Multi-robot Tethering Using Camera

by

Muhammad Rifdi Bin Rosli

16981

Dissertation submitted in partial fulfilment of

the requirements for the

Bachelor of Engineering (Hons)

(Electrical and Electronic)

JANUARY 2015

Universiti Teknologi PETRONAS

Bandar Seri Iskandar

32610 Tronoh

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Multi-robot Tethering Using Camera

by

Muhammad Rifdi Bin Rosli

16981

A project dissertation submitted to the

Electrical and Electronic Engineering Programme

Universiti Teknologi PETRONAS

in partial fulfilment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(ELECTRICAL AND ELECTRONIC)

Approved by,

_____

(Abu Bakar Sayuti Bin Hj Mohd Saman)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

January 2015

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

_____

MUHAMMAD RIFDI BIN ROSLI

ABSTRACT

An autonomous multi-robot or swarm robot able to perform various cooperative mission such as search and rescue, exploration of unknown or partially known area, transportation, surveillance, defence system, and also firefighting. However, multi-robot application often requires synchronised robotic configuration, reliable communication system and various sensors installed on each robot. This approach has resulted system complexity and very high cost of development. This project propose the integration of low-cost embedded camera in multi-robot application to perform object recognition and tracking, robot platoon coordination, and robot formation control using Proportional, Integral, and Derivative Controller and Fuzzy Logic Controller. The autonomous multi-robot will be developed using Intel Galileo as its microcontroller and low cost embedded computer vision called CMUcam. In the future, robot will be equipped with the ability of visual perception just like humans and animals in performing daily tasks.

ACKNOWLEDGEMENTS

This project was completed with the support of various individuals and communities whom I may not be able to address all of them here. However, their support may well be acknowledged in my mind and soul and I would wish them to have a blissful life here and after.

I would record my thankfulness to my lecturer and also my project supervisor, Abu Bakar Sayuti who has guided me until the completion of this project. He is a knowledgeable educator, an understanding person, and an inspirational person especially in robotics.

I am also in debt to each and every members of Electrical and Electronic Engineering Department of Universiti Teknologi PETRONAS whom has given their best support in terms of educations, morale, services, facilities, and also financials.

My gratitude is extended to my parents whom has never stop in believing and supporting me, whom has always love me without limits.

Not to forget the supports of my dear classmate friends, who has always spend some time to advice, to remind me, and to share their knowledge for the past four years of studies together.

Finally, I would like to give my full acknowledgement to Unversiti Teknologi PETRONAS organisations, to the communities of Tronoh and Perak, and to everyone who may directly or indirectly help to complete this project.

With warm regards,

_____

MUHAMMAD RIFDI BIN ROSLI

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLE

LIST OF EQUATIONS

ABBREVIATIONS AND NOMENCLATURES

Abbreviations	**Descriptions**
eFLL	Embedded Fuzzy Logic Library
FLC	Fuzzy Logic Controller
FOV	Field of view
GPS	Global Positioning System
HSI	Hue Saturation Intensity
LRF	Laser Range Finder
MF	Membership Function
PD	Proportional Differential
PV	Process variable
PWM	Pulse Width Modulation
RAD	Rapid Application Development
RF	Robot Follower
RL	Robot Leader
RRG	Robotics Research Group
SP	Set point
SPI	Serial Peripheral Interface
UESPI	State University of Piauí
WMR	Wheeled Mobile Robots

CHAPTER I
INTRODUCTION

1.1 Background

Human has outstanding ability in visual perception. Generally the human eyes works by the reaction of its retina with light. Human can recognise and differentiate its own family members, friends, peoples, objects, and colours rather instantaneously. Unlike human, machine vision has yet to achieve the human capability of visual perception. Most machines, specifically a robot, solely dependent on the sensors installed to interpret its surrounding area. Such sensors can be mechanical, ultrasonic, infrared, laser, camera, microwave and many more [29, 10, 15]. Typically robot 'visualise' through the information it gathered from the sensors and compute them. Current technology has proved that robot able to recognise objects and people by using complex image processing algorithm, for example Hue Saturation Intensity (HSI).

With the increasing popularity in robotics development, especially in the open source robotic system, it has resulted various type of robot being developed around the world. In swarm robotic, recognition between multiple robots has been achieved with the use of communication such as wireless networks. However swarm robotic mostly encompasses of the same robot configuration to ensure that they able to communicate with each other. Currently the interaction between a robot and another unidentified or unknown robot is rather difficult to achieve as they are pre-built differently in such a way they unable to communicate when different protocols and configuration are implemented. Hence, unique robot system that able to recognise each other and then coordinate their movement like human is needed.

A task would be easier being accomplished when it is being performed by

several people. There is a saying that two heads are better than one. Thus, building several robots to collaborate with each other will increase the efficiency of performing complex series of tasks. In addition, when multiple robots working together they now able to perform tasks that is almost impossible to be performed individually. For example of such tasks are search and rescue mission, exploration of unknown or partially known area, transportation, surveillance and defence system.

The purpose of this research is to develop an algorithm for multi-robot application to detect the presence of one another with the use of on-board camera, tracking and following, obstacle avoidance and lastly basic robot movement synchronisation such as line formation, leader-follower formation and many more. Each robot will be installed with monocular camera. The low-cost embedded camera called CMUcam5 Pixy is used in this research.

This research proposal is structured into 4 main chapters: The first chapter is to explain the research background and purposes. In Chapter II discusses the previous researches and studies that has been recorded. In Chapter III the research methodology and gantt chart are specified. Finally the conclusion and the feasibility of performing this project are elaborated in Chapter IV.

1.2 Problem Statement

Various type of communication and sensors are developed for multi-robot application. Most robotic system that uses visual sensors relies on the multiple fixed camera to provide the information of an area. Such information includes the 2D/3D area localisation and mapping, and also the real time position of the robots. This method limits the robot to a controlled area. Hence, the need of a better coordinating system that can operate outside the laboratory is required.

Global Positioning System (GPS) is another common approach for robot to track the location of its own or another. However GPS is known for its bad performances at indoor [34].

Triangular WiFi positioning may be the solution for indoor positioning system, however it is still in research progress and the accuracy and the stability of wireless

system to determine position of its client is still questionable.

Laser Range Finder (LRF) is a high accuracy range system that able to compute distance accurately. It also used in localisation and mapping. The only concern of using LRF is that the hardware individually is costly thus is not favour solution for multi-robot application.

The use of monocular camera mounted on robot is not new. Generally, robot with camera has the basic function to avoid obstacles, to estimate the distances between two points, recognising colours, and tracking human or objects. This technique should be explored to allow the robot to identify another robot visually and then to perform coordination and cooperative mission [16, 20, 25].

1.3 Objectives and scope of study

The proposed research objectives and scope of study are outlined in this section. The objectives of this research are:

i. To develop an algorithm for multi-robot tethering using camera.

ii. To demonstrate the application of the algorithm on multi-robot tethering.

This research focuses on the development of the multi-robot tethering algorithm. All of the algorithm of the image processing is handled by CMUcam5 Pixy. However, the algorithm of tracking and following robots will be developed in this research. In addition, the programming of multi-robot formations and coordination will be implemented.

CHAPTER II
LITERATURE REVIEW

This research proposed the development of algorithm for multi-robot application using a monocular camera mounted on the robots. The monocular camera will be used for robot leader tracking and following, distance approximation between robot leader and robot follower, and vector approximation of the robot leader. There will be no communication implemented between the robots. The robots will only rely on its on-board vision to perform its missions. Several past researches that studied the concepts of using monocular vision on Wheeled Mobile Robots (WMR) were found and discussed in this section.

2.1 Colour tracking

This paper proposed the use of fifth generation of low cost embedded colour vision system by Carnegie Melon University. CMUcam5 Pixy is the latest version of the embedded camera and it will be used in this research. The work in [28] describes that CMUcam utilises a low cost CMOS colour camera module, a frame buffer chip and all image data is processed by a low cost microcontroller. Based on the CMUcam5 Pixy datasheet, this embedded camera operates based on the hue-based colour filtering algorithm to detect colours. Specifically the HSV which stands for Hue, Saturation, and Value.

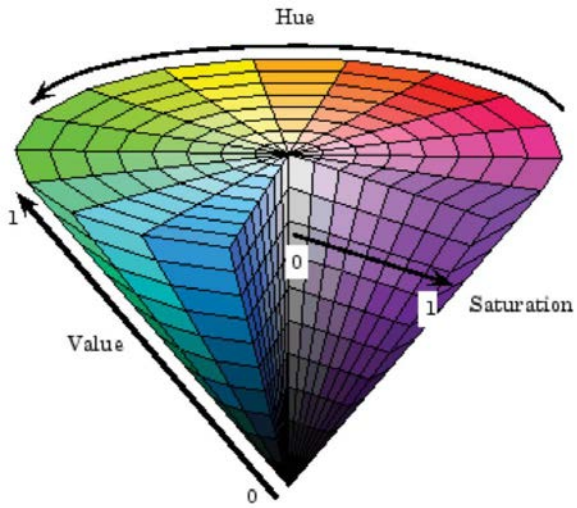


FIGURE 1: HSV cone space model [6]

The research in [25] explain that the hue represents the 'chromaticity, saturation describes the degree of concentration of the colour and value stands for the intensity. CMUcam5 Pixy calculates the hue and saturation of each RGB pixel from the image sensor and uses these as the primary filtering parameters. The hue of an object remains largely unchanged with the changes in lighting and exposure.

CMUcam5 Pixy able to recognise 7 different colour signatures and detect hundreds of objects at a time. It can process a full 640x400 image frame in 20 milliseconds which equal to 50 frames per second.

2.2 Formation control

Common formation control have been implemented in swarm robotics are the leader-follower methods, behaviour-based control, variable structure control techniques, and consensus based methods [26]. Formation control can be a centralised formation control and a decentralised formation control [16]. In a centralised formation control, all of the coordination information of the entire robots platoon are shared between a robot leaders with other robot follower. This method however requires high bandwidth, fast processors, and large storage hence it is not feasible research. In a decentralised formation control the robot autonomously operate using the local

information rather than relying on the information from a single leader. In addition, the position of follower and leader in decentralised formation are subjects to change anytime.

2.3 Coordination

A platoon of multiple mobile robot when deployed requires coordination of each robot motion, this is shown in [20]. The coordination can be through communication or various sensors. In this proposed research, the coordination between robots are only based on the information from local on-board camera. Hence, there will be no communication implemented in the robot platoons. The robot follower and robot leader must be positioned in a straight line formation. To achieve this, the camera will adjust the robot leader image to be centred. Setpoint determine the numerical centre frame value. The X and Y error are obtained and calculated to adjust the image to its centre frame [17].

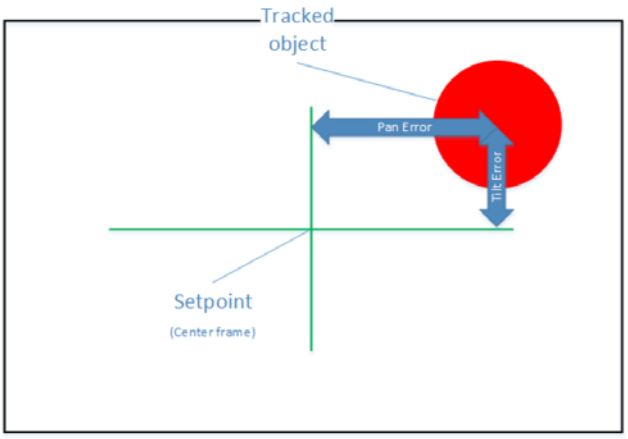


FIGURE 2: X-Y Error in camera view [10]

2.4 Algorithm

The algorithm shown in work [5] is designed for a robot to track an object and then to follow it. The algorithm firstly define the x and y centre and the minimum and maximum field of view (FOV) angle. The robot then will start to search random blob. Blob is the image colour of the object marker which are represented in pixelated form. The camera will start to looks for the desired blob, and when the desired blob has been found the camera will start to continuously track it. The camera will automatically reposition it's the blob to its centred frame. The robot will drive the motor forward and approach the block to its desired size in camera FOV. Once the correct size has been captured in the camera FOV, the robot will gradually decrease its motor speed.

2.5 Feedback Control system

The research in [26] explained that the control goal is to regulate the relative position between the robot follower and the robot leader, moving with bounded positive translational and angular velocity (vL, ωL) ∈ R + ×R. The CMUcam5 Pixy will be programmed with feedback control system to track and follow the robot leader.

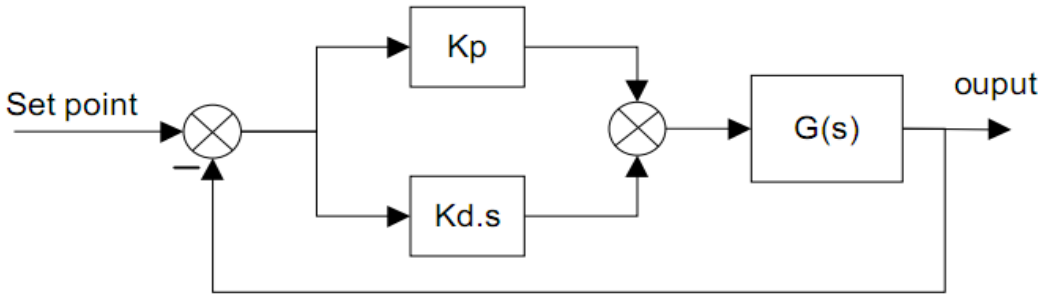


FIGURE 3: PD Block Diagram

The work in [6] described how the closed loop system was designed and implemented. PID controllers probably the most common controller used today. PID stands for Proportional, Integral, and Derivative control and it is actually a 3 types of controller in one. However not all application require PID control and most PID controllers are actually just operated as PI, PD or even just P.

The work in [17] state that the integral control is not suitable to be used in dynamic system as it integrates the error over time. If the measurement is not converging on the set point, the integral output keeps increasing to drive the system toward the set point. Integral control is good for nudging steady, predictable processes closer to perfection. Since CMUcam5 Pixy needs to always respond quickly to random unpredictable movements, integral control is not appropriate.

EQUATION 1: Proportional, Integral, and Derivative equation

$$MV(t) = Kpe(t) + Ki\int_0^t e(\tau)d\tau + Kd\frac{d}{dt}e(t)$$

The constants K_p, K_i, and K_d are used to set the sign and contribution gain of each part of this equation, whereas e (t) is your proportional "error" corresponding to set point (SP) – Process Variable (PV). The variable t corresponds to the current time in our system, and is simply a variable of integration.

