國立中興大學電機工程系

碩士學位論文

四輪全方位導覽機器人之適應性動態移動控制與SoPC實現

Adaptive Dynamic Motion Control and SoPC Implementation of a Four-Wheeled Omnidirectional Tour-Guide Robot

指導教授：蔡清池 博士 Ching-Chih Tsai

研 究 生：吳政叡 Zeng-Ruei Wu

中華民國九十八年七月

國立中興大學電機工程系

碩士學位論文

四輪全方位導覽機器人之適應性動態移動控制與SoPC實現

Adaptive Dynamic Motion Control and SoPC Implementation of a Four-Wheeled Omnidirectional Tour-Guide Robot

指導教授：蔡清池 博士 Ching-Chih Tsai

研 究 生：吳政叡 Zeng-Ruei Wu

中華民國九十八年七月

國立中興大學電機工程學系

碩士學位論文

題目(中文)：四輪全方位導覽機器人之適應性動態移動控制與 SoPC 實現

題目(英文)：Adaptive Dynamic Motion Control and SoPC Implementation of a Four-Wheeled Omnidirectional Tour-Guide Robot

姓名：吳政叡　　學號：79664214

經 口 試 通 過 特 此 證 明

論文指導教授　蔡清池

論文考試委員　李祖聖

黃國勝

蔡清池

中 華 民 國　98　年　7　月　24　日

致　謝

承蒙恩師 蔡清池教授於研究所就讀期間在課業上及研究上悉心指導與諄諄教誨，使得本論文得以順利完成，在此表示誠摯的敬意與謝忱。同時，也感謝口試委員中正大學 黃國勝教授以及成功大學 李祖聖教授撥冗指導，提供寶貴的意見與建議，使得本論文更臻周延與完善。

在課業上及生活上，感謝博士班學長黃旭志、王忍忠、李易餘、廖慶文在專業知識上的熱心指導，以及同窗好友炫全、明翰、衍翔與世泓在學業方面的切磋與砥礪。另外感謝實驗室學弟盈儒、祥雲、逢均與大學部專題學弟岳澧在實驗上面的協助與幫忙。

最後特別感謝父母親、家長以及妹妹長久以來的栽培與照料，使得我生活上與經濟上無後顧之憂。在此尚有未提及的師長與朋友們，也摯上萬分謝意。

願與家人、師長、同學、朋友分享此份畢業的喜悅!

吳 政 叡　　謹致於

國立中興大學電機工程研究所

尖端電控實驗室

中華民國九十八年七月

中文摘要

本論文之研究目的在於發展四輪全方位移動導覽機器人平台控制、ZigBee 定位與系統晶片化的方法與技術。ZigBee 模組用於完成初始全域定位，混合從 ZigBee 和航向估測元件所得到的訊號，可以得到機器人在任何時間任何地點的全域位置。在四個全方位輪呈現九十度的排列方式下，利用非線性適應動態學控制方法實踐全方位移動機器人點穩定度分析、軌跡追蹤實驗。四輪全方位移動導覽機器人的適應動態移動控制器以系統晶片的技術將其實現。文中也提出許多的實驗結果將此機器人的優點加以分析討論，確保以及驗證所提出方法的有效性與優點。

Abstract

This thesis develops methodologies and techniques for ZigBee localization, motion control and SoPC-Based Implementation of a tour-guide robot with a four-wheeled omnidirectional mobile platform. The ZigBee module has been adopted for accomplishing initial global localization. The information obtained from the dead-reckoning unit and ZigBee module is fused together to find the global localization of the robot at any place at any time. A nonlinear adaptive dynamic control method is presented for point stabilization and trajectory tracking of an omnidirectional wheeled mobile robot with four independent driving omnidirectional wheels equally spaced at 90 degrees from one to another. The proposed adaption dynamic motion controller has been implemented into an SoPC chip. The effectiveness and merit of the proposed techniques are well exemplified by conducting several simulations and experiments on an experimental four-wheeled omnidirectional tour-guide robot.

Contents

List of Figures

List of Tables

Nomenclature

$X_\omega Y_\omega$	World coordinate frame
$X_m Y_m$	Moving coordinate frame
M_G	Torque applied to the robot
${}^w_m R$	Transform matrix between the world frame and moving frame
u_i	Voltage applied to the motor
ω_i	Angular velocity of the motor
f_i	Coulomb friction of each wheel
r	Radius of wheel
L	Distance from the centre of mass to each wheel centre
I_w	Moment of inertia of the wheel
I_m	Moment of inertia of the mobile
M	Mobile robot mass
c	Viscous friction factor of the wheel
k	Driving gain factor or the gear ratio

List of Acronyms

AC	Alternating current
API	Application programming interface
ASIC	Application specified integrated circuit
CA	Cellular automata
DA	Digital-to-analog
DC	Direct current
DOF	Degree of freedom
DSP	Digital signal processor
FPGA	Field programmable gate array
IDE	Integrated development environment
IP	Intellectual properties
ISR	Interrupt service routine
LE	Logic element
PDNA	Parallel deoxyribonucleic acid algorithm
PEGA	Parallel elite genetic algorithm
PI	Proportional-integral
SOPC	System on a programmable chip
TCP/IP	Transmission control protocol/internet protocol
VHDL	VHSIC hardware description language

Chapter 1

Introduction

1.1 Introduction

Recently, museum guide robots have been widely deployed in the Europe, Japan and USA. The first museum guide robot, RHINO [1], was deployed in the "Deutsches Museum Bonn" in 1998. MINAVER is the second generation of tour guide robots [2], and has similar functions but better performance. In general, all of the existing guide robots have two major functional modules, namely navigation and interaction. However, how to implement these two modules differs from one guide robot to another. Figure 1.1 ~ 1.7 show the photographs of several tour guide robots.

Figure 1.1. Rhino robot.

Figure 1.2. Minerva robot.

Figure 1.3. Three entertainment robots developed by IPA.

Figure 1.4. RoboX robot.

Figure 1.5. Jinny robot.

Figure 1.6. NSTM robots.

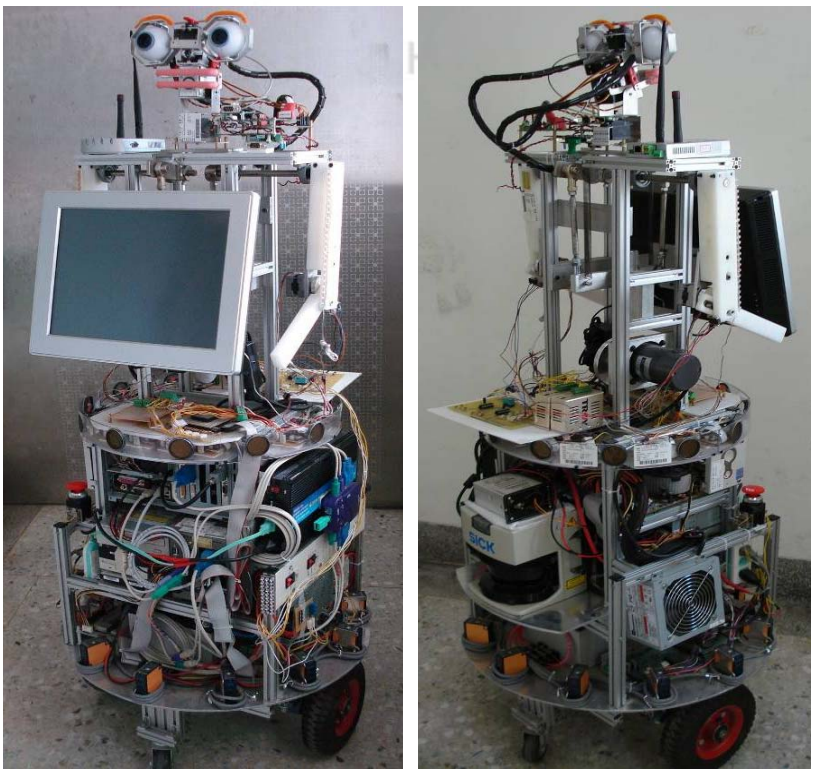
Figure 1.7. Photograph of tour-guide robot named UPITOR.

In addition to theses above-mentioned tour-guide robots developed in USA, Germany, Japan, Korea, Swiss and Taiwan, below are the histories of the tour-guide

mobile robot that were built in AECL, National Chung Hsing University. Figures 1.8 ~ 1.11 shows its evolution from the first-generation to the third-generation. The first-generation of the omnidirectional mobile robot was equipped with a differential-driving mechanism and three sensors: laser scanner, ultrasonic arrays, and infrared ray sensors. To solve the problem of the mobile robot localization and human robot interaction, Wang [3] considered methodologies and techniques for navigation and human-robot interaction of a tour-guide robot with an improved system configuration which was regarded as the second generation of the tour-guide robot. In the second-generation mobile robot, a least-squares algorithm was adopted to solve the RFID localization problem. A new tour-guide robot, called the third generation, based on a four-wheeled omnidirectional mobile platform was developed by Feng[4], This is because the differential mobile base was constrained by its nonhononomic property, thus resulting in a less flexible motion control in comparison with the omnidirectional mobile platform. Moreover, the four-wheeled omnidirectional mobile platform is able to carry more payloads, load more weight, and is very useful for dynamic, cluttered, and crowded environments. The omnidirectional mobile robot is indeed a holonomic robot that consists of omnidirectional wheel assembly constructed by several orthogonal-wheels inside the wheel, thereby enabling the robot to attain any pose and easy maneuver due to its small turning radius.

(a) (b)

Figure 1.8. Tour-Guide Robot Generation 1.

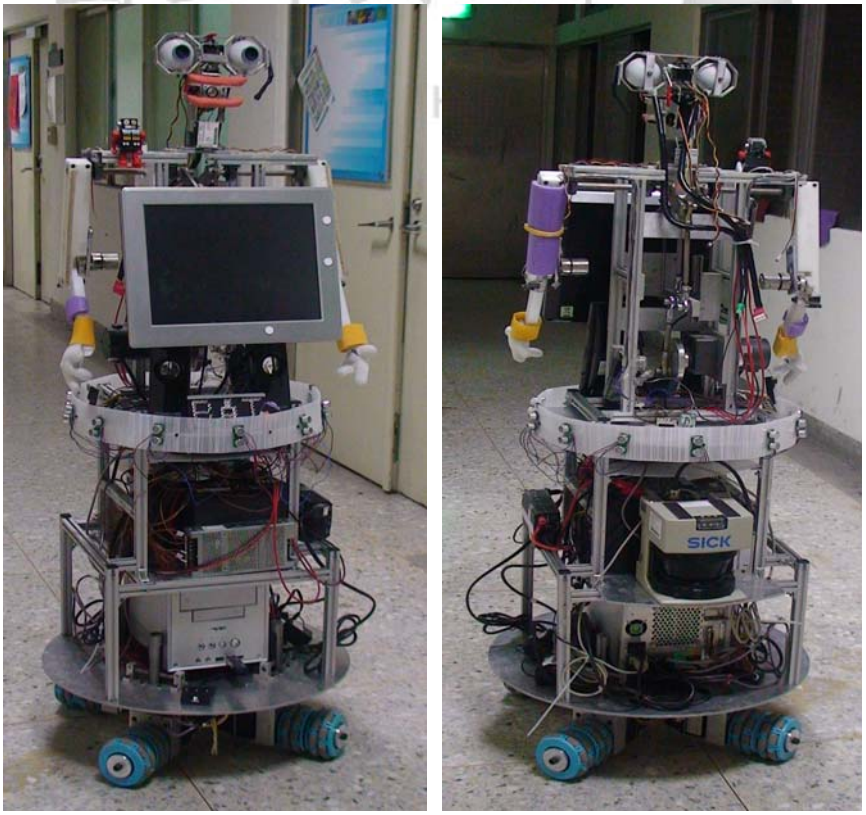


(a) (b)

Figure 1.9. Tour-Guide Robot Generation 2.

(a) (b)

Figure 1.10. Improved Tour-Guide Robot Generation 2.



(a) (b)

Figure 1.11. Tour-Guide Robot Generation 3.

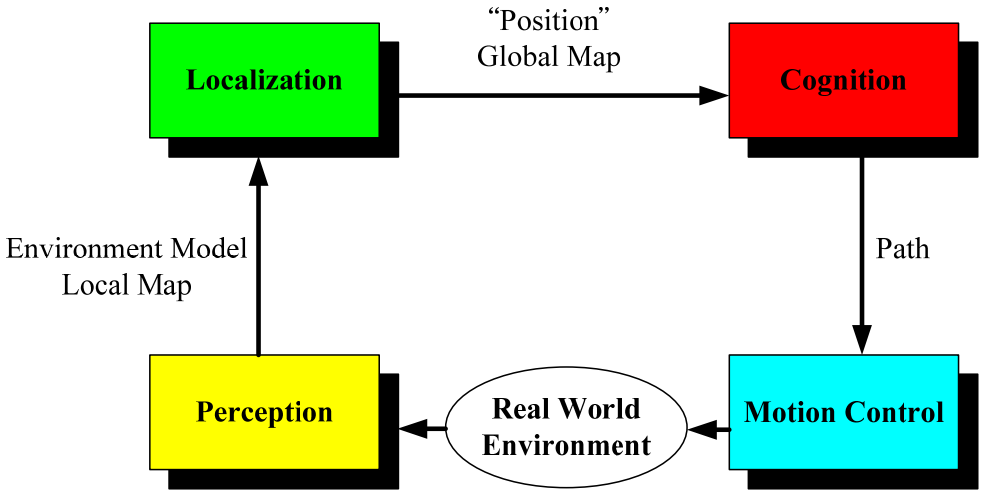


Figure 1.12. The general control structure of robots.

Generally speaking, the general control structure of the tour-guide robots, which have been regarded as one of autonomous wheeled mobile robots, includes four main modules: sensing and perception, localization and mapping, cognition and planning, and motion control, as proposed in [5] in some detail. Figure 1.12 depicts the general control structure. Although the four main modules may be included in any kind of tour-guide robot, this thesis is devoted to investigating some important issues for localization and motion control.

By looking into mobile platform structures of these tour-guide robots, most of them are commonly composed of two differential-driving wheeled mobility mechanism. Such mobile platforms have been shown inexpensive to production, but not easier to control or steer in comparison with other omnidirectional configuration. Therefore, this thesis attempts to design the four wheeled omnidirectional moving base to improve the driving characteristics. This kind of mobile mechanism is much more popular today because the omnidirectional motion base would move in the crowded environments and have the unique property to attain any position and

orientation (pose) simultaneously. Over the past decade, the omnidirectional mobile robots [6-7] have been proven useful and effective in Robocup competition. Therefore, with this motion mechanism, the tour-guide robot would achieve excellent mobility during tour-guide missions.

Localization is one of the most important modules to navigate tour-guide robots inside their working environments. Since the museum is often a large public space, allowing a great amount of people for visiting at the same tile slots, the localization schemes for such environments somewhat differ from home environments. On the other hand, visitors like children often stand around the robots so that the robot cannot use the on-board sensors to localize itself unless the ceiling navigation is used. An alternative to navigate tour-guide robot is to use wireless localization devices, such as WiMAX, Wi-Fi, RFID, ZigBee and so on. With these wireless localization devices along with the dead-reckoning method, global localization of this kind of robot would be achieved with some level of accuracy.

1.2 Literature Review

1.2.1 Related Work for Wireless Communication Module- ZigBee

ZigBee [8] is a specification for a suite of high level communication protocols using small, low-power digital radios based on the IEEE 802.15.4-2003 standard for wireless personal area networks (WPANs). ZigBee is targeted at radio-frequency (RF) applications that require a low data rate, long battery life, and secure networking. Based on the results from the experiments Pinedo-Frausto *et al.* [9] have performed, and from other aspects they have analyzed, they can situate ZigBee as a protocol well suited for applications in classes 3 to 5 according to the usage classes defined by ISA. But it would not be adequate for emergency applications or for closed loop control applications. F. Sottile *et al.* [10] proposed a complete system for nodes localization in

a Wireless Sensor Network (WSN) based on the ZigBee standard. The system includes a real-time location engine, which adopts a Received Signal Strength Indicator (RSSI)-based localization algorithm, and three tools, namely an Environment Description Tool (EDT), a Channel Modeling Tool (CMT) and a Network Planning Tool (NPT), which enable efficient deployment and accurate operation.

1.2.2 Related Work for Four-wheeled Mobile Platform

Omnidirectional mobile robots can achieve 3 DOF motion on a two-dimensional flat terrain due to the unique feature that such robots can move in an arbitrary direction without changing the direction of the wheels. To date, a variety of omnidirectional mobile robots have been proposed by several researchers; some popular examples among them are universal wheels [11-12], ball wheels [13] and off-centered wheels [14]. The omnidirectional mobile robots using omnidirectional wheels usually have 3 or 4 wheels. The three-wheeled omnidirectional mobile robots can be designed by driving 3 independent actuators [15,16], but they may have stability problem due to the triangular contact area with the ground, especially when traveling on a ramp with the high center of gravity owing to the payload they carry. It is desirable, therefore, that four-wheeled vehicles be used when stability is of great concern [17]. However, independent drive of four wheels creates one extra DOF.

Four-wheeled omnidirectional mobile robots have been investigated as well. Muir and Neuman [18] constructed a four-wheeled omnidirectional robot with Mecanum wheels, called Uranus, but they did not propose its dynamic controller. Byun *et al.* [19] constructed a four-wheeled omnidirectional mobile robot with a variable wheel arrangement mechanism. Wilson *et al.* [20] presented a dynamic model and a simple PD control for a redundant omnidirectional RoboCup goalie; however, they have not

show the stability of the closed-loop system. Haung *et al.* [21] described and implemented a kinematic speed controller of a four-wheel omnidirectional mobile robot. Purwin and Andrea [22] established an optimal trajectory planning approach for a four-wheel omnidirectional mobile robot. A T-S fuzzy path controller design for a four-wheeled omnidirectional mobile robot was constructed in [23]. Li *et al.* [24] developed a dynamic model and a fully fuzzy control for a four-wheeled omnidirectional surveillance robot. However, on the basis of the aforementioned studies, a fully dynamic model incorporating frictions and dynamic effects, and its corresponding dynamic controller using backstepping have not been addressed in detail yet.

1.2.3 Related Work for SoPC Implementation

SoPC technology is becoming more prevalent as a solution for the implementation of embedded computing systems. Campo *et al.* [25] presented the eefficient hardware/software implementation of an adaptive neuro-fuzzy system. The adaptive system was implemented in FPGA chip using hardware/software co-design approach and SoPC technology. Kung *et al.* [26] introduced a FPGA-based speed control IC for PMSM drive with adaptive fuzzy control law executed by a RISC soft-core processor embedded in the FPGA chip. Solano *et al.* [27] proposed a FPGA implementation of embedded fuzzy controllers for robotic applications. Some hardware circuits IP were developed and an embedded processor was applied for the fuzzy controller. The design and implementation of modular FPGA-based PID controllers was proposed in [28], in which the useful hardware circuits for PID controllers were presented in FPGA chip. The SoPC approach is shown more powerful than the conventional DSP-based designs [29] or hardware-oriented FPGA-based designs [30-32]. With the advantages of SoPC, these kind of embedded

system designs [33] are becoming more popular not only in robotics studies but also in many disciplines.

1.3 Motivation and Objective

Although the RFID system has been used for the posture estimation problem of mobile robots for years, it has suffered from some disadvantages; for example, active tags are more expensive than passive tags, and the battery life is too short to use on the mobile robots. Hence, a better solution should take place of the RFID system. ZigBee is an adequate choice because of its special features of low-cost, low-power, and long distance wireless sensor networks. To find out initial pose of the robot, the ZigBee system is firstly calibrated using a least-square method, the calibration curves are adopted to convert the measured RSSI values into the corresponding distances, and the other least-square approach is finally employed to obtain global initial location of the robot with some allowable errors. Once the ZigBee localization module for the robot has been established, the outcomes of the module should be fused with data the dead-reckoning unit, in order to get more reliable and precise position estimates of the robot.

On the basis of the previous literature surveys, this kind of tracking and stabilization control problem for four-wheeled omnidirectional mobile platform of the tour-guide robot with dynamic effect and uncertainties still deserves further study. This study is not only for industrial applications to material handling, but also for civilian applications to several kinds of autonomous wheeled service robots. Hence, the thesis will employ the adaptive backstepping approach to constructing an adaptive dynamic motion controller to achieve stabilization and trajectory tracking for the four-wheeled omnidirectional mobile platform incorporating with the dynamic effect and uncertainties

Although the adaptive dynamic motion controller can be realized by a small-scale personal computer, this kind of controller is too bulky, expansive and of high power consumption. To circumvent this shortcoming, the SoPC technology will be employed to implement such a controller in real time so as to meet industrial and commercial requirements.

1.4 Main Contributions

This thesis aims to develop methodologies and techniques for localization, dynamic control and SoPC implementation of the four-wheeled tour-guide robot executing tasks in indoor environments. Significant efforts have been paid to improve the functions of the omnidirectional robot, and make the robot more useful in its use on museum exhibition. The following are main contributions of the robot. First, the ZigBee localization module is applied to the tour-guide robot instead of the previous RFID localization module, and a complete localization module is proposed by fusing the ZigBee location module and a dead-reckoning unit. . Second, an adaptive dynamic control law is presented for both stabilization and trajectory tracking, and its closed-loop stability is ensured by Lyapunov stability theory. Third, the SoPC-based adaptive motion controller is constructed by incorporating with soft embedded processor and application IPs.

1.5 Organization of the Thesis

The remaining chapters of the thesis are organized as follows. Chapter 2 describes system structure and ZigBee localization module of the tour-guide robot, and Chapter 3 elaborates introduces the design, simulation and experiments of the adaptive dynamic motion controller for various motions, such as stabilization and trajectory tracking. The control law is synthesized by adaptive backstepping. In Chapter 4, we

describe the SoPC implementation of the proposed embedded adaptive controller for the four-wheeled mobile robot. Chapter 5 summarizes the thesis and presents a few recommendations for future work.

Chapter 2

System Structure and ZigBee Localization

2.1 Introduction

This chapter is aimed to describe the system structure and ZigBee localization module of the tour-guide robot. Many kinds of tour-guide robots have been developed and tested in many public institutions. As revealed in the literature, most of them adopted a differential driving mechanism which has two driving wheels and one or two caster wheels, thus generating nonholonomic behaviour. The nonholonomic property means that this kind of tour-guide robot must move along the perpendicular direction of the two driving wheels, namely that the robot cannot have lateral motions at any time. Such a property may cause some difficulties for tour-guide mission. For example, tour-guide robots may require omnidirectional motions during exhibition explanation, in order to give clearer elucidation to visitors; they may get a worse motion condition in a smaller exhibition space. To circumvent these difficulties and attain agile manoeuvres, a four-wheeled omnidirectional mobile platform has been used to replace the differential-driving mobile one. Hence, this chapter will be devoted to constructing an improved four-wheeled mobile platform.

On the other hand, localization has been considered as one of the most important functions to navigate tour-guide robots. Localization techniques can be divided into two types, local and global. Local localization aims to use external observations to compensate for odometer and dead-reckoning unit, while global localization is dedicated to find global position and orientation of the robots by assuming that they are kidnapped to some unknown environments. As mentioned in Chapter 1, many well-known tour-guide robots have their particular global localization techniques by fusing encoders, mounted on the driving wheels, and other external sensors, including

camera, laser scanner, RFID, and etc. Since the wireless ZigBee communication module has been regarded as a power means to achieve people tracking, object tracking, and wireless sensing networking, this kind of ZigBee communication module can be used to find the positions and orientations of mobile robots by triangulation techniques. Due to the advantages of low power consumptions and long distance communication, higher noise immunity in comparison with RFID, this ZigBee communication module will be adopted in the chapter to fuse with a dead-reckoning unit for achieving global localization.

The rest of the chapter is organized as follows. Section 2.2 briefly describes the system description of the tour-guide robot. Section 2.3 presents the ZigBee localization subsystem including ZigBee protocol and applications, advantage and localization algorithm, and the experimental setup, results and discussion. Section 2.4 presents dead-reckoning and signal fusion. Finally, the conclusions of the chapter are stated in Section 2.5.

