and obstacle avoidance for UAVs in dynamic environments. *2019 European Conference on Mobile Robots, ECMR 2019 - Proceedings*.

Prokopowicz, P., Czerniak, J., Mikolajewski, D., Apiecionek, L., & Slezak, D.

(2017). *Theory and Applications of Ordered Fuzzy Numbers* (Vol. 356).

Rasekhipour, Y., Khajepour, A., Chen, S. K., & Litkouhi, B. (2017). A Potential Field-Based Model Predictive Path-Planning Controller for Autonomous Road Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *18*(5), 1255–1267.

Raulcezar, M., & Lopes, C. R. (2016). Obstacle avoidance for mobile robots: a Hybrid Intelligent System based on Fuzzy Logic and Artificial Neural Network. *2016 IEEE International Conference on Fuzzy Systems*, 1038–1043.

Ren, J. (n.d.). *Fuzzy Logic in Control Fuzzy Logic*.

RENDYANSYAH, NURMAINI, S., SARI, S., & RAMARTA, D. B. (2020). *Implementation of Fuzzy Logic Control for Navigation System in Mobile Robot Omnidirectional*. *172*(Siconian 2019), 242–250.

Rohmer, E., Singh, S. P. N., & Freese, M. (2013). V-REP: A versatile and scalable robot simulation framework. *IEEE International Conference on Intelligent Robots and Systems*, 1321–1326.

Ross, T. J. (2004). Fuzzy Logic with Engineering Applications. In *Wiley*.

Rossomando, F. G., Soria, C., & Carelli, R. (2011). Adaptive neural dynamic compensator for mobile robots in trajectory tracking control. *IEEE Latin America Transactions*, *9*(5), 593–602.

Ruiz-Garcia, G., Hagras, H., Pomares, H., & Ruiz, I. R. (2019). Toward a Fuzzy Logic System Based on General Forms of Interval Type-2 Fuzzy Sets. *IEEE Transactions on Fuzzy Systems*, *27*(12), 2381–2395.

Seines, M., & Babuška, R. (2001). Rule base reduction: Some comments on the use of orthogonal transforms. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *31*(2), 199–206.

Selvachandran, G., Quek, S. G., Lan, L. T. H., Son, L. H., Long Giang, N., Ding, W.,

Abdel-Basset, M., & Albuquerque, V. H. C. (2019). A New Design of Mamdani Complex Fuzzy Inference System for Multi-attribute Decision Making Problems. *IEEE Transactions on Fuzzy Systems*, *6706*(c), 1–1.

Shamsfakhr, F., & Sadeghibigham, B. (2017). A neural network approach to navigation of a mobile robot and obstacle avoidance in dynamic and unknown environments. *Turkish Journal of Electrical Engineering and Computer Sciences*, *25*(3), 1629–1642.

Signifredi, A., Luca, B., Coati, A., Medina, J. S., & Molinari, D. (2015). A General Purpose Approach for Global and Local Path Planning Combination. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 996–1001.

Silva, W. A. S., Rabelo, R. A. L., & Santana, A. M. (2014). Safe autonomous navigation with a wall-following robot using interval Type-2 Fuzzy System in uncertain environments. *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 1222–1225.

Singh, B., & Mishra, A. K. (2015). Fuzzy Logic Control System and its Applications. *International Research Journal of Engineering and Technology (IRJET)*, *2*(8), 743–746.

Spica, R., Claudio, G., Spindler, F., & Giordano, P. R. (2016). *Interfacing Matlab / Simulink with V-REP for an Easy Development of Sensor-Based Control Algorithms for Robotic Platforms Interfacing Matlab / Simulink with V-REP for an Easy Development of Sensor-Based Control Algorithms for Robotic Platforms*. *MAY 2014*, 0–2.

Spolaor, S. (2019). *Fuzzy logic for the modeling and simulation of complex systems*.

Su, K., & Phan, T. (2014). Robot Path Planning and Smoothing Based on Fuzzy Inference. *2014 IEEE International Conference on System Science Ang Engineering (ICSSE)*.