2.6 Fuzzy Logic Controller

An intelligent controller namely the Fuzzy Logic Controller (FLC) has been implemented into various robotic application such as mobile robot for people following as described in [35]. Typically the FLC is used to manipulate the movement of the nonholonomic mobile robot as discussed in [30, 24] to achieve desired turning angle and to reduce the navigation error. Besides robot movement, FLC also has been simulated in ensuring the safety of robots while moving in dynamic movement, and obstacle avoidance scheme as presented in [14]. The research in [4, 1] has simulated the mobile robot to track a moving target along an unknown trajectory. Another simulation of fuzzy logic is discussed in [13] to maintain the desired formation control between robot leader and robot follower.

In [32] double fuzzy logics and extended kalman filter were combined and implemented for sensor fusion system of autonomous robot guidance. Combination of fuzzy logics and PID is also implemented to achieve trajectory tracking with minimum steady-state error and improving the dynamic behaviour (overshoot) as discussed in [2].

Autonomous robot that able to escape from a maze while avoiding any collisions using FLC is presented in [9, 33, 19]. Another paper [9] simulated Fuzzy logic to emulate skilled human driver experience of driving behaviour to perform reverse parking without needing accurate model equations, and handle any perturbation in the system.

The FLC generates a control law in a general form:

$$u(k) = F(e(k), e(k-1), \ldots e(k-p), u(k-1), u(k-2), \ldots, u(k-p))$$

Technical constraints limit the dimension of vectors. Also, the typical FLC uses the error change

$$\Delta e(k) = e(k) - e(k-1)$$

and for the control

$$\Delta u(k) = u(k) - u(k-1)$$

Such a control law can be written as

$$\Delta u(k)=F(e(k),\Delta e(k))$$ [Gupta et al., 1979],

$$u(k) = u(k-1) + \Delta u(k)$$ [Dubois & Prade, 1979]

The error e(k) and its change Δe(k) define the inputs included in the antecedents of the rules and the change of the control Δu(k) represents the output included in the consequents and it is represented in FIGURE 4 below.

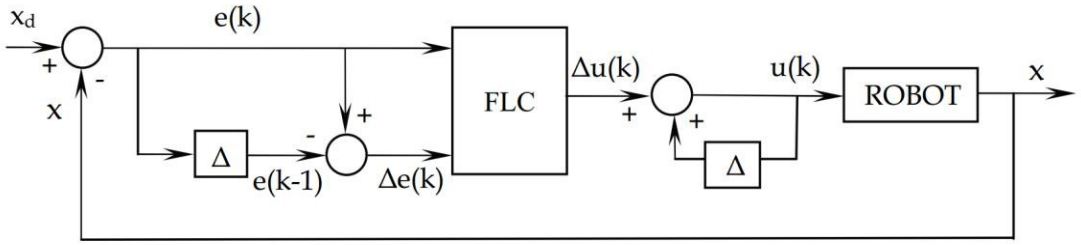


FIGURE 4: Block diagram of FLC

CHAPTER III
METHODOLOGY

3.1 Rapid Application Development (RAD)

The methodology of this research will be based on the Rapid Application Development (RAD) by James Martin. RAD is an ideal development life cycle for this proposed research because it was designed for faster development and give higher quality result compared to the conventional life cycle methodology. FIGURE 5 below is the RAD design for this proposed research. [21].

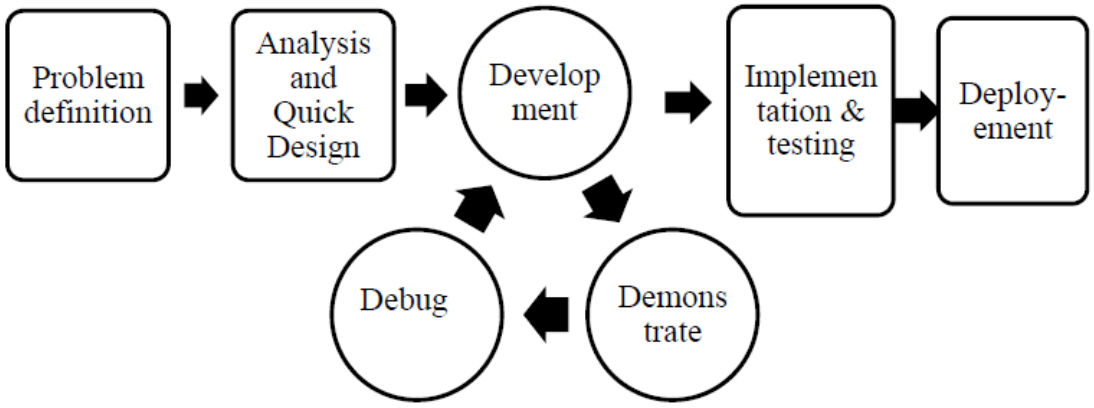


FIGURE 5: RAD block diagram

Problem definition – This process involves with identification of problems with the current robotic system. Research and proposal was outlined during this phase. Past research studies were studied and taken into account.

Analysis and Quick Design – Based on the problem definition, the solutions are figured here. A psuedo-code was created to for the algorithm. In addition, the consideration of the most suitable hardware, and programming language are planned during this process.

Development – The programming of the algorithm is executed in this process.

Demonstrate – The programming will be simulated in the computer.

Debug – Any troubleshooting of the program is run here. The cycle will continue until the program is working properly as expected.

Implementation & Testing – Multi-robot platforms will be adjusted to compatible with the algorithm in this phase. Then the algorithm will be uploaded into the robots application. They will be tested to ensure desired operation.

Deployment – The finalised algorithm is deployed. The robots now should be able to perform all the requirements and the objectives of the research are achieved.

3.2 Gantt Chart

Table below is the Gantt chart for FYP I and FYP II.

TABLE 1: FYP I Gantt Chart

No	Activity	Month 2014	Sep		Oct				Nov				Dec			
		Study Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Project Title Research															
2	Research and prepare extended proposal															
3	**Submission of Extended Proposal**															
4	Algorithm design and preparation of proposal defence															
5	**Oral presentation to supervisor and external examiner**															
6	Development of algorithm															
7	Demonstrate algorithm and prepare interim report															
8	**Submission of Draft Interim Report**															
9	**Finalisation and Submission of Interim Report**															

TABLE 2: FYP II Gantt Chart

No	Activity	Month 2015	Jan				Feb				Mar				April		
		Study Week	-	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Debugging of algorithm																
2	Development life cycle, Implementation, and testing																
3	**Submission of progress report**																
4	Deployment and finilisation of system																
5	**Pre-Sedex presentation**																
6	**Submission of draft final report**																
7	**Submission of Dissertation (soft bound)**																
8	**Submission of Technical Paper**																
9	**Viva**																

3.3 Proposed robot system

The proposed robot follower system will have a microcontroller, an on-board camera and also a robot platform with wheels and motors. The multi-robot system was design to minimise the robot cost. Intel Galileo GEN2 is utilised as the microcontroller to control the robot movement based on the information from the CMUcam5 Pixy, on-board monocular camera. Communication between the microcontroller and the on-board camera is via serial peripheral interface (SPI) to ensure high-bandwidth and high-speed communication. The camera is used as the only robot sensor to ensure the robot follower able to follow the robot leader. To maintain the leader-robot following formation, the Proportional controller is implemented and applied on the robot motors. In addition, the robot leader will be marked with high contrast and unique colour to ensure the camera able to track it.

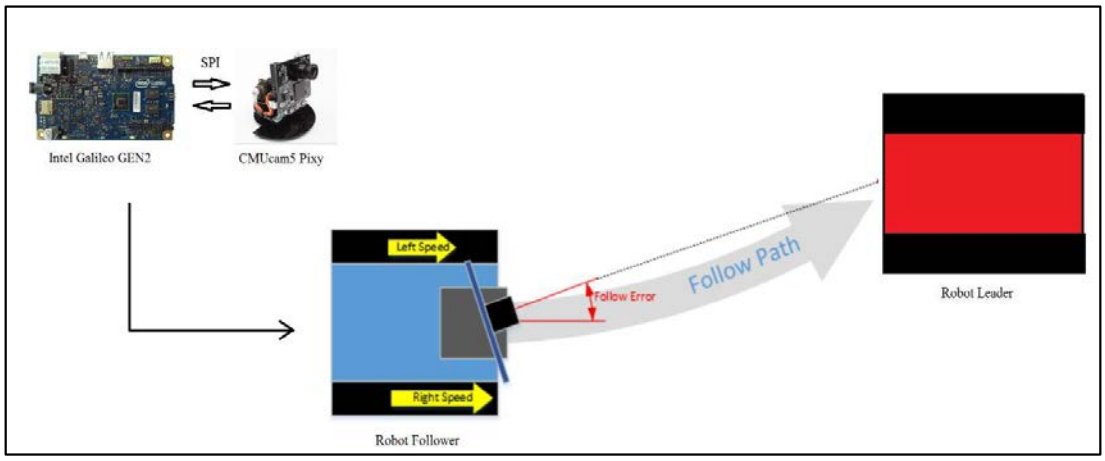


FIGURE 6: Proposed Robot System

3.3.1 Hardware

TABLE 3: Robot Follower Hardware

No.	Model	Qt	Price (MYR)
1	**Intel Galileo Generation 2 – Microcontroller** • Processor: Intel Quark SoC X1000 • Speed: 400MHz • Cores: 1/1 • ISA: 32-bit Intel Pentium processory-compatible • L1 Cache: 16KB • SRAM: 512KB on-die, embedded: 800 MT/s • Technologies supported: Integrated real-time clock, Optional 3V coin cell battery for operation between turn-on cycles • Firmward on 8MB NOR Flash • DRAM: 256MB DDR3 • SD Card (optional): Up to 32 GB • USB: compatible with any USB 2.0 storage device • EERPOM: 8KB • Dimensions: 123.8mm(L) x 72.0mm(W) • Connectors: o 6-pin console UART o 6-pin ICSP o 10-pin JTAG for Debugging o RJ45 Ethernet, Power over Ethernet capable o USB 2.0 Host (standard Type A) o USB 2.0 Client (micro-B) o Mini-PCI Express 1 x slot o Jack with increased range (7V to 15V DC)	1	338.14
2	**CMUcam5 Pixy – Vision System** • Processor: NXP LPC4330, 204 MHz, dual core • Image sensor: Omnivision OV9715, 1/4", 1280x800 • Lens field-of-view: 75 degrees horizontal, 47 degrees vertical • Lens type: standard M12 (several different types available) • Power consumption: 140 mA typical • Power input: USB input (5V) or unregulated input (6V to 10V) • RAM: 264K bytes • Flash: 1M bytes • Available data outputs: UART serial, SPI, I2C, USB, digital, analog	1	236.00

	• Dimensions: 2.1" x 2.0" x 1.4 • Weight: 27 grams		
3	**Pixy Pan/Tilt mechanism** • 5v servos	1	63.40
4	**Cytron MDD10A – Dual H-bridge Motor Driver** • Bi-directional control for 2 brushed DC motor. • Support motor voltage ranges from 5V to 25V. • Maximum current up to 10A continuous and 30A peak (10 second) for each channel. • Speed control PWM frequency up to 20KHz	1	73.00
5	**Parallax Stingray – Robot Platform** • Two DC spur gear motors • Two-wheel differential drive system with rear omnidirectional wheel • Multiple mounting locations for sensors, add-ons, etc. • Power requirements: 6 AA 1.2 V Rechargeable Batteries (not included) • Operating temperature: +32 to +158 Â°F (0 to +70 Â°C) • Dimensions, assembled: 13 x 10.9 x 5.5 in (330 x 277 x 140 mm)	1	1,300

TABLE 4: Robot Leader Hardware

No.	Model	Qt	Price (MYR)
1	**4WD Hercules Mobile Robotic Platform** ➢ Hercules Dual 15A 6-20V with Atmega328P IC ➢ Reducing Motor Working voltage: 4.0-7.0 VDC • Rotation: CCW • Startup voltage: ≤1.2V • Load torque: 1500.0 g.cm • Stalling torque: ≥7.0 Kg.cm • Stalling current: ≤9.0A No Load • Current: 280mA • Speed: 310±12% rpm Load • Current: 1600mA(max) • Speed: 260±12% rpm	1	600.00

3.3.2 Software

TABLE 5: List of software

No	Name	Version	Features	Qt	Price (RM)
1	PixyMon	0.1.49	➢ PixyMon is an application that runs on PC or Mac. ➢ It allows you to see what Pixy sees, either as raw or processed video. ➢ It also allows you to configure your Pixy, set the output port and manage color signatures	1	Open-source
2	Arduino-Intel IDE	1.5.3	➢ The open-source Arduino-Intel environment makes it easy to write code and upload it to the i/o board. ➢ It runs on Windows, Mac OS X, and Linux. ➢ The environment is written in Java and based on Processing, avr-gcc, and other open source software.	1	Open-source

3.4 Proposed algorithm design

The algorithm flow chart of the robot follower is presented in FIGURE 8 below. Instead of finding blocks as shown in Chapter II, Section 2.5, this algorithm is modified to locate the robot leader. The algorithm firstly define the x and y centre and the minimum and maximum FOV angle. The robot follower then will start to search the robot leader. Once the robot leader was located the camera on-board of the robot leader will continuously track it. The camera will reposition its view to centred frame. The robot follower will drive the motor forward and approach the robot leader. The robot follower will gradually decrease its motor speed once the desired distance (object size in FOV) has been computed.

