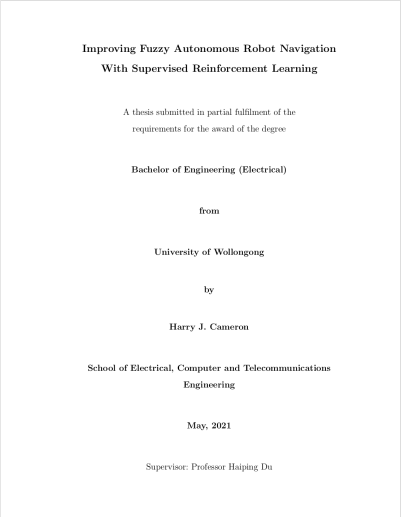
turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Harry Cameron
Assignment title:	ECTE458 Draft Report only Draft Report (Moodle TT)
Submission title:	ECTE458 Report Final
File name:	63101_Harry_Cameron_ECTE458_Report_Final_1979463_6101...
File size:	4.12M
Page count:	61
Word count:	19,585
Character count:	101,577
Submission date:	22-May-2021 03:36PM (UTC+1000)
Submission ID:	1589187715



Improving Fuzzy Autonomous Robot Navigation
With Supervised Reinforcement Learning

A thesis submitted in partial fulfilment of the
requirements for the award of the degree

Bachelor of Engineering (Electrical)

from

University of Wollongong

by

Harry J. Cameron

School of Electrical, Computer and Telecommunications
Engineering

May, 2021

Supervisor: Professor Haiping Du

Improving Fuzzy Autonomous Robot Navigation With Supervised Reinforcement Learning

A thesis submitted in partial fulfilment of the
requirements for the award of the degree

Bachelor of Engineering (Electrical)

from

University of Wollongong

by

Harry J. Cameron

School of Electrical, Computer and Telecommunications Engineering

May, 2021

Supervisor: Professor Haiping Du

Abstract

Autonomous Robots are mobile robots that can navigate and complete objectives autonomously using on-board sensors and without human intervention. Their ability to be autonomous has seen their widespread use in areas such as industry logistics to farming. In these applications, navigation occurs in unknown, complex and uneven terrain where the need for optimal navigation is critical. However, while current research has shown the strong navigational ability of fuzzy logic and Reinforcement Learning hybrids there is uncertainty in the comparative performance to other state-of-the-art controllers due to a lack of quantitative metrics and multi-controller comparative evaluations on simulation or hardware. Further, the control improvements achieved by Reinforcement Learning are difficult to understand and not implemented back onto the original Fuzzy Logic Controller.

Thus, this thesis investigates the ability of Reinforcement Learning to improve the navigational control of a Fuzzy Logic Controller when compared to alternative controllers in both simulation and hardware implementations. The improved control policies are also transferred back to the original Fuzzy Logic Controller via Policy Distillation to provide greater insight into how Autonomous Robot navigation can continue to be improved. The results highlight the strong improvement in navigational control, when compared to alternative controllers, obtained through Reinforcement Learning in both simulation and hardware implementations. The Reinforcement Learning tuning improvements were also successfully distilled onto the original Fuzzy Logic Controller, resulting in optimal control policies understandable by humans.

The findings from this thesis warrant further research into the application of Reinforcement Learning for improving fuzzy Autonomous Robot navigation. Additionally, further investigations must be conducted into the use of Policy Distillation for refining Autonomous Robot control and understanding how robot systems best navigate.

Acknowledgements

I would like to thank Professor Haiping Du for providing support and mentorship throughout the duration of this thesis.

Statement of Originality

I, Harry J. Cameron, declare that this thesis, submitted as part of the requirements for the award of Bachelor of Engineering, in the School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications or assessment at any other academic institution.

As author of this thesis, I also hereby grant, subject to any prior confidentially agreements, SECTE permission, to use, distribute, publish, exhibit, record, digitize broadcast, reproduce and archive this work for the purposes of further research and teaching.

(strike out that which does not apply)

This thesis;

~~IS~~

IS NOT

subject to a prior confidentially agreement.

Signature: 

Print Name: Harry Cameron

Student ID Number: 5013501

Date: Saturday 22nd May, 2021

Contents

List of Tables

List of Figures

Abbreviations and Symbols

AR	Autonomous Robot
FLC	Fuzzy logic controller
RL	Reinforcement learning
SL	Supervised Learning
$SOTA$	State-of-the-art
TTT	Time to target
TER	Total episodic reward
DPG	Deterministic Gradient Policy
$DDPG$	Deep Deterministic Gradient Policy
SRL	Supervised reinforcement learning
ML	Machine Learning
PD	Policy Distillation
$ANFIS$	Adaptive Neuro-Fuzzy Inference System
$RMSE$	Root mean squared error
$FPGA$	Field programmable gate arrays
FD	Front distance
DD	Differential distance
MF	Membership function
$SFSL$	Supervised Fuzzy Sarsa Learning
PSO	Particle swarm optimisation
FCL	Fully connected layer
$ReLu$	Rectified Linear Unit
IR	Infrared
v_L	Left wheel velocity
v_R	Right wheel velocity
D	Distance to the target
O	Odometer
U_S	Ultrasonic Sensors
U_D	Ultrasonic Distance
θ	Target angle
$\mu(x)$	Membership degree of x
γ	Discount factor
$r(t)$	Reward value at t
s_t	Observation state at time t
a_t	Action state at time t
$Q(s_t, a_t)$	Q-value at time t
π	Agent control policy
$H(x)$	Heaviside function

List of Changes

Section	Statement of Changes	Page Number
Chapter 2	Expanded introductory paragraphs to summarise secondary reviews of ARs and introduce what will be talked about.	5
Section 2.1	Added additional research and linked correlations and differences of research as suggested by ECTE451 feedback.	6
Section 2.2	Added additional research and linked correlations and differences of research as suggested by ECTE451 feedback.	11
Section 2.3	Added section on PD and the lack of research in AR about PD to emphasise importance.	20
Section 3.1	Updated details in robot simulation model and provided additional research that uses similar model	23
Section 3.2	Updated the FLC controller details to what was used in ECTE458 and provided research that affected design choices.	25
Section 3.3	Updated design, parameters, reward functions and model details that were changed in ECTE458 compared to ECTE451 to improve the preliminary navigation results of ECTE451. The work of ECTE458 has been to further extend the work of ECTE451 and obtain improved, convincing evidence of improvement of control. Thus, the work of ECTE458 relies of the code, models and technical development made in ECTE451, and has heavily extended this work. The section has been updated accordingly to highlight the changes.	26
Section 3.4	Added section on the application of PD	32
Section 3.5	Updated to further develop evaluation criteria through additional SOTA comparitive models and 16 starting locations	33
Section 3.6	Added section on hardware implementation and evaluation	36

Chapter 1

Introduction

Autonomous Robots (AR) are robots that can navigate in their environment and complete objectives without the intervention of human assistance. They have found wide applications in areas such as farming [1, 2], military [3], supply chain management [4] and search and rescue [5]. All of the applications where ARs are used require that they are able to safely navigate in environments, known or unknown, in the most efficient manner. The five key areas of development that are vital to ensure the continuous innovation of ARs are: "Artificial Intelligence, navigation, cost reduction, sensors and response capabilities and public policy" [4]. Through addressing these five keys areas, problems that AR research currently faces, such as navigation in dynamic and complex terrains according to a review of ARs [6], can be solved. From the perspective of AR control, the areas of 'Artificial Intelligence' and 'control' must be further refined and improved to ensure an Autonomous future.

AR navigation can either occur with the robot having global information about its environment, where it uses path planning to pre-plan a navigational path [7], or having only local sensor information. For many current, and potential future, applications of AR, such as in farming, navigation has to occur where the environment is unknown, highly dynamic and on uneven terrain where "nature is not uniform" [8]. In these types of environments the ARs will only have access to local information. Navigation through local information can be effectively addressed by the use of Fuzzy Logic Controllers (FLC) and Machine Learning (ML) based controllers [9]. A review of the challenges currently facing ARs found that these approaches struggle with navigating in complex environments [10], nor did many researchers implement their findings onto hardware platforms [9].

Fuzzy Logic Controllers (FLC) have been widely used in research to effectively address the two key AR development areas of 'control' and 'sensor and response capabilities' due to being both robust and understandable to humans. [11–22]. Researchers have utilised FLCs in achieving objectives of obstacle avoidance and target seeking in both simulation and hardware experiments. The broad application of

FLCs for AR navigation is due to their robust navigation control even in the presence of dynamic obstacles [23], sensor dead-zones or sensor noise [24]. FLCs have also maintain their navigational performance even when their computational complexity is reduced through multivariate data fusion [12] or having two separate controllers individually achieving the obstacle avoidance and target seeking objectives [20]. However, much of the current research fails to test FLCs on complex environments or against other state-of-the-art controllers (SOTA) using robust, quantitative metrics. Additionally, one of the main limitations of FLCs is their inability to learn new control policies without the intervention of expert tuning or training data. Further research is required to investigate improving FLC navigational performance while also quantitatively comparing its overall performance to alternative controllers.

The inability of FLCs to learn new, more optimal control policies is addressed through ML based AR controllers, which generate navigational improvements through adjusting model parameters to best suit the training environment [9]. The different application of ML in AR research is broad, with navigation taught through techniques such as Evolutionary Programming (EP) [25], Particle Swarm Optimisation (PSO) [26], Deep Supervised Learning (SL) [27, 28] and Reinforcement Learning (RL) [26, 29–31]. Yet, only Reinforcement Learning teaches improved control policies through the robot's own experiences. This allows for the robot to continually optimise its navigational control through its own exploration. Reinforcement Learning has been used to achieve SOTA performance in a wide range of applications such as poker [32], Go [33], video games [34, 35], stock trading [36], chess [37] and AR navigation [38–43]. Yet, none of these applications of RL for AR navigation use it to improve the expert control policies of already developed FLCs, rather they use Deep Neural Networks (DNN) and train from the beginning. Conqequently, the training process for RL takes a large amount of time, and may not necessarily converge to an optimal solution. Additionaly, understanding the learnt control policies stored in the DNN is difficult. To address both these issues, AR navigation RL solutions need to reduce training time through using expert knowledge while also storing the control policies in a architecture that is easily understood so that humans can understand how the key area of 'control' can be further addressed.

The long training period of RL based AR navigation can be reduced through transferring the expert knowledge of FLCs to the DNN before training in a Supervised Reinforcement Learning algorithm (SRL) [44–46]. One major limitation of the research in this area is in not quantitatively comparing the performance of the developed AR controllers to other SOTA controllers, in either simulation or physical experiments. This limitation is also mirrored in the research experiments for the Expert FLCs. Additionally, the improved policies taught through SRL are not transferred back onto the original Expert FLC for comparison. Due to the black box behaviour of DNNs, how the SRL improved the control of the FLC cannot be understood. This warrants further investigation into how the control improvements can be transferred back onto the Expert FLC.

The improved control policies taught through Reinforcement Learning can be used to refine the original Expert FLC through the technique of Policy Distillation (PD). PD is the process of transferring optimal control policies from one controller to another [47, 48]. In current research PD is used to transfer expert knowledge to DNNs to reduce RL training time [49, 50]. There is little research into the use of PD into refining Expert FLCs' navigation control from RL improved control policies. This gap presents a valuable opportunity to investigate the potential of back distilling optimal policies onto a FLC, thereby giving a greater understanding into the improvements made by RL for AR navigation. In turn, this will provide a deeper insight into how the five key developments for ARs can be addressed.

To address the current gaps in AR literature of the lack of quantitative comparison of SRL controllers, the need for evaluating in more complex obstacle environments and to investigate the use of PD in refining the original Expert FLC with the improved control policies, the research question for this thesis is;

Can Fuzzy Autonomous Robot navigation be improved by Supervised Reinforcement Learning?

To answer the research question, this thesis implemented a SRL algorithm to improve the performance of an Expert FLC. The expert control policies of the FLC were transferred to a DNN through a preliminary stage of SL, which was followed by a secondary stage of Reinforcement Learning to refine the performance of the model.

A Deep Deterministic Gradient Policy (DDPG) was used as the RL algorithm due to its ability to train in the robot's continuous state-action space [51]. The improved controller was then used to refine the control of the original Expert FLC through PD. The two developed models were also quantitatively compared to other SOTA controllers, both in simulation and hardware, to determine their performance and highlight areas of improvement for future research.

The research of this thesis will also be used to comment on the viability of implementing the findings and technical developments into the University of Wollongong's (UOW) School of Electrical, Computer and Telecommunication Engineering (SECTE) curriculum through its Intelligent Control subject ECTE441. To ensure the continuing innovation of ARs, students need to be exposed to the area through experimenting and demonstrating, which are of the best methods to create student learning [52–55]. ARs are "revolutionising" modern technology [56], it should also be revolutionising modern classrooms.

The objectives of the ECTE458 thesis project were as follows;

1. Further refine the preliminary navigation improvements of Fuzzy AR through SRL obtained in ECTE451
2. Refine the control of the Expert FLC by transferring the learnt control policies using PD
3. Quantitatively compare the performance of the SRL controller and Improved FLC to other SOTA controllers
4. Implement the Expert FLC and Improved FLC onto a Thymio hardware platform to quantitatively compare navigation performance in a physical environment

Chapter 2

Literature Review

This chapter briefly reviews the current state-of-the-art approaches to AR navigation including Fuzzy Logic Controllers, Discrete Reinforcement Learning, Continuous Reinforcement Learning and other Intelligent Control approaches. Additionally this section investigates methods to improve navigation performance, the difficulties faced by other researchers, weaknesses in other research and how this affects the research and experiments presented in this thesis.

AR navigation is a large area of research, with many different Intelligent Control approaches used to solve the navigation and obstacle avoidance problems. Pandey [9] conducted a review of AR navigation and Obstacle Avoidance Techniques and found autonomous navigation can be split into two separate areas: global and local navigation approaches. Global navigation provides the robot with complete knowledge of the environment, whereas local navigation provides the robot information only through its on-board sensors. Approaches for global navigation are Potential Fields, Dijkstra algorithms and many others. Fuzzy systems, Neural Networks and Particle Swarm Optimisation all provide optimal control in local navigation environments.

The research by Pandey [9] determined that most papers did not tackle the problem of dynamic environments using soft computing techniques. Additionally, many of these papers did not implement and test their controller on a hardware platform. The more complex implementations of AR control, such as RL, trained controllers that were hard to understand due to their black box behaviour, and too computationally large to implement onto lower end robotics platforms. Therefore, it is critical that transferring the control policies of RL controllers onto less complex models, such as FLCs, is investigated. These limitations in current research are discussed more in-depth below.

2.1 Fuzzy Logic Autonomous Robot Navigation

Fuzzy logic systems are effective at learning the control decisions used for robot navigation in a format that is easy for humans to understand the control decisions through their expression in linguistic variables. The use of fuzzy logic robot navigation was investigated by Bobyr et al. [15] where they designed an Adaptive Neuro-Fuzzy Inference System (ANFIS) to control the navigation of a robot through a maze utilising only on-board sensors. ANFIS expresses a FLC in the form of a neural network to enable the system to learn through supervised examples while maintaining the human readability of a FLC. The robot possessed a total of four ultrasonic sensors, three for short range obstacle detection and one for longer range. To evaluate the ANFIS robot controller, the experiment utilised simulation to observe the system's response to 16,471 different speed and angle inputs and compared the response to a known expert system. The experimental data highlighted the improved effectiveness of the ANFIS controller with a Root Mean Squared Error (RMSE) for system response of 2.1, where the expert system had RMSE of 26. Furthermore, the use of a fuzzy logic system reduced the size of the program by 25%. While the robot successfully navigated the maze when implemented on hardware, the maze used was simple and the successful navigation examples presented were 'cherry picked' to provide best examples. Further research is required into the application of fuzzy robot control for more complex obstacle navigation with more robust evaluation criteria. Regardless, the findings highlight the strong ability of fuzzy logic systems for AR navigation and the reduction in computational power required.

The robust navigation performance of FLC was also researched by Chiu et al. [13], where they explored the development and use of FLC for AR obstacle avoidance. The experiment used eight ultrasonic sensors spaced 20 degrees apart on a simulation based P3DX robot with a Mamdani FLC. Additionally, Berisha et al. [14] researched how FLC can be effectively used for hardware based AR navigation control. This robot possessed three ultrasonic sensors that had detection range of $[2, 400]cm$ to show the proximity of an approaching obstacle. The output of the controller were the two wheel velocities of the robot which took ranges of $[0, 4]ms^{-1}$. A grid simulation environment was then used to evaluate the performance of the Mamdani FLC

in reaching a specified target. Both experiments demonstrated the effective ability of FLC to avoid obstacles and navigate rapidly. One major gap in both experiment was the lack of metrics to compare the controller's performance to other obstacle avoidance controller. Also, Chiu et al. [13] notes that the ultrasonic sensors experience dead-zones in close range detection, and suggests image feedback could be used to address this. Hence, the FLCs for the Thymio hardware platform designed in this thesis will need to account for the ultrasonic dead-zone in their control policies. The lack of robust, quantitative metrics will be addressed in this thesis through comparing the control of FLCs to other SOTA models using a collection of metrics.

Where the research of Chiu et al. [13] and Berisha et al. [14] investigated the use of a single FLC to allow AR navigation, Omrane et al. [20] researched how the use of two independent FLCs, one for navigation and another for obstacle avoidance, could improve the navigation performance of an unicycle-based AR. The research designed a target seeking Sugeno FLC with 35 inference rules with two inputs; the distance and angle to the target. This initial FLC dealt with the navigation towards the target, with a second developed FLC that took inputs of the Robot's Infrared (IR) Sensors (Nine total covering a 360 degree observation of the robot's environment) and aimed to avoid incoming obstacles. A coordination unit was appended at the end of the control sequence to choose between either objective depending on the current state of the robot. The experiment concluded that the use of the dual FLCs led to a robust controller for AR navigation, although the research neglected to compare the performance to other SOTA controllers nor did it implement it on hardware. The robust performance of FLCs highlighted by the research of Omrane et al. [20] is one piece of many that justify the initial use of an Expert FLC for AR navigation in this thesis. The idea of a dual objective controller was also adapted in this thesis for a RL controller. The lack of quantitative comparisons conducted by Omrane et al. [20] will be remedied in this thesis to investigate the real-world viability of both controllers.

Fuzzy logic controllers are also effective in hardware-based obstacle avoidance, where sensor noise pose additional complexities for AR control. This was investigated by Ren et al. [21] where they designed a FLC for a HR-1 robot to both avoid obstacles and seek a target location. The experiment split the two objectives into individual

Mamdani FLCs, similar to the research by Omrane et al. [20]. Both controllers outputted the velocity of the wheels and the required angle of rotation, which take values $[0, 1]ms^{-1}$ and $[-0.3, 0.3]\text{rad}s^{-1}$ respectively. The inputs for the system were the distance and angle to the target location and obstacles. The experimental results highlighted the effectiveness of the Mamdani FLC controller in both simulation and hardware scenarios. The hardware used by the experiment simultaneously drove the robot motors and calculated the required velocities without loss of accuracy. One drawback of the experiment was not quantitatively comparing the performance of the Mamdani FLC to other AR Controllers to provide a metric baseline for performance improvement. Hence, when developing hardware based FLC sufficient hardware needs to be used to ensure simultaneous tasks can be run without loss of accuracy, and metrics need to be developed to test and compare the FLC performance to other controller architectures.