Tan, Y., Shum, H. P. H., Chao, F., Vijayakumar, V., & Yang, L. (2019). Curvature-

based sparse rule base generation for fuzzy rule interpolation. *Journal of Intelligent and Fuzzy Systems*, *36*(5), 4201–4214.

Taskin, A., & Kumbasar, T. (2015). An open source matlab/simulink toolbox for interval type-2 fuzzy logic systems. *Proceedings - 2015 IEEE Symposium Series on Computational Intelligence, SSCI 2015*, 1561–1568.

Thaker, S., & Nagori, V. (2018). Analysis of Fuzzification Process in Fuzzy Expert System. *Procedia Computer Science*, *132*, 1308–1316.

Tsai, C. P., Chuang, C. T., Lu, M. C., Wang, W. Y., Su, S. F., & Chang, S. L. (2013). Machine-vision based obstacle avoidance system for robot system. *ICSSE 2013 - IEEE International Conference on System Science and Engineering, Proceedings*, 273–277.

Tsekouras, G. E. (2016). Fuzzy rule base simplification using multidimensional scaling and constrained optimization. *Fuzzy Sets and Systems*, *297*(November 2015), 46–72.

Tuncer, a., Yildirim, M., & Erkan, K. (2012). A Motion Planning System for Mobile Robots. *Advances in Electrical and Computer Engineering*, *12*(1), 57–62.

V-Rep. (2012). *V-REP Quick User Manual*. *May*.

View, M. R., & Pandey, A. (2017). *Mobile Robot Navigation in Static and Dynamic Environments using Various Soft Computing Techniques Mobile Robot Navigation in Static and Dynamic Environments using Various Soft Computing Techniques Anish Pandey Department of Mechanical Engineering Nationa*. *October*.

Wahid, N., Zamzuri, H., Rahman, M. A. A., Kuroda, S., & Raksincharoensak, P. (2017). Study on potential field based motion planning and control for automated vehicle collision avoidance systems. *Proceedings - 2017 IEEE International Conference on Mechatronics, ICM 2017*, *2*, 208–213.

Wong, C., Yang, E., Yan, X., Gu, D., & Park, W. (2017). *Adaptive and Intelligent Navigation of Autonomous Planetary Rovers – A Survey*. 237–244.

Wu, D. (2013). Two differences between interval type-2 and type-1 fuzzy logic controllers: Adaptiveness and novelty. *Studies in Fuzziness and Soft Computing*, *301*, 33–48.

Wu, D., & Mendel, J. M. (2019). Similarity Measures for Closed General Type-2 Fuzzy Sets: Overview, Comparisons, and a Geometric Approach. *IEEE Transactions on Fuzzy Systems*, *27*(3), 515–526.

Wu, K., Yuan, S., Esfahani, M. A., & Wang, H. (2019). Depth-based obstacle avoidance through deep reinforcement learning. *ACM International Conference Proceeding Series*, *Part F1476*(February), 102–106.

Yan-ping, W., & Bing, W. (2010). Robot path planning based on modified genetic algorithm. *2010 2nd International Conference on Future Computer and Communication*, *3*, V3-725-V3-728.

Yanrong Hu, & Yang, S. X. (2004). A knowledge based genetic algorithm for path planning of a mobile robot. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, *April*, 4350-4355 Vol.5.

Yekinni, L. A., & Dan-Isa, A. (2019). Fuzzy Logic Control of Goal-Seeking 2-Wheel Differential Mobile Robot Using Unicycle Approach. *2019 IEEE International Conference on Automatic Control and Intelligent Systems, I2CACIS 2019 - Proceedings*, *June*, 300–304.

Yerubandi, V., Reddy, Y. M., & Reddy, M. V. K. (2015). Navigation system for an autonomous robot using fuzzy logic. *International Journal of Scientific and Research Publications*, *5*(2), 5–8.