3.4.1 State Machine Diagram

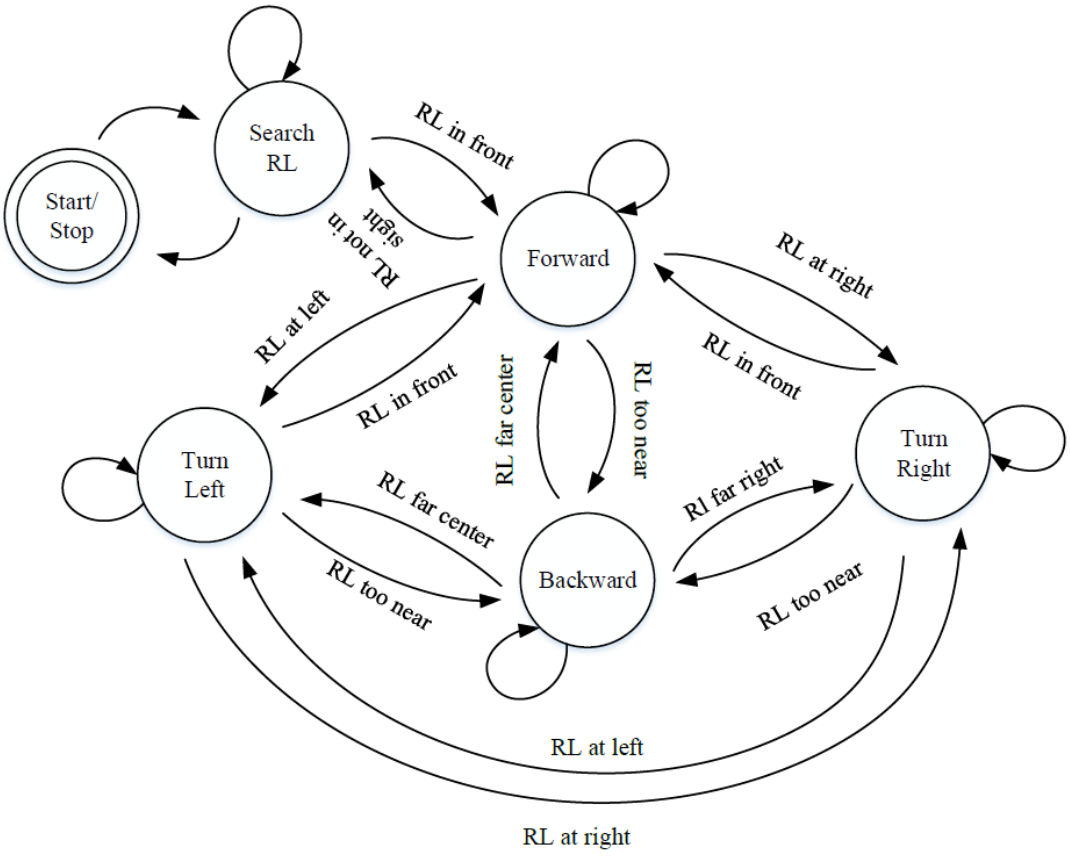


FIGURE 7: State Machine Diagram

3.4.2 Flow Chart

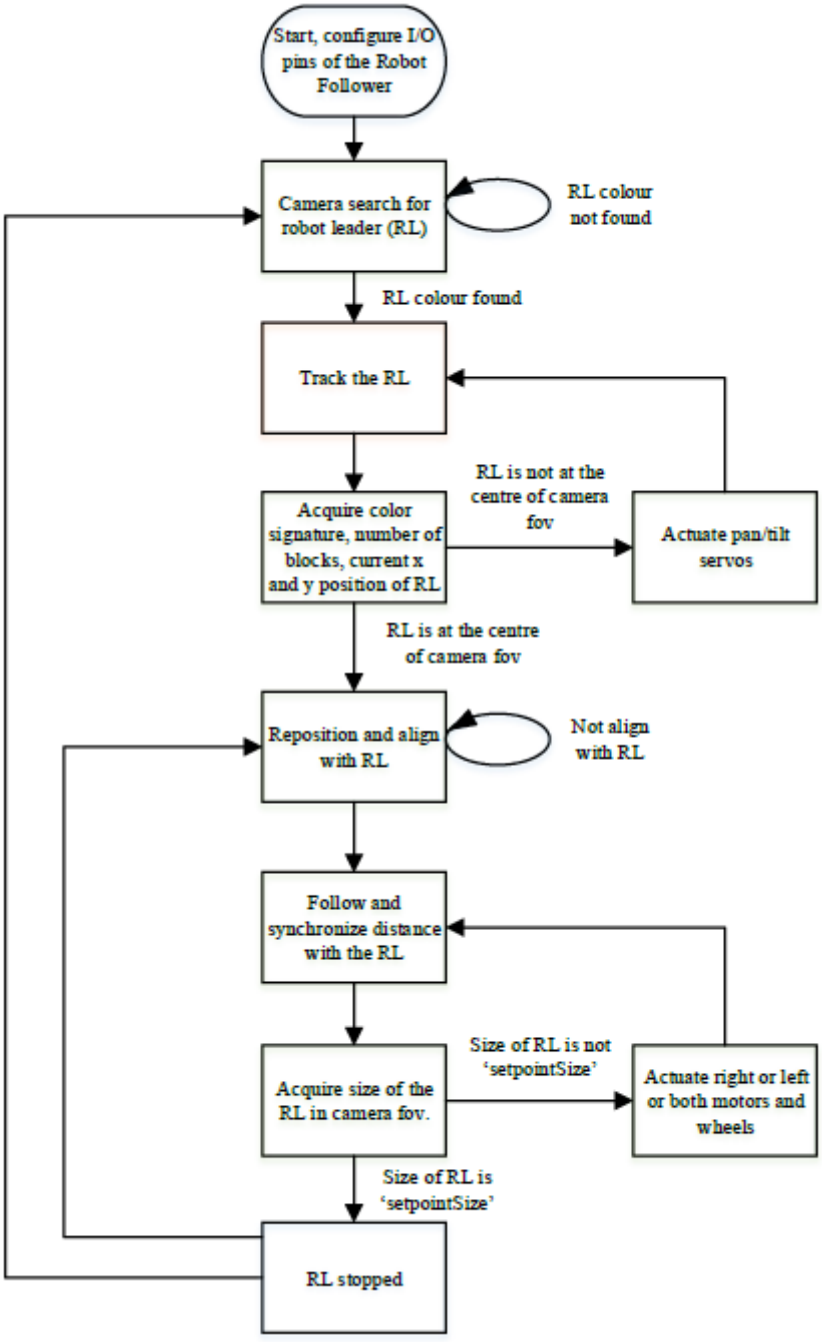


FIGURE 8: Flow Chart

3.4.3 Series of Diagram

FIGURE 9 explained the sequence of the algorithm in tracking and following the robot leader. The diagram is a 2 dimension and top view.

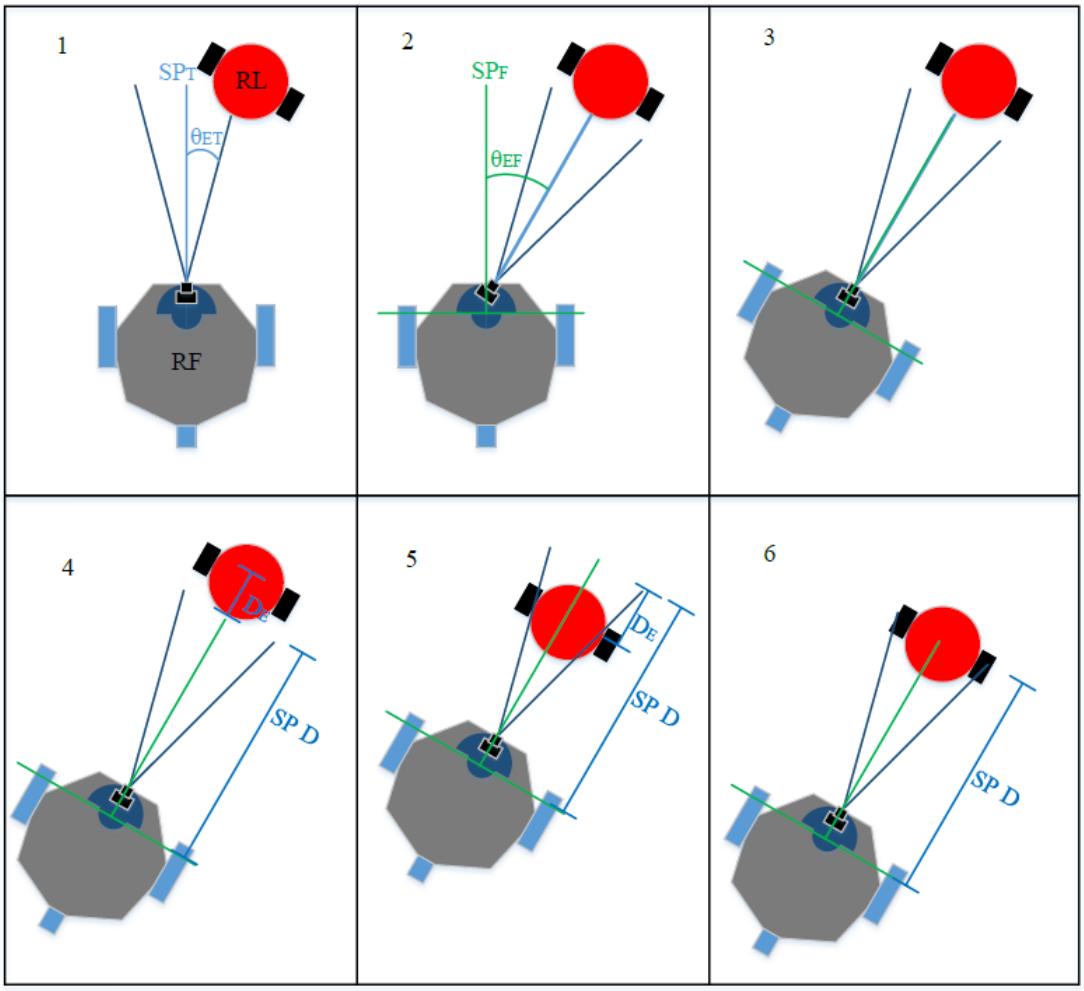


FIGURE 9: Series of Diagram

In the first diagram, the robot leader (RL) is a red coloured circle with black wheels. While the robot follower (RF) is a silver coloured pentagon shaped with 3 blue coloured wheels. The CMUcam5 Pixy with pan/tilt mechanism is shown installed on top of the RF. The set point for RL tracking is detonated as SPT (Set point track) in the diagram, which is at the centre of the camera field of view. In the same diagram, there is an approximate angle error, θET (angle error track), is the error between the centre of the RL and the SPT.

Next diagram showing that the camera pan/tilt mechanism has adjusted the camera field of view to the SPT angle. However the RF is still not facing the RL. Thus an angle error between the θEF (angle error follow) and the setpoint of SPF (set point follow) existed.

In the third diagram the RF has finally centred its position towards the RL. Hence there is no error existed.

In the fourth diagram, the RF has to keep up its speed to ensure the distance between herself and RL is synchronise. Another set point for distance between RL and RF was calculated and detonated as SP D (set point distance). Currently, the RF is lagging behind the desired SP D. A distance error, DE (Distance error) is shown in the diagram.

The RF accelerated to catch up with RL. However, it has miscalculated its distance due to the dynamic process, and noises such as frictions, lighting, and exposures. The DE now existed in the negative value.

Lastly, the RF managed to adjust its speed and distance with the RL. The algorithm sequence loop again.

3.5 Fuzzy Logic Controller Design

A single input, single output fuzzy logic controller (FLC) is designed to compute the linear velocity V of the robot follower in order to follow the moving target, the robot leader by maintaining a desired distance, Dd. The robot leader colour information is captured by CMUcam and then computed based on the desired pixel area size. The size is then translated into approximate distances forms the input to fuzzy controller while the outputs from the controller is pulse width modulated signals to control left and right motor speeds.

MATLAB® Fuzzy Logic Toolbox is utilized to aid in fuzzy logic controller (FLC) design. The toolbox contains functions, graphical user interfaces and data structures that allow the user to quickly design, test, simulate and modify a fuzzy inference system. The steps in FLC design are described in this section.

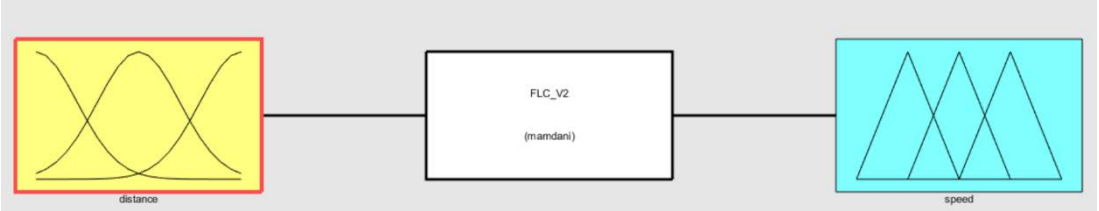


FIGURE 10: MATLAB Fuzzy Logic Toolbox

3.5.1 Fuzzification

Fuzzification is a process of transforming crisp values into grades of membership corresponding to fuzzy sets expressing linguistic terms. The distance relationship is determine by the size of the tracked blocks, where the block is the robot leader in this case. As the distance between the robot follower and the robot leader increases, the size of the block in the camera decreases. The size of block or distances are described by five fuzzy sets, very near, near, desired, far, very far. Their membership functions are expressed in the FIGURE 11 below:

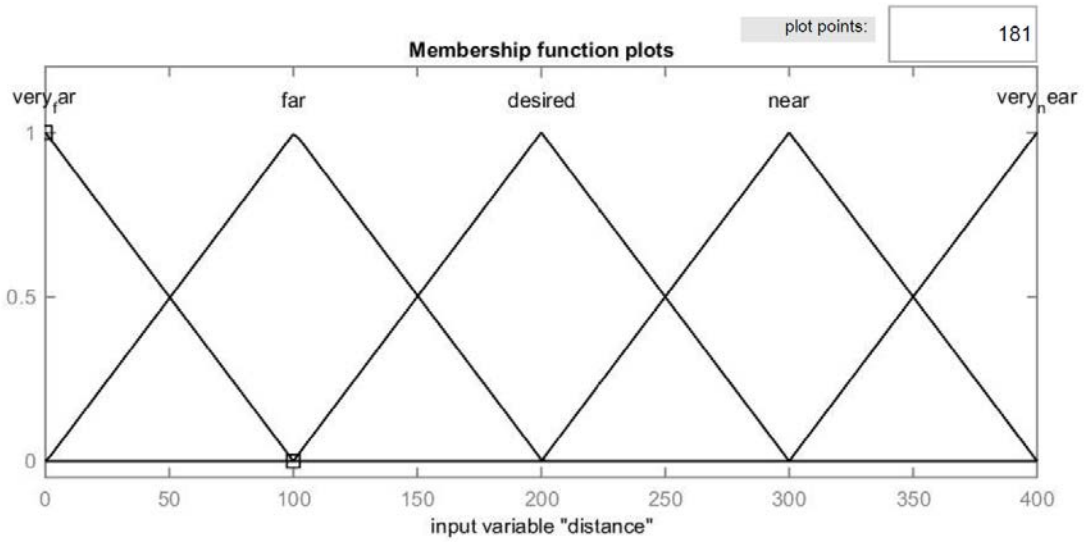


FIGURE 11: Fuzzification Membership Function Plots

TABLE 6: Fuzzification MF Parameters

Membership Functions	Type	Parameters
Very Far	Triangle	[-100 0 0 100]
Far	Triangle	[0 100 100 200]
Desired	Triangle	[100 200 200 300]
Near	Triangle	[200 300 300 400]
Very Near	Triangle	[300 400 400 500]

The rule of thumbs in designing fuzzification MF is to ensure that each neighbour functions will overlap at minimum of 25%.

3.5.2 Rule Implication

The Mamdani implication rule is used to map the input to the output. The Mamdani implication rule expression is given in EQUATION 2.

$$\emptyset[\mu_A(x), \mu_B(y)] = \mu_A(x) \hat{} \mu_B(y)$$

EQUATION 2: Mamdani Rule Implication Equation

where μA(x) is the input membership function, the distance, and μB(y) is the output membership function which is the speedPWM. The implication result is a fuzzy set, which is a minimum of the membership function of the input (distance) and output (speedPWM). The minimum membership values for the antecedent (input) propagates through to the consequent and truncates the membership function for the consequent (output) of each rule. The designed fuzzy rules are listed in TABLE 7 below.