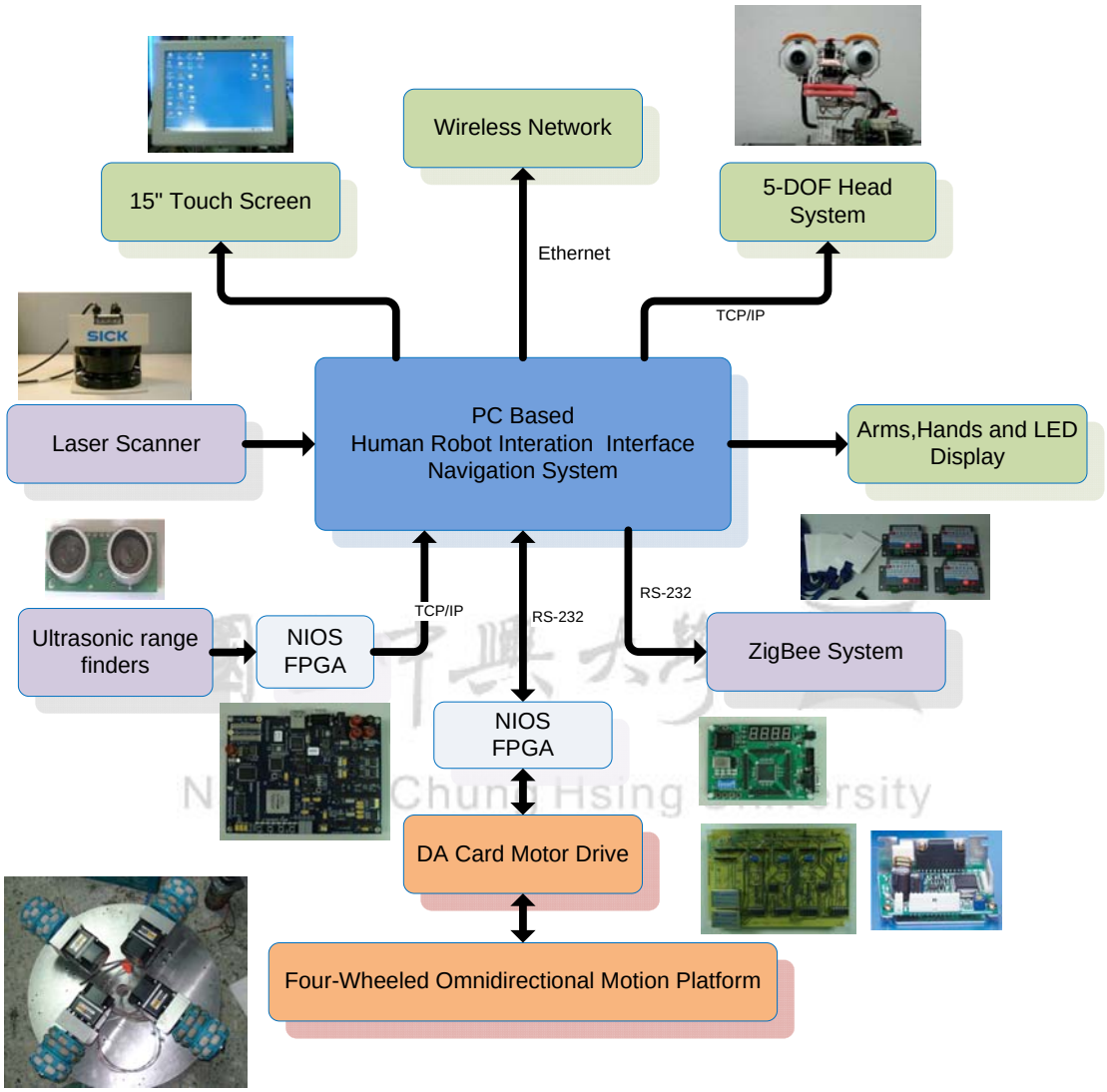


Figure 2.1. Block diagram of the overall system structure.

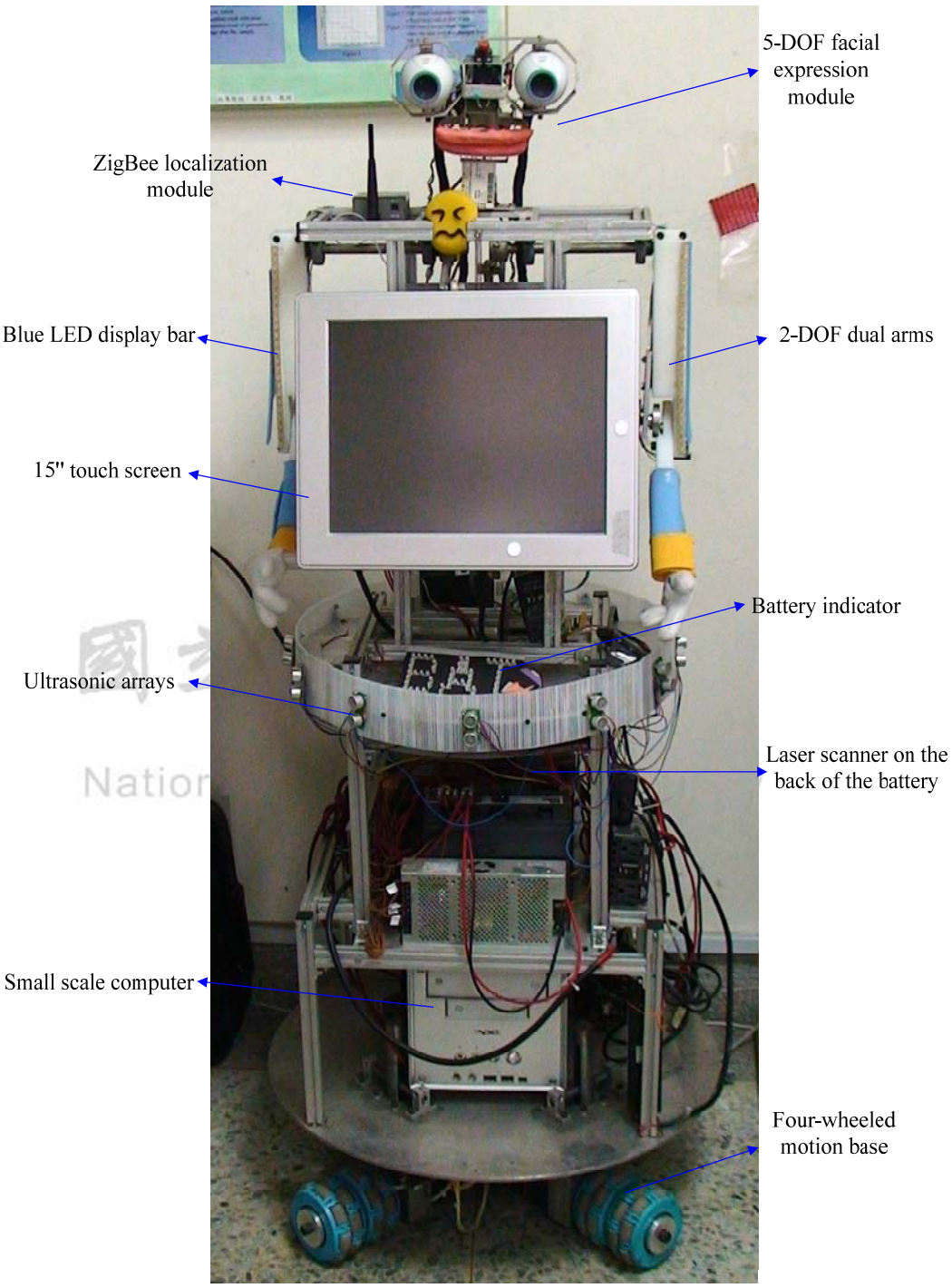


Figure 2.2. Physical structure of the experimental tour-guide robot.

2.2 System Structure and Mobile Platform of the Tour-guide Robot

2.2.1 Description of the System Structure of the Tour-guide Robot

Figure 2.1 shows the block diagram of the overall system structure. Such a robot system is equipped with an embedded personal computer, a ZigBee localization module, a laser scanner, twelve ultrasonic ranging modules, a FPGA-based motion controller implemented by a Stratix II edition Nios development board, a human-robot interaction module, and an four-wheeled omnidirectional mobile platform. The radius of the mobile robot is 25 cm; its height and weight are respectively 150 cm and 78kg. The physical configuration of the tour-guide robot is shown in Figure 2.2**.**

2.2.2 Description of the Four-Wheeled Omnidirectional Platform

The omnidirectional wheels used in this thesis are made by KORNYLAK Corporation. Such a mobile platform is equipped with four DC24V brushless Oriental servomotors modules including drivers (AXHM450k-10 and AXHD50K). The overall mobile motion control system consists of four brushless servomotors with four photo encoders, an AD/DA card made by DAC0800*4, a dual processer embedded personal computer, a FPGA-based motion controller and I/O port made by the Stratix II edition Nios development board. Via the RS-232 interface, the dual processer personal computer sends serial signals to the I/O port in FPGA, and communicates with the ZigBee Module and the laser scanner. The FPGA-based motion controller sends speed commands via the AD/DA card to control the servomotors. Four driving wheels are independently driven by four DC24V brushless servomotors with four mounted encoders outputting the signals of 300 pulses per revolution. Because the torque of motors is insufficient to steer the tour-guided robot, the gear ratio of each gearbox is changed from 10：1 to 20：1, in order to attain required torque. The physical photographs of the old and new motors are compared in figure 2.3. Figure 2.4 displays the block diagram of the motion control system for the mobile platform.

Figure 2.3. Physical photographs of the old and new motors.

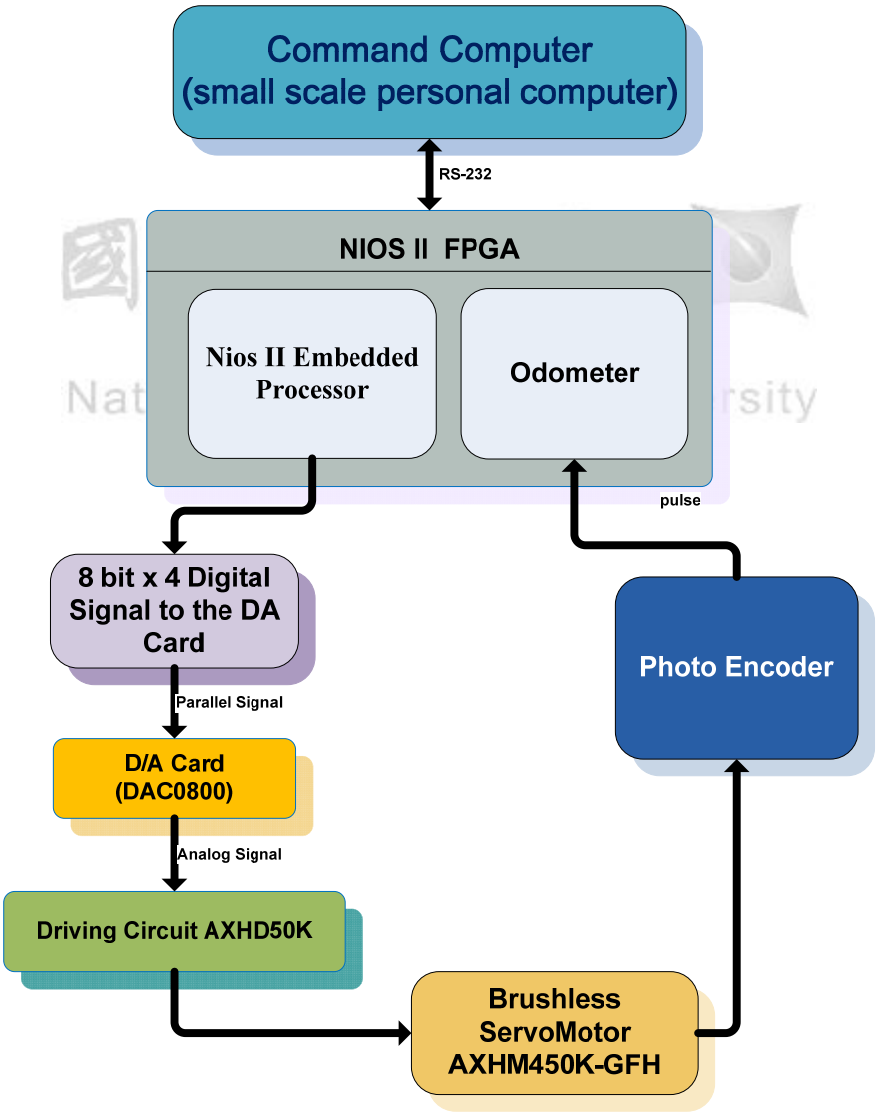


Figure 2.4. Block diagram of the motion control system for the mobile platform.

2.3 ZigBee Localization Subsystem

2.3.1 Introduction: ZigBee Protocol and Applications, ZigBee Advantages

ZigBee is a wireless technology developed as an open global standard to address the unique needs of low-cost, low-power, wireless sensor networks. The standard has full advantage of the IEEE 802.15.4 physical radio specification and operates in unlicensed bands worldwide at the following frequencies: 2.400–2.484 GHz, 902-928 MHz and 868.0–868.6 MHz. The 802.15.4 specification was developed at the Institute of Electrical and Electronics Engineers (IEEE). The specification is a packet-based radio protocol that meets the needs of low-cost, battery-operated devices. The protocol allows devices to intercommunicate and be powered by batteries that last years instead of hours.

The ZigBee protocol was engineered by the ZigBee Alliance, a non-profit consortium of leading semiconductor manufacturers, technology providers, OEMs and end-users worldwide. The protocol was designed to provide OEMs and integrators with an easy-to-use wireless data solution characterized by low-power consumption, support for multiple network structures and secure connections. ZigBee enables broad-based deployment of wireless networks with low-cost, low-power solutions. It provides the ability to run for years on inexpensive batteries for a host of monitoring applications: Lighting controls, AMR (Automatic Meter Reading), smoke and CO detectors, wireless telemetry, HVAC control, heating control, home security, environmental controls, drapery and shade controls, etc.

The ZigBee protocol was designed to carry data through the hostile RF environments that routinely exist in commercial and industrial applications. There are some reasons why we change RFID to ZigBee,

- Low duty cycle - Provides long battery life

- Low latency
- Support for multiple network topologies: Static, dynamic, star and mesh
- Direct Sequence Spread Spectrum (DSSS)
- Up to 65,000 nodes on a network
- 128-bit AES encryption – Provides secure connections between devices
- Collision avoidance
- Link quality indication
- Clear channel assessment
- Retries and acknowledgements
- Support for guaranteed time slots and packet freshness

Table 2.1 Comparison of various devices

Standard	**ZigBee 802.15.4**	**Wi-Fi 802.11b**	**Bluetooth 802.15.1**
Transmission Range(meters)	1 - 100	1 - 100	1 - 10
Battery Life(days)	100 - 1000	0.5 – 5.0	1 - 7
Network Size(# of nodes)	>64000	32	7
Application	Monitoring & Control	Web, Email, Video	Cable Replacement
Stack Size(KB)	4 - 32	1000	250
Throughput(kb/s)	20 - 250	11000	720

Table 2.1 shows the comparison of various devices. The ZigBee specification provides a security toolbox approach to ensuring reliable and secure networks. Access control lists, packet freshness timers and 128-bit encryption based on the NIST Certified Advanced Encryption Standard (AES) help protect transmitted data.

2.3.2 Mesh Networks

A key component of the ZigBee protocol is the ability to support mesh networks. In a mesh network, nodes are interconnected with other nodes so that at least two pathways connect each node. Connections between nodes are dynamically updated and optimized in difficult conditions. In some cases, a partial mesh network is established with some of the nodes only connected to one other node.

Mesh networks are decentralized in nature; each node is self-routing and able to connect to other nodes as needed. The characteristics of mesh topology and ad-hoc routing provide greater stability in changing conditions or failure at single nodes.

In our experimental setup, the ZigBee readers are used to receive the RSSI (Received Signal Strength Indication) of the tag on the robot. Then a calibration method is used to convert the RSSI data into the corresponding distances for the active RFID system. These distance data are then employed to compute the position and orientation approach for global localization of the robot by the least-squares method. Figure 2.5 displays the physical picture of the ZigBee readers and the tags. Figure 2.6 introduces application diagram of the ZigBee operation system. Table 2.2 illustrates the specifications of the ZigBee module.

Figure 2.5. Photograph of the ZigBee readers and the tags.

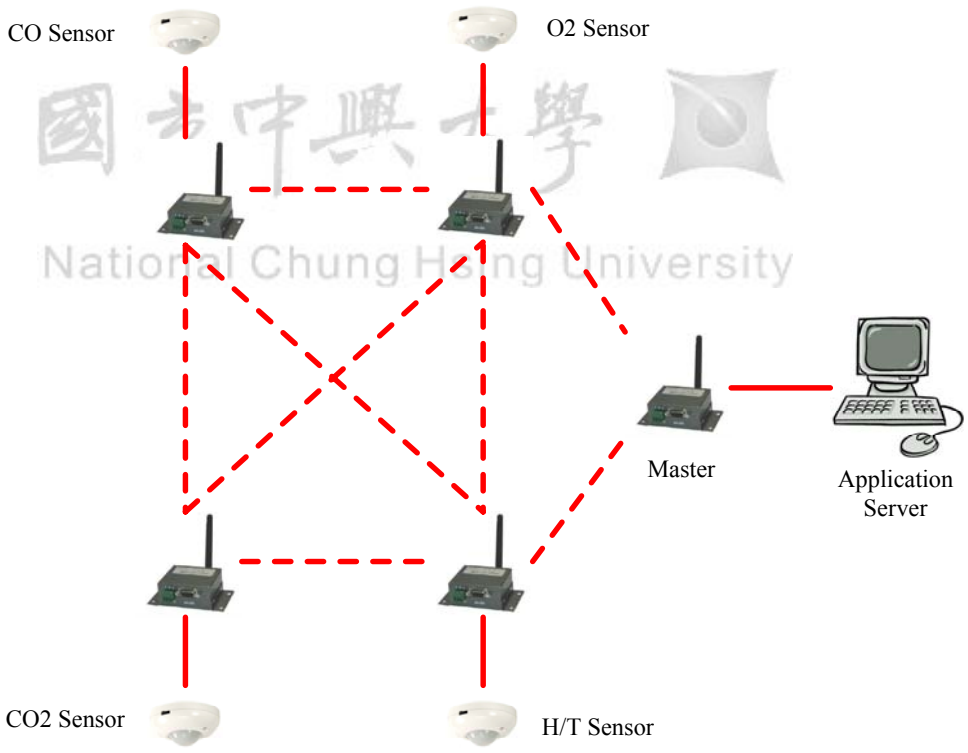


Figure 2.6. Application diagram of the ZigBee system.

Table 2.2 Specifications of the ZigBee module.

	Parameter	Master	Reader	Tag
Radio frequency	**Frequency**	2.4GHz	2.4GHz	2.4GHz
	Max Rate	250Kbps	250Kbps	250Kbps
	Sensitivity	-94dBm	-94dBm	-94dBm
	Trans Range	350m LOS	350m LOS	100m LOS
	Channel	16(5MHz)	16(5MHz)	16(5MHz)
	Launch Power	-15 ~ +10dBm	-15 ~ +10dBm	-25 ~ +0dBm
	Data Encryption	128-bit AES	128-bit AES	128-bit AES
	Antenna	Internal	External	Internal
Power consumption	**TX**	<100mA	<100mA	<37mA
	RX	<43mA	<43mA	<43mA
	Sleep	40uA	40uA	40uA
Physical parameter	**Size(mm)**	125*90*25	62*54*28	68*40*17
	Operating Temp	-20 ~ +70 ˚C	-20 ~ +70 ˚C	-20 ~ +70 ˚C
	Humidity	10% ~ 90%	10% ~ 90%	10% ~ 90%

2.3.3 Localization Algorithm

2.3.3.1 Global Localization Method Using RSSI

The proposed global localization method is based on the RSSI measurements between the tags and the reader, and the information of the given position of these

active tags. It is known that the RSSI measurements are corrupted by several factors, such as multi-path effect, environment, and the material inside the wave propagation path. Although the RSSI readings are not reliable, it is confirmed that the RSSI values in a specified environment will obey a certain model or a relationship. Inspired by the concept, this section will attempt to utilize the wave propagation theory and empirical experience to find an approximate calibration curve or model relating to the RSSI values and the corresponding distances. Once the calibration curve or model has been obtained in some accuracy level, we will use the distance information and a least-square method to calculate the position of robot. This method installs one tag on robot to transmit RSSI signals in the environment, receives RSSI data via the four readers, and then utilizes a least-square method to calculate the robot position. Worthy of mention is that in this global localization procedure, the calibration model must be done for each tag at an actual environment, and the RSSI model for each tag should be re-established at different environments.

2.3.3.2 Calibration of the ZigBee System

This subsection proposes a calibration method transforming the RSSI values from ZigBee tags into their corresponding distance. These RSSI values vary with the environments. In the follows illustrates the calibration method. For the m-th tag, the mathematical relation between the RSSI and the distance is expressed by

$$\mathrm{RSSI}_i = \frac{\mathrm{K}}{d_i^{\,n}}, \quad i = 1, \cdots, m. \tag{2.1}$$

where the parameters K and n are two unknown measurements. In order to find the parameters K and n, one takes the 10-based logarithmic operation for (2.1)

$$\log_{10} \mathrm{RSSI}_i = \log_{10} \mathrm{K} - n \cdot \log_{10} d_i = \mathrm{p} - \mathrm{q} \cdot \log_{10} d_i$$

which can be rewritten as

$$\begin{bmatrix}1 & -\log_{10} d_i\end{bmatrix}\begin{bmatrix}\mathrm{p}\\ \mathrm{q}\end{bmatrix} = \log_{10} \mathrm{RSSI}_i$$

where $\mathrm{p}=\log_{10}\mathrm{K}$ and $\mathrm{q} = n$. Given m pairs of the RSSI and distance measurement, (RSSI_i, d_i), i=1, 2, …, m, one obtains the following matrix equation

$$\begin{bmatrix}1 & -\log_{10} d_1\\ 1 & -\log_{10} d_2\\ & \vdots\\ 1 & -\log_{10} d_m\end{bmatrix}\begin{bmatrix}\mathrm{p}\\ \mathrm{q}\end{bmatrix} = \begin{bmatrix}\log_{10} \mathrm{RSSI}_1\\ \log_{10} \mathrm{RSSI}_2\\ \vdots\\ \log_{10} \mathrm{RSSI}_m\end{bmatrix} \quad \Rightarrow \quad \mathrm{AX} = \mathrm{B} \tag{2.2}$$

where

$$\mathrm{A}=\begin{bmatrix}1 & -\log_{10} d_1\\ 1 & -\log_{10} d_2\\ & \vdots\\ 1 & -\log_{10} d_m\end{bmatrix}, \quad B = \begin{bmatrix}\log_{10} \mathrm{RSSI}_1\\ \log_{10} \mathrm{RSSI}_2\\ \vdots\\ \log_{10} \mathrm{RSSI}_m\end{bmatrix}$$

The parameters p and q are solved via the least-square method

$$\mathrm{X}=(\mathrm{A}^{\mathrm{T}}\mathrm{A})^{-1}\mathrm{A}^{\mathrm{T}}\mathrm{B} = \begin{bmatrix}\mathrm{p}\\ \mathrm{q}\end{bmatrix} \tag{2.3}$$

which implies that $\mathrm{K}=10^{\mathrm{p}}$, $n=\mathrm{q}$.

The calibration procedure to find the two parameters K and n will be repeated for each tag until all the tags are calibrated.

2.3.3.3 Robot Position Estimation Using Least Square Method

This subsection aims to use the calibration models to obtain the distances between all the installed readers and the tag, in order to achieve global localization of the tour-guide robot. The basic idea to do so is to put m readers at known positions respectively given by $\mathrm{T}_1 = (x_1, y_1, z_1)^T$, $\mathrm{T}_2 = (x_2, y_2, z_2)^T$, $\cdots$, $\mathrm{T}_m = (x_m, y_m, z_m)^T$, to measure the RSSI data from the tag, and to convert the RSSI data into the

distances. Figure 2.7 shows the physical configuration of the proposed ZigBee localization system where $(x, y, z, \theta)^T$ represents the posture of robot.

