

**EMBEDDED SYSTEM FOR MOTION CONTROL  
OF OMNIDIRECTIONAL MOBILE ROBOT**

**BY  
MD. ABDULLAH AL MAMUN**

**JANUARY 2017**



**EMBEDDED SYSTEM FOR MOTION CONTROL OF  
OMNIDIRECTIONAL MOBILE ROBOT**

BY

**MD. ABDULLAH AL MAMUN**

A Thesis Presented to the  
DEANSHIP OF GRADUATE STUDIES

**KING FAHD UNIVERSITY OF PETROLEUM & MINERALS**

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the  
Requirements for the Degree of

**MASTER OF SCIENCE**

In

**COMPUTER ENGINEERING**

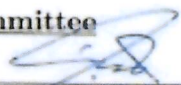
**JANUARY 2017**


KING FAHD UNIVERSITY OF PETROLEUM & MINERALS  
DHAHRAN 31261, SAUDI ARABIA


DEANSHIP OF GRADUATE STUDIES


This thesis, written by MD. ABDULLAH AL MAMUN under the direction of his thesis adviser and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE IN COMPUTER ENGINEERING.

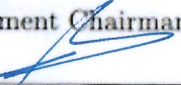
Thesis Committee

  
Dr. Ahmad Khayyat (Adviser)

  
Dr. Tarek Sheltani (Member)

  
Dr. Basem Almadani (Member)

  
Dr. Ahmad Almulhem  
Department Chairman

  
Dr. Salam A. Zummo  
Dean of Graduate Studies

29/1/17  
Date



© Md. Abdullah Al Mamun

2017



*To my Parents*  
*Every bit of me is a little bit of you*

# ACKNOWLEDGMENTS

*It is inevitable to extend my sincere gratitude and warmest affection to my family and friends who very graciously and voluntarily waived their rights on me to let me pursue this degree. I am and will always be heavily indebted to them for their continuous support, understanding and ever-needed prayers. Special credit is due to my thesis advisor Dr. Ahmad Khayyat and committee members Dr. Tarek R. Sheltami and Dr. Basem Almadani for their constant guidance and motivation right from the onset till the end of this research. Without their honest criticism and painstaking revisions, it would not have been possible to produce any quality work. Also, I want to thanks to Dr. Muhammad Mysorewala for useful guidelines. I am sincerely grateful to KFUPM for granting fully funded scholarship for my Master degree. The university and especially Department of Computer Engineering has been very kind and supportive in arranging necessary equipment and lab setup for the experiment. Also, I want to thanks to the team members of KFUPM RoboCup-SSL, specially Mohammad Nasir for benevolently providing the experimental support from their area of expertise in this project.*



# TABLE OF CONTENTS

<b>ACKNOWLEDGEMENT</b>	<b>v</b>
<b>LIST OF TABLES</b>	<b>ix</b>
<b>LIST OF FIGURES</b>	<b>x</b>
<b>ABSTRACT (ENGLISH)</b>	<b>xiii</b>
<b>ABSTRACT (ARABIC)</b>	<b>xv</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 Problem Statement . . . . .	3
1.2 Contribution . . . . .	5
1.3 Thesis Organization . . . . .	5
<b>CHAPTER 2 BACKGROUND AND LITERATURE REVIEW</b>	<b>7</b>
2.1 RoboCup Small Size League(SSL) . . . . .	8
2.2 Embedded System of RoboCup-SSL Robot . . . . .	11
2.3 Motion Control of Omnidirectional Robot . . . . .	18
<b>CHAPTER 3 EMBEDDED SYSTEM DESIGN</b>	<b>22</b>
3.1 Hardware Architecture . . . . .	24
3.2 Software Architecture . . . . .	26
3.3 Operation . . . . .	27

<b>CHAPTER 4</b>	<b>OMNIDIRECTIONAL MOTION CONTROL</b>	<b>28</b>
4.1	Robot Models . . . . .	29
4.1.1	Robot Kinematics . . . . .	29
4.1.2	Robot Dynamics . . . . .	32
4.1.3	Robot Model Linearization . . . . .	34
4.2	Controller Design . . . . .	35
4.2.1	Low-level Controller . . . . .	36
4.2.2	High-level Controller . . . . .	41
<b>CHAPTER 5</b>	<b>SIMULATION RESULTS</b>	<b>49</b>
5.1	Implementation . . . . .	49
5.2	Simulation Setup . . . . .	52
5.3	Results and Discussion . . . . .	52
<b>CHAPTER 6</b>	<b>EXPERIMENTAL RESULTS</b>	<b>66</b>
6.1	Robot Hardware Design . . . . .	66
6.1.1	Robot Mechanical Design . . . . .	68
6.1.2	Wheels and Gears . . . . .	68
6.1.3	Motors and Drivers . . . . .	69
6.1.4	Radio Devices . . . . .	71
6.2	Experimental Setup . . . . .	72
6.2.1	Pin Configuration and PCB . . . . .	72
6.2.2	Install Firmware . . . . .	73
6.2.3	Radio Message Format . . . . .	76
6.2.4	Experimental Procedures . . . . .	77
6.3	Results and Discussion . . . . .	78
<b>CHAPTER 7</b>	<b>CONCLUSION</b>	<b>81</b>
	<b>APPENDIX</b>	<b>83</b>
	<b>REFERENCES</b>	<b>85</b>





# LIST OF TABLES

4.1	Fuzzy rules for $K_P$ . . . . .	47
4.2	Fuzzy rules for $K_I$ . . . . .	47
5.1	Robot model parameters and values . . . . .	50
5.2	Error comparison with existing literature . . . . .	55
5.3	Comparison of velocity norm with existing literature for the two path planners . . . . .	65
6.1	Motor Data . . . . .	70
6.2	Pin configuration of motor driver. . . . .	71
6.3	The technical specification of XBee s2c zigbee. . . . .	71
6.4	Error comparison . . . . .	80



# LIST OF FIGURES

2.1	(Left) Physical drawings (Right) Wheel's position . . . . .	8
2.2	Small Size League match (source: RoboCup Federation) . . . . .	9
2.3	Vision system . . . . .	10
2.4	CMD Robot . . . . .	12
2.5	CMD Electronic Board . . . . .	13
2.6	Tigers Mannheim Electronics Board . . . . .	14
2.7	SKUBA main PCB board . . . . .	16
2.8	MRL main board . . . . .	16
3.1	System overview of RoboCup-SSL . . . . .	23
3.2	Hardware architecture . . . . .	25
3.3	Software architecture . . . . .	25
4.1	Local and global coordinates . . . . .	30
4.2	Comparison of linear and nonlinear model responses: Circular Trajectory . . . . .	35
4.3	Comparison of linear and nonlinear model responses: Rectangular Trajectory . . . . .	36
4.4	Hierarchical playing architecture [1] . . . . .	42
4.5	Configuration of fuzzy adaptive PI . . . . .	44
4.6	Membership function plotting of $E$ , $EC$ . . . . .	44
4.7	Membership function plotting of $\Delta K_P$ and $\Delta K_I$ . . . . .	45
5.1	Fuzzy-PI path planner and LQR controller schematic . . . . .	50

5.2	Optimal voltage (minimum) for motors on the rectangular trajectory for the PID and LQR controller . . . . .	51
5.3	The difference between tracked path and trajectory on a circular pathway by the LQR controller with robot velocity $V_{robot} = 1m/s$ $\phi = 0.05rad/s$ . . . . .	53
5.4	Robot position and orientation variation in rectangular trajectory	54
5.5	Surface area curve of fuzzy control logic . . . . .	55
5.6	The accuracy of rectangular path following for robots transitional speed (a) 3 m/s; (b) 4 m/s . . . . .	56
5.7	The horizontal and vertical deviations as the output of both PI-LQR and fuzzy-PI-LQR controller for the speed of 3 m/s . . . . .	57
5.8	The comparison of robot orientation error in both cases with the velocity (a) 3 m/s; (b) 4 m/s . . . . .	58
5.9	The positional error analysis for both PI-LQR and fuzzy-PI-LQR controller on a circular trajectory for robot velocity $V = 4m/s$ . . . . .	59
5.10	The velocity norm in both horizontal and vertical in 3 m/s . . . . .	60
5.11	The deviation from targeted trajectory in presence of randomly distributed disturbance on rectangular trajectory for the transitional speed of (a) $V = 2$ m/s; (b) $V = 4$ m/s . . . . .	62
5.12	Robot Position error analysis on the rectangular maneuver with external load with the velocity of 3m/s and 4m/s . . . . .	63
5.13	Circular Trajectory with $v = 4$ m/s . . . . .	64
5.14	DNA shape trajectory with $v = 2$ m/s . . . . .	64
5.15	Flower shape trajectory with $v = 2$ m/s . . . . .	65
6.1	The top view of KFUPM RoboCup-SSL robot . . . . .	67
6.2	The side view of KFUPM RoboCup-SSL robot . . . . .	67
6.3	Omni-wheels . . . . .	68
6.4	An EC 45 brush-less DC motor. . . . .	69
6.5	Motor driver. . . . .	70

6.6	A XBee s2c zigbee is communicating with another XBee on Arduino.	72
6.7	KFUPM SSL Robot . . . . .	73
6.8	Robot View . . . . .	73
6.9	Pin Configuration . . . . .	74
6.10	Flowchart of the embedded system firmware . . . . .	75
6.11	Message format . . . . .	76
6.12	Circular trajectory with max. allowable speed . . . . .	78
6.13	Position error comparison . . . . .	79
6.14	DNA trajectory with max. allowable speed . . . . .	79
6.15	Position error comparison . . . . .	80



# THESIS ABSTRACT

**NAME:** Md. Abdullah Al Mamun

**TITLE OF STUDY:** Embedded System for Motion Control of Omnidirectional  
Mobile Robot

**MAJOR FIELD:** Computer Engineering

**DATE OF DEGREE:** January 2017

In this thesis, an embedded system for motion control of omnidirectional mobile robots is presented. An omnidirectional mobile robot is a type of holonomic robots. It can move simultaneously and independently in translation and rotation. The RoboCup small-size league, a robotic soccer competition, is chosen as the research platform in this study. The first part of this research is to design and implement an embedded system that can communicate with a remote server using a wireless link, and execute received commands. Secondly, a fuzzy-tuned PI path planner and a related low-level controller are proposed to attain optimal input for driving a linear discrete dynamic model of the omnidirectional mobile robot. To fit the planning requisite and avoid slippage, velocity and acceleration filters are also employed. In particular, low-level optimal controllers, such as a

linear quadratic regulator (LQR), for multiple-input-multiple-output (MIMO) acceleration and deceleration of velocity is investigated, where an LQR controller is running on the robot with feedback from motor encoders or sensors. Simultaneously, a fuzzy adaptive PI is used as a high-level controller for position monitoring, where an appropriate vision system is used as a source of position feedback. A key contribution presented in this research is an improvement in the combined fuzzy-PI LQR controller over a traditional PI controller. Moreover, the efficiency of the proposed approach and PI controller are also discussed. Simulation and experimental evaluations are conducted with and without external disturbance. An optimal result to abate the variances between the target trajectory and the actual output is delivered by the on-board regulator controller in this study. Utilizing the new approach in trajectory-planning controller design results in more precise motion of four-wheeled omnidirectional mobile robots, and the modeling and experimental results confirm this claim.

## Thesis Abstract (Arabic)

### خلاصة الرسالة

الاسم: محمد عبدالله ال مأمون  
عنوان الرسالة: نظام مدمج للتحكم في حركة الروبوتات المتحركة في جميع الاتجاهات  
التخصص: هندسة الحاسب الآلي  
التاريخ: يناير ٢٠١٧

في هذه الأطروحة، نطرح نظاماً مدمجاً للتحكم في حركة الروبوتات المتحركة في كل الاتجاهات، والتي يمكنها الحركة عن طريق كل من الإزاحة والدوران.

يستخدم هذا البحث مسابقة روبوكب لكرة القدم للروبوتات الصغيرة (RoboCup-SSL) كمنصة لتطوير وتطبيق نتائجه. الجزء الأول من هذا البحث هو تصميم وتنفيذ نظام مدمج يمكنه التواصل مع خادم بعيد عبر وصلة لاسلكية، وتنفيذ الأوامر الواردة.

ثانياً، نقترح نظام تحكم تناسبي تكاملي يتم ضبطه بضابط ضبابي كمخطط مسار للحركة، بصحبة وحدة تحكم ذات مستوى منخفض، للحصول على مدخلات أمثل لتشغيل نموذج ديناميكي خطي متقطع للروبوت المتحرك في جميع الاتجاهات. لتتناسب مع ما يلزم من تخطيط وتجنب الزلل، تم كذلك استخدام مرشحات للسرعة والتسارع.

على وجه الخصوص، تمت دراسة استخدام وحدات التحكم الأمثل ذات المستوى المنخفض، مثل المنظم الخطي من الدرجة الثانية (LQR)، للتسارع والتباطؤ متعدد المدخلات والمخرجات (MIMO)، حيث يتم تشغيل وحدة التحكم في الروبوت في وجود تغذية مرتدة صادرة من مشفرات المحركات أو أجهزة الاستشعار.

في الوقت ذاته، يتم استخدام وحدة تحكم تناسبية تكاملية (PI) متكيفة تلقائياً بشكل ضبابي كوحدة تحكم عالية المستوى لمراقبة موقع الروبوت، حيث يتم استخدام نظام رؤية مناسب كمصدر للتغذية المرتدة. إن إحدى أهم مساهمات هذا البحث هي تحسين أداء وحدة تحكم المنظم الخطي من الدرجة الثانية المدمجة مع التحكم التناسبي التكاملي الضبابي (LQR Fuzzy-PI) مقارنة بوحدة تحكم تناسبي تكاملي تفاضلي (PID) تقليدية.

وعلاوة على ذلك، يناقش هذا البحث أيضاً كفاءة كل من النهج المقترح ووحدة التحكم التناسبي التكاملي التفاضلي (PID)، كما يشمل محاكاة النهج المقترح وتقييمه تجريبياً مع وبدون اضطراب خارجي.

تبين من خلال النتائج الحصول على نتيجة مثلى للحد من الفروق بين المسار الهدف والمسار الفعلي عند استخدام وحدة التحكم المدمجة المقترحة في هذه الدراسة. استخدام النهج الجديد في تصميم وحدة تحكم لتخطيط مسار الروبوت ينتج عنه تحكم أكثر دقة في حركة الروبوتات المتحركة رباعية العجلات، والنمذجة والنتائج التجريبية تؤكد هذه النتائج.

### درجة الماجستير

جامعة الملك فهد للبترول والمعادن

الظهران- المملكة العربية السعودية

## CHAPTER 1

# INTRODUCTION

An embedded system is a small computer system with a dedicated function inside a bigger electrical or mechanical system, often with real-time processing constraints [2]. In general, an embedded system is a combination of hardware and software that performs a specific task. The purpose of embedded systems is to control a device, a process or a larger system. It controls many devices in common use today [3]. Nowadays, embedded microcontroller accounts for more than 98% of all produced microprocessors, thus it has massively greater computing power in the IT industry [4] [5]. Firmware is used to program an embedded system unlike software which is meant to be used for interaction, productivity and activity like word processing, video editing. It is stored in chip flash memory.

In this research, the RoboCup Small Size League (RoboCup-SSL) is used as a concrete target application. The RoboCup-SSL competition is an active research platform for both students and researcher [6]. More details about the RoboCup-SSL is presented in the section 2.1. Two major challenges of this competition are



building a good robot and playing strategy. A robot is a combination of mechanical design, electronic circuit, a real-time embedded system and artificial intelligence algorithm. Although the robot of RoboCup-SSL are small in size, it needs to implement multiple function such as motion control, wireless communication and kicking & dribbling the ball.

In the future, mobile robot has many appropriate potentials in human society. The roles of the robot will no longer be limited to complete tasks in assembly and manufacturing at a secure position. A mobile robot has to be navigated efficiently in the real world in order to achieve hands-on jobs in which unpredictable variations take place.

Conventional wheeled mobile robot (WMR) unable to run sideways without an initial maneuvering causes constraint in their motion though several mechanisms have been advanced to progress the maneuverability of WMR but they have not yet overcome the issues of the non-holonomic system. For instance, there are a couple of motors mounted in static positions on the left and right side of the robot in a differential drive design. However, a kind of deficiency still exists in the differential wheel drive. This robot is called ‘non-holonomic’ because it cannot drive in all possible direction. On the contrary, using omnidirectional wheels, a holonomic robot, is capable of driving to any direction at any point of time. Due to having much more ambiguity than conventional WMR, the trajectory control of WMR is very hard and the dynamic equation is nonlinear. The objective of this research is to design and implement an embedded system for a specific robotic ap-

plication: The RoboCup-SSL soccer competition for robots. Also, ensure accurate motion control and path planning using a low-level and a high-level controllers. The low-level controller for controlling the wheel's speed and high-level controller is for controlling the position of the robot. A discrete time Linear Quadratic Regulator (LQR) controller is used as the low-level controller and a Fuzzy Control Logic (FLC) adaptive PI is used as the high-level controller.

Motion control for driving omni-wheels is stored on the robot. When team server decide to move a robot to a target location, it sends a message containing few vector data (e.g. angles and velocity) to radio device through a communication protocol. Now, the radio device (e.g. XBee) forward this message to the microcontroller for calculating the speed, direction and angular velocity of each wheel and send PWM to the motor driver that produce required analog signal to rotate motor accordingly. A feedback signal is maintained to monitor the accuracy. Similarly, Battery and sensor control module take care of power and sensor operations.

## **1.1 Problem Statement**

When a team thinks about creating their robot for the RoboCup-SSL competition, they may start with on hand open-source firmware of other teams rather than starting from scratch that can save enormous time. However, hardware matching is the main barrier in this method because; typically the firmware is written by the programmers for specific target hardware. So, the new team needs to change

a much amount of code to support the new hardware that is almost parallel time consuming to build a firmware from scratch. To deal with this problem, we are going to design and implement an embedded system that is easily adaptable.

However, It is a common fact that every RoboCup-SSL robots are driven by battery, so power consumption is one of the key concerns. Therefore, we need to find an energy efficient low-level controller that can produce optimal input for motor drivers with accurate wheel velocity.

Similarly, the special feature of the omnidirectional robot is it can move and rotate to any direction concomitantly and autonomously in the plane. Two main key responsibilities of motion control of a omnidirectional robot are the path planning and accurate trajectory following. Though there is many research has been done on the theoretical model implementation of kinematic and dynamics of omnidirectional robot with proper mathematical model, some traditional controller still have drawback of localization when the target path and surface condition is different.

A standard motion controller has some requirements. Similarly, there are some requirements, the proposed motion controller need to meet. According to the existing literature and game requirements, the data rate of wireless communication is 60 packets/sec is required. The controller stabilization time should not exceed 10 sec. and the tracking error should be less than 15%.

## 1.2 Contribution

The achievements from this thesis are listed below.

- Platform independent embedded system architecture for motion control of omnidirectional mobile robots. Tested on Arduino and mbed.
- Accurate and efficient motion control of four wheeled omnidirectional mobile robot.
- Wheel control: PI and LQR, with feedback from motor encoders.
- Position control: A fuzzy adaptive PI controller with vision feedback.
- Simulation and experimentation are conducted.
- A comparison study has been done with existing literature.

## 1.3 Thesis Organization

The foundation of the research area and the problem statement is presented in the first chapter. Chapter 2 will mention the backgrounds and fundamental concepts. Also, some of the most promising existing works in this field to show the big picture and assert the significance of this study will describe in this chapter. In addition, the relevance of our work in comparison with existing literature will briefly explain in this chapter. In chapter 3, the approach to design and implementation of the embedded system for controlling the motion of omnidirectional mobile robot will be discussed in depth. Chapter 4 will explain the robot model and omnidirectional



drive control using controller. Chapter 5.3 presents simulation setup and controller schematic. A detailed discussion and analysis of the result will be shown later in this chapter. Similarly, chapter 6 shows the experimental setup and performance comparison. This chapter will also describe the robot sub-systems. Finally chapter 7 will conclude this research and discuss future directions.

## CHAPTER 2

# BACKGROUND AND LITERATURE REVIEW

A simple embedded system is implemented for this thesis to control the motion of an omnidirectional mobile robot considering all required robot hardware design. The majority of teams made their robots using omnidirectional drive and used to compete on the SSL nowadays [7] [8]. This drive allows the robot to move to any point on the field without rotating its whole body on the way to the target quickly. To get rid of this, a special wheel called omnidirectional wheel is assembled with a motor that offers friction-less movement to any direction independently. The secondary small wheels are placed perpendicular to the main wheel. A geometrical drawing and a mechanical design of the omnidirectional robot using the omni-wheels, are shown in Figure 2.1 [9].

To make an omnidirectional drive robot, it needs more than two wheels. Nearly every team has a robot with four wheels. The main causes behind the use of more

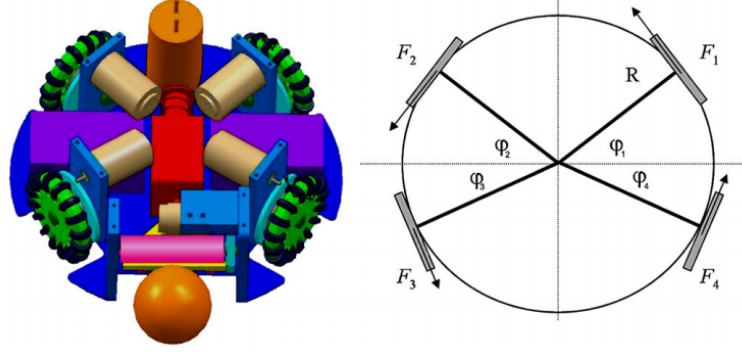


Figure 2.1: (Left) Physical drawings (Right) Wheel's position

than three wheels:

1. More torque comes from more motors, consequently, more acceleration, which is a major target on an SSL to cross the field quickly.
2. To ensure the robot speed is fast enough at least when it is driving forward.

## 2.1 RoboCup Small Size League(SSL)

RoboCup is an annual international robotic soccer competition that was founded in 1997. It was proposed by several researchers as a world-wide friendly football tournament [10]. RoboCup includes four different types of leagues: Standard platform league, small size league, middle size league and simulation league [11]. The competition of small robots, also known as small size league is one of the popular competitions. It is a robotic football tournament, which makes a rich research platform for areas of complex artificial intelligence algorithm, efficient hardware and firmware, computer vision and multiple distributed agent co-operation.

The core objectives of this project can be realized from this motivational and

inspirational sentence: “By 2050, a team of fully autonomous humanoid robot soccer players shall win a soccer competition, complying with the official FIFA rules, against the winner of the most recent World Cup of Human Soccer” [12]. The SSL competition is based on the FIFA rules [13]. Alike multi-robot automation task in systems assembly [14,15], collection of group works [16], and multi-robot space mission [17], the RoboCup-SSL is a multi-robot team work to achieve co-operative goal. In this competition, every team plays next to the opponent having six robots that is shown in Figure 2.2. There are few overhead cameras that are mounted on each side of the field. The cameras are sending captured images to an image processing unit that extracts the location of every robot and ball on the field and send it to each team server that is connected to a network cable. In common, constructing team requires intelligent playing strategy, execution and integration of software and hardware sub-components into a fully functional unit.