Hardware based fuzzy logic controllers are effective at obstacle avoidance and respond rapidly to changing environments when given a last action observation. AR navigation FLC implementation on field programmable gate arrays (FPGA) was investigated by Chiu and Lee [22]. The paper developed a Mamdani FLC using four inputs; three ultrasonic sensors and a last action observation informing the controller of the previous action taken. Two ultrasonic sensors were used to determine close-range obstacles and the third sensor used to determine range to incoming obstacles. The Mamdani FLC was ported to a FPGA Altera DE2-115. To evaluate the effectiveness of the hardware implementation, the experiment created several basic obstacle avoidance scenarios, and recorded the system's response with and without a last action observation. The results emphasised that by providing both obstacle and last action information, a hardware FLC more effectively avoided obstacles. One major gap in the research was the lack of a defined target, meaning the robot needed to navigate through a simple obstacle field. Further research needs to be conducted on the effectiveness of last action observations when a FLC is both avoiding obstacles and seeking a target. This thesis uses last action observation in implementing a RL controller to improve the navigational control.

Cui [12] investigated the use of multivariate data fusion for FLCs to improve performance. The robot used by the experiment possessed three IR sensors for close

range obstacle detection, six ultrasonic sensors for longer range obstacle detection and a electronic compass to determine the angle between direction of travel and target location. The information from each of these sensors was inputted into the pre-processing layer to fuse the information and calculated the front distance (FD) to an obstacle, the difference between left and right obstacle distances (DD) and the angle to target location. This reduced the input space dimension of the second layer FLC, which calculated the required velocities for the left and right wheels. The experiment then used a simulation grid-map environment to evaluate the performance of the controller. The experiment found that the reduced input FLC effectively navigated through obstacles to the target. The experiment did not include quantitative metrics of the controller's performance but used qualitative observations. While the FLC was shown to be an effective method for obstacle avoidance and target seeking, the paper only presented one example of successful robot navigation using circular obstacles. Further research must be conducted on the application of multi-sensor data fusion in more complex obstacle problems. Regardless, the paper highlights that reducing the input space of a FLC reduces the computational complexity without sacrificing the performance of the system.

Yang-zhi et al. [18] also investigated the use of data fusion in reducing computational complexity for AR navigation FLCs. The experiment used a two-layer FLC design to first reduce the dimensionality of the input data and then calculate the required robot wheel velocities, similar to the research by Cui [12]. When compared to a single layer FLC with inputs from all eleven ultrasonic sensors and 2048 total rules, the two-layer FLC had 67 total rules. In a simulation grid with several obstacles the two-layered FLC took 3.11 seconds to reach the target, while the single-layered FLC took 3.24 seconds. While the two-layered FLC has a reduced time to target (TTT), the paper did not investigate other quantitative metrics by which to compare the models and did not evaluate with other non-fuzzy based controllers, which was also a limitation in the research conducted by Cui [12]. In addition a hardware implementation of the developed two-layer FLC was not implemented. The experiment by Yang-zhi et al. [18] provides further avenues of research into testing performance with more specific metrics and on a hardware platform. Similar to Cui [12], it also highlights the ability to reduce computational complexity without sacrificing system

performance.

Both the research by Cui [12] and Yang-zhi et al. [18] highlight the relative stable performance of FLC AR controllers when their input array dimensions are reduced. This allows for smaller hardware platforms to be used while still maintaining optimal control. Both papers require further research using quantitative metrics and evaluation on hardware platforms to confirm the increase in performance successfully migrates across to hardware.

The performance of Fuzzy systems can be refined by the use of ANFIS, which have been shown to successfully navigate with multiple other robots present. The use of ANFIS in robot navigation was researched by Pothal and Parhi [11] to alleviate the black-box behaviour of Neural Networks. For a single robot, the average TTT over 10 navigation cases in simulation was 16.21 seconds while on hardware it took 18.10 seconds. With four robots the average TTT for simulation and hardware were 15.76 and 17.61 seconds respectively. Yet, while the ANFIS controller can refine the control of Fuzzy systems, it is limited by the provided expert knowledge. No new, more optimal policies can be learnt by the robot online during its exploration. Alternative methods of Fuzzy navigation improvement need to be investigated that allow for online learning while also maintain the ability to understand the navigational control decisions.

Summarising the research presented in this section of the Literature Review has demonstrated that Fuzzy-based AR Control provides both robust, effective navigation control while also having a model architecture that is easily understood by humans. The Fuzzy Logic Controllers used in research included the ANFIS system by Bobyr et al. [15] and Pothal and Parhi [11], single FLCs resesearched by Chiu et al. [13] and Berisha et al. [14], dual FLCs explored by Omrane et al. [20]. Reduced input dimension FLCs were developed by Ren et al. [21] and Chiu and Lee [22] that mainained robust and effective navigation control. Last action observation [22] and reducing the complexity of the state input space [12, 18] were two improvements to FLCs that will be utilised by this thesis. A collective gap in the reviewed research was the lack of complexity in obstacle fields navigated by the robots. Real world applications of ARs will require navigation, target seeking and obstacle avoidance

in complex and highly dynamic environments. Thus there needs to be an increased focus on evaluating the performance of AR navigation in suitably complex environments, an issue that will be addressed in this thesis. Additionally, many papers did not investigate implementing their FLCs onto hardware platforms to determine if optimal control is maintained in non-ideal conditions. This is important as consumers and real world applications of ARs will require the ability to function in a variety of harsh and unknown environments. Hardware based control was also an area that this thesis focused on developing to demonstrate robust controller performance of Fuzzy systems. The research of Chiu et al. [13] and Ren et al. [21] have highlighted the issues faced by hardware FLC implementations, such as sensor dead-zone and lack of computational power. These issues will be recognised when developing the hardware platform of this thesis.

2.2 Reinforcement Learning Autonomous Robot Navigation

Reinforcement Learning is an technique of teaching control through allowing the learning agent to explore the environment and to be rewarded or punished for its actions. It has been shown to be an effective and robust algorithm to teach ARs navigation, an area that was researched by Huang et al. [31]. The research used a Neural Network based Q-learning RL model to store the state-action pairs for the observed environment. To select a discrete action for an observed state of the environment, a greedy action selection was implemented which selects the action that is calculated to have the greatest current and future reward. The reward signal is calculated for each time step when the robot goes forward, rotates or collides with an obstacle with values of 0.01, -0.01 and -1 respectively. The research observed that through training over 500 epochs the robot reduced its obstacle collisions to six while a table based Q-learning architecture had 11 collisions. The research highlights the strong improvement in performance by using a Neural Network based Q-learning approach, with the system learning to avoid obstacles more quickly as opposed to the traditional lookup Q-table approach. One of the major gaps of the paper was not researching the training performance of the Q-learning controller when the robot was tasked with an additional goal-seeking objective, nor comparing the performance

improvements to other intelligent control methods. This thesis will investigate the use of a Neural Network Q-Learning with both obstacle avoidance and goal seeking objectives due to the highlighted strong performance in learning optimal control policies and will compare its performance to alternative controllers to address the gaps in the research by Huang et al. [31].

Similarly to Huang et al. [31], Duguleana and Mogan [38] researched the use of Neural Network based RL to teach navigation and obstacle avoidance to ARs. The obstacle field that the robot was trained in consisted of regular circle and square shapes. Global information of obstacle field position was provided to the robot and used to calculate its position relative to obstacles and the target location, which is a major difference to the local information provided in the research by Huang et al. [31]. The RL algorithm is implemented through the use of Q-learning with a NN based Q-table. The robot is rewarded for moving to a state that moves it away from obstacles or towards the target location, and is punished for entering a state of a higher risk of obstacle collision. After conducting training on four simulation scenarios for 200 training episodes each, the controller was implemented in a real-world obstacle field. After 20 trials in the real-world navigation problem, the proposed RL global view algorithm reached the target location approximately 12 seconds faster then the in-built navigation software, travelling at a faster speed through the environment. In real world applications of ARs, global environment information is typically not provided. Further research needs to be completed using only local environment information in conjunction with evaluating the performance in comparison to other SOTA AR navigation controllers to build on the research of both Huang et al. [31] and Duguleana and Mogan [38].

Alternatively, Macek et al. [29] demonstrated that AR navigation can also be learnt through combining unsupervised learning with reinforcement learning. The research used a Neural Network to represent the state-action Q-table. Inputs consisted of eight ultrasonic sensors taking values between $[-1, 1]$ and a discrete output space, similar to the continuous state - discrete action space in the research by Huang et al. [31]. A winner-takes-all learning algorithm was used to teach the network through back propagation, and greedy action selection was used to determine the action taken at each time step. The robot received a penalty when it collided with an obstacle

or was less than 15 centimetres away from an obstacle. After a period of training the robot successfully avoided obstacles while continuously moving forward. The use of unsupervised winner-takes-all reduced the time for the reinforcement learning algorithm to converge to a solution. A gap in the research design was penalising the robot for travelling close to obstacles while still avoiding them. This could restrict the RL algorithm from learning a more optimal solution. The research by Macek et al. [29] further highlights the findings of Huang et al. [31], which demonstrated the strong ability of Neural Networks to accurately represent the state-action table of a RL problem.This approach will be used by this thesis. It has also shown the ability of hybrid RL algorithms, which utilise an initial phase of training the model offline, to improve the control policies taught through RL. When designing the reward function for the project, the robot will only be punished for collisions, not for travelling close to obstacles, to train more optimal control policies.

The process of using RL to teach autonomous navigation can be improved through splitting it into two phases; an initial stage of expert observation and a second stage of learning through exploration. This was investigated by Smart and Pack Kaelbling [30] where they initially taught an RL algorithm through observing the robot navigation of an expert system's control policy. This is similar to the hybrid RL research by Macek et al. [29], but SL is used in place of unsupervised learning. Once the RL algorithm's policy had been taught a basic understanding of the environment, it moved onto a second stage of learning though exploration. It was rewarded with a value of 1 for reaching a target or penalised with a value of -1 for colliding with an obstacle. After 30 epochs of phase 1 training and 30 epochs of phase 2 training the robot actor reduced the number of time-steps required to reach the target from 150 to 58, with the second phase of training optimising the learnt policy from the expert system. The research found that by using two phase approach, the time required to converge to an optimal solution was significantly reduced. This thesis will utilise two phase training to reduce training time and enable the RL algorithm to more effectively learn through exploration.

Fathinezhad et al. [44] also investigated the use of combining SL examples and Reinforcement Learning by implementing a hybrid SRL, which greatly improves training time and performance of AR navigation FLCs. Fathinezhad et al. [44] implemented

Supervised Fuzzy Sarsa Learning (SFSL) to an E-puck robot that possessed IR sensors and VGA cameras. Initially the robot actor observed an expert system's navigation data to adjust its fuzzy membership functions, similar to the process researched by Smart and Pack Kaelbling [30]. In the RL second stage, the actor FLC continues to learn through exploration with the obstacle environment. Each training epoch consisted of 500 episodes, and at the end of the training the SFSL taught E-puck robot had an average path length to the target of 107.09 while a purely Fuzzy Sarsa Learning taught robot had an average path length of 124.001. These results highlight the improved performance for reinforcement learning by using an initial stage of SL to allow the actor to better understand and explore the environment.

The use of SRL was also researched by Ye et al. [45] to rapidly teach obstacle avoidance. The research designed a neuro-fuzzy logic controller which was initially taught obstacle avoidance through SL over 8000 training examples. It was then allowed to learn through exploration with an RL algorithm and taught for an additional 100,000 steps. After training the neuro-fuzzy logic controller was shown to effectively avoid obstacles and be robust to ultrasonic sensor noise while also maintaining a smooth navigational path. The initial supervised training reduced the training time to reach an optimal solution. Using prior expert knowledge accelerated the RL algorithm and ensured a robust controller for AR navigation.

The research of Smart and Pack Kaelbling [30], Fathinezhad et al. [44] and Ye et al. [45] have collectively shown through first-hand experimentation and quantitative data that using an initial phase of SL to supplement a secondary phase of Reinforcement Learning significantly reduced training time and improved navigation performance of AR navigation. One of the key requirements for the SL to be effective is to provide broad and appropriate training examples. These training examples need to highlight obstacle avoidance, target seeking and also the presence of no obstacles due to the sparse training environments.

In contrast to the SRL of the above research, where the experiments used ultrasonic sensor values as inputs for the SL phase, Kahn et al. [57] explored the use of SRL through providing the video feed from a forward-facing video camera to teach AR

navigation. The experiment represented a Generalised Computation Graph with a Recurrent Neural Network due to its ability to robustly process high dimensional data. A Bellman Error Function was used to determine the difference between the true H future time-step reward function value and the approximated value from the computation graph. The learning agent had a continuous action space of the vehicle's steering angle. Each training episode extended for 1000m of distance travelled or an obstacle collision, with the next training episode taking place at the previous end location. After training the research's Computation Graph approach achieved a 50% and 700% further final travel distance than the best other navigation method for the simulator and hardware implementation respectively. The significant improvement in controller performance aligns with the previous research discussed about SRL. The experiment did not investigate how the inclusion of a target seeking objective affected the performance of the model. Target seeking is an important aspect for commercial viability that requires research. It will be considered in this thesis, as it is a key objective in the real-world use of AR navigation.

Similar to the idea of creating two separate FLCs for navigation and obstacle avoidance in the research by Omrane et al. [20], Wang et al. [58] adapted the dual model to Deep Reinforcement Learning (DRL) for AR navigation. The two sub-networks select their individual optimal action and an action scheduler determines the final action of the controller. The simulated robot was equipped with a laser range finder and also knew the relative location of the target. The controller's action scheduler outputs an action from a set of five choices; up,down, left, right or no action. The environment consisted of obstacles moving in a constant direction with a constant speed. Criteria for ending a training episode were; obstacle collision, exceeding 100 seconds of simulation run time, or reaching the target. To train the robot's actions for obstacle avoidance a DRL double Q-learning agent was used. The double Q-learning agent was chosen as it mimics the human interpretation of obstacle detection and has been used previously to achieve SOTA performance. The target navigation side of the controller consisted of an offline and online model, both represented by fully connected neural networks. The offline model is adapted from a pre-trained simple path navigation model. This is then used in conjunction with

online RL to further optimise the controller to the dynamic environment. The experiment used an obstacle field with 20 moving obstacles, rewarding the controller for reaching the target and taking a step without collision, and penalising the controller if it collided with an obstacle and for each time step it wasn't at the target. It compared the model's performance to five other controller models. In total, 7000 training steps were used for training and evaluation. The double Q-learning agent outperformed all five other models, performing with a higher success rate in each of the four training scenarios while also requiring fewer time steps to reach the target location. The model performed 22% better in terms of success rate of reaching the target in comparison to the next best performing model. A dual RL controller will be used in this thesis as a comparative model due to its SOTA navigation control.

Di Mario et al. [26] observed that the navigation control of a Q-learning RL agent is reduced when the state observation space is discrete. Di Mario et al. [26] compared the performance of particle swarm optimisation (PSO) and RL in teaching obstacle avoidance to multiple robots. The RL Q-values were encoded into two Neural Networks with discrete action spaces, with one having a discrete state observation space and another having a continuous one. Each algorithm was trained for 1000 episodes and their best solutions evaluated through a quantitative fitness metric based on the robot's speed and obstacle collisions. The Q-learning agent obtained a fitness metric of 0.72 with a discrete state space, but increased its performance to 0.73 with a continuous state space. This demonstrates using a continuous state space for AR navigation RL algorithms improves obstacle avoidance performance. On account of this, a continuous state observation space will be used by this thesis.

Deep Deterministic Policy Gradient (DDPG) Reinforcement Learning outperforms stochastic actor-critic models in large dimensional, continuous state-action spaces. Continuous state action spaces are important in Reinforcement Learning as it allows more robust and effective AR navgation controllers as researched by Di Mario et al. [26]. The comparison of DDPG and stochastic actor-critic models was investigated by Lillicrap et al. [51] where they used a Deep Neural Network representation of a Deterministic Policy Gradient to estimate the Q-value function for a RL algorithm. Typical Q-learning techniques cannot be used in continuous state-action spaces as the dimensionality of the state-action data is too large. The paper compared the

learning progress of DDPG and discrete action Deep Q-learning agents on tasks including a pendulum swing-up, a driving simulator and a puck hitting task. The experiment trained each agent over one million steps and quantitatively compared their reward signals, revealing the DDPG maximised its reward in all tasks and learnt optimal control policies in a shorter amount of time. One drawback of using a DDPG in a high dimensional problem is the requirement of a large number of training episodes to allow the model to find the most optimal solution. Regardless, the research by Lillicrap et al. [51] has highlighted that in environments with continuous, high dimensional state-action spaces such as AR navigation, DDPG is an efficient and robust method to teach optimal control policies and will be used in this thesis to teach AR navigation.

Reinforcement Learning can not only be used to teach AR navigation to land-based robots as highlighted by all the previous papers. It can also be used in teaching navigation control to a fleet of unmanned surface vehicles (USV) for marine applications as shown in the research by Wu et al. [39]. The research proposes to use an RL ANOA (AR navigation and obstacle avoidance) to teach a fleet of USV robots to navigate in marine environments. A Deep Q-network (DQN) was used to handle the continuous state-action space of the marine environment. A duelling Q-network approach was used, where the Q-function estimator is split into two separate components; one part to calculate the state-action future reward and the second part to determine the future reward value for taking a specific action. This assists in learning a robust, optimal control policy. Model input consisted of a video feed from the USV. The outputs to control the robot were; yaw, surge and sway. The USV was trained in both a static and dynamic environment for over 10,000 episodes and was compared to DeepSarsa and non-duelling DQN learning agents. The research found all three learning algorithms taught autonomous navigation of the same robustness and accuracy, but the proposed ANOA approach learnt the optimal policy in a faster time than both other approaches. For the dynamic environment ANOA learnt the optimal policy in 750 episodes, while the other two approaches took approximately 2500 episodes. The research of Wu et al. [39] strengthens the findings of the previous research by showing that Reinforcement Learning is a SOTA method to teach navigational control not only for land based ARs, but also in marine environments.

While the research of Macek et al. [29] highlighted the robust control policies taught by UnSRL, the research of Smart and Pack Kaelbling [30], Fathinezhad et al. [44], Ye et al. [45] and Kahn et al. [57] highlighted the same for SRL, research completed by Luviano and Wen Yu [59] explored a hybrid Fuzzy Reinforcement Learning approach. The experiment used two Khepera robots with five ultrasonic sensors positioned around its body. The sensors provide three discrete values denoting the closeness of incoming obstacles. The robots have four actions they can take at each time-step: move forward, turn clockwise, turn anti-clockwise and don't move. The robots also have state information about their position and velocity locally, and were provided a pre-planned path to navigate. Each time-step is 0.4 seconds long, and each episode is 400 seconds long. The experiment used triangular membership functions to fuzzify the state information. They compared the performance of the Fuzzy Q-learning to a typical Q-learning approach. In the real-world experiment, the fuzzy Q-learning approach required 35 seconds of training time while the original Q-learning approach required 160 seconds, highlighting the reduced training time required for a Fuzzy based RL system. The fuzzy approach was also more accurate in travelling towards the target location. One limitation of the research was in failing comparing the performance of the Fuzzy Q-learning controller to a Fuzzy Reinforcement Learning model that was not provided a pre-planned navigation path. In a large number of commercial uses for ARs, such as military or search-and-rescue, pre-planned navigation paths would not be known or provided to the robot. Therefore, further research needs to be conducted to determine if the robust control policies taught by Fuzzy Reinforcement Learning transfer to the case where no pre-planned path is provided.