TABLE 7: Fuzzy Rules Assignment

Rule	Description
1	If (Distance is very far) then (Speed PWM is very fast)
2	If (Distance is far) then (Speed PWM is fast)
3	If (Distance is desired) then (Speed PWM is desired)
4	If (Distance is near) then (Speed PWM is slow)
5	If (Distance is very near) then (Speed PWM is reverse)

3.5.3 Defuzzification

Defuzzification is the process of producing a quantifiable result in fuzzy logic, given fuzzy sets and corresponding membership degrees. The output variable of the deffuzification is the speed of motor in terms of pulse width modulation (PWM). Although PWM is typically valued between 0-255, in this paper the PWM was mapped to better resolution of 400, and when it falls in negative values the motor shall be in reverse mode. Similar to the FLC inputs there are also 5 fuzzy sets which are detonated as reverse, slow, desired, fast, and very fast. This membership functions are plotted using MATLAB Fuzzy Logic Toolbox in the FIGURE 12, and the parameters and type of plots are listed in TABLE 8.

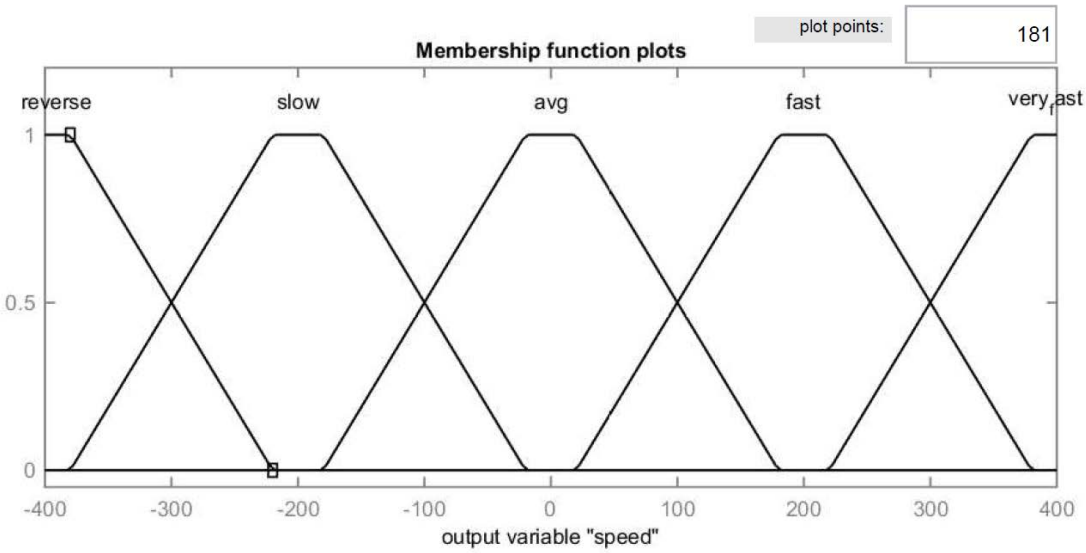


FIGURE 12: Defuzzification Membership Function Plots

TABLE 8: Defuzzification of MF

Membership Functions	Type	Parameters
Reverse	Trapezoid	[-580 -420 -380 -220]
Slow	Trapezoid	[-380 -220 -180 -20]
Desired	Trapezoid	[-180 -20 -20 180]
Fast	Trapezoid	[20 180 220 380]
Very Fast	Trapezoid	[220 380 420 580]

One of the most popular defuzzification methods is the centroid introduced by Sugeno in 1985. This method is also known as centre of gravity or centre of area defuzzification. Centroid defuzzification returns the centre of area under the curve. The formula of centroid defuzzification is shown in EQUATION 3.

$$x^* = \frac{\int \mu_{Speed}(x) x \, dx}{\int \mu_{Speed}(x) \, dx}$$

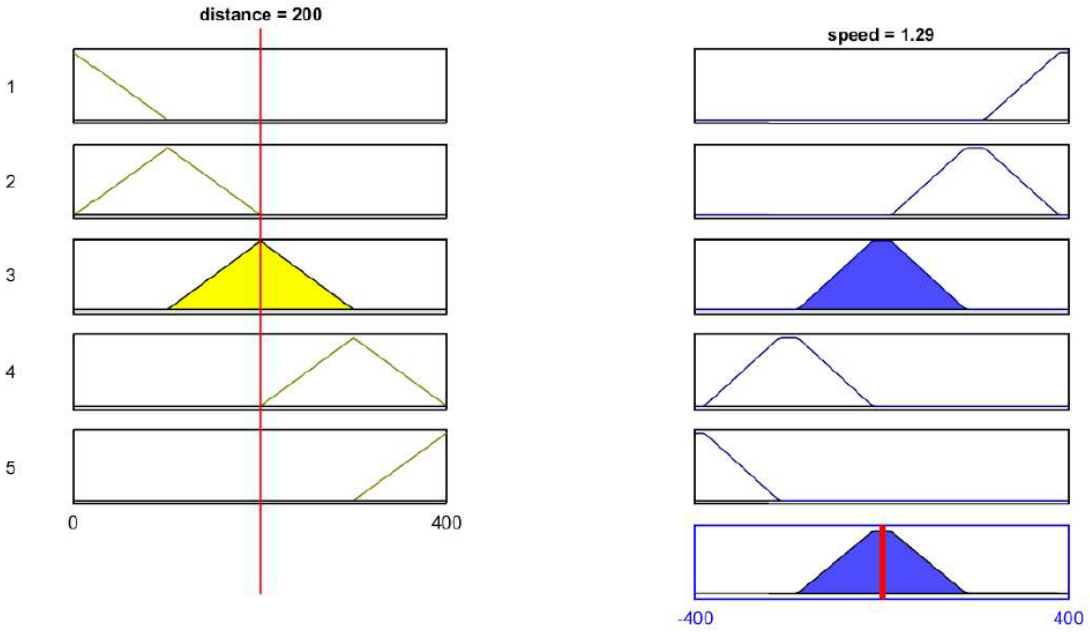
EQUATION 3: Centroid Defuzzification formula

Where x* is the defuzzified output, μi(x) is the aggregated membership function and x is the output variable.

3.5.4 Fuzzy Inference Diagram

According to Mathworks, the fuzzy inference diagram is the composite of all the smaller diagrams presented so far in this section. It simultaneously displays all parts of the fuzzy inference process you have examined.

The fuzzy inference diagram can be viewed in The Rule Viewer in MATLAB® Fuzzy Logic Toolbox. FIGURE 13 shows the fuzzy inference diagram of this research.



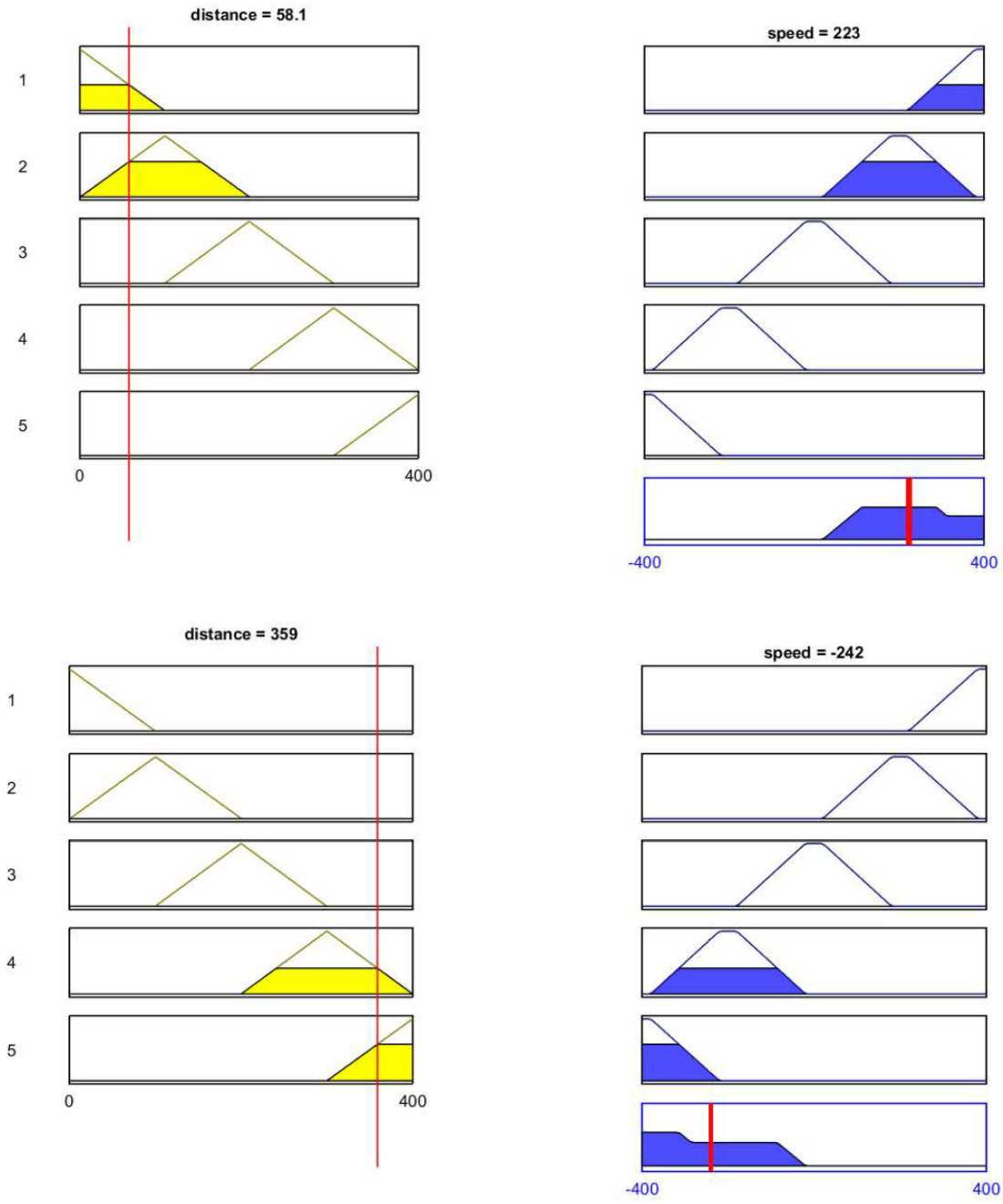


FIGURE 13: Rule Viewer MATLAB

The Surface Viewer is another GUI tool in MATLAB® Fuzzy Logic Toolbox that lets the user to examine the output surface of a FIS stored in a file, a.fis, for any one or two inputs. The surface plotted for this research is shown in FIGURE 14.

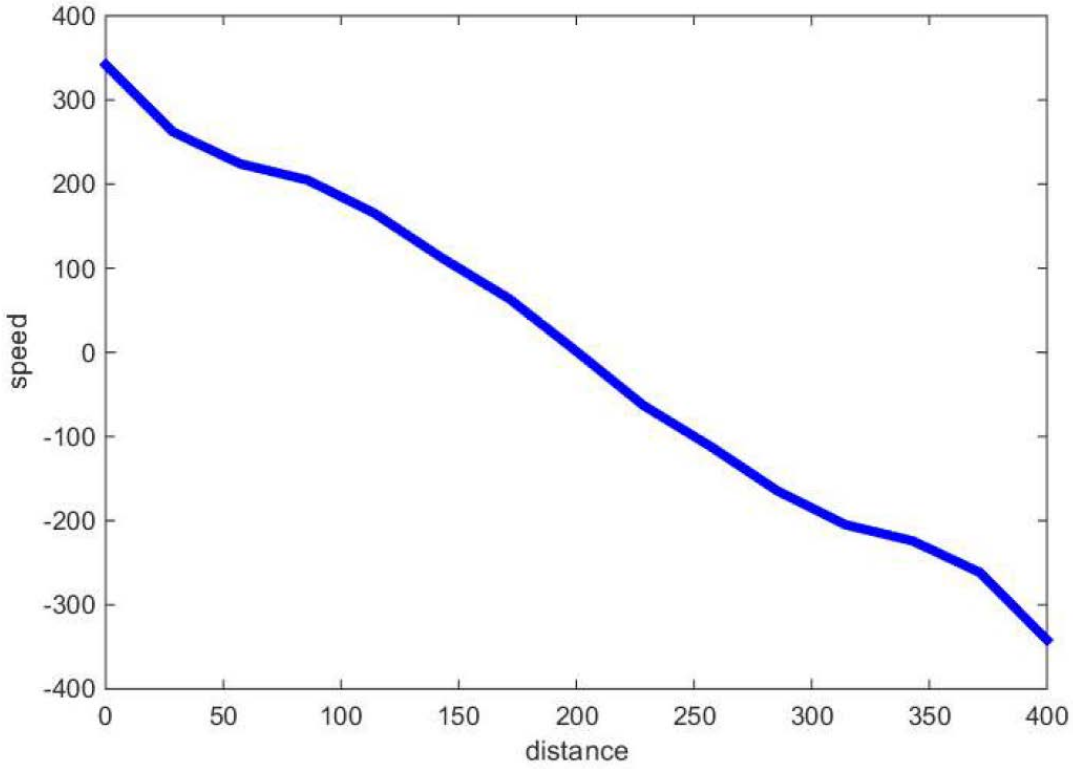


FIGURE 14: Surface Viewer MATLAB

The plotted graph shows that the relation between distance and speed is almost linear.

3.6 Validation Planning

A robot leader will be programmed with predetermined path. The path can be a circle, rectangular, or random path. The robot follower will follow the robot leader in a line formation. The author will measure the changing information for the distance, angle, and velocity as the time changes for both robot leader (expected value) and robot follower (true value). The accuracy of the measurement will be calculated and tabulated. Several errors such as deviation error, sensitivity error, hysteresis error, and so forth will be calculated as well.

To obtain the information of the measurement several ways will be implemented such as using fixed camera (top view) to record the robot movement (also can be used to calculate approximate distance and angle), FOV of on-board camera to get the approximate distance, and also rotary encoder sensor for velocity measurement.

CHAPTER IV
RESULTS & DISCUSSION

This section discusses the results that the author has obtained in FYP II. The project is still under development.

4.1 Camera configuration

Object detection is performed by using the embedded vision called the CMUcam5 Pixy. This embedded vision is an image sensor that able to detect practically anything. It uses a hue-based colour filtering algorithm to detect objects. CMUcam5 works by calculating the hue and saturation of each RGB pixel from the image sensor and uses these as the primary filtering parameters. The hue of an object remains largely unchanged with changes in lighting and exposure. Changes in lighting and exposure can have a frustrating effect on colour filtering algorithms, causing them to break.

The author has conducted several testing on the camera. By altering the configuration parameters inside the CMucam5 Pixy the object detection performances can be altered too. The TABLE 9 below shows the configuration parameters of CMUcam5 Pixy.