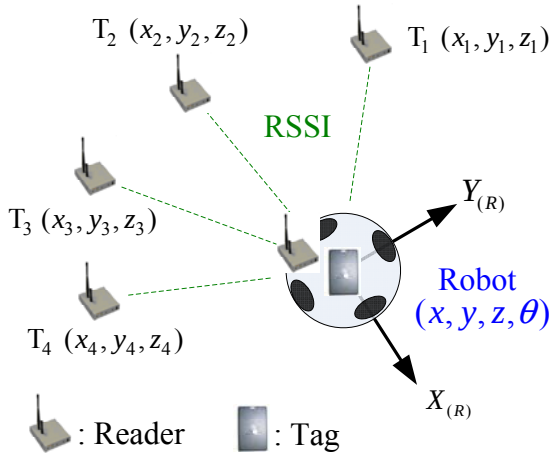


Figure 2.7. Physical configuration of the proposed ZigBee localization system

From the calibration model mentioned in section 3.2.2.1, the distances between the robot and the readers can be computed. Let $d_1, d_2, \cdots d_m$ denote the distances from the robot to the readers. Therefore, the following subsequent equation (2.4) can be obtained.

$$\begin{cases} (x_1 - x)^2 + (y_1 - y)^2 + (z_1 - z)^2 = d_1^2 \\ (x_2 - x)^2 + (y_2 - y)^2 + (z_2 - z)^2 = d_2^2 \\ \vdots \\ (x_m - x)^2 + (y_m - y)^2 + (z_m - z)^2 = d_m^2 \end{cases} \tag{2.4}$$

After some algebraic manipulation, (2.4) becomes

$$\begin{bmatrix} 2(x_1-x_2) & 2(y_1-y_2) & 2(z_1-z_2) \\ \vdots & & \\ 2(x_1-x_i) & 2(y_1-y_i) & 2(z_1-z_i) \\ \vdots & & \\ 2(x_1-x_m) & 2(y_1-y_m) & 2(z_1-z_m) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} d_2^2 - d_1^2 + (x_1^2 - x_2^2) + (y_1^2 - y_2^2) + (z_1^2 - z_2^2) \\ \vdots \\ d_i^2 - d_1^2 + (x_1^2 - x_i^2) + (y_1^2 - y_i^2) + (z_1^2 - z_i^2) \\ \vdots \\ d_m^2 - d_1^2 + (x_1^2 - x_m^2) + (y_1^2 - y_m^2) + (z_1^2 - z_m^2) \end{bmatrix}$$

which leads to

$$AX = B$$

where

$$A = \begin{bmatrix} 2(x_1-x_2) & 2(y_1-y_2) & 2(z_1-z_2) \\ \vdots & & \\ 2(x_1-x_i) & 2(y_1-y_i) & 2(z_1-z_i) \\ \vdots & & \\ 2(x_1-x_m) & 2(y_1-y_m) & 2(z_1-z_m) \end{bmatrix}, \quad B = \begin{bmatrix} d_2^2 - d_1^2 + (x_1^2 - x_2^2) + (y_1^2 - y_2^2) + (z_1^2 - z_2^2) \\ \vdots \\ d_i^2 - d_1^2 + (x_1^2 - x_i^2) + (y_1^2 - y_i^2) + (z_1^2 - z_i^2) \\ \vdots \\ d_m^2 - d_1^2 + (x_1^2 - x_m^2) + (y_1^2 - y_m^2) + (z_1^2 - z_m^2) \end{bmatrix}$$

The least square method is again used to solve the matrix equation.

$$X = (A^T A)^{-1} A^T B \tag{2.5}$$

which gives the robot position $(x, y, z)^T$.

2.3.3.4 Robot Orientation Estimation

This subsection presents an orientation estimation method to find the initial robot heading with respect to the world frame. Since the robot position $(x, y, z)^T$ can be calculated via (2.5), the robot heading would be found by letting the robot to move in a straight line with the original heading direction, where the straight line is given as follows:

$$y_i = mX_i + c \tag{2.6}$$

In (2.6) the parameter m is slope of the straight line and c is the constant offset.

Assuming two more robot positions to be obtained from (2.5), one re-arranges (2.6) in a matrix-vector form as

$$\begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \\ 1 & X_n \end{bmatrix} \begin{bmatrix} c \\ m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \Rightarrow \text{CY=D, } \text{Y=} \begin{bmatrix} c \\ m \end{bmatrix}, \text{ C=} \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \\ 1 & X_n \end{bmatrix}, \text{ D=} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{2.7}$$

Again, the least-square method is adopted to solve the matrix equation (2.7).

$$\begin{bmatrix} c \\ m \end{bmatrix} = \text{Y} = (\text{C}^{\text{T}}\text{C})^{-1}\text{C}^{\text{T}}\text{D} \tag{2.8}$$

From (2.8) determines the slop m where $m=\tan 2\theta$ which indicates $\theta=\text{Atan}(m)$. In order to obtain θ in the interval $(-\pi,\pi]$, the two-argument arctangent function $Atan2$ is utilized to acquire the robot orientation, i.e., $\theta=\text{Atan2}(m)$.

As a result, the initial pose (position and orientation) of the robot can be uniquely determined using (2.5) and (2.8).

2.3.3.5 Real-Time ZigBee Global Pose Initialization Algorithm

When the tour-guide robot is required to localize itself or to calibrate its current pose under the effective ZigBee propagation region, this subsection proposes the following real-time pose initialization algorithm to find the correct start-up pose of the robot with respect to the world frame.

Step 1: measure the all RSSI data between all the readers ($\text{T}_1, \cdots, \text{T}_l$) and the tag ($\text{R}_1$) and obtain n sets of the measured values for each tag,

$$\text{RSSI}_1(j),\cdots,\text{RSSI}_l(j),\ j=1,2,3,\cdots,n\ .$$

Step 2: apply the simple averaging method to obtain the means of the received RSSI values as follows;

$$\overline{\mathrm{RSSI}_i} = \frac{1}{n}\sum_{j=1}^{n}\mathrm{RSSI}_i(j), \quad i = 1,\cdots,l. \tag{2.9}$$

Step 3: use the model (2.1) with the calibrated parameters K_i and n_i to convert the averaged RSSI value into the corresponding distance, i.e.,

$$d_i = (\mathrm{K}_i/\overline{\mathrm{RSSI}_i})^{1/n_i}, \quad i = 1,\cdots,m. \tag{2.10}$$

Step 4: utilize the least-squares (2.5) to compute start-up position of the robot pose $(\hat{x}_0, \hat{y}_0, \hat{z}_0)$.

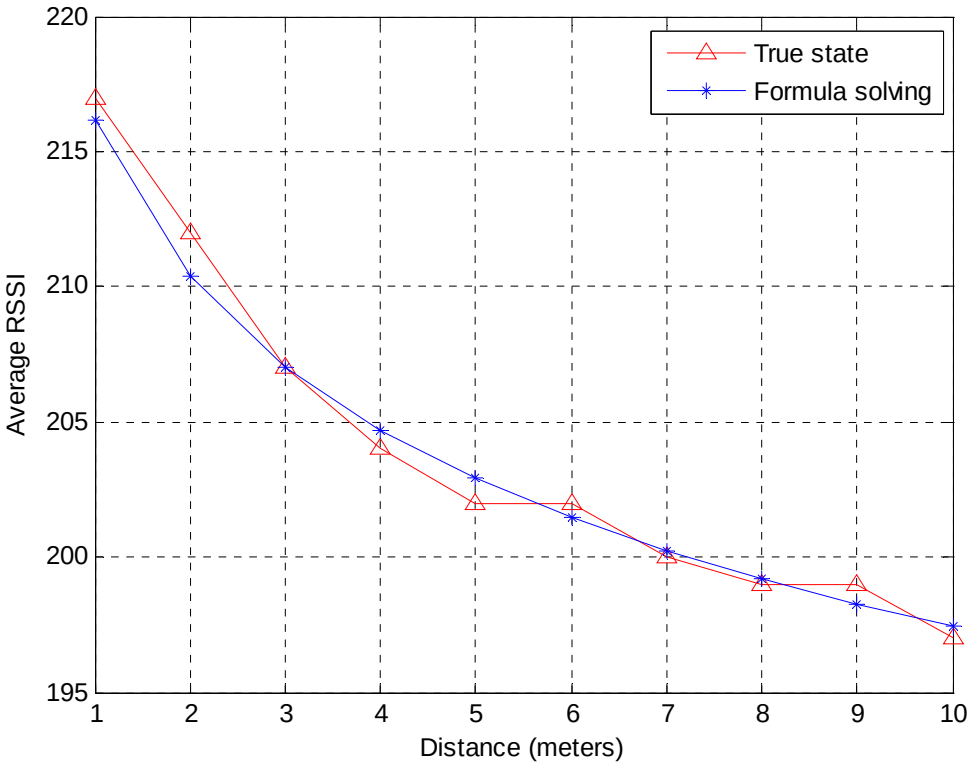
Step 5: control the robot to move a distance Δd along a straight line, find the present robot position $(\hat{x}_1, \hat{y}_1, \hat{z}_1)$ using (2.5), and then calculate the robot's orientation via (2.8), and obtain $\hat{\theta}_0 = A\tan 2\ (\hat{y}_1 - \hat{y}_0,\ \hat{x}_1 - \hat{x}_0)$

where $\hat{x}_1 - \hat{x}_0 = v\cos\hat{\theta}_0$, $\hat{y}_1 - \hat{y}_0 = v\sin\hat{\theta}_0$ and v represents the linear speed of the robot. Note that the angle range of two argument arc tangent function, $A\tan 2$ is between $-\pi$ and π.

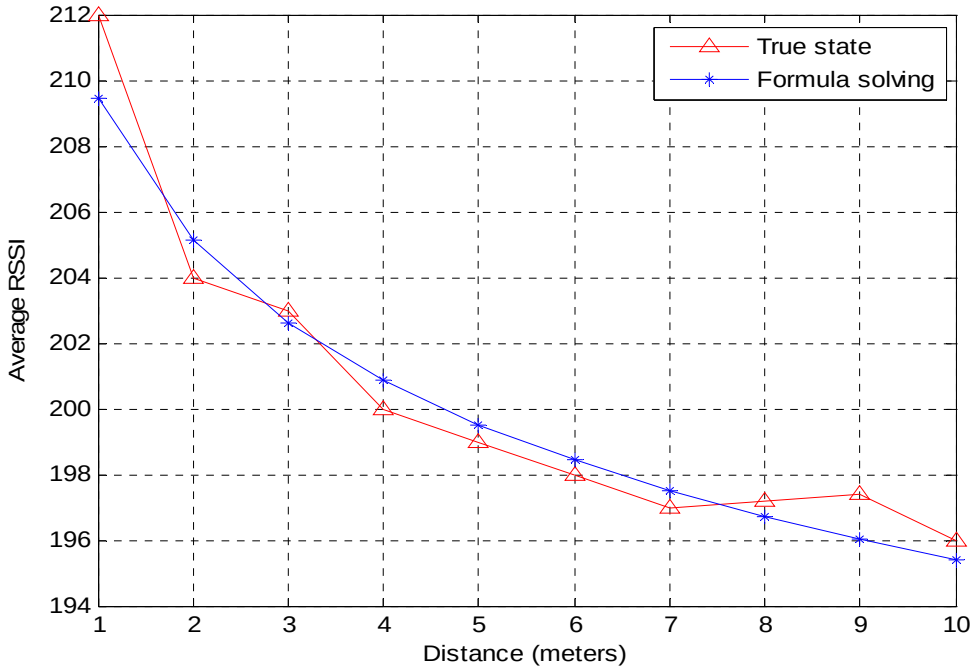
2.3.3.6 Experiment Result of Global Pose Initialization

The first experiment was performed to investigate the accuracy of the proposed method for ZigBee static pose estimate of the tour-guide robot with one tag on the robot and the three readers. The three readers were installed at the positions $(x_1, y_1, z_1) = (47, 544, 75), (x_2, y_2, z_2) = (400, 680, 74), (x_3, y_3, z_3) = (473, 313, 74)$ (unit: cm) with respect to the world frame. The true position of the robot in both x and y coordinate frame were given by (17.5cm, 240cm) , (70cm, 300cm) , (125cm, 325cm) . Before experimentation, all the three readers were calibrated using equation (2.1). Figure 2.8 shows the three calibration curves which covert the RSSI values into their corresponding distances. Afterwards, the real-time pose initialization algorithm was applied to calculate the pose estimates of the robot

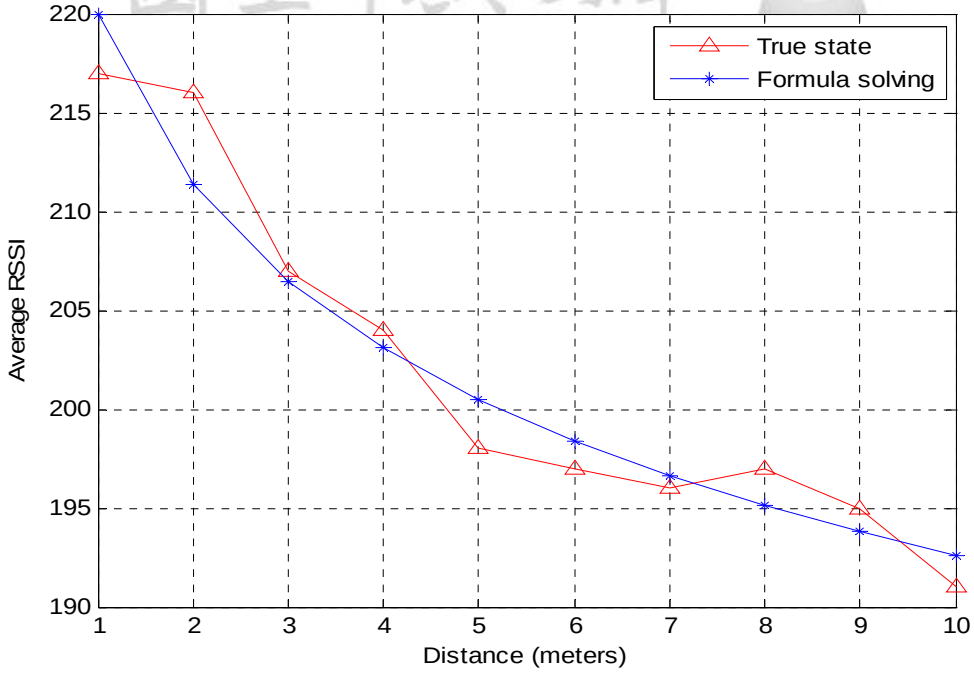
(0cm, 226.8455cm),(47.34cm, 313.28cm),(110.2, 349.16cm) In Figure 2.9, the circle represents the true position, and the cross represents the least-squares estimate. We observe that the proposed ZigBee global localization method is proven capable of having the position error of less than 30 cm and the heading error of less than 20^{o}. These experimental results indicate that the proposed pose initialization method can be effectively used to find the correct initial pose of the robot with respect to the world frame.



(a)



(b)



(c)

Figure 2.8. Calibration curves relating the RSSI values to their corresponding distances. (a)Reader 1.(b) Reader 2.(c) Reader 3.

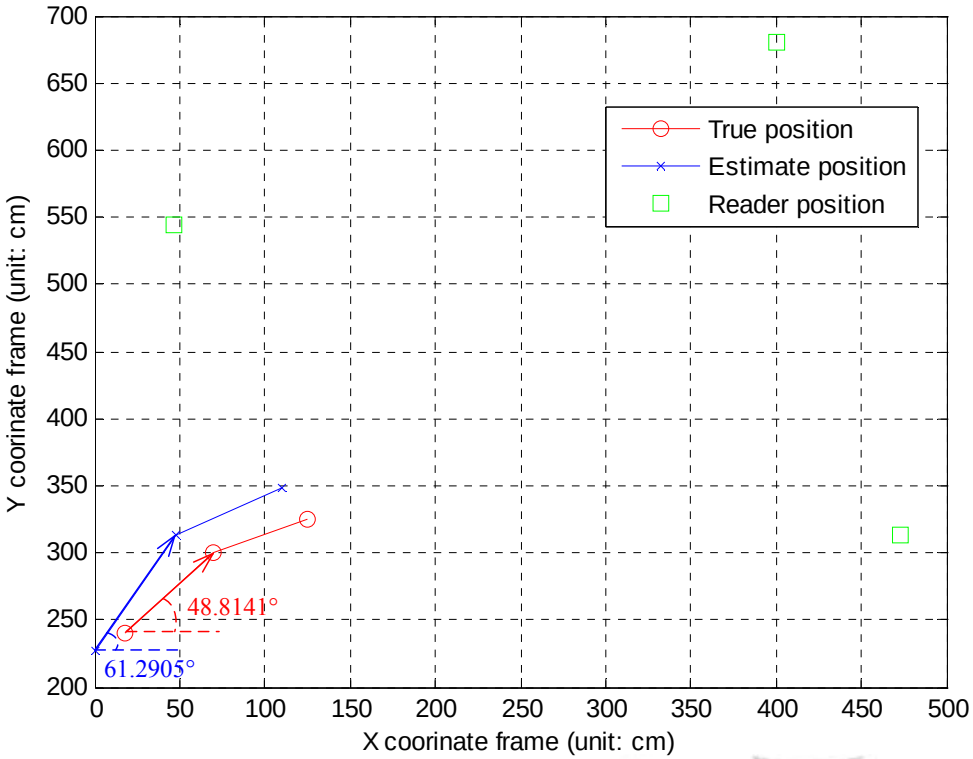


Figure 2.9. Static robot position and orientation estimates using the ZigBee pose initialization algorithm.

2.4 Dead-Reckoning and Signal Fusion

This section describes a kinematic model of the robot in the world frame, uses the model to find its position and orientation when the robot is travelling over short distances, and discusses how to fuse two measurements from the dead-reckoning unit and the ZigBee localization module. Below are detailed descriptions of the kinematic model, the dead-reckoning method and signal fusion approach.

2.4.1 Kinematic Model in the World Frame

Since the proposed omnidirectional mobile robot is equipped with four independent driving wheels equally spaced at 90 degrees from one to another; each driving wheel comprises an omnidirectional wheel. Figure 2.10 shows the commercial omnidirectional wheel from the Kornylak Corporation (kornylak.com), and Figure

2.11 shows the structure and geometry of the omnidirectional driving configuration with respect to the world frame, where θ represents the vehicle orientation which is positive in the counterclockwise direction, meanwhile θ also denote the angle between the moving frame and the world frame. The special feature of the omnidirectional mobile robot hinges on the wheels that allow free motions in any direction that is not parallel to the wheels' driving direction. Due to structural symmetry, the vehicle has the property that the center of geometry coincides with the center of mass.

Figure 2.10. Commercial omnidirectional wheel.

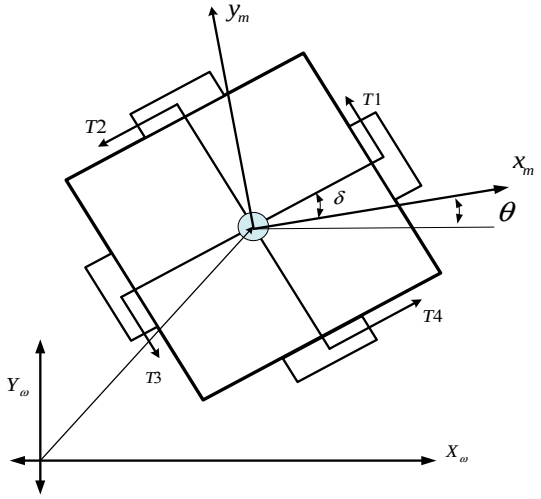


Figure 2.11. Structure and geometry of the omnidirectional driving configuration.

In what follows, particular effort will be paid to recall the kinematic model of this kind of robot. From the method proposed by Kalmár-Nagy *et al.*, it is easy to obtain the following inverse kinematic model of the four-wheeled omnidirectional mobile platform in the world frame

$$v(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \end{bmatrix} = \begin{bmatrix} r\omega_1(t) \\ r\omega_2(t) \\ r\omega_3(t) \\ r\omega_4(t) \end{bmatrix} = P(\theta(t)) \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} \tag{2.11}$$

where

$$P(\theta(t)) = \begin{bmatrix} -\sin(\delta+\theta) & \cos(\delta+\theta) & L \\ -\cos(\delta+\theta) & -\sin(\delta+\theta) & L \\ \sin(\delta+\theta) & -\cos(\delta+\theta) & L \\ \cos(\delta+\theta) & \sin(\delta+\theta) & L \end{bmatrix}$$

and $\omega_i(t)$, *i=1,2,3,4* denotes the angular velocity of each wheel, respectively; *r* denotes the radius of each wheel; *L* represents the distance from center of the platform to the center of each wheel. Notice that although the matrix $P(\theta(t))$ is singular for any θ, but it's left inverse matrix can be found, i.e., $P^{\#}(\theta(t))P(\theta(t)) = I_3$ and expressed by

$$P^{\#}(\theta(t)) = \begin{bmatrix} \frac{-\sin(\delta+\theta)}{2} & \frac{-\cos(\delta+\theta)}{2} & \frac{\sin(\delta+\theta)}{2} & \frac{\cos(\delta+\theta)}{2} \\ \frac{\cos(\delta+\theta)}{2} & \frac{-\sin(\delta+\theta)}{2} & \frac{-\cos(\delta+\theta)}{2} & \frac{\sin(\delta+\theta)}{2} \\ \frac{1}{4L} & \frac{1}{4L} & \frac{1}{4L} & \frac{1}{4L} \end{bmatrix} \tag{2.12}$$

2.4.2 Dead-Reckoning

The purpose of the dead-reckoning of the omnidirectional mobile robot is, given a correct initial pose, to continuously keep trace of its correct poses with respect

to the world frame. Dead-reckoning is the real-time calculation of the robot's position from wheel encoder measurements. The dead-reckoning problem of the robot can be easily solved by using the numerical approach, the velocity information from the encoders mounted on the driving wheels and the forward kinematical model described by

$$\begin{bmatrix} \dot{x}_w(t) \\ \dot{y}_w(t) \\ \dot{\theta}(t) \end{bmatrix} = P^{\#}(\theta(t))\upsilon(t) \tag{2.13}$$

where

To find the continuous poses of the robot from (2.13), many existing numerical approaches, such as the Euler's formula, the Runge-Kutta method and so on, can be employed according to the required numerical accuracy and the step size. One of the simplest dead-reckoning methods is based on the second-order Runge-Kutta formula which approximates the pose differentiation of the robot by the following equation,

$$\begin{bmatrix} x_w(k) \\ y_w(k) \\ \theta(k) \end{bmatrix} = \begin{bmatrix} x_w(k-1) \\ y_w(k-1) \\ \theta(k-1) \end{bmatrix} + T\frac{P^{\#}(\theta(k-1))\upsilon(k-1) + P^{\#}(\theta(k))\upsilon(k)}{2} \tag{2.14}$$

where T is the sampling period and sufficiently small; $x(k)$, $y(k)$ and $\theta(k)$ denote respectively the values of $x(t)$, $y(t)$ and $\theta(t)$ at the *k-th* sampling instant. Notice that (2.14) gives the recursive formula to approximately obtain the robot's position and orientation at the next sampling instant.

Worthy of mention is that the accuracy of the dead-reckoning (2.14) was inverse proportional to the sampling period. However, if the sampling period T is fixed and not significantly small, then the accuracy of the dead-reckoning method can be improved by using the fourth-order or higher-order Runge-Kutta numerical Method. Thus, this kind of dead-reckoning method can often be used as a useful pose

tracking approach for the robot over short traveling distances. That due to whatever the numerical method is used; the proposed dead-reckoning method always suffers from unavoidable accumulation errors caused by slippage, surface roughness, surface friction and even the mechanical structure of the robot.

2.4.3 Sensing Fusion

Due to the accumulation errors caused by slipping occurring between the four omnidirectional wheels and the ground, the dead-reckoning unit will eventually get lost of position tracking. To circumvent the shortcoming, the ZigBee location module is utilized to provide external position information of the robot and periodically correct the pose. Among many sense fusing algorithm, Kalman filtering approach [] is the most common useful method to merge the readings from the dead-reckoning unit and the ZigBee location module, thus reducing the position error significantly.

The Kalman filter is a set of mathematical equation that provides an efficient computational (recursive) means to estimate the state of the process of interest in a way to minimize the mean of the square error. The Kalman filter has been shown very powerful in several aspects, supporting smoothing, estimation and prediction of past, present, and even future states. In the following, the discrete-time Kalman filter for signal fusion is introduced.

Consider the signal fusion system with the following linear discrete-time state space model and measurement equation:

$$X(k+1) = AX(k) + T\frac{P^{\#}(\theta(k-1))\upsilon(k-1) + P^{\#}(\theta(k))\upsilon(k)}{2} + W(k) \tag{2.15}$$

$$Z(k) = CX(k) + V(k) \tag{2.16}$$

where k denotes the discrete-time index, $X(k)$ is an $n\times 1$ state vector, $Z(k)$ is an $m\times 1$ measurement vector, $W(k)$ and $V(k)$ two zero-mean mutually independent,

white Gaussian noise processes with covariance matrices Q and R, respectively. n the state dimension and m be the measurement dimension. Moreover,

$$A=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix},\quad C=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}$$

The Kalman filter provides a recursive state estimate $\hat{X}(k|k)$ at time k through a prediction estimate $\hat{X}(k+1|k)$ given all information up to time $k+1$ and a new measurement $Z(k+1)$, $K(k+1)$ is a Kalman filter gain and $r(k+1)$ is the innovation given by

$$r(k+1)=Z(k+1)-C\hat{X}(k+1|k) \tag{2.17}$$

$$K(k+1)=\hat{P}(k+1/k)C^T\left(CP(K+1/k)C^T+R\right)^{-1} \tag{2.18}$$

The Kalman filtering algorithm is described by the following equations.