Research has shown the robust and strong controller performance for AR navigation through environments with only local information available. Kato et al. [40] researched the performance of Reinforcement Learning in navigating through a topologically mapped dynamic environment, where the general environment was mapped prior to the robot navigating towards a target. An Adaptive Monte Carlo Localisation algorithm was used for identification of different types of obstacle nodes (such as T-intersection, etc.) and a rotary encoder on the robot was used to localise its position in the environment. The robot mapped the finer details with a laser range

finder. Reinforcement Learning was used to teach the robot to navigate between the calculated path nodes while avoiding obstacles in the path through detection via the laser range finder. The paper used the Fast Robot 2D Simulator to train the robot in a training environment. The RL agent was represented by a Deep Q-Network with four hidden layers, and outputting one of three actions for the robot. The research found that the robot was able to successfully navigate in both static and dynamic environments, where the first stage mapped the topographical map through random exploration that was used to path-plan and allow the robot to reach its destination. The robot did not perform optimally in avoiding dynamic obstacles, with only a 58% success rate. A major assumption of the research is that primarily that the environment is relatively small to allow mapping, and thus secondly that the RL navigation controller will have access to this global environmental information. Further research is required to verify that a Deep Q-Network approach is viable for navigation with only local environmental information available. There is potential to adapt the Adaptive Monte Carlo Localisation algorithm to identify types of obstacle nodes in real-time as the robot navigates and use those to inform the controller.

Reinforcement Learning bridges the gap that Fuzzy Logic Controllers potentially face, the gap between the programmed control policies from expert knowledge and optimal control policies through allowing the AR to learn through its own navigational experiences and mistakes. Deep Reinforcement Learning, which represents the Q-function and control policies through the use of Neural Network, allows the development of optimal, robust and fast-learning navigation controllers as shown by Huang et al. [31] and Duguleana and Mogan [38]. Combining the use of Reinforcement Learning with Unsupervised Learning [29] or SL [30, 44, 45, 57] has been shown to reduce training time while teaching robust control policies through giving the learning robot navigation experience prior to training via Reinforcement Learning. Other training methodologies to improve navigation performance include dual Deep Reinforcement Learning agents for goal seeking and obstacle avoidance objectives [58], using a continuous state space [26], implementing a collective buffer

playback between multiple learning agents [60], creating a hybrid Fuzzy Reinforcement Learning controller [59] and using DDPG as training algorithm in large dimensional, continuous state-action spaces [51]. All these methods of improvement will be incorporated into this thesis to ensure that robust and SOTA controller performance is taught to the AR agents. A gap identified in the AR navigation review by Pandey [9], and highlighted in the review of literature presented above, is that a large number of experiments did not implement their controllers onto a hardware platform and compare performance to the original software based controller. One of the main objectives of this thesis, influenced by this large collective gap in research, is to implement the trained controller onto a hardware platform to identify the difficulties in real-world applications of AR and potential areas of improvement or innovation.

2.3 Alternative Intelligent Control Approaches For Autonomous Robot Navigation

While Fuzzy Logic Controllers, Reinforcement Learning, hybrid Fuzzy Reinforcement Learning effectively teach AR navigation there are many alternative Intelligent Control techniques that have also been researched. The following section of the Literature Review will explore these alternative techniques, their performance, drawbacks and how they can be implemented to quantitatively compare to the performance of a Reinforcement Learning approach.

Evolutionary Programming is one Intelligent Control approach that can be used to teach AR navigation. The use of evolutionary programming (EP) to optimise the performance of a robot's autonomous navigation system was investigated by Boufera et al. [25]. The robot possessed three ultrasonic sensors measuring obstacle distances on the left, middle and right of the robot taking values between [0,8]. The research converted an expert Mamdani FLC to a genetic algorithm based problem with each gene of the chromosome consisting of 10 inference rules all relating to the same control decision scenario. The initial population was generated from the initial expert system and new population created through mutation. The fitness function used to calculate the suitability of each solution was determined by the number of obstacle collisions made, turns made in navigating, the distance, and time taken to

reach the target. After optimising the expert system with the evolutionary algorithm it was observed that the system had improved obstacle avoidance and target finding performance in the specific starting positions it was trained in. One of the major gaps in the research was the lack of verification on the robustness of each solution. While the EP improved performance for specific starting positions, it is unknown if this learning transferred over to new starting positions or in new obstacle fields. This prompts further research into the suitability of EP for robust navigation control policies.

Instead of using mutation of fit candidate solutions in the research by Boufera et al. [25], Krell et al. [61] researched the use of Particle Swarm Optimsation (PSO) to teach AR navigation. PSO uses a population of solutions to search the space for an optimal control policy, where each agent is affected by the global best and over time converge to an optimal solution. To navigate to the target the robots first explore the environment using a random walk and maps the obstacles with a LIDAR sensor. After this stage the PSO algorithm is used to plan the path for the robot. The paper used a Pure Pursuit Path Tracking algorithm to then smooth the planned path by the PSO algorithm. The environment was created through a grid pattern, where 0 denoted nothing and 1 denoted an obstacle. The environments used were an open field with several obstacles and a hallway environment. The research found that the PSO algorithm was able to effectively plan a path for the AR using under five waypoints in all environment scenarios. Within 200 iterations of a maximum 1000, the algorithm had converged to an optimal path-planning solution. Yet, the robot is required to randomly explore the environment prior to the navigation task to map out the obstacles. The PSO is then only used for path planning using the recorded global knowledge. This does not effectively tackle to problem of robot navigation in large, unknown areas as the robot would require a large amount of time to search the environment for all obstacles.

The investigations of using RL and other Intelligent Control approaches, such as PSO, to create optimal AR controllers all share one major limitation. The methods by which these investigations store the AR control policies, mostly using DNNs, are black boxes where the reasoning behind each control decision is unknown. Therefore, while these techniques result in SOTA navigational control, they cannot be used to

further understanding on how ARs navigate. PD is the technique of transferring control policies from complex controllers to simpler ones. The research by Rusu et al. [47] and Traoré et al. [50] have shown PD's ability to simplify control policies taught by RL while still maintaining SOTA performance. Yet, research has neglected the potential for PD to simplify control policies taught by SRL back onto the original FLC that was used as an expert. This gap presents an opportunity to investigate the use of SRL to improve the navigational control of a Fuzzy AR. PD will combine the interpretability and robustness of FLCs with the ability to train SOTA control policies through RL. It will allow for the limitations of FLCs and RL, discussed in Sections 2.2 and 2.1 respectively, to be simultaneously addressed.

While the alternative Intelligent Control approaches outlined in this section, the EP algorithm developed by Boufera et al. [25] and PSO algorithm by Krell et al. [61], result in optimal AR navigational control, they fail to allow the AR to learn from its own exploration experiences. To provide the ability for continual improvement of the AR, it needs to be able to continuously learn online as it navigates. Therefore, this is the reason why SRL is the technique chosen to improve AR control. Its development is discussed below in Chapter 3.

Chapter 3

Research Design

The following section discusses the technical details of the simulation environment, hardware platform, FLCs, SRL algorithm, PD and the metrics by which navigation performance was evaluated. This will answer the research question *'Can Fuzzy Autonomous Robot Navigation be improved by Supervised Reinforcement Learning?'*. It also describes the use and influence of the research presented in the Literature Review in the Research Design. The research design is set out in the following order:

- **Section 3.1: Robot Simulation Model -** Outlines the details of the simulation environment and robot model used in the thesis experiments.
- **Section 3.2: Mamdani Fuzzy Logic Controller -** Outlines the development of the Expert FLC and reasoning behind design choices.
- **Section 3.3: Reinforcement Learning Algorithm -** Outlines the technical details of teaching AR navigation through Reinforcement Learning.
- **Section 3.4: PD -** Outlines the process of refining the Expert FLC's control from the SRL controller's optimised policies
- **Section 3.5: Evaluating Model Performance -** Outlines the metrics and alternative controllers used to evaluate the developed controller's performance
- **Section 3.6: AR Hardware Implementation -** Outlines the development and evaluation of a hardware based AR.

3.1 Robot Simulation Model

MATLAB will be used to simulate the environment, robot exploration and Autonomous Navigation training algorithms. The robot simulation model that will be used is the model package provided to the University of Wollongong's ECTE441 students. As part of student's pre-existing project they are tasked with creating a Mamdani FLC to control a robot. By using the same simulation environment and robot model as that in ECTE441 it will allow for the viability of the developments of this thesis for use in the subject's practical projects to be evaluated. The robot model and obstacle environment of the simulation model are discussed below.

3.1.1 Robot Model Simulink Block

The simulation robot model used in this thesis is contained within a MATLAB Simulink block. The model has the kinematic parameters of the robot encoded into the Simulink block to simulate a physically realistic robot. The kinematic model of the robot is the same as the models used by Cui [12], Chiu et al. [13], Tzafestas et al. [24], Filipescu et al. [62]. At each time step of the simulation, the robot receives the environment state observation, which provides information on nearby obstacle and the target location, and it outputs the desired wheel velocities. The robot model possesses two wheels, each with their own controllable velocity value to adjust the direction of the robot. The following values can be inputted into the robot to control wheel actions of the robot at each time step:

- **Left wheel velocity** ($v_L \in [-2, 2] ms^{-1}$)
- **Right wheel velocity** ($v_R \in [-2, 2] ms^{-1}$)

At each time step the robot provides feedback via output signals of its relative position to the target location and any nearby obstacles in its field of vision. This is summarised in the following state outputs:

- **Distance to the target**: Euclidean distance between robot and target ($D \in [0, \infty) m$)
- **Odometer**: Distance travelled by robot ($O \in [0, \infty) m$)
- **Ultrasonic Sensors**: Integers denoting location of obstacles less than $0.5m$ away ($U_S \in \{-9, -5, -2.5, 0, 2.5, 5\}$)
- **Ultrasonic Distance**: Non-linear value of distance to obstacles immediately in-front of robot ($U_D \in [0, 25]$)
- **Theta**: Angle between target angle and direction robot is facing ($\theta \in [-180, 180]^o$)

The six integer values of the U_S input relate to obstacles in the following respective order: {no obstacle, left side, left, middle, right, right side}.

3.1.2 Obstacle Environment

The environment that the robot is simulated in is also the same used as ECTE441, with the environment consisting of a collection of complex shaped polygons. Obstacles are impassable areas and travelling into these obstacle areas results in a collision

and the simulation episode ending. If the robot reaches the target, the simulation episode ends. The simulation environment is shown in Figure 3.1 with the target located at the grid-point (0.9,3.4). The obstacle environment used by this thesis was chosen to be suitably complex to address the simple obstacles used by many AR researchers as discussed in the Literature Review.

3.2 Mamdani Fuzzy Logic Controller

The key objective of this thesis is to improve the performance of Fuzzy AR navigation through using SRL. Therefore, an Expert FLC needs to be developed to provide the initial controller for improvement. The development of such model is detailed below. Mamdani FLCs have been used widely in Fuzzy AR research [12, 15, 18, 20, 22, 25], and will also be used in this thesis as the Expert model.

3.2.1 Membership Functions

The Expert FLC controls the robot model through using the state observations as inputs to determine the required wheel velocities to reach the target location. To fuzzify the input state values of the environment, and defuzzify the left and right wheel velocity action values, triangular and trapezoidal membership functions (MF) will be used. Research has shown that these two types of MFs result in robust and effective AR navigation controllers [12–15, 17, 18, 21, 24, 25, 25, 45, 63].

The MFs for the environment state inputs are shown in Figures C.1, C.2, C.3 and C.4. The MFs for the left and right wheel velocities are shown in Figures C.5 and C.6.

The control decisions of the model are made by the IF-THEN inference rules, which are English based rules that can be easily understood by humans. The inference rules are adapted from the research of Cui [12], Omrane et al. [20], Boufera et al. [64]. The IF-THEN inference rules are described in Table 3.1.

The centroid method is used to defuzzify the wheel velocity outputs. It calculates the centre of the area created by the aggregated fuzzy inference rule outputs. Let x_i be a value in the range of output velocities and $\mu(x_i)$ is its respective membership, then the centroid of the aggregated fuzzy inference rule outputs is described in Equation 3.1. The functions used for AND, OR, aggregation and implication are

product, probalistic OR, minimum and maximum respectively.

$$\text{Centroid} = \frac{\sum_i \mu(x_i) x_i}{\sum_i \mu(x_i)} \tag{3.1}$$

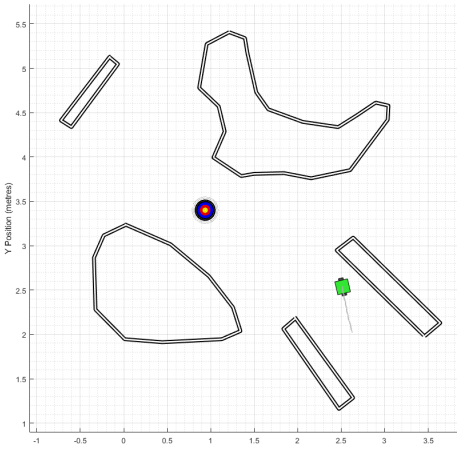


Figure 3.1: Simulation obstacle environment

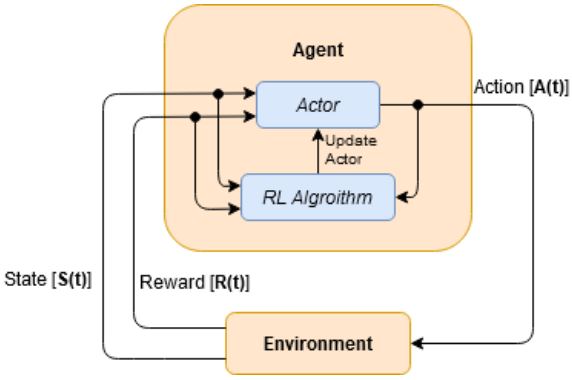


Figure 3.2: Reinforcement Learning process

If	(distance is not zero)	**then**	(left_vel is for_f) and (right_vel is for_f)
If	(ultrasonic is far) and (theta is right)	**then**	(left_vel is rev) and (right_vel is for)
If	(ultrasonic is far) and (theta is left)	**then**	(left_vel is for) and (right_vel is rev)
If	(ultrasonic is in_front) and (sensor is left)	**then**	(left_vel is for) and (right_vel is rev_f)
If	(ultrasonic is in_front) and (sensor is right)	**then**	(left_vel is rev_f) and (right_vel is for)
If	(ultrasonic is in_front) and (sensor is middle)	**then**	(left_vel is rev_f) and (right_vel is for_f)
If	(ultrasonic is in_front) and (sensor is left_side)	**then**	(left_vel is zero) and (right_vel is rev_f)
If	(ultrasonic is in_front) and (sensor is right_side)	**then**	(left_vel is rev_f) and (right_vel is zero)
If	(ultrasonic is close) and (sensor is left)	**then**	(left_vel is for) and (right_vel is zero)
If	(ultrasonic is close) and (sensor is right)	**then**	(left_vel is zero) and (right_vel is for)
If	(distance is zero)	**then**	(left_vel is zero) and (right_vel is zero)

Table 3.1: IF-THEN inference rules for Expert FLC

3.3 Reinforcement Learning Algorithm

One of the limitations of using FLCs for AR navigation is its dependence on Expert knowledge being encoded into the controller. To allow for new, potentially more optimal control policies the AR needs to be able to learn through its own experiences. Improving Fuzzy AR navigation performance through the use of SRL is the main objective of this thesis, and the process implemented will be discussed below.

3.3.1 Reinforcement Learning Background

Reinforcement Learning is an area of Intelligent Control that provides a family of algorithms to teach a learning agent through the use of environmental exploration. A reward function provides feedback to the agent during each training episode. Positively valued reward signals are received when the learning agent enters a state or

takes an action that is optimal, and a negative penalty value is received from the reward function when the state or action is poor or non-optimal. The overall process is summarised in Figure 3.2. At each time step of the episode, the environment provides the agent with a state observation input. From this the agent determines the action for the next time step which is inputted into the environment and simulated.

The state observations (s_t) that the learning agent experiences at time step t is used as an input to the controller. The action (a_t) that is determined to result in the greatest total future reward is selected for that time step. After the process is complete, the action is taken and the time step moves from t to $t+1$. Thus the agent attempts to maximise the expected future reward in Equation 3.2:

$$E\big(\sum_{\tau=t}^{\infty}\gamma^{\tau-1}r(\tau)\big) \tag{3.2}$$

Where γ denotes the discount factor, or how forward looking the agent is. γ takes a value between $[0, 1]$, and a value closer to $\gamma = 1$ results in a model that attempts to maximise the current and all future time-step rewards by the current time-step action. A value closer to 0 results in a model that only tries to maximise the current time-step reward, it does not consider the future time horizon.

To determine the action with the greatest future reward function, an algorithm is required to estimate the expected future reward for an agent in its current state and taking a certain action and use this information to determine the agent's action. The algorithm estimates the state-action value function $Q(s, a)$, which is the expected future reward for the model if the specified action is taken for the current time-step and the optimal policy is followed from thereon in. The state-action value Q^{π} for a policy π is shown in Equation 3.3. It is equal to the expected value of the discounted reward calculated from the current time step t onwards to time step $t = \infty$, or the end of the episode.

$$Q^{\pi}(s_t, a_t) = E\big(\sum_{\tau=t}^{\infty}\gamma^{\tau-1}r(\tau+1)|s_t, a_t\big) \tag{3.3}$$

The state-action value function $Q(s, a)$ cannot be explicitly represented and is estimated through a learning agent. There are a wide variety of agents that allow Q-value estimation in a discrete action space (Q-learning [65], Deep Q-Network [66] and SARSA [44]) or a continuous action space (Deep Deterministic Policy Gradient [51], Twin-Delayed Deep Deterministic Policy Gradient [67] or Soft Actor-Critic

[68, 69]).

The aim of this thesis is to determine if Fuzzy AR navigation control can be improved through the use of Reinforcement Learning. To achieve this the learned expert policies of the FLC need to transferred to the RL agent to enable further improvement. Since the state and action spaces of the FLC are continuous, the RL agent's state and action spaces must also be continuous to allow for a SL algorithm to transfer the expert knowledge effectively.

The Deep Deterministic Policy Gradient (DDPG) allows for a continuous state-action space while maintaining SOTA performance in large dimensional state-action spaces when compared to Stochastic Policy algorithms [26, 33]. Due to the optimal, improved control policies taught by DDPG, it has seen numerous applications for AR navigation and vehicle control [5, 70–72]. DDPG is readily programmed in MATLAB using its Reinforcement Learning library, with examples of using a SL initial phase for Reinforcement Learning readily available [73], which the thesis' code is adapted from.

The DDPG algorithm utilises an actor-critic model to learn the optimal control policies, which splits the Q-function estimation and policy updating into two separate processes. The critic estimates the Q-value function at each time step and is updated through the error between the estimated and true reward for each time step. The actor controls the learning agent through its stored policy information. The actor's policy is updated at each time step through the reward value estimated by the critic.

3.3.2 Actor-Critic Architecture

To implement the actor-critic model for the DDPG algorithm, a DDPG agent is initialised in MATLAB. The agent consists of an actor, critic and the RL algorithms to update each during training. Both the actor and critic are stored through a Deep Neural Network (DNN). DNNs are widely used in RL research to store actor and critic information [5, 29, 38, 58, 60, 70, 72, 74]. One main advantage of DNNs is its ability to "scale RL decision making to large dimensional problems, which was previously intractable" [56].

The actor DNN architecture is shown in Figure 3.3. The six observation inputs of

the network are the: ultrasonic distance, ultrasonic sensors, distance to target, angle to target and the left and right wheel velocities. The wheel velocities are provided as inputs into the system as last action observations have been shown to increase the navigation performance of ARs [22]. The hidden layers of the actor network consist of Fully Connected Layers (FCL) with 16 neurons each and Rectified Linear Unit (ReLu) activation blocks. The Hyperbolic Tangent output layer provides the velocities for the left and right wheels. The velocities are linearly scaled to between $[-2, 2]ms^{-1}$ before they are inputted into the robot.