Figure 2.2: Small Size League match (source: RoboCup Federation)

The RoboCup Small Size League, or F180 as it is otherwise known, focuses on the issues of artificial intelligence, embedded system, control theory in highly hostile circumstances with a distributed or centralized system [18]. The size of each robot must be with the dimensions given in the F180 rules: The robot ought



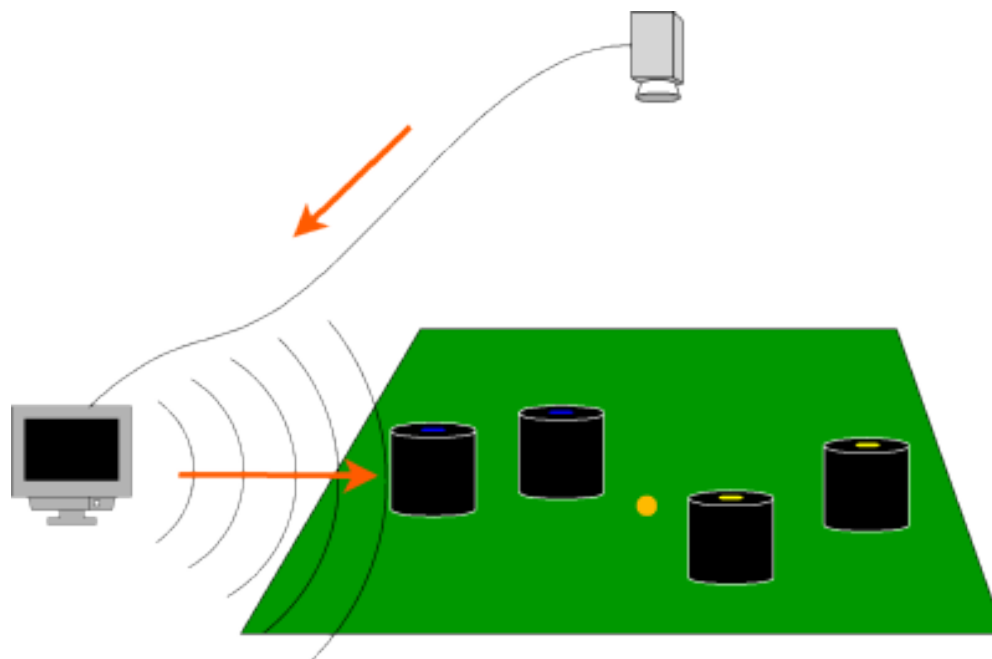


Figure 2.3: Vision system

to in shape within an 180mm diameter circle and should be no higher than 15cm. The robot plays on a green carpeted field that is  $(9m \times 6m)$  with an orange golf ball. In the vision system, few overhead cameras are used to identify the movement of robots in the field. The overhead camera is placed at 4m above the surface. A camera feedback can be processed either on robot or it sends back to the team server. All referee commands come from an off field PC. For wireless communication, a commercial RF receiver/transmitter is used. Few miscellaneous rules are given below

- The robots must not communicate with each other but only the server.
- Robots are identified by color patches on top of their shell.

## 2.2 Embedded System of RoboCup-SSL Robot

As the hardware and software combination is called embedded system, there are couple of hardware and software modules required to build a robot for RoboCup-SSL. First of all, there are four brushless DC motors connected to the omni-wheels to drive the robot. The motors can be driven by PWM or Analog or DAC signals generated from the microcontroller attached to the motor drivers. There is a radio device used to receive commands from the team server and sending feedback. A general overview of the embedded system design and implementation is given in chapter 3.

Approximately thirty teams participated in RoboCup-SSL in 2015 from different countries across the world [18]. Each team must publish their Team Description Paper (TDP). However, few teams among them published detail descriptions about their robot hardware and firmware. In this section, the investigation outcomes of the robot hardware and firmware of referenced teams are described.

### **CMDragons**

Every robot of CMDragons has four omnidirectional wheels. The wheels are run by 30-watt brushless motors [19] [20]. Each motor has a reflective quadrature encoder for accurate wheel travel and speed estimation. For velocity evaluation, every motor has a deep quadrature encoder. The kicker is a big diameter custom wound solenoid that gets better durability. The kicker mechanism and robot drive system are illustrated in Figure 2.4. The main control part of the robot electronics

is an ARMv7 core which is a group of three older 32-bit ARM processor cores running at 58MHz connected to a Xilinx Spartan2 FPGA. The main electronic board of the CMDragons is shown in Figure 2.5. The ARMv7 core is used for communication, Proportional Integral Derivative (PID) controlling estimation, and monitoring. The FPGA acts as a coordinator among the PWM generator, the quadrature decoders and the serial communication with additional on-board components.

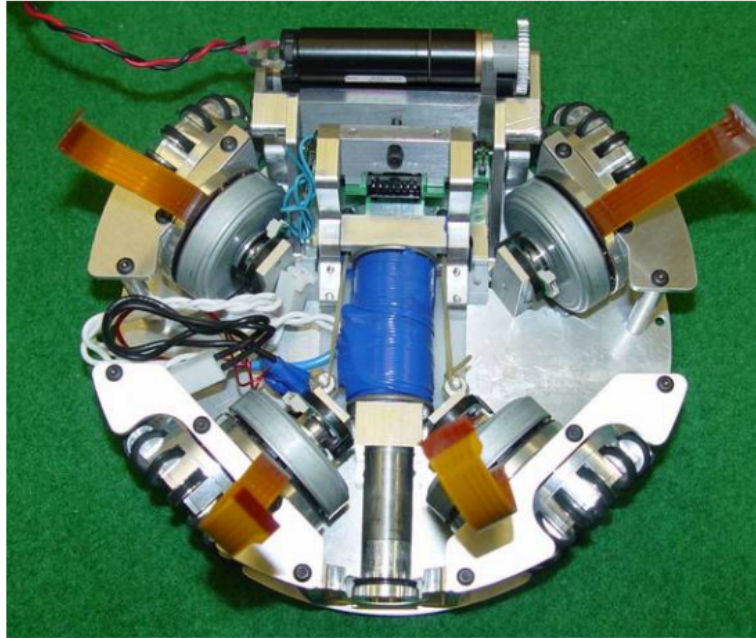


Figure 2.4: CMD Robot

For enhanced performance, the robot also gets feedback from a gyroscope. It was an open question by other teams: Why CMD uses the FPGA? Here is the answer: One new quality of CMD is the regular optimization of motion profile parameters. An easy trapezoidal motion profile that is slope the velocity curve to create predictable acceleration and deceleration rates. It is maintained to control the robot movement behavior which are specified by three different constants such

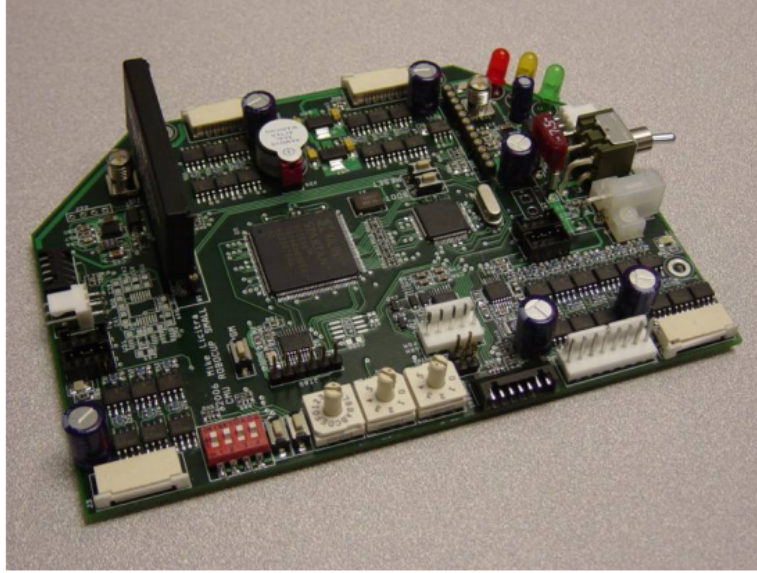


Figure 2.5: CMD Electronic Board

as acceleration, deceleration, and speed. They overcome this problem by using a programmed robot behavior which offers to find the best set of motion profile parameters. Their algorithm tried to solve this trade-off between the overall robot velocity and motion instability and this algorithm is run on FPGA.

### **Tigers Mannheim**

Tigers Mannheims omnidirectional drive of the robot has four wheels which are run by Maxon 30W EC motors [21]. The torque is generated with a gear ratio of approx. 3.33. As a result, high acceleration is achieved by optimizing drive and the speed is approximately 3 m/s when driving forward. Their robot has 3 electronic boards help to solve three different problems. The most significant board is the main board, a 140x120mm 4-layer PCB that is shown in Figure 2.6. Team Tigers Mannheim used a Cortex-M3 32-bit microcontroller as the main processor. It is from STmicroelectronics (STM32F103ZE) clocked at 72 MHz.

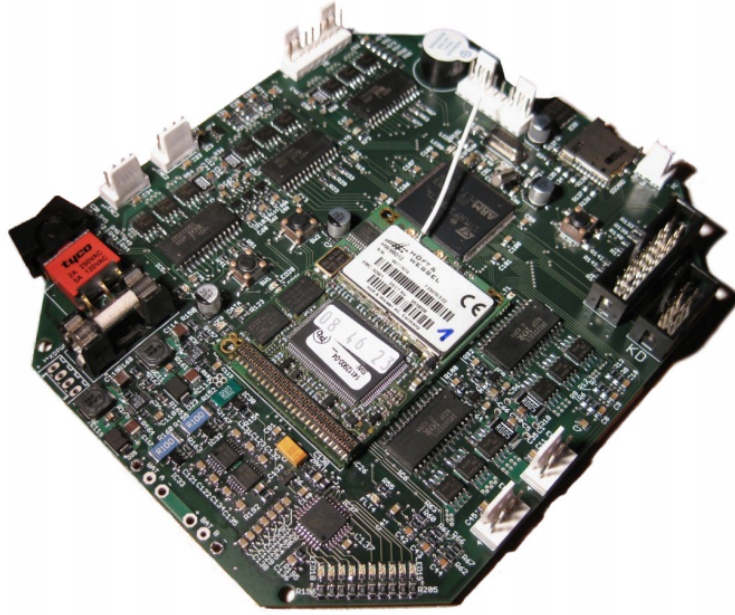


Figure 2.6: Tigers Mannheim Electronics Board

This processor is able to control all motors, perform wireless communication, and read the encoders. In order to requirements of hardware interrupts, they used the FreeRTOS real-time operating system [22]. An embedded Digital Enhanced Cordless Telecommunications (DECT) module from Hft & Wessel (HW86012) is used to do the bidirectional communication. The module has a frequency of 1.9 GHz to transfer unprocessed radio frames to the microcontroller. The main board uses two 8-bit ATmega microcontrollers as port expanders and Analog to Digital (A/D) converters. An SPI bus is used for communication between the microcontroller units. An ATmega microcontroller is used to control the main board. To charge and discharge transistors, PWM outputs are used. Also, the controller monitors heat sink temperatures as well as the capacitor voltage.

## SKUBA

A low level torque controller reduces the effect of surfaces [13]. The main board and the chipper and kicker board are major components of the electronic board. The main board executes all robotic tasks whereas the kicker board controls the kicking mechanism. A Xilinx Spartan-3 XC3S400 FPGA, a combination of motor driver, a debugging port and some add-on modules are used. To control the robot kinematics such as angular velocity and position, four brushless motors are used. These motors are controlled by the microprocessor core (RISC-32 architecture). 30W Maxon EC45 flat brushless motors are used in the robot for both driving the robot and kicking the ball. A feedback mechanism is built by using hall sensors that measure wheel velocity and send the signal back to the controller. Similarly, a speedy 15W Maxon EC16 motor is connected with the dribbler circuit through the encoder. However, the PWM signal generator - a specific series of MosFETs - is a must to drive all BLDC motors in the SKUBA main PCB, which is shown in Figure 2.7.

To control the driving sequence, a digital sequencer FPGA module is used. There are two big problems with the driving system which can destroy the whole robot: over-current and motor dead-time. In the motor data-sheet, the dead-time value is given. Dead-time is a blanking time period (upper & lower transistors in off-state simultaneously) of half-bridge power stage. The dead-time protection is built in the FPGA to prevent over current whereas a fully Integrated, hall-effect-based linear current sensor IC (ACS712) calculates the motor driving current to



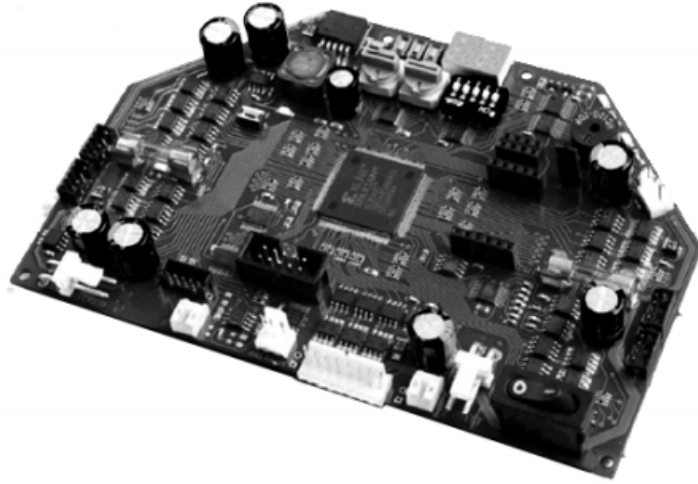


Figure 2.7: SKUBA main PCB board

prevent circuits. If the current exceeds the limit, the firmware will keep the PWM signals less than the motor's maximum range. An ultra low power 2.4GHz RF module (Nordic-nRF24L01+) is used as a radio communication device. In every robot, two radio devices are used for two way communication because the robot's current status is essential for the motion control, for instance the ball location.

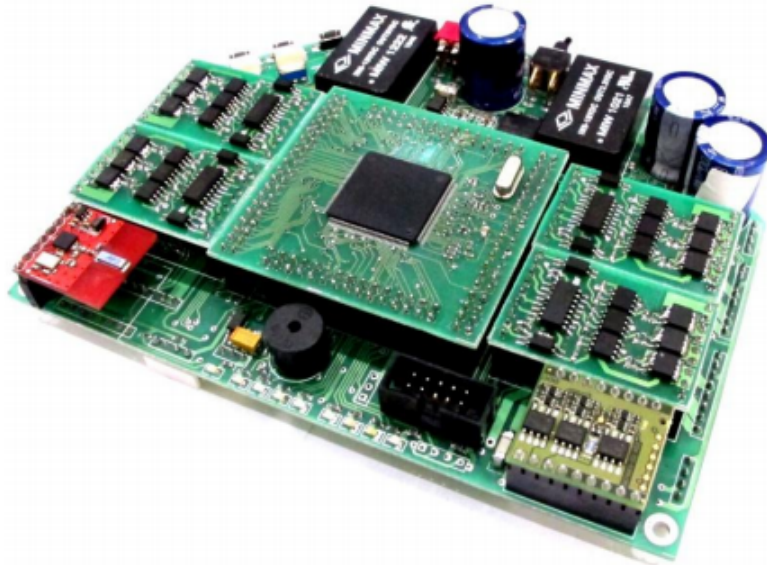


Figure 2.8: MRL main board

## MRL

The robot of MRL has a main board and different modules connected to it [23]. It includes Processor Daughter Board (PDB), motor driver modules, and solenoid driver circuit. A wireless board is also designed to send and receive data between robot and team server. The main board of the MRL robot is shown in Figure 2.8. The PDB consists of one FPGA (Altera Cyclone - EP1C6T144C8) and one ARM processor (NXP LPC2378) connected to each other.

The duty of the FPGA is to control the motors whereas the ARM processor is responsible for controlling the FPGA, wireless communication, compute control algorithms, debug the entire system, and log the data. In other words, the FPGA sends the encoder data whereas the ARM microcontroller prepares PWM data for FPGA to drive the motors. The team MRL used ARM7-TDMI core and developed the project in KEIL software. The ARM7 microcontroller was selected for several reasons such as its powerful debugging capabilities and low-power architecture. Only real-time tasks such as motor driving are executed on the FPGA, while all the remaining parts are implemented on the ARM7 microcontroller. The communication between robots and team server is done by using two radio module namely nRF2401 transceivers. These modules work in frequency between 2.4 to 2.525 GHz. The robot has 4 brush-less DC motors (BLDC) to perform precise motions. The BLDC motors are EC45 50 watts MAXON flat motor. The ARM7 microprocessor software architecture is designed by two main interrupts: Wireless Interrupts occurs when a new packet is received by wireless module whereas con-

trol Interrupt occurs when a PI control function runs for each wheel which gets its desired speed from wireless packet and its feedback speed from FPGA.

## 2.3 Motion Control of Omnidirectional Robot

In robotic applications where the environment is dynamic for plane trajectory following, speed and path following accuracy of omnidirectional robots are key motives for extensive study of such robots [24, 25]. Dynamic and kinematic equations with consistent control are core objectives in numerous studies of these robots. However, classic controllers suffer from a few weaknesses that prevents them from reaching the target location when the path has extensive variations of twists.

Omnidirectional robot dynamics and kinematics are discussed in a number of studies, where the robot dynamics model is approximated to a linear system [26–29]. More precisely, two autonomous PID controllers are built for controlling the direction and the position. However, a nonlinear relationship among the translational and rotational velocity has not yet investigated.

In the same way, the trajectory follower robot is made where a geometric path is given and robot tried to follow the exact same path using feedback; this classic problem of omnidirectional mobile robot is addressed on this article [30].

Creating a symmetrical path and using feedback to track the trajectory. The dynamic capabilities of the mobile robots are measured and applied to swiftly embryonic environments [31]. Moreover, a control approach and optimized maneuver planing for robot position control without considering orientation has es-

established [32, 33].

A continuous and nonlinear model is demonstrated where an applied technique to filter speed and dynamics to achieve a sleep-free (without slippage of the wheels) drive are presented [34].

To the contrary, acceleration control is an important approach to get an exact maneuver contains the resolve deceleration control of PD and PI [35]. The mentioned model is principally focused on feedback control that is performed for controlling the velocity as well as the orientation of the robot; Moreover, they conducted an experimental studies.

In addition, the sequential dynamics of an omnidirectional robot carried a lot of research studies of optimal controllers. Nonetheless the initial section of the current research emphasis on the advancement of a discrete time linear quadratic regulating method as the low-level controller with the combination of fuzzy logic as a high-level controller. Both are executed for trajectory following. This style offers optimal voltage to motor drivers as discuss in section 5.3.

In recent year, linear quadratic regulator controller is used to control many sophisticated robotic applications. The stochastic extended linear quadratic regulator (SELQR), an innovative optimization-based motion planner, that estimates a path and corresponding linear control rules aim to minimize the value of cost function [36]. A motion uncertain model of state-based nonlinear dynamics using a Gaussian distribution is applied in SELQR. In every iteration, to evaluate the cost-to-come and the cost-to-go for every state along a trajectory, SELQR

uses a combination of forward and backward value. Since SELQR optimizes each state locally along the trajectory in every iteration to minimize the anticipated total cost, dynamics linearization and cost function quadratization. Thus, SELQR gradually estimates the total cost.

Another example of an LQR controller, a dynamic-model control system to balance the speed of a unicycle robot is described in this research [37]. A sliding-mode controller with LQR is used to ensure the stability of speed tracking measurement. To minimize the switching function, a sigmoid function based mode controller is implemented in the roll controller. To follow the exact trajectory in real-time drive, an LQR controller has been introduced.

In biological systems investigation, a novel solution of the inverse LQR problem is provided [38]. In continuous- and discrete-time cases, a number of methods are followed to get a cost function for the LQR problem in this paper. Their approach is to find the optimal solution for  $K$  based on  $Q$  and  $R$  called inverse LQR problem. They get motivation from motion goals in biological system. When  $Q$  and  $R$  are unknown, they have created an efficient linear matrix inequality (LMI) to determine the solution for similar problems.

Apart from the above, an auto-adaptation of a PID controller with fuzzy logic for omnidirectional mobile robots is studied [33]. In order to keep the ideal performance for extremely nonlinear models, a fuzzy logic controller (FLC) rule assignment for tuning a PID controller parameter is discussed [39]. This FLC consist of two inputs recognized as error and its derivative, and one output, known as the

velocity control signal. This is a Mamdani type FLC, implemented for the speed task of the path planner in which the center of gravity method is used to perform defuzzification [40].

Based on fuzzy sets and defuzzification, fuzzy PID controllers were formerly established which is used in velocity control as discussed [41, 42]. The segmented membership functions that facilitates of perception as well as cognition to enhance the data processing efficiently. Additionally, the segmented members in the calculation of FLC is offerings more robust procedures for intelligent schemes [43, 44].

## CHAPTER 3

# EMBEDDED SYSTEM DESIGN

The architecture of a RoboCup small size league system consists of four main components: a single common vision system, a team server for each team, a single referee box and a number of remotely-controlled robots for each team [45]. The vision system consists of overhead cameras connected to a vision processing application that analyzes the captured images to extract location coordinates of the ball and all the robots in the field. The team server runs artificial intelligence software that controls the robots over wireless links.

The robot includes mechanical parts and electronic circuits controlled by an embedded system. The embedded system works as a mediator between the robot electronics and the team server. It receives and interprets commands sent by the server and drives the robot accordingly. Devices controlled by the embedded system include motors, sensors, and wireless modules. A brief overview of the RoboCup-SSL system is illustrated in the Figure 3.1. The hardware components with their interfaces are discussed in this chapter. The software architecture is

also described in this chapter.