(i) One-step-ahead Prediction:

$$\hat{x}(k+1|k)=A\hat{x}(k|k) \tag{2.19}$$

$$\hat{P}(k+1|k)=A\hat{P}(k/k)A^T+BQB^T \tag{2.20}$$

(ii) Estimation (Measurement Update):

$$\hat{x}(k+1|k+1)=\left[\hat{x}(k+1|k)+k(k+1)\right]r(k+1) \tag{2.21}$$

$$\hat{P}(k+1|k+1)=\hat{P}(k+1|k)-K(k+1)C\hat{P}(k+1|k) \tag{2.22}$$

2.5 Concluding Remarks

This chapter has described the system structure, the four-wheeled mobile platform and ZigBee localization module of the tour-guide robot. The overall system structure is almost similar to that developed by Feng [6], but the suspension spring and motors have been replaced in order to obtain higher output torques and have

smoother motion. In order to attain higher torques, the gear ratio of motors has been changed from 10：1 to 20：1. Moreover, the FPGA-based motion control chip has been used to substitute for original personal computer, thereby resulting in a lighter, more compact, low power-consumption motion controller. The ZigBee module has been adopted for accomplishing initial global localization. The information obtained from the dead-reckoning unit and ZigBee module is fused together to find the global localization of the robot at any place at any time. Through experimental results, the ZigBee module by the least-squares method has been shown capable of finding an acceptable pose estimate.

Chapter 3

Adaptive Dynamic Motion Control

3.1 Introduction and Control Architecture

This chapter develops an adaptive dynamic motion controller for position control and trajectory tracking of the omnidirectional mobile robot equipped with four independent omnidirectional wheels equally spaced at 90 degrees from one to another. Such a controller is synthesized by backstepping and will be proven globally asymptotically stable via the Lyapunov stability theory. Figure 3.1 shows the block diagram of the motion control system where the adaptive backstepping controller is used to achieve various motions and the estimator is employed to on-line estimate the required one controller parameter whose is unknown but constant. Finally, the feasibility and efficacy of the proposed control method are exemplified by conducting several simulations results on steering the mobile robot.

The rest of the chapter is organized as follows. Section 3.2 describes the dynamic model in the world frame. In Section 3.3 the dynamic motion controller design is presented. The adaptive dynamic motion controller is designed in Section 3.4 for the mobile robot with an unknown parameter k. Section 3.5 presents some simulations and discussion. Finally, the conclusions of the chapter are stated in Section 3.6.

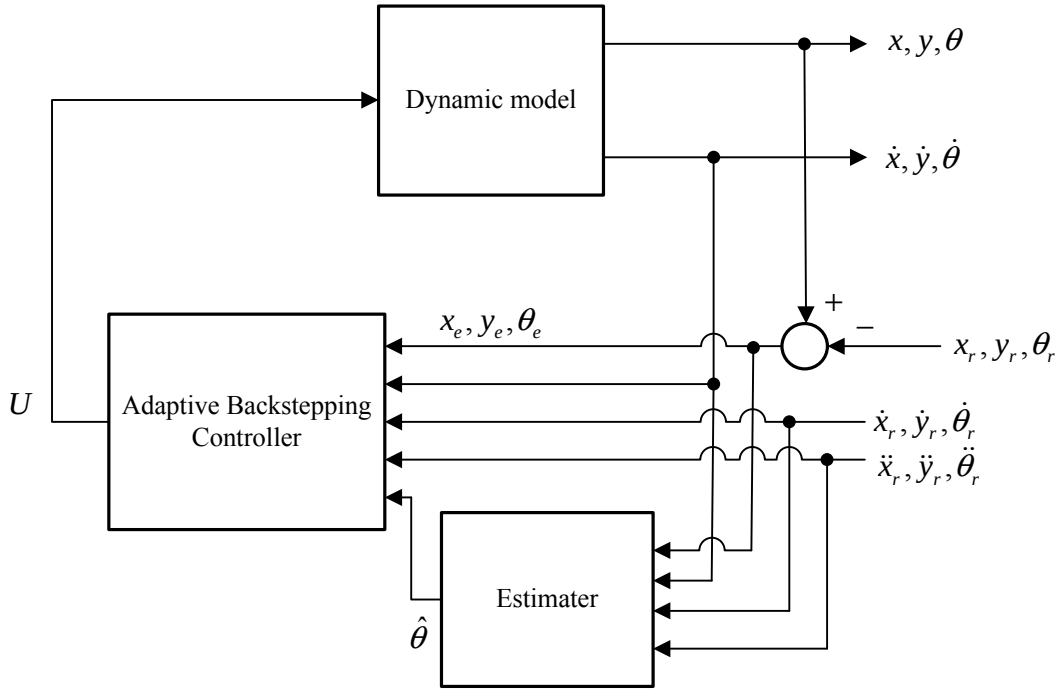


Figure 3.1. Block diagram of the motion control system.

3.2 Dynamic Model in the World Frame

This section derives a dynamic model of the omnidirectional mobile robot in the world frame. When the robot navigates along its planned path at slow speeds (less than 100cm/sec.), the kinematic model is valid and particular useful for finding the present pose of the robot with respect to a reference frame, or synthesizing regulators and controllers to steer the robot to achieve its scheduled missions. At high speeds, a dynamic model of the robot is needed for developing a dynamic controller to deal with uncertainties.

In order to derive the dynamic model for the four-wheeled omnidirectional robot, one develop the dynamic model in Cartesian coordinate with the world frame and moving frame first, the model $(\delta = 0)$ shown in Figure 3.2.

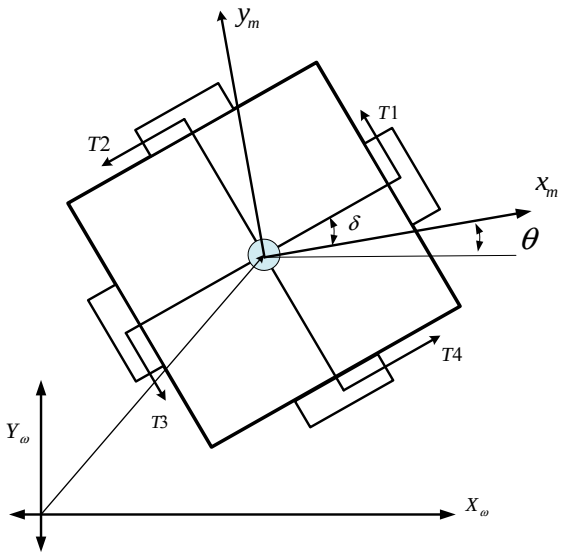


Figure 3.2. Geometry of the omnidirectional robot with simplified top-view.

By assumed the mobile robot is moving on a flat field with no slip, the initial world coordinate frame $X_\omega Y_\omega$ be fixed; and the moving coordinate frame $X_m Y_m$ is attached to the mass center of the mobile robot. Where the distance from the centre of mass to each wheel centre is L, δ is a fixed but adjustable angle and θ is the orientation of the mobile robot with respect to world frame X_ω. One defines two vectors with respect to world frame such that;

$$ {}^{w}S = \begin{bmatrix} x_w & y_w & \theta \end{bmatrix}^T , \tag{3.1} $$

and

$$ {}^{w}F = \begin{bmatrix} F_x & F_y & M_G \end{bmatrix}^T \tag{3.2} $$

where M_G is the torque applied to the robot; and ${}^{w}_{m}R$ is the transform matrix between the world frame and moving frame shown as follows;

$$ {}^{w}_{m}R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3} $$

Next, defines two vectors with respect to the moving frame such that

$$ {}^{m}s = \begin{bmatrix} x_m & y_m & \theta \end{bmatrix}^T \tag{3.4} $$

and

$$ {}^{m}f = \begin{bmatrix} f_x & f_y & M_G \end{bmatrix}^T \tag{3.5} $$

The transformations between the world and moving frames are

$$ {}^{w}\dot{S} = {}^{w}_{m}R\ {}^{m}\dot{s},\ {}^{w}F = {}^{w}_{m}R\ {}^{m}f \tag{3.6} $$

By applying the Newton's second law to the translation and rotation of the robot, one obtains

$$ \bar{M} \begin{bmatrix} \ddot{x}_m - \dot{y}_m\dot{\theta} \\ \ddot{y}_m + \dot{x}_m\dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ M_G \end{bmatrix} \tag{3.7} $$

where $\bar{M} = diag\{M, M, I_m\}$. Meanwhile, from Fig.3.3, since the instantaneous forces and moment balances are equal to

$$ \begin{bmatrix} f_x \\ f_y \\ M_G \end{bmatrix} = \begin{bmatrix} -\sin\delta & -\cos\delta & \sin\delta & \cos\delta \\ \cos\delta & -\sin\delta & -\cos\delta & \sin\delta \\ L & L & L & L \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}, \text{or} \quad {}^{m}f = Q(\delta)^T T \tag{3.8} $$

From the relation $\dot{q} = Q^T\ {}^{m}\dot{s}$, or expressed as follows;

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\\ \omega_4\end{bmatrix}=\frac{1}{r}\begin{bmatrix}-\sin\delta & \cos\delta & L\\ -\cos\delta & -\sin\delta & L\\ \sin\delta & -\cos\delta & L\\ \cos\delta & \sin\delta & L\end{bmatrix}\begin{bmatrix}\dot{x}_m\\ \dot{y}_m\\ \dot{\theta}\end{bmatrix}\Rightarrow \omega=\frac{1}{r}Q(\delta)^m\dot{s} \qquad (3.9)$$

The wheel dynamics for each assembly as given by [18] as follows;

$$I_w\dot{\omega}_i+c\omega_i=ku_i-f_i-rT_i \quad , \quad i=1,2,3,4 \qquad (3.10)$$

where u_i [v] is the voltage applied to the motor, and ω_i [rad/s] is the angular velocity of the motor shaft with neglected inductance; I_w is the moment of inertia of the wheel; f_i denote the coulomb friction of each wheel . The motor is characterized by the constants k [Nm/V] and c [Nm rad/s].

Expanding from (3.8), (3.9), and (3.10), one obtains

$$T_1=\frac{1}{r}\left(ku_1-f_1-I_\omega\dot{\omega}_1-c\omega_1\right)$$

$$T_2=\frac{1}{r}\left(ku_2-f_2-I_\omega\dot{\omega}_2-c\omega_2\right)$$

$$T_3=\frac{1}{r}\left(ku_3-f_3-I_\omega\dot{\omega}_3-c\omega_3\right)$$

$$T_4=\frac{1}{r}\left(ku_4-f_4-I_\omega\dot{\omega}_4-c\omega_4\right) \qquad (3.11)$$

and

$$\left\{\begin{array}{l}\omega_1=\frac{1}{r}(-\sin\delta\cdot\dot{x}_m+\cos\delta\cdot\dot{y}_m+L\dot{\theta})\\ \omega_2=\frac{1}{r}(-\cos\delta\cdot\dot{x}_m-\sin\delta\cdot\dot{y}_m+L\dot{\theta})\\ \omega_3=\frac{1}{r}(\sin\delta\cdot\dot{x}_m-\cos\delta\cdot\dot{y}_m+L\dot{\theta})\\ \omega_4=\frac{1}{r}(\cos\delta\cdot\dot{x}_m+\sin\delta\cdot\dot{y}_m+L\dot{\theta})\end{array}\right\}\Rightarrow\left\{\begin{array}{l}\dot{\omega}_1=\frac{1}{r}(-\sin\delta\cdot\ddot{x}_m+\cos\delta\cdot\ddot{y}_m+L\ddot{\theta})\\ \dot{\omega}_2=\frac{1}{r}(-\cos\delta\cdot\ddot{x}_m-\sin\delta\cdot\ddot{y}_m+L\ddot{\theta})\\ \dot{\omega}_3=\frac{1}{r}(\sin\delta\cdot\ddot{x}_m-\cos\delta\cdot\ddot{y}_m+L\ddot{\theta})\\ \dot{\omega}_4=\frac{1}{r}(\cos\delta\cdot\ddot{x}_m+\sin\delta\cdot\ddot{y}_m+L\ddot{\theta})\end{array}\right\} \qquad (3.12)$$

The substitution of (3.12) into (3.11) yields

$$T_1 = \frac{1}{r}\left(ku_1 - f_1 - \frac{I_\omega}{r}(-\sin\delta\cdot\ddot{x}_m + \cos\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r}(-\sin\delta\cdot\dot{x}_m + \cos\delta\cdot\dot{y}_m + L\dot{\theta}) \right)$$
$$= \frac{ku_1}{r} - \frac{f_1}{r} - \frac{I_\omega}{r^2}(-\sin\delta\cdot\ddot{x}_m + \cos\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(-\sin\delta\cdot\dot{x}_m + \cos\delta\cdot\dot{y}_m + L\dot{\theta}) \quad (3.13)$$

$$T_2 = \frac{1}{r}\left(ku_2 - f_2 - \frac{I_\omega}{r}(-\cos\delta\cdot\ddot{x}_m - \sin\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r}(-\cos\delta\cdot\dot{x}_m - \sin\delta\cdot\dot{y}_m + L\dot{\theta}) \right)$$
$$= \frac{ku_2}{r} - \frac{f_2}{r} - \frac{I_\omega}{r^2}(-\cos\delta\cdot\ddot{x}_m - \sin\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(-\cos\delta\cdot\dot{x}_m - \sin\delta\cdot\dot{y}_m + L\dot{\theta}) \quad (3.14)$$

$$T_3 = \frac{1}{r}\left(ku_3 - f_3 - \frac{I_\omega}{r}(\sin\delta\cdot\ddot{x}_m - \cos\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r}(\sin\delta\cdot\dot{x}_m - \cos\delta\cdot\dot{y}_m + L\dot{\theta}) \right)$$
$$= \frac{ku_3}{r} - \frac{f_3}{r} - \frac{I_\omega}{r^2}(\sin\delta\cdot\ddot{x}_m - \cos\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(\sin\delta\cdot\dot{x}_m - \cos\delta\cdot\dot{y}_m + L\dot{\theta}) \quad (3.15)$$

$$T_4 = \frac{1}{r}\left(ku_4 - f_4 - \frac{I_\omega}{r}(\cos\delta\cdot\ddot{x}_m + \sin\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r}(\cos\delta\cdot\dot{x}_m + \sin\delta\cdot\dot{y}_m + L\dot{\theta}) \right)$$
$$= \frac{ku_4}{r} - \frac{f_4}{r} - \frac{I_\omega}{r^2}(\cos\delta\cdot\ddot{x}_m + \sin\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(\cos\delta\cdot\dot{x}_m + \sin\delta\cdot\dot{y}_m + L\dot{\theta}) \quad (3.16)$$

Substituting (3.13 -16) into (3.7) and (3.8) gives

$$\bar{M}\begin{bmatrix} \ddot{x}_m - \dot{y}_m\dot{\theta} \\ \ddot{y}_m + \dot{x}_m\dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} -\sin\delta & -\cos\delta & \sin\delta & \cos\delta \\ \cos\delta & -\sin\delta & -\cos\delta & \sin\delta \\ L & L & L & L \end{bmatrix}\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}$$

or

$$M\left(\ddot{x}_m - \dot{y}_m\dot{\theta}\right) = -\sin\delta\cdot T_1 - \cos\delta\cdot T_2 + \sin\delta\cdot T_3 + \cos\delta\cdot T_4$$
$$M\left(\ddot{y}_m + \dot{x}_m\dot{\theta}\right) = \cos\delta\cdot T_1 - \sin\delta\cdot T_2 - \cos\delta\cdot T_3 + \sin\delta\cdot T_4 \quad (3.17)$$
$$\left(I_m\ddot{\theta}\right) = L(T_1 + T_2 + T_3 + T_4)$$

After manipulation, one obtains

$$M\left(\ddot{x}_m - \dot{y}_m\dot{\theta}\right)$$
$$= \frac{k}{r}\left(-\sin\delta\cdot u_1 - \cos\delta\cdot u_2 + \sin\delta\cdot u_3 + \cos\delta\cdot u_4\right)$$
$$+ \frac{1}{r}\left(\sin\delta\cdot f_1 + \cos\delta\cdot f_2 - \sin\delta\cdot f_3 - \cos\delta\cdot f_4\right) \quad (3.18)$$
$$- \frac{2I_w}{r^2}\ddot{x}_m - \frac{2c}{r^2}\dot{x}_m$$

and

$$
\begin{aligned}
& M\left(\ddot{y}_m + \dot{x}_m\dot{\theta}\right) \\
& = \frac{k}{r}\left(\cos\delta\cdot u_1 - \sin\delta\cdot u_2 - \cos\delta\cdot u_3 + \sin\delta\cdot u_4\right) \\
& + \frac{1}{r}\left(-\cos\delta\cdot f_1 + \sin\delta\cdot f_2 + \cos\delta\cdot f_3 - \sin\delta\cdot f_4\right) \\
& - \frac{2I_w}{r^2}\ddot{y}_m - \frac{2c}{r^2}\dot{y}_m
\end{aligned}
\tag{3.19}
$$

and

$$
\begin{aligned}
& \left(I_m\ddot{\theta}\right) = L(T_1 + T_2 + T_3 + T_4) \\
& = L\left\{\begin{array}{l}
\frac{ku_1}{r} - \frac{f_1}{r} - \frac{I_\omega}{r^2}(-\sin\delta\cdot\ddot{x}_m + \cos\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(-\sin\delta\cdot\dot{x}_m + \cos\delta\cdot\dot{y}_m + L\dot{\theta}) + \\
\frac{ku_2}{r} - \frac{f_2}{r} - \frac{I_\omega}{r^2}(-\cos\delta\cdot\ddot{x}_m - \sin\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(-\cos\delta\cdot\dot{x}_m - \sin\delta\cdot\dot{y}_m + L\dot{\theta}) + \\
\frac{ku_3}{r} - \frac{f_3}{r} - \frac{I_\omega}{r^2}(\sin\delta\cdot\ddot{x}_m - \cos\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(\sin\delta\cdot\dot{x}_m - \cos\delta\cdot\dot{y}_m + L\dot{\theta}) + \\
\frac{ku_4}{r} - \frac{f_4}{r} - \frac{I_\omega}{r^2}(\cos\delta\cdot\ddot{x}_m + \sin\delta\cdot\ddot{y}_m + L\ddot{\theta}) - \frac{c}{r^2}(\cos\delta\cdot\dot{x}_m + \sin\delta\cdot\dot{y}_m + L\dot{\theta})
\end{array}\right\}
\end{aligned}
\tag{3.20}
$$

which leads to

$$
\begin{aligned}
& \left(I_m\ddot{\theta}\right) = L(T_1 + T_2 + T_3 + T_4) \\
& = \frac{kL}{r}(u_1 + u_2 + u_3 + u_4) - \frac{L}{r}\left(f_1 + f_2 + f_3 + f_4\right) - 4\frac{I_\omega}{r^2}L^2\ddot{\theta} - 4\frac{c}{r^2}L^2\dot{\theta}
\end{aligned}
\tag{3.21}
$$

From (3.18-3.21) and setting $U = [u_1 \quad u_2 \quad u_3 \quad u_4]^T$, $F = [f_1 \quad f_2 \quad f_3 \quad f_4]^T$, one obtains the dynamic model of the robot in Cartesian coordinate with respect to the moving frame in a vector-matrix form as follows;

$$
P\left\{{}^m\ddot{s}\right\} + N\left\{{}^m\dot{s}\right\} = R\{U\} + K\left\{F\right\} \tag{3.22}
$$

where
$$P = \begin{bmatrix} \frac{2I_\omega}{r^2} + M & 0 & 0 \\ 0 & \frac{2I_\omega}{r^2} + M & 0 \\ 0 & 0 & I_m + \frac{4I_\omega}{r^2}L^2 \end{bmatrix}_{3X3} \tag{3.23}$$

$$N \left\{ {}^m\dot{s} \right\} = \begin{bmatrix} \frac{2c}{r^2}\dot{x}_m - M\dot{y}_m\dot{\theta} \\ M\dot{x}_m\dot{\theta} + \frac{2c}{r^2}\dot{y}_m \\ \frac{c}{r^2}4L^2\dot{\theta} \end{bmatrix}_{3\times 1} \tag{3.24}$$

$$R = \frac{k}{r}\begin{bmatrix} -\sin\delta & -\cos\delta & \sin\delta & \cos\delta \\ \cos\delta & -\sin\delta & -\cos\delta & \sin\delta \\ L & L & L & L \end{bmatrix}_{3X4} \tag{3.25}$$

$$K = \frac{1}{r}\begin{bmatrix} \sin\delta & \cos\delta & -\sin\delta & -\cos\delta \\ -\cos\delta & \sin\delta & \cos\delta & -\sin\delta \\ -L & -L & -L & -L \end{bmatrix}_{3\times4} \tag{3.26}$$

Next, we desire to express (3.22) with respect to the world frame. With (3.6), it follows that

$${}^w\ddot{S} = {}^w_m\dot{R}\,{}^m\dot{s} + {}^w_mR\,{}^m\ddot{s} \tag{3.27}$$

which gives

$${}^m\dot{s} = ({}^w_mR)^T\,{}^w\dot{S} \quad \text{and} \quad {}^m\ddot{s} = ({}^w_mR)^T({}^w\ddot{S} - {}^w_m\dot{R}({}^w_mR)^T\,{}^w\dot{S}) \tag{3.28}$$

Substituting (3.28) into (3.22) yields

$$P\left\{({}^w_mR)^T({}^w\ddot{S} - {}^w_m\dot{R}({}^w_mR)^T\,{}^w\dot{S})\right\} + \bar{N}\left\{{}^w\dot{S}\right\} = R\{U\} + K\{F\} \tag{3.29}$$

where $\bar{N}\left\{{}^w\dot{S}\right\} = N\left\{{}^m\dot{s}\right\}\Big|_{{}^m\dot{s}=({}^w_mR)^T\,{}^w\dot{S}}$. Since the mass matrix P is always invertible, (3.27) becomes

$$({}^{w}\ddot{S} - {}^{w}_{m}\dot{R}({}^{w}_{m}R)^{T}\,{}^{w}\dot{S}) + ({}^{w}_{m}R)P^{-1}\bar{N}\left\{{}^{w}\dot{S}\right\} = ({}^{w}_{m}R)P^{-1}R\{U\} + ({}^{w}_{m}R)P^{-1}K\{F\} \tag{3.30}$$

or

$${}^{w}\ddot{S} + \bar{\bar{N}}\left\{{}^{w}\dot{S}\right\} = ({}^{w}_{m}R)P^{-1}R\{U\} \tag{3.31}$$

where

$$\bar{\bar{N}}\left\{{}^{w}\dot{S}\right\} = ({}^{w}_{m}R)P^{-1}\bar{N}\left\{{}^{w}\dot{S}\right\} - {}^{w}_{m}\dot{R}({}^{w}_{m}R)^{T}\,{}^{w}\dot{S} - ({}^{w}_{m}R)P^{-1}K\{F\}$$

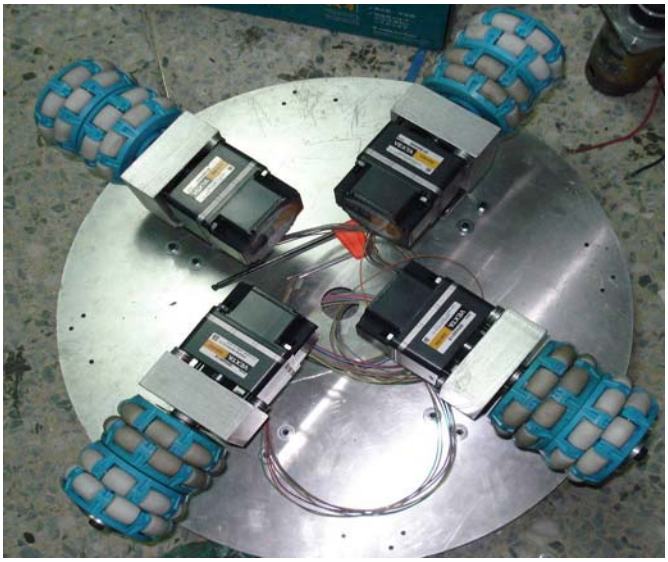
$$= \begin{bmatrix} \frac{2c/r^2}{\bar{M}}\dot{x}_w + (1-\frac{M}{\bar{M}}\cos\theta)\dot{\theta}\dot{y}_w - (\frac{M}{\bar{M}}\sin\theta)\dot{x}_w\dot{\theta} + \frac{\cos\theta}{r\bar{M}}(-f_1\sin\delta - f_2\cos\delta + f_3\sin\delta + f_4\cos\delta) \\ -\frac{\sin\theta}{r\bar{M}}(f_1\cos\delta - f_2\sin\delta - f_3\cos\delta + f_4\sin\delta) \\ (\frac{M}{\bar{M}}\cos\theta - 1)\dot{\theta}\dot{x}_w + \frac{2c/r^2}{\bar{M}}\dot{y}_w - (\frac{M}{\bar{M}}\sin\theta)\dot{y}_w\dot{\theta} + \frac{\sin\theta}{r\bar{M}}(-f_1\sin\delta - f_2\cos\delta + f_3\sin\delta + f_4\cos\delta) \\ +\frac{\cos\theta}{r\bar{M}}(f_1\cos\delta - f_2\sin\delta - f_3\cos\delta + f_4\sin\delta) \\ \frac{4cL^2}{4L^2I_w + r^2I_m}\dot{\theta} + \frac{L}{\frac{4L^2I_w}{r} + rI_m}(f_1 + f_2 + f_3 + f_4) \end{bmatrix}_{3\times 1}$$

and $\bar{M} = \frac{2I_w}{r^2} + M$.