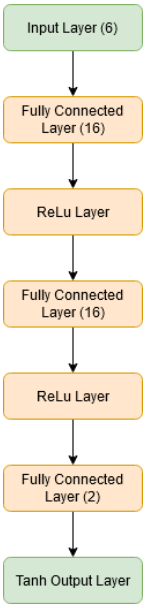


Figure 3.3: Actor DNN structure

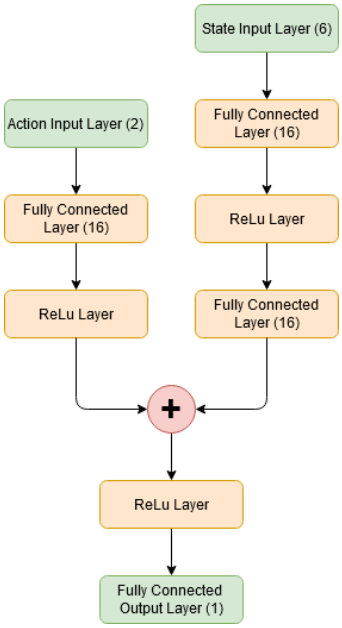


Figure 3.4: Critic DNN structure

The critic DNN architecture, adapted from the work of Castro [73], can be seen in Figure 3.4. It contains two input branches: an action branch and a state observation branch. The observation inputs are the same as those in the actor, and the action inputs are the left and right wheel velocities of the robot. Each branch consists of FCN and ReLu activation layers that are concatenated to a final FCN output layer which estimates the Q-value function for the state-action pair input into the network.

3.3.3 Phase 1: Expert System Knowledge Transfer

Phase 1 utilises SL to transfer the expert control policies stored in the FLC to the actor DNN. This allows the agent to learn more quickly as it already possesses the ability to navigate the environment [30, 44, 45]. This results in a reduced Phase 2

training time and more optimal control policies learnt.

The Expert FLC was simulated in 20 different navigational paths through the environment towards the target in an identical method to ECTE451. Both the observation state and corresponding action at each time step were saved for SL resulting in 24255 training examples. A 90/10 train-test split was used to train the actor DNN through SL. A final Root Mean Squared Error (RMSE) of 0.20046 was achieved with a learning rate of 10^{-3} over 30 training epochs. 16 neurons in each hidden FCN layer was chosen as it reached the desired RMSE of approximately 0.2 while keeping the size of the network small. The flowchart for this process can be seen in Figure C.7.

3.3.4 Phase 2: Learning Through Reinforcement Learning

Following Phase 1, a secondary phase of Reinforcement Learning is used to improve the navigation and obstacle avoidance control of the robot. The DDPG learning agent was implemented into the Simulink model to act as the controller for the AR. The agent's inputs consisted of the state observation, reward value and the IF-DONE signal to specify when the training episode had ended. The high-level Simulink model can be seen in Figure C.8.

The state observation input of the agent is the state output of the robot model, as discussed in Section 3.1.1, and the previous time step action taken by the controller.

The reward value input of the DDPG learning agent specifies if the state-action pair that the robot has selected is optimal or non-optimal and is used as the learning signal. Let D be the distance to the target, U_D be ultrasonic distance to obstacles, θ be angle to target, v_L and v_R the left and right wheel velocities respectively and $H(x)$ the heaviside function. The reward function value for the time step t is shown in Equation 3.4.

$$\begin{aligned} r(t) = \frac{0.1}{D - 0.2}(H(D_t) - H(D_t - 2)) + 100(H(D_t) - H(D_t - 0.35)) - \\ H(D_t - 3.8) - 25H(U_{Dt} - 24.85) + (H(|\theta_t|) - H(|\theta_t| - 30))\frac{30 - \theta_t}{60} + \\ 0.01(v_{Lt-1} + v_{Rt-1}) - 0.01|v_{Lt-1} - v_{Rt-1}|^2 \end{aligned} \quad (3.4)$$

The components of the reward function are as follows:

- **Close to target**: Reward the robot for travelling closer to the target $[\frac{0.1}{D-0.2}(H(D_t)-H(D_t-2))]$
- **At target**: Reward for reaching the target $[100(H(D_t)-H(D_t-0.35))]$
- **Far from target**: Punishment for travelling too far away from the target $[H(D-3.8)]$
- **Obstacle Collision**: Punish the robot for colliding with an obstacle $[25H(U_D-24.85)]$
- **Theta**: Reward robot for travelling in the direction of the target $[(H(|\theta|)-H(|\theta|-30))\frac{30-\theta}{60}]$
- **Forward Velocity**: Reward robot for travelling forward $[0.01(v_L+v_R)]$
- **Spin**: Punish robot for turning quickly/spinning $[0.01|v_L-v_R|^2]$

The 'at target' reward weighting wase increased from the value of 25 used in ECTE451 to 100 to provide greater positive feedback to the DDPG learning agent as it travelled closer to the target. During the initial development stages it was observed the increase resulted in the agent more consistently traveling towards the target. Similarly, the weighting for the 'theta' reward was also increased from $\frac{1}{600}$ to $\frac{1}{60}$ provide greater positive feedback to the learning agent when it was facing the location of the target. The theta reward weighting was increased from that used in ECTE451 to remedy the qualitative observation that the robot would briefly travel towards the target before turning to an alternative direction. The forward velocity reward was also increased from 0.005 to 0.01 to reward the robot more substantially for continuing to travel forward.

Each training episode is 30 seconds long or until one of the following conditions was met, which was inputted into the learning agent via the IF-DONE signal:

- **Target reached**: Robot has reached the target
- **Travelled too far from the target**: Robot has travelled more then 4 metres away from target
- **Obstacle collision**: Robot has collided with an obstacle

The training parameters used for Reinforcement Learning are summarised below in Table 3.2. The mini batch size specifies the number of previous time steps the agent

saves. From the saved samples, the agent randomly selects time steps to train on via the DDPG algorithm. A mini batch size of 256, increased from the value of 128 used in ECTE451, is used to allow the learning agent enough samples from which to learn on while maintaining minimal computational training complexity. Other similar implementations of the DDPG algorithm use the same mini batch size with optimal results [73].

The maximum training episodes specifies the number of training episodes that are run before the RL algorithm stops training the learning agent. The RL algorithm also stops training and saves the agent when the average reward over 100 training episodes exceeds 2500. This was increased from the value of 200 used in ECTE451 to correspond with the increased reward values. The sample time is the time step between each successive sampling window where the robot observes its state and determines the necessary action.

The learning rate was decreased from the value in ECTE451 of 10^{-2} to 10^{-3} to ensure that the agent learnt the most optimal policy by learning and adjusting network values more slowly.

Specification	**Value**
Sample Time	0.025s
Discount Factor	0.99
Mini Batch Size	256
Max Episodes	10000
Averaging Window Length	100
Stop Training Criteria	Average Episode Reward
Stop Training Value	2500
Learning Rate	10^{-3}

Table 3.2: Training specifications for DDPG RL Algorithm

The process for training the actor-critic agent is summarised in the pseudo-code flowchart of Figure C.9. The DDPG learning agent was evaluated when its average reward was above 2500 or it had reached the maximum number of training episodes.

3.4 Policy Distillation

One of the objectives of the thesis was in investigating methods to refine the control of the original Expert FLC through transferring the improved control policies stored in the DDPG actor. This transfer of control policies will be completed by using PD.

After optimising the control policies through SRL, the navigational paths taken by

the robot will be saved and appended to form a large training set. The training set will be the same structure as described in Section 3.3.3. The navigational training set will be used as the 'teacher', and the Expert FLC the 'student'.

The PD algorithm tunes the Expert FLC in two stages. It has been adapted from the MATLAB example of predicting fuel consumption [75] and inspired by the research of Rusu et al. [47], Buciluǎ et al. [48]. It has not been directly implemented directly from any research as the use of PD for FLCs has not being investigated. The first stage of the algorithm adjusts the inference rule base while keeping all other parameters of the model constant. The initial FLC has 11 inference rules, and PSO is used to select an additional nine rules that produce the lowest RMSE between the tuned model outputs and the navigational training data outputs. 20 training iterations was chosen to prevent over-fitting to the training data set.

Once the preliminary stage of the inference rule base tuning has finished, a secondary stage of tuning the input MFs using a pattern search algorithm is initiated. The algorithm modifies the parameters of the existing MFs for each input, but does not add or delete any MFs. The pattern search algorithm tunes the MFs to minimise the RMSE between the model and training outputs. 60 training iterations were used to tune the MFs to avoid overtuning. Once the PD algorithm was complete the Improved FLC was evaluated using the metrics discussed in Section 3.5. The results of PD and the performance of the Improved FLC is discussed in Section 4.2.

3.5 Evaluating Model Performance

To determine the performance of the learning agent's final control policies after both Phase 1 and 2, a robust set of evaluation metrics were developed, which are detailed below:

- **Time to reach the target (TTT)**
- **Total episodic reward (TER) [26]**

The TTT metric provides a quantitative measurement at the speed which the controller avoids obstacles and navigates towards the target. A smaller TTT will indicate more optimal control policies. While some research uses total path length as the

main performance indicator of a model, a TTT metric indicates both a fast and efficient path travelled by the robot and is used in research regularly [26, 30, 45, 58, 61].

In addition to the TTT, the TER metric is another measure of controller performance. While the TER for a controller that quickly navigates to the target location, it will be even higher for a controller that presents more optimal navigation behaviourssuch as smoothness of navigation and preemptive avoidance of collisions. By using both TTT and TER to evaluate the controller's performance the speed and quality of the robot's navigation can be robustly analysed. The TER is the cumulative reward value obtained by the robot over the length of the testing episode.

To determine the improvement in performance through the SRL optimisationit will be compared to several other SOTA models. As discussed in the literature review, AR research fails to provide sufficient quantitative comparison of the developed controllers to conclusively comment on the performance and viability in commercial applications. To address this limitation, this thesis will use the following SOTA models for comparison.

- **Expert FLC:** Detailed in Section 3.2
- **Dual Objective DDPG Agent:** Detailed in Section 3.5.1
- **Improved FLC:** Refined by PD as detailed in Section 3.4

All four controllers will be evaluated from 16 different starting positions in the simulation environment. These 16 starting positions were chosen to provide varied testing conditions with a range of concave, convex and multi-object avoidance scenarios. These obstacles provide a complex navigational environment for the robot to address the lack of complex environments in current research discussed in the Literature Review. The 16 starting positions are listed in Table C.1 below with the reasoning on their selection, and they can be seen visually in Figure 3.5. The evaluation metrics used for the hardware AR implementation is discussed in Section 3.6. Figure 3.6 presents a flowchart of the basis of each developed controller and how they were compared to determine if SRL can be used to improve Fuzzy AR navigation.

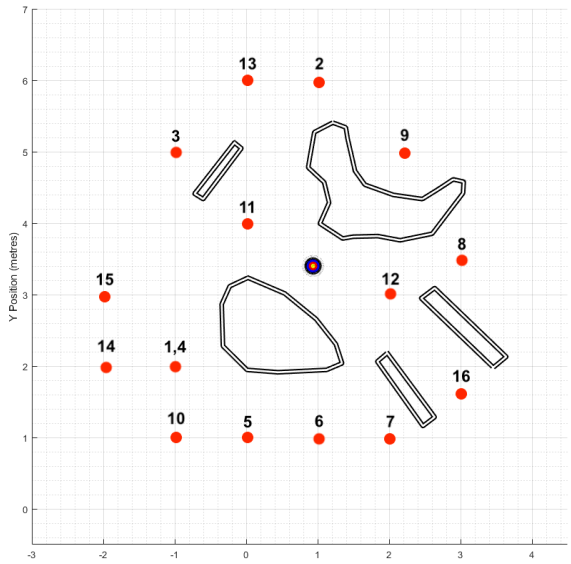


Figure 3.5: Evaluation starting positions shown in simulation environment

3.5.1 Dual Objective DDPG Learning Agent

Dual Objective controllers have been used in research for Fuzzy and Reinforcement Learning Autonomous Navigation controllers, resulting in improved navigation performance [20, 44, 58, 64, 76]. While the SRL learning agent developed in this thesis combined the objective of target seeking and obstacle avoidance into a singular controller, a dual objective approach uses an individual controller for each objective.

Through creating two controllers, the reward function for each is more specific on positive behaviours. This results in the two controllers learning optimal behaviours more tailored to their single objective [44]. The actions of each controller need to be inputted into an action scheduler that computes the final action to be taken by the robot from the two signals. The function used by the action scheduler is shown below in Equation 3.5. U_D is the ultrasonic distance to an obstacle, which is divided by its maximum value (27.5) to scale it to between $[0, 1]$. $a_a(t)$, $a_t(t)$ and $a_f(t)$ is the action of the obstacle avoidance controller, target controller and final action respectively.

$$\mathrm{a}_f(t) = (\tfrac{U_D}{27.5})^3 a_a(t) + \big(1 - (\tfrac{U_D}{27.5})^3\big) a_t(t) \tag{3.5}$$

Due to Expert FLC using a single controller for both obstacle avoidance and target seeking, its action outputs will have correlations to both objectives. This means that these actions can not be used as training examples in a SL phase for the Dual Objective DDPG learning agent. Therefore, the Dual Objective DDPG learning

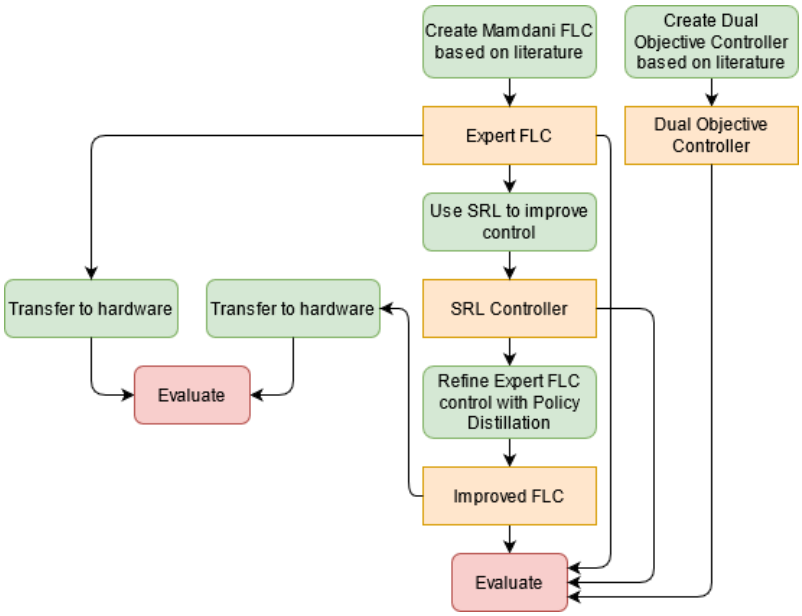


Figure 3.6: Process of controller development and evaluation

agent will have to be trained through a single RL phase. The reward function used to train the target seeking and obstacle avoidance components of the DDPG learning agent is shown in Equations 3.6 and 3.7 respectively. They use the same variables as Equation 3.4.

$$r(t) = \frac{0.05}{D - 0.2}(H(D_t) - H(D_t - 2)) + 500(H(D_t) - H(D_t - 0.35)) - H(D_t - 3.8) + (H(|\theta_t|) - H(|\theta_t| - 30))\frac{30 - \theta_t}{25} + 0.025(v_{Lt-1} + v_{Rt-1}) - 0.025|v_{Lt-1} - v_{Rt-1}|^2 \quad (3.6)$$

$$r(t) = -25H(U_{Dt} - 24.85) + 0.01(v_{Lt-1} + v_{Rt-1}) - 0.025|v_{Lt-1} - v_{Rt-1}|^2 \quad (3.7)$$

The overall Simulink model for the Dual Objective controller can be seen in Figure C.11.

3.6 Autonomous Robot Hardware Implementation

The second major component of the thesis was to implement the improved Fuzzy Autonomous Navigation controller onto a Thymio robot. Through evaluating the navigational performance of the Thymio in complex environments, the viability of SRL for use in classrooms and in commercial applications can be determined. SOTA performance needs to be demonstrated not only in simulation, but also in real-world

experiments as this is where ARs will be used.

3.6.1 Thymio Robot

Thymio is a small, open source robotics platform designed for educational and prototyping purposes. A Thymio II will be used for this thesis due to its ability to wirelessly connect to a computer. The Thymio II has 10 IR sensors, nine ITR9909 sensors for obstacle detection within 10cm and one IRM-3638T sensor for wireless communication with a computer [77]. The nine obstacle sensing sensors are split into five forward facing, two rear facing and two facing the ground. Wireless communication between the USB and Thymio is enabled through a 2.4GHz wireless signal sent using the 802.15.4 protocol [78]. Navigation is enabled through two DC motor driven wheels with a maximum speed of $14cms^{-1}$. The Thymio robotic platform is shown in Figure 3.7.

Figure 3.7: Thymio robotics platform

The Thymio robot is programmed through Aseba, an event driven coding language. Aseba uses asynchronous events to trigger different sub-routines on the hardware platform allowing the robot to react to received wireless data while also simultaneously avoiding obstacles [79]. Many different types of robotics platforms have been used to implement AR navigation [13–15, 22, 24]. The Thymio robotics platform has been used in research due to its low weight, small cost and ability to be easily programmed [64]. Due to these qualities and its similar sensor layout to the simulation based robot, it was selected as the robotics platform to be used in this thesis.

3.6.2 Autonomous Navigation Control

To test the performance of the Improved FLC on hardware, the membership functions and IF-THEN inference rules need to be programmed into the Thymio. There is no method to directly port MATLAB FLCs to an Aseba based controller. Aseba

code was developed to adapt the MATLAB FLC onto the Thymio platform. Due to the differences between the Thymio and the simulation based robot, as described in Table 3.3, several changes to the controller's behaviour were required.

	Sensor Range	***Sensor Angle Coverage***	***Active Sensor Feedback***	***Maximum Speed***
Simulation	0.5 m	270 degrees	Yes	$2ms^{-1}$
Thymio II	0.1 m	180 degrees	No	$0.14ms^{-1}$

Table 3.3: Simulation and hardware platform differences

Due to the decrease in obstacle detection range on the Thymio, the controller needed to be modified to react to obstacles more quickly. This was required as when the Thymio detects an object it would be nearer to a collision then the software controller. A faster reaction time was achieved through increasing the width of the IR obstacle detection triangle membership functions. The increase of MF width was also completed in an attempt to address the potential ultrasonic sensor dead-zones observed in the research of Chiu et al. [13].

Increasing the width of the of the IR obstacle detection MFs also effectively dealt with the reduction in the obstacle detection angle coverage. With a reduced angle coverage, the Thymio cannot detect obstacles that are located directly on its left and right side, whereas the simulation based robot can. By increasing the MF widths, the Thymio reacts faster once the obstacles enter its detection range allowing it to successfully avoid them. This does not fully address the inability to detect obstacles on the left and right side of the Thymio, which is one of the main limitations of the Thymio robotics platform.

Unlike the simulation based robot, the Thymio does not provide any active sensor feedback to the controller, which is signal specifying the sensor that is located closest to an obstacle. This was addressed through a sub-routine in the controller code to analyse the sensor information and generate an active sensor signal.

The reduction in the maximum robot speed did not affect the controller design due to the robustness and portability of Fuzzy logic. The output membership functions were rescaled to fit a $0.14ms^{-1}$ maximum speed.