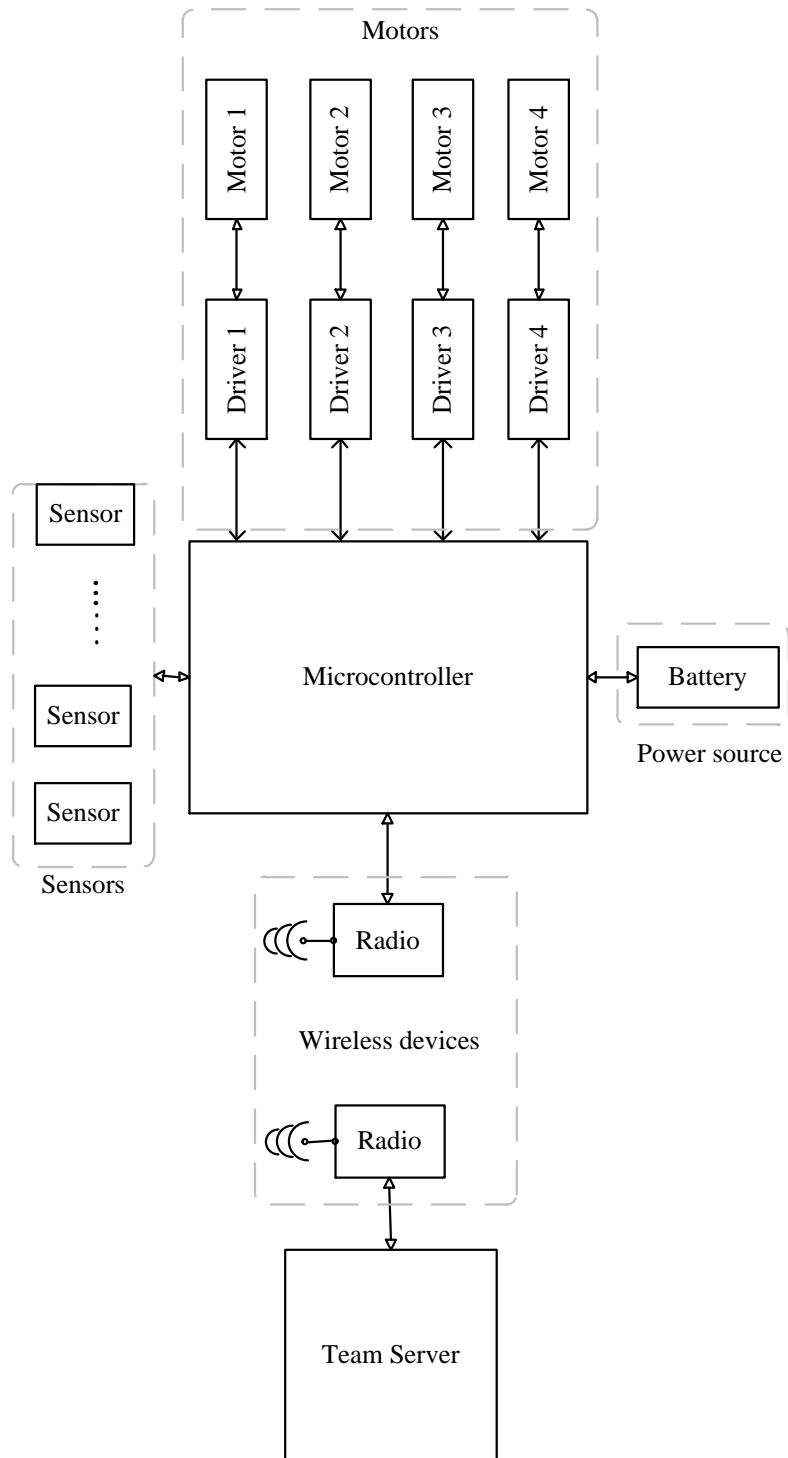


Figure 3.1: System overview of RoboCup-SSL



### 3.1 Hardware Architecture

A typical RoboCup-SSL robot is made of hardware such as a microprocessor, a radio device, sensors, motors, wheels and gears, battery, a dribbler and a kicker. The dribbler is a special motorized mechanical device that maintains ball possession and release, whereas the kicker is used to kick the ball. As our main target of this study is to implement an embedded system for accurate motion control of omnidirectional mobile robot, the dribbler and kicker devices are not discussed in this research.

Example, commonly used sensors include gyroscopes, accelerometers, infrared sensors. The gyroscope and accelerometer are used to measure the instant robot velocity as a feedback source for motion control whereas the infrared sensors are used for ball detection and obstacle avoidance. The radio devices are used for wireless communications. For each robot, the essential mechanical parts are the mechanical structure, four omni-wheels with gears, and four brushless DC motors. The hardware architecture of the embedded system is shown in Figure 3.1.

ZigBee wireless modules can be used as radio devices where the ZigBee PRO 2007 protocol is used between multiple ZigBee modules. A serial communication exists between microcontroller and the ZigBee modules. The sensors can be connected with the microcontroller using an SPI, I2C, or UART communication hardware interfaces. The motors can be driven through motor drivers using PWM, ADC, or analog signals generated by the microcontroller. An appropriate battery is required as a power source for the devices. A brief hardware architecture

is shown in the Figure 3.2 where the microcontroller is connected to four motor drivers, DC power source, sensors, and a wireless radio module.

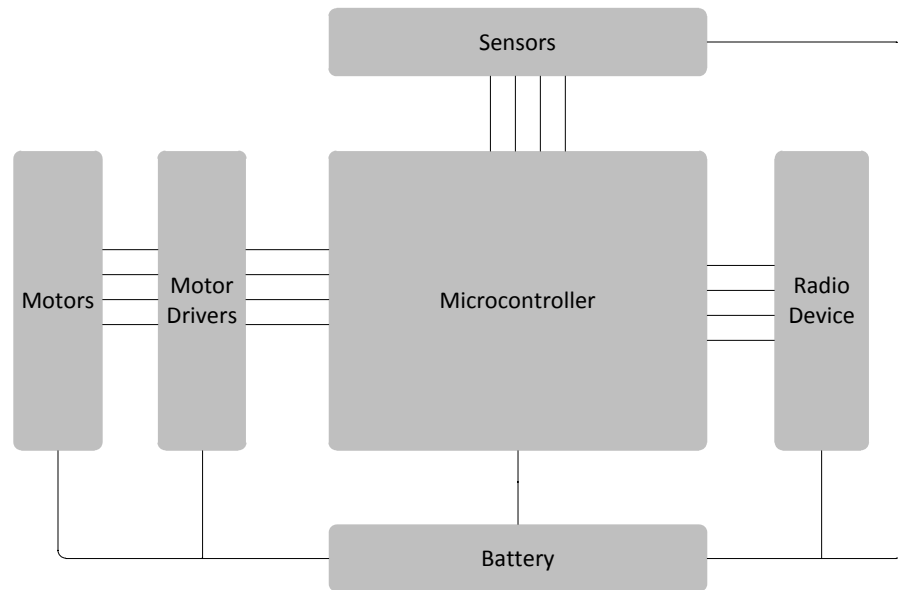


Figure 3.2: Hardware architecture

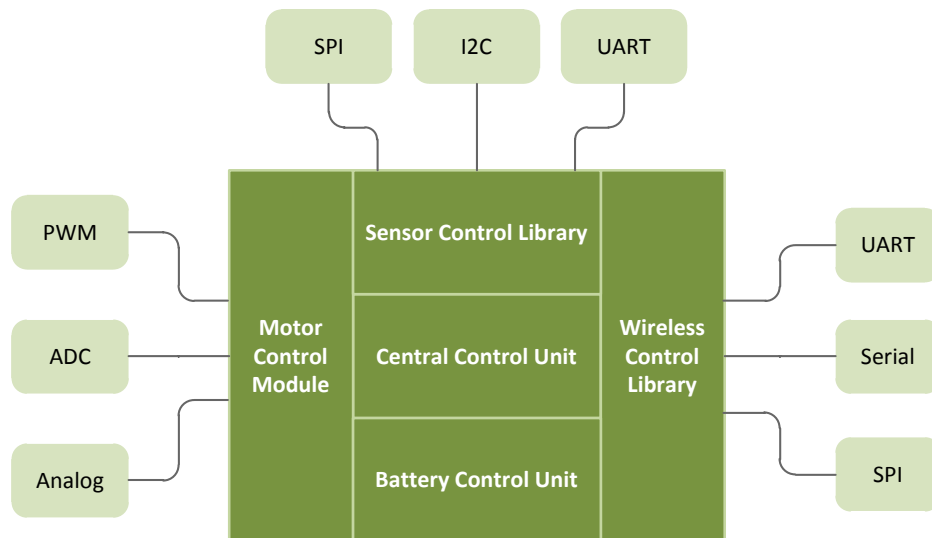


Figure 3.3: Software architecture

## 3.2 Software Architecture

The software architecture of the embedded system is shown in Figure 3.3. The firmware consists of motor, sensor, wireless, and battery control modules, which execute on the microcontroller. They are written in a system programming language such as C. A brief description of the software modules is given below.

**Central Control** The main function is in the central control unit. The main function calls the required function from the related library and pass the parameters to accomplish a define task.

**Sensor Control** There is a sensor library to process the sensor data. The sensors are connected using any of the hardware connection interfaces to the microcontroller. Similarly, the main controller program communicates with sensor library using some user defined function with floating parameters. For example, *readgyro()* function reads three values x, y, z and pass them to the main controller unit.

**Wireless Control** A wireless control library acts as a driver for the wireless devices to the microcontroller. The radio devices can be connected with microcontroller using various hardware interfaces like UART, serial or SPI. The main role of the wireless library is to communicate, control the data flow and buffer management.

**Battery Control** A battery or series of batteries are used as a power source. The main role of battery module are to supply power in a constant rate and

protect the main electric circuit board from over current.

**Motor Control** The most important part of the firmware is the motor control module. After receiving the commands from a wireless link, it decomposes the speed for each wheels. This module calculates a range of floating point data (from 0 to 255) of required PWM, DAC, or analog signal. When the microcontroller passes to the motor drivers, the microcontroller generates the required voltage to drive the motors with a targeted speed. Also, this module implements a PI or an LQR controller as the low-level motor controller where it gets the feedback from motor encoders or sensors.

### 3.3 Operation

When the team server decides to move a robot to a target location, it sends a message containing few vector data (e.g. target location, speed, and direction) to the radio device through a high-level communication protocol such as ZigBee PRO. Now, radio device (e.g. ZigBee) forwards this message to the microcontroller for calculating speed, direction and angular velocity of each wheel. The microcontroller sends required signals to the motor drivers that produce required analog signals according to the target velocity of the robot to rotate the motors. A feedback signal generated by motor encoder or sensors is used to monitor the speed accuracy.

# CHAPTER 4

## OMNIDIRECTIONAL MOTION

## CONTROL

Motion control of an omnidirectional mobile robot is a challenging task because its kinematics and dynamics are more complex compared to a traditional two wheeled mobile robot. The omnidirectional robot can move in any direction without rotating its body. Considering this, we developed kinematics and dynamics models. Two commonly used controllers for controlling the wheels angular velocity are Proportional Differential (PD) controllers and Proportional Integral (PI) controllers. PI controllers are much easier to build. However, they do not ensure the reliability if the field friction has varieties [1]. Therefore, we propose using a more advanced controller, namely a *linear quadratic regulator* (LQR) controller, to control the wheel's speed as a low-level controller that runs locally on the robot. Also, a *fuzzy tuned PI* controller is used as a high-level controller to control the position of the mobile robot remotely. The high-level controller determines the

robot's velocity and acceleration. The LQR -low-level controller- reads actual motor speed using motor encoders and uses it as a feedback source. On the other hand, the vision system is used as a feedback source to the high-level fuzzy tuned PI controller.

The objective of these controllers is to minimize robot position error. The position error is the distance between the target point and actual point. In this chapter, mathematical models of robot kinematics and dynamics are described. The relationship between the linear low-level controller (LQR) and the nonlinear robot model is explained. Finally, the design and development of the low-level controller and the high-level controller are discussed.

## **4.1 Robot Models**

The kinematics and dynamics models of the RoboCup-SSL robot are described in this section. The robot is driven by four motors, and has a cylindrical shape. In the robot models, two different frames are used: the robot body, denoted by  $b$ , and the global frame, denoted by  $g$ . The frame is a location coordinate vector in an x-y plane. When the robot starts moving, the body frame never changes with respect to its center. Similarly, the global frame is also fixed on the play ground.

### **4.1.1 Robot Kinematics**

An accurate kinematic model is a prerequisite to performing an accurate trajectory following task using a motion controller. The following sections discuss the

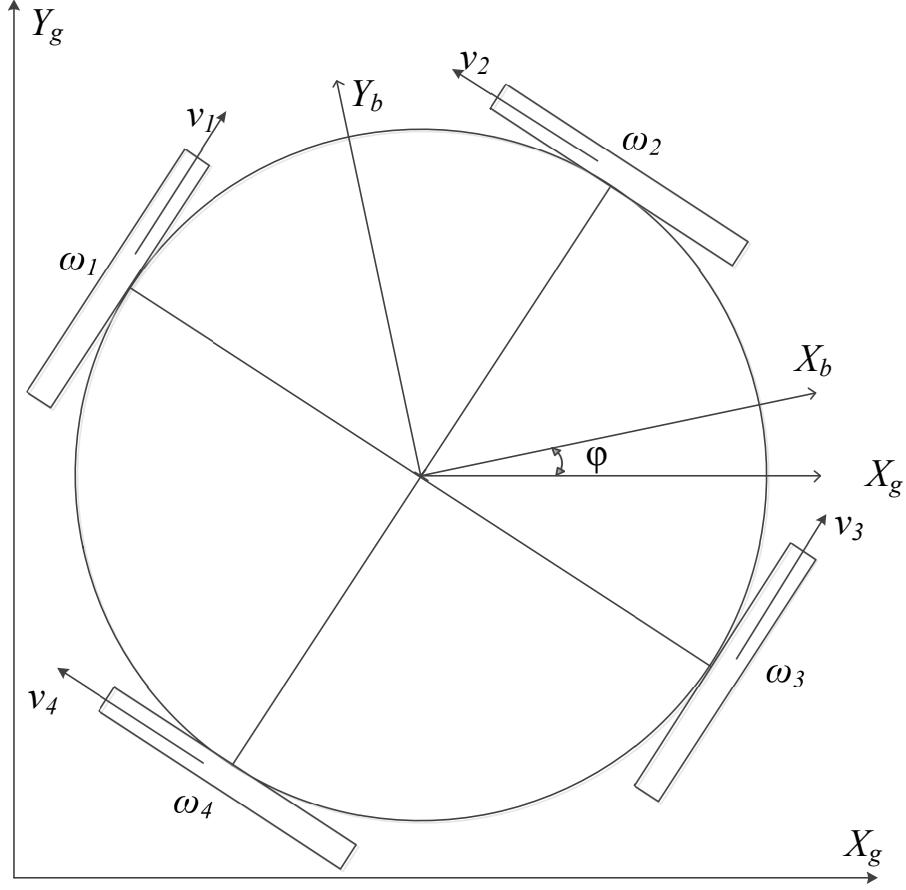


Figure 4.1: Local and global coordinates

relationship between different geometrical specifications and robot speed with basic kinematic formulas. The global and body coordinates of the robot position and orientation are illustrated in Figure 4.1.

$$X_b = \begin{bmatrix} x_b & y_b & \phi \end{bmatrix}^T \quad (4.1)$$

$$X_g = \begin{bmatrix} x_g & y_g & \phi \end{bmatrix}^T \quad (4.2)$$

The rotation matrix  ${}^gR_b$ , which is used to change the coordinates from the

body frame to the global frame, is given by:

$${}^g_bR = \begin{bmatrix} \cos \phi & -\sin \phi & 0 \\ \sin \phi & -\cos \phi & 0 \\ 0 & -0 & 1 \end{bmatrix} \quad (4.3)$$

By taking the time derivation of the robot position, the velocity vector can be stated as shown in Eq 4.4. The velocity vector is related to the angular velocity and the wheel geometry in global coordinates.

$$\dot{X}_g = {}^g_bR \dot{X}_b = {}^g_bR (G^T)^{-1} r \omega_L \quad (4.4)$$

Here, the geometrical matrix G is

$$G = \begin{bmatrix} -\sin \theta_1 & -\sin \theta_2 & -\sin \theta_3 & -\sin \theta_4 \\ \cos \theta_1 & \cos \theta_2 & \cos \theta_3 & \cos \theta_4 \\ l & l & l & l \end{bmatrix} \quad (4.5)$$

The mathematical description of the robot's acceleration vector is derived by taking the time derivative of Eq. (4.4), which results in:

$$\ddot{X}_g = {}^g_b\dot{R} (G^T)^{-1} r \omega_L + {}^g_bR (G^T)^{-1} r \dot{\omega}_L \quad (4.6)$$

The robot's gear ratio, robot geometry, and its matrix elements are used to estimate its angular velocity.



### 4.1.2 Robot Dynamics

In this section, the robot dynamics - the relationship between the motor driver's torque, actuating voltage, angular velocity, and angular acceleration - are described. Moreover, the required dynamic equations of the four-wheeled omnidirectional robots are given. There are some nonlinearities. For example, motor dynamic constraints may seriously disturb the robots performance particularly, during accelerating and deceleration [46]. So, a model is proposed considering the nonlinear parameters that are important in controller design. The relationship matrix between traction force and applied force is given by

$$F_b = GF_t \quad (4.7)$$

The traction force matrix of the robot is given by

$$F_t = \begin{bmatrix} f_1 & f_2 & f_3 & f_4 \end{bmatrix}^T \quad (4.8)$$

Here,  $f_i$  is the wheel's traction force values. The moment matrix and the functional forces can be stated as

$$F_b = \begin{bmatrix} F_{bx} & F_{by} & T_z \end{bmatrix}^T \quad (4.9)$$

The dynamic equation of the traction force of the robot is estimated by Eq. (4.10) which is derived from Eq. (4.6) and Eq. (4.7).

$$F_t = G^{-1} {}^g_b R^T m \left[ {}^g_b \dot{R}(G^T)^{-1} r \omega_L + {}^g_b R(G^T)^{-1} r \dot{\omega}_L \right]^T \quad (4.10)$$

Considering entire load and the rotational inertia of the motor, the required torque of the driver is:

$$\begin{aligned} \tau_m = & \left[ (J_m + \frac{J_L}{n^2}) I_{4 \times 4} + \frac{r^2}{n^2} G^{-1} {}^g_b R^T m {}^g_b R (G^T)^{-1} \right] \dot{\omega}_m \\ & + \left[ (c_m + \frac{c_L}{n^2}) I_{4 \times 4} + \frac{r^2}{n^2} G^{-1} {}^g_b R^T m {}^g_b \dot{R} (G^T)^{-1} \right] \omega_m \end{aligned} \quad (4.11)$$

Eq. (4.12) can be simplified as follows; the matrix elements of Z and V are listed in Appendix B.

$$\tau_m = Z \dot{\omega}_m + V \omega_m \quad (4.12)$$

Matrices Z and V are given in Eqs. (4.13) and (4.14)

$$Z = \begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_2 & k_1 & k_4 & k_3 \\ k_3 & k_4 & k_5 & k_6 \\ k_4 & k_3 & k_6 & k_5 \end{bmatrix} \quad (4.13)$$

$$Z = \begin{bmatrix} k_7 & -k_8\dot{\phi} & k_9\dot{\phi} & k_{10}\dot{\phi} \\ k_8\dot{\phi} & k_7 & -k_{10}\dot{\phi} & -k_9\dot{\phi} \\ -k_9\dot{\phi} & k_{10}\dot{\phi} & k_7 & -k_{11}\dot{\phi} \\ -k_{10}\dot{\phi} & k_9\dot{\phi} & k_{11}\dot{\phi} & k_7 \end{bmatrix} \quad (4.14)$$

To get an overall visualization of the torque control method, driver's dynamic behavior, driver's actuating voltage, back EMF, terminal resistance, and the motor torque constant are required. The relationship between them is given by

$$\tau_m = (E - EM.\omega_m) \frac{k_m}{R_t} \quad (4.15)$$

Combining Eq. (4.12) with (4.15) results in the following coupled nonlinear motion equation:

$$E = \left(\frac{R_t}{k_m}Z\right)\dot{\omega}_m + \left(\frac{R_t}{k_m}V + EM.I_{4 \times 4}\right)\omega_m \quad (4.16)$$

Eq. (4.16) illustrates the relationship between the dynamic behavior and the driver.

### 4.1.3 Robot Model Linearization

The previous sections demonstrate that the omnidirectional mobile robot is a nonlinear system. These nonlinearities are visible during concurrent linear and rotational paths. So, it is practical to estimate a linear model that establishes the system motion equations. Figure 4.2 and Figure 4.3 conclude a judgment

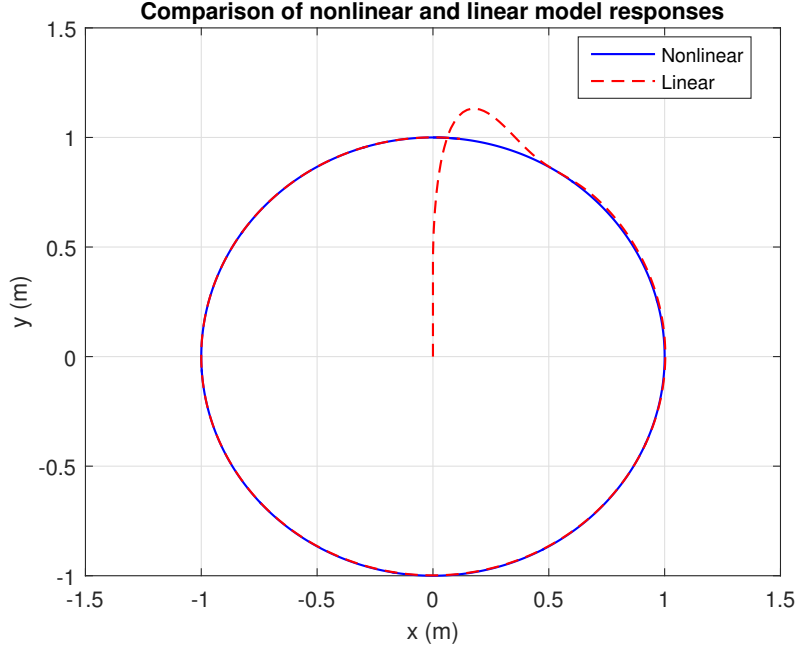


Figure 4.2: Comparison of linear and nonlinear model responses: Circular Trajectory

between the linear and nonlinear performance in a trajectory following task with time variation in a circular and rectangular path respectively. Since the linearized model offers more satisfactory outcome with a maximum deviation of 6.7% in ensuring the correct track than the nonlinear method which is shown in Figure 4.2 and Figure 4.3 respectively. It has a minor effect on robots current positioning in the field with approximately 1.5% error.

## 4.2 Controller Design

To achieve an accurate and efficient drive, two closed-loop controllers are employed in the proposed motion control approach. A discrete time linear quadratic regulator (LQR), low-level controller estimates the optimal input for the motors.

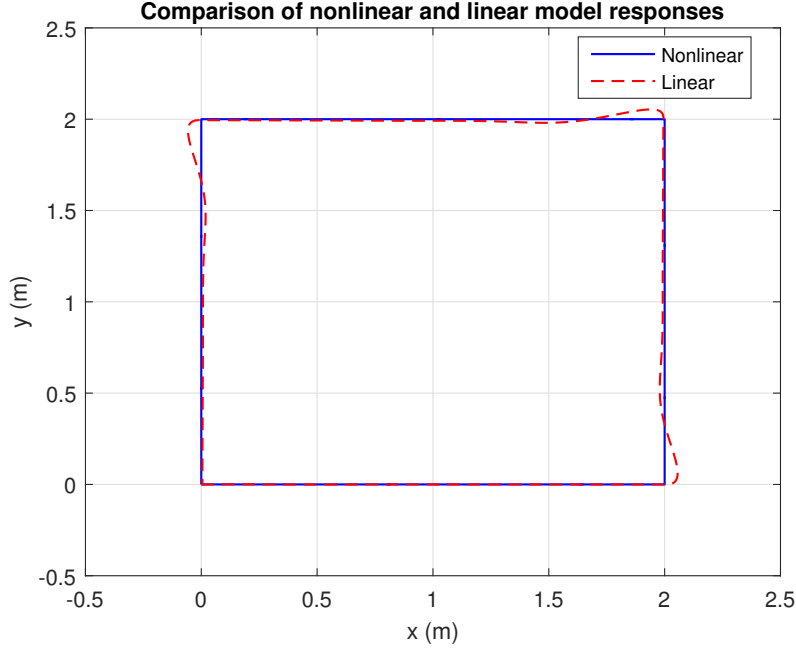


Figure 4.3: Comparison of linear and nonlinear model responses: Rectangular Trajectory

It runs on the robots main board. A fuzzy adaptive PI controller to calculate the right pathway runs in the team server. A detailed description of these two controllers is given below.