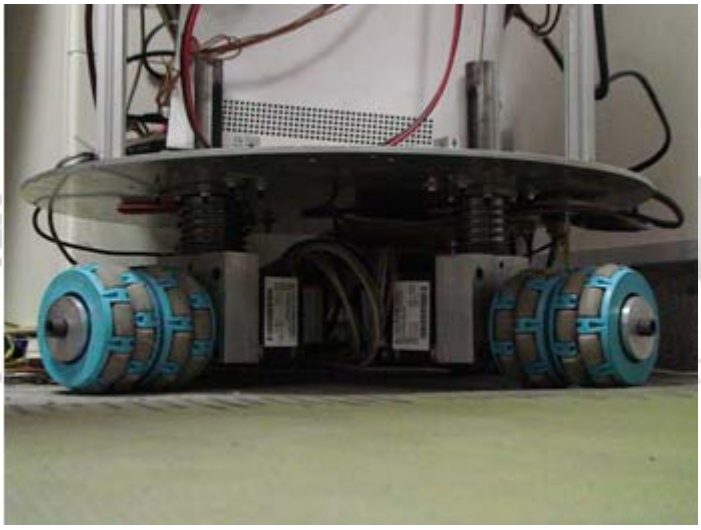
Rewrite (3.31) in terms of state vectors by defining $Y_1 = {}^{w}S$ and $Y_2 = {}^{w}\dot{S}$, and thus obtain

$$\begin{aligned} \dot{Y}_1 &= Y_2 \\ \dot{Y}_2 &= ({}^{w}_{m}R)P^{-1}R\{U\} - \bar{\bar{N}}\left\{{}^{w}\dot{S}\right\} \end{aligned} \tag{3.32}$$

which represents the three coupled multiple-input- multiple-output (MIMO) nonlinear state equations of dynamic motion of the robot with wheel torques as the inputs and the position and orientation in the world frame as the output.



(a)



(b)

Figure 3.3. Experimental omnidirectional four-wheeled mobile platform: (a) bottom-view; (b) side-view.

3.3 Dynamic Motion Controller Design

This section aims to design a dynamic control law for both regulation (stabilization) and trajectory tracking of the four-wheeled omnidirectional robot with the dynamic model (3.32) such that the omnidirectional mobile robot exactly follows the desired differentiable trajectory described by $Y_r(t) = \begin{bmatrix} x_r(t) & y_r(t) & \theta_r(t) \end{bmatrix}^T$. The

design procedure based on the backstepping technique is elaborated as follows. In doing so, define the tracking error vector by $Y_e = Y_1 - Y_r$. Differentiating Y_e and Y_r yields

$$\dot{Y}_e = \dot{Y}_1 - \dot{Y}_r = Y_2 - \dot{Y}_r \tag{3.33}$$

Considering Y_2 as a virtual control which is designed as $Y_2 = \phi(Y_1) = -K_P Y_e + \dot{Y}_r$, one obtains $\dot{Y}_e = -K_P Y_e + \dot{Y}_r - \dot{Y}_r = -K_P Y_e$ where the matrix K_P is diagonal and positive definite. The asymptotic stability of the Y_e dynamics can be shown via selecting the quadratic Lyapunov function $V_1 = \frac{1}{2} Y_e^T K_P^2 Y_e$, thus resulting in

$$\dot{V}_1 = Y_e^T K_P^2 \dot{Y}_e = Y_e^T K_P^2 (-K_P Y_e) = -Y_e^T K_P^3 Y_e < 0 . \tag{3.34}$$

To achieve the controller deign, the following backstepping error vector is defined by

$$\eta = Y_2 - \phi(Y_1) = Y_2 + K_P Y_e - \dot{Y}_r \tag{3.35}$$

From the definition (3.35), it follows that

$$\dot{Y}_e = Y_2 - \dot{Y}_r = (Y_2 + K_P Y_e - \dot{Y}_r) - K_P Y_e = \eta - K_P Y_e \tag{3.36}$$

Taking the time derivative of the backstepping error η gives

$$\begin{aligned} \dot{\eta} &= \dot{Y}_2 + K_P \dot{Y}_e - \ddot{Y}_r = ({}^w_m R) P^{-1} R\{U\} - \bar{\bar{N}}\{{}^w\dot{S}\} + K_P(\eta - K_P Y_e) - \ddot{Y}_r \\ &= ({}^w_m R) P^{-1} R\{U\} - \bar{\bar{N}}\{{}^w\dot{S}\} + K_P \eta - K_P^2 Y_e - \ddot{Y}_r \end{aligned} \tag{3.37}$$

Clearly, the control goal is to design the control law for U such that $\eta \to 0$ as $t \to \infty$ .This can be easily done by choosing the control law as

$$U = R^{\#} P ({}^w_m R)^T \left(\bar{\bar{N}}\{{}^w\dot{S}\} - (K + K_P)\eta + \ddot{Y}_r \right) \tag{3.38}$$

where the mass matrix K is also symmetric and positive-definite, and the right psudo-inverse matrix $R^{\#}$, satisfying $RR^{\#} = I_{3x3}$, is given by

$$R^{\#} = \begin{bmatrix} 0 & \frac{r}{2k} & \frac{r}{4kL} \\ \frac{-r}{2k} & 0 & \frac{r}{4kL} \\ 0 & \frac{-r}{2k} & \frac{r}{4kL} \\ \frac{r}{2k} & 0 & \frac{r}{4kL} \end{bmatrix} \tag{3.39}$$

Substituting (3.38) into (3.37) yields

$$\dot{\eta} = \bar{\bar{N}}\left\{{}^{w}\dot{S}\right\} - (K + K_P)\eta + \ddot{Y}_r - \bar{\bar{N}}\left\{{}^{w}\dot{S}\right\} + K_P\eta - K_P{}^2 Y_e - \ddot{Y}_r = -K\eta - K_P{}^2 Y_e \tag{3.40}$$

The asymptotical stability of the overall closed-loop error system is proven by choosing the radial, unbounded and quadratic Lyapunov function $V_2 = \frac{1}{2} Y_e{}^T K_P{}^2 Y_e + \frac{1}{2}\eta^T\eta$. Then the time derivative of the Lyapunov equation V_2 becomes

$$\begin{aligned} \dot{V}_2 &= Y_e{}^T (K_P{}^2)\dot{Y}_e + \eta^T\dot{\eta} \\ &= Y_e{}^T K_P{}^2\eta - Y_e{}^T K_P{}^3 Y_e - \eta^T K\eta - \eta^T K_P{}^2 Y_e = -Y_e{}^T K_P{}^3 Y_e - \eta^T K\eta \end{aligned} \tag{3.41}$$

which indicates that $\dot{V}_2$ is negative definite because the two matrices $K_P{}^3$ and K are symmetric, positive definite and even diagonal, and V_2 is a Lyapunov function for the proposed controller (3.38). Hence the two vectors Y_e and η approach zero as time goes to infinity, that is, $Y_1 \to Y_r$ and $Y_2 \to \dot{Y}_r$ as $t \to \infty$. This result indicates that the proposed dynamic control law (3.38) is capable of steering the robot with the dynamic model (3.32) to follow any destination pose or any differentiable and time-varying trajectory. Hence, this main result is summarized

as follows.

Theorem 3.1 For the robot's dynamic model (3.32) with the desired differentiable trajectory $Y_r = [x_r(t) \quad y_r(t) \quad \theta_r(t)]^T$ and the dynamic control law (3.38), the platform can be steered to reach any destination pose or follow any differentiable and time-varying trajectory in the sense of globally asymptotical stability, i.e., $Y_1 \to Y_r$ and $Y_2 \to \dot{Y}_r$ as $t \to \infty$.

3.4 Adaptive Dynamic Motion Controller Design

This section is devoted to developing an adaptive dynamic motion controller for the mobile robot with an unknown parameter *k*. In doing so, it is necessary to define $\theta = \frac{r}{k}$ and $\hat{\theta}$ as an estimate of $\theta = \frac{r}{k}$. With the symbol, one obtains

$$ {}^{w}_{m}R = \theta^{-1}\begin{bmatrix} 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ L & L & L & L \end{bmatrix}, {}^{w}_{m}R^{\#} = \theta\begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{4L} \\ -\frac{1}{2} & 0 & \frac{1}{4L} \\ 0 & -\frac{1}{2} & \frac{1}{4L} \\ \frac{1}{2} & 0 & \frac{1}{4L} \end{bmatrix} \tag{3.42} $$

Since the matrix ${}^{w}_{m}R$ contains the unknown parameter θ, the adaptive dynamic control law proposed by

$$ U = R^{\#}P({}^{w}_{m}\hat{R})^{T}\left(\bar{\bar{N}}\{{}^{w}\dot{S}\} - (K + K_P)\eta + \ddot{Y}_r\right) \tag{3.43} $$

where

$$ {}^{w}_{m}\hat{R} = \hat{\theta}^{-1}\begin{bmatrix} 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ L & L & L & L \end{bmatrix} $$

Substituting (3.38) and (3.42) into (3.37) yields

$$
\begin{aligned}
\dot{\eta} &= ({}^{w}_{m}R)P^{-1}R\{U\} - \dddot{\bar{N}} + K_P\eta - K_P{}^2Y_e - \ddot{Y}_r \\
&= \hat{\theta}\theta^{-1}\left[\dddot{\bar{N}} - \left(K + K_p\right)\eta + \ddot{Y}_r\right] - \dddot{\bar{N}} + K_P\eta - K_P{}^2Y_e - \ddot{Y}_r \\
&= \left(\hat{\theta}\theta^{-1} - 1\right)\dddot{\bar{N}} + \left(1 - \hat{\theta}\theta^{-1}\right)K_P\eta + \left(\hat{\theta}\theta^{-1} - 1\right)\ddot{Y}_r - K_P{}^2Y_e - \hat{\theta}\theta^{-1}K\eta \\
&= \left(\frac{\hat{\theta}}{\theta} - 1\right)\dddot{\bar{N}} - \left(\frac{\hat{\theta}}{\theta} - 1\right)K_P\eta + \left(\frac{\hat{\theta}}{\theta} - 1\right)\ddot{Y}_r - K_P{}^2Y_e - \frac{\hat{\theta}}{\theta}K\eta
\end{aligned}
\tag{3.44}
$$

To achieve the controller deign, the following estimate error is defined by

$$
\tilde{\theta} = \theta - \hat{\theta} \quad \left(\dot{\tilde{\theta}} = \dot{\theta} - \dot{\hat{\theta}} = -\dot{\hat{\theta}}\right) \tag{3.45}
$$

Then the time derivative of the backstepping error vector η becomes

$$
\begin{aligned}
\dot{\eta} &= -\frac{\tilde{\theta}}{\theta}\dddot{\bar{N}} + \frac{\tilde{\theta}}{\theta}K_P\eta - \frac{\tilde{\theta}}{\theta}\ddot{Y}_r - K_P{}^2Y_e - K\eta + \frac{\tilde{\theta}}{\theta}K\eta \\
&= \frac{\tilde{\theta}}{\theta}\left[\left(K + K_p\right)\eta - \dddot{\bar{N}} - \ddot{Y}_r\right] - K_P{}^2Y_e - K\eta
\end{aligned}
\tag{3.46}
$$

The asymptotical stability of the overall closed-loop error system is proven by choosing the radial, unbounded and quadratic Lyapunov function

$$
V = \frac{1}{2}Y_e^T K_p^2 Y_e + \frac{1}{2}\eta^T\eta + \frac{1}{2\gamma}\tilde{\theta}^2, \gamma > 0. \tag{3.47}
$$

With (3.36) and (3.45) ,the time derivative of the Lyapunov equation V becomes

$$
\begin{aligned}
\dot{V} &= Y_e^T K_p^2\dot{Y}_e + \eta^T\dot{\eta} + \frac{1}{\gamma}\tilde{\theta}\left(-\dot{\hat{\theta}}\right) \\
&= Y_e^T K_p^2\left(\eta - K_pY_e\right) + \eta^T\left\{-K_p^2Y_e + \frac{\tilde{\theta}}{\theta}\left[\left(K + K_p\right)\eta - \dddot{\bar{N}} - \ddot{Y}_r\right]\right\} + \frac{1}{\gamma}\tilde{\theta}\left(-\dot{\hat{\theta}}\right) \\
&= -Y_e^T K_p^3 Y_e - \eta^T K\eta + \frac{\tilde{\theta}}{\theta}\eta^T\left[\left(K + K_p\right)\eta - \dddot{\bar{N}} - \ddot{Y}_r\right] + \tilde{\theta}\left(\frac{-\dot{\hat{\theta}}}{\gamma}\right)
\end{aligned}
$$

or

$$
\dot{V} = -Y_e^T K_p^3 Y_e - \eta^T K\eta + \tilde{\theta}\left\{\frac{\eta^T\left[\left(K + K_p\right)\eta - \dddot{\bar{N}} - \ddot{Y}_r\right]}{\theta} - \frac{\dot{\hat{\theta}}}{\gamma}\right\} \tag{3.48}
$$

If the parameter adjustment rule for $\hat{\theta}$ is chosen as

$$\frac{\dot{\hat{\theta}}}{\gamma} = \frac{\eta^T \left[\left(K + K_p \right) \eta - \bar{\bar{N}} - \ddot{Y}_r \right]}{\theta}$$

or

$$\begin{aligned} \dot{\hat{\theta}} &= \gamma \left\{ \frac{\eta^T \left[\left(K + K_p \right) \eta - \bar{\bar{N}} - \ddot{Y}_r \right]}{\theta} \right\} \\ &= \gamma' \eta^T \left[\left(K + K_p \right) \eta - \bar{\bar{N}} - \ddot{Y}_r \right] \; where \; \gamma' = \frac{\gamma}{\theta} \end{aligned} \tag{3.49}$$

Then $\dot{V} = -Y_e^T K_p^3 Y_e - \eta^T K \eta \le 0$. Barbalat's lemma implies that $Y_e \to 0, \eta \to 0 \;\; as \; t \to \infty$.

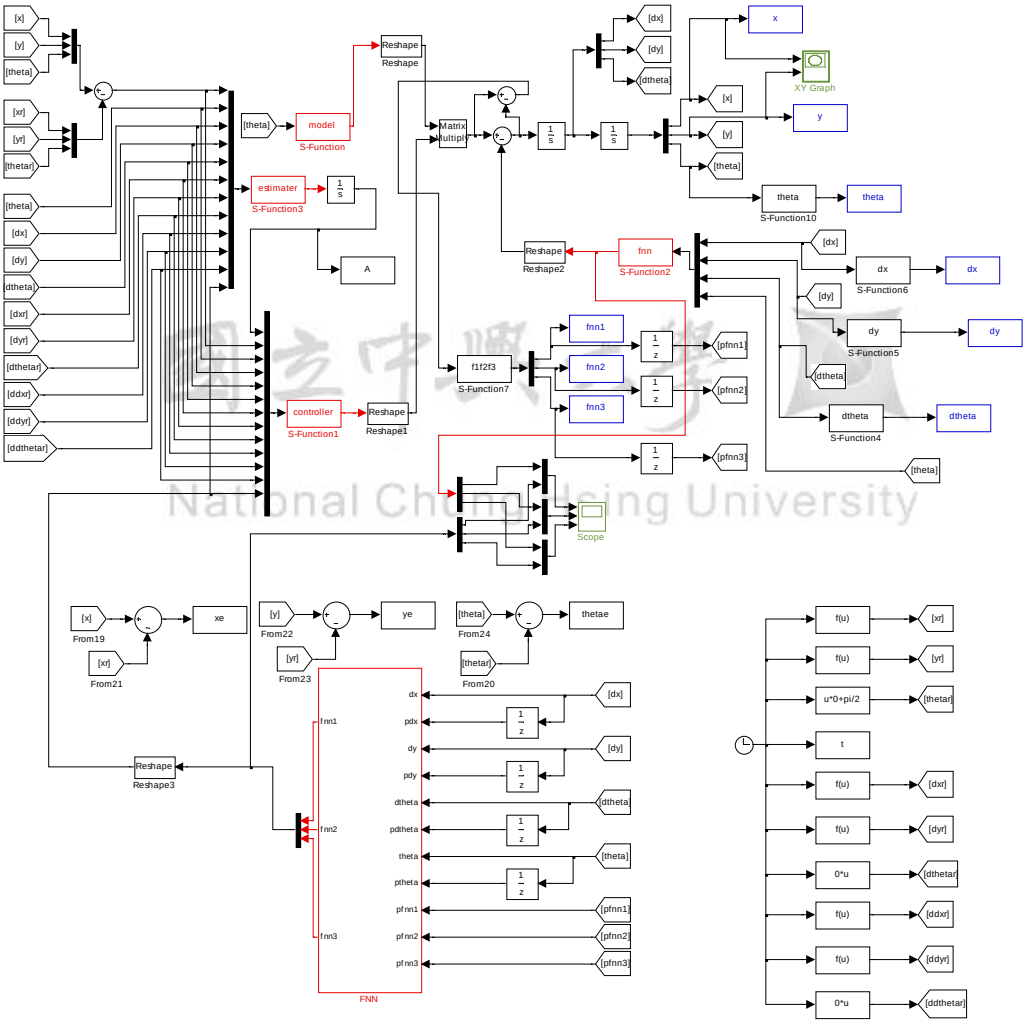
Theorem 3.2 For the dynamic model (3.32) with the desired differentiable trajectory $Y_r = \left[x_r(t) \quad y_r(t) \quad \theta_r(t) \right]^T$ and the adaptive dynamic control law (3.43) with the parameter updating rule (3.49), the mobile platform can be steered to reach any destination pose or follow any differentiable and time-varying trajectory in the sense of globally asymptotical stability, i.e., $Y_1 \to Y_r$ and $Y_2 \to \dot{Y}_r$ as $t \to \infty$.

3.5 Simulations And Experiments and Discussion

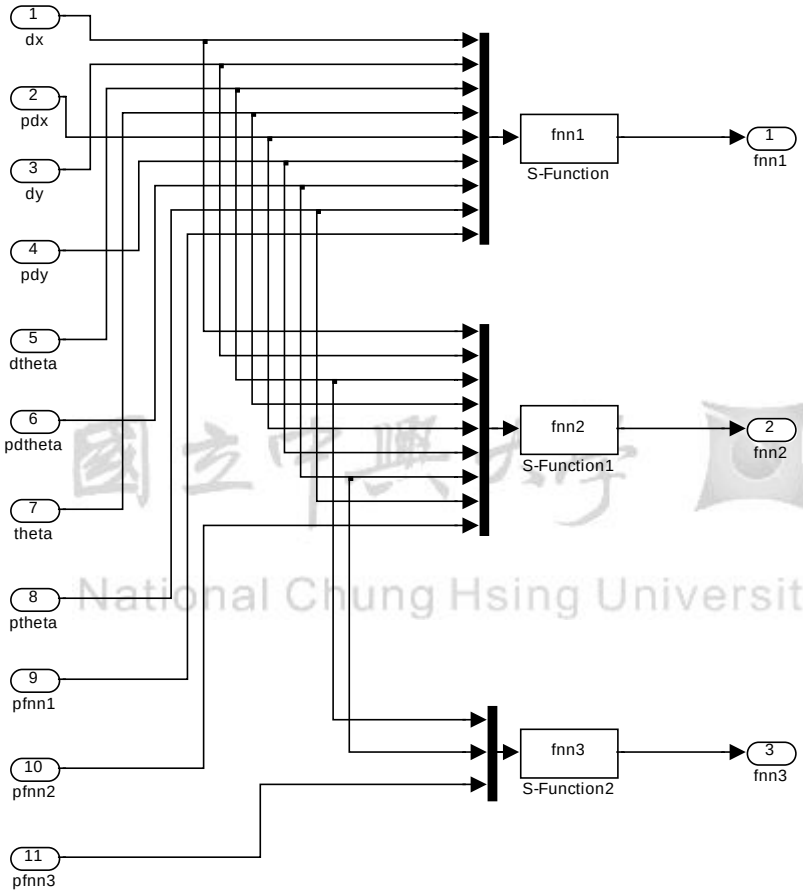
3.5.1 Brief Description of the Experimental Omnidirectional Mobile Robot

The aim of the simulations and experiments is to examine the effectiveness and performance of the proposed nonlinear adaptive controller. In applying **Theorem 3.2**, two diagonal gain matrices, $K_p = diag\{1.43, 1.5, 2.5\}$ and $K = diag\{278.57, 148.5, 37.5\}$. The control laws (3.43) and (3.38) together with the proposed dead-reckoning method were implemented using C++ codes and standard programming techniques.

Table 3.1 lists all the parameters and the actual values of the simulation. Fig. 3.4(a) and Fig. 3.4(b) depict the simulation results of the dynamic motion controller.



(a)



(b)

Figure 3.4. Simulink model of the adaptive dynamic motion controller: (a) main block diagram; (b) subsystem block diagram.

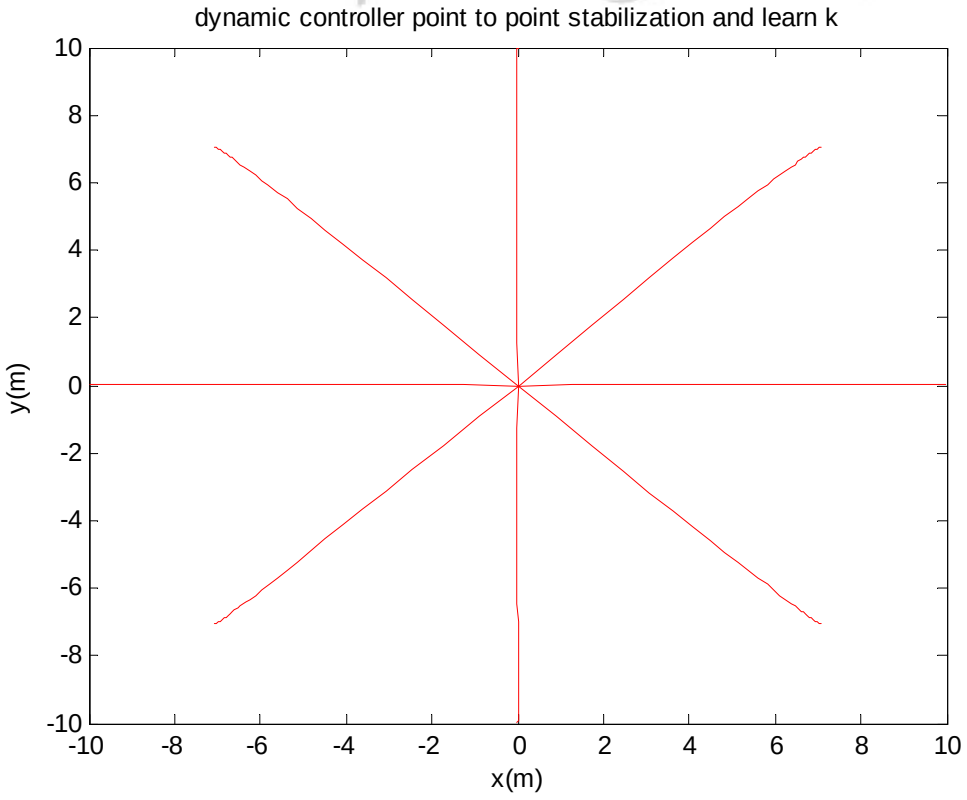
Table 3.1 List of the parameters and the actual values.