Yu, Y., Tingting, W., Long, C., & Weiwei, Z. (2018). Stereo vision based obstacle avoidance strategy for quadcopter UAV. *Proceedings of the 30th Chinese Control and Decision Conference, CCDC 2018*, 490–494.

Zadeh, L. A. (1965). Fuzzy Sets. *Inform Control*, *8*, 338–353.

Zhao, R., Lee, D. H., Li, T. T., & Lee, H. K. (2015). *Autonomous Navigation of a Mobile Robot in Unknown Environment Based on Fuzzy Inference Sv dt d Yr de*. 19–24.

Zhao, T., Chen, Y., Dian, S., Guo, R., & Li, S. (2020). General Type-2 Fuzzy Gain Scheduling PID Controller with Application to Power-Line Inspection Robots. *International Journal of Fuzzy Systems*, *22*(1), 181–200.

APPENDICES

Appendix I: MATLAB coding for simulation

```
clc

clear all

disp('Program started');

 vrep=remApi('remoteApi'); % use the remote API functionality in your Matlab program

  vrep.simxFinish(-1); % just in case, close all opened connections

   clientID=vrep.simxStart('127.0.0.1',19999,true,true,5000,5); % as soon as the scene is played, a server will be listening on port 19999)



  if (clientID>-1)

      disp('Connected to remote API server');

        vrep.simxAddStatusbarMessage(clientID,'Hello V-REP!',vrep.simx_opmode_oneshot);

      %%handles


[returnCodelm,left_Motor]=vrep.simxGetObjectHandle(clientID,'Pioneer_p3dx_left Motor',vrep.simx_opmode_blocking); % obtain a handle left motors of the robot


[returnCoderm,right_Motor]=vrep.simxGetObjectHandle(clientID,'Pioneer_p3dx_rig htMotor',vrep.simx_opmode_blocking); % obtain a handle left motors of the robot


[returnCodefs,front_Sensor]=vrep.simxGetObjectHandle(clientID,'Pioneer_p3dx_ult rasonicSensor5',vrep.simx_opmode_blocking); % detecting  front objects


[returnCodels,left_Sensor]=vrep.simxGetObjectHandle(clientID,'Pioneer_p3dx_ultra sonicSensor2',vrep.simx_opmode_blocking); % detecting  left  objects
```

```
[returnCoders,right_Sensor]=vrep.simxGetObjectHandle(clientID,'Pioneer_p3dx_ultrasonicSensor7',vrep.simx_opmode_blocking); % detecting  right objects

        %
[returnCodef1,handle]=vrep.simxGetDistanceHandle(clientID,'Pioneer_p3dx_ultrasonicSensor5',vrep.simx_opmode_blocking);

         %other code


%[returnCode]=vrep.simxSetJointTargetVelocity(clientID,left_Motor,VLW,vrep.simx_opmode_blocking);

       %
[returnCode,detectionState,detectedPoint,~,~]=vrep.simxReadProximitySensor(clientID,front_Sensor, vrep.simx_opmode_streaming);

         for i=1:3600


[returnCodel,detectionStatel,detectedPointl,~,~]=vrep.simxReadProximitySensor(clientID,left_Sensor, vrep.simx_opmode_blocking);

         pause(0.0001);


[returnCoder,detectionStater,detectedPointr,~,~]=vrep.simxReadProximitySensor(clientID,right_Sensor, vrep.simx_opmode_blocking);

         pause(0.0001);


[returnCodef,detectionStatef,detectedPointf,~,~]=vrep.simxReadProximitySensor(clientID,front_Sensor, vrep.simx_opmode_blocking);

         L=norm(detectedPointl);

         F=norm(detectedPointf);

         R=norm(detectedPointr);

         nonormdetp=0.9999;

         minnormdetp=0.2;

          if ((L>nonormdetp)||(L<minnormdetp))
```

```
    L1=nonormdetp;
  else
    L1=L;
  end
  if ((F>nonormdetp)||(F<minnormdetp))
    F1=nonormdetp;
  else
    F1=F;
  end
  if ((R>nonormdetp)||(R<minnormdetp))
    R1=nonormdetp;
  else
    R1=R;
  end
  disl=double(L1);
  disf=double(F1);
  disr=double(R1);
     D=[disl disf disr];
%       disp(D);
  fis=readfis('fuzzytrigaus');
  V=evalfis([disl disf disr],fis);%*******
  disp(V);
 % dV=single(V)
VlW =V(1,1);
```

```
        VrW =V(1,2);

        %      disp('Input variable is 0(zero is not alllowed as input)')

      %
[returnCode]=vrep.simxSetJointTargetVelocity(clientID,left_Motor,[VLW],vrep.sim
x_opmode_blocking);


[returnCode]=vrep.simxSetJointTargetVelocity(clientID,left_Motor,VLW,vrep.simx
_opmode_streaming);


[returnCode]=vrep.simxSetJointTargetVelocity(clientID,right_Motor,VRW,vrep.sim
x_opmode_streaming);

         % end



%[returnCode]=vrep.simxSetJointTargetVelocity(clientID,left_Motor,VLW,vrep.sim
x_opmode_blocking);


%[returnCode]=vrep.simxSetJointTargetVelocity(clientID,right_Motor,VRW,vrep.si
mx_opmode_blocking);




           pause(0.01);

         end



[returnCode]=vrep.simxSetJointTargetVelocity(clientID,left_Motor,0,vrep.simx_op
mode_blocking);


[returnCode]=vrep.simxSetJointTargetVelocity(clientID,right_Motor,0,vrep.simx_op
mode_blocking);

        vrep.simxFinish(clientID);
```

```
        else
        disp('Failed connecting to remote API server');
    end
    vrep.delete();

    disp('Program ended');
```