#### 4.2.1 Low-level Controller

Optimal performance with low-energy is a current hot topic for mobile robot [47]. An optimal control system is a system which estimates maximum return for a minimum cost [48]. Two leading methods were used in a related study: tracking control and regulator control [49]. Tracking control mainly focuses on a defined state time history in a way to optimize a specified performance index, whereas regulator control returns the system to the equilibrium state [50]. Furthermore, the regulator control approach delivers an ideal solution to minimize the distance

between the targeted path and the actual trajectory [51]. This paragraph explains the characteristics of a discrete time linear quadratic regulator controller that will be applied in this study to follow the reference path for the omnidirectional mobile robot [52].

The following state equation is used to describe the time-invariant and linear system

$$x(k+1) = Ax(k) + Bu(k) \quad (4.17)$$

with the output

$$y(k) = Cx(k) \quad (4.18)$$

The performance index that needs to be minimize is shown in Eq. (4.19), presenting the relationship between the control variables and parameters [53].

$$\begin{aligned} J = & \frac{1}{2} [Cx(k_f) - z(k_f)]^T F [Cx(k_f) - z(k_f)] \\ & + \frac{1}{2} \sum_{k=k_0}^{k_f-1} \{ [Cx(k) - z(k)]^T Q [Cx(k) - z(k)] \\ & + u^T(k) R u(k) \} \end{aligned} \quad (4.19)$$

Where an  $n$ -order and an  $r$ -order state, a control, and an output vector are presented as  $x(k)$ ,  $u(k)$  and  $y(k)$  respectively. Moreover, the positive semi-definite symmetric matrices  $F$  and  $Q$  that are of size  $n \times n$ , and the positive definite

symmetric matrix  $R$  that are of size  $r \times r$ . The free condition,  $x(k_f)$  with the fixed condition,  $x(k_0)$  is given to minimize the error  $e(k) = y(k)z(k)$  where  $z(k)$  is an  $n$ -dimensional target matrix. The formulation of Hamiltonian - an optimal control theory - is following as

$$\begin{aligned}
& H(x(k), u(k), \lambda(k+1)) \\
& = \frac{1}{2} \sum_{k=k_0}^{k_f-1} \{ [Cx(k) - z(k)]^T Q [Cx(k) - z(k)] \\
& \quad + u^T(k)Ru(k) \} \lambda^T(k+1)[Ax(k) + Bu(k)]
\end{aligned} \tag{4.20}$$

Where,  $x\star$  and  $\lambda\star$  are used in the state and the costate equations. All related calculations come from the Hamiltonian equations, as expressed in Eq. (4.21) to Eq. (4.24), where the star indicates the optimized parameter.

$$\frac{\delta H}{\delta \lambda \star(k+1)} = x \star(k+1) \tag{4.21}$$

$$x \star(k+1) = Ax \star(k) + Bu \star(k) \tag{4.22}$$

The costate equations from Hamiltonian are stated as

$$\frac{\delta H}{\delta x \star(k)} = \lambda \star(k) \tag{4.23}$$

$$\lambda \star (k) = A^T \lambda \star (k+1) + Vx \star (k) - Wz(k) \quad (4.24)$$

The final condition is

$$\lambda(k_f) = C^T F C x(k_f) - C^T F z(k_f) \quad (4.25)$$

Thus, we can get the optimal voltage for four motor drivers by minimizing the Hamiltonian considering the control input in open loop.

$$\frac{\delta H}{\delta u \star (k)} = 0 \quad (4.26)$$

$$u \star (k) = -R^{-1} B^T \lambda \star (k+1) \quad (4.27)$$

A canonical system is derived from costate systems as

$$\begin{bmatrix} x \star (k+1) \\ \lambda \star (k) \end{bmatrix} = \begin{bmatrix} A - E \\ V A^T \end{bmatrix} \begin{bmatrix} x \star (k) \\ \lambda \star (k+1) \end{bmatrix} + \begin{bmatrix} 0 \\ -W \end{bmatrix} z(k) \quad (4.28)$$

Here,  $E = B R^{-1} B^T$ ,  $V = C T Q C$  and  $W = C T Q$ . By eliminating the costate variable.

$$\lambda \star (k) = P(k) x \star (k) - g(k) \quad (4.29)$$

After deducing the costate  $\lambda \star (k)$  results the  $P(k)$  and  $g(k)$  stated as



$$P(k) = A^T P(k+1) [I + EP(k+1)]^{-1} A + V \quad (4.30)$$

$$g(k) = \{A^T - A^T P(k+1) [I + EP(k+1)]^{-1} E\} g(k+1) + Wz(k) \quad (4.31)$$

Terminator constraints are also given in Eqs. (31) and (32).

$$P(k_f) = C^T F C \quad (4.32)$$

$$g(k_f) = C^T F z(k_f) \quad (4.33)$$

The nonlinear Difference Riccati Equation (DRE) is given in Eq. (4.30) to get an acceptable solution of backwards considering the target constraint (4.32). Also, the linear vector difference Eq. (4.31) solves backwards problem using the final condition (4.33). These outcomes are achieved offline to be implemented in closed loop control method, expressed as:

$$u \star (k) = -L(k)x \star (k) + L_g(k)g(k+1) \quad (4.34)$$

The gains  $L_g(k)$ -forward and  $L(k)$ -back, are stated in Eqs. (4.35) and (4.36).

$$L(k) = [R + B^T P(k+1)B]^{-1} B^T P(k+1)A \quad (4.35)$$

$$L(g) = [R + B^T P(k+1)B]^{-1} B^T \quad (4.36)$$

As a result, the optimized path is stated as Eq. (4.37)

$$x \star (k+1) = [A - BL(k)]x(k) + BL_g(k)g(k+1) \quad (4.37)$$

The earlier closed-loop controller can follow the target path accurately, but in the presence of disturbance, there is a deviation between the desired and actual path. To tune the PI that can quickly stabilize the erratic system with a constant disturbance, a Fuzzy Logic Control (FLC) is used.

### 4.2.2 High-level Controller

In the preceding paragraphs, a general picture of linearization and a consistent routine for velocity profile determination are discussed. The LQR approach is used without a high-level controller such as the FLC. However, this section discusses how the fuzzy path planner and controller schemes are implemented as a high-level position controller.

For localization support, a vision system consisting of a number of fixed cameras mounted over the soccer field and an image processing system is continuously communicates with the team server. A typical vision system of RoboCup-SSL is shown in Figure 4.4. The robots and the server connected using wireless links. However, if the current position of the robot is not the target position, the distance function calculates the gap between the target and actual position. In this

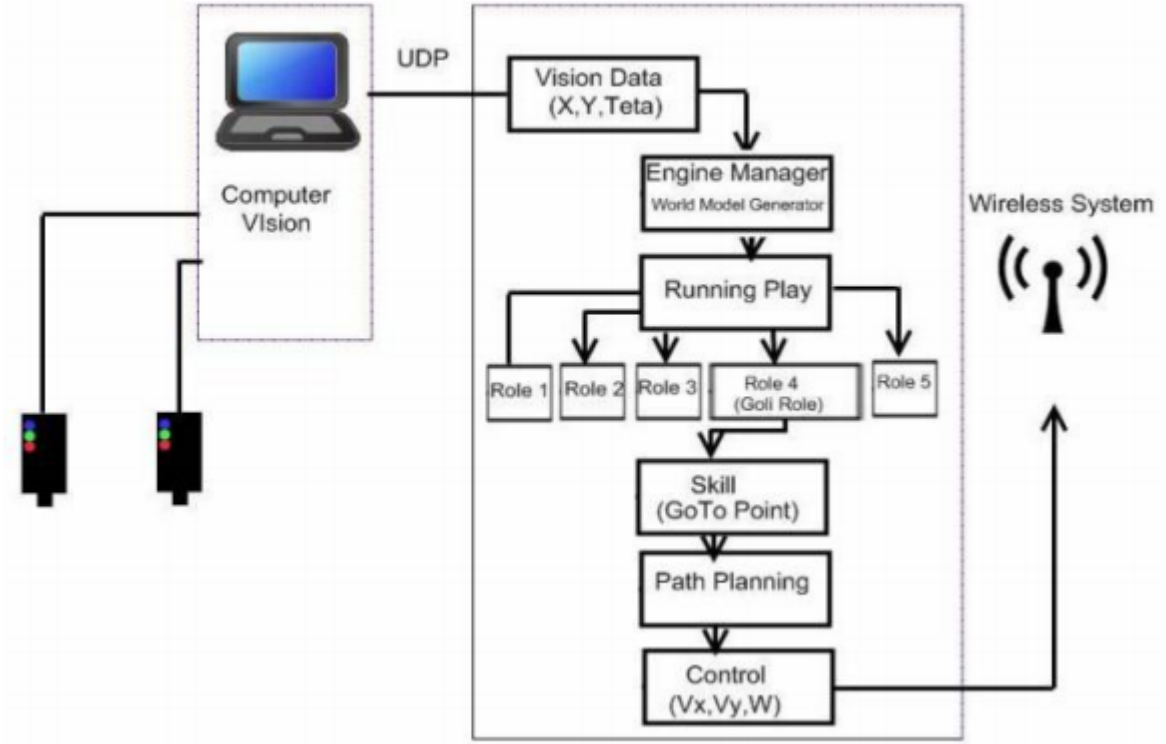


Figure 4.4: Hierarchical playing architecture [1]

study, a PI controller is used to control the position of the omnidirectional mobile robot which is tuned by an FLC [54]. As a result of applying linearized PI with FLC, the position errors decreases enormously, which is described mathematically in chapter 5.3. Currently, two major enhancements are gained by using a fuzzy adaptive PI path planner engaging the LQR. Firstly, the employment of the FLC to the PI parameter allocation process improves the trajectory following performance. The position and travel time errors decreases pointedly. Secondly, acceleration and deceleration take place more smoothly.

To achieve a better performance using the PI controller, selecting a suitable value for the P and I coefficients of the PI controller is important, and can be done automatically by using an FLC. However, FLC can be adjusted by either

regulating the input-output gains of the control system or changing the position of linked membership functions like [55, 56].

### **The PI Controller**

Only proportional and integral gains are considered in this study, which makes a PI controller. PI is a well-known and conventional control system commonly used in many complex systems because it is easy to implement. The stable time -the time it takes to stabilize- can be minimized by tuning the values of the  $P$  and  $I$  coefficients denoted by  $K_P$  and  $K_I$ . The mathematical linear relationship between the output  $u(t)$  and the error  $e(t)$  can be expressed by the following equation:

$$u(t) = K_P e(t) + K_I \int e(t) dt \quad (4.38)$$

where the proportional and integral gains are denoted by  $K_P$  and  $K_I$ . One of the common approaches to tune the values of the P and I coefficients is the Ziegler-Nichols method [57]. In this research, the constraints of the PI controller are set to  $K_P = 10$ ,  $K_I = 0.5$ .

### **Fuzzy Adaptive PI Controller**

The fuzzy control logic unit has of two inputs: the error,  $e$  and the change in the error,  $ec$ . The outputs  $\Delta K_P$  and  $\Delta K_I$  denote the changes of the proportional gain  $K_P$  and the integral gain  $K_I$ , respectively. The three key elements of the fuzzy adaptive PI: fuzzification, fuzzy inference and defuzzification, are illustrated in

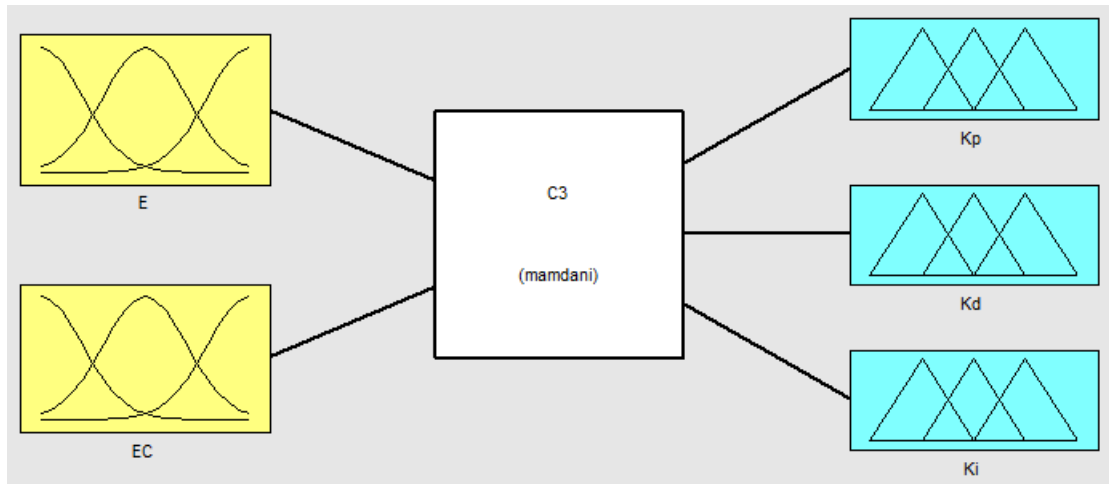


Figure 4.5: Configuration of fuzzy adaptive PI

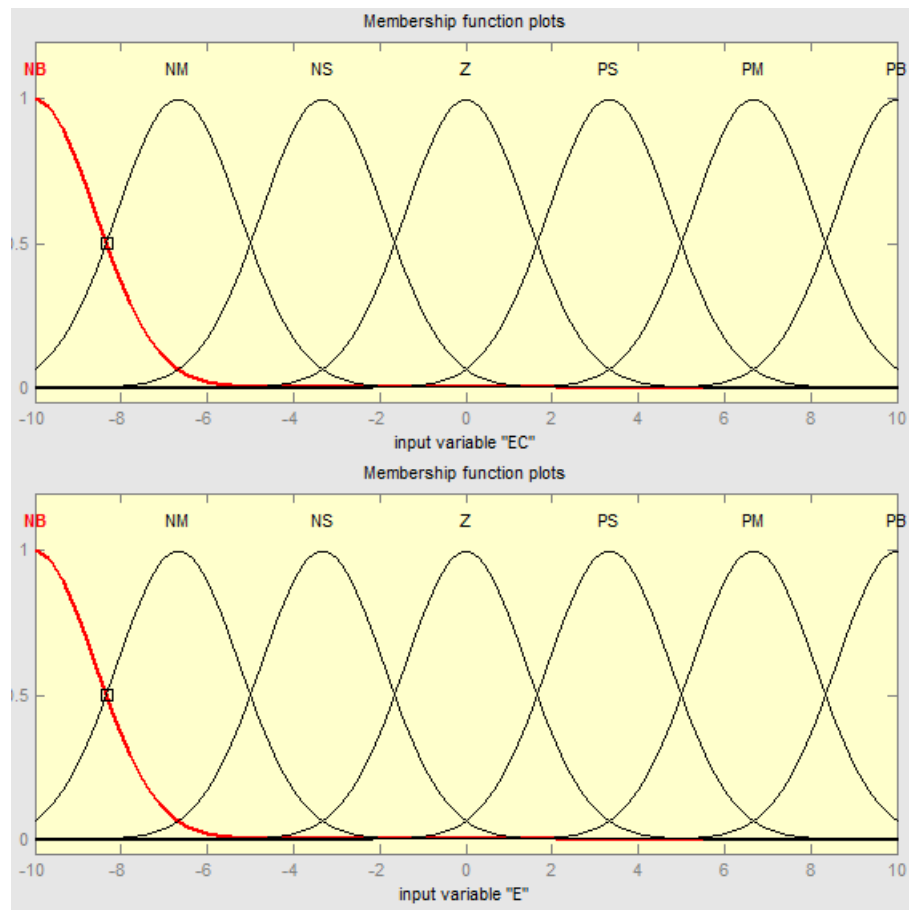


Figure 4.6: Membership function plotting of of  $E$ ,  $EC$

the schematic diagram in Figure 4.5. Firstly, to convert the input and output values to a semantic variables, fuzzification of the input and the output variables is employed, where the fuzzy range of the variables are  $e \in [-1, 10]$ ,  $ec \in [-1, 10]$ ,  $\Delta K_P \in [1, 1.5]$ , and  $K_I \in [0, 1.5]$ . More precisely, the fuzzy variety is segmented into seven semantic variables, and the equivalent fuzzy subsets are  $e, ec, \Delta K_P, \Delta K_I = \{NB, NM, NS, ZO, PS, PM, PB\}$

where

- NB: Negative Big
- NM: Negative Middle

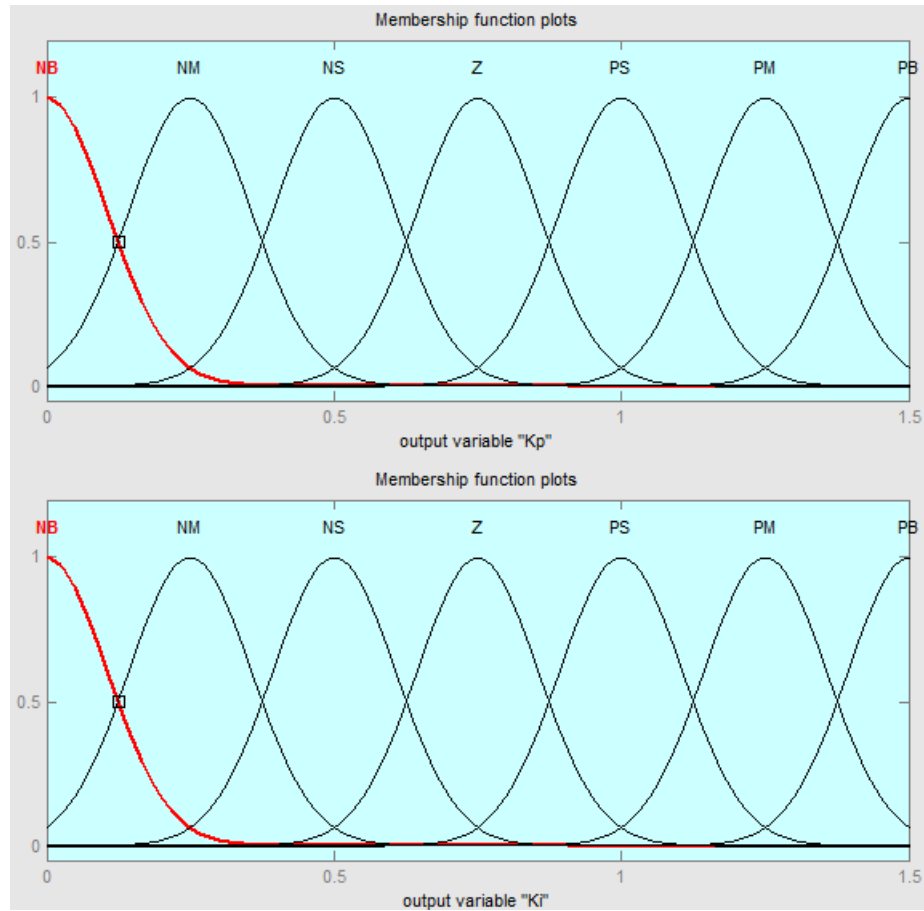


Figure 4.7: Membership function plotting of  $\Delta K_P$  and  $\Delta K_I$

- NS: Negative Small
- ZO: Zero
- PS: Positive Small
- PM: Positive Middle and
- PB: Positive Big

Every membership function is set to the Gaussian curve. Figure 4.6 and Figure 4.7 show the fuzzy membership functions. The significant step in designing a fuzzy control logic is the formation of a fuzzy inference rule among the input variables,  $e$  and  $ec$  and the output variables,  $\Delta K_P$  and  $\Delta K_I$ , based on input-output data. In this research, the rules of the PI parameters are applied in the team server and the value for P and I are extracted through simulation and experimentation. The fuzzy laws are shown in Table 4.1 and Table 4.2.

According to the Table 4.1, total forty-nine rules can be formulated as follows: if  $e$  is  $A_i$  and  $ec$  is  $B_j$ , then  $\Delta K_P/\Delta K_I$  is  $C_{ij}/D_{ij}$  where  $A_i, B_j, C_{ij}, D_{ij}$  are equivalent to the fuzzy subsets of  $e, ec, \Delta K_P, \Delta K_I$ . To achieve the fuzzy inference, Mamdani's Min-Max is employed. For example, the membership degree of the fuzzy subsets  $C_{ij}$  for the parameter  $\Delta K_P$  can be obtained as follows:

$$u_c(\Delta K_P) = \bigvee_{i,j=1}^7 \{[u_i(e) \wedge u_j(ec)] \wedge u_{cij}(\Delta K_P)\} \quad (4.39)$$

where  $u_x$  is the membership degree.

Table 4.1: Fuzzy rules for  $K_P$ 

$\Delta K_p \backslash e$	ec	NB	NM	NS	ZO	PS	PM	PB
NB		PB	PB	PM	PM	PS	ZO	ZO
NM		PB	PB	PM	PS	PS	ZO	NS
NS		PM	PM	PM	PS	PS	ZO	NS
ZO		PM	PM	PS	ZO	NS	NM	NM
PS		PS	PS	ZO	NS	NS	NM	NM
PM		PS	ZO	NS	NM	NM	NM	NB
PB		ZO	ZO	NM	NM	NM	NB	NB

Table 4.2: Fuzzy rules for  $K_I$ 

$\Delta K_I \backslash e$	ec	NB	NM	NS	ZO	PS	PM	PB
NB		NB	NB	NM	NM	NS	ZO	ZO
NM		NB	NB	NM	NS	NS	ZO	ZO
NS		NB	NM	NS	NS	ZO	PS	PS
ZO		NM	NM	NS	ZO	PS	PM	PM
PS		NM	NS	ZO	PS	PS	PM	PB
PM		ZO	ZO	PS	PS	PM	PB	PB
PB		ZO	ZO	PS	PM	PM	PB	PB

Defuzzification is the process of producing a quantifiable result in Crisp logic, given fuzzy sets and corresponding membership degrees. To get the crisp values, the center of gravity approach is applied. The coefficient  $\Delta K_P$  ( $\Delta K_I$  is similar) can be estimated as follows.



$$\Delta K_P(e, ec) = \frac{\sum_{k=1}^7 \Delta K_P u_c(\Delta K_p)}{\sum_{k=1}^7 u_c(\Delta K_p)} \quad (4.40)$$

After defuzzification, the parameters  $K_P$  and  $K_I$  can be:

- $K_P = K_{P0} + \Delta K_P$
- $K_I = K_{I0} + \Delta K_I$

Where  $K_{P0}$  and  $K_{I0}$  are the actual coefficients of P and I.