Parameters	Value
r	0.0508m
L	0.26m
I_w	$0.00080477\,kg\cdot m^2$
I_m	$7.02\,kg\cdot m^2$
M	78kg
c	0.025
k_1	278.57
k_2	148.5
k_3	37.5
k_{p1}	1.43
k_{p2}	1.5
k_{p3}	2.5
Z	0.01
k_0	0.015Nm/V

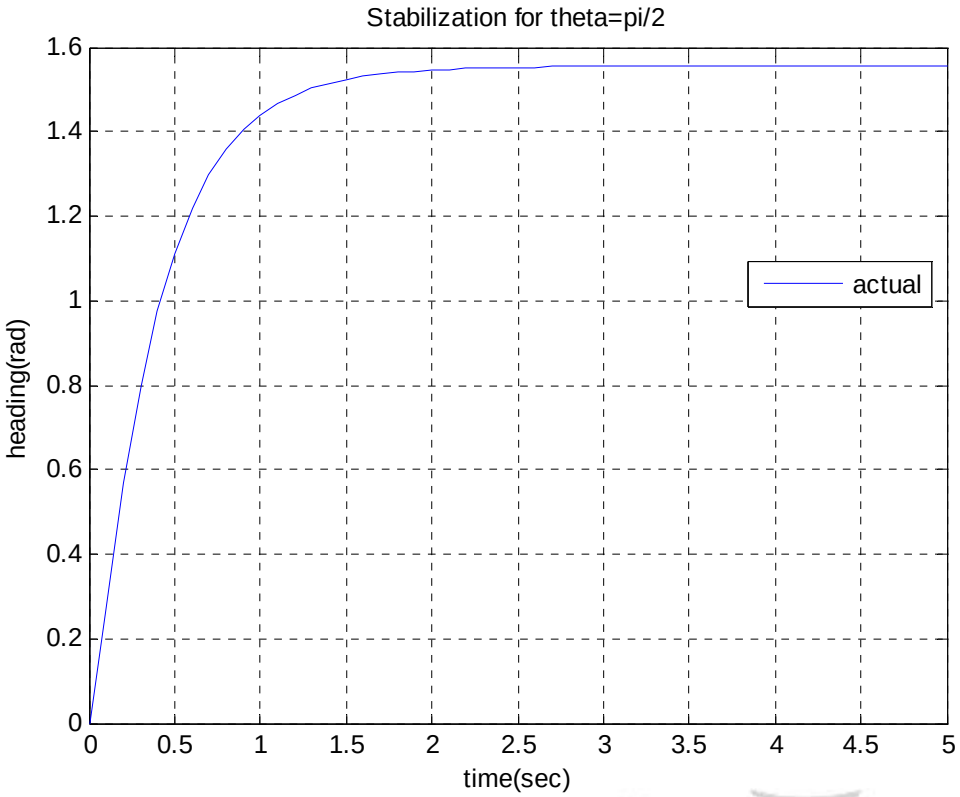
3.5.2 Regulation

The first simulation was conducted to investigate the performance of the proposed control law (3.38). The initial pose of the omnidirectional mobile robot was assumed at the origin, i.e., $[X_0 \quad Y_0 \quad \theta_0]^T = [0cm \quad 0cm \quad 0\,rad]^T$ and the desired poses were

located on a circle with the radius of 100cm and given by $\left(X_d,\ \mathrm{Y_d},\ \theta_d\right)^T = \left(10\times\cos(\theta),\ 10\times\sin(\theta),\ \theta\right)^T$, where $\theta = \frac{n\pi}{4}(n = 0:1:7)$, respectively. Fig. 3.5(a) depicts all the simulation trajectories of the omnidirectional mobile base from the origin to the goal poses, and Figure 3.5(b) shows the heading behavior of the proposed stabilization law for the robot moving towards the desired orientation $\frac{\pi}{2}$. The results in Figure 3.5(a) indicated that the trajectories have almost minimum distances, namely that the robot moved along with straight lines towards the goal poses. More importantly, the mobile robot moved toward any direction and simultaneously attained the desired orientation, showing the unique property of the omnidirectional wheeled mobile robot. Through simulation results, the mobile robot with the proposed stabilization method has been shown capable of reaching the desired postures with satisfactory performance.



(a)

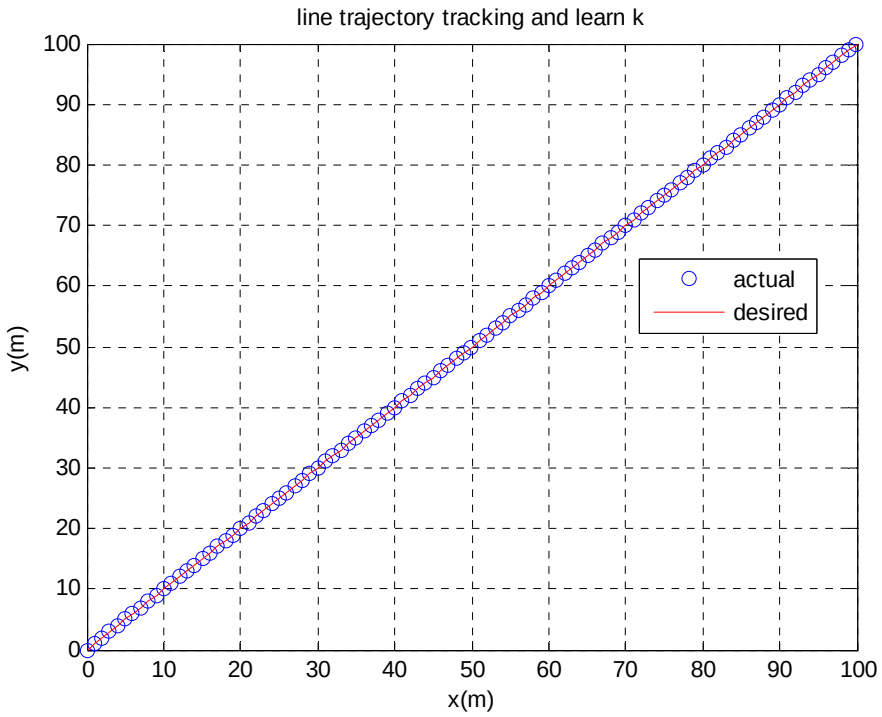


(b)

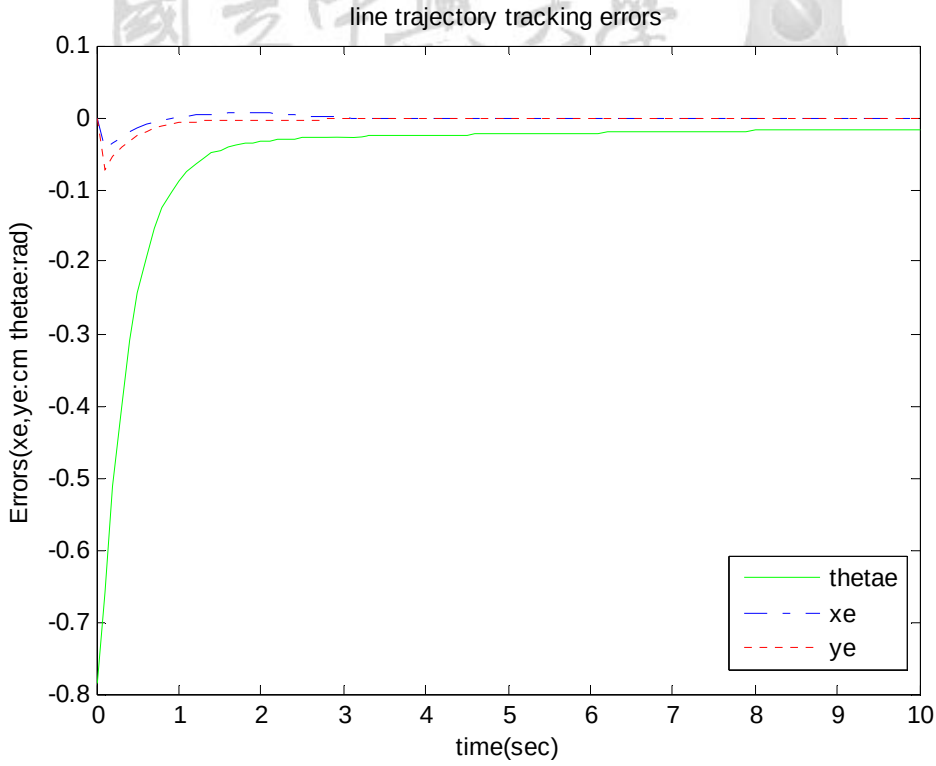
Figure 3.5. (a) Point to point stabilization result. (b) Time history of the vehicle orientation.

3.5.3 Straight-Line Trajectory Tracking

The second simulation is to study straight-line trajectory tracking performance of the control law (3.38). The initial pose of the robot was given by $(X_o, Y_o, \theta_o)^T = (30cm,\ 0\text{cm},\ 0rad)^T$ and the desired straight-line trajectory was described by



(a)



(b)

Figure 3.6 . (a) Straight-line trajectory tracking result. (b) Line trajectory tracking errors of the platform.

$$\left(X_d(t) \quad Y_d(t) \quad \theta_d(t)\right)^T = \left(X_{do} + V_x t \quad Y_{do} + V_y t \quad \frac{\pi}{4}\right)^T$$

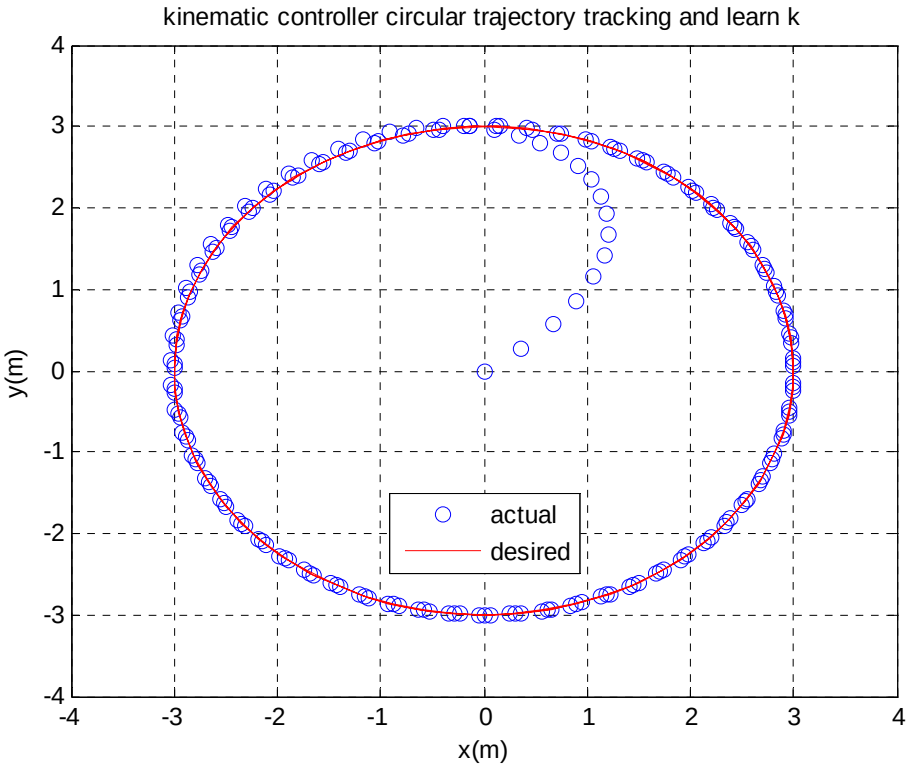
where $X_{do} = Y_{do} = 0\,cm,\ V_x = V_y = 10\ (m/s)$. Figure 3.6(a) shows the behavior of the simulation straight-line trajectory tracking of the omnidirectional mobile robot. As shown in Figure 3.6(b) that line trajectory tracking errors of the platform. The result indicated that the control (3.38) was capable of steering the robot to follow the desired line path.

3.5.4 Circular Trajectory Tracking

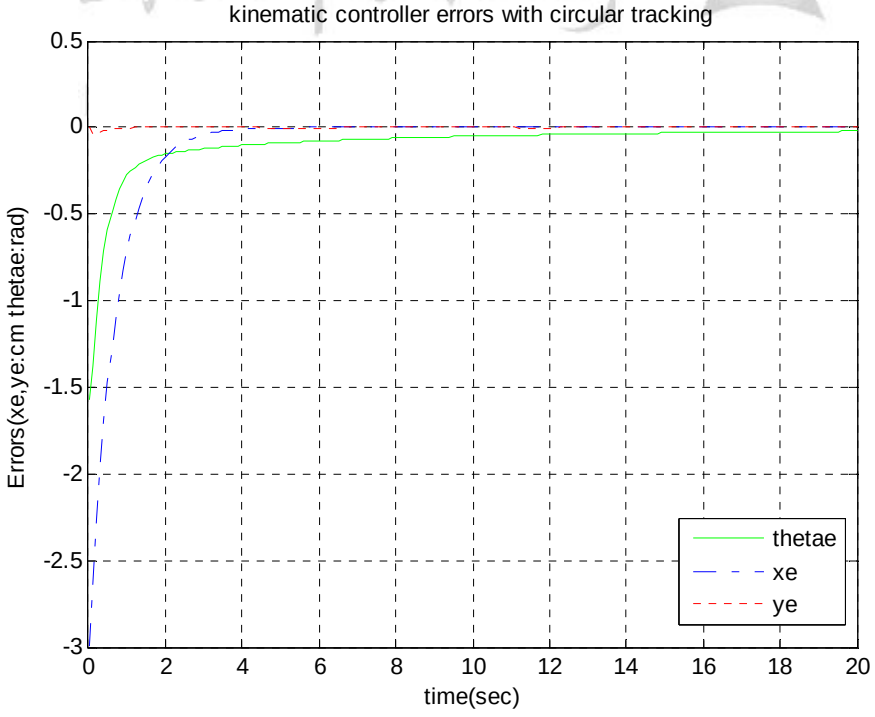
The circular trajectory tracking simulation was conducted with the following parameters: $\omega_o = 0$ (rad/sec) , $\omega_r = 1$ (rad/sec) , $r_c = 3$ (m) , $Y_{do} = 0$ (cm), $X_{do} = 0$ (cm). The starting pose was given by $(X_o, Y_o, \theta_o)^T = (0cm,\ 0cm,\ 0rad)^T$, and the desired circular trajectory was described by

$$\left(X_d \quad Y_d \quad \theta_d\right)^T = \left(X_{do} + r_c \times \cos(\omega_o + \omega_r t) \quad Y_{do} + r_c \times \sin(\omega_o + \omega_r t) \quad t\right)^T .$$

Figure 3.7(a) presents the simulation result of the circular trajectory tracking for the omnidirectional mobile robot with the control law (3.38) and Fig. 3.7(b) shown with the tracking errors. The result indicated that the control law (3.38) was useful in achieving the circular trajectory tracking.



(a)



(b)

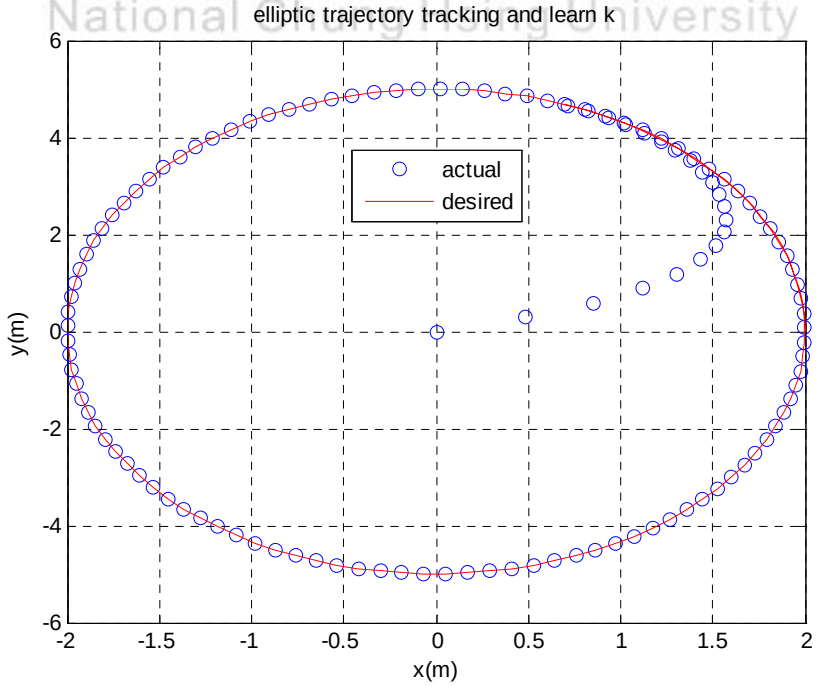
Figure 3.7. (a) Simulation result of the circular trajectory tracking. (b) The errors of the circular trajectory tracking.

3.5.5 Elliptic Trajectory Tracking

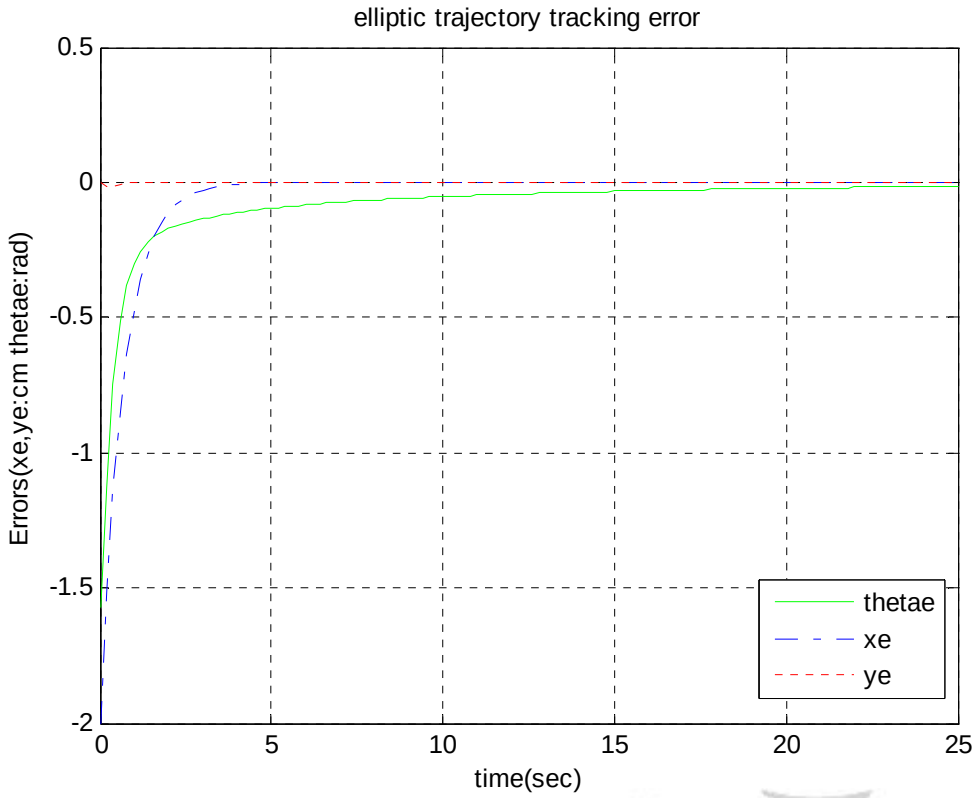
The next simulation for the tracking controller (3.38) was to steer the mobile robot following an elliptic trajectory described by

$$\begin{pmatrix} X_d & Y_d & \theta_d \end{pmatrix}^T = \left(X_{do} + r_1 \times \cos(\omega_o + \omega_r t) \quad Y_{do} + r_2 \times \sin(\omega_o + \omega_r t) \quad \frac{\pi}{2} \right)^T$$

The simulation assumed that the robot started with $(X_o, Y_o, \theta_o)^T = (0cm,\ 0cm,\ 0rad)$. The parameters in the elliptic trajectory tracking experiment were taken as follows: $\omega_o = 0$ (rad/sec) , $\omega_r = 0.3$ (rad/sec) , $r_1 = 2$ (m) , $r_2 = 5$ (m) , $Y_{do} = 0$ (cm) and $X_{do} = 0$ (cm). Figure 3.8(a) and Figure 3.8(b) depict the simulation of elliptic trajectory and errors response of the robot. The results indicated that the unified dynamic tracking controller was capable of steering the omnidirectional mobile robot to exactly track the elliptic trajectory.



(a)



(b)

Figure 3.8. (a) Result of the elliptic trajectory tracking. (b) Errors response of the elliptic trajectory tracking.

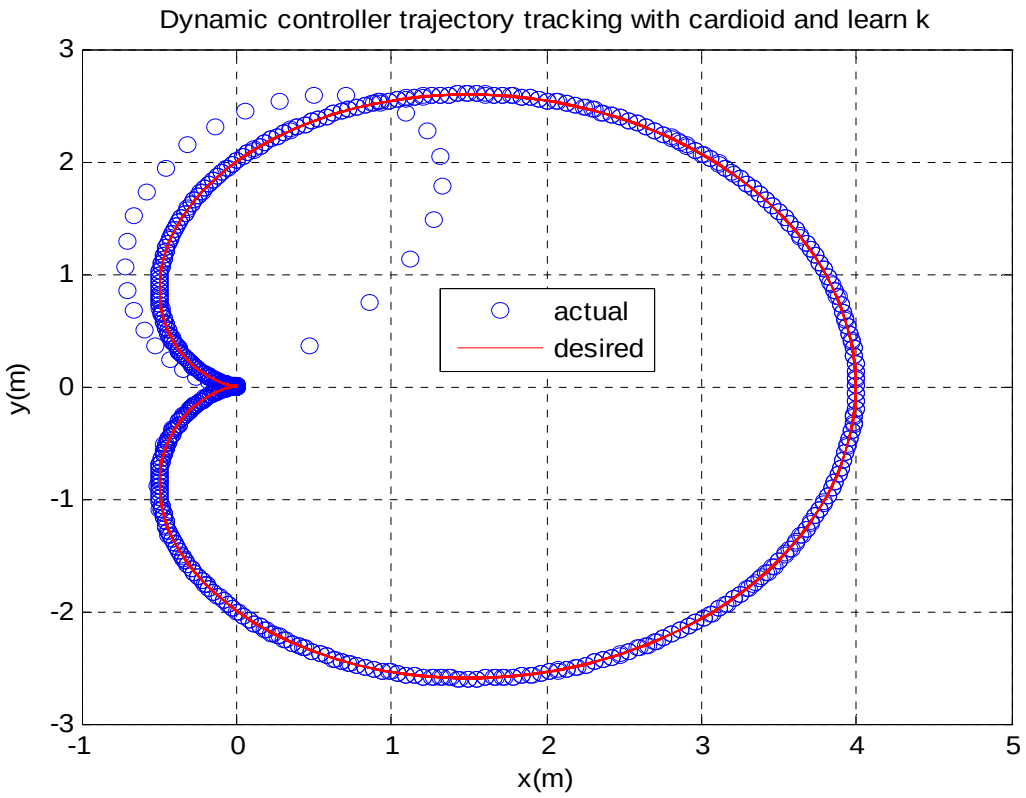
3.5.6 Cardioids trajectory Tracking

The last simulation was conducted to investigate the performance of the proposed dynamic control law (3.38).to steer the mobile robot following a cardioids trajectory described by

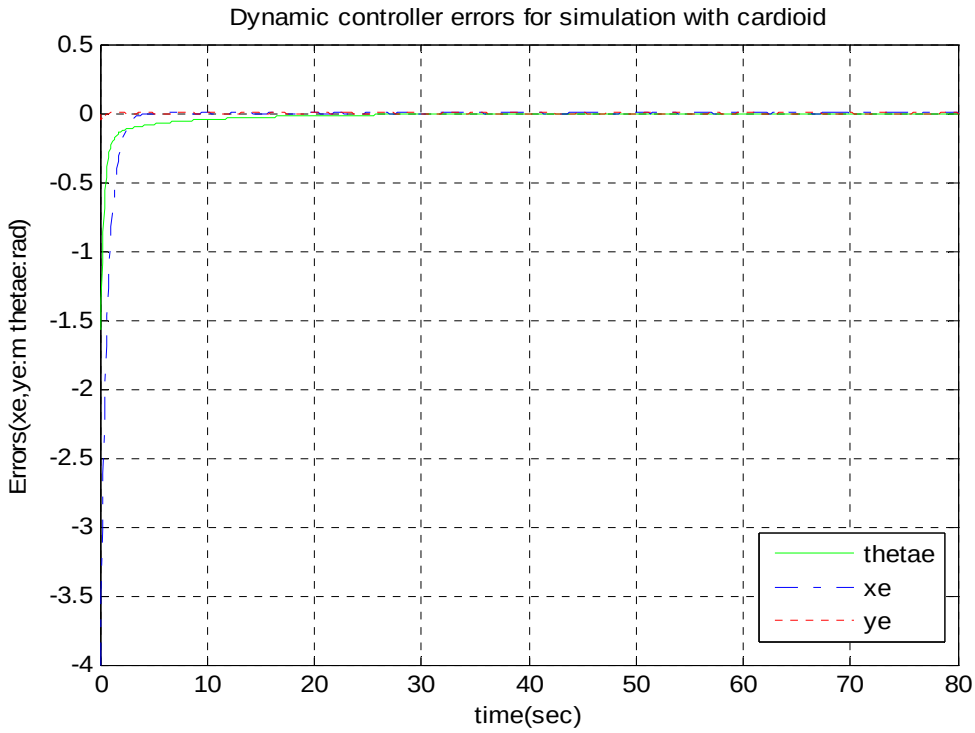
$$\begin{pmatrix} X_d & Y_d & \theta_d \end{pmatrix}^T = \begin{pmatrix} [(1+\cos(t))*\cos(t)]*2 & [(1+\cos(t))*\sin(t)]*2 & \pi/2 \end{pmatrix}^T$$

The simulation assumed that the robot started with $(X_o, Y_o, \theta_o)^T = (0cm,\ 0cm,\ 0rad)$. The parameters in the cardioids trajectory tracking experiment were taken as follows: $\omega_o = 0$ (rad/sec) , $\omega_r = 2$(rad/sec) , x0=4,y0=-0.09 , $Y_{do} = 0$ (cm) and $X_{do} = 0$ (cm), kp=0.0001, k=22; the start-up time is 0 and the end time is 2*π. In the

simulation, the fourth Range-Kutta method was adopted with a fixed step. Figure 3.9(a) and Figure 3.9(b) depict the simulation of cardioids trajectory and errors response of the robot. The results indicated that the unified dynamic tracking controller was capable of steering the omnidirectional mobile robot to exactly track the cardioids trajectory.