TABLE 9: CMUcam5 Pixy Configuration Parameters

Parameter	Function	Value
Max blocks	Sets the maximum total blocks sent per frame.	1000
Max blocks per signature	Sets the maximum blocks for each colour signature sent for each frame.	1000
Min block area	Sets the minimum required area in pixels for a block. Blocks with less area won't be sent.	20
Min saturation	Sets the minimum allowed colour saturation for when generating colour signatures. Applies during	20.0
Hue spread	Sets how inclusive the colour signatures are with respect to hue. Applies during teaching.	1.0
Saturation spread	Sets how inclusive the colour signatures are with respect to saturation. Applies during teaching.	1.0
Data out port	Selects the port that's is used to output data. 0=SPI, 1=I2C,2=UART, 3=analog/digital x,	0
I2C address	Sets the I2C address if you are using I2C data out port.	0x54
UART baud rate	Sets the UART baud rate if you are using UART data out port.	19200
Default program	Selects the program number that's run by default upon power-up.	0
Brightness	Sets the average brightness of the camera, can be between 0 and 255.	150

There are total of 7 colours signature can be configured on the Pixy. In addition to that the new firmware provided by the developer has abled Pixy to detect objects based on colour code. Colour code is a combination of two or more unique colours. By using colour code the Pixy now able to detect and identify objects more efficiently while reducing the colours hindrance on the background. The FIGURE 15 below shows the colour code being configured to Pixy.

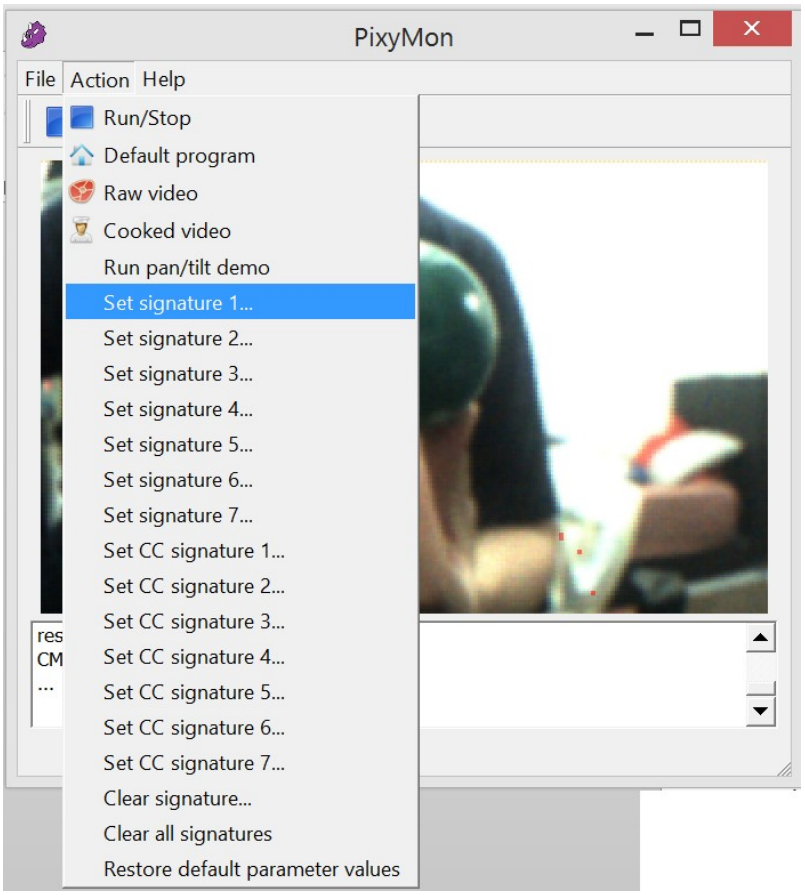


FIGURE 15: Setting colour signature in PixyMon

There are two ways to configure the colour signature. The first method is by using PixyMon (a graphical user interface) on either Windows platform, Linux, or Mac. The second method is by the physical switch on-board of the Pixy. In this project the author has preferred to configure the colour signature via PixyMon since it provides the ability for the user to actually see what the camera is output, hence ease the troubleshooting and calibration.

4.1.1 Tracking single colour and combined colour

To track object the Pixy must be installed with pan/tilt mechanism. The pan/tilt mechanism used in this project consists of two micro servos to change the viewing angle of the camera. An algorithm to track the object were uploaded into Pixy. This algorithm is to ensure that the object that is being track and is always at the centre of the camera view plane (where x-plane=160L, y-plane = 100L). Few balls with solid colours of red, green, blue, and yellow were experimented to identify the Pixy responses.

In this experiment, all the balls were tracked by Pixy under the same condition of lighting and exposure. However, the smoothness of the camera during the first testing is jerky and unstable. Hence, the proportional gain were reduced to reduce the pan/tilt movement errors. FIGURE 16 shows the Pixy detecting various colours and a colour code.

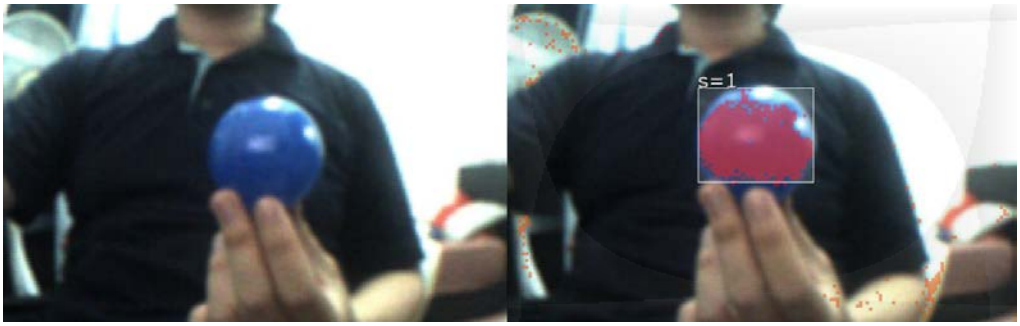


FIGURE 16: CMUcam5 Pixy detecting a signature

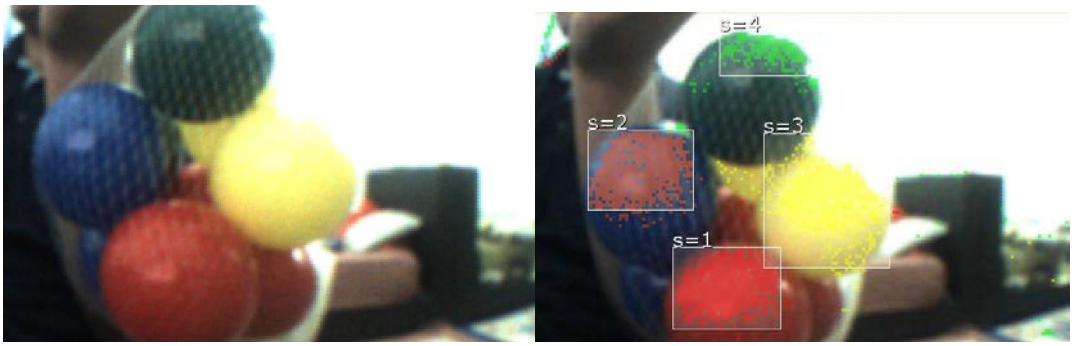


FIGURE 17: CMUcam5 Pixy detecting multiple signature

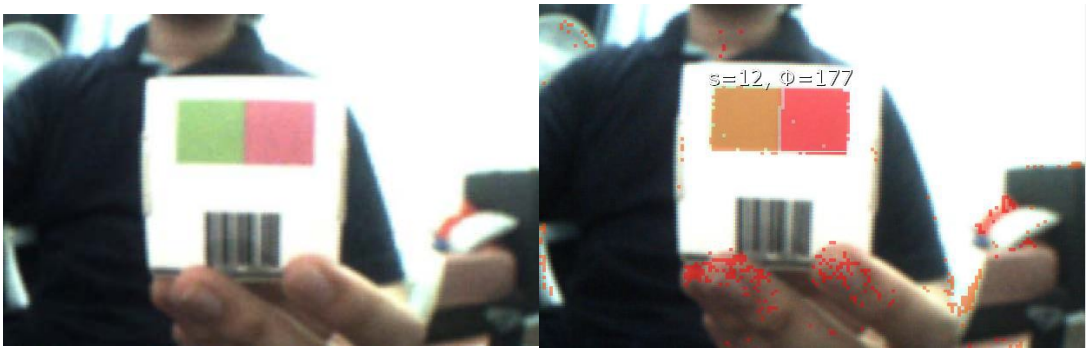


FIGURE 18: Pixy detecting colour codes with angle shown

The objective of this experiment is to identify how the camera measure the angle of the object. This is important as the robot leader (RL) will move dynamically and will change its direction of motions while it is moving. When the RL changed its direction or steering, the colour code (which is being tracked by the RF) attached at the rear of the RL will be angled and the block size will be reduced. Hence, this problem may incur noise data to the RF. In this experiment the author has recorded the experiment data and tabulated it into table and line chart. The results of the object colour code tracking is tabulated in the TABLE 10 below and in chart FIGURE 19.

TABLE 10: Colour code detection angle

Actual angle	Measured angle	Angle error
0	-2	2
45	42	3
90	86	4
135	139	4
180	180	0
-135	-141	6
-90	-91	1
-45	-37	8

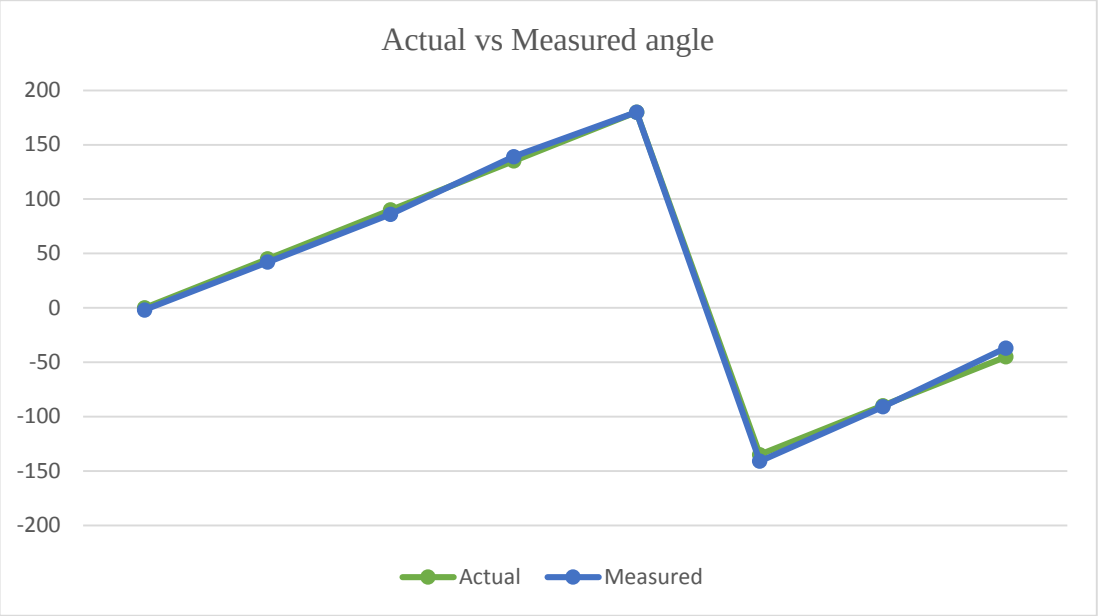


FIGURE 19: Actual angle vs Measured angle chart for CC

4.1.2 Tracking Infrared LED

By using custom firmware, IR-LOCK filter, and IR lens (3.6 mm) the CMUcam5 Pixy now able to detect IR lights source. This modification was developed by Thomas Stone, an engineer, and entrepreneur from Atlanta, United States. One of the advantages of this modification has allows the camera to track IR LED without being affected by external disturbances and noises, such as the variances of light exposure from sunlight and lamp, the background colours and the colours of the object as well. With this implementation, has allows the camera to be operable under extreme day light, and also in very dark environment. FIGURE 20 and FIGURE 21 shows the modified camera output.

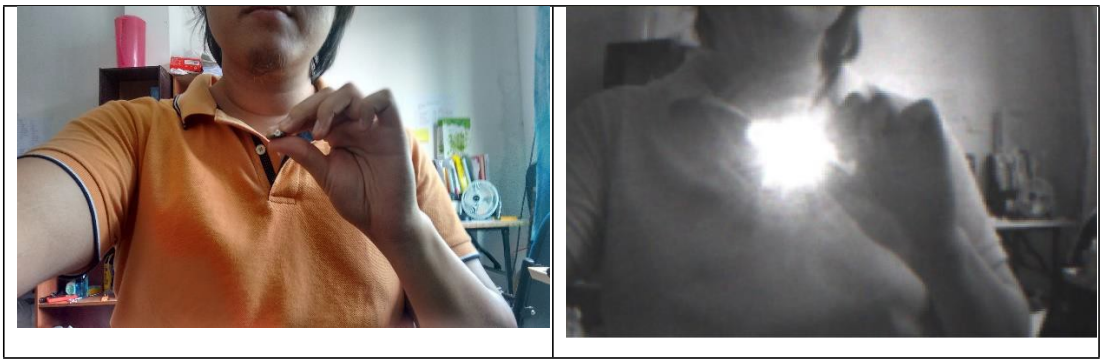


FIGURE 20: CMUcam5 Pixy IR LOCK output 1

FIGURE 21: CMUcam5 Pixy IR LOCK Output 2

4.1.3 Tracking Robot Leader

The robot leader is a green coloured wheel mobile robot. CMUcam5 Pixy was configured to detect the desired colour. Several information such as type of colour signature, the height and the width of the block, and the x-y coordinates of the block in the camera field of view (FOV) is then pass to the main microcontroller of the robot follower for further computation process.

FIGURE 22 shows that the CMUcam is detecting the robot leader platform. The computer vision has highlighted the green area that is bounded by a block.