## CHAPTER 5

# SIMULATION RESULTS

In this chapter, the simulation of the robot model with the proposed controller is discussed. Also, the design and implementation of the path planner and the related low-level LQR controller are simulated. The outcomes of the simulation are also illustrated in this chapter. The simulation has been done in Matlab Simulink 2015.

### 5.1 Implementation

A summarized controller schematic of the FLC tuned PI controller on the trajectory planner, and the related low-level controller LQR is shown in Figure 5.1. A circular or rectangular path is defined in the path planner, which generates the target positions. The velocity filter checks the transitional and rotational velocity/acceleration limit to avoid damage.

The fuzzy adaptive PI controller estimates the target position based on the robot's current position to minimize the position error. Then, inverse kinematics

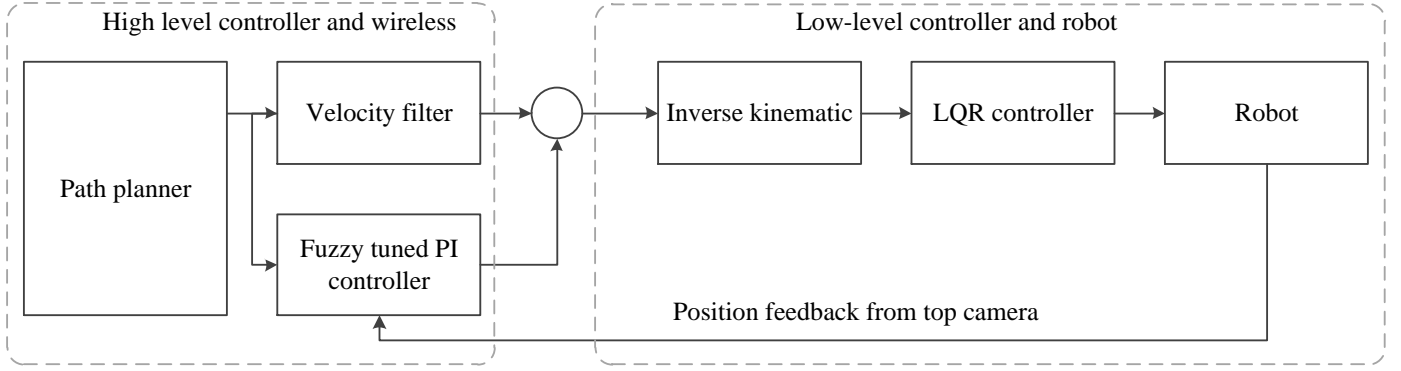


Figure 5.1: Fuzzy-PI path planner and LQR controller schematic

Table 5.1: Robot model parameters and values

Parameter	Unit	Value
Mass of robot	kg	0.743
Gear ratio of motor to wheel		$\frac{15}{51}$
Radius of omni-wheels	meters	0.0029
Distance between wheel's center and robot's center	meters	0.0798576
Resistance of motor from terminal to terminal	ohms	0.464
Back-emf constant of motor	$\frac{Vs}{rad}$	0.025
Torque constant of motor	$\frac{Nm}{A}$	0.0251
Viscous friction coefficient of wheel assembly	$\frac{Nms}{rad}$	0.0001
Moment of inertia of wheel assembly		2.43695253e-5
Max voltage applied to the motor phases	volts	18
Angle of each wheel axis relative to the +x axis	degree	295, 65, 135, 225

derives the robot body velocity from the global velocity. The LQR controller computes an optimal set of minimum input voltages for the motors considering the model specifications listed in Table 5.1.

According to the robot dynamics, the robot starts to move and tries to reach the desired point. There are two feedback sources used in this simulation. The

current velocity of the robot is fed into the LQR controller to calculate the new velocity vector. Also, position feedback is used in the fuzzy control logic to make accurate path planning. At the bottom of the model, there is an FLC that generates a set of optimal values for the PI controller considering the tracking errors,  $e$  and change in errors,  $ec$ .

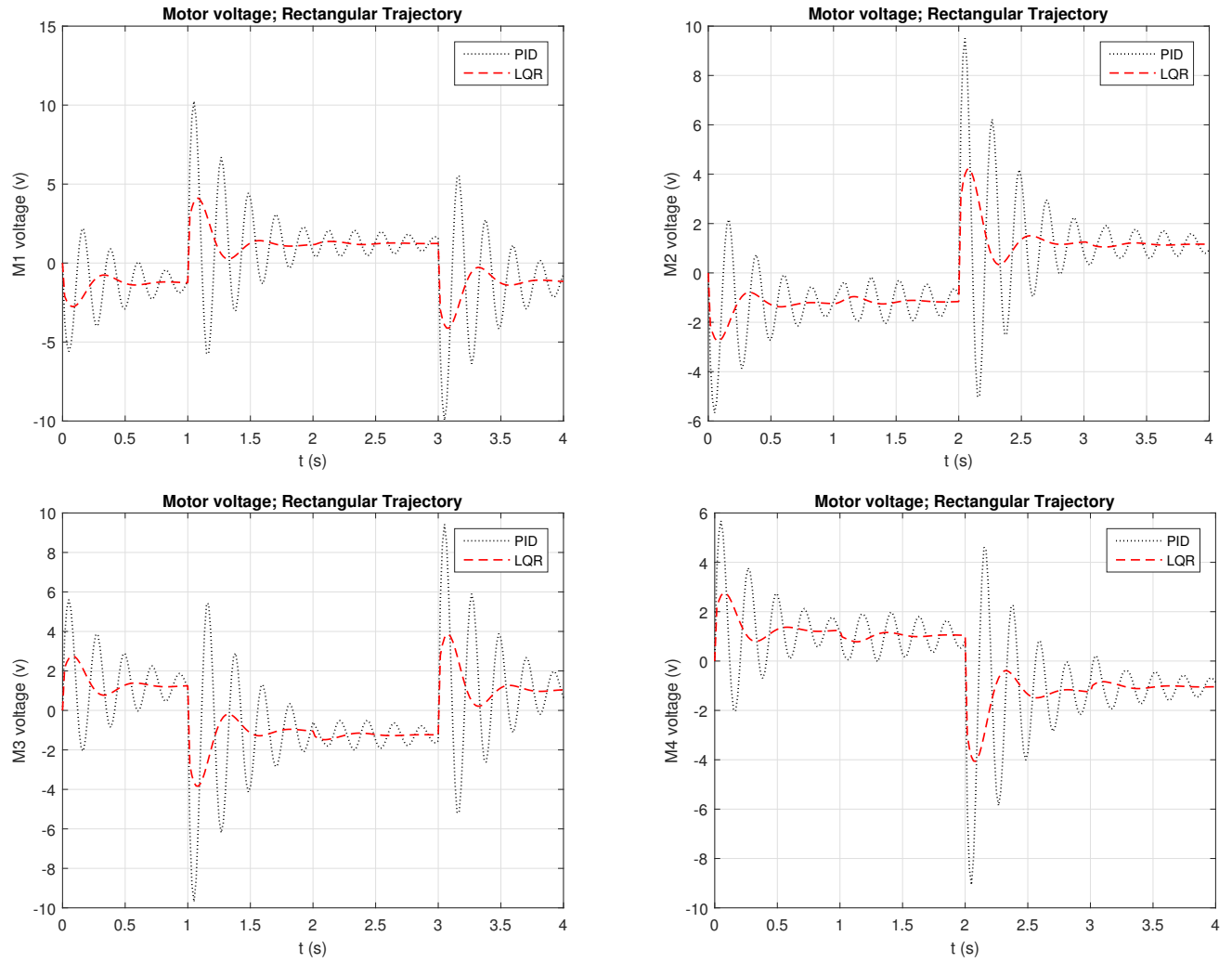


Figure 5.2: Optimal voltage (minimum) for motors on the rectangular trajectory for the PID and LQR controller

## 5.2 Simulation Setup

To simulate the four-wheeled omnidirectional mobile robots using FLC tuned PI with LQR controller, the Matlab Simulink 2015 software is used. The initial parameters of the robot model are given in Table 5.1. Also, the simulation model schematic is shown in Figure 5.1. These parameters are provided to the LQR controller as a parameter matrices  $A$  and  $B$ .

## 5.3 Results and Discussion

The primary concern of employing LQR is to make the model linear. An integrator is used on the error matrix in the low-level controller. This approach is energy-efficient because it estimates the optimal voltage for each motor driver to drive accurately with minimum tracking error. Figure 5.2 illustrates the desired and the actual controller output voltage for a particular circular trajectory. The robot position and orientation variation on a circular and a rectangular trajectories are depicted in Figure 5.3 and Figure 5.4 respectively where the robot's translational and rotational speed are the same (1 m/s and 0.05 rad/s). The line graph in Figure 5.3 reflects the performance of the proposed method in a circular trajectory. The outcome of the simulation results contains that the robot tracked the desired path with a very slight deviation.

The robot's global position on the playground and the wheels angular speed are estimated according to the given kinematics equations. In contrast, quick fluctuations in the path-following lead to slippage of wheels, as is noticeable from

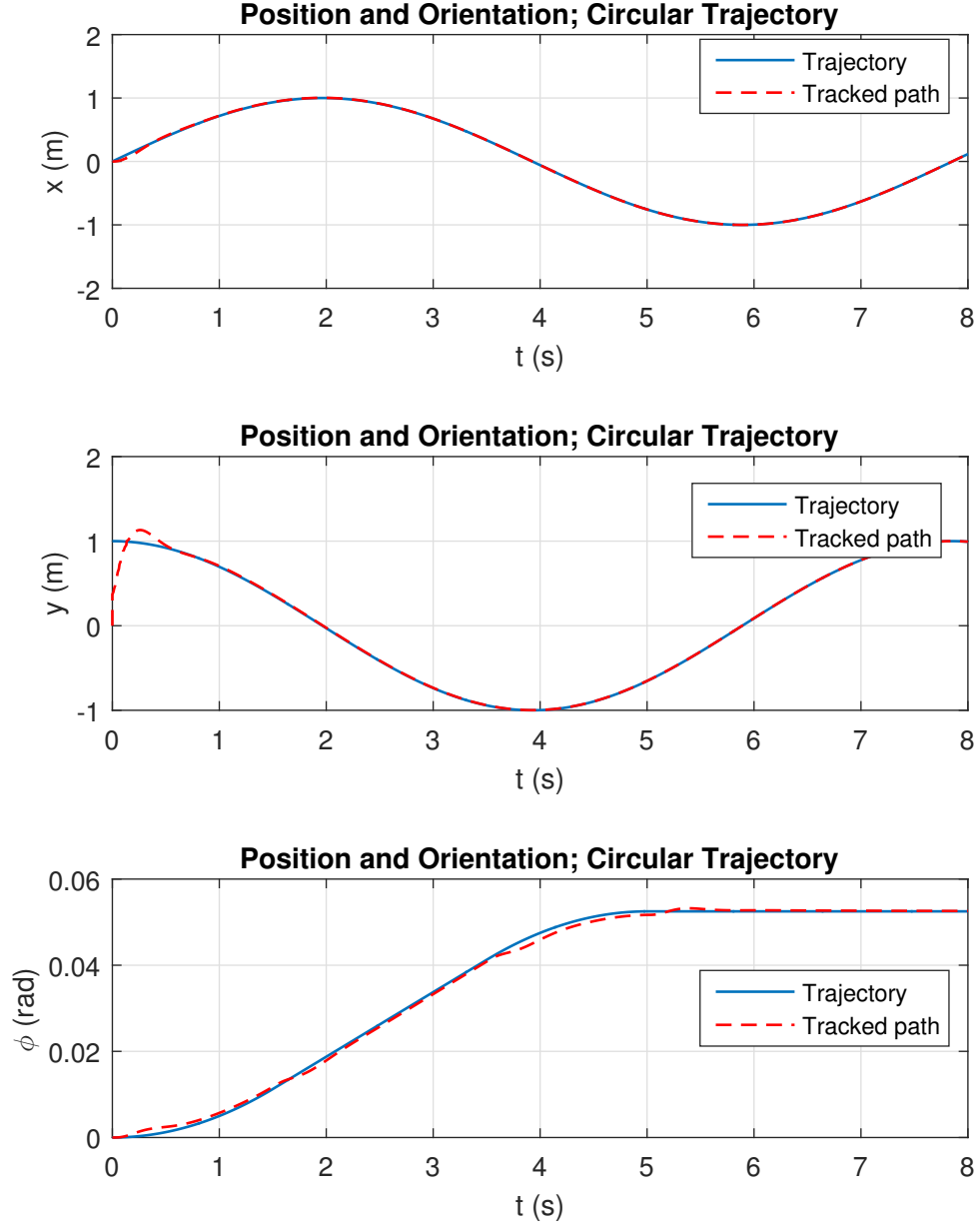


Figure 5.3: The difference between tracked path and trajectory on a circular pathway by the LQR controller with robot velocity  $V_{robot} = 1m/s$   $\phi = 0.05rad/s$

the first trajectory depicted in Figure 5.4. It depicts the relationship between the geometric coordinates of this pathway and the robot's position that result in 1.1% mean position and 3.2% orientation deviations. Furthermore, it is clearly observed that the error is decreasing along the straight pathways, which validates

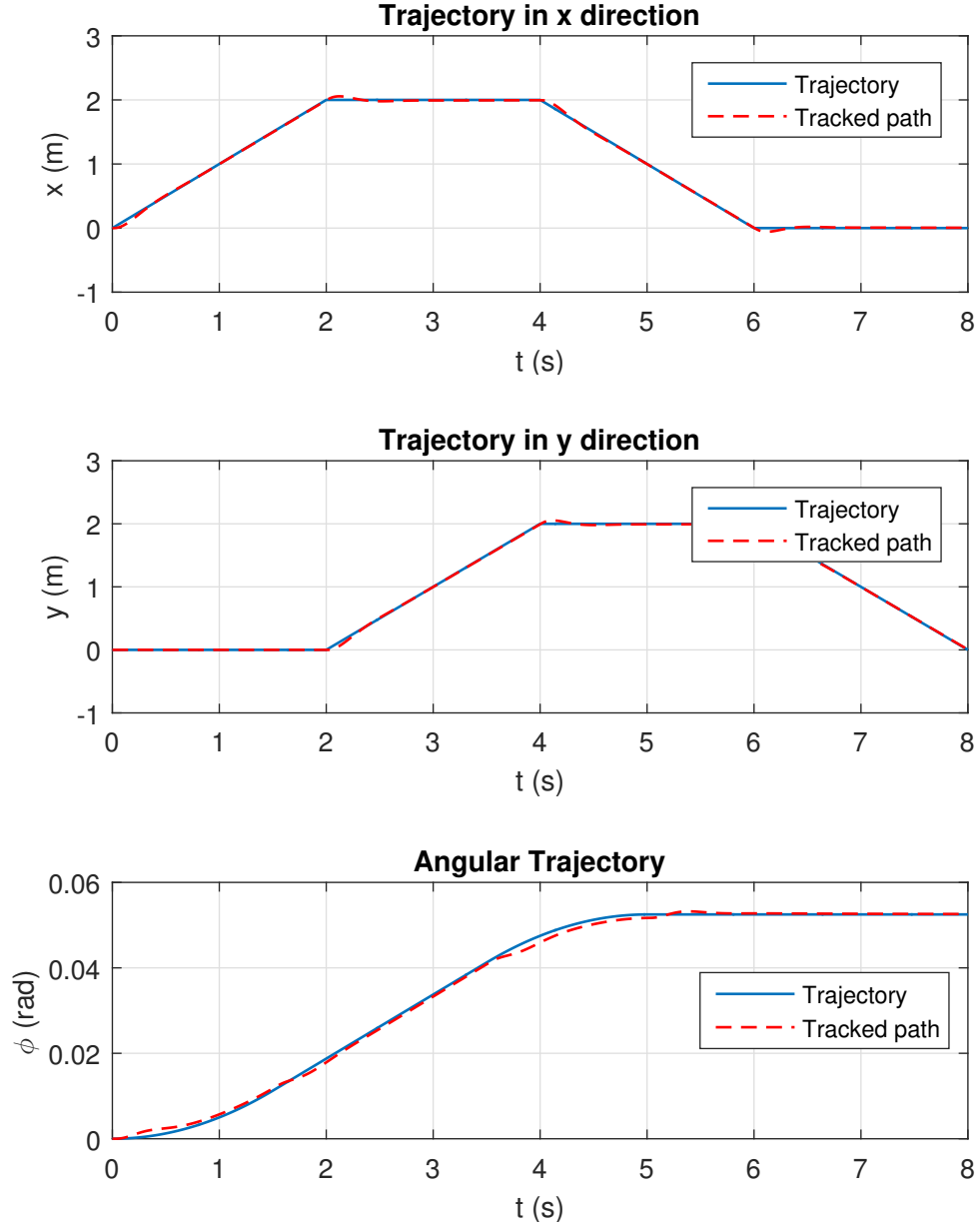


Figure 5.4: Robot position and orientation variation in rectangular trajectory

proper PI value selection of the high-level position controller.

The surface area of fuzzy controller that presents the correlation between the input parameters,  $e$  and  $ec$ , and the corresponding outputs,  $K_P$  and  $K_I$  depicted in Figure 5.5.

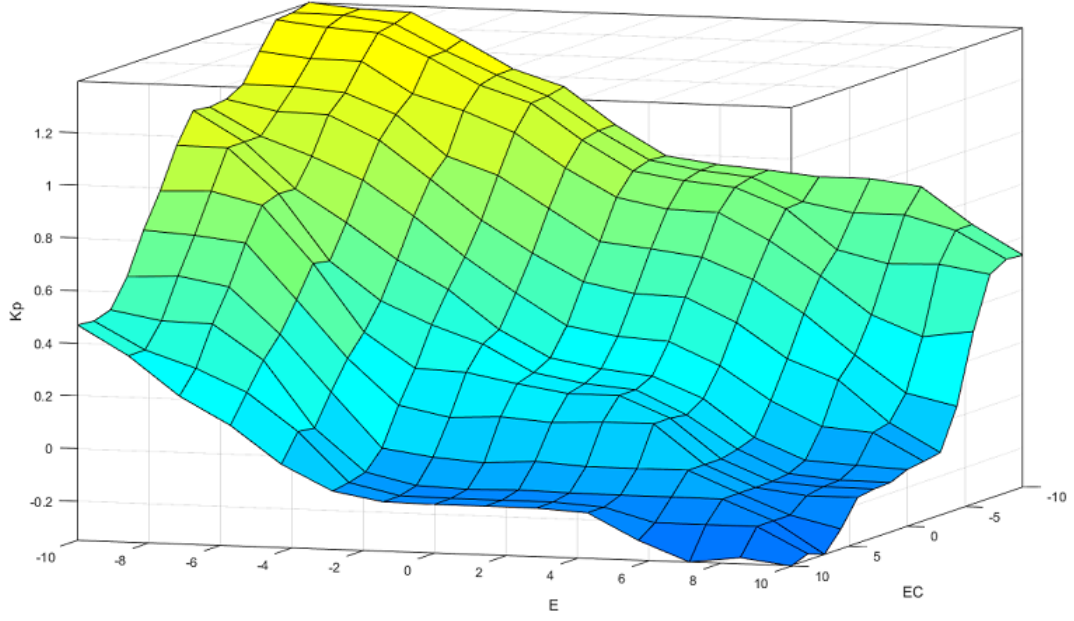


Figure 5.5: Surface area curve of fuzzy control logic

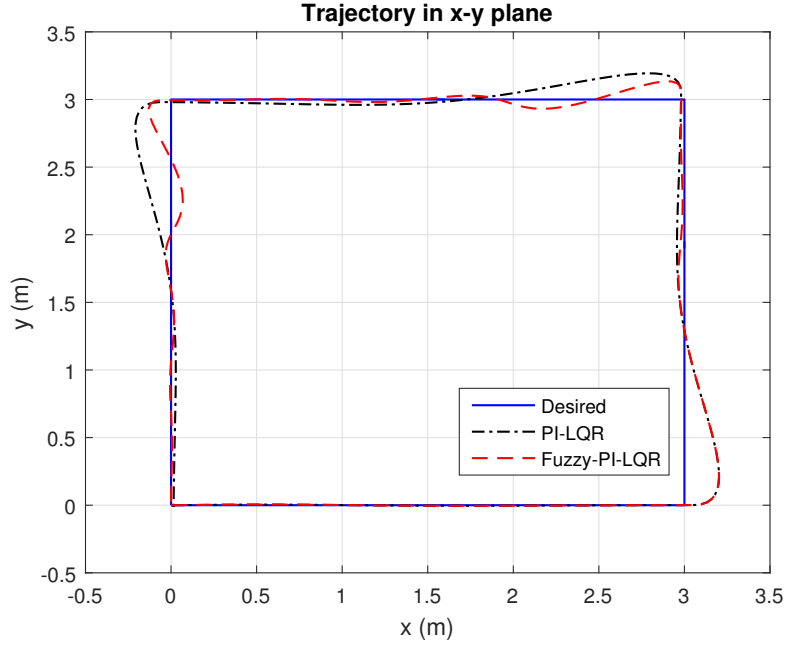
Table 5.2: Error comparison with existing literature

Reference	Controller Type	Horizontal(%)		Vertical(%)	
		3 m/s	4 m/s	3 m/s	4 m/s
Hashemi et al [1]	PI-LQT	7.1	5.2	14.2	16.3
	PI-Fuzzy-LQT	5.8	2.0	11.4	2.4
Proposed Model	PI-LQR	3.4	6.7	9.0	10.6
	Fuzzy-PI-LQR	1.2	0.9	6.6	8.7

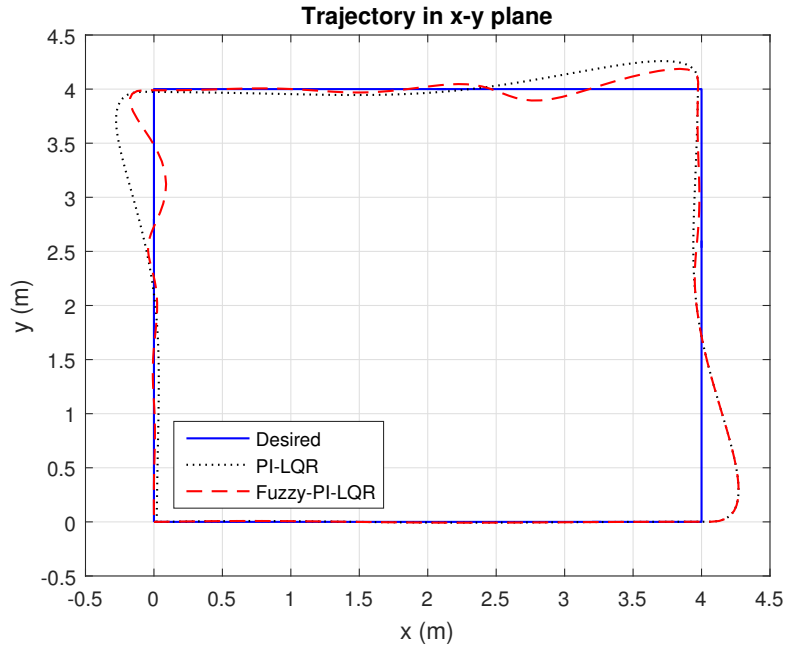
The following graphs provide the outcomes of the given controller for rectangular pathway with different robot velocities. The path-following characteristics of the experimental model for the PI-LQR and fuzzy-PI-LQR schemes is presented in Figure 5.6. The detected undulations in the speed profile curves, especially at the corners, are ascribed to adjusting the robot with given trajectories to encounter path's geometrical requirements and robot dynamics.