(a)



(b)

Figure 3.9. (a) Result of the cardioids trajectory tracking. (b) Errors response of the cardioids trajectory tracking.

3.5.7 Experimental Results and Discussion

The following two experiments are conducted to examine the effectiveness and performance of the proposed dynamic control method on the constructed four-wheeled omnidirectional robot. Both experiments adopted the dead-reckoning method mentioned in Section 2.4.2, and used the small-scale PC as the main motion controller such that the control codes can be easily implemented by C language. First, the line trajectory tracking experiment is used to study the tracking performance of the dynamic controller. The initial position and desired trajectory were respectively were respectively set by $[x_0 \quad y_0 \quad \theta_0]^T = [250\text{cm} \quad 100\text{cm} \quad \pi/2\text{rads}]^T$, and

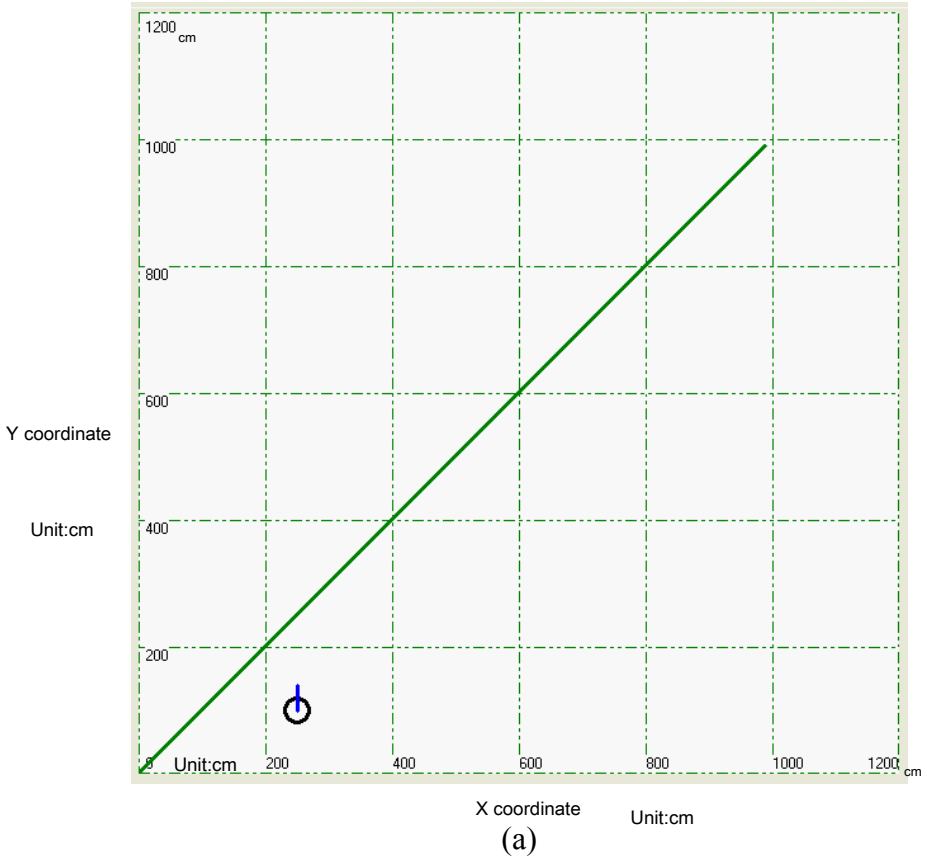
$$[x_r \quad y_r \quad \theta_r]^T = \left[Vt \cdot \cos\frac{\pi}{4} \quad Vt \cdot \sin\frac{\pi}{4} \quad \frac{\pi}{4}\right]^T, \; V = 20 \text{ cm/sec}, t \geq 0.$$

Figure 3.10 shows the behavior of the experimental straight-line trajectory and vehicle

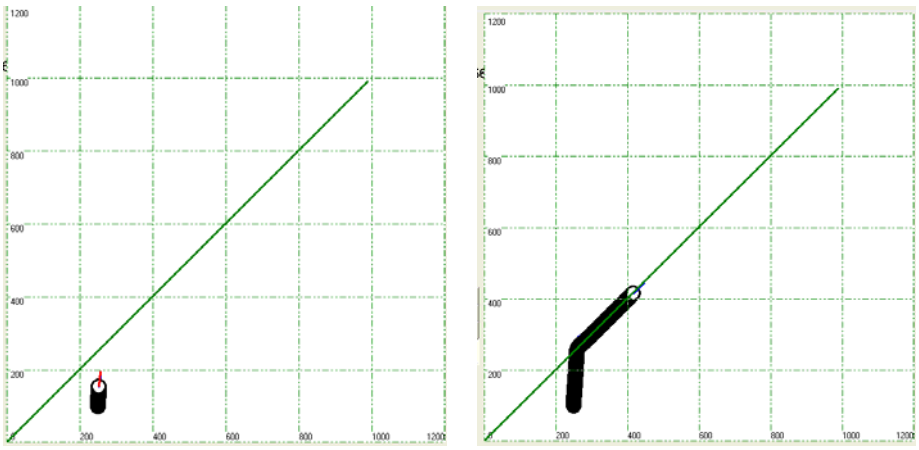
orientation of the omnidirectional mobile robot. Figure 3.11 illustrates the experiment historical pictures for the straight-line trajectory tracking. Table 3.2 depicts the mean error and the standard deviation from the line path experimental result where the sampling period is every 10 milliseconds and the total number of the samples is 4900.

Table 3.2 Mean error and standard deviation of the line path experiment.

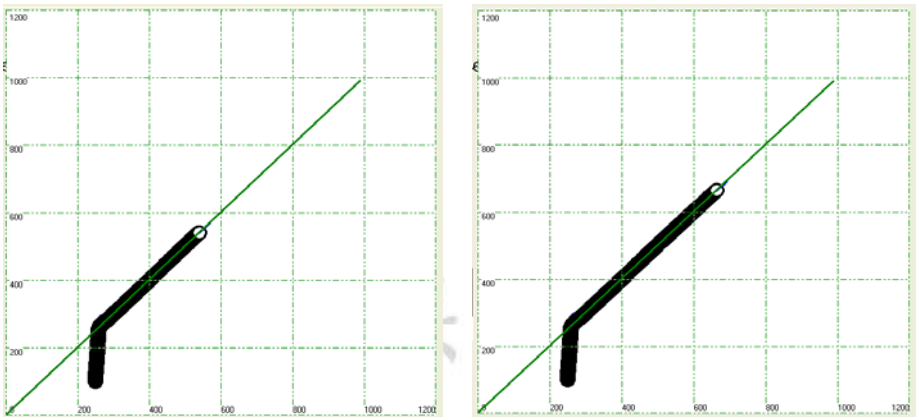
mean error	standard deviation
8.874cm	4.223cm



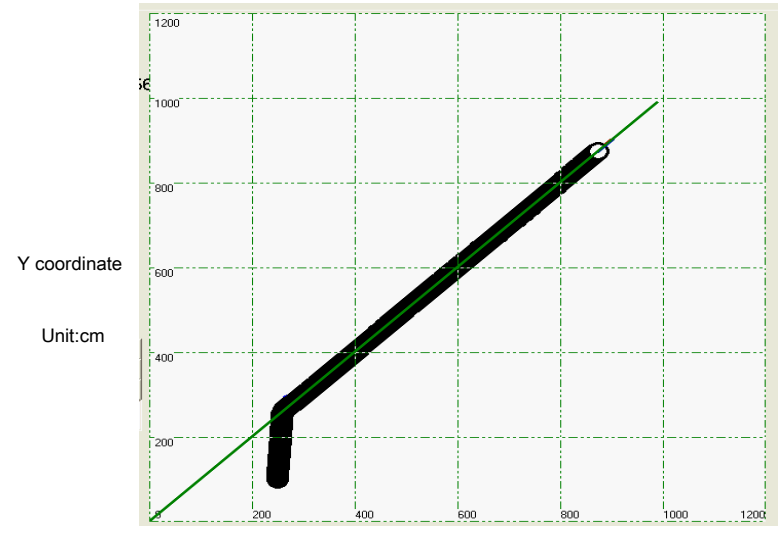
(a)



(b) (c)



(d) (e)



(f)

Figure 3.10. (a) Experiment straight-line trajectory tracking start point. (b~e)The trajectories of the mobile robot moving toward the straight line. (f) The overall tracking result.

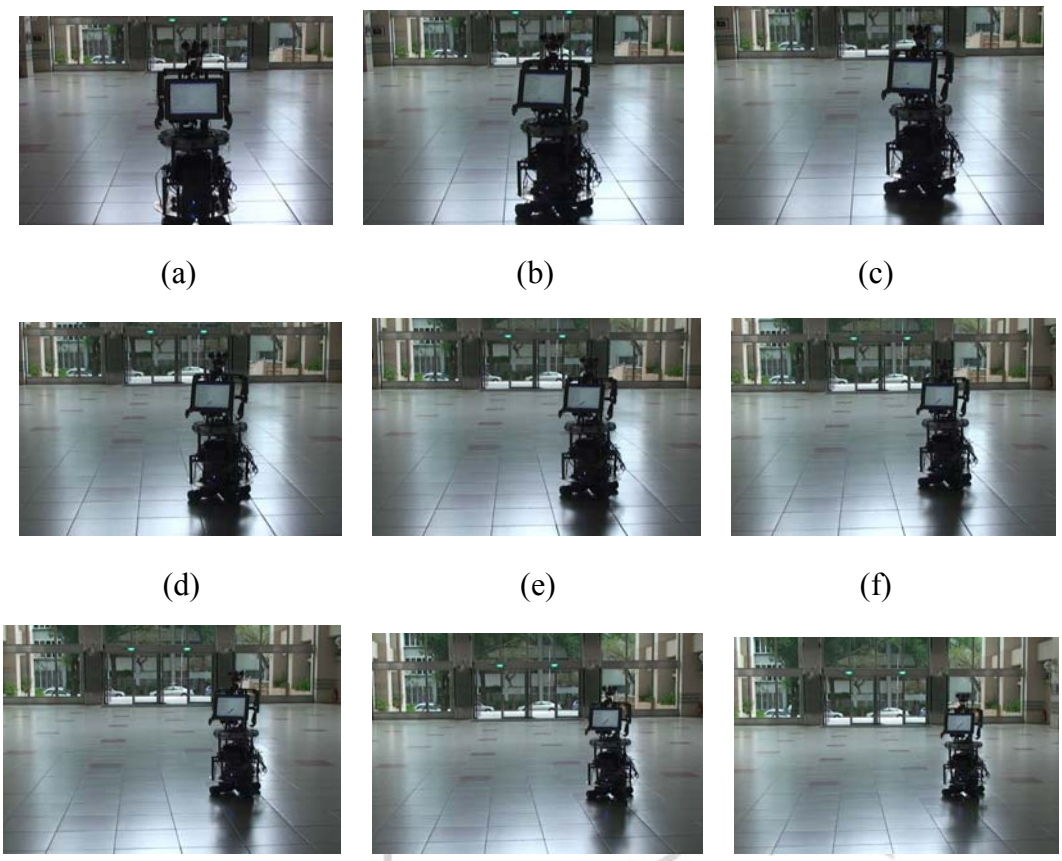


Figure 3.11. The time historical pictures of tracking the straight line trajectory.

Like the previous experiment, the second experiment was aimed to observe the behavior of the controller while the robot was moving toward the designed random destinations. The starting posture was $[x_0 \quad y_0 \quad \theta_0]^T = [400 \text{ cm} \quad 500 \text{ cm} \quad \frac{\pi}{2} \text{ rad}]^T$, and the designed via points are respectively

$$[x_d \quad y_d]^T = [350 \text{ cm} \quad 580 \text{ cm}], [300 \text{ cm} \quad 680 \text{ cm}], [350 \text{ cm} \quad 750 \text{ cm}]$$
$$[460 \text{ cm} \quad 800 \text{ cm}], [360 \text{ cm} \quad 850 \text{ cm}], [480 \text{ cm} \quad 900 \text{ cm}], [400 \text{ cm} \quad 1000 \text{ cm}]$$

The dynamic controller was with the constraints of the control signals, i.e, speed $\|v_i\| \leq 20 \text{ cm/sec}$. Figure 3.12 depicts the experimental random trajectory of the robot. These results show that the dynamic tracking controller was capable of steering the omnidirectional mobile robot to exactly track the random destinations. Figure 3.13

depicts the experimental pictures of the random trajectory tracking. Table 3.3 depicts the mean error and the standard deviation from the random trajectory tracking experimental result. The sampling period is every 10 milliseconds. The total number of the samples is 5400.

Table 3.3. Mean error and standard deviation of the random trajectory tracking experiment.

mean error	standard deviation
9.248 cm	5.086cm

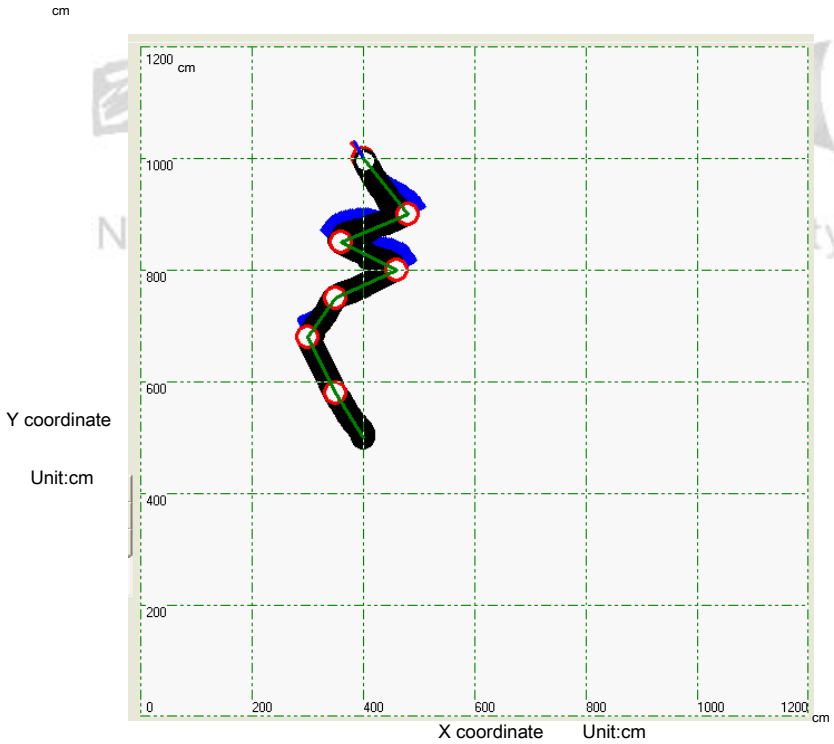


Figure 3.12. Experimental result of the random destination tracking.

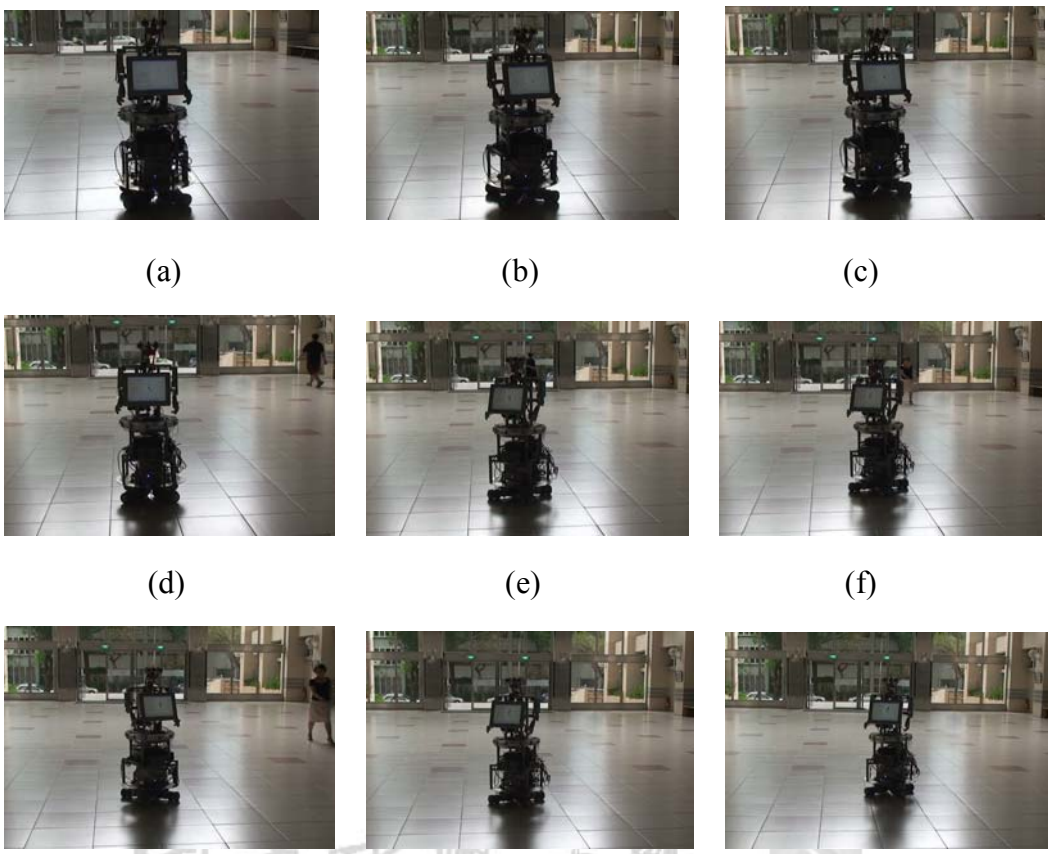


Figure 3.13. Time historical pictures of the random destination tracking.

3.6 Concluding Remarks

This chapter has developed methodologies for dynamic modeling and adaptive motion control of the four-wheeled omnidirectional mobile platform. The adaptive dynamic motion controller has been derived for position control, path following and trajectory tracking of the omnidirectional mobile robot equipped with four independent omnidirectional wheels equally spaced at 90 degrees from one to another. Such a controller is synthesized by backstepping and has been proven globally asymptotically stable via the Lyapunov stability theory. Through computer simulations and experimental results, the feasibility and efficacy of the proposed control method have also been well exemplified by conducting several simulations results on steering the mobile robot.

Chapter 4

SoPC-Based Implementation

4.1 Introduction

This chapter develops techniques for SoPC implementation of the adaptive dynamic controller designed in Chapter 4. Although the proposed motion control law has been successfully realized and tested on the small-scale embedded personal computer, this control algorithm can be implemented into a FPGA chip in order to reduce power consumption, volume and weight. Once the FPGA-base chip has been designed and verified, the FPGA codes can be further transformed into a application-specific integrated circuit (IC).

The SoPC technology is the third generation of the FPGA technology, and has been bringing a major revolution in the design of integrated circuits which efficiently integrate embedded processor intellectual properties (IPs) and application IPs into a FPGA chip. This technology has gained benefits of low cost, low power consumption, small circuit size, IP re-usability and reprogrammable hardware/software co-design. Due to these advantages, the SoPC technology has been shown as a powerful means to combine flexible software modules and high-performance hardware units for implementing sophisticated signal processing algorithms, and high-performance but computation-intensive algorithms. Comparing with the fixed-processor DSP which was shown to provide a feasible solution for developing a powerful controller for a robotic manipulator, the SoPC technology is capable of not only achieving the same software functions running in its embedded processor, but also providing additional hardware IP implementation and an embedded real-time operating system for further purposes. Furthermore, the SoPC technology may offer identical functions to hardware-oriented FPGAs. Comparing with VLSI-based chips, the SoPC has its own

merits and features, such as programmability, fast time-to-market and short design cycle; these unique strengths make the SoPC particularly suitable for small-volume but complicated products or applications. In addition, this FPGA-based SoPC approach also provides a compromise approach between the special-purpose application specified integrated circuit (ASIC) hardware and general-purpose processors.

The rest of the chapter is constructed as follows. Section 4.2 describes the SoPC implementation issues for the proposed embedded adaptive motion controller, and Section 4.3 designs the IP core library. In Section 4.4, the control software of the SoPC-based controller is described as well. Section 4.5 presents concluding remarks of the chapter.

4.2 FPGA Implementation Issues

This subsection is devoted to designing an embedded adaptive controller of the omnidirectional mobile platform for the tour-guide robot in order to achieve the adaptive control law (3.38). Figure 4.1 depicts the architecture of the Altera FPGA implementation for the proposed adaptive mobile platform controller. The Avalon memory-mapped (Avalon-MM) interfaces are used for reading/writing interfaces on master and slave components in a memory-mapped system. These components include microprocessors (Nios II), memories, UARTs, and timers, and have master and slave interfaces connected by a system interconnect fabric. The Avalon switch fabric is the collection of signals and logic that connects master and slave components, including address decoding, data-path multiplexing, wait-state generation, arbitration, interrupt controller, and data-width matching. The adaptive control law (3.38) for the mobile platform has been implemented into the 32-bit Nios II processor whose numerical precision and computation speed are high enough to

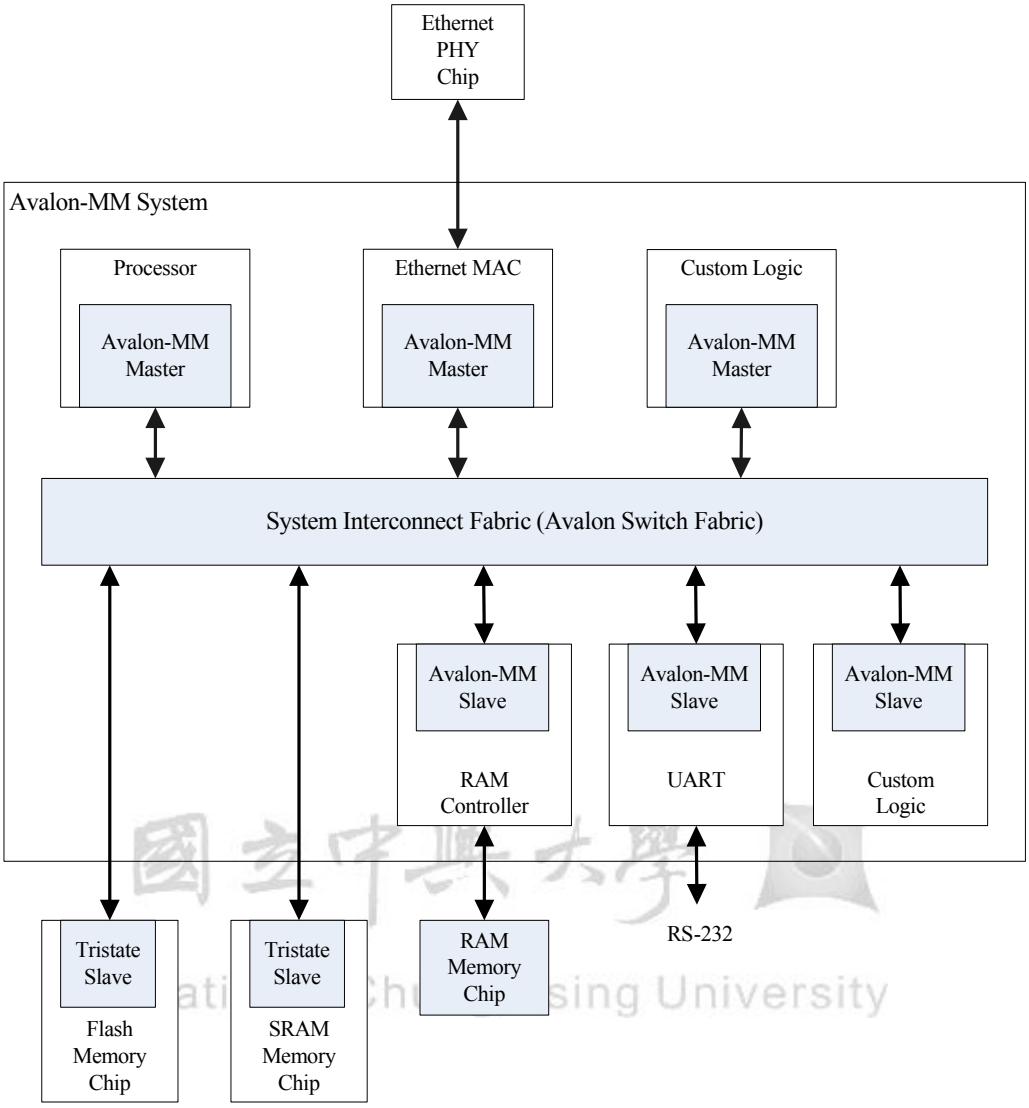


Figure 4.1. Architecture of the FPGA implementation for the proposed adaptive mobile platform controller.

realize the adaptive control law. The user IP cores (custom logic) for this robotic application have been developed by VHDL (VHSIC Hardware Description Language). The software-based adaptive controller and hardware-based custom logic are connected to the system interconnect fabric via Avalon-MM for achieving the adaptive controller in one FPGA chip.