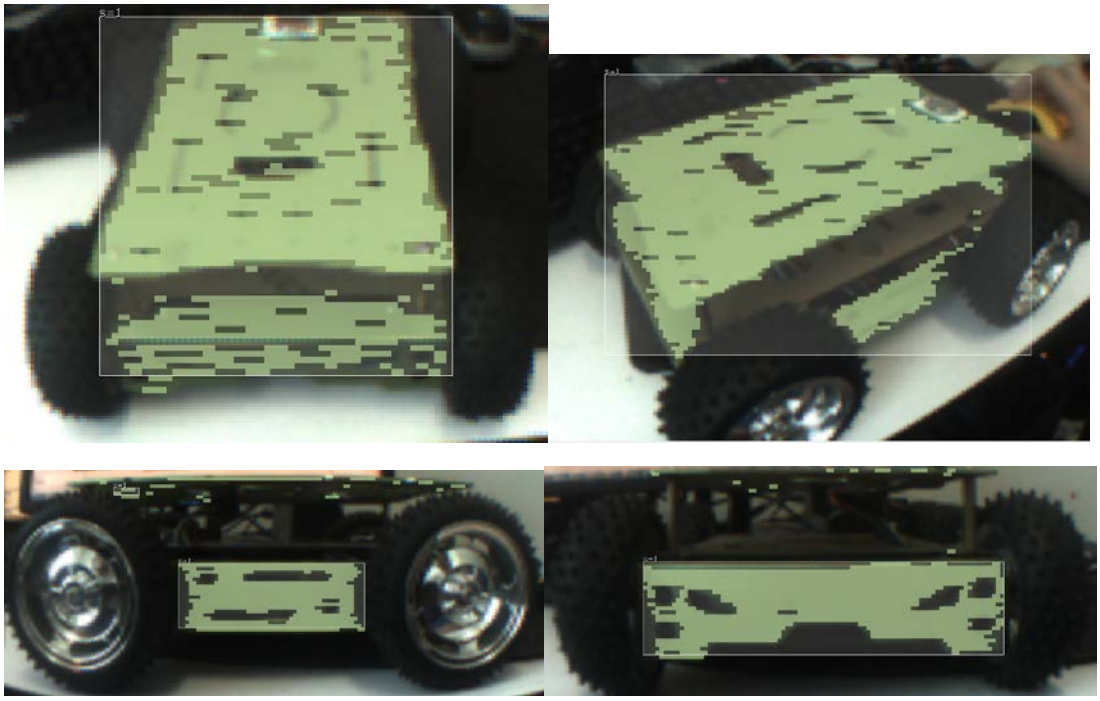


FIGURE 22: Tracking Robot Leader

4.2 System integration

The author will be using Intel Galileo (Gen 2) as the microcontroller for the robot follower. The main functions of the microcontroller is to control the angle of the CMUcam5 Pixy pan/tilt mechanism as well as the robot motors motions. The connection between Intel Galileo and Pixy is via SPI interface. The Intel Galileo operates at maximum of 5.0V, hence it is safe to connect the Pixy directly to Intel Galileo without the need of voltage regulator or second voltage supply. The connection of the Intel Galileo and Pixy is shown below.

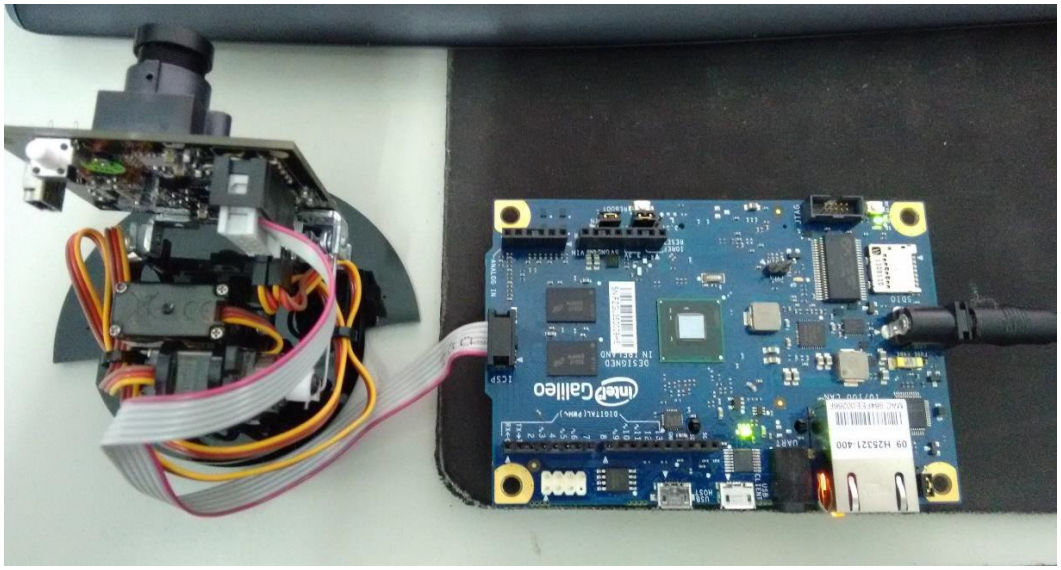


FIGURE 23: SPI connection between CMUcam5 Pixy and Intel Galileo

A program contained SPI library and program to catch the data from Pixy is shown below. This code can be downloaded from the Pixy developer website. FIGURE 24 below shows the serial output from Pixy.

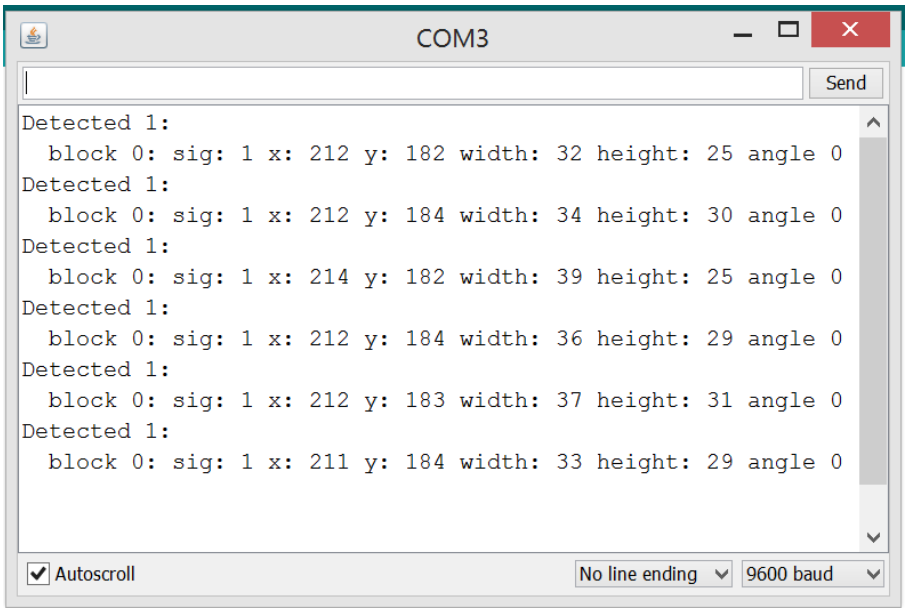


FIGURE 24: Serial output in Arduino-Intel Galileo IDE

Besides SPI interface, the Pixy can be communicated via other interfaces as well such as the UART, I2C, and also analog/digital pin. The author has experimented both the UART and I2C interface. To connect Pixy to microcontroller using another interfaces one must configure the Data out port inside the Pixy. The list below shows the available data out port and its configuration number.

- 0: SPI - this is the default port that uses 3 wires (pins 1, 3, and 4 of the I/O connector) and is used to communicate with Arduino
- 1: I2C - this is a multi-drop 2-wire port (pins 5 and 9 of the I/O connector) that allows a single master to communicate with up to 127 slaves (up to 127 Pixys).
- 2: UART - this is the common "serial port" (pins 1 and 4 of the I/O connector). Pixy receives data via pin 1 (input) and transmits data via pin 4 (output).
- 3: analog/digital x - this will output the x value of the largest detected object as an analog value between 0 and 3.3V (pin 3). It also outputs whether an object is detected or not as a digital signal (pin 1 of the I/O connector).

- 4: analog/digital y - this will output the y value of the largest detected object as an analog value between 0 and 3.3V (pin 3). It also outputs whether an object is detected or not as a digital signal (pin 1 of the I/O connector).

The speed of using both interfaces are relatively the same, however to simplify things, SPI was chosen as the communication interface.

4.2.1 Robot Follower Platform

The robot follower and robot leader is built with different hardware setup. The robot follower is a 2-Wheel Mobile Robot and is setup with Intel Galileo Generation 2 as the main microcontroller, an onboard camera using CMUcam5 Pixy, a rotary encoder to capture the robot velocity, and everything was mounted onto a Parallex Stingray robot platform. The robot follower platform pictures are shown in FIGURE 25.

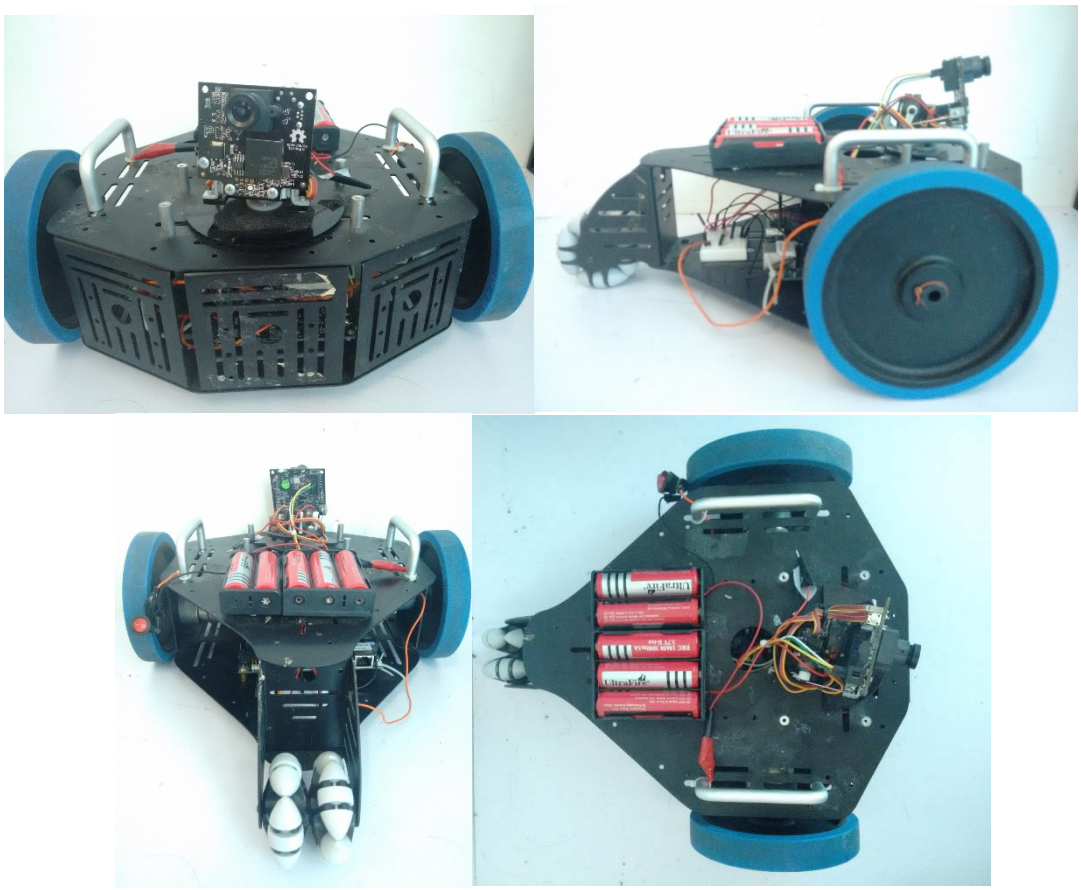


FIGURE 25: Robot Follower Platform

4.2.2 Robot Leader Platform

For the robot leader the configuration is rather simpler, it is a 4-wheel Mobile Robot by Seedstudio called the Hercules 4WD. It uses the Atmega328P as the microcontroller. The robot leader was programmed to autonomously move in specific formations. The robot leader platform pictures are shown in FIGURE 26.

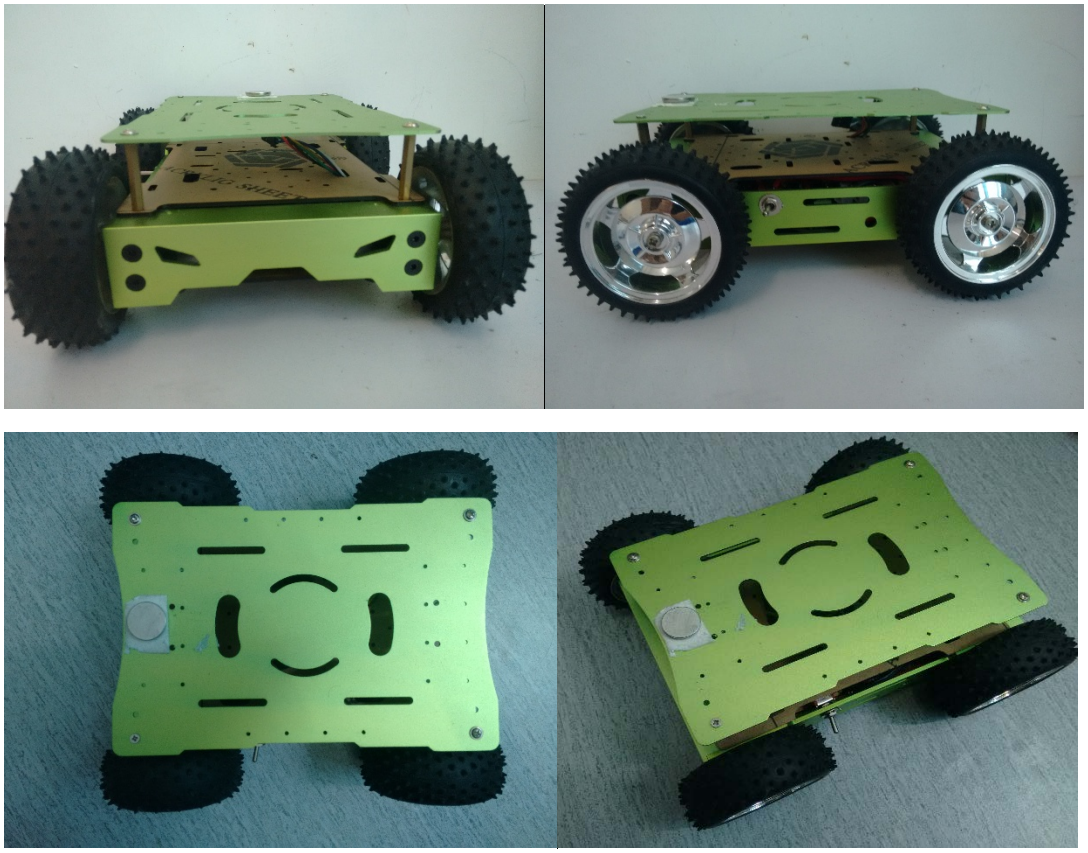


FIGURE 26: Robot Leader Platform

4.3 PD Controller vs FLC

An object, a blue coloured ball, is used to represent the robot leader. Parts of the algorithm is using proportional differential controller to control the robot acceleration toward the detected object.

4.3.1 Proportional Differential Controller Implementation

The Proportional, and Differential (PD) controller in this project was designed to control the camera pan/tilt servos, and to control the robot motions and speed. The PID equation as shown in EQUATION 1 is translated into the form of pseudo code and to C programming. In this particular multi-robot system we assume Ki to be 0.

The CMUcam5 Pixy system can be controlled using proportional control only or both proportional and derivative control. For robot following proportional control can be used alone. But for robot tracking proportional and derivative control are required.