Figure 5.6 indicates that the max. deviation along y-axis and x-axis movement





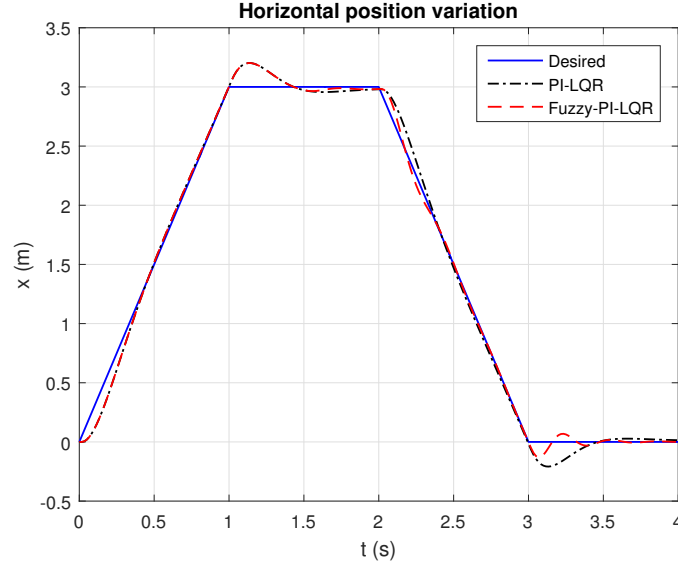
(a)



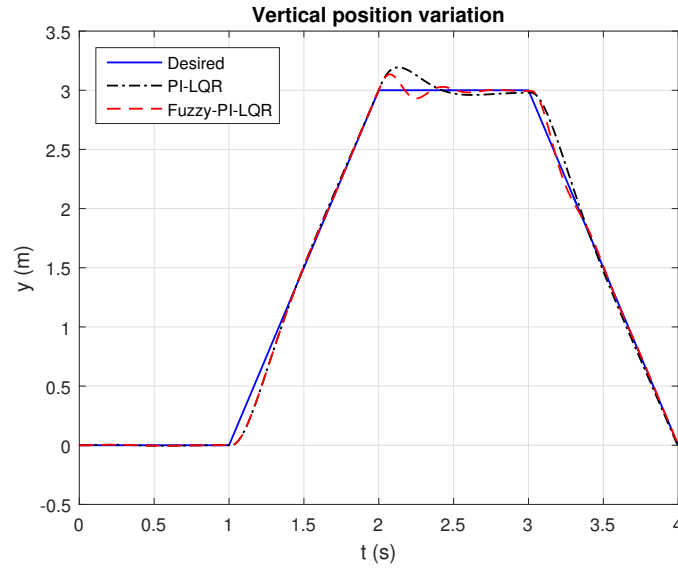
(b)

Figure 5.6: The accuracy of rectangular path following for robots transitional speed (a) 3 m/s; (b) 4 m/s

are 9.0% and 3.4% respectively for the PI-LQR approach with the velocity of 3 m/s and then the error augmented to 10.6% and 6.7% for a pathway with the



(a)

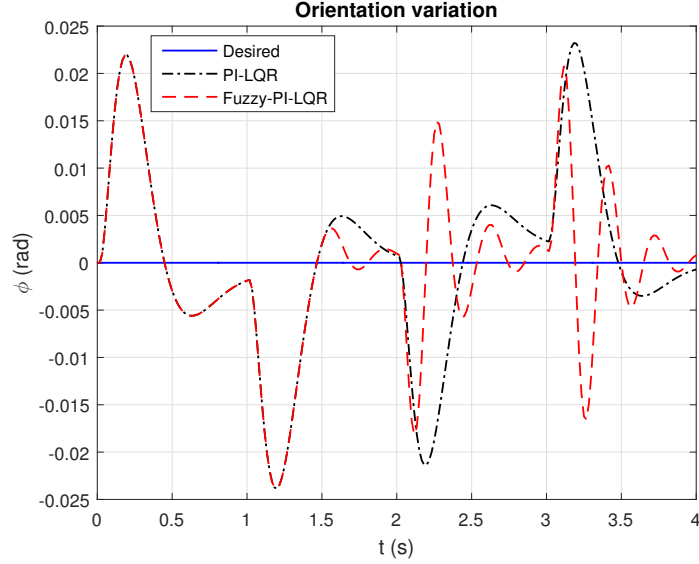


(b)

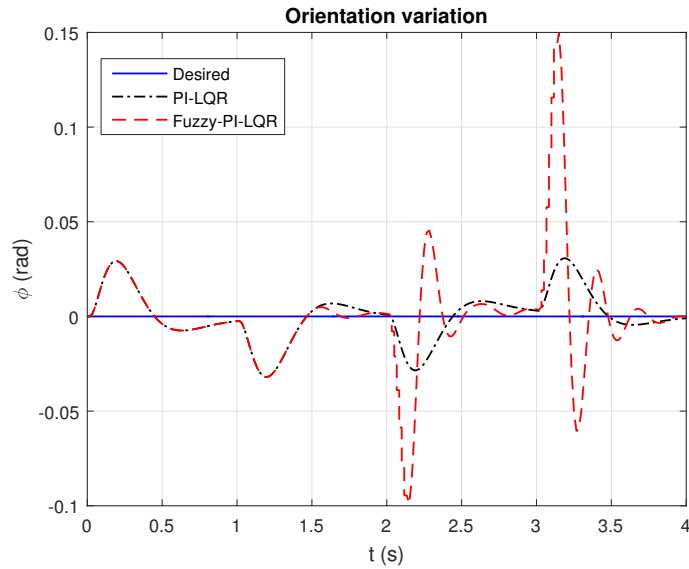
Figure 5.7: The horizontal and vertical deviations as the output of both PI-LQR and fuzzy-PI-LQR controller for the speed of 3 m/s

velocity of 4 m/s. The existence of the FLC on the maneuver planner diminishes the deviations to 6.6% and 1.2% for 3 m/s. 8.7%, 0.9% equally for velocities above the highest acceptable range. The error comparisons between PI-LQR and fuzzy-PI-LQR approaches with existing work are summarized in Table 5.2.

There is a satisfactory steadiness between the targeted path and the fuzzy-PI-LQR controllers outcome at low speed such as 3 m/s, but the deviation becomes higher when speed is high, i.e., 4 m/s. It is observed from Figure 5.6 that the established fuzzy-PI-LQR controller offers a further accuracy compared to the



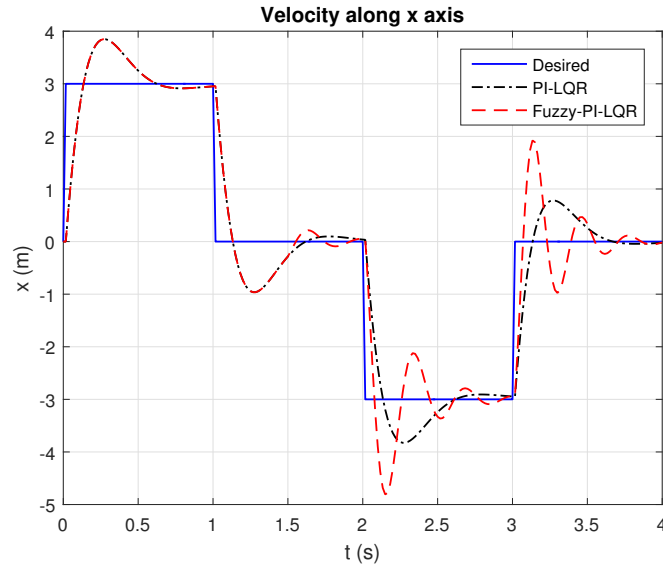
(a)



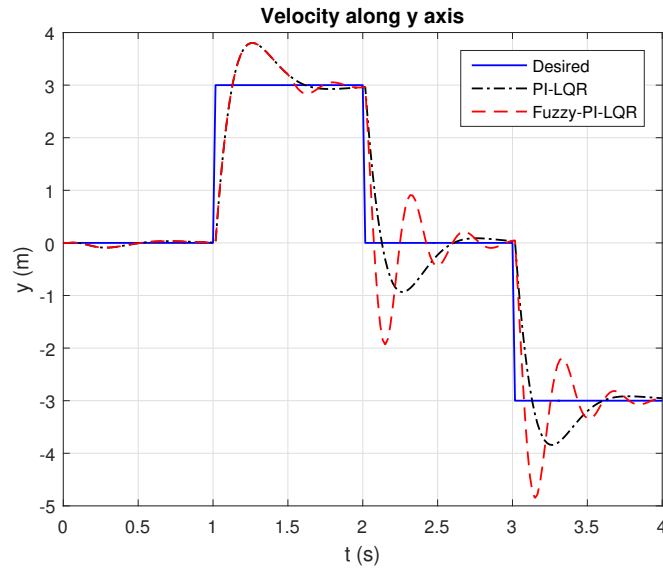
(b)

Figure 5.8: The comparison of robot orientation error in both cases with the velocity (a) 3 m/s; (b) 4 m/s

4 m/s is illustrated in Figure 5.8. It is clear that a sharp direction change is observed in robot alignment variations during the rectangular maneuver. These vicissitudes lead to the robot slip slightly, particularly for the max. velocity which is estimated in Section 4, as depicted in Figure 5.8b. These vacillation amplitudes are nearly 50% of the identical in the absence of FLC.



(a)



(b)

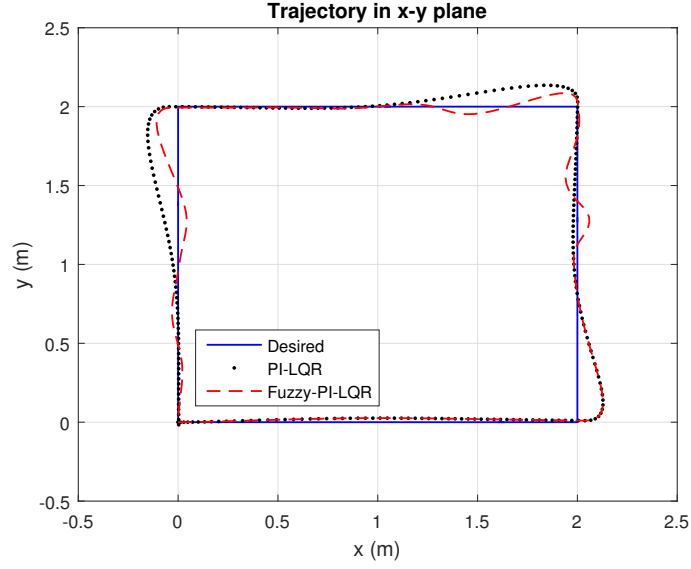
Figure 5.10: The velocity norm in both horizontal and vertical in 3 m/s

For the sake of analyzing the deviation, a circular maneuver is created using two existing trajectory planners. The outcome depicts in Figure 5.9, where the error variations of both approaches are illustrated. The detected slight fluctuation on proposed method caused for input and output rule segmentation.

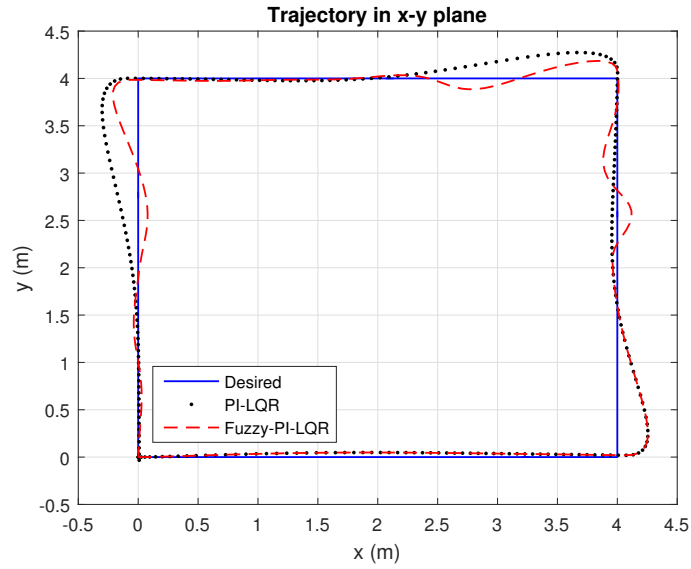
In Figure 5.10, there is an inclination of velocity increase and decrease respectively near the angular points that appear rational based on the stated pathway.

In Figure 5.11, the expected outcomes for the FLC trajectory planner is illustrated in the presence of randomly distributed disturbance applied for both trajectory planners with frequency and amplitude of 1 rad/s and 0.01m respectively. The difference of the robots orientation and position errors with the mentioned external loads for the traditional trajectory-following task is demonstrated in Figure 5.12. This experimental measurement is studied with the slip-free marginal velocity  $V = 4$  m/s. Robot position, orientation and motion smoothness of the proposed path planner in compared to the generated trajectory profile with the mentioned external noise are depicted in Figure 5.12.

Finally, to test our model on a complex and smooth trajectory following, the deviation between targeted path and tracked path is shown for both PI-LQR and fuzzy adaptive PI-LQR controller in following graphs. A circular trajectory following outcome is shown in Figure 5.13. Two asymmetric shape trajectories such as a DNA and a flower shape with the velocity of 2 m/s are depicted in the Figure 5.14 and Figure 5.15 respectively. Initially, the stable time of PI-LQR and fuzzy-PI-LQR was similar, but it decreased significantly after a while in case of



(a)



(b)

Figure 5.11: The deviation from targeted trajectory in presence of randomly distributed disturbance on rectangular trajectory for the transitional speed of (a)  $V = 2$  m/s; (b)  $V = 4$  m/s

fuzzy-PI-LQR.

The standard deviations of the desired velocity norm specified by the path planner,  $\sigma_{V_{nd}}$ , the velocity norm error,  $\sigma_{eV_n}$  and the maximum error of the velocity norm,  $eV_{n_{max}}$  are given in Table 5.3 for a range of robot velocities. The lower

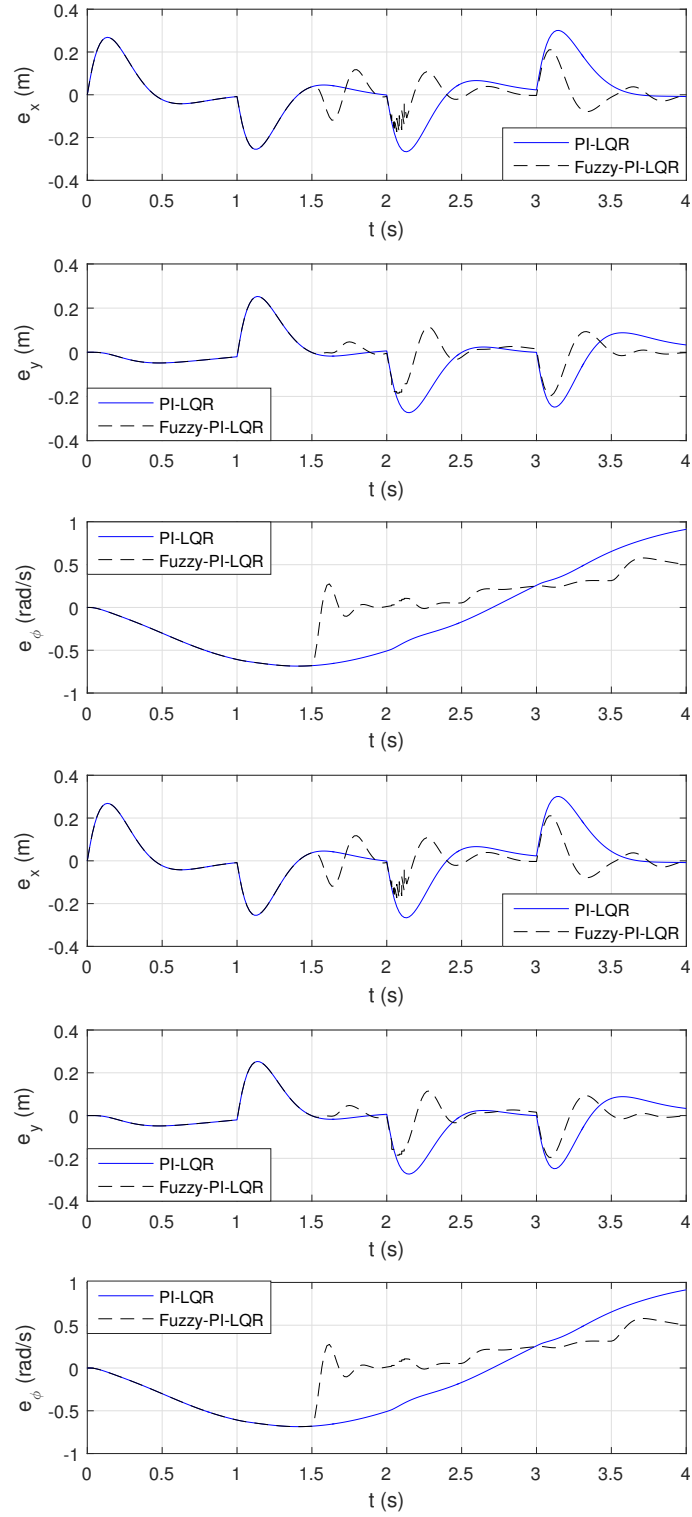


Figure 5.12: Robot Position error analysis on the rectangular maneuver with external load with the velocity of 3m/s and 4m/s

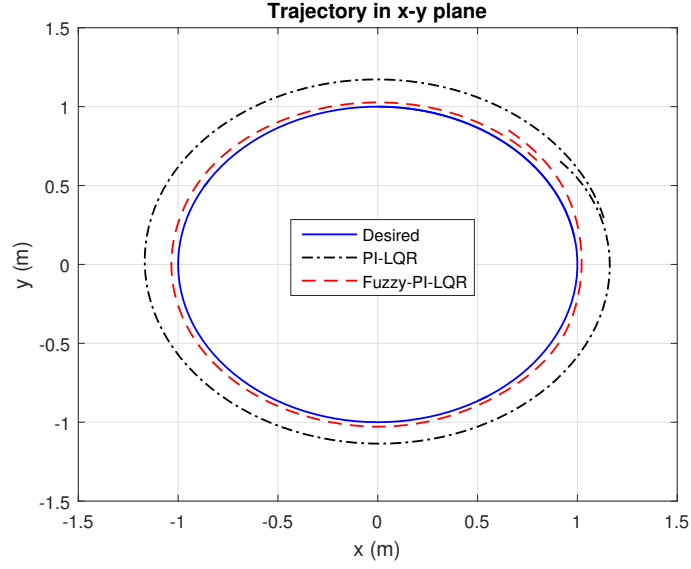


Figure 5.13: Circular Trajectory with  $v = 4$  m/s

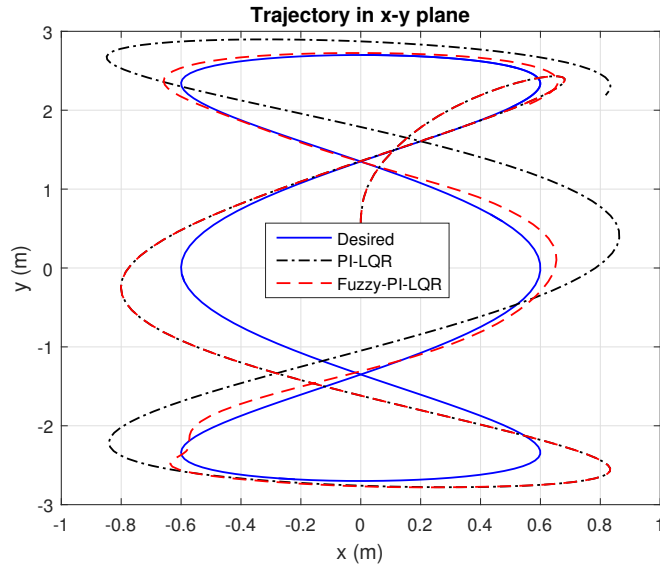


Figure 5.14: DNA shape trajectory with  $v = 2$  m/s

standard deviation observed for the fuzzy-PI-LQR approach for all three maneuvers, especially with the presence of noise, rationalizes employing this method which leads to lower fluctuations of desired velocity norms.



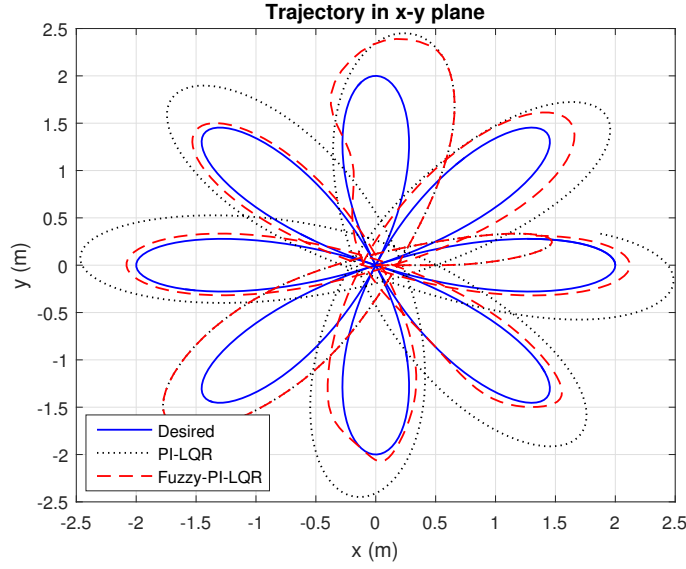


Figure 5.15: Flower shape trajectory with  $v = 2$  m/s

Table 5.3: Comparison of velocity norm with existing literature for the two path planners

Speed $V$ (m/s)	STD.	Hashemi et al [1]		Proposed Model	
		PI-LQT	PI-fuzzy-LQT	PI-LQR	Fuzzy-PI-LQR
$V = 2$ m/s	$\sigma_{Vnd}$	0.89	0.54	0.89	0.54
	$\sigma_{eVn}$	1.25	0.48	0.95	0.24
	$eVn_{max}$	3.45	1.96	2.19	1.06
$V = 3.15$ m/s	$\sigma_{Vnd}$	1.05	0.97	1.05	0.97
	$\sigma_{eVn}$	1.13	0.57	1.90	0.37
	$eVn_{max}$	3.70	2.42	2.30	1.13
$V = 4$ m/s	$\sigma_{Vnd}$	1.26	1.08	1.26	1.08
	$\sigma_{eVn}$	1.79	0.68	1.53	1.10
	$eVn_{max}$	4.27	2.46	2.9	1.5

## CHAPTER 6

# EXPERIMENTAL RESULTS

The fuzzy adaptive PI high-level controller is implemented for tracking the robot position. Concurrently, a PI low-level controller is employed in the robot to control the wheel speed. A single four-wheeled omnidirectional RoboCup-SSL robot is made to do the experiment of the proposed control system. To apply the proposed approach in a real-life application, an experimental execution of the embedded system for controlling the motion of given four wheel omnidirectional robot is described in this chapter. The robot hardware design, pin configuration of printed circuit board and the outcome of this experiment are discussed sequentially. Moreover, an outline of the experimental procedure is also given in this chapter.