Appendix II: MATLAB coding for sensor fusion

```
[returnCodefs,front_Sensor]=vrep.simxGetObjectHandle(clientID,'disf',vrep.simx_opmode_blocking); % detecting  front objects

        dd= pioneer_p3dx_ultrasonicSensor4

          de= pioneer_p3dx_ultrasonicSensor5

            df= pioneer_p3dx_ultrasonicSensor6

             disf=(dd+de+df)/3

[returnCodels,left_Sensor]=vrep.simxGetObjectHandle(clientID,'disl',vrep.simx_opmode_blocking); % detecting  left  objects

         da= pioneer_p3dx_ultrasonicSensor1

                db= pioneer_p3dx_ultrasonicSensor2

                       dc= pioneer_p3dx_ultrasonicSensor3

            disl=(da+db+dc)/3
[returnCoders,right_Sensor]=vrep.simxGetObjectHandle(clientID,'disr',vrep.simx_opmode_blocking); % detecting  right objects

                dg= pioneer_p3dx_ultrasonicSensor7

                       dh= pioneer_p3dx_ultrasonicSensor8

                              di= pioneer_p3dx_ultrasonicSensor9

                disr=(dg+dh+di)/3

[returnCodel,detectionStatel,detectedPointl,~,~]=vrep.simxReadProximitySensor(clientID,disl, vrep.simx_opmode_blocking);

        pause(0.0001);
```

```
[returnCoder,detectionStater,detectedPointr,~,~]=vrep.simxReadProximitySensor(cli
entID,disf, vrep.simx_opmode_blocking);

        pause(0.0001);


[returnCodef,detectionStatef,detectedPointf,~,~]=vrep.simxReadProximitySensor(cli
entID,disr, vrep.simx_opmode_blocking);

%           pause(0.0001);

        L=norm(detectedPointl);

        F=norm(detectedPointf);

        R=norm(detectedPointr);

        nonormdetp=0.9999;

        minnormdetp=0.2;
```