The Thymio Aseba code is summarised through a flowchart in Figure C.10. The Aseba code can be found in code repository listed in Figure B.1. One limitation of

using the Thymio for research is its small computational power, a critical element for AR hardware implementations identified in the research of Ren et al. [21]. After all the code had been transferred onto the Thymio, its memory was at 99%. Therefore, if any additions or more complex functions were to be added to the Thymio, there would be no memory to store it.

3.6.3 Target Seeking Data

In the simulation environment, the target and localisation data is known to the robot as it navigates. This is not the case for the Thymio hardware platform, and must be provided provided by an external signal. While two dimensional odometry, the act of calculating the robot's local position using its known wheel velocities and turning rotations [80] can be used for localisation, it is inaccurate on the Thymio. The inaccuracies exacerbated as the path length travelled increases [81].

AR localisation using a camera system was chosen as the method to localise the robot in its environment and calculate the necessary target data for the Thymio. Camera localisation is more accurate then the odometry approach for the Thymio platform and has been used in previous research [82, 83].

A webcam was positioned over the navigational environment of the robot with a portable work light used to illuminate the robot and obstacles as shown in Figure 3.8. Live video from the webcam is inputted into a MATLAB program that calculates theta and the distance to the target. This information is sent to the Thymio as it is autonomously navigating.

Figure 3.8: Image localisation camera setup

Figure 3.9: Hardware navigation environment setup

The target location is identified through a long, thick black line placed in the navigational environment, as shown in the top of Figure 3.9. The Thymio is identified through a long, thick black line with a large black circle at one end, shown in the bottom of Figure 3.9. The algorithm filters all other colours of the original image (Figure 3.10 (a)) but black out (Figure 3.10 (b)). The edge image transformation (Figure 3.10 (c)) is then computed before fitting lines and a circle to the target and robot identifiers in the filtered image through using a Hough transform (Figure 3.10 (d)). The robot line is determine to be the line with the least distanced point from the fitted circle's centre, and the target is then known to be the alternative line.

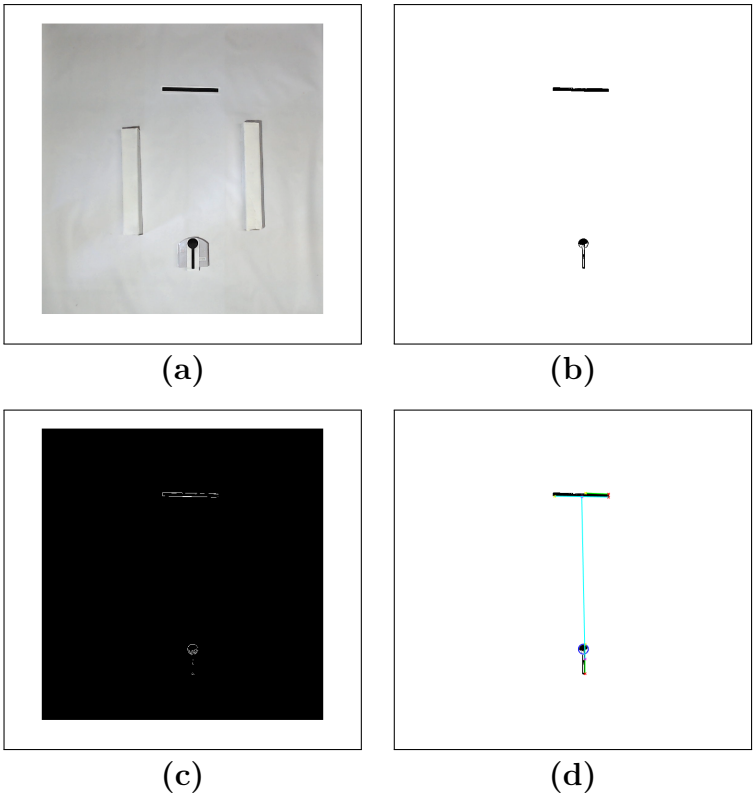


Figure 3.10: Process of calculating robot's local position: **(a)** Web cam image **(b)** Black and white image **(c)** Edge filter applied **(d)** Line vectors fitted to image

Let $T_1(x, y)$ and $T_2(x, y)$ specify either end point of the fitted line for the target. Similarly, let $R_1(x, y)$ and $R_2(x, y)$ denote the end points of the robot's line, and $R_C(x, y)$ be the center of the robot's circle. A new line is created between the midpoint of the two lines with end points $C_1(x, y)$ and $C_2(x, y)$, as shown in Equation 3.8.

$$C_1(x, y) = \frac{T_1(x, y) + T_2(x, y)}{2} \qquad C_2(x, y) = \frac{R_1(x, y) + R_2(x, y)}{2} \tag{3.8}$$

The distance to the target is then calculated through the Euclidean distance between $C_1(x, y)$ and $C_2(x, y)$ as shown in Equation 3.9.

$$D = |C_1(x, y) - C_2(x, y)| \tag{3.9}$$

Two vectors are also created between the two midpoints and another between the circle centre and robot line midpoint, as shown in Equation 3.10. Using this, the theta angle between the robot's pose and the target's midpoint is calculated using a four quadrant inverse tangent function in MATLAB, expressed in Equation 3.11. The four quadrant inverse tangent function takes an (x, y) coordinate and returns the angle $\theta \in (-180, 180]$ between the robot's forward facing pose and the target.

$$V_1 = C_1(x, y) - C_2(x, y) \qquad V_2 = R_C(x, y) - C_2(x, y) \tag{3.10}$$

$$\theta = \boldsymbol{atan2d}\Big(V_1(x)V_2(y) - V_1(y)V_2(x), V_1(x)V_2(x) + V_1(y)V_2(y)\Big) \tag{3.11}$$

Once both theta, θ, and the distance to the target, D, have been calculated they are wirelessly transmitted to the Thymio as it autonomously navigates using the 802.15.4 protocol. The target location is updated every second. The pseudocode flowchart for this process is shown in Figure C.12. The overall MATLAB code can be found in the thesis code repository at Figure B.1.

One limitation of using the Thymio is in the way data is wirelessly transmitted to it. The only way to transmit data is to use the Aseba IDE and open a dialogue window to type data to send. This thesis worked around this through using code to control the mouse. The data calculated from MATLAB was added to the system's cliboard, and then knowing the pixel locations of the necessary buttons, the code opened the dialogue window and entered the data. This means the image localisation code is custom built for the computer it was run on, as the pixel locations of the necessary buttons must be known. Future research should use a robotics platform that can run MATLAB.

3.6.4 Autonomous Navigation Evaluation

The navigation environment for the Thymio is constructed on a sheet of flat plywood covered in white paper. The white paper is used to improve the performance of the image localisation algorithm. All the obstacles are constructed out of a combination of white boxes. The obstacles are white as this reflects the most light back to the

Thymio's IR sensors, improving obstacle detection performance. The edge of the plywood surface is blocked by a white border to keep the robot from driving off the edge. An example of this setup is shown in Figure 3.9 around the sides of the surface.

Due to a physical size limit of the navigational environment setup, the obstacles in the simulation environment could not be recreated to scale. Four different navigational environments were created to model different sections of the simulation environment. Each of the navigational environments, and the simulation environment it is modelling, is shown below in Figure 3.11.

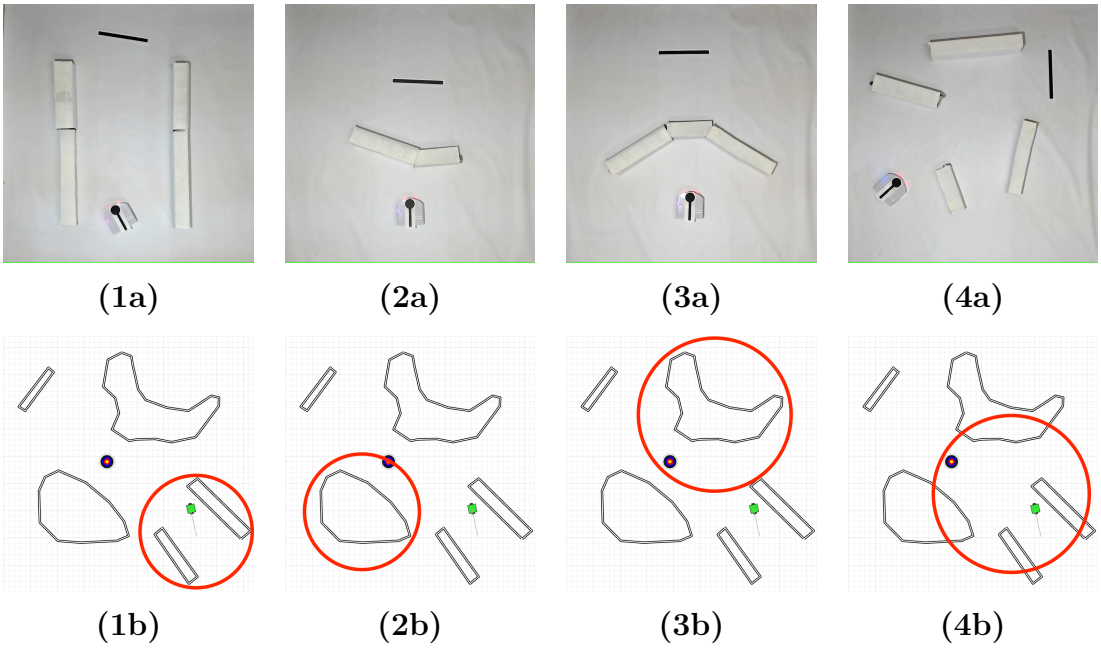


Figure 3.11: Comparison of simulation and hardware environments: **(1a)** Layout 1 **(1b)** Simulation recreation modelled by layout 1 **(2a)** Layout 2 **(2b)** Simulation recreation modelled by layout 2 **(3a)** Layout 3 **(3b)** Simulation recreation modelled by layout 3 **(4a)** Layout 4 **(4b)** Simulation recreation modelled by layout 4

The TTT in each of the four navigational environments was recorded for 10 attempts for both the Improved FLC and Expert FLC. The Expert FLC will be implemented onto the Thymio robotics platform in the identical manner as the Improved FLC. By evaluating the performance of each model, the ability of SRL to improve the performance of Fuzzy AR navigation can be conclusively determined. It will also provide insight into the real-world viability of SRL for ARs.

Chapter 4

Research Results

The following chapter presents the quantitative and qualitative comparative navigation performance of the SRL AR controller to alternative SOTA controllers in both simulation and hardware implementations.

4.1 Simulation Results

The SRL controller and the three comparative controllers discussed in Section 3.5 were evaluated from the starting positions listed in Table C.1. Each controller was evaluated once from each starting position. The TTT results for each controller is shown below in Table 4.1. The percentage difference in navigation time for the SRL controller, Dual Objective controller and Improved FLC when compared to the original Expert FLC is shown in Figure 4.1. The results of the Improved FLC are discussed in Section 4.2.

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Expert FLC	15.95	41.3	25.8	22.5	23.65	29.65	37.4	12.65	37.8	30.1	12.2	12.55	15.9	25.6	19.5	18.4
SRL Controller	15.325	21.125	17.325	17.525	24.625	17.075	28.525	12.125	29.425	19.825	6.025	7.325	16.025	24.375	16.075	17.15
Dual Objective Controller	21.175	23.475	22.525	19.775	24.725	22.575	29.825	18.075	44.3	22.275	4.45	4.975	16.925	21.775	18.725	18.925

Table 4.1: Comparison of TTT for Expert FLC, SRL controller and Dual Objective controller

From Figure 4.1, it can be seen that the SRL controller reaches the target location faster from a majority of the starting positions then the Expert FLC. Over the 16 starting positions the SRL controller reaches the target 21.9% faster. From positions 5 and 13 the SRL controller is slower then the Expert FLC by 4.1% and 0.8% respectively. The percentage by which the SRL controller is slower from these positions is minimal, and from all other positions it is faster. The Expert FLC, from positions 5 and 13, has a more optimal navigational control that wasn't further improved through RL. This discrepancy needs to be investigated further in future research to investigate why overall the SRL controller is faster, but from certain positions its performance has worsened. Regardless, the results have quantitatively highlighted the improvement in control policies taught through SRL and correlate with the findings of Smart and Pack Kaelbling [30], Fathinezhad et al. [44], Ye et al. [45].

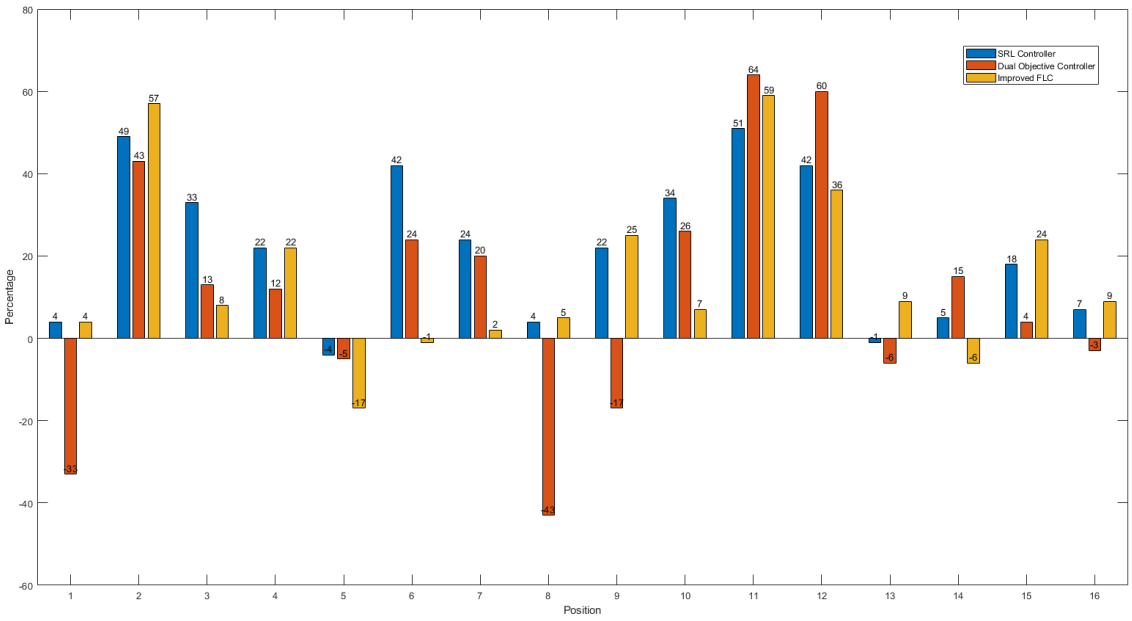


Figure 4.1: Comparison of percentage difference in TTT between Expert FLC and the other three controllers: SRL Controller, Dual Objective Controller and Improved FLC. A positive percentage corresponds to a faster TTT.

Comparing the SRL controller to the Dual Objective controller further highlights the optimal control policies taught through SRL. Over the 16 different positions times in Table 4.1, the SRL controller is faster overall by 12.8%. Yet from positions 11, 12 and 14 the Dual Objective controller is faster by 26.1%, 32.1% and 12.8% respectively. The quantitative comparison between the SRL controller and an alternative SOTA controller further demonstrates the improved navigational performance obtained through optimising an Expert FLC with SRL. While the quantitative comparison between the two controllers continues to agree with the experimental results of Smart and Pack Kaelbling [30], Fathinezhad et al. [44], Ye et al. [45], it contradicts the Dual Objective Controller findings of Omrane et al. [20]. The researched conducted by Omrane et al. [20] did not evaluate their developed model against other SOTA controllers. This thesis has addressed this issue, and shown a SRL controller to possess greater performance for AR navigation.

4.2 Policy Distillation Results

Following the improvement of the Expert FLC through SRL, the process of PD was used to distill the improved, optimal policies back onto the Expert FLC as detailed in Section 3.4. PD was used as an additional step in the improvement pipeline as a FLC ensures a robust control policy that can also be easily understood by humans.

36521 navigation examples were recorded from the SRL controller as it navigated in the environment during training and were used in the PD process.

After PD both the IF-THEN inference rules and input MFs were updated and tuned to improve the control of the Expert FLC. The original inference rules of the Expert FLC were kept the same as in Table 3.1, and an additonal nine rules were developed as shown in Table 4.2. Rules 1, 2 and 9 optimise the control of the robot when it is closing in on the target location while navigating near obstacles. Rules 3 to 6 have been distilled onto the Expert FLC to optimise control when the robot is at the target location. Rules 7 and 8 have optimised the control when adjusting its direction to travel towards the target location.

1	**If**	(distance is close) and (ultrasonic is in_front) and (theta is 0) and (sensor is right)	**then**	(left_vel is zero) and (right_vel is for_f)
2	**If**	(distance is close) and (ultrasonic is in_front) and (theta is right) and (sensor is no_obs)	**then**	(left_vel is for) and (right_vel is zero)
3	**If**	(distance is zero) and (ultrasonic is in_front) and (theta is zero) and (sensor is left)	**then**	(left_vel is rev_f) and (right_vel is rev_f)
4	**If**	(distance is zero) and (ultrasonic is far) and (sensor is right_side)	**then**	(left_vel is zero)
5	**If**	(distance is zero) and (theta is right) and (sensor is no_obs)	**then**	(left_vel is for) and (right_vel is rev_f)
6	**If**	(distance is zero) and (ultrasonic is in_front) and (sensor is left)	**then**	(right_vel is zero)
7	**If**	(ultrasonic is in_front) and (theta is right_small) and (sensor is no_obs)	**then**	(left_vel is zero) and (right_vel is zero)
8	**If**	(ultrasonic is far) and (theta is left) and (sensor is no_obs)	**then**	(left_vel is for) and (right_vel is for)
9	**If**	(distance is close) and (ultrasonic is close) and (theta is zero) and (sensor is no_obs)	**then**	(left_vel is rev_f) and (right_vel is rev_f)

Table 4.2: Additional IF-THEN inference rules for the Improved FLC

The updated MFs of the Expert FLC are shown in Figure 4.2. Figure 4.2 (1a) highlights the 'close' distance MF has been extended to have non-zero membership at larger distances away from the target. The ultrasonic distance MFs were kept the same as those in the Expert FLC as shown in Figure 4.2 (2a), highlighting that, from the SRL controller navigation data provided, the original MFs were optimal. From Figure 4.2 (3a) it can be seen that the Improved FLC has had its 'zero' MF's range extended to have non-zero membership to angles that reflect the robot is required to turn left to face the target. Further, the 'right' angle MF was changed from a trapezoidal MF to a triangular one and the 'left' MF had its non-zero membership extended all the way to the value 180 degrees, while originally the membership value at 180 degrees was 0. Figure 4.2 (4a) highlights the change made to the 'right' obstacle MF, but the ultrasonic location input only takes values of $\{-9, -5, -2.5, 0, 2.5, 5\}$. The original ultrasonic location MFs were created to ensure 100% membership to each of the respective values of the input set when they occurred, hence this update to the 'right' MF is non-optimal and could affect the controller's performance in certain scenarios. This is one of the main issues that can

arise from using PD. If the navigational data used in the distillation process does not contain a wide range of navigational examples, the final FLC may be tuned in a manner that will take non-optimal control decisions in certain situations.

One additional limitation to using PD is in understanding why certain updates were made to the original FLC. While FLCs allow humans to understand the control decisions being made by the controller, it cannot be understood why certain MFs were updated to improve control.