### 6.1 Robot Hardware Design

Two XBee modules (router AT and coordinator AT) with baud rate 9600 are used for wireless communication. There are four motor drivers used with the maximum

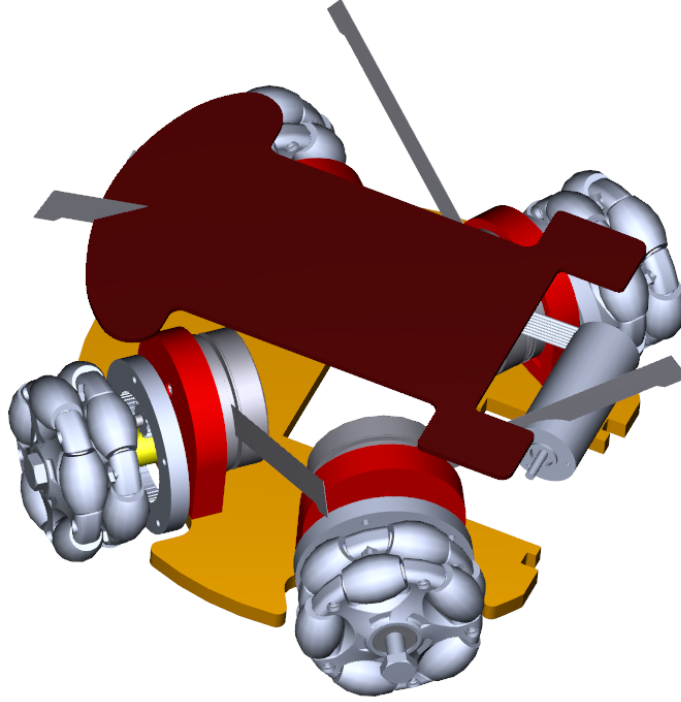


Figure 6.1: The top view of KFUPM RoboCup-SSL robot

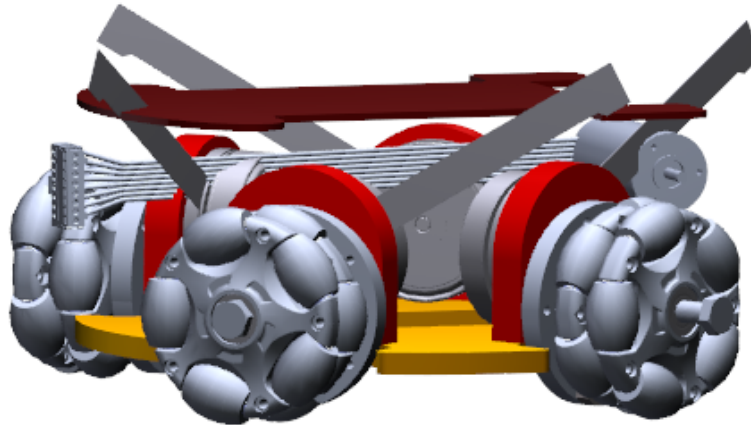


Figure 6.2: The side view of KFUPM RoboCup-SSL robot

rotation of 625rpm with the settings of  $N1=1$  and  $N2 = 0$  of the motor driver. The inner closed loop exists to control the wheels angular velocity using PI controller considering the motor encoder feedback. Moreover, there is an HD web-cam used in this experiment for vision feedback to the position controller.

### 6.1.1 Robot Mechanical Design

Robot's main structure is made of high-quality plastics to keep it robust and light. Because this is just a prototype of the main robot. To avoid short circuit problem among electrical components and to decrease the losses, parts are hardening anodized. The new designs characteristics are as follows:

- Robot Diameter: 178 mm
- Robot Height: 138 mm
- Robot Weight: 0.75 kg

The mechanical design of our RoboCup-SSL robot is shown in Figure 6.1 and Figure 6.2.



Figure 6.3: Omni-wheels

### 6.1.2 Wheels and Gears

The number of wheel of this omnidirectional robot is four. Each wheel is driven by a BLDC motor. The motor power is transmitted via two gears with a ratio of 51:15 to wheels. The specifications of the omni-wheels are as follows

- Body and Roller Color: Black

- Wheel Diameter: 58mm
- Roller Diameter: 13mm
- Body Material: Nylon
- Roller Material: Nylon and PE
- Load Capacity: 3kg
- Net Weight: 60g

An omni-wheel is illustrated on different views in Figure 6.3



Figure 6.4: An EC 45 brush-less DC motor.

### 6.1.3 Motors and Drivers

There are four brush-less DC motors used in each robot which has maximum speed of 10000 rpm. It has a hall sensor which is used to get the rotation feedback for an internal close loop. A sample 30 Watt EC 45 flat BLDC motor is shown in Figure 6.4.

Table 6.1: Motor Data

Title	Unit	Value
No load speed	rpm	4380
Nominal speed	rpm	2940
Nominal torque	nMm	55.5
Max efficiency	%	78
Terminal resistance	ohm	1.2
Torque constant	nMm/A	25.5
Rotor inertia	$gcm^2$	92.5

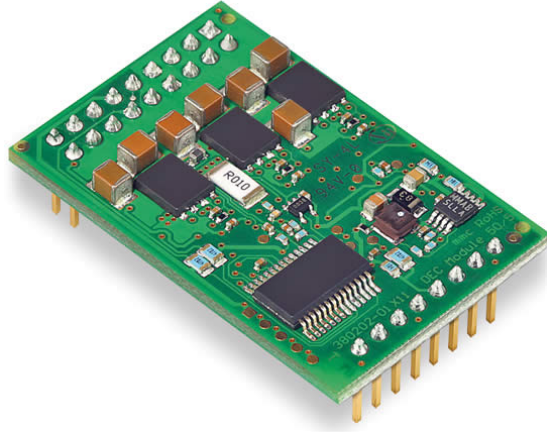


Figure 6.5: Motor driver.

In general, a brushed DC motor had a rotating armature and fixed magnetic field, and its commutation is mechanical. However, a BLDC motor is an inside-out DC motor. The main components of the BLDC motors are the stator and the rotor. The armature is in the stator and the magnets are on the rotor uses the position sensors, and an inverter to commutate electrically, which makes it a virtually maintenance-free motor. According to the datasheet, there are some important parameters in the motor specification shown in the Table 6.1.

A 1-Q-EC Amplifier DEC Module 50/5 used as a motor encoder in the robot which is shown in Figure 6.5. Also, the pin diagram of the motor driver is shown in Table 6.2.

Table 6.2: Pin configuration of motor driver.

Pin	Signal	Description
1-6	W1-W3	Motor winding
7,8	+Vcc	Supply voltage
9,10	Gnd	Ground
18	Monitor n	Speed monitor output
19	Ready	Status indicator output
20,21	DigIN1,2	Digital input
22	Enable	Enable input
23	Direction	Direction input
26	Set value speed	Speed input

Table 6.3: The technical specification of XBee s2c zigbee.

Feature	Characteristics
Data Rate	up to 1 Mbps
Range	200 ft
Receiver sensitivity	-100 dBm/-102 dBm
Serial data interface	SPI
Frequency band	ISM 2.4 GHz
Digital IO	15

#### 6.1.4 Radio Devices

To establish a reliable wireless communication between the robot and the team server, a low power 2.4 GHz XBee s2c ZigBee module is connected to the team server. Similarly, another ZigBee module is connected to the robot main board. Both are operated on ZigBee PRO 2007 protocol. The 16 channels ZigBee modules is capable of 128-bit encryption. There is a bidirectional communication exists to receive the command from and send the feedback to the team server. A brief specification of this device is given in Table 6.3. A communication between two XBee modules is shown in Figure 6.6. Finally, illustrations of our model robot are shown in the Figure 6.7 and Figure 6.8.

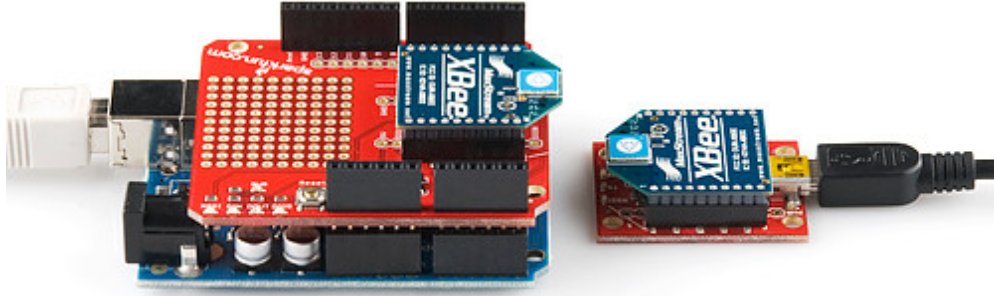


Figure 6.6: A XBee s2c zigbee is communicating with another XBee on Arduino.

## 6.2 Experimental Setup

The complete procedure of the experimental test bed is described in this section.

Before starting the experiment, it is prerequisite to check the pin configurations.

### 6.2.1 Pin Configuration and PCB

The pin configuration among Arduino, Xbee, and motor drivers is shown in Figure 6.9. Only 14 out of 26 pins of Dec 50/5 module are used. Four motor drivers and one XBee are connected to the main microcontroller (Arduino Due) board. Most important pin of motor driver is 26 (speed) which is connected to the PWM pins (2-5) of Arduino. The direction pin (23) of the driver required a digital binary signal, so it is connected to one of the digital pins (34-37) of Arduino. The enable (22) pin also needs a digital signal, so, it is connected to the pins (22-25). To select the range of motor speed, digital input pins N1 (20) and N2 (21) of the motor driver are connected to pins (40-49) of the Arduino. Finally, the ready pin of the driver is an output digital signal from the driver which is connected to pins (28-31) of Arduino.



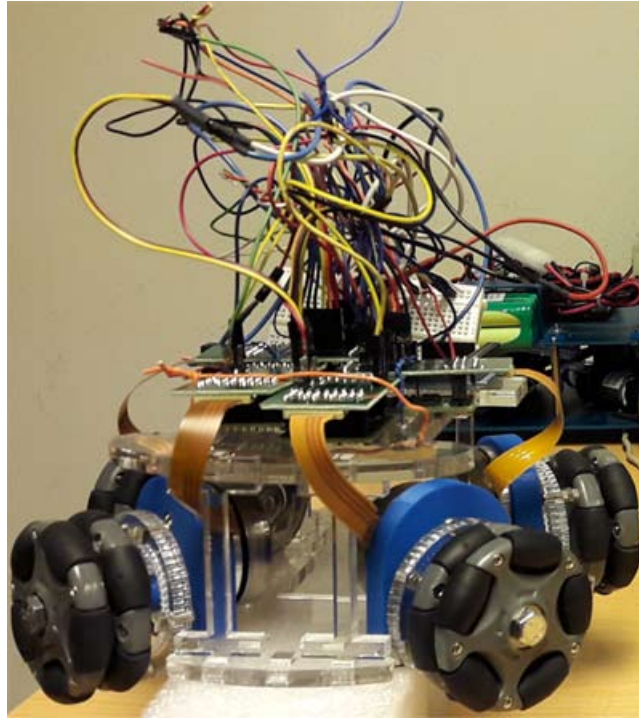


Figure 6.7: KFUPM SSL Robot

### 6.2.2 Install Firmware

At the very beginning of the firmware, motor drivers and radio devices initialization has been done. A test packet is sent and received between the robot and team server to confirm the wireless communication.

There are several tasks need to accomplish concurrently such as reading com-

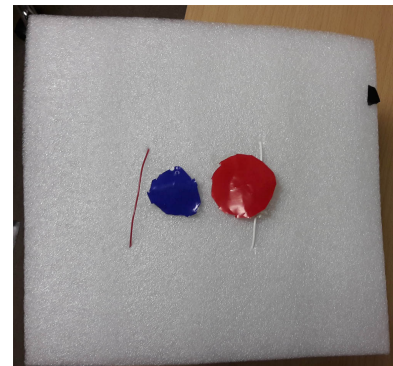
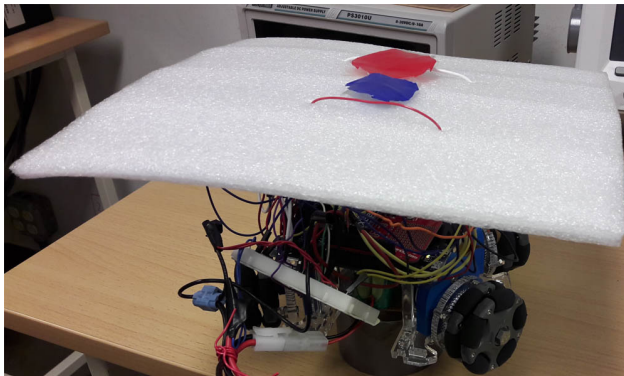


Figure 6.8: Robot View

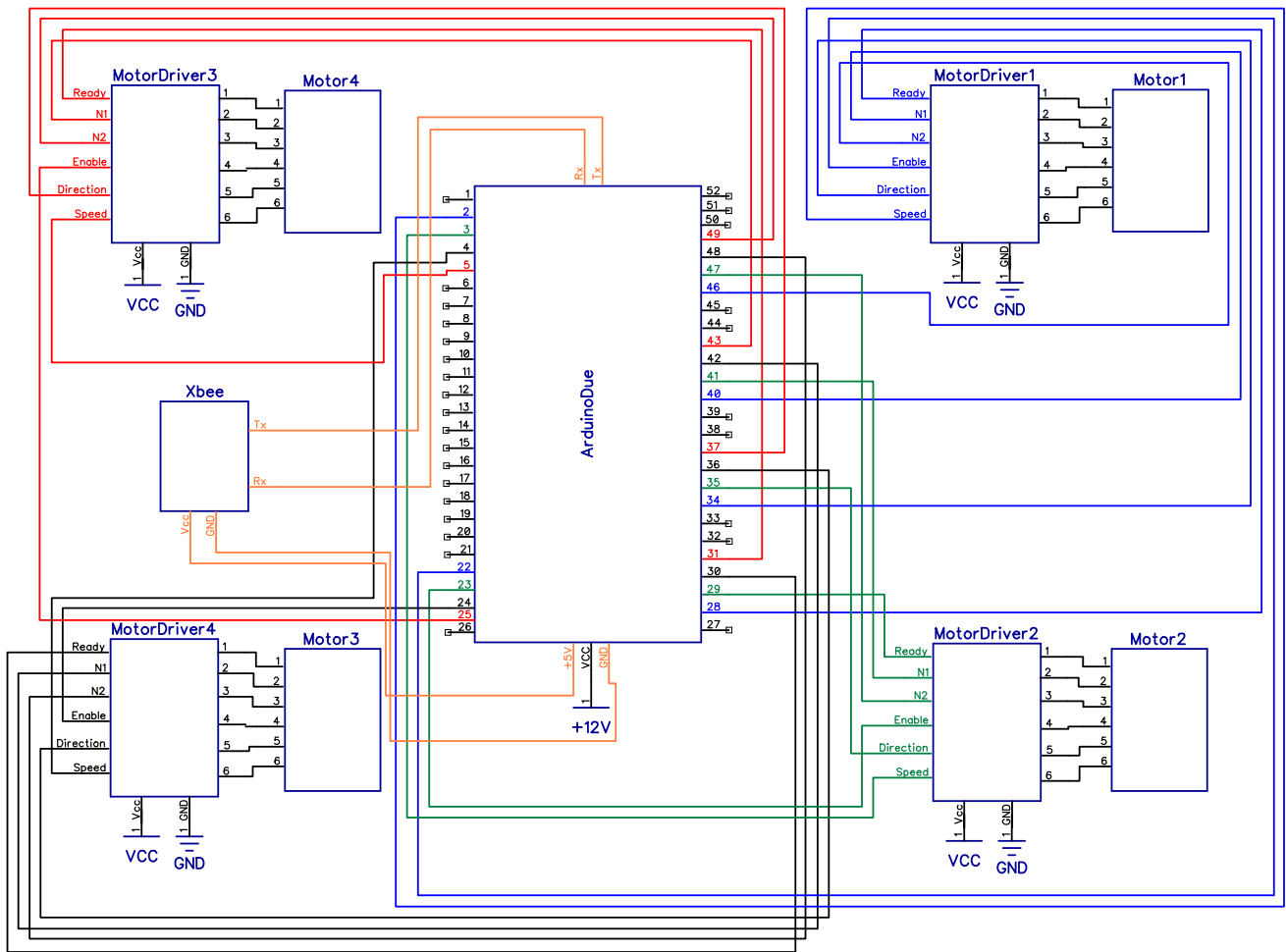


Figure 6.9: Pin Configuration

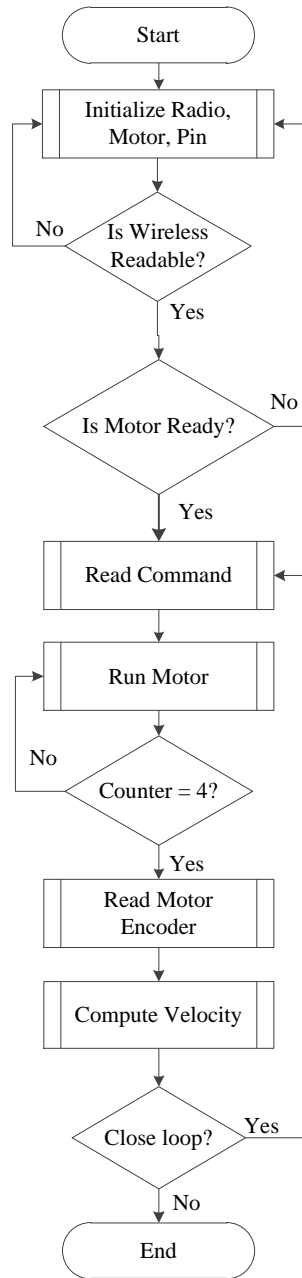


Figure 6.10: Flowchart of the embedded system firmware

mands, running the motors, and sending feedback. To do the multitasking, a multi-thread technique is used. Although Arduino does not support parallel tasks, but we can make use of a library to easily schedule the tasks with fixed or variable time between runs [58]. This library helps to maintain organized and to facilitate

the use of multiple tasks. It used timer interrupts to make it powerful by running pseudo-background tasks.

When motors are ready, the microcontroller sends PWM signal to each motor sequentially according to the target velocity of the robot. There is an internal close loop that can adjust the motor speed using PI controller. A recent velocity feedback is received from the *monitor n* of motor encoder to calculate the targeted PWM signal for the motors.

### 6.2.3 Radio Message Format

The message format for wireless communication is shown in Figure 6.11 where  $D$  means the direction and  $S$  mean the speed. The value of this message is in byte. If the value of  $D1$  is 0, the wheel will rotate clockwise and 1 for anti-clockwise. Similarly, from  $D2$  to  $D4$  for motor 2 to motor 4. The PWM signal range is 0 to 255. If  $N1$  and  $N2$  digital pin of the motor driver are set to 0 and 1 respectively, the lowest speed range will be 65 to 625 rpm.

**D1 . S1 . D2 . S2 . D3 . S3 . D4 . S4**

D – Direction (0: Clockwise, 1: Anti-clockwise)

S – Speed (0-255)

e.g. 1.100.0.45.1.255.0.3.1.200

Figure 6.11: Message format

## 6.2.4 Experimental Procedures

To run the experimental test, the following steps need to follow

1. Check connections according to the pin configuration.
2. XBee settings: The following settings is done for both receiver and sender XBee using XCTU tool.
  - **Mode:** One is Router AT mode and another is Coordinator AT mode.
  - **Baud rate:** 9600 bps
  - **PAN ID:** 2500
  - **Channel:** C
3. Vision system settings: Connect a USB webcam to the team server. Use robotLocation.m file to calibrate the camera. There are two dots on the top of the robot. The *red dots* means the center of the robot and *blue dot* means the heading.
4. Firmware upload: Connect Arduino and burn firmware.ino through USB. Press the reset button.
5. Connect 12V battery to the robot.
6. Create a trajectory e.g. line or rectangular trajectory in the controller.m file.
7. Run controller.m file on matlab

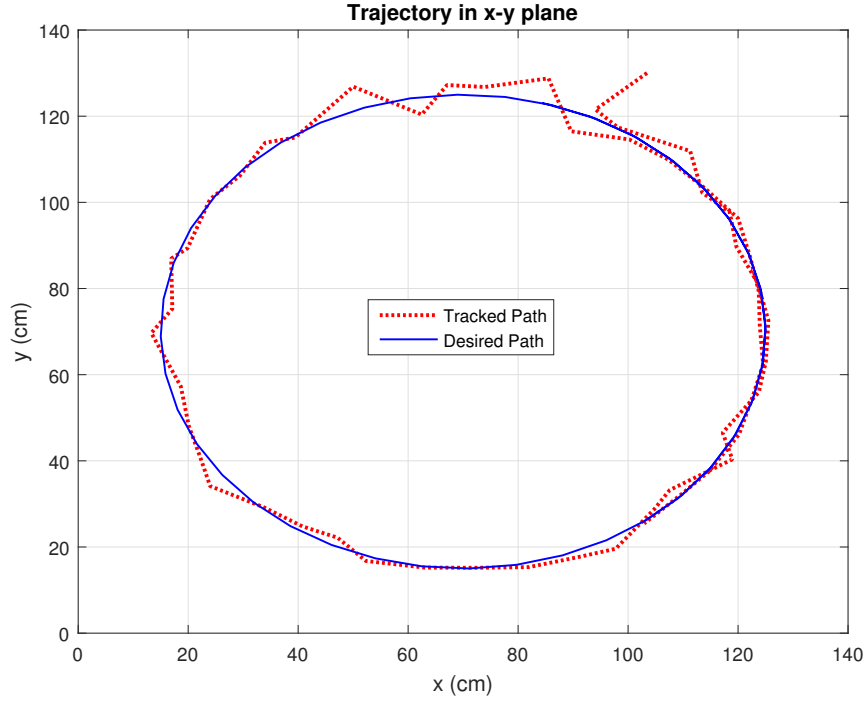


Figure 6.12: Circular trajectory with max. allowable speed

### 6.3 Results and Discussion

The performance of the experimental robot in a circular path with the maximum allowable velocity is recorded and shown in Figure 6.12. The position and orientation variation between actual and desired path is illustrated in Figure 6.13. The maximum deviation along x and y axis are 4.99%, and 16.01% respectively. A detailed error summery is listed in Table 6.4.