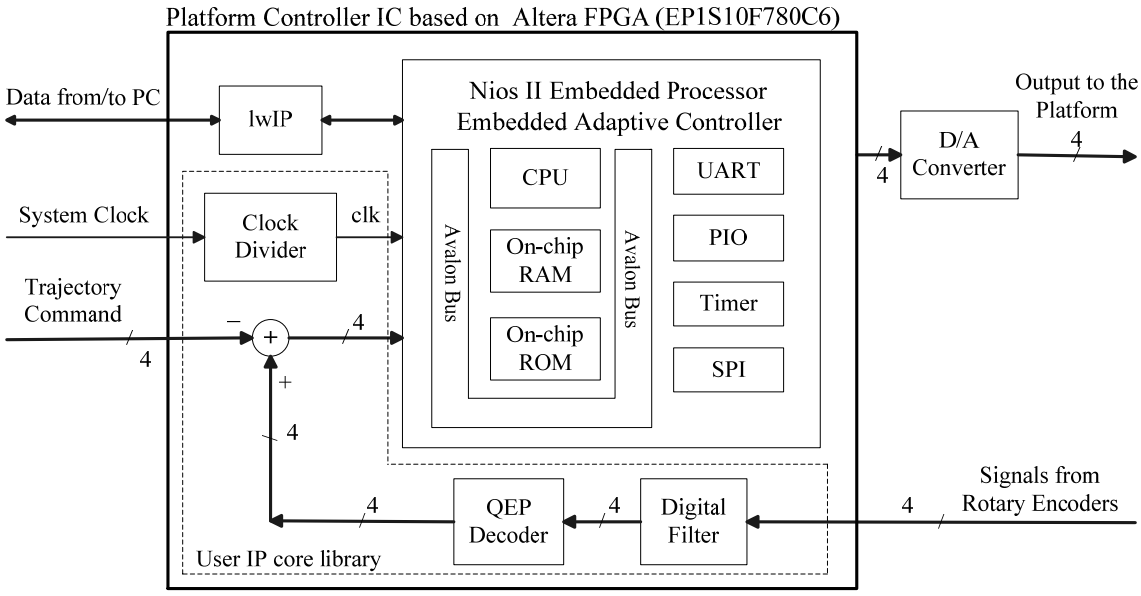


Figure 4.2. Embedded adaptive controller of the omnidirectional mobile platform.

As shown in Figure 4.2, the adaptive control law (3.38) has been efficiently implemented in a FPGA chip using a 32-bit Nios II processor. The 12-bit D/A converter, AD7541, is employed to convert the output commands into analog voltage signals for driving three DC brushless motors mounted on the four omnidirectional wheels. The three quadrature-encoder-pulse (QEP) signals generated from the four motors are fed back to the embedded controller. With the QEP signals, the real position and orientation of the mobile platform can be dead-reckoned by the embedded soft-core CPU Nios II. The QEP decoder circuit was implemented by VHDL, and the real-time OS MicroC/OS-II was ported into the FPGA chip to handle the data communication with PC via TCP/IP protocol. The TCP/IP stack was constructed using the prevalent real-time OS MicroC/OS-II which is regarded as a portable, ROMable, scalable, preemptive real-time, multitasking kernel for microprocessors. This kind of stack also includes the standard UNIX sockets

application programming interface (API). Moreover, the embedded soft-core Nios II processor works with the lwIP (lightweight IP) for the Ethernet connectivity, thereby significantly reducing resource usage. The FPGA chip is Altera Stratix EP1S10F780C6 with 10,570 LEs (Logic Element), 426 user I/O pins, 6 DSP blocks, 920,448 RAM bits memory, 6 PLLs (Phase-Lock Loop) and an embedded Nios II 32-bit RISC (Reduced Instruction Set Computer) processor.

4.3 Design of the User IP Core Library

This subsection presents a user IP core library (custom logic) connected to the Altera system interconnect fabric via Avalon-MM for the mobile platform controller in FPGA chip. This user IP core library is very useful not only for the presented mobile platform, but also for other embedded robotic studies. In what follows, special attention is paid to detail the design of the IP core library which consists of three modules: clock divider module, digital filter module and QEP circuit module.

4.3.1 Clock Divider Module

A 50 MHz quartz oscillator is used to supply the FPGA chip in Altera Stratix development board. The clock divider module is mainly employed to generate the desired 50/2N MHz clock frequencies to other clock domain modules in the controller, where N is an arbitrary integer. For efficient hardware realization, the well-known VHDL technique will be adopted to implement this module instead of the up-counter

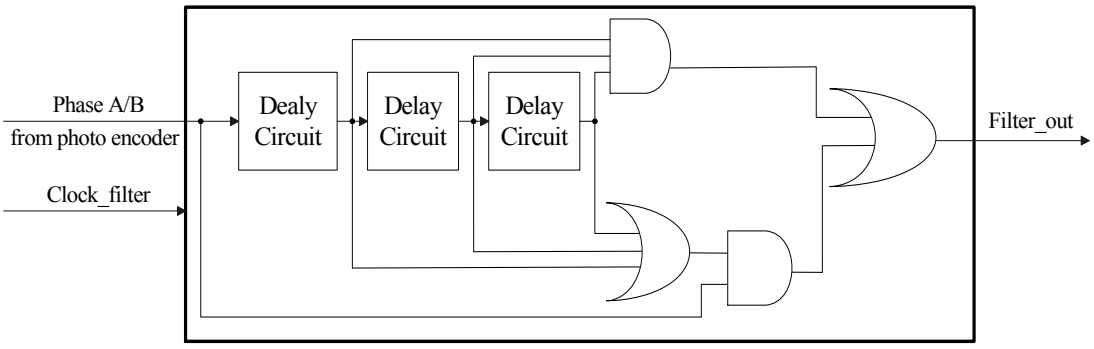


Figure 4.3. Block diagram of the digital filter module.

and D flip-flop based design [34]. This approach exploits the flexibility for generating different frequency clocks by simply adjusting the integer N, thereby providing the specified clock frequencies 50/2N MHz with 50% duty cycle.

4.3.2 Digital Filter Module

Photo encoders are often integrated in motor assembly in order to determine the current position of the motors by QEP processing. There are four sets of photo encoders in the mobile platform. Because the photo encoders are very sensitive to their working environments, the measurement noise must be filtered out by a digital filter prior to the QEP processing in order to obtain exact readings. The digital filter made by three delay circuits and relevant combinational logic elements can be easily done via the VHDL programming techniques. Figure 4.3 depicts the block diagram of the digital filter IP design in which the clock frequency of Clock_filter is 1 MHz with 50% duty cycle from the clock divider module in Section 4.3.1.

4.3.3 QEP Circuit Module

The QEP module is used for direct interface with a linear or rotary incremental encoder to obtain its position, direction and speed information for use in a

high-performance motion and position control system. Figure 4.4 illustrates the IP circuit design for the QEP modules, which have been implemented into the FPGA chip. In

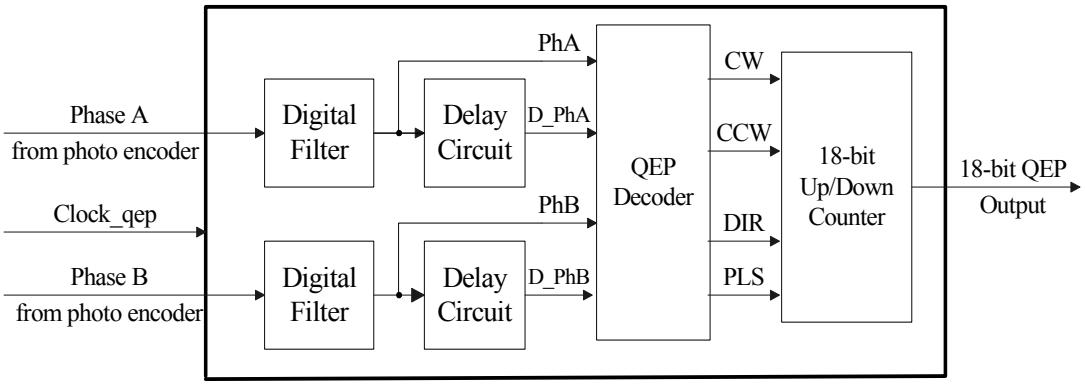


Figure 4.4. Block diagram of the QEP circuit module.

Figure 4.4, the clock rate of Clock_qep from the clock divider is 1 MHz with 50% duty cycle. Once the motors turn, the encoders output the QEP signals which are digitally filtered and sent into the VHDL-based 18-bit QEP counters to calculate the QEP pulses. Note that these 18-bit counters are designed to accomplish desired counting precision. With the QEP pulses, the real positions of the mobile platform can be estimated directly by the embedded Nios II processor. In the adaptive controller design, four QEP circuit modules are needed for the mobile platform. Compared with [34], these QEP modules integrated in FPGA chip saves the circuit size and keeps the cost down due to no extra A/D converters and no additional connection wires.

4.4 Control Software in the Embedded Adaptive Controller

The adaptive controller of the mobile platform is presented in Figure 4.2, in which the Nios II embedded processor is employed to execute the adaptive control law (4.30). The software program was developed for implementing the adaptive control algorithm incorporating with the hardware module (user IP), lwIP and embedded OS

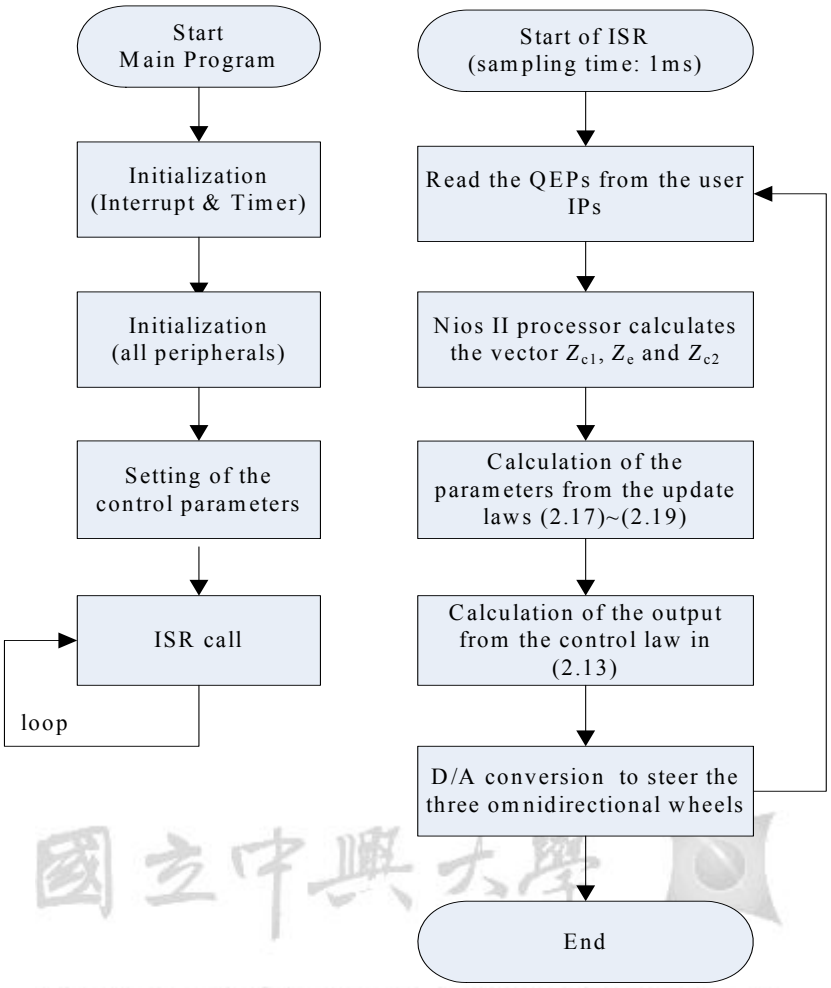


Figure 4.5. Main program and ISR for the proposed adaptive control scheme executed by Nios II processor.

in one Altera Stratix FPGA chip (EP1S10F780C6). With the hardware/software co-design and SoPC technology, the adaptive controller takes the advantage of software flexibility for complicated algorithm with low sampling frequency in motion control (less than 1kHz), and high sampling frequency required in hardware IP (greater than 1MHz) [35]. A simple left-side rule [36] for numerical approximation of integration/discretization is used to implement the adaptive control law (3.38). Figure 4.5 depicts the flow chart of the adaptive controller program, where the main program and the interrupt service routine (ISR) for the adaptive controller were coded in C/C++ language. With the developed QEP IPs, the embedded processor calculates the

current position and orientation of the mobile platform via dead-reckoning approach, and then obtains the values of Y_1, Y_e and Y_2. Once Y_1, Y_e and Y_2 are determined, they will be used to perform the parameter update law (3.43). Finally, the output of the control law (3.38) is sent out to the D/A converters, thus steering the three omnidirectional wheels. The interrupt interval is 1ms since the sampling time is 1 kHz. Through the Nios II integrated development environment (IDE), the adaptive controller program is cross compiled and downloaded into the SDRAM via JTAG interface. The executing time of the adaptive controller algorithm in Nios II processor is 150 μs.

4.5 Concluding Remarks

This chapter has developed a SoPC-based embedded adaption dynamic motion controller for four-wheeled mobile platform of a tour-guide robot. The SoPC technology has been successfully used to efficiently implement required hardware and software of the motion control algorithm into a FPGA chip. Several experiments for verifying the SoPC-based motion controller has been conducting, in order to confirm whether it is capable of achieving satisfactory trajectory tracking control performance of the robot.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This thesis has developed a ZigBee localization module and a SoPC-based adaptive motion controllers incorporating with an embedded processor. Simulations and experimental results are used for illustration of the effectiveness and applicability of the proposed methods. The main results of the thesis are threefold and summarized as below.

First, a localization system for the tour guide robot has been presented for global localization. The localization method for the ZigBee module has been presented based on the RSSI measurements, the proposed calibration method and the least square method. After the calibration procedure has been done, the calibration curves are adopted to convert the RSSI values into the corresponding distances; then the least-square approach is employed to obtain global localization of the robot. Once the global robot pose has been determined, data from the dead reckoning unit and the ZigBee module are then fused to compensate for the position error in order to achieve accurate position estimate. In order to steer the robot safely and smoothly, the gear ratios of the motors have also been increased.

Second, an adaptive dynamic control law has been derived for both stabilization and trajectory tracking of the four-wheeled omnidirectional mobile platform with dynamic effect, frictions and uncertainties. The proposed controller has been proved with the property of globally asymptotical stability via the Lyapunov stability theory. The proposed adaptive control method can be straightforwardly extended to address the path following problem such that any smooth time-independent path can be exactly followed. Simulations and experimental results have shown that the proposed

dynamic controller is capable of giving satisfactory control performance for the tour-guide robot.

Third, the SoPC-based adaptive motion controller incorporating with embedded processors has been constructed. This SoPC-based adaptive motion controller combining the hardware/software co-design and IP re-use concept takes the advantages of efficient implementation, excellent flexibility and cost-effective implementation, satisfactory performance characteristics with fast computations.

5.2 Future Work

Based on the aforementioned investigations and studies, the future work will be devoted to the following four main topics.

(1) Experimentation of the proposed SoPC-based dynamic controller

Several eexperiments have been under working, in order to ensure whether the proposed SoPC-based dynamic controller performs well in steering the tour-guide robot.

(2) Nonlinear function Approximation

As mentioned revealed in Section 3.2, the nonlinear dynamic state equation of the robot contains the nonlinear vector $\bar{\bar{N}}\{{}^{w}\dot{S}\}$ composed of many known terms and dynamic and static frictions. It has been known that dynamic and static frictions vary with terrain and the surface with different materials, thus causing parameter variations in the nonlinear vector $\bar{\bar{N}}\{{}^{w}\dot{S}\}$. If the nonlinear function vector $\bar{\bar{N}}\{{}^{w}\dot{S}\}$ can be approximated and learned by a kind of universal function approximators, then the controller would be more realistic. Several known

function approximators, such as adaptive neural fuzzy inference system (ANFIS), B-spline curve, radial-basis function networks, fuzzy basis function networks, fuzzy wavelet networks, interval type 2 fuzzy approximators and etc., would be used to automatically learn the nonlinear function vector $\bar{\bar{N}}\{{}^{w}\dot{S}\}$. Furthermore, weights of these kinds of function approximates could be derived via Lyapunov stability theory.

(3) Design and Implementation of SoPC-based Autonomous Navigation System

A complete autonomous navigation system generally includes four main modules: sensing and perception, localization and mapping, cognition and planning, and motion control. Although the techniques of these main modules have been separately investigated over the past and present, how to efficiently integrate them into a complete and efficient SoPC-based navigation system deserves much research effort.

(4)Speech Recognition

In interactive operation interface, the development of speech recognition technology provides the user not only via the touch screen but also through voice-command recognition inputs the commands to the computer, thereby directly communicating with the tour guide robot. The human-robot interactive module with a capability of speech recognition definitely facilitates the easy and convenient communication procedure between users and the robot.

References

[1] W. Burgard, “The interactive museum tour-guide robot,” *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, 1998, pp 11-18.

[2] S. Thurn, “MINERVA: A second-generation museum tour-guide robot, ”*Proc. of the IEEE International conference on Robotics and Automation*, 1999 Volume 3, 10-15 May 1999, pp 1999 – 2005.

[3] M. Y. Wang, *Autonomous navigation and interactive operation of a tour guide robot*, M.S. Thesis, Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan, July 2007.

[4] Y. J. Feng , *Motion Control, navigation and mission execution of a tour-guided robot with four-wheeled omnidirectional platform*, M.S. Thesis, Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan, July 2008.

[5] R. Siegwart and I. R. Nourbakhsh , *Introduction to autonomous mobile robots*, A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England, 2004.

[6] F. G. Pin and S. M. Killough, “A new family of omni-directional and homonymic wheeled platforms for mobile robots,” *IEEE Transactions on Robotics and Automation*, vol.10, pp.480-489, August 1994.

[7] M. J. Jung, H. S. Kim, S. Kim, and J. H. Kim, “Omni-directional mobile base OK-II,” *Proceedings of the 2000 IEEE international conference on robotics and automation*, San Franciso, CA, pp.3449-3454, April 2000.

[8] http://www.digi.com/technology/rf-articles/wireless-zigbee.jsp

[9] Pinedo-Frausto, E.D. and Garcia-Macias, J.A. ,” An experimental analysis of

ZigBee networks", *Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference*, pp. 723–729, Oct 2008.

[10] F. Sottile and R. Giannantonio and M.A. Spirito and F.L. Bellifemine," Design, deployment and performance of a complete real-time ZigBee localization system," *Wireless Days, 2008. WD '08. 1st IFIP*, pp. 1-5,Nov. 2008.

[11] J. F. Blumrich, *Omnidirectional vehicle*, United States Patent 3,789,947, 1974.

[12] B. E. Ilou, *Wheels for a course stable self-propelling vehicle movable in any desired direction on the ground or some other base*, United States Patent 3,876,255, 1975.

[13] M. West, H. Asada, "Design of ball wheel mechanisms for omnidirectional vehicles with full mobility and invariant kinematics," *Journal of Mechanical Design*, pp. 119-161, 1997.

[14] M. Wada, S. Mory "Holonomic and omnidirectional vehicle with conventional tires," *Proceedings of 1996 IEEE International conference on Robotics and Automation*, pp. 3671-3676, 1996.

[15] Carlisle, B, "A omnidirectional mobile robot," *Development in Robotics*, Kempston, pp.79-87, 1983.

[16] F. G. Pin, S. M. Killough, "A new family of omnidirectional and holonomic wheeled platforms for mobile robot," *IEEE Transactions on Robotics and Automation*, Vol. 15, No. 6, pp. 978-989, 1999.

[17] P. Muir, C.Neuman, "Kinematic modeling of wheeled mobile robots," *Journal of Robotic Systems*, Vol. 4, No. 2, pp. 281-340, 1987.

[18] F. G. Pin, and S. M. Killough, "A new family of omnidirectional and holonomic wheeled platforms for mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 4, pp. 480-489, 1994.

[19] K.S. Byun, S. J. Kim, J. B. Song, "Design of a four-wheeled omnidirectional

mobile robot with variable wheel arrangement mechanism, ” *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, Washington, DC, pp. 720-725, May 2002.

[20] L. Wilson, C.Williams, J.Yance, J.Lew, R.L. Williams II, “Design and modeling of a redundant omnidirectional RoboCup goalie,” [online available] http://zen.ece. ohiou.edu/ ~robocup/papers/mech/65.pdf.

[21] L. Huang, Y. S. Lim, D.C. E. L. Teoh, “Design and analysis of a four-wheel omnidirectional mobile robot, ” *The 2nd International Conference on Autonomous Robots and Agents*, Palmerston North, New Zealand, pp. 425-428. December 13-15, 2004.

[22] O. Purwin and R. D’Andrea, “Trajectory generation for four wheeled omnidirectional vehicles,” *Proceedings of 2005 American Control Conference*, Portland, OR, USA, pp. 4979-4984, June 8-10, 2005.

[23] C.-C. Shing, P.L. Hsu, S.S. Yeh, “T-S fuzzy path controller design for the omnidirectional mobile robot,” *the 32nd Annual Conference on IEEE Industrial Electronics (IECON 2006)*, Taipei, Taiwan, pp. 4142-4147, , 6-10 Nov. 2006.

[24] T.-H. S. Li, C.-Y. Chen, H.-L. Hung, and Y.-C.Yeh, “A fully fuzzy trajectory tracking control design for surveillance and security robots,” *E-proceeding of 2008 IEEE International Conference on Systems, Man and Cybernetics*, Singapore, October, 2008.

[25] I. Campo, J. Echanobe, G. Bosque and J. M. Tarela, “Efficient hardware/software implementation of an adaptive neuro-fuzzy system,” *IEEE Transactions on Fuzzy Systems*, vol.16, no.3, pp.761-778, June 2008.

[26] Y. S. Kung and M. H. Tsai, “FPGA-based speed control IC for PMSM drive with adaptive fuzzy control,” *IEEE Transactions on Power Electronics*, vol.22, no.6, pp.2476-2486, November 2007.

[27] S. S. Solano, A. J. Cabrera, I. Baturone, F. J. Moreno-Velo and M. Brox, "FPGA implementation of embedded fuzzy controllers for robotic applications," *IEEE Transactions on Industrial Electronics*, vol.54, no.4, pp.1937-1945, August 2007.

[28] Y. F. Chan, M. Moallem and W. Wang, "Design and implementation of modular FPGA-based PID controllers," *IEEE Transactions on Industrial Electronics*, vol.54, no.4, pp.1898-1906, August 2007.

[29] S. H. Han, M. H. Lee and R. R. Mohler, "Real-time implementation of a robust adaptive controller for a robotic manipulator based on digital signal processors," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: System and Humans*, vol.29, no.2, pp.194-204, March 1999.

[30] D. Zhang and H. Li, "A stochastic-based FPGA controller for an induction motor drive with integrated neural network algorithms," *IEEE Transactions on Industrial Electronics*, vol.55, no.2, pp.551-561, February 2008.

[31] C. F. Juang and C. H. Hsu, "Temperature control by chip-implemented adaptive recurrent fuzzy controller designed by evolutionary algorithm," *IEEE Transactions on Circuits and Systems-I: Regular Paper*, vol.52, no.11, pp.2376-2384, November 2005.

[32] C. F. Juang and J. S. Chen, "Water bath temperature control by a recurrent fuzzy controller and its FPGA implementation," *IEEE Transactions on Industrial Electronics*, vol.53, no.3, pp.941-949, June 2006.

[33] H. C. Huang and C. C. Tsai, "FPGA implementation of an embedded robust adaptive controller for autonomous omnidirectional mobile platform," *IEEE Transaction on Industrial Electronics,* vol. 56, no. 5, pp. 1604-1616 ,May 2009.

[34] T. H. Li, S. J. Chang and Y. X. Chen, "Implementation of human-like driving skills by autonomous fuzzy behavior control on an FPGA-based car-like mobile robot," *IEEE Transactions on Industrial Electronics*, vol.50, no.5, pp.867-880,

October 2003.

[35] Y. S. Kung and G.. S. Shu, “Design and implementation of a control IC for vertical articulated robot arm using SOPC technology,” *Proceedings of IEEE International Conference on Mechatronics*, pp.532-536, 2005.

[36] C. L. Phillips and H. T. Nagle, *Digital control system analysis and design*, 3rd edition, Prentice-Hall, Englewood Cliffs, N.J., 1995.