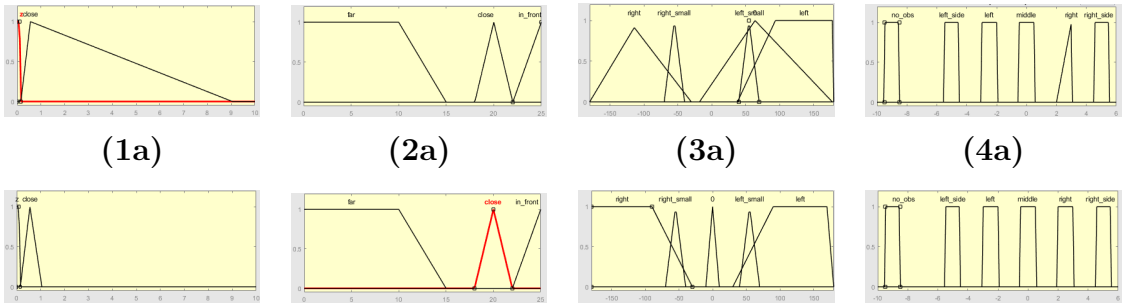


Figure 4.2: Refined MFs: **(1a)** Improved Distance MF **(1b)** Original Distance MF **(2a)** Improved Ultrasonic Distance MF **(2b)** Original Ultrasonic Distance MF **(3a)** Improved Theta MF **(3b)** Original Theta MF **(4a)** Improved Ultrasonic Location MF **(4b)** Original Ultrasonic Location MF

After improving the Expert FLC through distilling the policies learnt by the SRL controller, it was evaluated from the 16 starting positions listed in Table C.1. The TTT and TER is shown in Table 4.3. Overall, the Improved FLC is 15.2% faster then the Expert FLC. This further highlights the strong ability for SRL to improve the navigational performance of an Expert FLC. From positions 5, 6 and 14 the Improve FLC is slower then the Expert FLC by 17.3%, 0.7% and 5.9% respectively. So while overall the Improved FLC demonstrates more optimal control policies, there are certain navigational situations where its control has become less optimal then the Expert FLC.

One improvement that needs to be made to the SRL training and PD process is the refinement of the reward function to make it more robust. The TTT for the Improved FLC from position 11 is 59.4% faster then the Expert FLC, while its reward value is 81.7% lower. Overall, the Improved FLC's reward value from all 16 positions is 15.6% lower then the Expert FLC. Refinement of the reward function

needs to be made to ensure the reward of a controller that travels to the target faster is higher then that of a slower controller. As it stands now, the reward function was designed to encourage the robot to travel towards the target, but a high reward can be gained if the robot stays near the target while not travelling all the way there and finishing the episode.

The comparison of navigational TTT performance for the Improved FLC to the SRL and Dual Objective controllers is shown visually in Figure 4.1. Overall, the SRL and Dual Objective controllers as 21.9% and 10.9% faster then the Expert FLC, while the Improved FLC is 15.2% faster. As discussed in the Literature Review, little research has been conducted on the application of PD for AR. The results presented here provides strong evidence of the improved navigational performance obtained by using SRL and PD for FLCs.

Position	Expert FLC Time (s)	Improved FLC Time (s)	Expert FLC Reward	Improved FLC Reward
1	15.95	15.25	3609	4219
2	41.3	17.9	2424	2401
3	25.8	23.7	2871	2384
4	22.5	17.5	1772	1537
5	23.65	27.75	1752	9877
6	29.65	29.85	3438	3253
7	37.4	36.75	2594	2612
8	12.65	12.05	2017	2494
9	37.8	28.45	2650	2888
10	30.1	27.9	2677	9744
11	12.2	4.95	9839	1722
12	12.55	8	2532	2179
13	15.9	14.45	2389	2273
14	25.6	27.1	1634	9060
15	19.5	14.75	2848	3484
16	18.4	16.8	2417	2161

Table 4.3: Expert and Improved FLC TTT and TER evaluation

4.3 Hardware Results

Both the Expert and Improved FLCs were implemented onto the Thymio hardware platform, as detailed in Section 3.6, and tested over the four obstacle environments. Each controller was tested 10 times in each environment and all the TTT information is provided in Table C.2. A summary of the results is shown below in Table 4.4.

Overall, the Improved FLC hardware implementation is 18.8%, 3.2%, 7.7% and 57.5% faster then the Expert FLC from positions 1, 2, 3 and 4 respectively. It can also be observed that the standard deviation of the TTT for Improved FLC is less

then the Expert FLC from positions 1, 3 and 4. This highlights the improved robustness of the Improved FLC on the hardware platform. It has control policies that are more consistent at reaching the target location in approximately the same amount of time. From position 2, while the standard deviation is higher then the Expert FLC, the average TTT of the Improved FLC is lower. The standard deviation is high due to several samples positively skewing the data. Overall, the Improved FLC is still faster then the Expert FLC from position 2. The data presented strongly correlates with the findings of Fathinezhad et al. [44], that SRL improves the performance of Fuzzy AR navigation on hardware platforms.

There are two major limitations to using the Thymio hardware platform in combination with the image localisation algorithm. Firstly, the absence of ultrasonic sensors on the side of the Thymio robot result in the controller not knowing the presence of obstacles directly on the side of the robot. This resulted in the Thymio robot travelling extremely close to obstacles as it was navigating, and occasionally getting caught on the corners of obstacles. Secondly, the use of the image localisation algorithm was required as the Thymio does not inherently have any data about the target location, whereas in the simulation the robot does. The image localisation algorithm cannot consistently calculate the angle and distance to the target due to the shadows of obstacles and noise interfering with the algorithm. This resulted in target data being sent in inconsistent time intervals ranging between one to four second. It could also result, less commonly, in false target data being sent to the controller that would cause it to drastically change course.

Both of these limitations need to be addressed in future research to investigate more robustly the performance of each FLC on hardware platforms. Further research needs to explore different, more robust hardware platforms and localisation techniques.

	Position 1 Time		**Position 2 Time**		**Position 3 Time**		**Position 4 Time**	
	Expert FLC	**Distilled FLC**	**Expert FLC**	**Distilled FLC**	**Expert FLC**	**Distilled FLC**	**Expert FLC**	**Distilled FLC**
Average	29.989	24.344	38.817	37.564	60.660	55.989	46.659	19.846
Standard Deviation	3.095	1.107	3.211	4.581	6.344	4.189	18.961	1.781
Minimum	27.270	22.530	31.750	32.170	53.610	51.590	35.080	15.490
Maximum	36.700	25.910	42.600	45.940	74.190	65.440	89.010	21.780
CI 95%	0.061	0.022	0.064	0.091	0.126	0.083	0.376	0.035

Table 4.4: Hardware navigation performance summary

Chapter 5

Conclusion

The objective of this thesis was to investigate the use of SRL to improve the navigational performance of an Expert FLC. The investigation was complete through designing, with the use of current SOTA literature, a SRL controller which was compared to other SOTA controllers and evaluated on the Thymio hardware platform.

The simulation results in Chapter 4 have provided conclusive, quantitative evidence that an Expert FLC's navigational control policies can be further improved through the use of SRL. The SRL controller, over 16 different starting positions, was 21.9% faster then the Expert FLC, while a Dual Objective controller was only 10.9% faster. After transferring the improved control policies back to the Expert FLC through the process of PD, the Improved FLC was shown to be 15.2% faster. The quantitative navigational data highlights the strong improvement in control through the inclusion of a SRL and PD process.

The improvement in performance through the use of SRL and PD was further shown in the comparison of the Expert and Improved FLCs on the Thymio hardware platform. The Improved FLC was 21.8% faster then the Expert FLC over the four different obstacle environments. Both the simulation and hardware results both quantitatively confirm the improvement in navigational control through using SRL.

An additional objective of the thesis was to investigate the viability of including RL and the Thymio hardware platform into the experimental projects of ECTE441. Extensive time spent designing and implementing RL for improving control and FLC onto the Thymio has led to the conclusion that neither, in their current form developed in this thesis, is suitable for inclusion into ECTE441. The development and training of an Expert FLC through RL requires a broad understanding of the theory, coding implementation and time that would be too far out of the scope and time frame of ECTE441 and its projects. This is further highlighted in the fact that the Expert FLC will not be provided to students, they will have to develop it themselves. Additionally, the FLCs designed in MATLAB for the simulation experiment could not be automatically ported over to the Thymio due to the Thymio having a

simpler coding language and different ultrasonic sensor arrangement. Further investigation is required into determining a better method of including the findings and developments in this thesis into ECTE441.

While the experimental results highlight strong navigational performance by using SRL, there were several limitations of the experimental design that need to be addressed in future research. One limitation of the experimental design was the lack of robustness and generality present in the trained DDPG learning agent. While the DDPG learning agent possesses, overall, improved navigational control from each starting position, this improvement was localised to the starting position. Preliminary experimentation for this thesis found training more general policies through having the robot start at different locations each episode was slow and non-converging. Further research is required to investigate refinement of the SRL approach to allow for a more robust and generalised AR controller.

Another limitation of the experimental design was in the image localisation algorithm used to provide the Thymio with target location data while it was navigating. Due to shadows present in the environment, the use of a relatively low cost camera and the processing speed of the computer used to analyse the incoming video feed, the target data provided to the Thymio was either delayed and occasionally wrong. Additionally, the lack of ultrasonic sensors on the side of the Thymio meant it could not detect obstacles on its left or right side whereas the simulation based robot could. Future research needs to investigate the implementation of an SRL Improved FLC on a hardware platform that has both more accurate and faster methods of localising the robot and detecting obstacles.

Future research needs to further investigate the use of PD in improving the control of FLCs and other controllers. PD is an area of research in AR navigation that has not been investigated by many researchers. One interesting use of PD would be in distilling the control policies of a controller that has global information of its environment, and thus be able to plan the most optimal path, onto a controller with only local information. Quantitatively determining if an improvement in navigational control was obtained through this method would ascertain the viability of this technique for use in future research and commercial applications.

References

[1] L. Grimstad, R. Zakaria, T. Dung Le, and P. J. From. A Novel Autonomous Robot for Greenhouse Applications. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, October 2018. doi: 10.1109/IROS.2018.8594233. ISSN: 2153-0866.

[2] İlker Ünal, Önder Kabaş, and Salih Sözer. Real-Time Electrical Resistivity Measurement and Mapping Platform of the Soils with an Autonomous Robot for Precision Farming Applications. *Sensors*, 20(1):251, January 2020. doi: 10.3390/s20010251. URL `https://www.mdpi.com/1424-8220/20/1/251`. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[3] T. Kaur and D. Kumar. Wireless multifunctional robot for military applications. In *2015 2nd International Conference on Recent Advances in Engineering Computational Sciences (RAECS)*, pages 1–5, December 2015. doi: 10.1109/RAECS.2015.7453343.

[4] Autonomous robots and the future of supply chain, . URL `https://www2.deloitte.com/us/en/pages/manufacturing/articles/autonomous-robots-supply-chain-innovation.html`.

[5] Carlos Sampedro, Alejandro Rodriguez-Ramos, Hriday Bavle, Adrian Carrio, Paloma de la Puente, and Pascual Campoy. A Fully-Autonomous Aerial Robot for Search and Rescue Applications in Indoor Environments using Learning-Based Techniques. *J Intell Robot Syst*, 95(2):601–627, August 2019. ISSN 1573-0409. doi: 10.1007/s10846-018-0898-1. URL `https://doi.org/10.1007/s10846-018-0898-1`.

[6] Spyros G. Tzafestas. Mobile Robot Control and Navigation: A Global Overview. *J Intell Robot Syst*, 91(1):35–58, July 2018. ISSN 1573-0409. doi: 10.1007/s10846-018-0805-9. URL `https://doi.org/10.1007/s10846-018-0805-9`.

[7] Purushothaman Raja and Sivagurunathan Pugazhenthi. Optimal path planning of mobile robots: A review. *International Journal of the Physical Sciences*, 7(9), February 2012. ISSN 19921950. doi: 10.5897/IJPS11.1745. URL `http://www.academicjournals.org/ijps/abstracts/abstracts/abstract2012/23Feb/Raja%20and%20Pugazhenthi.htm`.

[8] Sajjad Yaghoubi, Negar Ali Akbarzadeh, Shadi Sadeghi Bazargani, Sama Sadeghi Bazargani, Marjan Bamizan, and Maryam Irani Asl. Autonomous Robots for Agricultural Tasks and Farm Assignment and Future Trends in Agro Robots. 13(03):6, 2013.

[9] Anish Pandey. Mobile Robot Navigation and Obstacle Avoidance Techniques: A Review. *IRATJ*, 2(3), May 2017. ISSN 25748092. doi: 10.15406/iratj.2017.02.00023. URL `https://medcraveonline.com/IRATJ/mobile-robot-navigation-and-obstacle-avoidance-techniques-a-review.html`. Number: 3.

[10] Mary B. Alatise and Gerhard P. Hancke. A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods. *IEEE Access*, 8:39830–39846, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.2975643. Conference Name: IEEE Access.

[11] Jayanta Kumar Pothal and Dayal R. Parhi. Navigation of multiple mobile robots in a highly clutter terrains using adaptive neuro-fuzzy inference system. *Robotics and Autonomous Systems*, 72:48–58, October 2015. ISSN 0921-8890. doi: 10.1016/j.robot.2015.04.007. URL `http://www.sciencedirect.com/science/article/pii/S0921889015000895`.

[12] Wenhao Cui. Multi-sensor Information Fusion Obstacle Avoidance based on Fuzzy Control. In *2019 IEEE 2nd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, pages 198–202, November 2019. doi: 10.1109/AUTEEE48671.2019.9033349.

[13] Chian-Song Chiu, Teng-Shung Chiang, and Yu-Ting Ye. Fuzzy obstacle avoidance control of a two-wheeled mobile robot. In *2015 International Automatic Control Conference (CACS)*, pages 1–6, November 2015. doi: 10.1109/CACS.2015.7378356.

[14] Jakup Berisha, Xhevahir Bajrami, Ahmet Shala, and Rame Likaj. Application of Fuzzy Logic Controller for obstacle detection and avoidance on real autonomous mobile robot. In *2016 5th Mediterranean Conference on Embedded Computing (MECO)*, pages 200–205, June 2016. doi: 10.1109/MECO.2016.7525740.

[15] M. V. Bobyr, S. A. Kulabukhov, and N. A. Milostnaya. Fuzzy control system of robot angular attitude. In *2016 2nd International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, pages 1–6, May 2016. doi: 10.1109/ICIEAM.2016.7910970.

[16] M.V. Bobyr, A.A. Dorodnykh, and A.S. Yakushev. Analysis of Fuzzy Models of Implication in the Task of Controlling a Mobile Robot. In *2018 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, pages 1–6, May 2018. doi: 10.1109/ICIEAM.2018.8728798.

[17] Xiaogang Ruan and Wangbo Li. Ultrasonic sensor based two-wheeled self-balancing robot obstacle avoidance control system. In *2014 IEEE International Conference on Mechatronics and Automation*, pages 896–900, August 2014. doi: 10.1109/ICMA.2014.6885816. ISSN: 2152-744X.

[18] Sun Yang-zhi, Xiao Shi-de, Hao Meng-jie, and Pan Shao-fei. Study on obstacle avoidance for intelligent robot based on hierarchical fuzzy control. In *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 525–528, June 2015. doi: 10.1109/CYBER.2015.7287994.

[19] Alessandro Saffiotti, Enrique H. Ruspini, and Kurt Konolige. Using Fuzzy Logic for Mobile Robot Control. In Hans-Jürgen Zimmermann, editor, *Practical Applications of Fuzzy Technologies*, The Handbooks of Fuzzy Sets Series, pages 185–205. Springer US, Boston, MA, 1999. ISBN 978-1-4615-4601-6. doi: 10.1007/978-1-4615-4601-6_5. URL `https://doi.org/10.1007/978-1-4615-4601-6_5`.

[20] Hajer Omrane, Mohamed Slim Masmoudi, and Mohamed Masmoudi. Fuzzy Logic Based Control for Autonomous Mobile Robot Navigation. *Computational Intelligence and Neuroscience*, September 2016. doi: https://doi.org/10.1155/2016/9548482. URL `https://www.hindawi.com/journals/cin/2016/9548482/`. ISSN: 1687-5265 Pages: e9548482 Volume: 2016 Publisher: Hindawi.

[21] Limin Ren, Weidong Wang, and Zhijiang Du. A new fuzzy intelligent obstacle avoidance control strategy for wheeled mobile robot. In *2012 IEEE International Conference on Mechatronics and Automation*, pages 1732–1737, August 2012. doi: 10.1109/ICMA.2012.6284398. ISSN: 2152-744X.

[22] Zheng-Kai Chiu and Pei-Jun Lee. A fuzzy control for obstacle avoidance implemented in the wheel robot with FPGA. In *2017 International Conference on System Science and Engineering (ICSSE)*, pages 95–98, July 2017. doi: 10.1109/ICSSE.2017.8030844. ISSN: 2325-0925.

[23] Mohammed Faisal, Ramdane Hedjar, Mansour Al Sulaiman, and Khalid Al-Mutib. Fuzzy Logic Navigation and Obstacle Avoidance by a Mobile Robot in an Unknown Dynamic Environment. *International Journal of Advanced Robotic Systems*, 10(1):37, January 2013. ISSN 1729-8814. doi: 10.5772/54427. URL `https://doi.org/10.5772/54427`. Publisher: SAGE Publications.

[24] S. G. Tzafestas, K. M. Deliparaschos, and G. P. Moustris. Fuzzy logic path tracking control for autonomous non-holonomic mobile robots: Design of System on a Chip. *Robotics and Autonomous Systems*, 58(8):1017–1027, August 2010. ISSN 0921-8890. doi: 10.1016/j.robot.2010.03.014. URL `http://www.sciencedirect.com/science/article/pii/S0921889010000801`.

[25] Fatma Boufera, Debbat Fatima, Nicolas Monmarché, Mohamed Slimane, and Mohamed Khelfi. Fuzzy Inference System Optimization by Evolutionary Approach for Mobile Robot Navigation. *International Journal of Intelligent Systems and Applications*, 10:85–93, February 2018. doi: 10.5815/ijisa.2018.02.08.

[26] Ezequiel Di Mario, Zeynab Talebpour, and Alcherio Martinoli. A comparison of PSO and Reinforcement Learning for multi-robot obstacle avoidance. In *2013 IEEE Congress on Evolutionary Computation*, pages 149–156, June 2013. doi: 10.1109/CEC.2013.6557565. ISSN: 1941-0026.

[27] Shi Bai, Fanfei Chen, and Brendan Englot. Toward autonomous mapping and exploration for mobile robots through deep supervised learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2379–2384, September 2017. doi: 10.1109/IROS.2017.8206050. ISSN: 2153-0866.

[28] Gregory Kahn, Pieter Abbeel, and Sergey Levine. BADGR: An Autonomous Self-Supervised Learning-Based Navigation System. *IEEE Robotics and Automation Letters*, 6(2):1312–1319, April 2021. ISSN 2377-3766. doi: 10.1109/LRA.2021.3057023. Conference Name: IEEE Robotics and Automation Letters.

[29] K. Macek, I. Petrovic, and N. Peric. A reinforcement learning approach to obstacle avoidance of mobile robots. In *7th International Workshop on Advanced Motion Control. Proceedings (Cat. No.02TH8623)*, pages 462–466, July 2002. doi: 10.1109/AMC.2002.1026964.

[30] W.D. Smart and L. Pack Kaelbling. Effective reinforcement learning for mobile robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 4, pages 3404–3410 vol.4, May 2002. doi: 10.1109/ROBOT.2002.

1014237.

[31] Bing-Qiang Huang, Guang-Yi Cao, and Min Guo. Reinforcement Learning Neural Network to the Problem of Autonomous Mobile Robot Obstacle Avoidance. In *2005 International Conference on Machine Learning and Cybernetics*, volume 1, pages 85–89, August 2005. doi: 10.1109/ICMLC.2005.1526924. ISSN: 2160-1348.