Similarly, a critical asymmetric maneuver is created for measuring the position and orientation error between targeted path and actual path is illustrated in Figure 6.14. Also, the position error of this trajectory is presented in the Figure 6.15. An indication of the result from Figure 6.15 is that the max. deviation along y-axis and x-axis movement are 9.60% and 17.96% respectively for the Fuzzy-PI

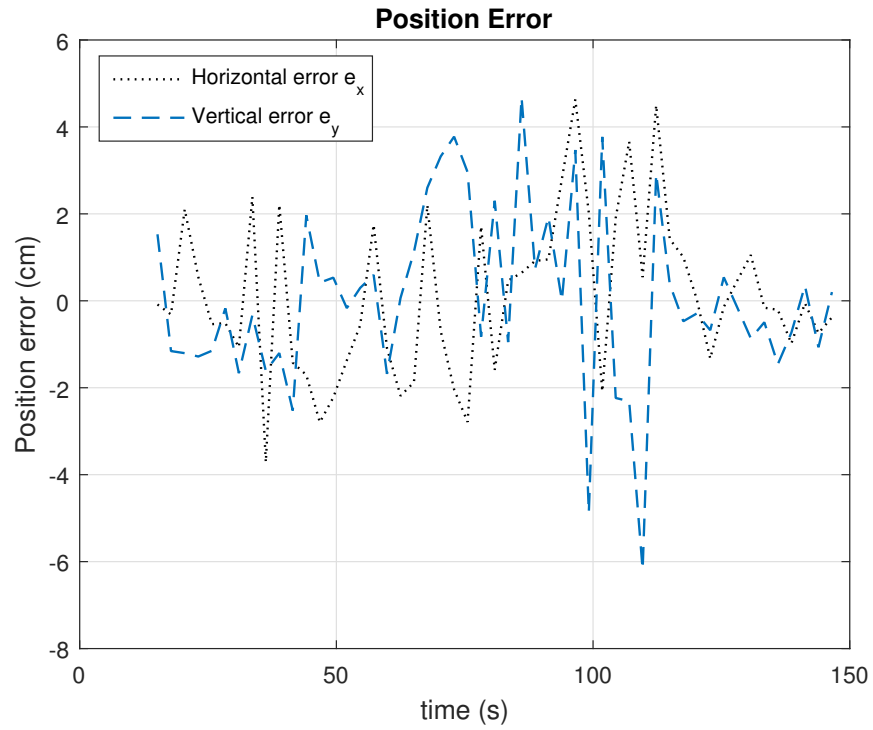


Figure 6.13: Position error comparison

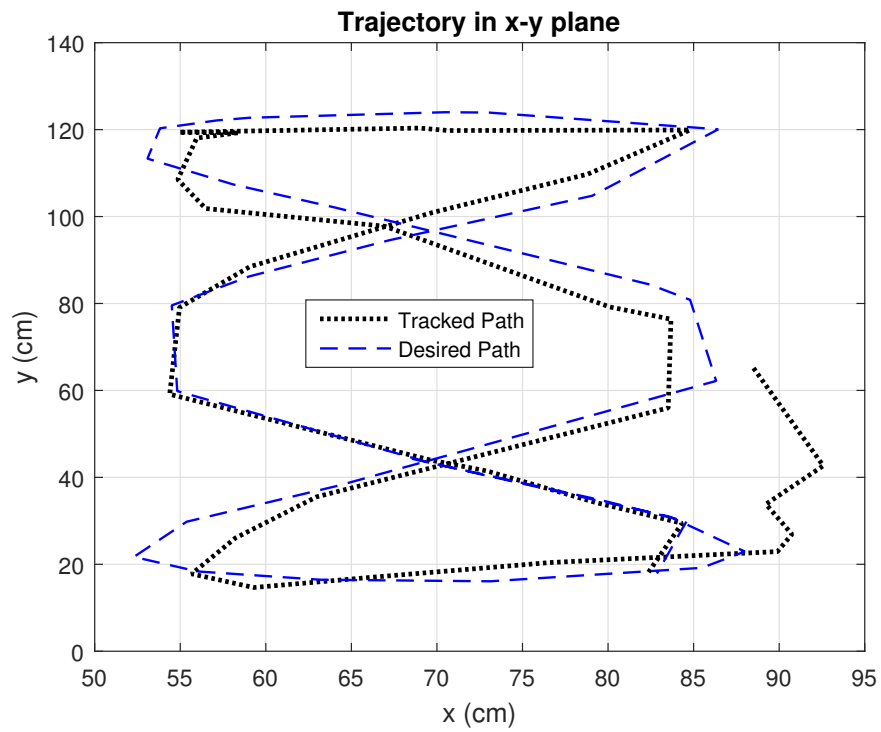


Figure 6.14: DNA trajectory with max. allowable speed

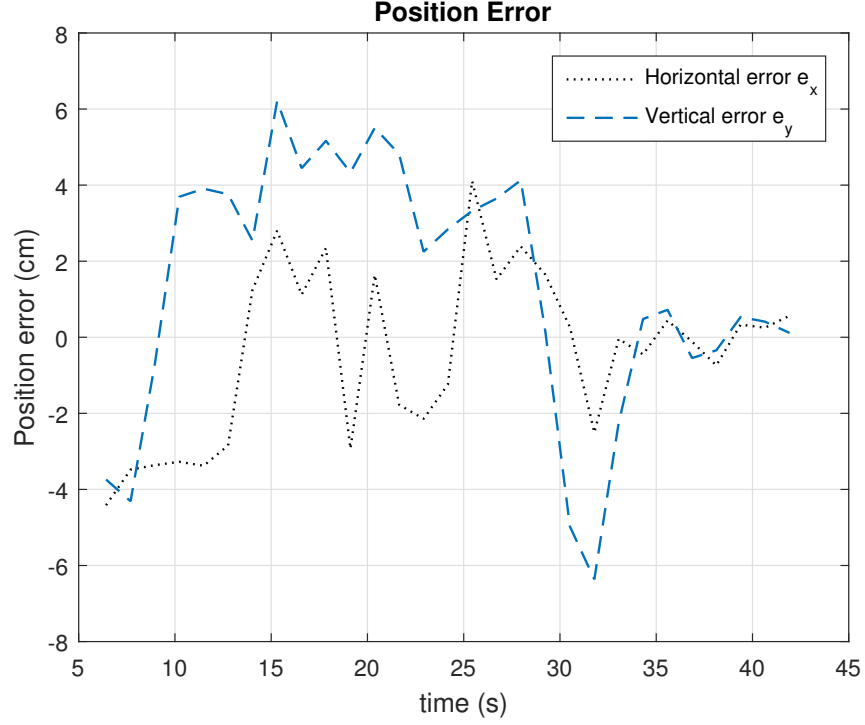


Figure 6.15: Position error comparison

approach with velocity 1 m/s that augmented to 5.17% and 19.66% for a pathway with the maximum allowable velocity. The existence of the Fuzzy logic controller on the maneuver planner diminishes the deviations.

Table 6.4: Error comparison

	Horizontal(%)		Vertical(%)	
	1 m/s	$V_{max}$ m/s	1 m/s	$V_{max}$ m/s
Circular Trajectory	8.08	4.99	11.54	16.01
DNA Trajectory	9.60	5.17	17.96	19.66



## CHAPTER 7

# CONCLUSION

In this research, an easy to use and the simple embedded system is developed for controlling the motion of four wheeled omnidirectional mobile robot using two closed loops structure control system that can successfully read the command from team server and executes the received command to drive the robot accurately and efficiently in an x-y plane. The outer loop controller, a fuzzy adaptive PI calculates the accurate target position and minimize the tracking errors using a vision feedback. Concurrently, an inner loop controller, a discrete time linear quadratic regulator is employed on the robot to obtain the optimal input for the motor drivers that can help to run the robot with exact wheel speed in an energy efficient way. An essential contribution illustrated in this thesis is a performance improvement in the combination of fuzzy tuned PI with LQR controller over classic PI controller. Furthermore, conducted simulation and experiments with and without external load given an optimal outcome that ensure the proposed claim. Implementing data distribution service using publish-subscribe method in

this study, performance can be improve. Also, the coefficients of the PI controller can be optimized in more suitable way using Genetic Algorithm or Particle Swarm Optimization in the further research direction.

# APPENDIX

## Appendix A. Nomenclature

Symbol	Quantity
$E$	Driver's voltage matrix $[E1 \ E2 \ E3 \ E4]^t$
$EM$	Driver's back EMF
$F_t$	Traction force matrix $[f1 \ f2 \ f3 \ f4]$
$F_b$	Force and moment on the robot in the moving frame $[F_{bx} \ F_{by} \ T_z]$
$F_g$	Force and moment on the robot in global coordinates $[F_x \ F_y \ T_z]$
$G$	Geometrical matrix
$J_l$	Wheel's inertia
$J_m$	Driver's rotor inertia
$J$	Robot's inertia
$R_t$	Driver's terminal resistance phase to phase
${}^g_b R$	Rotation matrix
$T_z$	Applied moment on the robot about vertical axis
$X_m$	Local position matrix, $[x_m \ y_m \ \phi]^T$
$X_g$	Global position matrix, $[x_g \ y_g \ \phi]^T$
$\dot{X}_g, \dot{X}_b, V$	Global and local velocity
$\ddot{X}_g, \ddot{X}_b, a$	Global and local acceleration
$k_m$	Driver's torque constant

Symbol	Quantity
$c_m$	Coefficient of damping of the driver
$c_L$	Coefficient of damping of the wheel
$l$	Wheel base
$m$	Robot's mass
$n$	Gear ratio
$r$	Wheel radius
$\theta$	Angle of wheels with respect to local coordinates
$\tau_m$	Drivers torque matrix, $[\tau_{m1} \ \tau_{m2} \ \tau_{m3} \ \tau_{m4}]^T$
$\dot{\phi}$	Angular velocity of the robot
$\omega_m$	Angular velocity matrix of the drivers
$\dot{\omega}_m$	Angular acceleration matrix of the drivers
$\omega_L$	Angular velocity matrix of the wheels
$\dot{\omega}_L$	Angular acceleration matrix of the wheels

## Appendix B.

$$k_1 = J_m + \frac{J_L}{n^2} + \frac{r^2}{n^2}(0.3m + 11.80J)$$

$$k_2 = \frac{r^2}{n^2}(0.03m + 11.8J)$$

$$k_3 = \frac{r^2}{n^2}(-0.25m + 9.1J)$$

$$k_4 = \frac{r^2}{n^2}(-0.06m + 9.1J)$$

$$k_5 = J_m + \frac{J_L}{n^2} + \frac{r^2}{n^2}(0.23m + 7J)$$

$$k_6 = \frac{r^2}{n^2}(0.7m + 7J)$$

$$k_7 = c_m + \frac{c_L}{n^2}; \quad k_8 = 0.3\frac{r^2}{n^2}m; \quad k_9 = 0.02\frac{r^2}{n^2}m; \quad k_{10} = 0.26\frac{r^2}{n^2}m; \quad k_{11} = 0.24\frac{r^2}{n^2}m$$

# REFERENCES

- [1] E. Hashemi, M. G. Jadidi, and N. G. Jadidi, “Model-based pi-fuzzy control of four-wheeled omni-directional mobile robots,” *Robotics and Autonomous Systems*, vol. 59, no. 11, pp. 930–942, 2011.
- [2] T. Bräunl, *Embedded robotics: mobile robot design and applications with embedded systems*. Springer Science & Business Media, 2008.
- [3] M. Barr and A. Massa, *Programming embedded systems: with C and GNU development tools*. ” O’Reilly Media, Inc.”, 2006.
- [4] M. Barr, “Real men program in c,” *DesignCon, IEEE*, 2015.
- [5] C. Ebert and C. Jones, “Embedded software: Facts, figures, and future,” *Computer*, no. 4, pp. 42–52, 2009.
- [6] A. Weitzenfeld, J. Biswas, M. Akar, and K. Sukvichai, “Robocup small-size league: Past, present and future,” in *RoboCup 2014: Robot World Cup XVIII*. Springer, 2015, pp. 611–623.
- [7] L. Iocchi, H. Matsubara, A. Weitzenfeld, and C. Zhou, *RoboCup 2008: Robot Soccer World Cup XII*. Springer, 2009, vol. 5399.

- [8] U. Visser, F. Ribeiro, T. Ohashi, and F. Dellaert, *Robocup 2007: Robot soccer world cup xi*. Springer, 2008, vol. 5001.
- [9] J. A. Gurzoni Jr, M. F. Martins, F. Tonidandel, and R. A. Bianchi, “On the construction of a robocup small size league team,” *Journal of the Brazilian Computer Society*, vol. 17, no. 1, pp. 69–82, 2011.
- [10] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, “Robocup: The robot world cup initiative,” in *Proceedings of the first international conference on Autonomous agents*. ACM, 1997, pp. 340–347.
- [11] L. S. Iliadis, I. Maglogiannis, and H. Papadopoulos, *Artificial Intelligence Applications and Innovations: 8th IFIP WG 12.5 International Conference, AIAI 2012, Halkidiki, Greece, September 27-30, 2012, Proceedings*. Springer, 2012, vol. 381.
- [12] R. Gerndt, D. Seifert, J. Baltes, S. Sadeghnejad, and S. Behnke, “Humanoid robots in soccer—robots versus humans in robocup 2050,” *Robotics & Automation Magazine, IEEE*, vol. 22, no. 3, pp. 147–154, 2015.
- [13] K. Sukvichai, T. Ariyachartphadungkit, and K. Chaiso, “Robot hardware, software, and technologies behind the skuba robot team,” in *RoboCup 2011: Robot Soccer World Cup XV*. Springer, 2012, pp. 13–24.
- [14] M. Dogar, R. A. Knepper, A. Spielberg, C. Choi, H. I. Christensen, and D. Rus, “Multi-scale assembly with robot teams,” *The International Journal of Robotics Research*, vol. 34, no. 13, pp. 1645–1659, 2015.

- [15] R. A. Knepper, T. Layton, J. Romanishin, and D. Rus, “Ikeabot: An autonomous multi-robot coordinated furniture assembly system,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 855–862.
- [16] S. Nouyan, R. Groß, M. Dorigo, M. Bonani, and F. Mondada, “Group transport along a robot chain in a self-organised robot colony.” in *IAS*, 2006, pp. 433–442.
- [17] T. L. Huntsberger, A. Trebi-Ollennu, H. Aghazarian, P. S. Schenker, P. Pirjani, and H. D. Nayar, “Distributed control of multi-robot systems engaged in tightly coupled tasks,” *Autonomous Robots*, vol. 17, no. 1, pp. 79–92, 2004.
- [18] [wiki.robocup.org](http://wiki.robocup.org), “Robocup-ssl wiki page,” 2017.
- [19] S. Zickler, J. Biswas, K. Luo, and M. Veloso, “Cmdragons 2010 team description,” 2010.
- [20] D. HANSELMAN, “Brushless motors: Magnetic design, performance and control,” *Orono: E-Man Press LLC*, 2012.
- [21] G. L. P. B. C. K. G. B. S. S. Bernhard Perun, Andre Ryll, “Team description for robocup 2011,” 2011.
- [22] FreeRTOS, “[www.freertos.org](http://www.freertos.org),” 2017.

- [23] M. A. Sharbafi, A. Azidehak, M. Hoshyari, O. Bakhshandeh, A. A.-M. Babarsad, A. Zareian, D. Esmaeely, A. Ganjali, S. Esmaeelpourfard, S. Ziadloo *et al.*, “Mrl extended team description 2011.”
- [24] C. Röhrig, D. Heß, C. Kirsch, and F. Künemund, “Localization of an omnidirectional transport robot using ieee 802.15. 4a ranging and laser range finder,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.* IEEE, 2010, pp. 3798–3803.
- [25] X. Li and A. Zell, “Motion control of an omnidirectional mobile robot,” in *Informatics in Control, Automation and Robotics.* Springer, 2009, pp. 181–193.
- [26] R. Rojas, “Omnidirectional control,” *Freie Universitat Berlin*, vol. 18, 2005.
- [27] J. Buchanan, S. Csukas, L. Langstaff, R. Strat, and R. Woodworth, “Robo-jackets 2014 team description paper.”
- [28] K. Sukvichai, P. Wasuntapichaikul, and Y. Tipsuwan, “Implementation of torque controller for brushless motors on the omni-directional wheeled mobile robot,” in *ITC-CSCC*, 2010, pp. 19–22.
- [29] K. Sukvichai and P. Wechsuanmanee, “Development of the modified kinematics for a wheeled mobile robot,” in *ITC-CSCC*, 2010, pp. 88–90.
- [30] Z. Chen and S. T. Birchfield, “Qualitative vision-based path following,” *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 749–754, 2009.



- [31] T. Kalmár-Nagy, R. DAndrea, and P. Ganguly, “Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle,” *Robotics and Autonomous Systems*, vol. 46, no. 1, pp. 47–64, 2004.
- [32] O. Purwin and R. DAndrea, “Trajectory generation and control for four wheeled omnidirectional vehicles,” *Robotics and Autonomous Systems*, vol. 54, no. 1, pp. 13–22, 2006.
- [33] Y. Liu, J. J. Zhu, R. L. Williams, and J. Wu, “Omni-directional mobile robot controller based on trajectory linearization,” *Robotics and Autonomous Systems*, vol. 56, no. 5, pp. 461–479, 2008.
- [34] E. Hashemi, M. G. Jadidi, and O. B. Babarsad, “Trajectory planning optimization with dynamic modeling of four wheeled omni-directional mobile robots,” in *Computational Intelligence in Robotics and Automation (CIRA), 2009 IEEE International Symposium on*. IEEE, 2009, pp. 272–277.
- [35] K. Watanabe, Y. Shiraishi, S. G. Tzafestas, J. Tang, and T. Fukuda, “Feedback control of an omnidirectional autonomous platform for mobile service robots,” *Journal of Intelligent and Robotic Systems*, vol. 22, no. 3-4, pp. 315–330, 1998.
- [36] W. Sun, J. van den Berg, and R. Alterovitz, “Stochastic extended lqr for optimization-based motion planning under uncertainty,” *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 437–447, 2016.

- [37] S. I. Han and J. M. Lee, “Balancing and velocity control of a unicycle robot based on the dynamic model,” *IEEE Transactions on Industrial Electronics*, vol. 62, no. 1, pp. 405–413, 2015.
- [38] M. C. Priess, R. Conway, J. Choi, J. M. Popovich, and C. Radcliffe, “Solutions to the inverse lqr problem with application to biological systems analysis,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 770–777, 2015.
- [39] Z. Kovacic and S. Bogdan, *Fuzzy controller design: theory and applications*. CRC press, 2005, vol. 19.
- [40] I. Iancu, *A Mamdani type fuzzy logic controller*. INTECH Open Access Publisher Rijeka, 2012.
- [41] A. Sheikhlari, A. Fakharian, and A. Adhami-Mirhosseini, “Fuzzy adaptive pi control of omni-directional mobile robot,” in *Fuzzy Systems (IFSC), 2013 13th Iranian Conference on*. IEEE, 2013, pp. 1–4.
- [42] A. G. Ram and S. A. Lincoln, “A model reference-based fuzzy adaptive pi controller for non-linear level process system,” *International Journal of Research and Reviews in Applied Sciences*, vol. 4, no. 2, pp. 477–486, 2013.
- [43] M. Nie and W. W. Tan, “Stable adaptive fuzzy pd plus pi controller for nonlinear uncertain systems,” *Fuzzy Sets and Systems*, vol. 179, no. 1, pp. 1–19, 2011.

- [44] F. Abdessemed, K. Benmahammed, and E. Monacelli, “A fuzzy-based reactive controller for a non-holonomic mobile robot,” *Robotics and autonomous Systems*, vol. 47, no. 1, pp. 31–46, 2004.
- [45] L. A. Martinez-Gomez, D. Sotelo, M. Soto, E. Torres, I. Diakite, O. Ponce, G. Rodriguez, and A. Weitzenfeld, “Eagle knights: Small size robocup soccer team,” *VII SBAI/II IEEE LARS. São Luís*, 2005.
- [46] A. S. Conceicao, A. P. Moreira, and P. J. Costa, “Practical approach of modeling and parameters estimation for omnidirectional mobile robots,” *IEEE/ASME transactions on mechatronics*, vol. 14, no. 3, pp. 377–381, 2009.
- [47] D. Meike and L. Ribickis, “Energy efficient use of robotics in the automobile industry,” in *Advanced Robotics (ICAR), 2011 15th International Conference on*. IEEE, 2011, pp. 507–511.
- [48] X. Zeng, P. Yi, and Y. Hong, “Distributed continuous-time algorithm for constrained convex optimizations via nonsmooth analysis approach,” *IEEE Transactions on Automatic Control*, vol. PP, no. 99, pp. 1–1, 2016.
- [49] H.-j. Zhang, J.-w. Gong, Y. Jiang, G.-m. Xiong, and H.-y. Chen, “An iterative linear quadratic regulator based trajectory tracking controller for wheeled mobile robot,” *Journal of Zhejiang University SCIENCE C*, vol. 13, no. 8, pp. 593–600, 2012.
- [50] R. Zhao, Y. Wang, and J. Zhang, “Maximum power point tracking control of the wind energy generation system with direct-driven permanent magnet

- synchronous generators,” *Proceedings of the CSEE*, vol. 29, no. 27, pp. 106–111, 2009.
- [51] Z. Zhong, R. J. Wai, Z. Shao, and M. Xu, “Reachable set estimation and decentralized controller design for large-scale nonlinear systems with time-varying delay and input constraint,” *IEEE Transactions on Fuzzy Systems*, vol. PP, no. 99, pp. 1–1, 2016.
- [52] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.
- [53] D. S. Naidu, *Optimal control systems*. CRC press, 2002.
- [54] A. Taskin and T. Kumbasar, “An open source matlab/simulink toolbox for interval type-2 fuzzy logic systems,” in *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, 2015, pp. 1561–1568.
- [55] K.-S. Tang, K. F. Man, G. Chen, and S. Kwong, “An optimal fuzzy pid controller,” *IEEE Transactions on Industrial Electronics*, vol. 48, no. 4, pp. 757–765, 2001.
- [56] R.-E. Precup and S. Preitl, “Pi-fuzzy controllers for integral plants to ensure robust stability,” *Information Sciences*, vol. 177, no. 20, pp. 4410–4429, 2007.
- [57] K. J. Åström and B. Wittenmark, *Adaptive control*. Courier Corporation, 2013.
- [58] Ivanseidel, “<https://github.com/ivanseidel/arduinothread>,” *GitHub*, 2017.

# Vitae

- Name: Md. Abdullah Al Mamun
- Nationality: Bangladeshi
- Date of Birth: January 8, 1988
- Email: *aamcse@gmail.com*
- Permanent Address: Vill: Chandipur, P.O. Sonka, P.S. Sherpur, Bogra-5800
- MS in Computer Engineering, King Fahd University of Petroleum & Minerals, 2016.
- BS in Computer Science & Engineering, Dhaka University of Engineering & Technology, 2012.