Appendix III: MATLAB coding T2MFLC

```
[System]

Name='fuzzyt2'

Type='mamdani'

Version=2.0

NumInputs=3

NumOutputs=2

NumRules=18

AndMethod='prod'

OrMethod='probor'

ImpMethod='prod'

AggMethod='sum'

DefuzzMethod='wtaver'

TypeRedMethod='EIASC'

outputType='interval'


[Input1]

Name='disl'

Range=[-1 1]

NumMFs=3

MF1U='Near':'trimf',[-1.8 -1 -0.2 1]

MF1L='Near':'trimf',[-1.6 -1 -0.4 0.7]

MF2U='Medium':'trimf',[-0.8 0 0.8 1]

MF2L='Medium':'trimf',[-0.6 0 0.6 0.7]

MF3U='Far':'trimf',[0.2 1 1.8 1]
```

```
MF3L='Far':'trimf',[0.4 1 1.6 0.7]


[Input2]
Name='disf'
Range=[-1 1]
NumMFs=3
MF1U='Near':'trimf',[-1.8 -1 -0.2 1]
MF1L='Near':'trimf',[-1.6 -1 -0.4 0.7]
MF2U='Medium':'trimf',[-0.8 0 0.8 1]
MF2L='Medium':'trimf',[-0.6 0 0.6 0.7]
MF3U='Far':'trimf',[0.2 1 1.8 1]
MF3L='Far':'trimf',[0.4 1 1.6 0.7]


[Input3]
Name='disr'
Range=[-1 1]
NumMFs=3
MF1U='Near':'trimf',[-1.8 -1 -0.2 1]
MF1L='Near':'trimf',[-1.6 -1 -0.4 0.7]
MF2U='Medium':'trimf',[-0.8 0 0.8 1]
MF2L='Medium':'trimf',[-0.6 0 0.6 0.7]
MF3U='Far':'trimf',[0.2 1 1.8 1]
MF3L='Far':'trimf',[0.4 1 1.6 0.7]


[Output1]
Name='Vl'
```

```
Range=[-1 1]

NumMFs=3

MF1='Big':'constant',[0 0]

MF2='Medium':'constant',[0.5 0.5]

MF3='Small':'constant',[1 1]


[Output2]

Name='Vr'

Range=[-1 1]

NumMFs=3

MF1='Big':'constant',[0 0]

MF2='Medium':'constant',[0.5 0.5]

MF3='Small':'constant',[1 1]


[Rules]

1 1 2, 2 3 (1) : 1

1 1 3, 1 2 (1) : 1

1 2 1, 3 3 (1) : 1

1 2 3, 1 2 (1) : 1

1 3 1, 2 2 (1) : 1

1 3 3, 1 2 (1) : 1

2 1 1, 3 1 (1) : 1

2 2 1, 3 1 (1) : 1

2 2 2, 2 2 (1) : 1

2 2 3, 2 3 (1) : 1

2 3 1, 3 1 (1) : 1
```

```
2 3 2, 2 2 (1) : 1

3 1 2, 3 2 (1) : 1

3 1 3, 3 1 (1) : 1

3 2 1, 2 2 (1) : 1

3 2 2, 2 1 (1) : 1

3 3 1, 3 2 (1) : 1

3 3 3, 1 1 (1) : 1
```

Appendix IV: ANOVA test using SPSS.

1. Click Analyze > Compare Means > One-Way ANOVA.
2. Add the variable *triangular MF* to the Dependent List box, and add the variable *Gaussian MF* to the Factor box.
3. Click Options. Check the box for Means plot, then click Continue.
4. Click OK to finish.

Appendix V: Independent Samples t- test in SPSS.

1. “Click Analyze > Compare Means > Independent-Samples T Test”.
2. Move the variable to Grouping Variable field, and move the variable MileMinDur to the Test Variable(s) area. Now variable is defined as the independent variable and MileMinDur is defined as the dependent variable.
3. Click Define Groups, which opens a new window. By default, the option to use specified values is selected. Type "0" in the first text box and "1" in the second text box because our grouping variable is numerically coded (0 = "Non-variable," 1 = "variable"). This means we'll be comparing groups 0 and 1, which represent non-variable and variable, respectively. When you're finished, click Continue.
4. Click OK to run the Independent Samples t Test. Output for the analysis will display in the Output Viewer window.