[32] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, May 2017. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aam6960. URL `https://science.sciencemag.org/content/356/6337/508`. Publisher: American Association for the Advancement of Science Section: Research Article.

[33] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic Policy Gradient Algorithms. page 9, 2014.

[34] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. doi: 10.1038/nature14236. URL `https://www.nature.com/articles/nature14236`. Number: 7540 Publisher: Nature Publishing Group.

[35] Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, May 2019. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aau6249. URL `https://science.sciencemag.org/content/364/6443/859`. Publisher: American Association for the Advancement of Science Section: Research Article.

[36] Zhiyong Tan, Chai Quek, and Philip Y. K. Cheng. Stock trading with cycles: A financial application of ANFIS and reinforcement learning. *Expert Systems with Applications*, 38(5): 4741–4755, May 2011. ISSN 0957-4174. doi: 10.1016/j.eswa.2010.09.001. URL `http://www.sciencedirect.com/science/article/pii/S095741741000905X`.

[37] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *arXiv:1712.01815 [cs]*, December 2017. URL `http://arxiv.org/abs/1712.01815`. arXiv: 1712.01815.

[38] Mihai Duguleana and Gheorghe Mogan. Neural networks based reinforcement learning for mobile robots obstacle avoidance. *Expert Systems with Applications*, 62:104–115, November

2016. ISSN 0957-4174. doi: 10.1016/j.eswa.2016.06.021. URL http://www.sciencedirect.com/science/article/pii/S0957417416303001.

[39] Xing Wu, Haolei Chen, Changgu Chen, Mingyu Zhong, Shaorong Xie, Yike Guo, and Hamido Fujita. The autonomous navigation and obstacle avoidance for USVs with ANOA deep reinforcement learning method. *Knowledge-Based Systems*, 196:105201, May 2020. ISSN 0950-7051. doi: 10.1016/j.knosys.2019.105201. URL http://www.sciencedirect.com/science/article/pii/S0950705119305350.

[40] Y. Kato, K. Kamiyama, and K. Morioka. Autonomous robot navigation system with learning based on deep Q-network and topological maps. In *2017 IEEE/SICE International Symposium on System Integration (SII)*, pages 1040–1046, December 2017. doi: 10.1109/SII.2017.8279360. ISSN: 2474-2325.

[41] Xiaogang Ruan, Dingqi Ren, Xiaoqing Zhu, and Jing Huang. Mobile Robot Navigation based on Deep Reinforcement Learning. In *2019 Chinese Control And Decision Conference (CCDC)*, pages 6174–6178, June 2019. doi: 10.1109/CCDC.2019.8832393. ISSN: 1948-9447.

[42] Delong Zhu, Tingguang Li, Danny Ho, Chaoqun Wang, and Max Q.-H. Meng. Deep Reinforcement Learning Supervised Autonomous Exploration in Office Environments. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7548–7555, May 2018. doi: 10.1109/ICRA.2018.8463213. ISSN: 2577-087X.

[43] Junyan Hu, Hanlin Niu, Joaquin Carrasco, Barry Lennox, and Farshad Arvin. Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning. *IEEE Transactions on Vehicular Technology*, 69(12):14413–14423, December 2020. ISSN 1939-9359. doi: 10.1109/TVT.2020.3034800. Conference Name: IEEE Transactions on Vehicular Technology.

[44] Fatemeh Fathinezhad, Vali Derhami, and Mehdi Rezaeian. Supervised fuzzy reinforcement learning for robot navigation. *Applied Soft Computing*, 40:33–41, March 2016. ISSN 1568-4946. doi: 10.1016/j.asoc.2015.11.030. URL http://www.sciencedirect.com/science/article/pii/S1568494615007486.

[45] Cang Ye, N.H.C. Yung, and Danwei Wang. A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(1):17–27, February 2003. ISSN 1941-0492. doi: 10.1109/TSMCB.2003.808179. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics).

[46] Chin-Teng Lin and Chong-Ping Jou. GA-based fuzzy reinforcement learning for control of a magnetic bearing system. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 30(2):276–289, April 2000. ISSN 1941-0492. doi: 10.1109/3477.836376. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics).

[47] Andrei A. Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy Distillation. *arXiv:1511.06295 [cs]*, January 2016. URL http://arxiv.org/abs/1511.06295.

arXiv: 1511.06295.

[48] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 535–541, New York, NY, USA, August 2006. Association for Computing Machinery. ISBN 978-1-59593-339-3. doi: 10.1145/1150402.1150464. URL https://doi.org/10.1145/1150402.1150464.

[49] Wojciech M. Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. Distilling Policy Distillation. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1331–1340. PMLR, April 2019. URL http://proceedings.mlr.press/v89/czarnecki19a.html. ISSN: 2640-3498.

[50] René Traoré, Hugo Caselles-Dupré, Timothée Lesort, Te Sun, Guanghang Cai, Natalia Díaz-Rodríguez, and David Filliat. DisCoRL: Continual Reinforcement Learning via Policy Distillation. *arXiv:1907.05855 [cs, stat]*, July 2019. URL http://arxiv.org/abs/1907.05855. arXiv: 1907.05855.

[51] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. September 2015. URL https://arxiv.org/abs/1509.02971v6.

[52] M Suzanne Donovan and John D Bransford. Committee on How People Learn, A Targeted Report for Teachers. page 265.

[53] Douglas B. Luckie, Jacob R. Aubry, Benjamin J. Marengo, Aaron M. Rivkin, Lindsey A. Foos, and Joseph J. Maleszewski. Less teaching, more learning: 10-yr study supports increasing student learning through less coverage and more inquiry. *Advances in Physiology Education*, 36(4):325–335, December 2012. ISSN 1043-4046. doi: 10.1152/advan.00017.2012. URL https://journals.physiology.org/doi/full/10.1152/advan.00017.2012. Publisher: American Physiological Society.

[54] Patricia Burrowes and Gladys Nazario. Promoting Student Learning Through the Integration of Lab and Lecture. *Journal of College Science Teaching*, 37:6, 2008.

[55] Michelene T.H. Chi, Matthew Lewis, Peter Reimann, and Peter Glaser. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, 13 (2):145–182, April 1989. ISSN 0364-0213. doi: 10.1016/0364-0213(89)90002-5. URL http://www.sciencedirect.com/science/article/pii/0364021389900025. Publisher: No longer published by Elsevier.

[56] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. A Brief Survey of Deep Reinforcement Learning. *IEEE Signal Process. Mag.*, 34(6):26–38, November 2017. ISSN 1053-5888. doi: 10.1109/MSP.2017.2743240. URL http://arxiv.org/abs/1708.05866. arXiv: 1708.05866.

[57] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine. Self-Supervised Deep Reinforcement Learning with Generalized Computation Graphs for Robot Navigation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5129–5136, May 2018.

doi: 10.1109/ICRA.2018.8460655. ISSN: 2577-087X.

[58] Y. Wang, H. He, and C. Sun. Learning to Navigate Through Complex Dynamic Environment With Modular Deep Reinforcement Learning. *IEEE Transactions on Games*, 10(4):400–412, December 2018. ISSN 2475-1510. doi: 10.1109/TG.2018.2849942. Number: 4 Conference Name: IEEE Transactions on Games.

[59] David Luviano and Wen Yu. Continuous-time path planning for multi-agents with fuzzy reinforcement learning. *Journal of Intelligent & Fuzzy Systems*, 33(1):491–501, September 2017. ISSN 10641246. doi: 10.3233/JIFS-161822. URL `https://ezproxy.uow.edu.au/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=iih&AN=123765526`. Number: 1 Publisher: IOS Press.

[60] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P Lillicrap, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. page 10.

[61] Evan Krell, Alaa Sheta, Arun Prassanth Ramaswamy Balasubramanian, and Scott A. King. Collision-Free Autonomous Robot Navigation in Unknown Environments Utilizing PSO for Path Planning. *Journal of Artificial Intelligence and Soft Computing Research*, 9(4):267–282, October 2019. doi: 10.2478/jaiscr-2019-0008. URL `https://content.sciendo.com/view/journals/jaiscr/9/4/article-p267.xml`. Number: 4 Publisher: Sciendo Section: Journal of Artificial Intelligence and Soft Computing Research.

[62] Adrian Filipescu, Viorel Minzu, Bogdan Dumitrascu, Adriana Filipescu, and Eugenia Minca. Trajectory-tracking and discrete-time sliding-mode control of wheeled mobile robots. In *2011 IEEE International Conference on Information and Automation*, pages 27–32, June 2011. doi: 10.1109/ICINFA.2011.5948958.

[63] H.A. Malki, D. Misir, D. Feigenspan, and Guanrong Chen. Fuzzy PID control of a flexible-joint robot arm with uncertainties from time-varying loads. *IEEE Transactions on Control Systems Technology*, 5(3):371–378, May 1997. ISSN 1558-0865. doi: 10.1109/87.572133. Conference Name: IEEE Transactions on Control Systems Technology.

[64] Fatma Boufera, Debbat Fatima, Francesco Mondada, and M. Khelfi. Fuzzy Control System for Autonomous Navigation and Parking of Thymio II Mobile Robots. *International Journal of Computer and Electrical Engineering*, 6:321–325, January 2014. doi: 10.7763/IJCEE.2014.V6.846.

[65] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[66] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with Deep Reinforcement Learning. *arXiv:1312.5602 [cs]*, December 2013. URL `http://arxiv.org/abs/1312.5602`. arXiv: 1312.5602.

[67] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. *arXiv:1802.09477 [cs, stat]*, October 2018. URL `http://arxiv.`

org/abs/1802.09477. arXiv: 1802.09477.

[68] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *arXiv:1801.01290 [cs, stat]*, August 2018. URL http://arxiv.org/abs/1801.01290. arXiv: 1801.01290.

[69] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic Algorithms and Applications. *arXiv:1812.05905 [cs, stat]*, January 2019. URL http://arxiv.org/abs/1812.05905. arXiv: 1812.05905.

[70] Sen Wang, Daoyuan Jia, and Xinshuo Weng. Deep Reinforcement Learning for Autonomous Driving. *arXiv:1811.11329 [cs]*, May 2019. URL http://arxiv.org/abs/1811.11329. arXiv: 1811.11329.

[71] Pin Wang, Hanhan Li, and Ching-Yao Chan. Continuous Control for Automated Lane Change Behavior Based on Deep Deterministic Policy Gradient Algorithm. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1454–1460, June 2019. doi: 10.1109/IVS.2019.8813903. ISSN: 2642-7214.

[72] Wenhui Huang, Francesco Braghin, and Stefano Arrigoni. Autonomous Vehicle Driving via Deep Deterministic Policy Gradient. American Society of Mechanical Engineers Digital Collection, November 2019. doi: 10.1115/DETC2019-97884. URL https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings/IDETC-CIE2019/59216/V003T01A017/1069916.

[73] Sebastian Castro. Deep Reinforcement Learning for Walking Robots Video. URL https://www.mathworks.com/videos/deep-reinforcement-learning-for-walking-robots--1551449152203.html.

[74] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016. ISSN 1476-4687. doi: 10.1038/nature16961. URL https://www.nature.com/articles/nature16961. Number: 7587 Publisher: Nature Publishing Group.

[75] Tune Mamdani Fuzzy Inference System - MATLAB & Simulink - MathWorks Australia, . URL https://au.mathworks.com/help/fuzzy/tune-mamdani-fuzzy-inference-system.html.

[76] Wahidin Wahab. Autonomous mobile robot navigation using a dual artificial neural network. In *TENCON 2009 - 2009 IEEE Region 10 Conference*, pages 1–6, January 2009. doi: 10.1109/TENCON.2009.5395892. ISSN: 2159-3450.

[77] Specifications - Thymio & Aseba, . URL http://wiki.thymio.org/en:thymiospecifications.

[78] IEEE 802.15.4-2011 - IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), . URL https://standards.ieee.org/standard/802_15_4-2011.html.

[79] The Aseba language - Thymio & Aseba, . URL http://wiki.thymio.org/en:asebalanguage.

[80] Moti Ben-Ari. Projects for the Thymio Robot in the Aseba Studio Environment. page 79.

[81] Mohammad O. A. Aqel, Mohammad H. Marhaban, M. Iqbal Saripan, and Napsiah Bt. Ismail. Review of visual odometry: types, approaches, challenges, and applications. *SpringerPlus*, 5 (1):1897, October 2016. ISSN 2193-1801. doi: 10.1186/s40064-016-3573-7. URL https://doi.org/10.1186/s40064-016-3573-7.

[82] Jae-Hong Shim and Young-Im Cho. A Mobile Robot Localization using External Surveillance Cameras at Indoor. *Procedia Computer Science*, 56:502–507, 2015. ISSN 18770509. doi: 10.1016/j.procs.2015.07.242. URL https://linkinghub.elsevier.com/retrieve/pii/S1877050915017238.

[83] Konstantinos K. Delibasis, Vasilios P. Plagianakos, and Ilias Maglogiannis. Real Time Indoor Robot Localization Using a Stationary Fisheye Camera. In Harris Papadopoulos, Andreas S. Andreou, Lazaros Iliadis, and Ilias Maglogiannis, editors, *Artificial Intelligence Applications and Innovations*, volume 412, pages 245–254. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-41141-0 978-3-642-41142-7. doi: 10.1007/978-3-642-41142-7_25. URL http://link.springer.com/10.1007/978-3-642-41142-7_25. Series Title: IFIP Advances in Information and Communication Technology.

Appendix A

Original Project Proposal

University of Wollongong 

SCHOOL OF ELECTRICAL, COMPUTER AND TELECOMMUNICATIONS ENGINEERING
ECTE451 PROJECT PROPOSAL FORM

1. Candidate Details	
Name: Harry Cameron	**Student No: 5013501**
Supervisor: Haiping Du	
Title of Project: Hardware Based Reinforcement Learning Fuzzy Logic Controller to Facilitate Student Learning	
Brief Overview: A key aspect for students learning ECTE441 Intelligent Control as part of an Electrical Engineering degree is to experiment and develop their own control algorithms. One major area of intelligent control is reinforcement learning where the system learns through exploration of the problem space and is rewarded according to the actions it takes, although the initial learning period is slow. Another important area is the application of intelligent control theory and simulation onto hardware to solve real-world problems, such as autonomous robot navigation. Currently these two intelligent control areas are not offered as autonomous robot project components to students. Hence this project proposes the design and implementation of a reinforcement learning algorithm to teach a robot fuzzy logic controller to avoid obstacles and navigate through an obstacle course and implement it onto hardware. The project will also explore and address the problems encountered by research of these two areas. Through this, the feasibility and best practices to incorporate these intelligent control components into ECTE441's major project will be researched to ensure the greatest amount of student learning and understanding is achieved.	

2. Project Description: (Expand to one page maximum)

Students best learn and gain knowledge through demonstration, experimentation and inquiry which allows them to apply their theory through a practical problem over an extended period [22-24]. The University of Wollongong's School of Electrical, Computer and Telecommunications Engineering offers a fourth-year subject, ECTE441, teaching students intelligent control systems. These systems include fuzzy logic control (FLC), supervised, un-supervised and reinforcement learning (RL), genetic algorithms (GA) and adaptive neuro-fuzzy inference systems (ANFIS).

Fuzzy inference systems are a major component of the subject and are used widely in industry and research from robot navigation, controlling flexible-joint robot arms and magnetic bearing systems [1-21]. Out of these systems the major project for the subject allows students to experiment and develop a MATLAB-based FLC, ANFIS and neural network (NN) system to control a simulated autonomous robot through an obstacle field to reach a pre-defined target location.

This project does not encapsulate the student developing a reinforcement learning algorithm to teach the robot to avoid obstacles, a major component of subject theory, nor do they get to observe a real-world implementation of the control algorithms, a key step in learning and understanding [22-24].

To address these two problems this project will investigate the feasibility of designing a RL-trained obstacle avoidance FLC as well as hardware implementation of fuzzy based obstacle avoidance systems. Following this, an investigation will be conducted to determine how to best integrate these components into ECTE441 to give students the ability to develop a wider array of intelligent control systems. Through evaluating the feasibility of the above implementations, the areas of research can be implemented into ECTE441 bolstering its connection between theory and application and immerse students in state-of-the-art (SOTA) research and control systems. It will also ensure that incorrect conceptions of intelligent control will be eliminated through a deeper understanding of the underlying control algorithms, an important part of the experimental learning process [22].

The Reinforcement Learning algorithms will allow students to understand better the process of learning through reinforcement as well as how the reward system affects the robot's performance. By using FLC for RL students will be able to understand the difference in linguistic variable use in relation to their hand-made FLC, whereas a NN would present a black-box of control and would offer no learnings to students on how to better construct a fuzzy system or how the robot understands its surroundings through its sensors.

By providing a platform for ECTE441 students to implement their software onto hardware it will enable them to gain further experience into autonomous control and the difficulties associated when dealing with a physical implementation of a control system such as sensor noise, friction, wheel slip and other non-ideal phenomena. Other experimental research only train and experiment with autonomous obstacle avoidance through simple-shape obstacle fields [1-3,8-10,12,14]. An implementation on more complex obstacle fields would provide further insight into developments of obstacle avoidance, which will be further bolstered through development and comparison to alternative FLC obstacle avoidance algorithms.

This project will also explore the difficulties in the development pipeline of training fuzzy systems through reinforcement learning, most predominantly the slow period of initial learning [3,15-16]. The methods by which these problems are solved will highlight to students the power of hybrid intelligent control systems, further strengthening the theory-experimentation learning link for ECTE441 students.

Thus, the research question for this project is:

Can a Reinforcement Learning Fuzzy Logic Controller be designed and implemented on hardware for autonomous robot navigation to effectively facilitate student learning?

The combined objectives of the ECTE451 & ECTE458 project are:

- Design a Supervised Fuzzy RL (SFRL) algorithm to train a FLC that effectively achieves obstacle avoidance and goal seeking behaviours for an autonomous robot
- Compare SFRL FLC model performance to alternative models such as expert-based FLC and ANFIS controllers
- Implement SFRL model on hardware to navigate physical obstacle field
- Develop software/process to seamlessly port MATLAB fuzzy system onto available hardware
- Design platform to enable students to learn about developing intelligent control by hardware implementation

3. Project Plan: (Two pages maximum)

The aim of the project is to train a FLC to accurately control a robot through an obstacle field to reach a specified target and avoid obstacle collisions in a simulation environment. Once this has been completed, the FLC will be implemented onto a hardware robot to achieve the same objectives in a physical environment. Through developing these control solutions, it will inform the feasibility and best method of implementing them into ECTE441 to maximise student learning of intelligent control.

The SFRL algorithms will be developed and implemented on MATLAB through the reinforcement learning and fuzzy libraries. The expert knowledge data used in the initial learning stage of the algorithm will be gathered from previous ECTE441 work and generated by expert interaction with the simulation software. The simulation software will be the same as the MATLAB based simulation present in ECTE441 to ensure feasibility of implementation. Alternative FLC comparative models will be trained in MATLAB using its ANFIS program.