Proportional control allows for a much smoother response than simple on or off control. Proportional control calculates an output value that is proportional to the magnitude of the error. Small errors yield a small response. Larger errors result in a more aggressive response. Proportional control can be used alone, or augmented with Integral or Derivative control as needed.

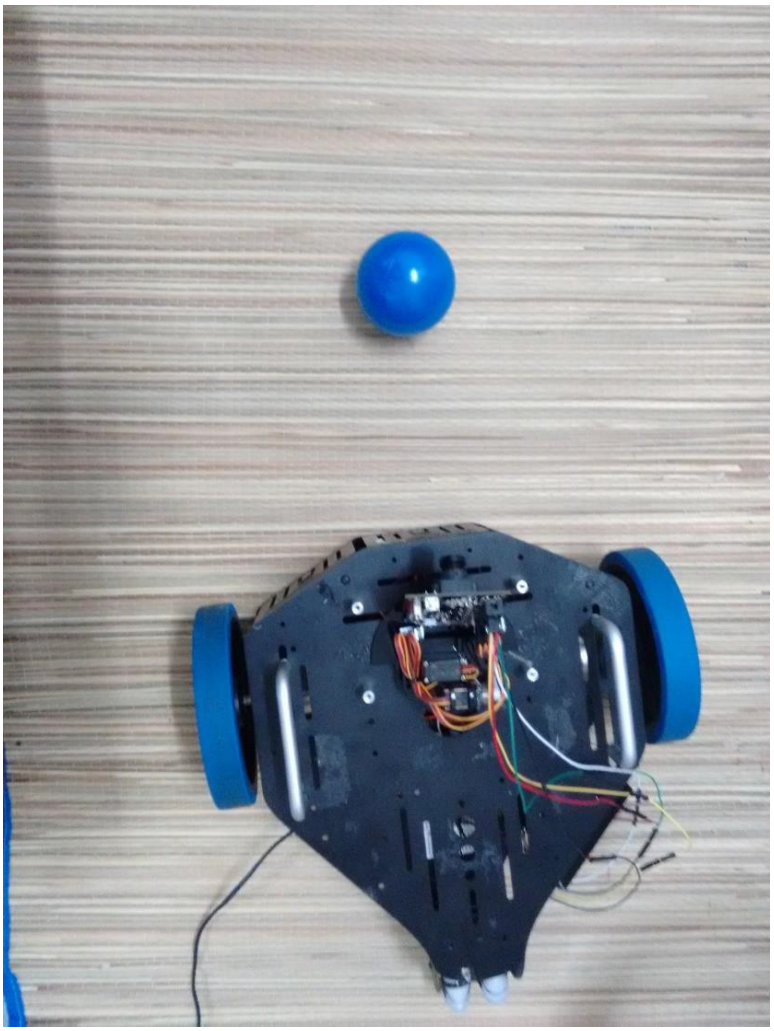


FIGURE 27: Robot follower is tracking a blue colour ball (top view)

Derivative control looks at the rate of change in the error. If the error is rapidly approaching zero, the output of the derivative calculation attempts to slow things down to avoid overshooting the set point. The CMUcam5 Pixy object tracking algorithm uses derivative control in conjunction with the proportional control to help prevent over-correction when tracking robot leader. Below is the pseudo code for PD.

PD:

Error = Setpoint - Actual

Derivative = (Error - Previous_error)/dt

Drive = (Error*kP) + (Derivative*kD)

Previous_error = Error

wait(dt)

GOTO PID

Tuning methods was applied in determining the Kp and Kd. The value of Kp and Kd tuned for the camera pan mechanism are Kp = 250, Kd = 400, while tilt mechanism are tuned at Kp = 350, Kd = 300.

The Kp for robot motion and speed was programmed with an auto-tune method. The Kp value changes over the size of the object it is following. For the differential value, Kd = 100.

4.3.2 Fuzzy Logic Controller Implementation

The FLC controller designed in MATLAB typically will be saved in fuzzy interference system (.fis) file type. One can convert the .fis into C/C++ codes and then modify it for particular microcontroller. This method however may become more mathematical complex as the number of membership functions increases, and other type of membership function (MF) plots were used. In addition, using this method will prone to programming error as well.

In this research, the author has utilised an Embedded Fuzzy Logic Library (eFLL) library which was developed by the Robotics Research Group (RRG) at the State University of Piauí (UESPI-Teresina). eFLL is a standard library to deploy fuzzy logic into Embedded Systems for easy deployment and programming simplicity. In the next section discuss how the eFLL library works and being implemented into this research.

TABLE 11: eFLL Library list

```
#include <FuzzyRule.h>
#include <FuzzyComposition.h>
#include <Fuzzy.h>
#include <FuzzyRuleConsequent.h>
#include <FuzzyOutput.h>
#include <FuzzyInput.h>
#include <FuzzyIO.h>
#include <FuzzySet.h>
#include <FuzzyRuleAntecedent.h>
```

The library of eFLL includes all the files shown in the codes snippet in TABLE 11.

TABLE 12: eFFL Fuzzification

```
    //Fuzzification of Inputs - Creating the
    FuzzyInput distance
 1  FuzzyInput* distance = new FuzzyInput(1);
    //Creating FuzzySet that make up the
    distanceFuzzyInput
 2  FuzzySet* very_far = new FuzzySet(-180, -20,
    100,150); //Distancia very_far
 3  distance->addFuzzySet(very_far); // AddingFuzzySet
    very_far in distance
 4  FuzzySet* far = new FuzzySet(50, 150, 150, 250);//
    Distance far
 5  distance->addFuzzySet(far); // Adding FuzzySet
    farin distance
 6  FuzzySet* desired = new FuzzySet(150, 200,
    420,500); // Distance desired
 7  distance->addFuzzySet(desired); // AddingFuzzySet
    desired in distance
 8  FuzzySet* near = new FuzzySet(400, 500, 500,
    600);// Distance near
 9  distance->addFuzzySet(near); // Adding
    FuzzySetnear in distance
10 FuzzySet* very_near = new FuzzySet(500, 600,
    840,880); // Distance very_near
11 distance->addFuzzySet(very_near); //
    AddingFuzzySet very_near in distance
12 fuzzy->addFuzzyInput(distance); //
    AddingFuzzyInput in Fuzzy object
```

The fuzzification of fuzzy inputs are processed in the code shown in TABLE 11. In the code, the line 1 initialised a single fuzzy input called distance. The line 2 is the first MF declaration and parameters. As of now the eFFL library only able to compute trapezoid and triangle parameters. Since there is total of 5 MF, the line 2 to line 11 declares all of the MF.

TABLE 13: eFFL Rule Implication

```
    // FuzzyRule "IF distance = desired THEN speed
    = desired"
1.  FuzzyRuleAntecedent* ifDistanceDesired = new
    FuzzyRuleAntecedent(); // Instantiating one
    antecedent to express
2.  ifDistanceDesired ->joinSingle(desired); //
    Adding the corresponding FuzzySet the Background
    object
3.  FuzzyRuleConsequent* thenSpeedDesired = new
4.  FuzzyRuleConsequent(); // Instancinado
    one Consequent to the expression
5.  thenSpeedDesired->addOutput(desired);// Adding
    the corresponding FuzzySet the consequent object
    // Instantiating one FuzzyRule object
6.  FuzzyRule* fuzzyRule03 = new FuzzyRule(3,
    ifDistanceDesired, thenSpeedDesired); //
    Passing the antecedent and the consequent of
    expression
7.  fuzzy->addFuzzyRule(fuzzyRule03); // Adding
    to FuzzyRule Fuzzy object
```

The fuzzy rules are defined in the code as shown in TABLE 13. This code only shows 1 out of 5 rules. The rules shown is IF distance is desired, THEN speed is desired.

TABLE 14: eFFL Defuzzification

```
    //Defuzzification of Outputs - Creating the FuzzyOutput speed
1.  FuzzyOutput* speed = new FuzzyOutput(1); // Creating FuzzySet
    that make up the FuzzyOutput speed
2.  FuzzySet* reverse = new FuzzySet(-580, -420, -320,
    -270); // slow speed
3.  speed->addFuzzySet(reverse); // Adding the slow in speed FuzzySet
4.  FuzzySet* slow = new FuzzySet(-350, -270, -270, - 200); // slow speed
```

```
5. speed->addFuzzySet(slow); // Adding the slow in speed FuzzySet
6. FuzzySet* desired = new FuzzySet(-270, -200, 80, 200); // normal
   speed
7. speed->addFuzzySet(desired); // Adding FuzzySet desired speed in
8. FuzzySet* fast = new FuzzySet(80, 200, 200, 320);
   // fast speed
9. speed->addFuzzySet(fast); // Adding in the fast speed FuzzySet
10.        FuzzySet* very_fast = new FuzzySet(200, 320, 420, 580); //
   very fast speed
11.        speed->addFuzzySet(very_fast); // Adding FuzzySet very fast
   speed in
12.        fuzzy->addFuzzyOutput(speed);    //    Adding
   FuzzyOutput in Fuzzy object
```

The defuzzification code shown in TABLE 14. Similarly to the Fuzzy Inputs, all of the MF fuzzy output sets are defined.

The defuzzification consist of computing the area and centroid of the trapezoid and the triangle which is shown in the code snippet in Table 15 below.

TABLE 15: Trapezoid defuzzification code

```
// Trapezoid
area = ((length _bottom_base + length_top_base) / 2.0) * (height);
X-axis centroid = ((x_axis_point2-x_axis_point1) / 2.0) + x_axis_point1;
numerator    += middle * area;
denominator += area;
```

4.4 Validation

By using MATLAB with Simulink, the PD Controller and FLC was simulated. The results of the simulation is then validated with the real implementation performance of the developed multi robot system. The validation of real implementation performance is based on the observation and assisted by image processing software to track the robot movement. The graph of the simulated controlled is shown in the FIGURE 28 and FIGURE 29.

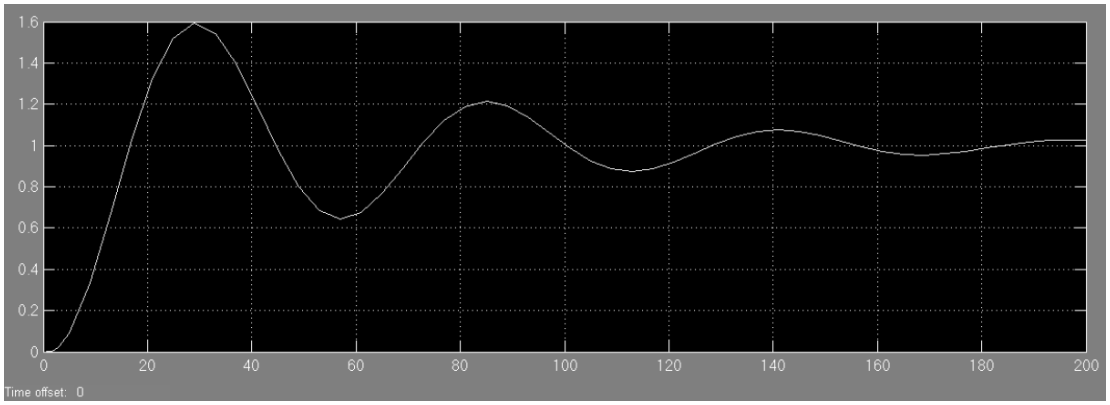


FIGURE 28: PD Controller Step Response

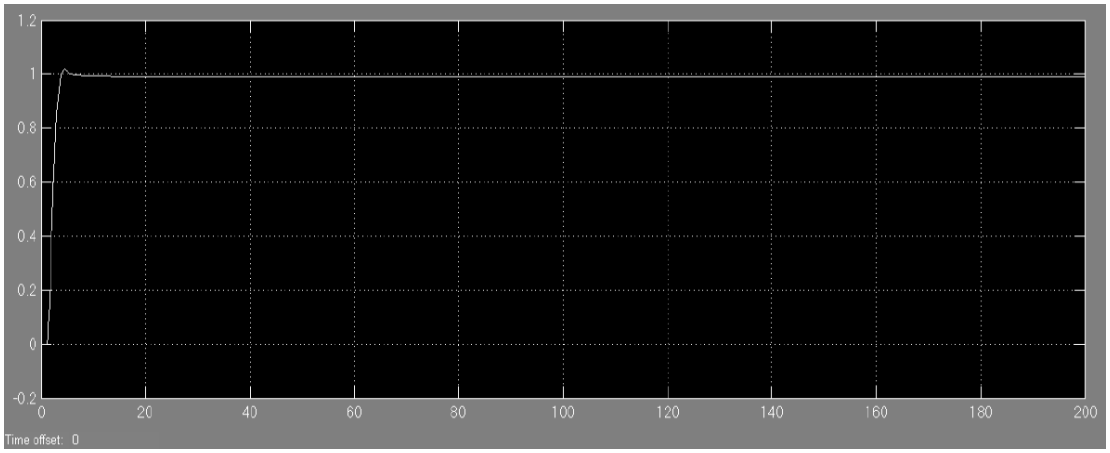


FIGURE 29: FLC Step Response

The graph indicate that FLC has faster response, very minimal overshoot, and converge to steady state really fast as compared to PD controller. The simulation is also validated in real performance via observation. A video of the same robot implemented with either controller was experimented. In the experiment, the robot

follower is required to follow the robot leader. The robot leader will stop at some point. The performance of both controller is observed when the robot leader is moving and also stopping. Based on the observation, the robot follower using PD controller has very high overshoot, and unable to reach steady state and is oscillatory. While when implemented with FLC the robot follower has very minimal overshoot, and reach steady state almost instantaneously.

Another validation process was took where the robot follower implemented with FLC will follow the robot leader in a circle formation. The screenshot of the experiment is shown in the FIGURE 30 and the robot path was recorded into a graph in FIGURE 31.