The method to complete this project is as follows:

1. Generate expert navigation data to train ANFIS, NN and initial stage of SFRL – require enough examples to ensure models gain diverse situation responses (using previous research for reference [15-16])
2. Use RL to continue to train SFRL based model through simulation environment (using previous research for reference on implementation of RL and FLC [1-4,6-16,18-21])
3. Use robot analytics (path length, path time, obstacle collisions, number of full robot rotations) to robustly conclude on performance of each FLC [6,20]
4. Implement SFRL model onto hardware robot to evaluate performance and feasibility through coding FLC onto Aseba language [26-27]
5. Create process to port software FLC onto hardware system efficiently
6. Research best method of implementation of robot hardware control and RL into ECTE441 projects

For the hardware implementation of the control system a Thymio robot will be used. Thymio is an open source robotic platform that is used for educational and research purposes [26]. It is coded through the Aseba language and is equipped with a range of sensors to allow it to detect and respond to its surroundings. The Thymio robot has been used for implementing fuzzy obstacle avoidance control algorithms for research projects [27], and the wireless communication offered by specific models will be required to send control input signals to the robot.

Previous research has highlighted the problem of initially slow learning for reinforcement learning obstacle avoidance systems due to the lack of understanding of the environment and how to interact with it [3,15-16]. To address this a SFRL algorithm is used to accelerate the initial learning of the control system to a level where it has an ability to effectively continue to learn through reinforcement learning.

A FLC is used to contain the RL state-action information about the obstacle environment due to being easily interpretable to humans. Alternative models, such as neural networks, act as control black boxes where the inner workings and decisions are unknown to a human observe. FLC provide linguistic input and output variables connected by plain English rules. This system will allow students to be able to better understand the criteria the reinforcement learning algorithm finds that achieves obstacle avoidance and enables them to gain insight into the learning process of the controller system.

The criteria used to evaluate robot navigation control systems is adapted from previous research [6,20], as it provides a robust set of metrics that can be applied to any control system. This criterion is:

- Path length robot has travelled to reach target (indicating efficiency of control)
- Number of 360-degree rotations taken during navigation (indicating control sequences that have needlessly increased robot path)
- Number of object collisions/failed episodes of selected robot navigations
- For RL trained FLC the total reward gained will also be analysed

This criterion will be used to evaluate and validate the performance of the SFRL algorithm with other SOTA methods of teaching a control system obstacle avoidance. To judge the validity of the experimental results the criteria will be used to compare the performance of the simulate and the experimental obstacle avoidance control systems. Identical obstacle fields will be used in both simulation and experiment to provide the same problem to the robot.

If time permits, part of the research and experimentation for ECTE458 will be an in-depth analysis on the best method of delivery of these new topics to ECTE441 students via the project.

For ECTE451, no hardware is required and hence no contingency plan is required for slow deliveries due to COVID-19. Still, hardware will be ordered immediately for expected use in 2020/2021 Summer break and Session 1 2021 ECTE458. For the hardware implementation, Thymio provides an inexpensive robotic platform that be able to be budgeted for and is potentially already in the possession of SECTE from previous Thesis Students eliminating the need for its purchase.

A detailed outline of the milestones and deliverables for the ECTE451 Project are as follows:

- **Monday of Week 4:** Submit Project Proposal and begin work on develop SFRL training algorithm
- **Monday of Week 5:** Begin writing thesis outline and literature review
- **Monday of Week 6:** Complete literature review. Begin training of alternative FLC models
- **Monday of Week 7:** Complete training & implementation of SFRL training algorithm
- **Mid-Session Recess:** Compare performance of SFRL trained FLC and alternative FLC models through simulation. Begin compiling results for thesis.
- **Monday of Week 8**: Begin writing main body of thesis. Begin designing poster
- **Monday of Week 10:** Draft thesis complete to allow for review
- **Monday of Week 10:** Draft poster complete to allow for review. Begin working on presentation
- **Monday of Week 12:** Final Thesis completed for submission
- **Monday of Week 13:** Final Poster completed for submission
- **Friday of Week 13:** Presentation prepared and recorded

4. Resources Required: (Expand to a half a page maximum)

ECTE451 will require only MATLAB based software and simulation, which access is already available to. ECTE458 will require a Thymio robot platform with additional webcam camera feed for input information. Obtaining a Thymio robot is being conducted, and I have a webcam already that can be used for the project. Should the webcam/video setup need changing, suitable arrangements will be made.

Appendix B

Thesis Code

The code for the thesis was too extensive to list in the appendix. A link to the GitHub code repository is provided below.

`https://github.com/harrycam/ECTE458_Thesis`

Figure B.1: GitHub Code Repository For Thesis

Appendix C

Thesis Figures

C.1 Chapter 3

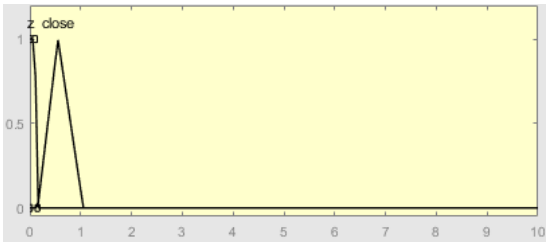


Figure C.1: Distance to target membership function

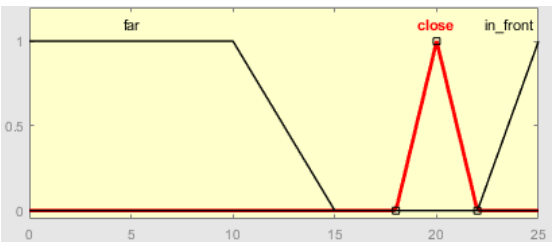


Figure C.2: Ultrasonic distance membership function

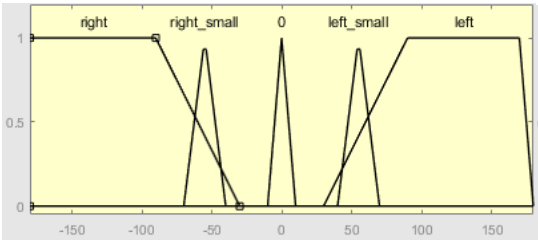


Figure C.3: Angle to target membership function

Figure C.4: Obstacle detection membership function

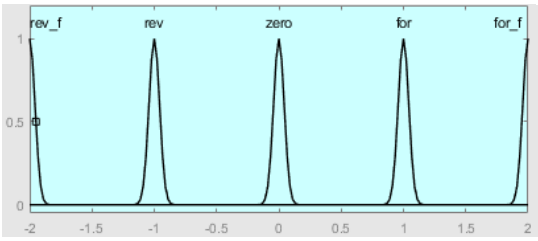


Figure C.5: Left wheel velocity membership function

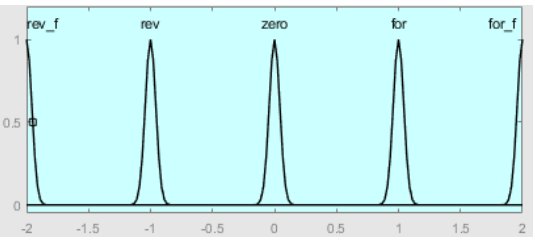


Figure C.6: Right wheel velocity membership function

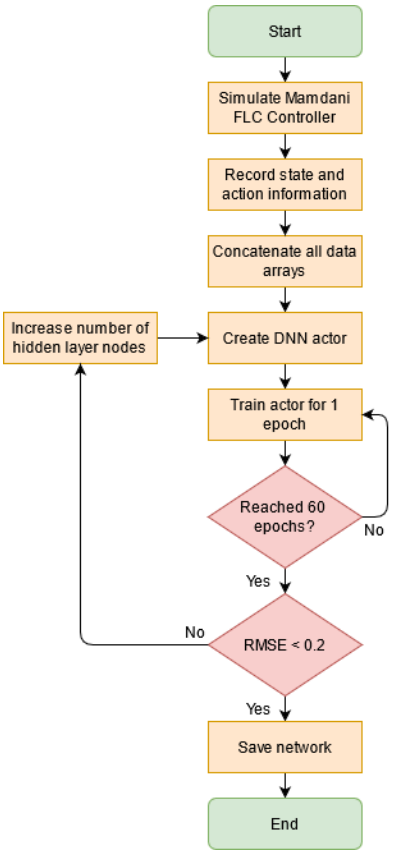


Figure C.7: Actor network supervised training process

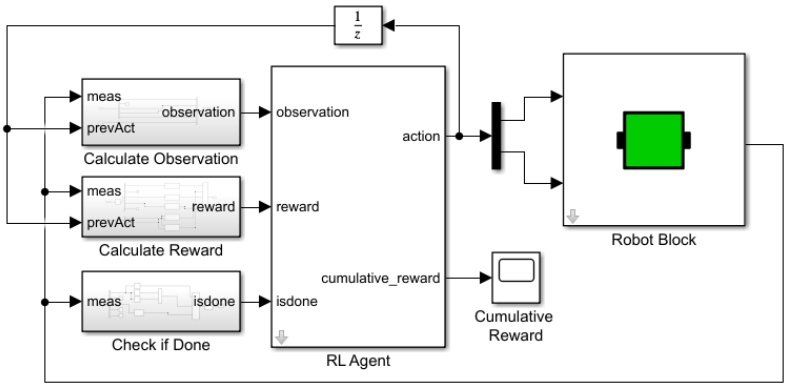


Figure C.8: High-level SRL Simulink model

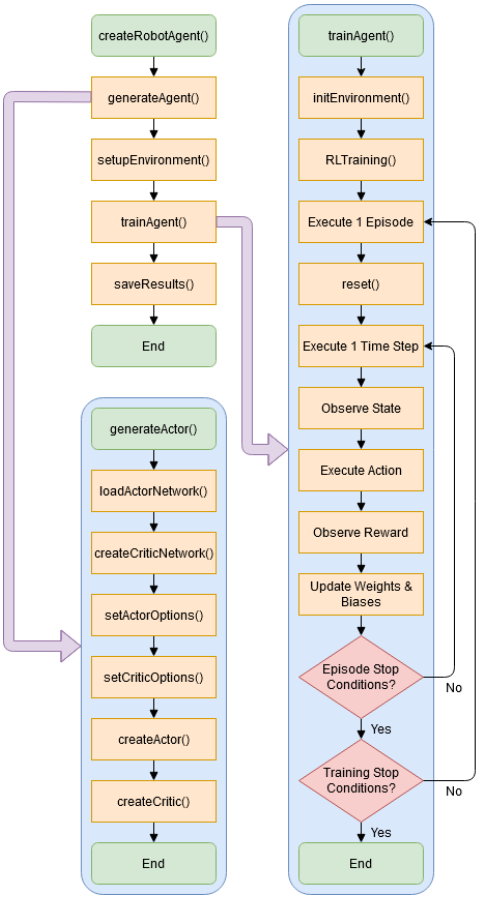


Figure C.9: Process for DDPG agent-critic training

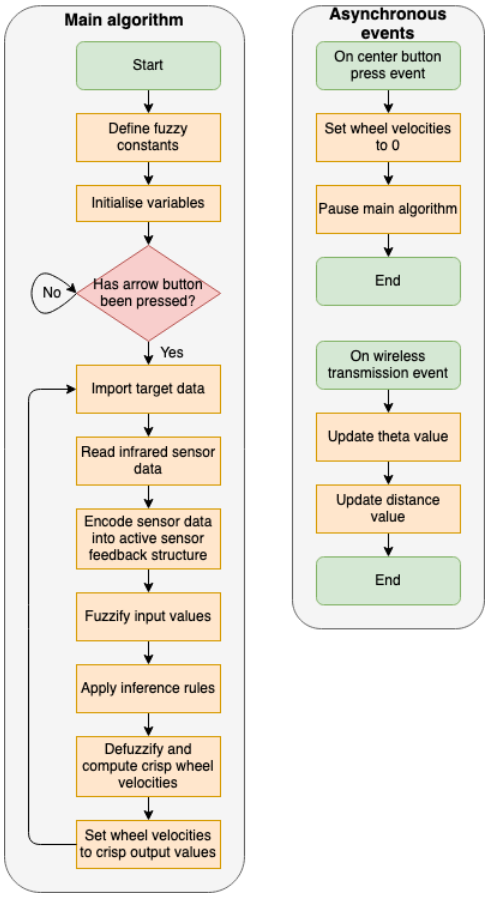


Figure C.10: Thymio Aseba Navigation Code Flowchart

Position	Coordinate	Angle	Reason
1	$(-1, 2)$	$\frac{\pi}{2}$	Convex obstacle and gap present
2	$(1, 6)$	$\frac{\pi}{2}$	Convex obstacle and gap present
3	$(-1, 5)$	$\frac{-\pi}{2}$	Straight obstacle
4	$(-1, 2)$	0	Convex obstacle and must determine best direction of travel
5	$(0, 1)$	$\frac{\pi}{2}$	Convex obstacle
6	$(1, 1)$	0	Small gap to test controller performance with multiple obstacles
7	$(2, 1)$	0	Small gap to test controller performance with multiple obstacles
8	$(3.5, 3.5)$	$\frac{\pi}{2}$	Gap present with multiple obstacles
9	$(2.2, 2)$	π	Concave obstacle
10	$(-1, 1)$	$\frac{\pi}{4}$	Far distance and convex obstacle
11	$(0, 4)$	$\frac{3\pi}{4}$	Small distance to test speed of controller
12	$(2, 3)$	$\frac{\pi}{4}$	Small distance to test speed of controller
13	$(0, 6)$	$\frac{\pi}{2}$	Gap present with multiple obstacles
14	$(-2, 2)$	0	Far distance and convex obstacle
15	$(-2, 3)$	$\frac{\pi}{2}$	Far distance and convex obstacle
16	$(3, 1.6)$	$\frac{3\pi}{4}$	Gap present with multiple straight obstacles

Table C.1: Simulation evaluation starting positions

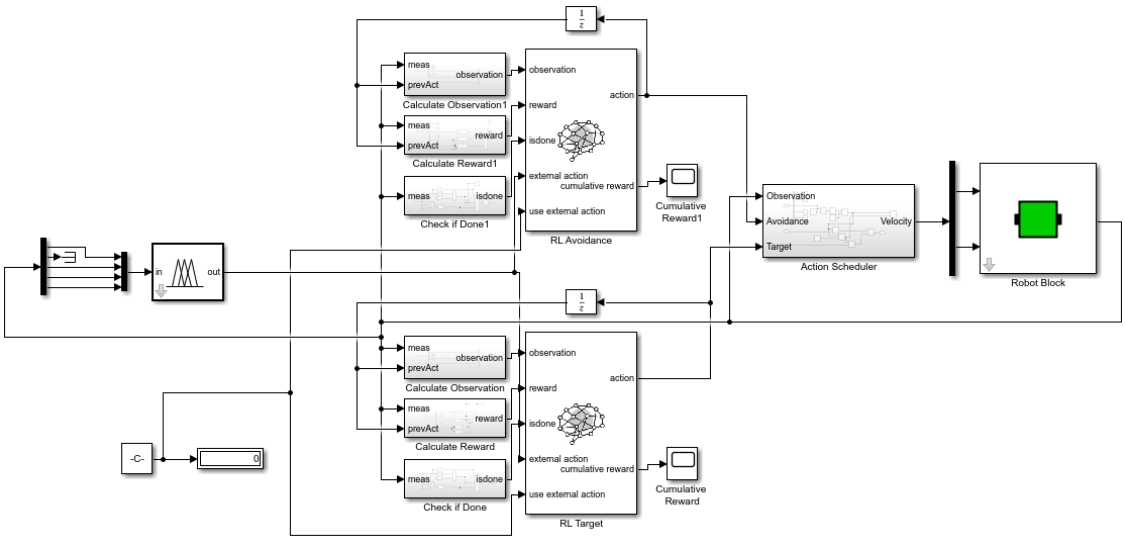


Figure C.11: High-level Dual Objective Simulink model

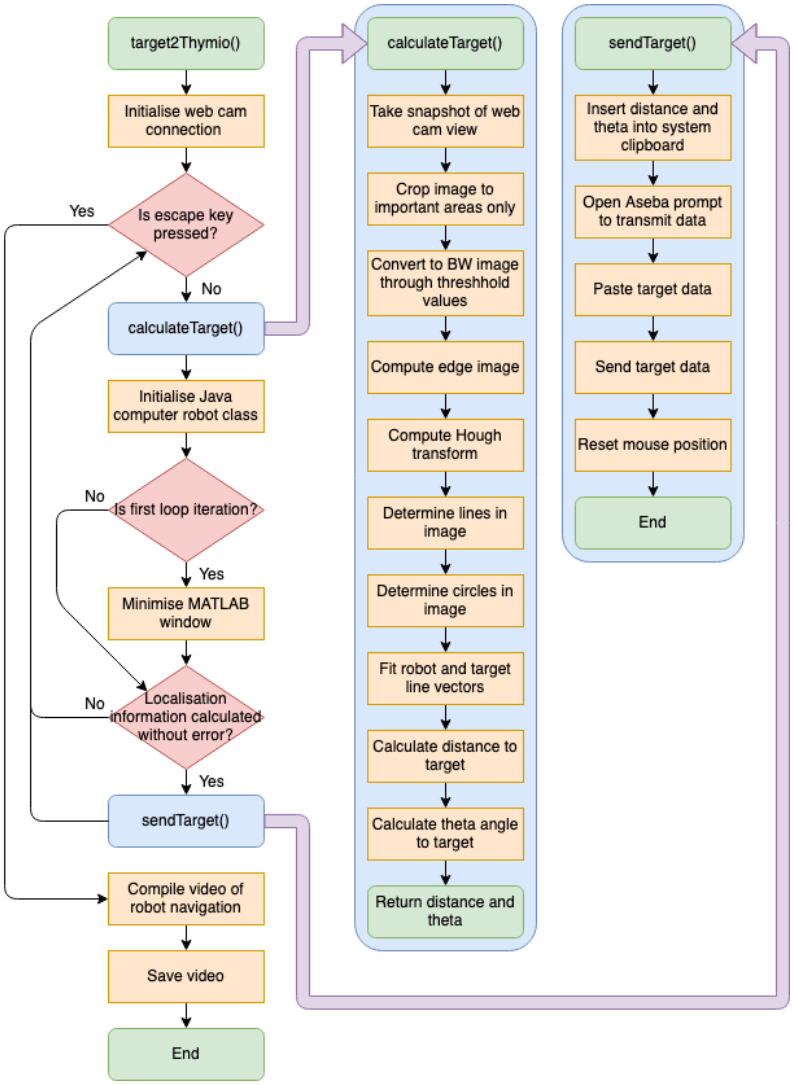


Figure C.12: MATLAB image localisation psuedocode

	Position 1 Time		Position 2 Time		Position 3 Time		Position 4 Time	
Position	**Expert FLC**	**Distilled FLC**	**Expert FLC**	**Distilled FLC**	**Expert FLC**	**Distilled FLC**	**Expert FLC**	**Distilled FLC**
1	29.14	23.86	38.44	45.94	53.61	53.44	35.43	15.49
2	33.67	25.28	42.6	37.69	61.47	55.9	37.18	20.55
3	27.27	24.51	31.75	39.12	58.29	60.76	74.78	19.15
4	27.7	23.97	39.14	35.11	56.82	65.44	35.08	20.01
5	28.38	24.88	41.37	34.85	74.19	53.75	89.01	20.58
6	36.7	22.53	41.06	44.32	59.25	51.59	37.07	20.93
7	27.83	23.57	38.36	32.51	59.57	56.81	39.35	19.98
8	27.71	23.22	35.2	36	60.34	53.35	38	18.76
9	31.32	25.91	39.96	32.17	68.73	55.53	39.59	21.78
10	30.17	25.71	40.29	37.93	54.33	53.32	41.1	21.23

Table C.2: Hardware navigation TTT for Expert Improved FLCs

Appendix D

Thesis Meeting/Weekly Updates Proof

All proof of weekly updates and meeting scheduling has been provided in the additional materials ZIP file. The 'Meeting Proof' folder provides a collection of email correspondence between myself and Professor Haiping Du.