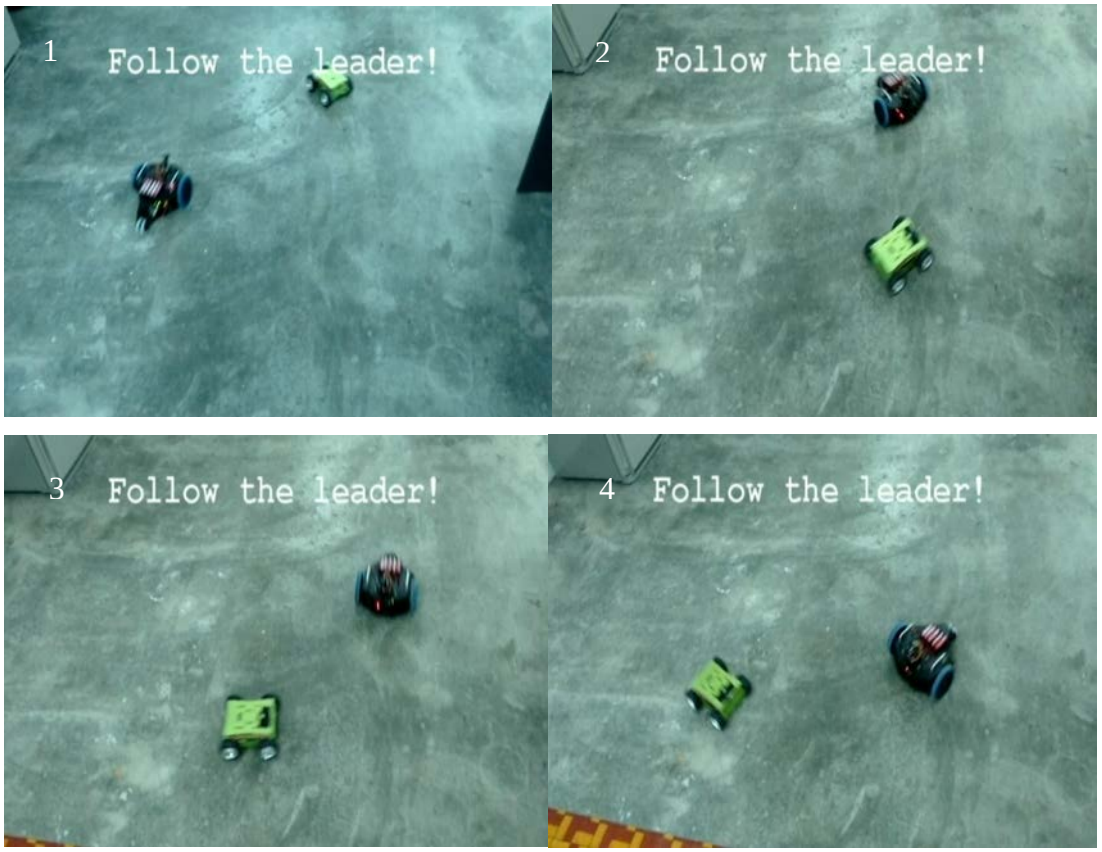


FIGURE 30: Screenshot of follow leader line formation

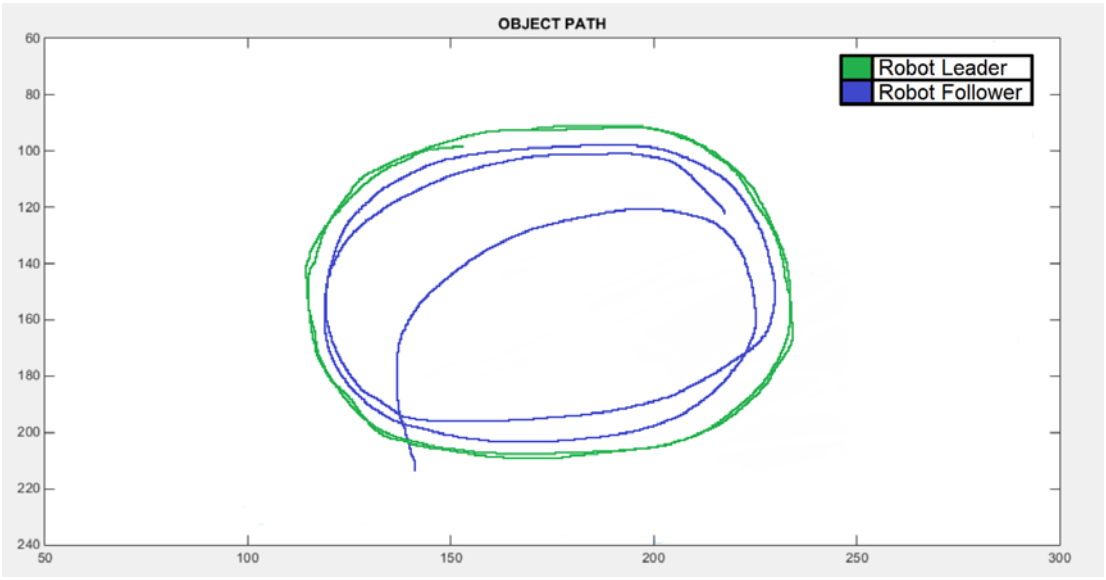


FIGURE 31: Robot Leader and Robot Follower Path

Based on FIGURE 31, the robot follower able to follow the robot leader almost accurately. However, due to differences of robot specifications, motor speed, the robot follower is somehow lagging behind the robot leader. However the trajectories of the robot follower and robot leader is as desired.

The study has shown that FLC has better performances compared to PD controller both in simulation and also real multi-robot applications.

CHAPTER V
CONCLUSION & RECOMMENDATION

5.1 Conclusion

This research proposed the development of multi-robot tethering using camera and intelligent contoller algorithm. Typical multi-robot application requires various sensors and communication devices in order for them to interact. However this is a highly cost solutions since large quantity of sensors and communications devices would be required to equip on every robots.

This project has designed and implemented conventional controller and intelligent controller namely the Proportional-Derivative Controller and Fuzzy Logic Controller into autonomous robot application. The results has shown that the FLC improved the robot navigation performances with reduced overshoot, faster system response, and faster to converge stable state.

In addition, this research has also proved that multi-robot tethering can be achieved by using only a low cost embedded camera on each robot. Robot vision perception has potential of evolving into the next level like living creature vision perception. This study will prove that multi-robot application can be applied efficiently with the use of only monocular vision.

5.2 Recommendation

Image processing holds the future of machine vision. Robotic vision has the ability of more than just colour sensing. A robot should be installed with the ability to recognise objects and human faces. Further study on robot with object recognition and face recognition could improve robotic application greatly. Camera like human eyes, are easily affected by the dynamic lighting environment. The need of vision that able to overcome the poor lighting conditions would be a major achievement.

Design and implementation of modern controller algorithm in robotic application should be looked into as well. Modern controller such as state feedback, observer, and adaptive control able to equip the robot with predictive and accurate performances.

REFERENCES

[1] N. Abdelkrim, K. Issam, K. Lyes, and C. Khaoula, "Fuzzy logic controllers for Mobile robot navigation in unknown environment using Kinect sensor," in Systems, Signals and Image Processing (IWSSIP), 2014 International Conference on, 2014, pp. 75-78.

[2] S. Z. S. Al-khayyt, "Comparison between fuzzy logic based controllers for robot manipulator trajectory tracking," in Engineering Sciences (FNCES), 2012 First National Conference for, 2012, pp. 1-6.

[3] E. Ayari, S. Hadouaj, and K. Ghedira, "A Fuzzy Logic Method for Autonomous Robot Navigation in Dynamic and Uncertain Environment Composed with Complex Traps," in Computing in the Global Information Technology (ICCGI), 2010 Fifth International Multi-Conference on, 2010, pp. 18-23.

[4] K. Benbouabdallah and Q.-d. Zhu, "Design of a fuzzy logic controller for a mobile robot tracking a moving target," in Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on, 2012, pp. 634-638.

[5] A. Bindal, S. Jit, and A. Simpson, "The design of autonomous mobile predator and prey robots," in Computational Intelligence and Multimedia Applications, 2005. Sixth International Conference on, 2005, pp. 232-239.

[6] D. Chi-Tinh, P. Hoang-The, P. Thanh-Binh, and T. Nguyen-Vu, "Vision based ground object tracking using AR.Drone quadrotor," in Control, Automation and Information Sciences (ICCAIS), 2013 International Conference on, 2013, pp. 146-151.

[7] B. Earl, "Pixy Pet Robot - Color vision follower," 26 August 2014.

[8] D. Elayaraja and S. Ramabalan, "Fuzzy logic based motion control of mobile robot in a rough terrain," in Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on, 2012, pp. 36-40.

[9] S. A. L. El-Teleity, Z. B. Nossair, H. M. A. K. Mansour, and A. TagElDein, "Fuzzy logic control of an autonomous mobile robot," in Methods and Models in

Automation and Robotics (MMAR), 2011 16th International Conference on, 2011, pp. 188-193.

[10] T. Emter, T. Emter, A. Saltolu, and J. Petereit, "Multi-Sensor Fusion for Localization of a Mobile Robot in Outdoor Environments," in Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK), 2010, pp. 1-6.

[11] U. Farooq, H. Arif, M. Amar, B. M. Tahir, M. Qasim, and N. Ahmad, "Comparative analysis of fuzzy logic and neural network based controllers for line tracking task in mobile robots," in Signal Processing Systems (ICSPS), 2010 2nd International Conference on, 2010, pp. V2-733-V2-739.

[12] U. Farooq, K. M. Hasan, M. U. Asad, and S. O. Saleh, "Fuzzy logic based wall tracking controller for mobile robot navigation," in Industrial Electronics and Applications (ICIEA), 2012 7th IEEE Conference on, 2012, pp. 2102-2105.

[13] M. Ghiasvand and K. Alipour, "Formation control of wheeled mobile robots based on fuzzy logic and system dynamics," in Fuzzy Systems (IFSC), 2013 13th Iranian Conference on, 2013, pp. 1-6.

[14] J. Ghommam, H. Mehrjerdi, and M. Saad, "Leader-follower formation control of nonholonomic robots with fuzzy logic based approach for obstacle avoidance," in Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, 2011, pp. 2340-2345.

[15] T. T. Hoang, P. M. Duong, N. T. T. Van, D. A. Viet, and T. Q. Vinh, "Multi-sensor perceptual system for mobile robot and sensor fusion-based localization," in Control, Automation and Information Sciences (ICCAIS), 2012 International Conference on, 2012, pp. 259-264.

[16] M. Hyeun Jeong, A. Drenner, and N. Papanikolopoulos, "Vision-based leader-follower formations with limited information," in Robotics and Automation, 2009. ICRA '09. IEEE International Conference on, 2009, pp. 351-356.

[17] Y. Kang, B.-W. Liu, K. Huang, M. Sheng, and Y. Liu, "Design of a real-time tracking robot based on simplified binocular positioning model," in Automatic Control and Artificial Intelligence (ACAI 2012), International Conference on, 2012, pp. 811-816.

[18] G. K. Kloss, S. Heesang, and N. H. Reyes, "Dynamic colour adaptation for colour object tracking," in Image and Vision Computing New Zealand, 2009. IVCNZ '09. 24th International Conference, 2009, pp. 340-345.

[19] M. F. R. Lee and F. H. S. Chiu, "A fuzzy logic navigated service mobile robot," in Fuzzy Theory and Its Applications (iFUZZY), 2013 International Conference on, 2013, pp. 183-188.

[20] S. B. Marapane, M. Holder, and M. M. Trivedi, "Coordinating motion of cooperative mobile robots through visual observation," in Systems, Man, and Cybernetics, 1994. Humans, Information and Technology., 1994 IEEE International Conference on, 1994, pp. 2260-2265 vol.3.

[21] J. Martin, Rapid Application Development: Macmillan, 1991.

[22] E. M. Mellouli, S. Sefriti, and I. Boumhidi, "Combined fuzzy logic and sliding mode approach's for modelling and control of the two link robot," in Complex Systems (ICCS), 2012 International Conference on, 2012, pp. 1-6.

[23] A. Mohagheghi, F. Shabaninia, and M. Salimifard, "Fuzzy logic & fuzzy sliding mode tracking control of non-holonomic unicycle wheeled mobile robots," in Electrical Engineering (ICEE), 2013 21st Iranian Conference on, 2013, pp. 1-6.

[24] R. M. Nor, A. Suhaib, K. S. Talha, N. Hassan, K. Wan, D. Hazry, et al., "The effects of membership function of the input and output fuzzy logic controller in a mobile robot's straight line navigation," in Electronic Design (ICED), 2014 2nd International Conference on, 2014, pp. 47-52.

[25] L. E. Parker, B. Kannan, T. Fang, and M. Bailey, "Tightly-coupled navigation assistance in heterogeneous multi-robot teams," in Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, 2004, pp. 1016-1022 vol.1.

[26] H. Poonawala, A. C. Satici, N. Gans, and M. W. Spong, "Formation control of wheeled robots with vision-based position measurement," in American Control Conference (ACC), 2012, 2012, pp. 3173-3178.

[27] M. P. N. Rathnayake, K. J. Samarasekera, S. C. Rajapakshe, E. W. M. P. W.

Wikramasinghe, H. B. M. U. L. Mahagedara, M. G. C. R. Samaranayake, et al., "Design and implementation of autonomous robot with solid object identification algorithm," in Industrial and Information Systems (ICIIS), 2010 International Conference on, 2010, pp. 632-637.

[28] A. Rowe, C. Rosenberg, and I. Nourbakhsh, "A Second Generation Low Cost Embedded Color Vision System," in Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on, 2005, pp. 136-136.

[29] T. Said, S. Ghoniemy, and O. Karam, "Real-time multi-object detection and tracking for autonomous robots in uncontrolled environments," in Computer Engineering & Systems (ICCES), 2012 Seventh International Conference on, 2012, pp. 67-72.

[30] M. S. Saidon, H. Desa, R. Nagarajan, and M. P. Paulraj, "Vision based tracking control of an autonomous mobile robot in an indoor environment," in Control and System Graduate Research Colloquium (ICSGRC), 2011 IEEE, 2011, pp. 1-6.

[31] O. K. Sayidmarie, M. O. Tokhi, A. M. Almeshal, and S. A. Agouri, "Design and real-time implementation of a fuzzy logic control system for a two-wheeled robot," in Methods and Models in Automation and Robotics (MMAR), 2012 17th International Conference on, 2012, pp. 569-572.

[32] L. Seung-Hwan, L. Tae-Seok, and L. Beom-Hee, "A sensor fusion system using enhanced extended Kalman filter with double fuzzy logics for autonomous robot guidance," in System Integration (SII), 2011 IEEE/SICE International Symposium on, 2011, pp. 579-584.

[33] M. Shayestegan and S. Din, "Fuzzy logic controller for robot navigation in an unknown environment," in Control System, Computing and Engineering (ICCSCE), 2013 IEEE International Conference on, 2013, pp. 69-73.

[34] A. Troppan, E. Guerreiro, F. Celiberti, G. Santos, A. Ahmad, and P. U. Lima, "Unknown-color spherical object detection and tracking," in Autonomous Robot Systems (Robotica), 2013 13th International Conference on, 2013, pp. 1-4.

[35] W.-J. Wang and C. Jun-Wei, "Implementation of a mobile robot for people

following," in System Science and Engineering (ICSSE), 2012 International Conference on, 2012, pp. 112-116.

[36] Q. Zhao and J. Hou, "Fuzzy logic system design for mobile robot motion control," in Instrumentation and Measurement, Sensor Network and Automation (IMSNA), 2013 2nd International Symposium on, 2013, pp. 631-633.

[37] H. Zhenlong, "Localization system of rescue robot based on multi-sensor fusion," in Computing, Control and Industrial Engineering (CCIE), 2011 IEEE 2nd International Conference on, 2011, pp. 178-181.